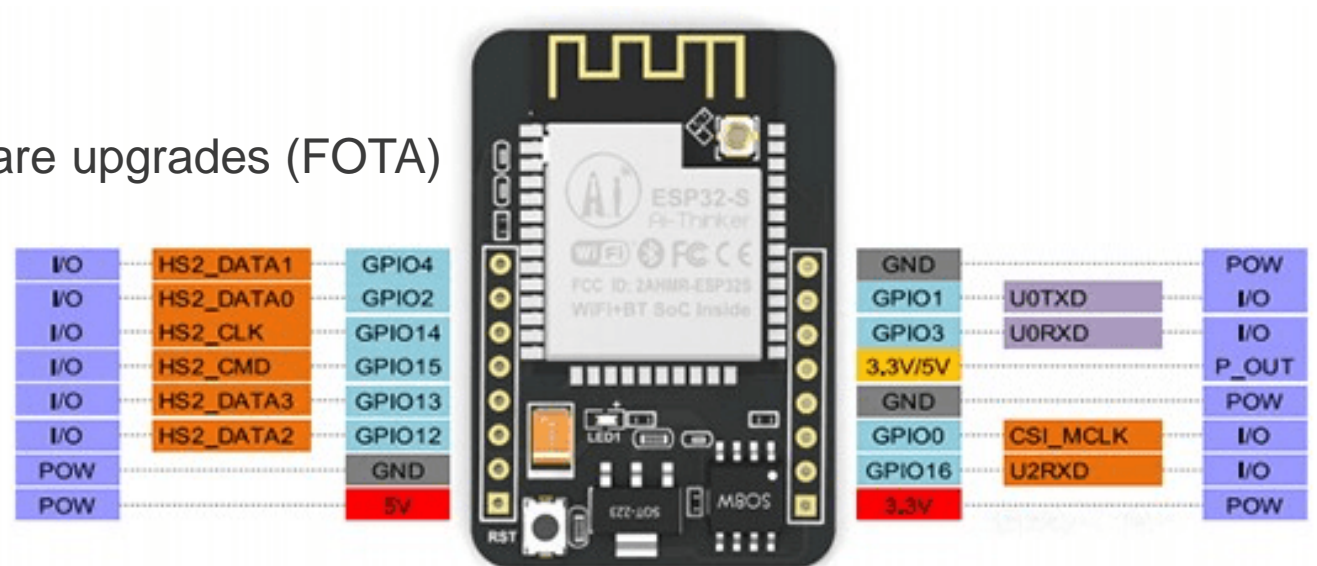


Features

Here is a list with the ESP32-CAM features:

- The smallest 802.11b/g/n Wi-Fi BT SoC module
- Low power 32-bit CPU,can also serve the application processor
- Up to 160MHz clock speed, summary computing power up to 600 DMIPS
- Built-in 520 KB SRAM, external 4MPSRAM
- Supports UART/SPI/I2C/PWM/ADC/DAC
- Support OV2640 and OV7670 cameras, built-in flash lamp
- Support image WiFi upload
- Support TF card
- Supports multiple sleep modes
- Embedded Lwip and FreeRTOS
- Supports STA/AP/STA+AP operation mode
- Support Smart Config/AirKiss technology
- Support for serial port local and remote firmware upgrades (FOTA)



ESP32-CAM Pinout

The following figure shows the ESP32-CAM pinout (AI-Thinker module). There are three GND pins and two pins for power: either 3.3V or 5.5V.

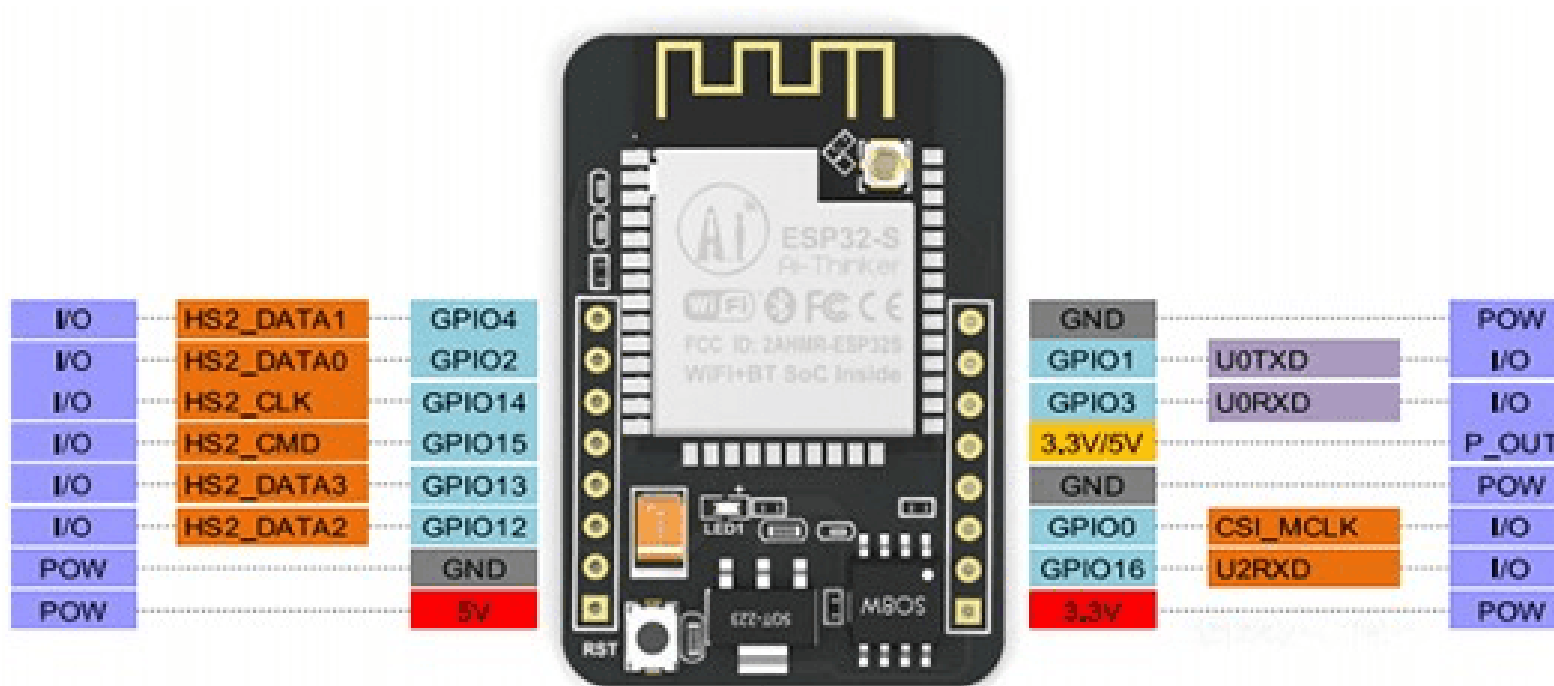
GPIO1 and GPIO3 are the serial pins.

You need these pins to upload code to your board.

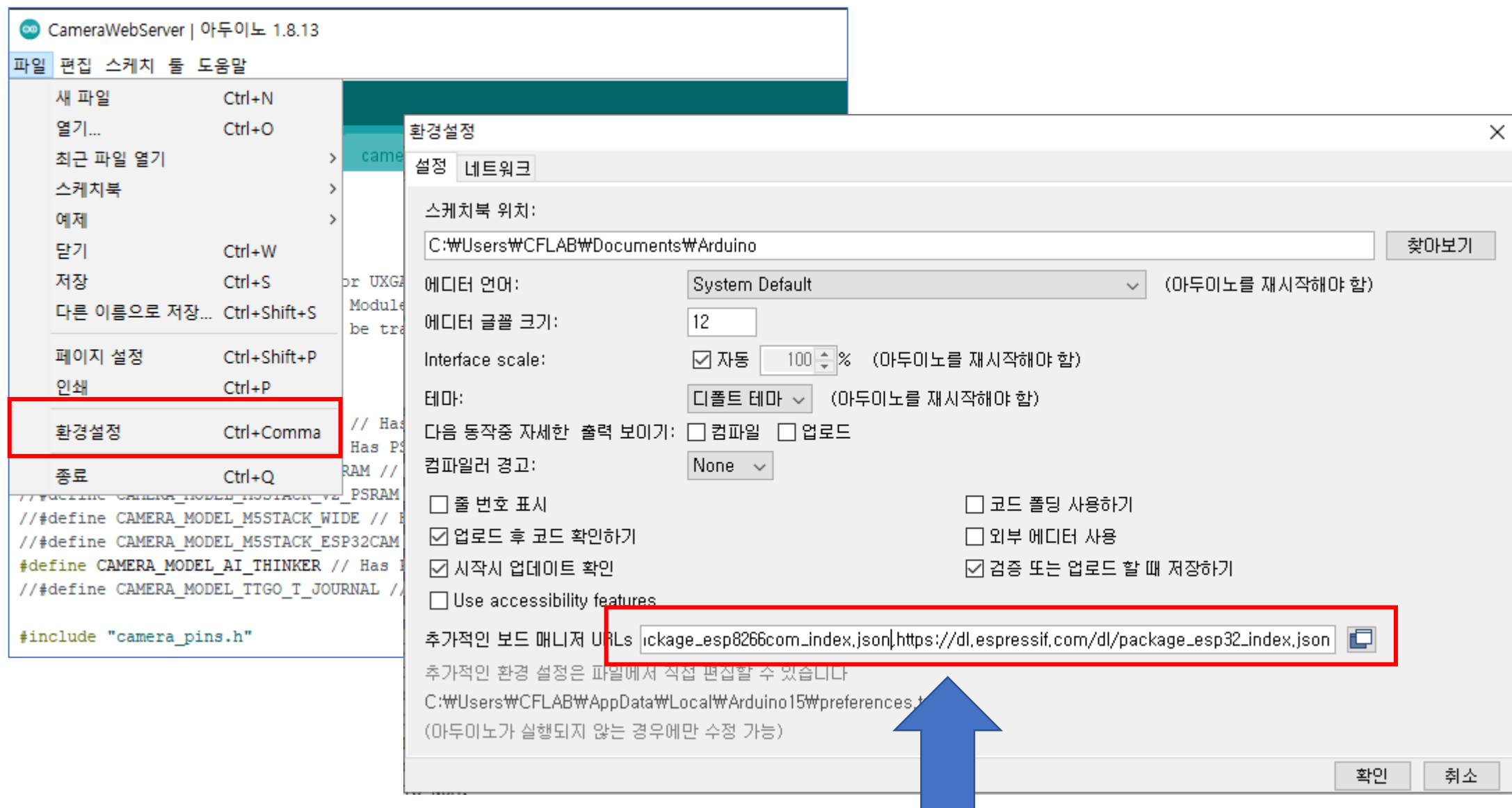
Additionally, **GPIO 0** also plays an important role, since it determines whether the ESP32 is in flashing mode or not. When **GPIO 0** is connected to GND, the ESP32 is in flashing mode.

The following pins are internally connected to the microSD card reader:

- GPIO 14: CLK
- GPIO 15: CMD
- GPIO 2: Data 0
- GPIO 4: Data 1 (also con
- GPIO 12: Data 2
- GPIO 13: Data 3

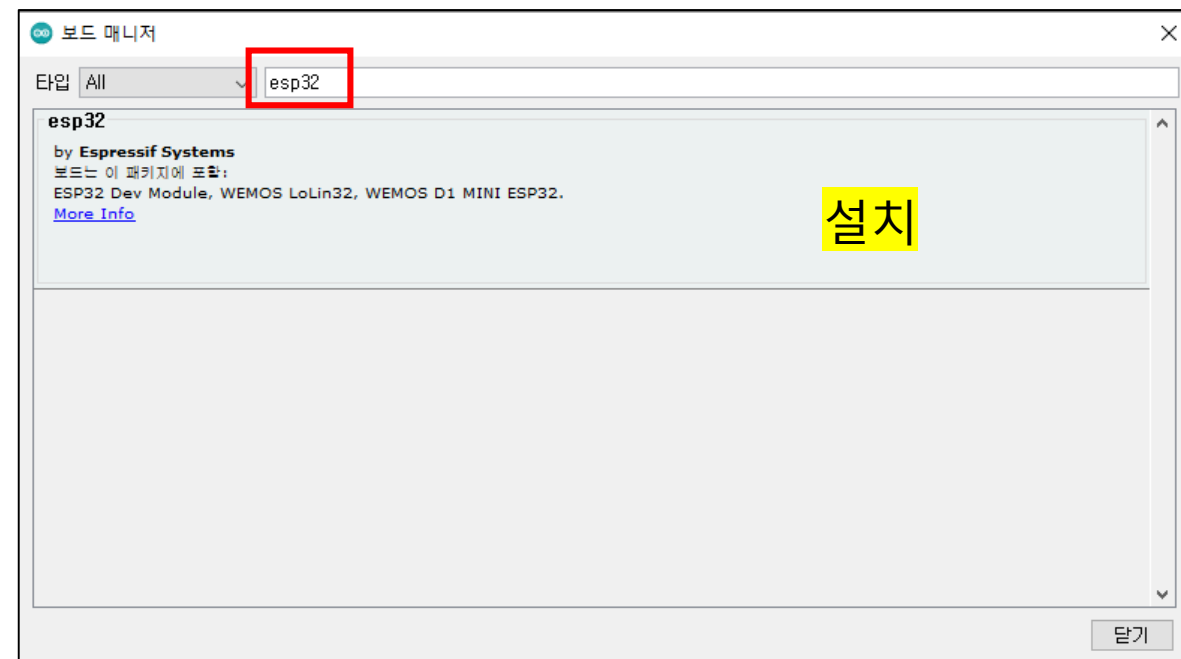
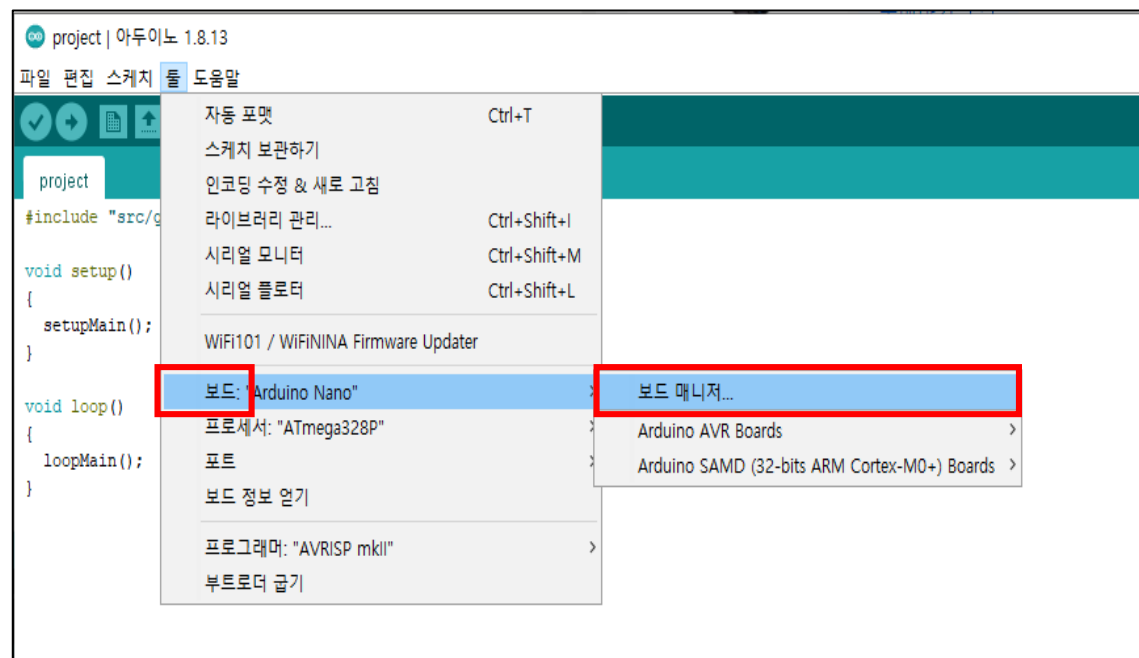


ESP32 CAM 보드 설치 1-3

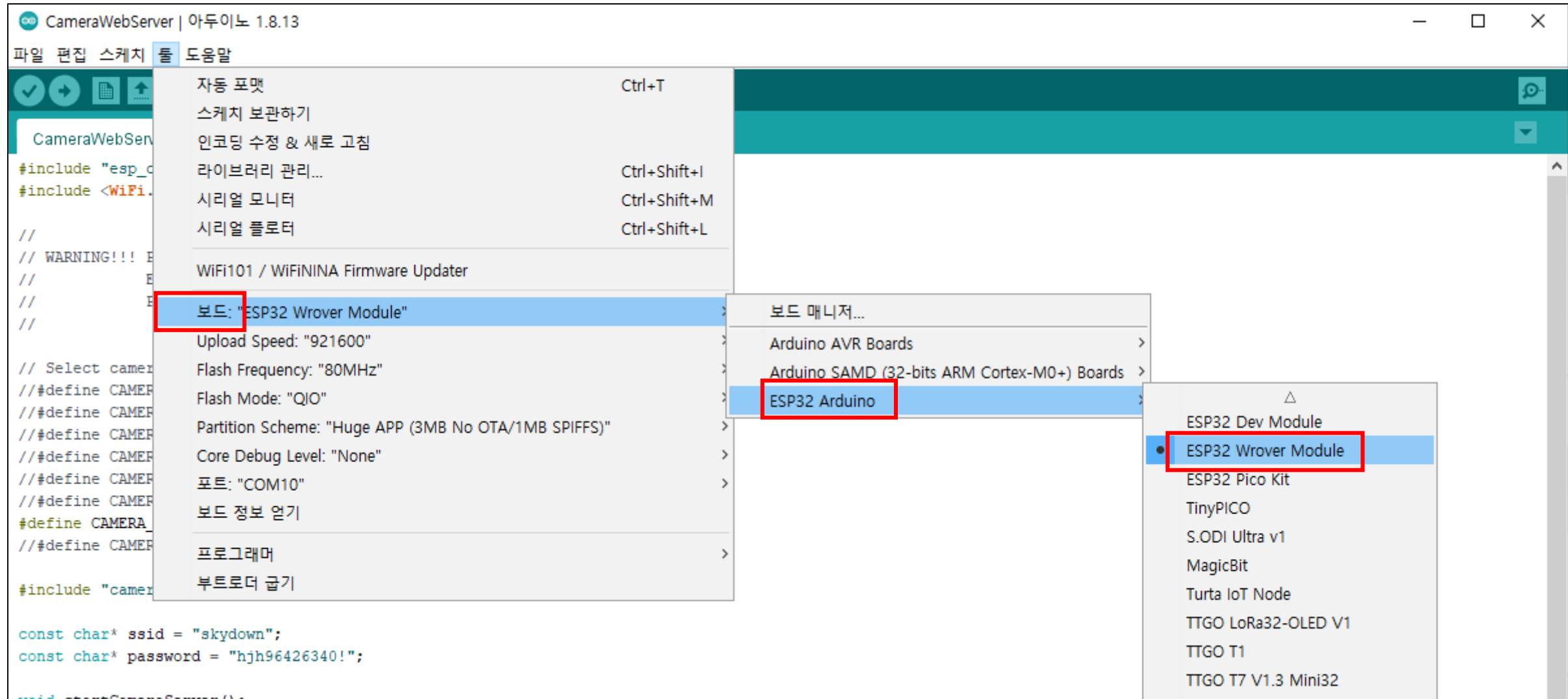


https://dl.espressif.com/dl/package_esp32_index.json,
http://arduino.esp8266.com/stable/package_esp8266com_index.json

ESP32 CAM 보드 설치 2-3



ESP32 CAM 보드 설치 3-3



ESP32 CAM 예제소스 테스트

```
CameraWebServer | 아두이노 1.8.13
파일 편집 스케치 툴 도움말

CameraWebServer app_httpd.cpp camera_index.h camera_pins.h

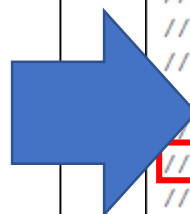
#include "esp_camera.h"
#include <WiFi.h>

//
// WARNING!!! PSRAM IC required for UXGA resolution and high JPEG quality
//           Ensure ESP32 Wrover Module or other board with PSRAM is selected
//           Partial images will be transmitted if image exceeds buffer size
//

// Select camera model
#define CAMERA_MODEL_WROVER_KIT // Has PSRAM
// #define CAMERA_MODEL_ESP_EYE // Has PSRAM
// #define CAMERA_MODEL_M5STACK_PSRAM // Has PSRAM
// #define CAMERA_MODEL_M5STACK_V2_PSRAM // M5Camera version B Has PSRAM
// #define CAMERA_MODEL_M5STACK_WIDE // Has PSRAM
// #define CAMERA_MODEL_M5STACK_ESP32CAM // No PSRAM
// #define CAMERA_MODEL_AI_THINKER // Has PSRAM
// #define CAMERA_MODEL_TTGO_T_JOURNAL // No PSRAM

#include "camera_pins.h"

const char* ssid = "*****";
const char* password = "*****";
```



```
CameraWebServer | 아두이노 1.8.13
파일 편집 스케치 툴 도움말

CameraWebServer app_httpd.cpp camera_index.h camera_pins.h

#include "esp_camera.h"
#include <WiFi.h>

//
// WARNING!!! PSRAM IC required for UXGA resolution and high JPEG quality
//           Ensure ESP32 Wrover Module or other board with PSRAM is selected
//           Partial images will be transmitted if image exceeds buffer size
//

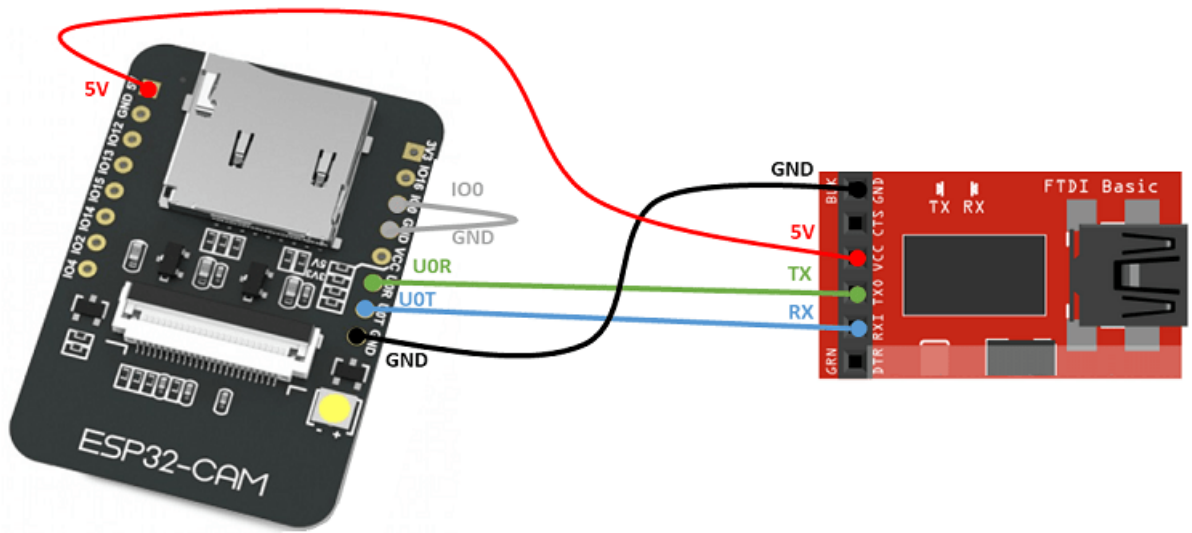
// Select camera model
#define CAMERA_MODEL_WROVER_KIT // Has PSRAM
// #define CAMERA_MODEL_ESP_EYE // Has PSRAM
// #define CAMERA_MODEL_M5STACK_PSRAM // Has PSRAM
// #define CAMERA_MODEL_M5STACK_V2_PSRAM // M5Camera version B Has PSRAM
// #define CAMERA_MODEL_M5STACK_WIDE // Has PSRAM
// #define CAMERA_MODEL_M5STACK_ESP32CAM // No PSRAM
// #define CAMERA_MODEL_AI_THINKER // Has PSRAM
// #define CAMERA_MODEL_TTGO_T_JOURNAL // No PSRAM

#include "camera_pins.h"

const char* ssid = "*****";
const char* password = "*****";
```

ESP32 CAM Upload Code

Connect the ESP32-CAM board to your computer using an FTDI programmer. Follow the next schematic diagram:



```
COM10
ets Jun  8 2016 00:22:57

rst:0x1 (POWERON_RESET),boot:0x13 (SPI_FAST_FLASH_BOOT)
configsip: 0, SPIWP:0xee
clk_drv:0x00,q_drv:0x00,d_drv:0x00,cs0_drv:0x00,hd_drv:0x00,wp_drv:0x00
mode:DIO, clock div:1
load:0x3fff0018,len:4
load:0x3fff001c,len:1216
ho 0 tail 12 room 4
load:0x40078000,len:10944
load:0x40080400,len:6388
entry 0x400806b4
$
.....
WiFi connected
Starting web server on port: '80'
Starting stream server on port: '81'
Camera Ready! Use 'http://192.168.0.2' to connect
```

ESP32 OV2640

Toggle OV2640 settings

Resolution: UXGA(1600x1200)

Quality: 10

Brightness: -2

Contrast: -2

Saturation: -2

Special Effect: No Effect

AWB: ON

AWB Gain: ON

WB Mode: Auto

AEC SENSOR: ON

AEC DSP: ON

AE Level: -2

AGC: ON

Gain Ceiling: 2x

BPC: ON

WPC: ON

Raw GMA: ON

Lens Correction: ON

H-Mirror: ON

V-Flip: ON

DCW (Downsize EN): ON

Color Bar: ON

Face Detection: ON

Face Recognition: ON

Get Still

Stop Stream

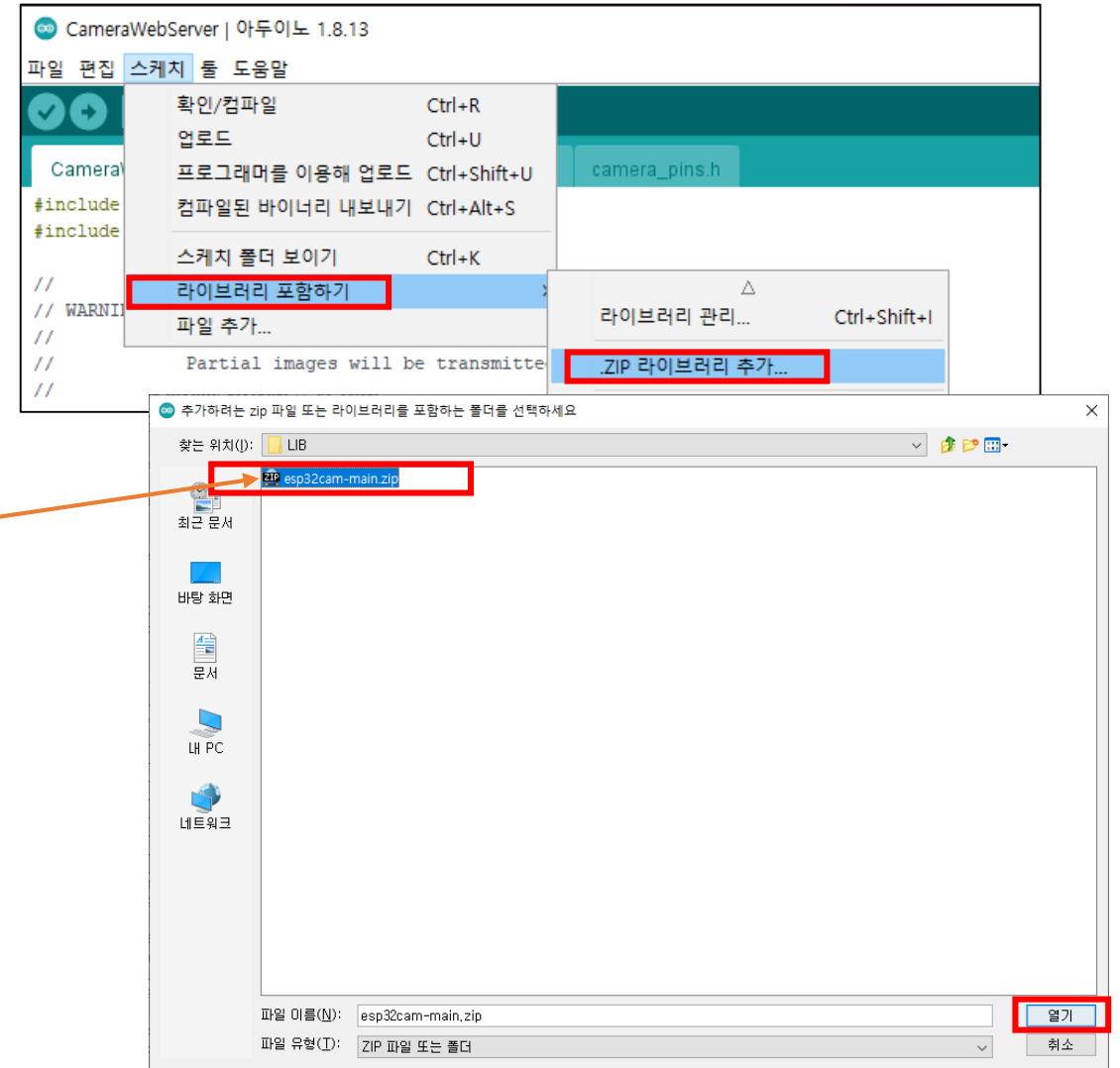
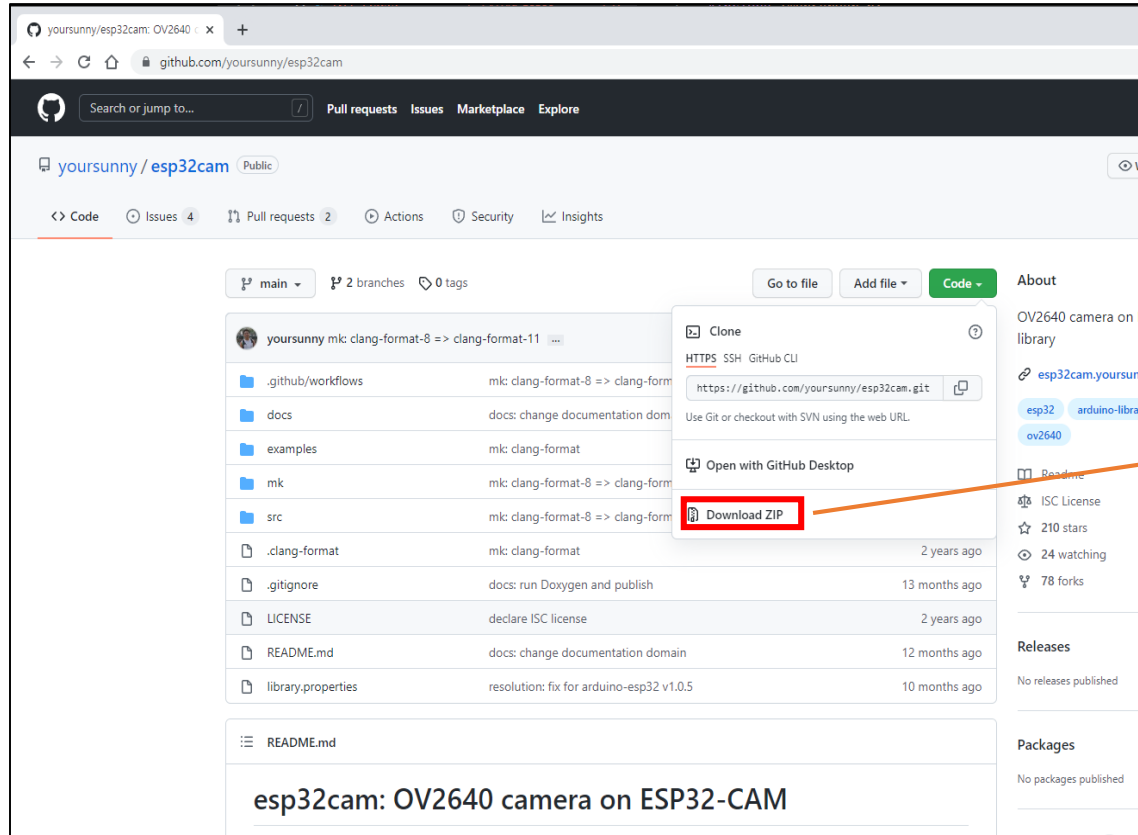
Enroll Face

ESP32-CAM	FTDI Programmer
GND	GND
5V	VCC (5V)
U0R(RX)	TX
U0T(TX)	RX
GPIO 0 == GND	

Upload 할 때만 연결

Esp32cam LIB 받기

<https://github.com/yoursunny/esp32cam>



1. YOLO(You Only Look Once)와 호환되는 가장 많이 사용되는 프레임워크

- 1) **Darknet**: YOLO를 위해 특별히 제작된 YOLO 개발자가 만든 프레임워크

장점: 빠르고 GPU또는 CPU와 함께 사용가능

단점: 리눅스에서 지원

- 2) **Darkflow**: Darknet을 텐서플로우에 적용

장점: 빠르고 GPU 또는 CPU와 함께 사용 가능하고 리눅스, 윈도우, 맥에서 호환

단점: 설치 복잡

- 3) **OpenCV**: 최소 3.4.2버전 필요

장점: 설치가 간단하다(openCV외에 설치할 것이 없다)

단점: CPU에서만 작동하기 때문에 비디오를 실시간으로 처리하는 데 속도가 느리다.

2. YOLO 주요 파일

1) Weight file

- 훈련된 model의 값

2) Cfg file :

- 환경설정 파일
- 알고리즘에 관한 모든 설정이 있다.

3) Name files :

- 알고리즘이 감지할 수 있는 객체의 이름들

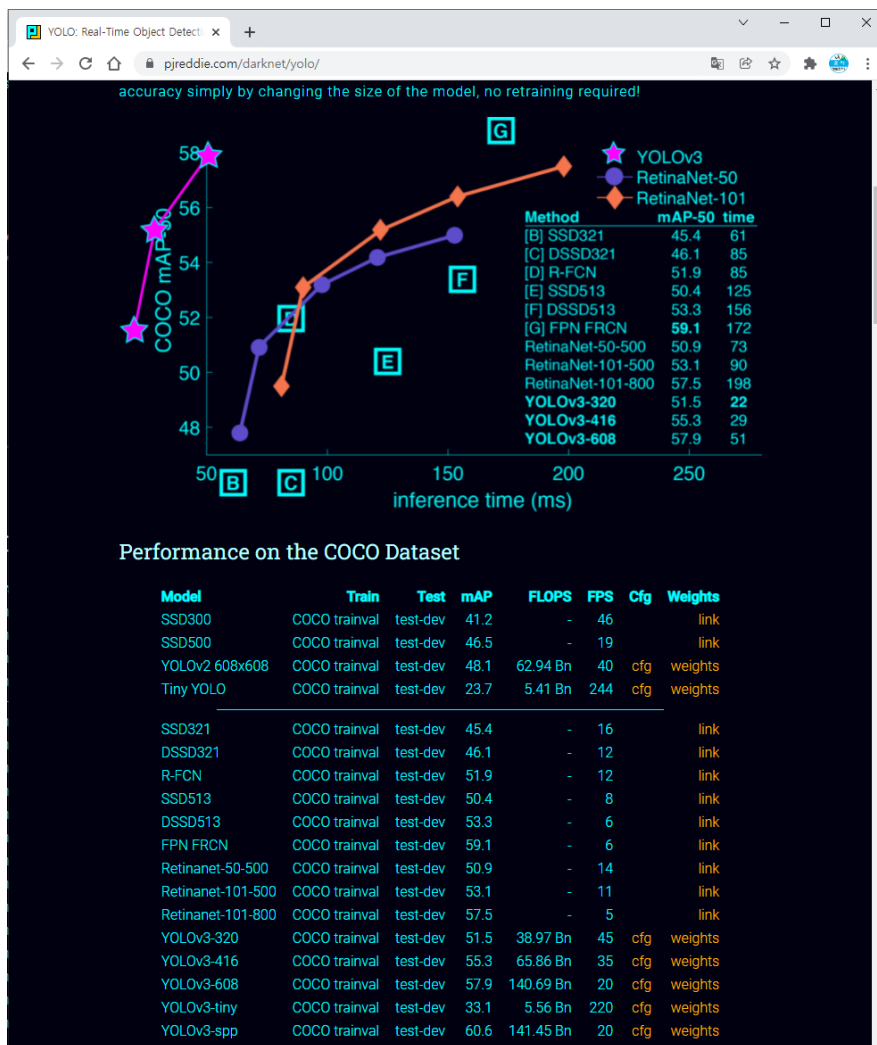
3. YOLO가 허용하는 세가지 크기

- 1) 320×320 : 정확도는 떨어지지 만 속도 빠름
- 2) 416×416 : 중간
- 3) 609×609 : 정확도는 더 높지만 속도 느림

YOLO3 download

Weights data는 학습이 필요함.

<https://pjreddie.com/darknet/yolo/>



<https://github.com/JhoelRN/ESP32-CAM-wireless-computer-vision-objects-detection>

JhoelRN / ESP32-CAM-wireless-computer-vision-objects-detection

Public

Code Issues Pull requests Actions Projects Wiki Security Insights

main

Go to file Add file Code

About

ESP32 CAM wireless computer vision objects detection.

Readme

5 stars

2 watching

2 forks

Releases

No releases published

Packages

No packages published

Languages

C++ 57.4% Python 42.6%

README.md

ESP32-CAM-wireless-computer-vision-objects-detection

ESP32 CAM wireless computer vision objects detection.

DETECCION DE OBJETOS CON ESP32 CAM | VISION ARTIFICIAL PYTHON + OpenCV + YOLOv3 (TIEMPO REAL)

Se realiza con "coco" que se encuentra en el mismo directorio. En la parte de python se hace referencia en las lineas 13 al 26: donde se realiza la configuración y pesos de YOLOv3 con la ayuda del modulo "dnn" de openCV. El archivo coco.names contiene los nombres de distintos objetos que se han entrenado para detección de objetos... Luego se almacena en "classNames", y así como "net" se basa en usar librerías para para capas de calculo de salida, cargar y procesar qyue ya fueron implementados y más información se encuentra en Internet.

어느 정도 학습이 된 weights와 테스트 소스