Memo

Date: November 13, 2015

To: Inst. Schrock and GTA Yang

From: Group A - Nick Bova, Bryan Check, Tyler Sargent, Jordan Scully, Brad Sievers

Subject: Lab 09 Performance Test 3 - Energy

Introduction

The purpose of this lab was for the team to create and test two Arduino codes in order to determine the energy efficiency of the AEV while using each. According to the MCR, the AEV design should be as efficient as possible and complete the run in a timely manner. By using two different methods in order to complete the test run, the team could easily discern which sorts of commands best met these requirements and therefore which should be used during final testing. Because this was the final lab before testing, the team also used this time to pinpoint the proper timing and mark counts used in each code. This final modification stage is of the utmost importance, for in the engineering world employers and clients expect their product to perform flawlessly while accomplishing its objectives. As a result, engineers must utilize this crucial time period to ensure it does so. In this memo, the results of the lab are displayed and discussed as well as a few relevant figures. Individual conclusions are then given by each team member, and an appendix of data and references are included on the final pages.

Results & Discussion

At the beginning of this lab, only a few tasks needed to be completed in order to obtain the most efficient vehicle possible. Test code #1 was identical to the code used in Performance Test 2, therefore the team merely needed to run the AEV on the track and upload the data from the run to MATLAB. To create test code #2, the tea brainstormed possible new code ideas and decided upon modifying code #1 instead of starting from scratch. One group member increased the motor power of certain commands and added reverse thrust braking commands while the others dictated their ideas to him. After that, the code was uploaded to the AEV and tested on the track just as the first code was. The EEPROM data was uploaded to the computer, and the students used the AEV Analysis App to locate data trends and values to come up with the most efficient code.

The team's programming strategy during this lab was to produce a code that was clear, concise, and allowed the AEV to finish it’s objective in the most energy efficient manner possible. The two codes were designed with completely opposite methods in terms of how the AEV traveled to each section of the run. The first code (Figure A1) utilized a more coast-heavy method to gradually reduce its speed and reach each mark while traveling at a low speed (23% power on the way to the cargo and 30% power on the return trip). The second code (Figure A2) utilized a reverse thrust approach when

traveling to the gate and transporting the cargo back to the start to attempt to stop accurately and consistently at each checkpoint during every run. It also commanded the AEV to travel at higher speeds (30% power on the way to the cargo and 45% power on the way back) in hopes that this quicker run time would offset the amount of power used in the reverse-thrusting brake method. Designing two completely different approaches to the AEV proved helpful as it allowed a very obvious conclusion of which code was more energy efficient. Before the discussion of the individual performances of the AEVs, it is important to note that the tracks of either test room are not the same. While appearing to be of the same specifications it was observed that while in one testing room the AEVs traveled further than in the other testing room while using the same code, design, and motor power.

During the runs it became apparent that AEV code #2 (Figure A2) was far more accurate as it depended much less on the battery strength than AEV code #1 (Figure A1). AEV code #2 was able to very precisely approach the gate sensors and promptly slow down using a burst of reverse thrust almost every run, while the coasting prevalent in code #1 was inconsistent. However, the burst of thrust as well as the overall longer time spent at high motor power made the energy output of code #2 increase greatly, causing the group to question the validity of the code. This can be observed in Figure 1 below where AEV code #2 is represented by the orange plot and AEV code #1 is represented by the blue plot.

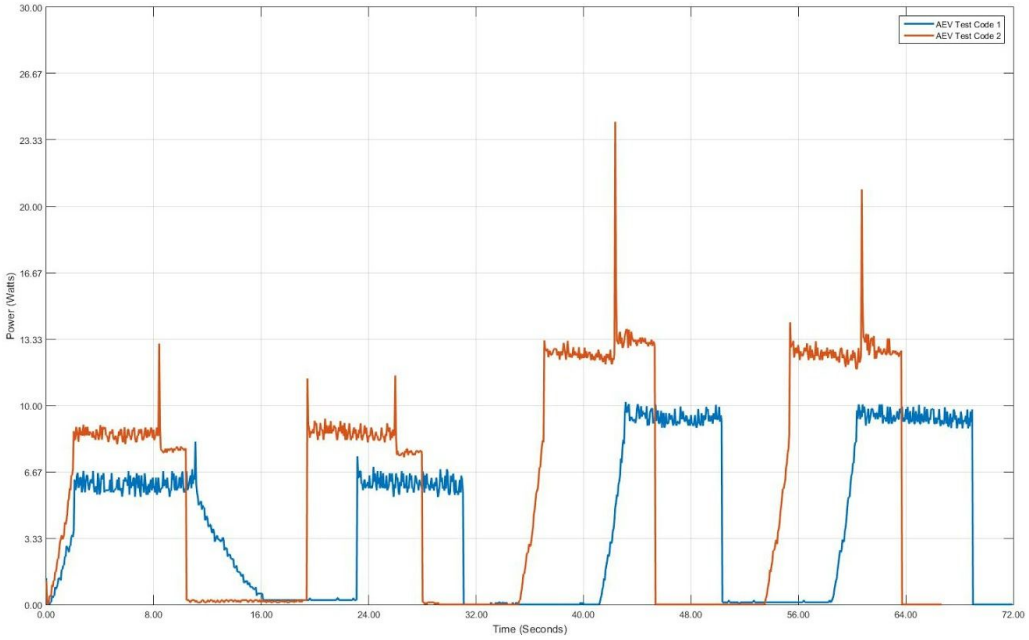

Figure 1: Graph of AEV Power vs. Time for Both Test Codes

Looking at the plots, it is quite evident that code #2 uses more energy during every phase of the test run. While AEV code 1 was less accurate and slightly slower, it was far more efficient than AEV code 2. Using code #1 allowed the AEV to start off with an initial speed and then drop off the power to allow a gentle coast towards the gate or cargo. This was more efficient as the motors spent less time actually running and it didn't require large bursts of energy to reduce the speed of the AEV.

Looking at the phase diagrams below (Figures 2 and 3), it can be seen that the motorSpeed command required the most amount of energy to use. In AEV code 1, these are phases 2, 5, 8, and 11. In AEV code 2, these are phases 2, 3, 5, 6, 9, 10. The high energy usage is a result of the nature of the command, which forces the motors to maintain a constant speed. This requires constant energy. AEV code #2 used the motorSpeed command far more frequently and also used much higher motor speeds, which caused much higher energy usage than AEV code #1.

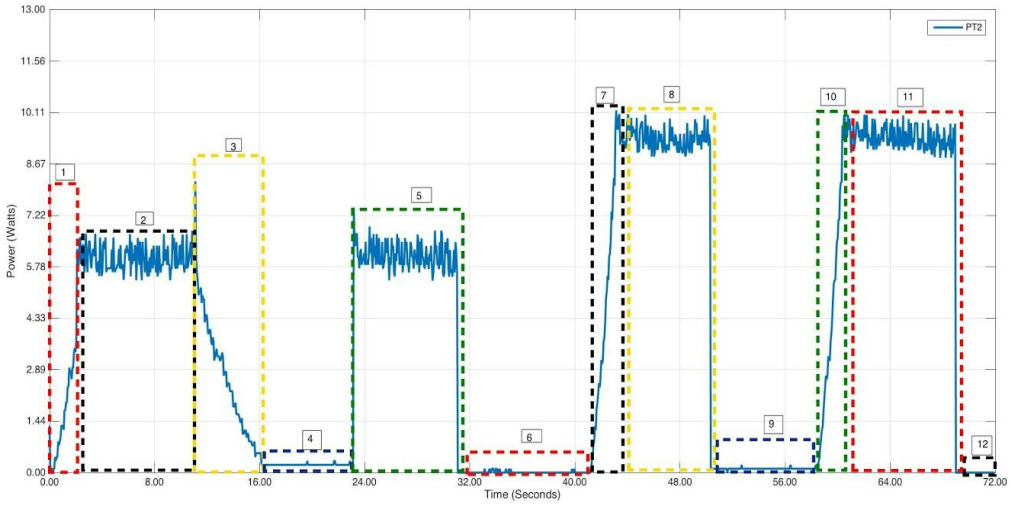

Figure 2: Graph of AEV Power vs. Time with Phase Divisions for Test Code #1

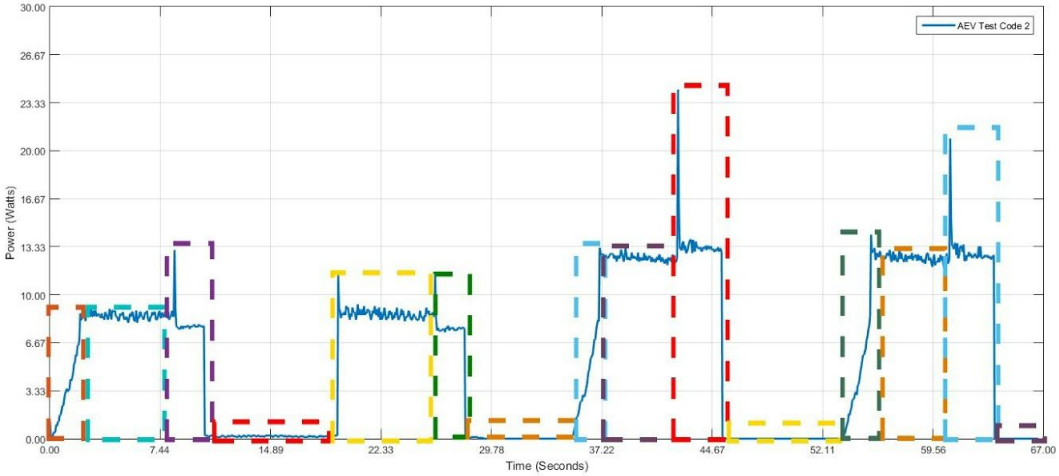

Figure 3: Graph of AEV Power vs. Time with Phase Divisions for Test Code #2

One can also observe that the AEV completed the mission in approximately 72 seconds using code #1 and in approximately 62 seconds using code #2. Time of run is indeed a factor in efficiency, however the amount of energy expended is a much more important factor so therefore this advantage is quite minimal. Also, although code #2 is faster, it actually runs more commands than code #1 does, decreasing efficiency. As seen in the phase tables (Table 1 and Table 2) below, the first code entailed 12 command phases while the second was composed of 15 phases.

Table 1: Phase Energy Breakdown Data For Test Code #1

Phase	Arduino Code	Time (s)	Total Energy (J)
1	celerate(4,0,23,2)	2	3.1178
2	motorSpeed(4,23)	5.52	32.9382
3	celerate(4,23,0,5)	5	28.4128
4	brake(4)	7	7.21
5	motorSpeed(4,23)	11.5	27.1768
6	brake(4)	10	0.0921
7	celerate(4,0,35,2)	2	6.194
8	motorSpeed(4,35)	7.4	68.4441
9	brake(4)	8	0.8979
10	celerate(4,0,35,2)	2	5.1353
11	motorSpeed(4,35)	9.06	83.9184
12	brake(4)	2.4	0
		Total Energy:	285.6431 J

As one can see, code #1 used a fairly balanced mixture of commands throughout the course of the program. The “celerate()” and “motorSpeed()” commands during this run performed quite similarly, as evidenced by phases 2 and 3. Phase two utilized “motorSpeed()” at 23% power and ran for 5.52 seconds while expending 32.9 J of energy. Phase three was “celerate()” to 23% power and although 4 less joules of energy were expended, it ran the command for half of a second less. This suggests that perhaps “motorSpeed()” is the more desirable command since it does not require the AEV to gradually build up to its maximum speed. This idea was explored in test code #2. The code in which the phase table represents is actually identical to the program used in Performance Test 3 and therefore has already been discussed at length in the previous memo. Because of this, the table will serve mainly as a comparative tool when analyzing the phase table of test code #2 below (Table 2). Arduino test code #1 expended a total of 285.6431 J of energy throughout the pickup and transport of the R2-D2 cargo.

Table 2: Phase Energy Breakdown Data For Test Code #2

Phase	Arduino Code	Time (s)	Total Energy (J)
1	celerate(4,0,30,2)	2	6.1266
2	motorSpeed(4,30)	6.2	46.3247
3	motorSpeed(4,30)	2.2	17.3141
4	brake(4)	9	1.1247
5	motorSpeed(4,30)	4.9	52.9853
6	motorSpeed(4,30)	2.1	39.6999
7	brake(4)	7	0.6785
8	celerate(4,0,45,2)	2	26.5853
9	motorSpeed(4,45)	2.7	41.4558
10	motorSpeed(4,45)	2	25.8743
11	brake(4)	8	0
12	celerate(4,0,45,2)	2	25.1071
13	motorSpeed(4,45)	2.8	45.66
14	motorSpeed(4,45)	2.7	49.0976
15	brake(4)	2	0
	Total Energy:		378.0339

As mentioned above, the 15 phases of AEV test code #2 proved to be less efficient than the 12 phases of AEV test code #1. This is evidenced by the fact that it took the AEV approximately 98 J of energy to reach the cargo (phases 1-5) using code #1 while it took the AEV about 162 J to reach the cargo (phases 1-6) using code #2. This large discrepancy in efficiency caused the team to seriously question the second code despite its super consistency. It was determined that this spike was due to the reverse thrust in phases 3, 6, 10, and 14. These four phases account for nearly 135 J of energy by themselves, almost half the entire amount of energy used by code #1. This was unacceptable to the team, and the total expenditure of 378.0339 J was not adequate in the eyes of the group. This large amount of power used was also due in part to the increased speed at which the AEV moved. Although it was determined in an earlier lab that the lower the motor power the better the AEV efficiency, the team thought that the faster run time may be worth it. This proved to be untrue.

There were a few potential sources of error in this lab. One example was the location of testing. The team performed this lab on a Friday and a Monday, meaning two different tracks were used. As touched upon earlier these tracks have different characteristics and require different parameters to complete. This affects the readings of efficiency of the codes as code one was exclusively created and tested on friday and code two was created and tested monday on another track. This variance should be considered when comparing the two codes. Another potential source of error was the starting

point of the AEV. In the Mission Concept Review it is stated that the vehicle must be started behind the piece of tape on the track, however there are a few inches of variability in where the AEV was actually started. In a lab as detail-oriented as this one, if the mark readings are off in the slightest the AEV could pass a crucial sensor or fail to reach a gate. As a result, the the team's code could prove to be slightly off when testing during the final lab.

Conclusions and Recommendations - Bryan Check

In this lab, the team was tasked with creating two Arduino codes in order to determine the most efficient commands and coding patterns for the AEV to use while completing its mission. The first code devised by the group (Figure A1) utilized coasting and low motor speed to try to conserve energy, and although it did indeed accomplish this goal the AEV did not run consistently. Due to track inconsistencies and battery power lossage, it proved to be quite difficult to get the vehicle to stop in the same place when run multiple times. In an attempt to combat this inconsistency, the group created a second code (Figure A2) that moved faster but also used reverse thrusting in order to halt the vehicle abruptly. This technique was successful, however when analyzing the results it was concluded that far too much energy was used (378 J) in comparison to the first code (285 J) to be viable. It was concluded that the higher motor power was the main reason for this as well as a few unnecessary reverse thrusts. Therefore, it is recommended that the team create a final code utilizing parts of both test codes. Most realistically, the group should simply add some reverse thrust commands at the points during the run that have resulted in inconsistent stopping so that the AEV can complete its mission correctly every time. It is advised that the speed of the AEV be kept at 23%, for it is just enough to move the AEV while maximizing efficiency.

The two major errors in this lab can most definitely be resolved. To combat the issue of inconsistent tracks, the group should simply write their code for the lower level track and ensure that they do their final testing on that track as well. If there is work time in the upstairs room, the group should not alter their mark numbers but instead work on future lab reports. The second potential error present in this lab was that the starting point wa not always the same. To ensure that this is not an issue, the student responsible for starting the vehicle should be sure to put the front of the front wheel on the back of the piece of tape every time the AEV is tested. That way, the mark readings will be the same for every run and the values will not have to be altered at all.

Conclusions and Recommendations - Jordan Scully

This lab had the team creating and comparing two different codes for the AEV. The two codes were created to complete the requirements as efficiently as possible. One code used more initial speed (code 2) and late braking and another code used more slow steady speed (code one). Each code had its own benefits. For example code one was more efficient and code two was easier to fine tune on the track. This is because code one used coasting as its main feature, meaning there was a lot more variance in the distance traveled because of different factors such as the battery level. The takeaway from this lab is finding a balance between efficiency and reliability in the AEV code. A sweet spot would lie between the two codes that were used. This knowledge can be used to create the perfect balance between the two used codes and creating a code that is both reliable and does not consume copious amounts of power. The error discussed above in practice does not affect the results enough to make analyzing difficult. This is because the power usage differs greatly between the two. A solution to this problem is the team is going to refrain from doing any further testing on the track used on fridays and exclusively use the monday tuesday track to avoid any conflicts with the track and codes. A recommendation for the future would be to prevent groups from having to switch rooms and waste time on two different tracks. Or at the very least preface the difference between the tracks earlier. Another potential change is to assign the groups a specific battery or two so that there are less issues with wild battery variance.

Conclusions and Recommendations - Tyler Sargent

In this lab students were to complete and test two AEV codes to see which of the two different methods was more efficient. AEV code 1 was far more efficient than AEV code 2 however, code 2 was far easier to control the exact position of the AEV. Code 1 utilized a coasting approach, saving a large amount of energy. This was efficient, yet at times ineffective for completing the MCR. Code 2 was created to fix this flaw giving more control over the exact location of the AEV during its stopping procedures. Code 2 fixed the issue of inaccuracy but used far more power than code 1. The comparison of energy consumption of each code can be seen in Figure 1. Code 2's use of a reverse thrust to help brake for the gates and crago phases used a high amount of energy on top of using more energy to keep the motors running for longer periods. Code 1's use of coasting allowed for minimal power usage, but suffered from major deviations in distance traveled due to heavy reliance on the power of the battery. The way to most effective meet the MCR would be to make a third code design utilizing low power and a coast as in code 1, as well as a very minimal reversed thrust celerate. This would make a hybrid utilizing the energy efficiency of code 1 while maintaining the accuracy of code 2. The main sources of error in this lab were battery charge levels, inconsistencies of the batteries, as well as the differences in the track between either testing room. For future labs or groups perhaps numbering the batteries and assigning each team a battery just as each team is assigned an AEV kit would help to keep the test more consistent in terms of power supplied. In order to solve the issues with the differences of the track, it could be maintained to minimize the differences.

Conclusions and Recommendations - Nick Bova

Many different factors must be taken into account in order to get the AEV running properly on the day of the team's final run. Specifically, while code 1 offered a great improvement in energy efficiency from code 2 (from 285J compared to 378J respectively), code 1 proved more difficult to work with due to its reliance on coasting to decrease energy usage. Considering the difficulties the team have with getting consistent test results due to factors such as battery charge and testing the vehicle on different rail systems, it is recommended that the coasting methods are not the only one relied on when finalizing the code. Many of the same errors experienced in this lab were also experienced in previous labs, so

Conclusions and Recommendations - Brad Sievers

During this lab, two separate AEV codes were designed by the team in order to test the energy efficiency of the AEV while using different techniques. One code utilized reverse thrust in order to abruptly stop, while the other used coasting in order to slow to a stop. AEV code 2 used much more energy than code 1, which can be seen in Figure 1. However, AEV code 2 was much easier to control on the test track. Therefore, it is recommended that a combination of AEV codes 1 and 2 be used in the final test run. This will allow for both aspects of the codes to be used: efficiency and ease of control. One source of error the team encountered dealt with the difference in test tracks. The distance that the AEV travelled was inconsistent between the two tracks. If the team is to complete the tasks in the MCR, using purely the coasting method or the reverse thrust method will not result in efficient completion. Using a combination of these two test codes will let the team complete the required tasks while still maintaining energy efficiency.

Appendix A

```
reverse(4);                         //Reverse all motors
celerate(4,0,23,2);                         //Accelerate all motors from 0 to 23% power in 2 sec

motorSpeed(4,23);                 //Run all motors at 23% power.
goToAbsolutePosition(-362);       //Go to an absolute position of -362 marks.

reverse(4);                        //Reverse all motors.
celerate(4,23,0,5);                  //Accelerate all motors from 23 to 0% power in 5 sec

brake(4);                             //Brake all motors
goFor(7);                             //Go for 7 seconds
                                      //Stop at gate

reverse(4);                           //Reverse all motors
motorSpeed(4,23);                  //Run all motors at 23 % power
goToRelativePosition(-338);       //Go to a relative position of of -338 marks

brake(4);                               //Brake all motors
goFor(10);                            //Go for 10 seconds
                                      //Pick-up Cargo

reverse(4);                               //Reverse all motors.
celerate(4,0,35,2);                //Accelerate all motors from 0 to 35%  power in 2 sec

motorSpeed(4,35);                  //Run all motors at 35% power
goToRelativePosition(355);        //Go to a relative position of 355.

brake(4);                               //Brake all motors
goFor(8);                             //Go for 8 seconds
                                      //Stop at gate

celerate(4,0,35,2);                //Accelerate all motors from 0 to 35% power in 2 sec

motorSpeed(4,35);                  //Run all motors at 35% power
goToRelativePosition(355);       //Go to a relative position of 355 marks

brake(4);                             //Brake all motors
                                      //Run Finished
```

Figure A1: Arduino Code #1 Used to Complete Mission

```
reverse(4);                          //Reverse all motors.
celerate(4,0,30,2);                  //Accelerate all motors from 0 to 30% in 2 sec

motorSpeed(4,30);                    //Run all motors at 30% power
goToAbsolutePosition(-368);          //Go to an absolute position of -240

//Brake for gate.
reverse(4);                          //Reverse all motors
motorSpeed(4,30);                    //Run all motors from 30 power for 2 seconds
goFor(2);

//Stop at Gate First Time
brake(4);                            //Brake all motors
goFor(9);                            //Go for 9 seconds

reverse(4);                          //Reverse all motors
motorSpeed(4,30);                    //Run all motors at 30 % power
goToRelativePosition(-390);          //Go to a relative position of of -350

//Brake for Cargo.
reverse(4);                          //Reverse all motors
motorSpeed(4,30);                    //Run all motors at 30 power for 2 seconds
goFor(2);

brake(4);                            //Brake all motors for 7 seconds.
goFor(7);

//Pick Up Cargo
celerate(4,0,45,2);                  //Accelerate all motors from 0 to 45 % in 2 sec
motorSpeed(4,45);                    //Run all motors at 45% power
goToRelativePosition(330);           //Go to a relative position of 355


//Brake for gate second time
reverse(4);                          //Reverse all motors.
motorSpeed(4,45);                    //Set all motors to run at 45% power for 3 sec
goFor(3);

brake(4);                            //Brake all motors for 8 seconds
goFor(8);
//Stopped at gate second time.

reverse(4);
```

```
celerate(4,0,45,2);                //Accelerate all motors from 0 to 45% in 2 sec
motorSpeed(4,45);                  //Run all motors at 45% power
goToRelativePosition(345);     //Go to a relative position of 360.

//Brake for final stop.
reverse(4);                          //Reverse all motors
motorSpeed(4,45);                //Run all motors at 45% power for 3 sec
goFor(3);

brake(4);                            //Brake all motors.
//Run Finished
```

Figure A2: Arduino Code #2 Used to Complete Mission

References

1. “Advance Energy Vehicle Design Project Lab Manual” https://eeiccourses.engineering.osu.edu/sites/eeiccourses.engineering.osu.edu/files/uploads/1182/AEVLab/AEVDocuments/LabManual/AEV_Lab_Manual_Rev_2015_08_07.pdf

2. “Technical Communication Guide” https://eeiccourses.engineering.osu.edu/sites/eeiccourses.engineering.osu.edu/files/uploads/resources/TechCommGuide/Tech_Comm_Guide_Rev_2015_07_16.pdf