ros2로 모터제어

ros2 rolling 버전 받음 → 라즈베리파이5는 ubuntu 23이상만 가능.

따라서 ubuntu 24.04를 받았고, 이에 호환되는 rolling과 jazzy 중 먼저 나와 안정화가 되어있는 rolling 버전을 사용한다.

ros rolling 설치는 생략.

실행 : `source /opt/ros/rolling/setup.bash`

ros가 실행 중인 지 확인하고 싶으면 echo $ROS_DISTRO (return값은 ros버전)

새로운 ROS2 패키지 생성

```
source /opt/ros/rolling/setup.bash
mkdir -p ~/ros2_ws/src
cd ~/ros2_ws/src
ros2 pkg create --build-type ament_python motor_controller
```

패키지 디렉토리로 이동

```
cd ~/ros2_ws/src/motor_controller
```

```
import rclpy
from rclpy.node import Node
from std_msgs.msg import String
import serial

class MotorController(Node):
    def __init__(self):
        super().__init__('motor_controller')
        self.publisher_ = self.create_publisher(String, 'mo
tor_command', 10)
        self.subscription = self.create_subscription(
            String,
            'motor_command',
            self.listener_callback,
```

```
            10)
        self.subscription  # prevent unused variable warnin
g

        self.serial_port = serial.Serial('/dev/ttyACM0', 96
00) # Adjust the port according to your setup

    def listener_callback(self, msg):
        self.get_logger().info('Received command: "%s"' % m
sg.data)
        self.serial_port.write(msg.data.encode())

def main(args=None):
    rclpy.init(args=args)
    motor_controller = MotorController()
    rclpy.spin(motor_controller)
    motor_controller.destroy_node()
    rclpy.shutdown()

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

```
mkdir motor_controller
touch motor_controller/motor_controller.py
chmod +x motor_controller/motor_controller.py
sudo nano motor_controller/motor_controller.py
motor_controller.py
```

`motor_controller.py` **파일 생성**

키보드 추가(0806)

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

class MotorController(Node):
```

```
    def __init__(self):
        super().__init__('motor_controller')
        self.subscription = self.create_subscription(
            Twist,
            'cmd_vel',
            self.listener_callback,
            10)

        self.last_msg_time = self.get_clock().now()
        self.timer = self.create_timer(0.1, self.check_time
out)

        try:
            self.serial_port = serial.Serial('/dev/ttyACM
0', 9600)
        except serial.SerialException as e:
            self.get_logger().error(f'Failed to connect to
serial port: {e}')
            self.serial_port = None

    def listener_callback(self, msg):
        self.last_msg_time = self.get_clock().now()
        linear_x = msg.linear.x
        angular_z = msg.angular.z
        self.get_logger().info(f'Received command - Linear:
{linear_x}, Angular: {angular_z}')

        if self.serial_port is not None:
            try:
                if linear_x > 0:
                    self.serial_port.write('F'.encode())  #
Forward
                elif linear_x < 0:
                    self.serial_port.write('B'.encode())  #
Backward
                else:
                    self.serial_port.write('S'.encode())  #
Stop
```

```
            except serial.SerialTimeoutException as e:
                self.get_logger().error(f'Serial write time
out: {e}')
            except serial.SerialException as e:
                self.get_logger().error(f'Serial write erro
r: {e}')
        else:
            self.get_logger().error('Serial port is not con
nected.')

    def check_timeout(self):
        current_time = self.get_clock().now()
        time_since_last_msg = (current_time - self.last_msg
_time).nanoseconds / 1e9
        if time_since_last_msg > 0.1:  # 0.1초 동안 새로운 명
령이 없으면 정지
            self.get_logger().info('No command received for
0.1 seconds, stopping motor.')
            if self.serial_port is not None:
                try:
                    self.serial_port.write('S'.encode())  #
Stop
                except serial.SerialTimeoutException as e:
                    self.get_logger().error(f'Serial write
timeout: {e}')
                except serial.SerialException as e:
                    self.get_logger().error(f'Serial write
error: {e}')

def main(args=None):
    rclpy.init(args=args)
    motor_controller = MotorController()
    rclpy.spin(motor_controller)
    motor_controller.destroy_node()
    rclpy.shutdown()

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

`setup.py` 파일 수정

```
sudo nano setup.py
------------------------------------------------------------
from setuptools import setup

package_name = 'motor_controller'

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='your_name',
    maintainer_email='your_email@example.com',
    description='ROS2 Motor Controller',
    license='Apache License 2.0',
    tests_require=['pytest'],
    entry_points={
        'console_scripts': [
            'motor_controller = motor_controller.motor_cont
roller:main'
        ],
    },
)
------------------------------------------------------
```

패키지 빌드

```
cd ~/ros2_ws
colcon build
```

```
source install/setup.bash
```

ROS2 노드 실행 : 터미널 여러개를 사용해야함 by tmux (창 생성 : ctrl+b ,c)

```
터미널1
ros2 run motor_controller motor_controller
터미널2
ros2 topic pub /motor_command std_msgs/String "data: 'F'"
ros2 topic pub /motor_command std_msgs/String "data: 'B'"
ros2 topic pub /motor_command std_msgs/String "data: 'S'"
```

gui를 사용하지않아서 아두이노 ide는 불가.

따라서 arduino cli를 사용해서 아두이노 코딩을 한다.

아두이노 CLI 설치

```
curl -fsSL https://raw.githubusercontent.com/arduino/arduin
o-cli/master/install.sh | sh
export PATH=$PATH:~/bin
#도구초기화
arduino-cli config init
```

보드와 라이브러리 업데이트 및 설치

```
arduino-cli core update-index
arduino-cli core install arduino:avr
```

보드 포트 설정(안해도되는듯)

```
arduino-cli board attach --port /dev/ttyACM0 arduino:avr:un
o
된 적이 없긴 함 항상 오류
```

아두이노 프로젝트 디렉토리 생성 및 스케치 파일 작성

```
mkdir -p ~/arduino/motor_controller
cd ~/arduino/motor_controller
```

```
sudo nano motor_controller.ino
----------------------------------------
#include <Arduino.h>
const int ENA = 9; // L298N ENA pin
const int IN1 = 8; // L298N IN1 pin
const int IN2 = 7; // L298N IN2 pin
const int ENB = 3; // L298N ENB pin
const int IN3 = 5; // L298N IN3 pin
const int IN4 = 4; // L298N IN4 pin

void setup() {
  Serial.begin(9600);
  pinMode(ENA, OUTPUT);
  pinMode(IN1, OUTPUT);
  pinMode(IN2, OUTPUT);
  pinMode(ENB, OUTPUT);
  pinMode(IN3, OUTPUT);
  pinMode(IN4, OUTPUT);
}

void loop() {
  if (Serial.available() > 0) {
    char command = Serial.read();
    switch(command) {
      case 'F': // Move forward
        digitalWrite(IN1, HIGH);
        digitalWrite(IN2, LOW);
        analogWrite(ENA, 255);
        digitalWrite(IN3, HIGH);
        digitalWrite(IN4, LOW);
        analogWrite(ENB, 255);
        break;
      case 'B': // Move backward
        digitalWrite(IN1, LOW);
        digitalWrite(IN2, HIGH);
        analogWrite(ENA, 255);
        digitalWrite(IN3, LOW);
        digitalWrite(IN4, HIGH);
```

```
        analogWrite(ENB, 255);
        break;
      case 'S': // Stop
        digitalWrite(IN1, LOW);
        digitalWrite(IN2, LOW);
        analogWrite(ENA, 0);
        digitalWrite(IN3, LOW);
        digitalWrite(IN4, LOW);
        analogWrite(ENB, 0);
        break;
    }
  }
}
-----------------------------------------
```

코드 컴파일 및 업로드

```
arduino-cli compile --fqbn arduino:avr:uno ~/arduino/motor_
controller
arduino-cli upload -p /dev/ttyACM0 --fqbn arduino:avr:uno
~/arduino/motor_controller
```