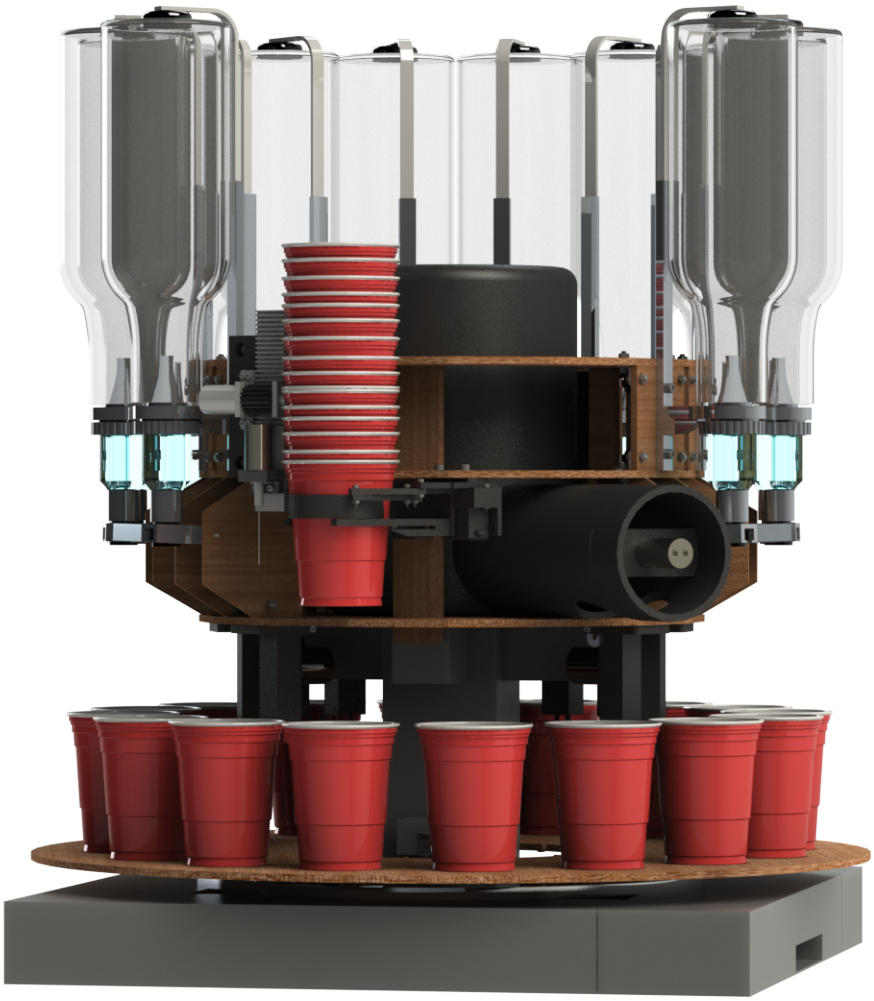
Bartender Robot

MCEN 4228/5228: Mechatronics and Robotics 2 Final Project

Team: Drinkers with an Engineering Problem

May 3, 2022



Contents

1 Introduction

The University of Colorado Spring 2022 Mechatronics class attempted to build autonomous bartender robots as the team project for the semester. The project specifications required the robot to include a user interface for ordering drinks from multiple options and a system for delivering the selected drink to the user. Aside from those requirements, the project was open-ended, allowing teams to choose which class concepts and elements of bartending to focus on.

This report details the design, construction, programming, and performance of the robot for team Drinkers with an Engineering Problem. The team chose to focus on the mechatronic challenge of preparing and serving a large selection of cocktails, with less focus on the user interface. The design pursued was a circular stationary robot that used a "lazy Susan" turntable to move each cup sequentially through various stations for different phases of the cocktail-making process. The stations included 12 liquid dispensers for liquors and mixers as well as a cup dispenser, ice dispenser, and drink mixer, allowing the robot to prepare a wide variety of cocktails from start to finish. The simple user interface was voice-activated, allowing the user to order a drink verbally in an "open bar" setting.

Overall, the envisioned robot functionality and performance was largely but not fully achieved by the time of the presentation. The team was able to get all subsystems functioning independently and coarsely integrated, but several subsystems failed prior to presenting. However, the team plans to continue work on the robot in order to fix the failed subsystems and improve system integration and performance, in effort to achieve its original vision.

2 Design

The design process for the robot was iterative, but most of the major design decisions were made at the start. The team's goal was to create a cocktail robot that could be used in a social setting to make a wide variety of common cocktails, and make multiple drinks simultaneously. The team also wanted to focus on designing and integrating multiple electro-mechanical systems into a complex autonomous device, with less focus on creating a highly-developed user interface.

With these design goals in mind, and strong consideration of budget and available parts, the team conceived its robot design. The design, shown in Figure 1, was a stationary tabletop robot with a rotating "lazy Susan" style table that would move drink cups sequentially through 15 different stations, where the cup would receive the correct combination of ingredients for the user's requested drink. To "take orders," the robot used a voice-activated user interface that interpreted verbal drink commands from users in an "open bar" setting, i.e. without requiring payment or ID. If it recognized the requested drink and had all the ingredients necessary to make it, the robot initiated the drink-making process by dispensing a cup onto the rotating table at the first station. The cup was then rotated to the second station, where it received ice from a central cooler via a mechanical ice dispenser. The rotating table then stopped the cup under any combination of 12 liquid dispensers, which each dispensed a liquor or mixer into the cup from an inverted bottle. Once the cup had received all drink ingredients, it was rotated to the final station where it was mixed by a mechanical mixer, and was then ready for the user to pick up off the table. Sections 3-6 describe the design, construction, electrical wiring, and programming of the robot and all its subsystems in more detail.

The robot was designed to make multiple drinks simultaneously by continuously initiating the drink-making process for new drinks as orders came in, even as previous orders were still in the process of being made elsewhere on the turntable. However, the embedded program enabling this functionality was not completed by the time of the presentation, so the presented robot could only make one drink at a time. Section 7.3 discusses plans for future work to achieve the robot's full intended functionality.

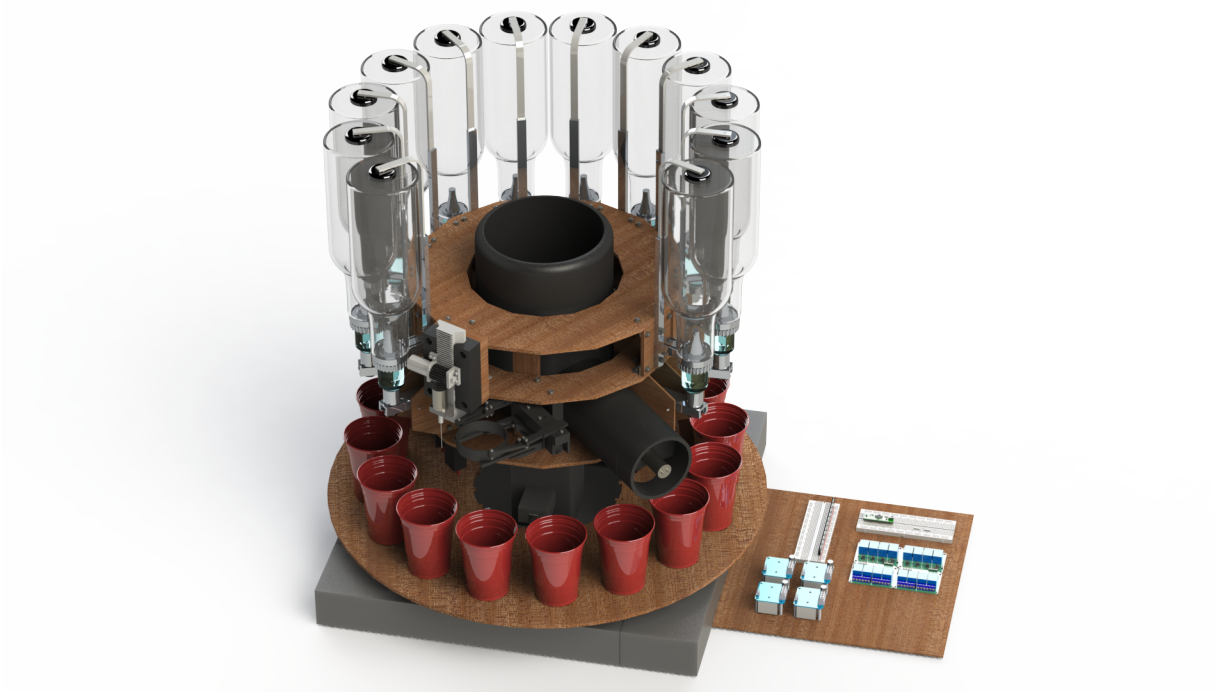


Figure 1: Render of full robot design

3 Structure

The structure of the robot was designed for manufacturability and ease of maintenance. The structure housed 12 identical liquid dispensers and accompanying actuators as well as several other mechanical components, so these needed to be easy to mount. The team also anticipated these components may need to be replaced at some point due to questions about their durability and reliability. Thus, their removal and reinstallation needed to be easy even after the final assembly was complete.

The robot's structure was supported by a 18.5" square base which was made out of wood. Wood was chosen because it was easy to cut to size and was already available to the team, thus not contributing to the team's budget. The square base served as the mount for the rotating turntable and accompanying motor, described in Section 4.1. The base also contained holes on one side for all electronic wiring to run through. Mounted above the turntable was the primary component of the robot's structural design, a central cage supporting all the liquids, ice, and mechanical components used to make the drinks. This central cage was mounted to the square base of the robot via four 9" 2 x 4 boards which were spaced so as to fit through the 12" hole in the center of the turntable. 2" wood screws were used to attach the wood components together.

The structural design chosen for the central cage was a 9" high ribbed structure with a 17" wide 16-sided polygon for its base. The 16-sided polygon would allow 12 different walls for the 12 different liquids that were used, a wall for the mixer, a wall for the cup dispenser, a wall for the ice dispenser, and an extra wall for flexibility or any future attachments. Meanwhile, the cooler holding the ice could sit centrally inside the cage. To make manufacturing easy and economical, this main structural cage was made out of laser-cut 1/8" eucaboard pieces, attached with metal brackets and fasteners. To support the weight of the 12 bottles mounted to the outside, the structure included two wide, flat ribs around the inside of the cage. Two layers of eucaboard were used for the base plate of the cage to better support the weight of the bottles and ice.

3.1 Bill of Materials

Category	Material/Part	Quantity	Cost
Structure & Hardware	1/8" Eucaboard 2'x 4'	3	22.53
	2x4 Board, 9" section	4	**
	2x2 Board, 18.5" section	6	**
	1x2 Board, 14" section	6	**
	18.5" x 18.5" x 0.75" Board	1	**
	M5 Brackets	120	14.99
	M5 x 14mm Bolts	300	14.99
	M5 x 20mm Bolts	50	**
	M5 Nuts	350	9.99
	M3 x 15mm Bolts	20	**
	M3 x 20mm Bolts	48	**
	M3 Nuts	32	**
	#8 x 2" Construction Screws	20	4.68
Liquid Dispensers	Liquid Dispenser and Mount	12	61.58
	3mm x 60mm Rods	12	8.99
	12V Car Door Lock Actuators	12	55.76
	5" Aluminum Lever	12	**
Ice Cooler & Dispenser	2.5 gal Ice Cooler	1	4.35
	Auger	1	22.63
	12V High Torque DC Motor	1	16.34
	4mm to 3/8" Shaft Coupler	1	5.44
	1/8" x 1" x 4.5" Aluminum Motor Mount Bar	1	0.54
	4" PVC Pipe, 1' section	1	0.54
	0.75" PVC Pipe, 1' section	1	0.54
	Silicone Sealant	1	1.09
Turntable	29" Circular Wood Table	1	5.44
	18" Lazy Susan Turntable Ring	1	**
	12V High Torque DC Motor with Mount	1	**
	17mm Spur Gear	1	7.61
	1" x 12" Containment Ring	1	1.64
	Hall Effect Sensor	1	5.99
	Magnets	32	5.76
Cup Dispenser & Mixer	3D Printer Filament Roll	1	4.99
	9V Servo Motors	2	4.99
	12V DC motors Low Torque Motor	2	**
	Milk Frother Stir Stick	1	2.17
Electronics	Teensy 4.0	1	25.00
	Raspberry Pi 4	1	**
	12V 30A Power Supply	1	14.99
	12V to 5V Buck Converter	2	6.99
	16-Channel 12V Relay Board	1	15.99
	74HC595 Shift Register	2	**
	Level Shifter	1	**
	Mosfets	12	**
	80mm 12V Cooling Fan	1	8.99
	Motor Drivers	4	**
	Miscellaneous Connectors	Many	**
	Miscellaneous Wires	Many	**
Total Cost			**357.27**

** Indicates items that were provided to team or available from Mechatronics 1

4 Mechanical Subsystems

Included on the robot were 16 mechanical subsystems for the drink-making process—the rotating table, cup dispenser, ice dispenser, drink mixer, and 12 identical liquid dispensers. The details of each subsystem are described in Sections 4.1-4.5.

4.1 Turntable

The robot moved drink cups through the drink making process via a "lazy Susan" style rotating table, as seen in Figure 2. The table was made from a 29" diameter circular wood table top with a 12" circular hole cut in the center, and was mounted to the base via an 18" aluminum turntable swivel ring. To power the table's rotation, a 12 V DC motor was mounted to the base with a 17 mm, 15-tooth steel pinion gear attached to its shaft. A 300-tooth internal gear was designed in CAD to mesh with the pinion gear, and was laser cut out of two 1/8" eucaboard layers and mounted to the table inside of the swivel ring. The 20:1 gear ratio provided more than enough mechanical advantage for the motor to rotate the table without stalling. The table also included a 1"-high containment ring around the 12" interior hole, sealed to the table with silicone sealant to prevent any liquid leakage into the interior area.

To allow cups on the turntable to be aligned with the 16 drink-making stations, 16 equally-spaced magnets were mounted around the inside of the table's inner ring. A hall effect sensor attached to one of the structure's four wood support posts could then be used to detect the magnets and align the table properly, as shown in Figure 3.

Figure 2: Render of Turntable

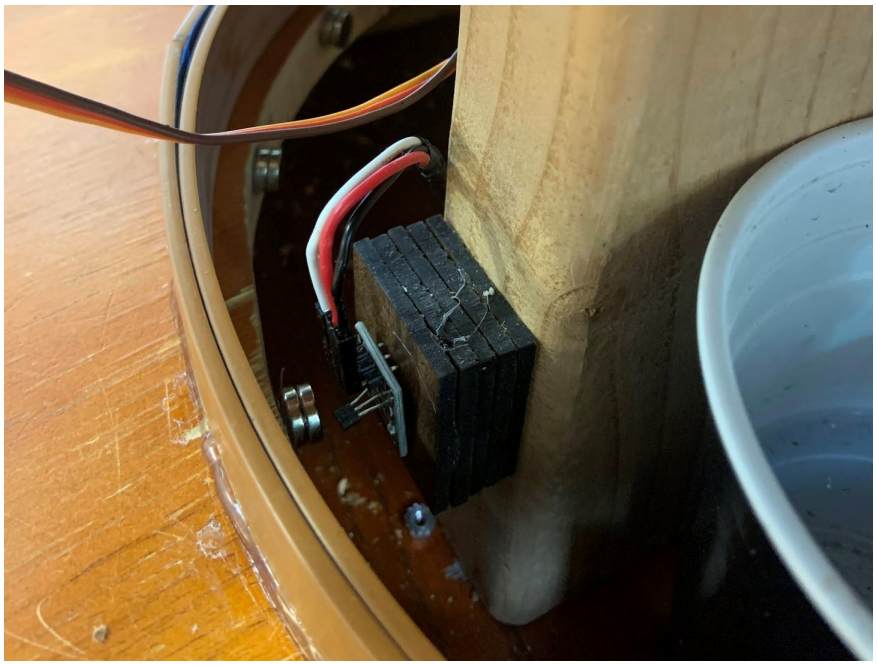


Figure 3: Hall Effect Sensor Mounted with Magnets

4.2 Cup Dispenser

Figure 4 shows the custom built mechanism used for dispensing 200 ml disposable cups onto the rotating table at the first position. The dispenser is actuated by 2 MG995 digital high torque servos moving a 2 link system. The link manipulator moves the cup-stack holder back and forth along a guide-way. The stack of cups rests on a separator, as shown in Figure 5. The gap between the two layers of the cup separator allows the bottom cup's rim to pass through to a wider hole in the bottom layer of the separator which releases it onto the table below, while the rest of the stack remains above the top layer of the separator. After releasing one cup, the linkages move the stack back to the original position.

The dispenser was bolted on to a custom built 3D mount, which was further bolted on to the main structure at the designated position. All the elements of the subsystem were connected using M3 screws and bolts. The servo housing structure and linkage mechanisms were 3D printed using Polyactic (PLA) material on the Lulzbot Mini 2. MG995 servos were chosen as the actuators primarily due to their high torque output (10 kgf-cm at 6V).

Due to limitations of the 3D printer, the linkage element teeth could not mesh perfectly with the servos, which caused the mechanism to fail after some cycles. By altering the design of the arm to house a servo horn, the repeatability of the mechanism was improved. However, using hot glue to make the armature connections permanent on the final day caused the plastic material to deform and the meshing of the linkages failed during final testing.

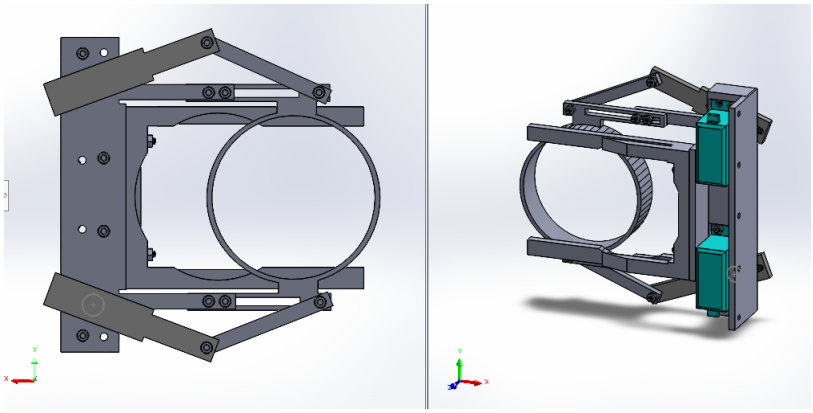


Figure 4: Top and Isometric View of Cup Dispenser Subsystem

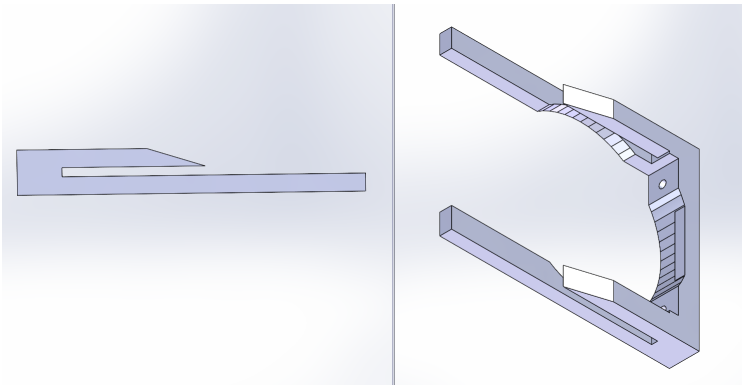


Figure 5: Side and Isometric View of Separator Element

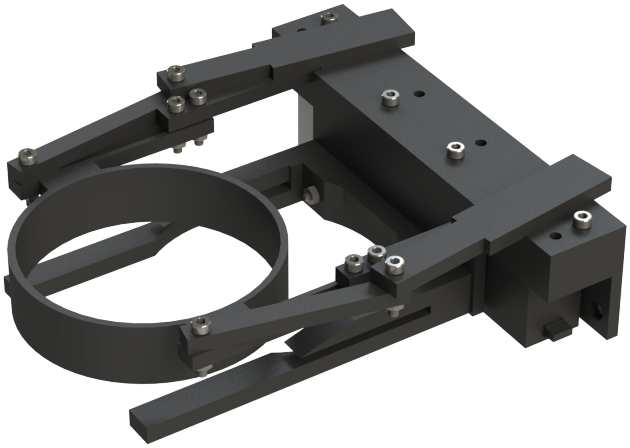


Figure 6: Render of Cup Dispenser

Figure 7: Cup Dispenser Finished Result

4.3 Ice Dispenser

Figure 8 shows a diagram of the robot's ice dispensation system. Ice was stored in a 2.5-gallon cylindrical cooler housed in the center of the robot. A circular hole was cut through the bottom of the cooler's side wall to fit a 4-inch PVC pipe which carried ice radially outward from the cooler to the cup. A standard refrigerator ice machine auger was used to push ice from the cooler outward through the PVC tube. This auger was powered by a high-torque DC econ gear motor mounted inside the outer end of the PVC tube via an aluminum crossbar, which was fit diametrically through the tube. The motor was connected to the auger via a 4 mm to 3/8" stainless steel coupler. The motor, which had a nominal no-load speed of 19 RPM and a stall torque of 2,678 oz-in, was chosen due to its high torque which was necessary overcome the weight and friction of the ice. A 2-inch circular hole was cut in the PVC tube centered 11.3 inches from the center of the robot, allowing ice to fall directly from the tube into the cup when pushed by the rotating auger.

To contain water leakage from melted ice, the seam between the cooler and the tube was sealed with silicone sealant. Drainage holes were then drilled in the bottom of the cooler at it's seam with the 4" tube, and in the bottom of the tube just before the ice exit hole. Two drainage pipes made from 3/4" PVC pipe were then installed to drain water from these drainage holes into a cup sitting inside the turntable below the cooler.

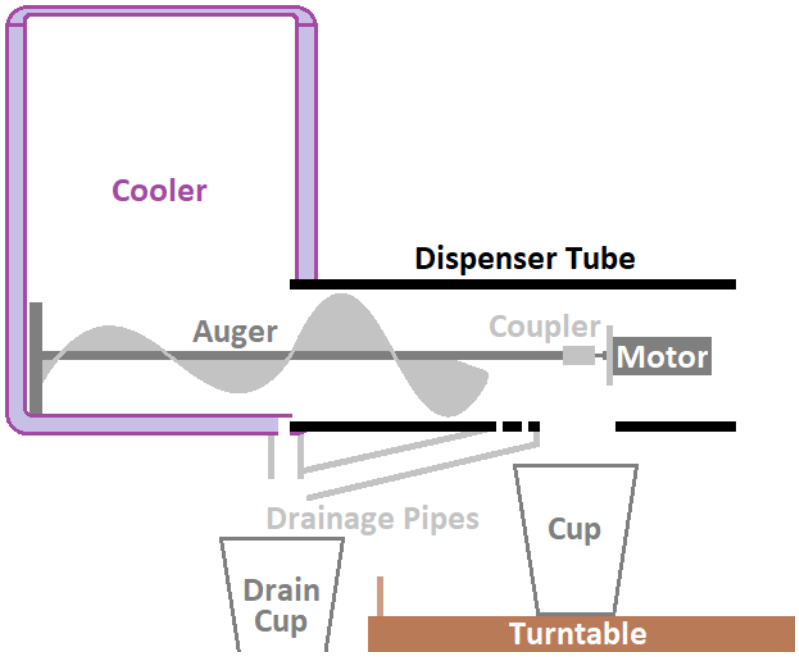


Figure 8: Cross-Sectional Diagram of Ice Dispensation System

Figure 9: Render of Ice Dispensation System

Figure 10: Render of Cutaway of Ice Dispensation System

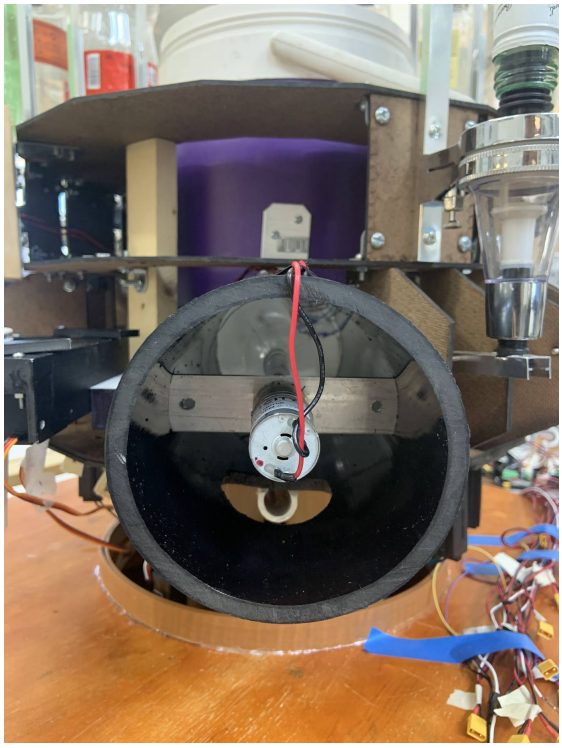


Figure 11: Ice Dispenser Finished Result

4.4 Liquid Dispensers

Figure 12 shows a diagram of the mechanical system used to dispense liquids at each of the robot's 12 bottle stations. Each alcohol or mixer bottle was mounted upside down to a dispenser nozzle, with a spring-loaded arm holding the top of the bottle in place. The nozzles were removable, making it easy to load new bottles by attaching the nozzle to the top of the opened bottle first and then flipping and snapping into place.

The spring-loaded valve on each nozzle was actuated by an actuator-lever system as shown in Figure 12. The actuators used were 12 V car door lock actuators, mounted to the bottom of the structural cage base plate with custom 3D-printed plates. Per specifications, the actuators each provided about 6.7 lbf of force, while the levers needed more force to consistently open the mechanical valve. The pivot point on each lever was chosen to give the lever about a 3:1 mechanical advantage for the actuator to overcome the spring force of the valve, while still allowing the valve enough distance of travel to fully open and seal shut. Each lever was manufactured from an aluminum rod, with a 5/8" hole drilled at the end to fit the nozzle tip.

The nozzles used were designed to dispense exactly one standard shot (50 mL) of liquid each time the valve was opened, and refill the dispenser chamber only after the valve closed. However, it was found through testing that the nozzles did not consistently refill after dispensation, making it possible that minimal or no liquid would be dispensed at a station when the valve was actuated. To solve this problem, the stopper inside each nozzle was removed, allowing liquid to continuously refill the nozzle as liquid was dispensed. Additionally, a straw was added to the bottle side of each nozzle, providing a separate release for air inside the nozzle to travel into the bottle as liquid from the bottle flowed into the nozzle.

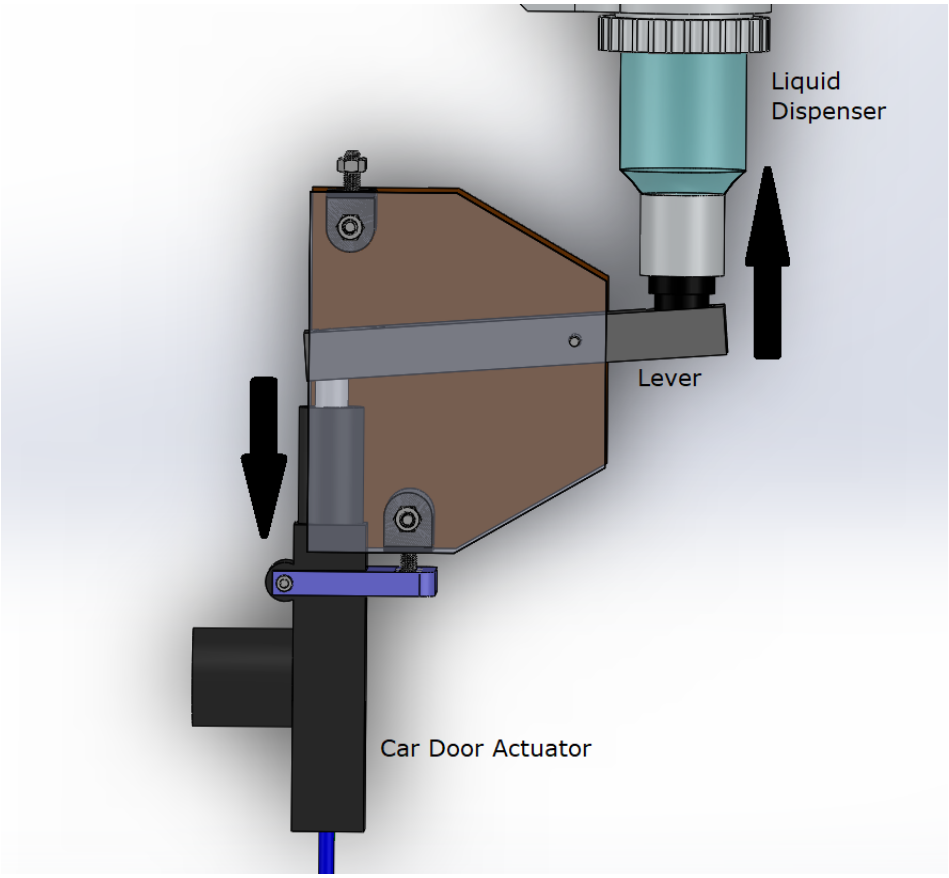


Figure 12: Liquid Dispenser Subassembly

4.5 Mixer

After pouring all the required ingredients into the cup, the final stop of the rotating table has a mixer as shown in Figure 13 for stirring the drink. The mixer was powered by two identical 12 V DC motors. The first motor was mounted on the base plate of the mixer and powered the pinion gear on a vertical rack and pinion, which moved the mixer vertically in and out of the cup. The second motor was mounted to the bottom end of the rack with a stir stick attached to its shaft to stir the drink when lowered into the cup.

The rack, pinion, motor mount plates, and the mixer base plate which housed the rack were all 3D printed using PLA material. To support the weight of the stirrer motor on the end of the rack, all components were made wide and bulky. The motors were mounted onto the base and the rack using M3 fasteners and the base was mounted on to the main structure using M5 screws, after ensuring the stirrer was properly aligned with the cup position.

It was noticed that when the stirrer motor actuates, the counter-torque on the rack would dislodge the mesh between the rack and pinion. By using a modified rack design and improving supports on the pinion, the smoothness of operation of the subsystem was greatly increased. Synthetic grease was also used between the rack and base plate to reduce friction and allow for smoother motion.

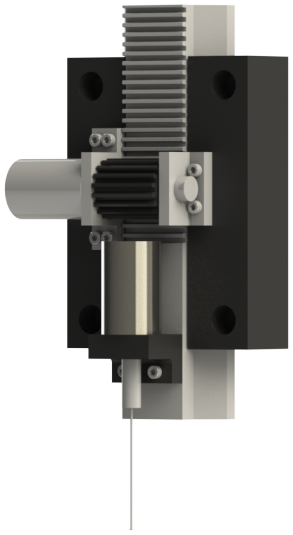


Figure 13: Isometric view of Mixer Subsystem

5 Electronics

The electronics for the robot can be broken into two subsections: power distribution and control. As the name implies, the power distribution system transfers power from the 12V-30A power supply to all of the electrical components. The control system includes the microprocessors and other electronics used to control the device, such as the hall effect sensor, motor drivers, and display.

5.1 Power Distribution

The entire robot was powered from the 12V-30A power supply, which was connected to 120V AC. The motors, actuators, and relays all needed 12V, so they were directly powered from the supply. The microcontrollers and servos require lower voltages, so buck converters were used to step the voltage down. One buck converter

was adjusted to 5V to power the Teensy, Raspberry Pi, and PC fan. A second buck converter was set to 9V to provide power to the servos used in the cup dispenser. The full schematic is shown in figure 15 below.

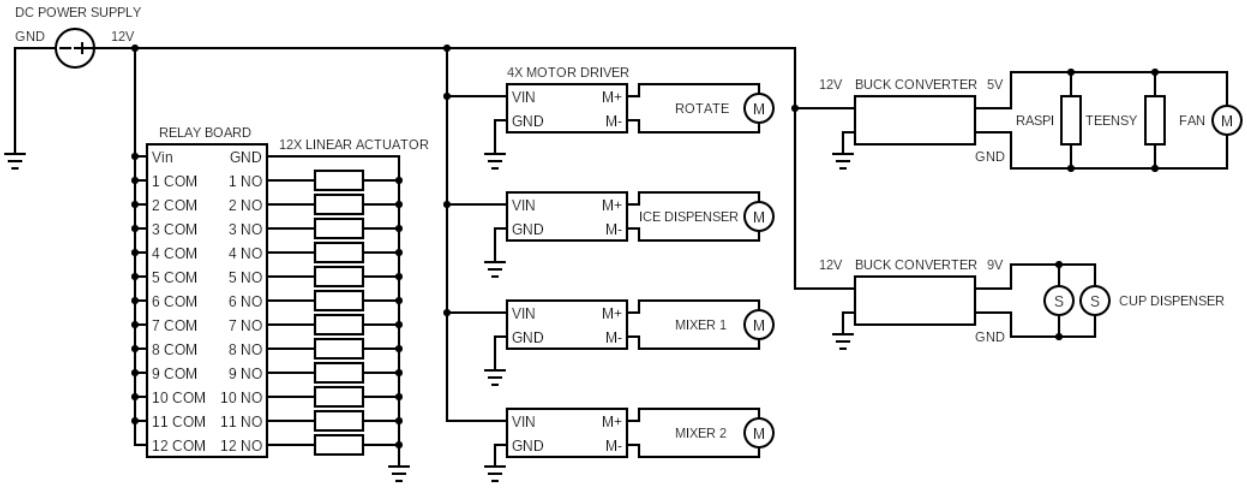


Figure 14: Schematic of the Power Distribution for the Robot.

The relay board was used to send power to the actuators based on inputs from the Teensy. The common terminal of relays 1-12 were connected to the +12V power supply. The normally open terminal of each relay was connected to the positive lead of the corresponding actuator, and the negative lead of the actuator was connected to ground. In practice, male XT30 connectors were installed after each relay and female XT30s were installed on each actuator connection. Then extension cords were made with a male and female end to connect the relays to the actuators. While this took longer than directly wiring everything together, it made it much easier to assemble or remove individual components as needed. It also allowed the actuators to be wired to the relays after they had been installed.

The motor drivers were also connected to the 12V supply using a single XT30 connector which then sent 12V and ground to each driver. Similar to the relays, a male XT30 was connected to the motor driver output and a female XT30 was connected to each motor's positive/negative leads. An extension was made to connect from the male XT30 on the drivers to the female XT30 on the motors so that they could be connected after assembly.

Both buck converters were connected to the 12V. The input and output of the buck converters was also connected to an XT30 so that a buck converter could easily be replaced. A multimeter was attached to the output of the buck converters and their potentiometers were adjusted until one read 5V and the other read 9V. The XT30 on the output side of the 5V converter split into three sets of wires to power the Teensy, Raspi, and fan in parallel. The other buck converter powered the two servos in parallel.

5.2 Microcontrollers and Integration

The Teensy 4.0 microcontroller was used to control the motor drivers, relays/actuators, and servos. A Raspberry Pi 4.0 (RPi) was also used to handle the voice activation and display. The two microprocessors were interfaced using serial communications so that the RPi could send drink orders to be carried out by the Teensy.

Four motor drivers were used to control the motors being used for the table roation, ice dispenser, and mixer. The VCC and the two enable pins on each driver were connected to the 3.3V output of the tTeensy. PWM signals were sent to the forward and reverse PWM pins in order to control the speed in each direction. The two servos' signal lines were also connected to PWM pins on the Teensy, which sets the position of the servos via a PWM frequency.

A hall effect sensor (HES) was used to track the position of the table as it rotated. The HES output a 5V high logic signal, but the Teensy needed 3.3V logic. Using a level shifter, the voltage was stepped down from 5V to 3.3V so the signal could be read.

In order to activate a relay the signal pin must be pulled to ground. This was achieved this by using shift registers in order to generate the 12 separate signals corresponding to the 12 relays. Each signal was sent to the gate of a MOSFET through a 220 Ωresistor. The drain of the MOSFET was connected to the signal pins on the relays and the source of the MOSFET was connected to ground. When activated, the MOSFET pulls the relay signal to ground, actuating the relay, and in turn triggering the actuator to dispense liquid.

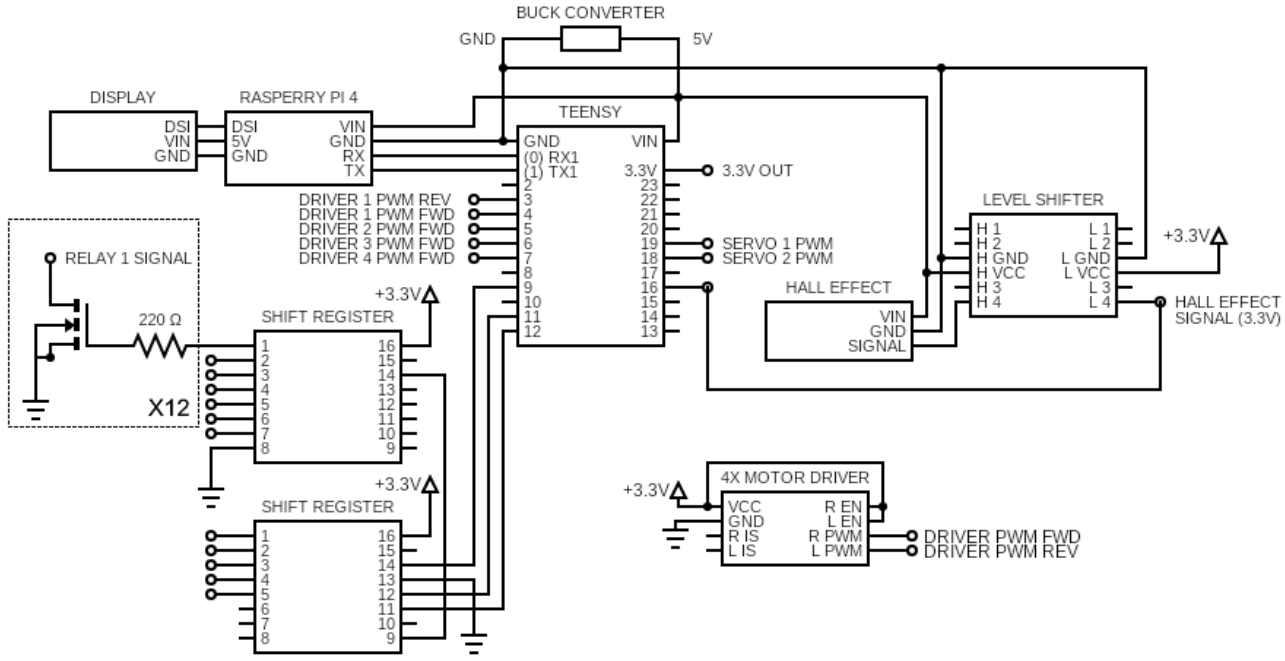


Figure 15: Schematic of the Control System for the Robot.

6 Programming

6.1 Raspberry Pi

The Raspberry Pi, programmed in Python, served three main purposes: listen to the voice commands of the user, compile drink recipes into machine code, and send that machine code as data packets to the Teensy 4.0 microcontroller. These three tasks were split up into three packages imported into a main Python script. The outside data needed was a CSV (comma separated value) list of all drink recipes consisting of the drink names, their required mixers, and the amount needed of each mixer. Also provided was a CSV of the current drinks the bartender has, in their order.

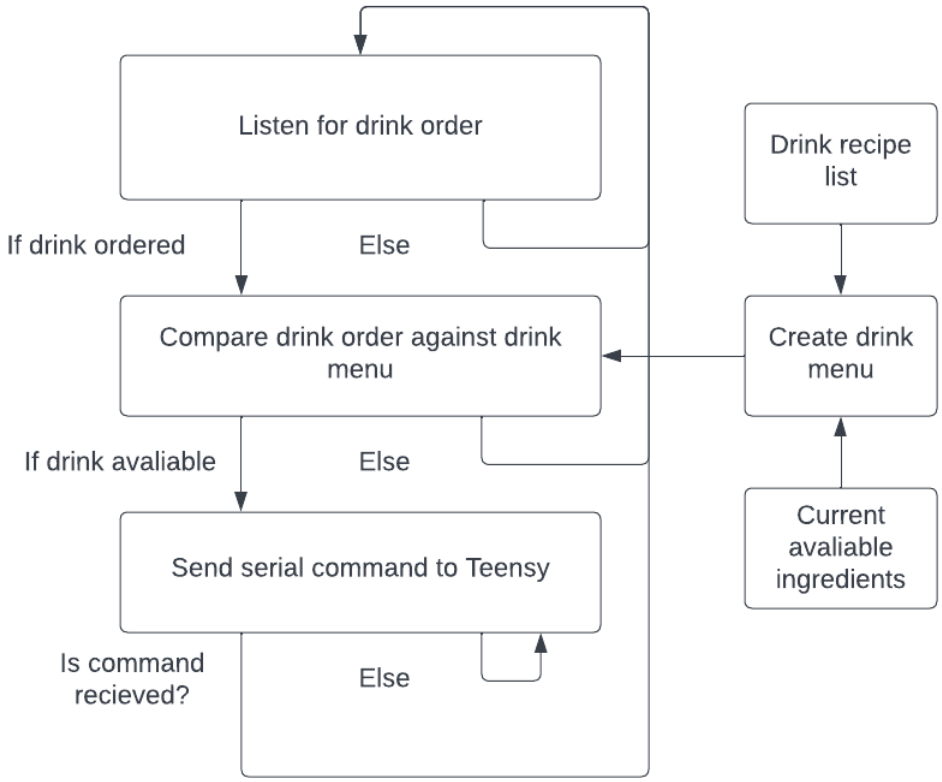


Figure 16: Raspberry Pi Code Flowchart

The voice recognition was done by a packaged called Picovoice. Picovoice uses the Porcupine wake word detection engine and the Rhinoceros speech intent engine. An AI model had to be trained for each of these engines to be interfaced with the Python package. This was done through the Picovoice servers, making the process easy and quick, and validating the choice of using Picovoice. After the AI models were trained, a package was developed from some framework code released by the developer. The PicoVoice class is a thread, letting it run continuously in the background awaiting voice commands. When a drink order is detected, that drink order is saved until it is detected by the main Python script.

The menu creation package has the task of parsing the two given CSVs to create a menu of the drinks that can be currently made. The full recipe list is cut down so that only the available mixers are used. This shortened list is then ordered in the order the mixers are inserted in the physical bartender. This is the drink menu. This drink menu can be compared against the recipe list to determine if a drink can be made or not. This functionality can also be used to print every possible drink that can be made at the moment. This process uses the NumPy library so it takes as little time as possible and would scale well with a larger drink recipe list.

The serial command package utilizes a UART serial connection between the RPi and Teensy for serial communication. The code takes in a recipe from the menu package, and transforms it into data packets which are sent sequentially to the Teensy. This process can be seen in Figure 17. An initial recipe is given, where the additional mechanism commands are added on to it. These commands are then read one by one and sent as data packets to the Teensy. This interface between the two devices is shown in Figure 18, where the RPi sends a packet and waits until the Teensy responds with “true” to continue. The structure of the data packets sent are represnted in Table 6.1.

The main script of the Python code executes the three mentioned packages. It has the added responsibility of compiling the ordered drinks in a queue so that if a drink is ordered during another’s creation, it will begin to be made directly after the first drink is finished.

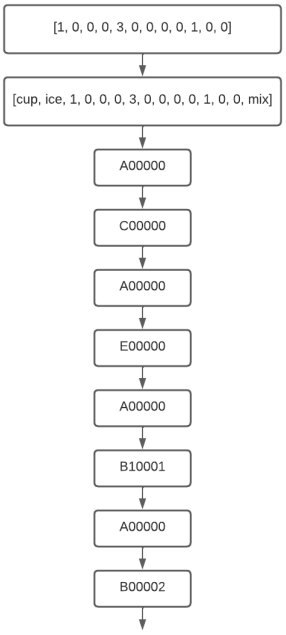


Figure 17: Data Packet Creation

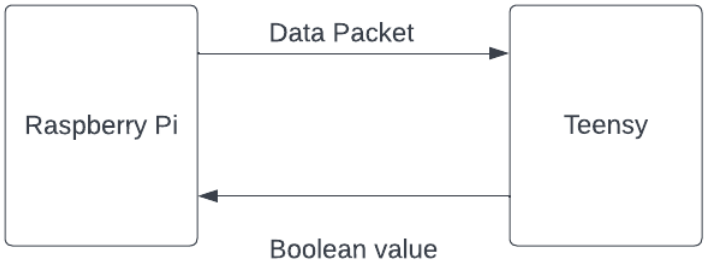


Figure 18: Serial Interface Between Devices

Data Packet Format

Command	**Data Packet**	**Subsystem**	**Value**	**Identifier**
Rotate Table	A00000	A	None	None
Dispense Drink	B10001	B	Number of drinks (1.00)	Drink Dispenser (1)
Dispense Cup	C00000	C	None	None
Mix	D00000	D	None	None
Dispense Ice	E00000	E	None	None
Error	X00000	X	None	None
Ping	Z00000	Z	None	None

6.2 Teensy

The Teensy 4.0 microcontroller is programmed in C++. The Teensy is compatible with the Arduino IDE, but this was avoided and a VSCode extension called PlatformIO was used to structure the code. This allowed for more readability and structure to be given to the code with the downside of it being incompatible with running on the Arduino IDE. The Teensy acts as an extension of the RPi. It waits until a serial packet is received, parses the packet to decipher what command it should execute, and once the command is executed it sends a "true" Boolean over serial back to the RPi to allow the RPi to continue. The Teensy code is kept as simple as possible on purpose to allow it to run quickly and work within the constraints of a microcontroller.

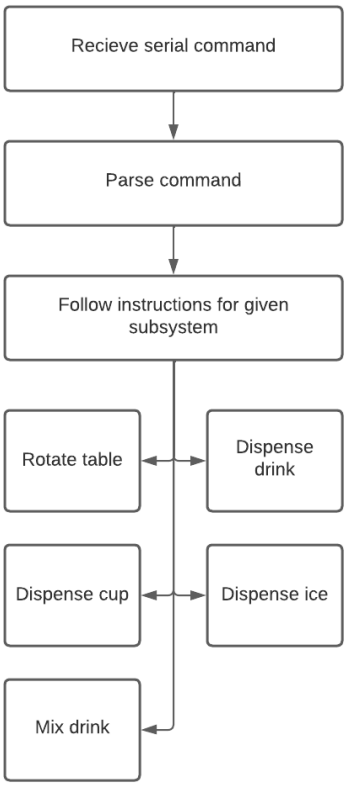


Figure 19: Teensy 4.0 Microcontroller Flow Diagram

Each individual mechanism is given its own class. The only classes that are inherited by other classes are the shift register, hall effect sensor, motor drivers, and sensors. The other classes are named for their representative mechanism. All classes other than the Servo class were developed specifically for this project. The Servo class is the Arduino Servo library but drastically cut down for simplicity and reduction of bloat.

Most of these classes are simple in their functionality and simply act as drivers for the actuators they are connected to. The most complicated class to understand at first glance is the shift register class. The shift registers are used to control the 12 drink actuators and they were chosen for the future ability to actuate multiple drink dispensers at once. The shift registers utilize SPI (Serial Peripheral Interface) to be programmed. A bit of data is "shifted" down the shift register one by one until it is in its desired location. This allows he register to act as a 11bit array" and store a combination of "on" signals to actuate the drink dispensers. The code is implemented to only be able to turn on one bit from the register at a time by reading a preprogrammed hexadecimal representation of the bit array (for example: 00000100 is 0x04 in hexadecimal).

7 Results and Conclusion

7.1 Performance

The robot was partially successful in its ability to make a drink at the time of the runoff demonstration. The structure of the bartender was sturdy and was able to support 12 full bottles of liquid, a full ice cooler, and all mounted mechanical components. The code and electronics, as well as the rotating table and liquid dispenser assemblies, all worked as intended. While real alcoholic beverages could not be used during the demonstration, the system was able to move a cup through most of the drink-making process.

While the team got all subsystems working before the presentation, several subsystems failed prior to demonstration. The cup dispenser stopped working because of a stripped joint between one of the servos and its connecting linkage. The cups used were also slightly too large for the cup support ring, resulting in friction between the ring and the cups that prevented cups from dispensing consistently.

The ice dispenser was fully functional the day before the presentation, provided that pellet or nugget ice was used since larger ice cubes sometimes jammed the auger. However, the ice dispenser's motor was inadvertently wired backwards during one trial run. This resulted in the motor and auger running backwards and jamming the ice into the cooler. The resulting torque on the motor was excessive enough to stall and damage it with no spare available.

The mixer assembly was also fully functional, but reaction between the paint used on the mixer base, and the grease used between the base and the rack for lubrication, seemed to cause the rack to stick in place when the mixer was not being run consistently. This resulted in only the stirrer spinning during the first demonstration run while the rack and pinion could not move the mixer vertically. However, the rack was subsequently freed from the base and re-lubricated, and the mixer was fully functional after that.

Figure 20: Fully Assembled and Finished Bartender

7.2 Lessons Learned

The team learned a wide variety of lessons from working on this project. One major lesson is that assembly, integration, and testing of a mechatronic system generally takes longer than expected, especially for a system

as complex as this robot. The team had considered simpler robot designs with fewer subsystems and limited functionality, but wanted to pursue a robot that would actually be usable outside the classroom to make complete cocktails. This always meant that it was possible some subsystems or functionality would not be completed, and a simpler version of the robot would have to be demonstrated. Even so, the team could have had all subsystems fully working and integrated had they been completed further in advance, with more time for testing, fixing issues, and ordering replacement parts if necessary.

Among additional lessons learned are that power requirements and PWM signal direction needed to be documented for integration to go smoothly. Spray paint proved to significantly hinder moving mechanisms and should not be applied to 3D printed parts that require tight fits. While difficult on a tight budget, spare parts ideally would be available in advance for immediate replacement when parts break.

7.3 Future Work

While some of the functionality originally envisioned by the team was not achieved by the time of the class demonstration, the team intends to continue working on the robot to achieve full functionality and pursue all of its "stretch goals." The most immediate work planned is get the broken subsystems functional again by replacing the ice dispenser motor and the servo linkage joints on the cup dispenser. Several parts on the cup dispenser and mixer may also need to be re-printed to eliminate friction caused by the paint and achieve consistent performance. A mechanism to slide cups off the table after being mixed is also a potential upgrade to the robot.

The robot was also originally designed to be an assembly line drink maker, which would continually start new drinks as orders came in and work on fulfilling existing drink orders simultaneously. To allow multiple drinks to be made at once, the robot code would need to be enhanced and a scheduling library would need to be developed for the Teensy. This is not a native feature of the Teensy, but it has a Cortex M7 processor which supports threading and would allow this functionality to be implemented. A GUI was also envisioned as a "stretch goal," which would allow the user to order available drinks from a touchscreen device. The GUI was intended to be developed in React Javascript with the Python code acting as the back end to the GUI. The team intends to continue working on implemeting this GUI, the framework of which can be found on the Github linked to in Appendices A and B.

References

Appendix A: Raspberry Pi Code

See the link below for the Python code used on the Raspberry Pi.
`https://github.com/nicolasgarzione/robot-bartender/tree/main/RPi`

Appendix B: Teensy Code

See the link below for the C++ code used on the Teensy.
`https://github.com/nicolasgarzione/robot-bartender/tree/main/Teensy`