Noah Teal, John Jeong, Tim Regrut
Professor Janiszewska
Executive Summary Lab 6
24 February 2016

Introduction

The objective of this lab was to develop a program to test the AEV. The purpose of this program was to check the balance of the AEV with the battery, find out which propeller speed would be the smartest choice, and make any changes if needed. The next task was to download the Arduino EEPROM data and convert the EEPROM data into Physical parameters. The student's following task was to write a script file in MATLAB to make the proper conversions. After the conversions were made, the students made a performance analysis and a graph to go with the data. This helped optimize the student's understanding of what happened in the lab and contribute to future decisions.

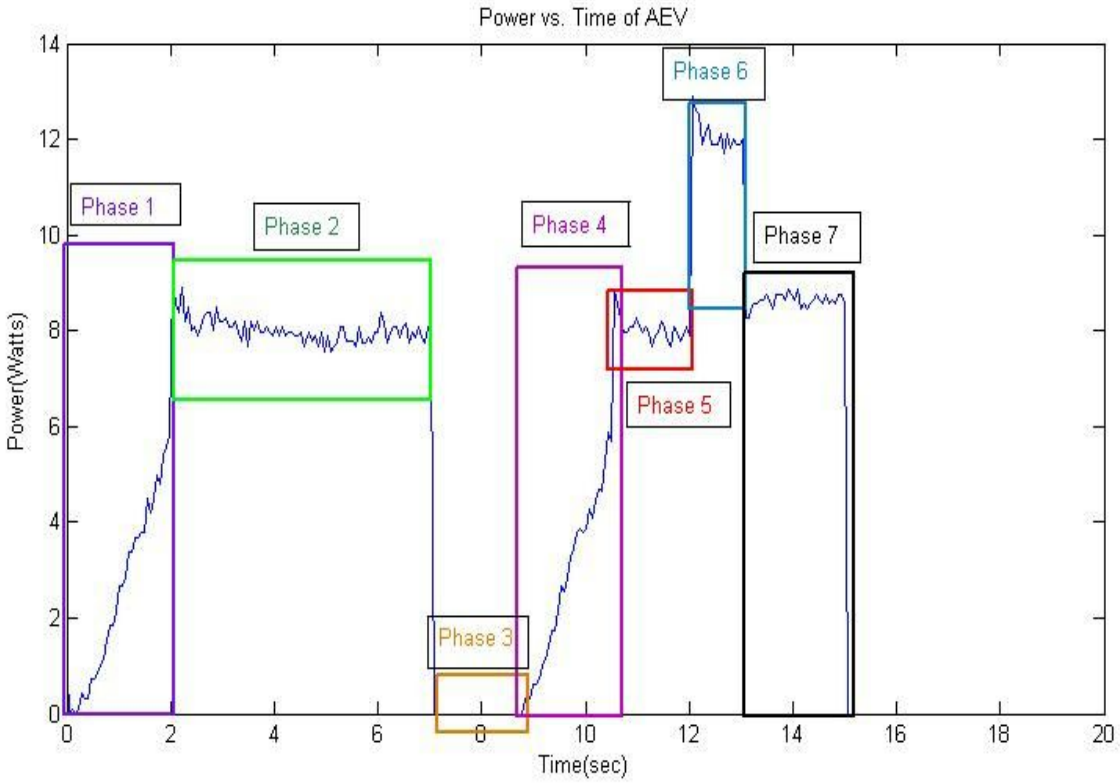
Results

The plots will help the team with their coding strategy in a number of ways. One way is that every phase where there is an increase or a decrease on the graph, there would be an acceleration of the AEV. Every phase where the graph is a straight line (or close to straight) is where the acceleration of the AEV is equal to zero and the velocity of the AEV is maintained. The results on the graph on the next page show that it takes more energy for the AEV to maintain its speed than it would for the AEV to accelerate or decelerate. This will affect the team's coding strategy because it will be more efficient to accelerate when the AEV needs power and then decelerate afterword to save energy. The code will have to be created with the AEV track in mind. The effective way to program the code would be to use conditional statements based on position to change the speed of the AEV to the lowest speed possible in order to maximize the efficiency of the AEV.

Certain recurring errors occurred in the coding department of design, where often initializations of variables were not set, indexes were out of bounds, etc. A quick crash course or review in the previous course's material (ENGR 1181.01) would resolve this error, since coding is a much knowledge-based skill. The format of the for loop, which starts with "loop" and an index to which it runs on, would not be known initially without a read in last semester's notes.

Since coding requires time—with editing, creating, and so on—a specific time frame should be devoted for it, preferably in an environment that is non-distracting and intellectually stimulating. One best example would be at office hours in Prof. Janiszweska's room, where any student could ask questions from an experienced resource. It is recommended to be at a campus room to perform optimally for creation of an academic piece.

Power Graph


Power vs. Time of AEV
Power(Watts)
Time(sec)
Phase 1
Phase 2
Phase 3
Phase 4
Phase 5
Phase 6
Phase 7
0
2
4
6
8
10
12
14
16
18
20

Phase	Arduino Code	Distance (m)	Time(s)	Total energy in each phase (j)
1	celerate(4,0,27, 2);	**0.0865**	**2**	**5.5242**
2	motorSpeed(4,2 7);goFor(5);	**1.8104**	**5**	**84.3222**
3	motorSpeed(4,0); goFor(1.5);	**0.6696**	**1.5**	**29.0003**
4	celerate(4,0,27, 2);	**0.0124**	**2**	**30.3225**
5	motorSpeed(4,2 6); goFor(1.5);	**.60347**	**1.5**	**28.1074**
6	motorSpeed(4,3 6);goFor(1);	**0.2356**	**1**	**17.6905**
7	motorSpeed(4,2 8);goFor(2);	**0.9796**	**2**	**33.9866**
total energy				**228.9537**

Arduino Code (w/ comments)
reverse(4); **(propellers start backwards, so the reverse function is required to orient pushing position forward)**

celerate(4,0,27,2); **(all motors accelerate to 27% power within a span of 2 seconds)**
motorSpeed(4,27); **(the same power is kept for an additional 5 seconds)**
goFor(5);
motorSpeed(4,0); **(all motors are put to a stop for 1.5 seconds)**
goFor(1.5);
celerate(4,0,27,2); **(all motors accelerate to previous power within a span of 2 seconds)**
motorSpeed(4,26); **(the power is lowered by 1% and kept for 1.5 seconds)**
goFor(1.5);
motorSpeed(4,36); **(all motors are increased in power to 36% for 1 seconds)**

```
goFor(1);
motorSpeed(4,28); (all motors are decreased in power to 28% for 2 seconds)
goFor(2);
```

Matlab Code

```
clear,clc

% this is initialization of all variables from the test run
data=xlsread('Data_AEV_run1.xlsx');
te=data(1:307,1);
ve=data(1:307,3);
ie=data(1:307,2);
marks=data(1:307,4);
pos=data(1:307,5);
pos=abs(pos);

% this is changing the EEPROM parameters into workable units and values
t=te./1000;
v=(15.*ve)./1024;
i=(ie./1024)*(2.46)*(1/.185);
d=0.0124.*marks;
s=.0124.*pos;
p=v.*i;

% graph calculated power and time
figure(1)
plot(t,p);
xlabel('Time(sec)')
ylabel('Power(Watts)')
title('Power vs. Time of AEV')

%observe total incremental energy of all the points
ej=1:length(p);
for i=1:1:length(p)-1;
   ej(i)=((p(i)+p(i+1))/2)*(t(i+1)-t(i));
end

%pass this point is the division and calculation of each phases; the function knnsearch
was used to pick the data range in the x axes; afterwards, the energy sum was calculated
with sum

%Phase 1
```

```
xR1 = 2.15;
xL1 = 0;
iL1 = knnsearch(t,xL1);
iR1 = knnsearch(t,xR1);
E_phase_1 = ej(iL1:iR1);

ep=1:length(E_phase_1);
for i=1:length(E_phase_1)-1;
   ep(i)=((E_phase_1(i)+E_phase_1(i+1))/2)*(t(i+1)-t(i));
end
y1=sum(ep);


%Phase 2
xR2 = 7.674;
xL2 = 2.15;
iL2 = knnsearch(t,xL2);
iR2 = knnsearch(t,xR2);
E_phase_2 = ej(iL2:iR2);

ep=1:length(E_phase_2);
for i=1:length(E_phase_2)-1;
   ep(i)=((E_phase_2(i)+E_phase_2(i+1))/2)*(t(iL2-1+i)-t(iL2-2+i));
end
y2=sum(ep);

%Phase 3
xR3 = 9.004;
xL3 = 7.674;
iL3 = knnsearch(t,xL3);
iR3 = knnsearch(t,xR3);
E_phase_3 = ej(iL3:iR3);

ep=1:length(E_phase_3);
for i=1:length(E_phase_3)-1;
   ep(i)=((E_phase_3(i)+E_phase_3(i+1))/2)*(t(iL3-1+i)-t(iL3-2+i));
end
y3=sum(ep);

%Phase 4
xR4 = 10.95;
xL4 = 9.004;
iL4 = knnsearch(t,xL4);
```

```
iR4 = knnsearch(t,xR4);
E_phase_4 = ej(iL4:iR4);

ep=1:length(E_phase_4);
for i=1:length(E_phase_4)-1;
   ep(i)=((E_phase_4(i)+E_phase_4(i+1))/2)*(t(iL4-1+i)-t(iL4-2+i));
end
y4=sum(ep);

%Phase 5
xR5 = 12.89;
xL5 = 10.95;
iL5 = knnsearch(t,xL5);
iR5 = knnsearch(t,xL5);
E_phase_5 = ej(iL5:iR5);

ep=1:length(E_phase_5);
for i=1:length(E_phase_5)-1;
   ep(i)=((E_phase_5(i)+E_phase_5(i+1))/2)*(t(iL5-1+i)-t(iL5-2+i));
end
y5=sum(ep);

%Phase 6
xR6 = 13.4;
xL6 = 12.89;
iL6 = knnsearch(t,xL6);
iR6 = knnsearch(t,xR6);
E_phase_6 = ej(iL6:iR6);

ep=1:length(E_phase_6);
for i=1:length(E_phase_6)-1;
   ep(i)=((E_phase_6(i)+E_phase_6(i+1))/2)*(t(iL6-1+i)-t(iL6-2+i));
end
y6=sum(ep);

%Phase 7
xR7 = 14.43;
xL7 = 13.4;
iL7 = knnsearch(t,xL7);
iR7 = knnsearch(t,xR7);
E_phase_7 = ej(iL7:iR7);

ep=1:length(E_phase_7);
```

```
for i=1:length(E_phase_7)-1;
   ep(i)=((E_phase_7(i)+E_phase_7(i+1))/2)*(t(iL7-1+i)-t(iL7-2+i));
end
y7=sum(ep);
```

Conclusion

The lab served to familiarize the students with the arduino and matlab syntax in coding. A draft of the arduino code was created, while the EEPROM parameters were outputted to observe the AEV with arduino code. The function knnsearch was introduced to set x axis intervals for phase calculations, where for loops were used to determine the incremental energies. Phase 2 used the most energy, with approx. 83 joules, and it was concluded that energy is significantly dependent on time the motor was ran. Overall, the students gained a greater understanding of the AEV and the programming needed to run the AEV.

Acknowledgments

The team would like to thank the Ohio State University, Their instructor, Professor Janiszewska, and the engineering department for giving the team the tools and resources necessary to complete this lab.