**Exercise -7**

**Name – Akshit Jain**

**Reg. no. – 21BRS1088**

**Question**

Open a gazebo world with burger turtlebot3.  
write a publisher code to move the bot in the following conditions  
  
1. Forward movement  
2. Forward movement with stop for 3 second and move again in the forward direction and repeat  
3. Forward movement with stop for 3 second and move again in the forward direction for 30 seconds and then move in the bot reverse direction with stop for 3 second and move again in the reverse direction for 30 seconds. Again, change the direction of movement in forward direction and continues

**Steps**

1. **Install gazebo first.**
2. **Install packages using - sudo apt install ros-noetic-gazebo-ros-pkgs**
3. **Install turtlebot**

**mkdir -p ~/turtlebot\_ws/src**

**cd ~/turtlebot\_ws/src**

**git clone https://github.com/ROBOTIS-GIT/turtlebot3.git**

**git clone https://github.com/ROBOTIS-GIT/turtlebot3\_msgs.git**

**git clone https://github.com/ROBOTIS-GIT/turtlebot3\_simulations.git**

**cd ~/turtlebot\_ws**

**catkin\_make**

1. **Let’s select the robot by typing in the terminal**

**export TURTLEBOT3\_MODEL=burger**

1. **Let’s bring the gazebo simulation environment**

**roslaunch turtlebot3\_gazebo turtlebot3\_empty\_world.launch**

1. **Open an another terminal**

**export TURTLEBOT3\_MODEL=burger**

**roslaunch turtlebot3\_teleop turtlebot3\_teleop\_key.launch**

1. **Now go to scripts folder and create files for different conditions**
2. **Forward Movement**

**#!/usr/bin/env python**

**import rospy**

**from geometry\_msgs.msg import Twist**

**def move\_forward():**

**rospy.init\_node('turtlebot\_controller', anonymous=True)**

**pub = rospy.Publisher('/cmd\_vel', Twist, queue\_size=10)**

**rate = rospy.Rate(10) # 10 Hz**

**twist = Twist()**

**twist.linear.x = 0.2 # Forward linear velocity**

**while not rospy.is\_shutdown():**

**pub.publish(twist)**

**rate.sleep()**

**if \_\_name\_\_ == '\_\_main\_\_':**

**try:**

**move\_forward()**

**except rospy.ROSInterruptException:**

**pass**

**A screenshot of a computer

Description automatically generated**

1. Forward movement with stop

**#!/usr/bin/env python**

**import rospy**

**from geometry\_msgs.msg import Twist**

**import time**

**def move\_forward\_with\_stop():**

**rospy.init\_node('turtlebot\_controller', anonymous=True)**

**pub = rospy.Publisher('/cmd\_vel', Twist, queue\_size=10)**

**rate = rospy.Rate(10) # 10 Hz**

**twist = Twist()**

**twist.linear.x = 0.2 # Forward linear velocity**

**while not rospy.is\_shutdown():**

**pub.publish(twist)**

**time.sleep(3) # Stop for 3 seconds**

**twist.linear.x = 0.0 # Stop**

**pub.publish(twist)**

**time.sleep(3) # Wait for 3 seconds before moving again**

**if \_\_name\_\_ == '\_\_main\_\_':**

**try:**

**move\_forward\_with\_stop()**

**except rospy.ROSInterruptException:**

**pass**

A screenshot of a computer

Description automatically generated

1. Forward movement with stop and reverse

**#!/usr/bin/env python**

**import rospy**

**from geometry\_msgs.msg import Twist**

**import time**

**def move\_forward\_and\_reverse():**

**rospy.init\_node('turtlebot\_controller', anonymous=True)**

**pub = rospy.Publisher('/cmd\_vel', Twist, queue\_size=10)**

**rate = rospy.Rate(10) # 10 Hz**

**twist\_forward = Twist()**

**twist\_forward.linear.x = 0.2 # Forward linear velocity**

**twist\_reverse = Twist()**

**twist\_reverse.linear.x = -0.2 # Reverse linear velocity**

**while not rospy.is\_shutdown():**

**for \_ in range(3): # Repeat the sequence 3 times**

**for \_ in range(30): # Move forward for 30 seconds**

**pub.publish(twist\_forward)**

**rate.sleep()**

**twist\_forward.linear.x = 0.0 # Stop**

**pub.publish(twist\_forward)**

**time.sleep(3) # Stop for 3 seconds**

**for \_ in range(30): # Move reverse for 30 seconds**

**pub.publish(twist\_reverse)**

**rate.sleep()**

**twist\_reverse.linear.x = 0.0 # Stop**

**pub.publish(twist\_reverse)**

**time.sleep(3) # Stop for 3 seconds**

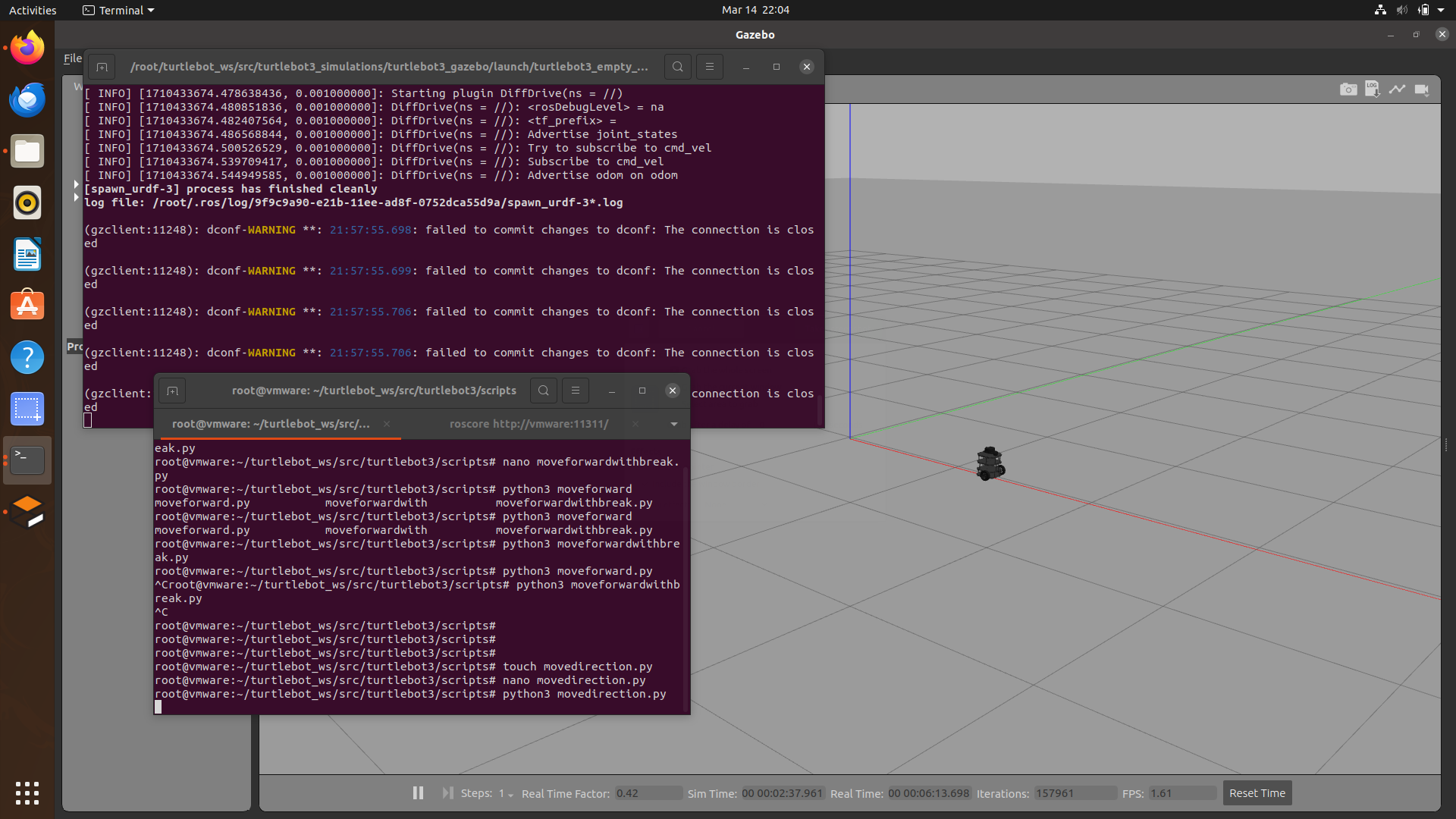
**if \_\_name\_\_ == '\_\_main\_\_':**

**try:**

**move\_forward\_and\_reverse()**

**except rospy.ROSInterruptException:**

**pass**

****