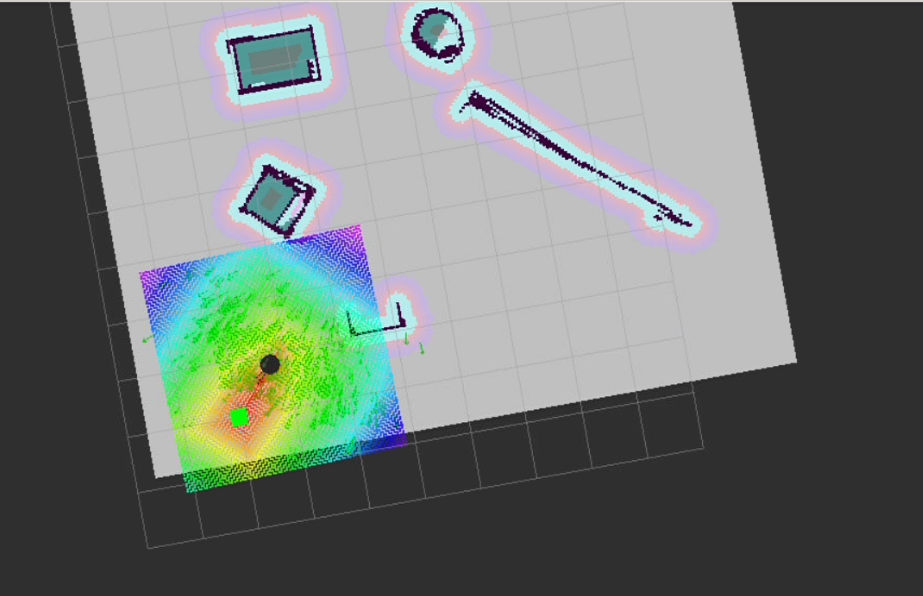
**Udacity Robotics Software Engineer Nanodegree Program Project 5 - Home Service Robot (Writeup)**

**Intro**

The task of this project was to use create a robot in a virtual environment along with a virtual object. The task of the robot was to navigate to the initial goal position, pick up the object and drop it off at the next goal position. It had to do that while navigating the environment on it own. The robot is able to perform Simultaneous Localisation and Mapping or SLAM. [](https://github.com/murthybalaji0725/udap5/blob/master/Extra%20file/path%20planning%201.png)

**The robot performed :**

1. Localisation using AMCL
2. SLAM using Gmapping package with uses a Grid Based FastSLAM
3. Navigation using Dijkstra's algorithm

**AMCL**

The Adaptive Monte Carlo Localization algorithm uses a particle filter to track the pose of the robot against a known map. The particles are initially distributed randomly in the environment with some weights. As the robot localises the weights are updated and the particles that predict the pose of the robot more accurately survive.

Other localisation algorithms are

1. Extended Kalman Filter
2. Markov Localisation
3. Grid Localisation
4. Monte carlo Localisation

**Dijkstra's algorithm**

It is another type of Uniform Cost Search algorithm. This algorithm is used by the ROS Navigation Stack. Dijkstra's algorithm (or Dijkstra's Shortest Path First algorithm, SPF algorithm) is an algorithm for finding the shortest paths between nodes in a graph. Refer this [link](https://www.geeksforgeeks.org/dijkstras-shortest-path-algorithm-greedy-algo-7/) to better understand the working of the algorithm and test its implementation in C++ and python.

**ROS Packages used**

**GMapping**

[Documentation available here](http://wiki.ros.org/gmapping) The package provides laser-based SLAM. Using slam\_gmapping, you can create a 2-D occupancy grid map. Using this we can create a floor map for the environment.

**turtlebot\_teleop**

[Documentation available here](http://wiki.ros.org/turtlebot_teleop) Allows user to control the robot using commands from joystick or keyboard.

**turtlebot\_rviz\_launchers**

[Documentation available here](http://wiki.ros.org/turtlebot_rviz_launchers) It contains the view\_navigation.launch file using which can we load a preconfigured RViz workspace. It automatically loads the RobotModel, Trajectory, Maps and other relevant things in thr RViz workspace and saves lots of time

**turtlebot\_gazebo**

[Documentation available here](http://wiki.ros.org/turtlebot_rviz_launchers) One of the most important packages. We can deploy the turtlebot in gazebo with our own custom world file.