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 (in terminal)

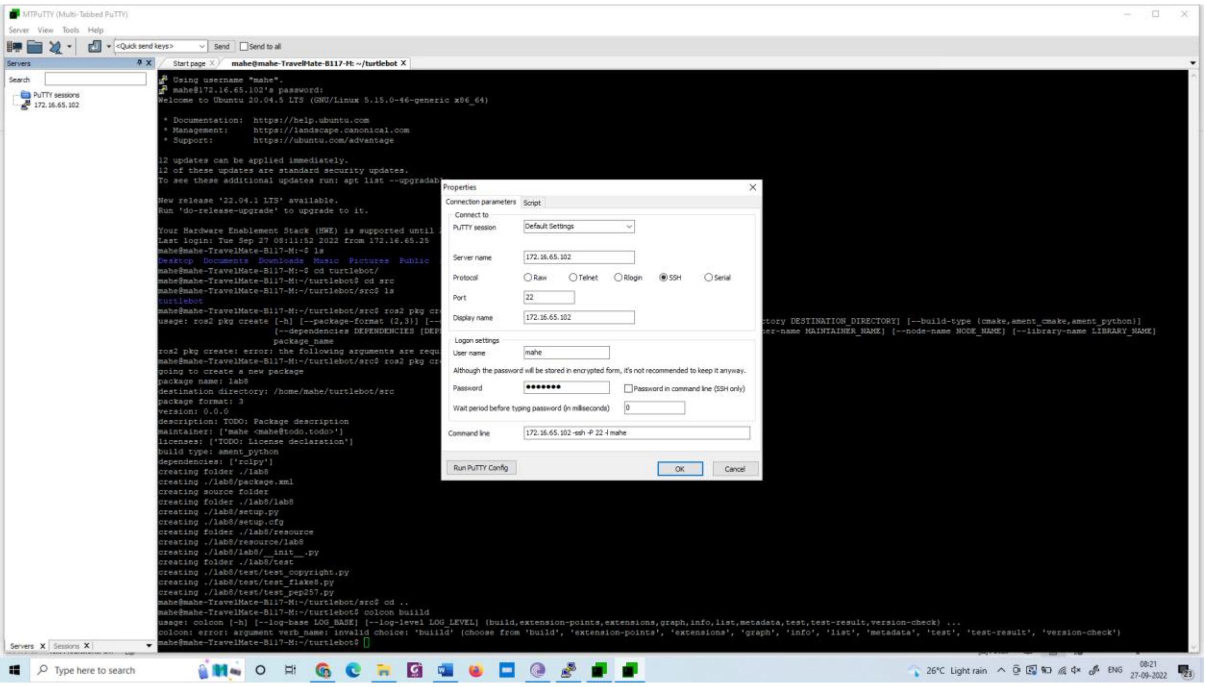
Connect camera USB and Kubuki USB and switch on Kubuki.
Switch on the netbook.
Use Remmina to connect to turtlebot from Ubuntu/mtputty from windows
See that the turtlebot laptop and remote laptop are connected to same WiFi
Find the address of netbook placed on turtlebot
Open terminal and type
$ifconfig
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
```



```
Using username "mahe".
mahe@172.16.65.102's password:
Welcome to Ubuntu 20.04.5 LTS (GNU/Linux 5.15.0-46-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

12 updates can be applied immediately.
12 of these updates are standard security updates.
To see these additional updates run: apt list --upgradab

New release '22.04.1 LTS' available.
Run 'do-release-upgrade' to upgrade to it.

Your Hardware Enablement Stack (HWE) is supported until
Last login: Tue Sep 27 08:11:52 2022 from 172.16.65.25
mahe@mahe-TravelMate-B117-M:~$ ls
Desktop  Documents  Downloads  Music  Pictures  Public
mahe@mahe-TravelMate-B117-M:~$ cd turtlebot/
mahe@mahe-TravelMate-B117-M:~/turtlebot$ cd src
mahe@mahe-TravelMate-B117-M:~/turtlebot/src$ ls
turtlebot
mahe@mahe-TravelMate-B117-M:~/turtlebot/src$ ros2 pkg cr
usage: ros2 pkg create [-h] [--package-format {2,3}] [--                                            ctory DESTINATION_DIRECTORY] [--build-type {cmake,ament_cmake,ament_python}]
                       [--dependencies DEPENDENCIES [DEP                                            er-name MAINTAINER_NAME] [--node-name NODE_NAME] [--library-name LIBRARY_NAME]
                       package_name
ros2 pkg create: error: the following arguments are requ
mahe@mahe-TravelMate-B117-M:~/turtlebot/src$ ros2 pkg cr
going to create a new package
package name: lab8
destination directory: /home/mahe/turtlebot/src
package format: 3
version: 0.0.0
description: TODO: Package description
maintainer: ['mahe <mahe@todo.todo>']
licenses: ['TODO: License declaration']
build type: ament_python
dependencies: ['rclpy']
creating folder ./lab8
creating ./lab8/package.xml
creating source folder
creating folder ./lab8/lab8
creating ./lab8/setup.py
creating ./lab8/setup.cfg
creating folder ./lab8/resource
creating ./lab8/resource/lab8
creating ./lab8/lab8/__init__.py
creating folder ./lab8/test
creating ./lab8/test/test_copyright.py
creating ./lab8/test/test_flake8.py
creating ./lab8/test/test_pep257.py
mahe@mahe-TravelMate-B117-M:~/turtlebot/src$ cd ..
mahe@mahe-TravelMate-B117-M:~/turtlebot$ colcon buiild
usage: colcon [-h] [--log-base LOG_BASE] [--log-level LOG_LEVEL] {build,extension-points,extensions,graph,info,list,metadata,test,test-result,version-check} ...
colcon: error: argument verb_name: invalid choice: 'buiild' (choose from 'build', 'extension-points', 'extensions', 'graph', 'info', 'list', 'metadata', 'test', 'test-result', 'version-check')
mahe@mahe-TravelMate-B117-M:~/turtlebot$
```

Properties

Connection parameters | Script

Connect to
PuTTY session: Default Settings
Server name: 172.16.65.102
Protocol: ○ Raw ○ Telnet ○ Rlogin ● SSH ○ Serial
Port: 22
Display name: 172.16.65.102

Logon settings
User name: mahe
Although the password will be stored in encrypted form, it's not recommended to keep it anyway.
Password: ●●●●●●● □ Password in command line (SSH only)
Wait period before typing password (in milliseconds): 0

Command line: 172.16.65.102 -ssh -P 22 -l mahe

Run PuTTY Config | OK | Cancel

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()
```