

Lesson 1 The Use of Monocular Camera

Loaded with USB mono camera which is equivalent to “eye”, ArmPi Pro robotic arm can obtain the transmitted image.

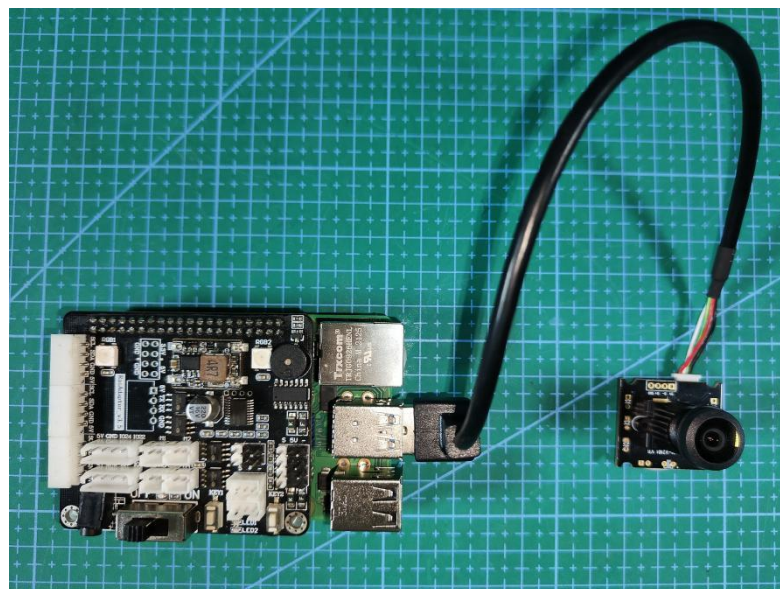
Mono camera has sensing, surround view, and monitoring functions.

Compared with other types of cameras, it not only features low delay, low jitter and immediate use after connecting but also small in size and cost-effective, which mono camera becomes a popular choice for industrial design.

1.Operation Steps

1.1 Connect Raspberry Pi

The wiring between monocular camera and Raspberry Pi board is as the figure shown below:




1.2 Turn on Camera

i It should be case sensitive when entering command, and the “Tab” key can be used to complete the keywords.

1) Turn on ArmPi Pro robotic arm and connect it to ubuntu desktop via NoMachine.

2) Click  Applications in the lower left corner and select

 Terminal Emulator to open the terminal.

3) Enter command “sudo systemctl stop start_node.service” and press “Enter” to stop the node of Raspberry Pi.

```
ubuntu@ubuntu:~$ sudo systemctl stop start_node.service
```

4) Enter command “cd armpi_pro/src/armpi_pro_demo/CameraCalibration/” and press “Enter” to enter the program.

```
ubuntu@ubuntu:~$ cd armpi_pro/src/armpi_pro_demo/CameraCalibration/
```

5) Enter command “python3 Camera.py” and press “Enter” to start the game.

```
ubuntu@ubuntu:~/armpi_pro/src/armpi_pro_demo/CameraCalibration$ python3 Camera.py
```

6) If want to exit the game, you can press “Ctrl+C”. If fail to exit, please try few more times.

2. Project Outcome

After starting the game, the transmitted image will be displayed on screen.



3. Program Parameter Instruction

The source code of program is located in :
 /home/ubuntu/armpi_pro/src/armpi_pro_demo/CameraCalibration/Camera.py

```

9  if sys.version_info.major == 2:
10     print('Please run this program with python3!')
11     sys.exit(0)
12
13  class Camera:
14     def __init__(self, resolution=(640, 480)):
15         self.cap = None
16         self.width = resolution[0]
17         self.height = resolution[1]
18         self.frame = None
19         self.opened = False
20
21         self.th = threading.Thread(target=self.camera_task, args=(), daemon=True)
22         self.th.start()
23
24     def camera_open(self):
25         try:
26             self.cap = cv2.VideoCapture(-1)
27             self.cap.set(cv2.CAP_PROP_FOURCC, cv2.VideoWriter_fourcc('Y', 'U', 'Y', 'V'))
28             self.cap.set(cv2.CAP_PROP_FPS, 30)
29             self.cap.set(cv2.CAP_PROP_SATURATION, 40)
30             self.opened = True
31         except Exception as e:
32             print('Camera open on failure:', e)
33
34     def camera_close(self):
35         try:
36             self.opened = False
37             time.sleep(0.2)
38             if self.cap is not None:
39                 self.cap.release()
40                 time.sleep(0.05)
41             self.cap = None
42         except Exception as e:
43             print('Camera close on failure:', e)

```

◆ Open Camera

Use external camera by calling VideoCapture() function in cv2 library, as the figure shown below:

```
24 def camera_open(self):
25     try:
26         self.cap = cv2.VideoCapture(-1)
```

◆ Read and Display Image

If the parameter in “VideoCapture(-1)” is “0”, it means the built-in camera is called; if it is other values, it means the external camera is called.

```
48 if self.opened and self.cap.isOpened():
49     ret, frame_tmp = self.cap.read()
50     if ret:
51         self.frame = cv2.resize(frame_tmp, (self.width, self.height), interpolation=cv2.INTER_NEAREST)
52     ret = False
```

Then return “True or False” value by “cap.read()” function. If the frame is read correctly, the returned value is True.

Finally, display image by calling imshow() function.

```
75 print('The original image of the camera and does not be corrected for distortion')
76 while True:
77     img = my_camera.frame
78     if img is not None:
79         cv2.imshow('img', img)
80         key = cv2.waitKey(1)
81         if key == 27:
82             break
83     my_camera.camera_close()
84     cv2.destroyAllWindows()
```

Take the code “cv2.imshow("img", frame)” as example. The meaning of parameters in parentheses is as follow:

The first parameter “img” is the title of display window.

The second parameter “frame” is the the image transmitted by camera.