|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Task. No.: | 5 | Points: | 5 | TurtleBot3 movement |

|  |
| --- |
| Objectives:  1. Installing Turtlebot. 2. Using Gazebo simulation. 3. Using launch files. |

|  |
| --- |
| Description: The goal of this exercise is to install the TurtleBot packages and launch the Gazebo simulation and move the robot to a fixed distance. The distance is global distance and the initial origin of the robot is 0,0,0. |

|  |  |
| --- | --- |
| Step | Action |
| 1 | Update the list of packages.  ~$ sudo apt-get update ~$ sudo apt-get upgrade |
| 2 | Install turtlebot simulation packages.  ~/catkin_ws/src$ git clone https://github.com/ROBOTIS-GIT/turtlebot3_msgs.git -b noetic-devel ~/catkin_ws/src$ git clone https://github.com/ROBOTIS-GIT/turtlebot3.git -b noetic-devel ~/catkin_ws/src$ catkin_make |
| 3 | Install simulation environments.  ~/catkin_ws/src$ git clone https://github.com/ROBOTIS-GIT/turtlebot3_simulations.git ~/catkin_ws/src$ catkin_make |
| 4 | Change the *bashrc* file.  ~/catkin_ws$ cd .. ~$gedit .bashrc  source /opt/ros/noetic/setup.bash source /home/smoz/catkin_ws/devel/setup.bash export TURTLEBOT3_MODEL=waffle export SVGA_VGPU10=0 |
| 5 | Load world environment.  ~$ roslaunch turtlebot3_gazebo turtlebot3_world.launch |
| 6 | Move the robot around the environment. Open a new terminal and launch the following command.  $ roslaunch turtlebot3_teleop turtlebot3_teleop_key.launch |
| 7 | The velocity topic is */cmd\_vel\_mux/input/teleop*. The Message type is *geometry\_msgs/Twist.*  Robot position feedback topic is */odom* and its message type is *nav\_msgs/Odometry.*  See these messages with *rosmsg show* command. |
| 6 | Write the following code in *move\_turtlebot.py* file.  #!/usr/bin/env python import rospy from geometry_msgs.msg import Twist from nav_msgs.msg import Odometry import sys robot_x = 0 def pose_callback(msg): global robot_x #Reading x position from the Odometry message robot_x = msg.pose.pose.position.x  rospy.loginfo("Robot X = %f\n",robot_x) def move_turtle(lin_vel,ang_vel,distance): global robot_x rospy.init_node('move_turtlebot', anonymous=False) #The Twist topic is /cmd_vel_muc/input/teleop pub = rospy.Publisher('/cmd_vel_mux/input/teleop', Twist, queue_size=20) #Position topic is /odom rospy.Subscribed'/odom',0dometry, pose_callback) rate = rospy.Rate(10) # 10hz vel = Twist() while not rospy.is_shutdown():  vel.linear.x = lin_vel vel.linear.y = 0 vel.linear.z = 0 vel.angular.x = 0 vel.angular.y = 0 vel.angular.z = ang_vel #rospy.loginfo("Linear Vel = %f: Angular Vel = %f",lin vel,ang_vel) if(robot_x >= distance): rospy.loginfo("Robot Reached destination") rospy.logwarn("Stopping robot") break pub.publish(vel)  rate.sleep() if_name_++ "_main_': try: move_turtle(float(syst.argv[1]),float(sys.argv[2]),float(sys.argv[3])) except rospy.ROSInterruptException: pass |
| 7 | We can run this code by using the following command.  $rosrun hello_world move_turtlebot.py 0.2 0 3 |

 This is shared under a Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International Public License