Perception and Decision Making in Intelligent Systems
Homework 3: A Robot Navigation Framework
Announce: 11/5 , Deadline: 11/26 23:55

1. Introduction

In the HW2, we successfully reconstructed a 3D semantic map of **apartment_0**. Our next goal is to have a robot move to a desired location (for example, **navigate the robot to find a specific item**). Therefore, in the third homework, we focus on how to navigate from point A to B on the **first floor of apartment_0** using the RRT algorithm.

There are three main tasks, i.e., **2D semantic map construction, RRT algorithm implementation, robot navigation**

1. 2D semantic map construction:
   In the second homework, we already have a semantic point cloud. We need a 2D semantic map for navigation.

2. RRT algorithm implementation:
   We use the RRT algorithm to find a navagitable path from starting point A to goal point B (a specific item).

3. Robot navigation:
   An agent can navigate automatically by following the path calculated by RRT on the **first floor of apartment_0** to find specific items.

2. Requirements

- OS : ubuntu 18.04
- Python 3.6
- opencv-contrib-python
- Open3d
- matplotlib
- Habitat sim and Habitat lab

3. Tasks

Part 1: 2D semantic map construction

The following describes the steps for generating a 2D semantic map.

1. Remove the ceiling and floor points of the point cloud constructed in HW2
2. Save coordinates and colors of the points (the color should be the same as HW2) color map for 101 categories
3. Use any libraries (such as matplotlib) to plot a scatter graph (x-coordinate, z-coordinate in the point clouds) with with the points you saved
4. Save the map as **"map.png"**

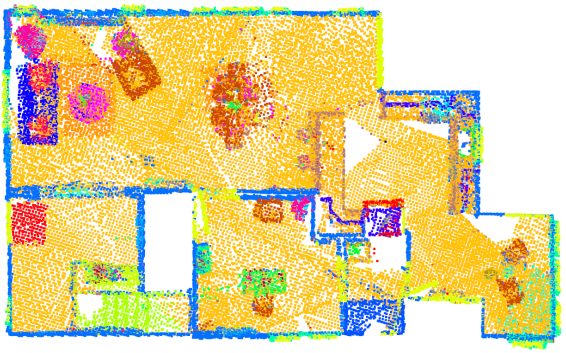
We provide a 2D semantic map of the first **floor of apartment_0 for your reference**. You can download it from this link. If you choose to use the provided 2D semantic map, you can start from **Step 3 (i.e., plot scatter graph)**. Definitely, please free to use your own reconstructed map.

Notes for **Step 3**:

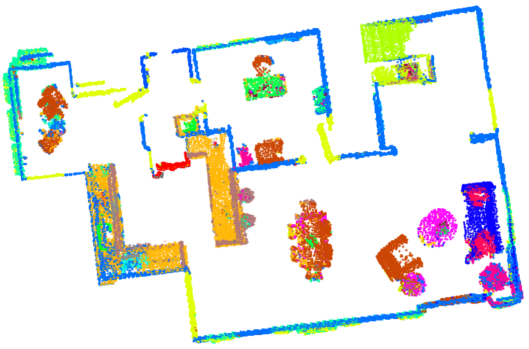
- **Scale relationship** between coordinates in **point.npy** and coordinates in apartment_0 is:
  apartment_0 = points array * 10000. / 255.
- There are two versions of the color array, the value of rgb in **color_01.npy** is in range [0, 1] while **color_0255.npy** is in range [0, 255]. If you have problems using **color_0.npy** to find labels because of floating-point errors, you can use **color_0255.npy.**

Note: You need to find the coordinate relationship between the map (pixel) and the coordinates on the **first floor of apartment_0**. (in Part 3) (**Feel free to add extra points to calculate the scale and create different map may help you do the job**)

Example map:



[After removing ceiling]



[Removing ceiling and floor]

Part 2: RRT Algorithm

You will implement the RRT algorithm, which calculates a navigable path from the **"map.png"**. For more detail of the algorithm, please check details on this link RRT or the slide on the class (**Lecture 3-2 Sample based Path Planning**)

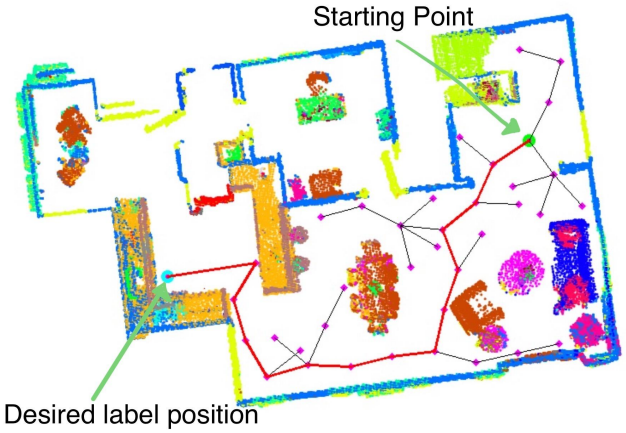
Input:

1. A target category you want to search. Just input a string (our target name ex: **rack**)
    - Check the colors of the corresponding categories from color map for 101 categories
      Our target categories are **refrigerator, rack, cushion, lamp, and cooktop**
    - In this homework, you only need to search for designated categories
2. Choose a **starting point** on the map

Output:

1. A map with a path from a starting point to a target point. Note that the target point is a point in front of the target item. It is up to you to decide the location of the point.
    - Save the map as **“route.png”**
2. Path points on this route
    - The xy-coordinate calculated by RRT is in the pixel coordinate. To navigate in Habitat, you need to convert them into the coordinate **of the first floor of apartment_0**

Example of a RRT output:



Part 3: Robot navigation

- The xy-coordinate calculated by RRT is in the pixel coordinate. We need to convert them from pixel coordinate to xyz-coordinate on the first floor of **apartment_0** in Habitat
- we can define the amount of how much the agent should move and turn for each step (By adding the following lines in function: make_simple_cfg)

```
# Here you can specify the amount of displacement in a forward action and the turn angle
agent_cfg.action_space = {
    "move_forward": habitat_sim.agent.ActionSpec(
        "move_forward", habitat_sim.agent.ActuationSpec(amount = 0.25)
    ),
    "turn_left": habitat_sim.agent.ActionSpec(
        "turn_left", habitat_sim.agent.ActuationSpec(amount = 10.0)
    ),
    "turn_right": habitat_sim.agent.ActionSpec(
        "turn_right", habitat_sim.agent.ActuationSpec(amount = 10.0)
    ),
}
```

- Based on those path points, let the agent move using these three commands : “move_foward”, “turn_left”, “turn_right”

- Use the load.py to navigate (please feel free to make any modifications to the code)
- Collect RGB images of moves, and save them as a video **“Observation.avi”**

Note: In the video, please highlight the target with a **transparent mask** while navigating, so that we can clearly visualize the target category.

For instance, we search for **“refrigerator”**.

4. Examples

Example Demo Video

5. Submission

1. **File Formate:**

```
HW3_StudentID   | --- Readme
                | --- Codes
                        | --- build_2D_map.py ( to create map)
                        | --- navigation.py  --- RRT.py
                        or
                        | --- main.ipynb
                | --- Map
                        | --- map.png
                        | --- route.png
                        | --- other map …(if you need extra map)
                | --- Observation.avi
```

Note : You can use either .py or .ipynb to show your results.
Please choose a target from our list.

- route.png(RRT result)
- Observation.avi(navigation result)

ReadMe:
Please describe how to run your program and the usage of maps if you have extra ones.

Please put all the files into folder **HW3_StudentID** and compress your folder into **a "zip" file**. Please submit the file to New E3 System with the naming format **HW3_StudentID.zip.**

2. The **deadline** for this homework is **11/26 23:55.**
3. Late submission leads to **-20 points per day**.

6. Grading

1. Online Questions (30%)
   - Explain RRT Algorithm
   - Describe the way you calculate the scale
2. Online Demo (70%)
   - We will specify a target label and a starting point, you should show us the corresponding navigation result on the first floor of aprartment_0.