Assignment -1

Shivam Shah (66503541)

Basics of Using ROS2

- Initially I started creating a folder of workspace named “Shah_Shivam_ws” where I build a package named “autoturtle” in this folder using this command “ros2 pkg create --build-type ament_python autoturtle”.
- Then, I created “my_teleop_node.py” inside my “Shah_Shivam_ws/autoturtle/autoturtle” and modified the “setup.py” file where I added my node in “console script”.
- Then, I built my workspace using “colcon build” command and ran my node successufully.

Code: -

- I used “rclpy”, “sys”, “tty” and “termios” libraries to where “rclpy” is used for ROS2 and other libraries were used to capture the keyboard button pressed to simulate the turtle according to the button pressed.
- I also used “Node” from “rclpy.node” library and “Twist“ from “geometry.msgs.msg” to send/publish the messages on “turtle/cmd_vel” having a buffer size of 10.
- I tried to use “msvcrt” library, but it needed to install, so instead of this library I used “termios” and other dependable libraries to capture the keys pressed.
- I created a “catch_button” function in which I tried to capture key buttons. I tried using input function but user need to press enter key to start the simulation. So, I used “termios”. To capture the key and return that char to main fuction for further processing.

- Then, I created “My_Teleop_Node” function which publishes the movements on the “turtle1/cmd_vel” which sending the linear and angular displacement.
- In main, I opened the “cmd_vel” where I will be publishing the movements and tried to capture the keys and send the following movements to “cmd_vel” for turtle simulation. And when we exit the function, we destroy the node and shut down the publishing page.

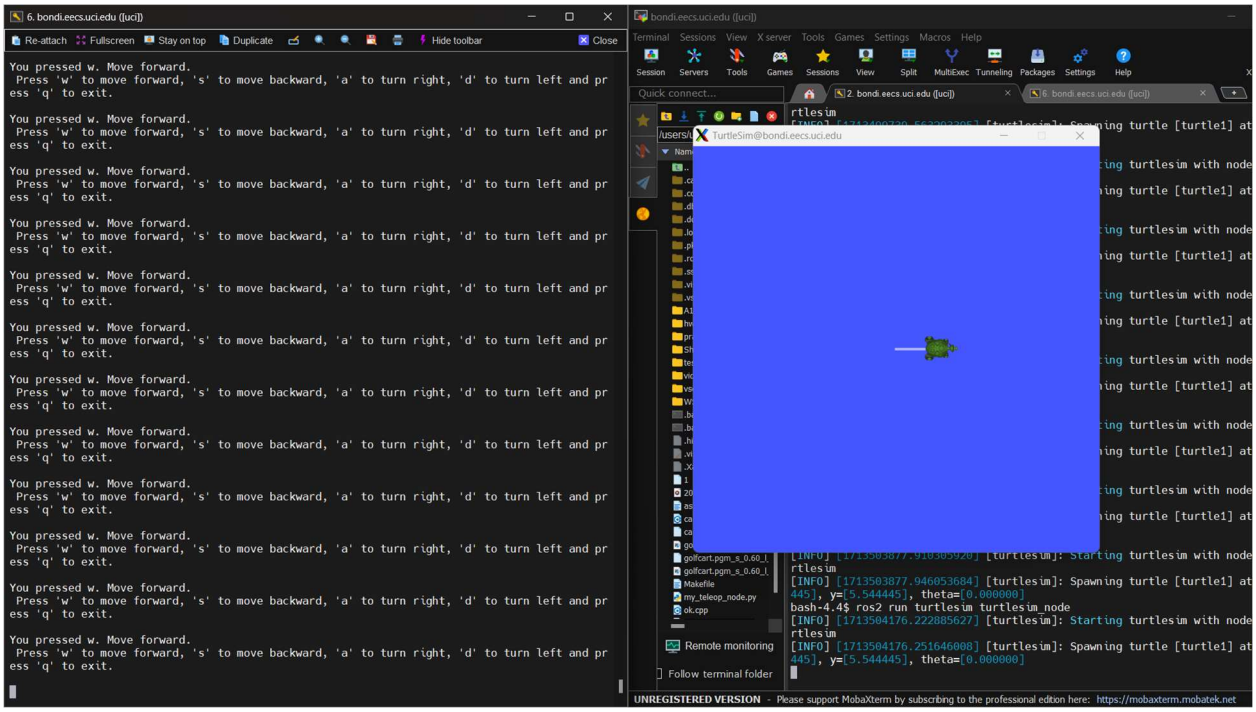
Code: -

```
import rclpy
from rclpy.node import Node
from geometry_msgs.msg import Twist
import sys
import tty
import termios


print("Press 'w' to move forward, 's' to move backward, 'a' to turn right, 'd' to
turn left and press 'q' to exit.\n")

# running the code
def main(args=None):
    rclpy.init(args=args)
    my_teleop_node = My_Teleop_Node()
    try:
        while True:
            button = catch_button()
            if button == 'w' or button == 'W':
                my_teleop_node.publish_twist(2.0, 0.0)
                print(f"You pressed {button}. Move forward.\n Press 'w' to move
forward, 's' to move backward, 'a' to turn right, 'd' to turn left and press 'q'
to exit.\n")
            elif button == 's' or button == 'S':
                my_teleop_node.publish_twist(-2.0, 0.0)
                print(f"You pressed {button}. Move backward.\n Press 'w' to move
forward, 's' to move backward, 'a' to turn right, 'd' to turn left and press 'q'
to exit.\n")
            elif button == 'a' or button == 'A':
                my_teleop_node.publish_twist(0.0, 1.2)
```

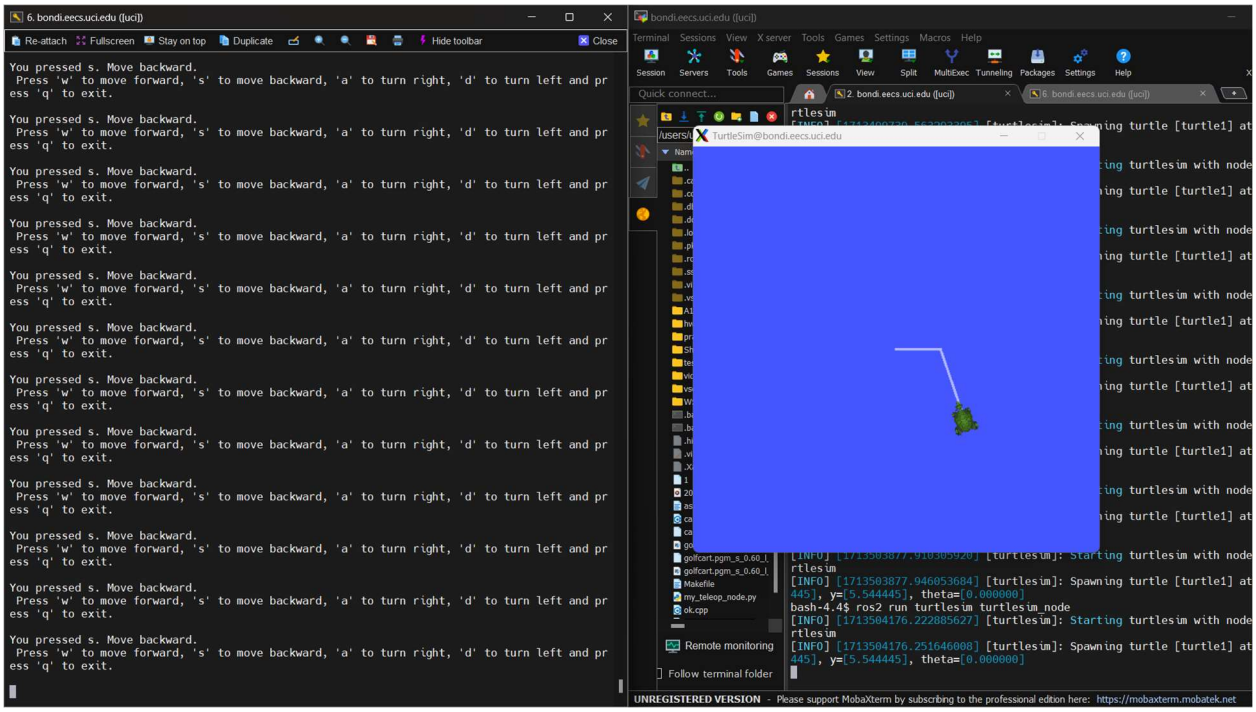
```
                print(f"You pressed {button}. Move left.\n Press 'w' to move
forward, 's' to move backward, 'a' to turn right, 'd' to turn left and press 'q'
to exit.\n")
            elif button == 'd' or button == 'D':
                my_teleop_node.publish_twist(0.0, -1.2)
                print(f"You pressed {button}. Move right.\n Press 'w' to move
forward, 's' to move backward, 'a' to turn right, 'd' to turn left and press 'q'
to exit.\n")
            elif button == 'q' or button == 'Q':
                print("You exited by pressing 'q'. Good bye.")
                break
    finally:
        my_teleop_node.destroy_node()
        rclpy.shutdown()

# to publish the message on cmd_vel
class My_Teleop_Node(Node):
    def __init__(self):
        super().__init__('my_teleop_node')
        self.publisher_ = self.create_publisher(Twist, '/turtle1/cmd_vel', 10)
        self.twist_msg_ = Twist()

    def publish_twist(self, linear_x, angular_z):
        self.twist_msg_.linear.x = linear_x
        self.twist_msg_.angular.z = angular_z
        self.publisher_.publish(self.twist_msg_)

#to catch the user input from the keyboard
def catch_button():
    f = sys.stdin.fileno()
    old_settings = termios.tcgetattr(f)
    try:
        tty.setraw(f)
        a = sys.stdin.read(1)
    finally:
        termios.tcsetattr(f, termios.TCSADRAIN, old_settings)
    return a

if __name__ == '__main__':
    main()
```

Simulation: -

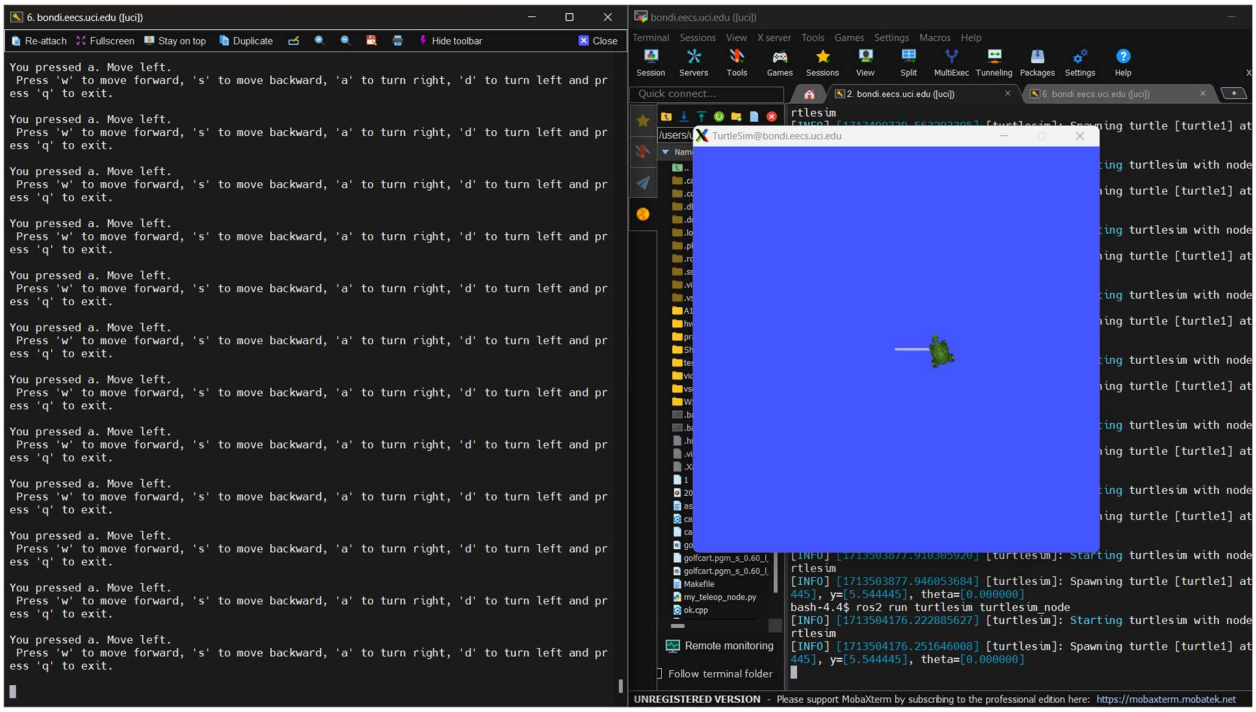
- When user presses “W” or “w” button from keyboard, the turtle will move forward with a speed of 2 units.



- When user presses “S” or “s” button from keyboard, the turtle will move backward with a speed of 2 units.



- When user presses “A” or “a” button from keyboard, the turtle will turn left with a speed of 1.2 units.


6. bondi.eecs.uci.edu ([uci])
Re-attach
Fullscreen
Stay on top
Duplicate
Hide toolbar
Close
You pressed a. Move left.
Press 'w' to move forward, 's' to move backward, 'a' to turn right, 'd' to turn left and press 'q' to exit.
bondi.eecs.uci.edu ([uci])
Terminal
Sessions
View
X server
Tools
Games
Settings
Macros
Help
Session
Servers
Tools
Games
Sessions
View
Split
MultiExec
Tunneling
Packages
Settings
Help
Quick connect...
2. bondi.eecs.uci.edu ([uci])
6. bondi.eecs.uci.edu ([uci])
TurtleSim@bondi.eecs.uci.edu
golfcart.pgm_s_0.60_l.
golfcart.pgm_s_0.60_l.
Makefile
my_teleop_node.py
ok.cpp
Remote monitoring
Follow terminal folder
rtlesim
[INFO] [1713503877.946053684] [turtlesim]: Spawning turtle [turtle1] at
445], y=[5.544445], theta=[0.000000]
bash-4.4$ ros2 run turtlesim turtlesim_node
[INFO] [1713504176.222885627] [turtlesim]: Starting turtlesim with node
rtlesim
[INFO] [1713504176.251646008] [turtlesim]: Spawning turtle [turtle1] at
445], y=[5.544445], theta=[0.000000]
UNREGISTERED VERSION - Please support MobaXterm by subscribing to the professional edition here: https://mobaxterm.mobatek.net