**Lab9: ROS2 control for Turtlebot 2**

sudo apt-get install ros-foxy-teleop-twist-keyboard

sudo apt-get install ros-foxy-joint-state-publisher

sudo apt-get install ros-foxy-xacro

sudo apt-get install ros-foxy-kobuki-\*

ros2 launch turtlebot\_interface interface.launch.py

ros2 run teleop\_twist\_keyboard teleop\_twist\_keyboard

sudo apt-get update && sudo apt-get upgrade -y && sudo apt-get dist-upgrade -y

Open Terminal

hostname -I

or ifconfig

note down the IP address

From a remote computer you can connect to turtlebot Laptop

Using PuTTY (in windows)

Reminna (in Ubuntu)

ssh [username@ipaddress](mailto:ubuntu@192.168.245.41) (in terminal)

Connect camera USB and Kubuki USB and switch on Kubuki.

Switch on the netbook.

Use Remmina to connect to turtlebot from Ubuntu/mtputty from windows

See that the turtlebot laptop and remote laptop are connected to same WiFi

Find the address of netbook placed on turtlebot

Open terminal and type

$ifconfig

Copy the address inet addr: ----(For example 172.16.65.109)

Open Remmina

Establish SSH connection

Type Name (for example turtlebot)

Server Name: 172.16.65.109

User name: mahe or robolab

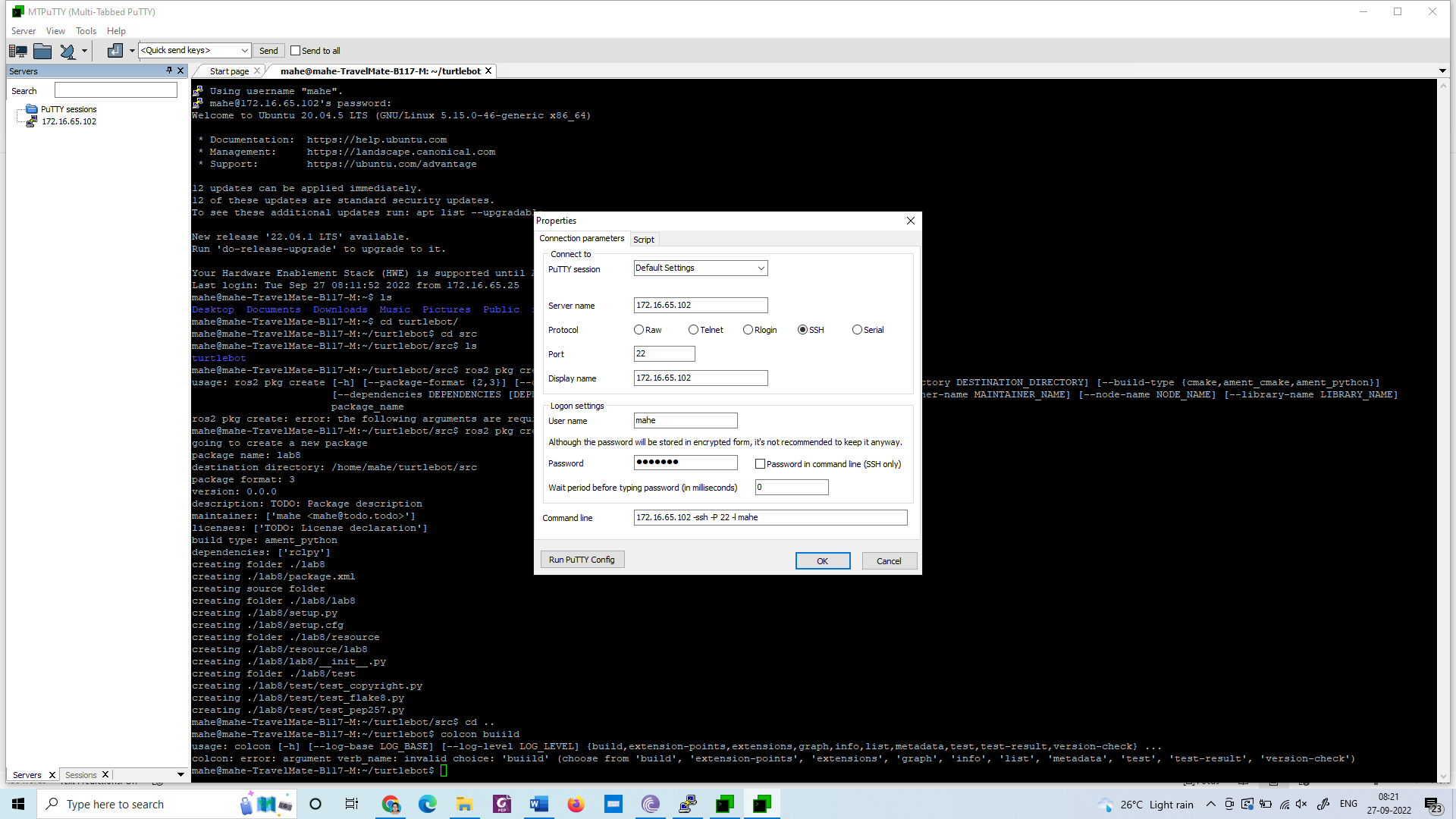
Password:robolab

Open the terminal and type

$ ros2 launch turtlebot\_interface interface.launch.py

Open another terminal and type

$ ros2 run teleop\_twist\_keyboard teleop\_twist\_keyboard



cd turtlebot

cd src

ros2 pkg create lab8 –build-type ament\_python –dependencies rclpy

cd ..

colcon build

Open lab8 using visual studio

Create a python file move\_robot.py inside lab8 folder

#!/usr/bin/env python

import rclpy

from geometry\_msgs.msg import Twist

from nav\_msgs.msg import Odometry

from rclpy.node import Node

import sys

class MoveRobot(Node):

def \_\_init\_\_(self):

super().\_\_init\_\_("move\_robot")

self.lin\_vel = 0.1

self.ang\_vel = 0.0

self.distance = 1.0

self.publisher = self.create\_publisher(Twist, "/cmd\_vel", 10)

self.subscriber = self.create\_subscription(Odometry, "odom", self.control\_loop, 10)

def control\_loop(self, msg):

X=msg.pose.pose.position.x

print("position", X)

vel = Twist()

if abs(X) < self.distance:

vel.linear.x = self.lin\_vel

vel.angular.z = 0.0

else:

vel.linear.x = 0.0

vel.angular.z = 0.0

print('speed : {}'.format(vel))

self.publisher.publish(vel)

def main(args=None):

rclpy.init(args=args)

node = MoveRobot()

rclpy.spin(node)

rclpy.shutdown()

if \_\_name\_\_ == "\_\_main\_\_":

main()

Edit setup.py as follows

from setuptools import setup

package\_name = 'gotogoal'

setup(

name=package\_name,

version='0.0.0',

packages=[package\_name],

data\_files=[

('share/ament\_index/resource\_index/packages',

['resource/' + package\_name]),

('share/' + package\_name, ['package.xml']),

],

install\_requires=['setuptools'],

zip\_safe=True,

maintainer='mahe',

maintainer\_email='mahe@todo.todo',

description='TODO: Package description',

license='TODO: License declaration',

tests\_require=['pytest'],

entry\_points={

'console\_scripts': [

'move = lab8.move\_robot:main'

],

},

)

cd ~/turtlebot/

colcon build

Terminal 1:

ros2 launch turtlebot\_interface interface.launch.py

Terminal 2:

ros2 topic list

Terminal 2:

ros2 run lab8 move

Example 2:

#!/usr/bin/env python

import rclpy

from geometry\_msgs.msg import Twist

from nav\_msgs.msg import Odometry

from rclpy.node import Node

import math

import time

from std\_srvs.srv import Empty

import sys

class MoveRobot(Node):

def \_\_init\_\_(self):

super().\_\_init\_\_("move\_robot")

self.lin\_vel = 0.1

self.ang\_vel = 0.0

self.distance = 1.0

self.publisher = self.create\_publisher(Twist, "/cmd\_vel", 10)

self.move(0.1,5,True)

time.sleep(2)

self.rotate(30,125,False)

self.stop()

def move(self, speed, time, is\_forward):

t0= self.get\_clock().now()

self.velocity = Twist()

if(is\_forward):

self.velocity.linear.x = abs(speed)

self.get\_logger().info(“Turtlebot moving forward”)

else:

self.velocity.linear.x =-abs(speed)

self.get\_logger().info(“Turtlebot moving backward”)

t1= self.get\_clock().now()

if (t1-t0)>time:

self.get\_logger().info(“Time closed”)

self.get\_logger().warn(“Stopping the robot”)

self.velocity.linear.x =0

self.publisher.publish(self.velocity)

def stop(self):

self.velocity = Twist()

self.velocity.linear.x=0

self.publisher.publish(self.velocity)

def rotate(self, ang\_speed\_deg,relative,speed\_deg,clockwise):

self.velocity = Twist()

self.velocity.linear.x=0

ang\_speed=math.radians(abs(ang\_speed\_deg))

if(clockwise):

self.velocity.angular.z=-abs(ang\_speed)

else:

self.velocity.angular.z=abs(ang\_speed)

angle\_moved = 0

t0= self.get\_clock().now()

while(True):

self.publisher.publish(self.velocity)

self.get\_logger().info(“Turtlebot ratates”)

t1= self.get\_clock().now()

current\_ang = (t1-t0)\*ang\_speed\_degree

if(current\_ang > relative\_speed\_deg):

self.get\_logger().info(“Reached”)

break

self.velocity.angular.z=0

self.publisher.publish(self.velocity)

def main(args=None):

rclpy.init(args=args)

node = MoveRobot()

rclpy.spin(node)

rclpy.shutdown()

if \_\_name\_\_ == "\_\_main\_\_":

main()