

Task 1

- For task 1: Provide the screenshot of the terminals from step 17.

17. Action: In three separate terminals run the turtlebot server, the turtlebot client, and the teleop node. You should now be able to drive your turtlebot using the keyboard.

The screenshot shows three terminal windows on a Linux desktop environment. The top window displays Python Turtle Graphics with a small black star icon in the center. The middle window shows the command line output for starting the turtlebot server, and the bottom window shows the command line output for starting the turtlebot client. Both windows show the ROS command-line interface (CLI) output.

Terminal 1 (Top): Python Turtle Graphics

```
parallels@ubuntu-linux-22-04-02-desktop:~/Documents/ECSE4965/Lab4/roscourse_ws$ ros2 run teleop_twist_keyboard teleop_twist_keyboard
This node takes keypresses from the keyboard and publishes them as Twist/TwistStamped messages. It works best with a US keyboard layout.
-----
Moving around:
  u   t   o
  j   k   l
  m   ,   .
For Holonomic mode (strafing), hold down the shift key:
-----
  U   I   O
  J   K   L
  M   <   >
t : up (+z)
b : down (-z)
anything else : stop
q/z : increase/decrease max speeds by 10%
w/x : increase/decrease only linear speed by 10%
e/c : increase/decrease only angular speed by 10%
CTRL-C to quit
currently:    speed 0.5      turn 1.0
currently:    speed 0.55     turn 1.0
currently:    speed 0.49500000000000005  turn 1.0
currently:    speed 0.49500000000000005  turn 0.9
currently:    speed 0.49050000000000005  turn 0.9
currently:    speed 0.5390550000000002  turn 0.9
currently:    speed 0.5929050000000002  turn 0.9
currently:    speed 0.6522565500000003  turn 0.9
currently:    speed 0.7174822050000004  turn 0.9
currently:    speed 0.7892304255000004  turn 0.9
currently:    speed 0.8681534680500006  turn 0.9
currently:    speed 0.9549688148550007  turn 0.9
currently:    speed 1.0504650963405008  turn 0.9
currently:    speed 1.15551226597451  turn 0.9
currently:    speed 1.2710634925720061  turn 0.9
currently:    speed 1.3981698418292060  turn 0.9
```

Terminal 2 (Middle): ROS Command Line

```
parallels@ubuntu-linux-22-04-02-desktop:~/Documents/ECSE4965/Lab4/roscourse_ws$ source install/setup.bash
parallels@ubuntu-linux-22-04-02-desktop:~/Documents/ECSE4965/Lab4/roscourse_ws$ ros2 run python_turtle turtlebot_server
[INFO] [1708996791.31454739] [turtleServer]: Turtlebot server started!
```

Terminal 3 (Bottom): ROS Command Line

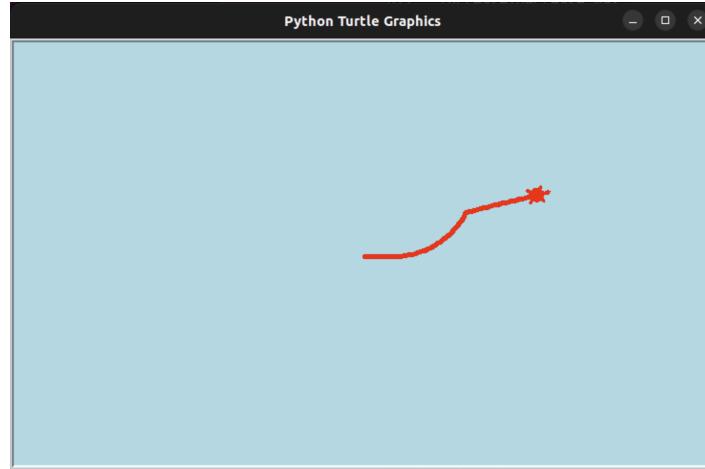
```
parallels@ubuntu-linux-22-04-02-desktop:~/Documents/ECSE4965/Lab4/roscourse_ws$ source install/setup.bash
parallels@ubuntu-linux-22-04-02-desktop:~/Documents/ECSE4965/Lab4/roscourse_ws$ ros2 run python_turtle turtlebot_client
[INFO] [1708996794.241812727] [turtleClient]: Turtlebot Client Started!
```

Task 2

- For task 2: Provide the modified turtlebot_client.py file with appropriate comments and formatting from step 14.
- For task 2: Show the turtle from step 12. It should be a different color than the default parameter value.

```
python_turtle > python_turtle > ⚡ turtlebot_client.py
 1  import rclpy
 2  from rclpy.node import Node
 3  import math
 4  import random
 5
 6  import turtle
 7
 8  from geometry_msgs.msg import Twist, Pose
 9
10 from rclpy.parameter import Parameter
11 from rcl_interfaces.msg import ParameterDescriptor
12
13 from turtle_interfaces.srv import SetColor
14 from turtle_interfaces.msg import TurtleMsg
15
16 class TurtleClient(Node):
17     def __init__(self):
18         super().__init__('turtleClient')
19
20         ##### Display/Turtle Setup #####
21         self.screen = turtle.Screen()
22         self.screen.bgcolor('lightblue')
23         self.turtle_display = turtle.Turtle()
24         self.turtle_display.shape("turtle")
25         self.turtle = TurtleMsg()
26
27         ##### publisher define #####
28         self.twist_pub = self.create_publisher(Twist, 'turtleDrive', 1)
29         #####
30
31         ##### subscribing turtlebot state #####
32         self.turtle_sub = self.create_subscription(TurtleMsg, 'turtleState', self.turtle_callback, 1)
33
34         # Pen size
35         # Declare a new parameter for pen size with a default value
36         self.declare_parameter('pen_size', 5)
37         # Retrieve the pen size parameter value
38         pen_size = self.get_parameter('pen_size').get_parameter_value().integer_value
39         # Set the turtle's pen size
40         self.turtle_display.pensize(pen_size)
41
42         # Initial turtle color
43         self.declare_parameter('turtleColor', 'red', ParameterDescriptor(description= 'Initial color of the turtle'))
44         turtleColor = self.get_parameter('turtleColor').get_parameter_value().string_value
45         self.turtle_display.color(turtleColor)
46
47         # Turtle color service
48         self.color_cli = self.create_client(SetColor, 'SetColor')
49         while not self.color_cli.wait_for_service(timeout_sec=1.0):
50             self.get_logger().info('Color service not available, waiting...')
51         self.color_req = SetColor.Request()
52         self.color_req.color = self.get_parameter('turtleColor').get_parameter_value().string_value
53         self.server_call = True
54         self.service_future = self.color_cli.call_async(self.color_req)
```

Modified init method of turtlebot_client.py



Default color

A screenshot of a terminal window titled "parallels@ubuntu-linux-22-04-02-desktop: ~/Documents/ECSE...". The terminal shows ROS commands being run:

```
parallels@ubuntu-linux-22-04-02-desktop:~/Documents/ECSE4965/Lab4/roscourse_ws$ ros2 run python_turtle turtlebot_server
[INFO] [1709184578.665935293] [turtleServer]: Turtlebot server started!
[INFO] [1709184584.556260092] [turtleServer]: Turtle color set: blue
```

```
parallels@ubuntu-linux-22-04-02-desktop:~/Documents/ECSE4965/Lab4/roscourse_ws$ source install/setup.bash
parallels@ubuntu-linux-22-04-02-desktop:~/Documents/ECSE4965/Lab4/roscourse_ws$ ros2 run python_turtle turtlebot_client --ros-args --param turtleColor:=blue
[INFO] [1709184584.561153699] [turtleClient]: Turtlebot Client Started!
```

The terminal also displays the Python Turtle Graphics window with a blue line drawing. The window has a status bar with movement controls and a message about keyboard input. The status bar includes:

```
currently: speed 22.629627784088036 turn 1.0
currently: speed 24.892598562496842 turn 1.0
currently: speed 27.38184961874653 turn 1.0
currently: speed 30.120034580621184 turn 1.0
currently: speed 33.132038038683305 turn 1.0
currently: speed 36.44524184255164 turn 1.0
```

This node takes keypresses from the keyboard and publishes them as Twist/TwistStamped messages. It works best with a US keyboard layout.

Moving around:

u	i	o
j	k	l
m	,	.

For Holonomic mode (strafing), hold down the shift key:

U	I	O
J	K	L
M	<	>

t : up (+z)
b : down (-z)
anything else : stop

q/z : increase/decrease max speeds by 10%
w/x : increase/decrease only linear speed by 10%
e/c : increase/decrease only angular speed by 10%

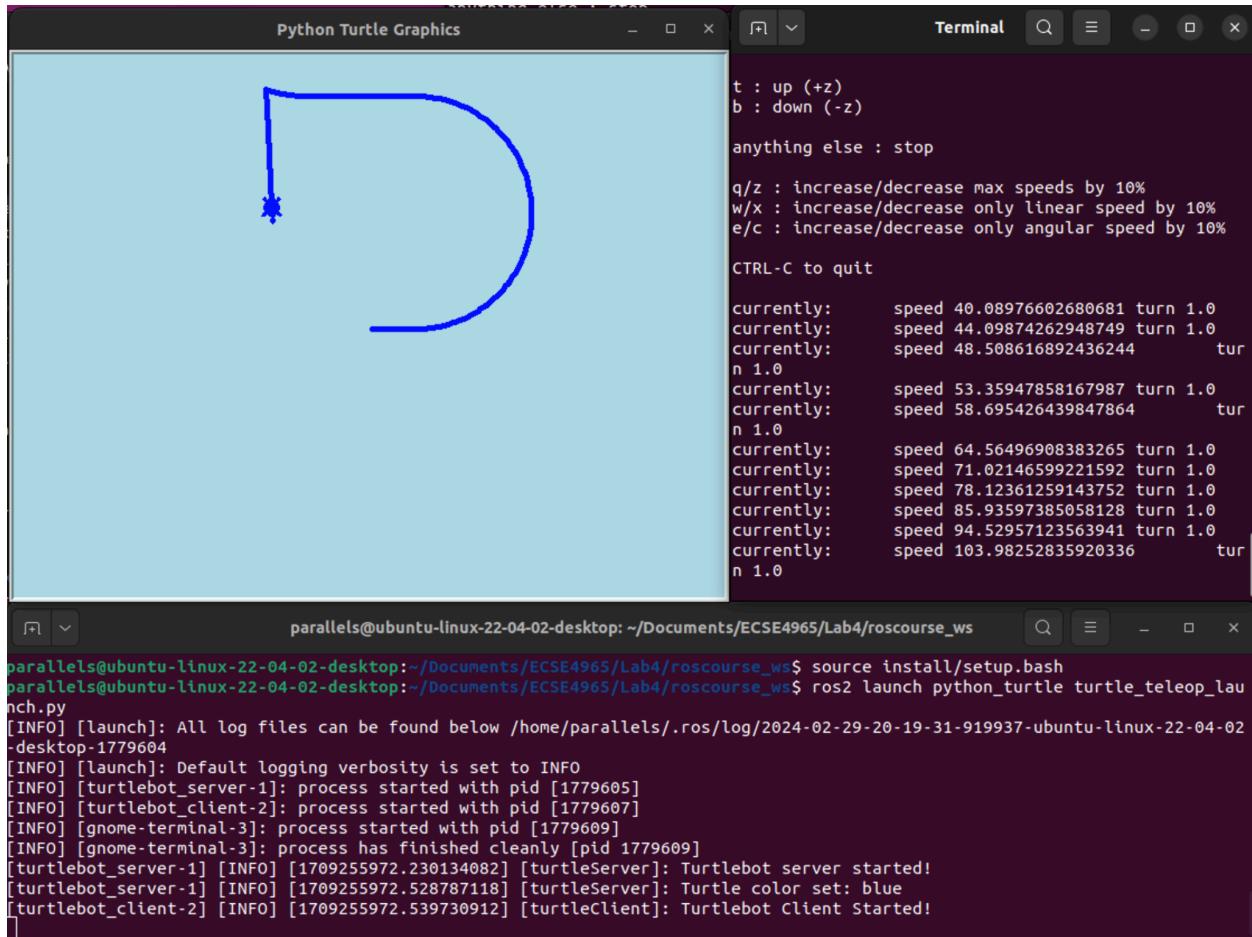
CTRL-C to quit

Overridden color

Task 3

- For task 3: Provide a screenshot of the terminals from step 8.

8. Action: Build and source. Now run everything with the command ros2 launch python_turtle turtle_teleop_launch.py



Task 4

- For task 4: Provide a screenshot from the host (the one with all the turtles) and the terminals performing remote control for step 12.

