Automated Mail Sorter

Daniel Emerson, ME, James Finn, CSE, Harrison Liu, CSE, and Long Nguyen, CSE

Abstract—**Sorting mail is a mundane and routine task that can be significantly enhanced through automation. Automation would improve the cost efficiency compared to a traditional mail carrier. Currently there are no automated last-mile delivery implementations available on the market. We plan to fulfill this gap by designing a small scale automatic mail sorter for individual office mailbox arrays. Our design features a hopper system that dispenses one 9 ½ " by 4 ⅛" envelope at a time into a delivery tray. From there, a photo is taken of the envelope, and text image processing is applied on the photo to determine the address of the recipient. Finally, the delivery tray moves the envelope to the corresponding mailbox location and returns to its original position. This process repeats until the hopper is empty.**

I. Introduction

Sorting letters is a part of every step of mail delivery. It is routine, mundane, and traditionally fulfilled by manual human labor. We want to focus on addressing the last stage in the mail delivery process, specifically after the mail has been delivered to the appropriate building address. Often in large buildings there are walls of mailbox arrays. When the letters arrive in the building, someone has to read through each letter address and deposit the letter into the correct mailbox. Our product can be mounted on each wall of mailbox arrays to automate the last stage of the mail delivery process.

A. Significance

Our project has the potential to reduce mundane and routine tasks in mail sorting, especially in the context of big buildings with mailbox arrays. This can help free up time for other more important tasks and increase productivity. Additionally, machine labor is much cheaper than human labor, allowing our product to make a financial impact by saving human labor costs. "Delivery is the Postal Service's largest cost center accounting for more than 40 percent of expenses, and having carriers manually sort mail takes time and money. Carrier routes are configured to take eight hours to complete, and those eight hours include time spent in the office . . . primarily manually sorting mail, as well as time spent on the street" [1].

B. Context and Existing Products

Mail sorting has been a problem long before our project. Traditionally, it has been a task for humans to manually sort letters into corresponding mailboxes. This is a primary solution that our product aims to compete against.

On the market right now, there is another large scale solution for mail sorting, the OPEX mail matrix, with the smallest possible model costing around $180,000 [2]. This product is mainly used in very large mail sorting centers, before they are sent off to the last mile of the delivery. The goal of our project is not to automate large mail centers, but to automate the last stage of mail delivery, to put letters into the correct mailbox once the letters arrive at the correct address. Ultimately, our product and the OPEX mail matrix both seek to automate the mail delivery process, but we target different stages in the mail delivery process. Both our product and the OPEX mail matrix also need to compete against manual human labor.

C. Societal Impacts

As with any automation project, we must consider that we are automating jobs that people rely upon. Such is the nature of innovation and technological advancement. Our mail sorter will still require one person to load letters into the hopper and resolve any jams of the machine. When visiting UMass Mail Services, we received a variety of responses regarding our project. The manager and supervisor seemed impressed and enthusiastic to hear the outcome of our project. The mail sorters and receptionist replied to the tune of, "so you are replacing our jobs?"

D. Requirements Analysis and Specifications

Requirement	Specification	Value
Image Processing	Accuracy in recognizing typed font	≥99 % accuracy
Speed	Letters processed per day	≥ 1000 letters/day
Delivery	Accuracy of bringing the delivery tray to the correct position	≥ 95% accuracy
Dispensing	Accuracy of dispensing exactly 1 letter	≥ 95% accuracy

Table 1: Requirements and Specifications for CDR

II. Design

A. Overview

To automate the mail sorting process we created an autonomous system designed to complete the task mentioned above. The main technologies we used to solve the problem were image processing and motor control. At its simplest level, the project overview can be described as the following: Take a photo of a letter, process the letter's address, then move motors a certain amount based on the address.

We divided the project into three main subsystems. First we have the dispense mechanism, of which the goal is to dispense exactly one letter from a hopper into the staging area. Second is the processing unit, which completes all image processing and address recognition. Third is the delivery mechanism, which is the subsystem responsible for moving the letter from the staging area to the appropriate mailbox. Figure 1 shows a block diagram of how these subsystems interact with each other. Each subsystem is described in depth below in their respective sections.

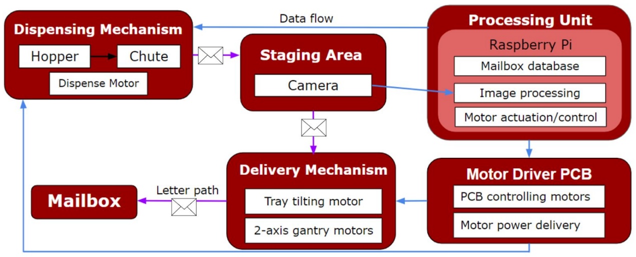

Figure 1: CDR Block Diagram

In order to meet our image processing and delivery specification, we had to compromise a lot of speed, which is also one of our specifications. If we chose to move our gantry too fast, the stepper motors that drive the gantry would not have enough torque, occasionally causing slipping. When the stepper motors miss steps, the gantry will eventually fall out of calibration. Over time these missed steps aggregate and our gantry would no longer be accurate enough to deliver the letter tray to its appropriate destination mailbox. Our solution was to simply not run the gantry motors too fast, thus eliminating this problem. The trade off here is of course gantry speed. One way we could have improved the operating speed of the motors was by adding encoders to create a closed loop feedback system. Unfortunately this would have added a lot more complexity to our project and caused us to go over budget. Large scale linear encoders are very expensive and would be complex to integrate into our design.

With our image processing, we started with barcode reading at MDR, which worked very well and had fast processing speed (on the Raspberry Pi 4), achieving an average of 3.5 seconds per processing time. At CDR, we upgraded our image processing to recognize typed text, which took more processing time, averaging around 8.6 seconds per processing time, which further slowed our entire processing speed. In order to achieve our speed specification of dispensing 1000 letters in an average 8 hour work day, we would need to process a letter within 28.8 seconds. At our current state of the project during CDR, we needed to increase the speed of the delivery mechanism in order to achieve this specification. On average, our system would take around 2 seconds to dispense the letter, 9 seconds to process the address, and 17 seconds to move the delivery tray to the address location and back to the original position.

B. Dispense Mechanism

The dispensing mechanism, as seen in Figure 2, consists of a hopper, gate, servo motor, and a chute. The hopper is built out of wood and sized to be roughly larger than the size of the standard 9 ½" by 4 ⅛" letters. The stepper motor is mounted below the hopper, with a rubberized wheel protruding through a slit in the bottom of the hopper. This wheel grips the bottom letter when rotated, pushing the letter out of the hopper. An adjustable gate made of plexiglass sits at the front of the hopper to prevent more than one letter from being dispensed at a time. In our testing for CDR, the dispense mechanism typically achieved a success rate of ~95%. This design was derived from that which we had at MDR. We originally saw inconsistent results in dispensing the last letter in a stack. The lack of counter weight that the stack provides led to the last letter being dispensed at an angle, causing it to jam. For CDR, we used a block of wood to act as weight to ensure all letters are level when dispensed. While this was not the permanent solution we envisioned for FPR, it did lead to a dispense mechanism that worked very well.

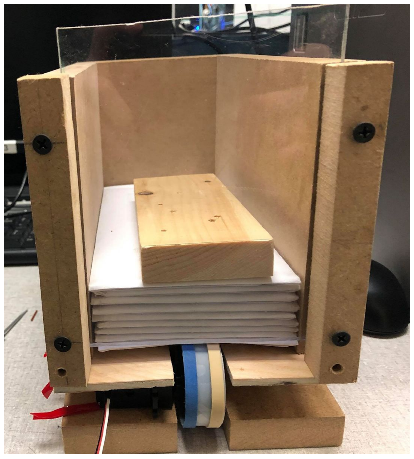

Figure 2: Dispensing Mechanism

C. Processing Unit

For our processing unit, we decided to use a Raspberry Pi 4 equipped with 1.5 GHz processing speed and 1 GB RAM [3]. We also have an RPi Camera Module v2 with 8 megapixel camera quality attached through the camera port.

The main functions that the RPi accomplishes for our project are image processing, motor actuation/control, and storing the mailbox database.

The project's main program, main.py, was written in Python. At its core it consists of a while loop, within which functions for image processing and motor control are called in a sequential order. The main program ensures that the system works as a cohesive unit, even when met by challenging edge cases. For example, in the case that a letter is unmarked and no address can be derived, the main program will deliver the letter to a 'junk' mailbox. Importantly, the main program also tells the system to stop operating when there are no more letters in the hopper by breaking the loop.

The mailbox database is stored within main.py. It keeps a list of the addresses on the mailbox. Each address corresponds to a number of motor steps. When it receives the address from image processing, it inputs the number of motor steps to rotate as a parameter for the motor programs.

For motor actuation and control, we made Python programs to control each motor's movement in terms of direction and rotational distance. These motor programs run on the RPi and send signals from the RPi's GPIO pins to the appropriate motor drivers, which in turn move the appropriate motors.

For image processing by CDR, we were able to take a photo of the envelope, locate the position of the printed text from that photo, and decipher the recipient's address from the text. This was accomplished using the PyTesseract and OpenCV Python libraries. First, the RPi system calls camera.py which takes a photo and stores it to 'envelope.jpg'.

Then, the RPi system calls text_recognition.py which takes in the envelope.jpg image, converts the image to grayscale, scales pixels down for faster image processing, and then locates the region of text and deciphers it. The recipient's address is then stored in 'address.txt' for the delivery mechanism to reference.

To test the accuracy for reading addresses, we conducted a test of placing 40 different addresses and envelope orientations under the RPi camera. Some of the addresses were long text like McLaughlin, while others were short like Hollot. We also tested in Times New Roman and Arial font, and deviated the font size from 12-48pt. Out of all 40 tests, the program was still able to decode the text 40 out of 40 times for an accuracy of 100%.

D. Delivery Mechanism

By CDR, the delivery mechanism moved the letter in the X axis to the correct mailbox. The Y axis did not work correctly due to a missing lead screw. For FPR, the delivery mechanism needed to move in both the X and Y directions. The final delivery mechanism design consists of three lead screws, each with two linear bearing rails to provide support on each side of the lead screw. These two rails are important for absorbing the torque on the lead screw and allowing for smooth operation of the stage. When the lead screw is torque by the weight of the horizontal stage we see issues akin to our CDR demo.

The parallelism of the lead screws and linear bearing rails is very important. For CDR we aligned the rails as best we could when screwing them in directly to the mailbox's wood frame. However, this was not perfect and caused some difficulty in operating the horizontal stage. To address this issue, we planned to use a mixture of machined metal components and nylon 3D printed components for mounting of the lead screws and rails for FPR.

We saw a large improvement in speed from MDR to CDR when we switched from the Bosch R146520000 Linear Actuator to the lead screws. The Bosch linear actuator was powered by a ball screw linear actuator with a very fine pitch that was good for positional accuracy, but too slow for our application. Once we switched to the lead screws with a coarser pitch we saw a big improvement in the translational speed of the gantry. Once the third lead screw was added and alignment issues were addressed, we were anticipating a much faster gantry system.

In MDR the Bosch linear actuator often caught and caused the motor to stall briefly. This resulted in a positional error of <5% on the longest letter delivery. In our testing we manually corrected this positional error at the end of every testing cycle. We noted that if this problem persisted that we would need to purchase some sort of encoder to obtain positional data and create a feedback loop that corrects the position of the gantry at the end of every delivery. It became apparent as we approached CDR that it would be quite difficult to integrate a linear encoder, and would put us far over budget. Instead we decided that our efforts would be focused on careful alignment of the lead screws so minimized torque on the lead screws and consequently decrease the load on the stepper motors.

For FPR we also needed to implement a mechanism to move the letter from the delivery tray into the mailbox. We considered two methods, a tray tilting mechanism or a letter pushing mechanism. We extensively tested a tray tilting mechanism for CDR and found the stepper motors did not have enough holding torque to support the letter, and in some cases were not even strong enough to support the tray. This was likely because of how far away from the motor shaft the load was applied, which caused a large multiplication of the torque force. Instead we decided to go with a design similar to the dispense mechanism. This mechanism has proven itself to be effective and reliable in the open loop dispensing of letters, and would work similarly when implemented in the delivery tray.

III. THE PRODUCT

A. Product Overview

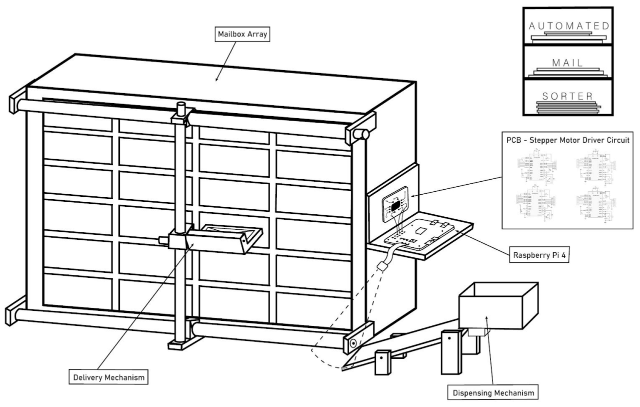

Figure 3. Overview product sketch

We divided the project into three main subsystems. Figure 3 shows the subsystems of our products all integrated into one cohesive system. First we have the dispense mechanism, of which the goal is to dispense exactly one letter from a hopper into the staging area. Second is the processing unit, which completes all image processing. Third is the delivery mechanism, which is the subsystem responsible for moving the letter from the staging area to the appropriate mailbox for delivery via a 2-D gantry.

B. Electronic Hardware Component

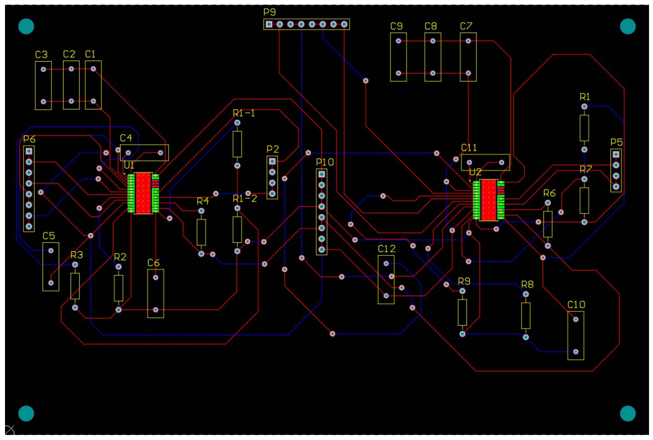

Figure 4. Altium screenshot of PCB

We chose to use our PCB as a stepper motor driver board for all of our motor drivers. Figure 4 shows the top side of our PCB housing two stepper motor driver chips surrounded by the supporting passive components. The bottom side of the PCB does not house anything. At MDR, we were using a stepper motor driver board, but it only supported one single stepper motor, which was sufficient for our project at the time. Now however, we've expanded our gantry to 2-D and thus need more stepper motors. Our PCB was designed to support two stepper motors which together would drive our upgraded gantry and share a common power source on the board. At the heart of our PCB, the stepper motor driver chips are the surface mounted Texas Instruments DRV-8825PWP chips. Each chip has its own set of resistors for different purposes like current limiting, pulling up, pulling down, and capacitors for different purposes like decoupling and bypassing. These passive components are not surface mount, but rather through hole for ease of component sourcing from the labs. We have rows of wire clamp blocks on the board to our DC power supply, as well as connect to our stepper motors. We also have rows of pins on the board to take in logic control signals from the Raspberry Pi. At the four corners of the board, there are mounting holes that allow us to easily station our PCB wherever we might need it. The board itself measures 4 inches by 6 inches. We are not able to obtain a picture of the finished soldered board because the SDP lab is now inaccessible.

C. Product Functionality

At CDR, the two major changes in our product was scaling up the gantry from 1-D to 2-D and changing image processing from barcode recognition to typed text recognition. The gantry did not function properly at CDR. We put our gantry on two mounting rails at CDR, but this caused our gantry to slant towards one side and this sag inhibited the vertical movement of our gantry. The horizontal movement of our gantry was functional at CDR, but that's only half of the 2-D gantry. In order to fix our gantry, we needed to add in another lead screw and mounting rails to better support our gantry and prevent it from slanting to one side. Our image processing was able to recognize typed text from letter labels at CDR. We accomplished this using OpenCV and PyTesseract library functions, and could locate and decipher the correct address text 95% of the time at CDR. Aside from the major functionality changes between MDR and CDR, we also made tweaks and optimizations to parts of our product. The letter dispenser was given a new dispensing wheel that was able to both better balance the letters and better grip onto the letters. Additionally, we rotated the direction in which the letters placed into the letter dispenser, allowing the letters to be better balanced on the wheel. This helped to vastly improve our dispensing goal of exactly one letter at a time. The hopper also got a new gate that was clamped more securely in place, leading to more repeatable results and thus better dispensing consistency.

D. Product Performance

At CDR, some subsystems worked well while others needed improvement. The letter dispenser worked very well. It consistently dispensed one letter from the stack at a time when triggered. In our testing of twenty trials, we had a success rate of 19/20, which is 95%, and exactly met our 95% dispensing accuracy specification. Our image processing was also very consistent, and was able to process envelope addresses with 99% accuracy. However, we were not able to meet our delivery specification due to our 2-D gantry being incomplete. Our 2-D gantry was supported by two mounting rails and driven by two stepper motors, causing the gantry to slant towards one side. To rectify this issue, we simply need to add in a third mounting rail with a third stepper motor. In order to achieve our speed specification of 1000 letters per day, we would have needed to achieve an entire system speed of 28.8 seconds per letter. This means our delivery speed would need to on average achieve a speed of ~17 seconds, as dispensing took an average of 2 seconds and image processing an average of 8.6 seconds.

IV. Conclusion

Due to the unfortunate circumstances of the second half of our senior year, the current state of the project accomplished all of the CDR goals we set previously, but failed to meet our FPR specifications. During our CDR presentation, the dispense mechanism was able to dispense one letter at a time with 95% accuracy, the text address reading worked 99% of the time, and our delivery tray achieved the correct mailbox location ~50% of the time. This is because when we implemented the 2-D gantry system, we did not have a third lead screw assembly. We used one lead screw for horizontal movement. The only issue with horizontal movement arose from slight misalignment of the bearing rails that run parallel to the lead screw. We planned to ensure parallelism of this stage by printing brackets rather than building the stage out of wood. The vertical stage requires two lead screws, one mounted on each side of the mailbox. Since we only had one lead screw for MDR, the vertical stage only received torque on one side of the stage. This caused the stage to lag behind on the unpowered side, as evidenced by the tilting and catching of the horizontal stage. Additionally the extra lead screw kit would provide us with two more linear bearing rails which could help even out the torque on the vertical lead screws allowing for smoother, faster operation.

For FPR, we hoped to improve the dispensing and delivery mechanism to an accuracy of 99%, dispensing one letter at a time and delivering one letter to the correct address respectively. We also needed to create the delivery tray to move the letter into the mailbox, which we considered two possible solutions for.

Acknowledgment

Our team would like to especially thank our advisor, Professor Holcomb, for taking the time and effort to provide critical feedback and advice when needed. We would also like to thank our evaluators and SDP coordinators Professor McLaughlin, Moritz, Hollot, Soules, and Caron for their time. Finally, we would like to thank M5, the All Campus Makerspace, and the MIE Machine Shop for allowing us to use their makerspace, various equipment, and materials.

References

[1] *Automation and the Life of the Letter Carrier*, USPS, 10-Oct-2019. Available: https://www.uspsoig.gov/blog/automation-and-life-letter-carrier

[2] “OPEX Mail Matrix Information,” *OPEX Mail Matrix Information*, 02-Oct-2019.

[3] Raspberry Pi (Trading) Ltd, Raspberry Pi Compute Module 3+ datasheet. Jan-2019

[4] National Electrical Manufacturers Association, “NEMA 23 Stepper Motor,” datasheet. Jun-2018.

[5] Texas Instruments, “DRV8825 Stepper Motor Controller IC,” SLVSA73F datasheet, Apr. 2010 [Revised Jul. 2014].

Appendix

A. Design Alternatives

During our design phase, we broke the project up into smaller subsystems when viewing alternatives. There were good design choices for each subsystem that we considered before ultimately deciding on our current implementations.

Processing Unit: We considered using an Arduino Uno, a Raspberry Pi 4, or a BeagleBone Black to be our main computing unit. In the end, we chose the Raspberry Pi 4 for its high computing power for image processing. The Arduino Uno would simply not have enough computing power to support our use case. The BeagleBone Black was a close contender, but it has a lot of GPIO pins that we do not need, which drives up its price compared to the Raspberry Pi 4.

Motors: There were a variety of motors we could have chosen for this project. To drive our 2-axis gantry, we could use DC motors with closed loop positional feedback to move the delivery tray, or use stepper motors with no closed loop feedback. Ultimately, we decided to use stepper motors for our 2-axis gantry for its very accurate step sizes. Even if there were significant errors with the motor step sizes, the error would not accumulate, because it will be consistent every rotation, every rotation will produce the same linear motion. This made stepper motors very appealing to our specification constraints. We did not want to use a DC motor because we would need some kind of closed loop feedback to not have positional errors cripple our product. For our dispensing mechanism, we had the option of using a servo motor for its controlled speed or a stepper motor or its accurate step sizes. Because of its size, cost efficiency, and controlled speed, we chose to use a servo motor instead of a stepper motor in order to dispense letters one by one.

B. *Technical Standards*

A primary standard that was incorporated into the project was that of letter size. Our product is compatible with letters measuring 9 ½ " by 4 ⅛". This is known as a No. 10 envelope, and is the most common standard used by the USPS. Additionally, we filled our envelopes with 8 pieces of paper for testing purposes. This brought us to a letter thickness of ¼", which is the maximum thickness allowed by the USPS. Any larger, and the piece of mail is no longer eligible for shipping at standardized letter prices.

In the case of IEEE, there were no relevant standards to adhere to for this project. The same can be said about OSHA.

C. *Testing Methods*

The primary method that we used to test each of our subsystems was to run each subsystem in isolation and record its success and failure over many trials. With enough data over many trials, we calculated success rates from all of our trial data. We kept improving each subsystem until our data showed a success rate high enough to meet the specifications that we had promised for MDR.

With our image processing subsystem, we manually placed letters in front of the camera, let our code read the letters in different orientations and addresses, then noted in our data if our code was able to identify the correct name in its image processing. Our data came out to 30 trials, with 28 of them being correct, yielding a success rate of 28/30, or 93.33%.

With our dispensing subsystem, we let the dispenser dispense many letters, then recorded if each run was a successful trial with success being exactly 1 letter dispensed. Our data came out to 20 trials, with 16 being successful, yielding a success rate of 80%.

With our delivery subsystem, we manually controlled where the delivery tray should go for each trial, and recorded whether or not the delivery tray successfully travelled to the correct mailbox height. Our data came out to 30 trails, 29 of which were successful, yielding a success rate of 96.67%.

Overall, our testing was heavily guided by the goals that we had promised for MDR. We tested each subsystem with the promised goal in mind. Whenever our subsystem fell short of a goal, we knew that we had to improve it in order to meet the numbers we promised. We were only satisfied after seeing that the data collected in our testing met each requirement.

D. *Team Organization*

Each member of our team has a well defined role. James acts as the team manager and leads in software integration across the subsystems. Long is the team's PCB lead and motor control expert. Harrison is the lead in developing image processing functions. Dan is the team's mechanical engineer and thus oversees the many moving and mechanical aspects of the project. Overall, our skill sets complement each other well.

For much of the year, we worked independently on our given tasks, providing updates to each other when certain milestones were reached. When we did reach a point where we could start integrating our parts together, team communication and collaborative work went smoothly. The members that needed to integrate their parts together met at SDP lab to address any issues that may arise during the integrating phase. During our first semester, we failed to take full advantage of our advising meetings by going in unsure of what material needed to be covered on meeting day. We fixed this by discussing problems and potential solutions in our group chat, which served as a log for our meetings with Professor Holcomb. We all helped each other out by providing opinions on how each subsystem could be improved, serving as a pair of second eyes.

E. *Beyond the Classroom*

This project has been a valuable learning experience for us as young developing professionals. We were able to develop our own project, divide work up into our corresponding areas of expertise, consult other people for advice, present in a formal setting, and work towards one collective goal. This has been the biggest project for us so far, and we are truly grateful for this year-long experience.