Open-Source Quadruped Robot for Agricultural Research

Austin Neff, Kevin Nelson, Jordan Raver, Sam Spencer, Ben Siesser, Yang Yang

1. Introduction

The Open-Source Quadruped Robot for Agricultural Research is an eight-degree-of-freedom quadruped based on the design of the Open Dynamic Robot Initiative (ODRI) project on GitHub. Due to complications with acquiring the materials and code of the ODRI project, the team shifted to using the ODrive S1 brushless motor control boards to drive the legs of the robot. The purpose of this robot is to provide a customizable and easily recreated robot platform that can be used as a launchpad for other developers down the line. This document will go over the design decisions made by the Fall 2024-Spring 2025 to allow for streamlined changes in the future.

1. Robot Body
   1. Chassis

The chassis was augmented to hold eight ODrive S1 motor boards. A mount was designed to attach the S1 board along with a braking resistor to vertical pegs in the robot’s body. A similar mount was made for the Orange Pi board in the center of the robot. This design was chosen to save as much space as possible to retain the original dimensions of the ODRI robot.

* 1. 2-DoF Legs

The leg actuators are designed according to [ODRI specifications](https://github.com/open-dynamic-robot-initiative/open_robot_actuator_hardware/blob/master/mechanics/actuator_module_v1/actuator_module_v1.1.md#brushless-actuator-module-core-v11). Our design used a modified version of the actuator shell found in the documentation. In case ODRI goes defunct, refer to the actuator build notes for building instructions.

Key Building Notes:

* + 1. Strain Relieving Wires

A big issue with the motors was the solid core wires snapping off in motion. To relieve this issue, the team hot-glued the male end of the banana cables to the actuator shell to decrease the risk of the motor phase wires snapping.

* + 1. Heated Inserts

ODRI prescribed Helicoil inserts to reinforce the screw holes. This, however, required additional Helicoil tools that they didn’t mention in the bill of materials. Instead, the team used heated inserts to fit the screws. These were less precise but more convenient. After insertion, thread the screw holes to remove the residual melted PLA that might have been pushed into the path of the screw.

* + 1. Motor Preparation

**Make sure to always support the underside of the rotor when removing the drive shaft**. Use something sturdy to support it, or use the 3D printed support part from ODRI, and be very careful without making large strikes. Attempt to use something like a pick to concentrate force directly on the shaft to remove.

The rotor should then be reamed to 3 mm. Support the rotor in a clamp with light force so as not to bend it.

* 1. Storage and Carrying

A cover was printed with a carrying handle to carry the robot, and a stand was provided to set the robot on for walking tests. A definitive carrying case was not in scope as the robot was still in walking development.

* 1. Printing Material

The legs of the robot are printed out of CF-PLA from Bambu for use with the Senior Design Lab Bambu 3D printers. Through robot testing, the leg modules notably got hot enough to warp the legs. This is a concern because in normal operation, the motors are likely to get hot enough to at least soften the CF-PLA to a point where the weight on the legs can cause permanent deformation. An idea to get around this is to use a 3D printed material that cannot soften (resin 3D print) or a material with higher heat resistance (APS).

1. Robot Components
   1. ODrive S1 Motor Control Board

The ODrive S1 Motor Control Board is a driver board that allows velocity, position, or torque control. The ODrive S1 has extensive documentation and supports ROS2 (see 4B). The board has compatibility with CAN, UART, I2C, and SPI communication. With the Orange Pi 5’s limited pins, this project uses **CAN communication** as the ODrive natively supports CAN daisy-chaining. The ODrive supports 12V-50V and up to 40A and comes with a braking resistor to dissipate heat. At our voltage (24V), we draw around 5A at max load and do not require the heat spreader plate. **The website recommends a USB isolator when the ODrive S1 is connected to offboard power to avoid shorting USB power.** Due to our timeline, we were unable to purchase the ODrive Micro, which is cheaper and lower power, which is perfect for this project. Unfortunately, the ODrive Micro is on pre-order during the creation of our robot. **We highly recommend future teams look into using the Micro boards instead as the code will likely transfer over seamlessly.**

ODrive S1: <https://odriverobotics.com/shop/odrive-s1>  
ODrive Micro: <https://shop.odriverobotics.com/products/odrive-micro>

* 1. Orange Pi 5 (Standard Model)
     1. CAN Communication Limits

The default Orange Pi 5 **does not include CAN transceiving**, requiring additional off-board CAN transceivers to operate. The Orange Pi 5 has 2 built-in CAN receivers that can be used.

CAN Transceivers used: <https://www.amazon.com/SN65HVD230-CAN-Board-Communication-Development/dp/B0B5DTN62K?source=ps-sl-shoppingads-lpcontext&ref_=fplfs&psc=1&smid=A50C560NZEBBE>

The CAN protocol must be activated manually if using Ubuntu 22.04 LTS. [insert activation steps here]

The CAN communication lines are set to 250 kHz on default and this speed can be changed in the configuration of the ODrive boards. At this baud rate, all eight ODrive S1 boards can be daisy-chained via CAN. The Orange Pi by default has set its CAN baud rate to 1 MHz by default; ensure the baud rate matches on the Orange Pi and ODrive boards. The side CAN cables are connected in parallel and were confirmed by testing the resistance on the lines.

* + - 1. Daisy-Chaining Issues and the Tale of the CAN Wires From Hell (but mostly China)

**Make sure all CAN ports are connected CAN-H to CAN-H and CAN-L to CAN-L with the ODrive board’s termination resistors activated only at the last board.**

Upon connecting all 8 leg segments, issues began on CAN connectivity. The biggest issue is **alternating ODrive boards being missed along the chain**. This issue was caused by the pre-made 4-pin JST-GH connectors which have reversed connections. The boards were alternating as CAN-H and CAN-L lines were reversed between boards. Thankfully, the ODrive S1 boards are isolated and this incident did not affect the functionality of the CAN daisy chain.

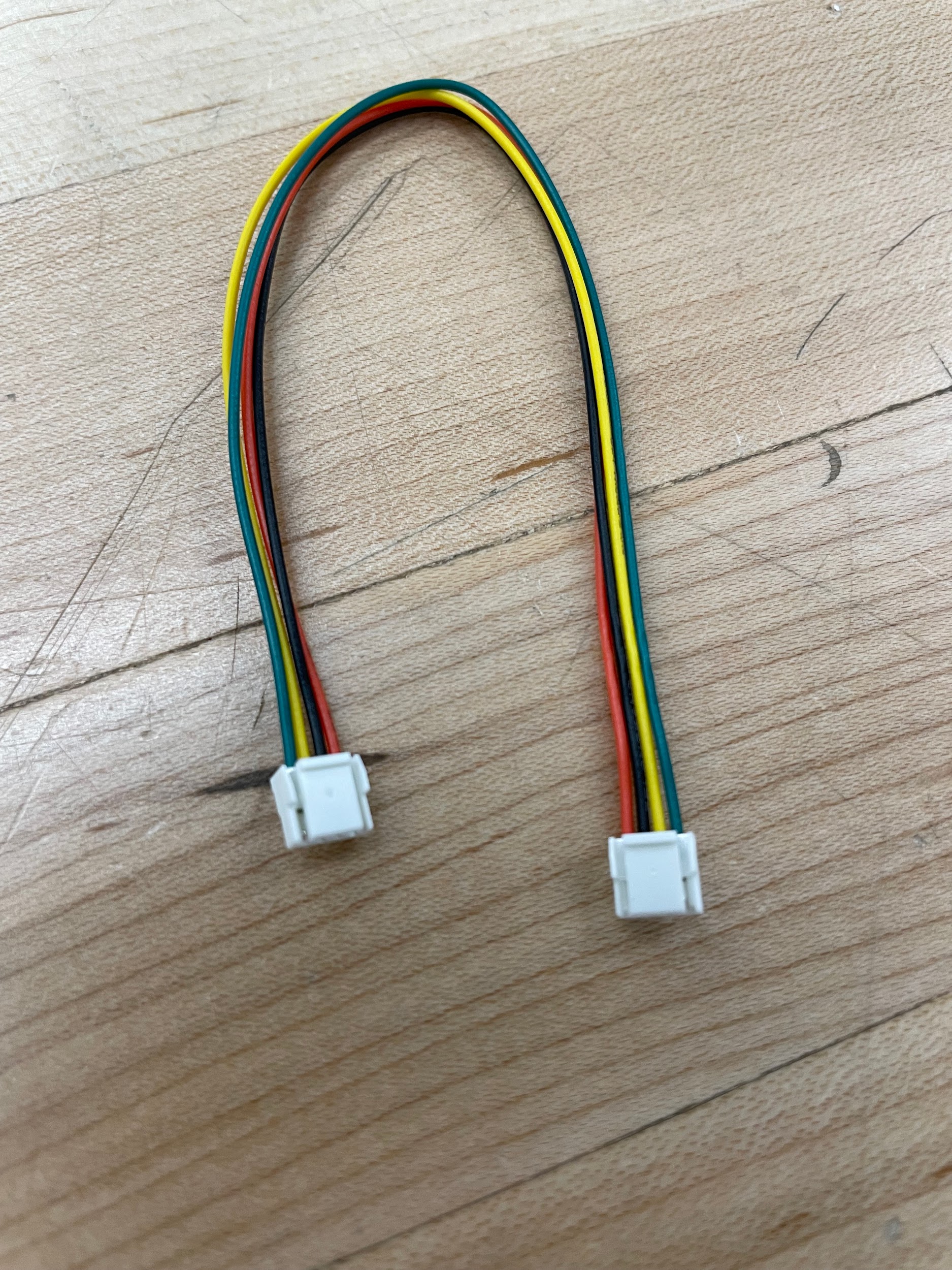


Figure: Take the wires out and make sure they’re in the same order on both sides

* + 1. SSH Remote Access

The robot can be reached via SSH by console commands. SSH protocol is pre-installed on all Ubuntu systems and theoretically all Linux systems. Windows systems require a dedicated SSH program. In the Ubuntu console, type “ssh quadrupedrobotgroup@192.168.91.111”. When prompted with the password, type “opi”.

“ssh quadrupedrobotgroup@192.168.91.111”  
Username: quadrupedrobotgroup  
Local IP: 192.168.91.111  
Password: opi

If the protocol does not go through, it’s likely the IP has been reset. Attempt to connect keyboard, mouse, and monitor directly to the Orange Pi 5 to access the system and re-acquire the IP from there. The password for the system is also “opi”.

* 1. Adafruit LSM6DSOX 6-DoF Accelerometer and Gyroscope

The LSM6DSOX was acquired as a replacement for the IMU prescribed by the ODRI project. The purpose of this component is to **determine the rotation of the robot**. The IMU used by the ODRI project could use 6-DoF due to the “Kalman Filtering” process to isolate the gravity vector from the accelerometer and uses statistics to accurately determine the rotation of the robot. Kalman Filtering is complex and likely out of scope for our team. Our team uses a combination of gyroscope and accelerometer measurements to determine the rotation of the robot (see 4C). **We do not recommend using the 6-DoF IMU for future projects unless you want to move forward with Kalman Filtering.** **Adafruit and Sparkfun makes IMUs with magnometers with libraries to determine the IMU rotation imbedded.** The purchase of the LSM6DS was made prior to discovering ODRI’s intentions, and we were able to make it work.

Adafruit 6-DoF LSM6DSOX: <https://www.adafruit.com/product/4438>

* 1. Offboard Power Using Power Tool Batteries

The robot uses two 18V 5Ah Milwaukee batteries. These should provide adequate safety features to protect the electronics. The batteries lead into a switch, then the control board. However, these batteries are really heavy. If possible, replace the batteries with LiPo battery packs with a battery control module, if that will save weight.

1. Software Implementations

Refer to the coding documentation for further details.

* 1. ROS2 Humble Hawkbill

ROS2 was used to organize the code into packages. ROS2 Humble Hawksbill has the best compatibility with Ubuntu LTS 22.04. It’s possible to install the Linux distribution on a USB drive and to boot from the USB whenever you need to work on the project.

* 1. ODriveTool

ODriveTool is a console script to change configuration settings. Theoretically, this tool can be used via CAN communication. We recommend that future teams look at its functionality.

* 1. Adafruit LSM6DS and CircuitPython “Blinka”

1. Common Issues
   1. Calibration Step Issues
      1. **Leg Unmoving But ODrive Board Flashing Green - Current Dumping**

**This is the most serious issue that can be encountered** when dealing with the robot. When phase wires snap, there’s a chance the leg is visibly not moving but current is being dumped into the motor. This is extremely bad for the robot and will cause the motor to heat rapidly to a dangerously high temperature if not detected. This will cause warping to the legs and will require rebuilding if severe. Immediately disconnect the robot if you suspect this to be happening and examine the banana cables for disconnects.

* + 1. Jittering Legs - Motor Phase Disconnected

This is the less concerning cousin of the first issue. If the leg is appearing to jitter in place, turn off the robot and check the banana cable for disconnects.

* + 1. Leg Not Changing Direction - Encoder Issue

If the legs are not changing direction in calibration, this is likely an encoder issue. The most common issue is the ribbon cables falling out of the crimp pins. Check the crimp pins on the female side of the 31-pin JST connector for loose ribbon wires. If that doesn’t solve the issue, check the soldering connections between the ribbon wires on the encoder reader or replace the ribbon cable entirely. If this doesn’t work, it may be time to replace the encoder reader altogether.

* 1. Leg Resistance Issues
     1. Metal chips in rotor magnets

If you move the leg and detect that the leg is locking up a little before moving again, there might be metal shavings between the rotor and stator. Loosen the set screws on the rotor and pull hard to remove. Use something tacky (tape, putty, etc.) to wipe around the magnets to make sure all shavings are removed.

* + 1. Belt Tensioning

If skipping is encountered, open the leg up and replace the belt tensioners. The belt tensioners are two bearings on the lid with red/white 3D printed covers. Make 2-3 shallow cuts not to damage the bearing, and the bearing should come out easily. Refer to the print files or ODRI documentation for printing .stl files and there should be a few spare in the parts box.

\*\*\*When opening up, compress the center gear with a pick to make sure the belt doesn’t deform the encoder wheel.