



Bluetooth Controlled Motorized Camera Slider

Created by Ruiz Brothers



<https://learn.adafruit.com/bluetooth-motorized-camera-slider>

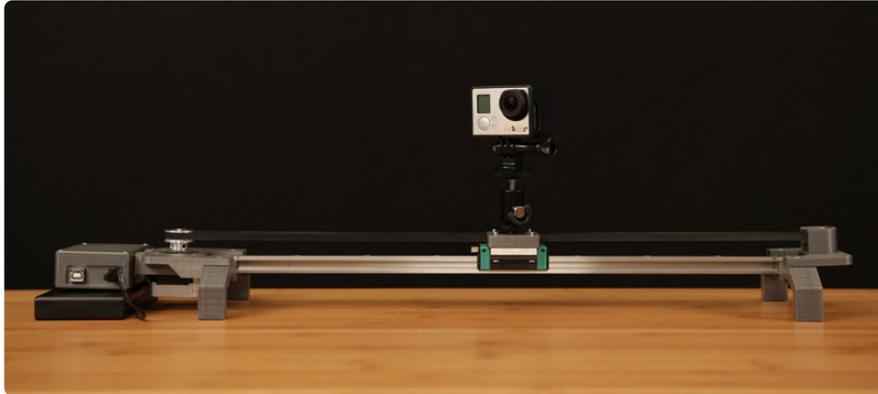
Last updated on 2024-06-03 01:47:56 PM EDT

Table of Contents

Overview	5
<ul style="list-style-type: none">• Arduino Uno + Motor Shield V2 + Bluefruit LE SPI Friend• Portable, Wireless and Light-weight• Prerequisite Guides• Parts• Tools and Supplies• Hardware Screws	
3D Printing	8
Circuit Diagram	9
Software	10
<ul style="list-style-type: none">• Uploading Sketches• Advanced Controller Motor Shield Sketch• Simple Controller Motor Shield Sketch	
Hardware	14
<ul style="list-style-type: none">• Prep Boards• Install headers to Bluefruit SPI Friend• Install Adafruit Bluefruit LE SPI Friend to Motor Shield• Prep Wires for Motor Shield• Solder wires into the pins on the Motor Shield• Motor Shield + Bluefruit LE SPI Friend Wiring Diagram• Measure and cut headers for the Motor Shield• Install headers to Motor Shield• Double check for any cold solder joints.• Install the VIN Jumper to the Motor Shield• Wiring Complete	
Assembly	23
<ul style="list-style-type: none">• Install the GT2 Timing Pulley• Add Hardware to Camera Mount• Install belt to camera mount• Install Tripod Screw and Ball Head• Install Camera Mount to Railing Platform• Install railing platform onto support rail• Add some Lube (Optional)• Install feet onto support rail• Install stepper motor to mount• Install Arduino Uno into Enclosure• Install Stepper Wires• Secure Motor mount to enclosure• Install Stepper Wires to Motor Shield• Close enclosure• Install bearing and belt to pulley mount• Install Pulley Mount to foot• Install belt to timing pulley on stepper motor• Load Power Pack• Connect and Control	

-
- [Adafruit Bluefruit LE Connect Mobile App](#)
 - [Control Features](#)
 - [What Cameras should I use?](#)
 - [Perspective, Angles and Composition](#)
 - [Optimal Slide Direction](#)
 - [Timelapse / Video](#)

Overview



Arduino Uno + Motor Shield V2 + Bluefruit LE SPI Friend

We really love photography and camera gear here at Adafruit, so we came together and upgrade our original [3D Printed DIY camera slider](https://adafru.it/iqa) (<https://adafru.it/iqa>).

It's now motorized using a single NEMA-17 stepper driven by the Adafruit Motor Shield on top of an Arduino Uno.

The Adafruit Bluefruit SPI Friend allows us to remotely control the movement of the camera slider using the Adafruit Bluefruit LE Connect app on a mobile gadget like an Android or iOS device!



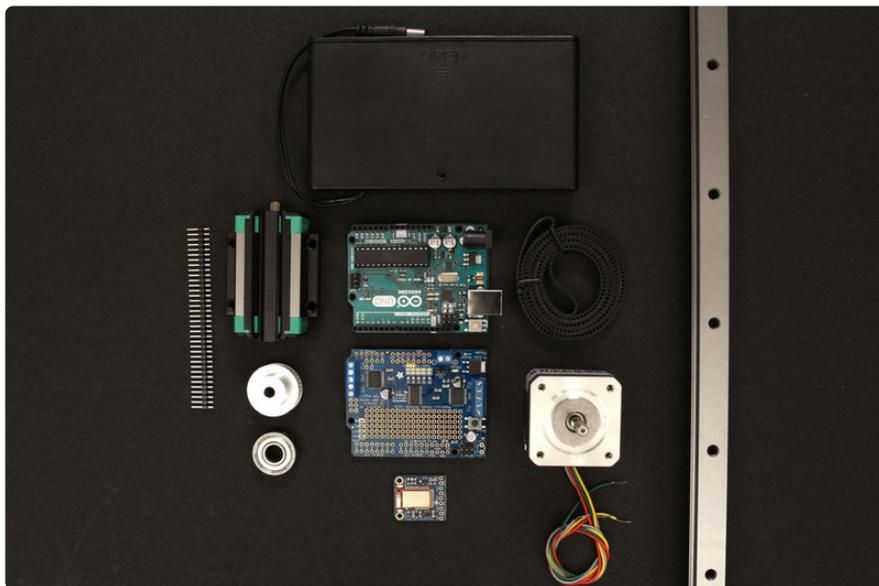
Portable, Wireless and Light-weight

Using a 500mm long support slide rail and platform allows you to create a basic DIY camera slider. A GT2 timing pulley mounted to the stepper motor and a radial ball bearing allows a GT2 timing belt to pull the platform across the support slide rail. With a few 3D printed parts, an Arduino Uno and some components, we can build a very capable DIY motorized camera slider.

Prerequisite Guides

Before starting this project, we recommend checking out the guides listed below.

- [Adafruit Guide to Excellent Soldering \(https://adafru.it/drl\)](https://adafru.it/drl)
- [Arduino Motor Shield V2 for Arduino \(https://adafru.it/doc\)](https://adafru.it/doc)
- [Adafruit Bluefruit LE SPI Guide \(https://adafru.it/iqb\)](https://adafru.it/iqb)



Parts

You can obtain all the parts to build this project in the Adafruit shop. You can get all the parts you need by clicking on the side bar on the right.

- [Arduino Uno \(http://adafru.it/50\)](http://adafru.it/50)
- [Adafruit Motor Shield V2 \(http://adafru.it/1438\)](http://adafru.it/1438)
- [Adafruit Bluefruit LE SPI Friend \(http://adafru.it/2633\)](http://adafru.it/2633)
- [NEMA 17 Stepper Motor \(http://adafru.it/324\)](http://adafru.it/324)
- [500mm x 15mm Slide Rail \(http://adafru.it/1861\)](http://adafru.it/1861)

- [15mm Slide Railing Platform](http://adafru.it/1866) (<http://adafru.it/1866>)
- [36 Tooth GT2 Timing Pulley](http://adafru.it/1253) (<http://adafru.it/1253>)
- [608ZZ Radial Ball Bearing](http://adafru.it/1178) (<http://adafru.it/1178>)
- [1164mm long GT2 Timing Belt](http://adafru.it/1184) (<http://adafru.it/1184>)
- [8 x AA battery holder](http://adafru.it/875) (<http://adafru.it/875>)
- [3/8" to 1/4" Adapter Screw](http://adafru.it/2392) (<http://adafru.it/2392>)
- [Shoe Mount / Bracket](http://adafru.it/2410) (<http://adafru.it/2410>)
- [Swivel-Head Pan Tilt](http://adafru.it/2464) (<http://adafru.it/2464>)
- [1/4" Screw with D-Ring](http://adafru.it/2629) (<http://adafru.it/2629>)
- Tripod Adapter for Mobile Phone / GoPro

Tools and Supplies

The following tools and supplies are necessary to build and assemble this project.

- [3D Printer](https://adafru.it/doT) (<https://adafru.it/doT>)
- [Soldering Iron](https://adafru.it/doU) (<https://adafru.it/doU>)
- [Panavise Jr.](http://adafru.it/151) (<http://adafru.it/151>)
- [Helping Third Hand](http://adafru.it/291) (<http://adafru.it/291>)
- [Wire Strippers](http://adafru.it/527) (<http://adafru.it/527>)
- [Filament](http://adafru.it/2080) (<http://adafru.it/2080>)
- [Solder](https://adafru.it/doU) (<https://adafru.it/doU>)
- [26AWG Wire](http://adafru.it/1970) (<http://adafru.it/1970>)
- [Screwdriver](http://adafru.it/829) (<http://adafru.it/829>)



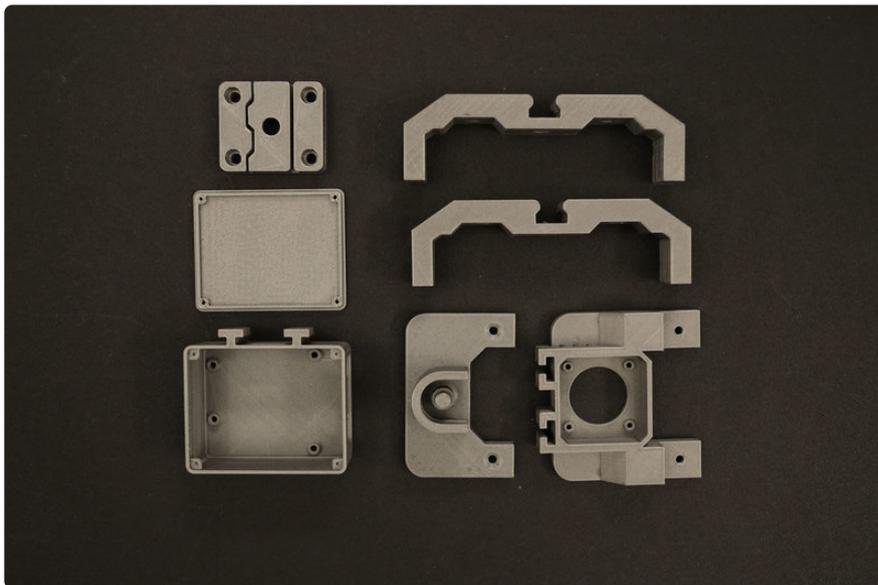
The number of screws in the photo do not match up with the actual quantity!
Follow the listing below.

Hardware Screws

The bits listed below are required to assemble this project. Yeah, we know... Imperial.

- 8x #6-32 3/4in
- 4x #2-56 3/8in
- 6x #6-32 3/8in
- 4x 8mm long M3
- 1x 1/4 20
- 2x #6-32 1in

3D Printing



These parts will fit on 3D printers with a minimum bed of 100mm x 100mm. The parts can be printed without any support material. They're oriented in the center and positioned to print "as-is".

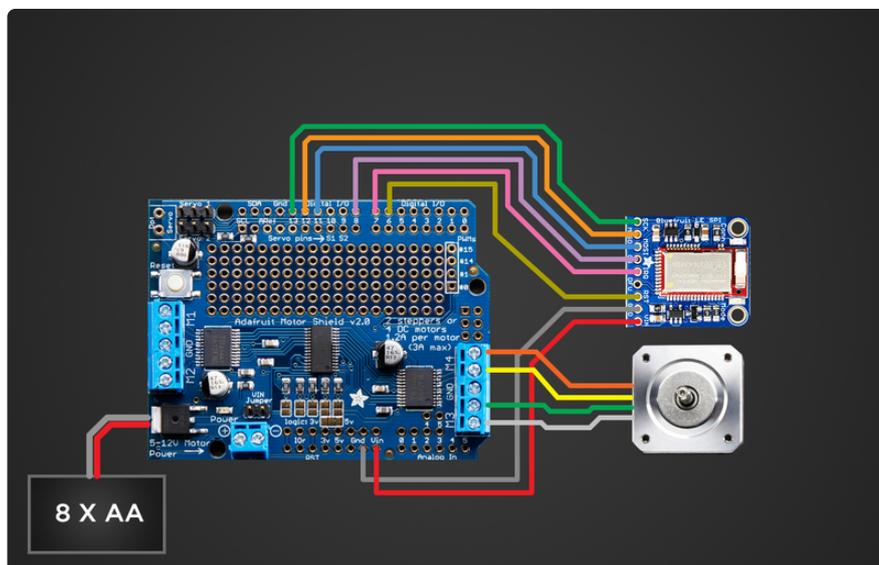
You can download the files for this project by clicking the button below.

[Download STLs](https://adafru.it/iqc)

<https://adafru.it/iqc>

enclosure-top.stl	1x cover to the enclosure box	
enclosure-box.stl	1x enclosure for Uno and Adafruit Motor Shield V2	
rail-foot.stl	2x feet for supported slide rail	
camera-mount.stl	1x camera to slide railing platform adapter	
bearing-mount.stl	1x mount for radial ball bearing	
motor-mount.stl	1x mount for NEMA-17 size stepper motor	

Circuit Diagram



To build the camera slider with the Adafruit Motor Shield V2 for Arduino, wire the components together as shown in the diagram above.

- Connect Arduino **VIN** to Bluefruit LE SPI **VIN** pin
- Connect Arduino **GND** to Bluefruit LE SPI **GND** pin
- Connect Arduino **D13** to Bluefruit LE SPI **SCK** pin

- Connect Arduino **D12** to Bluefruit LE SPI **MISO** pin
- Connect Arduino **D11** to Bluefruit LE SPI **MOSI** pin
- Connect Arduino **D8** to Bluefruit LE SPI **CS** pin
- Connect Arduino **D7** to Bluefruit LE SPI **IRQ** pin
- Connect Arduino **D6** to Bluefruit LE SPI **RST** pin

The diagram above shows the wiring for **NEMA-17** size stepper. Connect the four wires from the stepper to motor outputs on **M3** and **M4**.

An 8 x AA battery pack will supply the necessary 12V to the Arduino Uno via a 2.1mm barrel jack.

Software

To use the controller motorshield sketch you'll want to make sure you're using the [latest version of the Arduino IDE \(https://adafru.it/fvm\)](https://adafru.it/fvm) (1.6.5 at the time of this writing).

If you're totally new to Arduino take a little time to go through some introductory tutorials like [how to make a LED blink \(https://adafru.it/iqd\)](https://adafru.it/iqd). This will help you understand how to use the IDE, load a sketch, and upload code.

Next you'll need to make sure the libraries used by the sketch are installed. With the latest Arduino IDE you can use its [library manager \(https://adafru.it/fCN\)](https://adafru.it/fCN) to easily install libraries, or check out this [guide on how to manually install a library \(https://adafru.it/dNR\)](https://adafru.it/dNR). You'll want to install the following libraries:

- Adafruit Motor Shield V2 Library
- Adafruit Adafruit BluefruitLE nRF51

Search for the libraries in the library manager and they should be easy to find and install:

If you already have one or more of these libraries installed then make sure to update it to the latest version.

Uploading Sketches

To load the sketch make sure the libraries above are installed, and the Arduino is connected to the computer through a USB cable. Make sure under the **Tools -> Board**

menu the **Arduino Uno** is selected, and under the **Tools -> Port** menu the serial port for the Arduino is selected (it should say **Arduino Uno**).

Then press the upload button or click the **Sketch -> Upload** item to send the code to the Arduino. Woo-hoo the sketch should be running!

Advanced Controller Motor Shield Sketch

This sketch features several functions for moving the slider. You can set speed and choose linear or easing movements.

```
// Smartphone- or tablet-activated timelapse camera slider.
// Uses the following Adafruit parts:
//
// Arduino Uno R3 or similar (adafruit.com/product/50 or #2488)
// Bluefruit LE SPI Friend (#2633)
// Motor/Stepper/Servo Shield v2 (#1438)
// NEMA-17 Stepper motor (#324)
// Miscellaneous hardware and 3D-printed parts; see guide for full list.
//
// Needs Adafruit_BluefruitLE_nRF51 and Adafruit_MotorShield libs:
// github.com/adafruit
// Use Adafruit Bluefruit LE app for iOS or Android to control timing.
// Buttons 1-4 select interpolation mode (linear vs. various ease in/out).
// Up/down select speed. Left = home. Right = start slider.

#include <SPI.h>;
#include <Wire.h>;
#include <SoftwareSerial.h>;
#include <Adafruit_BluefruitLE_SPI.h>;
#include <Adafruit_MotorShield.h>;

#define LED 13 // Built-in LED on pin 13

// Bluefruit config -----
Adafruit_BluefruitLE_SPI ble(8, 7, 6); // CS, IRQ, RST pins

// Stepper motor config -----
#define STEPPER_STEPS 1100 // Length of slider
#define STEPPER_RPM 20

Adafruit_MotorShield AFMS = Adafruit_MotorShield();

// Stepper motor w/200 steps/revolution (1.8 degree) on port 2 (M3 & M4)
Adafruit_StepperMotor *motor = AFMS.getStepper(200, 2);

// Global stuff -----

// Four motion interpolation modes are supported (corresponding to buttons
// 1-4 in the Bluefruit LE app for iOS & Android): linear is constant speed,
// ease_in_out accelerates/decelerates at the ends (fastest in middle),
// ease_in accelerates (fastest at end) and ease_out decelerates (fastest
// at beginning):
typedef enum interp { linear, ease_in_out, ease_in, ease_out };

// A few different time periods are supported, but not a whole lot. Since
// there's no display connected, we just blink the onboard LED to indicate
// which setting is active, and there's only so many speed changes one can
// reliably 'read' this way. Feel free to edit, but keep in mind there are
// upper and lower limits to the time interval -- the stepper can only move
// so fast and if you try to press beyond that it'll just move linearly
```

```

// regardless of the selected interpolation mode, and the micros() function
// rolls over about every 70 minutes, so the duration can't exceed that.
// Blink rate is similarly constrained due to BLE comm; about 2 Hz max.
struct {
  uint32_t totalTime; // Total duration of movement along slider
  uint32_t LEDflashTime; // LED blink rate indicates selected speed
} speed[] = {
  5 * 60 * 1000000L, 1000000L / 2, // 5 min slide, 2 Hz blink
  10 * 60 * 1000000L, 1000000L, // 10 min slide, 1 Hz blink
  20 * 60 * 1000000L, 1000000L * 2, // 20 min slide, 1/2 Hz blink
  60 * 60 * 1000000L, 1000000L * 3 // 60 min slide, 1/3 Hz blink
};
#define N_SPEEDS (sizeof(speed) / sizeof(speed[0]))

uint32_t startTime = 0; // micros() value when slider movement started
interp mode = linear; // Selected interpolation mode
uint8_t speedIdx = 0; // Selected speed index (0 to N_SPEEDS-1)
boolean moving = false; // True if motor active & interpolating

// Setup function - runs once at startup -----
void setup(void) {
  Serial.begin(57600);
  pinMode(LED, OUTPUT);
  digitalWrite(LED, HIGH); // LED steady on during init
  if(!ble.begin(false)) for(;;); // BLE init error? LED on forever
  ble.echo(false);
  AFMS.begin();
  motor->setSpeed(STEPPER_RPM);
  motor->release(); // Allow manual positioning at start
  digitalWrite(LED, LOW); // LED off = successful init
}

// Loop function - repeats forever -----
void loop(void) {
  static uint16_t prevSliderPos = 0; // Slider position on prior call
  static uint8_t led; // Blink status on/off

  if(moving) { // Motor currently engaged?
    float p, t = (float)(micros() - startTime) /
      (float)speed[speedIdx].totalTime; // Time 0.0 - 1.0

    switch(mode) { // Cubic easing functions from http://gizma.com/easing/
      case linear:
        p = (float)STEPPER_STEPS * t;
        break;
      case ease_in_out:
        t *= 2.0;
        if(t < 1.0) {
          p = (float)STEPPER_STEPS * 0.5 * t * t * t;
        } else {
          t -= 2.0;
          p = (float)STEPPER_STEPS * 0.5 * (t * t * t + 2.0);
        }
        break;
      case ease_in:
        p = (float)STEPPER_STEPS * t * t * t;
        break;
      case ease_out:
        t -= 1.0;
        p = (float)STEPPER_STEPS * (t * t * t + 1.0);
        break;
    }

    // Move stepper to new position (if required)
    uint16_t sliderPos = (int)(p + 0.5);
    if(sliderPos > prevSliderPos) {
      // Microstepping is used to reduce vibration
    }
  }
}

```

```

    motor->step(sliderPos - prevSliderPos, FORWARD, MICROSTEP);
    prevSliderPos = sliderPos;
    if(sliderPos >= STEPPER_STEPS) { // At end of motion?
        motor->release();           // Turn off motor
        moving = false;
    }
} else {
    if((micros() - startTime) > (speed[speedIdx].LEDflashTime / 2)) {
        digitalWrite(LED, led++ & 1);
        startTime = micros();
    }
}

// Process any pending Bluetooth input
if(ble.isConnected()) {
    ble.println(F("AT+BLEUARTX")); // Request string from BLE module
    ble.readline();                // Read outcome
    if(!strncmp(ble.buffer, "B", 2) && // Controller button command
        checkCRC(255-'!'-'B', 4) && // Verify checksum
        (ble.buffer[3] == '1')) { // Button press? 1=press 0=release
        switch(ble.buffer[2]) {
            case '1': if(!moving) mode = linear; break;
            case '2': if(!moving) mode = ease_in_out; break;
            case '3': if(!moving) mode = ease_in; break;
            case '4': if(!moving) mode = ease_out; break;
            case '5': // Up (faster)
                if(!moving && speedIdx) speedIdx--;
                break;
            case '6': // Down (slower)
                if(!moving && (speedIdx < (N_SPEEDS-1))) speedIdx++;
                break;
            case '7': // Left (home)
                digitalWrite(LED, LOW);
                motor->step(prevSliderPos, BACKWARD, DOUBLE);
                motor->release();
                prevSliderPos = 0;
                moving = false;
                break;
            case '8': // Right (start/pause)
                if(!moving) { // Don't double-start
                    digitalWrite(LED, LOW);
                    moving = true;
                    startTime = micros();
                }
                break;
        }
    }
}

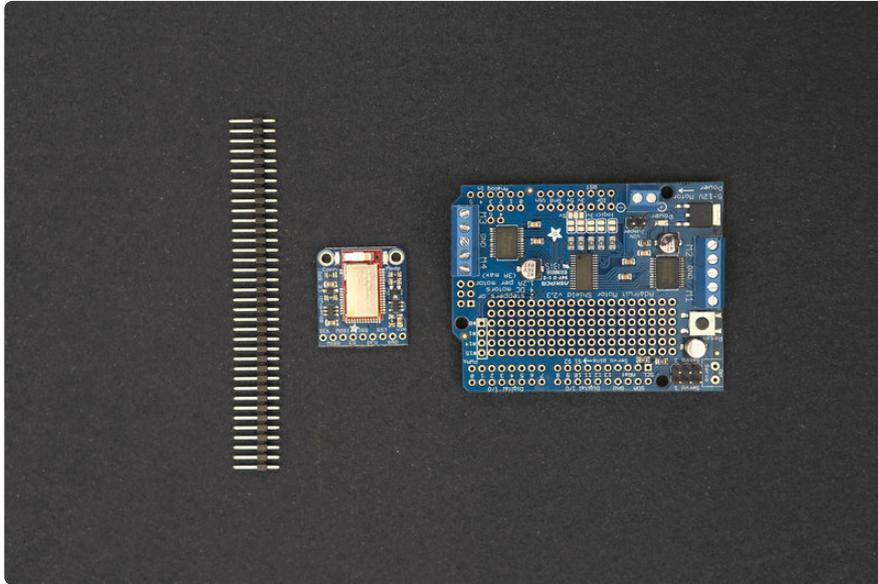
boolean checkCRC(uint8_t sum, uint8_t CRCindex) {
    for(uint8_t i=2; i<CRCindex; i++) sum -= (uint8_t)ble.buffer[i];
    return ((uint8_t)ble.buffer[CRCindex] == sum);
}

```

Simple Controller Motor Shield Sketch

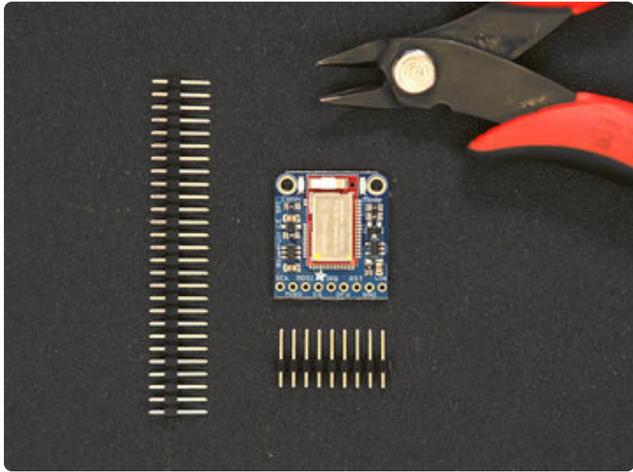
This sketch do not feature any easing functions. It moves linearly and do not have any periodic stops. This sketch is ideally for heavier cameras that want to use double steps. The controls are easier to modify and understand.

Hardware



Prep Boards

Grab the Adafruit Motor Shield V2 and the Adafruit Bluefruit LE SPI Friend module. We need to stack the module on top of the motor shield. To do that, we need to solder the included headers to the pins on the Adafruit Bluefruit LE SPI Friend.



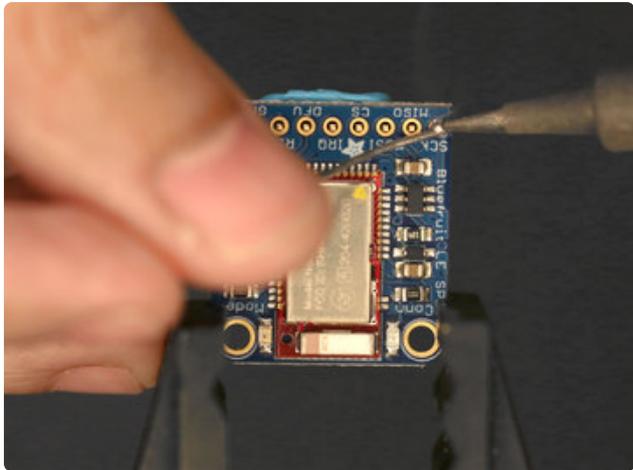
Install headers to Bluefruit SPI Friend

Insert the header into pins and cut it down to size.

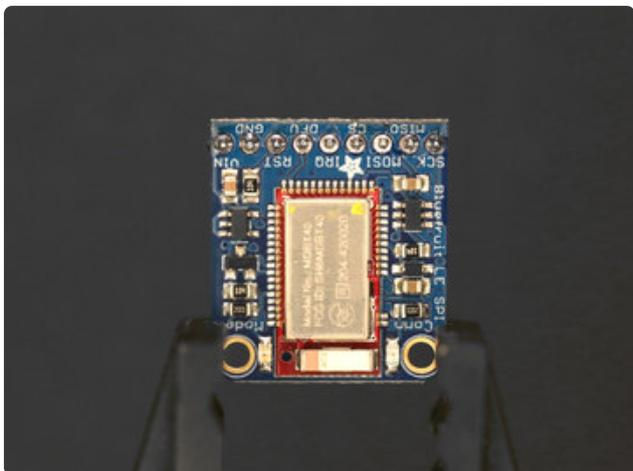


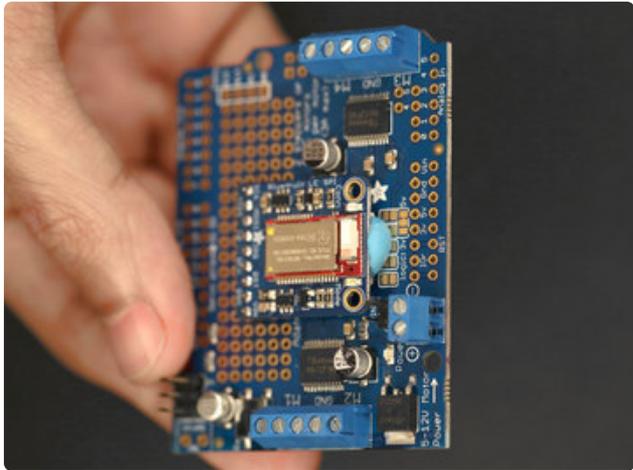
I recommend using tack putty to hold the header in place while soldering.

You can use a panavise to hold PCB in place while you solder the header to the pins on the module.



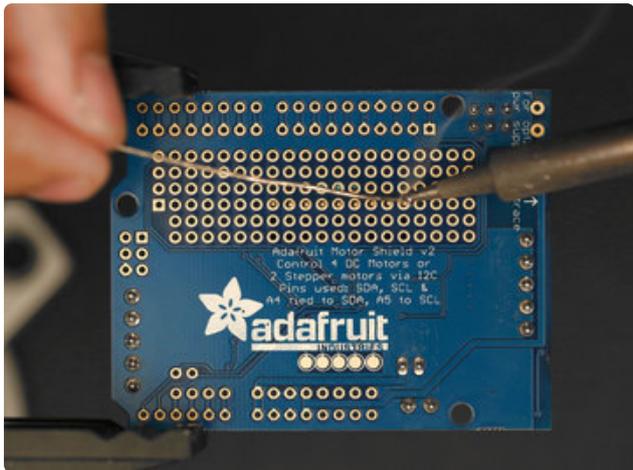
Be sure to solder all the pins in place and double check your work for any cold solder joints.



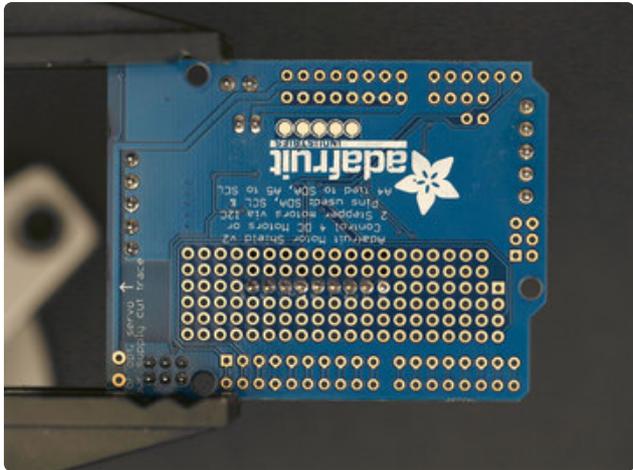


Install Adafruit Bluefruit LE SPI Friend to Motor Shield

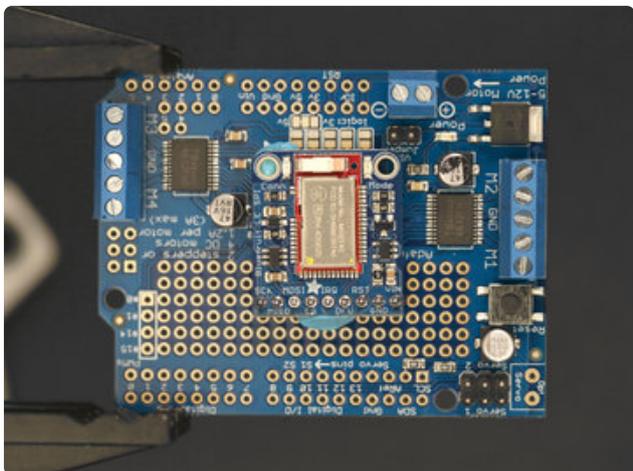
Once the headers are soldered to the SPI Friend, insert them on top of the prototyping area on the motor shield. Reference the photo to see best placement.



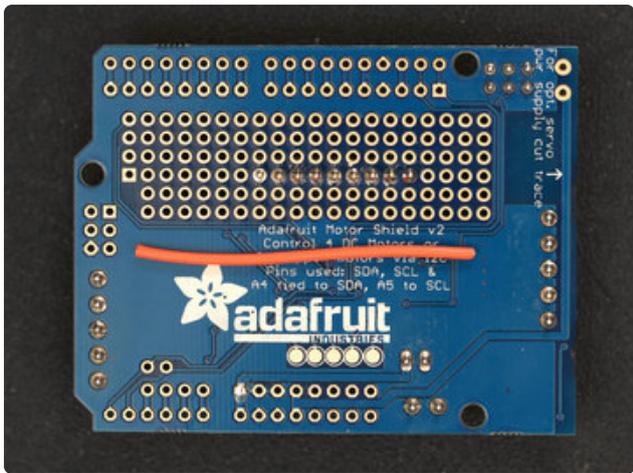
Again, I recommend using tack putty to hold the PCB in place while you solder. Fitting a small amount underneath the PCB will elevate it.



Flip the board over and use a panavise to hold it in place.



Solder the header pins in place.



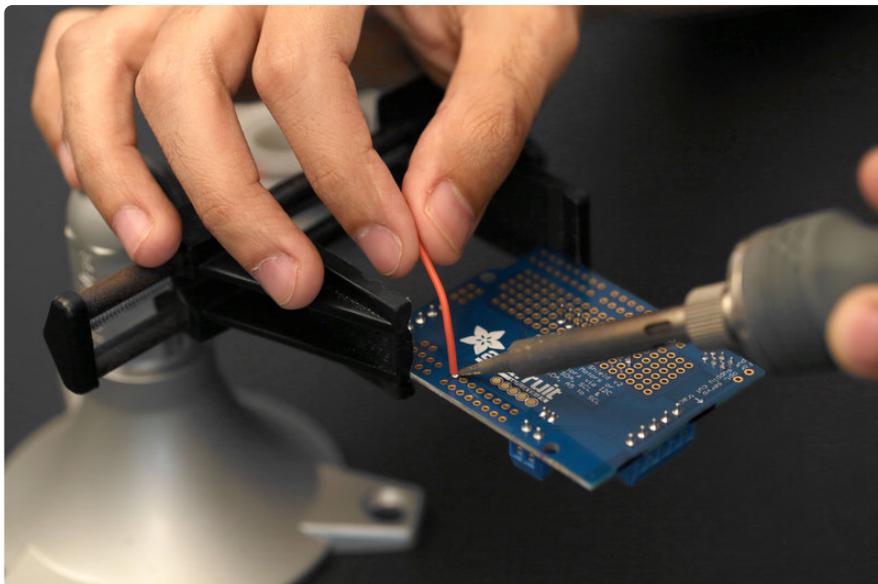
Prep Wires for Motor Shield

Next up we need to prep some wires for connecting the SPI friend to the Motor Shield. The prototyping area on the shield doesn't connect to any of the Arduino pins, so we'll use wires to connect the pins from the SPI friend, to the pins on the Arduino.

Measure the length of the wires using prototyping area as a measuring guide.

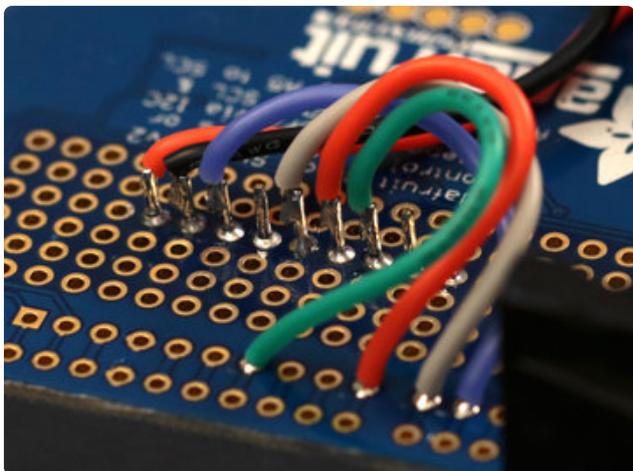
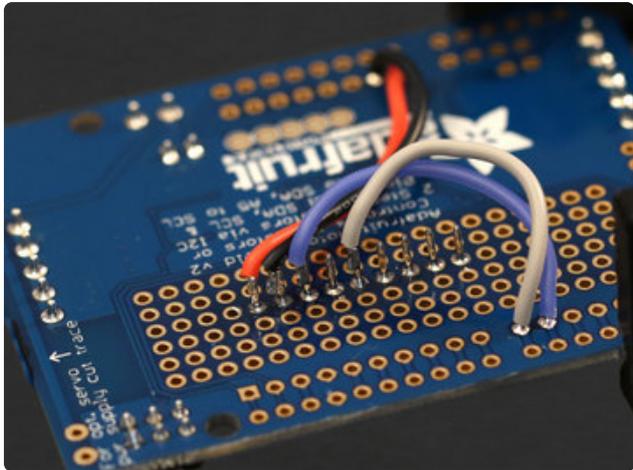
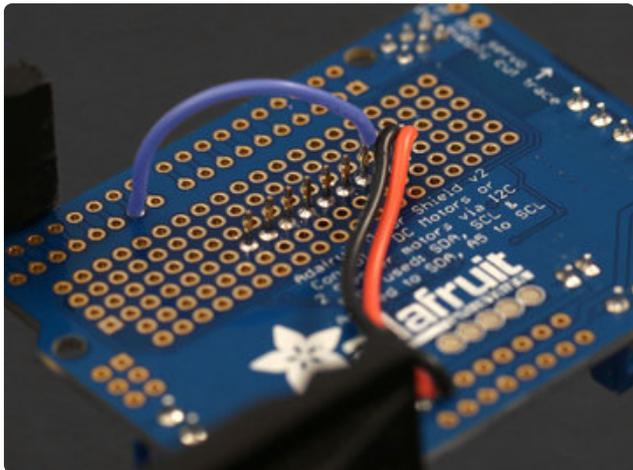
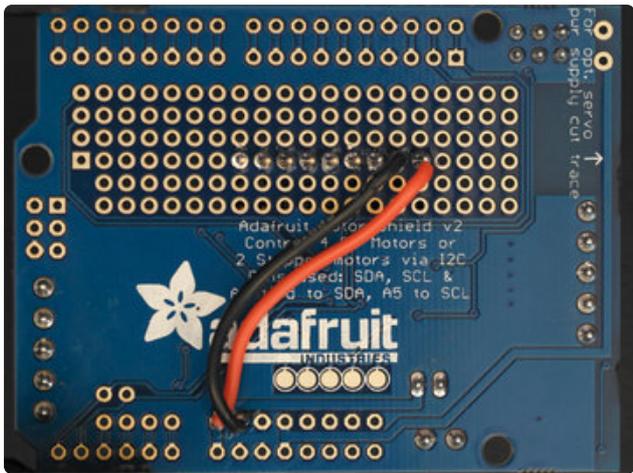
I recommend using 26AWG silicone-coated stranded wire because they're flexible.

Measure and cut and 8 wires in total. Use wire stripper to remove about 5mm of insulation from the tips of each wire.

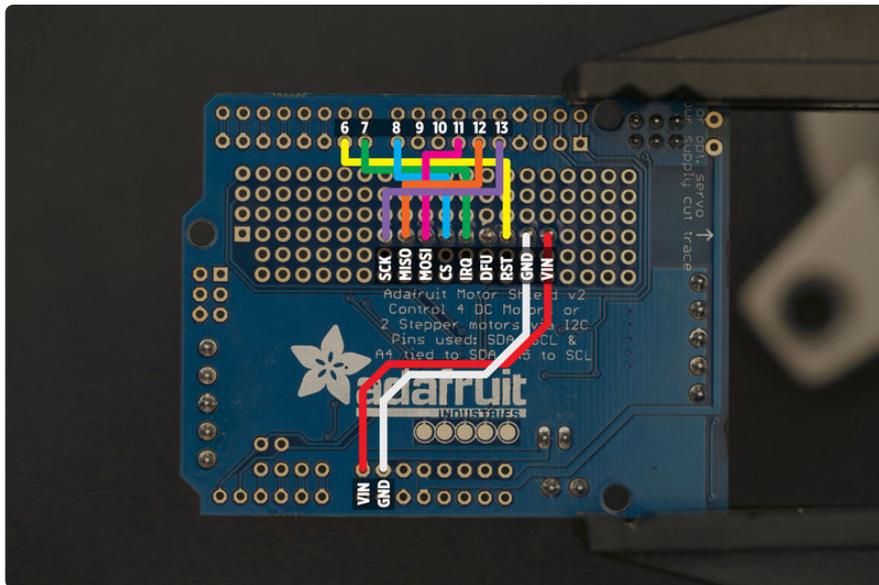


Solder wires into the pins on the Motor Shield

OK, now that we have some wires to work with, let's start by soldering the voltage and ground pins. Locate the VIN and GND pins on the Arduino. We need to connect a wire from those to the VIN and GND pins on the Bluefruit LE SPI friend.



Start soldering by wiring to the ground and voltage in pins to the pins on the motor shield. Tin the header pins on the Bluefruit LE SPI friend and solder the wire from the motor shield to the header pin. Repeat this process for the 8 connections.

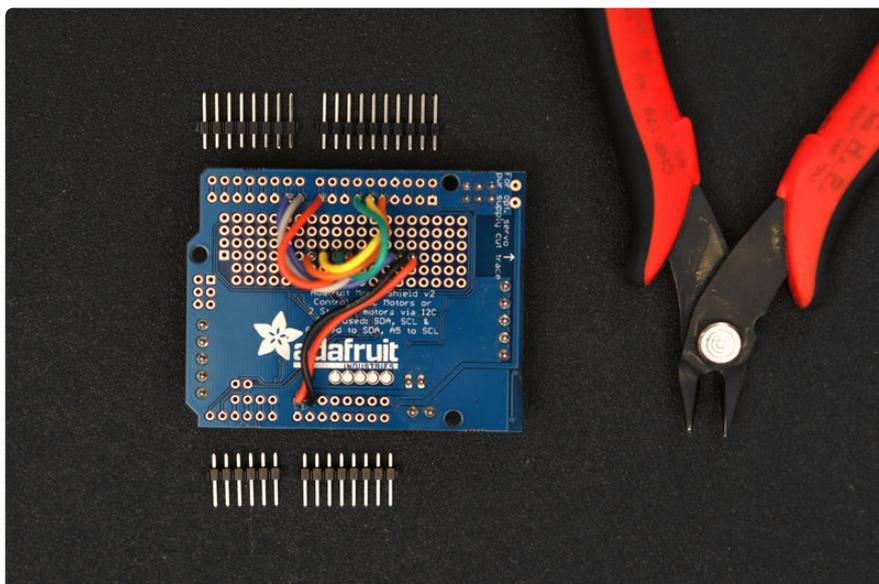


Motor Shield + Bluefruit LE SPI Friend Wiring Diagram

Use the diagram above to reference the wiring connections. Because we're working underneath the PCB's, it's hard to gauge which pin is what. The diagram includes labels and colored lines to represent connections.

The wiring may require a bit of finesse. I'm left handed and found myself in some tight spots where I had to solder the wires in a certain order. Ensure there isn't any cold solder joints and that the wires are properly soldered to the header pins.

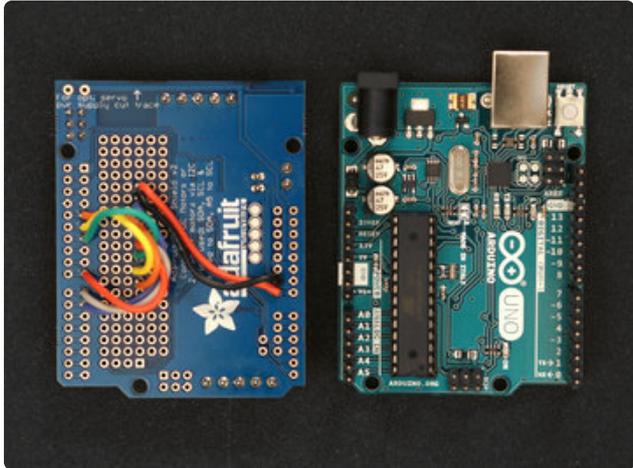
This might be the most difficult part of this whole project. Take your time and have patience.



Measure and cut headers for the Motor Shield

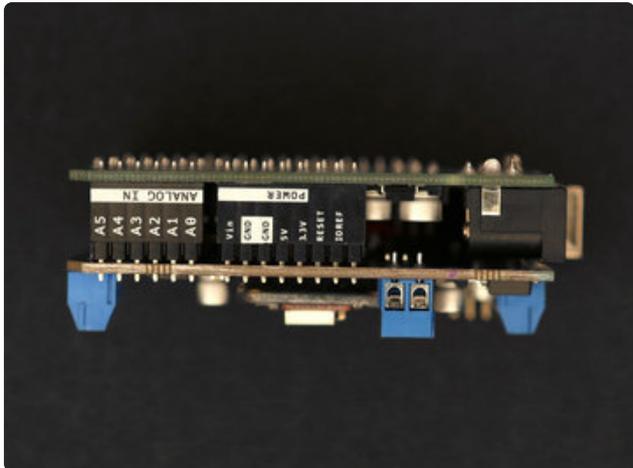
The final wiring looks like a knot, I know - wish they were as clean as the diagram! Anyway, I used 26AWG silicone-coated wires cause it's less likely for these wires to break with the given stress-relief.

Next up grab some headers left over from the SPI friend and cut them down to size so they fit the pins on the Arduino Uno.

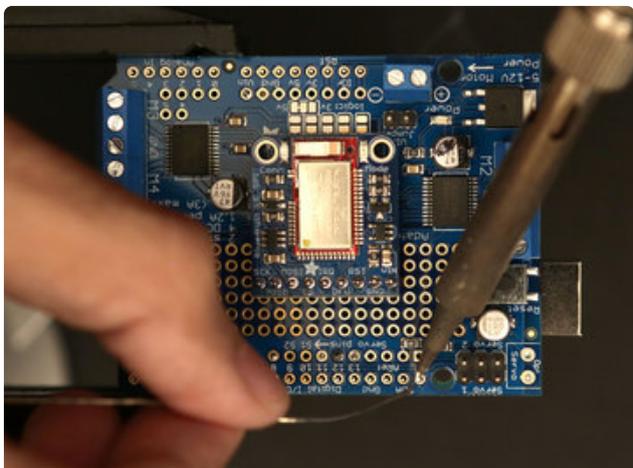


Install headers to Motor Shield

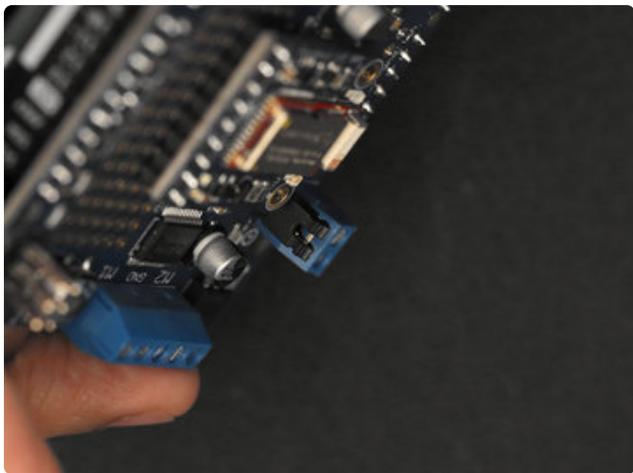
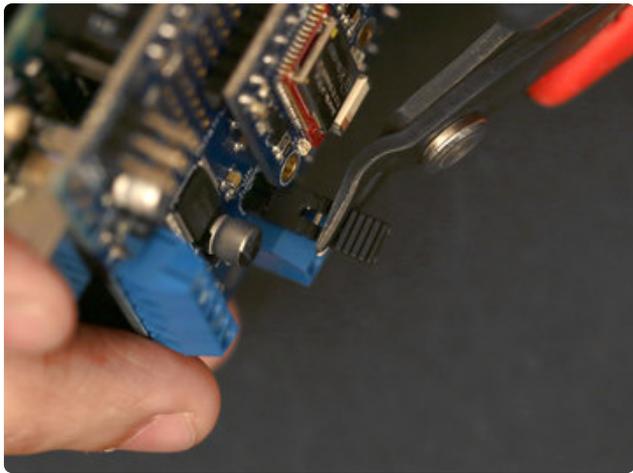
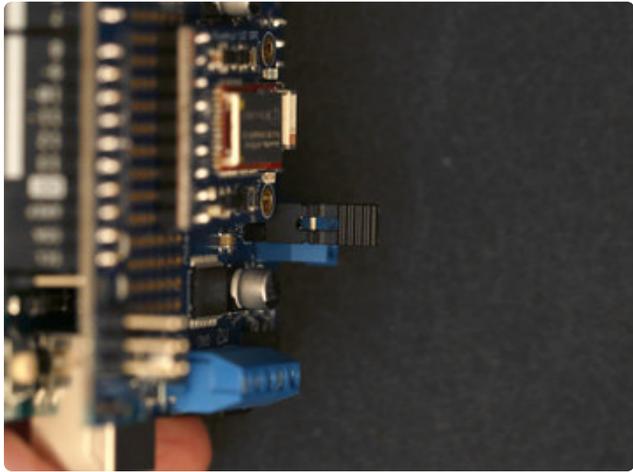
The easiest way I found to solder the headers to the Motor Shield is to insert them onto the Uno first.



Once the headers are seated, lay the Motor Shield PCB over the Arduino Uno and line up the pins so they are seated to the Motor Shield.



With that in place, go ahead and solder them in place. Again, double check and ensure there is no cold solder joints.



Install the VIN Jumper to the Motor Shield

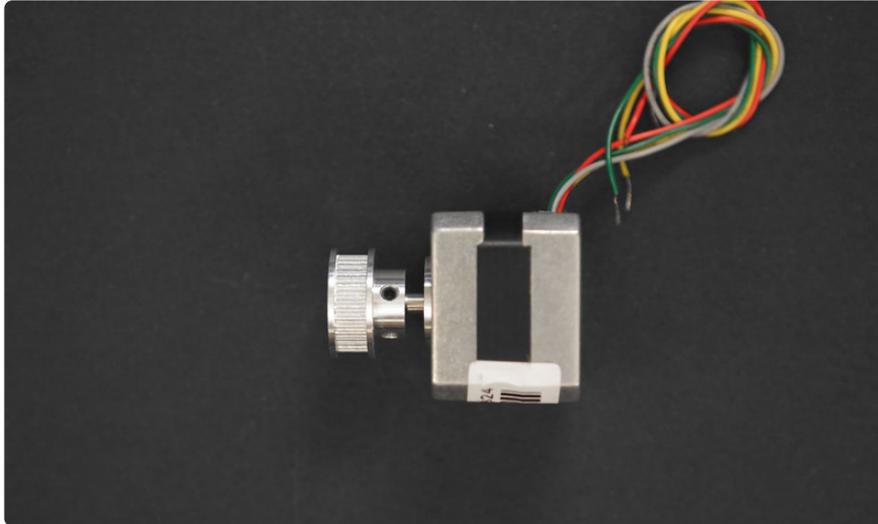
The Motor Shield comes with a little black piece. This is the VIN jumper. **Insert the VIN Jumper** shown as the tall black handle right next to the green Power LED below. Without it, the thing won't power!

It's a little bit too tall to fit inside the enclosure so you can snip it short using wire cuts or scissors.

Wiring Complete

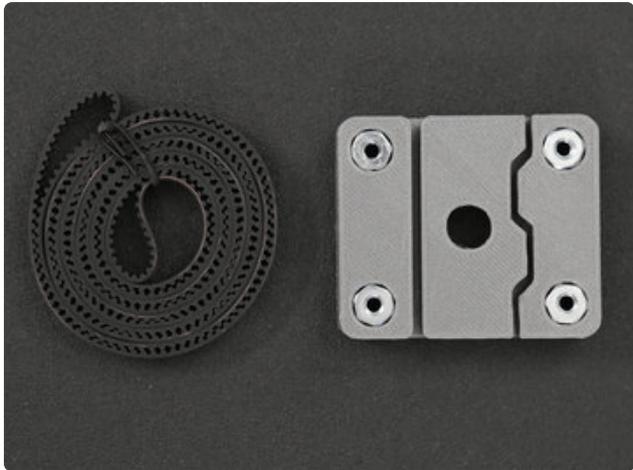
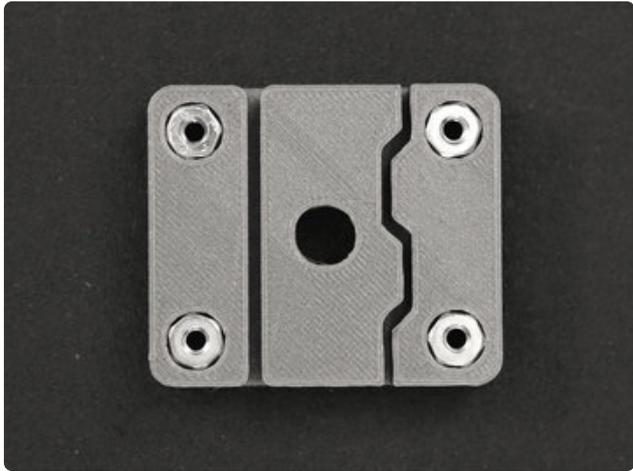
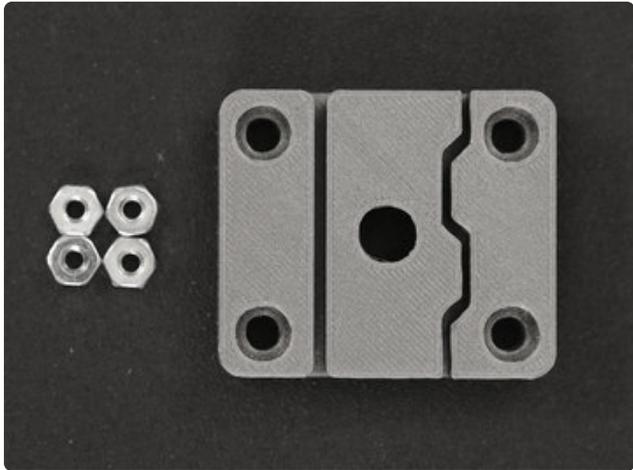
Congrats! The wiring portion of this project is done. Next up, we'll assemble the hardware and 3D printed parts.

Assembly



Install the GT2 Timing Pulley

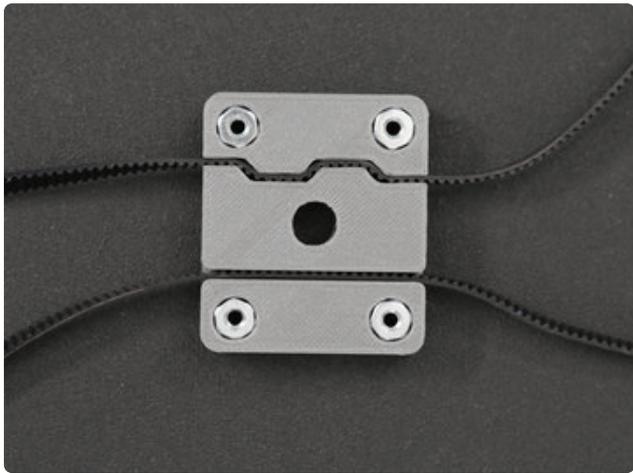
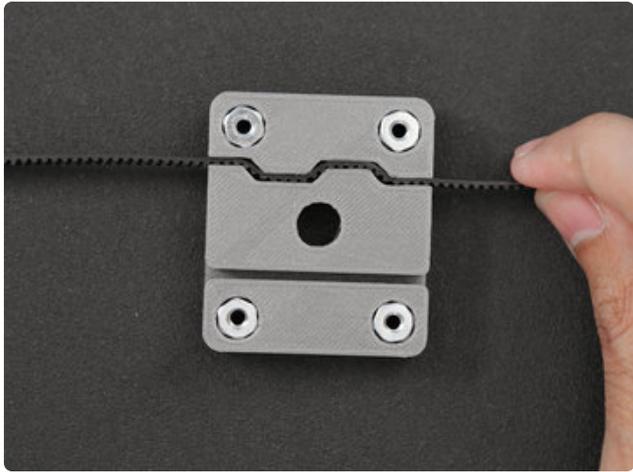
Insert the timing pulley into shaft of the stepper motor with the tightening bore end first. Reference the photo for the position.



Add Hardware to Camera Mount

The side with the counterbores will tightly fit #6-32 hex nuts. The easiest way to get'em in, is to press hex nuts into counterbores by laying the mount upside down top the nuts and using your hand/body weight to push down, ideally on a counter top table.

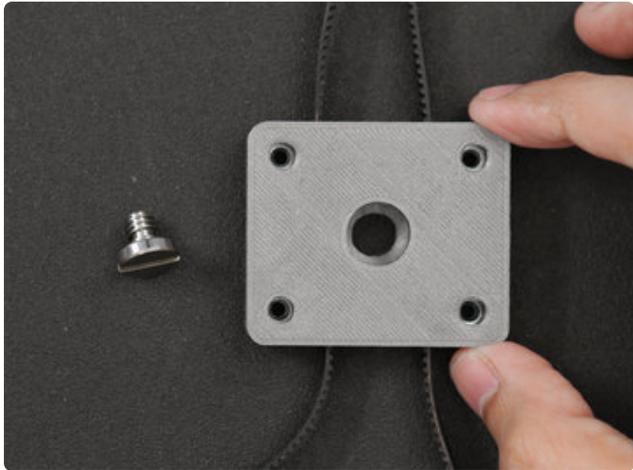
Once they're in, go ahead and grab the timing belt.



Install belt to camera mount

Now it's time to seat the belt onto the zigzag slot on the camera mount. The tension is pretty tight, so you'll need to wedge the belt in, little by little starting on one side. The other slot is the "idler", the tolerance is loose so the belt can pass through. You'll need to keep the belt in place while we work with it.

Take note of the orientation the teeth on the belt are. They should be pointing inwards. The tension slit may be slightly different (simplified) but will still function the same.



Install Tripod Screw and Ball Head

Flip the camera mount over and you'll see the center hole has a counterbore. We need to insert a tripod screw here so the thread is protruding through the other side.



A standard tripod screw is typically 1/4"-20 size but the screw head may vary. The diameter of the opening here is 13.40 mm.

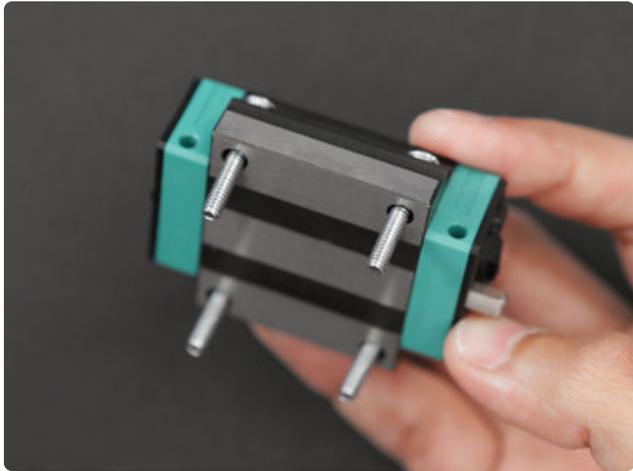
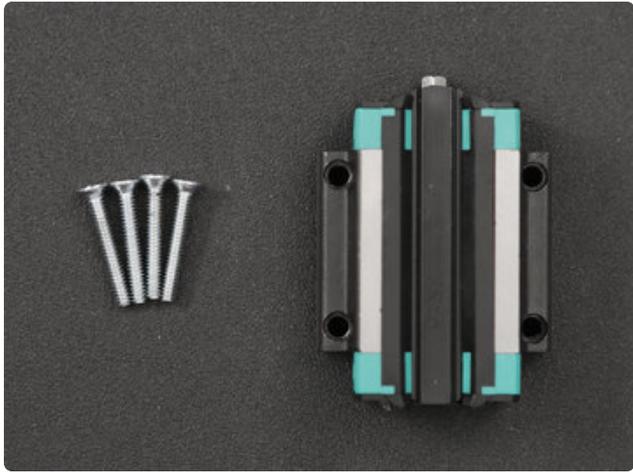
If your tripod screw doesn't fit, you may need to use a filing tool or adjust the model in CAD to widen the bore.



Once it can fit, insert and fasten it to the screw thread on the bottom of the tripod ball head. Tightly fasten them together.

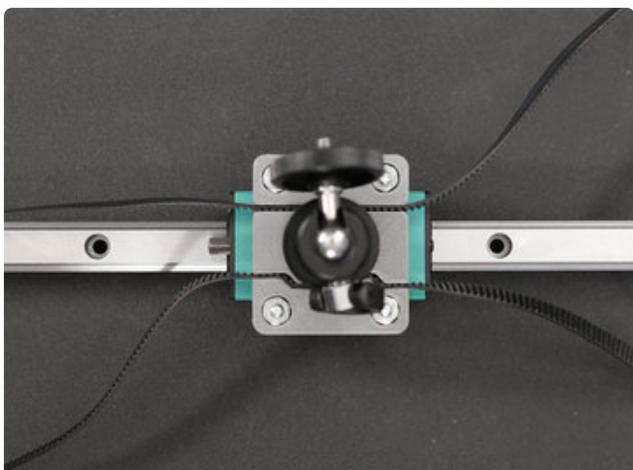
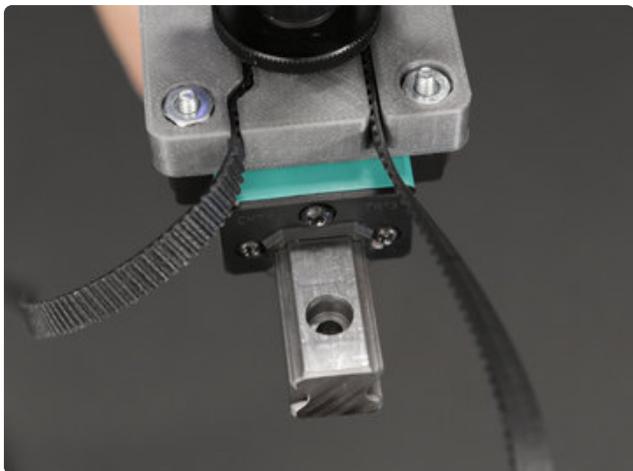
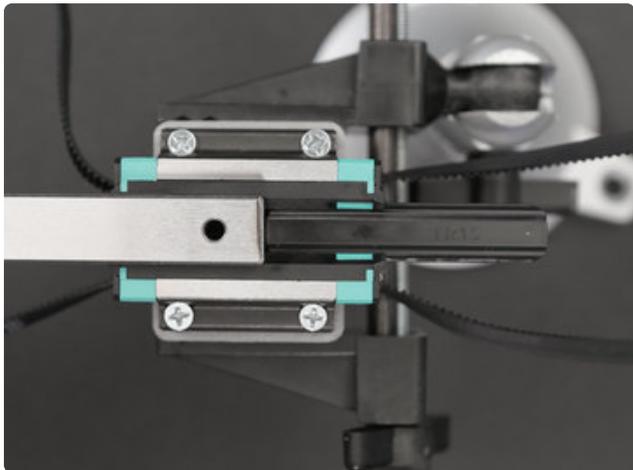
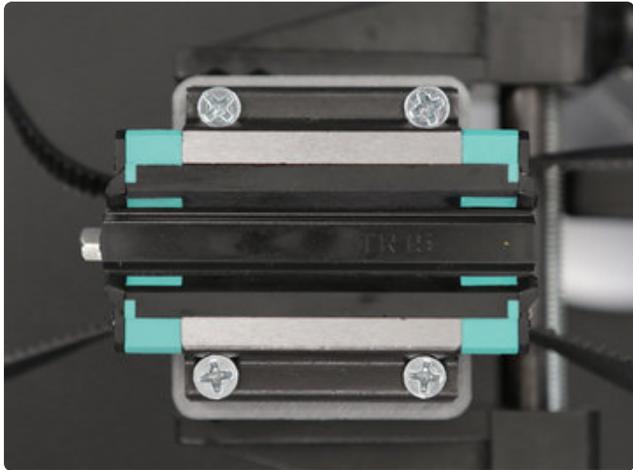
The timing belt needs to be install before tripod mount because this keeps the belt from being unseated.





Install Camera Mount to Railing Platform

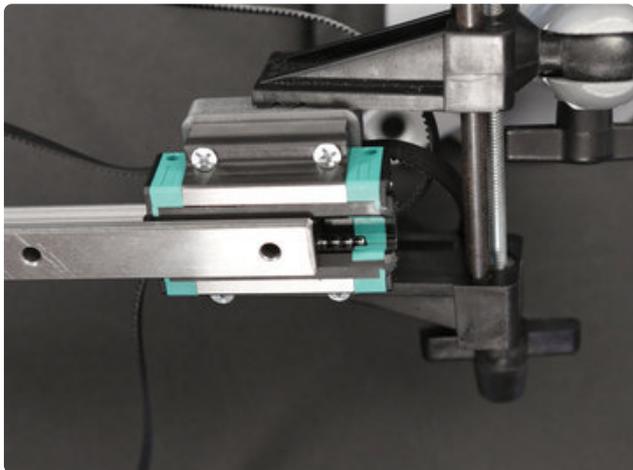
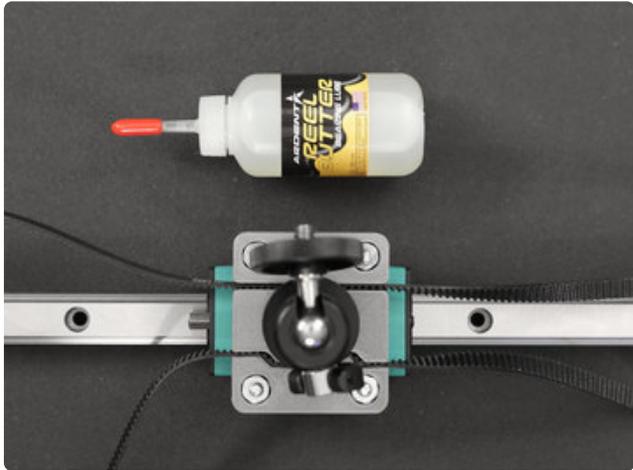
Insert and fasten four #6-32 3/4" screws into mounting holes on the platform. Grab the camera mount and lay that on top of the railing platform. The screw threads should go through the bores. Use a screwdriver to fasten the screws into nuts.



Install railing platform onto support rail

OK, so this part requires you to be super careful. The railing platform comes with a plastic insert. **DO NOT REMOVE IT!!!** This plastic holder actually holds tiny ball bearings in place. The trick here is to insert the rail into the platform by pushing it through the rail, incrementally pushing the plastic holder out. To do this, I secured the platform to a panavise and slowly pushes the rail through the slider, pushing out the plastic bit. Make sure you have it in the proper orientation and do it slowly!

If you remove the plastic holder piece from the platform without the rail in place, tiny ball bearings can fall out!



Add some Lube (Optional)

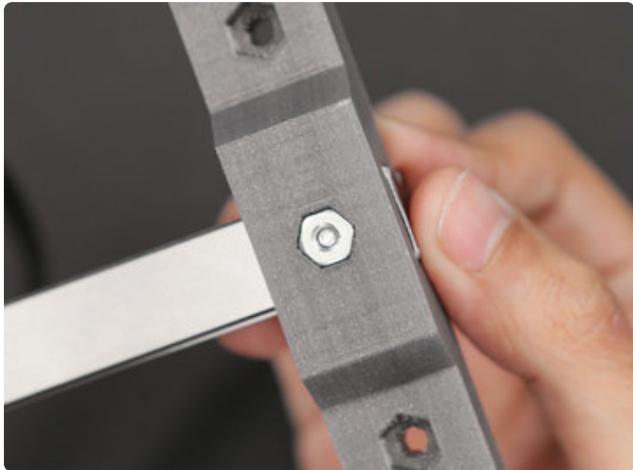
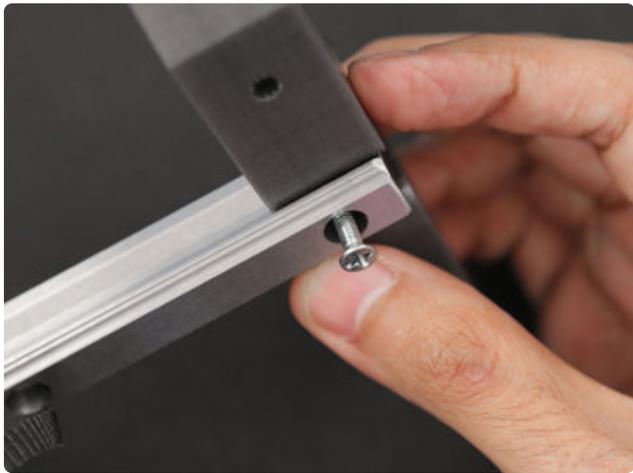
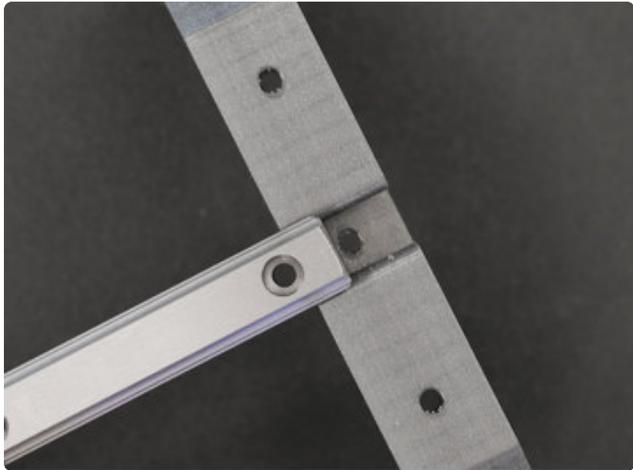
To make the railing platform slide across the support rail buttery smooth, I recommend adding some lubrication to the ball bearings.

Be super care if you decide to do this. When I did this the second time (the build for this tutorial) three little beads came out because the support rail came off too quick.

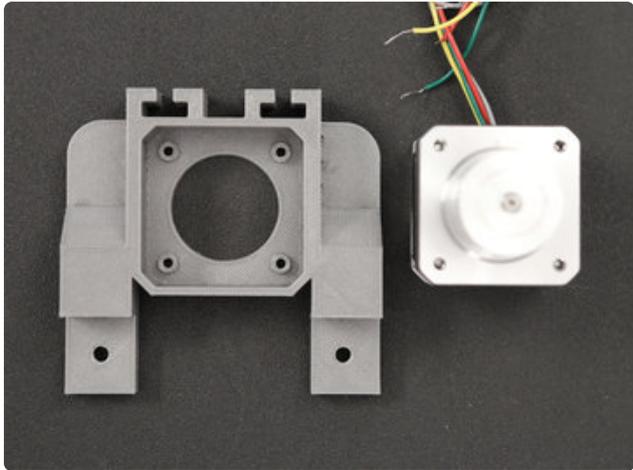
I was able to pop them back in though, but that was super lucky cause I found them on the desk.

Anyway, slowly and carefully pull the rail out of the platform, revealing only a small portion of the bearings. There's actually two rows of bearings on each side - it's difficult to see and I couldn't get any good photos of it but you'll see them. You only need to add one drop of Reel Better (synthetic oil for skate bearings) to each row. Once one side is lubed, carefully slide the platform the other way and lube the other side. Again, being super carefuls.

Danger! I felt pretty bummed when those tiny ball bearings popped out. Don't let that happen to you!

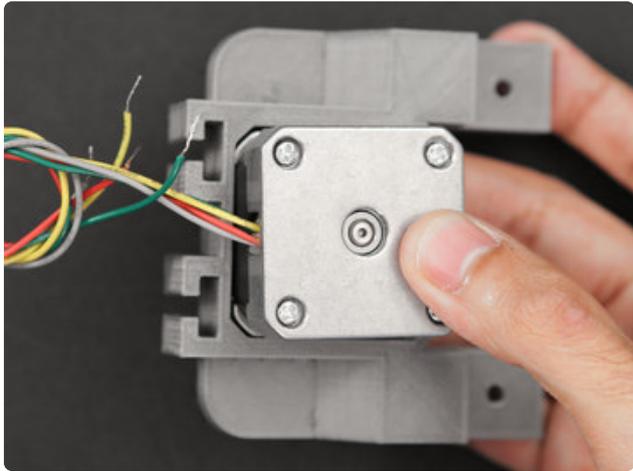


Install feet onto support rail
The feet should freely slide onto the support rail. Line up the mounting holes and insert a #6-32 3/8" screw through. Insert a hex screw to the bottom of the foot and fasten tightly. The hex nut should be seated in the counterbore. Go ahead and repeat this for the other side.

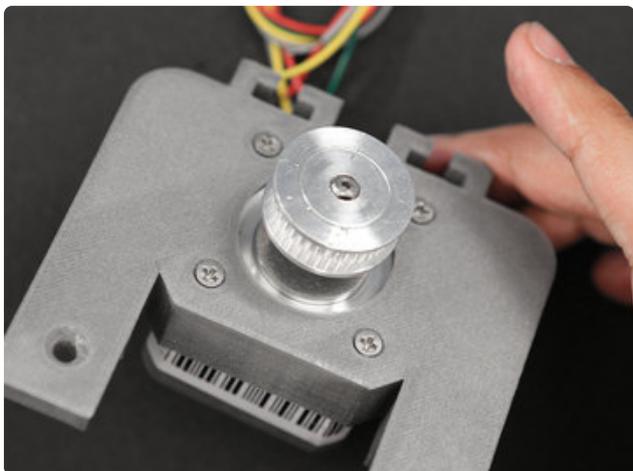
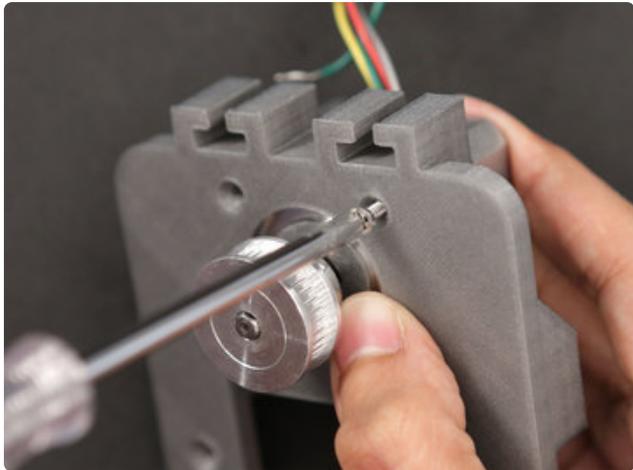


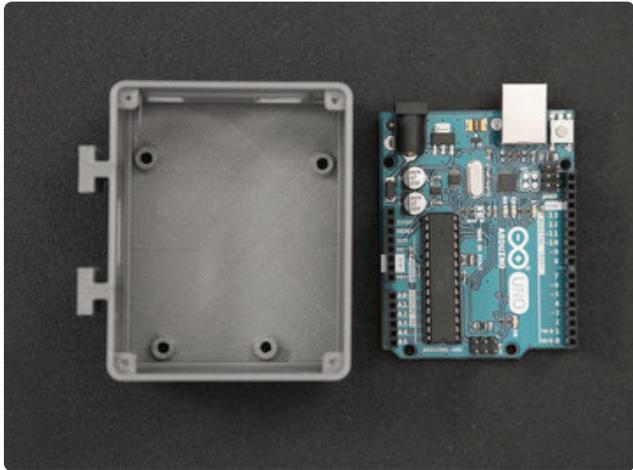
Install stepper motor to mount

Insert the stepper motor into the motor mount with the timing pulley going through the hole in the center. The stepper should be oriented with the wires facing the clips on the mount.



Insert four #4-40 3/8 screws into the mounting holes and fasten until they're flush.





Install Arduino Uno into Enclosure

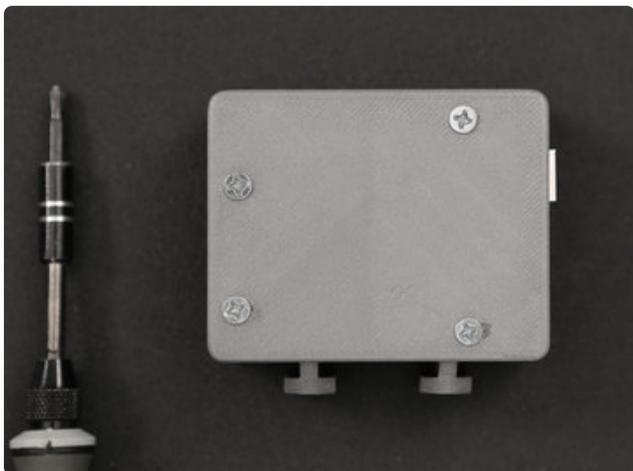
Next up we need to mount the Uno to the enclosure box. First, insert the PCB at angle with ports going in port into cutouts. This requires a bit of finesse. The stand-offs on the corners may get in the way, so you'll need to wiggle the PCB until it pops into place.

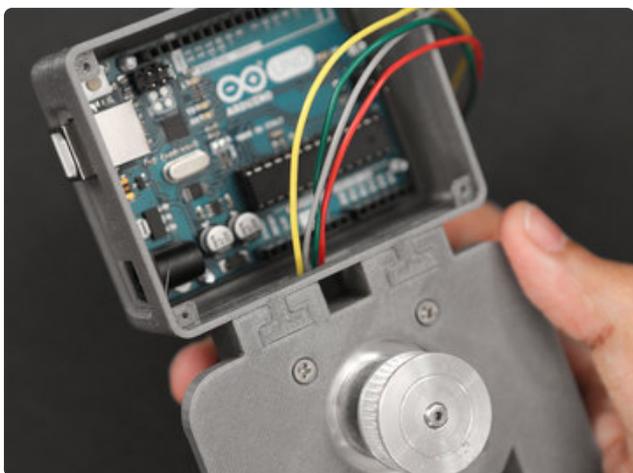
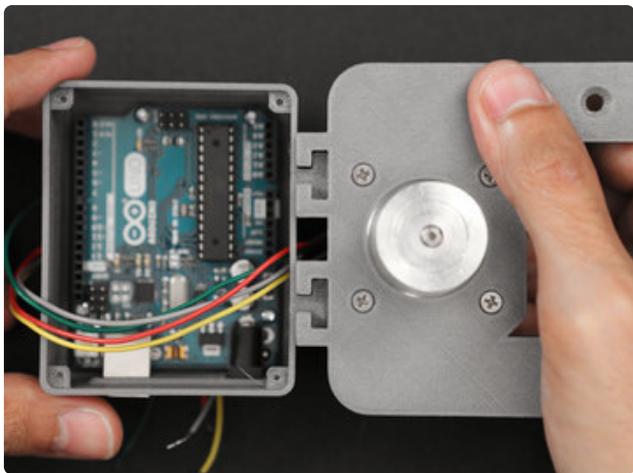
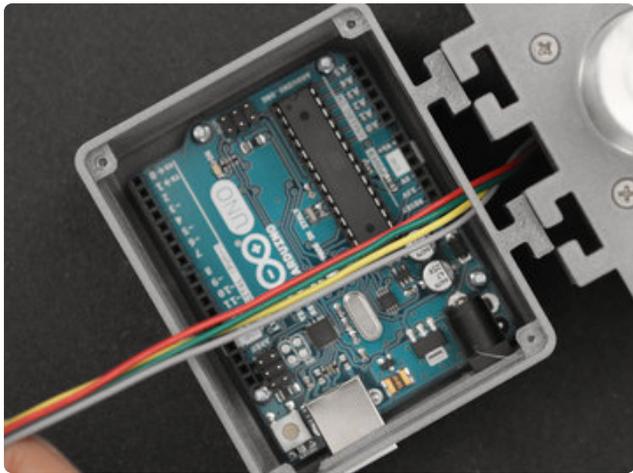
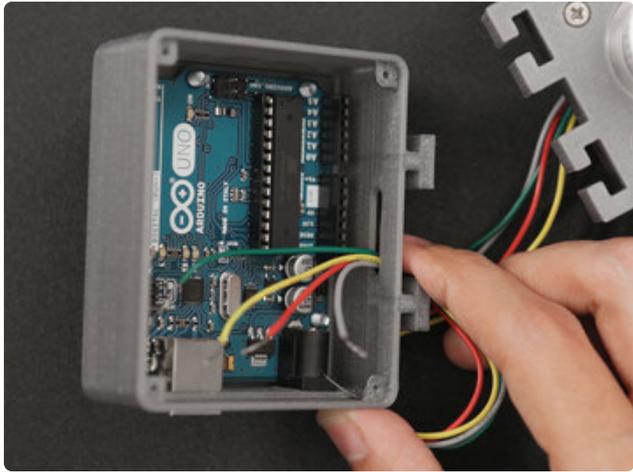


Once you get it in there, hold the PCB down in place with the mounting holes line up and insert four #6-32 3/8" screws. Fasten the screws tightly.



Rather than using nuts, these #6-32 screws "bite" into the PCB holes instead. This is normal.

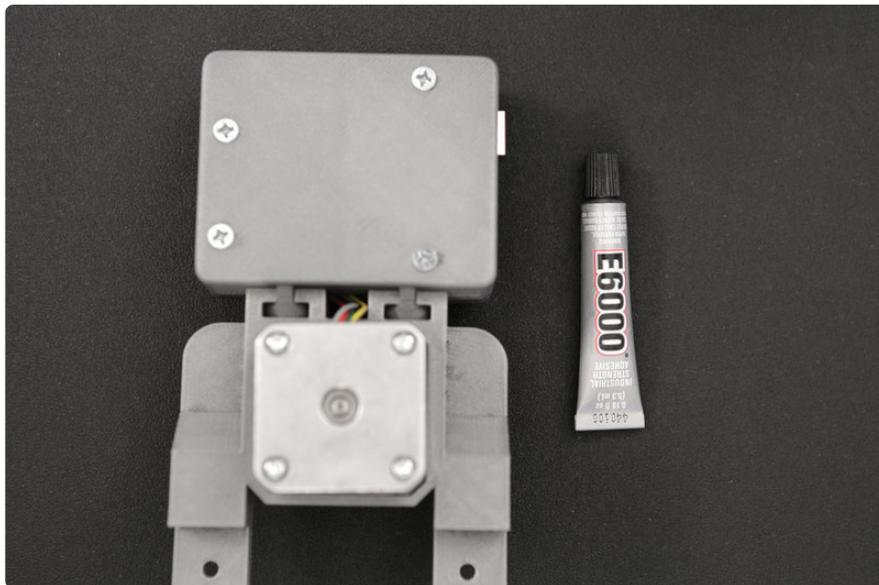




Install Stepper Wires

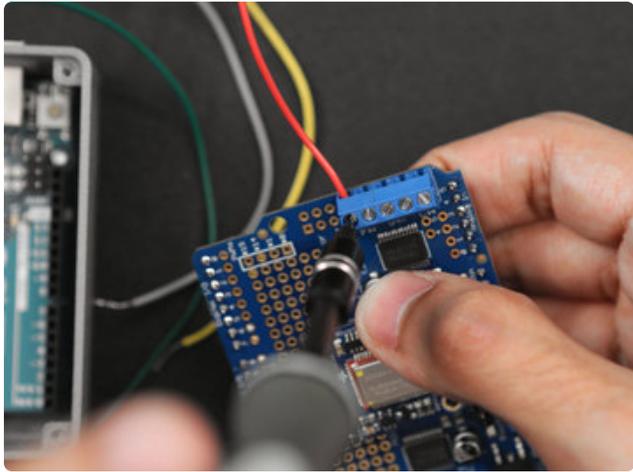
OK, grab the stepper motor mount and position it closely to the enclosure box. Thread wires from stepper motor into cut out on enclosure. Pull them all the way through.

Now we can install the enclosure to the motor mount. Insert "T" shapes from enclosure to clips on stepper mount, being aware of the stepper wires - avoid clipping these while joining the parts!



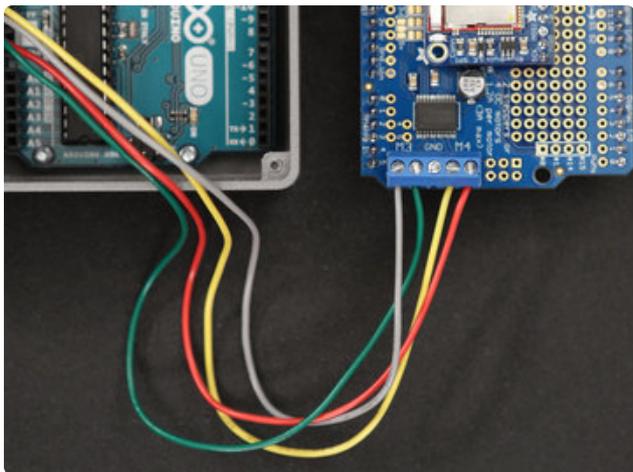
Secure Motor mount to enclosure

Depeneding on how loose or tight the tolerances are, you may need to apply some adhesives to permanently attach the parts together. I recommend applying directly to the bottom of the clips and sliding them in.

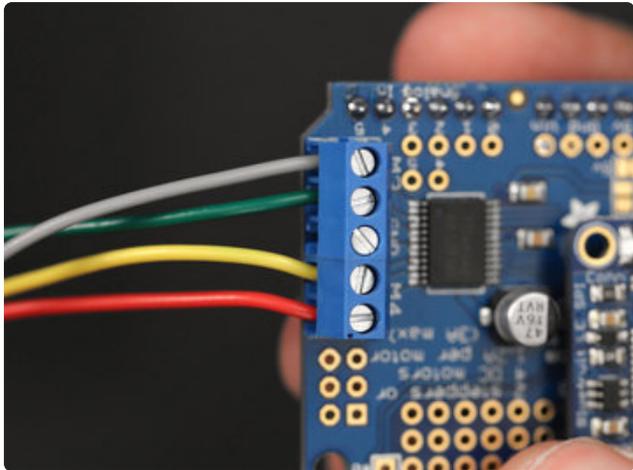


Install Stepper Wires to Motor Shield

The Motor Shield has terminals on the motor inputs so it's easy to insert them and tighten the screws to secure them in place.

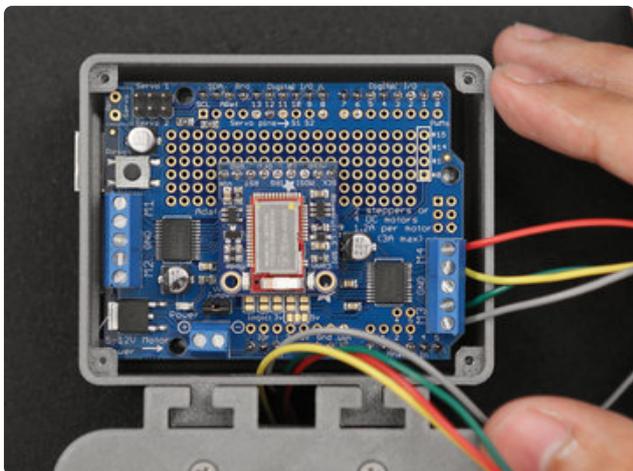


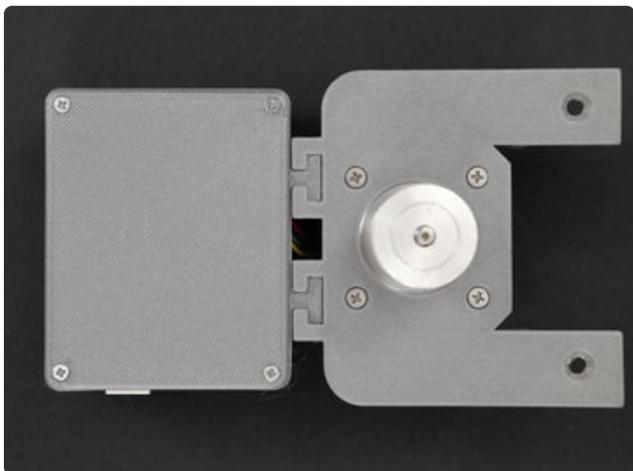
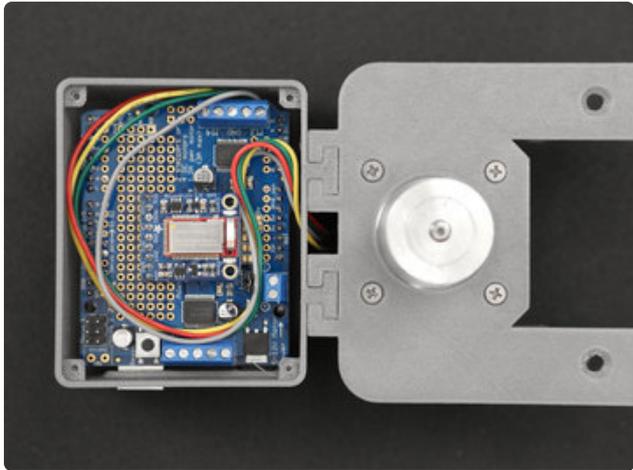
Insert the wires from stepper motor to motor input M3 and M4. Follow the photo to reference order of wires using color as an indicator.



Next, insert the motor shield into enclosure with pins lined up with Arduino Uno.

Firmly press PCB down to install shield onto uno.



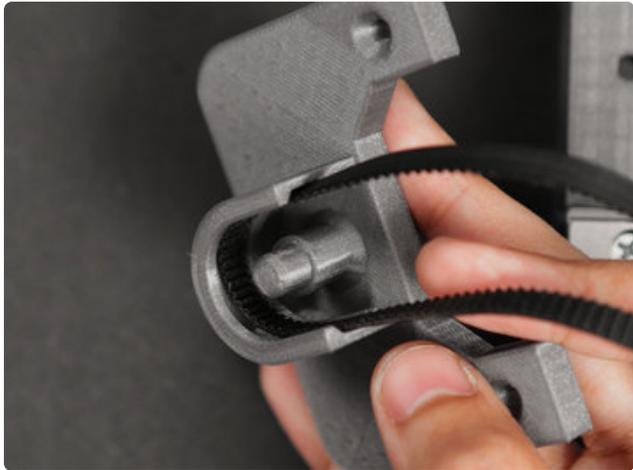


Close enclosure

Fit stepper wires into enclosure so it's nicely packed inside.

Next, let's insert and fasten four #2-56 3/8" screws into enclosure cover. Then, lay the cover on top of the enclosure and fasten the screws tightly, until they're flush.

Now our motor and electronics portion is complete!



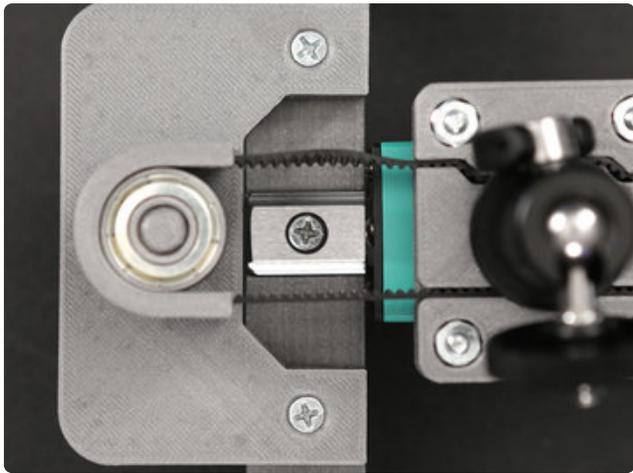
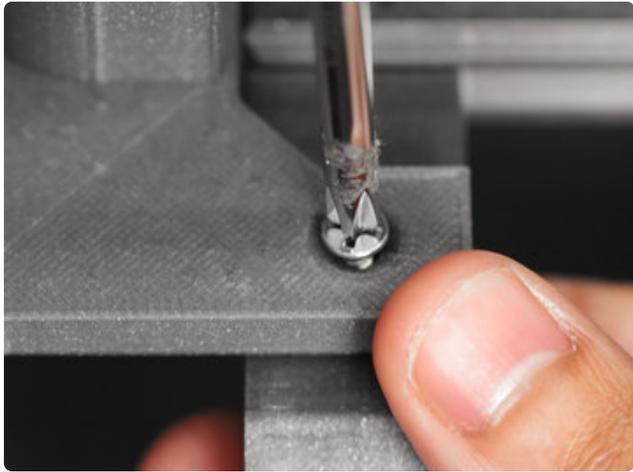
Install bearing and belt to pulley mount

OK, so before we can mount the bearing to the part, we need to insert the timing belt. Place the belt over the shaft and position it so the belt is up against the wall of the pulley guard. There's two lips that should keep the belt in line with the bearing. Once the belt is positioned, place the bearing over the shaft so it fits in. Firmly press it down until it snaps into place.

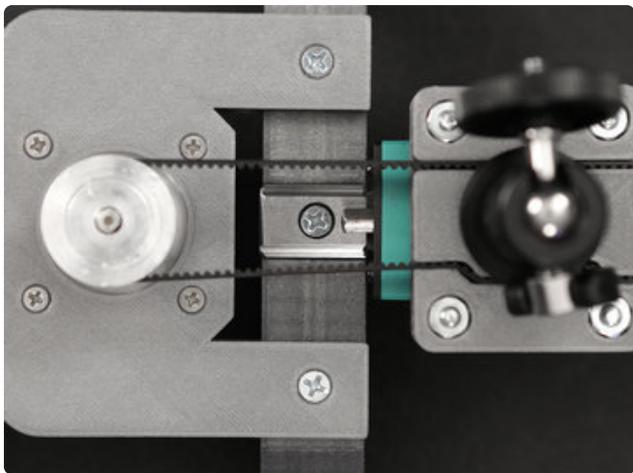
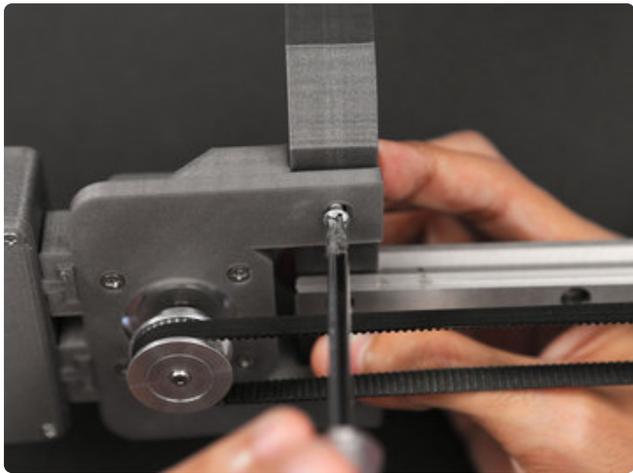
With the bearing and belt in place, you should be able to free pull on the belt to rotate the bearing. The tension should be not too tight or loose.

If it's too tight, you may need to clean the inner wall of the bearing guard. There could be little bits of plastic left over from retraction.

If the timing belt gets dislodged while inserting the bearing, remove the bearing from the shaft using a vise grip and try again.



Install Pulley Mount to foot
Insert two hex nuts on the bottom of the foot. Lay the mount over the foot and line up the mounting holes. Insert two #6-32 3/4" screws and fasten them tightly.



Install belt to timing pulley on stepper motor

Now we need to install the belt onto toothed timing pulley that's on the stepper motor. Once it's in place, play the mount over the foot and line up the mounting holes. Just like before, insert two #6-32 3/4" screws and fasten tightly.

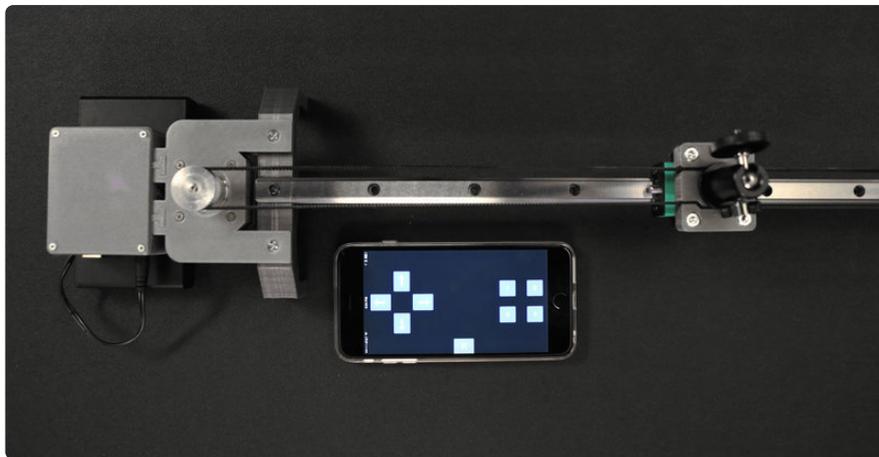
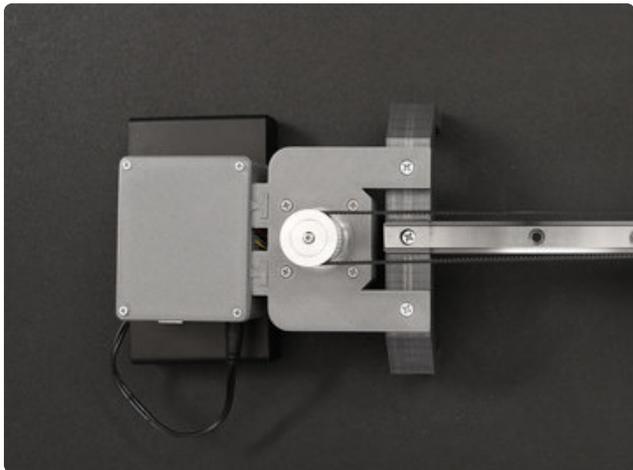


Load Power Pack

Install 8 x AA batteries into battery holder. I recommend using rechargeable ones. The battery holder features a standard barrel jack and plugs into the Arduino Uno.

It's a good idea to attach the battery pack to the enclosure. I didn't design a mounting bracket or anything special to do this because you may want a different battery pack or power source. I simply used double stick foam-tape but you could use velcro straps, zip ties or even adhesives to attach the battery box to the enclosure. You can acct it to either the bottom or top of the enclosure.

The 8 x AA battery holder also features a power switch!



Connect and Control

Congratulations, the build is complete!! All that's left is to do is test the connection and get some epic timelapses!

Download and launch the Adfruit bluefruit LE connect mobile on your device. Power on the Arduino. The Bluefruit LE module should appear in the list of devices. Tap on the connect button and then the control pad.

Now you can press the button to move the slider! Check the next page for usage details.

Usage

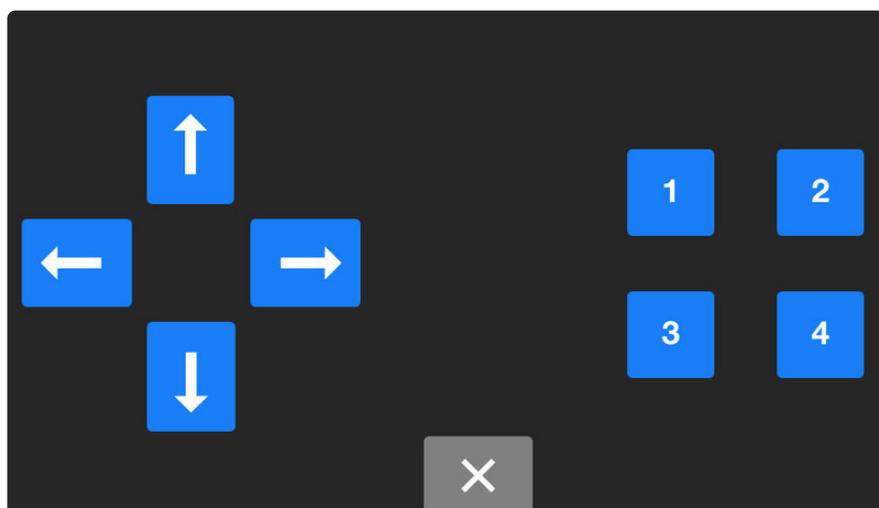
Adafruit Bluefruit LE Connect Mobile App

You'll need to download the mobile to control the movement of the slider. Follow the steps below.

- Get the Adafruit Bluefruit LE Connect app
 - [Download for iOS \(https://adafru.it/ddu\)](https://adafru.it/ddu)
 - [Download for Android \(https://adafru.it/f4G\)](https://adafru.it/f4G)
- Under Peripherals, select connect on **Adafruit Bluefruit LE**
- Select **Controller**
- Under Module, select **Control Pad**

Control Features

A few different time periods are supported, but not a whole lot. Since there's no display connected, we just blink the onboard LED to indicate which setting is active, and there's only so many speed changes one can reliably 'read' this way. Feel free to edit, but keep in mind there are upper and lower limits to the time interval -- the stepper can only move so fast and if you try to press beyond that it'll just move linearly regardless of the selected interpolation mode, and the micros() function rolls over about every 70 minutes, so the duration can't exceed that. Blink rate is similarly constrained due to BLE comm; about 2 Hz max.



Up	Faster Speed	Increases the speed of the movement
Down	Slower Speed	Slows the speed of the movement.
Left	Home	Retracts movement to starting position
Right	Start	Begins the movement
1	Linear	Moves at a constant rate
2	Ease In and Out	Starts slow, ramps up speed and then ends slow
3	Ease In	Starts off slow, ramps up and ends with ramped speed
4	Ease Out	Starts off with constant speed and slowly ends

What Cameras should I use?

The slider can push about 3-4 pounds of weight. I've tested it with a manfrotto fluid head and a Canon 5D with a 24-70mm lens. Ideally, we made this for a camera phone or GoPro, but it can push a full frame DSLR. It's really up to you. Point and shoot, camcorders and webcams are all fine so as long as you can tightly mount it to the platform. Whatever you choose, be sure things are really tight to avoid any shakiness.

Perspective, Angles and Composition

I recommend avoiding wide shots and landscapes. Only because it start's looking like nothing is even moving. Since the length of the rail is short, you dont get much sliding distance. So try getting elements in the foreground to get the illusion of sliding a bigger distance. You can get pretty low to the ground or set it on a table. Get creative

and prop it up at an angle. A tripod ball head can help position your camera in interesting angles.

Optimal Slide Direction

The slider performs the smoothest movement when the timing pulley on the stepper pulls the slider to it. Ideally, the platform should be homed when its closest to the radial ball bearing. When the platform slides across in the other direction, the timing pulley pushes the belt, which introduces a slight shake. The shake is noticeable in our tests, so we recommend sliding one way.

Timelapse / Video

The stepper motor performs the smoothest when it's set to run in microsteps. This makes the speed very slow, but gives very smooth movement. Ideally, this is great for capturing timelapses. However, the speed of the stepper motor can be adjusted (double steps) but that introduces shake to the movement (with light-weight cameras).

You can however use double steps if your camera is ~3-4lbs. A heavy DSLR body with a lens does not capture any shake, so you can capture desirable live video.