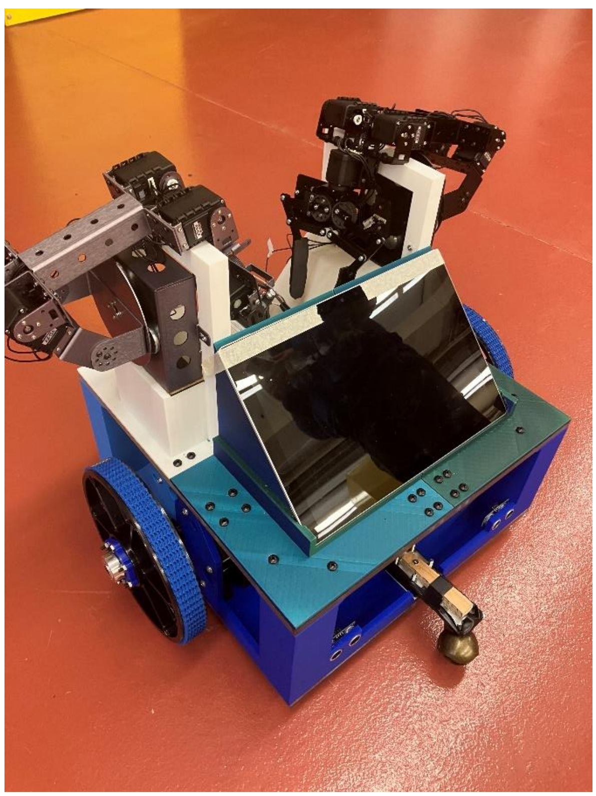
Robot Project: Mr. Grabs' Construction Guide

Nicholas Mutlak

University of Toronto

August 31st, 2023



Introduction

During the summer of 2023, a robot that will be referred to as 'Mr. Grabs' was designed, built, and programmed in a 6-week timeline. The purpose of this robot was to accompany a separate project being developed on a computer by another student, that would involve both facial and voice recognition and interaction with human beings. The robot would allow for the computer of that project to be mounted onto it, and from there make the features of that project mobile and more expressive through the use of mechanical arms. This guide entails details about the robot's design process, construction, and programming.

Parts List

Part	Qty	Link
Vex Robotics SS Gearbox, 84:12	1	https://www.vexrobotics.com/single-speed-single-reduction.html
CIM Motor	2	https://www.andymark.com/products/2-5-in-cim-motor
8" Plaction Wheels	2	https://www.andymark.com/products/8-in-plaction-wheel-with-blue-nitrile-tread
1.5" Castor Wheels	2	https://www.amazon.ca/Caster-Wheels-Furniture-Profile-Roller-Included/dp/B07T8NMYFZ
½ Hex Spacers	8	AndyMark
½ Hex Shaft Collars	2	AndyMark
Motor Controller (FAULTY)	1	https://www.amazon.ca/H-Bridge-Control-IRF3205-Response-Arduino/dp/B08JM4Z1HP
Arduino Nano 33 BLE	1	https://store-usa.arduino.cc/products/arduino-nano-33-ble
Adafruit Feather 32u4 Bluefruit LE	2	https://www.adafruit.com/product/2829
30A Breaker	1	https://www.amazon.ca/iplusmile-Circuit-Breaker-Manual-Marine/dp/B086X2SZL2
12V Battery	1	https://batteryclerk.ca/products/sigmastek-sp12-7-5-12v-7ah-sealed-lead-acid-replacement-battery
4V Battery	1	https://www.amazon.ca/Volt-Sealed-Lead-Acid-Battery/dp/B00ZCPF274
HC-SR04 Ultrasonic Sensors	4	https://www.digikey.ca/en/products/detail/adafruit-industries-llc/3942/9658069
AX12-A Smart Robotic Arm	2	https://www.crustcrawler.com/products/AX12A%20Smart%20Robotic%20Arm/
Android Phone	1	Borrowed (provided by students)
Microsoft Surface Pro 7	1	Borrowed (provided by students)
Assortment of 10-32 Screws	125+	Amazon

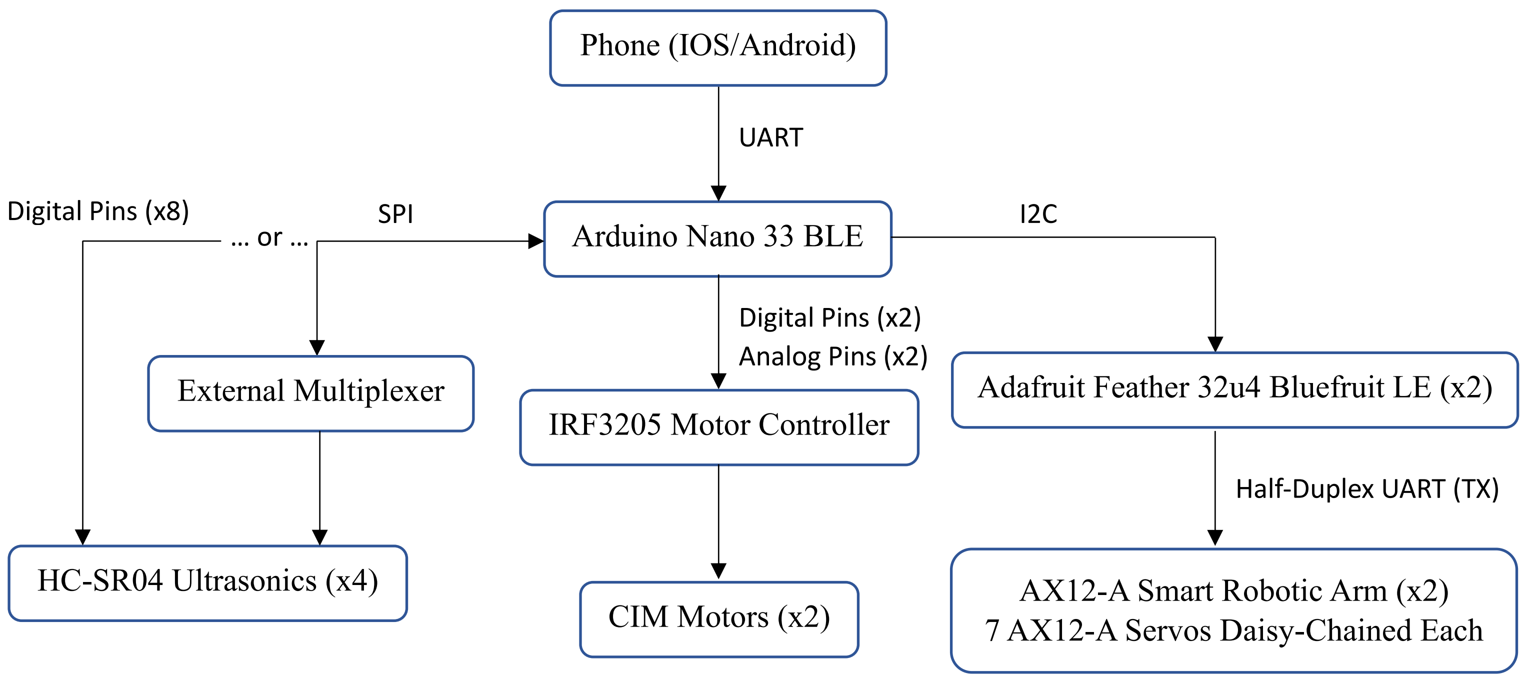
The chassis on which all these parts rest on, was 3D printed by a Bambu Lab X1 Carbon 3D Printer in eight, 8x8" sections, for a full 16x16" chassis. Two sheets of 16x16" quarter inch thick plexiglass were used to secure the 3D printed chassis together with machine screws and nuts.

In addition to the main body, mounts for the castor wheels where printed after a few design iterations. These mounts were not successful in upholding the weight of the robot. A splint made out of wood, a stronger material than the 15% infill plastic mount, was designed to fix this problem.

The files for the 3D print design were made in Fusion360, which you can find in the CAD_PrintFiles folder. The CAD models these designs were made around can be found in the CAD_StepFiles folder.

Communication Setup

The robot was designed to have the following parts talk to each other through different communication protocols:



Applications made through MIT App Inventor are compatible with both IOS and Android devices, however the BLE extension used for the app, is not compatible with IOS devices as of 2023. From the bluetooth app, the phone would connect wirelessly with the Arduino Nano 33 BLE, with its UART pins. Using two digital and analog pins, the Arduino Nano 33 BLE would interpret signals from the app to control the CIM motors through the IRF3205 Motor Controller.

For actions related to the mechanical arms sent from the app, the Arduino Nano 33 BLE would send signals through I2C communication protocol to the two Adafruit Feather 32u4 Bluefruit LE boards. The Adafruit boards would then interpret those signals and transmit machine code instructions through a packet handler to the 7 daisy-chained AX12-A servos that control the arms movement. It should be noted that due to limitations of the Adafruit boards, they are only able to either send bytes (TX) or receive bytes (RX) during runtime, and are not able to do both. This means the Adafruit boards are not able to read information like a servo's position, and as such from a programming side, resulting in having to resort to using delays to make an educated guess when the arm will have finished moving to a new position during runtime.

Lastly the robot comes equipped with four HC-SR04 Ultrasonic Sensors. Due to lack of time and additionally having the unanswered question of, "If the Arduino Nano 33 BLE is reading from four ultrasonic sensors sequentially through polling would it slow down the control of the robot too much?" the sensors were left unwired. It was intended for them to be used as emergency stops, to ensure the robot does not crash into objects during operation. To wire them up, two options were considered. Option 1 would be to use 8 digital pins, which the Arduino Nano 33 BLE can support. Option 2 would be to use SPI communication protocol with an external multiplexer that could be used for the chip select, as HC-SR04 do not have an internal chip select.

Programming

Arduino IDE was used to program in C++ and upload sketches stored as .ino files to all of the Arduino and Adafruit boards. The Arduino Nano 33 BLE has been uploaded the Main_Nano_Board.ino sketch, while the two Adafruit boards have been uploaded the Left_Arm_Adafruit_Board.ino and Right_Arm_Adafruit_Board.ino files with respect to how they are physically labelled on the robot. To upload a new sketch to any of the boards, connect a USB to a Micro-USB cable between the board and your computer, and upload via the Arduino IDE.

AX12-A Smart Robotic Arm

The AX12-A Smart Robotic Arms were acquired for 4th year robotics labs, at a minimum of 10 years ago. The labs were designed for Matlab, and documentation regarding them has been lost. The setup guide to put build an arm from a kit that comes with the AX12-A Smart Robotic Arm, should be stored with the robot.

The arms were purchased from CrustCrawler and documentation can be found here: https://www.crustcrawler.com/products/AX12A%20Smart%20Robotic%20Arm/

Further documentation about the AX12-A servos and how their control tables are setup can be found here: https://emanual.robotis.com/docs/en/dxl/ax/ax-12a/

Over the timeline of this robot project, there were two methods of interacting with the robotic arm. Method 1 was through Arduino/Adafruit boards using the AX12-A library for Arduino IDE, documentation about which can be found here: https://github.com/jumejume1/AX-12A-servo-library. This method is somewhat lacking, as the Arduino/Adafruit boards used on the robot can only either read or send data through a TX or RX pin, but was necessary for the arms to be controlled independently of a computer.

Method 2 was through how the robotic arms were initially designed to function, with a USB2Dynamixel board and the Dynamixel Software Development Kit (SDK), which would allow a computer to interface with the robotic arm via Visual Studio Express. To do this requires a considerable larger amount of setup, for which documentation can be found here: https://emanual.robotis.com/docs/en/software/dynamixel/dynamixel_sdk/download/#repository and here: https://emanual.robotis.com/docs/en/software/dynamixel/dynamixel_sdk/library_setup/cpp_windows/#cpp-windows.

The benefit of this method is the USB2Dynamixel board is able to both write and read data to the arms during runtime. This was used to manipulate primarily to manipulate arms into desired positions by hand, and then read the position of each arm and record them as states the arm would be in during operation.

In addition to either of these methods, it is necessary to use a computer with a USB2Dynamixel to set the baud rate and ids of any new-out-of-the-box servos to 1,000,000 and 1 through 7 in the daisy chain, if you are building a new robotic arm. This requires the installation of RoboPlus and use of the Dynamixel Wizard, documentation on which can be found here: https://robotis.co.uk/software/roboplus-1-0.html and here: https://emanual.robotis.com/docs/en/software/rplus1/dynamixel_wizard/.

Bluetooth App

There are several applications on IOS that can connect to an Arduino and interact with it, however there were none that provided a decent controller interface, were free, and worked with the Arduino Nano 33 BLE specifically. As such, a custom bluetooth app was created using MIT App Inventor. To build your own app using the MIT App Inventor code block sketch interface or better understand the design of BLE5.aia already made for this project, refer to this tutorial: https://www.youtube.com/watch?v=RvbWl8rZOoQ to get started. It should be noted that BLE5.aia project makes use of more than what the basic tutorial provided covers. As such, further documentation about the BluetoothLE extension used for the project can be found here: https://iot.appinventor.mit.edu/#/bluetoothle/bluetoothleintro

Recommendations for the Future

1. It is HIGHLY recommended in the future that for a project of this scale, to allot more than 6 weeks and a single person to work on the project, for the sake of allowing more time to think through the design in its initial phases, different ideas to come up from different people, and minimize oversights.
2. It is HIGHLY recommended that stronger mounts for castor wheels are designed in any new design iterations of this robot, as that is the robot's biggest current design weakness and point of failure during operation.
3. It is HIGHLY recommended that if the design is to be 3D printed again, methods be implemented to reduce the use of supports as much as possible, while taking into consideration the structural integrity of the design. Reducing the use of supports, both reduces the costs spent on materials and reduces the printing time of each piece, which

can make the difference between printing 1 piece or 2 pieces in a single day. Talk to the person who is managing the 3D printer you want to use, for ideas.

4. It is recommended to look into methods of recharging the robot's 12V and 4V battery, as the Energy Systems Lab currently does not have a battery charger, as of 2023. The professor has stated that using a DC power supply to charge the battery is viable.
5. It is recommended to invest in a Roborio for both now and all future projects, rather than using Arduino or Adafruit boards, as a Roborio can support several different kinds of I/O at a processing speed much faster than an Arduino/Adafruit board would be able to provide.
6. It is recommended to swap out the bolts on the bottom of the robot for shorter 1'' ones to allow for more clearance underneath the robot.
7. It is recommended to find a different method of wiring hardware parts of the robot together, as to not use cheap/old jumper wires and solder to hold them together under heat shrink.
8. It is also recommended in future design iterations to allow more time to think about wiring and include design features like hooks to help secure wires down.
9. It is recommended that if the Ultrasonics Sensors are to be wired up to the Arduino Nano 33 BLE board, that an additional SPI multiplexer chip is used to reduce the number of digital pins used up by the Nano. This is to leave room for potential future additions.

Special Thanks

This section is a thank you to all the people that helped make the completion of this project possible, and can also be used as a list of potential resources for new students working on this robot.

Thank you to:

- Professor Hamid Shokrollah-Timorabadi for providing the funding required to build the robot and a good sense of humor.
- Iman Koohi for allowing the use of his 3D printer and materials.
- Michael Colacci for allowing full use of his tools, supplies, and advice.
- Mehrdad Mehramiz for providing the mechanical arms used on the robot.
- Ryan Seto for providing integral advice with regard to 3D modelling and designing the robot.
- Simon Xu for providing manual assistance in putting the robot together.
- Shuntaro Wakamatsu for creating the logo displayed on the robot.