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 will feature a hopper system that dispenses one 4 ⅛" x 9 ½" envelope at a time into a delivery tray. From there, a photo will be taken of envelope, and barcode image processing will be applied on the photo to determine the address of the recipient. Finally, the delivery tray will move the envelope to the corresponding mailbox height and return to its original position. A tilting delivery mechanism will be implemented in further designs. 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 will aim 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	≥90 % 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	≥ 75% accuracy

Table 1: Requirements and Specifications for MDR

II. Design

A. Overview

To automate the mail sorting process we will create an autonomous system designed to complete the task mentioned above. The main technologies we will use to solve the problem are 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. 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.

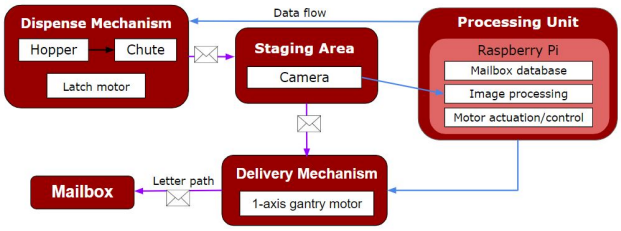

Figure 1: MDR Block Diagram

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 ¼" by 4 ⅛" letters. There is a plate of stainless steel on the bottom of the hopper to reduce friction for the bottommost letter. 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 aluminum sits at the front of the hopper to prevent more than one letter from being dispensed at a time. In our testing for MDR, the dispense mechanism typically achieved a success rate of ~80%, and typically dispensed two letters on the 5th rotation. This systematic error will need to be addressed with a new or modified design of the dispense mechanism for FPR to achieve our specification of "accurately dispenses exactly one letter 95% of the time."

Figure 2: Dispensing Mechanism

C. Processing Unit

For our processing unit, we decided to use a Raspberry Pi 3 equipped with 1.2 GHz processing speed and 1 GB RAM [3]. This will be transitioned to a Raspberry Pi 4 equipped with 1.5 GHz processing speed and 1 GB RAM, but by MDR we did not have the right cables shipped to the lab in time so we could not operate on the RPi 4. All files that we use on the RPi 3 are cross-compatible with the RPi 4. 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 projects 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 MDR, we were able to take a photo of the envelope, locate the position of the barcode from that photo, and decipher the recipient's address from the barcode. This was accomplished using the Pyzbar 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 barcode.py which takes in the envelope.jpg image, converts the image to grayscale, and then locates the barcode and decodes it using the Pyzbar function decode(). The recipient's address is then stored in 'address.txt' for the delivery mechanism to reference.

To test the accuracy for reading barcodes, we conducted a test of 30 different addresses and envelope orientations. Some of the addresses were long barcodes like McLaughlin, while others were short like Hollot. The envelope was barcode was placed horizontally, tilted, and flipped upside down in respect to the camera angle. Out of all 30 tests, the program was still able to decode the barcode 28 out of 30 times for an accuracy of 93.33%.

D. Delivery Mechanism

The delivery mechanism moves the letter in the y axis to the correct mailbox in MDR. For FPR the delivery mechanism will need to move in both the x and y directions. The delivery mechanism consists of one Bosch R146520000 Linear Actuator, one 5mm to ¼" flexible shaft coupling, one NEMA 23 stepper motor, and one stepper motor driver [4]. The linear actuator was found in the M5 scrap room, and has worked well for MDR, but will be replaced by a lead screw for FPR. The current linear actuator catches and causes the motor to stall briefly. This results in 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. If this problem persists we will 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.

We will also need to drastically speed up the rate of delivery for FPR. This will be in part resolved by the new lead screw, which has a larger pitch than the ball screw in the current linear actuator. This means the lead screw converts one rotation into a larger translational motion, but will require more torque. Hopefully this will not be an issue with the large NEMA 23 stepper motors, but with the additional weight of the horizontal gantry, we may need to explore stronger motors.

Another potential challenge in building out the two axis gantry for FPR will be the precise alignment of all the components. We will need to achieve perfect parallelism between the guide rails and the lead screw to ensure smooth operation.

For FPR we will also need to implement a mechanism to move the letter from the delivery tray into the mailbox. We have considered two methods, a tray tilting mechanism or a letter pushing mechanism. Part of this decision revolves around whether we will consider if the mailboxes are full, and how we process this knowledge. If we need to place letters on top of a partially full mailbox the letter pushing mechanism will be the better solution.

E. Motor Driver Circuit Board

For MDR, we opted to use a product available on the market, the Pololu DRV-8825 stepper motor driver chip [5]. We only needed to get one stepper motor running, and we needed to do so as soon as possible to get our prototype started and meet MDR deadlines. Because of this, designing and manufacturing our own custom stepper driver would not be feasible, so we opted for a product already available on the market. Now that we can drive one stepper motor with one driver chip, we can scale it up to driving four or more motors on four or more chips as needed.

Our next step will be to move away from products available on the market. Currently, the only motor driver chips on the market at chips that only support one single motor driver. This means that if we needed to drive five motors, we would need five corresponding motor driver chips. To move away from this messy solution available, we plan to design a custom PCB to hold our own stepper motor drivers. Centralizing all of our motor drivers onto one single PCB carries two significant advantages: easy and simple wiring and better centralized cooling for heat dissipation. Having all of our motor drivers together on one board will allow us to simply wire the whole board to our RPi, instead of having four or more motor driver chips all wired back to the RPi from all directions. Putting all of our motor drivers onto one centralized PCB will also allow us to implement better cooling solutions as needed. If we needed to, we could simply use one big fan to cool the whole custom PCB, instead of having to use four or more smaller fans, each fan distributed to each motor driver chip. After we finish implementing our custom PCB solution for the motor drivers, we can start to explore microstepping to smooth out the motion of our stepper motors. As of right now on our prototype, our stepper motor makes too much noise and vibrations during its operation. Microstepping will help reduce both noise and vibrations, but will also reduce the torque on the motors. We will need to find a balance between noise, vibrations, and torque with microstepping.

We did not have to do much testing to verify the function of our motor driver. It either works or it does not, and it can turn on or it cannot turn on. If the motor driver can turn on power for the motor, thus making the motor move, then we knew it was working. We encountered no issues with this part of our prototype.

III. Project Management

We have accomplished all of the goals that were set for MDR and successfully demonstrated our prototype. Our prototype currently incorporates a 1-D array of mailboxes, stacked vertically on top of each other. We have a staging and delivery tray that is mounted on a 1-axis gantry which

allows us to move the tray along the mailboxes. We have a hopper that holds a stack of letters that can dispense letters down a slide onto the staging tray. We have an RPi camera that takes an image for image processing. Finally, we have an RPi that processes the image to match the letter to a mailbox, then sends signals to the motors to deliver the letter to said mailbox height. All of these components functioned together for a successful MDR demo. With that said, our product still needs to be polished both in terms of aesthetics and functionality. First, we need to scale our prototype to support 2-axis movement instead of 1-axis. Second, we need to implement a final delivery mechanism that deposits the each letter once it arrives at the appropriate mailbox. Third, we would like to implement typed text reading for image processing, These are the most immediate 3 goals for us. After completing these 3 tasks, we will still need to polish the final product aesthetically as well as sure all subsystems operate smoothly together.

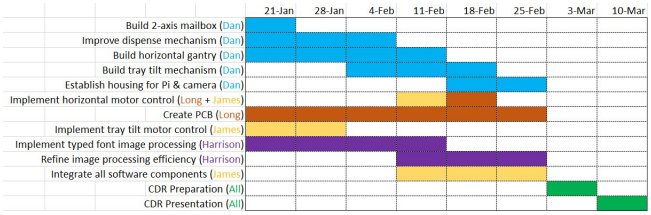

Figure 3: Gantt Chart

James: Timing and control logic, software architecture
Long: Motor control and actuation, hardware architecture
Harrison: Image processing, text recognition
Dan: Mechanical systems

Dispensing Mechanism: Dan Lead, Long Support

Dan will produce a hopper to hold the letters, Long will a motor driven mechanism to dispense one letter at a time. Dan will also produce a slide or chute to guide the dispensed letter to its staging area.

Staging Area: Dan Lead

Dan will implement a staging tray to hold the dispensed letter while its information is processed. This tray will also be part of the delivery mechanism, moving along the 2-axis gantry.

Processing Unit: James and Harrison Lead, Long Support

James will be the software system integrator, stitching together all the software pieces from Long's motor control and Harrison's image processing. James will ensure that software operates sequentially and as a cohesive unit by implementing proper timing constraints and checking all edge cases.

Harrison will be working on the image processing and RPi camera aspect of the project, implementing typed text recognition through image processing and making sure it is able to be processed quickly and efficiently.

Delivery Mechanism: Dan Lead, Long Support.

Dan will produce the 2-axis gantry to mount onto the mailbox array, Long will control the motors to move the delivery tray along the 2-axis gantry.

Dan will also produce a motor driven tilt mechanism for the tray to put the letters into its mailbox, Long will control the motor to tilt the tray to deliver the letter.

Motor Drivers: Long Lead

Long will produce a custom PCB to house all of the necessary motor drivers, as well as the code to run these motor drivers.

Mailbox: Dan Lead

Dan will construct a mailbox array or source one from UMass Mail Services.

IV. Conclusion

The current state of the project accomplished all of the MDR goals we set previously. During our MDR demo, the dispense mechanism was able to dispense one letter at a time with 80% accuracy, the barcode reading worked 95% of the time, and our delivery tray achieved the correct mailbox height 100% of the time.

In the future, we hope to improve the dispense mechanism to be able to dispense one letter with 95% accuracy. We are also working to implement typed text image processing so that our project is applicable towards more standard forms of address writing. We will also create a delivery mechanism to deliver the envelope into the mailbox.

Acknowledgment

We would like to especially thank our advisor, Professor Holcomb, for taking the time and effort in providing critical feedback and advice when needed. We would also like to thank our evaluators and SDP coordinators Professor McLaughin, Moritz, Hollot, Soules, and Caron for their time. Finally, we would like to thank M5 for allowing us to use their makerspace, various equipment, and well stocked 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 was 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 have the option of using a servo motor for its controlled speed or a stepper motor or its accurate step sizes. Our prototype is currently using a servo motor, which worked well enough for MDR, but if consistent rotational motion becomes an issue as we try to achieve a higher dispensing accuracy, we could switch out the servo motor for a stepper motor.

We will replace the Bosch linear actuator with lead screws and linear guide rails for FPR. These lead screws should move faster than the linear actuator and will require lots of care to properly mount and align. They will also require more torque to move compared to the linear actuator.

B. 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.

C. Team Organization

Each member of our team has a well defined role. James acts as the team manager and leads in software integration. 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 semester, we worked independently on our given tasks, providing updates to each other when certain milestones were reached. When we did reach a point at which 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. Nonetheless, there are ways we plan to further improve our team organization and communication next semester. Most importantly, we will focus on organizing our thoughts more effectively prior to our weekly team advising meetings with Professor Holcomb. Often times this 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 plan to fix this by scheduling a team-only meeting the day before our advising meeting. Having these two weekly meetings should allow the team to work much more efficiently.

D. Beyond the Classroom

This project has been a valuable learning experience for

us as young developing professionals. We are able to develop our own project, divide work up into our corresponding areas of expertise, consult other people, communicate and all work towards one collective goal. This has been the biggest project for us so far, and even though we are only half way through this year long project, we have already learned a lot about ourselves and how to function effectively as a team.