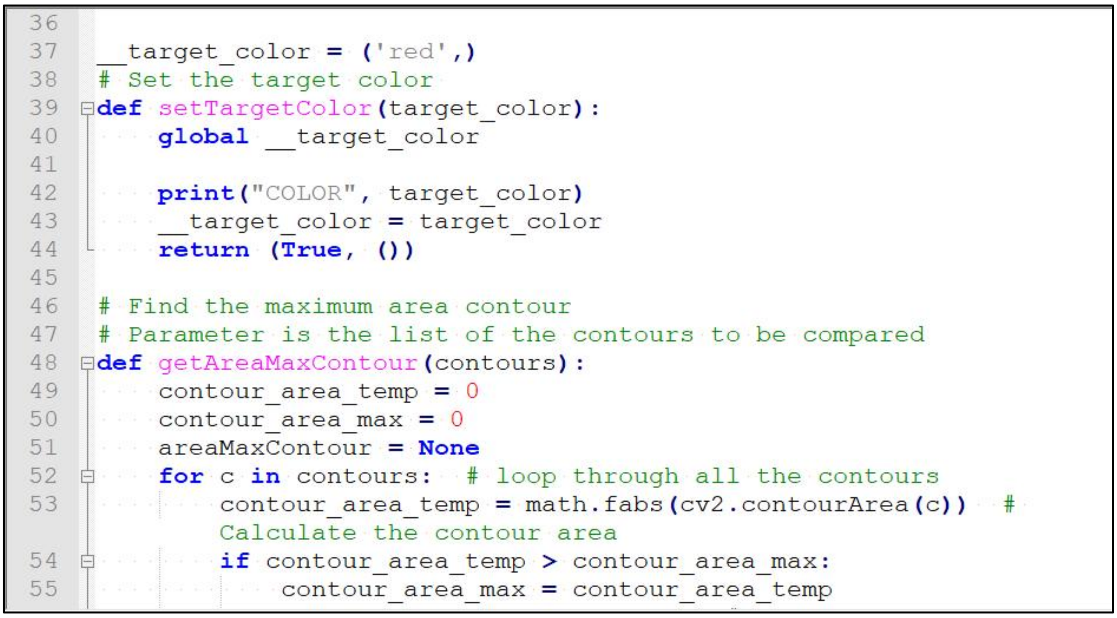
Lesson 3 Target Tracking

1. Working Principle

Firstly, convert the color of object through Lab color space and then frame target outline with a circle after processing the image. When the object is recognized, it can be tracked. Tracking mode includes robotic arm tracking and car following.

Robotic arm tracking mode is that robotic arm tracks target while car keeps stationary. Car following mode is that car follows the movement of the target but robotic arm keeps stationary.

The source code of program is located in: **/home/pi/MasterPi/Functions/ColorTracking.py**



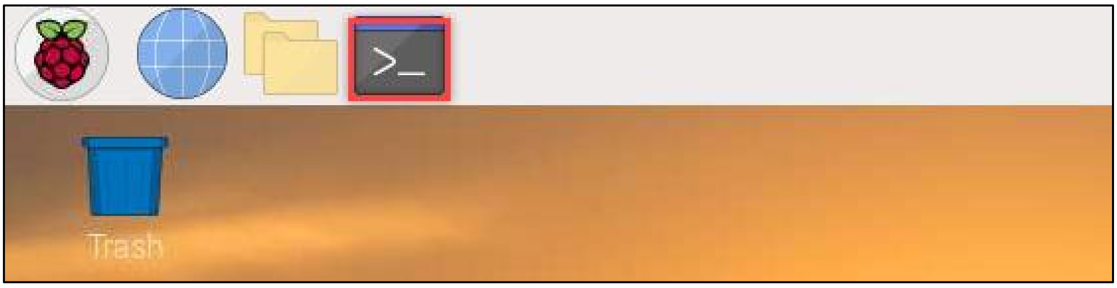
```python

__target_color = ('red',)
# Set the target color
def setTargetColor(target_color):
    global __target_color

    print("COLOR", target_color)
    __target_color = target_color
    return (True, ())

# Find the maximum area contour
# Parameter is the list of the contours to be compared
def getAreaMaxContour(contours):
    contour_area_temp = 0
    contour_area_max = 0
    areaMaxContour = None
    for c in contours:  # loop through all the contours
        contour_area_temp = math.fabs(cv2.contourArea(c))  # Calculate the contour area
        if contour_area_temp > contour_area_max:
            contour_area_max = contour_area_temp
```

2. Operation Steps

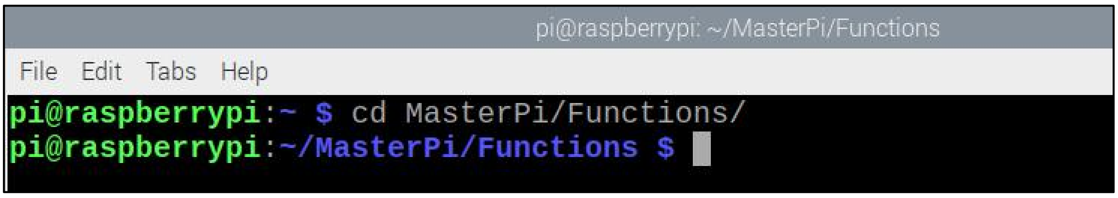
The entered command should be case sensitive. And the keywords can be complemented by Tab key.

Step 1: Turn on MaserPi, then connect to the Raspberry Pi system desktop through VNC.

Step 2: Click >_ or press "Ctrl+Alt+T" to enter LX terminal.



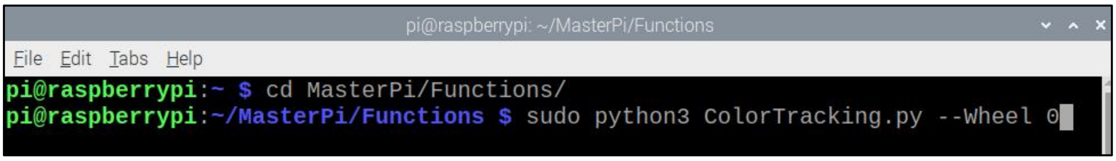
Step 3: Enter "cd MasterPi/Functions/" command, and then press "Enter" to come to the category of game programmings.



```
pi@raspberrypi:~ $ cd MasterPi/Functions/
pi@raspberrypi:~/MasterPi/Functions $ 
```

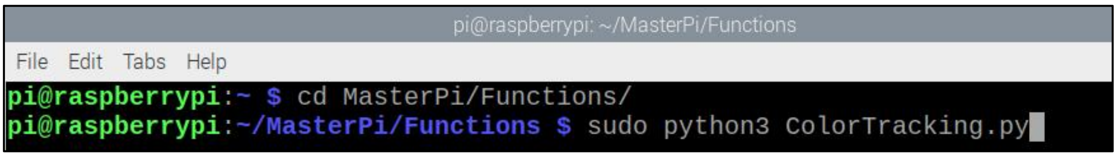
2.1 Robotic Arm Tracking

1) After entering the games directory, input "**sudo python3 ColorTracking.py --Wheel 0**" command and then press "Enter" to enter robotic arm tracking mode.



```
pi@raspberrypi:~ $ cd MasterPi/Functions/
pi@raspberrypi:~/MasterPi/Functions $ sudo python3 ColorTracking.py --Wheel 0
```

Note: MasterPi defaults to robotic arm tracking mode. You can also directly enter "sudo python3 ColorTracking.py" command to enter this mode.

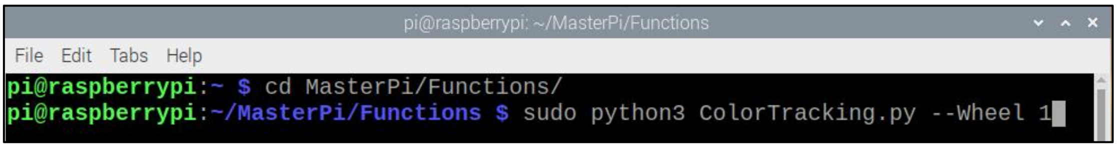


```
pi@raspberrypi:~ $ cd MasterPi/Functions/
pi@raspberrypi:~/MasterPi/Functions $ sudo python3 ColorTracking.py
```

2) If you want to exit the programming, press "Ctrl+C" in LX terminal interface. If fail to exit, please try it few more times.

2.2 Car Following

1) After entering games directory, enter “sudo python3 ColorTracking.py --Wheel 1” command and then press “Enter” to enter car following mode.



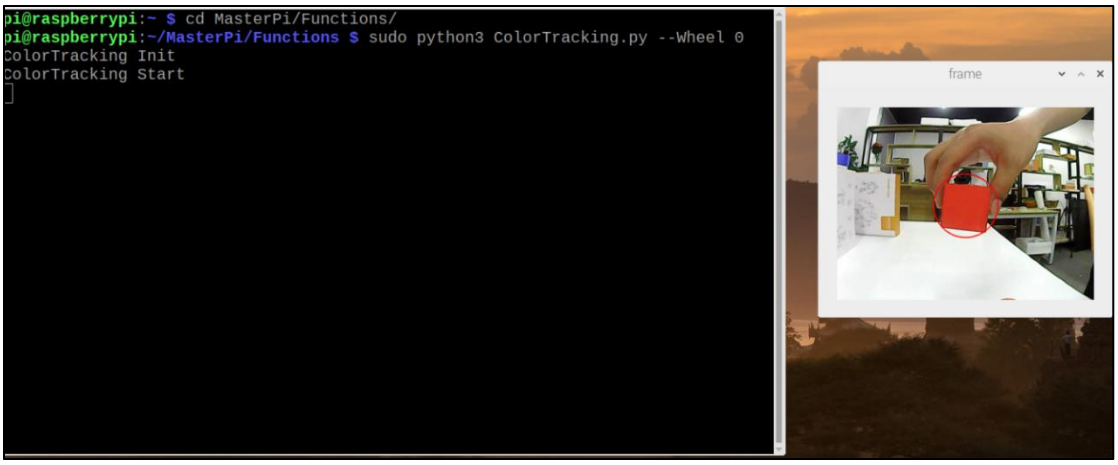
```
pi@raspberrypi:~ $ cd MasterPi/Functions/
pi@raspberrypi:~/MasterPi/Functions $ sudo python3 ColorTracking.py --Wheel 1
```

2) If you want to exit this program, press “Ctrl+C” in LX terminal interface. If the exit fails, please try it few more times.

3. Project Outcome

The default tracking color is red.

Mode	Outcome
Robot arm tracking	The robotic arm will follow the movement of the block while the car keeps still.
Car following	The car will follow the movement of the block while the robotic arm keeps still.



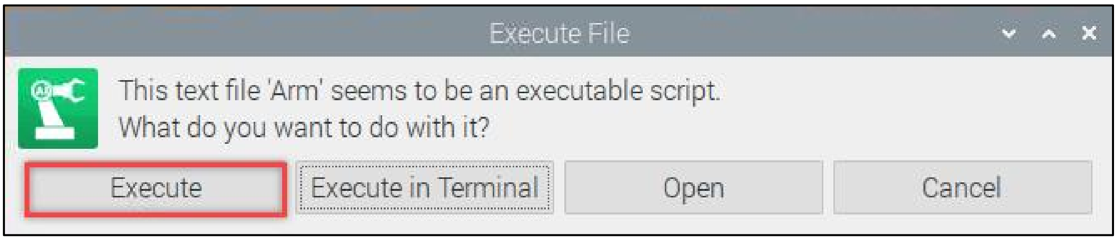
```
pi@raspberrypi:~ $ cd MasterPi/Functions/
pi@raspberrypi:~/MasterPi/Functions $ sudo python3 ColorTracking.py --Wheel 0
ColorTracking Init
ColorTracking Start
```

4. Function Extension

4.1 Color Threshold Adjustment

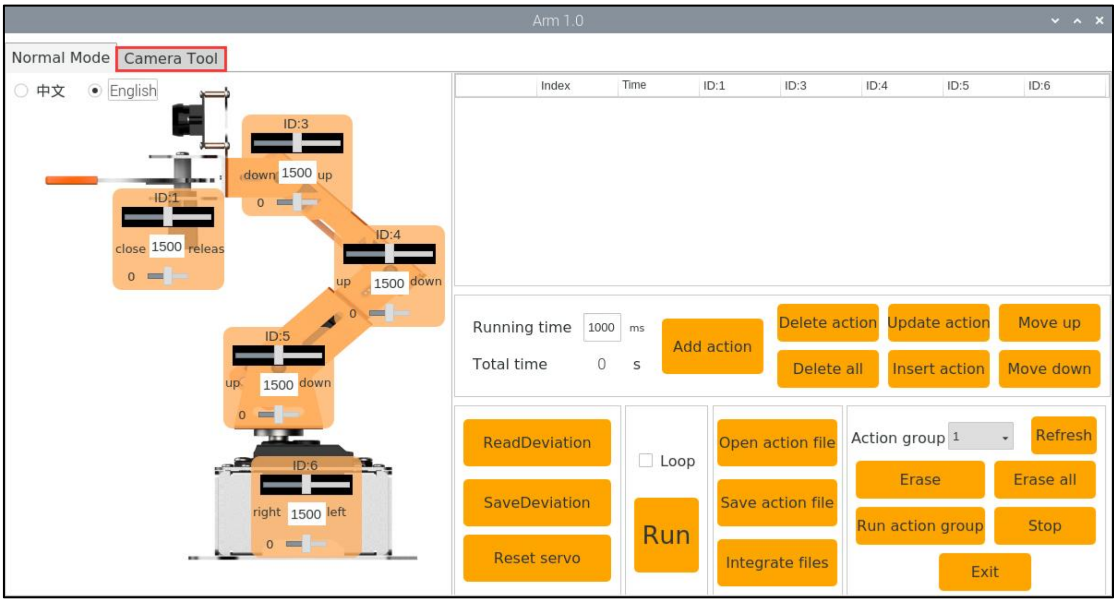
In the process of game experience, if the color recognition effect of the object is not good enough, you need to adjust the color threshold. This section will modify red as an example. The adjustment method is same to other colors. The operation steps is as follow:

1) Double-click Arm on system desktop, and then click “Execute” in the pop-up window.

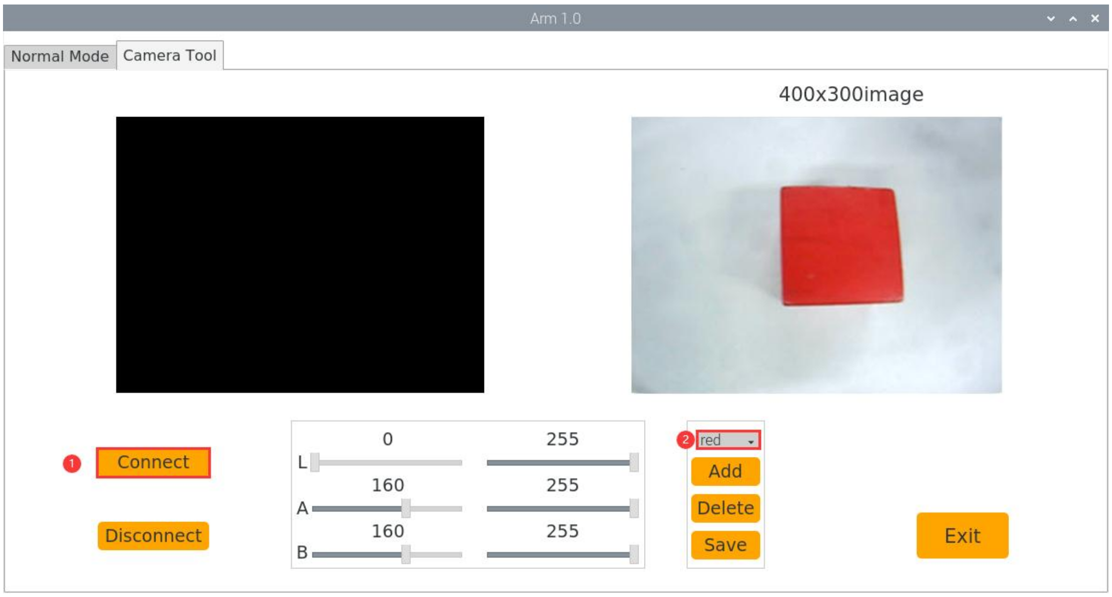


This text file 'Arm' seems to be an executable script.
What do you want to do with it?

2) After entering the interface, click “Camera Tool”.



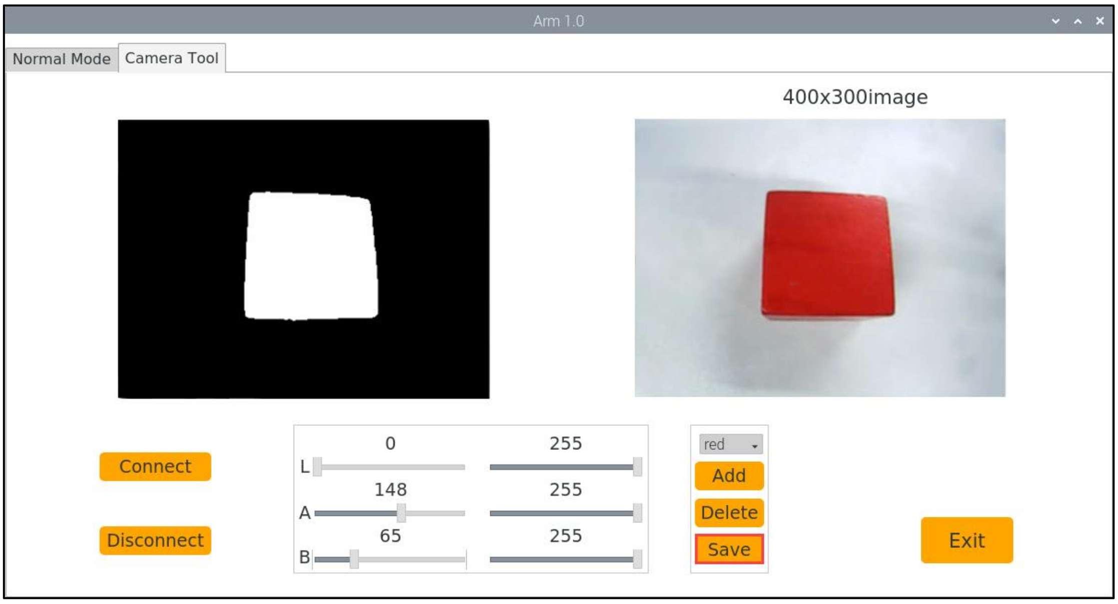
3) Then click “Connect” button. After connecting successfully, select “red” in color option bar.



4) If no real-time image transmitted by camera appears in the pop-up window, it means the camera fails to connect and need to check whether camera cable is connected normally.

The right side of the interface below is the real-time image transmitted by the camera. The left side is the color to be collected. Point the camera at the red block, and then drag the following six sliders until the the red area becomes white and other areas become black.

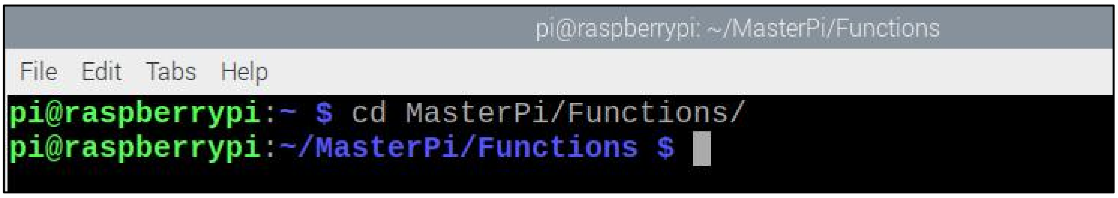
5) Then click “Save” to save the data.



4.2 Modify Tracking Color

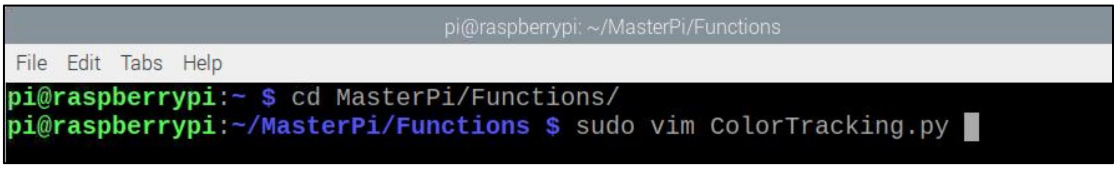
The default tracking color is red. Take modifying the default color to blue as an example:

Step 1: If want to modify the tracking color, enter “**cd MasterPi/Functions/**” command and then press “Enter” to the directory where the game programs are located.



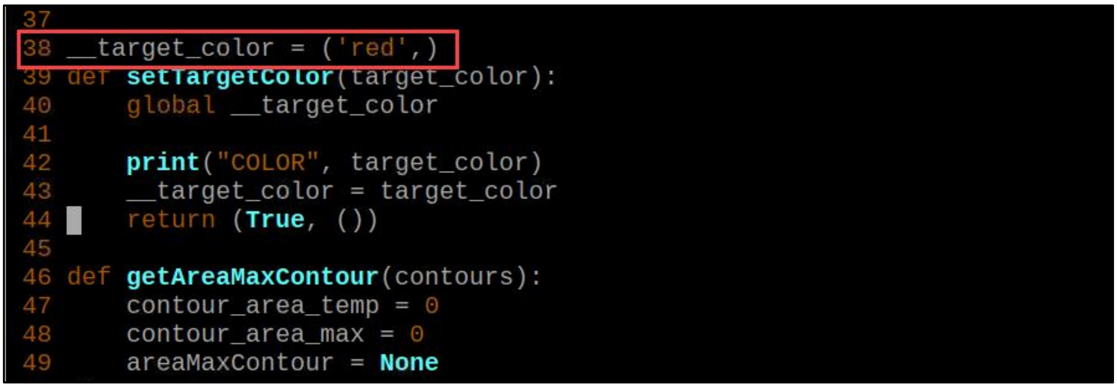
```
pi@raspberrypi:~ $ cd MasterPi/Functions/
pi@raspberrypi:~/MasterPi/Functions $ 
```

Step 2: Enter command “sudo vim ColorDetect.py”, and then press “Enter” to open program file.



```
pi@raspberrypi:~ $ cd MasterPi/Functions/
pi@raspberrypi:~/MasterPi/Functions $ sudo vim ColorTracking.py 
```

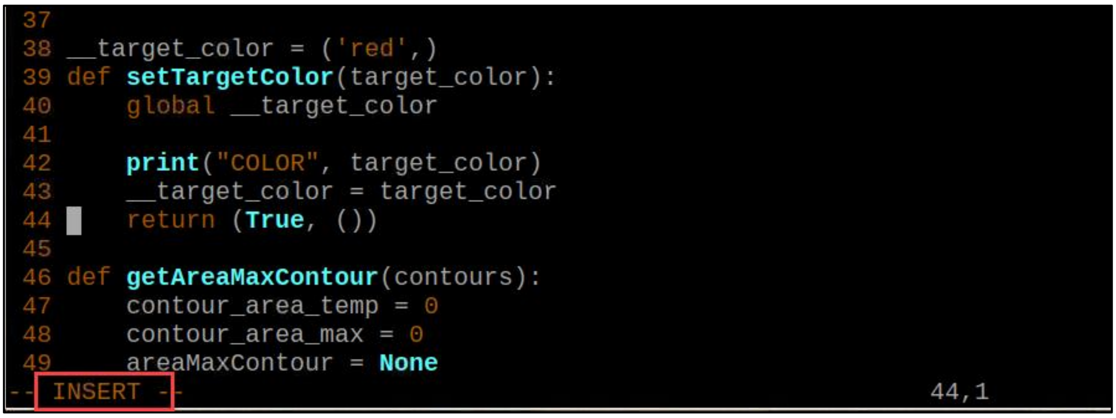
Step 3: Find the code shown in the following red box:



```

__target_color = ('red',)
def setTargetColor(target_color):
    global __target_color

    print("COLOR", target_color)
    __target_color = target_color
    return (True, ())

def getAreaMaxContour(contours):
    contour_area_temp = 0
    contour_area_max = 0
    areaMaxContour = None
```

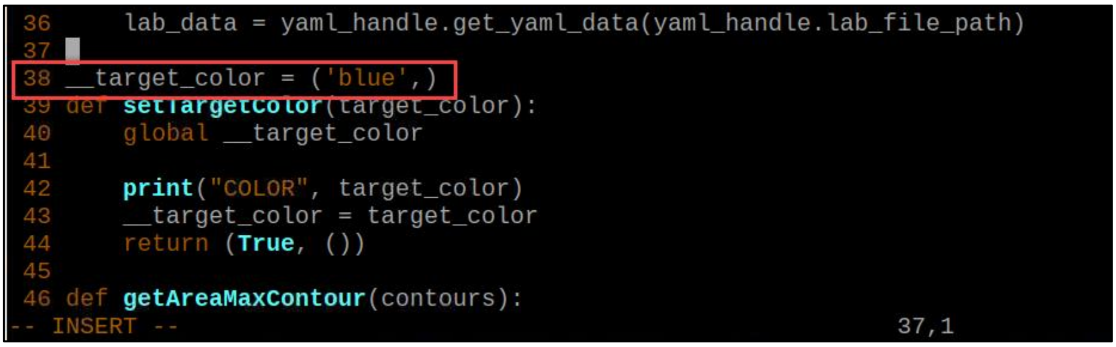
Note: After entering the position number of code, press “Shift+G” to jump to the corresponding position. (The position number of the code in figure is for reference only.)

Step 4: Press “i” on keyboard. Then enter “editing mode” when the word “INSERT” appears.



```

__target_color = ('red',)
def setTargetColor(target_color):
    global __target_color

    print("COLOR", target_color)
    __target_color = target_color
    return (True, ())

def getAreaMaxContour(contours):
    contour_area_temp = 0
    contour_area_max = 0
    areaMaxContour = None
-- INSERT --                                              44,1
```

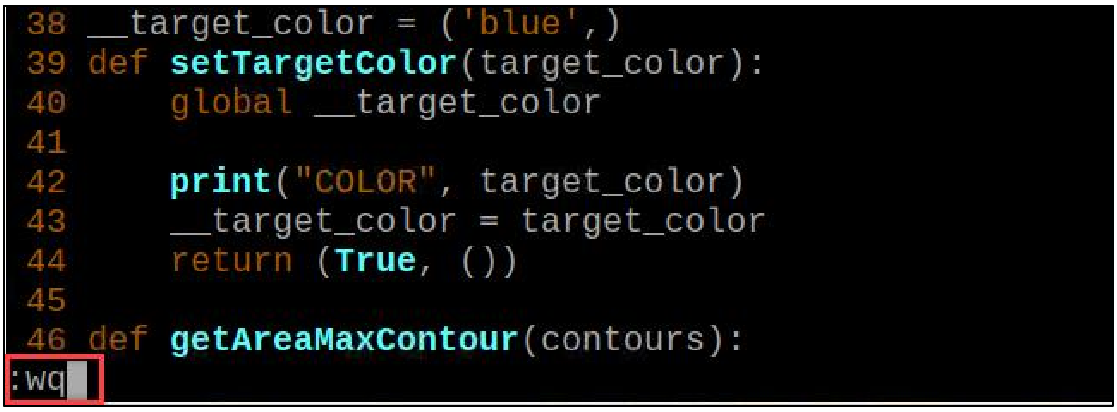
Step 5: Find the corresponding code and modify “red” in “_target_color = (‘red’)” to “blue” as shown in the figure below:



```
    lab_data = yaml_handle.get_yaml_data(yaml_handle.lab_file_path)

__target_color = ('blue',)
def setTargetColor(target_color):
    global __target_color

    print("COLOR", target_color)
    __target_color = target_color
    return (True, ())

def getAreaMaxContour(contours):
-- INSERT --                                          37,1
```

Note: The color to be modified must be in color option bar. If want to modify to other colors, you can refer to “4.3 Add Recognized Color” to add a new recognition color.

Step 7: After the modification is completed, press “Esc”. Then enter “:wq” and press “Enter” to save and exit.

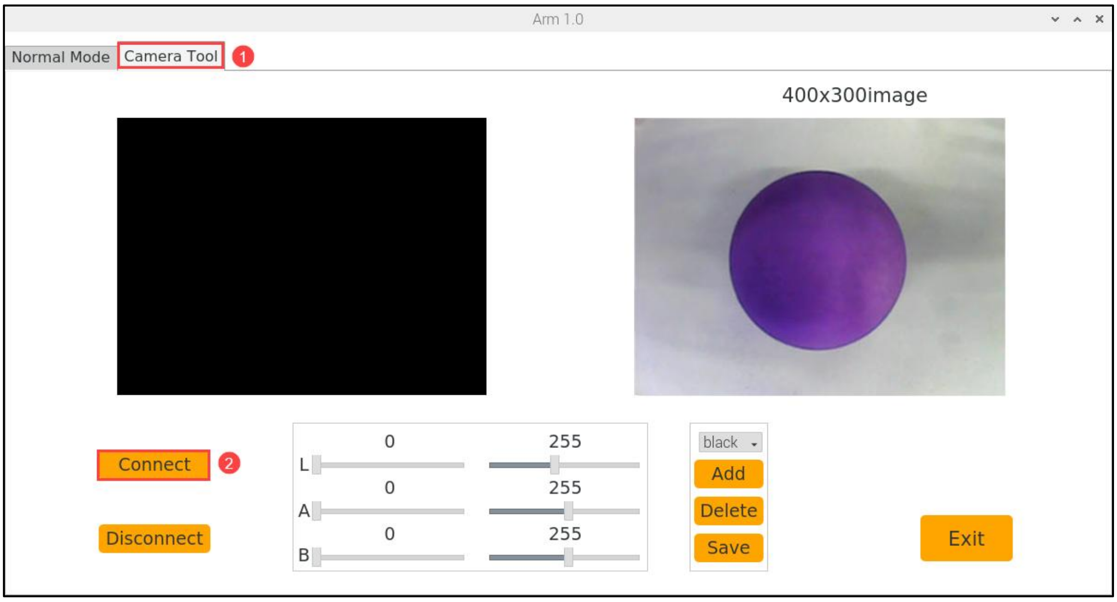


```
__target_color = ('blue',)
def setTargetColor(target_color):
    global __target_color

    print("COLOR", target_color)
    __target_color = target_color
    return (True, ())

def getAreaMaxContour(contours):
:wq
```

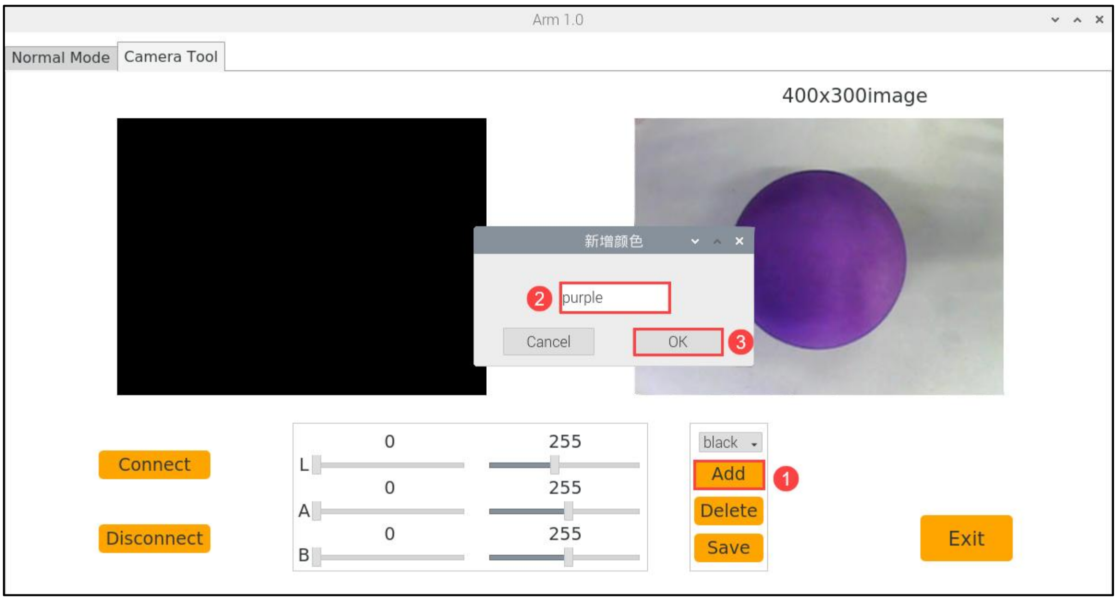
4.3 Add Recognized Color

There are three built-in colors red, green and blue in this game program. In addition to the built-in recognized colors, you can set other recognized colors in this game. Take purple as example:

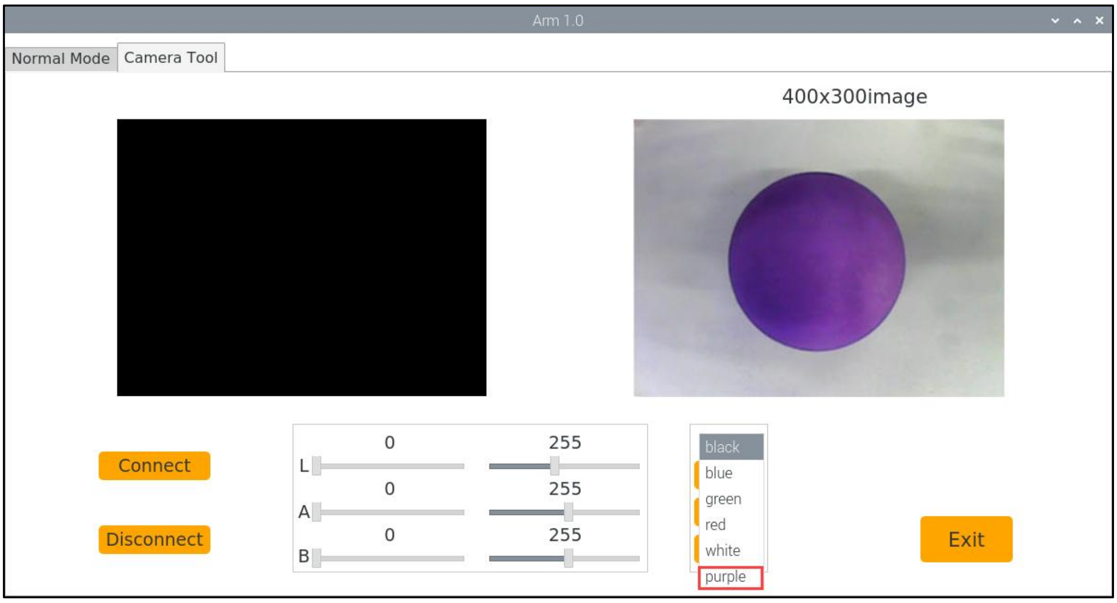
Step 1: In the pop-up interface, select “Camera Tool” and “Connect” in turn.



Step 2: Click “Add”. Then name the added color (Take “purple” as an example) and click “OK”.

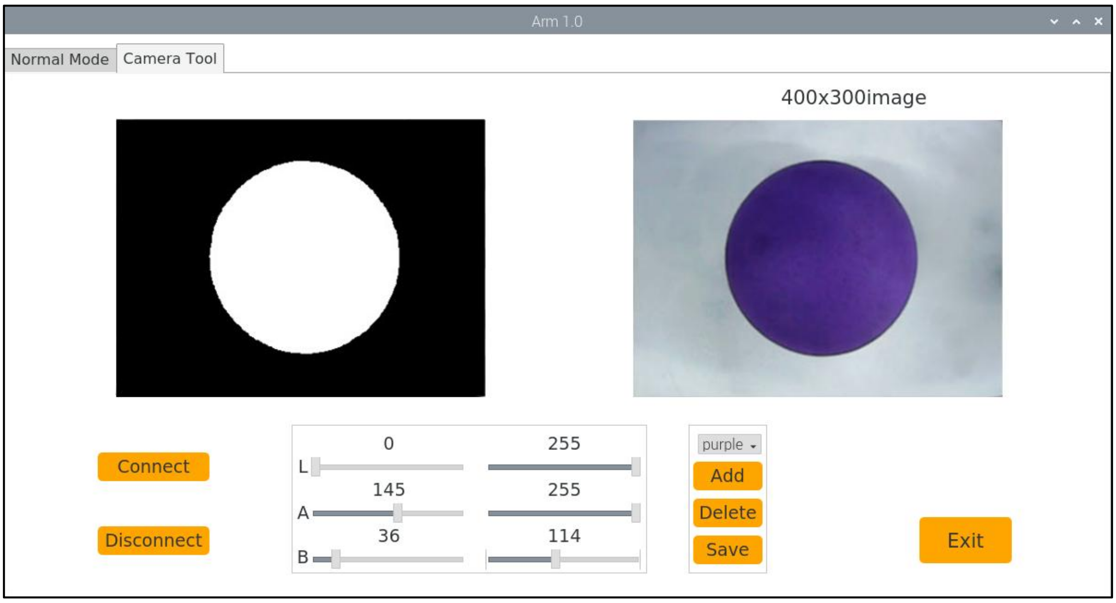


Step 3: Then select “purple” in the color potion bar.



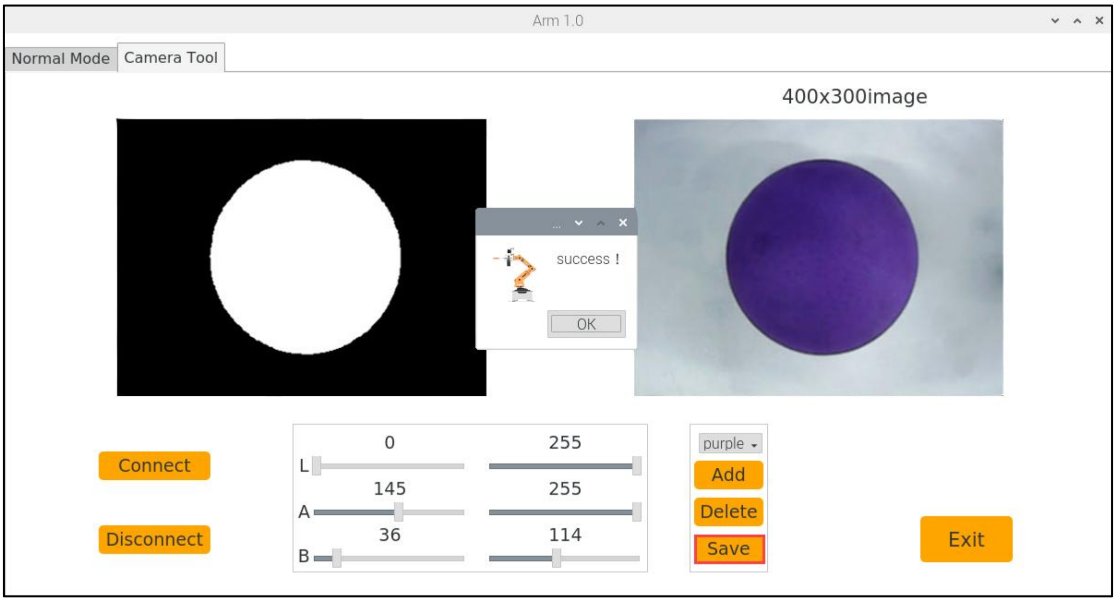
L 0 255
A 0 255
B 0 255

Step 4: Point the camera at the purple object. Drag the corresponding sliders of L, A, and B until the color area to be recognized becomes white and other areas become black.

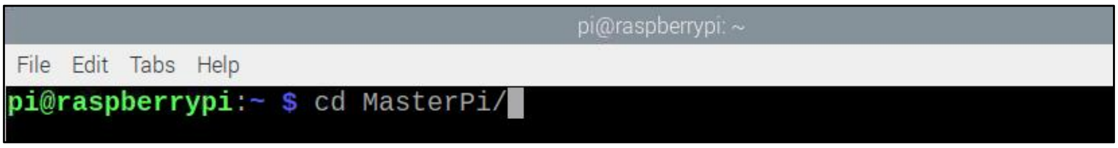


L 0 255
A 145 255
B 36 114

Step 5: Click “Save” to save the adjusted color threshold.

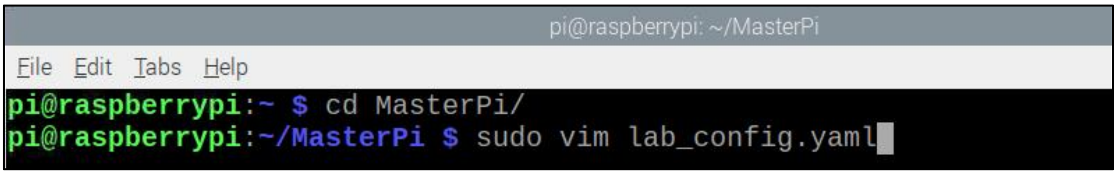


Step 6: After the modification is complete, check whether the modified data is written in successfully. Enter “cd MasterPi/” command and then press “Enter” to come to the directory of the game programmings.



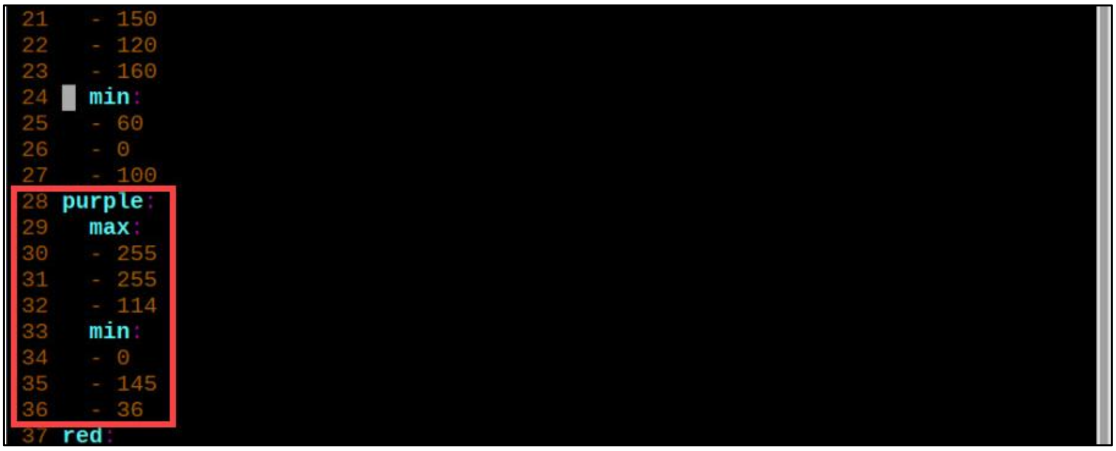
```
pi@raspberrypi:~ $ cd MasterPi/
```

Step 7: Enter “sudo vim lab_config.yaml” command, and then press “Enter” to open the program file.



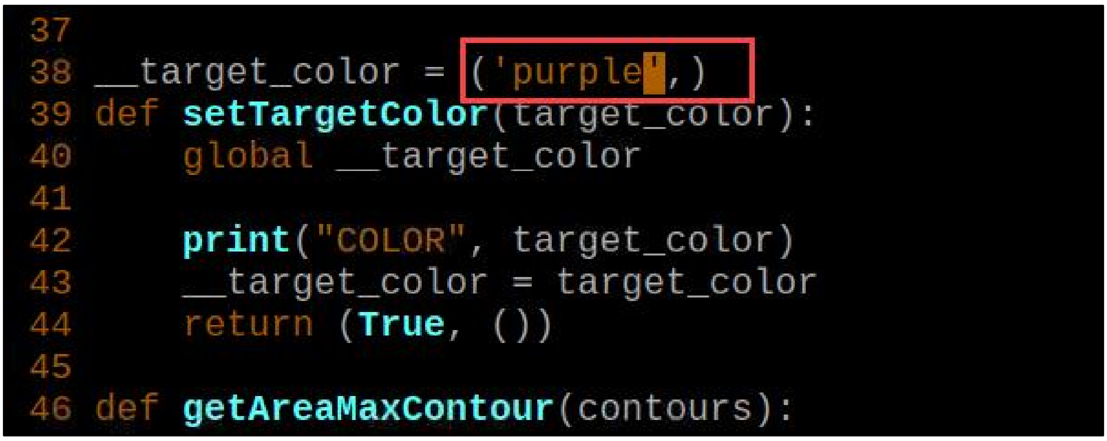
```
pi@raspberrypi:~ $ cd MasterPi/
pi@raspberrypi:~/MasterPi $ sudo vim lab_config.yaml
```

Step 8: After opening the color threshold program file, you can view the purple threshold parameter.



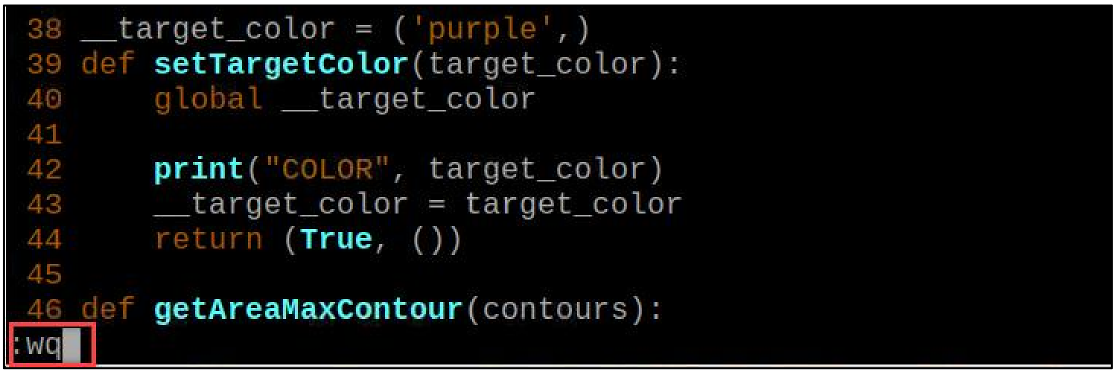
```
    - 150
    - 120
    - 160
  min:
    - 60
    - 0
    - 100
purple:
  max:
    - 255
    - 255
    - 114
  min:
    - 0
    - 145
    - 36
red:
```

Step 9: According to the steps of “4.2 Modify tracking color”, open the program file ,and enter the editing mode, and then modify “red” in “_target_color = (‘red’)” to “purple” as the figure shown below:



```

__target_color = ('purple',)
def setTargetColor(target_color):
    global __target_color

    print("COLOR", target_color)
    __target_color = target_color
    return (True, ())

def getAreaMaxContour(contours):
```

Step 10: After the modification is complete, press “Esc”. Then enter “:wq” and press “Enter” to save and exit.



```
__target_color = ('purple',)
def setTargetColor(target_color):
    global __target_color

    print("COLOR", target_color)
    __target_color = target_color
    return (True, ())

def getAreaMaxContour(contours):
:wq
```

Step 11: Refer to “2. Operation Steps” to start the game. Then MasterPi will track the purple object.