MECENG 100 Final Project Fall 2024

The SmartCycle

By: Team Astro

Nathan Luis Saldana , Larry Bermudez , Rayme Aquino

Table of Contents

1 Introduction

More than half of the world's population is unable to ride a bike, especially one that requires manual gear changes. Despite being more user-friendly, electric bikes with advanced equipment can be excessively costly with an average price of $1,000. Additionally, commuting with gears can be dangerous, and bike theft is a common issue on campuses and in urban areas. By offering non-electric bikes an inexpensive modification that makes them trackable, safe, and simple to operate, the SmartCycle project tackles these issues. The primary objective of the SmartCycle project is to use the Internet of Things to develop a smart system that will improve a user's safety, convenience, and accessibility. By integrating multiple sensors and actuators including an OLED Display, accelerometer, GPS Module, keypad, and several DC Motors, the SmartCycle provides the user with real-time data and automated responses which improves the cycling experience for all.

2 System Design

The SmartCyle uses a multitude of sensors to achieve our goals. Each component was selected based on accuracy, resolution and performance. Below is a comprehensive list of the components that were used as well as their individual use:

1. 3 ESP32 Feather V2 microcontroller boards per each member. The first ESP32 was responsible for controlling a DC Motor to automatically shift the gears of the bike depending on the incline angle measured by an LSM6DSO accelerometer that was used as a gyroscope. The accelerometer also sends temperature and the incline angle measured to the second ESP32
2. The second ESP32's main function was the UI that used an SSD1306 OLED Screen to display various data readings. It also controlled the GPS Module so that the user is able to see their speed, the date, and time on the OLED Screen as well as flashed a red LED light indicating the incline angle has been exceeded and so a gear shift has occurred. It is also possible to display the latitude and longitude coordinates obtained from the GPS. Both the OLED and GPS Module had their own library ensuring its compatibility with the ESP32 and MicroPython.
3. The last ESP32 was responsible for our locking mechanism controlled by another DC Motor that incorporated a keypad as well as an email sendout depending on whether the code to unlock/lock the mechanism was correct. The Keypad also contained its own library.

3 Design Methodology

To demonstrate the SmartCycle we decided on the following considering the limitations imposed upon us:

- For the locking mechanism we used a 0.96 OLED I2C screen,a 4x4 keypad, DC Motor with Encoder, and a CAD Design that took the rotational motion of the motor and turned it into translational motion to clamp onto the wheel. We also have a feature that sends emails to the user(s) when the bike is unlocked, locked, or an incorrect code is inputted.
    - The smaller OLED screen fit perfectly on the breadboard and was still large enough to show a user what passcode was being inputted. By using I2C we were able to only use 4 pins and allow for a much simpler design.
    - The DC Motor was supplied with a 12V battery that provided the power necessary to operate the locking mechanism. A smaller motor, and smaller power source would not have been sufficient to open/lock the mechanism.
    - We originally wanted to attempt to implement a fingerprint sensor that would unlock the device, however, after much consideration, we decided upon the keypad because it allowed us to have 16 input values and only use 8 GPIO pins.

Furthermore, if were to manufacture the SmartCycle in a commercial setting this is what we would change or improve:

- Ideally, we would combine the functionality of the keypad and OLED into a single larger OLED screen. This OLED screen would have a touch screen capability, working similarly to a phone's lock screen.

- The CAD design for the clamping mechanism would be redesigned to be smaller and fit better around tires. Instead of it being completely 3D printed, we would use different manufacturing processes as well as different materials such as aluminum so that there is a balance between strength, manufacturability, cost, and also durability.
- Instead of sending an email, ideally we would have a method to send push/notifications to a phone, asking to receive verification confirmation messages.

4 Communication Protocols

The communication protocols between the ESP32's were learned completely from the course. Implementing ESPNOW that was introduced from Lab 4, we set up the Sender and Receiver ESP32 program between the ESP32 detecting the movement of the accelerometer and the other that was displaying information on the OLED Screen. The ESP32 that is connected to the accelerometer sends information on the incline detected and its surrounding temperature every second. The ESP32 connected to the OLED screen then receives this information and displays it in the OLED and updates in real time as new data is received. The ESP32 controlling our locking mechanism uses SMTP (Simple Mail Transfer Protocol) to communicate with the user(s) based on the code input. When a specific passcode is inputted into the keypad of the ESP32 it will send an email to the user stating whether the bike was locked, unlocked, or incorrect code was inputted.

5 Outcome and Conclusions

After successfully implementing and debugging our code, 3D-Printing the locking mechanism and assembling each ESP32 onto the bike we went outside to test each component ensuring the OLED Screen updated in real time according to the GPS Data and the data being received by the accelerometer mentioned earlier. It was also crucial that our locking mechanism functioned properly and so it needed to be mounted strategically to ensure it did not move while operating as that was causing us some issues. These issues could have been because of the lead screw we used to convert rotational movement into translational movement through the motor. The lead screw was also 3D-Printed along with the slider that moved forward and black to close the mechanism. By 3D-Printing threads as opposed to having them machined and the hole tapped, it may have caused issues with the sliding motion. Although we were able to successfully mount and test it, in the future we may invest more time into designing a new motor housing and mounting structure for it. The gear shifter worked very nicely and did make riding the bike much easier when going up inclines through the automatic upshifts as well as going downhill with the downshifts. We were also able to demonstrate our knowledge of course material through the implementation of techniques learned through the labs such as ESPNow Communication, I2C, as well as DC Motor control and encoding. Overall we were all extremely happy and proud of our work as we were able to create a project we thought of and designed completely ourselves and believe it could be an even better project given more resources.

7 Project Video and Presentation

Project Slides

8 References

[1] Luiz. (2024, August 15). *MicroPython: ESP32 with NEO-6M GPS module*. Random Nerd Tutorials. https://randomnerdtutorials.com/micropython-esp32-neo-6m-gps/

[2] Festing, D., Santos, S., Maximiliano, Aart, Charles, Javi, Wouters, M., & Diego. (2023, January 26). *MicroPython send emails with esp32/ESP826*. Random Nerd Tutorials. https://randomnerdtutorials.com/micropython-send-emails-esp32-esp826/

[3] Medienverbinder, Santos, S., Ng, M., 9a3xz, M., Filip, Dehmel, M., Caneda, E., Tim, Michal, Paul, S., Escasa, D., Till, Maurizio, Antonella, Eyal, Cyril, Paul, Serhii, Ezekiel, … Sina. (2023, July 5). *MicroPython: OLED display with esp32 and ESP8266*. Random Nerd Tutorials. https://randomnerdtutorials.com/micropython-oled-display-esp32-esp8266/

[4] PyPI · the python package index. (n.d.). https://pypi.org/project/micropython-simple-keypad/