Clutterbot

SOFTWARE GET - ASSESMENT

Simulation setup (ROS2)

1. You can take up any open-source robot model (turtlebot, husky, etc.) and/or an entire gazebo simulation setup and get it up and running on your laptop.
2. This setup should include the following:
    a. A non-holonomic robot capable of publishing
        i. Wheel Encoder odometry data
        ii. IMU data
        iii. 2D Lidar data (range limited at 5m and have 2 configuration files, one with angular resolution of 0.9 degree and other with 2.5 degree)
    b. The environment setup should be about 10.0 x 5.0 m^2 with three or more rooms and ~30% obstacle coverage, with objects like, cupboards, tables, chairs or just simple 3D shapes with dimensions similar to the above mentioned objects. Eg. A cupboard could just be a plain cuboid of size 1.5 x 1.0 x 0.4 m^3.
    c. All obstacles in the environment should be visible at lidar height.
    d. You should be able to run some open-source 2D-SLAM packages using data from the above.
3. **Bonus:** You can also set up the ability to perform autonomous navigation tasks, using an open-source navigation pipeline in tandem with above achieved tasks.
    a. This could include manually giving tasks through a ros2 node or rviz2.

Compare any two 2D-SLAM approaches

1. With the above simulation set up, you have to choose any 2 open-source 2D-SLAM packages (eg. gmapping, etc.) and run them.
2. Comparative observations need to be logged for the same. These observations and their explanations can be along the lines, but **not** limited to:
    a. % Compute usage (here the variation in the trends also need to be explained with context from the SLAM approach being used)
    b. Variations in the mapping output when compared to varying input velocity
3. Run the packages on both lidar configurations and comment on the differences observed and also your thoughts on what other issues one may face due to this change.
4. Comment on whether you are able to observe a loop closure/ global correction step during your tests. Elaborate on why you did or didn't observe the same and what changes you could make to the simulation to make this step visible.
5. Suggest one package from the two, that you think is better with a brief reasoning.

Introduce noise to the system

1. Write a C++ ROS2 node (in a package) to subscribe to the odometry topic and introduce some noise (random) to it of reasonable amplitude as one may expect in a home environment.
2. Broadcast this new odometry data as a transform and a topic, while ensuring no other sources of odometry transform are active.
3. Run the ‘better’ SLAM package from above again with this new source of odometry and compare the difference in observations.
4. Explore through the parameters of the SLAM package of how the newly observed issues could be fixed.

Documentation and Submission

1. A doc file summarizing the comments and explanations for tasks 2 and 3 from above. Feel free to add any plots that seem relevant for explanation.
2. A github link or .zip file with all the code and launch files etc. and a supplementary readme for any installation requirements and instructions to run the setup.
3. A screen recording video for task 2, for at least one SLAM method.

Evaluation Criteria

You will be evaluated based on the following:

1. Completion of simulation setup
2. Thoroughness and coverage of observation scenarios for the SLAM package comparison.
3. Structure of your code and package directory