Robotics: HW1

R11922178 曾青山

Demo video link:
https://drive.google.com/drive/folders/1kcgpTVXN1n-Ug3QVz0elWHAxDeHmS3KI?usp=sharing

Part C: Keyboard Control

Use WASD for the robot's movement and backquote (`) to stop all movements. Key presses are attached to ArKeyHandler, and each press has its own functions. For forward movement (W), pressing W causes current velocity to be added by an acceleration value of 50 m/s^2 , but the top velocity is capped at 300 m/s. Backward movement (S) is the same as forward move, but instead of addition, velocity is subtracted from acceleration value. Left (A) and right (D) movements also have the same idea as the former movements, however instead of *setVel*, we use the function *setRotVel* to set rotation velocity. Rotation velocity is capped at 50 degree/s. When backquote (`) is pressed, it will set both *setVel* and *setRotVel* to 0, stopping all movements.

Part D: Collison Avoidance using Sonar Sensor

Collision avoidance feature is implemented in the *CollisionAvoidance* function. To read from the sensor, we use the ArSensorReading class to get sonar reading. After it is read, we need to find the range of the environment from the sonar reading. Next, we need to check if any of the range is less than the predetermined range. I set the range as 500, so if the range is less than 500, it will stop all movements.

Part E: Robot Pose and Odometry

1. Odometric pose is different from the true pose. Odometry pose is relative to the position of the robot, so when the map is loaded for the first time, the odometry pose of the robot is 0. True pose, however, is relative to the position of the world (map).

2. According to the question, we need to set the odometry pose of the robot to a certain point. We can use ArPose to do this. Then, we ask for user input of the target position. We calculate the target position then we use ArPose to set the target location. We calculate the angle and distance from the current position to the target position. First, we face toward the target angle and wait until it completes its move. Next, we move the robot to the calculated distance and wait until it reaches the destination. Lastly, make the robot face Th.