UDACITY

PROJECT SPECIFICATION

3D Motion Planning

Writeup

CRITERIA	MEETS SPECIFICATIONS
Provide a Writeup / README that includes all the rubric points and how you addressed each one. You can submit your writeup as markdown or pdf.	The writeup / README should include a statement and supporting figures / images that explain how each rubric item was addressed, and specifically where in the code each step was handled.

Explain the Starter Code

CRITERIA	MEETS SPECIFICATIONS

CRITERIA	MEETS SPECIFICATIONS
Test that `motion_planning.py` is a modified version of `backyard_flyer_solution.py` for simple path planning. Verify that both scripts work. Then, compare them side by side and describe in words how each of the modifications implemented in `motion_planning.py` is functioning.	The goal here is to understand the starter code. We've provided you with a functional yet super basic path planning implementation and in this step, your task is to explain how it works! Have a look at the code, particularly in the `plan_path()` method and functions provided in `planning_utils.py` and describe what's going on there. This need not be a lengthy essay, just a concise description of the functionality of the starter code.

Implementing Your Path Planning Algorithm

CRITERIA	MEETS SPECIFICATIONS
In the starter code, we assume that the home position is where the drone first initializes, but in reality you need to be able to start planning from anywhere. Modify your code to read the global home location from the first line of the `colliders.csv` file and set that position as global home (`self.set_home_position()`)	Here you should read the first line of the csv file, extract lat0 and lon0 as floating point values and use the `self.set_home_position()` method to set global home.

CRITERIA	MEETS SPECIFICATIONS
In the starter code, we assume the drone takes off from map center, but you'll need to be able to takeoff from anywhere. Retrieve your current position in geodetic coordinates from `self._latitude`, `self._longitude` and `self._altitude`. Then use the utility function `global_to_local()` to convert to local position (using `self.global_home` as well, which you just set)	Here as long as you successfully determine your local position relative to global home you'll be all set.
In the starter code, the `start` point for planning is hardcoded as map center. Change this to be your current local position.	This is another step in adding flexibility to the start location. As long as it works you're good to go!
In the starter code, the goal position is hardcoded as some location 10 m north and 10 m east of map center. Modify this to be set as some arbitrary position on the grid given any geodetic coordinates (latitude, longitude)	This step is to add flexibility to the desired goal location. Should be able to choose any (lat, lon) within the map and have it rendered to a goal location on the grid.

CRITERIA	MEETS SPECIFICATIONS
Write your search algorithm. Minimum requirement here is to add diagonal motions to the A* implementation provided, and assign them a cost of sqrt(2). However, you're encouraged to get creative and try other methods from the lessons and beyond!	Minimal requirement here is to modify the code in `planning_utils()` to update the A* implementation to include diagonal motions on the grid that have a cost of sqrt(2), but more creative solutions are welcome. In your writeup, explain the code you used to accomplish this step.
Cull waypoints from the path you determine using search.	For this step you can use a collinearity test or ray tracing method like Bresenham. The idea is simply to prune your path of unnecessary waypoints. In your writeup, explain the code you used to accomplish this step.

Executing the flight

CRITERIA	MEETS SPECIFICATIONS

CRITERIA	MEETS SPECIFICATIONS
This is simply a check on whether it all worked. Send the waypoints and the autopilot should fly you from start to goal!	At the moment there is some mismatch between the colliders map and actual buildings in the scene. To ensure success build in a 5+ m safety margin around obstacles. Try some different goal locations. Also try starting from a different point in the city. Your reviewer will also try some random locations so be sure to test your solution! There is no firm constraint or requirement on how accurately you land exactly on the goal location. Just so long as your planner functions as expected.

Suggestions to Make Your Project Stand Out!

For a standout submission, consider using some of the more advanced techniques presented in the lessons, like the probabilistic roadmap, receding horizon planning and automatic replanning. Play around with a dynamical model for the vehicle, deadzones to allow smooth transitions through waypoints and even a potential field modification to your planner!