Juno Tracker 5000

Developer Documentation

Valencia College | Summer 2022

Derek DiLeo

# Document Updates

|  |  |  |  |
| --- | --- | --- | --- |
| Revision ID | Date | Author | Notes |
| 1.0 | 05/20/2022 | Derek DiLeo | Initial draft |
| 1.2 | 06/08/2022 | Derek DiLeo | Complete 'Handling Data' Section using screen shots with test data |
| 1.3 | 07/03/2022 | Derek DiLeo | Review and Submission |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

****

Table of Contents

[**Document Updates** 1](#_Toc105569182)

[**Introduction 3**](#_Toc105569183)

[Installation Guide: 3](#_Toc105569184)

[Overview 4](#_Toc105569185)

[Purpose 4](#_Toc105569186)

[Collecting Data 4](#_Toc105569187)

[Handling Data 4](#_Toc105569188)

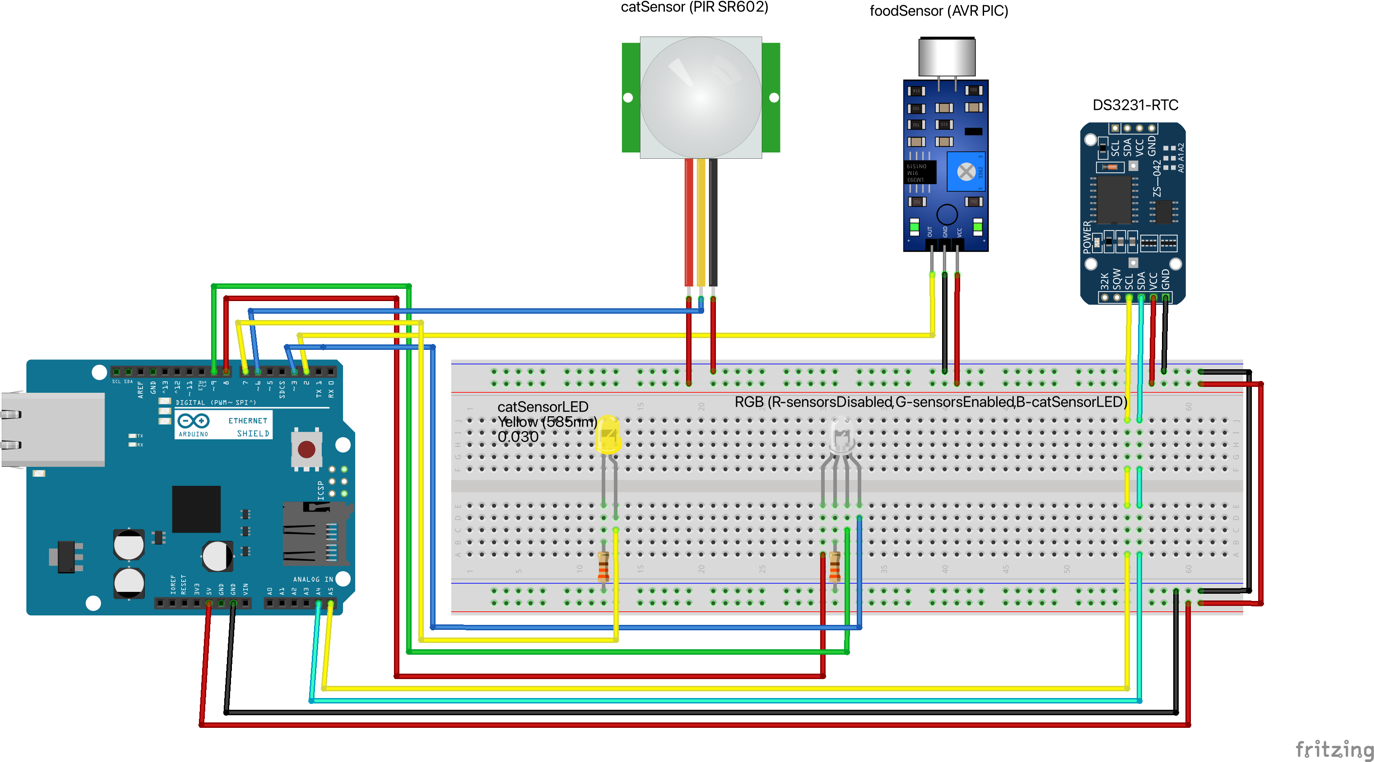
[Conclusion 8](#_Toc105569185)

[Results 7](#_Toc105569189)

[Conclusion 8](#_Toc105569190)

# Introduction

This software project is focused on the concept of 'small data' to answer the age-old questions: "How quickly does my cat get to his food bowl after his auto-feeder begins dispensing his food? How can this data be collected and presented?"



## Installation Guide:

1. Download the Arduino JunoTracker5000.ino sketch [here](https://github.com/derekdotdev/cat-tracker). **Notes:**
   1. See repo for alternate circuit using UNO and SD Card Module
   2. You must update date and time in Initialize\_RTC() function (line 215)
   3. You must update timeIsValid() function to match your pet's feeding time(s)
2. Assemble the circuit shown above (pins are outlined in comments at top of sketch).
3. [Prepare your Micro SD card](https://learn.adafruit.com/arduino-ethernet-sd-card?view=all#sd-card-preparation) for use with Arduino SD Library.
4. Carefully insert SD card into the slot.
5. Position sensors. foodSensor audio threshold calibration is required.   
    **Note:** Both sensors are ignored outside 'feeding windows' set in step 1c and catSensor is ignored until foodSensor is triggered.
6. Install [Arduino.app](https://www.arduino.cc/en/software) to your Applications folder.
7. Download the [DS3231 RTC Module library](http://www.rinkydinkelectronics.com/download.php?f=DS3231.zip) from Rinky-Dink electronics and add to project with Sketch > Include Library > Add .ZIP Library...
8. Connect your Arduino to an available USB port. UNO rev3 with Ethernet Shield 2 is preferred, but any SD module will work if default Chip Select pin is considered.
9. Upload the sketch to your Arduino ~10 seconds before time set in step 1b (to account for delay() functions in setup() function) and let it do its thing!

# Overview

## Purpose

The goal of this project is to aid in better understanding the speed at which my cat, Juno, can run to his feeding bowl after it begins automatically dispensing a meal by collecting the time differences between two sensors being activated. The first is tripped when the food begins dispensing and the second is tripped when Juno arrives at his bowl. The comma-separated value (CSV) data is written to an SD card with a timestamp where it can later be imported into a spreadsheet for analysis (see Handling Data).

## Collecting Data

This application is programmed to run without the need to be connected to a computer and depending on the number of meals per day, can take some time to collect sufficient data. To allow the user to validate functionality without a UI, LEDs are incorporated to notify the user of the status. During startup, the RGB LED will illuminate GREEN for a period of four seconds and turn off to indicate that the setup function is complete. The RGB LED is also used to indicate the status of the sensors which are only programmed to be active for a six-minute 'feeding window' around each feeding time. These windows are used to mitigate false sensor events from being recorded and to allow for slight time differences between the auto-feeder and RTC module. When they are active, the RGB will glow GREEN and when inactive, it will glow RED. The RGB will also glow BLUE when the catSensor is tripped. This, however, can only occur once the foodSensor has been triggered which is indicated by the YELLOW LED. To recap:

RGB glows GREEN at startup for 4 seconds to indicate program setup is complete

RGB glows RED while outside of 6-minute feeding window (3-min before / after)

RGB glows GREEN once inside 6-minute feeding window (until foodSensor trips)

When foodSensor is tripped:

* RGB RED and GREEN LEDs are disabled
* RGB YELLOW LED glows for 500ms and turns off
* Current time (in ms) is saved to foodTime variable
* At this point, all LEDs remain off until second sensor trips

When catSensor is tripped:

* RGB glows BLUE until motion is no longer detected
* foodTime (and LED delays) subtracted from current time to get responseTime
* Date, Time, and responseTime are written to SD card as CSV

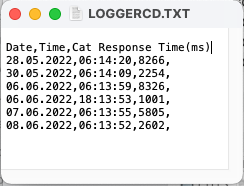
Finally:

* RGB is set to glow RED again
* A 30-minute (1800000 millisecond) delay is set to avoid repeat sensor event data during current feeding window
* The process repeats

## Handling Data

Once a reasonable amount of data has been collected, it is time for analysis. Feel free to follow the steps below or [watch a video tutorial here](https://www.youtube.com/watch?v=EDtZcM9CFKY). Unplug the Arduino, carefully eject the SD card and insert into a card reader attached to your computer. Locate the file LOGGERCD.TXT on the SD and open it in a preferred text editor.

Each time the program starts and Initialize\_SDcard() runs, the first line of the data file is written which, if multiple exist, can affect our import. To avoid any issues, remove all test data prior to the final write of the first row. Your data should resemble the following:

Text

Description automatically generated with medium confidence

Figure 1 - Before Test Data Removal

Figure 2 - After Test Data Removal

Open a new Excel spreadsheet, navigate to the Data tab, and select Get Data from Text and choose the updated LOGGERCD.TXT from the SD in the Get Data dialogue.

Graphical user interface, application, table, Excel

Description automatically generated

Figure 3 - Get Data From Text (Legacy)

Navigate the Text Import Wizard:

Step 1: Select Delimited and Click Next

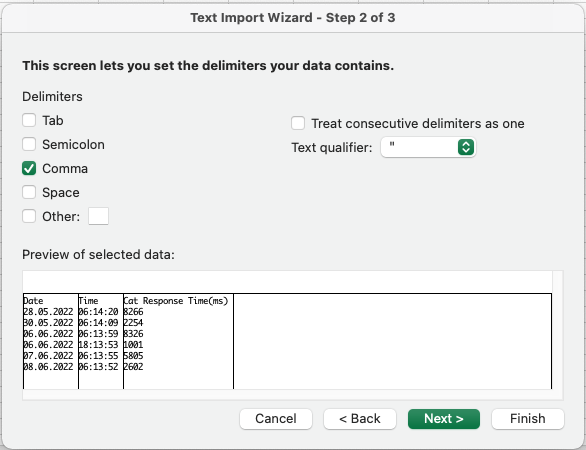
Step 2: Select Comma and Click Next

Step 3: Select 'Date', change to DMY to set correct Date format, and click Finish

Step 4: Import Data to Existing sheet at =$A$1

The following screenshots will serve as a guide to navigate the Text Import Wizard.

Graphical user interface, application

Description automatically generatedGraphical user interface, application

Description automatically generated

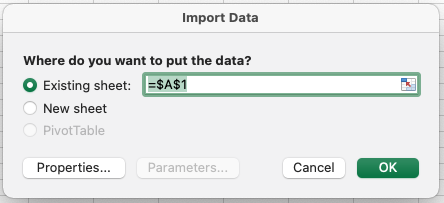


Figure 4 - Navigate the Text Import Wizard Steps 1-4

Calculate SUM and AVERAGE Values:

Step 1. Select the next empty cell below the list of times

Step 2. Navigate to Formulas tab and select Sum

Step 3: Press Enter

Step 4: Select the next empty cell below the list and repeat for Average

Graphical user interface, application, table, Excel

Description automatically generated Graphical user interface, application, table, Excel

Description automatically generated

Figure 5 - Calculate Sum and Average Response Time

Graph Results:

Select all rows and columns, navigate to Insert tab, and select a 2-D Column Bar Graph

Graphical user interface, application, table, Excel

Description automatically generated

Figure 6 - Insert a 2-D Bar Graph to Display the Results

# Conclusion

## Hardware

Photos of the system once fully operational.

A picture containing electronics

Description automatically generated

Figure 7 - Circuit Overview

A picture containing indoor

Description automatically generated

Figure 8 - AVR Sound Sensor Placed Directly on Body of Auto-Feeder

A picture containing plastic

Description automatically generated

Figure 9 - AVR Sound Sensor Placed Directly on Body of Auto-Feeder (close)

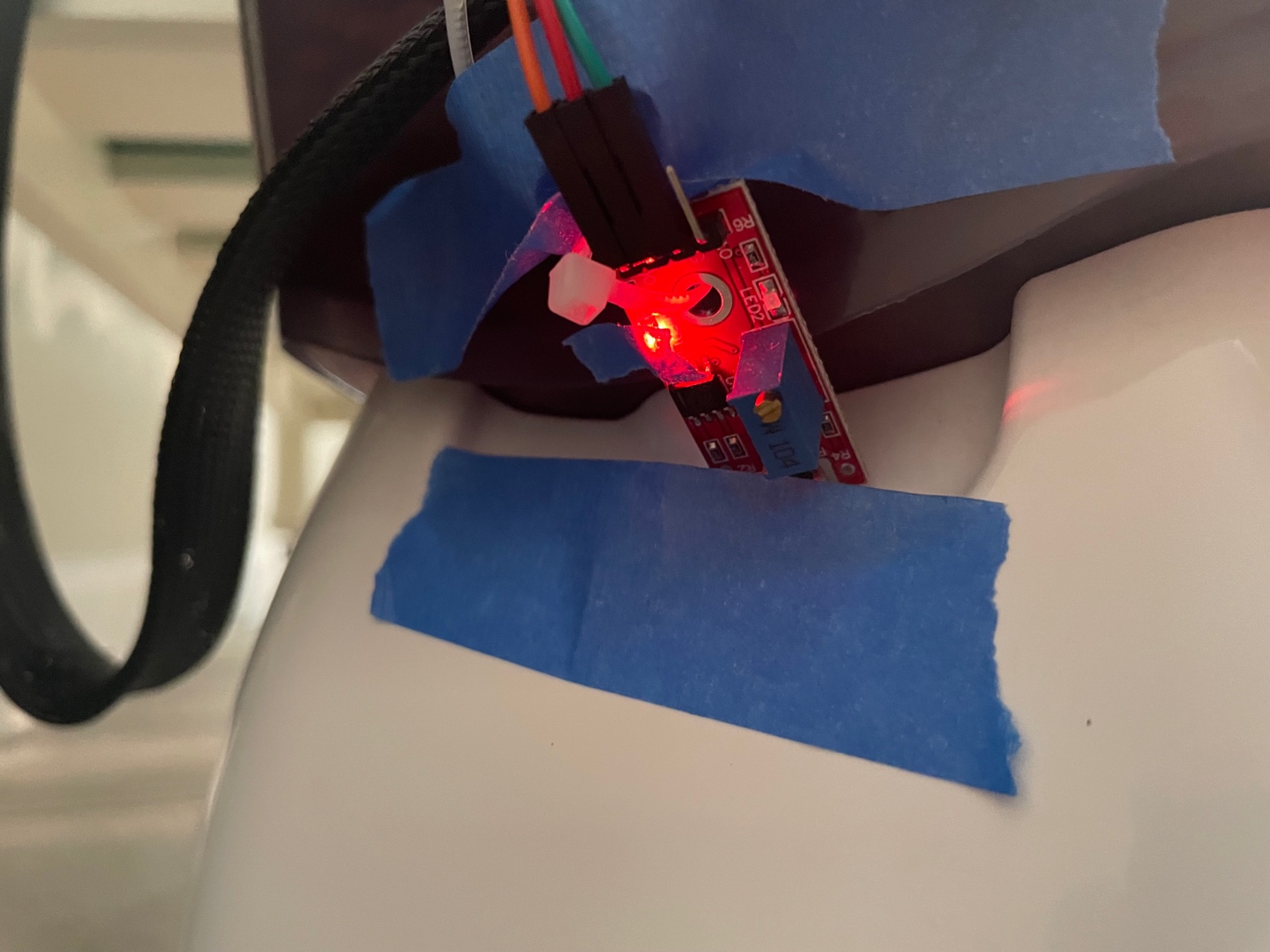


Figure 10 - AVR Sound Sensor Placed Directly on Body of Auto-Feeder (close)

## Results

After one month, the following data about Juno's response time was captured.

Minimum: 1001ms (GREEN)

Maximum: 8326ms (RED)

Average: 2958ms (YELLOW)

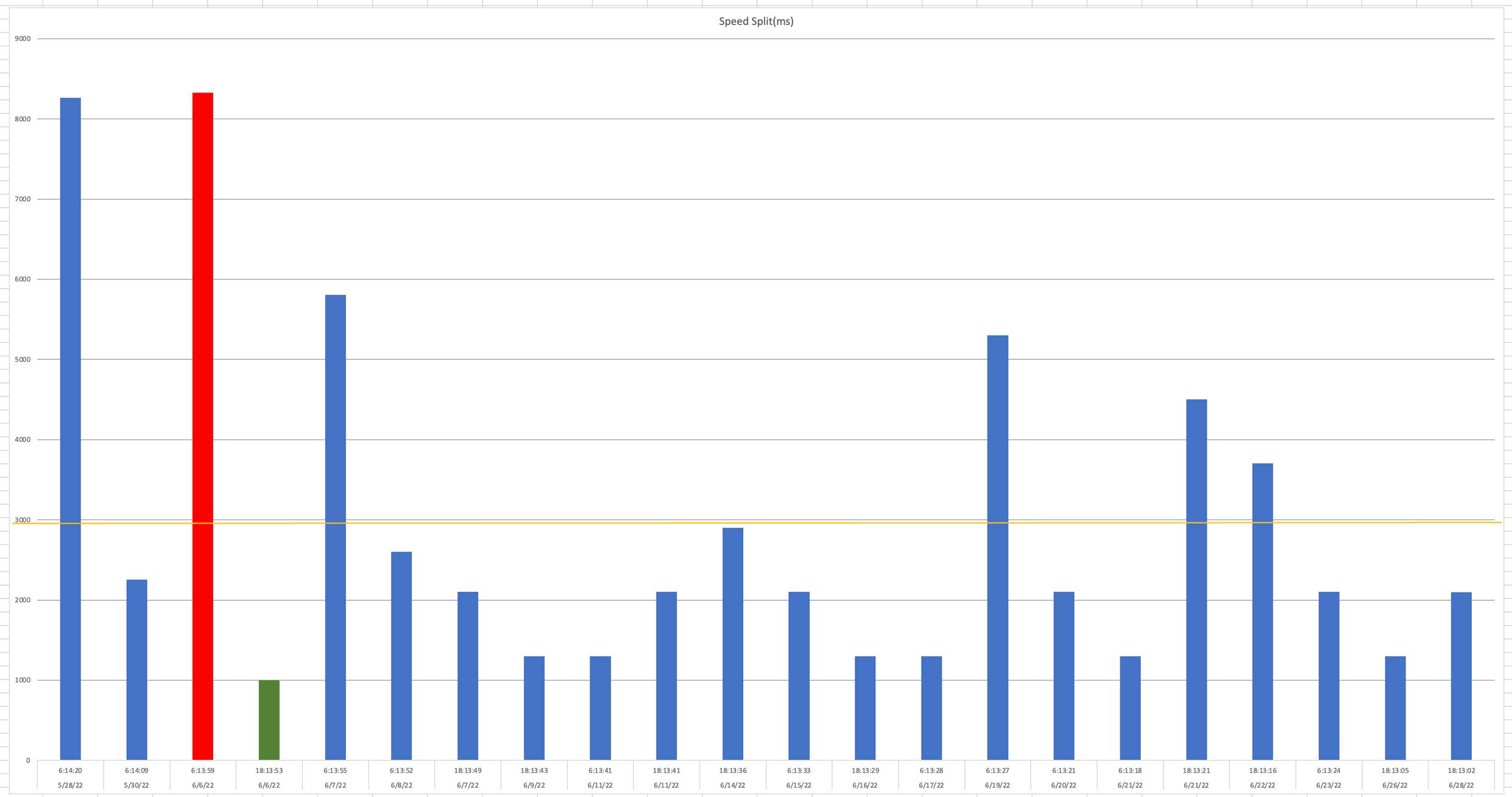


Figure 11 - Results Graph

## Conclusion

The Juno Tracker 5000 project was more difficult to implement than initially expected, but it turned out to be a great learning experience for many reasons. First, where the concepts of requirements gathering, and research intersect is where one can attain the most insight about the overall cost and schedule of the project. If I had conducted more thorough research up front, I would have known that PIR sensors are not ideal for capturing dry cat food in motion and, thus, avoided the additional cost and timeline delays that were introduced by purchasing the AVR sound sensor. Second, because the data were written to an onboard SD card, the only feedback from the device while it was deployed was from a couple of onboard LED indicators. Without access to the data in real time, I was essentially blind to the performance and overall status of the device. If I could start this project over, I would set up a local network server and database and connect the Arduino to it so the data could be accessed whenever without worry of taking the device offline. Lastly, I discovered that the more interest I took in Juno's feeder, the closer he would be to it at mealtime. Some data had to be deleted because his response time was essentially instant since he was sitting right next to it when it began dispensing food.

Overall, this was a fun project and an awesome way to collect 'small data'. Some people want a 10-second muscle car, but I've got a 3-second cat!! [Please follow this link to see the full project presentation](https://vimeo.com/726744707). Thanks for reading!!