Executive Summary

Autonomous Package Delivery Robot

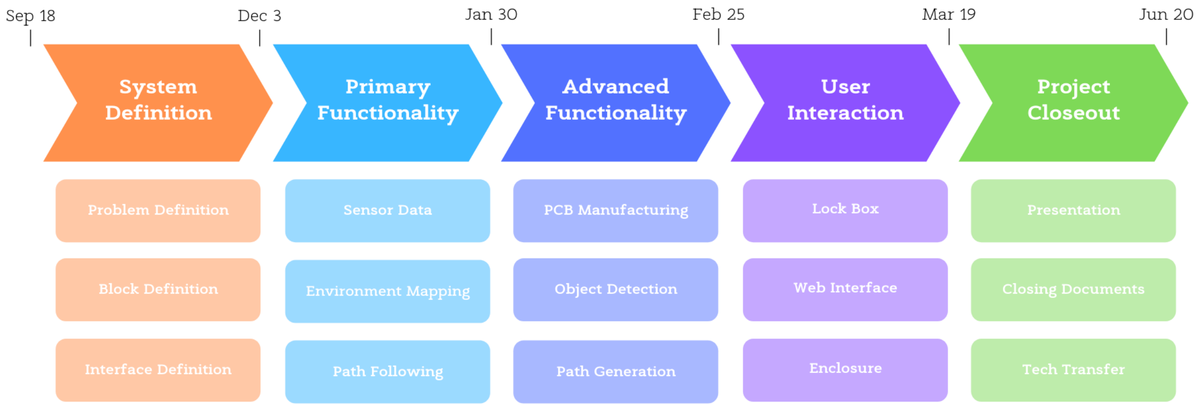
ECE Group 025

Nick McBee, Drew Gehrke, Andrew Pehrson, Nathan Searles, Tyrone Stagner

The purpose of this project is to create a robotic package delivery system operating in the context of an environment with well-developed pedestrian-tailored infrastructure, such as a college campus. The Autonomous Package Delivery Robot (APDR) will be capable of carrying packages while autonomously navigating along sidewalks and avoiding obstacles to reach its final destination. The scope of this project also contains a user interface in the form of a website that will allow individuals to initiate and receive deliveries at a specified destination.

The goal of this project is to join the increasing number of autonomous delivery robots that provide contactless deliveries of food and goods to customers. This project will also introduce a solution to the rising issue of electronic waste, which will be achieved by using recycled electronics such as the base, motors and batteries of an electric wheelchair.

This project was inherited from a previous Oregon State University EECS Capstone group (2020-2021). This team will be working with Hanna Anderson, project sponsor, and previous team member on this project. In its current state, the robot is capable of movement under manual control, avoidance of stationary obstacles, and waypoint creation using GPS. The technical goals for the team inheriting this project are developing a secure package delivery system and increasing the capability for autonomous outdoor travel of the APDR. The developed product will be incredibly aware of stationary objects and dynamically moving pedestrians and vehicles, as well as provide an intuitive and reliable courier service to distributors and customers alike.



Project Timeline

Many changes and improvements have been made to the system by the new team. The APDR system now has a way to store packages in its lockbox mounted right on top of the electric wheelchair base. Other hardware changes include the addition of a team made PCB and new circuitry to properly distribute power to all the various electronic components of the system. Another key change is that many of the sensor modules have been off-boarded from the Raspberry Pi and are now processed on an ESP32 microcontroller. This allows for more processing speed on the Raspberry Pi. The inertial measurement unit, or IMU, and global positioning system, or GPS, aid in the navigation of the robot by sending the data first to the ESP32 to be processed, which the Raspberry Pi then receives and sends to the various topics which require the data. Speaking of topics, the entire system has been migrated from the first version of the Robot Operating System, or ROS, into the newer version, ROS2 Galactic. Many custom topics have been developed to get the APDR system working, including a MCU (microcontroller unit) driver, USB-to-Serial driver, motor controller driver, and many more. Several other topics have been utilized to aid in traversal, such as the navigation stack built into ROS2 and the robot translocation topic.

The team learned many valuable lessons over the course of this project. One of which is team communication and collaboration. The team set up a form of communication extremely quickly and were able to continuously update each other based on progress and easily ask questions of each other. Ensuring every team member was aware of due dates, hardships, and any other concerns with the project were easily voiced with the use of a Discord server as our primary form of communication. Another valuable lesson learned was time management. As this project lasted several months, the tendency to procrastinate was apparent. However, the team was able to consistently reach deadlines and have effective progress every week. Another big part of time management with the project was the time it took to integrate each component into the entire system. The team learned that it is important to integrate components as early as possible. This is to allow for debugging of any potential problems in the system and to allow for sufficient time for testing.