결과 보고서

임베디드응용및실습.
8주차 실습 과제

과 목 명	임베디드응용및실습
학 번	2019161068
이 름	백승진
제 출 일	2024년 10월 23일

1. Bluetooth 통신으로 움직이는 자동차 만들기

1) 코드

```
import threading
import serial
import RPi.GPIO as GPIO
import time

# 핀 번호 설정
PWMA = 18
PWMB = 23
AIN1 = 22
AIN2 = 27
BIN1 = 25
BIN2 = 24

# 초기 설정
GPIO.setwarnings(False)
GPIO.setmode(GPIO.BCM)
GPIO.setup(PWMA, GPIO.OUT)
GPIO.setup(PWMB, GPIO.OUT)
GPIO.setup(AIN1, GPIO.OUT)
GPIO.setup(AIN2, GPIO.OUT)
GPIO.setup(BIN1, GPIO.OUT)
GPIO.setup(BIN2, GPIO.OUT)
bleSerial = serial.Serial("/dev/ttyS0", baudrate=9600, timeout=1.0)

# 출력 주파수 설정
L_Motor = GPIO.PWM(PWMA, 250)
R_Motor = GPIO.PWM(PWMB, 250)

# 초기 정지 상태
L_Motor.start(0)
R_Motor.start(0)

# 수신 데이터 저장 변수
gData = ""

# 수신 데이터를 실시간 저장 함수
def serial_thread():
    global gData
    while True:
        data = bleSerial.readline()
        data = data.decode()
        gData = data

def main():
    global gData
    try:
        while True:
            # "go"를 수신 받으면
            if gData.find("go") >= 0:
```

```
                gData = ""
                # 방향 및 속도 설정 후 출력 (전진)
                GPIO.output(AIN1, 0)
                GPIO.output(AIN2, 1)
                GPIO.output(BIN1, 0)
                GPIO.output(BIN2, 1)
                L_Motor.ChangeDutyCycle(50)
                R_Motor.ChangeDutyCycle(50)
            # "back"을 수신 받으면
            elif gData.find("back") >= 0:
                gData = ""
                # 방향 및 속도 설정 후 출력 (후진)
                GPIO.output(AIN1, 1)
                GPIO.output(AIN2, 0)
                GPIO.output(BIN1, 1)
                GPIO.output(BIN2, 0)
                L_Motor.ChangeDutyCycle(50)
                R_Motor.ChangeDutyCycle(50)
            # "left"를 수신 받으면
            elif gData.find("left") >= 0:
                gData = ""
                # 방향 및 속도 설정 후 출력 (좌회전)
                GPIO.output(AIN1, 0)
                GPIO.output(AIN2, 1)
                GPIO.output(BIN1, 0)
                GPIO.output(BIN2, 1)
                L_Motor.ChangeDutyCycle(25)
                R_Motor.ChangeDutyCycle(75)
            # "right"를 수신 받으면
            elif gData.find("right") >= 0:
                gData = ""
                GPIO.output(AIN1, 0)
                GPIO.output(AIN2, 1)
                GPIO.output(BIN1, 0)
                GPIO.output(BIN2, 1)
                L_Motor.ChangeDutyCycle(75)
                R_Motor.ChangeDutyCycle(25)
            # "stop"을 수신 받으면
            elif gData.find("stop") >= 0:
                gData = ""
                L_Motor.ChangeDutyCycle(0)
                R_Motor.ChangeDutyCycle(0)
    except KeyboardInterrupt:
        pass

# main 모듈에서만 동작
if __name__ == '__main__':
    task1 = threading.Thread(target = serial_thread)
    task1.start()
    main()
    bleSerial.close()
```

2) 해석

첫 번째는, 필요한 모듈과 함수를 import 하는 단계이다. 과제를 수행하는 데 필요한 모듈 및 함수는 멀티 쓰레드를 위한 threading, 시리얼 통신을 위한 serial, 라즈베리파이를 제어하기 위한 RPi.GPIO 모듈, sleep 함수 사용을 위한 time 함수이다.

두 번째는, 사용할 모드에 맞는 핀 번호를 확인 후 명시적으로 정의한다. 과제를 수행하는 데 필요한 모드는 모터 속도 2개와 모터 방향 4개로, 번호는 각각 18, 23, 22, 27, 25, 24번이다.

세 번째는, 처음 실행할 때 환경 설정을 하는 단계이다. GPIO.setwarnings는 이전의 결과가 현재 실행에 영향을 줄 위험이 있는 경우에 경고 문구를 출력할지 말지를 결정하는 함수이다. 과제에서는 False로 설정한다. 다음으로 GPIO를 BCM 모드로 사용할 예정이므로 BCM 모드로 설정하고, 모터의 속도와 방향은 모두 출력 모드로 설정한다. 마지막으로 시리얼 통신의 경우 baudrate는 9600, 최대 timeout 시간은 1초로 설정한다.

네 번째는, 모터의 주파수를 설정한다. 좌우 모두 250으로 설정하였다.

다섯 번째는, 초기 상태를 정지 상태로 설정한다.

여섯 번째는, 수신받은 데이터를 저장하기 위한 변수 gData를 선언 및 초기화한다.

일곱 번째는, 수신 데이터를 실시간으로 저장하기 위한 함수를 정의한다. 내부 블록은 프로그램이 종료되지 않는 한, 계속 동작하도록 True 조건의 while 문을 이용하였다.

여덟 번째는, while 문 내부의 블록으로, 시리얼로부터 한 줄씩 수신 후 문자열로 변환하여 입력 데이터를 전역 변수 gData에 저장한다.

아홉 번째는, 실제로 동작을 위한 main 함수로, 위에서 확인한 입력 데이터를 통해 어떤 버튼이 눌린 지를 확인하고, 해당 버튼별로 적합한 동작을 수행하는 코드를 작성한다. 이때, 예외 처리를 위해 try-except 문을 사용한다. 여기서 예외 처리는 실행 중 키보드 입력으로 'Ctrl + C'가 들어왔을 경우 try-except 문을 빠져나가는 것이다. 아닌 경우에는 try 문 내부의 블록을 실행한다.

열 번째는, try 문 내부의 블록이며, 마찬가지로 프로그램이 종료되지 않는 한, 계속 동작하도록 True 조건의 while 문을 이용하였다.

열한 번째는, while 문 내부의 블록으로, 버튼 입력에 대응되는 입력 데이터를 확인 후, 모터의 방향과 속도를 조절한다. 만약 "go" 버튼을 눌러 go 값을 수신했다면 자동차는 전진하고, "back" 버튼을 눌러 back 값을 수신했다면 자동차는 후진한다. "left"와 "right", "stop"도 같은 방법으로 작성한다.

열두 번째는, main 모듈에서만 동작하도록 조건문을 이용하였고, 프로그램 실행 시 쓰레드와 main 함수가 동작하도록 작성한다.

마지막으로, 다음 실행에 영향을 주지 않도록, bleSerial.close 함수를 이용하여 종료 및 초기화한다.

3) 결과

출력 결과
"HW8" 영상 참조