

## Lesson 3 AI Fan

---

**NOTE:** If the face models have been burnt to WonderCam in advance, you can start this game directly.

If fail to recognize face, please re-burn the firmware according to “4. Firmware (including method)” in “Appendix/ WonderCam AI Vision Module /3.Appendix /4.Firmware (Installation method included)”.

---

### 1. Project Principle

In this game, when face is detected by WonderCam module, fan module will turn on and dot matrix module will display a smiley expression, and then robotic arm will track the face,

The tracking function is implemented by PID algorithm which controls robotic arm to move within a certain area.

PID control algorithm combines Proportional (P), Integral (I), and Derivative (D). The controller attempts to correct the error between a measured process variable and desired setpoint by calculating the difference.

Based on the position information obtained in face recognition, and the coordinates of the center point of the vision module are compared to calculate the distance between robot arm and target position.

The path of program file is “7. AI Vision Game/ Arduino Development/ Program Files/ AI Fan/Face\_tracking\_fan/ Face\_tracking\_fan.ino”

```
39 void loop() {
40     float pos[3];
41     pos[0] = 0;
42     pos[1] = -120;
43     pos[2] = 150;
44     set_position(pos, 1500);
45     delay(1500);
46     bool display_st = false;
47
48     while (true) {
49         cam.updateResult();
50         if (cam.anyFaceDetected()) {
51             if (!display_st) {
52                 display_st = true;
53                 FanModule_on();
54                 Matrix.setDisplay(smiling_buf, 16);
55             }
56             WonderCamFaceDetectResult p;
57             cam.getFaceOfIndex(1, &p);
```

## 2. Preparation

### 2.1 Hardware

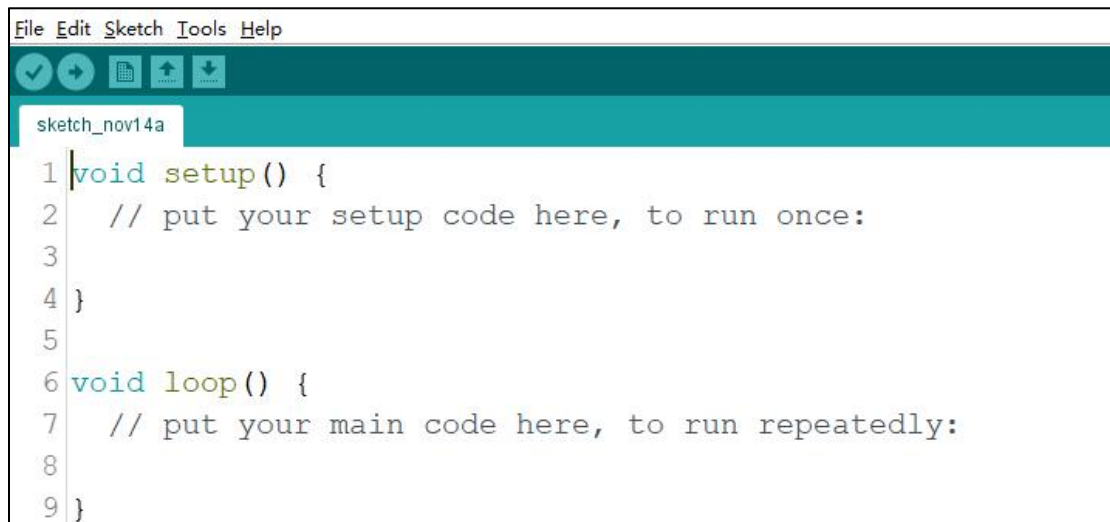
Please refer to “Lesson 2 Module Assembly” to assemble fan module to the corresponding position on MaxArm (**WonderCam AI vision module has been assembled by default**).

### 2.2 Software

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

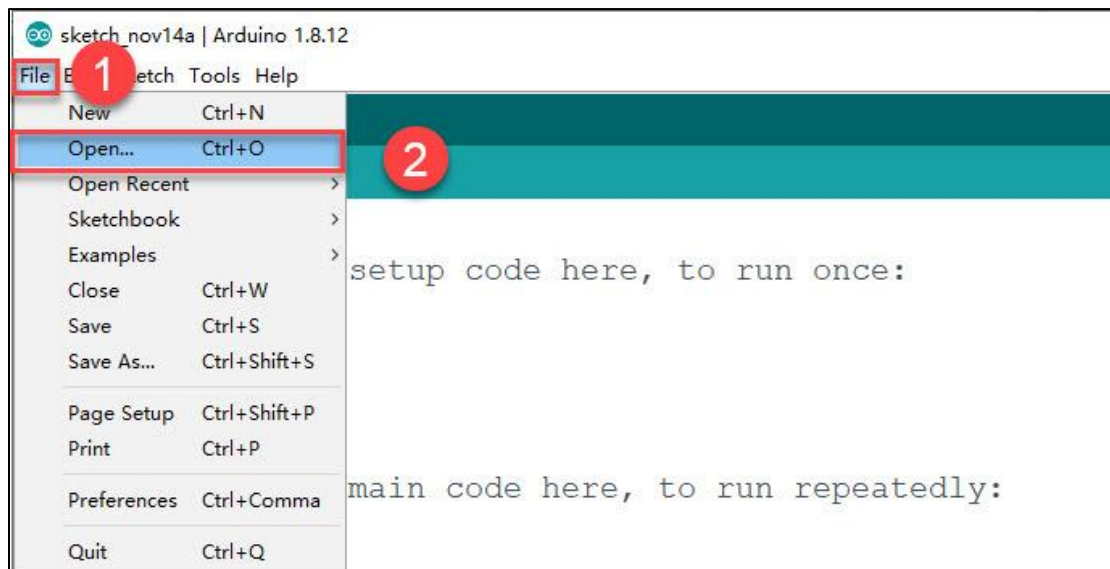
## 3. Program Download

- 1) Double click to open Arduino IDE .

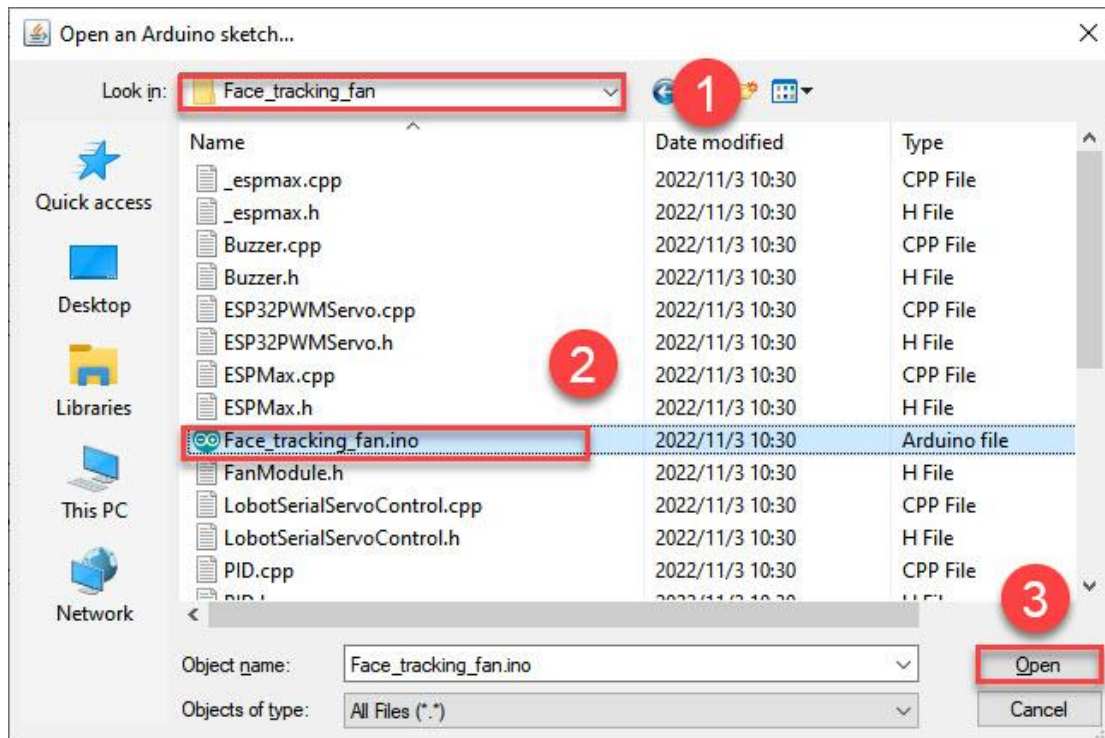


```
File Edit Sketch Tools Help
sketch_nov14a
1 void setup() {
2   // put your setup code here, to run once:
3
4 }
5
6 void loop() {
7   // put your main code here, to run repeatedly:
8
9 }
```

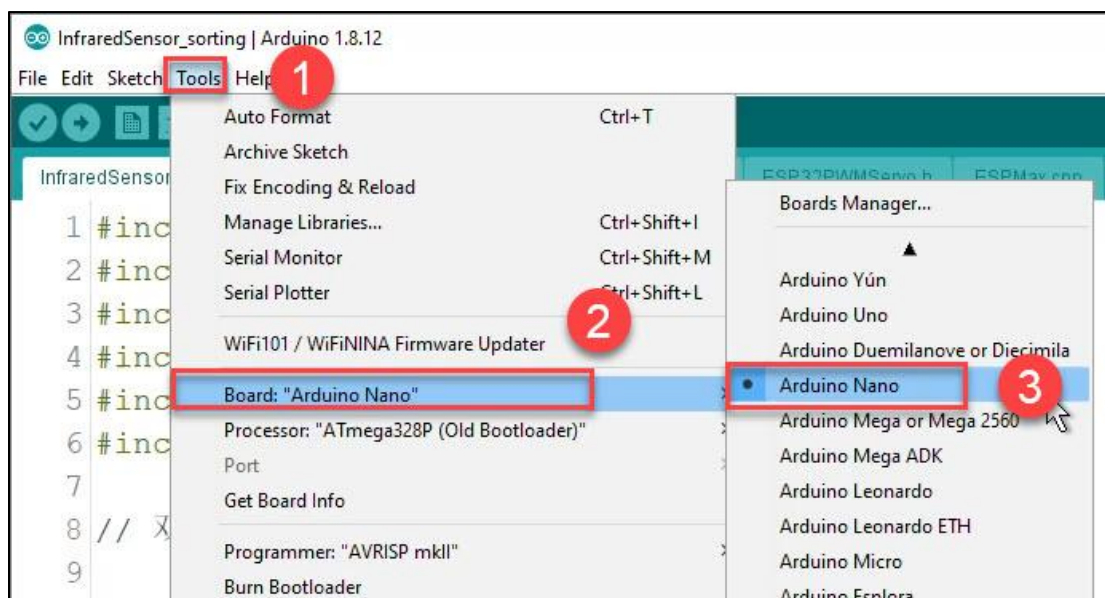
2) Click “File -- Open”.



3) Open the program “Face\_tracking\_fan.ino” in the folder “7. AI Vision Games/ Arduino Development/ Program Files/ AI Fan/ Face\_tracking\_fan”.

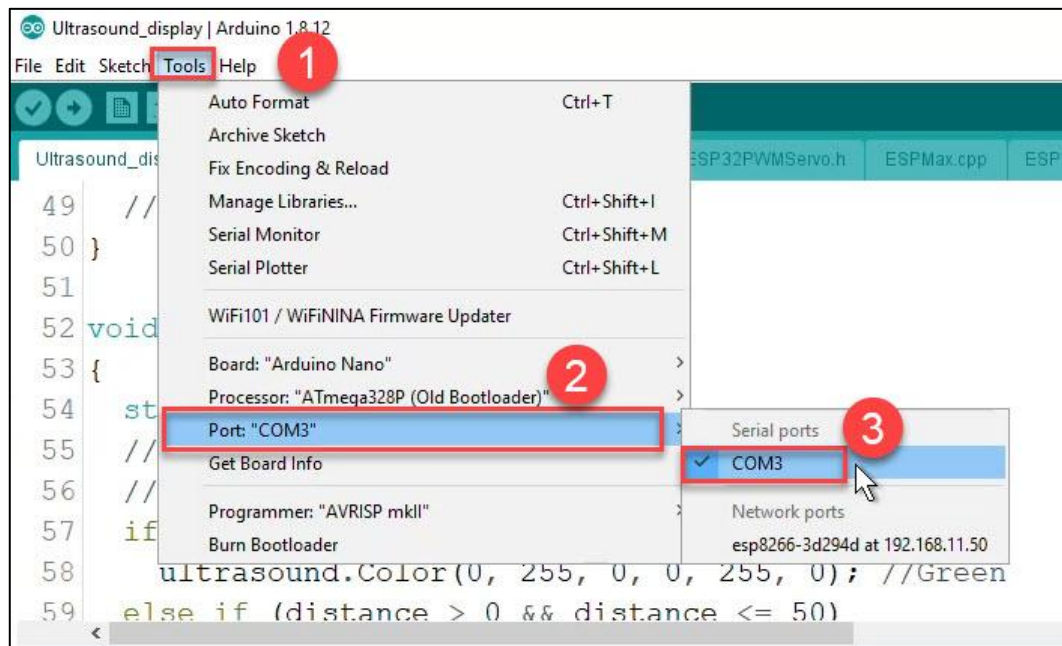


- 4) Select the corresponding board model. Click “Tool -- Board” and select “ESP 32 Dev Module” (If the board model has been configured when setting development environment, you can skip this step.)

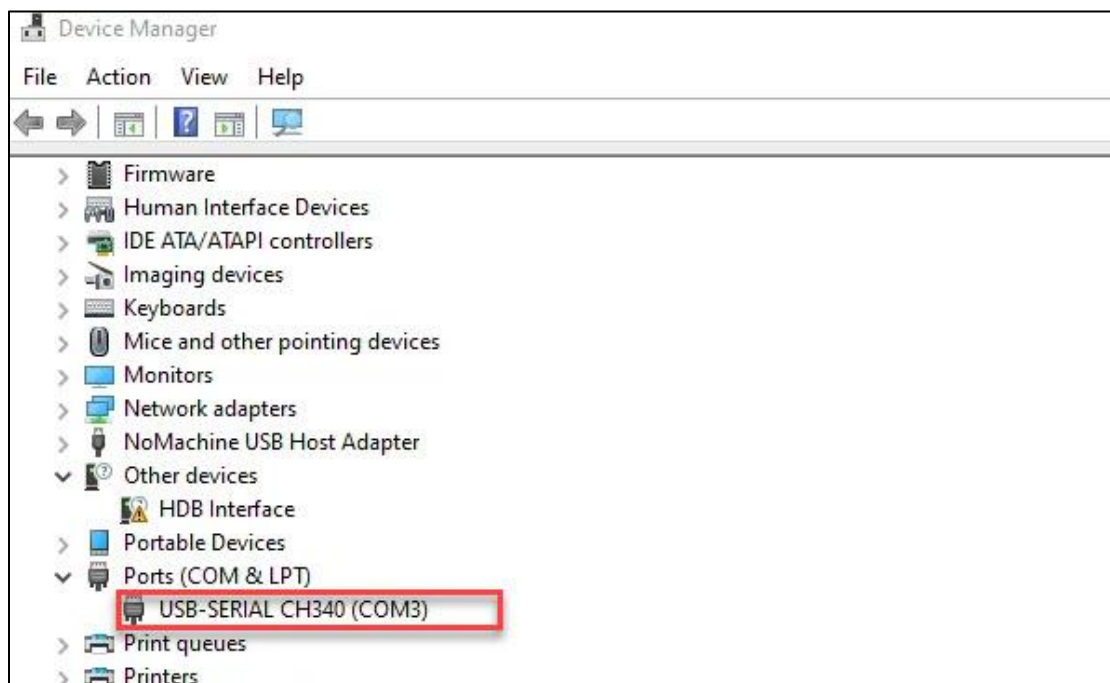


- 5) Select the corresponding port in “Tools->Port”. (Here take the port “COM5” as example. Please select the port based on your computer. If COM1 appears, please do not select because it is the system communication port

but not the actual port of the development port.)




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

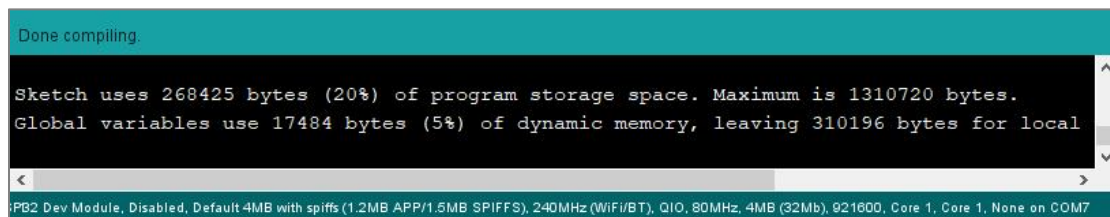



- 7) After selecting, confirm the board "ESP32 Dev Module" in the lower right corner and the port number "COM5" (it is an example here, please refer to

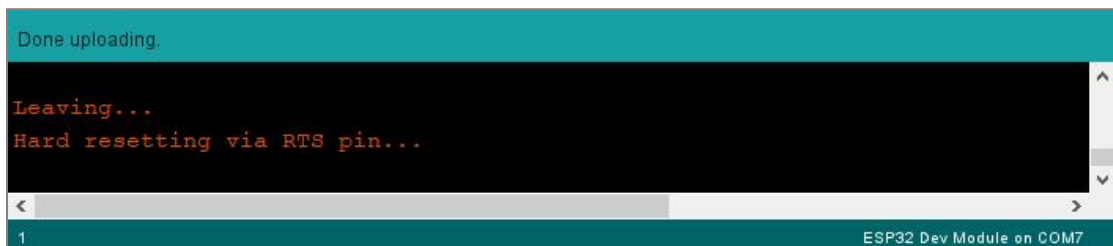
the actual situation).



- 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.



- 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.



## 4. Project Outcome

After the game is started, MaxArm keeps searching human face within a certain range. When a human face is detected, fan will turn on and a smile expression will display on dot matrix module. If the human face disappears, the fan will stop and dot matrix module will go black.



## 5. Program Analysis

### 5.1 Import Function Library

The path of the program file: 7. AI Vision Game/ Arduino Development/ Program Files/ AI Fan/ Face\_tracking\_fan/ Face\_tracking\_fan.ino

Before the program is executed, I2C protocol, recognition module, buzzer, inverse kinematics, PWM servo, bus servo, air pump and other related function libraries are required to be imported first.

Face_tracking_fan	Buzzer.cpp	Buzzer.h	ESP32PWMServo.cpp
1	#include	"PID.h"	
2	#include	"ESPMax.h"	
3	#include	"Buzzer.h"	
4	#include	"TM1640.h"	
5	#include	"WonderCam.h"	
6	#include	"FanModule.h"	
7	#include	"SuctionNozzle.h"	
8	#include	"ESP32PWMServo.h"	

### 5.2 Initialization

Initialize the required module and set WonderCam vision module as facial recognition mode.

```
22 void setup() {
23   cam.begin();
24   Buzzer_init();
25   ESPMax_init();
26   Nozzle_init();
27   PWMServo_init();
28   FanModule_init();
29   Valve_on();
30   SetPWMServo(1, 1500, 1000);
31   Valve_off();
32   setBuzzer(100);
33   Serial.begin(115200);
34   Serial.println("start...");
35   Matrix.setDisplay(empty_buf, 16);
36   cam.changeFunc(APPLICATION_FACEDETECT); |
```

The program uses Valve\_on() function to turn on fan, and Valve\_off() function to turn off fan, as the figure shown below.

```
29 Valve_on();
```

```
31 Valve_off();
```

Next, use setBuzzer() function to control buzzer. The parameter 100 represents the sound duration is 100ms, as the figure shown below.

```
32 setBuzzer(100); // The buzzer is set to sound for 100ms
```

Then, use Matrix.setDisplay(empty\_buf, 16) function to clear the screen of the dot matrix module, as the figure shown below.

```
35 Matrix.setDisplay(empty_buf, 16);
```

And set WonderCam vision module as facial recognition mode.

```
36 cam.changeFunc (APPLICATION_FACEDETECT);
```

Robot arm is controlled to turn through the inverse kinematics function set\_position(), as the figure shown below.

```
39 void loop() {  
40     float pos[3];  
41     pos[0] = 0;  
42     pos[1] = -120;  
43     pos[2] = 150;  
44     set_position(pos, 1500);  
45     delay(1500);  
46     bool display_st = false;
```

Take “set\_position(pos, 1500)” above as example:

The first parameter “pos” is the position coordinate of the suction nozzle.

Assign pos. The first parameter "pos" is the position coordinates of the suction



nozzle. Assign pos as 0, -120, 150, that is, the position coordinates of the end-effector of the robot arm are (0, -120, 150). (About the determination of the spatial coordinate system and position, please go to " 6 Secondary Development/ Arduino Development/ Inverse Kinematics Learning (must-see!!!)"

The second parameter "1500" is the running time and the unit is ms.

## 5.3 Face Detection

Robotic arm rotates continuously, and constantly detects a human face through WonderCam AI vision module, and keeps updating the data of the vision module .

Then judge the returned data, as the figure shown below:

```
49      cam.updateResult();  
50      if (cam.anyFaceDetected()) {
```

If a human face is recognized, then determine whether the fan is turned off. If it is off, use the fan.on() function to turn on fan and display smile expression on the dot matrix module.

```
51          if (!display_st) {  
52              display_st = true;  
53              FanModule_on();  
54              Matrix.setDisplay(smiling_buf, 16);  
55          }
```

Compared with the data returned by vision module, if no human face is detected, the fan will turn off and the screen of dot matrix will go black through the tm.update\_display() function, as the figure shown below:

```
81     if (display_st) {
82         display_st = false;
83         FanModule_off();
84         Matrix.setDisplay(empty_buf, 16);
85     }
86 }
87 delay(50);
```

## 5.4 Search Human Face

Use PID algorithm to set the x, y and z values of end-effector to search human face to realize the tracking function.

```
61     if (abs(face_x - 160) < 15) {
62         face_x = 160;
63     }
64     x_pid.setTarget(160);
65     x_pid.setInput(face_x);
66     pos[0] -= x_pid.getOutput();
67
68     if (abs(face_y - 120) < 10) {
69         face_y = 120;
70     }
71     y_pid.setTarget(120);
72     y_pid.setInput(face_y);
73     pos[2] += y_pid.getOutput();
74
75     if (pos[0] > 100) pos[0] = 100;
76     if (pos[0] < -100) pos[0] = -100;
77     if (pos[2] > 180) pos[2] = 180;
78     if (pos[2] < 100) pos[2] = 100;
79     set_position(pos, 50);
80 } else {
```

Firstly, obtain the coordinate of the human face on vision module, as the figure shown below:

```
56 WonderCamFaceDetectResult p;  
57 cam.getFaceOfIndex(1, &p);  
58 int face_x = p.x;  
59 int face_y = p.y;
```

Calculate the coordinates of the human face and center point obtained from vision module. If the calculated x-axis value satisfies the following conditions, PID algorithm is used to calculate the distance between the robot arm and the face coordinate.

```
61 if (abs(face_x - 160) < 15) {  
62     face_x = 160;
```

The specific calculation method is as follow:

```
64 x_pid.setTarget(160);  
65 x_pid.setInput(face_x);  
66 pos[0] -= x_pid.getOutput();
```

Because the movement of the robot arm on space coordinate has a certain range, it is necessary to set a movement range for the x and z.

```
75 if (pos[0] > 100) pos[0] = 100;  
76 if (pos[0] < -100) pos[0] = -100;  
77 if (pos[2] > 180) pos[2] = 180;  
78 if (pos[2] < 100) pos[2] = 100;
```

The calculation method of y and z axes is the same.

After calculating the coordinate position of the face on screen, substitute the value into the inverse kinematics function to achieve the tracking function as the figure shown below.

79	<code>set_position(pos, 50);</code>
----	-------------------------------------

Among them,

The first parameter “pos” represents the coordinate values of x, y and z axes of end effector.(The difference between the center coordinate of the face and the center coordinates of screen).

The second parameter “50” represents the time it takes for the robot arm to move to the coordinate position.