Package Sensing Prototype

Haiden Cramer, Eric Frazer

University of Montana

CSCI491

Anh Nguyen Ph.D

12/11/2024

# *Project Report*

# Abstract

This project aims to explore a new type of device that can record real-time data for package deliveries. With this project we intend to record our data using multiple different data types. These data types include Temperature, Light Levels, Gyrographic, and Acceleration. Temperature can be implemented in packages where keeping the contents inside need to be within a certain range. The Gyro and Accelerometer are going to be used to make sure the package isn’t being tossed around in the process of delivery. Finally our light sensor will measure whether or not the box has been opened.

# Project Report: Package Prototype

The goal of this project is to provide another level of transparency between buyers and sellers of online goods using means of collecting data during the shipping and handling processes. We thought about this idea because of hearing about a story of someone who had ordered computer parts online, and when they received them and opened the box the parts were non-functional, and it was too late to return them. This seemed to raise a big question during the shipping process of how customers so willingly trust someone else to ensure their goods are not broken, stolen, or tampered with. Our solution to this was to create some sort of device that is not only practical, but also delivers on some of our initial ideas, and that is where we decided to create a device to give accurate readings of a package, and the state that it is in during the whole shipping process. There already exists a product used for larger containers called Shock/Impact indicators and they are able to tell if a package had been dropped or crushed, but we wanted to see if we could implement our idea on a similar, but smaller scale. We want to create a system that aims more around data collection for packages containing software parts, fragile items, glass goods, etc. Due to our group size being just two people, we decided that working on each step together would be helpful in tackling the challenges this project proposed.

## 

## Proposal: Package Sensing System

To get our idea together, and start implementing knowledge of what we had learned so far in IOT, we decided to order a few components. To start we needed to decide whether or not our package had been opened which is critical in the delivery process to ensure the package hadn’t been tampered with. To achieve this we needed to order a light sensor which we obtained from Adafruit, and we chose the Adafruit BH1750 Light Sensor. During this process of deciding what sensors to order we decided we would need another sensor designed for accounting for Gyrographic, Acceleration, and Temperature data. For this part we ordered it with the Light Sensor, and it’s called the Adafruit LSM6DS3TR-C 6-DoF Accel + Gyro IMU Sensor. After obtaining these parts we needed a way for them to communicate with each other, and the best route was to use a serial connection between the Raspberry Pico, and Raspberry Pi Zero we obtained from our IOT class.

To use these components all together, and in sync we needed somewhere to display the data that was collected, and so we made the decision to host it locally through Apache (external server tool). The communication flow is as follows: Raspberry Zero (Gyro/Accel/Temp) → Raspberry Pico (Light Sensor via Serial Port) → Raspberry Zero → Locally Hosted Server (Using Apache). This allows us to send a command from our master program which was created on the Zero to allow the Pico to communicate its data back and then to be displayed on a server. The data’s input sources come from the area’s Temperature that the device is contained in, Axis data to tell the position of the box relative to itself, Light data to tell whether or not the package had been opened based on light levels received, and finally the Accelerometer data to let us know if the package has been shifted drastically. With all of these components combined we were able to develop the current system we wanted to achieve.

### Challenges Faced

During this project we faced many obstacles, but worked together using critical thinking skills, and prior experience as software engineering majors to overcome such challenges. Initially, our first big challenge was deciding how we wanted to go about collecting data, and deciding what sorts of tools we were going to use to not only collect our data, but to output this data in a readable, and user-friendly way. We spent a lot of time looking for different sensors that fit within our system and once we settled with a few, and ordered them then we didn’t have to order anything else.

Initially, getting the sensors connected to the Raspberry Pi and Raspberry Pico wasn’t extremely difficult, but making them communicate together was proven to be a difficult endeavor. For starters, we ran into a hurdle with our light sensor in the very beginning phase of our project, and had to read a lot of different documentation that would explain to us what the issue was. After we figured that out we had to decide how we were going to establish communication between the Pi and the Pico which in our case turned out to be a serial connection. At the beginning of this process we were having difficulties getting the connection to properly establish, and it was proving to be stressful as we had attempted many different ideas. Talking to our TA Nhan gave us a lot of insight into what the issue could be, and without his help at that time we might not be where we are currently.

After the connection issue was solved, and we had data from the Pico being sent to the Zero we needed somewhere to display the data, and keep track of specific occurrences of events that a user would want to be alerted for. Using Apache was almost seamless as we only needed to download it, and run a command on linux to allow the server to be locally hosted, but getting the data to correctly format was a massive challenge that took us multiple attempts at achieving what we wanted. Our last, and current challenge we are faced with is figuring out a way to implement a pressure sensor to take into account if our package had been crushed or not. We never came up with a good solution for this, and decided that the inaccurate readings from the pressure sensor would not be important to our data analysis. All in all this whole system probably took us about One to Two months to get fully functioning, and operating as we handled the tasks detailed here on a weekly basis due to other labs we were working on in IOT. As we approached the last lab this freed up a lot of time that we used to finish up the rest of our working prototype.

## 

## Prototype Development

For the first round of our project design, we used a simple layout with our breadboard and all the devices attached to it through their respective wiring schemes. We started by testing each component for connectivity, followed by getting readings from each device. The light sensor took the most time to get running, as the libraries for it were not native to our hardware and arduino. This process took a couple weeks, but once we had all the components wired properly and working we began testing thresholds and behaviors from the sensors. It should be noted that during this time we decided to remove the pressure sensor from the prototype, and instead implement logic for the temperature sensor built into the gyro/accelerometer. This decision was based on the data being gathered not accurately reflecting the forces acting on the prototype. This process lasted about a week, where we ran tests on the sensors to gather data we could analyze.

After finding out the thresholds for our gyro/accelerometer sensor, we started to bundle the components in a compact scheme, allowing for them to be placed in a small box (acting as our prototype housing). After ensuring the wiring was correct and the device could function, we added the ability to run independent from an external power source with the use of a battery pack. The next step in the prototype was to take the data being gathered and display the analysis conducted on the raspberry pi to a localhost webpage that stands as the user interface for the device. We began developing a basic webpage to show the sensor data live from the raspberry pi, and upgraded the functionality to show when certain sensors found their thresholds had been surpassed. With each event being timestamped and placed into a log, the next and final step was to incorporate an easy to understand color bar, signifying the severity of the actions taken upon the package. We finished this with the use of green for the package being untampered, yellow for an unknown state showing that the package has experienced distress, and finally read for if the light sensor was triggered and the package was opened.

## 

## Evaluations and Results

After running testing on the components and having previous package handling experience, we tested the final product to see what readings would need to be registered by the device, and which should be flagged by our software. The gyro measurements in the x, y, and z axes have a threshold set to 1.5 rad/s, while the accelerometer values for x, y, and z are compared against a threshold of 1.5 m/s². The temperature is measured in Celsius, with high and low thresholds set to 38°C and 0°C, respectively. All values are rounded to the nearest second decimal place. The system performs very well, and the light sensor is able to detect punctures in the box from a pen-sized hole. The user-interface allows for both an easy-to-comprehend and extensive range of data, giving the user detailed insights into the performance of the package and its environment. Alerts are triggered when the thresholds are exceeded, ensuring that potential damage or unusual conditions are quickly flagged.

**Lesson Learnt and Conclusion**

In the end of this project we both agree that we learned a lot more about how a system running different electrical components operates, and communicates with each other. There was a massive stretch of using critical thinking skills, and figuring out different ways to think about some of the issues the system was facing in the beginning stages. Overall this was a great opportunity to learn and operate our very own IOT device. Some of the other components involved in this project made us better programmers, and helped us gain further insight into the world of computer science.

**System Setup and Operation Guideline (included in README.md)**

<https://github.com/haidencramer/PackageSystem?tab=readme-ov-file>

For this section Haiden created a GitHub repository that has all of our files, screenshots, link to youtube video, and other source code used for testing. Feel free to check this out as it details the system setup as well as a step-by-step guide on how to get the main program up and running. GitHub was the best option because it gives other users a chance to pull our setup directly from the website, and hopefully help us create a better product in the future.

**Link to Demonstration Video** <https://youtu.be/E0emrvPqTsI>

**Group Work**

For this project there were only two of us in a group, Haiden Cramer and Eric Frazer so it was best that we both worked on every single part together. A huge part of our communication outside of class can be attributed to discord where we can work together and share files back and forth to troubleshoot. GitHub was also another option for us to use but we ended up implementing that further towards the end, but it is a great resource when working on projects with a big code flow. Working together on every task, and not moving forward without the other groupmate knowing was the most efficient way for us to achieve our final prototype. Personally, we both agree that smaller group sizes might be best for a project such as this. As far as delivering both of our components we both worked really hard to make sure that we could turn our idea into a real working prototype.

# 

# 

# 

# 