First-Year Engineering Program:

ADVANCED ENERGY VEHICLE
DESIGN PROJECT

AEV Lab Manual

C. Whitfield: Assistant Professor of Practice (Whitfield.22@osu.edu)
D. West: Graduate Research and Teaching Associate (West.425@osu.edu)
J. Allenstein: Graduate Research and Teaching Associate (Allenstein.1@osu.edu)
# Table of Contents

Lab Rules and Guidelines ......................................................................................................... 1

Introduction .................................................................................................................................. 2

*Curriculum Objectives* ............................................................................................................. 2

*AEV Design Cycle* .................................................................................................................... 3

Lab 1: Creative Design Thinking ............................................................................................. 4

*Example Concept Sketch* ......................................................................................................... 5

Lab 2: Arduino Programming Basics ...................................................................................... 6

*Arduino Sketchbook Setup* ....................................................................................................... 8

*Basic Function Calls* ............................................................................................................... 11

*Trouble Shooting* ................................................................................................................... 13

Lab 3: Concept Screening and Scoring ................................................................................ 15

Lab 4: External Sensors ........................................................................................................... 17

*Installing Reflectance Sensors* ............................................................................................... 17

*Testing the Reflectance Sensors* ............................................................................................ 18

*Sensor Function Calls* ............................................................................................................ 20

*Trouble Shooting* ................................................................................................................... 21

*Additional Sensor Function Calls* .......................................................................................... 21

Lab 5: System Analysis 1 ......................................................................................................... 22

*Downloading Arduino EEPROM Data* ................................................................................... 22

*EEPROM Data to Physical Parameters* .................................................................................. 23

*Supplied Power and Energy* .................................................................................................... 23
Lab Rules and Guidelines

The guidelines that must be followed at all times in the lab are as follows:

1. ALWAYS get circuit and program verified by instructional staff prior to plugging in the power.
2. DO NOT STAND IN THE PROPELLER PLANE.
3. DO NOT PUT YOUR FINGER IN THE PROPELLER.
4. No dangling jewelry or loose clothes.
5. No ‘open’ shoes. Close-toed shoes or boots only.
6. No climbing or standing on chairs or tables.
7. Be aware of sharp corners and edges which may exist on tables or on apparatus and tools.
8. Always know the location of the phone and of the first-aid kit.
9. Report to the instructor ALL injuries occurring during lab.
10. ALWAYS FOLLOW THE PROPER TESTING PROCEDURE (See Appendix)

Failure to follow these rules and guidelines may result in losing lab privilege with loss in lab participation grade for the course.
**Introduction**

AEVs are small (<500 grams), autonomous, electric motor-powered, propeller-driven vehicles that are suspended from and maneuver along monorail track systems hung from the laboratory ceilings. The AEV Kit includes PVC sheet structure, monorail wheels, electric motor(s), propeller(s), 2-cell lithium-polymer battery, and an in-house built AEV- Autonomous Control System (AEV-ACS) consisting of an ArduinoNano and motor controller(s), with externally mounted and adaptable sensors for feed-back control. An example AEV is shown below.

![Example AEV](image1.png)

**Figure 1: Example AEV**

**Curriculum Objectives**

The AEV design project has three main curriculum objectives:

1. *Project Management and Teamwork* - which includes, but is not limited to; time management and task scheduling, team communications and meetings, fair division of labor and team member responsibilities.

2. *Design Process* - which consists of identifying the project requirements and constraints, gathering background information, brainstorming, identification of management of materials, preliminary analysis and initial design, and the build/test/document cycle.

3. *Project Documentation* - which includes three parts:
   i. Project notebook - complete documentation of the project, and which was reviewed on a weekly basis
   ii. Final oral presentation - overview of design experience
   iii. Midterm and final written report - complete summary of all aspects of the design
AEV Design Cycle

Figure 1 below is schematic of the AEV design process.

Figure 2: AEV Design Process

AEV Project Objective
(Problem Definition)

INITIAL CONCEPTS
(Brainstorming)

EXPERIMENTAL RESEARCH
(Programming) (System Analysis)

ANALYZE

COMPARE

DESIGN DECISION

RESEARCH

FINAL DESIGN

Present AEV Design

Figure 2: AEV Design Process
Lab 1: Creative Design Thinking

Objectives:
1. Become familiar with techniques used for creative design thinking.
2. Practice two techniques.

Lab Activity: Work on constructing the sample AEV as a team. Brainstorm ideas for vehicle designs as a team. Work on individual concept sketches.

1.) Download the SampleAEV.pdf on the course website and start work on constructing the sample AEV as a team. Vehicle should be fully constructed by Lab 3: Concept Screening and Scoring. NOTE: You do not need to mount the motors and Arduino controller until after Lab 2.

2.) Work on individual concept sketches. The individual concept sketch should include a 3-view (orthographic) of the concept with major components identified. See example sketch in Figure 4.
Example Concept Sketch

Figure 3: Example AEV Concept Sketch
Lab 2: Arduino Programming Basics

Objectives:
1. Become familiar with the automatic control system hardware components.
2. Setup the software.
3. Program the basic function calls in controlling the AEV.
4. Be able to upload programs on the Arduino and test.
5. Become familiar with troubleshooting techniques.

Lab Activity: Follow this section of the AEV Lab Manual and the Grading Guidelines to acquire software, program basic function calls, upload the program to the Arduino to test, and demonstrate to a member of the instructional team.

Equipment: The following figure shows the AEV parts that are needed for this lab, and includes the following parts: motor stand, AEV controller, two motors, two propellers, and one battery.

There are two types of motor stands, a base made of nylon (shown above) or metal. This one is shown specifically to demonstrate that the Arduino should **NEVER** be set on **any metal** (including the metal hardware in the AEV Kit)!

Connecting the motors & battery to the AEV motor controller

**ALWAYS** make sure that the power switch on the side of the board is in the off position before making any connections.
Refer to Figure 5 and 6 and perform the following steps:

1) Select a pair of propellers to use for this lab.

2) Before we mount the propellers, it is important to note that propellers provide more thrust when they’re mounted with the dull side of the propeller (the side with writing) facing the direction the vehicle is traveling. For this lab, mount the propellers with the dull side facing away from the electric motor casing.

3) Place the propeller hub flat on the table (dull side facing the table). Now place the motor over the propeller and push the motor spindle into the propeller. Refer to Figure 6 for a visual representation.

4) The propeller should have a tight fit onto the motor spindle. If a propeller feels like it will slide off easily notify an instructional team member for a replacement.

5) Place Arduino controller and motors in motor stands as displayed in Figure 6.

6) Connect motors 1 & 2. Note: The motor connections on the Arduino will have male connections.

7) Make sure the power switch on the Arduino is in the OFF position. Orient the Arduino controller such that the red On/Off switch is on the left hand side facing you. The switch should be all the way to the right in the OFF position.

8) Connect the battery. Note: The battery connection on the Arduino will have a female connection.

9) DO NOT EVER PLUG THE MOTOR WIRE INTO THE BATTERY WIRE.

Your setup should look similar to Figure 6 with either nylon or metal stand. If using a metal stand do NOT set Arduino on the stand.

Figure 5: Mounting a Propeller.
**Arduino Sketchbook Setup**

To setup the Arduino software with the AEV program, follow these steps:

1. Save the **sketchbook** folder from the website on a flash drive or in your Z: drive.
2. Open the Arduino software from Start → All Programs → Arduino
3. Set the preference by selecting File → Preferences (you should see a window like the one in Figure 7).
4. Select "Browse" and select the sketchbook folder, then click ok. Note: Do not click into the sketchbook folder. Only select the folder (by making sure the folder is highlighted).

5. Now select File, then sketchbook. You should see the files that are in your sketchbook folder as shown in Figure 8.
6. Now close out of Arduino completely and restart the program. We are doing this step because now that we changed our sketchbook location Arduino needs to “reset” itself to take into account the change made. If you do not do this step, when you verify or upload your program Arduino will give you a “Metro” error.

7. Select AEV_Controller under File → Sketchbook. You should see a second Arduino window pop open. Select the tab named “01_myCode,” you will see a sample program is already included. You can close the original blank window. This is where you will be programming the functions for controlling your AEV.

8. Now refer back to the lab guidelines and complete the second part of Lab 2: Arduino Programming Basics Lab Procedure. Refer to the basic function calls below to complete your task. Once the basic function programming is complete and ready to upload to the Arduino, follow steps 9 through 20.
Basic Function Calls

- `celerate(m,p1,p2,dt);`
  - Four Arguments
    - `m`: Motor number 1, 2, 3, and 4 for all
    - `p1`: Start % speed (0 – 100)
    - `p2`: End % speed (0 – 100)
    - `dt`: Time span for acceleration in seconds (1 – 10)
  - Example: `celerate(4,20,45,2);`
    - Will accelerate all motors from 20% to 45% full power in 2 seconds
  - Example: `celerate(2,40,0,4);`
    - Will decelerate motor 2 from 40% to 0% full power in 4 seconds

- `motorSpeed(m,p);`
  - Two Arguments
    - `m`: Motor number 1, 2, 3, and 4 for all
    - `p`: % Speed (0 – 100)
  - Example: `motorSpeed(2,16);`
    - Sets motor 2 speed to 16% full power

- `goFor(dt);`
  - One Argument
    - `dt`: Time span in Seconds
  - Example: `goFor(5);`
    - Continues the last command for 5 seconds

- `brake(m);`
  - One Argument
    - `m`: Motor number 1, 2, 3, and 4 for all
  - Example: `brake(2);`
    - Cuts the power from motor 2.

- `reverse(m);`
  - One Argument
    - `m`: Motor number 1, 2, 3, and 4 for all
  - Example: `reverse(2);`
    - Reverse motor 2
9. Once you have saved your program you should see under File → sketchbook, the name of the saved program.

10. Connect your AEV board to your computer using the USB cable provided. Make sure the power switch is in the OFF position.

11. Now the proper Board and Serial Port must be set in the Arduino IDE.
   a. To set the board, go to: Tools → board, then select Arduino Nano w/ATmega328.
   b. To set the serial port, go to: Tools → serial port, and select the last COM port available in the list.
12. Compile your program to check for syntax errors by selecting the compile button. Then upload the program to the Arduino by selecting the upload button on the tool bar. See trouble shooting section if you have difficulties with this step.

![Figure 11: Arduino Compile and Load Button](image)

13. The Arduino software informs you that it is uploading and when it is complete.
   - While uploading, observe the lights flashing on the Arduino. If they are flashing fast, that is an indication that you have the proper board and com port set.
   - See trouble shooting section below if it does not upload properly.

14. Unplug the USB cable from the Arduino.

15. Turn on power switch (see Figure 9 above).

16. Wait until yellow light turns solid.

17. **Make sure that your team knows that you are about to run the program.**

18. Make sure nothing is in the propeller plane! It’s not common, but propellers can fly off and hit you.

19. Push the start button that is near the Arduino board of the automatic control system. See Figure 9. You will see the yellow light on the Arduino board flash 4 times before starting.

20. Once the program is finished, wait approximately 10 seconds, and then turn off the power switch.
   - The waiting portion is to allow the Arduino to download data. Even though we will not download the data until Lab 5, it is good practice to start.

**Trouble Shooting**

Problems that can occur that would cause an uploading error:

1. The Preference is not set properly to the "actual" sketchbook folder (not the folders inside of sketchbook). See step 3.
2. The proper board is not selected. See step 11.
3. The proper COM port is not selected. See step 11. (Most common trouble.)
4. Check the programming syntax. Are you misspelling a function call? Pay attention to the errors at the bottom of the Arduino window. They will guide you in finding out what is going wrong with the compiling or uploading of your program.

5. “Metro” error: Make sure the preferences are set according to step 3 and then CLOSE out of Arduino completely and re-open. This error will disappear.
Lab 3: Concept Screening and Scoring

Objectives:
1. Become familiar with techniques for design decision making.
2. Become familiar with a structured method to screen and score design concepts.
3. Program the sample AEV for a specific operation and test on classroom track.
4. Perform concept screening and scoring methods with AEV design concepts using the sample AEV as the reference.

Lab Activity: Follow this section of the AEV Lab Manual and the Lab Grading Guidelines to show an instructional team member a completed sample AEV construction (without wheel count sensor installed), program the automatic control system with specified characteristics, test the AEV on the classroom track following PROPER TESTING PROCEDURES (see Appendix), and perform the concept screening for each of the design concepts.

Procedure for uploading program to the sample AEV:
1. Open Arduino software
2. Select File → sketchbook → AEV_Controller
3. Follow Lab Grading Guidelines for specific programming instructions
4. Save Program
5. Get instructional team verification

Example Concept Screening Scoresheet
The following is an example of a concept screening scoresheet. Your team is required to determine the success criteria that you see fit for the AEV design, and screen and score the design concepts using the sample AEV as the reference.
<table>
<thead>
<tr>
<th>Success Criteria</th>
<th>Reference</th>
<th>Design A</th>
<th>Design B</th>
<th>Design C</th>
<th>Design D</th>
<th>Design E</th>
<th>Design F</th>
</tr>
</thead>
<tbody>
<tr>
<td>Balanced</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>-</td>
<td>0</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Minimal blockage</td>
<td>0</td>
<td>0</td>
<td>-</td>
<td>-</td>
<td>0</td>
<td>+</td>
<td>0</td>
</tr>
<tr>
<td>Center-of-gravity location</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>+</td>
<td>+</td>
<td>0</td>
<td>+</td>
</tr>
<tr>
<td>Maintenance</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>-</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Durability</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>+</td>
<td>0</td>
</tr>
<tr>
<td>Cost</td>
<td>0</td>
<td>+</td>
<td>-</td>
<td>-</td>
<td>0</td>
<td>-</td>
<td>0</td>
</tr>
<tr>
<td>Environmental</td>
<td>0</td>
<td>+</td>
<td>+</td>
<td>0</td>
<td>+</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>Sum +’s</th>
<th>Sum O’s</th>
<th>Sum –’s</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sum +’s</td>
<td>0</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>Sum O’s</td>
<td>7</td>
<td>5</td>
<td>4</td>
</tr>
<tr>
<td>Sum –’s</td>
<td>0</td>
<td>0</td>
<td>2</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>Sum +’s</th>
<th>Sum O’s</th>
<th>Sum –’s</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sum +’s</td>
<td>2</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Sum O’s</td>
<td>1</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>Sum –’s</td>
<td>2</td>
<td>3</td>
<td>1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>Sum +’s</th>
<th>Sum O’s</th>
<th>Sum –’s</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sum +’s</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>Sum O’s</td>
<td>2</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>Sum –’s</td>
<td>2</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

| Net Score             | 0        | 2        | -1       | -2       | 1        | 0        | 0        |

| Continue?             | Combine  | Yes      | No       | No       | Yes      | Combine  | Revise   |

*Figure 12: Example Concept Screening Scoresheet*
Lab 4: External Sensors

Objectives:

1. Become familiar with the external sensor hardware components.
2. Become familiar with troubleshooting techniques.
3. Program function calls for using external sensors with AEV control.

Lab Activity: Follow this section of the AEV Lab Manual and the Lab Grading Guidelines to install external sensors, test to make sure that it is working properly and demonstrate to instructional team member. Program basic function calls, upload the program to the Arduino to test, and demonstrate to a member of the instructional team.

*Installing Reflectance Sensors*

The following figure shows the reflectance sensor and a wheel with reflective tape that is used with the sensor to track the AEV distance and use a feedback control mechanism with the basic commands from Lab 2.

![Figure 13: Reflectance Sensor (left) and Wheel with Reflective Tape (right)](image)

The reflectance sensors are wired to connect directly to the automatic control system. Both of the vertical support arms provided in your kits have direct capability to mount the sensors on the opposite side of the wheel. You will use the small #2 bolt, nut, and zip tie in the AEV kit to secure the two reflectance sensors to the vertical support. See Figure 14.
The figure below shows the reflectance sensor ports on the automatic control system. The 3-prong plug must be oriented such that the white wire must be in the port closest to the mini-USB connection on the automatic control system. See Figure 15 below.

**Important Note**: Make sure the connections are installed with the white wire facing the Arduino mini-USB connection.

**Figure 15: Reflectance Sensor Connection**

It is always a good idea to verify, where possible, any new addition to a project. In this case, after mounting the sensors, you need to test that they work properly. Conduct the following steps:

**Testing the Reflectance Sensors**
1. Open Arduino software: Start → All Programs.
2. Set sketchbook preferences if needed (see Lab 2).
3. Type in “reflectanceSensorTest();” into 01_myCode
4. Connect the Arduino controller using the USB cord provided and upload the code to your Arduino. NOTE: Keep the Arduino controller connected to the computer throughout this test.

5. When the code is uploaded, open the serial monitor: Tools → Serial Monitor.

6. Set the monitor to 115200 baud, located on the bottom of the serial monitor.

7. Once the serial monitor is open, there should be a series of “1” scrolling. Press the start button on the Arduino board and after 4 seconds, spin the wheel on your AEV. You should see numbers increasing or decreasing by 1 and whether the system is moving forward or reverse. If this is not what you see, go back over Installing Reflectance Sensors and check your work. If you continue to have problems ask an instructional team member for help.

8. Now take the time to orient your sensors appropriately. If your vehicle is moving in the forward direction but you’re decreasing in counts, then the sensors connections on the Arduino controller need to be switched. Get instructor team verification.

The counts in the serial monitor are similar to how the counts are accumulated in the AEV_Controller code. The goToRelativePostion and goToAbsolutePostion functions were written specifically for this sensor to assist in stopping the vehicle, changing speed, or reversing the rotation of your propeller.

Note: Each wheel has 8 marks and a circumference of ~3.9 inches. Therefore the conversion from marks to distance traveled is 3.9 inches per 8 marks or 0.4875 inches/mark.
**Sensor Function Calls**

- `goToRelativePosition(c);`
  - One Argument; works specifically with reflectance sensor
  - `c`: Number of wheel counts (from current position)
  - The `goToRelativePosition` function continues the previous command for `c` marks from the vehicle's current position. `c` can be a positive or negative value. A positive value indicates the vehicle is moving forward from its current position. A negative value indicates the vehicle is moving backward from its current position.
  - Example Scenario 1: Your vehicle has already traveled 200 marks from the starting point on the track and is stopped at this position. The next lines in the code are:
    ```
    motorSpeed(4,20);
    goToRelativePosition(30);
    ```
    The code above will set all motors to 20% power and run the motors until the vehicle reaches 230 marks.
  - Example Scenario 2: Your vehicle has already traveled 200 marks from the starting point on the track and is stopped at this position. The next lines in the code are:
    ```
    reverse(4);
    motorSpeed(4,20);
    goToRelativePosition(-30);
    ```
    The code above will reverse all motors, set all motors to 20% power and run the motors until the vehicle reaches mark 170.

- `goToAbsolutePosition(c);`
  - One Argument; works specifically with reflectance sensor
  - `c`: Number of wheel counts (from current position)
  - The `goToAbsolutePosition` function continues the previous command until the vehicle has traveled to mark `c` relative to the starting position of the vehicle.
  - Example Scenario 1: Your vehicle has already traveled 200 marks from the starting point on the track and is stopped at this position. The next lines in the code are:
    ```
    motorSpeed(4,20);
    goToAbsolutePosition(300);
    ```
    The code above will set all motors to 20% power and run the motors until the vehicle reaches 300 marks from the starting point. The equivalent of this using `goToRelativePosition` is:
    ```
    motorSpeed(4,20);
    goToRelativePosition(100);
    ```
Example Scenario 2: Your vehicle has already traveled 200 marks from the starting point on the track and is stopped at this position. The next lines in the code are:

```
reverse(4);
motorSpeed(4,20);
goToAbsolutePosition(0);
```

The code above will reverse all motors, set all motors to 20% power and run the motors until the vehicle reaches mark 0 (starting point). The equivalent of this using `goToRelativePosition` is:

```
reverse(4);
motorSpeed(4,20);
goToRelativePosition(-200);
```

**Trouble Shooting**
The possible problems that can occur that would cause an uploading error (in order):

1. Check to make sure that the sensors are not mounted too far away from the wheel.
2. Check to make sure that reflectance sensors are installed in the right 3-prong ports on the automatic control system.
3. Check to make sure that the ground wire is the port that is closest to the mini-USB port.
4. Check the actual reflectance sensors by running the “reflectanceSensorTest();” Arduino Code.

**Additional Sensor Function Calls**
NOTE: The Arduino program, AEV_Controller, can be used to write more complex algorithms to assist in AEV operation. Arduino programming is NOT required for this class thus any experimentation or exploration done by you is solely your responsibility.

Arduino has a large community that shares programming details and information. For those that would like to dig further the two best places to start learning Arduino is (1) Arduino reference page: [http://arduino.cc/en/Reference/HomePage](http://arduino.cc/en/Reference/HomePage) and (2) googling your question. There are many different sites where Arduino users share helpful programming tips.

In addition, there are function calls available to you that may be helpful to use in various ways such as in while loops, for loops, conditional statements, etc. These function calls and details on how to appropriately call them can be found in tab “_00_AEV_key_words” in the AEV_Controller program.
Lab 5: System Analysis 1

Objectives:
1. Be able to download data from the automatic control system.
2. Be able to convert EEPROM Arduino data readouts to physical parameters.
3. Be able to calculate the supplied power to the AEV.
4. Be able to calculate the supplied incremental and total energy to the AEV.

Lab Activity: Follow this section of the AEV Lab Manual and the Lab Grading Guidelines to program the automatic control system with specified characteristics, test the AEV on the desktop track, download data from the automatic control system, and convert data to physical parameters and conduct supplied power and energy calculations.

Downloading Arduino EEPROM Data
To download data from the automatic control system, follow these steps:

1. First and foremost, after your AEV program is done running turn off the power to conserve battery life (this is **AFTER** the yellow light becomes solid again, else you will lose all your data!).
2. Open the sketchbook folder.
3. Go to the AEV Documents page on the EEIC website and download the zip file labeled “Data Recorder”. You can save this file where you have saved your sketchbook. Unzip the file. You should see a MATLAB file called “aevDataRecorder”. Open this in MATLAB. This MATLAB function downloads EEPROM data from the Arduino controller and saves the data to an Excel spreadsheet for further analysis.
4. Connect the Arduino controller to the computer using the mini-USB cord. You should see the yellow LED on the Arduino flash several times before turning solid.
5. If the LED is not turning solid, ask an instructional team member for help.
6. There is one of three ways to use the aevDataRecorder function provided to you. In the MATLAB you can execute the function in one of three ways:
   i. Type aevDataRecorder in the MATLAB command window:
      - No input needed. The program will ask you for a file name and you can place the file anywhere you like outside the MATLAB directory.
   ii. Type aevDataRecorder('System_Analysis_1'); in the MATLAB command window:
      - Including a string input such as ‘System_Analysis_1’ will use this as the excel file name and save the Excel file in the current MATLAB directory (your sketchbook folder since the aevDataRecorder file is in your sketchbook folder).
   iii. Pressing the Run button at the top at the screen.
• Executes the function just as first method described.

7. When you call the aevDataRecorder function the program will indicate in the command window what steps it is going through. When the Program indicates it is finished you have successfully downloaded the Arduino EEPROM data.

8. If you run into any errors ask for help from an instructional team member.

9. Refer back to the System Analysis 1 Procedure Guidelines for further instruction.

**EEPROM Data to Physical Parameters**

The data recorded with the automatic control system is in the form of integers (bytes) that must be converted to physical parameters to use in calculations. The Arduino data available includes: EEPROM time (TE), current (IE) and voltage (VE) supplied to the electric motors. The automatic control system also records the Marks from the reflectance wheel count sensor. Using Mark data will be the focus of System Analysis 2.

The following microprocessor conversion equations are used to convert the EEPROM equivalent data to the physical parameters of time (sec.), current (amps), and voltage (volts), where VR is the integer data for the reference voltage for the system.

**Time:**

\[ t = \frac{t_E}{1000} \]

where:

- \( t \) = time (seconds)
- \( t_E \) = EEPROM time (milliseconds)

**Current:**

\[ I = \left( \frac{I_E}{1024} \right) \times V_R \times \left( \frac{1 \text{ Amp}}{0.185 \text{ Volts}} \right) \]

where:

- \( I \) = current (amps)
- \( I_E \) = EEPROM equivalent current
- \( V_R \) = Arduino reference voltage

**Voltage:**

\[ V = \frac{15 \times V_E}{1024} \]

where:

- \( V \) = voltage (volts)
- \( V_E \) = EEPROM equivalent voltage

**Supplied Power and Energy**

Now that the EEPROM equivalent data is converted into physical parameters, the next step is to analyze the data to determine important parameters relevant to operating the AEV. For the first
system analysis activity, this includes calculating the power and energy supplied to the vehicle. First calculate the power (watts) supplied to the vehicle using the voltage and current:

\[ P_{in} = V \times I \]

where:  
- \( P_{in} \) = power (watts)  
- \( V \) = voltage (volts)  
- \( I \) = current (amps)

The supplied energy (joules) is a function of the supplied power (watts) and time (sec.). The energy supplied to the AEV can be thought of as both the incremental energy, similar to the incremental power, and the total energy supplied throughout the entire AEV operation. The amount of energy used corresponds to the area under the curve of power-time plots, with power on the y-axis and time on the x-axis. The method used to determine the energy will be using a rectangular approximation of the power and time data. There are three approximation methods that use neighboring data points for both power and time, they include: left-end point, right-end point, and midpoint (shown below).

\[ \text{Incremental Energy:} \quad E_i = \frac{P_{1in} + P_{2in}}{2} \times (t_2 - t_1) \]

The total energy corresponds to the sum of all of the incremental energies.

\[ \text{Total Energy:} \quad E = \sum_{n=1}^{N} E_n \]
This is a very powerful tool to be able to use in summing up portions of the AEV operation to determine how components of the programming and AEV operation contribute to the total supplied energy.
Lab 6: System Analysis 2

Objectives:
1. Demonstrate capability to download data from the automatic control system.
2. Demonstrate capability to convert EEPROM Arduino data readouts to physical parameters.
3. Demonstrate capability to calculate the supplied power, incremental and total energy to the AEV.
4. Be able to calculate performance characteristics of the AEV.

Lab Activity: Follow this section of the AEV Lab Manual and the Lab Grading Guidelines to program the automatic control system with specified characteristics, test the AEV on the classroom track, download data from the automatic control system, convert data to physical parameters, conduct supplied power and energy calculations, and calculate AEV performance characteristics that include velocity, kinetic energy, propeller RPM, propeller advance ratio, and propulsion efficiency.

**AEV Performance Characteristics**

Now that the EEPROM equivalent data is converted into physical parameters, the next step is to analyze the data to determine important parameters relevant to operating the AEV. The first step is to utilize the reflectance sensor to calculate distance traveled, velocity, and, with the AEV mass, the kinetic energy of the AEV.

**Distance:**

\[ s = 0.0124 \times \text{Marks} \]

where:  
- \( s \) = distance (meters)  
- Marks = EEPROM reflectance sensor marks

For velocity and subsequently, kinetic energy, a backward differencing scheme will be used. Visually this can be compared to left-end point approximation from System Analysis 1, where the velocity at point 2 is the difference between point 2 and 1 divided by the difference of time between 2 and 1. (Velocity at point 1 is equal to zero since this is the starting point.) This is mathematically represented by:

\[ v_i = \frac{(s_i - s_{i-1})}{(t_i - t_{i-1})}; \text{ Example: } v_2 = \frac{(s_2 - s_1)}{(t_2 - t_1)} \]

where:  
- \( v \) = velocity (meters / sec.)  
- \( s \) = distance (meters)  
- \( t \) = time (sec.)
Kinetic Energy: \[ KE = \frac{1}{2}mv^2 \]

where:
- \( KE \) = kinetic energy (joules)
- \( m \) = AEV mass (kg)
- \( v \) = velocity (meters / sec.)

**Propulsion Efficiency**

The efficiency of the AEV propulsion system \( \eta_{sys} = \eta_{motor} \& \eta_{propeller} \) is defined as the ratio of the power available (output) and the power supplied (input). The efficiency is a function of both the forward speed \( (v) \) of the vehicle and the propeller RPM \( [\eta = f(v, RPM)] \). This in turn is a function of the mass of the vehicle and the layout of the components on the AEV.

Needless to say, that for a simple ratio, the physics behind input and output power can be very complex. However, understanding the propulsion efficiency for your AEV is very important, specifically, when closing the loop between supplied power and energy (input) & available power and energy (output).

The next lab, System Analysis 3, will discuss how to determine the available propulsion characteristics, including efficiency, in detail for your AEV. The purpose of System Analysis 2 is to expose you to the variables used in determining the propulsion efficiency and how they are related to each other.

**Background:** There is a powerful tool that all engineers should at least be aware of when dealing with a variable that is a function of multiple independent variables, such as efficiency above. This tool is known as Buckingham Pi Theorem. When applied using the fundamental variables such as mass, time, and distance, the theorem determines non-dimensional parameters that are very useful by decreasing the amount of independent variables required.

For example, the AEV propulsion efficiency \( (\eta) \) which is a function of velocity and RPM can be reduced to a single non-dimensional parameter. This non-dimensional parameter is called the propeller advance ratio \( (J) \). Now \( \eta = f(J) \) only.

The propeller advance ratio is determined by the following equation:

**Propeller Advance Ratio:**
\[ J = \frac{v}{(RPM/60) \times D} \]

where:
- \( v \) = velocity (meters / sec.)
- RPM = propeller revolutions per minute
The propeller RPM is a function of the supplied current, and was determined using a strobe tachometer. The following trendline equations are provided for both propeller options:

**Propeller RPM:**

\[
RPM_{3\text{inch}} = -64.59I^2 + 1927.25I - 84.58
\]

\[
RPM_{2.5\text{inch}} = -17.64I^2 + 690.375I + 99.77
\]

where:

- \( \text{RPM} \) = propeller revolutions per minute
- \( I \) = current (amps)

For System Analysis 2, you will be provided a sample efficiency plot. However, it is very important to understand that each propeller and electric motor is different, and that System Analysis 3 will discuss how to analyze your own propeller and motor combination.

**Note:** After conducting System Analysis 3, you will return to this section and update the sample equation with your own propulsion system characteristics; closing the loop between supplied power and energy (input) & available power and energy (output).

**Sample propeller efficiency trendline:**

**Propulsion Efficiency:**

\[
\eta = -1205 * J^3 + 1033 * J^2 - 179.4 * J + 17.91
\]

where:

- \( \eta \) = propulsion efficiency
- \( J \) = propeller advance ratio (non-dimensional parameter)

**Note:** Experimental equipment, Arduino included, has what is termed “noise.” This is where you have data that is received by the hardware that is outside of the bounds tested. To account for this, there is a topic known as experimental methodology that discusses how to properly filter experimental data. This advanced topic can be very complex, and out of interest of time, we will simply apply conditions to the data above, specifically with propeller advance ratios, that we know to be beyond the bounds of the trendline and that would require extrapolation.

For the data above, the advance ratio, and hence the propulsion efficiency curve trendline, ranged from \( 0.15 \leq J \). Therefore any \( J \) value less than 0.15 is either 0 (off) or 0.15, depending on where within the program this occurs, i.e. no power or partial power supplied to the motors.

Keep in mind that a max value may have occurred at a different power setting. We will not be able to acquire the proper value because of this, so by setting the maximum value of \( J \) we are minimizing the error to the known value accordingly. This can be represented for clarity by the following expression:
**Propeller Advance Ratio:** \[ J = \begin{cases} 0 & \text{for } J \leq 0.15 \text{ with no power} \\ 0.15 & \text{for } J \leq 0.15 \text{ with power} \end{cases} \]

This should be applied to your experimental data in System Analysis 2. Hint: To apply constraints like the one above we can use nested IF statements in Excel. An IF statement in Excel has the following syntax

\[
\text{IF(logical_test, [value_if_true],[value_if_false])}
\]

Since we have three logical statements we need to test for we can create nested if statements as follows:

Say for example, we have a value for the advance ration, J, in cell T12 and we have a value for supplied power, in cell M12, the Excel input would be:

\[
=\text{IF(AND(T12<=0.15,M12=0),0,IF(AND(T12<=0.15,M9>0),0.15,T12))}
\]
Lab 7: System Analysis 3

Objectives:
1. Become familiar with propulsion system efficiency.
2. Become familiar with the wind tunnel and thrust stand equipment.
3. Conduct wind tunnel testing on electric motor and propeller.
4. Conduct analysis of wind tunnel testing results.
5. Update propulsion efficiency performance characteristics.

Lab Activity: Follow this section of the AEV Lab Manual and the Lab Grading Guidelines to conduct wind tunnel tests, analyze results, and update your AEV propulsion efficiency performance characteristics from System Analysis 2.

Equipment: The following figure shows the wind tunnel and testing equipment that are needed for this lab. This includes the wind tunnel, speed controller, power supply, thrust stand, and data acquisition system. See Figure 17.

![Wind Tunnel Equipment Diagram]

Figure 16: Wind Tunnel Equipment

The goal is to determine how efficient the propulsion system is by (1) setting the input power (voltage and current) supplied to the motor, (2) measuring the power output from the electric motor and the propeller using a thrust stand. Recall that the propulsion efficiency is the ratio of
the input and output power, and that the output power for the AEV is a function of the vehicle velocity (v) and propeller RPM, or the non-dimensional advance ratio parameter (J).

To measure the power available (output) from the electric motor and propeller, you will use a calibrated thrust stand to measure thrust for known settings of input power, by incrementing the voltage and recording the voltage and current supplied to the electric motor from the power supply.

Recall that the efficiency, hence the power available, is a function of velocity and propeller RPM. We will use the wind tunnel to control the simulated velocity of the AEV, and known propeller RPM characteristics with the current supplied by the power supply, to calculate the propeller advance ratio. The thrust available will be measured and the power available calculated using the wind tunnel speed. This ultimately obtains the propulsion efficiency for your specific AEV.

In this lab, there will be multiple wind tunnels in the front of the classroom. Each wind tunnel will have a different propeller configuration. Propeller configuration is an important aspect to consider in your team’s AEV design. There are two classifications of propeller configuration. There is the (1) tractor (or puller) configuration and (2) the pusher configuration. See Figures 17 and 18. For the purposes of this lab, your team will be testing out one of these configurations (if time permits, both configurations can be tested).

Figure 17: Electric Motor Mounted in Puller (tractor) Configuration
Wind Tunnel Testing Procedure

The following procedure will help you set up and conduct the wind tunnel test.

1. When your team is called up to the wind tunnel, one team member should bring a printed copy of the Wind_Tunnel_Testing.xlsx spread sheet and a pen/pencil.
2. Verify which configuration you are working with by writing down either “pusher” or “puller” on the wind tunnel spread sheet.
3. Turn on the power supply. Verify that the voltage on the power supply reads 7.4 volts and the current knob is turned all the way clockwise. Note: The 7.4 voltage setting is representative of what a fully charged battery would supply to the Arduino Motor controller.
4. Turn on the Thrust Stand Data Acquisition System (DAQ) (see Figure 20).

![Figure 20: Thrust Stand Data Acquisition (DAQ) System](image)

5. Unlock the moment arm so that the arm is resting on the scale. DO NOT REZERO THE SCALE AT THIS POINT. See Figure 21.

![Figure 21: Moment arm disengaged (LEFT) and Moment arm engaged (RIGHT)](image)

6. Turn on the wind tunnel speed controller. See Figure 22 (left image) below.

7. Verify that the velocity speed indicator is set to approximately ~2.8 meters/sec. See Figure 22 (right image) below.
8. Take an initial reading. Record the (1) percent power, (2) current, and (3) thrust scale reading. This value will be used to reference zero.

9. The Arduino Control System (see Figure 23) will control the power supplied to the motor. The (1) button increases the percent power in 5 percent increments. The (2) button decreases the percent power by 5 percent increments. The (3) supplies and cuts power to the motor. The (4) button changes the direction the propeller rotates.
10. Check that the propeller is rotating in the right direction.
   a. Set the power to 25%.
   b. Look very carefully at the propeller. As you push the (3) button determine if the propeller is rotating counterclockwise.
   c. Push the (4) button to reverse the direction of the propeller if needed.
11. On the Arduino Control System, push the (1) button until the Power reads 60%.
12. By selecting the (3) button, this will supply power to the propeller.
13. Take a reading [(1) percent power, (2) current, (3) thrust scale reading and (4) RPM] at each power setting going from 60% to 10% in 5% increments. Use the (2) button to decrease the power in 5% increments.
14. When your team reaches 10 percent power press the (3) button to cut power to the motor. Turn off the wind tunnel and the power supply.

After the wind tunnel testing is completed clean up around the area for the next team. Continue with the following procedure to analyze the wind tunnel test results.

15. Transfer hardcopy recorded data to Wind_Tunnel_Testing.xlsx and save the file.
16. Distribute the Wind_Tunnel_Testing.xlsx to each team member.
17. Conduct analysis calculations requested in the Lab Grading Guidelines.

The following equations are used for the performance calculations of the propulsion system from the wind tunnel testing.

Wind Tunnel Data Analysis
The thrust stand set up uses a 200 gram counter weight. Since we used a counter weight in our set up we need to account for this in our thrust reading through calibration. The equation for thrust calibration is:

$$ T_c = 0.411 * (T - T_0) $$

where:
- $T_c$ = Calibrated Thrust (grams)
- $T$ = Thrust scale reading (grams)
- $T_0$ = Thrust scale reading at 0% power (grams)

The input power supplied to the electric motor is determined by the voltage and current recorded and the power setting.

Power Input:

$$ P_{in} = V * I * (P_{\%} / 100) $$

where:
- $P_{in}$ = power input (watts)
- $V$ = voltage (volts)
- $I$ = current (amps)
- $P_{\%}$ = Arduino Power Setting
The power available (output) is measured by the velocity at which you are moving and the thrust required to create that movement. **NOTE: See the conversions below this formula for calculating the output power correctly.**

\[
\text{Power Available: } \quad P_{\text{out}} = T_c \times v
\]

where:  

- \( P_{\text{out}} \) = power available (watts)  
- \( T_c \) = calibrated thrust (Newton)  
- \( v \) = wind tunnel velocity (m/s)

In order to compute the output power correctly, the product of the thrust (recorded in grams, not newtons) and the wind tunnel velocity (m/s) can be converted to watts using the following unit conversions below. Notice that even though typically mechanical systems express power in units such as horsepower they can easily be converted to watts.

- 1 gram = 0.002205 lbs.  
- 1 m/s = 3.28 ft/s  
- 1 horsepower = 550 lb*ft/s  
- 1 horsepower = 745.7 watts

Or we could convert the measured value of thrust in grams to force by multiplying by the acceleration due to gravity.

\[
\text{Thrust} = m \times a \text{ (Newton)}
\]

where:  

- \( m \) = mass (kg)  
- \( a \) = acceleration due to gravity = 9.81 m/s\(^2\)

Now multiplying the thrust (in Newtons) by the wind tunnel velocity (m/s) will result in the output power in watts.

Now you can determine your propulsion system efficiency for your electric motor and propeller.

\[
\text{Propulsion Efficiency: } \quad \eta_{\text{sys}} = \frac{P_{\text{out}}}{P_m} \times 100\%
\]

Multiply the efficiency by one-hundred to determine the efficiency in percentage.

In order to update the energy analysis spreadsheet from System Analysis 2 with your own propulsion system characteristics, closing the loop between supplied power and energy & power and energy output, the non-dimensional propeller advance ratio from the wind tunnel tests must be calculated. Recall that the advance ratio equation is as follows:
**Propeller Advance Ratio:**

\[
J = \frac{v}{(RPM/60) \cdot D}
\]

where:
- \(v\) = velocity (meters/sec.)
- \(RPM\) = propeller RPM
- \(D\) = propeller diameter (meters)

The velocity was simulated by the wind tunnel, and set a constant value of 2.8 meter/sec.

In order to determine the propeller efficiency as a function of the advance ratio, replace the sample propeller efficiency trendline coefficients provided in System Analysis 2, and make a plot of efficiency (y-axis) and advance ratio (x-axis) and add a 3rd order trendline. This new trendline will represent the propeller efficiency as a function of the advance ratio.
Lab 8: Design Analysis Tool

Objectives:
1. Become familiar with MATLAB based design analysis tool.
2. Be able to upload wind tunnel data into the design analysis tool.
3. Be able to upload Arduino data into the design analysis tool.
4. Be able to conduct performance analysis of an AEV operation.
5. Be able to export plots for reports.

Lab Activity: Follow this section of the AEV Lab Manual and the Lab Grading Guidelines to upload wind tunnel and classroom track performance data into the design analysis tool, conduct performance analysis, and export plots for writing reports.

The goal of this lab is to learn how to use a design analysis tool that provides an efficient and productive method to evaluate AEV performance. This program was designed to communicate with the Arduino motor controller, which will download data collected from the Arduino motor controller after an AEV test run. The following describes in detail how to use the AEV Analysis program.

**Installing and Running the Design Analysis Tool**

The following steps are for installation of the AEV Analysis tool for both home computers and computers in the class room.

1.) In the Lab 8 Design Analysis Tool folder, double click on the install app called “AEV Analysis Tool.mlappinstall”. MATLAB will open if it is not already running and display the window shown in Figure 22.

![MATLAB Installer Image]

Figure 24: MATLAB Installer
2.) Click on the “Install” button and MATLAB will install the AEV Analysis tool in the Apps bar.
3.) To view the App, select the “App” tab, then select the down arrow on the far right of the menu and you will see the “AEV Analysis” app installed under “My Apps”. See Figure 23.

![MATLAB Apps](image)

**Figure 25: MATLAB Apps**

Simply click on the “AEV Analysis” App and the software window will open. See Figure 24.

![AEV Analysis Software](image)

**Figure 26: AEV Analysis Software**
There are two tabs for this program. The current tab that is displayed when the program is run is the Propulsion Efficiencies tab. The next tab is the Performance Analysis tab. Selecting these buttons will bring up two different screens on the figure window.

The AEV Analysis tool has a File and Tools menu optioning the top left hand corner of the figure window. Below is a brief description of these options

1.) File
   a. **Open File (.xls)**: Opens, loads, and analyzes an Excel spreadsheet containing System Analysis 3 data recordings from wind tunnel testing. (Details discussed in “Loading Propulsion Efficiency Data”).
   b. **Open File (.mat)**: Opens, loads, and analyzes a MATLAB data file containing EEPROM data recorded by the Arduino motor controller. (Details discussed in “Loading Performance Analysis Data”).
   c. **Save As (.xls)**: Saves voltage, current, and thrust data entered into the table by the user to and Excel spreadsheet for later use by the AEV Analysis tool. (Details discussed in “Loading Propulsion Efficiency Data”).

2.) Tools
   a. **Download Arduino Data**: Downloads EEPROM data from the Arduino motor controller following an AEV track run and saves the data as a (.mat) file. (Details discussed in “Downloading Arduino Data”).
   b. **Export Figure**: Allows the user to export plots displayed in the app for documentation and presentation.
   c. **Clear File**: Clears one of the four files in the performance analysis tab so another data set can be observed (Note: “Clear File” does not permanently delete the data set).
   d. **View Energy Analysis**: Opens up the data set in Excel and so the user can view the raw and calculated data (similar to the Excel spreadsheets created in System Analysis 1 and 2).
   e. **View Arduino Program**: Allows the user to see an appended Arduino code (if copied and pasted). The user can then use this as a reference if needed to revert back to what Arduino code was used to create the data set.
   f. **Append Arduino Program**: Allows the user to copy and paste an Arduino code to a data set for they can look back at the code at a later time.

**Downloading Arduino Data**

The AEV Analysis tool communicates with Arduino using a serial port (COM #). In the Arduino IDE, the COM port needs to be specified for loading a program onto the Arduino motor controller. The AEV Analysis tool automatically detects which comport the AEV is connected to so there is
no need to take this step anymore. The following steps will describe how to download Arduino data.

1.) Connect the AEV to a PC using the provided USB cord.
2.) On the Arduino motor controller you will see a yellow light flashing, wait until the yellow light becomes solid.
3.) Under Tools, select “Download Arduino Data”. You should see the yellow and green light on the Arduino board turn solid. The progress bar at the bottom left hand side of the figure window should display the progress of the download.
4.) When the EEPROM data has been downloaded successfully, a prompt will display asking the user if they would like to store the Arduino code used for the run.
5.) The next prompt that will display is for saving the EEPROM data into a (.mat) file (make sure to save in a location you know where to find the data set and save the data set with a unique save name so you can go back if needed. Also, do not have a save name with numbers, else MATLAB will return with an error.)
6.) The download is now complete. The (.mat) file can now be used for viewing performance analyses following a propulsion efficiency analysis.

**Propulsion Efficiency**

In AEV Analysis, loading propulsion efficiency data is the first step that must be taken before viewing any performance analysis of your AEV. As we saw in the System Analysis 3 lab, knowing the diameter of the propeller determines what RPM equation we use in order to compute the propeller advance ratio and ultimately the propulsion efficiency of the AEV.

The following explains how to properly record and load data. For first time users, data will be loaded using sample data provided in the Lab 8 Design Analysis Tool folder.

1.) In the Lab 8 Design Analysis Tool, one file that is provided is the Wind Tunnel Recording.xlsx. Open this file and you will see that it is filled with sample data. The only data required in this file is the voltage, current, and thrust.
2.) Loading this file only requires a couple of steps:
   a. Under the File menu, select “Open File (.xls)”.
   b. Select the Wind Tunnel Recording Excel file.
   c. A list box will appear asking for you to select the proper Excel spreadsheet. In this case, it is on “Wind Tunnel EP-3030”. Select Ok.
   d. Another list box will appear asking which propeller was used for this wind tunnel test. Select “EP-3030” and then select Ok.
   e. The next prompt will ask for the user to input the wind tunnel velocity tested.
f. The final prompt is for the user to highlight the data set. Highlight the raw wind tunnel data set (consists of Current, Thrust Scale Reading, RPM and Arduino Power Setting). Note: You only want to highlight the numerical data, not the text!

g. You should see all tables populated and the push buttons above are now enabled. We can now view our propulsion efficiency data graphically.

Three plot options are available under the “Select a Plot Type...” pop-up menu. Thrust vs. Voltage, Power vs. Voltage, and Power Efficiency are the three plots provided. Take the time now to select all three and view the results of this sample data in the graphing window.

For documentation and presentation purposes, the tool allows you to export the figures you see in the program. The following steps provide an example of how to export a figure.

1.) Under the “Select a Plot Type...” pop-up menu, select the “Power Efficiency” plot option.
2.) Next to the File menu you should also see a Tools menu. Under Tools, select “Export Figure”. You should see something similar to Figure 25.

**Performance Analysis**

Before going through the following steps in this section, Propulsion Efficiency data must be loaded. See “Loading Propulsion Efficiency Data” above.

Knowing how your vehicle is performing is crucial to meeting the operational objectives for your project. In System Analysis 1 and 2, you learned how to calculate performance characteristics of the AEV through experimental research. The project is now entering into the cycle consisting of

1.) Analyzing data
2.) Making design decisions from the analyzed data
3.) Researching design decisions
4.) Testing and comparing to previous designs

This cycle will then continue until you’ve reached your final design. The AEV Analysis tool is designed to provide an in-depth analysis of the AEV performance to aid in design decisions.

This step would typically be followed by the “Downloading Arduino Data” section of this document. For a first time user, a MATLAB file named “Test_1.mat” should be provided with this tool. If the intent is to analyze data from an AEV run, first refer to the “Downloading Arduino Data” and then follow the following steps:

1.) Under File, select “Open File (.mat)...” and select a (.mat) file.
2.) The tool will load the EEPROM data stored from an AEV run and perform an analysis similar to System Analysis 2. You should see the file name and total energy appear in one of the file boxes. This is illustrated in Figure 29.

![Figure 27: Performance Analysis](image)

Two plot options are available under the “Select a Plot Type...” pop-up menu. Energy Analysis vs. Time (s) and Energy Analysis vs. Distance (m). In order to view data graphically:

1.) Select a plot type, (for first time users, select Energy Analysis vs. Time (s))
2.) The “Select a file...” pop-up menu next to the plot type menu will be enabled, as well as the box for entering the AEV mass, in kilograms. First, enter in the vehicles mass. (For first time users, enter 0.250)
3.) Under the “Select a file...” pop-up menu there will be a list of up to 4 files to choose from, depending on how many files are loaded. Select a file and you should see a full energy analysis. See Figure 27.

![Figure 28: Visual of Performance Analysis using Sample Data](image)

For documentation and presentation purposes, the tool allows you to export the figures you see in the program. Under Tools, select “Export Figure”. The figure that is exported can be saved as different file types such as JPEG or PNG. When saving the file, just change the “Save as type:” See Figure 31.
Figure 29: Figure save as file type options
APPENDIX: AEV Operation Procedure

The following outlines the testing procedure for AEV operation on the track to minimize the potential of damaging the AEV, AEV components, and YOU.

**AEV Desktop-track Setup and Controller Software Confirmation**

- Each AEV must demonstrate proper balance on the desktop track with the AEV center-of-gravity in between and directly below the two monorail track wheels.
- The vertical support arm must be equal to or less than 6 inches from the monorail track.
- IF major changes are made to the AEV Controller Software in between runs, confirmation of the desired programmed power control must be checked statically on the desktop track prior to testing on the classroom track.
- The AEV power must be off when transferring AEV from the desktop to the classroom track.

**AEV Classroom-track Testing Procedure**

1. AEV power OFF.
2. A team member must be at each station.
3. Teammate #1 takes the AEV to the docking station, Station 1.
4. All other teammates must be at their designated stations and be prepared to stop the AEV if it does not pass the required operation procedure in designated location.*
5. Inform classmates (if in operational area) that AEV operation is going to start.
6. Teammate #1 sets AEV on the monorail track and turns AEV power on.
7. Receive confirmation from all other Teammates that they are ready.
8. Begin AEV operation by pressing the AEV Automatic Control System start button.

*The Teammate must place their hand above the track at the designated stopping area.

**AEV Operation Pass/Fail Conditions**

- If the AEV stops within the designated stopping area, the vehicle passes the required operation stopping procedure.
- If the AEV stops prior to the designated stopping area the AEV does not pass and MUST be stopped by grabbing the AEV’s monorail track wheels.
- If the AEV continues through the designated stopping area (without stopping at all), the Arvoes not pass and MUST be stopped by grabbing the AEV’s monorail track wheels at the end of the designated stop.