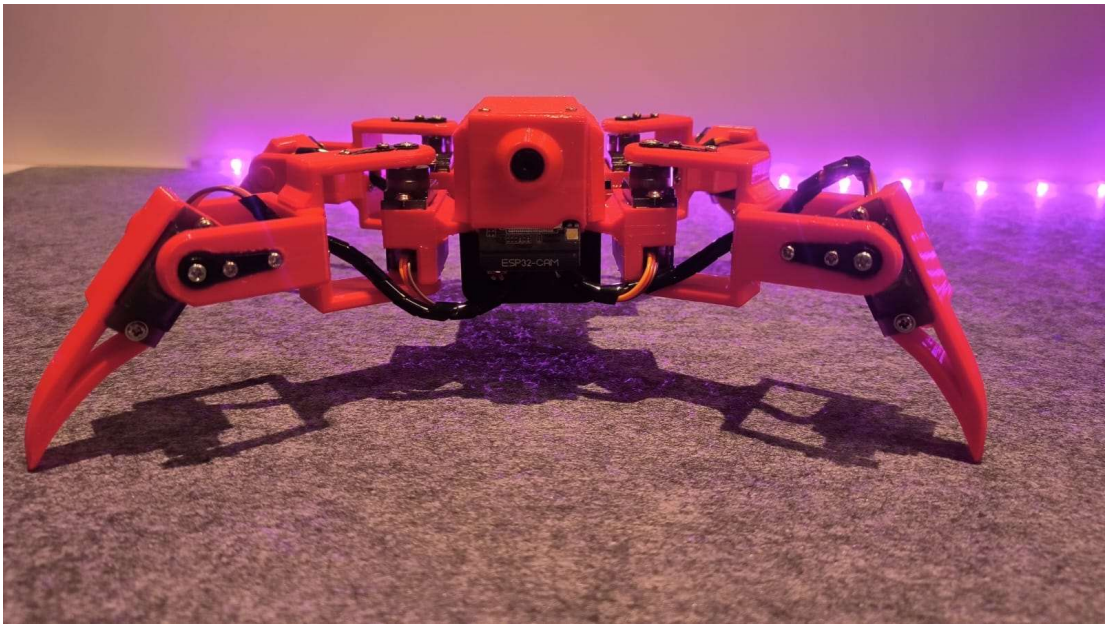
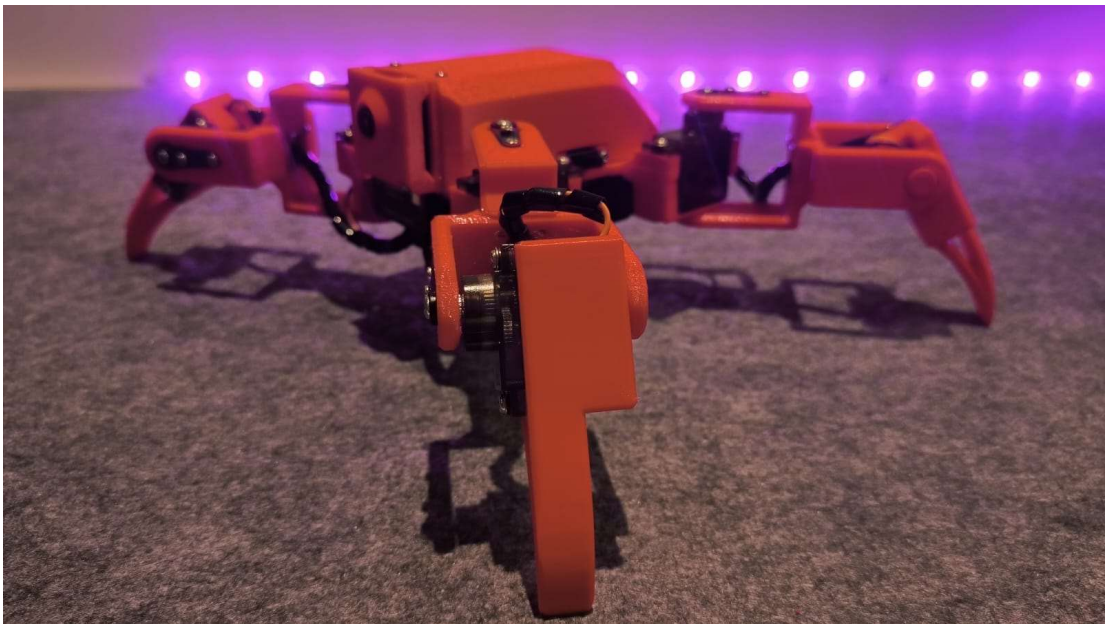
# delta3Robotics



## ESPiderman

Documentation

## Table of Contents

# 1 Introduction

ESPiderman is a project dedicated for people that want to start Robotics and Engineering. The robot is fully 3D-printed, and everything is published for free. The lowest price for the parts is around 22€.

To steer the robot, you need a Tablet with any browser installed. Ther ESP32cam sets up an access point after powering it up. After connecting the user interface and camera feed is shown.

If you get stuck somewhere you can ask me, write a DM on Instagram @delta3Robotics or send an email to chris@delta3robotics.de

## 2. Assembly instruction

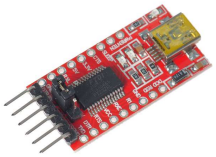
This chapter will guide you step by step through the building process. In general, building this robot takes between 3-4 Hours, don't get frustrated if your first attempt takes longer.

### 2.1 What to Buy

Not all parts can be printed, therefore please refer to the List below and buy the parts at any source you like.

Full amazon List: https://amzn.to/3wQY4Tq

| | Titel | Note | Amazon link | |
|---|---|---|---|---|
| 1x | ESP32cam | | https://amzn.to/48imAdp |  |
| 8x | Servomotors | The design should fit every kind of MG90s or MF90 and SG90 | https://amzn.to/3wpSuHu |  |
| 1x | PCA9685 | | https://amzn.to/48dSzeM |  |
| 1x | Battery holder | Battery holder for two 18650 LiPo batteries LxW 76x40mm | https://amzn.to/3uAwUzw |  |

| | | | | |
|---|---|---|---|---|
| 2x | 18650 Batteries | Get some of AMAZON or check Craigslist for used ones. Any 18650 can power the robot. | https://amzn.to/3I30PU6 |  |
| Bits and bolds | | | | |
| | Socket header | Two times with 8 Sockets or get the larger ones and cut them | https://amzn.to/3UHVklf |  |
| | Screws | Pan Head self-tapping Screws M2*4 and M2*8 | https://amzn.to/49hOEyN |  |
| | Jumper wires | Only must have is that one side is the female part | https://amzn.to/3TacOpz |  |
| 1x | Switches | Sliding button three position 2P2T DPDT SS23D32 F3E4 | https://amzn.to/3OIg1tu |  |
| 1x | FDTI Adapter | Check that the USP port of the adapter fits any cable you have to connect it to the computer | https://amzn.to/49CoDKs |  |
| 1x | Charger | | https://amzn.to/49woMyM | |

| | | | | |
|---|---|---|---|---|
| | Cable Conduit | | https://amzn.to/4bDlKuE | |

If you can wait Aliexpress normally get the pest prices. I added some Amazon links to all the parts. Aliexpress is cheaper but like this you work with the same items I uses for testing, and I get some bucks when you buy with these links.

You can start out using cheap SG90 servos. If you like you can upgrade, it later.

There is an affordable a plug and play alternative for the FTDI chip you can use, since I don't have any experience with that device I will refer to the good old FTDI chip in this document.

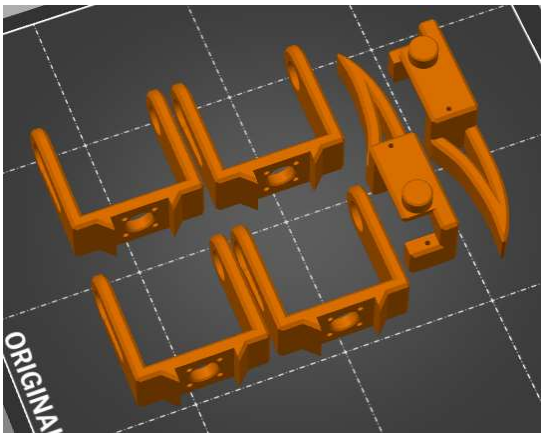
## 2.2 Printing

This chapter describes the process on how to print all the parts you need. Feel free to differ from this manual if you feel confident. Here I will list some settings that worked for me.

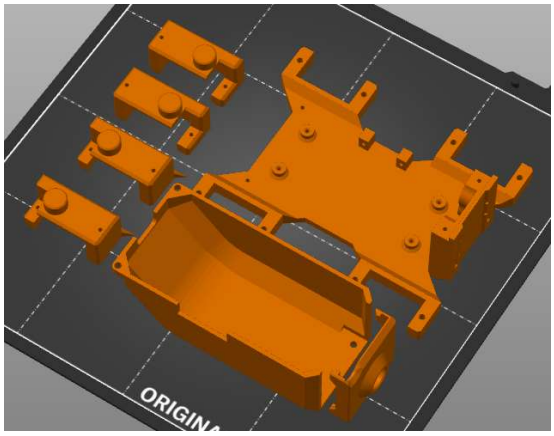
| | | |
|---|---|---|
| 1x | Head | |
| 1x | Body | |
| 2x | Cap1 | |
| 2x | Cap2 | |
| 1x | BodyLid | |
| 8x | LegUPart | |
| 2x | LegOuterPart | |
| 2x | LegOuterPart | mirrored |

Here you can see hoe to orientate the Parts for the print. I use my Prusa mini and split everything into 3 Prints. There is one print for Body, Lid, Head and the four Caps. Another

print holds two legs each with LegOuterPart, LegOuterPart (mirrored) and four times LegUPart. This print is done twice to get 4 legs.



*Figur 1 Print orientation for two legs*



*Figur 2 Build orientation for the remaining body parts*

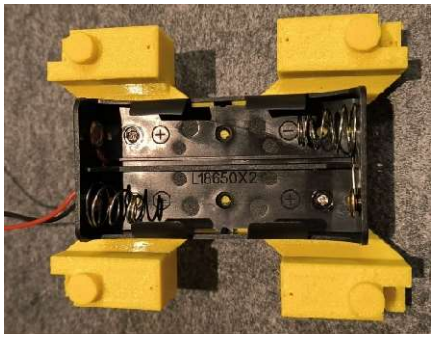
Support Structure is definetly needed. Print at least four parameters to have a rigid and durable part.

## 2.3 Assembly

Now we take care for the mechanical assemly.

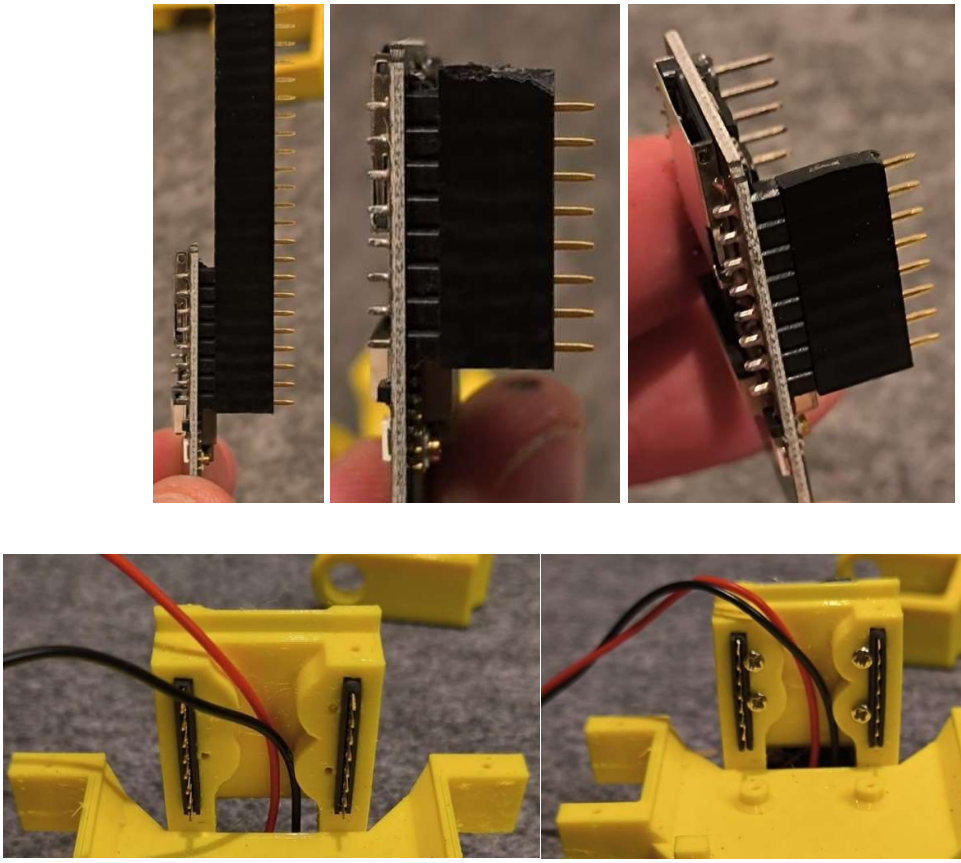
### 2.3.1 Mechanical Assembly

First mount the batteriy box unter the body. The cables must be at the front. Under the neck section is space to guide them into the robot. Take two M2x4 srews and use the dedicated holes. If your batteriy box differs in layout try to screw it directly into the plastic.



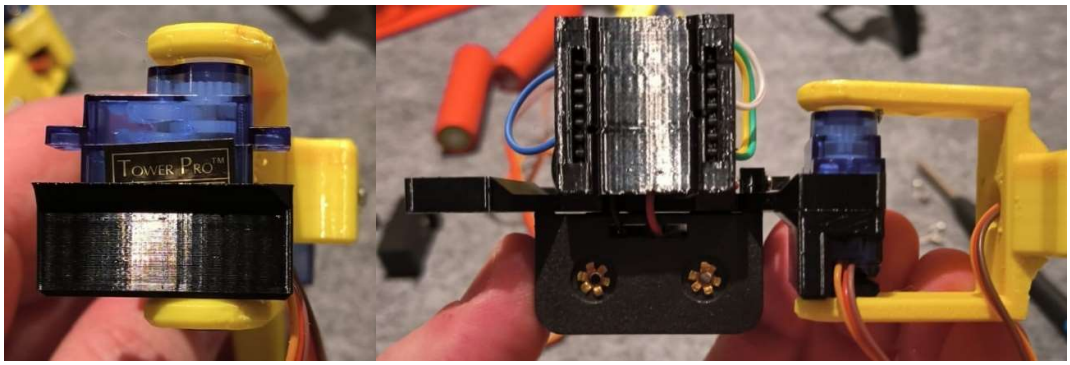

*Figur 3Body with battery box*

Then take the socket header and cut it to length. You will need two of them. Ther is the option to buy two times 1x8 socket header but if yu buy the log ones and cut them, you'll end up cheaper. After you cut it to length tae a file and remove the access material. Important is, the socket headers that are pushed into the body to take up the ESP32cam afterward must be sled in very carefully. If the pushing needs to much effort try to work on the slots, otherwise the bodypart might breake. After pushing them in completely secure them with two M2*4 each.



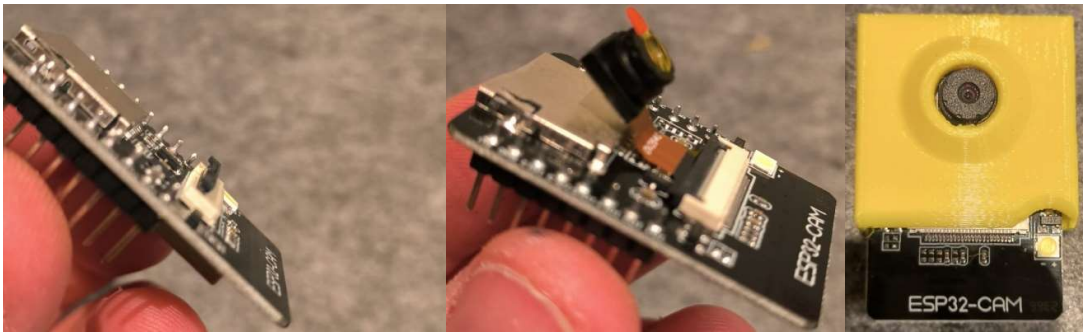
*Figur 4 Mounting the socked header*

Update: I split the body to save printtime and support structure. The Caps can be fixed with the screws used to mount the servos. Maybe you need to take longer 2mm screws if the ones that come with the sevos are too short. M2x12 work like a charm. Slide the Caps onto the servo and then push everything in place before tigthen it up with two screws each.

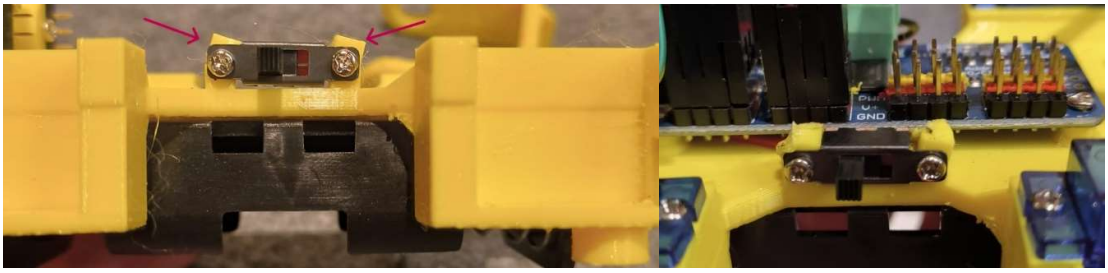



*Figur 5 Mounting the first serv motors with caps*

Now we can assemble the ESP32cam. Open the flap by pulling up the lid. Slide in the camera module all the way and close the flap again. The printed part for the head can be gently pushed onto the ESP. Therfore, push it all the way up and then click it onto the controller.  The ESP can be put in place now. Gently slide all 8 pins into the socket header.

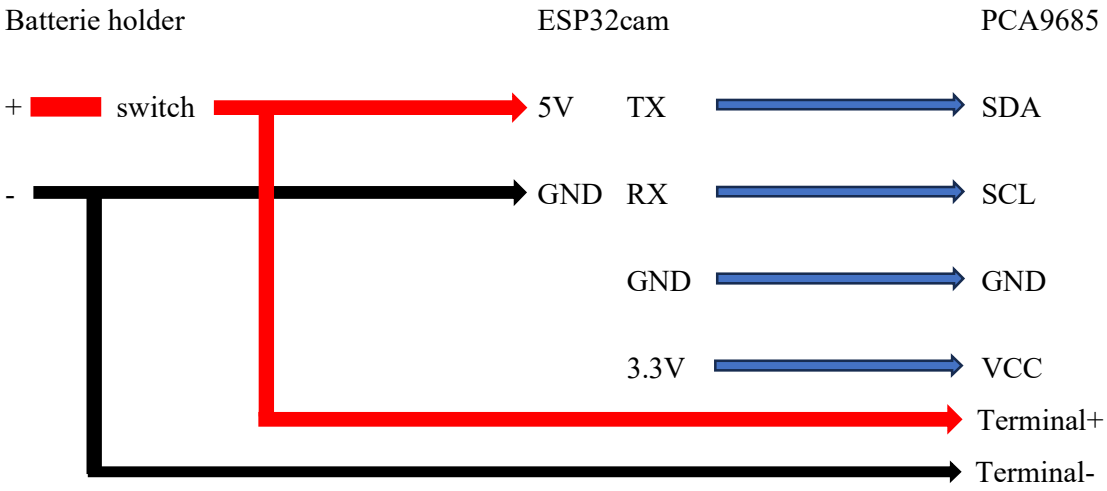



Now take the swith and screw it into place. The mouning is super fragile and might brack if you are not careful enough. Gently fix the swith with 2 M2x4 screws. If the mount is damaged like senn in the pictures below one can use a soldering iron to repair it.

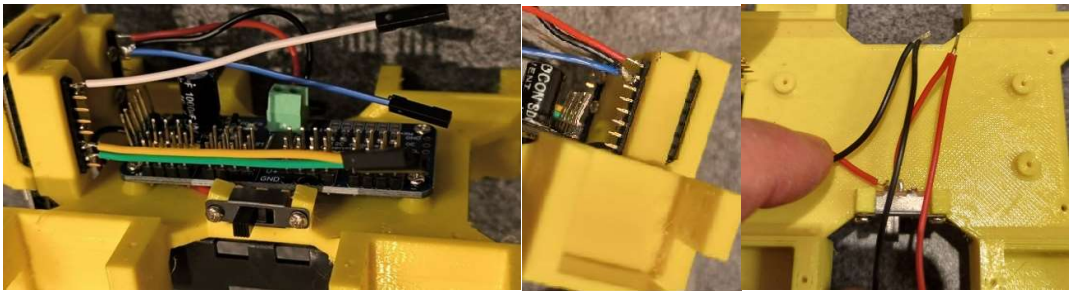


### 2.3.2 Wireing

I get a lot of questions about the voltage level for the ESP and PCA. 7.4 Volts is not too much for the devices. The internal power management supports up to 15V on both chips. For the ESP that holds true for the 5v pin and for the PCA it is the terminals, the logic side does not support that much current, take the 3.3 V output pin from the ESP.

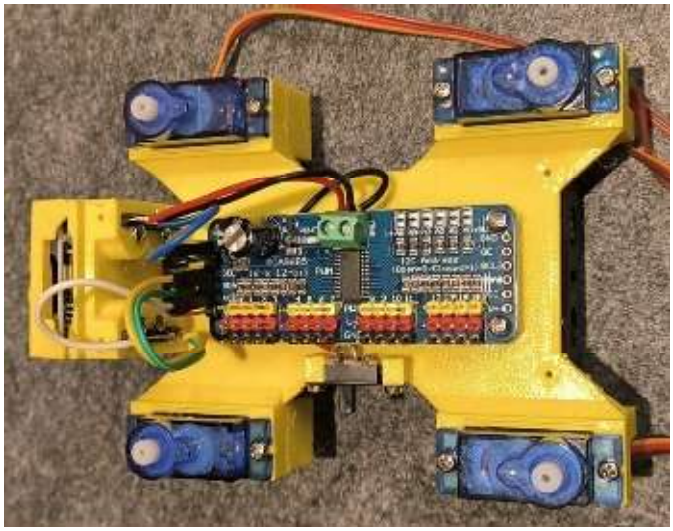



All the connections on the socket header need to be soldered. The connections on the PCA can be plugged with jumper ends, but to do so the Pins on the PCA must be bend upwards, if the pins are the ones with a 90° angle.

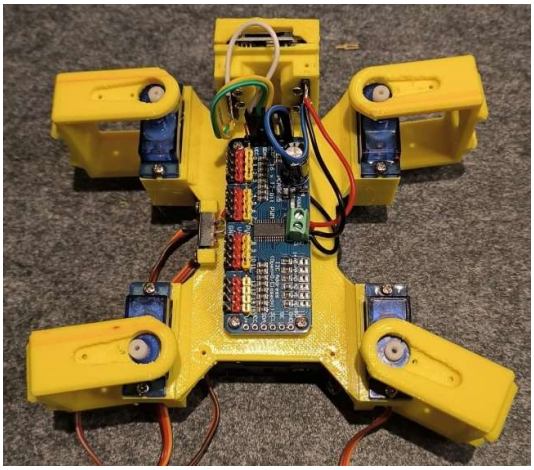


Fix the PCA9685 with 4 M2x4 screwys and connect the jumperwires according to the schematic.

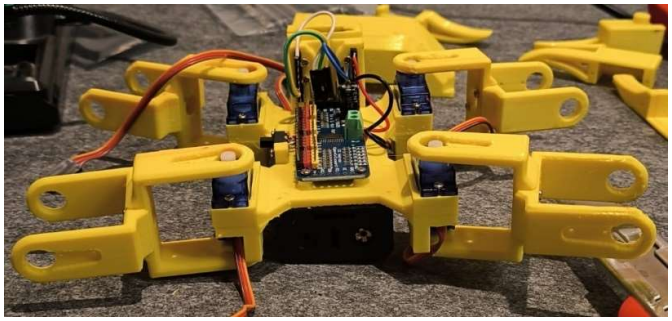
Mount the first 4 servos like shown in the foto.



Put on the first LegUPart per servo. Do not attach the servo horns for now.



Now Put on another LegUpart per leg. Make sure that the spot for the servo horn points to the Front for the front legs and backwards for the back legs. Fix each connection with 4 M2x12 screw.



Now slide one servo into each LegOuterPart and fix it with screws.


Mount the four remaining parts to the legs.

### 2.3.3 Location for Servos on PCA9685

Logical counting start at 0. The first servo is 0 and the port on the PCA for servo 0 is called 0, too. For the legs we count in mathematical quadrants, therefore leg 0 holds the servos 0 and 1. Leg 1 holds the servos 2 and 3 etc.

After everything is connected, screw the lid on top to hide all the nasty cable connections. Before you screw the lid on fully, make sure that there is enough cable length for the legs to be fully operated.

### 2.3.4 Calibration

First thing to do is to set the servos on neutral (90°), I personally like to use a servo tester, a cheap device. If you do not have one, here is what to do! If we upload the code as it is the robot will start in its idle position which is not the one, we need!

We do not want it to be like that, so open the code and set legIdle = korr = korr2 = 0

```
63  //Angle of the outerLegParts
64  int legIdle = 30; //  for calibaration 0 for usage 30
65  //value that can be used to manipulate the idle posution after arming
66  int korr =25; //0for calibration 25 for usage
67  //Offset to manipulate the center of gravity for each height
68  int korr2 = 5; //0 for calibration 5 for usage
69
70  //Compensation values to calibrate the servos
71  int comp0=3;
72  int comp1=-5;
73  int comp2=3;
74  int comp3=0;
75  int comp4=3;
76  int comp5=0;
77  int comp6=0;
78  int comp7=-12;
```

*Figur 6 orig snippet from my code*

Further make sure that all the compensation values (comp0-comp7) are set to zero.

```
63  //Angle of the outerLegParts
64  int legIdle = 0; //  for calibaration 0 for usage 30
65  //value that can be used to manipulate the idle posution after arming
66  int korr = 0; //0for calibration 25 for usage
67  //Offset to manipulate the center of gravity for each height
68  int korr2 = 0; //0 for calibration 5 for usage
69
70  //Compensation values to calibrate the servos
71  int comp0=0;
72  int comp1=0;
73  int comp2=0;
74  int comp3=0;
75  int comp4=0;
76  int comp5=0;
77  int comp6=0;
78  int comp7=0;
```

*Figur 7 code with adapted values*

#### *2.3.4.1 Upload the code.*

The internet is full on how to download the Arduino IDE and how to connect the ESP32cam to the computer to upload code! Therefor I will not explain it further. If you never programmed an ESP32cam I would suggest doing some testing with the Webserver example sketch you can find after integrating all the stuff you need to program ESP32cam with the Arduino IDE. It is not easy to do for the first time, but you can do that!

As soon as you are done with your testing, and you are confident enough open the adapted code and upload it to the controller. Put the controller into the socket header and power the robot on. The controller sets all servos to 90°, that is exactly what we want to mount the servo horns. Put the links into the correct position and mount the horns, after mounting all the horns the robot should look like this.

*Figur 8 Robot pose for calibaration*

This will not be in a perfect 90° angle, do the best you can, the rest will be done in code. For this we must play a bit with the values comp0-comp7. Take your time and find values to drive the servos exactly to 90°. For this adapt values, I would recommend changing one value at the time, upload, power on the robot and check if the link is set to 90°.

As soon as you are satisfied with your calibration, set korr, korr2 and legIdel back to the values mentioned in the comment next to each value and upload it again. The robot should look this now.

*Figur 9Robot idle position*

Congratulations! Your robot should be calibrated now.

## 2.4 Connect a client to the access point.

To connect a client to your webserver which is running on the ESP, power on your bot. After some seconds Take a Tablet (right now only touchscreen devices are supported, and I would highly suggest taking a tablet, otherwise the buttons are way to small :/ ) and connect to the Wi-Fi “ESPiderman” the password is “delta3robotics”. Btw you can change that easily in the code.

After the connection is established, there will come a warning since your device can not connect to the internet. Little side tip, if your device switches automatically to other Wi-Fi networks due to quality of connection it will switch out of “ESPiderman” so turn that setting off, I cannot do it in the code. Now you can enter the following IP-address to any browser: 192.168.4.1 and maybe hit the refresh button. Safe the tab as favorite so you do not have to input the IP every time.

If there are any questions left feel free to send me a DM, @delta3Robotics on Instagram is my main channel!

And that is it! You built your own robot! I am proud of you!

## 3. Trouble Shooting

### 3.1 Shaking / jittering servo motors

Due to bad resolution and backlash, cheap servos tend to jitter if no load is applied. This will most likely happen with SG90 but can also affect more expensive servo types. Ther are two solutions that may work.

#### 3.1.1 Lower the legs to have contact in idle position

This solution is implemented in the code you have. By setting the legs to legIdle the feet should have contact with the ground. If this is not the case, you have to increase this value a bit to stop it from shaking.

#### 3.1.2 Increase the friction inside the affected links

You can take a piece of cloth or sponge and put it inside the link. Like this you increase the friction, and the backlash will not resonate, so the jittering stops.

### 3.2 Flipped camera feed

Some camera modules seem to be flipped. From the outside you cannot see it but the camera feed at the webserver is flipped upside down. To change that you can use inbuilt functions in the code.

```
860   sensor_t * s = esp_camera_sensor_get();
861   //flip the camera vertically
862   //s->set_vflip(s, 1);          // 0 = disable , 1 = enable
863   // mirror effect
864   //s->set_hmirror(s, 1);          // 0 = disable , 1 = enable
865
866   int res = 0;
```

By uncommenting the corresponding line, you can flip and/or mirror the camera feed.

## 3.3 Servos are not working

The robot is fully assembled but the motors are not driven? Sadly, this can have several issues.

### 3.3.1 Wiring made incorrect

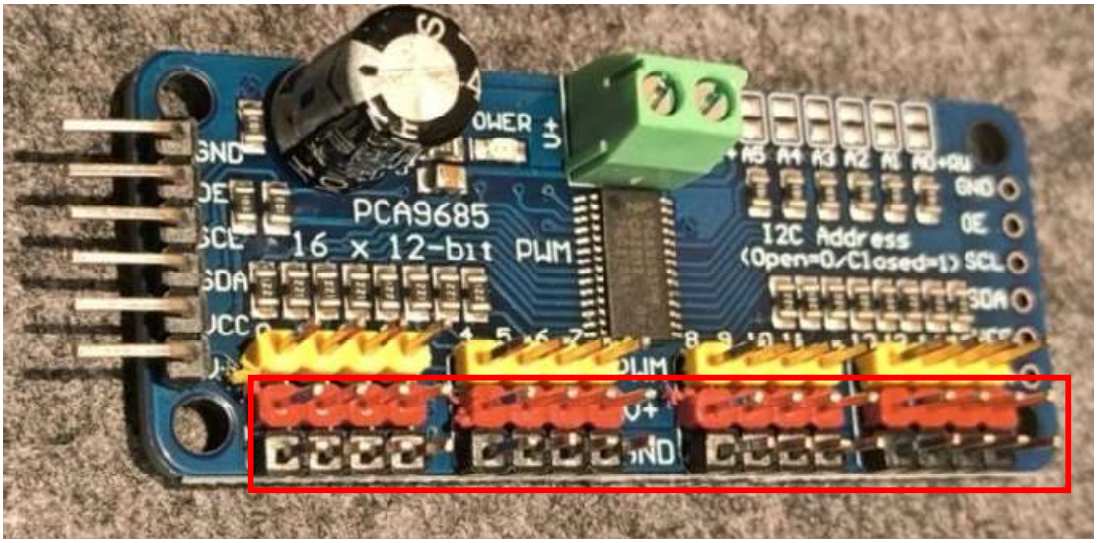
Please double check that the RX/TX/SDA/SCL are connected like described in Section 2.3.2 Wireing. If the SDA and SCL lines are crossed ther will be no communication between PCA and ESP. Make sure that works.

### 3.3.2 Insufficient Power supply

Charge the batteries! One servo can pull 400mA peak. That means if all servos are actuated the robot needs more that 3.2 A peak! The 18650 can do so, but only if charged fully. A good sign to charge the batteries is if it worked before or everything works fine until the servos should start working.

### 3.3.3 Broken PCA Board

Check that the Voltage between the V+ and GND pins on the PCA. For that you need a Multimeter. Power on the robot and measure at any red and black pin.




Most PCA9685 boards have a transistor that blocks the voltage if the logic part of the PCA is not active. Sadly, a lot of board out there exactly this transistor is broken. If the robot is powered on but there is no voltage > 7.4 V at the indicated pins your board is most likely broken, and you must replace it.

### 3.4 Buttons are not working

You powered up the robot and connected to the webserver, but the robot does not respond? Check if the flashlight is working. If the flashlight is working, there is a problem with the I2C connection. Why? Because the flashlight is the only device that is integrated directly into the ESP, everything else is send from the microcontroller to the PCA board.

If the flashlight does not work as well, you most likely use the wrong device to control the robot. Right now, the buttons only work with touch devices. This is on my issue list but with a very low priority.

### 3.5 Slippery feet

You'll notice that the feet are slippery. To increase the friction and reduce slipping you can apply hot glue onto the tips of the feet.

## 4. Acknowledgements

This section shall be used to thank people that put in work and shared their findings. Like this, we all can make this project great.

### 4.1 Muxiddin

MF is one of the first Instagram user to join this project. The robot was printed and assembled very fast. Parts were modified to fit other components like bigger switches and an ESP with a bigger camera. Also Muxiddin came up with the solution to insert pieces of sponges into the links and therfore decreasing the jittering of servo hysteresis.