Andrew Wood
aew61

PS1

My strategy for maneuvering in the simulator was to incrementally run my ROS node and make path adjustments each time until I made it from my initial position to my final position. To ease this process, I developed some utility functions that provided basic rotation and translation services. Therefore, implementing additional movement instructions meant adding additional function calls in my main function.

I found that, while maneuvering, the rotation process was the most fragile. Despite using 36 significant digits of Pi and turning at an extremely slow rate (turning Pi/2 radians in 3 seconds updating every 100Hz per turn), that sometimes my turns are imprecise enough to cause my hard-coded trajectory to fail in moving the robot to the goal position. This scenario is rare enough that I am not worried of failure when my code is reproduced, but often enough that it is frustrating. I could theoretically solve this by increasing the frequency of message updates, or by slowing my turning rate (yaw rate) even more.

Finally, my main OS is Ubuntu 16, so I am running ROS kinetic locally where developed this solution. Therefore, I was worried that my solution would not build or run on ROS indigo depending on whether <ros/ros.h> or the stdr_simulator has changed between versions (or the environment in which they are built has changed significantly enough). To solve this, I created a 14.04 virtual machine and installed indigo on it, and I confirmed that my solution builds and runs as-is.