Holcombe Department of Electrical and Computer Engineering
Clemson University

Final Report:
Motor Driven System for Automatic Latte Art

Spring 2023

Team 4 – Lazy Latte Art

Jacqueline Bendziewicz
Leah Brown
Elise Ferkler
Derrick Joyce
Luke Swetonic
Mariah Tam

Executive Summary

Introducing Lazy Latte: the automated latte art pouring machine. Lazy Latte is an appliance capable of creating latte art with the push of a button. The current edition of Lazy Latte supports two latte art designs: heart and snowman. In order to successfully operate the machine, it is incredibly important for the user to utilize a cup of espresso and a cup of steamed whole milk. The type of milk, consistency, and amount directly affects the quality of the produced latte art. Lazy Latte features two cup holders: one for the steamed milk, and the second for the espresso. Through a sequence of pre-defined commands, Lazy Latte will create an aesthetic art design on top of the espresso.

To enhance the user experience, Lazy Latte comes with an easy-to-use Graphical User Interface (GUI) which provides a graphical display for selecting the latte art designs. Images of the heart and snowman are shown to give an expectation of the latte art when it is finished pouring. Buttons on the GUI display makes ordering your custom latte straightforward and efficient. The display not only improves the quality for the user, but also prevents the user from needing any technical background to operate the system. The heart and snowman designs were chosen for simplicity and recognizability. Whether it is the morning of New Years or the morning of Valentines Day, Lazy Latte is here to help you achieve your coffee dreams.

Since Lazy Latte only supports three axes of motion, more complicated designs are not currently supported. In future iterations of Lazy Latte, it will be a focus to increase the complexity of available designs. Current challenges of operating the system centers around the consistency and temperature of the steamed milk. Unfortunately, variability in these measurements directly affects the quality of the latte art that is produced. This is because Lazy Latte creates an initial canvas which requires a different consistency than the actual froth that creates the design on top of the canvas. Otherwise, Lazy Latte is a fully functional, automated latte art maker.

Abstract

This project resulted in a system that creates two designs for latte art. The user was able to pick the desired design - a heart or a snowman - and the system would execute a specific set of commands to create the design. These designs were replicated with a decent consistency, but sometimes the designs looked different or failed completely. This was due to inconsistency in materials used and not due to any fault of the hardware or software used. The espresso and steamed milk are both made by a separate Nespresso machine, with the latte art system pouring the designs. The way the latte art machine works is by having one cup that can rotate up to 45 degrees, which holds the espresso. The other part of the machine is a platform that can move up and down as well as side to side. On top of this platform is a

spouted cup, which holds the steamed milk. This cup can rotate up to 135 degrees. Timed movements pour the milk into the espresso to design the art in the coffee.

1. Introduction

The objective of this project was to create an automated latte art design machine. This process is incredibly intricate and artistic. Few individuals can master the technique and art manually. To overcome the human error and difficulty of learning this task, we created a machine to replicate these latte art designs on a consistent basis. This project would be a large benefit to every coffee shop around the world. Latte art is something that customers enjoy, but it takes many hours to train employees to do this task. Having an automated latte art machine would benefit these businesses to consistently sell lattes with beautiful art.

The formulation and design process in this project covers a number of areas. On the software side, a knowledge of Python was necessary for coding this project. Additionally, motor control and hardware design were important in the building of our machine. General coding knowledge has been gained throughout many classes in the curriculum, especially for the computer engineers. Motors are discussed in various classes such as Power and in Linear Control Systems. We also needed knowledge about various Pulse Width Moduation's (PWM) which we initially covered in Microcontrollers. This project was an exciting opportunity for our group to combine our hardware and software skills on a creative, unique, and useful project.

2. Problem Definition

The idea for our machine was born because of the difficulty in creating consistent latte art manually. Creating latte art designs is an art that takes many months to master. With an automated machine that can produce consistent latte art designs, many coffee shops would be intrigued. The latte art machine can consistently produce 2 designs, with potential to add in more. In order to run, the machine needs one cup of steamed milk, and one cup of freshly brewed espresso. The steamed milk must be at a warm temperature and frothed to a medium setting. The project is powered by a power supply with 12V and 6V sources for the Raspberry Pi and hardware components. The project is fairly mobile with the entire latte art machine cased in one box with a few extraneous pieces on top. The other components are the power supply and monitor, which can be moved fairly easily. The design itself is robust in that all of the critical electrical components are covered by the housing. In order to prevent spilling from the latte art, there is a container underneath the cups to catch any liquid that may escape either cup.

Our choice to add this enclosure around the electronic components stemmed from the problem of potential damage to our central electronics area because of the use of liquids in our project. Additionally,

we did not want any movement to disrupt any electronics or someone to accidentally move something as they were working on the project. This enclosure helps protect our electronics system. In the same light, we added the container on top of the box to catch any potential spills from doing the latte art.

To run a latte art design, the machine takes roughly 40 seconds to 1 minute from start to finish. The machine only takes about 10 seconds longer than a person to perform latte art designs because of system and motor initialization. The machine is otherwise efficient and does not waste any milk or coffee in the process of making designs. Successful runs of each latte art design, result in no spills and a clear shape for each selected design.

The design of the motor and actuator system was refined and specified exactly for our needs. To start, the coffee cup with espresso needs to move on only one axis, to tilt the cup. The cup with the steamed milk needs to move in three directions throughout the pouring process: vertically, horizontally, and angularly. The only requirement for the espresso cup is to move angularly in which specific angles are specified in the software depending on the design being created.

In the Python code, we ran 4 different threads for each different hardware component. This allows each hardware component to run simultaneously. The GUI was designed for customers to easily place their orders. It features pictures of each possible design with a start and exit button. The simplicity and clarity of this design is meant to make it easier for customers to see their options and order their latte accordingly.

The requirements we laid out for this project specified how the hardware components would move. As specified in the PMP, the horizontal actuator can move 3-4 inches, and the vertical linear actuator 4-5 inches. The espresso cup can rotate from an initial 0 to 45 degrees. The milk cup can move from 0 to 150 degrees. Each of these measurements was an intentional guideline for achieving each of our desired latte art pours.

3. Final Design

The Lazy Latte Art system consists of a Raspberry Pi , two linear actuators, and two DC servo motors. The servos are powered by 5V from a DC power supply, and the actuators are powered by 12V. The servos receive a signal directly from the Pi. The Pi signal for the actuators is sent to a motor driver along with the power. The motor driver then connects to the actuators. A motor driver is required because the pi and the motors have different operating voltages. Figure 1 shows the final implementation of the system with white boards covering and protecting all the equipment. Figure 2 shows the block diagram, which did not change from the PMP. The two servos and the two actuators move together to create the latte art. All of the control for the hardware is controlled by the code running on the Pi; there is no hardware logic in the Lazy Latte Art system.

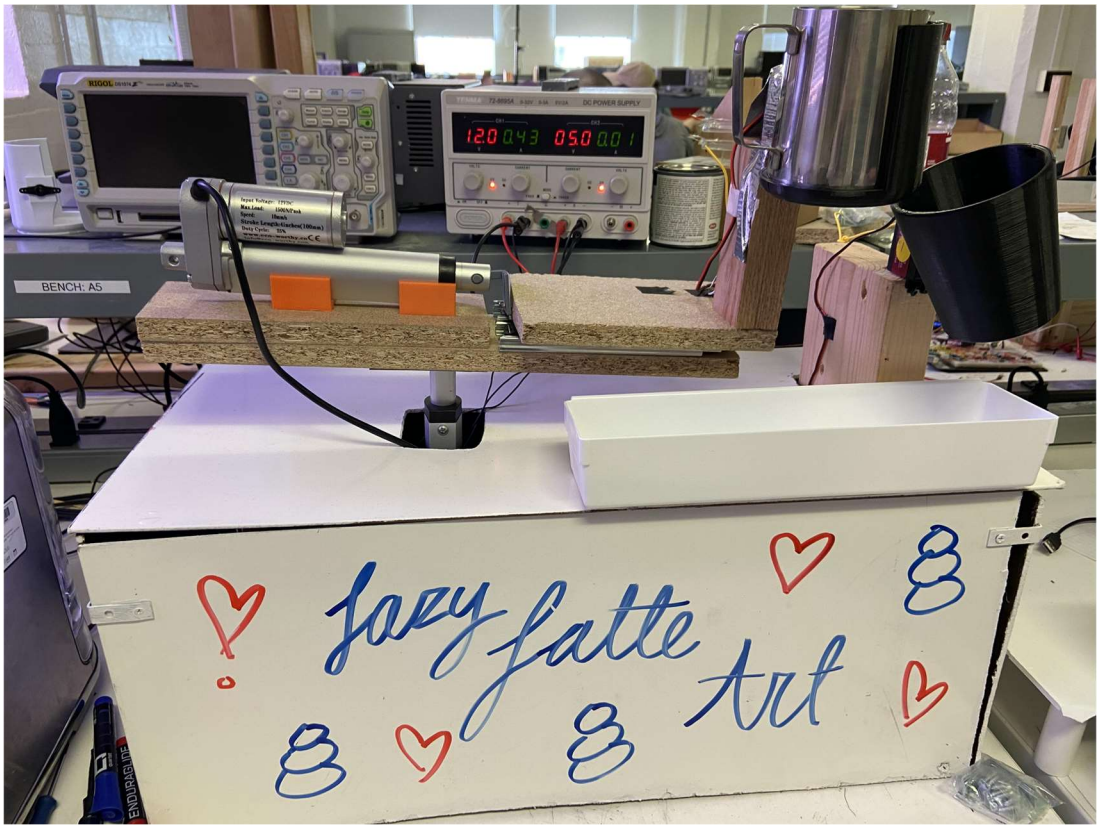

Figure 1: Final Implementation

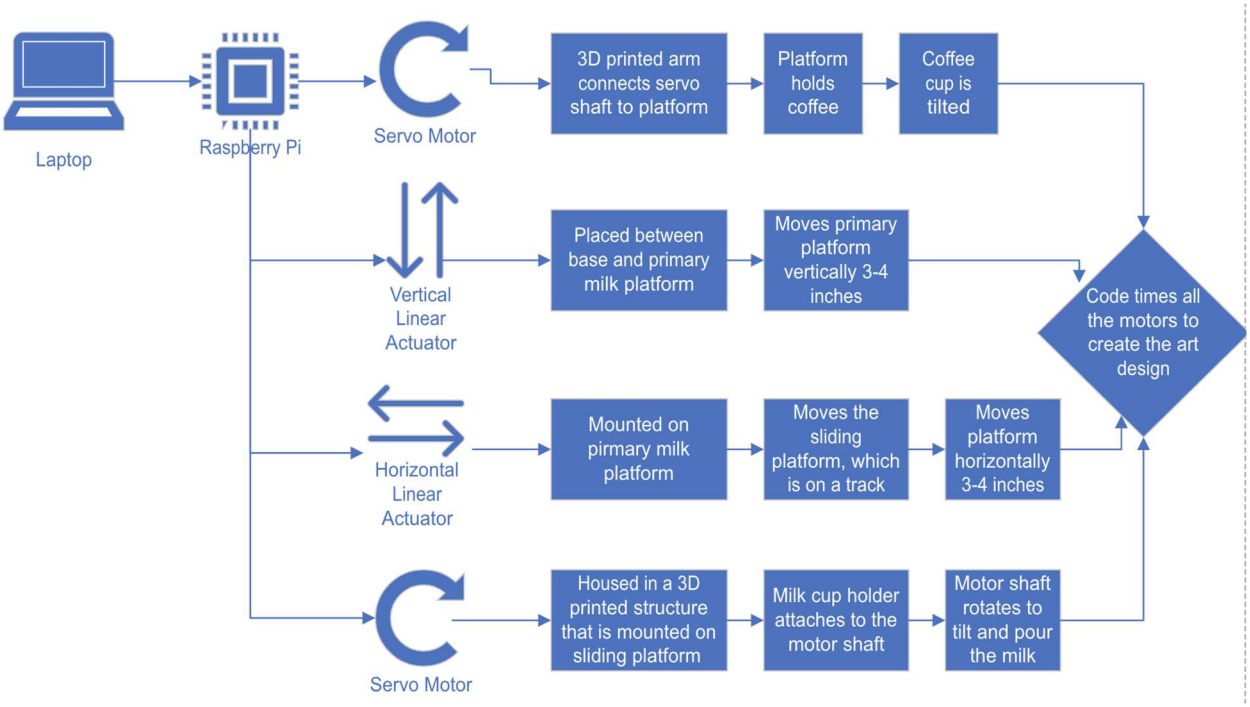

Figure 2: Block diagram

The final graphic user interface is shown in *Figure 3: GUI* and the associated flowchart is shown in *Figure 4: GUI Flowchart.* The GUI consists of two separate screens: the 'Main Menu' and the 'Loading Screen'. The 'Main Menu' consists of five different buttons for user interaction: 'Design One', 'Design Two', 'Start', and 'Exit'. When a 'Design' button is pressed the system is triggered to store the selected design for future use. If no 'Design' button is pressed, the system defaults to 'Design 1'. The design to be executed can change until the user selects 'Start'. Once 'Start' is selected, the program switches over to the 'Loading Screen' and the code that controls the movement of the components is run based on the design the user selected. The 'Loading Screen' shows until the hardware program is finished running. Once finished, the program returns to the 'Main Screen'. This process repeats until the user selects the 'Exit' button which terminates the program. The GUI and *Figure 4: GUI Flowchart* both slightly differ from what was written in the PMP, due to the elimination of 'Design 3'. Besides the reduction to only two design buttons the GUI has remained consistent with the PMP.

Figure 3: GUI

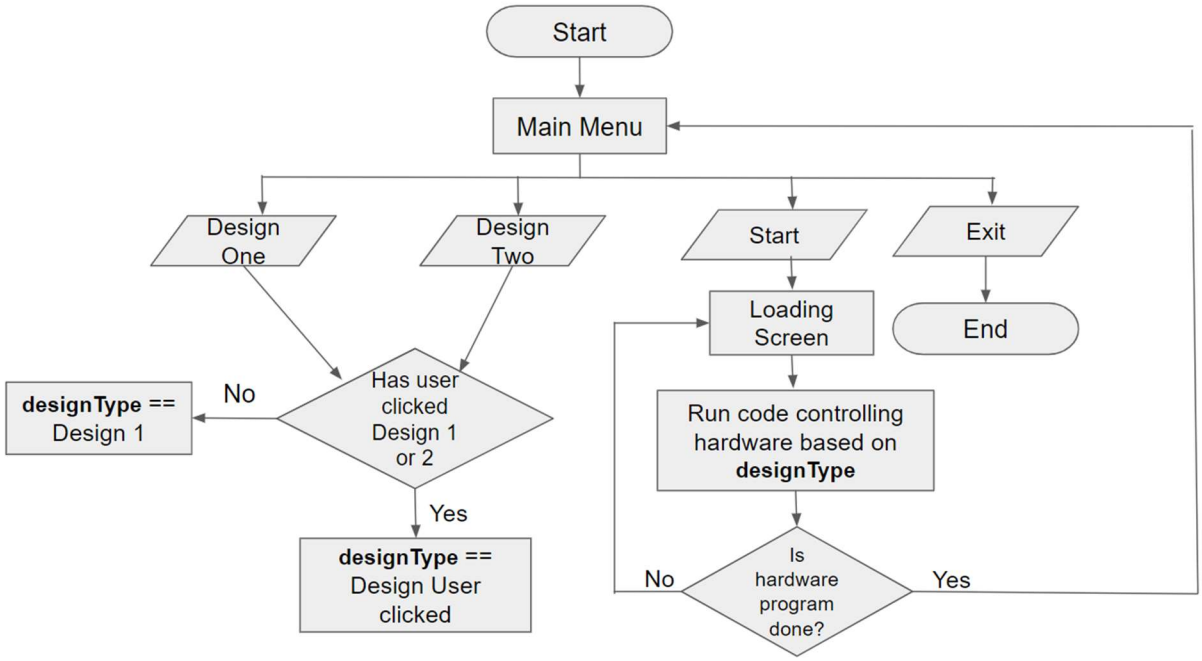

Figure 4: GUI Flowchart

The component control Flowchart is shown in *Figure 5: Component Flowchart.* This flowchart is the main concept behind the movement of the hardware when the design code is run. When the program starts, every component remains at its initial position until the hardware program receives the start signal from the GUI. Once the signal is received, four threads are created, one for each of the four components. The four threads move independently from one another, until they receive an exit signal. Once the threads finish their processes, they return back to the same initial position. Since the PMP, the component flow chart has been altered. The once previous separate 'Motor Flowchart' and 'Actuator Flowchart' have been combined into *Figure 5: Component Flowchart,* due to redundancy.

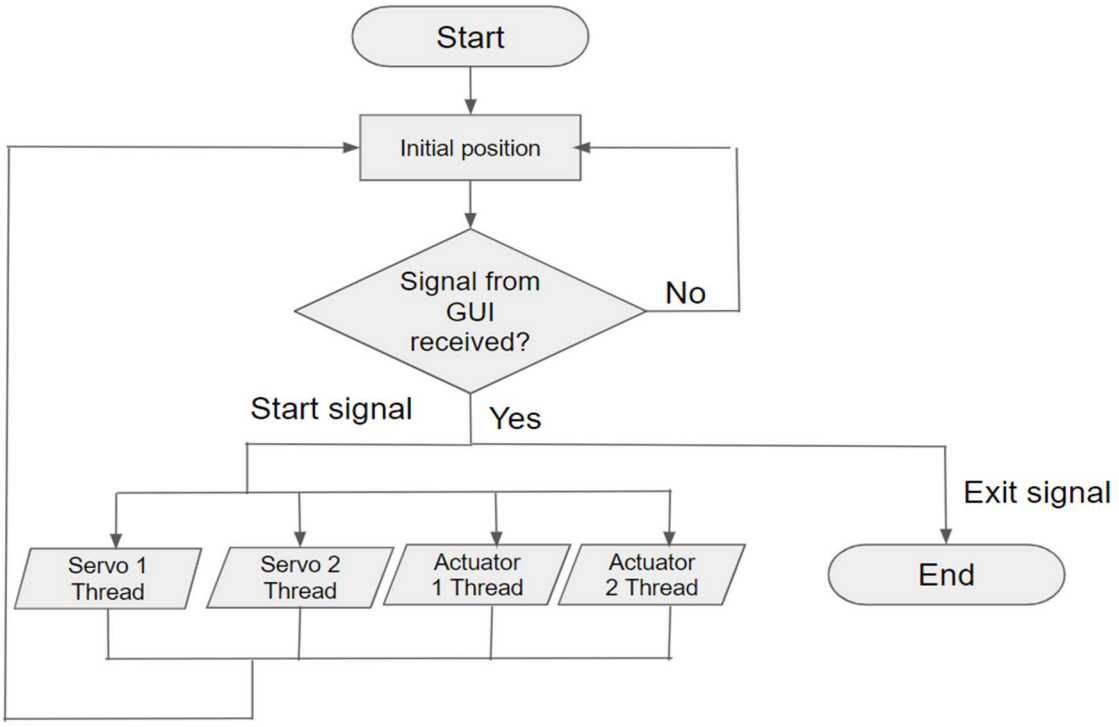

Figure 5: Component Flowchart

This first part of the code was when the components' gpio pins were initialized. The different linear actuators and servos were initialized using ‘gpiozero’ and ‘pigpio’ respectively. Gpiozero is a standard way to initialize gpio pins. This is used to identify which pins the code is being run on. The other library we used to initialize the servos was ‘pigpio’. The benefit of this library is that it uses a hardware pulse width modulation (PWM) rather than a software PWM. Initially, a software PWM was used for the servos, but this would cause them to jitter. This was caused by the software PWM sending interrupts to the servos whenever they were “sleeping”. Using the hardware PWM fixed this issue.

Next aspect of the code was the movement functions for the different components. The first movement function was for the servos. This function would take in a servo pin, an angle, and a gpio pin. This function calls the set_servo_pulsewidth function. In order to accomplish this, the angle the user inputs must be converted to a pulsewidth to have the servo turn the desired amount. The other movement function in our code is for the linear actuators. The function for the linear actuators takes in an actuator pin, a pulsewidth pin, a length, a direction, and which number actuator it is. First this function checks which direction the user desires the actuator to go. Next it determines which actuator the user wants to move. This is important because the actuators move at different speeds. The actuators move based on time rather than distance, so the distance needs to be multiplied with the speed to have the actuator move the

desired distance. The actuators are then run for the desired amount of time and then turned off until called again.

The largest part of the code is the different designs. Each design has its own class. Within each of the classes 4 functions are created, one for each moving component. These functions include sequences of movement function calls and sleeps to move each component at the proper time. All the components move to create the desired design. At the end of each of the functions in every class, all of the components are set to return to their initial position to be ready for the next latte design.

In order for all 4 of the component functions to run concurrently, it was necessary to include threading into the code. To do so, the 'threading' library was imported. Without the implementation of threading, only one component would be able to move at a time, however for the application of the project it was crucial that all the components could run at the same time. So after the design class was set up, a corresponding thread for each of the component functions was created. The threads were all started by using the '.start' from the 'threading' library. The '.join' function was also used to ensure all threads finished their processes before continuing to the next lines of code.

4. Performance Characterization and Discussion

Latte art is an artform, so it is difficult to quantify specific numbers. This project involved mostly trial and error to determine which designs looked correct, which type of milk to use, and what settings the milk needed to be steamed on. The overall motions on how to pour latte art were determined from watching videos and trying to do it manually, but the final designs needed to be tested repeatedly and the code modified slightly to determine if the desired design was produced.

The first design that was produced by the lazy latte was the heart design. When creating this design it was difficult to determine how much frothy milk would need to be poured to make a good size heart. Figure 6 shows the progression of pouring the blob for the heart. Once it was determined to be the proper size, the motion of creating the dimple at the top was added to truly define the heart shape. Once the final design resembled a heart, it needed to be repeated multiple times in a row to ensure no changes needed to be made. There would sometimes still be inconsistencies in the design, but this was due to consistency in the milk, which will be discussed later.

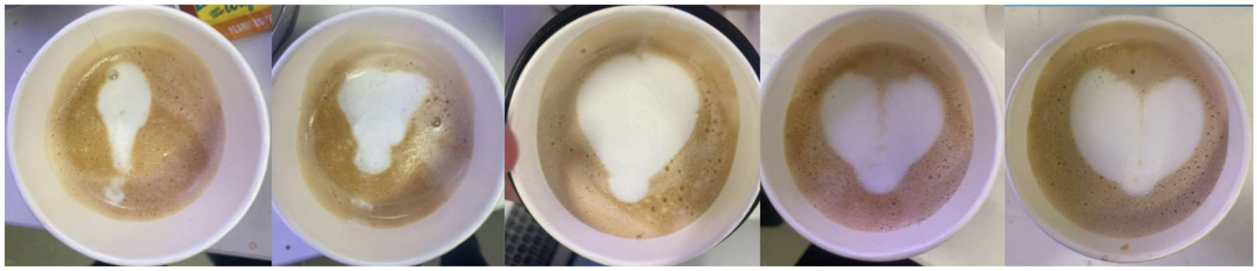

Figure 6: Heart Latte Art Progression

The second design the lazy latte can produce is a snowman. This design is produced by 3 properly spaced blobs getting progressively smaller. When this design was first being created, the blobs were merging together, so only two blobs could be seen. As time went on and it was more practiced the spacing was fixed to create 3 distinct blobs. Finally, in order to get the proper snowman shape, the blobs were perfected to be different sizes to show the base, the body, and the head of the snowman. This progression can be seen in Figure 7. Similarly to the heart, this design needed to be practiced repeatedly to ensure consistency. The milk did also play a role in how the snowman would turn out each time, but if the milk was correct then the snowman would look how it was supposed to look.

Figure 7: Snowman Latte Art Progression

Another aspect of the latte art was the milk that was being used. At the beginning of the project, different milks were used. Mainly 2% and whole milk were used in order to get properly frothed milk. For the last 3 weeks of the project, whole milk was used to be consistent, and because it created the best frothed milk. It was determined that the proper amount of milk was half a cup. This is what would produce the proper shapes in the 8oz cups. For the designs and timing it was important to always use the same amount of milk. The Nespresso machine used to make the coffee and steam the milk had different settings to adjust for the milk. When steaming the milk, a temperature and an amount of froth needed to be chosen. The temperature settings of the milk steamer have the options of low, warm, ideal, hot, and very hot. The Nespresso machine also has a temperature sensor, which ensures that the milk is heated to the specified temperature. For the latte art the lazy latte made it was determined that warm was the best

setting. This would allow the milk to be silky smooth and not have too many bubbles. The settings for the texture of the milk ranged from 1 to 8. This level identified how much froth was in the milk. For the latte art it was determined that 3 was the best setting for the milk. This would give the proper amount of froth to create the art. This would allow for enough milk to make the canvas and enough froth to do the designs on top. It was tested with high and lower froth levels, but the lower levels would not produce enough milk and the designs would not show up. When the froth levels were higher, the blobs that were produced would be too large and would fill the entire cup making the design indistinguishable. The settings that were used on the Nespresso machine are shown in Figure 8.

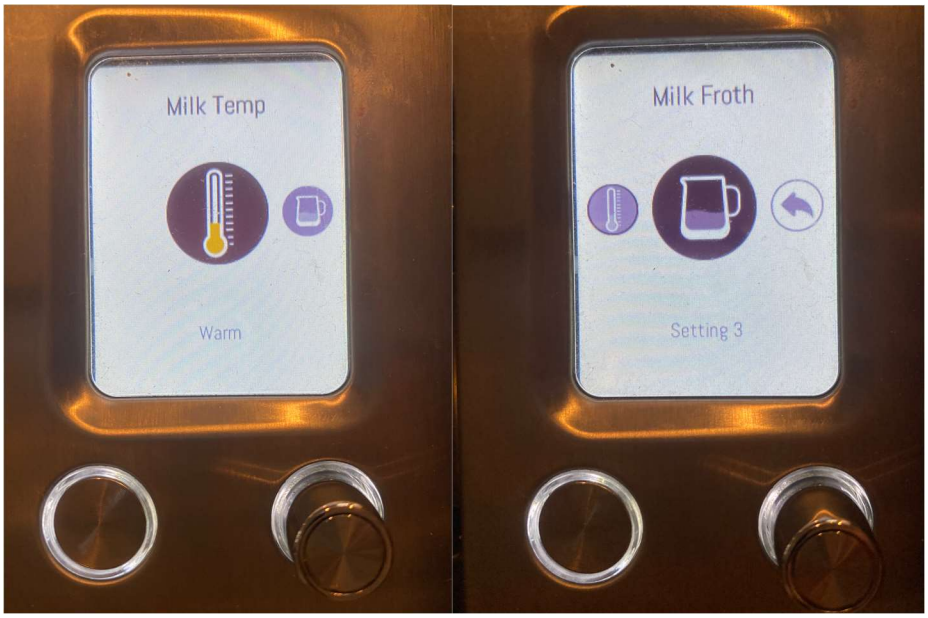

Figure 8: Milk settings used on Nespresso Machine

5. Cost Accounting

The total cost for the project was $247.57 as seen in Appendix A. Other parts included in the project did not have any cost, including several 3D modeled parts printed at the Watt Makerspace, spare wood from the lab, screws, and wires. In addition, the lab provided a monitor, keyboard, and mouse. The items that were not reimbursed were the paper cups and the coffee pods. This brings the reimbursed development cost to 191.46. In addition, a Breville espresso machine was utilized for this project but it was owned by a group member. There were no spare items purchased for this project so the artifact cost is the same as the total cost.

6. Project Postmortem

6.1. Technical Postmortem

Although the output of our system is subjective, the system made the two final designs well. The final codes for both the heart and the snowman created images that could objectively be identified as such. One aspect of the system that caused difficulties was the stability. When we ran initial testing to determine the code sequences, both cups were empty. Once we started testing with liquids, first water then coffee, the system's inability became obvious. When the platform was raised and lowered, it would wobble and shift sideways. To fix this, we modified the connection between the actuator and the platform, making it thicker. This decreased the lateral swaying. After testing with liquids and cups, we realized that this added weight made the platform imbalance. The cups were also at risk of detaching from the servo. We glued the cups to the servo attachments to prevent this. To improve stability, we added another horizontal slider to better distribute the added weight.

The two ultimate designs were strong, but we were limited in the designs we could produce due to only having a few axes of movement. More designs could have been produced if the system had another axis of rotation. This would either be with an additional servo or actuator to allow left/right movement of the milk cup instead of just forward/backward. This extra movement would allow the system to produce more typical latte art.

The initial goal for this system was to produce three distinct designs with the milk. Ultimately, the system was able to produce two consistent designs. This performance did not meet what we believed to be capable of. However, we were pleased with the results due to the time given. We discussed options and decided that two strong, consistent designs would be better than three inconsistent or unclear designs. We managed our time well, but the testing and fine-tuning of the design took longer than expected. We could have accomplished the third design with perhaps an extra two weeks of time.

6.2. Nontechnical Postmortem

For the most part, the three electrical engineers focused on the physical aspects of the project while the three computer engineers focused on the Python code. The physical structure was built quite early into the project for milestone 1. These three people were able to meet to work on building, with some 3D modeling and printing occurring outside of meeting times. This group also handled the wiring of components and determining the power requirements. While the structure was being built, the computer engineering group began developing the code to control the actuators and servos, involving building classes and creating a threading code. This division of labor worked well for our team to accomplish a lot before the first milestone.

After the first milestone when the semester began picking up, it was difficult for the whole team to meet to work on the project together. There were time conflicts with classes during the day and other

obligations in the evening. To combat this, we would schedule a three to four hour period to be in the lab where people could drop in when they were available. That way, we were able to put in a significant amount of time into developing without needing everyone present at once. This stage is also where the lines between majors blended. Since the physical aspects were mostly complete, the group was all working on modifying and fine-tuning the code for the designs. In addition, if a group member could not make it to a physical meeting, they could focus on another aspect of the project, such as the milestone powerpoints or the GUI code. This system worked well for our team. It could have been made easier if there were more times during the day to meet without worrying about time-conflicts. If the senior design class were scheduled during the day, instead of the 5-7 pm time slot, this would ensure that everyone in the group could meet at the same time.

6.3. Future Directions

While the team was satisfied with the final results of the project, there were aspects that could have been improved with more resources. The main resource needed was more time, as each design is complex and minute, requiring significant testing to perfect. The project could have implemented a third design with more time. One improvement to the system would be adding another servo to add an axis of rotation to the milk cup. This would allow the system to create more advanced designs. Another improvement to the system would be on the software side: adding a computer vision aspect. This could be used as design verification to ensure that the designs are meeting our standards and staying consistent. The system as a whole was quite loud as the actuators had to move a substantial weight. This could be improved by buying higher quality actuators that have a quieter operating volume. Using higher quality actuators would also give more control and perhaps even more designs if the speed were adjustable. We had to reject some design ideas as the speed of the horizontal linear actuator was too slow to accomplish it.

7. Acquiring New Knowledge

Derrick Joyce: I have learned Python, how to create GUI's with PyQT and Kivy, how to create a technical poster, how to create latte art, and how to work within an engineering team. My strategy to learning Python is expanding on my previous basic skills by researching quicker and more efficient ways to achieve tasks. My strategy to learning how to create GUI's in PyQT and Kivy was looking through documentation and YouTube videos of engineers creating simple widgets like buttons, images, and layouts. We decided to use both PyQt and Kivy for the GUI to see which would be easier to integrate with the hardware. My strategy for learning how to create a technical poster was to review previous posters created by myself, and other senior design teams from the past. Following the clear-cut guidelines provided by Dr. Raza also streamlined the process. My strategy for learning how to create latte art was asking my teammates lots of questions and watching many YouTube videos of how latte art is made from start to finish. My strategy for learning how to work within an engineering team was to take a step back to listen more than I spoke, always making myself available to assist others, and offer ideas to solving our problems we encountered.

Elise Ferkler: I have learned Python, the importance of commenting code, how linear actuators work, how to make latte art, how to time manage, and how to adjust scheduling as needed. My strategy to learn Python has been through researching the specific functions we would need to implement. My strategy for learning the importance of commenting code has been by working on code with others and having trouble knowing what other people's codes mean. My strategy to learn how linear actuators work has been through researching the necessary code and learning that each linear actuator takes a different amount of time to go the same distance. My strategy to learn how to make latte art has been by watching youtube videos and by trial and error. My strategy to learn how to manage time has been to try to set meetings towards the beginning of the week. My strategy to learn how to adjust scheduling as needed has been to keep track of how closely we are following the gantt chart and making changes to our goals as we go along.

Jackie Bendziewicz: Throughout this project I have learned some basic coding skills, as we implemented Python coding for our main software implementation. In this section, I learned a lot from my teammates who were focused on the software side. Each of them took the time to help me understand the programming done and some background knowledge for Python. I also learned more about the hardware systems such as the Raspberry Pi and the motor driver. I was not familiar with the Raspberry Pi before, since I had only used Arduinos. While they are somewhat similar systems, they operate very differently. To learn about the Raspberry Pi I watched many tutorial videos on youtube and various websites. Finally,

I had to learn about the world of latte art. I had no experience pouring latte art myself, so I needed to watch tutorial videos and step by step articles breaking down each step and motion of creating latte art.

Mariah Tam: I have learned how to control components in Python, how to make a GUI using the Kivy and Python, how to operate a Raspberry Pi this semester, and how to make latte art . My strategy to learn controlling components in Python has been researching different libraries and the components we integrated, along with trial and error. My strategy on learning to make a GUI using Kivy and Python, was watching youtube tutorials and researching the Kivy library. My strategy to learn how to operate a Raspberry Pi was to look at the specific pinout diagram of our Raspberry Pi, and research modules that worked the best with the components we were using. My strategy to learn how to make latte art was researching the different variables of making latte art and watching videos of other people making the latte art and frothing the milk.

Leah Brown: I have learned how to use a Raspberry Pi, some basic coding, best practices for making latte art, and how to manage time and workload efficiently. My strategy for learning about the Raspberry Pi was looking at the wiring diagrams when we began to connect everything as well as research modules for a hardware PWM. My strategy for learning basic code was to look over it and make basic changes as needed throughout testing. My strategy for learning about latte art was watching youtube videos, reading articles, and practicing by hand. My strategy for learning about time management and teamwork was to schedule meetings in advance and ensure the majority of the group could attend. Also I learned which tasks were better to take care of outside of team meetings as they only needed one person to work on it.

Luke Swetonic: I have learned Python, how to operate a Raspberry Pi, how to integrate different motors into the same machine, how to manage time better, how to allocate proper roles, how to take advantage of strengths, and how to make latte art. My approach to learning Python was to research specific problems that I was having or needing to overcome as well as learning the various libraries that were available for use with the Raspberry Pi. My approach to learning the Raspberry Pi was to read articles that explained the uses of the Raspberry Pi that we specifically needed and how to program the pins to do so. My approach to learning how to integrate motors to work together was by researching how to get multiple instantiations of a chunk of code to run simultaneously, which was achieved through multithreading. My approach to learning to better manage my time was learning over the semester how long certain tasks take to complete and allocating my personal time to work on those accordingly. My approach to learning how to better take advantage of strengths was by noticing over the semester what people did exceptionally well within our project and allowed for more of that to be taken care of by individuals. My approach to

learning how to make latte art was to watch various videos on proper pouring techniques as well as proper milk frothing consistency.

8. Technical Standards

802.11-2020 - IEEE Standard for Information Technology--Telecommunications and Information Exchange between Systems - Local and Metropolitan Area Networks--Specific Requirements - Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications

NFPA 70E® Standard for Electrical Safety in the Workplace®

ISO 21.040.10 Metric screw threads

ISO 22000 Food safety management

ISO 13849-1:2023 Safety of machinery — Safety-related parts of control systems — Part 1: General principles for design

29119-2-2021 ISO/IEC/IEEE International Standard - Software and systems engineering - Software testing -- Part 2: Test processes

Appendix A - Part Costs

Item Description	Part Number	Vendor	Quantity	Total Cost	Payment Type
4 inch linear actuator	AM-TGF12V100-T-1	Eco-Worthy	1	41.99	RP
6 inch linear actuator	AM-TGF12V150-T-1	Eco-Worthy	1	41.99	RP
Raspberry Pi 4	RAS-4-4G	Raspberry Pi	1	55.00	PO
DC Servo Motor, pack of 2	YLK2020414	Betu	1	32.99	PO
Motor Driver	MDD3A	Cytron	1	8.50	PO

Drawer Sliders	BT-004	Btibpse	1	10.99	RP
Nespresso Pods	B098YH1981	Starbucks	1	33.33	PP
Paper Cups	B09DNS3Y37	Racetop	2	11.39	PP

Acknowledgements

Clemson Machining and Technical Services

Watt Center Makerspace

References

[1] “Gpiozero,” *gpiozero*. [Online]. Available: https://gpiozero.readthedocs.io/en/stable/.
[2] M. Bilal, “Ampere debuts 80-core arm server processor,” *PIC Microcontroller*, 11-Apr-2020. [Online]. Available: https://pic-microcontroller.com/ece-4760-latte-art-machine/amp/.
[3] Sebastian, “GPIO programming on the Raspberry Pi: Python libraries,” *Medium*, 31-Aug-2021. [Online]. Available: https://medium.com/geekculture/gpio-programming-on-the-raspberry-pi-python-libraries-e12af7e0a812.