Obstacle_Avoidance

September 5, 2021

1 Obstacle Avoidance in ROS

In this tutorial, I'll show you how to make an obstacle avoidance robot in ROS. First of all, you have to install some kind of mobile robot simulator. I am using turtlebot3 in gazebo simulation env., running on Ubunutu 18.04 and ROS melodic. Obstacle avoisdance alorithm in this simulation does not use map, and robot goes only forward unless it encounters any obstacle in front, ledft-side or right-sides. when the obstacle is perfectly in front, it simply stops untill the obstacle is gone/removed. In case of left-side or right-side ostacles, robot takes CW and CCW turnes respectively, untill obstacle is beyond given distance-threshold (in our case Threshold=1m). Let's see the code and logic below:

```
In [ ]: #!/usr/bin/env python
        # Dependecies
        import rospy
        from geometry_msgs.msg import Twist
        from sensor_msgs.msg import LaserScan


        # call back function
        def callback(msg):
            if (msg.ranges[20]<=1):     # Turn CW.
        # Lidar has 0-360 degree range, with 1-deg increments.
        # Hence, ranges[20] : is 20th lidar ray at 20deg from x-axis (front).
        # Similarly, ranges[340]: is at -20deg from x-axis,
        # and ranges[0]: is front ray (along x-axis).
                move.linear.x=0.0
                move.angular.z=-0.3
                print("Dist_Left",msg.ranges[20])

            elif (msg.ranges[340]<=1): # Turn CCW
                move.linear.x=0.0
                move.angular.z=0.3
                print("Dist_Right",msg.ranges[340])

            elif (msg.ranges[0]<=1):     # STOP
                move.linear.x=0.0
                move.angular.z=0.0
        print("Dist_Front",msg.ranges[0])

    else:
        move.linear.x=0.3    # Go Forawad
        move.angular.z=0.0
        print("Go--Go--Go!")



rospy.init_node('obstacle_node') # Initialize node as ('obstacle_node')
pub = rospy.Publisher('/cmd_vel', Twist, queue_size=1) # make Publisher which will
# be used to send cmd_vel comonds to robot base.
sub=rospy.Subscriber("/scan", LaserScan, callback) # make Subscriber which subscribes
# to lidar data ( topic name;  /scan).


rate = rospy.Rate(2)
move = Twist() # defining the way we can allocate the values


while not rospy.is_shutdown():
  pub.publish(move)
  rate.sleep()
```