teleop_python.py

```python
#!/usr/bin/env python3
import rclpy
import os
import select
import sys
from rclpy.node import Node
from geometry_msgs.msg import Twist

if os.name == "nt":
    import msvcrt
else:
    import termios
    import tty

BURGER_MAX_LIN_VEL = 0.21
BURGER_MIN_LIN_VEL = 0.01
BURGER_MAX_ANG_VEL = 2.50
BURGER_MIN_ANG_VEL = 0.1

msg = """
Control Your TurtleBot3!
---------------------------
Moving around:
            up
   left    down    right

u : increase linear velocity and angular velocity
j : reset linear velocity and angular velocity
m : decrease linear velocity and angular velocity

CTRL-C to quit
"""


def get_key(settings):
    if os.name == "nt":
        return msvcrt.getch().decode("utf-8")
    tty.setraw(sys.stdin.fileno())
    rlist, _, _ = select.select([sys.stdin], [], [], 0.1)
    if rlist:
        key = sys.stdin.read(1)
        if key == "\x1b":
            key += sys.stdin.read(2)
    else:
        key = ""

    termios.tcsetattr(sys.stdin, termios.TCSADRAIN, settings)
    return key
```

```python


def print_vels_set(target_linear_velocity, target_angular_velocity):
    print(
        "setting :\tlinear velocity {0}\t angular velocity {1} ".format(
            target_linear_velocity, target_angular_velocity
        )
    )


def print_vels_run(target_linear_velocity, target_angular_velocity):
    print(
        "currently :\tlinear velocity {0}\t angular velocity {1} ".format(
            target_linear_velocity, target_angular_velocity
        )
    )


def main(args=None):
    settings = None
    if os.name != "nt":
        settings = termios.tcgetattr(sys.stdin)
    rclpy.init(args=args)
    teleop_node = rclpy.create_node("teleop_publisher")
    teleop_pub = teleop_node.create_publisher(Twist, "cmd_vel", 10)

    current_speed = [0.0, 0.0]
    max_speed_linear = 0.01
    max_speed_angular = 0.1
    twist = Twist()

    print(msg)

    while True:
        try:
            key = get_key(settings)
            if key == "\x03":
                print("Exiting...")
                break

            if key == "u":
                if max_speed_linear < BURGER_MAX_LIN_VEL:
                    max_speed_linear += 0.01
                else:
                    max_speed_linear = BURGER_MAX_LIN_VEL
                if max_speed_angular < BURGER_MAX_ANG_VEL:
                    max_speed_angular += 0.1
                else:
                    max_speed_angular = BURGER_MAX_ANG_VEL
                print_vels_set(max_speed_linear, max_speed_angular)
            elif key == "m":
                if max_speed_linear > BURGER_MIN_LIN_VEL:
                    max_speed_linear -= 0.01
                else:
                    max_speed_linear = BURGER_MIN_LIN_VEL
                if max_speed_angular > BURGER_MIN_ANG_VEL:
                    max_speed_angular -= 0.1
                else:
                    max_speed_angular = BURGER_MIN_ANG_VEL
                print_vels_set(max_speed_linear, max_speed_angular)
            elif key == "j":
                max_speed_linear = 0.01
                max_speed_angular = 0.1
                print_vels_set(max_speed_linear, max_speed_angular)

            if key == "\x1b[A" or key == "\x1b[B" or key == "\x1b[C" or key == "\x1b[D":
                if key == "\x1b[A":
                    current_speed[0] = max_speed_linear
                if key == "\x1b[B":
                    current_speed[0] = -max_speed_linear
                if key == "\x1b[C":
                    current_speed[1] = -max_speed_angular
                if key == "\x1b[D":
                    current_speed[1] = max_speed_angular
                print_vels_run(current_speed[0], current_speed[1])
            else:
                current_speed[0] = 0.0
                current_speed[1] = 0.0

            twist.linear.x = current_speed[0]
            twist.angular.z = current_speed[1]

            teleop_pub.publish(twist)
        except KeyboardInterrupt:
            break


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