3D Motion Planning

Explaining Starter Code:

The starter code given consists of two basic files one is motion\_planning.py and other is Planning\_utils.py. The planning\_utils.py contains several methods such create\_grid that help us discretize the world into grid format, then a function valid\_action that specifies what valid actions can be taken by the drone. Finally, it contains A\* algorithm which helps drone is finding a path.

The motion\_planning.py file contains a drone class which has several functions defined in it. At first it is initialized by the function \_\_init\_\_. Then a few callback functions are defined which based on the conditions help drone transition from one state to another. Later, all the transitions functions are defined, and which dictate how the transition happens. Then the plan\_path function which takes in the information from planning\_utils.py file and helps in reading the data, set start and goal states implement A\*, find the waypoints and send them to drone. Finally, the main function helps building the connection with the simulator provided.

Comparing Motion\_Planning.py and Backyard\_Flyer\_Solution.py:

The code in motion\_planning.py is more sophisticated the backyard\_flyer. The most important reason for this the A\* star algorithm which helps is finding the optimal path between any start and goal position on the grid map if available. On contrary in backyard\_flyer the waypoints are just 4 fixed points that are calculated in calculate\_box\_function. Other than this, the callbacks are called in a similar manner and all other functions are implemented in the same way before.

Implementation of Path Planning:

* Read the first line of the csv file. Then iterated over the data and added global home position dictionary. Then using lat0 and lon0 floating point values global home position was updated when used with set\_home\_position function.

Text

Description automatically generated

* Global\_to\_local function was used to convert the global coordinated to local coordinates that were read from the colliders.csv file.

Text

Description automatically generated

* To add the flexibility to start and goal location so that the vehicle can start from its current location and go to any goal that if feasible on the map we set the grid\_start equal the current local position minus the offset. Then convert the existing coordinates to global coordinates before adding desired (lat, lon) values. Finally, convert them to local coordinates back to use them in next steps.

Text

Description automatically generated

* Then to include diagonal motions modified the A\* search algorithm. Included 4 diagonal motions such NORTH-WEST, SOUTH\_WEST, NORTH\_EAST and SOUTH\_EAST. Validation checks were added to keep a check if vehicle was near the boundary of map or near an obstacle and remove the respective diagonal motion.

A picture containing diagram

Description automatically generated

* Using the collinearity test to check the collinearity in the waypoints and removal of those in between waypoints was achieved so that the drone’s motion is smooth through the path. Then the path with was pruned.

Text

Description automatically generated