Product: AutoFold
Individual Report - S2157884

1. Introduction

AutoFold is an innovative system (Figure 1), designed to automatically fold clothes and neatly stack them in a pile. The system consists of two key components: the folding platform and the loading mechanism.

Users interact with the product by simply hanging clothes on the drying rack using custom hangers and then initiating the folding cycle with a button press. The system autonomously manages subsequent folding and stacking processes, utilising the folding platform and a mechanism for transferring clothes to the platform once ready. A detailed step-by-step breakdown of the process has been provided in section 4.1 in the appendix.

The primary aim of AutoFold is to provide a convenient and accessible solution for individuals with mobility limitations, such as rheumatoid arthritis or dyspraxia, whether in home environments or care facilities. Additionally, a secondary market of busy individuals and students has been identified, highlighting the versatility and broad applicability of the system.

In this report, I will provide insights into my key contribution to the AutoFold project, as well as reflections on the teamwork involved and the skills acquired throughout the course of its development.

Figure 1. Final Product Render

2. Key Contributions

2.1. Brief Overview

My primary focus revolved around system design, i.e. conceptualising the working of the complete system and then breaking it down into smaller components. I worked on building multiple of these individual components along with integration-based tasks essential for the operation of the AutoFold system. The major tasks undertaken by me include:

- **Bluetooth Communication Implementation**: Enabled wireless communication between the loading mechanism's moving component and the Raspberry Pi, facilitating garment detection for stepper motor control. Detailed in section 2.2
- **Stepper Motor Circuit Setup and Coding**: Led the setup of the stepper motor circuit, transitioning from Arduino to Raspberry Pi for cost-effectiveness. Developed code for stepper motor operation and collaborated on custom PCB design. Detailed in section 2.3
- **Establishment of Serial Communication**: Implemented USB serial communication between the Raspberry Pi and a Pico micro-controller driving servo motors. Created a user-friendly interface with Grove buttons, simplifying system operation to a single click. Detailed in section 2.4

In addition to my core responsibilities, I actively contributed to various other aspects of the project, including general soldering, product assembly, designing various 3-D components, creating presentations, and delivering these presentations. Throughout my involvement, I dedicated approximately 230 hours to the project.

2.2. Bluetooth Communication Implementation

We required a solution to enable communication between the moving component on the loading mechanism and the main computer (Raspberry Pi). This was essential for reversing the stepper motor rotation to pull the garment onto the folding platform upon reaching a hanger, as well as for detecting when the moving component had reached the folding platform to re-initiate the cycle.

The most straightforward method was to utilise WebSockets (MozDevNet, 2020) as I had prior experience with the same. However, this approach would have required user setup, whereas a Bluetooth based solution could be preconfigured, minimising the user intervention required. Additionally, wired solutions were also considered, but the complexity of wire management for such a moving component and potential safety hazards (Qviro, 2024) made Bluetooth the more viable choice for our project.

Challenges were encountered during the implementation phase, notably due to ambiguity in the Pico documentation. Extensive research and experimentation were required to overcome these challenges. I discovered that traditional Bluetooth protocols were not supported by the Pico W micro-controller, which only supported Bluetooth Low Energy (BLE). This required additional modules to be installed on the Raspberry Pi. Also, as the Pico used MicroPython and the Pi used regular Python, I had to go through various documentation and libraries to develop the code, as online guides and tutorials for the Pico W were limited to communicating between two devices using MicroPython. (Foundation, b)

Leading the development of the core code for Bluetooth communication, I came up with a working solution, including integrating the button-based sensors into the Pico. After testing this solution, I collaborated closely with Khalid Belhadj to integrate it into the final software framework. Our efforts were focused on optimising the communication process, using threading to enable near-simultaneous detection from the two buttons so that we could instantly transmit messages. For final integration with the system, we loaded the file onto the Pico such that it automatically ran when powered (Foundation, a) and secured the Pico along with other components in a laser-cut wooden box. This thus created the moving component as seen in figure 4. This was attached to a timing belt being rotated with a stepper, as described earlier.

To confirm the reliability and effectiveness of the Bluetooth communication setup, we subjected it to rigorous testing processes. Over the course of 30 iterations, we observed that 22 connections were successfully established on the first attempt, with 7 connections on the second attempt and 1 connection on the third. Since our protocol allowed up to 5 connection attempts, and each connection was established within 1-2 seconds, we experienced no difficulties. Despite the limited sample size, we encountered no issues with establishing Bluetooth connections during both the development and testing phases. This 100% successful connection rate instilled confidence in the reliability of the Bluetooth connection for consumer use.

2.3. Stepper Motor Circuit Setup and Coding

In approaching the stepper motor circuit setup and coding, I relied on extensive online research, documentation, and data-sheets supplemented by tutorials from various sources, including YouTube and UCreate Makerspace at the University. Despite the scarcity of specific resources for our stepper motor model, I navigated through trial and error to incorporate general principles seen across various stepper motor setups into our project, gradually building proficiency in circuit setup and writing code to control the motor.

The decision to initially configure the circuit on an Arduino platform was driven by its user-friendly interface and suitability for experimentation. As the project progressed, considerations of system optimisation and cost efficiency led to the transition to a Raspberry Pi, which proved not to be too difficult as it just involved converting existing C++ code to Python and re-configuring the pin setup.

In developing the code for controlling the stepper motor operation, I studied documentation to understand the required pulse signals and their impact on motor behaviour. Challenges included fine-tuning pulse parameters to achieve desired motor performance and mitigating issues such as resonance and torque fluctuations. (Portescap, 2021) Through systematic experimentation and iterative refinement, I successfully addressed these challenges, resulting in robust and efficient motor control algorithms. The code was structured in an object-oriented manner to abstract the implementation of these functions, allowing them to be called simply as functions from an external library. This code is run continuously throughout the machine's running. The moment AutoFold starts, the stepper motor drives the moving component towards the hangers, and then once the hanger is grabbed and the Pi receives a Bluetooth signal, it reverses direction to pull the garment on the hanger onto the folding platform. I finally fixed the stepper in a 3-D printed housing I designed and secured it to the machine frame as shown in figure 7.

Furthermore, I partnered with Sarthak Mangla to develop a custom PCB design using Target CAD software, replacing the initial breadboard setup, and can be seen in figure 6. This collaboration resulted in a more streamlined and reliable system featuring a significantly reduced form factor. This enhancement not only improved the overall functionality and robustness of the stepper motor control within the project but also eliminated the need for loose wires, ensuring a more professional and tidy setup.

2.4. Establishment of Serial Communication

I consulted Lab Technicians and researched online to determine the most suitable communication method from our main logic computer, the Raspberry Pi, to a Pico microcontroller responsible for driving five servo motors. With two viable options available; a) a wired serial connection and, b) a wireless solution using WebSockets or Bluetooth . Therefore careful consideration was rendered essential. As wireless connectivity posed complexities and required the use of a Pico W, which came at a slightly higher cost than the regular Pico, I ultimately chose the stability and simplicity offered by a USB serial connection. This decision was based on its ease of implementation and minimal cable management requirement due to the stationary nature of the two devices.

After conducting thorough documentation research, I successfully established this communication system, allowing our folding platform to operate directly from the Pi without needing to connect to the Pico physically from a laptop. Additionally, leveraging my recently acquired familiarity with serial connections and basic electronics on the Pi, I developed a user interface using Grove buttons. I even designed and 3-D printed the button caps and casing for the same as shown in figure 5. This streamlined system operation to a single button click, significantly enhancing the usability and accessibility of our system, and ensuring seamless interaction for users.

3. Learning Outcomes

3.1. Acknowledgment

The System Design Project course has provided me with a range of experiences and opportunities to expand my skills and knowledge. I am grateful for the support and guidance from Mr. Garry Ellard and our mentor Guoyu Zhang. I also want to thank everyone involved in the course's success, including the technicians, professors, and PhD judges. Moreover, SDP also allowed me to meet and befriend various people from the university whom I hadn't known before. Lastly, I would also like to thank my team without whom this project would not have been possible. I have picked up various skills on the way to making our AutoFold machine a reality and would like to mention some of them in this section.

3.2. Technical Learning

Throughout the course, I acquired multiple skills necessary to bring our AutoFold machine to fruition. One significant area of growth was in Electronics and Circuits, where my understanding was previously limited to basic Physics experiments in high school. I had long desired to expand my knowledge in this field but lacked the opportunity to do so. My journey began with the initial servo setup using the Pico, which sparked my interest and demonstrated it was not as challenging as it seemed. Delving into the documentation to comprehend the functionality of various ports and pins on a Raspberry Pi further deepened my understanding. The stepper motor configuration presented additional challenges, requiring me to grasp concepts such as connecting the motor to a driver and configuring the driver with the Pi. Having gone through these various experiences of controlling various types of motors, buttons and sensors, I am now confident in my ability to experiment with and make such personal electronic projects and actually have a physical machine rather than just being limited to an online application or website.

I found great enjoyment in learning to use Fusion 360 software for designing various 3D prints and laser cuts, an area in which I had minimal prior experience. It was a skill I wished to acquire, and I had the opportunity to learn from Alex Bailey, who excelled in this software and was familiar with its advanced features. His guidance proved invaluable, and I quickly picked up techniques that enhanced my design capabilities. I also got to learn "Target 3001!" - the PCB design software from Sarthak who was proficient in it.

Additionally, I acquired proficiency in soldering during the course. While working on the robot, I took on the responsibility of most of the soldering tasks. The technicians provided helpful explanations of the basics, and with practice, I was able to produce fairly neat results. This skill was particularly beneficial during the stepper motor setup process, contributing to the overall success of the project.

3.3. Lessons Learnt

Despite our strong group dynamic, we acknowledged areas where we could have performed better. Firstly, although we had a clear task list, our task sequencing could have been improved. For instance, starting software development too early without concurrently calibrating it with hardware and leaving integration until the last minute resulted in inconsistencies during hardware evolution, wasting time and effort. Additionally, better time management was needed. With everyone's fixed schedules and sporadic group meetings except before each submission, setting stricter internal deadlines and mandatory meetings could have improved coordination and led to a more refined end product.

Moreover, discussing the final product's specifications and functionalities in more detail early on could have provided clearer project milestones and minimised misalignment in task prioritisation. While we created Gantt charts to outline project timelines, delving deeper into the final product's intricacies could have offered greater clarity and direction throughout the development process.

Despite these shortcomings, I take pride in the accomplishments of our group. We have successfully developed a fully functional prototype, with the ambitious loading mechanism operating as intended.

3.4. Personal Reflection

Throughout the System Design Project, I experienced significant personal growth, particularly in non-technical areas. One notable area was learning to delegate tasks effectively and trust my teammates to complete them successfully. Previously prone to micromanaging, this time I consciously shifted towards empowering my teammates, delegating tasks based on their strengths and expertise. This fostered a more collaborative and efficient workflow, enhancing overall project performance.

Additionally, the project provided me with the opportunity to rediscover my passion for presenting and public speaking. While I excelled in these skills during high school, they had taken a backseat in recent years. Taking the lead in designing and delivering presentations reignited my enthusiasm, reminding me of the value of effective communication in both academic and professional settings.

Moreover, I developed adaptability and flexibility in response to unforeseen challenges and obstacles encountered throughout the project. Whether troubleshooting technical issues or adjusting project timelines, embracing these challenges with a positive mindset allowed me to develop resilience and problem-solving skills.

Furthermore, working in a team environment honed my collaboration and conflict resolution skills. By fostering open communication and a supportive team culture, I effectively navigated conflicts and maintained positive working relationships with my teammates.

Overall, the System Design Project provided invaluable opportunities for personal growth, allowing me to develop essential skills that will benefit me both academically and professionally in the future.

References

Foundation, Raspberry Pi. Getting started with raspberry pi pico, a. URL https://projects.raspberrypi.org/en/projects/getting-started-with-the-pico/9.

Foundation, Raspberry Pi. Connecting to the internet with raspberry pi pico w, b. URL https://datasheets.raspberrypi.com/picow/connecting-to-the-internet-with-pico-w.pdf.

MozDevNet. The websocket api (websockets) - web apis: Mdn, 2020. URL https://developer.mozilla.org/en-US/docs/Web/API/WebSockets_API.

Portescap. Avoiding resonance with stepper motors, 2021. URL https://www.portescap.com/en/resources/motor-specifications-and-literature/white-papers/avoiding-resonance-with-stepper-motors.

Qviro. Good robotics cable management: Full guide - qviro blog, Apr 2024. URL https://qviro.com/blog/robot-cable-management/.

4. Appendix

4.1. System Workflow

Here are the steps outlining how the system operates:

1. The machine is turned on.
2. The machine establishes a wireless connection and ensures the moving part is in its initial position.
3. The user initiates the process by pressing a button.
4. A computer component, Raspberry Pi, activates a motor, causing the moving part to approach a series of hanging items.
5. A sensor detects when the moving part encounters a hanging item. A small electronic device, Pico W, attached to the moving part, moves a grabbing mechanism and sends a wireless signal to the Raspberry Pi to pause and reverse the motor.
6. Upon receiving the signal, the Raspberry Pi halts the motor briefly and then changes its direction.
7. The moving part pulls the hanging item along, laying the garment flat on a folding surface.
8. As the moving part reaches the end of its path, it triggers a mechanism to release the garment onto the folding surface. Simultaneously, a button on the Pico W device is pressed against a block, signalling the Raspberry Pi that the garment has been deposited.
9. The Raspberry Pi stops the motor and sends a signal through a wired connection to another Pico device to start the folding process.
10. The second Pico device receives the command and operates servo motors to fold the garment in a predetermined pattern. The final step involves flipping the folded garment onto a stack behind the folding surface.
11. The process repeats until the moving part completes its path without encountering any hanging items, indicating that there are no more garments to fold. At this point, the Raspberry Pi resets the system to its initial state.

An emergency stop button is available next to the start button. Pressing it immediately halts the machine. When the system is restarted after an emergency stop, it returns to its initial state.

4.2. Various Components

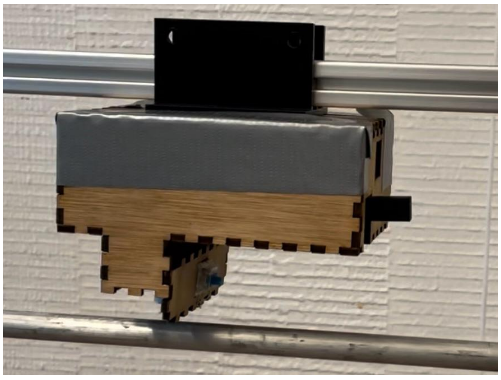

Figure 2. Moving Component in initial state

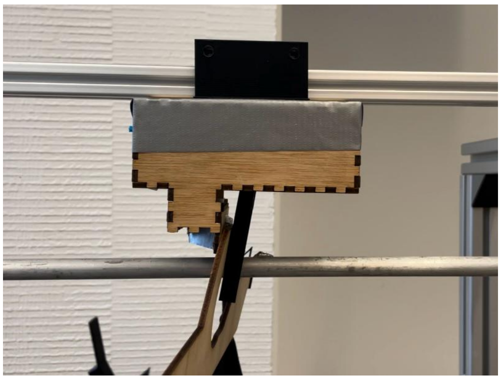

Figure 3. Moving Component Grabbing Hanger

Figure 4. Moving Component

Figure 5. 3-D Printed User Interface

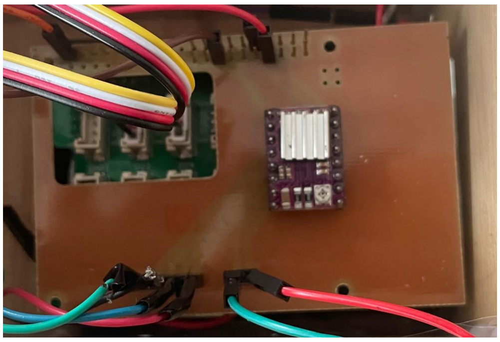

Figure 6. PCB Hat attached to Raspberry Pi

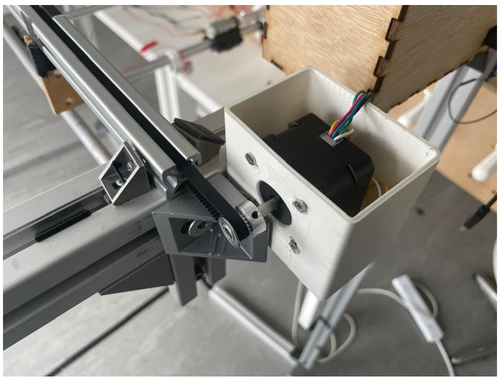

Figure 7. Stepper Motor