Timbo Robot 개발 문서

이재진

June 13, 2024

Contents

1 개요

1.1 프로그램 소개

이 문서에서 설명하는 기능은 모두 MicroPython code로 작성되었습니다. 섹션당 프로그램별 상세한 설명을 기술합니다.

2 기능

2.1 ESPNow 콜백 함수

우선 다른 기능들을 살펴보기 전에 ESPNow의 통신 콜백 함수인 **ESPNow.irq()**에 대해 이해할 필요가 있습니다. 이 함수는 인터럽트 기반 함수로, 다른 peer로부터 메시지가 오는 순간 인터럽트가 걸리면서 메시지를 수신하게 됩니다. 이를 활용한 구현이 root/main_program/main.py 프로그램 내의 **recv_cb_slave**, **recv_cb_queen** 함수입니다. 이름에서 알 수 있듯이 각각 slave와 queen을 위한 메시지 수신 콜백 함수입니다.

이 함수에서 가장 중요한 요소는 **received_msg** 변수입니다. 여기에 도착한 메시지 내용이 저장됩니다. 메시지 포맷은 어떤 요청을 하는지 알려주는 **tag** 필드 (String), queen인지 slave인지 나타내는 **is_queen** 필드 (Boolean), 그 외에 유동적으로 바뀔 수 있는 페이로드가 존재할 수 있습니다.

이 콜백 함수의 구조적인 단점은 인터럽트가 걸릴 때마다 하나의 메시지만 수신할 수 있다는 점입니다. 따라서 한번에 여러 개의 메시지가 들어올 때 수신하기는 어렵습니다.

2.2 전체적인 코드의 흐름

맨 처음에 boot.py에 있는 코드가 main_program/main.py에 있는 메인 코드를 실행합니다. 이 메인 코드의 구조는 main.py의 runner 함수 안에 있는 큰 while loop 하나를 돌면서 버튼을 검사하면서 사용자와 상호작용하는 방식으로 구성되어 있습니다. 아래에 설명할 모든 모드에서 버튼을 항상 검사하도록 되어 있습니다.

2.3 모션 블록들의 모드

버튼을 한번씩 짧게 누름으로써 다음 모드로 이동할 수 있습니다.

mode 0 : (LED 색 : 무색) **아무 동작도 하지 않는 모드**로, 1) 만약 queen과 slave 모션 블록이 둘 다 이 모드에 존재하고 둘 다 버튼을 충분히 길게 눌렀다면 그루핑/언그루핑 동작을 진행하게 됩니다. 2) 만약 queen 혼자 충분히 길게 버튼을 눌렀다면 자신이 가진 network table 을 자신의 network table 내에 존재하는 모든 slave들에게 배포하게 됩니다. 3) 만약 queen이 mode 0으로 이동하게 되면 그룹 내의 모든 slave들도 일제히 mode 0으로 같이 이동하게 됩니다.

mode 1 : (LED 색 : 빨간색) 사용자가 모터를 돌려 원하는 동작을 모션블록에 저장할 수 있는 **record 모드**입니다. 만약 queen이 이 mode로 이동하게 되면 그룹 내의 모든 slave들이 mode 5 (record와 play를 동시에 하는 모드) 로 이동하게 됩니다.

mode 2 : (LED 색 : 연두색) 사용자가 mode 1에서 저장한 동작을 재생할 수 있는 **play 모드**입니다. 만약 queen이 이 mode로 이동하게 되면 그룹 내의 모든 slave들이 mode 2로 이동하게 됩니다.

mode 3 : (LED 색 : 무색) **AI 모드**로, 길게 누르면 Edge Impulse에서 생성한 모델 파일 (.tflite)을 mode 4 (웹서버 모드) 에서 모션 블록의 저장소에 다운로드한 것을 활용하게 되는 AI 애플리케이션 모드입니다. 현재는 간단한 MLP 예제가 들어가 있고, 추후 더 개발이 되어야 하는 모드입니다. 실행은 우선 root/model 폴더에 model.tflite가 있는지 확인하고, mode 3에서 버튼을 길게 누른 후 Thonny IDE에서 Run-Plotter를 선택하여 확인할 수 있습니다.

mode 4 : (LED 색 : 노란색) **웹서버 모드**로, 사용자가 연결한 WiFi를 활용하여 ESP32에서 연 웹소켓 서버를 활용하여 1) 임의의 파일을 ESP32 root 폴더에 업로드하거나, 2) main.py 파일을 업데이트하거나 (수동 OTA), 3) tflite 모델을 업로드하거나 (mode 3에서 쓰임), 4) mode 1과 2에서 사용하게 될 총 5개의 메모리 버퍼 중 몇 번째를 사용할지 고르거나, 5) 5개의 메모리 버퍼에 기록된 동작의 정보를 텍스트 파일로 다운로드받는 기능을 사용할 수 있습니다.

연결 조건은 다음과 같습니다 :

1) 와이파이에 연결이 되어 있어야 한다 : **섹션 1.4 (커뮤니케이션 블록)** 에서 더 자세히 설명함.

2) mode 4에서 버튼을 충분히 길게 누른다.

mode 4에서는 여러 개의 모션 블록들이 동시에 웹서버에 연결할 수 있습니다. 왜냐하면, 모션 블록마다 서로 다른 IP 주소를 할당받기 때문입니다.

mode 5 : (LED 색 : 빨간색) queen이 record 모드 (mode 2)로 이동했을 때, queen과 같은 ESPNow peer 그룹 (network table) 에 있는 slave들은 이 모드로 이동하게 됩니다. **이 모드는 버튼을 눌러서 도달할 수 없고, slave만이 앞의 조건을 만족했을 때만 도달할 수 있습니다.** 이 모드에서 slave는 queen이 record 하고 있는 동작을 실시간으로 따라서 돌게 됩니다. 여기서 버튼을 한 번 더 누르면 mode 0으로 되돌아가게 됩니다.

2.4 커뮤니케이션 블록

먼저, 커뮤니케이션 블록이 가질 수 있는 모드는 아래와 같습니다.

mode 0 : (LED 색 : 무색) **아무 동작도 하지 않는 모드**로, 만약 queen과 커뮤니케이션 블록이 둘

다 이 모드에 존재하고 둘 다 버튼을 충분히 길게 눌렀다면 그루핑/언그루핑 동작을 진행하게 됩니다. 커뮤니케이션 블록도 모션블록으로부터 메시지를 받는 **recv_cb_slave** 함수가 존재하는데, tag = distribute 메시지가 들어오면 mode 1로 전환됩니다. 또한 만약 mode 1로 한번이라도 전환된 적이 이전에 있다면, 별도의 기다림 없이 바로 mode 1로 전환됩니다.

mode 1 : (LED 색 : 연두색) **뒤에 기술할 AP 모드 제어/ Firmater 모드** 입니다. **mode 2** : (LED 색 : 주황색) **아무 동작도 하지 않는 모드**로, 버튼을 한번 누르면 mode 0으로 돌아가게 됩니다.
커뮤니케이션 블록은 아래의 기능들을 가지고 있습니다.

1) **Wi-Fi 초기 설정 기능** : 와이파이 설정을 하지 않았거나, 현재 와이파이 설정이 사용자 위치에서 스캔되지 않을 때 처음 부팅하면 boot.py의**boot_with_update** 함수에서 커뮤니케이션 블록을 자동으로 AP 모드로 전환합니다. 그 이후에 스마트폰이나 기타 전자기기로 인터넷 주소 192.168.4.1에 접속하여 현재 사용 가능한 와이파이의 비밀번호를 입력하면, 기기 자체의 Wi-Fi가 설정됩니다.

2) **Entry 를 이용한 Serial Firmater 및 AP 모드를 활용한 그룹 내의 모션 블록들을 제어** : 우선 1)의 설정을 마친 상태에서 시작합니다. Wi-Fi 공유를 원하는 다른 모션 블록과 완전히 그루핑을 마치고, queen 블록이 자신의 테이블을 배포하도록 합니다. 이렇게 함으로써 각 모션 블록과 커뮤니케이션 블록이 그룹 내의 모든 peer들의 정보를 가지고 있고, mode 1로 모드가 전환됩니다. 모드 1에 진입하면 먼저 시리얼 메시지가 들어오고 있는지 확인합니다. 만약에 들어오고 있다면 Entry에서 시리얼 메시지를 보내는 경우로 생각하고 Firmater 모드로 전환합니다. 아니라면 자동으로 AP로 전환되어 192.168.4.1 주소에서 서버에 접속할 수 있습니다. 이 웹서버에서 각각의 모션블록을 'PLAY' 하거나 'STOP' 할 수 있습니다.

3) **Wi-Fi 배포 기능** : 1)에서 설정한 Wi-Fi 설정 파일을 2)의 AP 웹 서버 사이트에서 그룹의 모든 모션 블록에게 배포할 수 있습니다.

2.5 그루핑, 언그루핑, 테이블 배포

세 가지 기능은 모두 micropython 상의 ESPNow 라이브러리를 활용하고 있고, 메커니즘은 src/espnow 폴더 안의 **grouping.py** 파일에 구현되어 있습니다.

2.5.1 그루핑, 언그루핑

그루핑과 언그루핑은 네트워크 테이블 net_table에서 peer가 추가되거나 삭제되는 것을 제외하고 정확히 똑같은 동작을 합니다. 두 동작을 하기 위해서는 Queen 블록의 버튼과 Slave 블록의 버튼을 mode 0 상태에서 적당히 길게 누르면 됩니다.

1) **그루핑** : 처음에 만약 slave와 queen의 버튼이 둘 다 길게 눌렸다면, 이 때 grouping.py의 grouping 함수에서는 만약 이 함수를 실행 중인 모션 블록이 queen이라면 1.5초 동안 delay를 주게 됩니다. 이 시간 동안 slave에서 만약에 메시지를 보낸 게 있다면 queen은 자신의 네트워크 테이블에 slave의 mac address를 추가하게 됩니다. 추가한 이후에 slave에게 자신의 mac address를 포함하는 tag = grouping 메시지를 보내게 됩니다. 반대로 slave의 입장에서 버튼을 처음 길게 누른 시점으로 돌아가보면, slave는 grouping 함수가 실행되는 즉시 tag = grouping 메시지를 브로드캐스팅하게 됩니다. 그리고 3.5초동안 delay를 주게 되는데, 그 이유는 앞에서 queen이 네트워크 테이블에 slave를 추가하고 자신의 mac address를 포함하는 tag = grouping 메시지를 보내는 시간 전에 slave가 먼저 grouping 함수에서 빠져 나오게 되면 queen의 grouping 메시지를 받지 못하게 되어 grouping이 되지 않기 때문입니다.

2) **언그루핑** : 1)과 동일하고, 네트워크 테이블에서 각자의 mac address를 제거한다는 점만 다릅니다.

2.5.2 테이블 배포

테이블 배포는 Queen 블록의 버튼만을 mode 0 상태에서 길게 누르면 됩니다. 구현 상세는 아래와 같습니다.

grouping 함수가 실행되고, 만약 1.5초를 기다려도 slave로부터 아무 메시지도 오지 않는다면 테이블 배포의 의미로 생각하고 자신의 네트워크 테이블 정보를 포함한 tag = distribute 메시지를 멀티 캐스팅하게 됩니다. (멀티 캐스팅은 브로드 캐스팅과는 다르게 현재 자신과 같은 ESPNow peer 그룹 내의 모션 블록에게만 메시지를 전달하도록 되어 있습니다.) 그러면 slave 쪽에서 멀티 캐스팅한 메시지를 recv_cb_slave

함수에 의해 인터럽트가 걸려 받게 되고, 그 메시지에 포함된 queen의 네트워크 테이블 정보로 자신의 네트워크 테이블 정보를 overwrite 합니다.

2.6 레코딩 기능

만약 임의의 모션블록이 mode 1에 있다면, 여기서 모터를 움직여서 해당 모터 동작을 모션 블록 메모리에 저장할 수 있습니다. 이 기능은 src/motion_block 폴더 내부의 BlockIODevices.py 파일에서 **record_motor** 과 main_program 폴더 내부의 main.py 파일의 **runner** 함수의 while 문 중 일부에 구현되어 있습니다.

record_motor 함수에서 모터에 해당하는 ADC 센서 값을 읽어들이는데, 읽는 주기를 **mem_counter** 변수로 관리하고 있습니다. record_motor가 runner 함수의 while loop에서 한번 실행될 때마다 mem_counter는 모듈러 1씩 증가하고, 30의 주기마다 그 값을 5개의 duty_memories 슬롯 중 하나(기본은 0으로 설정)에 순서대로 저장합니다. 추가로 mode 1에 처음 도달했을 때 이전에 저장되어 있는 메모리가 모두 지워지게 되는데, 이를 원치 않으면 mode 0에서 버튼을 두 번 빠르게 눌렀을 때 기존의 저장된 메모리를 유지할 수 있습니다.

2.7 플레이 기능

만약 임의의 모션블록이 mode 2에 있다면, mode 1에서 각 모션블록이 메모리에 저장하고 있는 동작을 플레이하게 됩니다. 이 기능은 src/motion_block 폴더 내부의 **move_motor** 함수와 main.py 파일의 **runner** 함수 while 문 중 일부에 구현되어 있습니다.

move_motor 함수는 메모리에 저장된 어떤 k번째 모터 값 m_k로 모터의 위치값을 이동시키고 싶다고 할 때, 이 m_k 값과 실제 모터 관련 ADC 센서 값 a_k 간의 오차를 활용하는 PID 제어를 활용하여 모터를 원하는 값에 가까운 값으로 유지되도록 효율적으로 제어합니다. 이 move_motor 함수는 main.py의 runner 함수의 while 문에서 1ms 마다 한번씩 실행되게 됩니다.

2.8 AI 기능

만약 임의의 모션블록이 mode 3에 있고, 이 때 버튼을 적절한 시간 동안 누르고 있으면 AI 모드에 진입합니다. 이 기능은 src/ai/ai_app.py에 구현되어 있습니다. 현재 구현된 기능은 root/model 폴더 안의 .tflite 형식 모델 파일을 읽어들여서 간단하게 주어지는 입력에 대한 출력을 예측하는 regression을 수행하고 있습니다. 추후에 구현되어야 하는 부분은 Edge impulse를 사용하여 모델을 만들고 이것을 뒤에 소개될 웹서버 모드(mode 4)를 활용하여 기기로 모델을 다운로드하여 mode 3에서 이를 불러들여 사용하는 것입니다.

2.9 웹서버 기능

만약 임의의 모션블록이 mode 4에 있고, 이 때 버튼을 적절한 시간 동안 누르고 있으면 웹서버 모드에 진입합니다. 이 기능은 src/webserver/webserver.py에 구현되어 있습니다. 단 이 기능이 제대로 동작하려면 **기기가 와이파이 설정이 되어 있어야 합니다.** (와이파이 설정을 하려면 커뮤니케이션 블록을 통해서 해야만 합니다.) 현재 웹서버에 접속하려면 접속하는 다른 기기가 모션블록 와이파이 설정과 같은 와이파이에 연결되어 있어야 하고, 현재 이 주소를 확인하려면 IDE에서 출력되는 메시지를 보는 방법밖에 없어 잠시 웹 서버를 개설하여 사용자에게 이 주소를 보여주는 식으로 수정하려고 합니다.

웹서버에 접속하면, 1) 임의의 파일을 ESP32 root 폴더에 업로드하거나, 2) main.py 파일을 업데이트하거나 (수동 OTA), 3) tflite 모델을 업로드하거나 (mode 3에서 쓰임), 4) mode 1과 2에서 사용하게 될 총 5개의 메모리 버퍼 중 몇 번째를 사용할지 고르거나, 5) 5개의 메모리 버퍼에 기록된 동작의 정보를 텍스트 파일로 다운로드받는 기능을 사용할 수 있습니다.

2.10 AP 제어 기능

커뮤니케이션 블록에만 있는 기능으로, 사용하려면 다음 과정을 거쳐야 합니다:

1) 커뮤니케이션 블록과 queen 블록을 그루핑하고, 그 외에 다른 slave들도 queen과 그루핑한다.

2) queen 블록에서 테이블을 배포한다.

위 과정을 거치면 자동으로 커뮤니케이션 블록이 mode 1로 넘어가게 되고, Entry 프로그램에서 시리얼 메시지가 오지 않으면 몇 초 후에 AP 모드로 전환됩니다. 이 때 다른 기기로 AP에 연결하여 192.168.4.1로 접속하면, PLAY, STOP 등의 버튼으로 기기를 제어하거나 Share Wi-Fi 버튼으로 Wi-Fi 설정을 그룹 내 다른 모션 블록에 공유할 수 있습니다.

2.11 Entry를 활용한 Firmate 기능

커뮤니케이션 블록에만 있는 기능으로, 사용하려면 다음 과정을 거쳐야 합니다:

1) Entry 프로그램의 블록 리스트에서 [하드웨어] 를 선택하고 [연결 프로그램] - [아두이노 Uno 호환 보드] 를 선택한 후 나타나는 COM 포트를 선택한 상태로 둔다.

2) 커뮤니케이션 블록과 queen 블록을 그루핑하고, 그 외에 다른 slave들도 queen과 그루핑한다.

3) queen 블록에서 테이블을 배포한다.

위 과정을 거치면 자동으로 커뮤니케이션 블록이 mode 1로 넘어가게 되고, Entry에 연결되며 Firmate 모드에 들어가게 됩니다. Entry에서 읽어들일 때 (모션 블록에 PLAY, STOP 명령을 내릴 수 있음)는 [**디지털 i 번 핀 켜기/끄기**] 블록을 사용하면 되고, Entry에 입력할 때는 [**디지털 i 번 센서값**] (Entry에서 모션블록의 센서값을 읽어들일 수 있음) 블록을 적절히 사용하면 됩니다. Firmate 모드에 커뮤니케이션 블록이 진입한 채로 Entry 블록 프로그램의 [시작하기]를 누르면, 시리얼 통신이 시작되면서 모션 블록이 블록 코딩한 대로 제어가 됩니다.