Zach Janice, znj
PS4

The implementation for assignment 4 is set up similarly to that of assignment 3; the client constructs a path of poses for the server to execute from start to finish. With modifications made in this assignment, an action server is now in use instead of a service, and interruptible helper methods for twist commands allow for halting of the current goal at any time.

The mentioned helper methods, along with other associated methods that pertain to the movement and management of the robot in two-dimensional space, are contained in an included resource named "stdr_twist" (see the /include directory of the project files). The goal of this resource is to streamline twist commands given to a robot. While the root versions of the methods are still being worked on, as they cause segmentation faults for currently unknown reasons, the partial ("interruptible") "...Iter" versions of the methods work as intended. As these methods are the ones needed for this assignment, the resource is utilized.

The differences between the previous assignment and this assignment - in detail - are as follows: The movement commands in the server (beyond being moved to the resource) allow for a boolean check to occur after each iteration. This check, allowing for an abort flag from the client to be raised, will stop the robot upon returning true. The remaining poses of the path are then given back in the result of the goal, along with the (estimated) current pose of the robot (these estimates come from refactoring the movement methods to return floats of what portion of twisting is left to be done after execution).

In receiving this result back in the failed case, the client takes the (estimated) current pose and has the robot revert the path backwards a small amount. After reaching this reversion pose, the original path is tacked on to be attempted again.

In execution of these nodes, the robot attempts to make it to the upper corner of the maze, by the same path as in the previous assignment. In observing a possible obstacle (including squeezing through a narrow pathway), the goal is halted by the client and sent again with the reversion. This translates to the robot turning around and moving back a small amount, then turning around again to see if a second attempt at following the path will work. By the current design of the nodes, this process will repeat infinitely until enough error in twists would cause the robot to find a successful (and wrong) path. The process of aborting the current goal and starting a new one, however, is clearly demonstrated through the action server.