

Lesson 3 Waste Sorting

Note: If the mask recognition models have been burnt, please re-download the firmware of WonderCam vision module before starting this game. The specific operation refers to “WonderCam Firmware (Installation method included)”.

1. Project Principle

Using image classification function, robot arm is able to recognize waste cards, and then sort them into the corresponding position.

Waste classification is mainly implemented by the trained waste models, and the sorting function is implemented by inverse kinematics function to control robotic arm. The specific control method refers to “5. Program Instruction” in this lesson.

The path of the program file: “7. AI Vision Game/ Python Development/ Sensor-extension Game/ Program Files/ Waste Sorting/main.py”

```
34 = while True:
35     cam.update_result()
36     result_data.append(cam.most_likely_id())
37 = if len(result_data) == 30:
38     result = sum(result_data) / 30.0
39     result_data = []
40
41 = if result != int(result):
42     result = 0
43     continue
44
45 = if 2 <= result and result <= 4:
46     print('id:',int(result),' Hazardous waste')
47     angle = 38
48     move_time = 1000
49     (place_x, place_y, place_z) = (-120,-170,60)
50
51 = elif 5 <= result and result <= 7:
52     print('id:',int(result),' Recyclable material')
53     angle = 52
54     move_time = 1200
55     (place_x, place_y, place_z) = (-120,-120,60)
56
57 = elif 8 <= result and result <= 10:
58     print('id:',int(result),' Kitchen garbage')
59     angle = 68
```

2. Preparation

2.1 Hardware

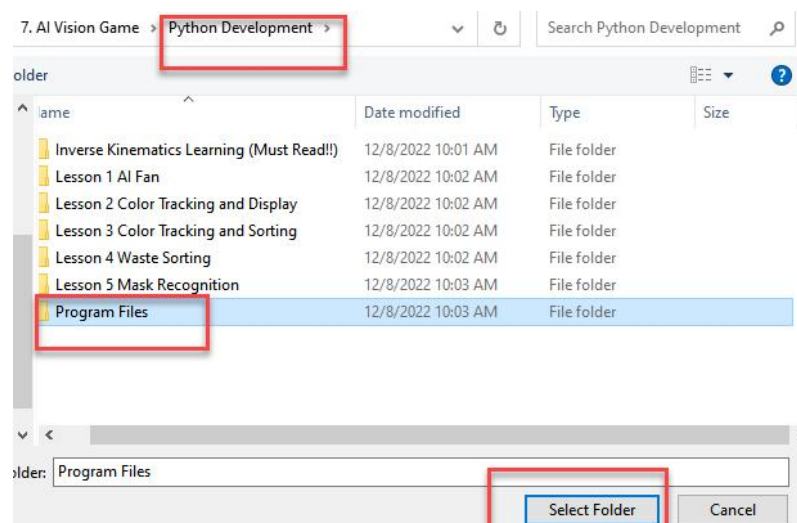
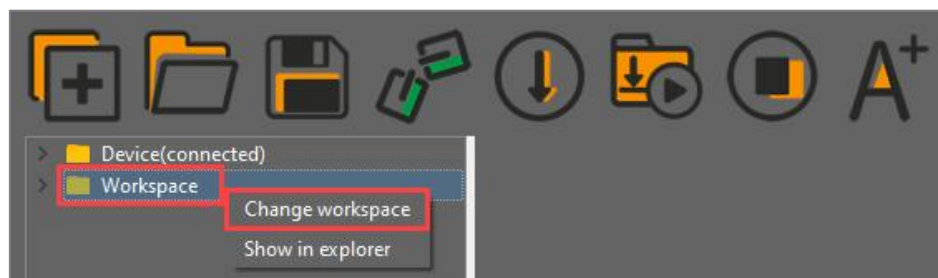
Please refer to “7. AI Vision Game/ WonderCam AI Vision Module Learning (Must read!!)/ Lesson 3 Assembly” to assemble WonderCam to the corresponding position on MaxArm.

2.2 Software

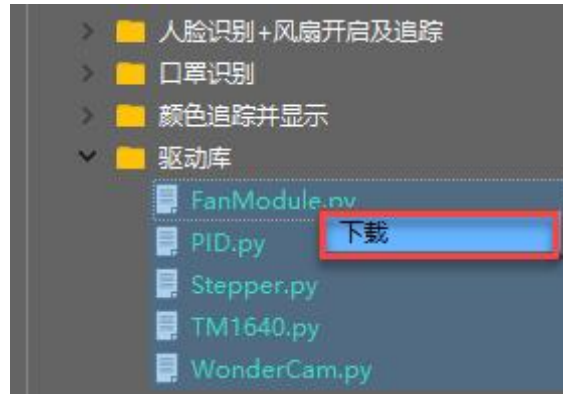
Please connect MaxArm to Python editor according to the tutorial in folder “4. Underlying Program Learning/Python Development/Lesson 1 Set Development Environment”.

3. Program Download

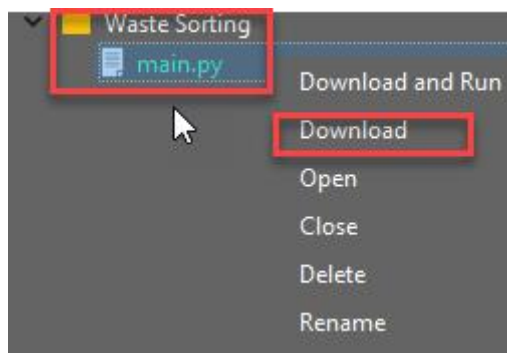
1) After connecting, change the path of Workspace to “7. AI Vision Game/ Python Development”, and select “Program Files”.



2) Then double click “Driver Library” folder, and select all library files, and right click and select “Download” to download the library files to the controller. (If you have downloaded this library file before, please skip this step.)



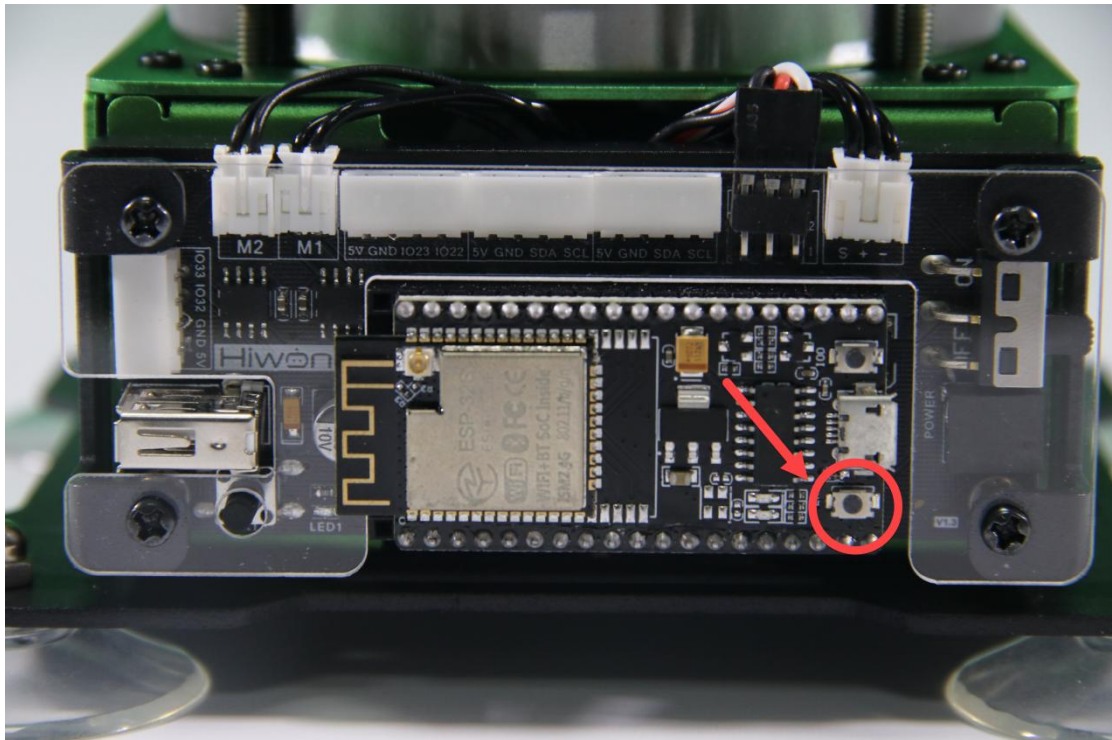
3) Then double click “Waste Sorting” folder and select “main.py” folder. Then right click and select “Download” to download all the program files to the controller.



When the terminal prints the prompt as shown in the image below, it means download completed.

```
>>>
Downloading.....
main.py Download ok !
>>>
```

4) After downloading, click on the reset icon or press the reset button on ESP32 controller to run program.



4. Project Outcome

After the program is downloaded, press reset button according to the operation steps in “3. Program Download”. Then robot arm will be in preparation for recognition.

Then place a waste card to the recognition area. When the card is recognized, MaxArm will suck and transfer it to the corresponding position according to the card category.

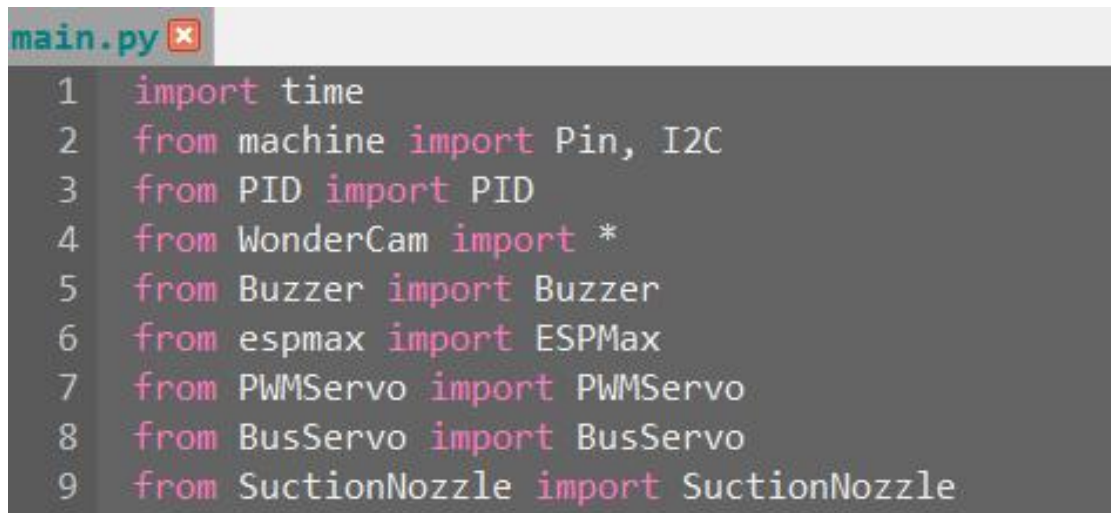
5. Program Instruction

The path of the program file: “7. AI Vision Game/ Python Development/ Sensor-extension Game/ Program Files/ Waste Sorting/main.py”. 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 inverse kinematics library, WonderCam driver library, buzzer library, PWM servo library, suction nozzle library and other function library files corresponding to this game.



```
main.py x
1 import time
2 from machine import Pin, I2C
3 from PID import PID
4 from WonderCam import *
5 from Buzzer import Buzzer
6 from espmax import ESPMax
7 from PWMServo import PWMServo
8 from BusServo import BusServo
9 from SuctionNozzle import SuctionNozzle
```

Then initialize each modules, as the figure shown below:


```

14  pwm = PWMServo()
15  buzzer = Buzzer()
16  pwm.work_with_time()
17  bus_servo = BusServo()
18  arm = ESPMax(bus_servo)
19  nozzle = SuctionNozzle()
20  i2c = I2C(0, scl=Pin(16), sda=Pin(17), freq=400000)
21  cam = WonderCam(i2c)
22  cam.set_func(WONDERCAM_FUNC_CLASSIFICATION)
23  cam.set_led(True)
24
25  = if __name__ == '__main__':
26      x, y, z = 0, -120, 150
27      buzzer.setBuzzer(100)
28      nozzle.set_angle(0, 1000)
29      arm.set_position((x, y, z), 2000)
30      time.sleep_ms(2000)
31      result_data = []
32      result = 0
33

```

5.2 Image Classification Recognition

Call the `cam.updateResult()` function to update the recognized result, and call the `cam.classIdOfMaxProb()` function to obtain ID number the save it to `result_data`. (Each waste card has a specific ID number).

```

34  = while True:
35      cam.update_result()
36      result_data.append(cam.most_likely_id())

```

To prevent misidentification, we take the average value of 30 detections and assign it to `result`.

```

37  = if len(result_data) == 30:
38      result = sum(result_data) / 30.0
39      result_data = []

```

If the result is not an integer value, it means the test result is unstable and the program needs to restart the next round of testing.

```
41 = if result != int(result):  
42     result = 0  
43     continue
```

Then sort the card into corresponding card category according to the result ID.

```
45 = if 2 <= result and result <= 4:  
46     print('id:',int(result), ' Hazardous waste')  
47     angle = 38  
48     move_time = 1000  
49     (place_x, place_y, place_z) = (-120,-170,60)  
50  
51 = elif 5 <= result and result <= 7:  
52     print('id:',int(result), ' Recyclable material')  
53     angle = 52  
54     move_time = 1200  
55     (place_x, place_y, place_z) = (-120,-120,60)
```

5.3 Suck Card

After the card is recognized, buzzer makes sound and MaxArm sucks and place the block to the corresponding card category, as the figure shown below:

```

72     d_y = 65
73     buzzer.setBuzzer(100)
74     arm.set_position((x,y-d_y,100),1000)
75     time.sleep_ms(1000)
76     arm.set_position((x,y-d_y,50),600)
77     nozzle.on()
78     time.sleep_ms(1000)
79     arm.set_position((x,y-d_y,150),800)
80     time.sleep_ms(1000)
81     arm.set_position((place_x,place_y,150),move_time)
82     nozzle.set_angle(angle,move_time)
83     time.sleep_ms(move_time)
84     arm.set_position((place_x,place_y,place_z),800)
85     time.sleep_ms(1000)
86     nozzle.off()
87     arm.set_position((place_x,place_y,150),800)
88     time.sleep_ms(1000)
89
90     x, y, z = 0, -120, 150
91     arm.set_position((x, y, z), move_time)
92     nozzle.set_angle(0,move_time)
93     time.sleep_ms(move_time)

```

Use the “setBuzzer()” function to control buzzer, as the figure shown below.
Among them, the parameter 100 represents the sound duration of buzzer and its unit is ms.

```

73     buzzer.setBuzzer(100)

```

Use the “arm.set_position()” function to control MaxArm to locate above the card, as the figure shown below:

```

74     arm.set_position((x,y-d_y,100),1000)

```

The parameter “(x,y-d_y,100)” represents the coordinate of the target position.
The value of x, y-d_y, 100 respectively represents the value of x, y and z.

5.4 Sorting

Set up card placement areas and sort the card into the corresponding area according to the card category. However, the sorting function is implemented

by the inverse kinematics function, as the figure shown below:

```
6 from espmax import ESPMax

45 = if 2 <= result and result <= 4:
46     print('id:',int(result),' Hazardous waste')
47     angle = 38
48     move_time = 1000
49     (place_x, place_y, place_z) = (-120,-170,60)
```

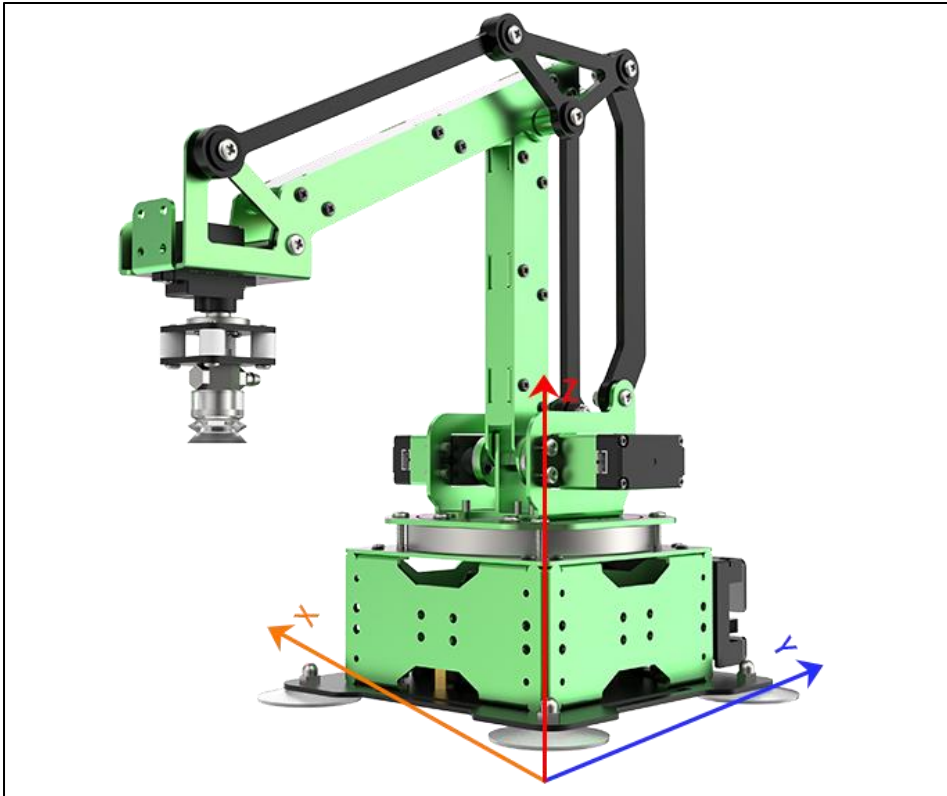
Among them, “(place_x, place_y, place_z)” represents the coordinate of end-effector (suction nozzle). Here the coordinate of end effector is “(-120,-170,60)”, which means robotic arm needs to transfer the hazardous card to (-120,-170,60).

6. Function Extension

The program defaults to place the cards in a row in front of the left side of the robot arm. In this section, you can learn how to modify the placement position of cards. Take placing the cards around robotic arm for example.

Firstly, take robotic arm as the first person view. The positive direction of x axis corresponds to the right side of robotic arm; the positive direction of y axis is the rear of robotic arm; the positive direction of z axis is the top of robotic arm. All position adjustment is based on this coordinate system.

The specific coordinate system refers to the following figure:



After connecting to the ports according to the steps 1-2 in “2.Preparation and 3. Program Download”, open and download the corresponding program files of Waste Sorting. Then find the codes in “main.py”:

```
43         continue
44
45     if 2 <= result and result <= 4:
46         print('id:',int(result),' Hazardous waste')
47         angle = 38
48         move_time = 1000
49         (place_x, place_y, place_z) = (-120,-170,60)
50
51     elif 5 <= result and result <= 7:
52         print('id:',int(result),' Recyclable material')
53         angle = 52
54         move_time = 1200
55         (place_x, place_y, place_z) = (-120,-120,60)
56
57     elif 8 <= result and result <= 10:
58         print('id:',int(result),' Kitchen garbage')
59         angle = 68
60         move_time = 1400
61         (place_x, place_y, place_z) = (-120,-70,60)
62
63     elif 11 <= result and result <= 13:
64         print('id:',int(result),' Other garbage')
65         angle = 90
66         move_time = 1600
67         (place_x, place_y, place_z) = (-120,-20,60)
```

```

72     d_y = 65
73     buzzer.setBuzzer(100) |
74     arm.set_position((x,y-d_y,100),1000)
75     time.sleep_ms(1000)
76     arm.set_position((x,y-d_y,50),600)
77     nozzle.on()
78     time.sleep_ms(1000)
79     arm.set_position((x,y-d_y,150),800)
80     time.sleep_ms(1000)
81     arm.set_position((place_x,place_y,150),move_time)
82     nozzle.set_angle(angle,move_time)
83     time.sleep_ms(move_time)
84     arm.set_position((place_x,place_y,place_z),800)
85     time.sleep_ms(1000)
86     nozzle.off()
87     arm.set_position((place_x,place_y,150),800)
88     time.sleep_ms(1000)

```

Take modifying the codes in red box as example. The parametrer place_x, place_y and place_z represents the coordinate of x, y, and z axes of the card placement position.

Change the codes in red box to the following codes:

```

45 = if 2 <= result and result <= 4: # 有害垃圾
46     print('id:',int(result),' Hazardous waste')
47     angle = 38 # 设置位置初始角度
48     move time = 1000
49     (place_x, place_y, place_z) = (120,-120,200) # 设置位置坐标位置
50
51 = elif 5 <= result and result <= 7: # 可回收物
52     print('id:',int(result),' Recyclable material')
53     angle = 52
54     move time = 1200
55     (place_x, place_y, place_z) = (-120,-120,200)
56
57 = elif 8 <= result and result <= 10: # 厨余垃圾
58     print('id:',int(result),' Kitchen garbage')
59     angle = 68
60     move time = 1400
61     (place_x, place_y, place_z) = (120,60,200)
62
63 = elif 11 <= result and result <= 13: # 其他垃圾
64     print('id:',int(result),' Other garbage')
65     angle = 90
66     move time = 1600
67     (place_x, place_y, place_z) = (-120,60,200)
68

```

```

79     arm.set_position((x,y-d_y,150),800) # 机械臂抬起来
80     time.sleep_ms(1000)
81     arm.set_position((place_x,place_y,200),move_time) # 移动到放置位置上方
82     nozzle.set_angle(angle,move_time) # 调整喷嘴角度，使卡片放正
83     time.sleep_ms(move_time)
84     arm.set_position((place_x,place_y,place_z),800) # 放置卡片
85     time.sleep_ms(1000)
86     nozzle.off() # 关闭气泵
87     arm.set_position((place_x,place_y,200),800) # 机械臂抬起来
88     time.sleep_ms(1000)

```

Take the code of hazardous card for example:

place_x is modified from -120 to 120, which corresponds to the adjustment of the card position from -120 to 120 on x-axis, i.e. the card position is changed from the right to the left and the robot arm moves to the left.

place_y is modified from -170 to -120, which corresponds to the adjustment of the card position from -170 to -120 on y-axis, i.e. the robot arm moves backwards.

place_z is modified from 60 to 200, which corresponds to the adjustment of the card position from 60 to 200 on z-axis. Thus robotic arm will lift up.

Therefore, the position of the hazardous card will be adjusted from (-120, -170, 60) to (120, -120, 200).

The parameters of arm.set_position() functions are modified from 150 to 200 which represents the height of the robot arm after it picks up the card and the height after it moves to the placement area, respectively.

If want to modify the placement position of other cards, you can refer to the following table to modify the corresponding coordinate parameters:

	Parameter Adjustment	Robot Movement Direction
X axis	Increase	left
	Decrease	Right
Y axis	Increase	Backward
	Decrease	Forward
Z axis	Increase	Upward
	Decrease	Downward

Note: Take yourself as the first person view to determine the movement direction.

After modification, please refer to the steps 8-10 in “3. Program Download” to check the effect.

After the cards are recognized, the kitchen waste will be placed to the rear of robotic arm, the other waste card will be placed to the right rear of the robot arm, the hazardous waste card will be placed to the left front of the robot arm, the recyclable waste will be placed to the right front of the robot arm. Please refer to the following image:



(The props in image just for demonstration)

The specific parameters of the codes are as follow:

Recyclable Waste: place_x= -120; place_y = -120; place_z = 60;

——》 place_x = -120; place_y = -120; place_z = 200;

Kitchen Waste: place_x = -120; place_y = -70; place_z = 60;

——》 place_x = 120; place_y = 60; place_z = 200;

Other Waste: place_x = -120; place_y = -20; place_z = 60;

——》 place_x = -120; place_y = 60; place_z = 200;

Hazardous Waste: place_x= -120; place_y = -170; place_z = 60;

——》 place_x = 120; place_y = -120; place_z = 200;