Simultaneous Localization and Mapping (SLAM) for Person Mapping

Group Members

Saad Rafiq - SLAM Integrator
Mason Melead - Depth Detection
Rose Ochoa - User Interface
Colton Richards - Person Detection
Jimmy Abouhamzeh - Flask API

Repository Link: https://github.com/SrBlank/SLAM-Person-Mapping

Project Overview

This project uses Simultaneous Localization and Mapping (SLAM) to track and map people in real-time within a changing environment. By combining a stream of footage and machine learning, the system will identify and follow people as they move, updating the map continuously. This project can be applied to areas like indoor navigation, autonomous robots, and smart surveillance, with a focus on reliable human detection.

Introduction

Term	Description
Hector-SLAM	Specific version of SLAM used maps environments in 2D
Occupancy Grid	A 1024x1024 array (in our case) with each index of the array being one of three values: 0 - free space, 100 - occupied space, -1 - obstacle
Robot Operating System (ROS)	A popular framework for robotics which works on a publisher/subscriber model. Users can publish data (occupancy grid) and gather it when needed (subscriber)
Intel Realsense D345i	Depth camera of choice

Unmanned aerial vehicles (UAVs) have emerged as versatile tools for exploring and mapping environments. Combining UAV mobility with advanced sensors enables the development of systems capable of detecting and locating individuals in real time. This paper presents a project that integrates an Intel Realsense camera with a drone to perform human mapping using Hector SLAM. The project focuses on balancing computational efficiency with mapping accuracy, ensuring scalability for practical deployment.

Methodology

Subsystem	Tools Required
Depth Detection	Hardware: Intel Realsense D345i Software: PyRealsense2
Person Detection	Hardware: Any camera Software: PyRealsense2, Mobilenet-SSD Compute: MSI GL745 Leopard Laptop
Application Programming Interface (API)	Software: Flask
Hector-SLAM	Hardware: RPLidar A1 Software: ROS Melodic Compute: Nvidia Jetson Nano
Graphical User Interface (GUI)	Software: Pygame

The subsystems for this project included depth detection, person detection, an application programming interface (API), Hector-SLAM, and a user interface. Table 1 shows the requirements for each subsystem. The High-Performance Engineering (HiPE) Research Lab provided these resources. The following steps outline our methods for tackling this project.

The first step in our project was to successfully gather person detections and their depths since the project is reliant on this data. We chose the application of autonomous indoor drone mapping and thus mounted the Intel Realsense on a drone. We installed the necessary drivers to get the camera up and running. Basic RGB output was working, the next step was to choose, load, and inference on a model. We chose Mobilenet-SSD for our model as it was easy to integrate and fast for person detection. Finally, once we had the bounding box, we were easily able to find the middle and capture the depth using the pyrealsense2 library.

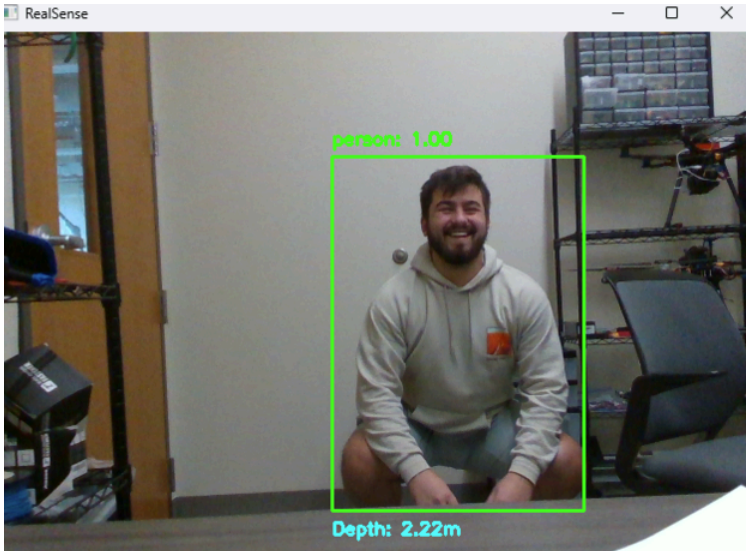
The issue we came across was that pyrealsense2 uses Python 3.6 - 3.11 however our Hector-SLAM setup uses Python 2. Hector-SLAM was already setup and running on the drone by the HiPE Lab so we could not change this configuration. The overcome this issue we created an API to publish the person and depth detection data and capture it by SLAM when needed. We used the Python flask library to accomplish this. It is essentially a loop always inferencing the frames and pushing it to the API waiting for requests.

Simultaneous Localization and Mapping (SLAM) for Person Mapping

The third component of our project was grabbing the person, detection, and SLAM data and mapping it onto the occupancy grid. This was convenient as we know the exact size of each index of the array which in our case was .05 meters. This information is published on */map* as well as */slam_out_pose* which tells us where the drone is in the grid and which way it is facing, the yaw value. By taking the cos and sin of the yaw and multiplying the distance detected we can find the true distance a person is from the drone. Once the position is found we can introduce a fourth value to the occupancy grid, 50, which represents a person.

Finally, we have all the components required for the last subsystem. We utilized Pygame to draw the 1024x1024 grid assigning colors for each value of the occupancy grid: 0 - green, 100 - red, -1 - grey, 50 - blue. An issue we came across was the amount of time it takes to process and display the large grid to solve this we created a mask to isolate the relevant cells (0, 100, 50) and cv2.boundingRect to compute the smallest size grid needed which converted the 1024x1024 grid to 428x428.

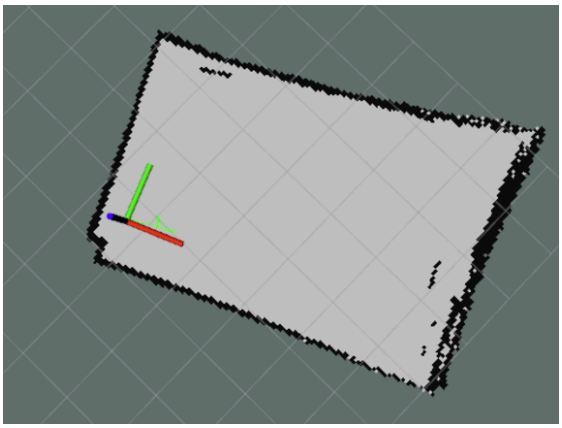
Results

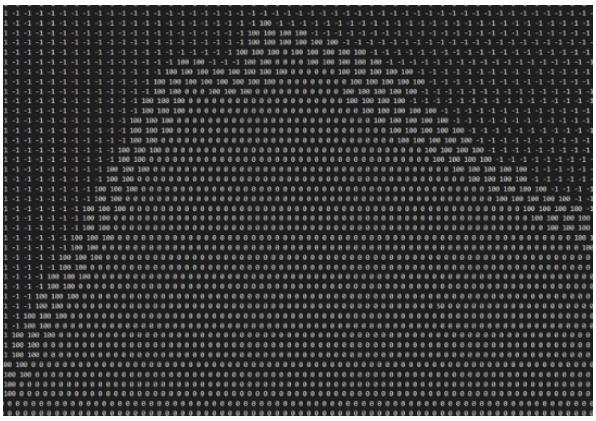


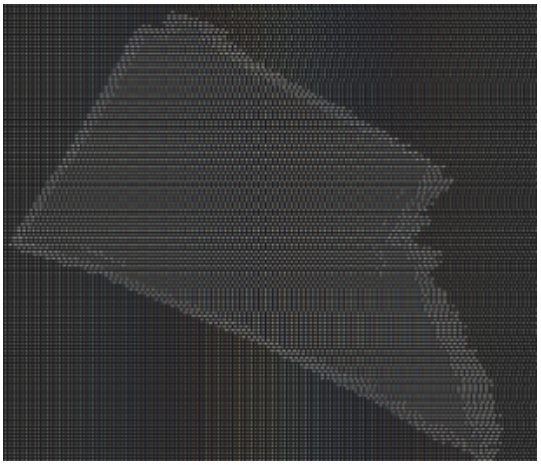
Person and Depth Detection Results
The image to the left shows a successful person and depth detection. We can observe that our first subsystems work accurately as Mobilenet-SSD is 100% confident in its detection as well showing a depth of 2.2 meters.

Application Programming Interface (API)
The image to the left shows the success of our second subsystem, the API. The API is successfully updating showing the confidence of the model's detection and the distance the person is from the camera.

Simultaneous Localization and Mapping (SLAM) for Person Mapping

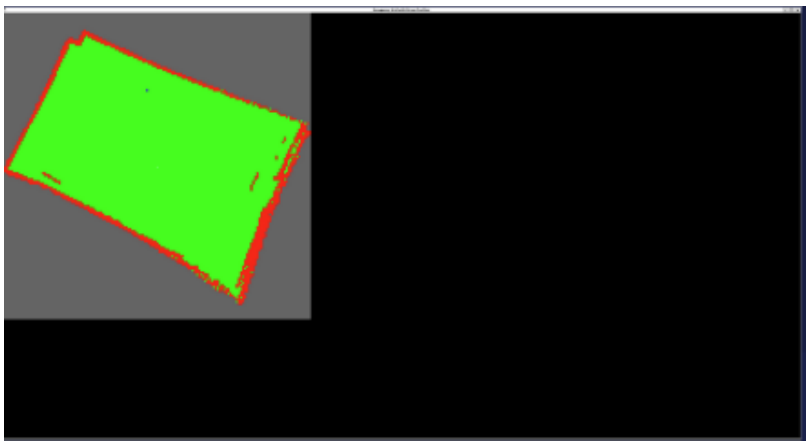






Hector-SLAM and Plotting Results

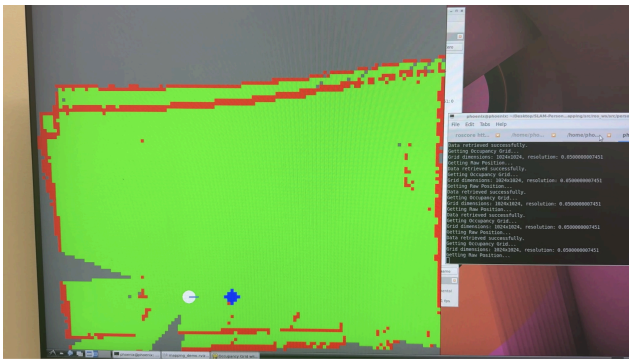
The image on the top left shows the base Hector-SLAM and its initial scan of the environment. From there we are working with the top right and bottom left images which are the array and its values. While difficult to see the middle image shows the value 50 successfully being plotted in the array.

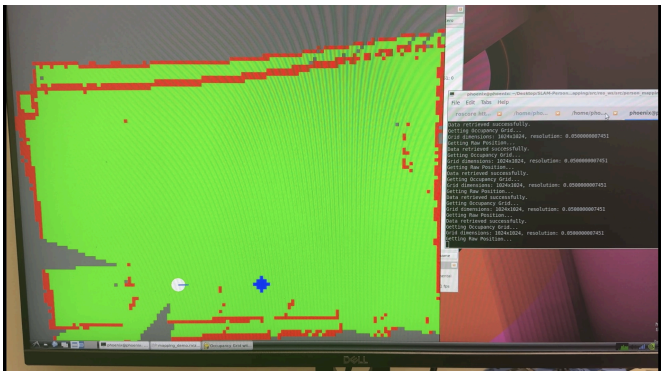


Graphical User Interface (GUI) Results

Here we can see our modified occupancy grid being plotted as well as the person. This shows the initial results of all of our subsystems working as expected. We can also observe the significant shrinkage of the grid for faster computing. We can now test the system as a whole.

Simultaneous Localization and Mapping (SLAM) for Person Mapping





Final Results

Here we can see all subsystems put together, this is running in real-time. A person is detected, the depth is perceived, and published the API. Hector-SLAM map is updated then the data is plotted on the occupancy grid and finally updated on the GUI. We can see in the above figures that the person is moving back and forth which is being updated in near real-time.

Conclusion

This project combines cutting-edge technologies—Hector SLAM for mapping and MobileNet-SSD for human detection—to create a system capable of identifying and mapping people in real time. By integrating these technologies into a drone, we provide a practical solution for scenarios where quick, accurate situational awareness is critical. The system is lightweight and efficient. The design makes it suitable for various real-world applications, including search and rescue missions, crowd monitoring, and security operations. Through this project, we aim to demonstrate how drones can become more than just tools for aerial imaging—they can actively interact with and understand their environment. The results of this work can lay the foundation for more advanced systems, helping improve safety, efficiency, and decision-making in complex situations.