Lesson 3 Mask Recognition

Note: Please refer to "Lesson 1 Image Classification Learning" to burn mask recognition model into WonderCam AI vision module.

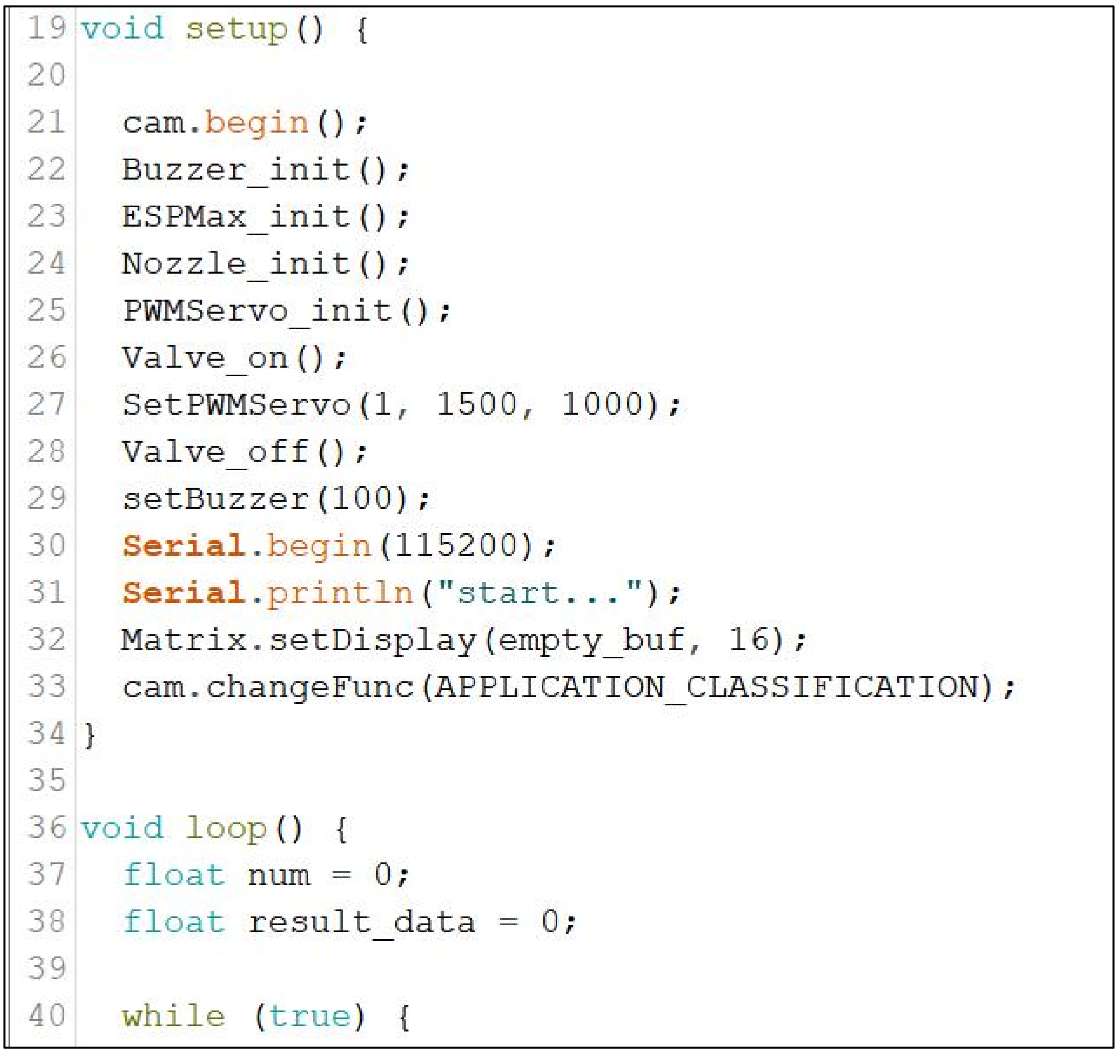
If do not finish this step, please burn the models first according to Lesson 1, otherwise, you will fail to start this game.

1. Project Principle

When WonderCam recognizes a human wearing a mask, dot matrix module will display a smiley face, and when it recognizes a face without a mask, the dot matrix module displays a cross mark.

The function of image classification is used to identify the ID number with the highest confidence, and then judge whether it is a human face and whether the face is wearing a mask, and finally combine with the dot matrix module to set the prompt feedback.

The path of program: "7. AI Vision Game/ Arduino Development/ Program Files/ Mask Recognition /Masks_identify/Masks_identify.ino".



```
void setup() {

  cam.begin();
  Buzzer_init();
  ESPMax_init();
  Nozzle_init();
  PWMServo_init();
  Valve_on();
  SetPWMServo(1, 1500, 1000);
  Valve_off();
  setBuzzer(100);
  Serial.begin(115200);
  Serial.println("start...");
  Matrix.setDisplay(empty_buf, 16);
  cam.changeFunc(APPLICATION_CLASSIFICATION);
}

void loop() {
  float num = 0;
  float result_data = 0;

  while (true) {
```

2. Preparation

2.1 Hardware

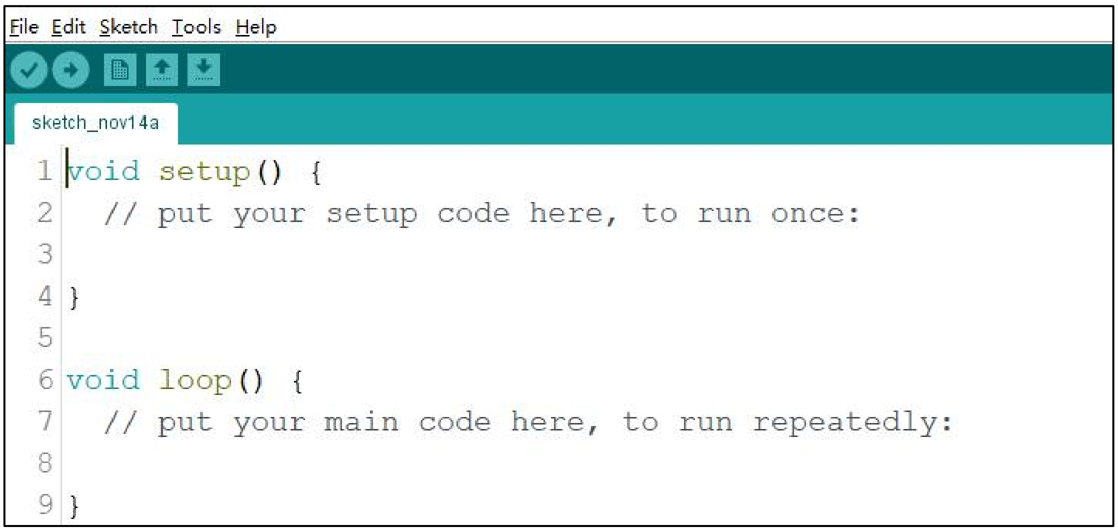
Please refer to “Lesson 1 Module Assembly” under the same directory to assemble module to the corresponding position on MaxArm

2.2 Software

Please refer to “4. Underlying Program Learning/ Arduino Development/ Lesson 1 Set Development Environment” to connect MaxArm to Arduino.

3. Program Download

1) Double click to open Arduino IDE .



File Edit Sketch Tools Help

sketch_nov14a

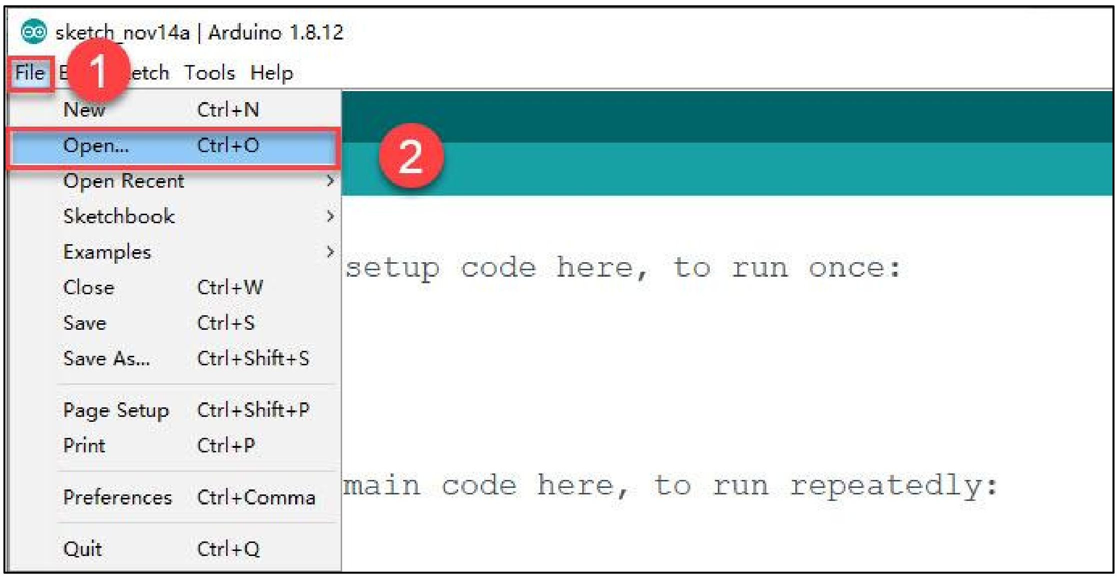
```
void setup() {
  // put your setup code here, to run once:

}

void loop() {
  // put your main code here, to run repeatedly:

}
```

2) Click “File -- Open”.

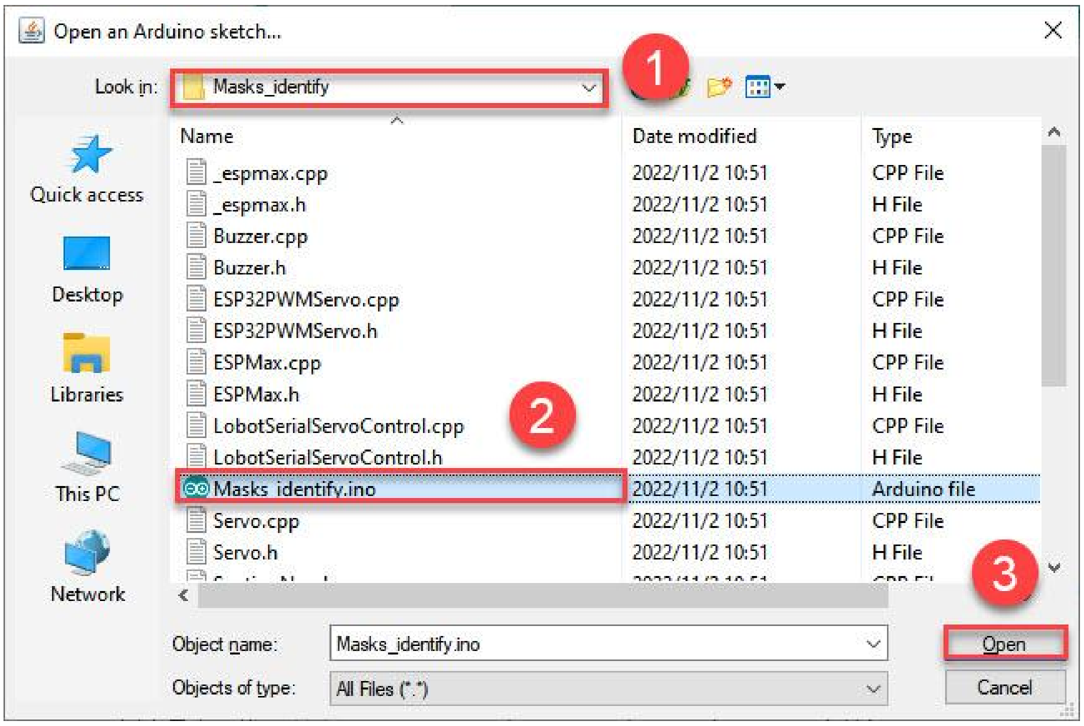


sketch_nov14a | Arduino 1.8.12

```
setup code here, to run once:

main code here, to run repeatedly:
```

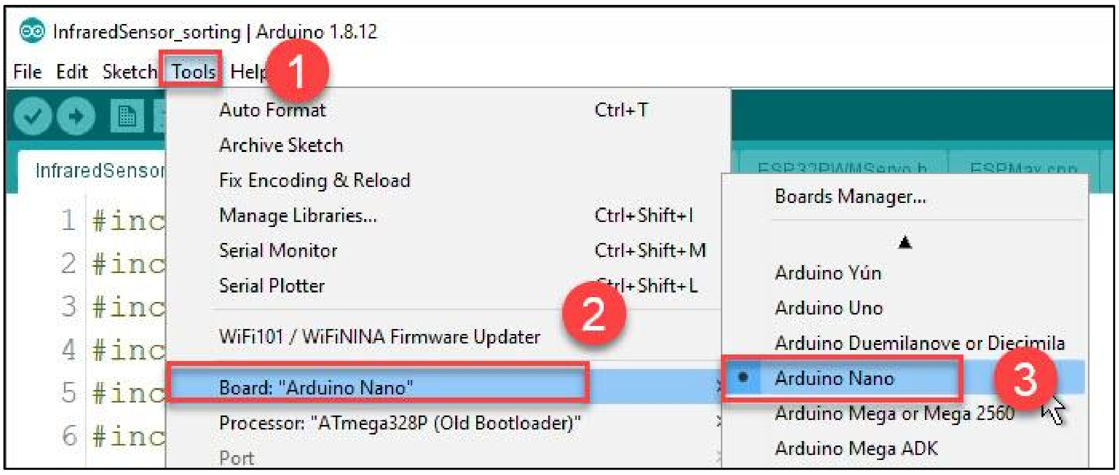
3) Open the program “Masks_identify.ino” in the folder “7. AI Vision Games/ Arduino Development/ Program Files/ Mask Recognition/Masks_identify”.



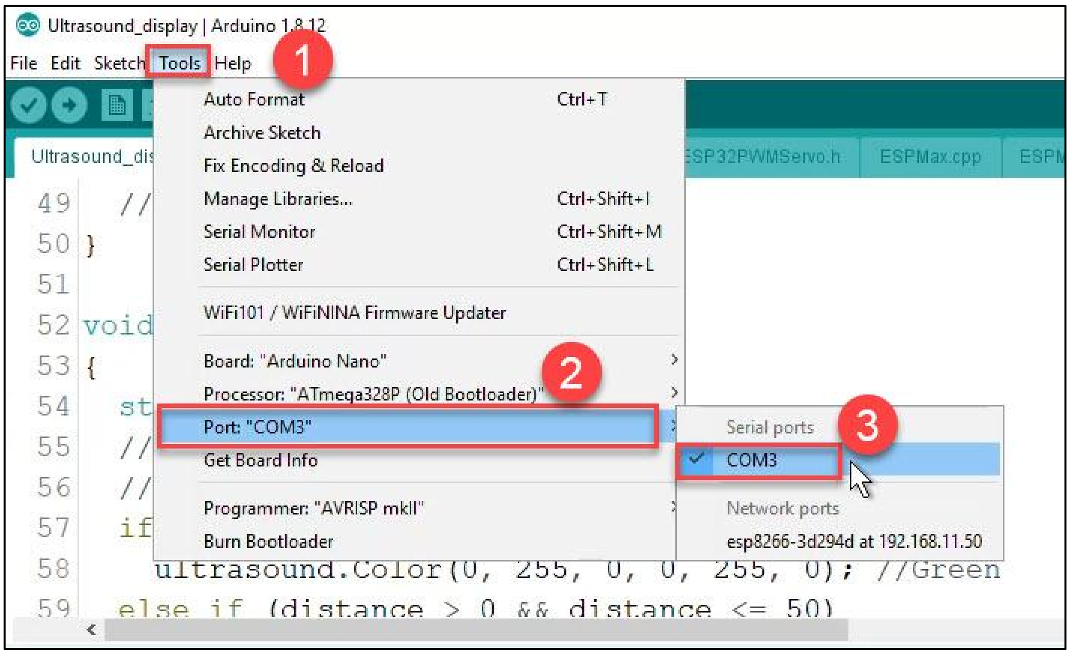
Name	Date modified	Type
_espmax.cpp	2022/11/2 10:51	CPP File
_espmax.h	2022/11/2 10:51	H File
Buzzer.cpp	2022/11/2 10:51	CPP File
Buzzer.h	2022/11/2 10:51	H File
ESP32PWMServo.cpp	2022/11/2 10:51	CPP File
ESP32PWMServo.h	2022/11/2 10:51	H File
ESPMax.cpp	2022/11/2 10:51	CPP File
ESPMax.h	2022/11/2 10:51	H File
LobotSerialServoControl.cpp	2022/11/2 10:51	CPP File
LobotSerialServoControl.h	2022/11/2 10:51	H File
Masks_identify.ino	2022/11/2 10:51	Arduino file
Servo.cpp	2022/11/2 10:51	CPP File
Servo.h	2022/11/2 10:51	H File

Object name: Masks_identify.ino
Objects of type: All Files (*.*)

4) Select the corresponding board model. Click “Tool -- Board” and select “ESP 32 Dev Module” (If the board model has been configured when setting development environment, you can skip this step.)

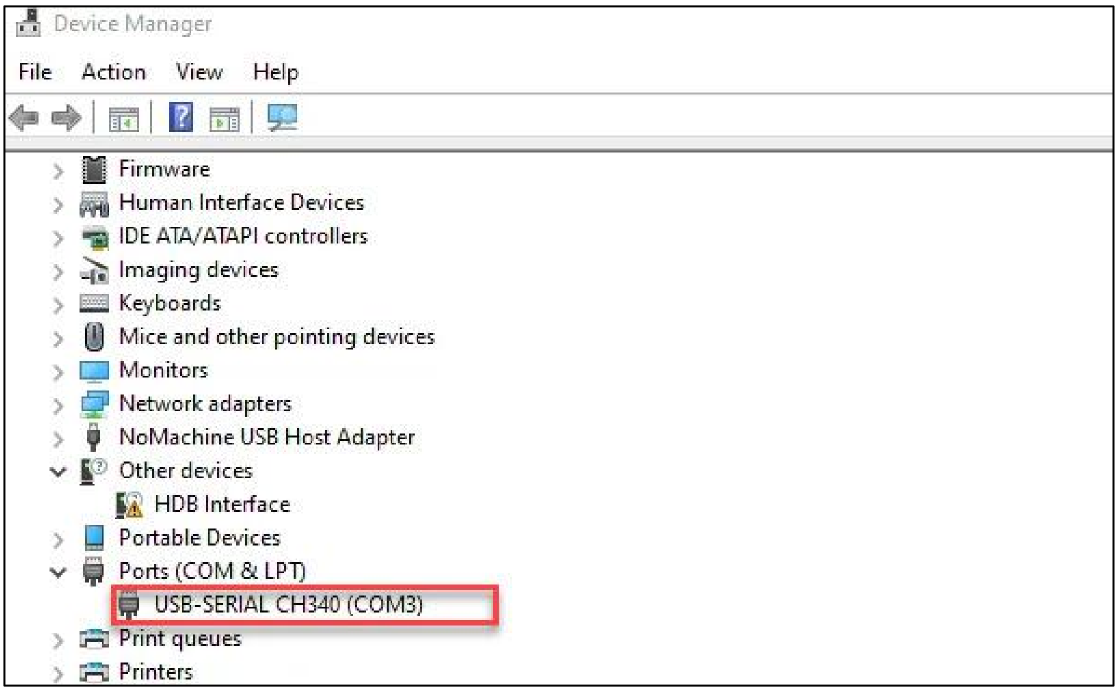

InfraredSensor_sorting | Arduino 1.8.12
File Edit Sketch Tools Help
1
Auto Format Ctrl+T
Archive Sketch
Fix Encoding & Reload
Manage Libraries... Ctrl+Shift+I
Serial Monitor Ctrl+Shift+M
Serial Plotter Ctrl+Shift+L
WiFi101 / WiFiNINA Firmware Updater
2
Board: "Arduino Nano"
Processor: "ATmega328P (Old Bootloader)"
Port
Boards Manager...
Arduino Yún
Arduino Uno
Arduino Duemilanove or Diecimila
Arduino Nano
3
Arduino Mega or Mega 2560
Arduino Mega ADK

5) Select the corresponding port in “Tools->Port”. (Here take the port “COM5” as example. Please select the port based on your computer. If COM1 appears, please do not select because it is the system communication port but not the actual port of the development port.)

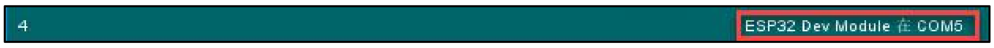

Ultrasound_display | Arduino 1.8.12
File Edit Sketch Tools Help
1
Auto Format Ctrl+T
Archive Sketch
Fix Encoding & Reload
Manage Libraries... Ctrl+Shift+I
Serial Monitor Ctrl+Shift+M
Serial Plotter Ctrl+Shift+L
WiFi101 / WiFiNINA Firmware Updater
Board: "Arduino Nano"
2
Processor: "ATmega328P (Old Bootloader)"
Port: "COM3"
Get Board Info
Programmer: "AVRISP mkII"
Burn Bootloader
Serial ports
3
COM3
Network ports
esp8266-3d294d at 192.168.11.50
ESP32PWMServo.h
ESPMax.cpp

```
ultrasound.Color(0, 255, 0, 0, 255, 0); //Green
else if (distance > 0 && distance <= 50)
```

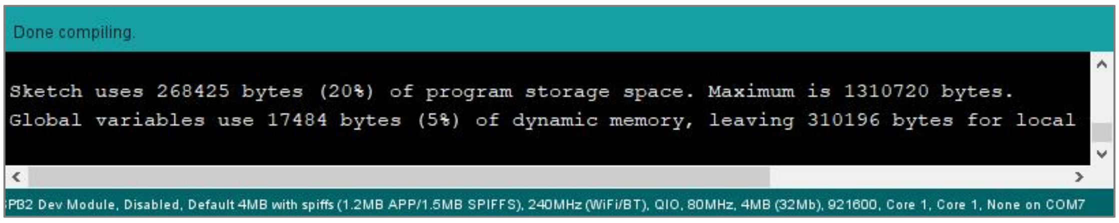
6) If you’re not sure about the port number, please open the “This PC” and click “Properties->Device Manger” in turns to check the corresponding port number (the device is with CH340).



7) After selecting, confirm the board “ESP32 Dev Module” in the lower right corner and the port number “COM5” (it is an example here, please refer to the actual situation).



8) Then click on icon to verify the program. If no error, the status area will display "Compiling->Compile complete" in turn. After compiling, the information such as the current used bytes, and occupied program storage space will be displayed.

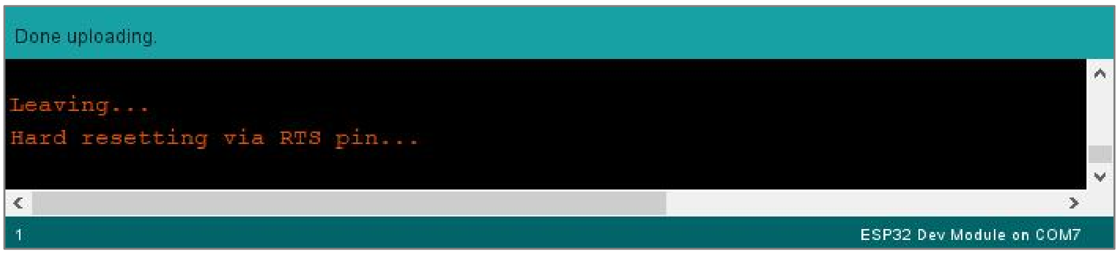


```
Done compiling.

Sketch uses 268425 bytes (20%) of program storage space. Maximum is 1310720 bytes.
Global variables use 17484 bytes (5%) of dynamic memory, leaving 310196 bytes for local
```

PB2 Dev Module, Disabled, Default 4MB with spiffs (1.2MB APP/1.5MB SPIFFS), 240MHz (WiFi/BT), QIO, 80MHz, 4MB (32Mb), 921600, Core 1, Core 1, None on COM7

9) After compiling, click on icon to upload the program to the development board. The status area will display "Compiling->Uploading->Complete" in turn. After uploading, the status area will stop printing the uploading information.



```
Done uploading.

Leaving...
Hard resetting via RTS pin...
```

4. Project Outcome

After the program is downloaded completely, turn on robot and LED dot matrix will display a smiley face.

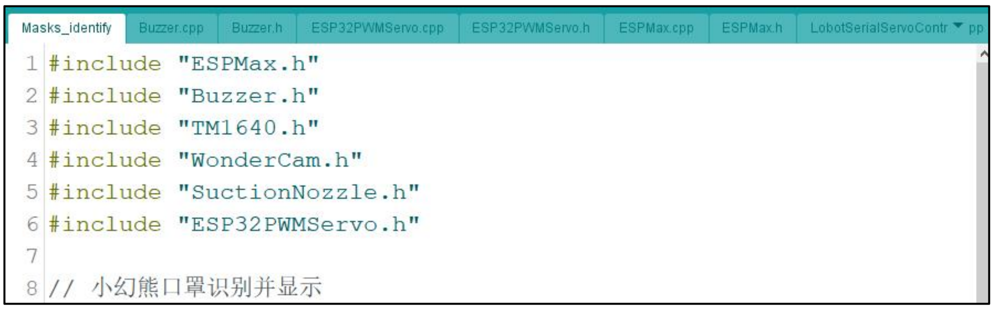
When detecting a face wearing no mask, robot will rotate left and right, all RGB lights will light up red and buzzer will alarm. After wearing a mask, all RGB lights will light up green.

5. Program Analysis

The path of program: “7. AI Vision Game/ Arduino Development/ Program Files/ Mask Recognition /Masks_identify/Masks_identify.ino”. You can find the backup program file in Appendix if the original program file is modified.

5.1 Import function library and Initialization

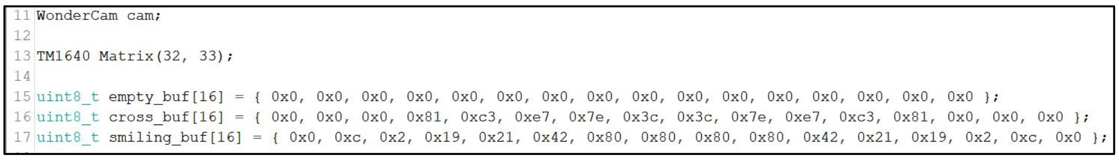
Firstly, import the kinematics package library, buzzer library, dot matrix driver library, WonderCam driver library, suction nozzle library, PWM servo library and other related function library files for mask recognition.



```
#include "ESPMax.h"
#include "Buzzer.h"
#include "TM1640.h"
#include "WonderCam.h"
#include "SuctionNozzle.h"
#include "ESP32PWMServo.h"

// 小幻熊口罩识别并显示
```

Next, instantiate WonderCam module and dot matrix driver library file.



```
WonderCam cam;

TM1640 Matrix(32, 33);

uint8_t empty_buf[16] = { 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0 };
uint8_t cross_buf[16] = { 0x0, 0x0, 0x0, 0x81, 0xc3, 0xe7, 0x7e, 0x3c, 0x3c, 0x7e, 0xe7, 0xc3, 0x81, 0x0, 0x0, 0x0 };
uint8_t smiling_buf[16] = { 0x0, 0xc, 0x2, 0x19, 0x21, 0x42, 0x80, 0x80, 0x80, 0x80, 0x42, 0x21, 0x19, 0x2, 0xc, 0x0 };
```

Then, initialize robotic arm.

```
void setup() {

  cam.begin();
  Buzzer_init();
  ESPMax_init();
  Nozzle_init();
  PWMServo_init();
  Valve_on();
  SetPWMServo(1, 1500, 1000);
  Valve_off();
  setBuzzer(100);
  Serial.begin(115200);
  Serial.println("start...");
  Matrix.setDisplay(empty_buf, 16);
  cam.changeFunc(APPLICATION_CLASSIFICATION);
```

5.2 Image classification Recognition

Set the accumulation amount (num) to store the number of tests and the result cache amount (result_data) to store the test results.

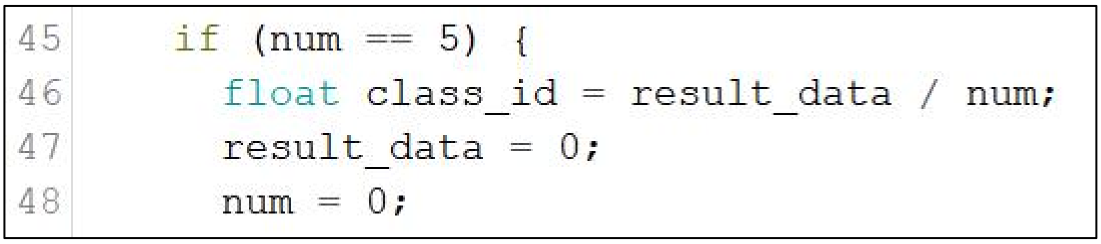
```
void loop() {
  float num = 0;
  float result_data = 0;

```

For each detection, num+1, call cam.updateResult() function to update the detection result, call cam.classIdOfMaxProb() function to get the maximum ID detected and stored in result_data (the ID for mask wearing is 2.0 and for no mask is 3.0).

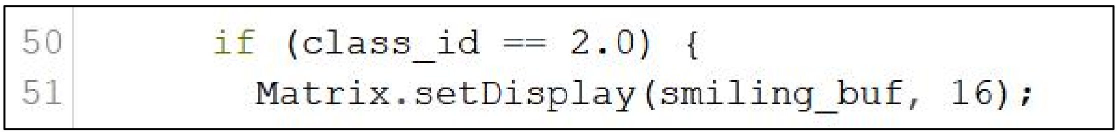
```
  while (true) {
    num += 1;
    cam.updateResult();
    result_data += cam.classIdOfMaxProb();
```

In order to prevent wrong detection, we take the average value after 5 times of detection and assign it to class_id to determine whether the person detected is wearing a mask or not, at this time the value of class_id is (2.0, 3.0, not equal to 2.0 or 3.0).



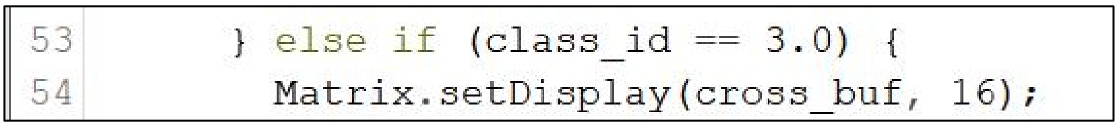
```
    if (num == 5) {
      float class_id = result_data / num;
      result_data = 0;
      num = 0;
```

If class_id == 2.0, the face detected wears a mask. Then call the setDisplay() function in the TM1640.h library file to display a smiley face on dot matrix module .



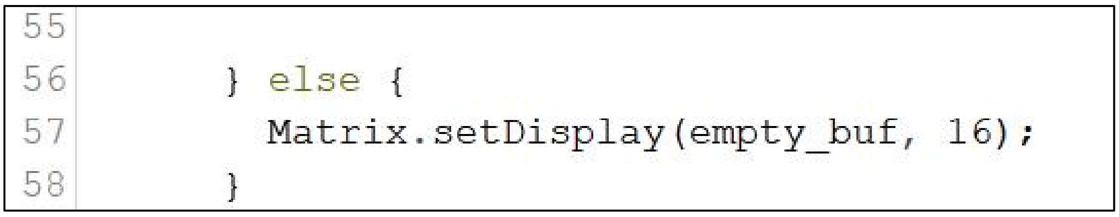
```
      if (class_id == 2.0) {
        Matrix.setDisplay(smiling_buf, 16);
```

If class_id == 3.0, it means the person detected does not put on a mask. At this time, dot matrix module display ✕ .



```
      } else if (class_id == 3.0) {
        Matrix.setDisplay(cross_buf, 16);
```

Otherwise, it is assumed that the background is detected and the dot module is cleared.



```

      } else {
        Matrix.setDisplay(empty_buf, 16);
      }
```

6. Function Extension

In the program, three display states of the LED dot matrix module are set, which are off, smiley, and cross. If want to change the display pattern, you can refer to the following operations:

The LED dot matrix module has 8 rows and 16 columns, each column represents an 8-bit binary number, when it is 0, the LED is off, when it is 1, the LED is on. To facilitate writing, we will convert it into hexadecimal numbers to record "cross_buf[16] = { 0x0, 0x0, 0x0, 0x81, 0xc3, 0xe7, 0x7e, 0x3c, 0x3c, 0x7e, 0xe7, 0xe7, 0xc3, 0x81, 0x0, 0x0, 0x0, 0x0 };". Take displaying "✕" for example, as shown in the following table, coded from the bottom up.

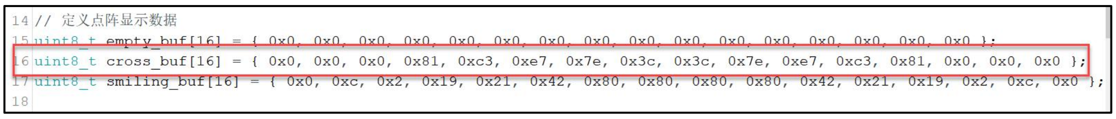
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
0x0	0x0	0x0	0x81	0xc3	0xe7	0x7e	0x3c	0x3c	0x7e	0xe7	0xc3	0x81	0x0	0x0	0x0
0	0	0	1	1	1	0	0	0	0	1	1	1	0	0	0
0	0	0	0	1	1	1	0	0	1	1	1	0	0	0	0
0	0	0	0	0	1	1	1	1	1	1	0	0	0	0	0
0	0	0	0	0	0	1	1	1	1	0	0	0	0	0	0
0	0	0	0	0	0	1	1	1	1	0	0	0	0	0	0
0	0	0	0	0	1	1	1	1	1	1	0	0	0	0	0
0	0	0	0	1	1	1	0	0	1	1	1	0	0	0	0
0	0	0	1	1	1	0	0	0	0	1	1	1	0	0	0

Here is an example of changing the ✕ displayed by cross_buf to a sad expression. The specific steps are shown below.

1) First calculate the hexadecimal numbers of the sad expressions, which are { 0x0, 0x0, 0x2, 0x2, 0x22, 0x12, 0x8, 0x8, 0x8, 0x8, 0x12, 0x22, 0x2, 0x2, 0x0, 0x0 } as shown in the table below, coded from the bottom up.

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
0x0	0x0	0x2	0x2	0x22	0x12	0x8	0x8	0x8	0x8	0x12	0x22	0x2	0x2	0x0	0x0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	1	1	1	1	0	0	0	0	1	1	1	1	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	1	1	1	1	0	0	0	0	0	0
0	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0
0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

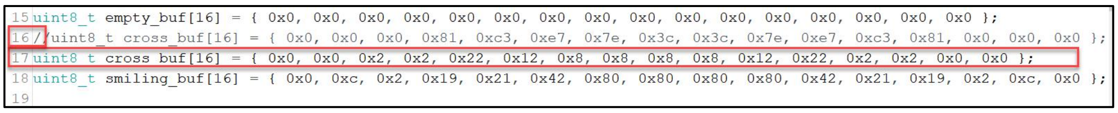
2) Find the following code.



```
// 定义点阵显示数据
uint8_t empty_buf[16] = { 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0 };
uint8_t cross_buf[16] = { 0x0, 0x0, 0x0, 0x81, 0xc3, 0xe7, 0x7e, 0x3c, 0x3c, 0x7e, 0xe7, 0xc3, 0x81, 0x0, 0x0, 0x0 };
uint8_t smiling_buf[16] = { 0x0, 0xc, 0x2, 0x19, 0x21, 0x42, 0x80, 0x80, 0x80, 0x80, 0x42, 0x21, 0x19, 0x2, 0xc, 0x0 };

```

3) Use “//” to comment it out and input “uint8_t cross_buf[16] = { 0x0, 0x0, 0x2, 0x2, 0x22, 0x12, 0x8, 0x8, 0x8, 0x8, 0x12, 0x22, 0x2, 0x2, 0x0, 0x0 };” in the next line.



```
uint8_t empty_buf[16] = { 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0 };
//uint8_t cross_buf[16] = { 0x0, 0x0, 0x0, 0x81, 0xc3, 0xe7, 0x7e, 0x3c, 0x3c, 0x7e, 0xe7, 0xc3, 0x81, 0x0, 0x0, 0x0 };
uint8_t cross_buf[16] = { 0x0, 0x0, 0x2, 0x2, 0x22, 0x12, 0x8, 0x8, 0x8, 0x8, 0x12, 0x22, 0x2, 0x2, 0x0, 0x0 };
uint8_t smiling_buf[16] = { 0x0, 0xc, 0x2, 0x19, 0x21, 0x42, 0x80, 0x80, 0x80, 0x80, 0x42, 0x21, 0x19, 0x2, 0xc, 0x0 };
```

4) After inputting, press “Ctrl+S” to save program and refer to “3. Program Download” to compile and upload. After the game starts, when a human face wearing no mask is detected, the module will display a sad expression.