Individual Lab Report 5

Name: Zhihao Zhu

Team: PhoeniX (H)

Date: 04/11/2019

Team Member: Akshit Gandhi

Shubham Garg

Parv Parkhiya

I. Individual Progress:

My work for the last two weeks are mainly focused on two parts: 1. completing the implementation of the micro-controller, 2. establish the simulation for perception abstract.

1. Micro-controller:
Since the last two weeks I have configured the necessary working environment for the Teensy 3.6 (mainly the ROS package and Serial communication), these two weeks mainly focus on implementing the function that: micro-controller (Teensy) receiving the ROS message sent from the Jetson side, and sent appropriate control signal to the laser.

The control logic is such that thermal camera first capture the thermal images, and we run the image segmentation algorithm (implemented previously) to get the processed image. Once we detect the segmented target object (heat source) and the target object lies close to the center of the image, we will send a "fire" signal (we use a boolean variable in our code, "1" represents for "fire", and "0" represents "not fire") to the micro-controller, and then micro-controller sets one of its digital output port(connected with laser) as HIGH, to activate the laser.

During the process of implementation of above functionality, we met a problem that when we are writing the ROS subscriber's callback function (which is used to decide whether "fire" signal is received or not) on the Teensy side, the message comparison block does not work as expected. To be more specific, we are writing an *if* condition to determine whether the ROS message we received is the fire (1) signal or not, and we use *msg.data == "1"* and int(*msg.data) == "1",* but both of the above don't work. So to check where the problem might be, we have to verify the message content received by Teensy. But soon, we met another problem: the current serial port is used for transmitting data from Jetson to Teensy. So in order to use Teensy's serial monitor to print some contents, we wired another serial port and attach that port on Teensy to Jetson via a Serial-USB transfer cable. Finally, it turned out that when we do *int()* (string_to_int) on the received ROS message (which is std::String type), we are getting a weird number: 5306999, instead of the expected integer 1. Until now, we still couldn't figure out the problem. But we still managed to solve the message comparison problem by directly casting the *std::String* type into C++ string, and compare it to the string "1" and "0". Finally, we are able to "fire" the laser once the thermal camera detects heated source.

2. Simulation of Perception Abstract

In this part, the goal is to establish the simulation environment using Gazebo for our robot's perception system. Currently, two simple goals are: 1. given the camera's pose (including position and orientation) and all the other objects' pose, we want to determine whether a designated target object is visible or not to the camera. 2. the perception system should be able to return the geometry information of two specific target object: window and door (haven't implemented yet).

First, I wrote a ROS node, which can subscribe to the */Gazebo/states/* ROS topic, and get each object's pose information given the object's name. Then, this node publishes the each object's pose information via different topic channel (named according to object's name), so that other ROS node can select to utilize certain objects' information when needed.

Then, after obtaining the pose information of all the objects, we wrote a node to implement the first goal: determine whether a specific object is visible or not by camera. Our approach is sampling-based. Within the sector of camera's perceptive field, we sample "ray" sources with a constant distance of 1 degree, and along that "ray", we sample points with a distance of 0.01m. If there is a sample point within an object, it means that object is visible to the camera. If there are different samples within two or more objects along the same "ray", it means that the farther objects are occluded by the nearer object, and is not visible to the camera at least from this particular ray.

II. Teamwork

Akshit and I worked together on solving the "message comparison" mentioned above. Parv helped me familiarize with the Gazebo environment, and Shubham helped me by explaining the serial communications.

III. Challenges

For the micro-controller part, the main challenge is the ROS message type problem as stated above. For the simulation of perception abstract part, the challenges lie in two perspectives:1. I'm not familiarized with Gazebo previously, so I have to learn this tool, as well as how Gazebo interacts with ROS. 2. To determine whether an object is visible or not to the camera, current sampling-based method is not optimal. So, we have to find an analytic method which is more accurate.

IV. Future Plan

My future plan is to finish the simulation for our robots' perception system in the Gazebo, including the object detection, segmentation and visual servoing of the UAV.