DRONE VIDEO

I. Abstract

Our goal is to be able to deploy a vision system for an unmanned aerial vehicle, specifically the Intel Aero Drone, using its onboard equipment and any additional sensors we might need to achieve a 3D mapping of its environment along with labeled objects. We plan to test this in a lab by setting up some items and boxes and we will determine how accurately we are identifying the objects and also how accurate our mapping of the environment is by comparing it to ground truths. Our implementation will also be tested on various publicly accessible databases. Additionally, the goal is to develop the system to run in real time for a cost and computationally inexpensive implementation while also performing at state of the art accuracy. Our system is meant to help aid in extending the capabilities of autonomous, unmanned vehicles by creating a 3D map of the surrounding environment along with calculating potentially useful information. The system will be able to measure the distances between objects and the dimensions of objects in its view. This approach will build off existing Simultaneous Localization And Mapping (SLAM) systems to accommodate for our needs and hopefully remedy their shortcomings for our application. Lastly, it will be able to create labels for common objects in the view to determine how to interact with the environment if necessary. This will be done through training an object detection algorithm with a suitable dataset that matches common object a drone might interact with such as doors, tables, hallways, and people.

II. Objectives

Our group is working on a solution to let a drone map out a 3D environment all while extracting objects and labeling them from that map. We separate this task into two main subtasks, first we construct the 3D map from RGB video and depth images. We also work on object detection both trying to detect objects from RGB video, but also working on trying to detect directly from the map. Once both of these parts are done we combine to create a coherent map whose objects are correctly labeled and also provide information like the distance between them.

III. Materials and Methods

1. 3D Mapping

-In order to implement our solution, we first need to accurately build a 3D Map of the drone's environment. This is accomplished using a modified version of ORB-SLAM2, a SLAM library for Monocular, Stereo and RGB-D cameras that computes the camera trajectory and a sparse 3D reconstruction, and turns it into a PCD file format so that it can be analyzed using the Point Cloud Library tools.

- Extracting points from images before sending them to the computer on the ground to act as a form of data compression. This more evenly distributes computation between computers and can allow for adding multiple drones in the future to map out even larger areas. No externally libraries are used for this.

- To reconstruct the 3D map using PCL, we are first converting the data to PCD. We are using Euclidean Cluster Extraction in order to reconstruct the point cloud map and organize similar points into the same label. Ultimately we are going to perform object detection and label each of them in a box.

2. **Object Recognition**
   - We are taking different approaches to find the most effective and accurate method to perform object detection. YOLO, TensorFlow are main methods we are testing to accomplish the goal. After getting the basic object detection to work, we implement the code with OpenCV tracker to improve the detection consistency and performance of detecting partially occluded objects.

     - YOLO: This model is based on python platform. It goes through the input images or videos only once, and it builds up bounding box around the objects and extracts features from them. Then the model compares the features with the trained datasets to classify the name of the objects. To run the model, Nvidia CUDA is needed, as well as some python libraries such as torch and tensorflow.

     - TensorFlow: In order to perform object detection accurately and efficiently on our drone, we used TensorFlow object detection API to achieve this task. Written in Cuda, C++, and Python, the machine learning platform is capable of identifying multiple objects in a single image and can be modified and implemented on videos and livestreams. In order to run the object detection API, a trained model, which contains objects of interest that we try to detect, is necessary in order to perform detection.

<u>IV. References</u>

- R. Mur-Artal and J. D. Tardós, "ORB-SLAM2: An Open-Source SLAM System for Monocular, Stereo, and RGB-D Cameras," in *IEEE Transactions on Robotics*, vol. 33, no. 5, pp. 1255-1262, Oct. 2017.

- Euclidean Cluster Extraction from Point Cloud Library

- Y. Li and G. Liu, "Learning a scale-and-rotation correlation filter for robust visual tracking," *2016 IEEE International Conference on Image Processing (ICIP)*, Phoenix, AZ, 2016, pp. 454-458.

- D. S. Bolme, J. R. Beveridge, B. A. Draper and Y. M. Lui, "Visual object tracking using adaptive correlation filters," *2010 IEEE Computer Society Conference on Computer*

Vision and Pattern Recognition, San Francisco, CA, 2010, pp. 2544-2550.

- A. Coates, P. Baumstarck, Q. Le and A. Y. Ng, "Scalable learning for object detection with GPU hardware," *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*, St. Louis, MO, 2009, pp. 4287-4293.

V. Results

3D mapping

- At the current stage, we are able to track the drone's trajectory and build a reconstructed 3D map although it is not as accurate as we would like. There tend to be some overlapped points while creating the point cloud representation of the environment which we believe is due to versions of the operating system and used libraries.

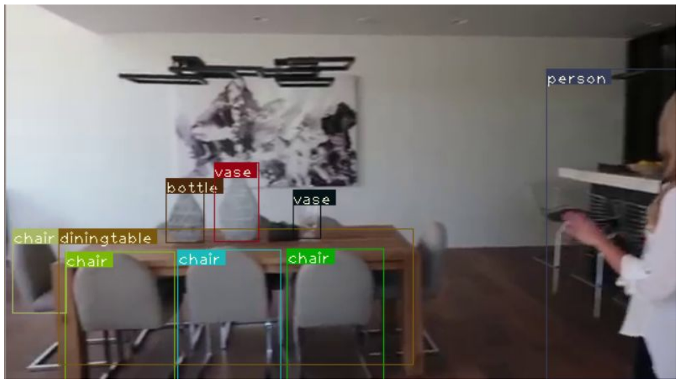
Object Recognition

- Currently, we can run our object detection model for general objects such as chairs and tables. Beyond that, we have already trained our own model that could detect customized objects. Specifically, our model detects boxes in the video took in the lab. We also implemented the object detector with OpenCV tracker to improve the detection consistency and performance of detecting partially occluded objects.
- Also, we are planning to add more classes into our model to detect more objects.

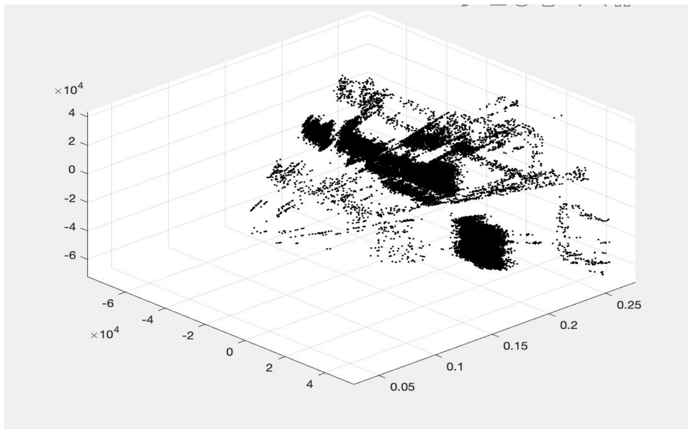
VI. Conclusions

- Real-time object detection using Tensorflow Object Detection API and YOLO

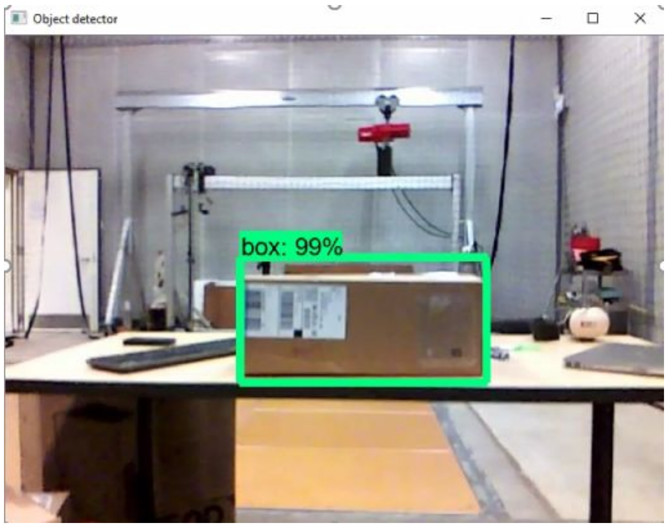
VII. Images

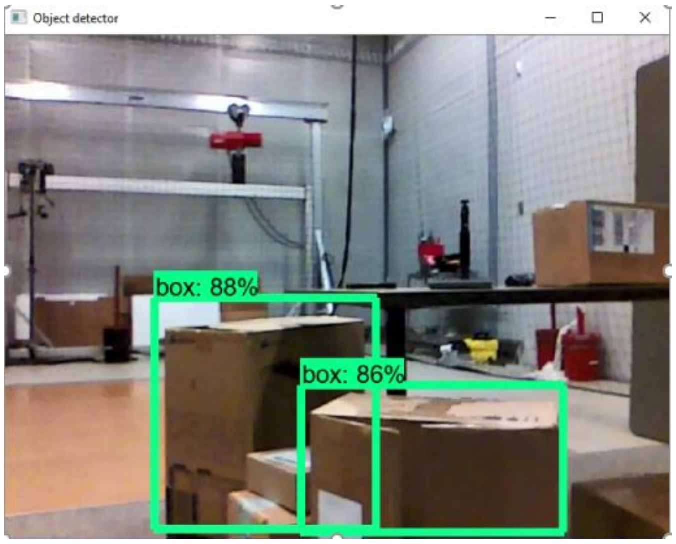


Object Recognition



Extraction of point cloud data and reconstructed to point cloud map





Object Detection for detecting boxes