Module 10

Mario Rodriguez-Montoya
ESE 105
Washington University in Saint Louis
St. Louis, Missouri
mario.rodriguez-montoya@wustl.edu

Matthew Kuchak
ESE 105
Washington University in Saint Louis
St. Louis, Missouri
kmatt@wustl.edu

***Abstract*—Through the usage of raspberry PIs, the Adeept Rover Car, and the PICar class, we were able to accomplish three objectives, those being a successful control car, successful movement of the car to a certain point, and a successful implementation of both the previous objectives simultaneously.**

I. Introduction

These three goals will be named **Objectives #1, #2,** and **#3**. **Objectives #1** and **#2** deal with completely separate parts of the Rover Car, **#1** being an implementation of PID control to maintain a target rotations per second (RPS), and **#2** being an automatic control for the speed and direction of the rover using a camera and ultrasonic sensor.

II. Methods

A. Objective One: Control

The first objective deals only with calculating the RPS, using PID (Proportional, Integral, Derivative) control to adjust it, then using the result to adjust the duty cycle of the motor. Calculating RPS was accomplished through the use of a photoresistsor sensing the difference between a rotating object colored differently keeping pace with the spinning of the wheels. Because the wheels could spin at varying speeds and the rover car would move from one place to another, and due to the nature of a photoresistor, would need to account for background light, readings had be taken frequently. The background light was taken into account by adjusting the values of the AD readings of the photoresistor and counting transition using a threshold value, which was calculated using the following formula:

$$threshold = C_1(max[f(t-100), f(t-99), \\ ...f(t)] - min[f(t-100), f(t-99), ...f(t)])$$

Where C_1 is a constant used to adjust using a proportion (experimentally determined to be 0.2 in our case), and f(t) is the AD readings from the photoresistor over time. This equation essentially took the difference between the maximum and minimum for the last 100 readings and multiplied it by some proportion (C_1) to generate an accurate threshold for the code to accurately consider the passing above or below the threshold value by the AD readings as a transition of the wheel the photoresistor is taking readings from as going from the dark to light sections of the wheel. To allow for more accuracy, the AD readings were taken every 0.005 seconds, or a frequency of 200 readings per second. The RPS, because of its variability and because allowing the photoresistor time to record AD readings would result in more acurate readings, had a period of 0.250 seconds, or a frequency of four readings per second. This meant the photoresistor would have about 50 readings for every one RPS calculation, and because the threshold value was taken between each RPS calculation from the last 100 AD readings, this made for more accurate RPS readings. This allowed for an accurate counter of the RPS, and so gave us the ability to routinely measure the RPS as it changed due to hardware issues and later due to the PID control. To first translate our initial duty cycles into a rough starting point for the RPS, the rover was run three different times at different duty cycles, then the measured RPS was recorded and used to create a trend line, giving a rough RPS to duty cycle conversion. this would help as the closer to the target RPS the rover started at, the easier it would be for RPS to adjust and reach stability, as well as allowing for the programmer to choose a rough RPS for the rover to start at.

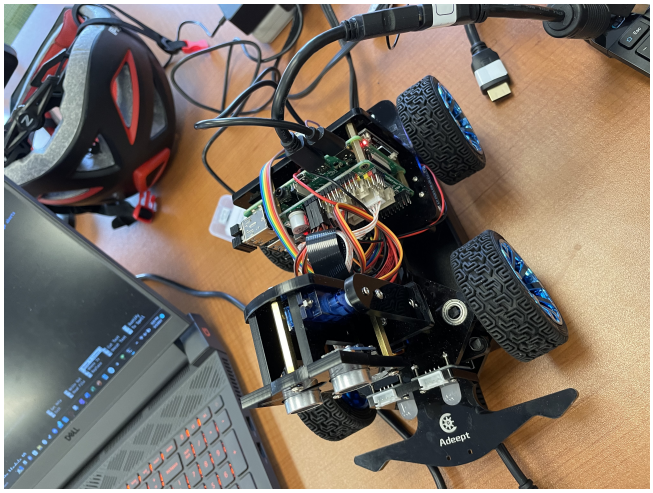

Fig. 1. The Adeept Rover Car used to complete the three objectives. The swivel and steering servos can be seen, which were used to turn the camera and turn the angle at which the wheels drove, respectively.

Fig. 2 shows the results of our experimentation with the relationship between RPS and duty cycle. While, because of outside factors like friction, a bare minimum amount of duty

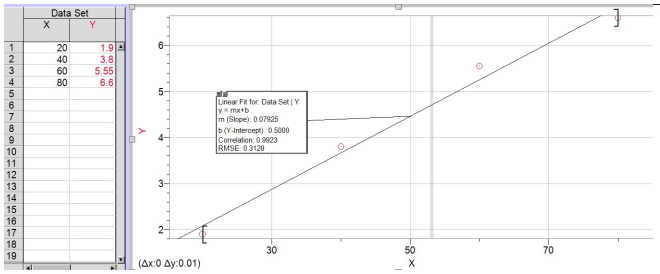

Fig. 2. The duty cycle/RPS line generated empirically by testing the rov car at different duty cycles using RPS recording software.

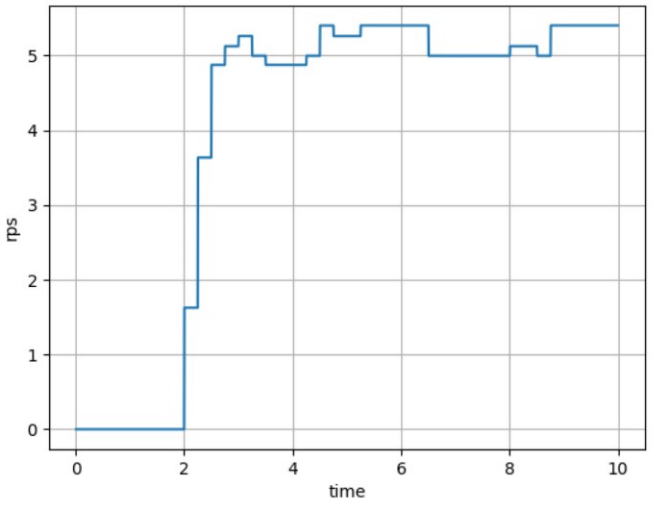

Fig. 3. A graph of the RPS over time using PID control with the rover suspended over a wooden block, the target RPS was five.

cycle needed to overcome the coefficient of static friction, etc. the relationship between duty cycle and RPS is not absolutely linear, because the duty cycle to RPS conversion only serves as a starting point for the RPS, it can be approximated as such so that PID control can have an easier time adjusting it to a target value.

After being certain of the efficacy of our RPS calculations and RPS to duty cycle trend line, PID control was then incorporated to reach and maintain a target RPS. To create effective PID control, values for the weights each portion of the PID should have were necessitated, and were determined experimentally. Derivative control did not have much of a positive impact on the performance of PID control, so its proportion was set to zero, essentially neutralizing the effect of derivative control. As for proportional and integral control, the value experimentally determined for proportional control was six, while integral control's was three. These numbers gave the rover its best performance in adjusting to and maintaining a target RPS.

To experimentally find these values, it was necessary to run control for the RPS with the rover suspended over a wooden block, and then look at the corresponding recorded

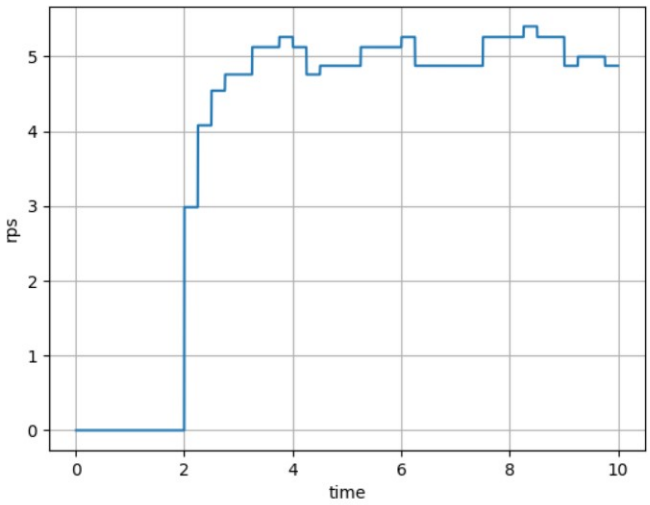

Fig. 4. A graph of the RPS over time with the rover driving on the ground, the fluctuations in the graph are due to the nature of **Objective #3**, and involve the rover turning from left to right often.

data to see how the PID control preformed. Using this trial-and-error method through analysis, the proportions for PID were determined. It is possible to analyze the performance of the system using three parameters, response time of the system, initial overshoot of the system, and the error after the system stabilizes, or steady state error. Fig. 3's, for example, has a rise time of about a second, an overshoot of about 0.2 RPS, or 4% over the target, and a steady state of about 0.25 RPS. For Fig. 4, the response time was about two seconds, the overshoot was also about 0.2 or 4%, and the steady state was about 0.2 RPS. This shows the performance of identical Ki, Kp and Kd parameters in two different settings, and shows how in this case, might not have been necessitated to accurately control the RPS of the rover either while driving or stationary.

To show the effects of Proportional control, the control code was run on a rover suspended by a wooden block with Integral and Derivative proportions set to zero, and a Proportional control proportion of one half the experimentally determined value and ten times the experimentally determined value. This is seen in Fig. 6 and 7, and serves to show the effects of Proportional control. Fig. 6 shows that a lack of integral control skews the steady state, as with a target RPS of five, the system stabilizes at about 5.5-5.6 RPS, matching or even exceeding the initial overshoot. Fig. 7 shows this to an even greater extent, as with a target RPS of five, the system overshoots to about 5.5 and remains there for the duration of the program, meaning it immediately reaches its steady state while off by about 10%. This shows the considerable effect proportional control has on the system, as from 0.5 times to ten times, while the overshoot did not change by very much, the settling time decreased to be near non-existent for 7. This also shows the impact that integral control has, as the steady

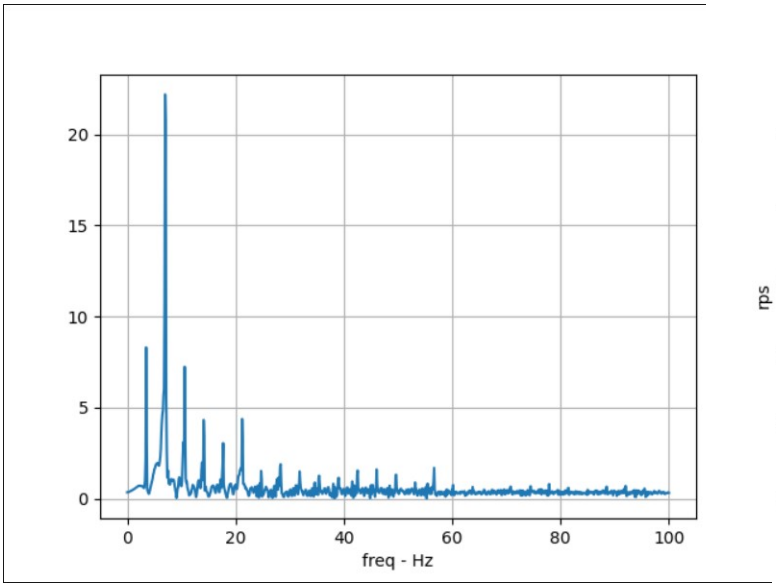

Fig. 5. An FFT of the steady state of Fig. 4. The fist spike can be seen around 5.5, which is about the value of the steady state of 4.

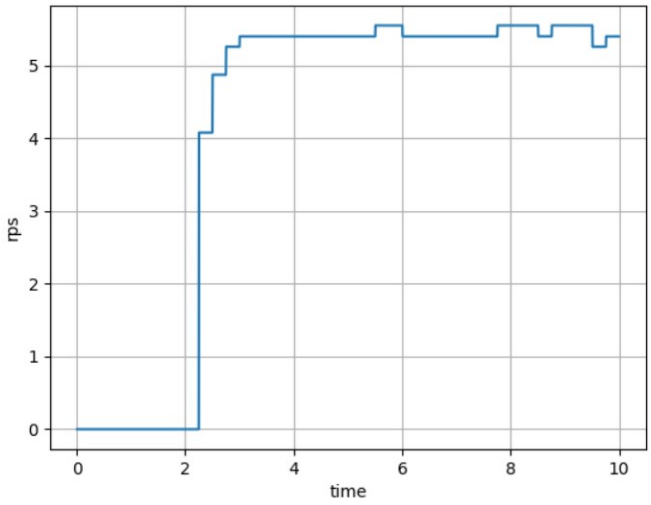

Fig. 7. A graph of the RPS over time using only proportional derivative control, and at ten times the experimentally determined ideal, or a proportion of 60 compared to the ideal six.

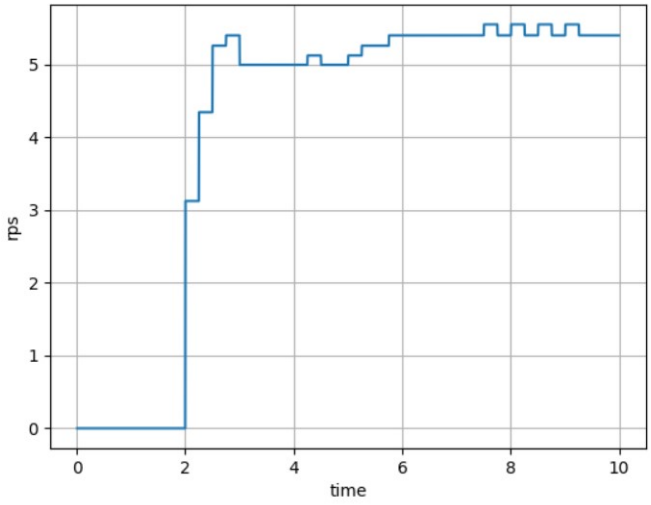

Fig. 6. A graph of the RPS over time using only proportional derivative control, and at half of the experimentally determined ideal, or a proportion of three compared to the ideal six.

state error was much higher than without integral control.

Based on this FFT, the value of the frequency of the spinning is about 5.5 which matches our RPS calculations. This reinforces the conclusion the RPS calculations are accurate, and ensures the AD readings are being interpreted correctly.

B. Objective Two: Movement

Objective #2 was to drive a rover up as close as possible to an object from a distance away without hitting the object, with the caveat the car could be turned up to thirty degrees from the target and would have to turn as it drove to face the object. To know from what distance away to stop the car, the ultrasonic sensor was used to ascertain accurate distance readings, and then the motor duty cycle was then reduced accordingly so as to stop the car as close as possible to the object. As for the angle at which the car could be turned from the object, the angle between the car's middle (i.e. midway between the car's front right and front left tire) and the object was calculated using a picture from the car's middle using trigonometric properties and the duty cycle of the camera's swivel was the adjusted to face the object, using come control as the calculated angle was not the true angle due to the nature of a 2-D image's inability to see three dimensional space. The steering servo were then set up so they would match the camera's swivel duty cycle, so that the car would turn so its middle was facing the object and would drive straight to the object. In the future, a different sort of control ought to be used which would minimize the change in duty cycle based on the angle to control the car's steering servo, as it would swing from the left to the right. Using the camera's swivel servo is also a possibility, but it would come with a much more difficult method of implementation, though it would allow the car to more easily follow a moving object.

C. Objective Three: Movement With Control

This **Objective** combined the goals from both previous objectives, and was to maintain a target RPS while driving towards an object, then stopping as close as possible to the object. To this end, the PID control, and so the success of **Objective #1**, was crucial to the success of this **Objective**. **Objective #2**'s method for stopping the car as close as possible from the object was beneficial, but could be circumvented using PID control by setting the target RPS to be zero, and so was in no way as vital as PID control. To fully

operate the picar for **objective #3** the following parts were used: an ultrasonic sensor, an AD converter, a photoresistor, three servos, a DC motor, and a camera. The AD converter, photoresistor and DC motor were used to control the speed of the picar. The three servos, the camera and the ultrasonic sensor were used to tell the picar which direction to travel. As explained above **objective #3** was the combination of **objective #1** and **objective #2**. So the explanation of the this **objective** is similar. The major differences were that the code had to be adjusted for the rps input instead of the duty cycle input. this caused the picar to take corners at a different rate so adjustments to the boundaries for the picar to slow down based on the ultrasonic sensor's output had to be made. To improve the functionality of the picar's turning it would be necessary to fine tune the cars turning and the rate at which it turned. In our picar we achieved the goal of getting close to the recycling bin without hitting it but the turns it took to get there were quite large turns. Another way that could improve functionality of the picars is to tighten the pivots on the steering arm. This causes a lot of turbulence when steering which causes some error in the turning radius of the picar. This means that the picar may be set to turn to a certain degree but due to the hardware error in the steering the servo may turn to a different degree.

III. Results and Discussion

The results of the objective are as follows. From **objective #1** the results are that in figure 3. The results show that the picar had little over shoot to the average rps. This shows that the values chosen for the PID controller were good values that did not cause overshoot, did not over damp the system. It as a good set of values to get a good response time. For **objective #2** the results were that the picar was able to drive to the blue recycling bin without running into it. The picar was able to do this by using the camera to take a photo then making a mask that looks for blue. From the mask photo the code analyzes the white part of the mask to determine which side of the photo has more blue, the picar would then turn in that direction. Once the picar got close to the bin the ultrasonic sensor would take over. Once the sensor detected the bin with in a certain range the power in to the motor that drived the picar was set to zero. So the result was the car not running into bin. The results of **objective #3** is that the picar was able to drive at the desired rps and get close to the recycling bin with out hitting it. The picar was able to get to the bin by similar code that was in **objective #2**.

IV. Conclusion

Using **Objective #1** and **Objective #2** to complete **Objective #3** was the goal of experimentation with the rover, and this was accomplished using hardware alongside code to interpret simple readings as several different pieces of information to create driving rover. More generally speaking, through the raspberry PI, it became possible to analyze data from several outputs (photoresistor, camera, ultrasonic sensor, etc.) to extrapolate information using formulas and strategies. Through this data, changes to several inter-working systems could be implemented through engineering strategies and design choices. Control strategies were a major component of the project, PID being the most prominent. A proportional control managed the steering servo duty cycle while PID managed the motors duty cycle, which encapsulates all of the changing duty cycles in this project. It would be possible to use more than those two servos to accomplish the goal in a more efficient manner (e.g. tighter turns, different landscape etc.) but this could require dependent systems working simultaneously, while the used method in the project only necessitated two independent systems. While the implemented methods in this project was relatively simplistic compared to other methods, it was able to accomplish the stated goals and was a product of many engineering strategies and their accompanying empirically determined design choices.

A. REPO LINK

https://github.com/ESE205/module-10-rodriguezmontoya-kuchak