Flight Simulator Chair Proof of Concept

Dezeray Kowalski, Jamison Bauer, John Young, and Seth Kingston

***Abstract*—Our mission is to provide a unique, immersive, and enjoyable arcade gaming interface. It provides a more realistic and exciting experience through unique physical interaction. Unlike similar creations, we aim to prove an idea for this new gaming experience that could become an affordable arcade product.**

I. Introduction

As video game-supporting technology has improved over the years, new and unique gaming interfaces have transitioned from large arcade style systems to small in-home systems. Due to this trend, users have lost the excitement of leaving their home to experience gaming in a special, unique way. To revive this experience, new large-scale video game interfaces must offer an experience far more exciting than the current in-home gaming standard.

We will provide a new level of immersive gaming by creating a motion-simulating interface for games. As a proof of concept, we will produce a scaled-down version of our design. Our design includes both hardware and software; the hardware is a cage-like structure that is rotated by stepper motors and the software controls the positioning of that structure. A micro-controller and two motor-controller boards manage the system. These boards move the motors to account for both pitch and roll rotational axes and are capable of moving a seat in 6 degrees of freedom. Our design will also interface with driving and flying games to provide input to the micro-controller for motor control. The seat will mimic the player's in-game spatial orientation and forces of acceleration acting on the player. To cut costs and improve user immersion our system design is aimed towards using a virtual reality headset, instead of a monitor. The design and construction of the scaled down structure are done with stability, durability, and safety in mind.

Gamers around the world are always seeking a new way to experience games, especially in a physical sense. Our product will allow the user to feel as though they are in the game. It will also allow people to get out of their home to have a 21st century arcade experience. Because of this, our product will be marketable and in demand.

II. Motivation

With the evolution of small, in-home gaming systems the classic arcade style of entertainment is becoming much less prevalent. The connection with games and the experiences they provide is being lost due to the diminishing interest in leaving home to play games. It seems that bigger, more immersive gaming systems are now the upcoming technology to combat the less engaging home console. The idea of immersive gaming technology is to have a system that can physically envelop the gamer in the game. There are currently many thoughts on how to do this, including using different input devices or giving the user physical feedback. However, there are few popular systems in the market that create this gaming style.

The lack of immersive gaming in the world is a challenge that has yet to be tackled. Often times, systems do not effectively absorb the gamer because of a lack of eliminating distractions. This means that a user cannot focus fully on the information presented to them in game because of their real world surroundings. An example that relates to our system is sitting in a chair that is moving as if the user is in a car game, but the user can look to the side and see that they are, in fact, not on a road or even in a car.

These kinds of distractions from the outside world are the reason that things like virtual reality (VR) headsets were invented; these devices that you wear on your head move with you as you look around and greatly minimize distractions from your peripheral vision. However, the emerging problem regarding gaming with only a VR headset is that the user does not physically feel the virtual world that exists around them, no matter how clearly they can see it.

So, the clearest solution is to create a hybrid of the visual and physical world of a game, by combining the technologies of an arcade chair and a VR headset. In this type of system, a user can see the virtual world without distraction and physically feel like they are in that world. But still, challenges emerge in this design, a main one being the monetary cost to create a system of this scale and complexity.

That is why the goal of our project is to create a gaming experience that fully immerses the user– without distractions from unwanted stimuli– at an affordable price. We plan to create a proof of concept of this system that involves a mechanical chair with 720 degrees of motion and a VR headset to interface with a flight simulator. This project allows us to work out the most effective design for absorbing the user and providing the most natural feeling motion, while maintaining the lowest cost possible.

III. Background

Our design is based on similar designs that have been produced by other small teams around the world. Most of these designs provide 720 degrees of motion. They achieve 720 degrees of motion by using two motors, one to rotate the outer frame, and another to rotate the inner frame. Their main

structure contains a large motor to rotate the bigger frame. The bigger frame is rotated by the large motor around the roll axis, with respect to the user. Additionally, inside the bigger frame there is a sub-frame that contains a motor to rotate the inner frame around the pitch axis, with respect to the user. Every design we have encountered uses a large monitor to display the game and a joystick for game input. From our, research there is not a lot of competition in the market for these systems.

Our structural design will be similar to other designs but with a slightly different structure. We will build a scaled down demo version unless adequate funding is received. We will provide a unique visually immersive experience by using a VR headset instead of a monitor. This design specification will allow the user to feel more naturally immersed in the game and help prevent the possibility of motion sickness. This will also reduce the weight of the cage and potentially reduce the torque requirements of the motors.

As we are implementing our design on the hardware and software level, we will leverage many building blocks. We will use the STM32f0-Discovery board with an ARM micro-controller and various peripherals including a gyroscope. We will also leverage a motor control PCB we have designed for small-scale single motor control as a starting point for our large-scale dual motor PCB design. On the software side, we will start with many C modules we have previously implemented for motor control, interfacing with a gyroscope through I2C, and communicating through UART. We will use an open-source program called ”Flight Gear” as our flight simulator. Our focus will be on upgrading these building blocks for our specs and connecting all of them together to produce a final product.

IV. Materials

A. Hardware

The hardware framework of the project currently consists of a PVC pipe structure, as can be seen in the figure below. This structure has a base that extends upwards to hold a rectangular piece; this piece controls the pitch axis of the structure, which is represented by a blue line in the figure. This axis's movement is controlled by a motor that will be located in the joint indicated by the light blue circle. The pitch piece also holds another, smaller rectangular piece inside of it. This inner piece controls the roll axis, which is represented by a red line in the figure. The motor that controls this axis's movement will be located in the joint indicated by the pink circle.

The design uses two motors, a small one to control the roll axis and a bigger one to control the pitch axis. In our current implementation, the motors work individually, but not at the same time. At this point, the inner piece of the structure can rotate around the roll axis. However, the bigger motor still needs verification to know if it can sufficiently rotate the structure on the pitch axis. This is a tricky task that requires more time because the pitch piece carries the smaller roll piece and the small motor; this added weight creates uncertainties regarding the current motor's ability to rotate the entire structure.

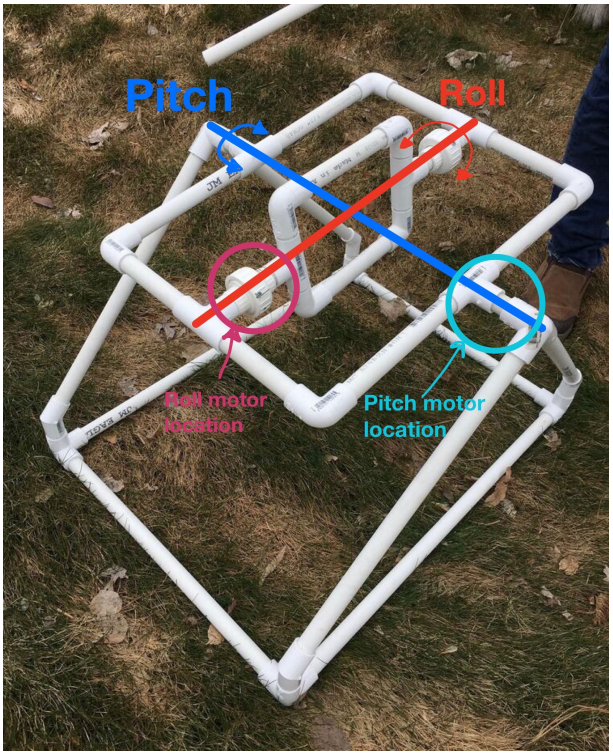

Fig. 1. The beginning of the Flight Simulator structure.

The small inner piece of the structure also currently houses the STM board and represents the place where a user would sit. This allows us to use the on-board gyroscope to get data on the movement of the chair.

We are still looking into designing a custom motor control PCB that will control both motors. We are also discussing other options for unique custom made hardware components.

B. Software

The software portion of our project includes a user interface, sensor and motor controllers, and a FlightGear connection. The user interface is created by connecting to a Putty window and providing a menu to the user via a UART connection. This menu uses keyboard input to send commands back to the STM board. The commands allow for actions like selecting between debug and simulation modes, enabling and disabling the motors, and changing the direction and speed of the motors. Once the commands are received, the program decides what actions need to be done and executes them. This is how the sensors and motors are controlled. In addition, a FlightGear connection is made with a socket in Python. This connection allows us to pull data from the flight simulation, which will be used to decide how the motors should move.

V. Results

The final system can be seen in 2. It consisted of a chair PVC structure, with force sensing resistors (FSRs), motor drivers, stepper motors, an inertial measurement unit (IMU), and a micro-controller. The chair matched it's orientation with the orientation supplied by the video game. The chair output

values for the game developer to allow for a more interactive experience. The game developer was able to see how much force the player was feeling and detect the physical orientation of the player through our systems feedback. The final system was completed and intended to be a plug and play device.

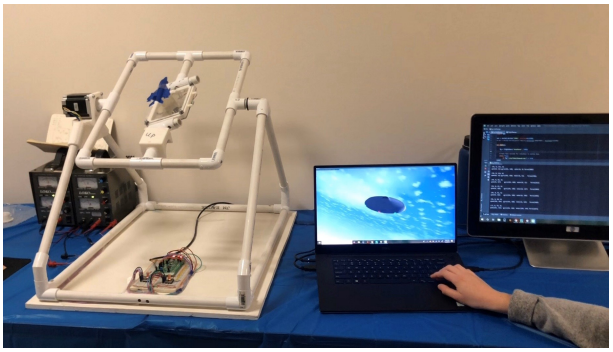

Fig. 2. The Flight Simulator Chair Hooked Up to the Game on Demo Day.

A. Structure & Wiring

The structure of the system was built using 1/2" PVC piping and fittings. This was chosen over metal because it is lightweight, cheap, sturdy, and easy to cut and fit. As the project progressed this proved to be the correct choice. The structure consists of 3 main parts: The base and supports, the outer platform, and the inner platform. The left support was connected to our custom 3d printed motor mount for the large stepper motor as you can see in figure 2. This mount had two openings for 1/2" PVC pipe to fit into, above which the motor was mounted using 4 screw holes. In the center of the mount was a large opening for the motor shaft. The shaft of the large stepper motor was connected to one side of the outer platform. The right support was connected to the other side of the outer platform using a small 24-wire slip ring so that we could wire everything through a freely rotating joint. We did experience some minor issues with the slip ring we used. As seen in figure 3, one end of the slip ring has a very small rotating piece to connect the platform to. We designed a 1/2" diameter connection shaft with one end that had an opening for this small end of the slip ring to fit into. This allowed us to connect the slip ring to the outer platform. We did break one slip ring because the small end is quite fragile. If the piece that connected to the small end of the slip ring moved up or down too much, it would snap the slip ring. This happened to us once, but we were very careful after this incident to ensure that it didn't happen again.

The outer platform rotated to simulate pitch and also contained the inner platform which rotated to simulate roll. One side of the inner platform was connected to the outer platform through another slip ring and custom connection shaft, and the other side connected to the outer platform through a small stepper motor. This small stepper motor was also mounted in another custom 3d printed mount. One end of the mount was designed to slip into the 1/2" PVC connector, and the other side had 4 holes to mount the motor and one big hole for the motor shaft to fit through. The inner platform encased an IMU sensor as described in subsection C, as well as a center weight with 4 legs, each sitting on top of a FSR. The FSRs functionality will be described in subsection D. Figure 7 shows an up-close photo of the center platform.

A custom wiring harness was constructed to wire up the motors and the sensors to the motor drivers and the micro controller. The IMU and FSRs were wired through the inner slip ring which led to the outer slip ring. The sensor wires from the inner slip ring and the small motor were wired to the outer slip ring which had extended wires leading to the control platform that was mounted to the base of the system. Three slip ring wires were used for each small motor wire due to the current limitations on the small slip ring wires. The large motor had extended wires which also led to the control platform. This control platform contained the two motor drivers and the micro-controller as seen in figure 4. A wiring diagram, which can be seen in 5, was created to keep everything organized and make troubleshooting easier. During the wiring process, a multi-meter was used to check for conductivity from one end to the other, and each neighboring wire was checked to ensure that none of the wires were crossed.

Fig. 3. 24-wire slip ring [1].

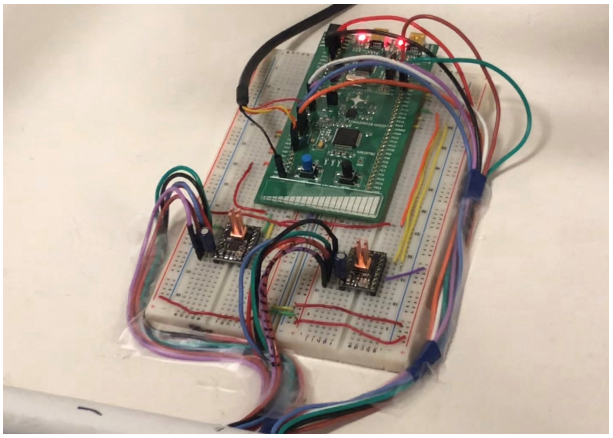

Fig. 4. STM32 Micro-controller with two motor drivers and wiring running into PVC pipe to Sensors and Motors.

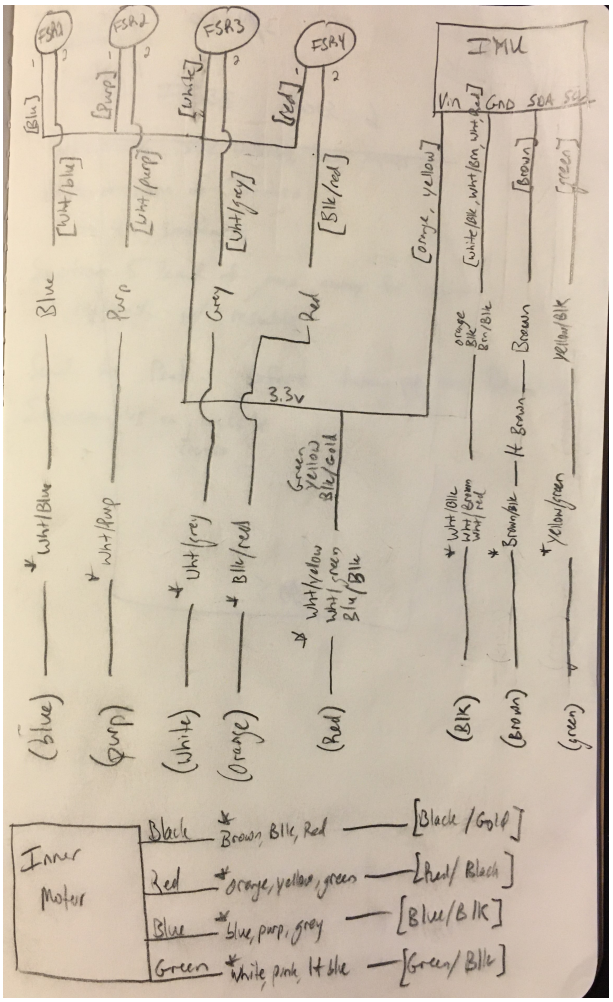

Fig. 5. Wiring diagram from the center platform to the control platform.

B. Motors and Motor Drivers

The motors and motor drivers presented continual challenges throughout the project. Initially we ordered two bi-polar stepper motors and motor drivers which we thought would be sufficient for our purposes. As we tested the motors and their motor drivers, we quickly found that the motor driver for the large motor did not operate as advertised. Our large motor required about 2.5A per phase for maximum torque. Our motor driver was advertised to supply 1.7A per phase without additional cooling, and up to 4.5A per phase with additional cooling. We found that it could not even supply 1A per phase without additional cooling, so we installed a heatsink on the chip and it still had issues. A month before the project deadline, we performed testing with the structure finalized and the motors in place. The large motor driver was unable to provide enough current for the torque that we needed, and the small motor didn't have enough torque even at max current. We ordered two new motors that provided higher torque with less current. We also ordered a new motor driver for our new small motor. From this point on we had no issues with the large motor. We used the same motor driver, but it only needed to provide about 0.6A per phase to the new motor for the required torque. The new small motor and its accompanying motor driver also worked much better overall, but it still didn't have enough torque. The night before the project deadline, we stripped the frame off of the inner platform to shorten the lever arm and to reduce weight drastically. The small motor worked flawlessly from this point on.

The motor drivers can be seen in figure 6. Each one has a current limiting potentiometer to easily tune the amount of current that is output to the motor [2]. It was important to ensure this was properly set so that the current to the motors did not exceed its specifications and cause damage. The inputs for the motor drivers include: Vdd, ground, step, and direction. The driver for the large motor received a 10V power input, and the driver for the small motor received a 20V power input. Each driver had a 100uF ceramic capacitor in parallel with Vdd and ground to protect the chip from LC voltage spikes. The step and direction inputs came from the micro controller telling the motor when to step and in what direction. Each board also had 4 outputs to the motors, one output per motor lead.

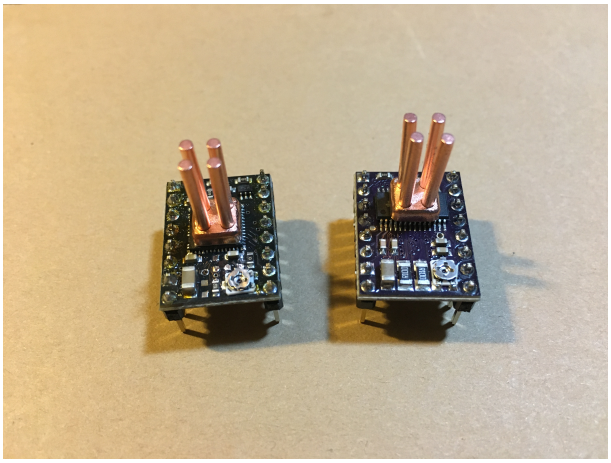

Fig. 6. Motors driver boards. Left one for the large motor, right one for the small motor.

C. Inertial Measurement Unit

The Inertial Measurement Unit (IMU) used was the the BNO055. This IMU was placed in the center of the chair in order to get the most accurate orientation. This is shown in figure 7, where the IMU is in the middle of the Force Sensing resistors, which will be explained in the next section.

Initially we tried to use the Library for the BNO055 supplied by Bosch and Adafruit because it has easy to use functions that provide quick setup and data readings [3]. Upon trial and error, we found that these libraries were designed for an Arduino in C++ and therefore would not work for our project. So, we rebuilt them from scratch using just the data sheet. We also added an initialize function that would not allow the program to continue before the system was calibrated, this way the structure always starts off in a stable position.

In our implementation of the libraries, the system sends requests byte by byte to initialize the IMU and ensure that it is calibrated. For this calibration we set the IMU to NDOF mode, which means that it uses the chip's magnetometer, gyroscope, and accelerometer; using all of these sensors together allows for a better reading on the orientation of the structure. Our system uses this information to provide useful messages to the user on what stage of the calibration the system is in and when it is complete. We also added limitations on the ranges for when the IMU is allowed to interfere with and correct the chair, since we found that the orientation range was limited and not always accurate for the pitch axis.

Overall, this IMU proved to be quite helpful in initializing the system and ensuring that the user begins in a stable state. We had hoped to use it to keep the structure in a more accurate position throughout game-play, however, the IMU would often over-correct for FlightGear causing the structure to move in odd ways. When we saw this happening, we ran tests without the IMU correction and realized that the structure was already responding quite well to the movements in FlightGear.

D. Force Sensing Resistor

We connected the force sensing resistors [4] to the STM32 micro-controller, and we were originally hoping to use the many ADC Channels the STM32 micro-controller had to track the 4 force sensors separately. But, we found that all the channels shared the the same peripheral where you would retrieve the data. The problem we ran into was we had to get the right data from the right sensor at the right time. Originally I tried turning on and off each ADC channel at a time and pull the sensor data when the channel was on. This method was too slow and didn't work. Another attempt was to use a DMA buffer to store the values from each of the force sensors so we could read them off and use them. We had issues getting the DMA working correctly and data from the 4 force sensors would interfere with each other. We tried many different methods to get the DMA working for multiple sensors with no avail. The DMA seemed to only work for one FSR. We did go back and forth between using the ADC Channels alone with the DMA and back to the DMA. In the end we settled on the DMA even though either one would have been fine. Using only one force sensor still helped us see the force felt by the person in the seat as the pitch changed. With this information, the game developer could use it to create the illusion of acceleration and decelleration.

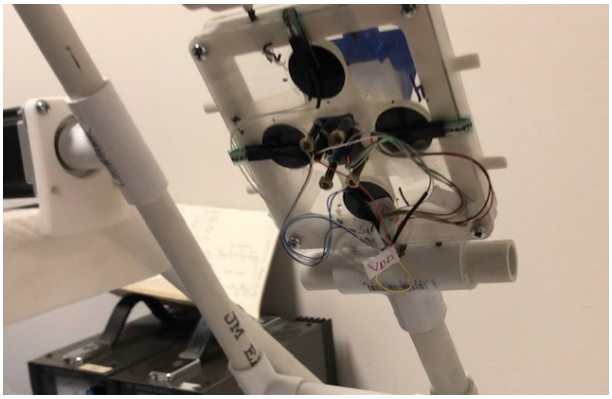

Fig. 7. The chair center with an IMU and FSRs hooked to the bottom of it.

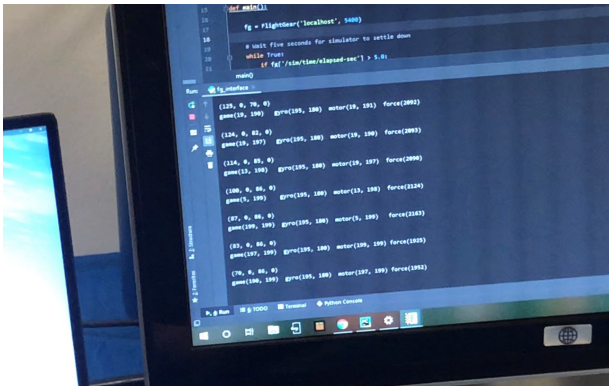

Fig. 8. The packets being sent to and from the chair.
(Top String = Game Data; Bottom String = Chair Data.)

E. Firmware

Our code was written in embedded C. The code ended up being split into 8 main modules, `main`, `FSRs`, `game_parser`, `putty`, `USARTs`, `GPIOs`, `BNO055`, and `motors`. The firmware revolved around a feedback system that would take the desired orientation (pitch and roll) and match the motors to that orientation. The system would start up and use the gyro to detect the orientation of the physical chair and set the actual orientation in steps based on the gryo. The chair would then match it's orientation to desired orientation supplied by the video game.

The chair would send a start signal to the game developer to let them know when the initialization was complete. The game would in turn respond a start byte to the micro-controller of [0xFF]. If the micro-controller parsed a valid roll and pitch value from the game it would respond with updated information about the chair. It would send information about the orientation of the motors, gyro and force felt by the user. This feedback data is shown as the bottom line in the output window in figure 8.

The micro-controller had inputs from the BNO055 that came in as values -90°to +90°for pitch, and -180 to +180

for roll. The code had to convert from these ranges and units to the ranges and units of the motors. The code converted both the roll and pitch to steps of 0 to 200. The input from the game was 0 to 360 and was also converted to steps of 0 to 200. The IMU provided accurate data when the system wasn't in motion but inaccurate data when the motors were moving. This meant that we couldn't fully rely on the gyro. We used the gyro to synchronize the motors to the game and then the motors would keep track of their steps there after. This caused more accurate and smoother movements while keeping the system synchronized.

F. Gaming Interface

Our system received target orientation data over UART from the game. For our demo we chose a software called Flight Gear to act as our game. We chose Flight Gear because it allowed us to poll data every second about a simulated plane. We connected to the Flight Gear API by sending it HTTP requests using Python. The API would respond with the target roll and pitch data requested. Once the target data was retrieved the Python script sent it over UART to the micro-controller.

The game data protocol consisted of a start byte [`0xFF`] followed by 2 roll bytes [`LSB_Roll`][`MSB_Roll`] and 2 pitch bytes [`LSB_Pitch`][`MSB_Pitch`]. Both roll and pitch ranged from 0-365°. The roll and pitch game data could be seen as the top line in the output window in figure 8. On demo day the system appeared to have a second of latency between the game and the chair. This limitation was not from our system but from the open source software, Flight Gear. The time it took to retrieve data from Flight Gear over HTTP requests was very slow. If a game developer used our system they could send values as fast as UART would allow, minimizing the latency to just a few milliseconds.

VI. Conclusion

At the onset of this project, we aimed to produce a proof of concept flight simulator chair. The chair was to be a plug-n-play device that simulates the movement of any object which has pitch, roll, or both pitch and roll axis of movement. We faced many setbacks and difficulties with our structure, motors, firmware, and sensors. As a team, we collaborated and overcame the roadblocks we faced delivered a finished product according to our original design goals.

References

[1] A. Group, "Cap type conductive slip ring," Alibaba Group, Taobao, Taiwan, Dec. 2019. [Online]. Available: https://www.aliexpress.com/item/33033275044.html

[2] P. Corporation. (2019, Apr.) Drv8825 stepper motor driver carrier. Pololu Corporation. Las Vegas, USA. [Online]. Available: https://www.pololu.com/product/2133

[3] "BNO055 data sheet," BOSCH Sensortec, Reutlingen, Germany. [Online]. Available: https://cdn-shop.adafruit.com/datasheets/$\mathrm{BST}_B NO055_D S000_1 2.pdf$

[4] "FSR03CE data sheet," Ohmite, Warrenville, USA. [Online]. Available: https://www.mouser.com/datasheet/2/303/$\mathrm{res}_f sr - 1590094.pdf$