Capstone Project: KiwiBot Construction Guide

Nicholas Mutlak

University of Toronto

April 10th, 2025

Introduction

During the academic year of 2024-2025, a robot that will be referred to as 'KiwiBot' was designed, built, and programmed. The purpose of this robot was to spark interest in engineering for high school students through a technically impressive design and a hands-on interactive controller, that would do what most other hobby robots don't. This guide entails brief details about the robot's design process, construction, and programming. For more detailed information refer to the project's Final Report. For control and troubleshooting refer to the project's User Guide.

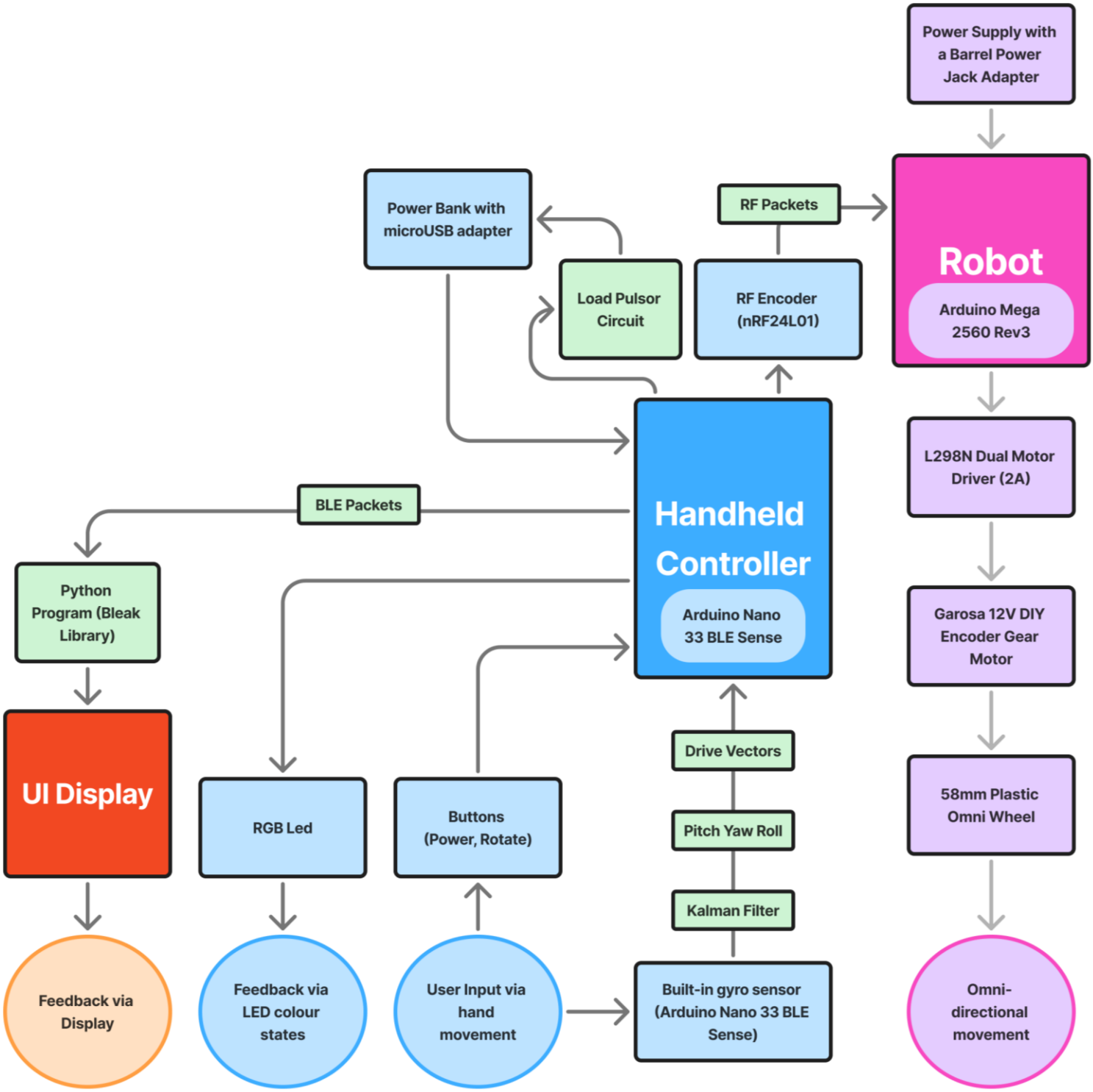
Parts List

Part	Qty	Link
Laptop	1	Borrowed
Arduino Mega 2560 Rev3	1	https://www.canadarobotix.com/collections/featured-1/products/266
L298N Motor Drivers	1	https://www.amazon.ca/PChero-2Packs-Controller-Stepper-Arduino/dp/B07C4B3DL4/
58mm Plastic Omni Wheels	3	https://ca.robotshop.com/products/58mm-plastic-omni-wheel-compatible-servos-lego-mindstorms-nxt
Garosa 12V DIY Encoder Gear Motor	3	https://www.amazon.ca/dp/B07WRYYG4G
USB A/B Connector	1	Personal Donation
Talentcell Rechargeable 12V 3000mAh Lithium ion Battery Pack	1	https://www.amazon.ca/dp/B01M7Z9Z1N
DC Adapter Female Barrel Power Jack	1	https://www.amazon.ca/dp/B0D3VDG88R
Arduino Nano 33 BLE Sense Rev2	1	https://www.canadarobotix.com/products/3031
TNTOR Power Bank 5V 2500mAH	1	https://www.amazon.ca/dp/B0CJFMNKBV
USBC / MicroUSB Adapter	1	https://www.amazon.ca/Adapter-BabyElf-Convert-Connector-Compatible/dp/B087WKS9NZ/
Push Buttons	2	MyFab
nRF24L01 RF Module	2	https://www.canadarobotix.com/products/2437
Assorted Wires	X	MyFab
Assorted Machine Screws / Bolts	X	Personal Donation / RONA

The chassis on which all these parts are mounted, as well as the hex shafts for the wheels were 3D printed by a Ender Pro 3 v2 in 5 main sections for the robot and 3 main sections for the controller. The robot's sections are connected together in place through M10 machine screws and nuts. The controller's battery case is standalone, while the button case is two distinct pieces connected together by M4 machine screws and bolts. Superglue was used for different mounts.

The files for the 3D print design were made primarily in Fusion360, which you can find in the CAD_Files folder.

Hardware Setup



The design is meant to take tilt controls measured from the controller's Arduino Nano 33 BLE Sense Rev2 and convert them to control signals of X,Y (planar) and W (rotational) velocities. The buttons on the robot control when these velocities are allowed to be non-zero or if they get scaled up or down. There are 3 main states which are displayed by the LED. The Power button is primary, displaying Red when off and the zero-ing out the velocities regardless of tilt. When pressed the controller will either be in the Green or Yellow state (green being only planar movement and yellow allowing for rotational movement). The states can be swapped with the secondary Rotate button. All this information is packaged into RF and BLE packets, to be sent to the robot and a computer respectively to control to the robot as well as show the controls in real time on a display.

The communication range on these devices is reliable to at minimum 80ft and in theory can reach a max range of 900ft if there are no obstructions.

Both batteries of the controller and robot are rechargeable. Theoretical runtime of robot on max motor output is 2.5 hours with tested runtime being at minimum 1 hour of use. Theoretical runtime of controller is 24 hours with tested runtime being at minimum 1 hour of use. Recharging the robot usually takes approximately an hour and the glove approximately under 30 minutes depending on how much the devices are drained.

The robot's Arduino Mega 2560 Rev3 has the pinout capacity to support PID control with the Garosa motor's encoders. It was deemed in the scope of the project and the floaty-feeling control of the glove controller that it would not be necessary. Additionally, the robot's MCU is also capable of supporting more peripherals such as a 4^{th} motor, collision sensors or other communication modules if desired.

The TNTOR Power Bank for the controller has an auto shut off feature, as it's original intention is to charge phones and the current draw of all the controller's components is not large enough to register above its detector's threshold. The load pulser circuit was then designed, with a simple resistor network of a ~50 Ohm load wired the 5V USB line to draw 100mA of current from the battery. Doing this through the Arduino Nano board itself is a risk as its not designed to draw that much current at a constant rate due to risks of overheating, and as such a transistor is used to pulse the circuit only every 30 seconds, saving battery life as well as providing protection to the MCU.

Programming

Arduino IDE was used to program in C++ and upload sketches stored as .ino files to both of the Arduino boards. The Arduino Nano 33 BLE Sense Rev2 has been uploaded the glove_main_ble.ino sketch, while Arduin Mega 2560 Rev3 has been uploaded the robot_main.ino sketch. To upload a new sketch to any of the boards, connect a USB cable between the board and your computer, and upload via Arduino IDE.

BMI270 / BMM150 IMU Sensors

The greatest challenge of this project was getting accurate tilt controls that felt good and intuitive. Of the ~25 people that used the design, it's clear that for some it comes very easily, while for others it's hard to understand. That aside, during the early phases of construction, one of the major choices being made was to use either an MPU6050 or an MCU with built-in IMUs like the Arduino Nano 33 BLE Sense Rev2 (BMI270 + BMM150). Due to a desire to simplify the hardware on the Controller as much as possible along with budget constraints, the team would select the Arduino Board with built-in sensors. It should be noted that both IMUs are able to achieve a IMU measurement range of +/- 4 Gs on the accelerometer and +/- 2000 deg/s on the gyroscope, but the Arduino also has the BMM150 to make it a 9-axis sensor. A more comparable external IMU would be the MPU9250. Another consideration going into choosing the Arduino is that with an internal measurement unit built-in, sampling rate is automatically handled and overhead to prompt the sensors is minimized, whereas with an external measurement unit, communication would have to be over a serial bus and potentially susceptible to noise from other hardware on the controller.

It should also be noted after much testing, that the magnetometer (BMM150) is highly unreliable and susceptible to noise in the environment, hence its influence in the Kalman Filter design was heavily reduced to be a simple error tracking term for the gyroscopic sensor's yaw predictions.

Over the course of the project several different filters would be tested to get better results. The process can be seen in the gyro_tuning.ino file, which contains the functions used to tune the gyroscopic and magnetometer sensors, as well as manually analyzing the predictions of independent sensors alone, complementary filters, and Madgwick filters. To see these, simply connect the Arduino Nano to a computer, uncomment a desired function in the main loop of the program in Arduino IDE and upload the sketch. The Kalman filter would however be selected and implementing in the gyro_main_ble.ino file because it was highly tunable to the design, leading to the best results given the condition that the Controller is not turned upside down as a result of using Euler angles over quaternions.

Python UI App

This is a rudimentary matplotlib application, initially deemed infeasible, as I'm personally inexperienced with proper app design and working with BLE on Windows. This program using the Bleak python library to connect to the advertising service that the Controller sends out to receive information and visually displays how the Robot handles that information with the drive vectors shown on screen. There are two threads that run asynchronously; one updates a list of shared global variables using the Bleak library, while the other updates the UI based on these variables. It was made as an aid to first time users that struggled to understand how to control the robot, however it's effectiveness in doing so is questionable, as a user would have to focus their attention between the Robot's movements, their hand movements, as well as the UI display.

Recommendations for the Future

1. Single-handed tilt controls are unique and technically impressive to students, as well as covers up the floaty control and motor drift during runtime. That said, there are ultimately better methods of control. It was determined about halfway through the project that a controller with 2 analog potentiometer joysticks (one to control orientation and one to control direction of movement; or simply just one for the latter) rather than a glove would be easier and snappier to control, assuming PID control is implemented to remove motor drift. It is entirely possible to implement a controller that does both joystick and tilt control and can swap between those methods of control, even while maintaining the wearable glove concept.
2. One of the major recommendations for taking this project further is reducing costs to turn it into a workshop kit. A drastic way to do this would be to cut out the Robot entirely. With the hardware of the Controller alone, the project would consist of the Arduino Nano 33 BLE Sense Rev2, two push buttons, an RGB Led, and the power bank. However, even such a large power bank used in the project is not needed and can be swapped out for AA batteries instead, given the low power nature of the Controller. This would reduce costs approximately to $80 per kit.
3. In tandem with Recommendation 2, without a physical Robot the simulation software should be heavily advanced. It is possible to let students CAD out their own KiwiDrive Robot or use a base template Robot, for which they can port into a 3D software simulation and control the Robot through a digital landscape with the Controller. Once they have done a software simulation test, they could try their controller out on the real thing, with one shared physical Robot, that we've already made. This would be a major undertaking potentially worth another Capstone Project.
4. Another method of reducing costs would be to not use Arduino boards that come with a lot of extra bloat hardware but rather have a person who can design custom PCBs with select chips, which was originally intended at the start of this project. Currently ~25% of the project costs are from the Arduino boards. This would also help reduce space these components require.
5. The wiring of the Controller is exposed and very vulnerable to weather conditions. Although it wasn't in the scope of this project, it was initially desired and is recommended that the hardware for the Controller be contained within a protective casing.
6. Occasionally PWM/DIR wires connecting to the Arduino Mega 2560 (on the second layer) come loose from the L298Ns (on the base layer). Find a fix for this; currently using electrical tape as a stand-in.
7. Fix sticky buttons on Controller.