

Lesson 3 Introduction to the Servo Fine-tuning

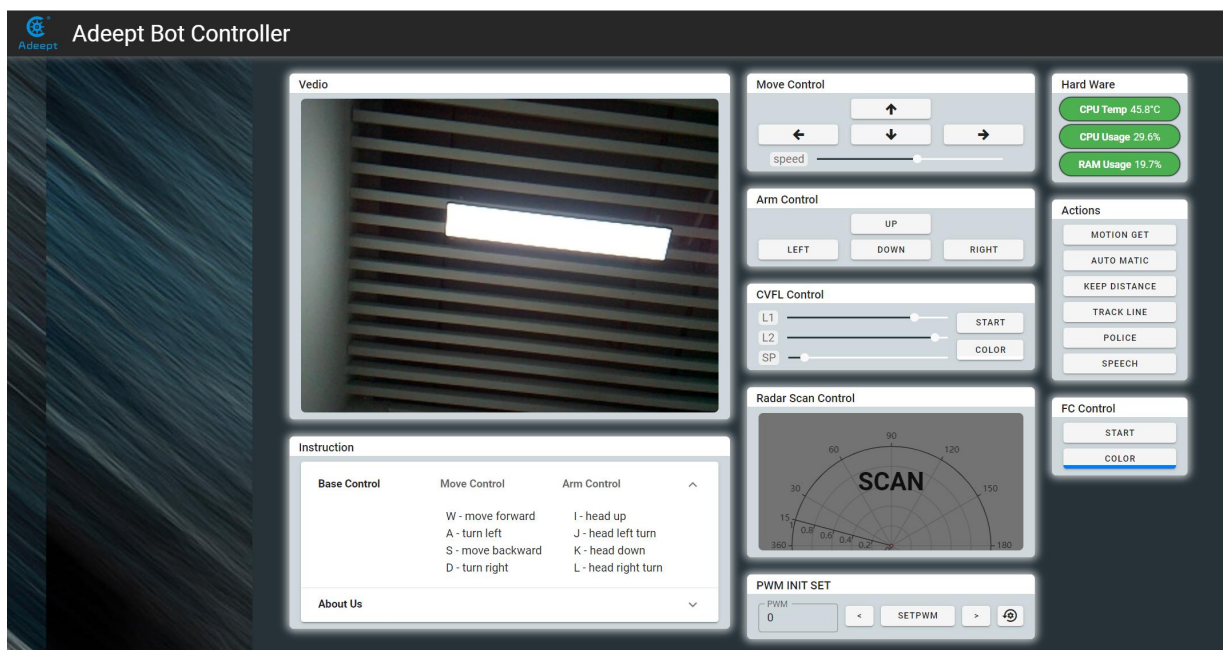
3.1 Function Introduction

Available to fine-tuning the default angle of the servo and correct the angle error caused by the servo installation. For the crawling robots, fine-tuning the default angle of the servo will improve their walking smoothness. For robot products with manipulators, it helps to adjust the initial angle of the manipulator chuck for correcting the error caused by the servo installation.

3.2 The fine-tuning function of the Servo

Run the servo fine-tuning program

1. Start up Raspberry Pi robot for about 30s-60s.
2. When PiCar-B is started up, enter the IP address of your Raspberry Pi on the Google browser of the mobile phone or computer, and access Port 5000, for example: 192.168.3.44:5000. Then the web controller will be displayed on the browser.



3. The module "PWM INIT SET" is the operating area of the fine-tuning function of the servo.
(It is recommended to restart PiCar-B before fine-tuning the servo)



4. Introduction to the servo fine-tuning module



- "PWM": for selecting the servo number and the No. 0 servo is selected in the figure. Click the box to enter the number or click the buttons of up and down in the box to select the

servo number that needs adjusting.



- "<": Adjust the selected servo to rotate counterclockwise slightly.
- ">": Adjust the selected servo to rotate clockwise slightly.
- "SETPWM": After adjusting the servo, save the current servo angle as the default value.
- "@": Initialize the default angles of all servos to factory setting.

5. Select the servo number to be adjusted. The servo number is the interface number of the servo connected to Robot HAT. For example, when the servo cable is connected to the Servo 1 interface, the servo number is 1.

6. After selecting the servo to be adjusted, clicking "<" and ">" to adjust the servo to a right position, and then click "SETPWM" to save.

3.3 Main Code

```
01 def configPWM(command_input, response):
02     global init_pwm0, init_pwm1, init_pwm2, init_pwm3, init_pwm4
03
04     if 'SiLeft' in command_input:
05         numServo = int(command_input[7:])
06         if numServo == 0:
07             init_pwm0 -= 1
08             T_sc.setPWM(0,init_pwm0)
09         elif numServo == 1:
10             init_pwm1 -= 1
11             P_sc.setPWM(1,init_pwm1)
12         elif numServo == 2:
13             init_pwm2 -= 1
14             scGear.setPWM(2,init_pwm2)
15
16     if 'SiRight' in command_input:
17         numServo = int(command_input[8:])
18         if numServo == 0:
19             init_pwm0 += 1
20             T_sc.setPWM(0,init_pwm0)
21         elif numServo == 1:
22             init_pwm1 += 1
23             P_sc.setPWM(1,init_pwm1)
24         elif numServo == 2:
25             init_pwm2 += 1
26             scGear.setPWM(2,init_pwm2)
27
28     if 'PWMMS' in command_input:
29         numServo = int(command_input[6:])
30         if numServo == 0:
31             T_sc.initConfig(0, init_pwm0, 1)
32             replace_num('init_pwm0 = ', init_pwm0)
33         elif numServo == 1:
34             P_sc.initConfig(1, init_pwm1, 1)
35             replace_num('init_pwm1 = ', init_pwm1)
36         elif numServo == 2:
37             scGear.initConfig(2, init_pwm2, 2)
38             replace_num('init_pwm2 = ', init_pwm2)
39
40
41     if 'PWMINIT' == command_input:
42         print(init_pwm1)
```

```
43     servoPosInit()
44
45     elif 'PWMD' in command_input:
46         init_pwm0,init_pwm1,init_pwm2,init_pwm3,init_pwm4=90,90,90,90,90
47         T_sc.initConfig(0,90,1)
48         replace_num('init_pwm0 = ', 90)
49
50         P_sc.initConfig(1,90,1)
51         replace_num('init_pwm1 = ', 90)
52
53         scGear.initConfig(2,90,1)
54         replace_num('init_pwm2 = ', 90)
```