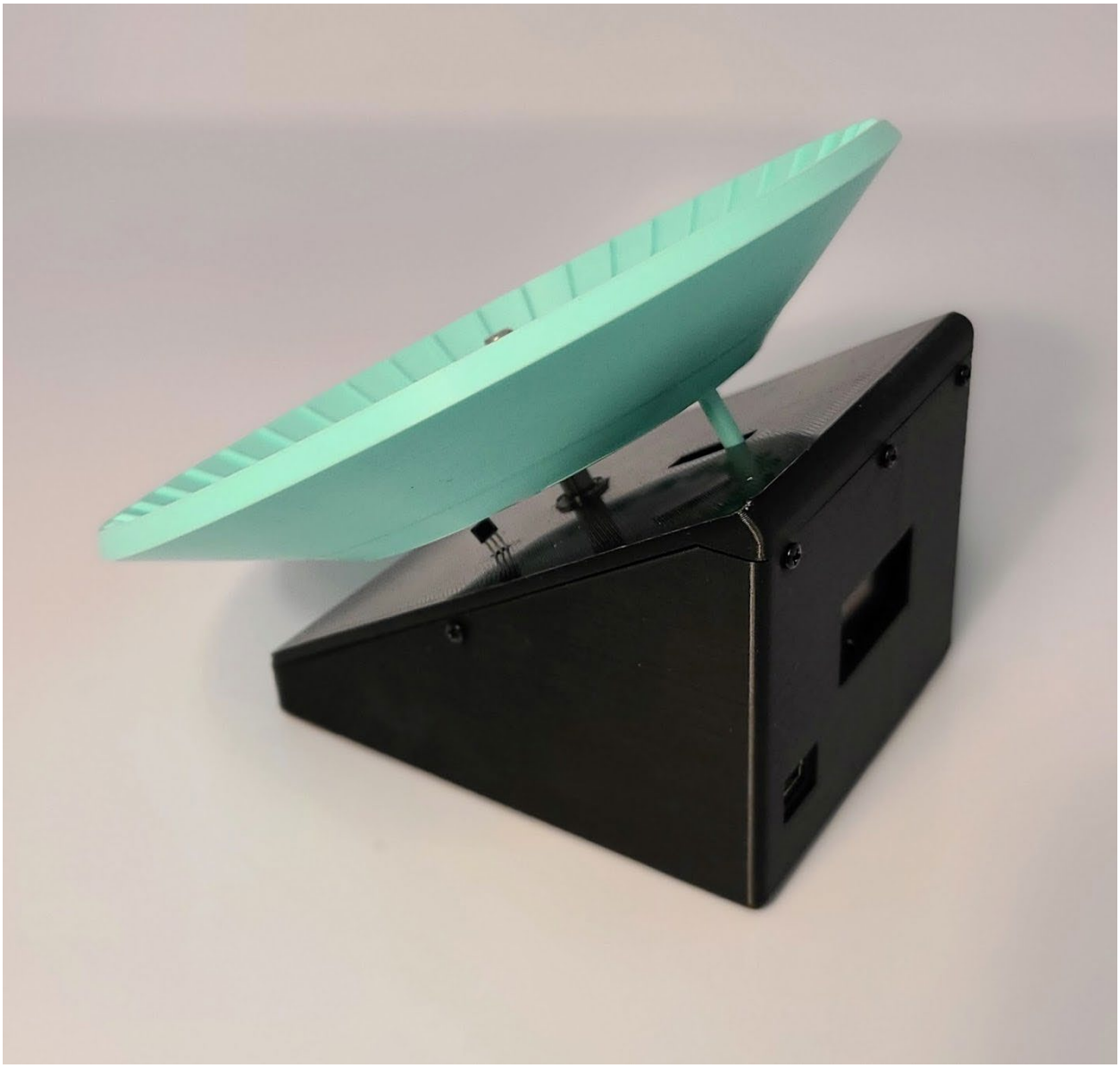


LOST-Wheel Manual

Version 1 – October 2021

JJ Bivona

Welcome

The LOST-Wheel was born out of the necessity to create an inexpensive, data collecting mouse wheel with locking features. Whereas current products can track distances, or be manually locked, they often require cage modifications, human intervention, purchasing additional hardware/software, and are cost prohibitive. Using easily accessible parts, the LOST-Wheel can be assembled for around $15 USD, requires no modifications to a standard shoebox cage, and can be locked without human intervention. We hope that this wheel can assist you in your research and allow some freedom in your budget.

This manual will walk you through the assembly, programming, and operation of the LOST-Wheel. It is designed to make the process as easy as possible. A few notes before you begin:

3D Printing: All parts for the wheel can be printed in standard polylactic acid (PLA) on entry level fusion deposition modeling printers. These types of printers have substantially decreased in price and can be purchased for less than $300 USD. Our lab has an Ender 3 V2 (MSRP $280) that has been a workhorse for our needs. Alternatively, many universities have a 3D printing core or a fabrication lab that can produce the 4 parts required.

Programming: The Arduino sketches and R application are designed so that there is very little programming involved – users only need change simple, clearly labeled parameters to customize the wheel to their experiments. Before starting, be sure to download the Arduino IDE. If you would like to use the LOST-Wheel Logger application, you will need the R programming language (4.1 or newer), and RStudio.

Soldering: This part can be tricky, but with some patience and practice any bench scientist can handle this skill. It's like pipetting, but hotter. **Soldering is not required**, however it is recommended for more durable connections to the microcontroller. To use pin-style connections to the Arduino Nano, see section 3.5.3 and be sure to download and 3D print the "no solder version" of the LOST-Wheel Bottom.

Assembly: There are only a handful of screws that hold the LOST-Wheel together, and many pieces are press-fit into place.

All required files may be downloaded from www.github.com/jjbivona/LOSTwheel

Table of Contents

1. Materials

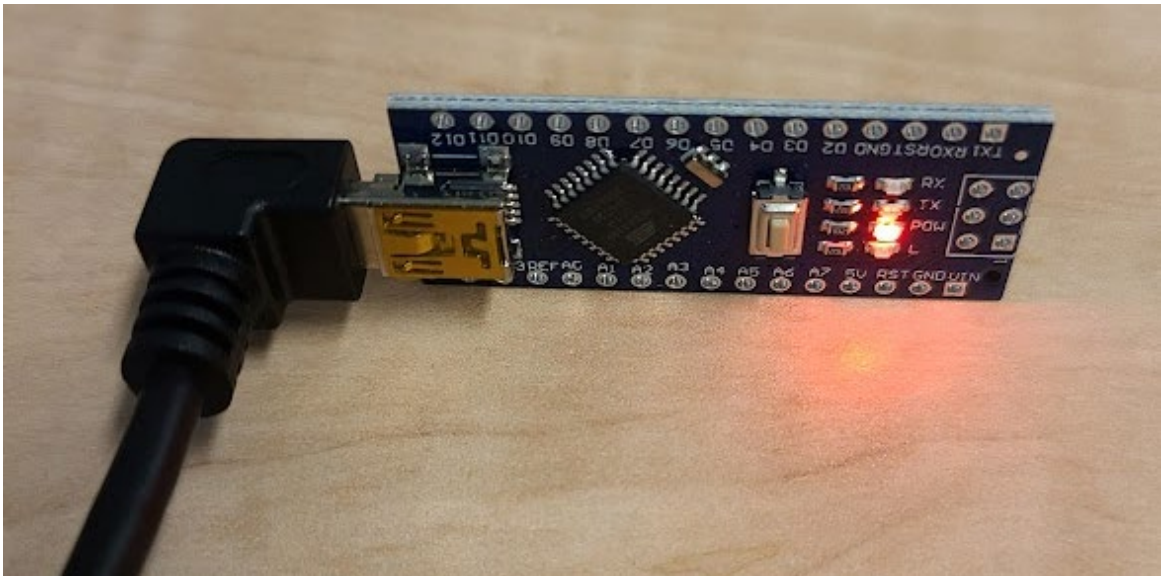
1.1 LOST-Wheel Component List

Component	Quantity	Link:
6 mm ID, 10 mm OD, x 3 mm bearing	3	6 mm Bearing
6 mm D axel cut to 65 mm	1	250 mm Axle
M3x5 self-tapping screw	1	M2, M2.3, M3 Screw Set
M2.3x8 self-tapping screw	8	
M2x6 self-tapping screw	1	
M1.7x6 self-tapping screw	8	M1 Screw Set
10 mm x 5 mm x 3 mm neodymium magnet (or similar)	2	10x5x3 mm Magnet
22-gauge, 2.54 mm breadboard jumper wires, 3 male, 7 female	10	Elegoo Jumper Wires
Arduino Nano (or similar)	1	Elegoo Nano Clone or Arduino Nano
9g micro servo	1	Plastic Gear or Metal Gear
KY-003 hall effect sensor	1	Hall Effect Sensor
0.96 inch 128x64 OLED Screen I2C connection SSD1306 Driver	1	OLED Screen

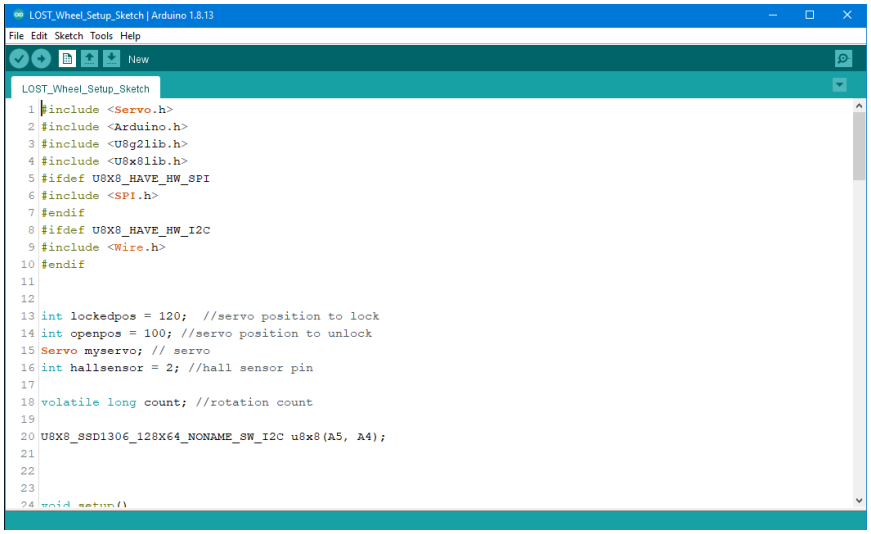
2. Assembly Instructions

2.1 Uploading the setup sketch

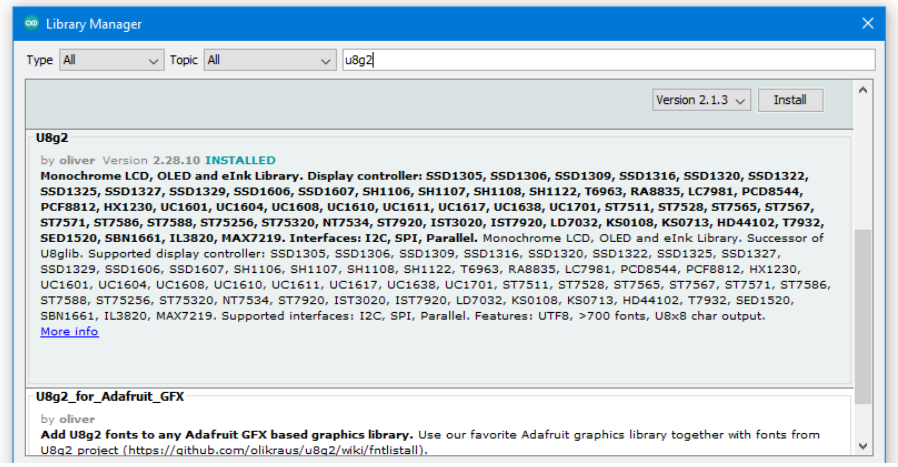
2.1.1 Connect microcontroller to computer through USB



2.1.2 Open LOST_Wheel_Setup_Sketch

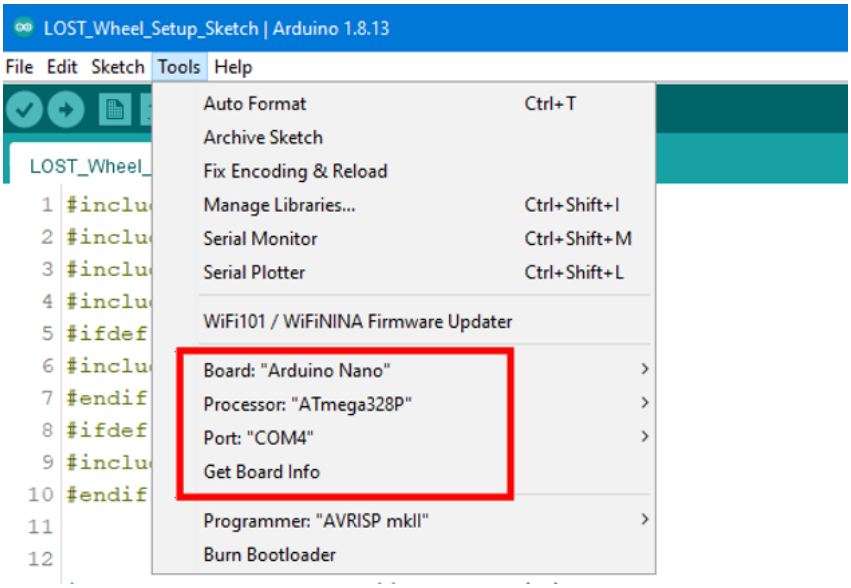


Note: if this is your first time making the LOST-Wheel be sure to install the **U8g2 library**. To install libraries, go to Tools → Manage Libraries → type in U8g2 into the search bar → install the latest version of U8g2

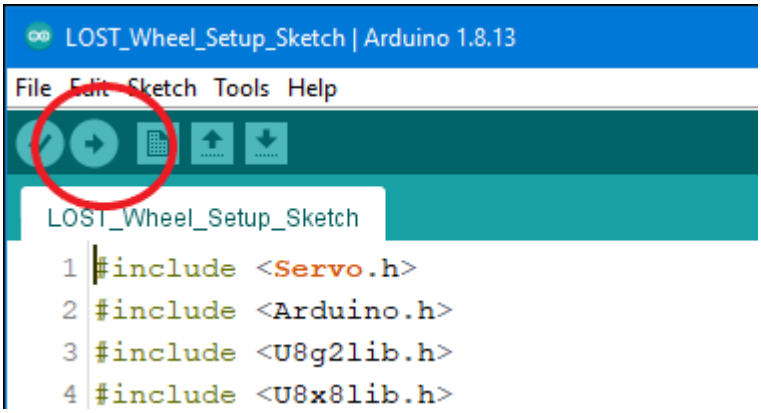


2.1.3 Confirm connection between Nano and computer

a. The “Port:” may need to be changed. This varies based on which USB connection is being used.



2.1.4 Upload sketch to Nano by pressing “Upload”

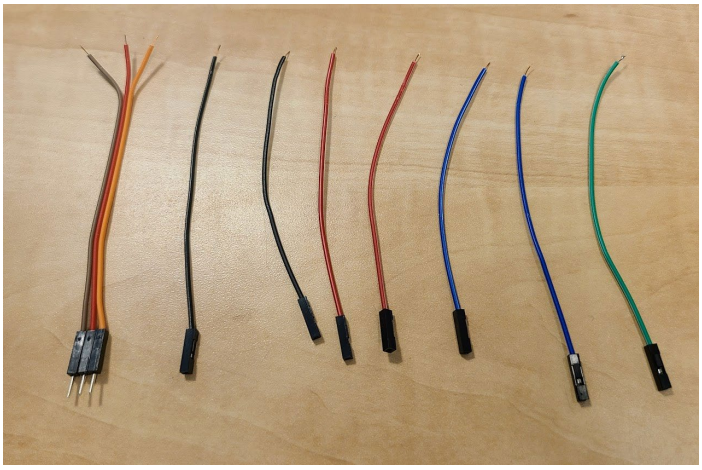


2.2 Wiring

Notes:

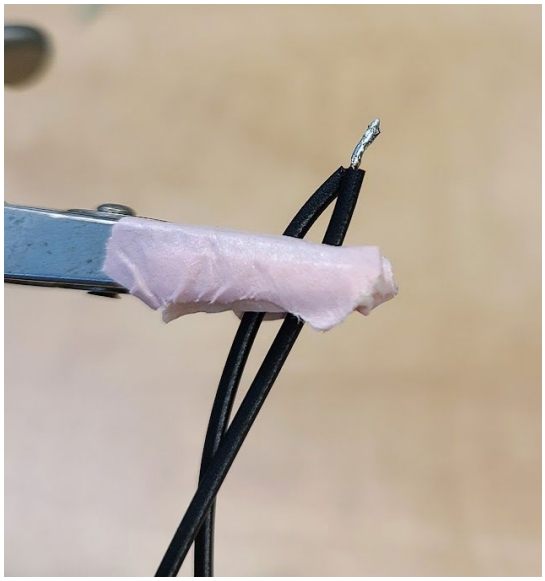
- Soldering is not necessary to create the LOST-Wheel, but is recommended. Alternatively, pin-style connections can be used with the Arduino Nano. **Should this route be taken make sure to download and 3D print the correct model labeled "no solder version."** The location of the USB connector has been changed to account for the pins. See section 3.5.3 for additional details.
- Please refer to the wiring diagram on **page 7** for this section of the manual.
- Unplug Nano before soldering.
- Take necessary safety precautions while soldering.

2.2.1 Cut and strip 3 male jumper wires and 7 female jumper wires

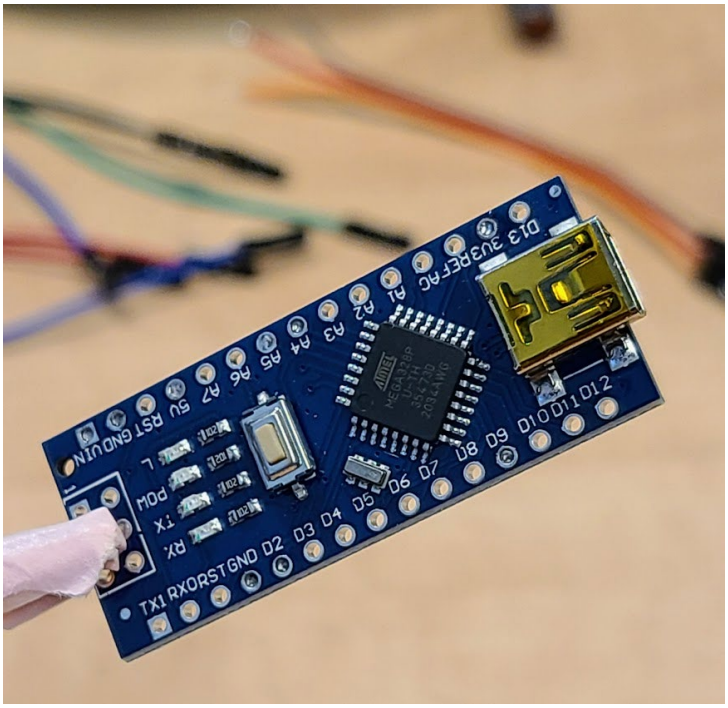


2.2.2 Apply solder to exposed ends of wires

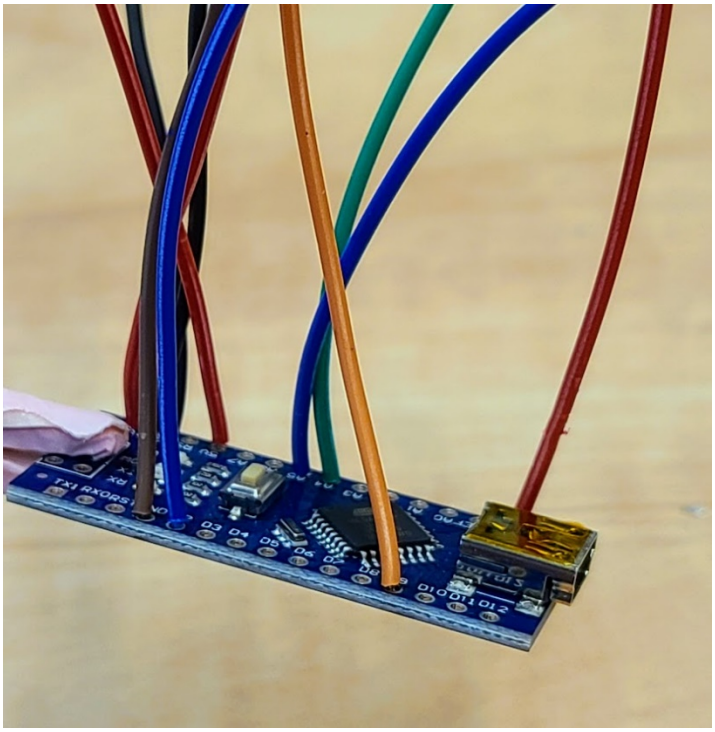
a. For the ground wires, twist two wires together and apply solder to the twist

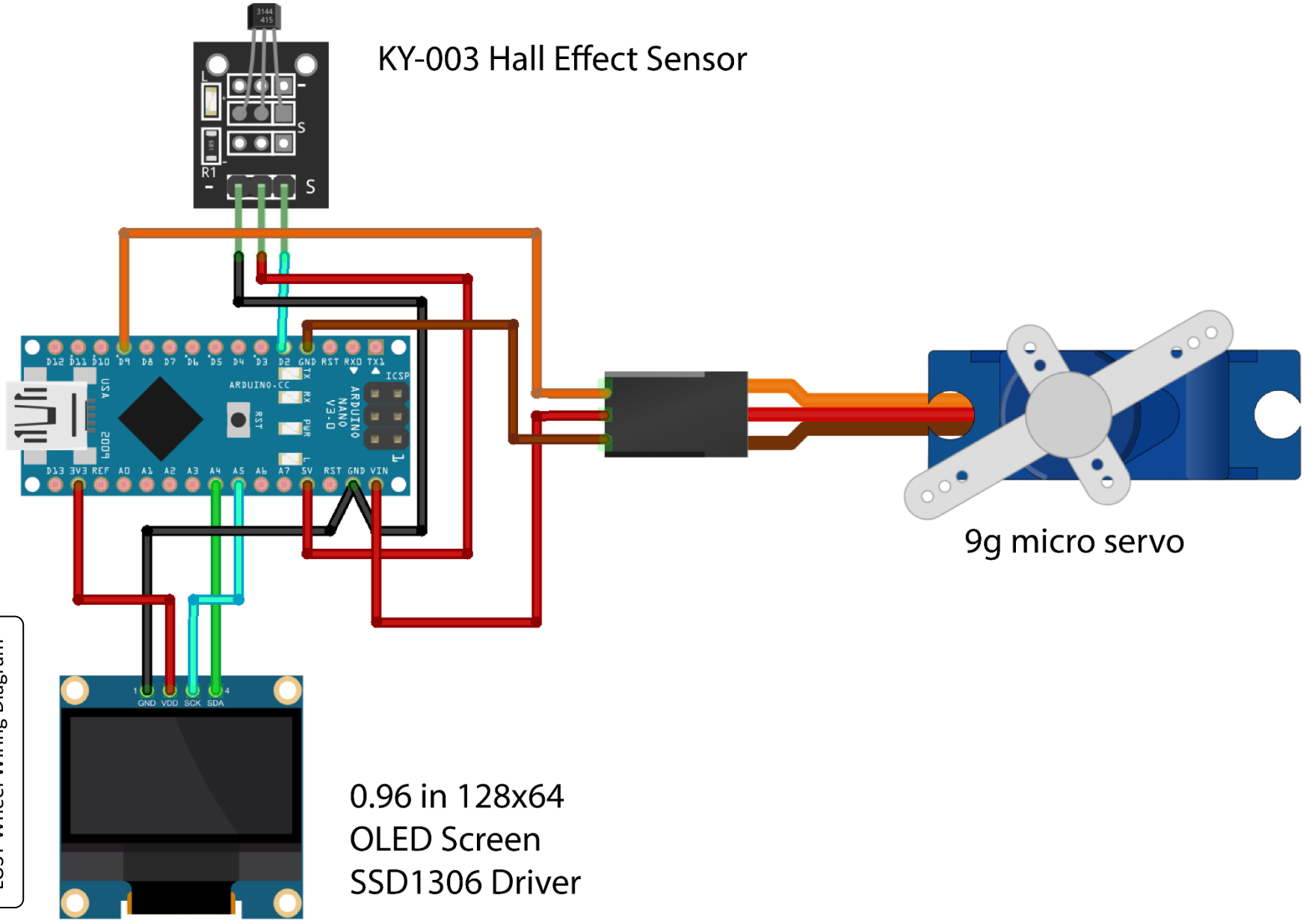


2.2.3 Apply solder to appropriate pins of the Nano



2.2.4 Fuse wires to the pins of the Nano



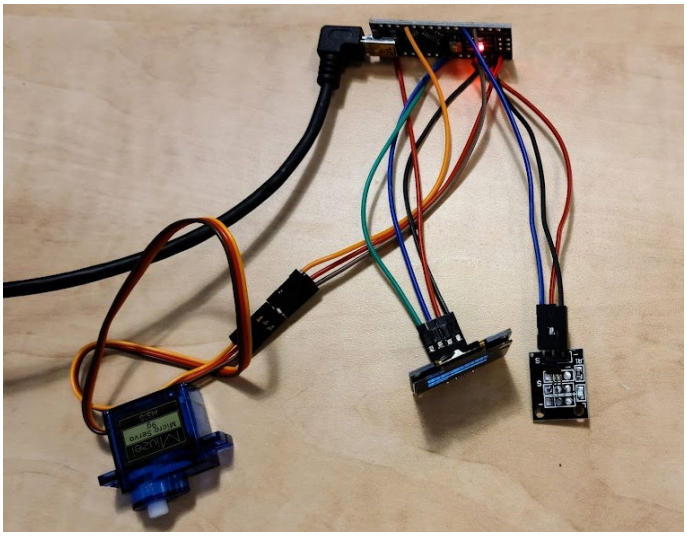


LOST Wheel Wiring Diagram

2.3 Testing components

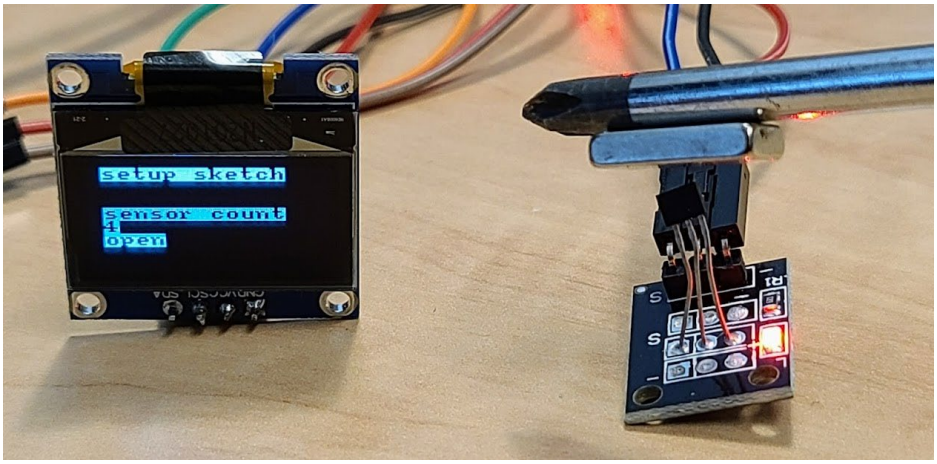
2.3.1 Refer to the wiring diagram to attach the servo, screen, and Hall effect sensor to the correct wires

a. Plug Nano into USB power source



2.3.2 Confirm screen turns on and wave a magnet in front of Hall effect sensor to test

a. A red LED should flash, and the counter should increase by 1



2.3.3 Confirm servo movement

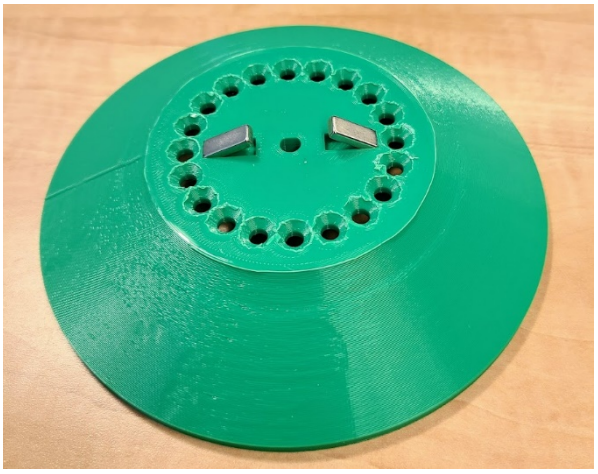
a. The servo will move 20° every 10 seconds

If any component is defective, replace and retest before proceeding.

2.4. Device assembly

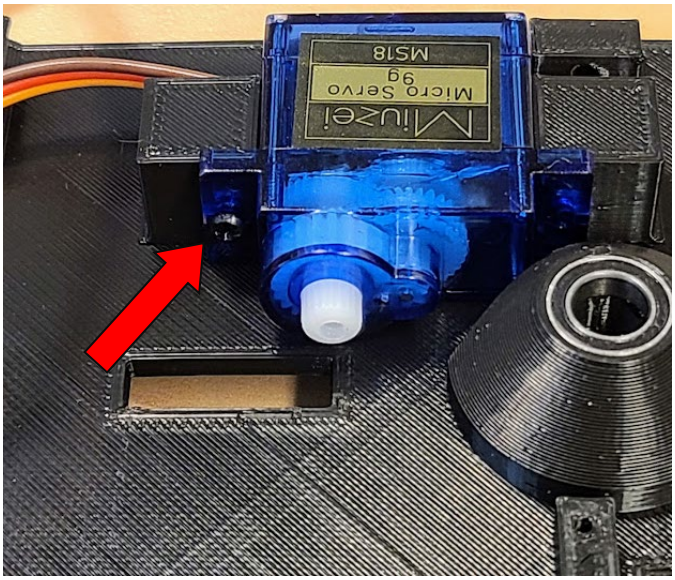
2.4.1 Press magnets into wheel

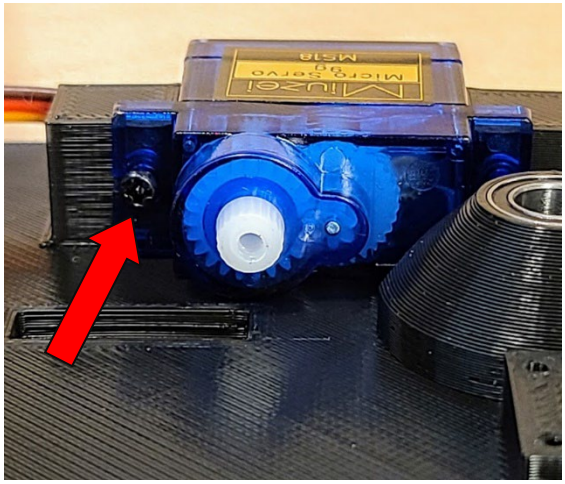
a. **Tip: wedge one side of the magnet into the slot, flip the wheel over and press down on a hard surface. The magnet should snap into place. Repeat for second magnet.**



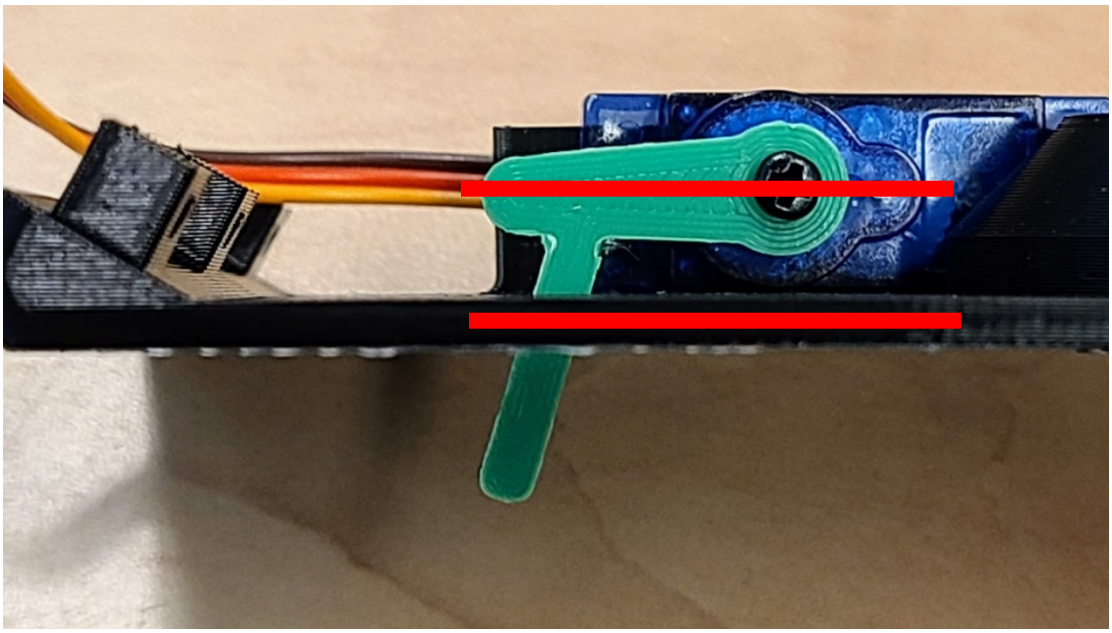
2.4.2 Press bearings into both sides of top and into the base

2.4.3 Attach servo using 1 M1.7x6 screw



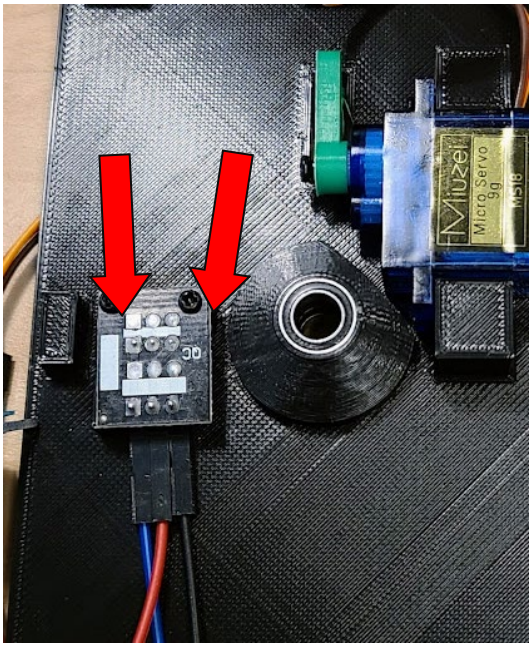


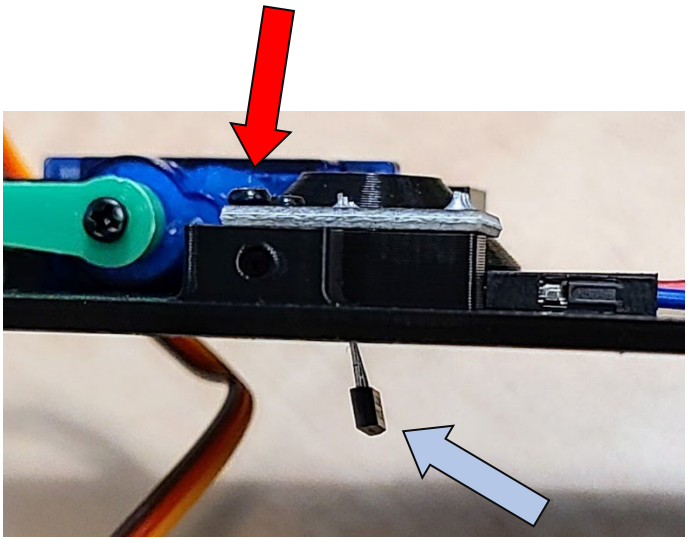
2.4.4 Power on the LOST-Wheel and while the servo is in the OPEN position as indicated by the OLED screen, press the locking pin onto the servo so that it is parallel to the top plate. Secure locking pin to servo using 1 M2x6 screw.



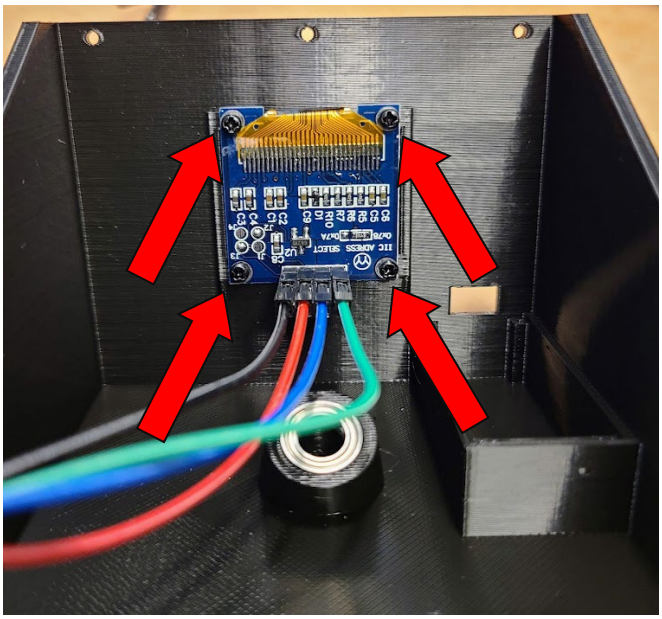
2.4.5 Attach hall effect sensor using 2 M1.7x6 screws

a. **Thread the sensor head (blue arrow) through the LOST-Wheel top before fastening screws.**

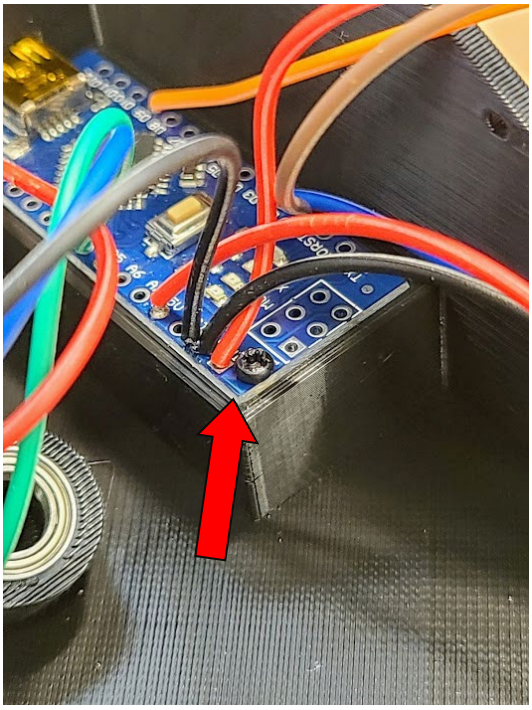




2.4.6 Attach screen using 4 M1.7x6 screws

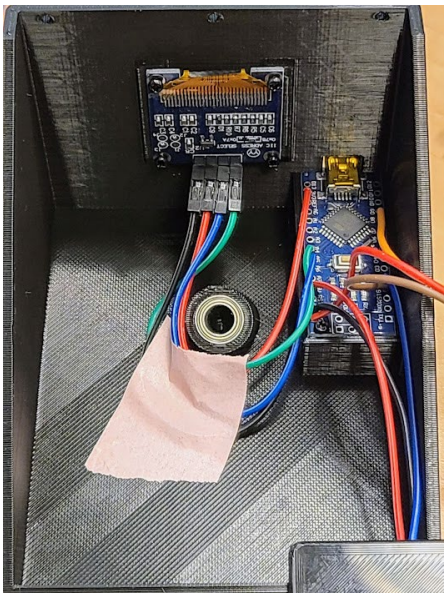


2.4.7 Attach Nano using 1 M1.7x6 screw

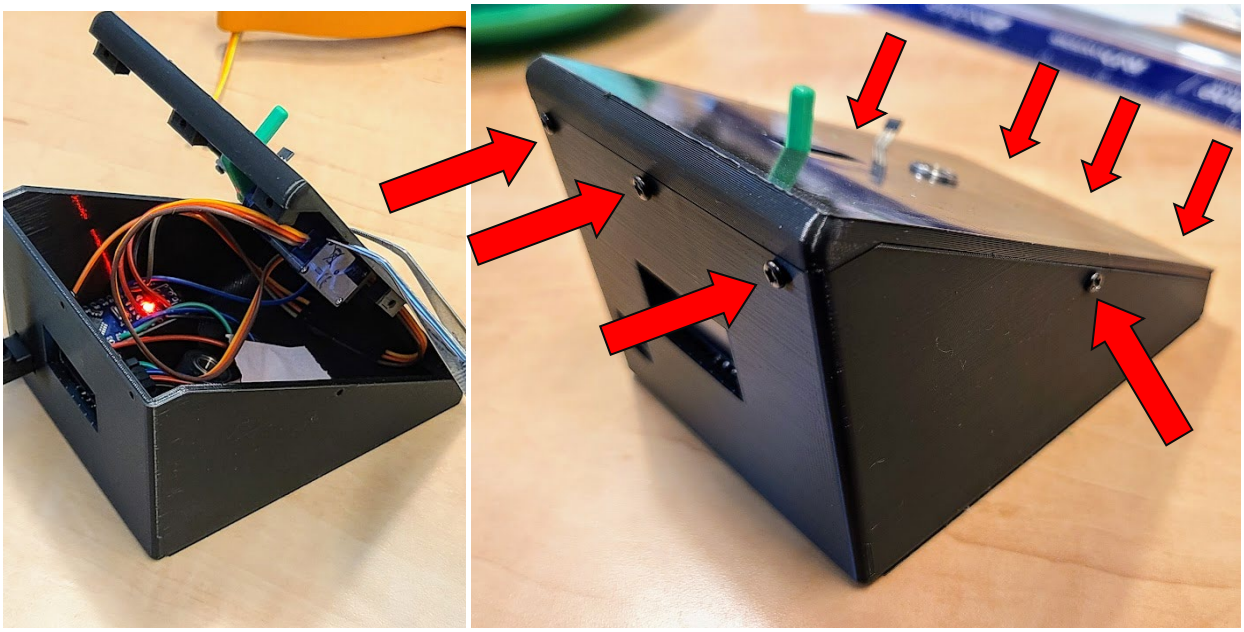


2.4.8 Position screen wires away from axle mounting point.

a. **Optional: Secure wires with tape**

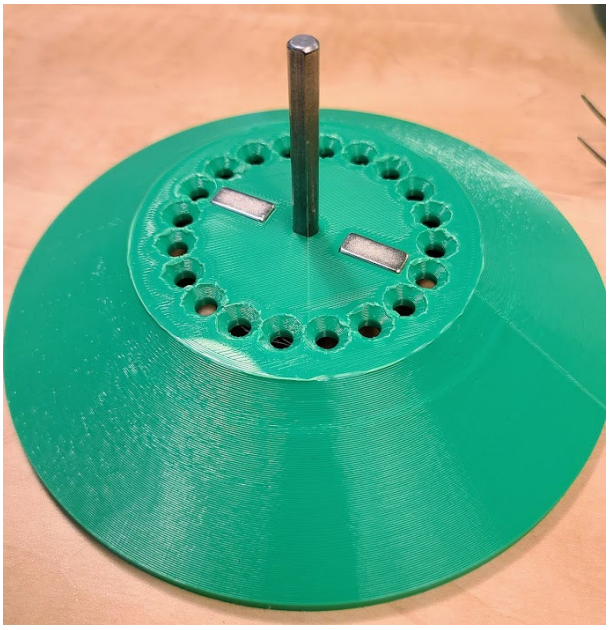


2.4.9 Secure the top and base pieces using 8 M2.3x8 screws



2.4.10 Attach wheel to axle

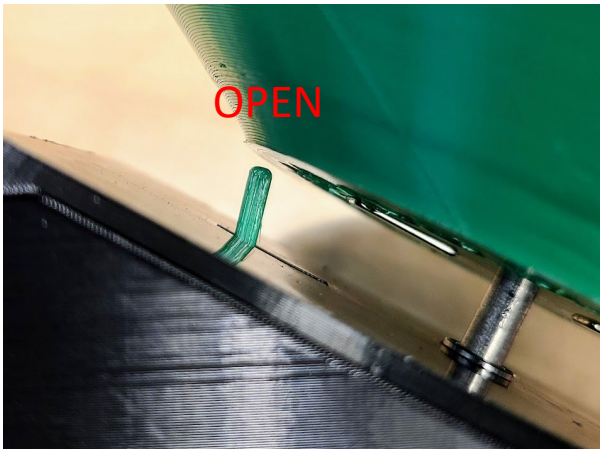
a. **Mark drive shaft 49mm from one end**
b. **Slide wheel onto axle until mark is covered**

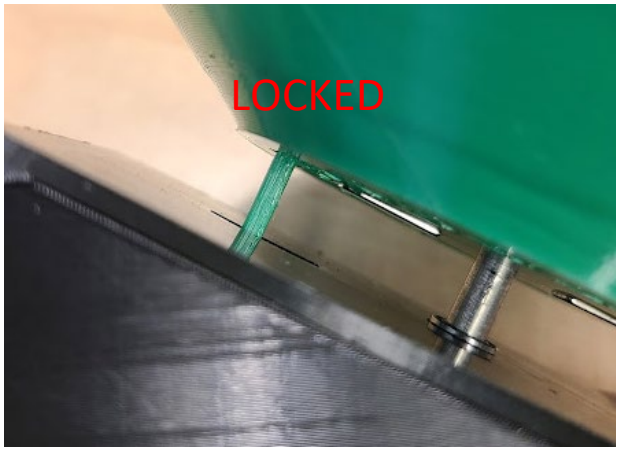


2.4.11 Insert axle into the base

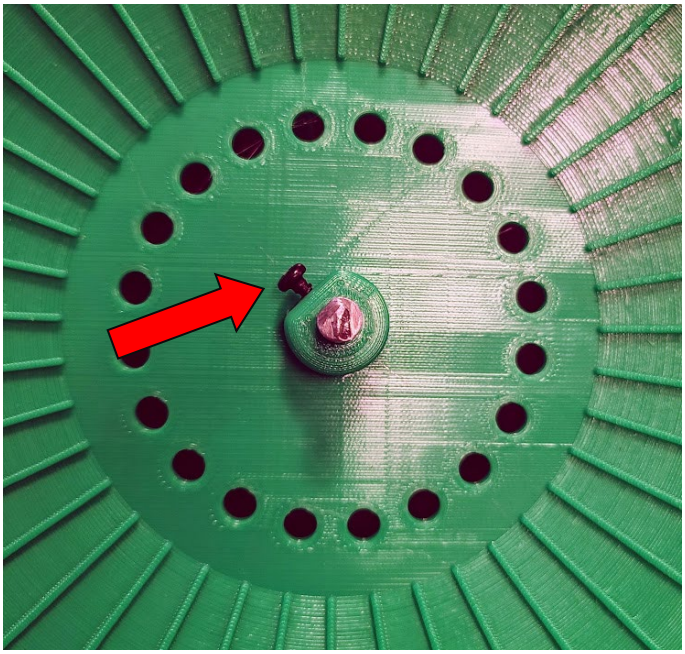
2.4.12 Confirm locking mechanism operation by powering wheel and using the setup sketch

a. **Subtle adjustments of the wheel/shaft may be required for proper locking**
b. **When in the open position, there should be 2-3 mm of clearance between wheel and pin**





2.4.13 Secure wheel to axle with a M3x4 screw



3. Programs and modes

The LOST Wheel can be programmed for either distance-based locking (Distance Mode) or for timed intervals of running and rest (Timer Mode). Both modes can be used for a standalone wheel (no computer required) or in combination with the LOST-Wheel Logger application to obtain more detailed data about speed and time duration.

Distance Mode – the wheel begins unlocked and will lock upon reaching a set milestone.

Timer Mode – the wheel can be set to unlock or lock at certain intervals that are based on when the device is powered on.

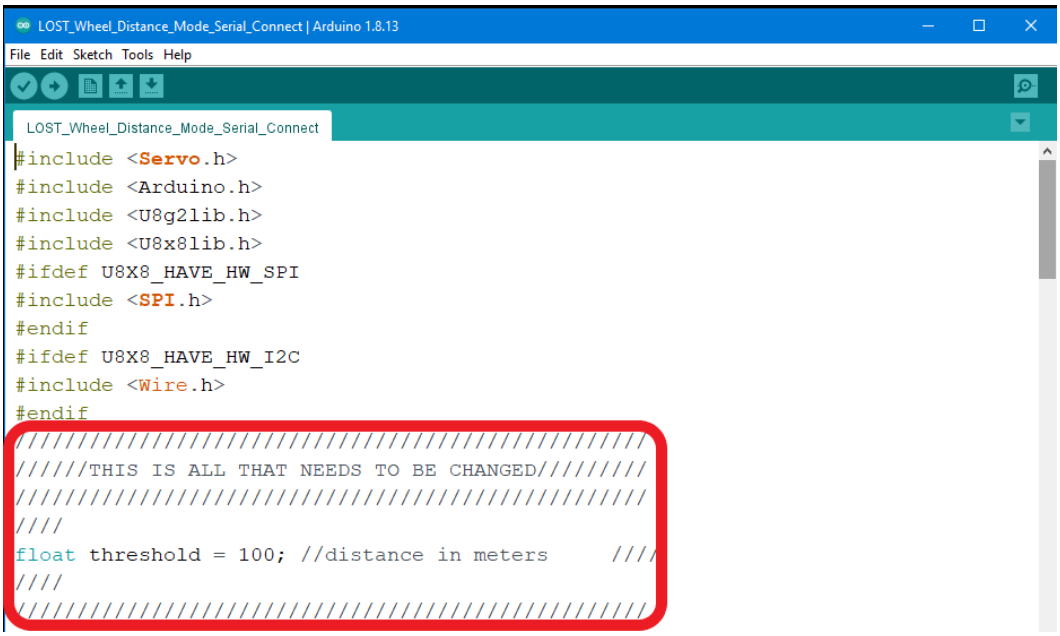
LOST-Wheel Logger – this data collection application is run through RStudio. It prompts users to input the length of data collection, the USB port that the wheel is connected to, and a unique ID for the wheel for recordkeeping. Multiple wheels can be attached to one computer through a USB splitter and by opening several RStudio windows.

3.1 Distance Mode

This mode will lock the wheel once the distance milestone is reached. To reset, disconnect and reconnect from a power source.

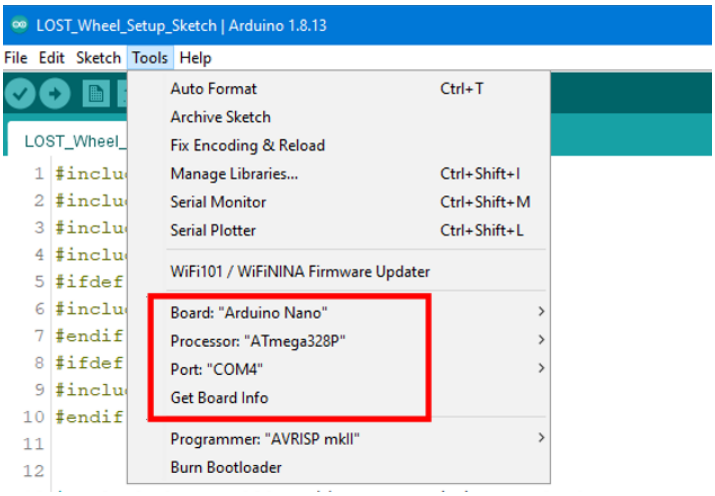
3.1.1 Open Arduino IDE and the LOST_Wheel_Distance_Mode_Serial_Connect sketch

a. Enter the distance at which wheel will lock and prevent any further running

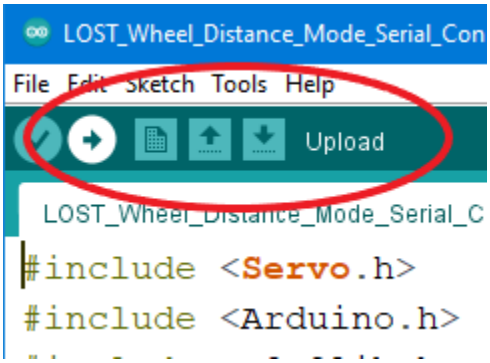


3.1.2 Confirm connection between Nano and computer

The "Port:" may need to be changed. This varies based on which USB connection is being used.



3.1.3 Upload the sketch to your LOST-Wheel



3.1.4 The LOST-Wheel is now ready to be used in Distance Mode!

3.2 Timer Mode

This mode allows users to specify times at which the wheel will be locked and unlocked. It is important to note that the Arduino Nano does not have a real time clock built in. This means it cannot tell time in the conventional sense. The timer used in this mode is based on when the device is powered on. Therefore, some coordination is required between programming and turning the device on. For example, if you would like the wheel to unlock at 7 PM, and plan on resetting the wheel when you leave the lab at 5 PM, set the initial **lockedPeriod = 2**

3.2.1 Open Arduino IDE and the LOST_Wheel_Timer_Mode_X_States sketch

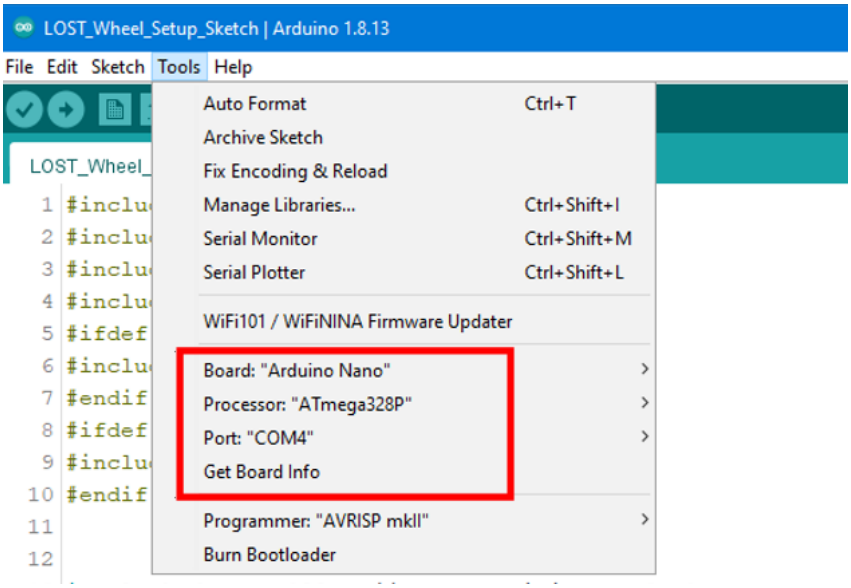
a. **Choose the sketch that corresponds to the number of states you require for your experiment**
b. **This mode is set to immediately lock upon startup**
    i. **To have the wheel unlocked at startup, set lockedPeriod = 0**
c. **This mode will always lock after the final unlocked period**

LOST_Wheel_Timer_Mode_4_states §

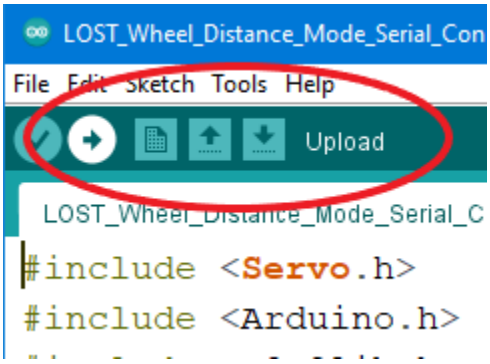
```
#include <Servo.h>
#include <Arduino.h>
#include <U8g2lib.h>
#include <U8x8lib.h>
#ifdef U8X8_HAVE_HW_SPI
#include <SPI.h>
#endif
#ifdef U8X8_HAVE_HW_I2C
#include <Wire.h>
#endif
////////////////////////////////////////////////
//THIS IS ALL THAT NEEDS TO BE CHANGED/////////
////////////////////////////////////////////////
float lockedPeriod = 0; ///// time after startup for which the wheel will stay locked - enter in hours

float unlockPeriod = 4; //// time the wheel will stay unlocked for - enter in hours

float lockedPeriod2 =  7; //// time the wheel will stay locked for after first unlock period - enter in hours

float unlockPeriod2 =  12; //// time the wheel will stay unlocked for after the second locking - enter in hours
////////////////////////////////////////////////
```

3.2.2 Confirm connection between LOST-Wheel and computer

The “Port:” may need to be changed. This varies based on which USB connection is being used.



3.2.3 Upload the sketch to your LOST-Wheel



3.2.4 The LOST-Wheel is now ready to be used in Timer Mode!

3.3 LOST-Wheel Logger

The LOST-Wheel Logger is an R-based application that interfaces with the LOST-Wheel that has been pre-programmed as outlined in 3a. and 3b. The application is launched from RStudio and requires several packages (prewritten functions) to be installed. The first time the Logger is started, these packages will automatically download.

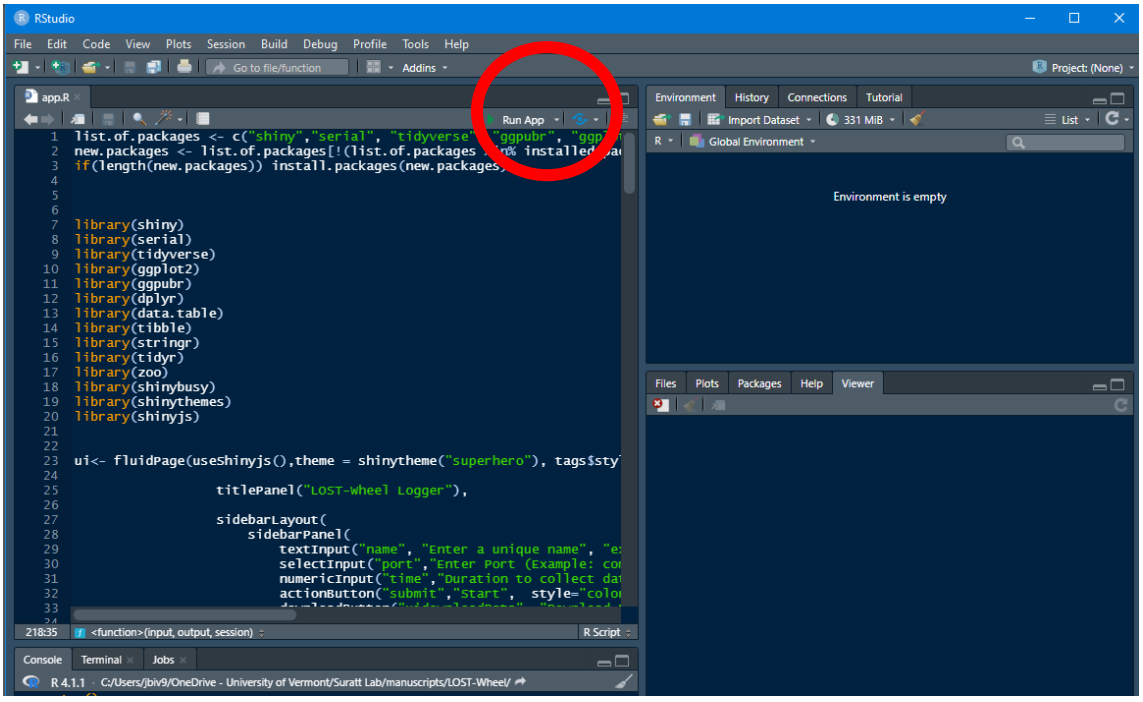
3.3.1 Before launching application:

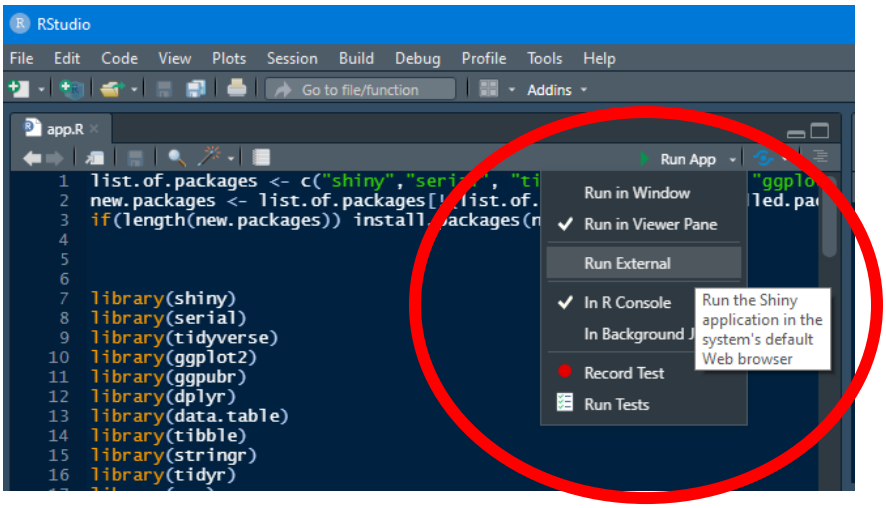
a. **Turn off automatic updates**
b. **Turn power saving settings off – the computer cannot sleep while collecting data**
c. **Plug in LOST-Wheel to USB**

3.3.2 To open the LOST-Wheel Logger, launch RStudio and open the file titled “app.R”

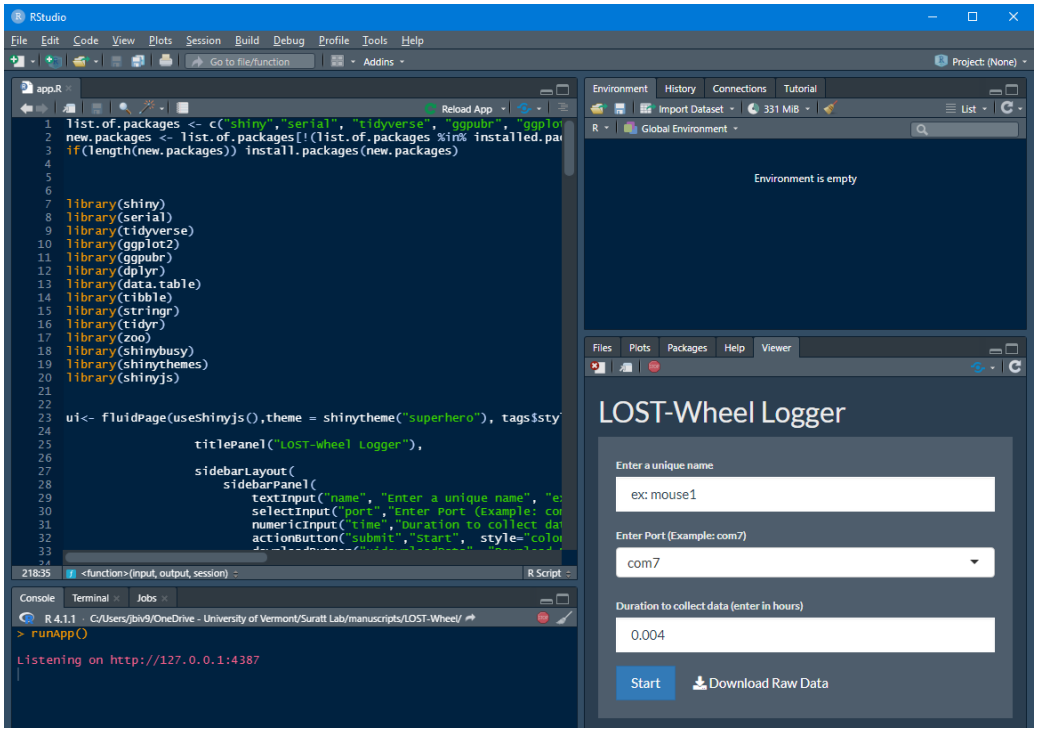
3.3.3 Click “Run App”

3.3.4 The logger can be launched in a default internet browser by selecting the option from the dropdown menu





3.3.5 The Logger will run in the "Viewer" pane of RStudio.



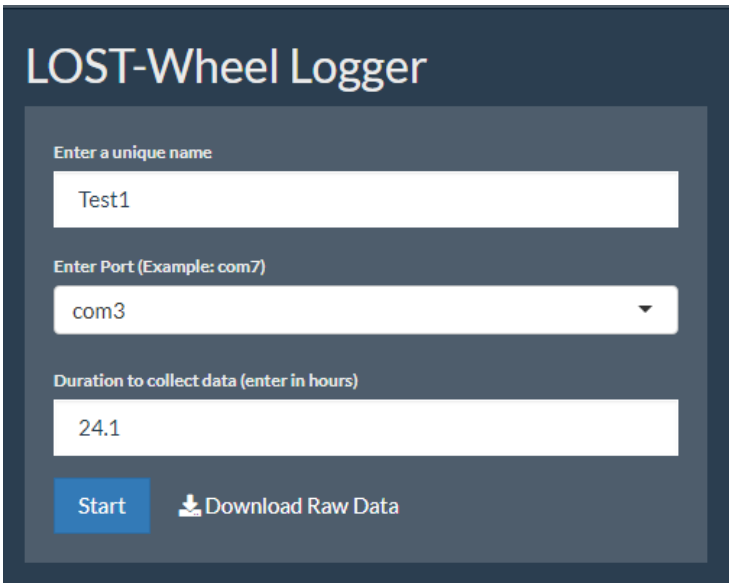
3.3.6 To start data collection:

Enter a unique subject ID

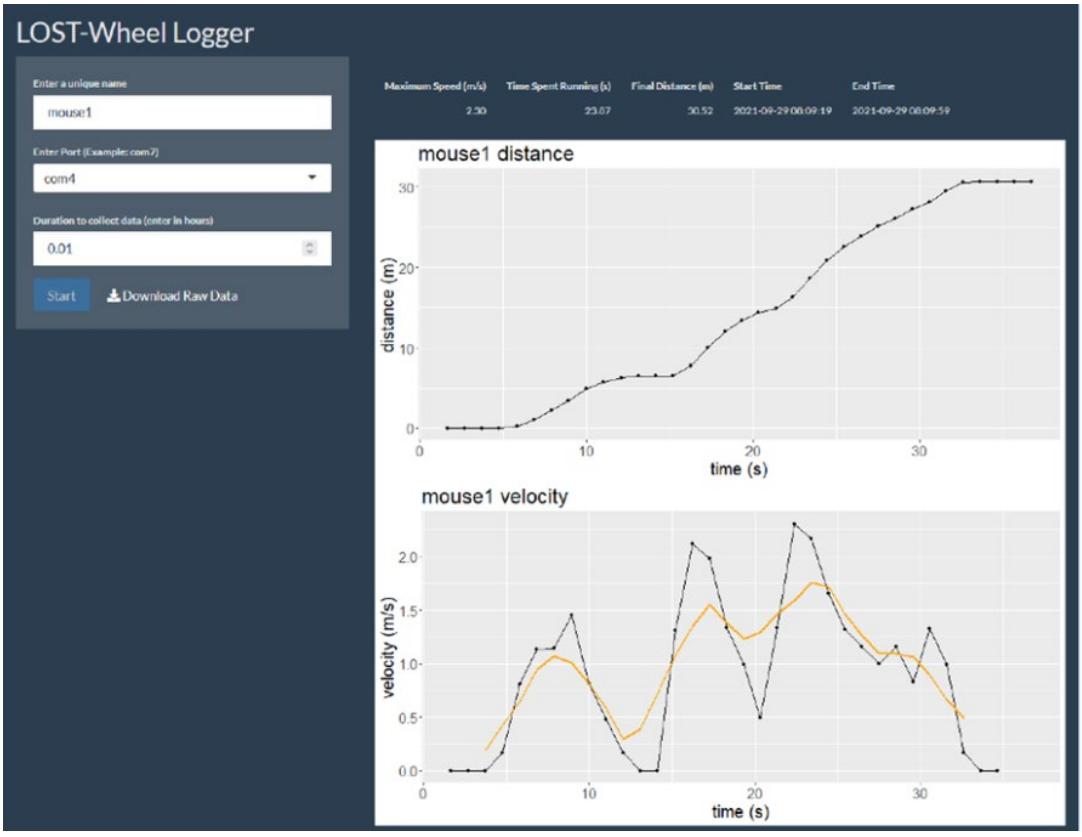
Select the correct COM port for which the wheel is connected to

Enter the time of data collection

Select "Start"



3.3.7 The "Start" button will become inactive and a loading screen will appear. After data collection has finished, the Logger will display the following screen containing maximum speed, time spent running, distance traveled, as well as Distance/Time and Velocity/Time graphs. These graphs can be saved by using the right mouse button and saving the file. Raw data can also be saved in a .csv format by using the "Download Raw Data" button.

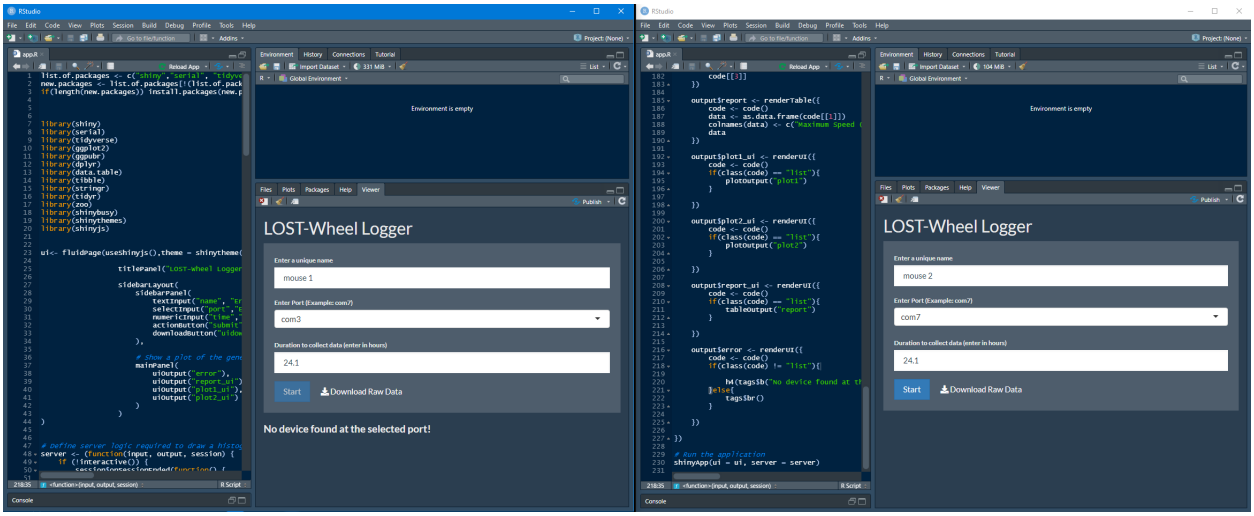


3.4 Multiple Wheel Collection

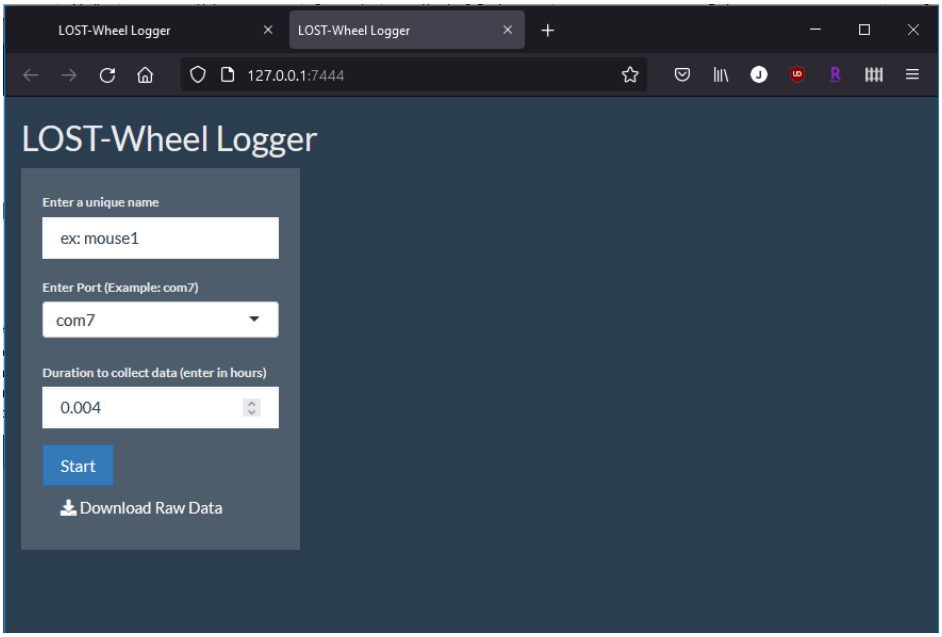
3.4.1 Simultaneous collection from multiple wheels

3.4.1.1 To collect data from multiple wheels using the LOST-Wheel Logger, open several instances of RStudio, and the “app.R” file.

3.4.1.2 Collecting data from multiple wheels using the Viewer Panel:



3.4.1.3 Collecting data from multiple wheels in an internet browser using the “Run External” option:



3.5 Troubleshooting

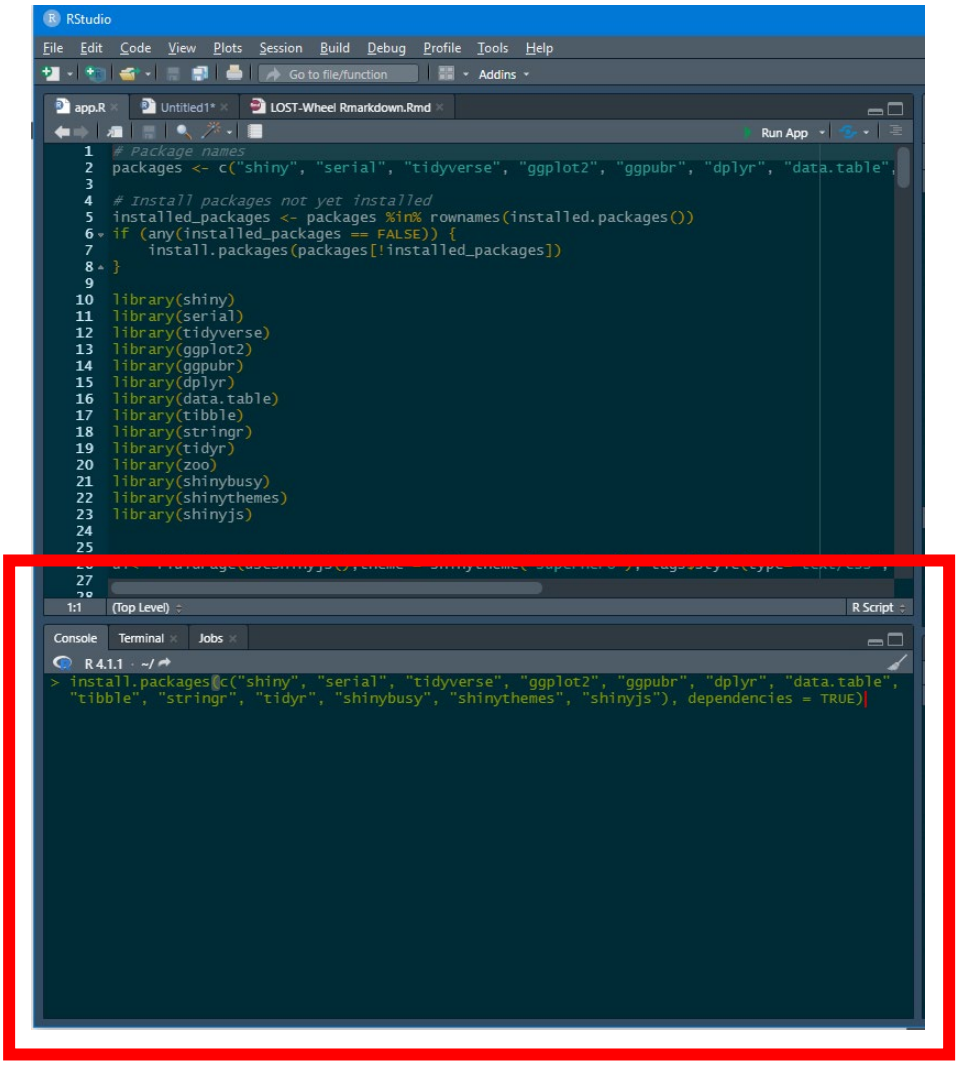
3.5.1 Packages not installed

3.5.1.1 The LOST-Wheel Logger requires the following R packages which should be automatically installed on the first use of the application:

"shiny", "serial", "tidyverse", "ggplot2", "ggpubr", "dplyr", "data.table", "tibble", "stringr", "tidyr", "shinybusy", "shinythemes", "shinyjs"

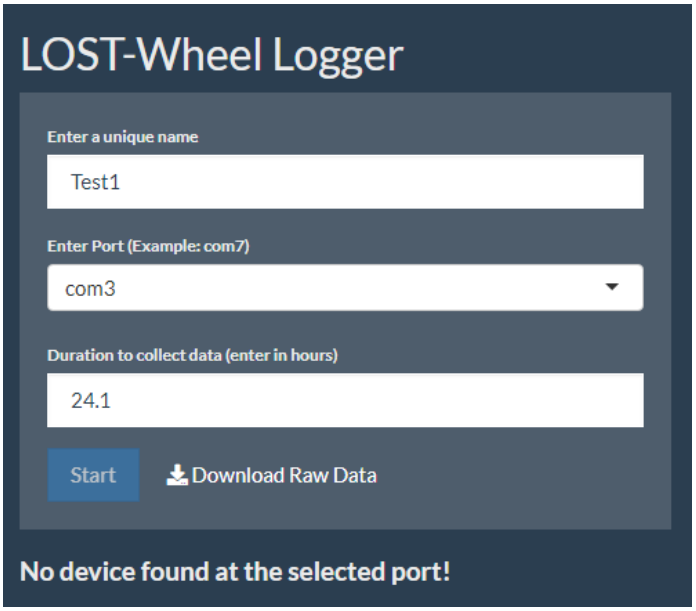
3.5.1.2 If the application does not automatically install these packages, COPY and PASTE the following lines of code into the console. Then hit ENTER to run. This will install the missing packages.

```
install.packages(c("shiny", "serial", "tidyverse", "ggplot2", "ggpubr", "dplyr", "data.table", "tibble", "stringr", "tidyr", "shinybusy", "shinythemes", "shinyjs"), dependencies = TRUE)
```



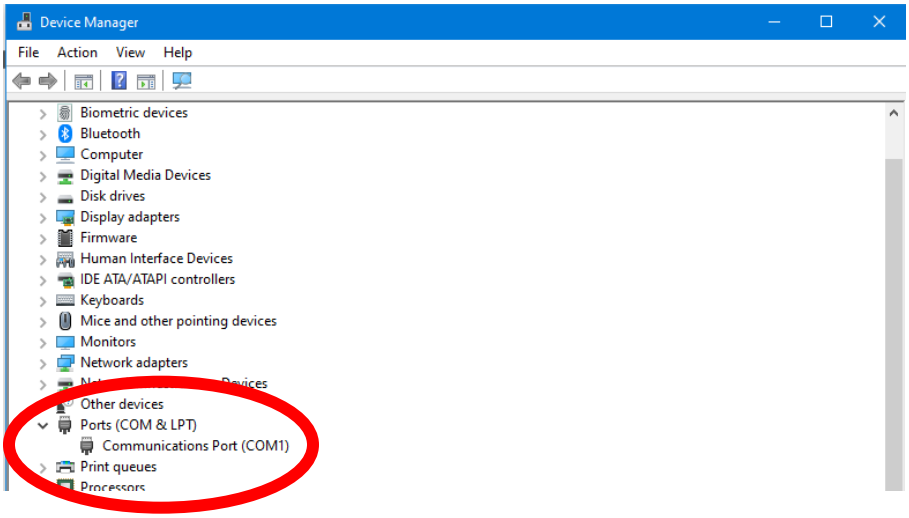
3.5.2 COM Port Error

3.5.2.1 If the incorrect COM port is selected, the following error will be displayed



3.5.2.2 To determine the correct COM port, open the computer's Device Manager

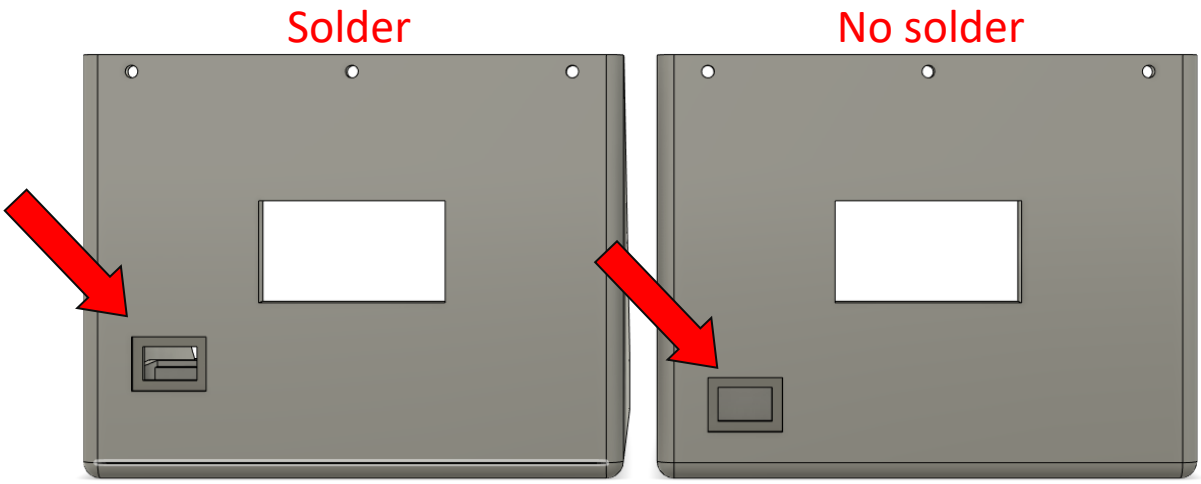
Navigate to "Ports (COM & LPT)" and expand the options to view active ports



3.5.2.3 Restart the application and select the correct COM port

3.5.3 Using pin-style connections to the Arduino Nano

3.5.3.1 It is pivital to download and print the correct bottom section. If using the pin-style connections, please use the file labeled "LOST-Wheel Bottom – no solder version."



The USB port has been moved to account for the microcontroller pins

3.5.3.2 Two ground wires will need to be connected together: visit this link for a tutorial

3.5.3.2.1 https://mschoeffler.com/2017/12/26/diy-y-adapter-jumper-wire/

3.5.3.3 Connect all wires to the Nano