

# Teleoperate control in python

## 一、前置設定

### 1. 設定機器型號

在 機器人端 與 控制電腦端 執行：

```
export TURTLEBOT3_MODEL=burger
```

建議加到 `~/.bashrc` 。

### 2. 啟動 TurtleBot3 Bringup（機器人端）

```
ros2 launch turtlebot3_bringup robot.launch.py
```

確認 `cmd_vel` 可用：

```
ros2 topic list
```

應該看到 `/cmd_vel`、`/odom`、`/tf`、`/scan`。

### 3. 確認 `cmd_vel`

```
ros2 topic info /cmd_vel
```

---

## 二、Python Teleop 程式（實體機器專用）

建立檔案 `teleop_turtlebot3.py`：

```
#!/usr/bin/env python3
import sys
import termios
import tty
import rclpy

from rclpy.node import Node
from geometry_msgs.msg import Twist
```

```

def get_key():
    fd = sys.stdin.fileno()
    old_settings = termios.tcgetattr(fd)
    try:
        tty.setraw(fd)
        key = sys.stdin.read(1)
    finally:
        termios.tcsetattr(fd, termios.TCSADRAIN, old_settings)
    return key

class TurtleBot3Teleop(Node):
    def __init__(self):
        super().__init__('turtlebot3_teleop_real')
        self.pub = self.create_publisher(Twist, '/cmd_vel', 10)

        # 實體 Burger 安全速度
        self.linear_speed = 0.15      # m/s
        self.angular_speed = 1.5      # rad/s
        self.get_logger().info(
            '\nReal TurtleBot3 Teleop\n'
            'w/s: forward/backward\n'
            'a/d: turn left/right\n'
            'x: stop\n'
            'q: quit\n'
        )

    def run(self):
        twist = Twist()

        while rclpy.ok():
            key = get_key()

            # 預設每次都先停（避免暴衝）
            twist.linear.x = 0.0
            twist.angular.z = 0.0

            if key == 'w':
                twist.linear.x = self.linear_speed

            elif key == 's':
                twist.linear.x = -self.linear_speed

            elif key == 'a':
                twist.angular.z = self.angular_speed

```

```
        elif key == 'd':
            twist.angular.z = -self.angular_speed

        elif key == 'x':
            pass # already stopped

        elif key == 'q':
            self.pub.publish(twist)
            self.get_logger().info('Teleop 結束')
            break

    self.pub.publish(twist)

def main(args=None):
    rclpy.init(args=args)
    node = TurtleBot3Teleop()

    try:
        node.run()

    except KeyboardInterrupt:
        pass

    finally:
        node.destroy_node()
        rclpy.shutdown()

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

---

## 三、操作方式

### 1 啟動 Bringup（機器人端）

```
ros2 launch turtlebot3_bringup robot.launch.py
```

### 2 執行 Teleop（控制端）

```
chmod +x teleop_turtlebot3.py
python3 teleop_turtlebot3.py
```

鍵盤對應：

按鍵	動作
w	前進
s	後退
a	左轉
d	右轉
x	停止
q	離開程式

⚠ Terminal 必須有 focus，否則讀不到鍵盤。

---

## 四、安全與注意事項

- 若按鍵沒反應：Terminal 沒 focus / ROS\_DOMAIN\_ID 不一致 / bringup 未啟動
- 速度太快：調整 `self.linear_speed` 與 `self.angular_speed`
- 放開鍵後仍動：程式已設計每次按鍵都重新發送速度訊息，避免累積速度  
建議初次使用：

```
self.linear_speed = 0.1  
self.angular_speed = 1.0
```

## 五、進階調整

- 速度即時調整 (+ / -)
- 安全 watchdog (無輸入自動停)
- 與 Navigation2 / SLAM 整合
- 急停 (空白鍵)
- 封裝成 ROS 2 package