Lab 5

1. What are the names of everyone in your lab group?
    - Wei Jiang
    - Nikko Gajowniczek
    - Patrick Nguyen
2. Why is sensor noise problematic when mapping?
    - It tricks the robot/our algorithm to think that there is an object where there isn't.
3. How did you choose the value with which you increment the map entry? What happens if the value is too small, what happens if it is too large?
    - We incremented the probability that each pixel contains an obstacle by 0.005 every time the robot detects there's an obstacle there.
    - If this value were too small we would not be able to identify obstacles until the robot has consistently seen it which can reduce the false positive but also increase the false negative rate.
    - If the value was too large, the robot would likely be overly sensitive to the lidar readings making it think there are random obstacles everywhere. Too large of a value means that the robot will mark the pixel as an obstacle even though it only saw it a small number of times.
4. How did you choose the value to threshold your map? What happens if the value is too small, what happens if it is too large?
    - To definitively say there is an obstacle, we chose to use probability > 0.8
    - If the threshold is too small (e.g., 0.3):
        - More sensitive to detecting obstacles
        - Requires fewer observations to mark obstacles
        - Many false positives
        - Noise and temporary objects might be marked as obstacles

    - Path planning becomes difficult due to excessive "obstacles"
- If the threshold is too large (e.g., 0.95):
    - Very high confidence in obstacle detection
    - Minimal false positives
    - Might miss real obstacles that weren't observed enough times
    - Requires many more observations to confirm obstacles

5. As Tiago is traveling along the path that the planner provided, suppose it detects an

object in its way. How could you modify your solution to plan around/gracefully handle this unforeseen object in the robot's path?

- Instead of using a pre saved map from manually moving the robot and mapping the space, we can use its real time feed + the pre saved map.
- Perhaps when using the real time feed we can allow the robot to be more sensitive to the objects to reduce how many observations it requires to consider something an object.
    - This should allow it to quickly identify obstacles in a scenario where it might not be able to make constant observations of a certain location
    - When new obstacles are updated to the map, quickly re simulate the path to check for collisions and adjust the path if needed
    - Can use RRT for scenarios like these

6. Could we use an algorithm like RRT to generate a viable path instead of Dijkstra's algorithm/A*? If yes, how would the path look different? If no, why not?
    - Yes, RRT would still find a path, however, it will not be the most optimal straight line path. The path would look more organic rather than straight line as RRT would generate more random looking paths.
    - RRT paths would also be non deterministic meaning the same location would not have the same path every time when using RRT.

7) Roughly how much time did you spend programming this lab?

- 3 hours: mapping, creating map etc.
- 10 hours:
    - finding out that the robot coordinates are just plain wrong and the orientation was not bounded within [-2pi, 2pi] instead it was offset by like pi/2 or something.
    - The freaking caster wheels didn't let me move the robot straight after turning which means i have to somehow smoothen out its turns or just be patient until the wheel slowly moves to the right postitions
- 10+ hours:
    - Aligning the camera frame to world frame (we never got it to work)
    - Getting the arm to move to where the object is using IK and the camera frame (didn't work because idk where the arm is going like its highkey kinda random)
    - Trying to understand how to recognise the orange because the duck could be recognised but the orange couldn't (why?)

By the way, in our submissions we also added a lab5_controller_old.py because this one has all the other functions like the planner mode, etc.