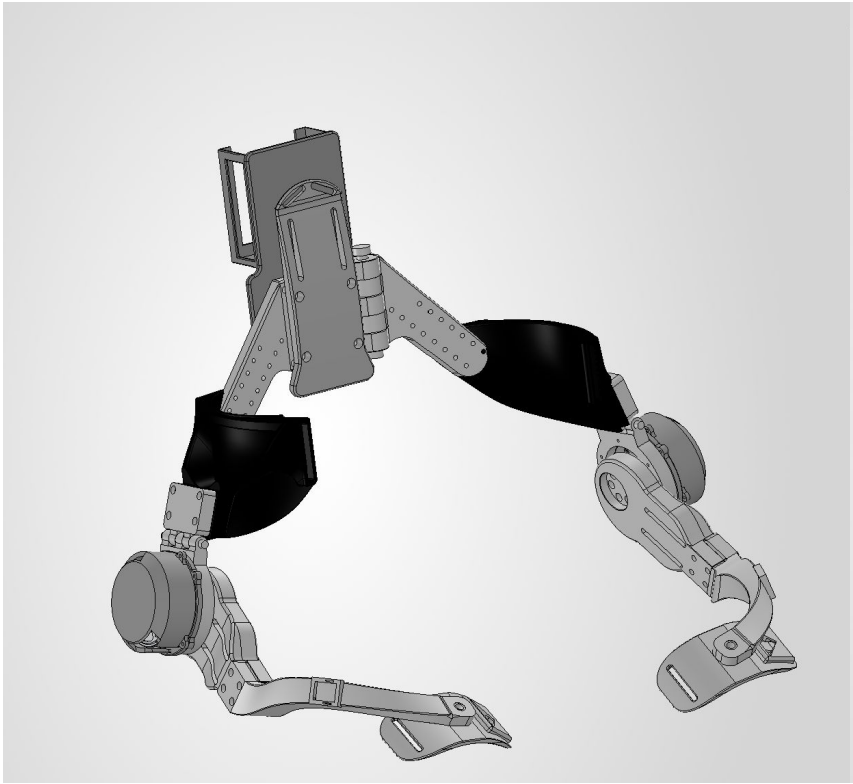
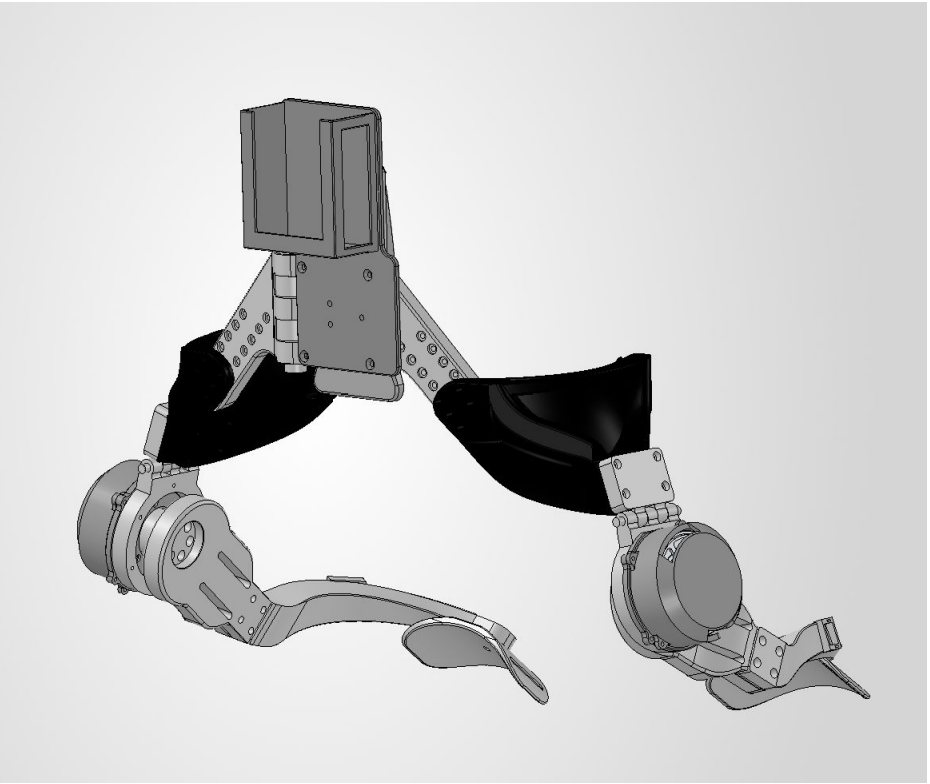
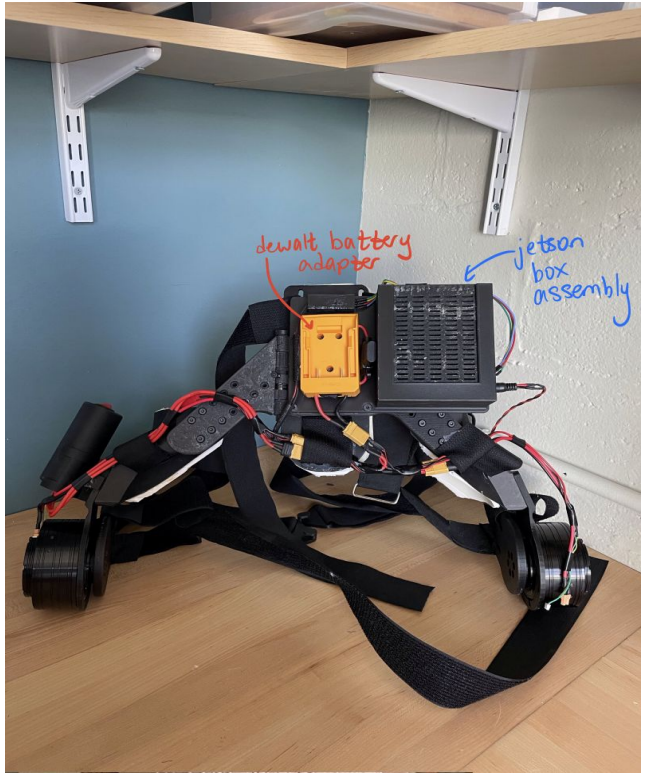
Mechanical Parts Updated Assembly





3D Printed Assembled Parts

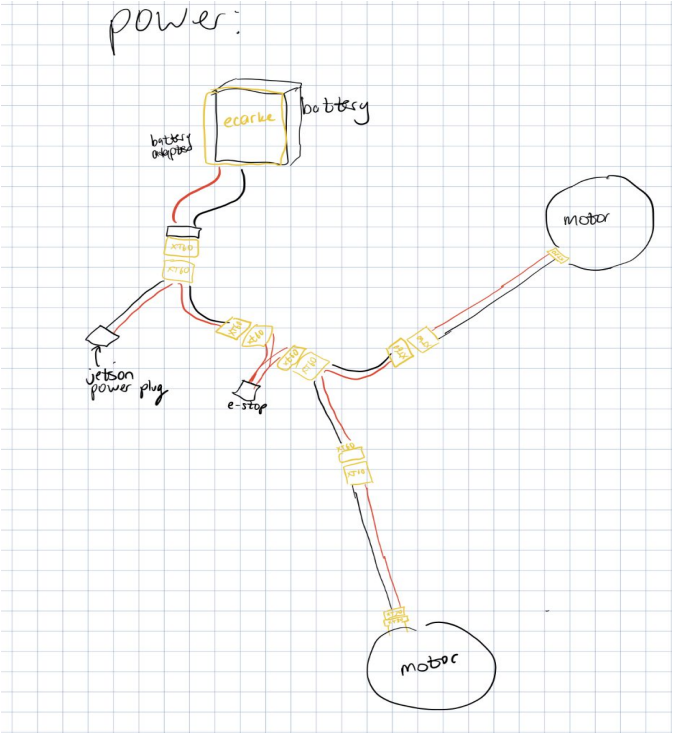


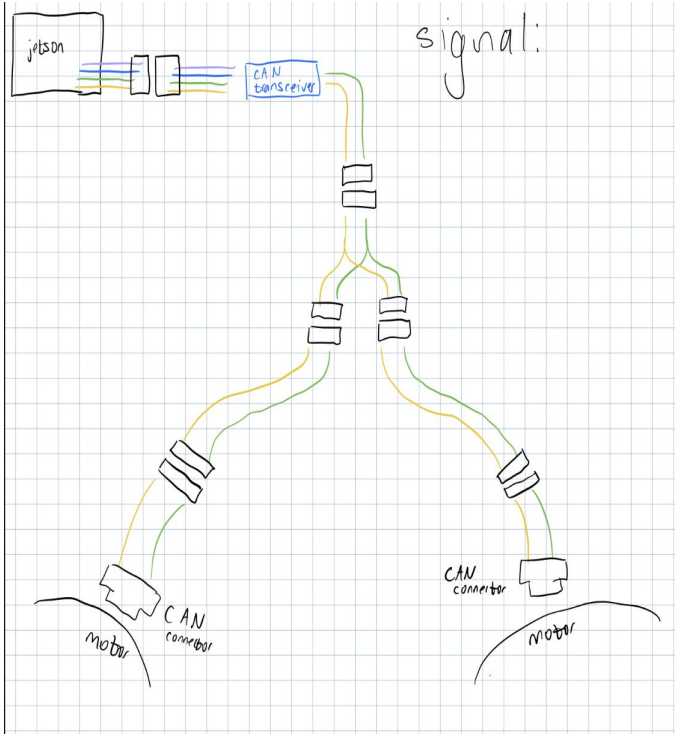


Full system worn on a user



Power & Signal Connection Sketch



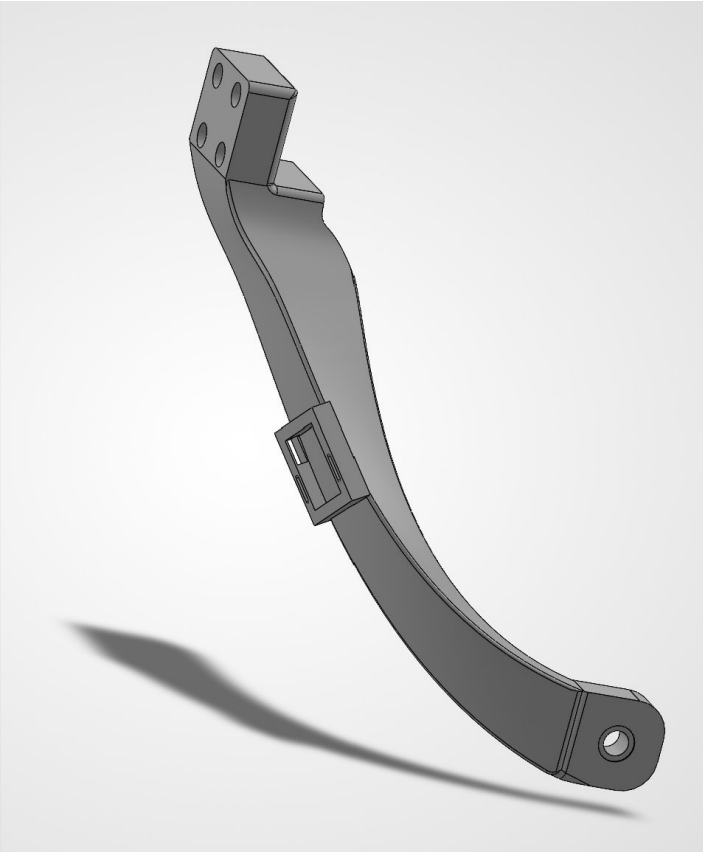


Curved Thigh Strut

- IMU casing
- Thickened variation (15 → 47 mm width)
- Deeper connecting screw holes



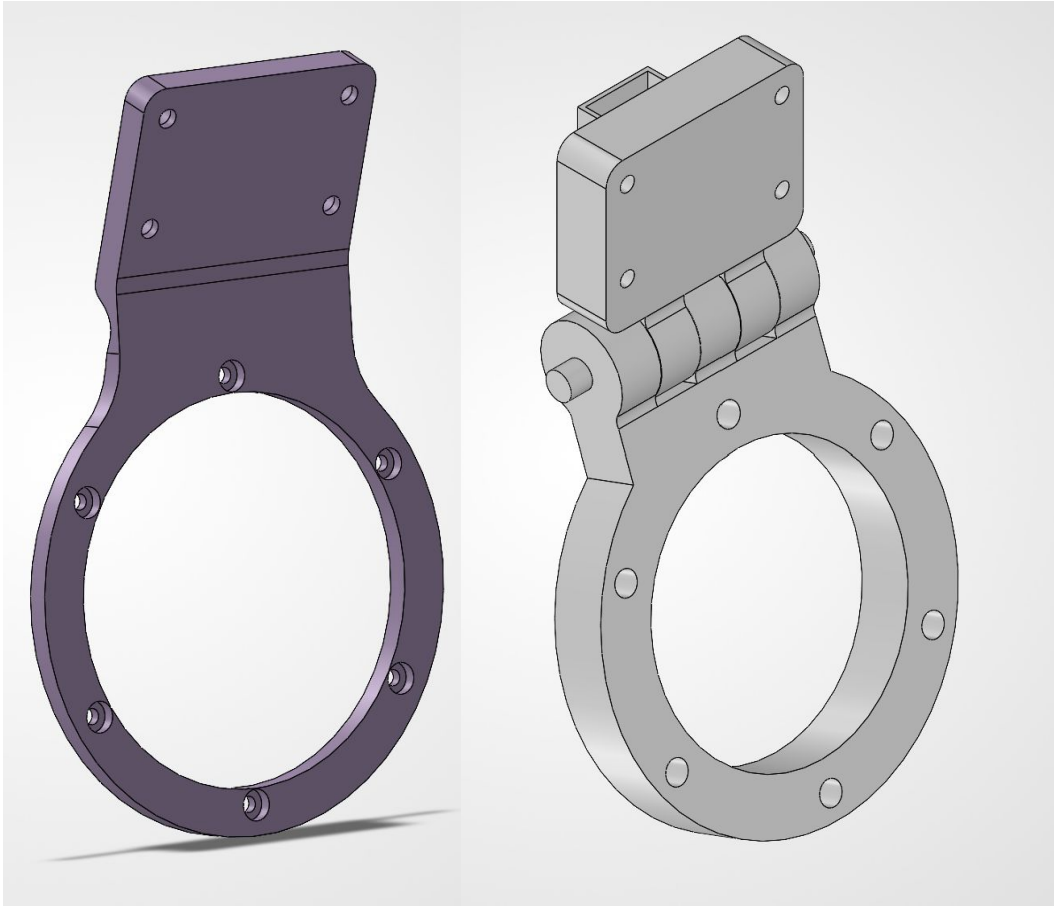
Original



Current
(at 50% infill)

Angled Motor Connector

- Thickened body (7 → 15 mm
- CAN casing
- 8 → 6 motor screw holes (all holes deeper)
- Hinge design

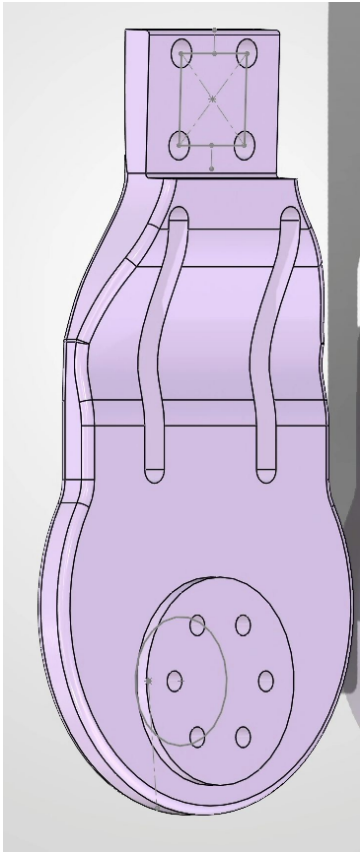


Original

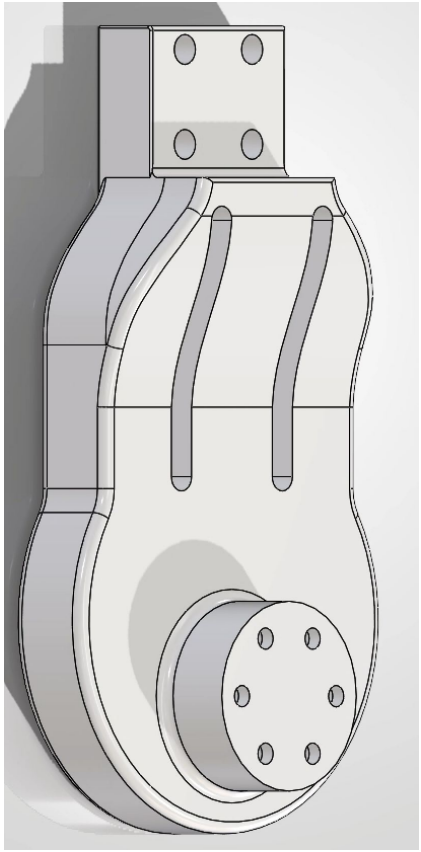
Current
(at 25% infill)

Flat Connector

- Thickened body 10 to 20 mm
- Increased the height of the circular connector attached to the motor from 3 mm to 17 mm
- Created an extruded-cut cavity behind the piece to provide easier access for installing the motor screws
- Changed the hole type from a flat-head screw hole to a countersunk screw hole.

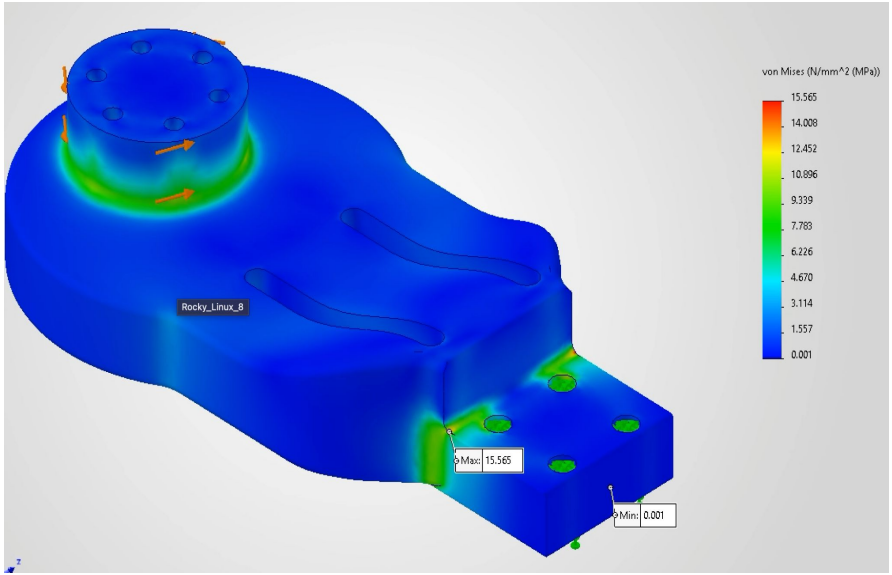


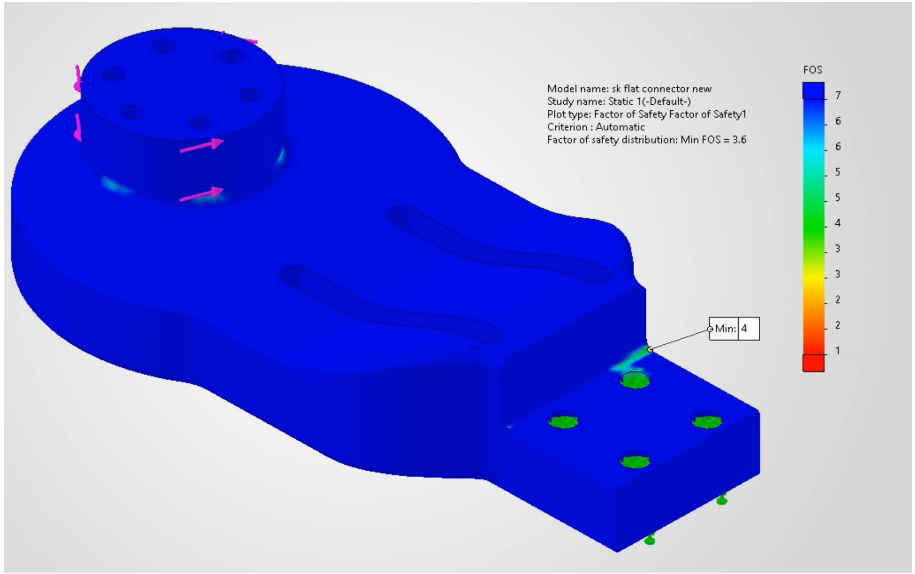
Original



Current
(at 50% infill)

FEA Simulation for Flat Connector





- Material used PETG
- Print Settings(Prusa): 0.20 mm layer height, 50% infill density, and gyroid infill pattern
- Inserted M4 Heat Sink Threaded inserts to mate with the curved thigh strut with M4x14 screws

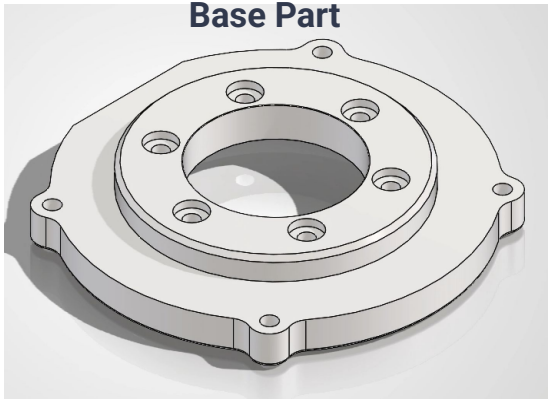


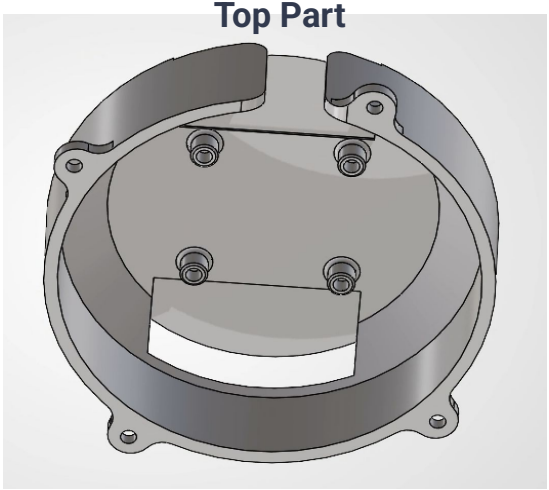
Older Version

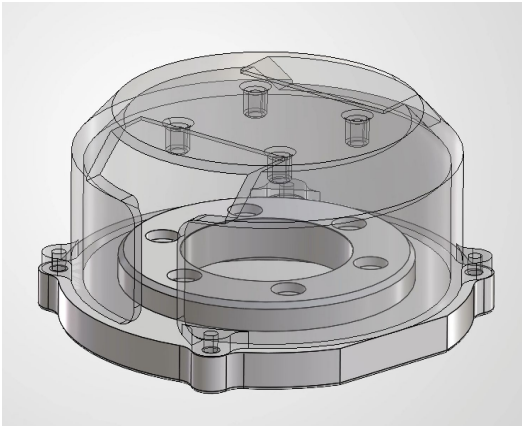


Latest Version

- Solidworks motor encasement assembly
- The motor rests on the base part and is secured using M3 counterbore screws.
- The top enclosure is then fastened to the base using M2 countersunk screws and threaded inserts.

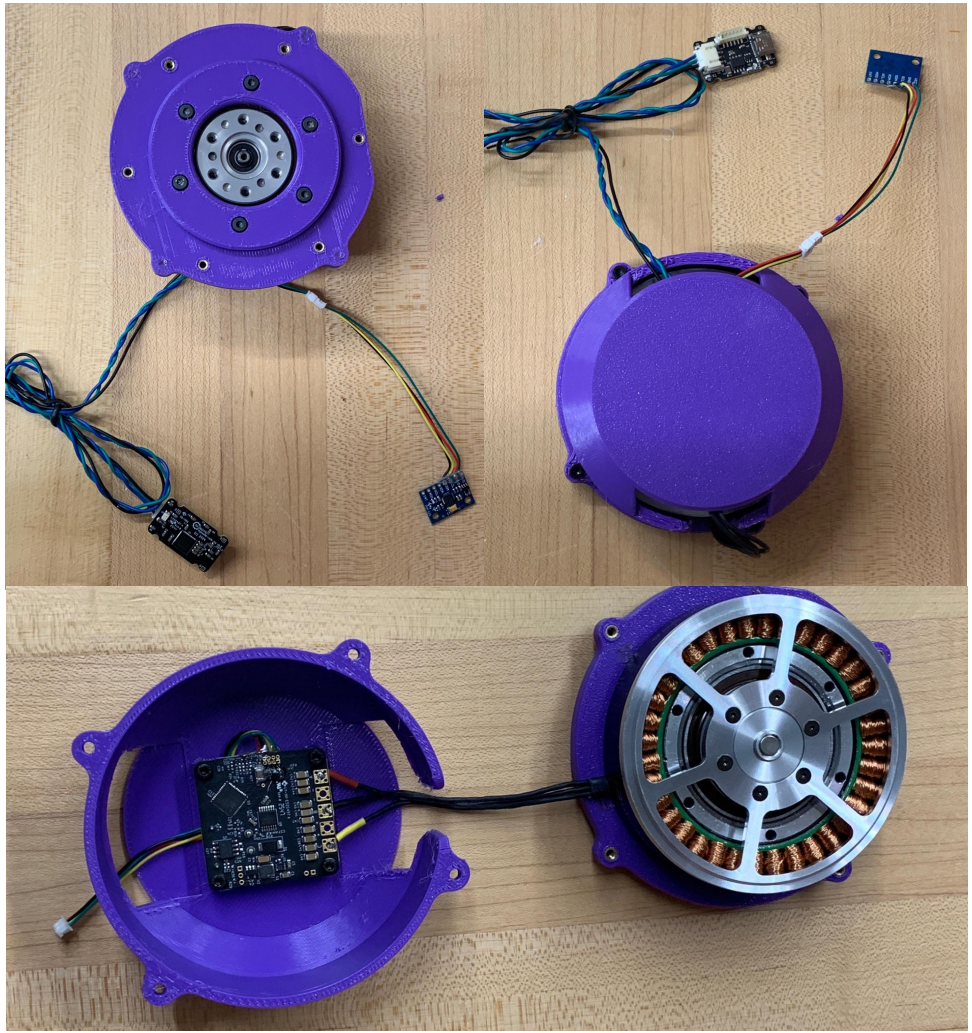






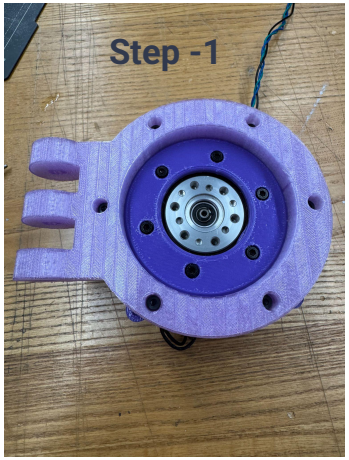
Assembly

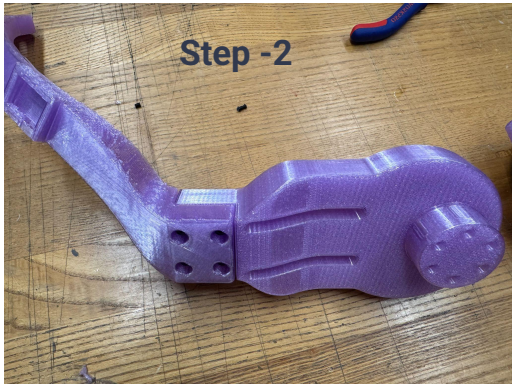
- New motor encasement: which includes AKE80-8 motor and Moteus-C1 with modified firmware which supports MPU-6050 IMU

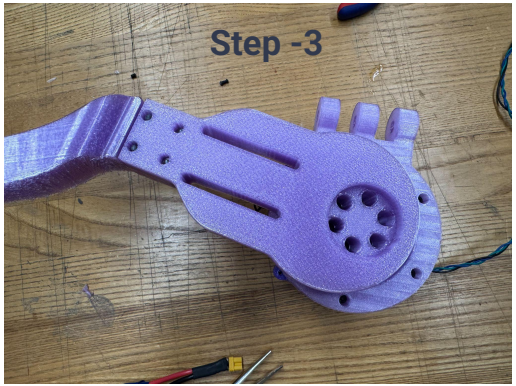


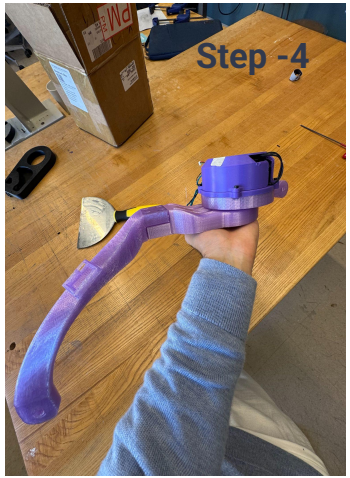
Assembly of Motor Connector

- First, attach the angled connector to the motor using M3×10mm x6 screws
- Next, attach the flat connector to the curved thigh strut using M4 × 10mm x4 screws
- Finally, attach the flat connector to the motor using M3 × 14 screws









Manual

Screenshots from manual, solidworks description, print settings, assembly instructions, etc

Assembly

1. Build the "Legs"

Part list:

- Motor encasement (moteus and motor holders)
- AKE80-8 KV30 motors and moteus-c1 controllers
- Curved thigh strut
- IMU lid
- Flat connector
- Angled motor connect hinge (1 pin, 2 caps, bottom and top pieces)
- Thigh cuff

Screw list:

- Motor encasement:
  - M3x6mm (x4 qty)
  - M3x8mm (x6)
  - M2.5x6mm (x4)
- M4x10mm (x4)
- M3x10mm (x6)
- M3x14mm (x6)
- M5 shoulder screw

Threaded Inserts

- M4 Heat Sink (x4)

Setting Up the Hip Exo Motor

This guide walks a first-time user through setting up a Moteus servo driver with an IMU (MPU-6050) for the hip exoskeleton, from flashing firmware to wiring the sensor to physically assembling the motor. It is written so that someone with no prior experience can follow it step by step. The guide has three sections: Firmware Update, Wiring, and Assembly.

What You Will Need

Hardware

- Moteus C1 servo driver (with the motor already calibrated)
- MPU-6050 IMU breakout board
- CAN-FD adapter (fdcanusb / mjcanfd-usb-1x)
- Power supply for the motor (e.g. XT30 battery/bench supply)
- 4 thin wires for the IMU (SCL, SDA, 3.3V, GND)
- Motor + encasement hardware (see Assembly section for screws/parts)

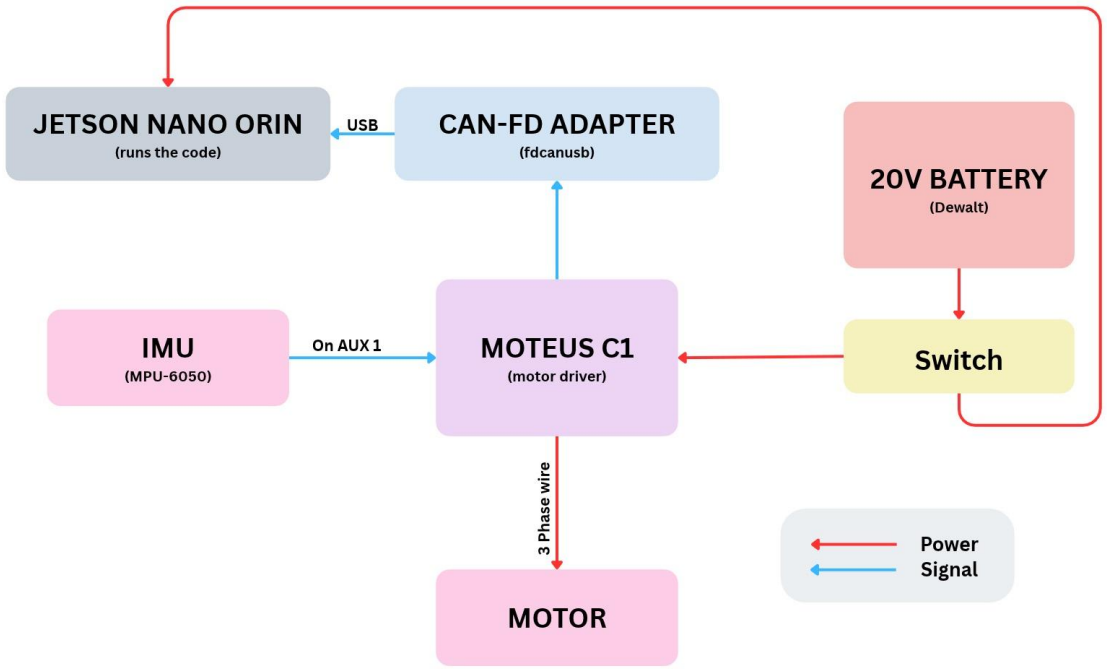
Tools & Software

- Soldering iron + solder
- Wire cutters/strippers
- Helping-hands or a small vise
- Set of Allen keys
- A computer (Mac, Windows, or Linux) with Python 3 installed
- The moteus Python tools (installed in Section 1)

Electrical Parts

Summary of new system

- Switched to the Moteus C1 driver
- Added MPU-6050 IMU, read through the Moteus (custom type-6 firmware)
- Upgraded Jetson to JetPack 6.2.3
- Two environments now: canenv (CAN setup) + moteusenv (Moteus/IMU)



Manual

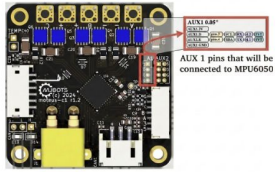
Documentation produced:

- Moteus + IMU setup guide (firmware → wiring → config)
- Jetson firmware flashing guide (normal + SDK Manager fallback)
- System wiring overview (power + signal)
- Common issues / troubleshooting

2.3.3 Identify the pins on the Moteus AUX1

On the Moteus C1, find the AUX1 pads. Using the Moteus C1 pinout diagram for your board revision, locate these four points:

- AUX1 pin 3 — SCL
- AUX1 pin 4 — SDA
- 3.3V
- GND

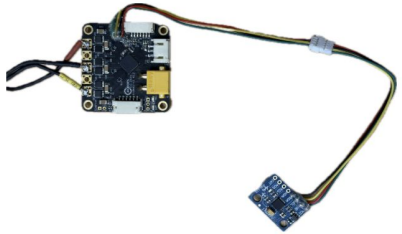


2.3.4 Solder the connections

Connect each MPU-6050 pin to its matching Moteus AUX1 point:

1. MPU-6050 SCL → Moteus AUX1 pin 3
2. MPU-6050 SDA → Moteus AUX1 pin 4
3. MPU-6050 3.3V → Moteus AUX1 3.3V
4. MPU-6050 GND → Moteus AUX1 GND

You can use JST connectors; the connectors in the reference figure below are JNT eCable 100mm male (for Moteus) and female (for imu).



2.2 Connect the power supply to the Moteus

The Moteus does NOT run off USB or the CAN adapter. It needs its own external power supply connected to its power-input terminals.

For this setup we use a DeWalt external 20V, ~5Ah supply. (The Moteus C1 accepts a range of input voltages; 20V is what this build uses.)



NOTE: Double-check polarity before powering on. Reversing + and - can damage the board. Add an inline switch and fuse on the positive lead for safety.

2.3 Wire the IMU (MPU-6050) to the Moteus

2.3.1 Understand the connection

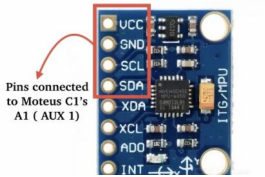
The MPU-6050 talks to the Moteus over I2C, which needs four wires: two for data (SCL and SDA) and two for power (3.3V and GND). On the Moteus C1, the IMU goes on the AUX1 port.

NOTE: The Moteus C1 does NOT have a plug/connector for AUX1; only AUX2 has one. That means the four AUX1 wires must be soldered directly to the pads on the board.

2.3.2 Identify the pins on the MPU-6050

Locate these four labeled pads on the MPU-6050 breakout board:

- VCC / 3.3V — power in (use 3.3V, NOT 5V)
- GND — ground
- SCL — I2C clock
- SDA — I2C data

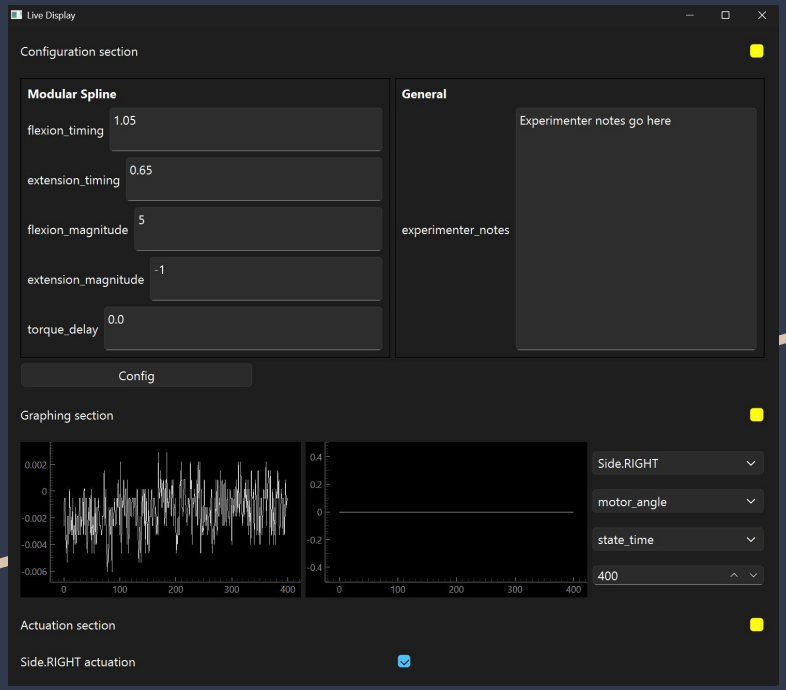


Code/GUI

Integrating Moteus and CAN

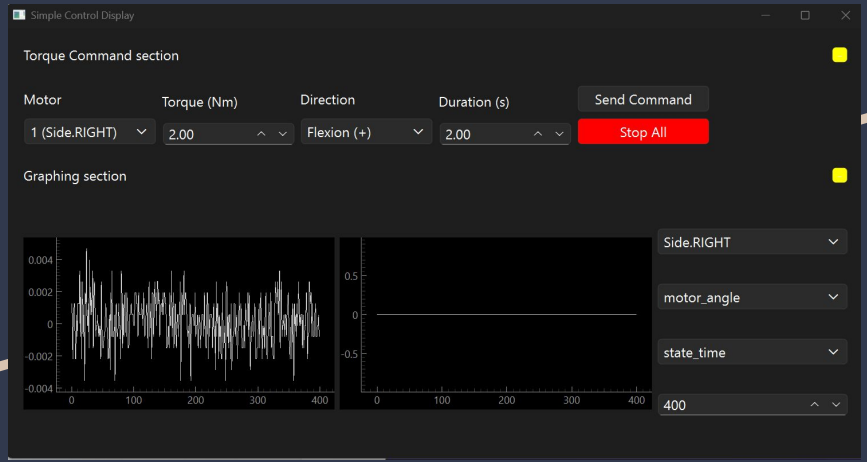
- Added a couple new files and edited some existing files in order to allow for integration of the Moteus system to work alongside the CAN system.
- Updated the OS version of the Jetson, since the previous version was unable to run the moteus files
- Created a separate virtual environment to use separate from the canenv environment since the Moteus and CAN setups have different requirements and this made it safer and more resistant to package/permission conflicts
- Added Moteus functionality to the hip exo system
    - Based heavily off of the CAN system and functionality

Edits to Walking Controller & GUI



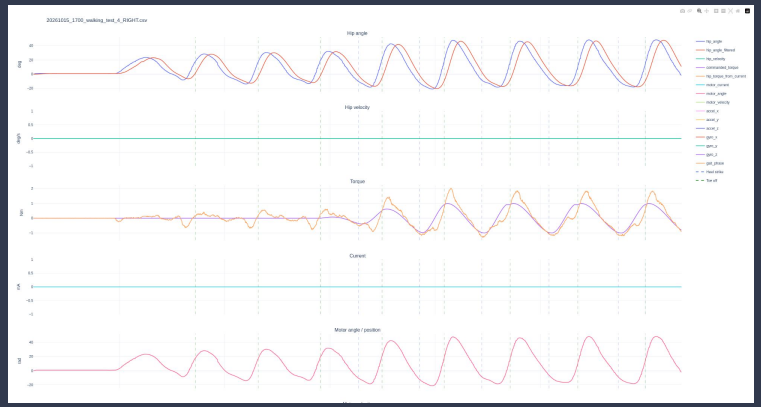
- Small edits to existing Walking Controller
- Changed the setup so that the GUI/Controller will launch safely while only one motor is connected.
    - A warning is sent to the terminal informing the user that only one motor is connected at the time, this will also be transferred into the startup GUI prompts when it is detected.
- Edited the Code such that this GUI and controller now works seamlessly with both the Moteus and CAN systems.

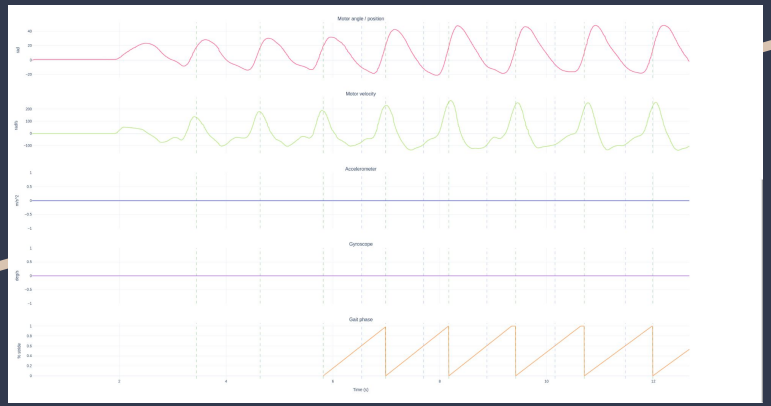
New Simple Controller & GUI



- Created a new Simple Controller with its own GUI
- This controller is designed specifically for testing
- Capabilities:
  - Send torque commands to one motor at a time
  - Direction of torque applied (flexion vs extension)
  - Duration of torque command with stop button
- Plotting:
  - This GUI has all the same plotting capabilities as the original GUI. This includes plotting values such as motor angle, torque applied/commanded, and more
- Maintains the same ability as Walking controller to be able to be used with one or both motors connected
- Maintains usage with both Moteus and CAN systems

Post Trial Plotting





- Created a script that when run will open an html file with the complete values recorded during the previous (or specified) trial
- Works with both Moteus and CAN systems
- Has access to every value that was recorded and these plots are interactable, allowing user to zoom in and out on specific moments in time

