



UFACTORY

UARM Swift Pro

Quick-Start Guide

V1.0.12
July. 2017

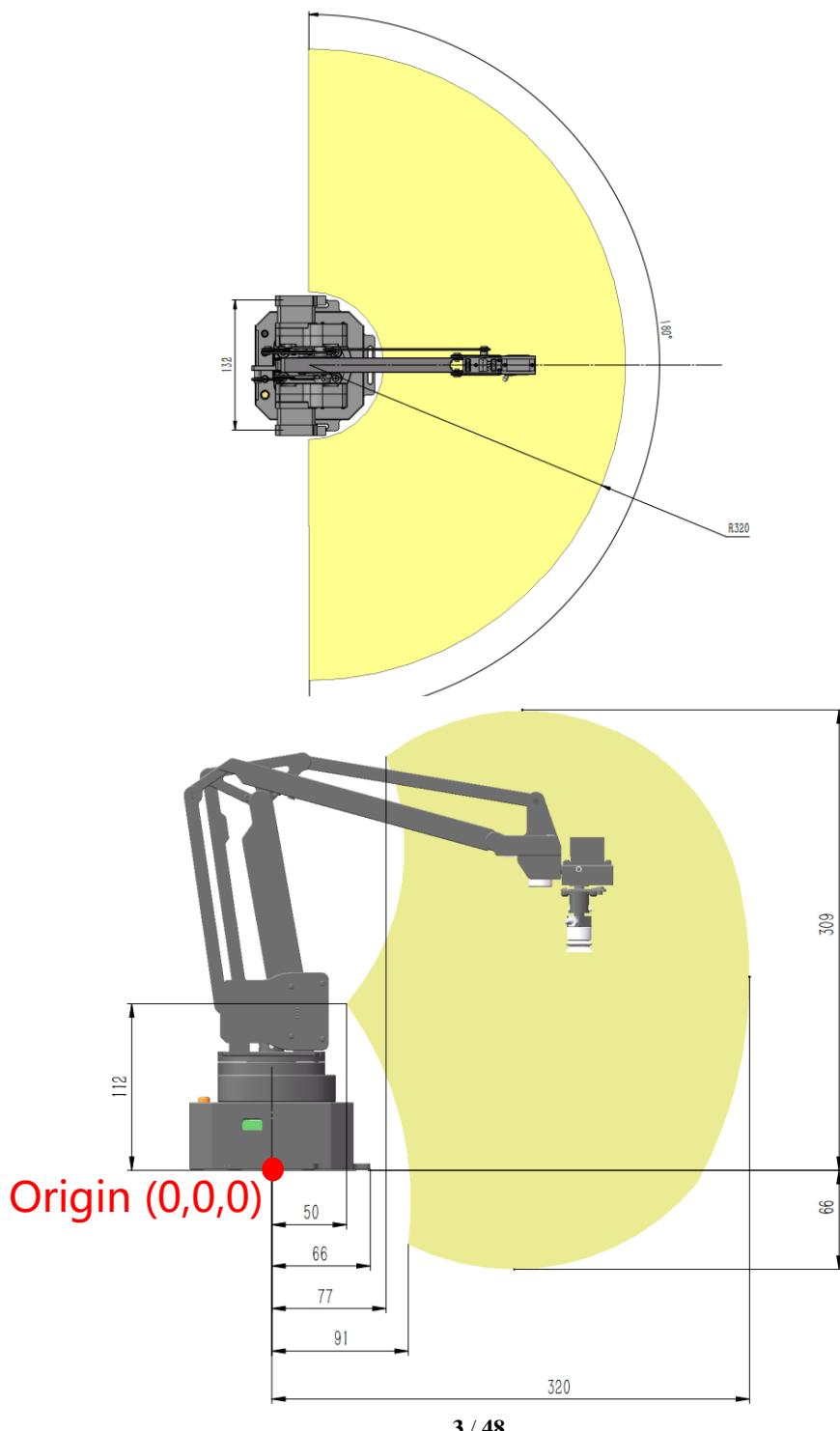


Contents

SAFETY INSTRUCTIONS	3
PRODUCT OVERVIEW	4
1.REFERENCE FRAME	4
2. BUTTONS & INDICATOR LIGHTS.....	5
3.EXTENSION DESCRIPTION.....	6
HARDWARE INSTALLATION.....	7
1. SUCTION CUP (DEFAULT).....	7
2. LASER	9
3. 3D PRINTING	11
4. SWIFT GRIPPER.....	15
5. SWIFT UNIVERSAL HOLDER	17
6. SEEED GROVE MODULES.....	19
7. OPENMV MODULE	22
OFFLINE LEARNING MODE	28
SOFTWARE: UARM STUDIO (WIN/MAC)	30
1.DOWNLOAD UARM STUDIO	30
2.DEVICE CONNECTION	30
3.DRAWING/LASER ENGRAVING	31
4.3D PRINTING	32
5.TEACH & PLAY: LEARNING MODE	35
6.BLOCKLY: VISUAL PROGRAMMING.....	37
FOR DEVELOPERS	39
1.COMMUNICATION PROTOCOL	39
UARM COMMUNITY	47
RELEASE NOTE	48

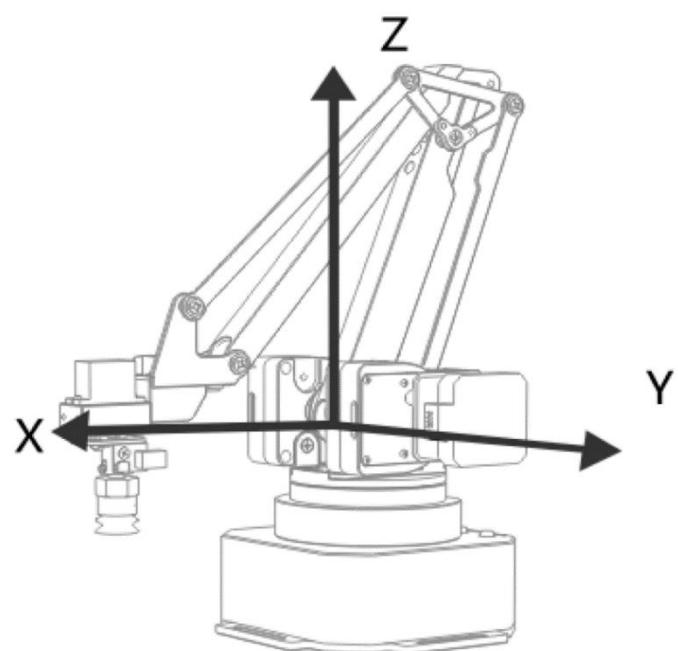
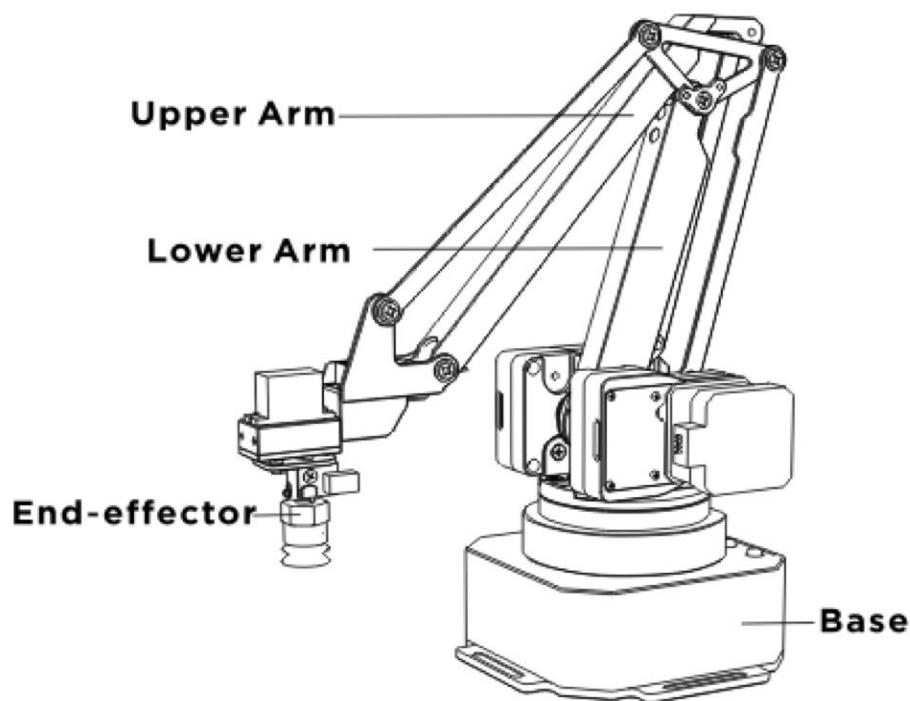
Safety Instructions

1. Please don't put your hands between the arms when uArm is moving.
2. Please use the official power supply for safety reasons.
3. Please clear a space for uArm, in case of knocking down anything.

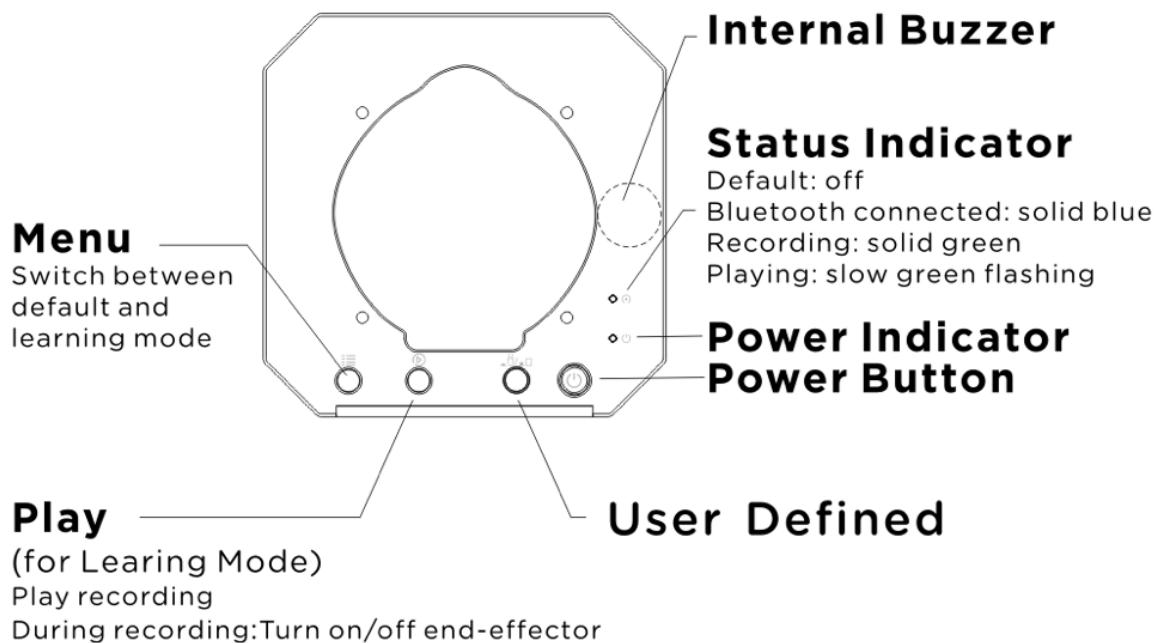


Product Overview

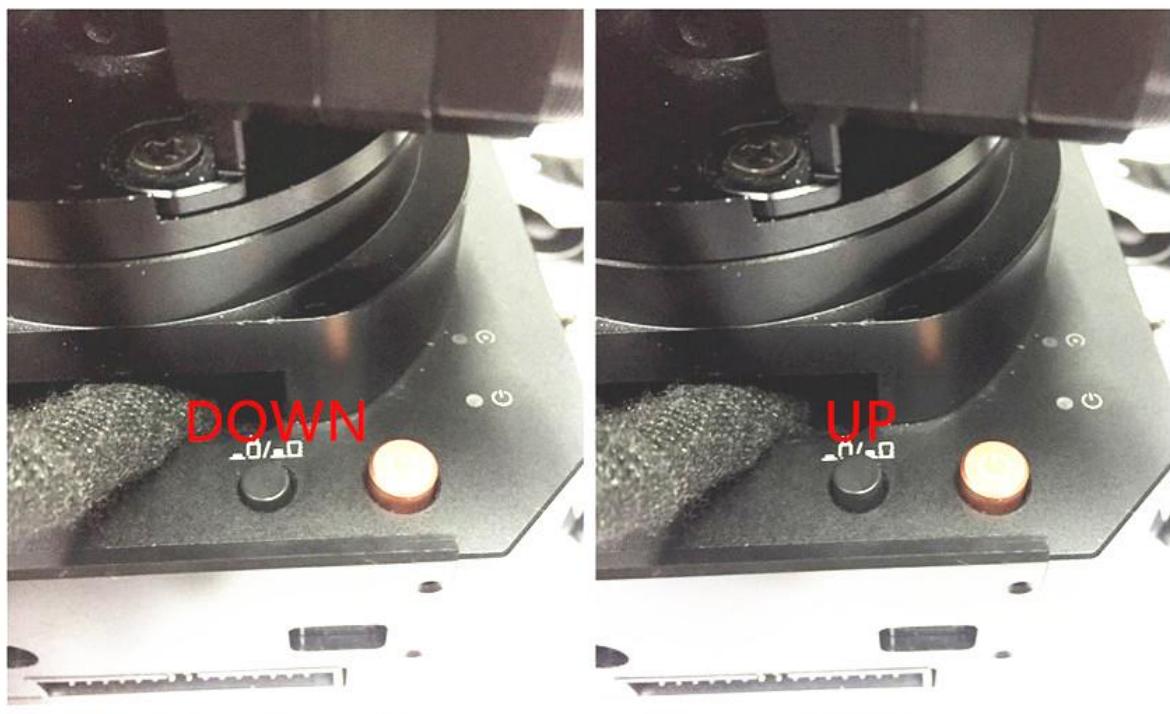
1. Reference Frame



2. Buttons & Indicator Lights



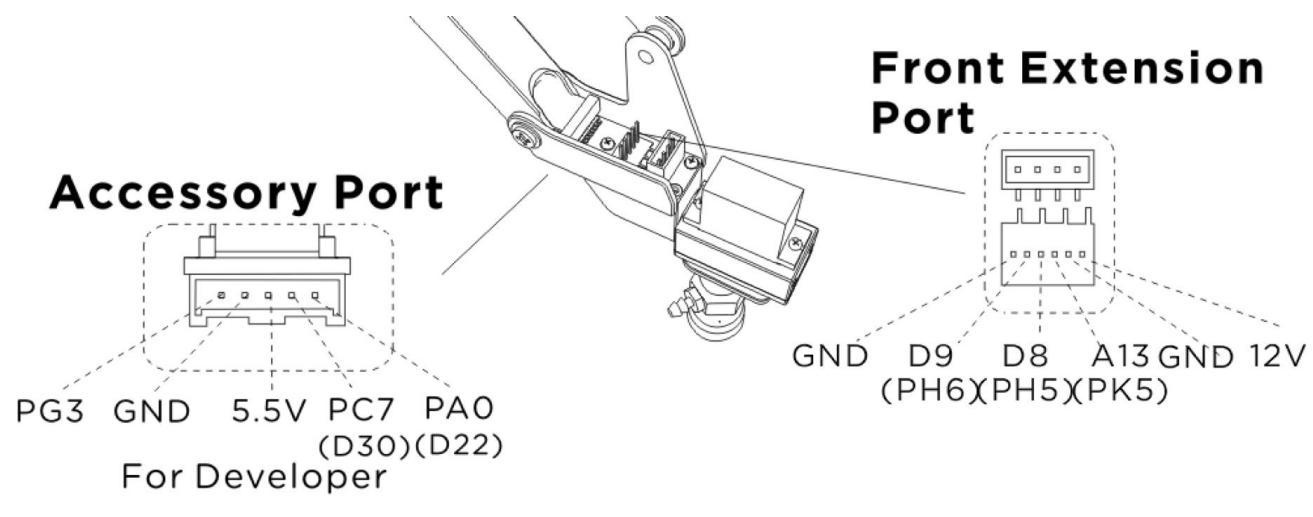
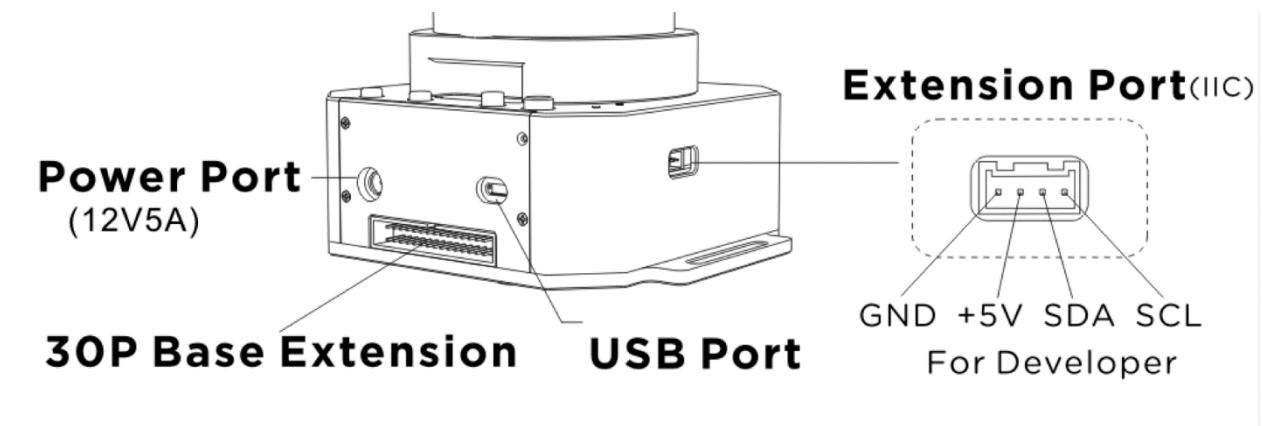
Caution: By default, the user defined button is for switching between Bluetooth and USB mode. Please ensure the button is UP while communicating with uArm via USB.



Bluetooth

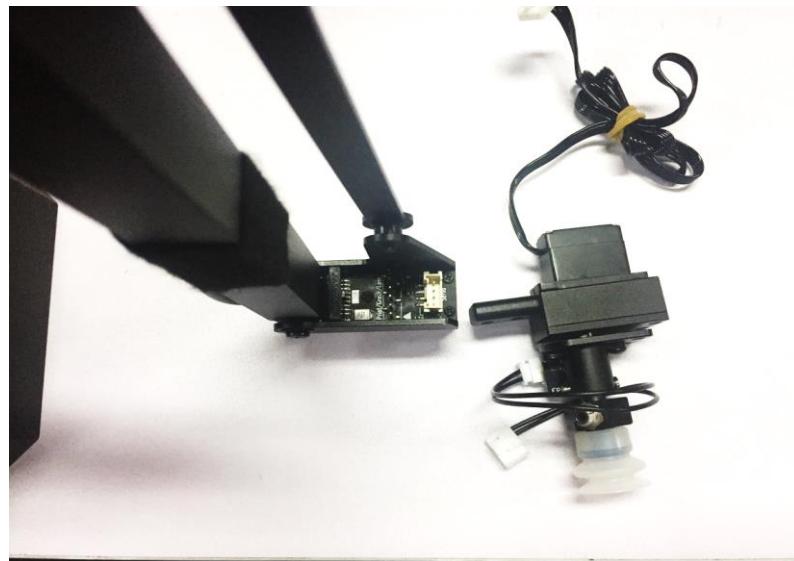
USB

3.Extension Description



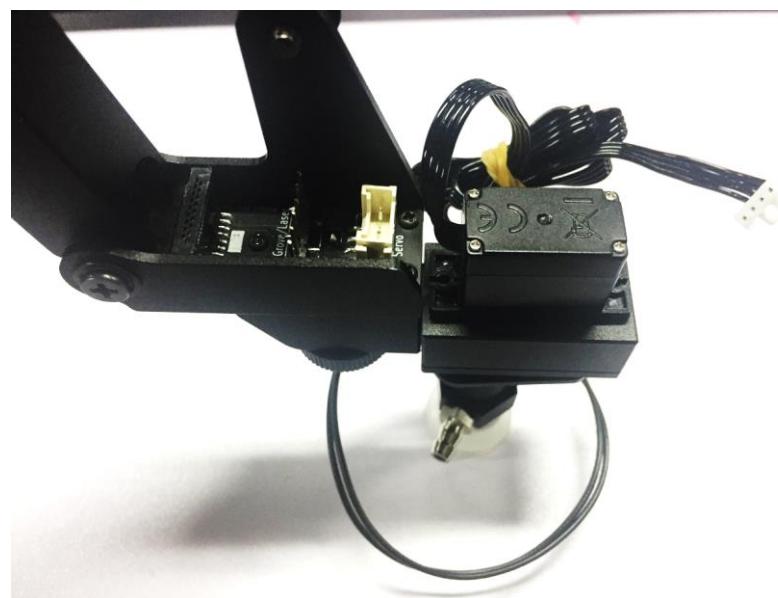
Hardware Installation

1. Suction Cup (Default)



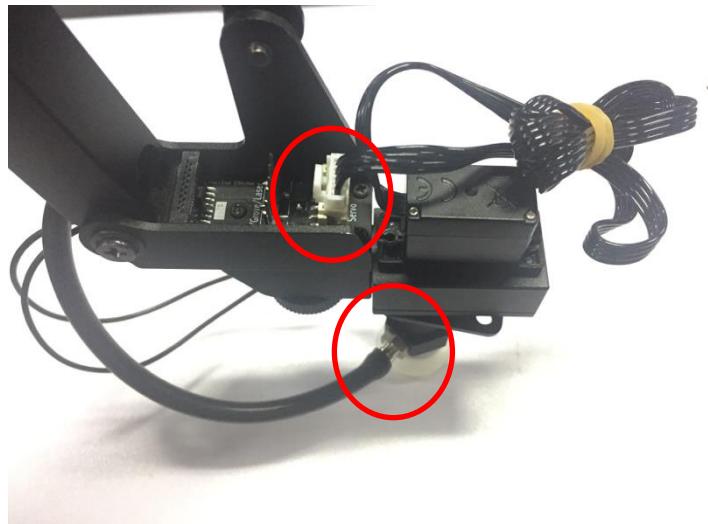
Preparation

Step 1: Install the suction to the end-effector and lock the nut tightly



Note: Similarly, if you want to uninstall suction cup, unlock the nut.

Step 2: Plug the wire of 4th axis motor, suction tube and limited switch

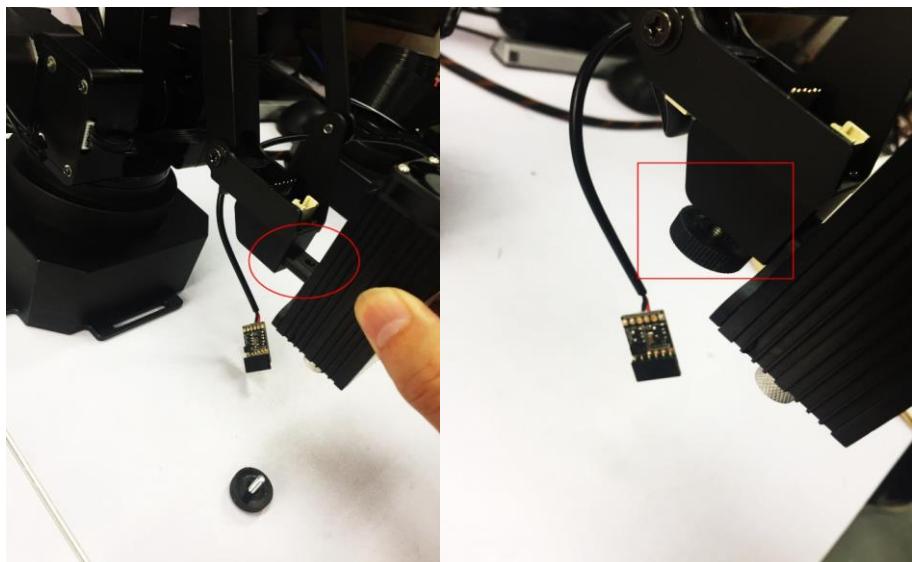


2. Laser

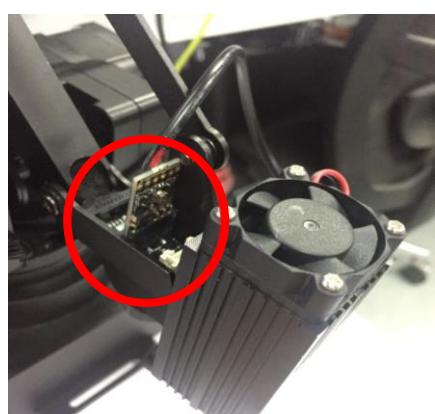
Preparation (Required Parts: Laser head, Thumb nut)



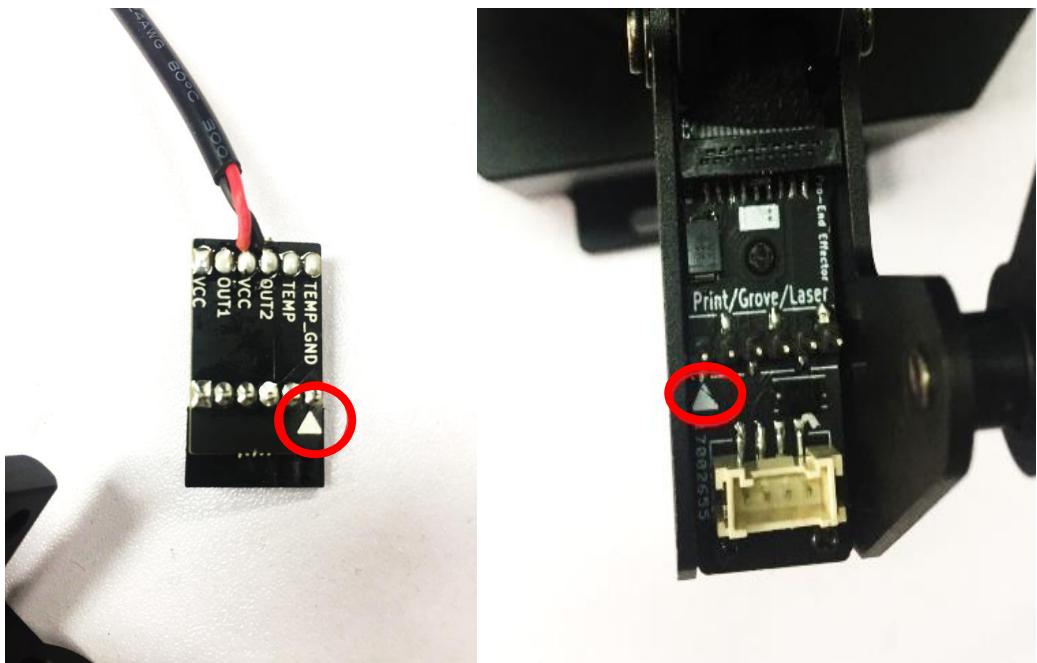
Step 1: Install the laser head and lock the nuts tightly



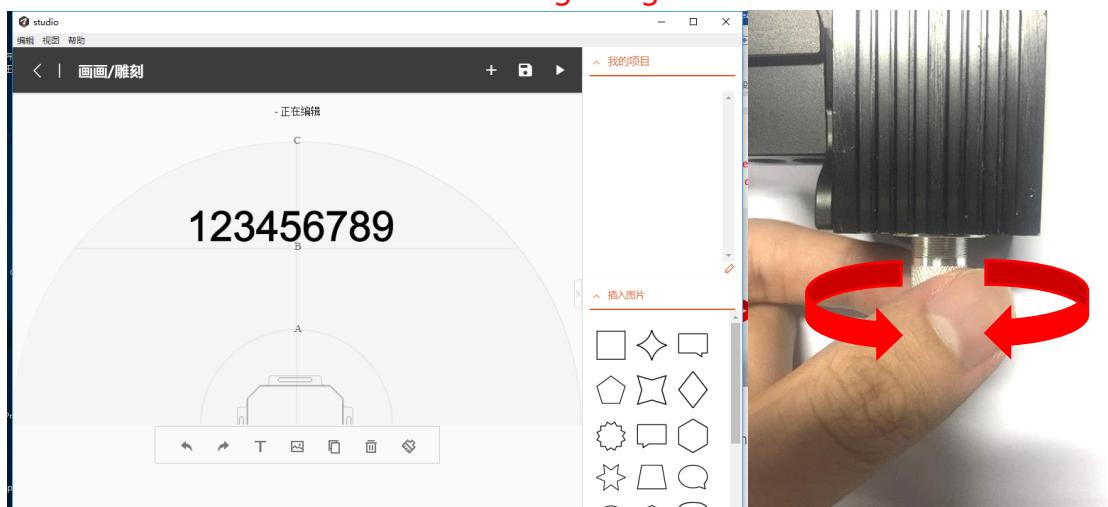
Step 2: Plug in the board of laser to the end-effector



(Please pay attention to the direction)



Caution: If the laser could not engrave the paper, please open the uarm studio and start the laser engraving, then focus adjust the lens of laser slowly. Please do not touch the light of laser during the engraving.

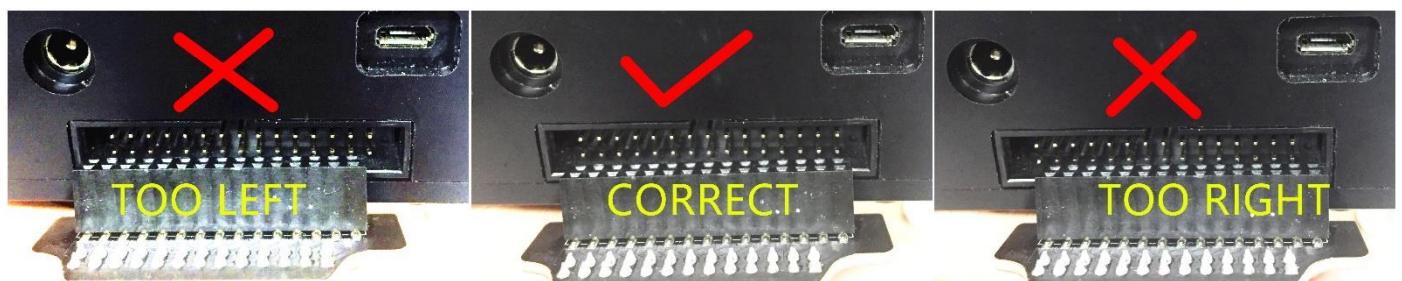
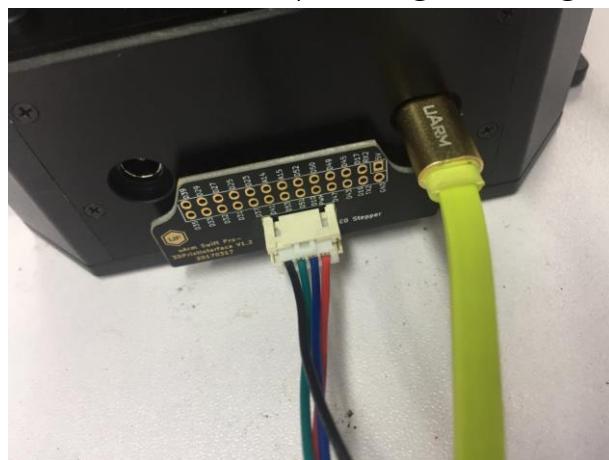


3. 3D Printing

Step 1: Install the 3D printing extruder and locked the nut tightly

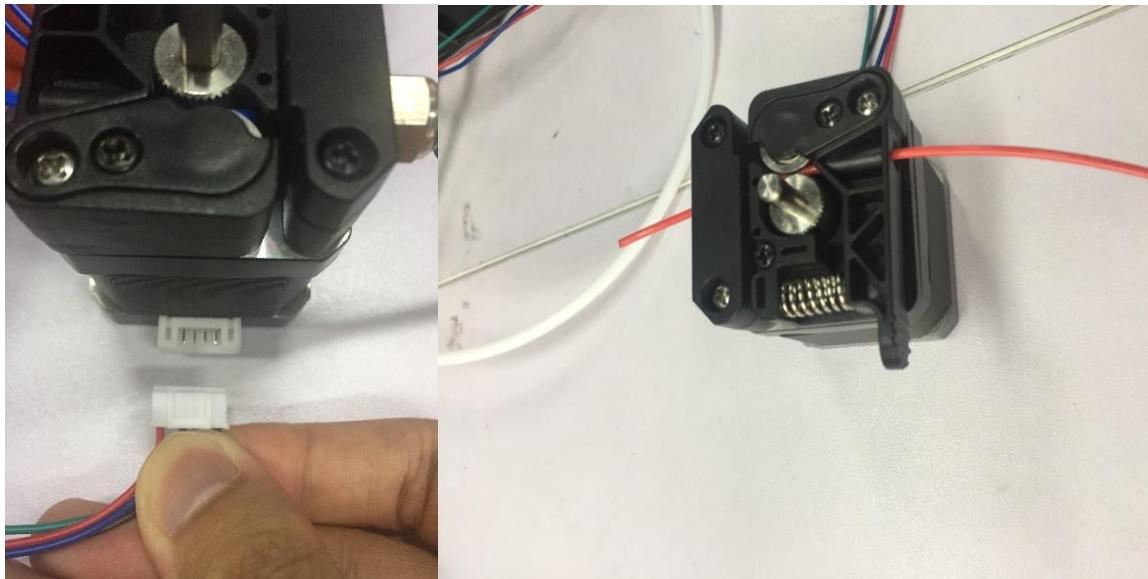


Step 2: Install the 3D printing feeding system



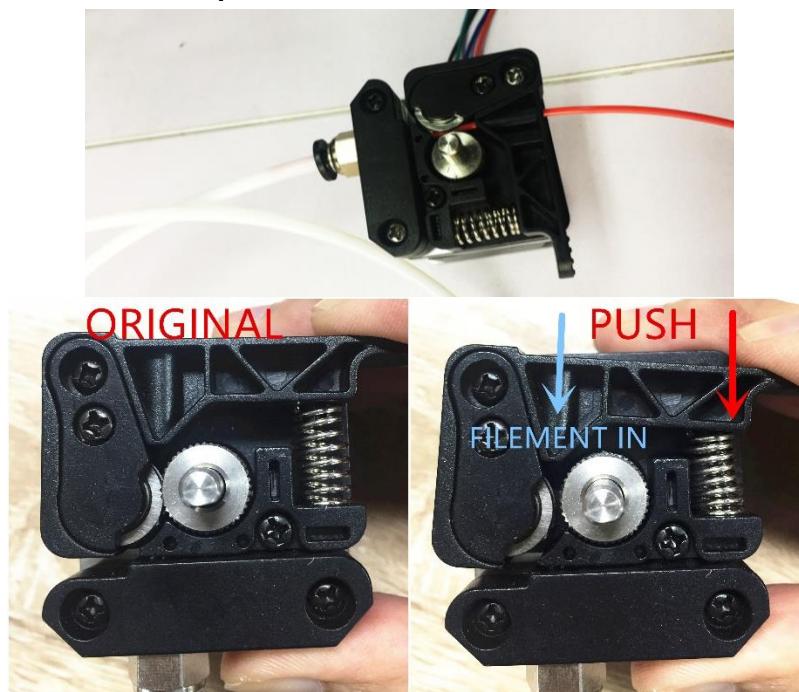
Caution: Please ensure the connection is correct. Or the computer wont recognize the uarm.

(Connect the motor with the extension board with the 4-color cable)



(Feed the PLA material we offered into the feeding system)

Step 3: Install the PTFE tube



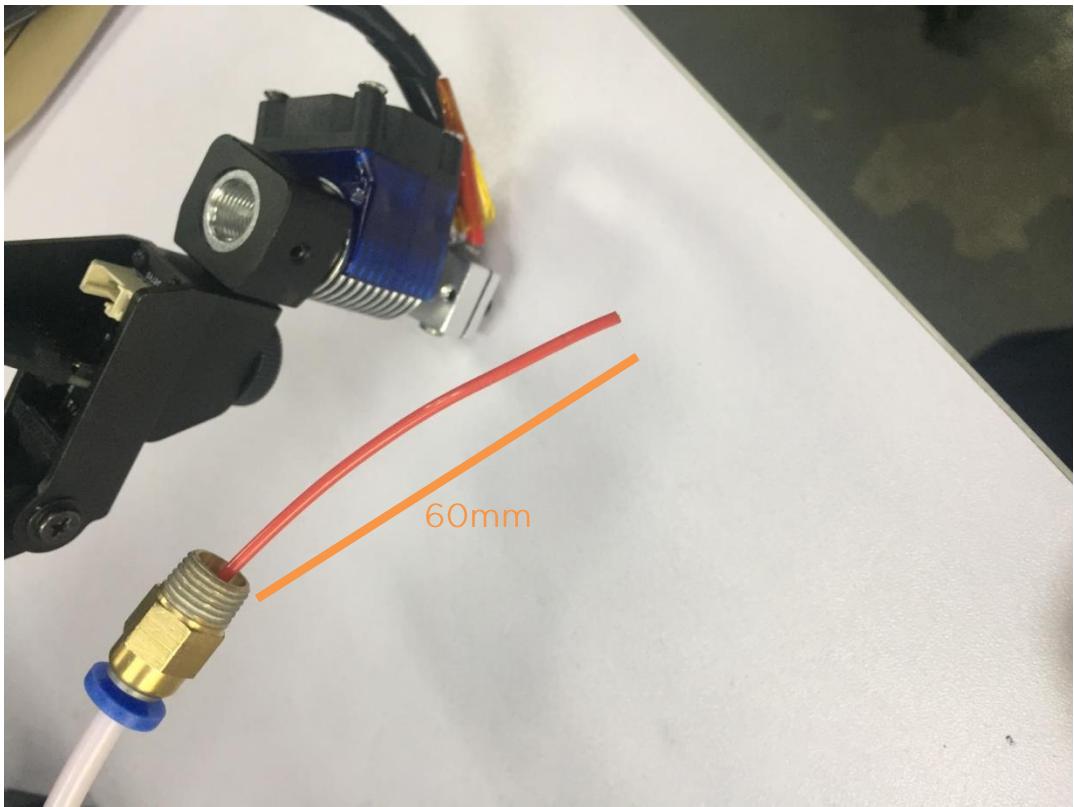
Feeding the filament



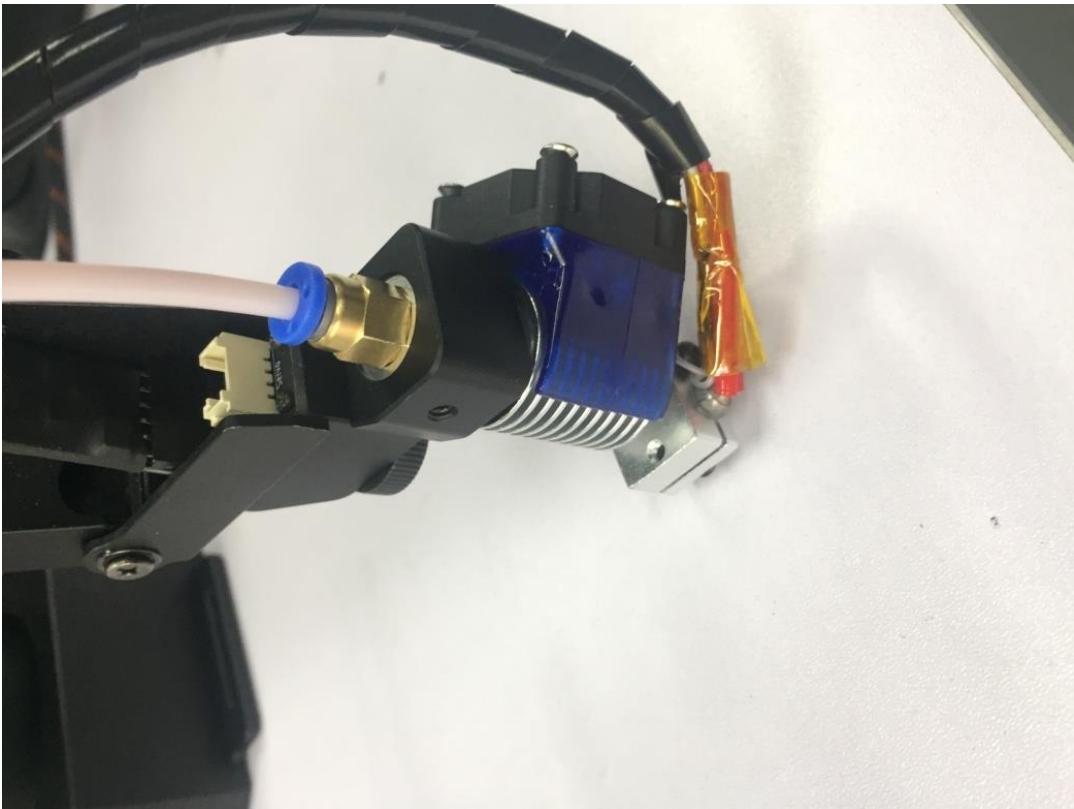
Installing the tube

Step 4:

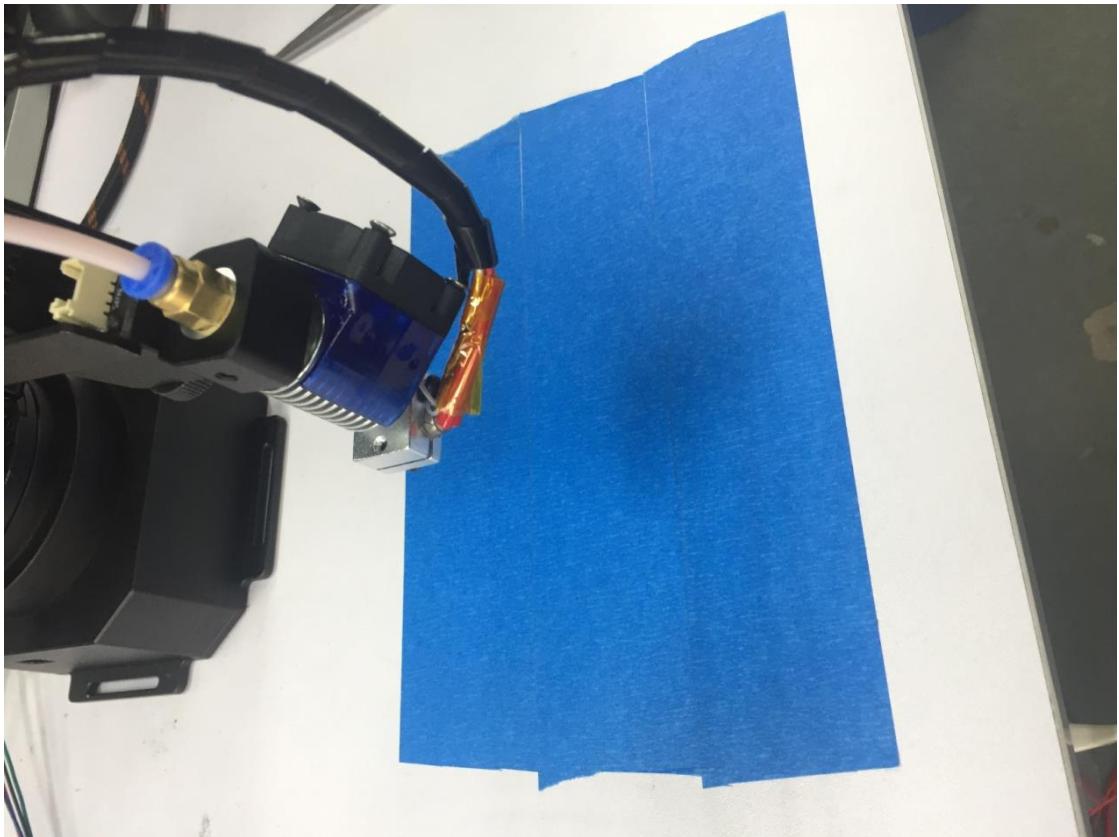
Keep feeding the material until it's 60mm out of the other side of PTFE tube.



Step 5: Install the tube to the extruder



Step 6: Stick the masking tape on the table

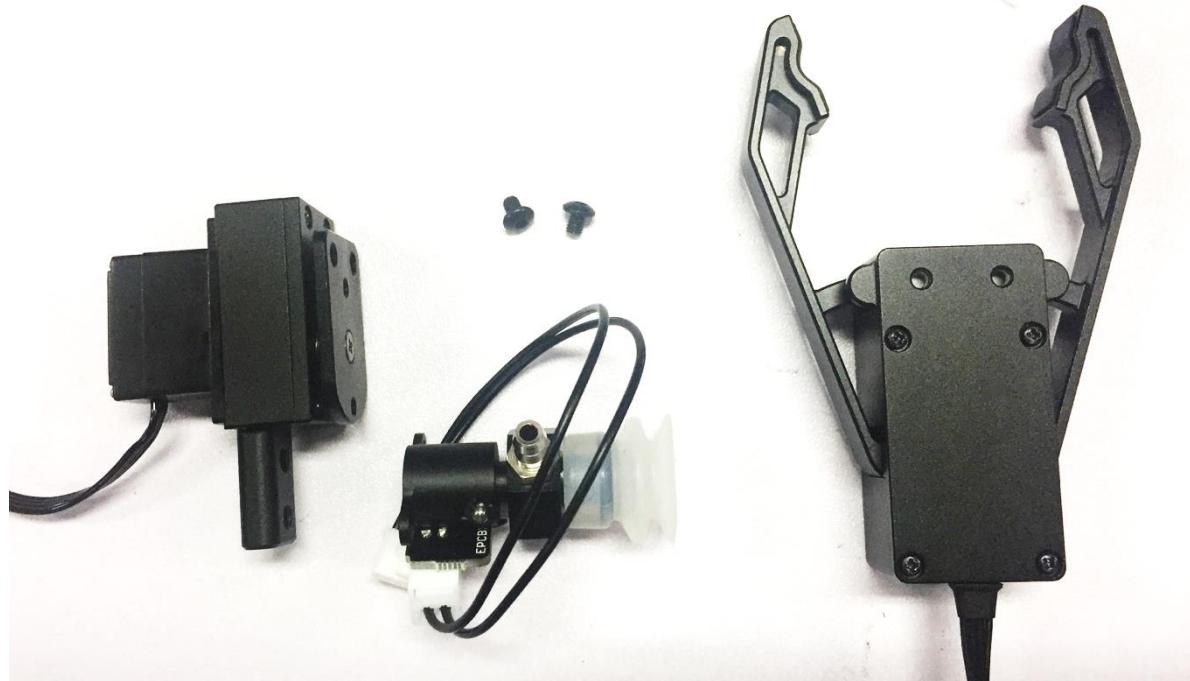


4. Swift Gripper

Preparation



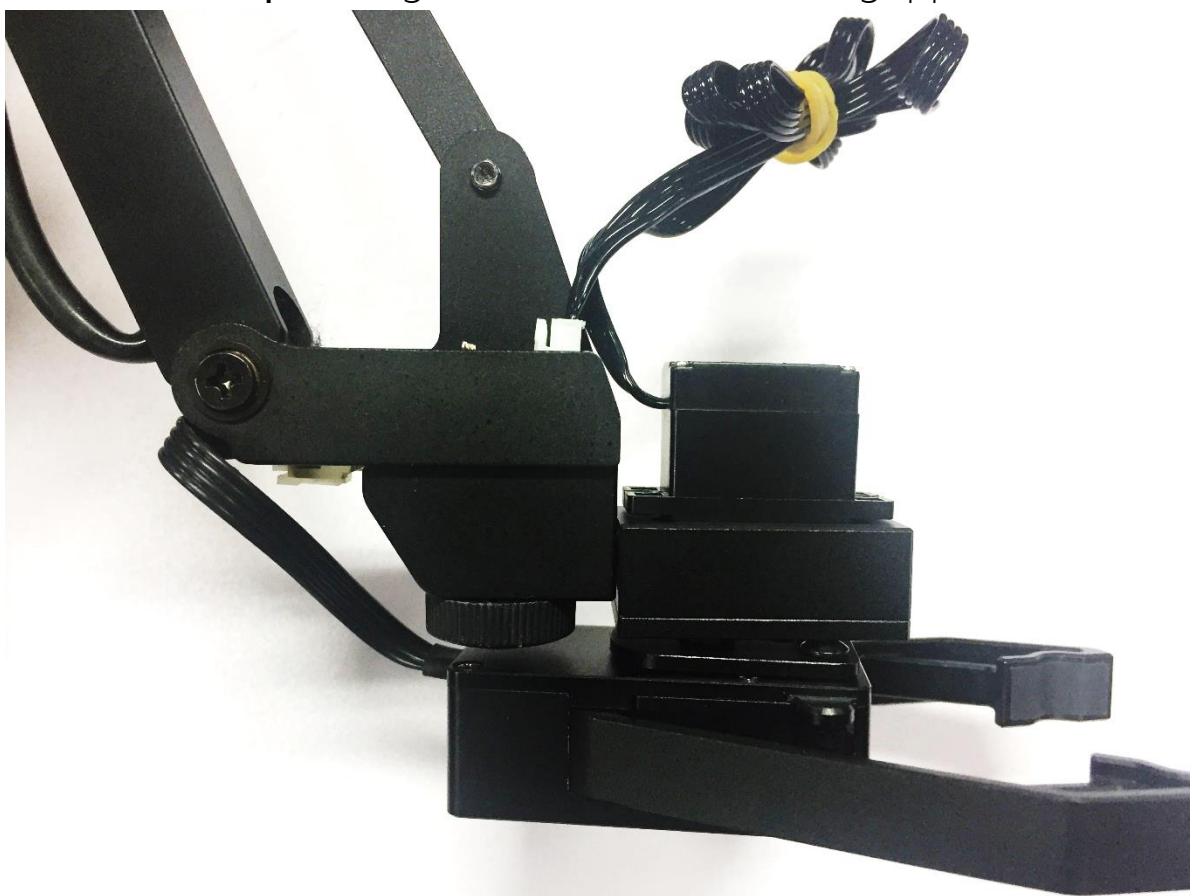
Step 1: Unscrew suction cup with the hex bar wrench.



Step 2: Fix the gripper and lock the nut tightly

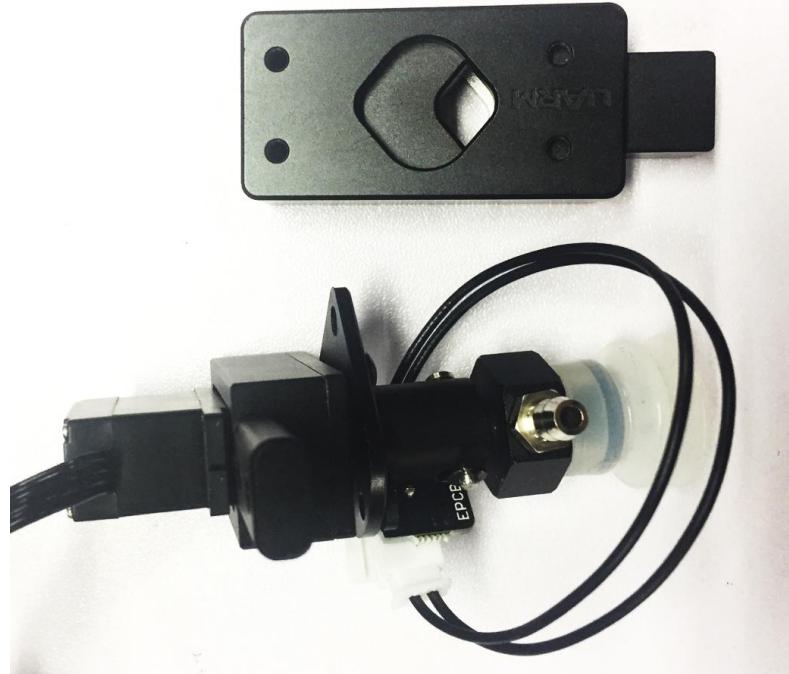


Step 3: Plug the 4th axis motor and gripper



5. Swift Universal Holder

Preparation

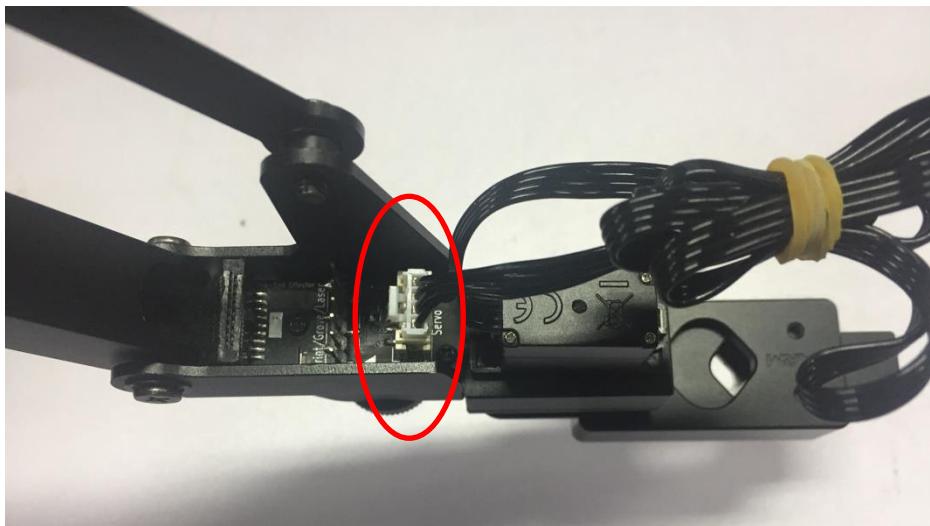


Step 1 : Unscrew suction cup with the hex bar wrench.



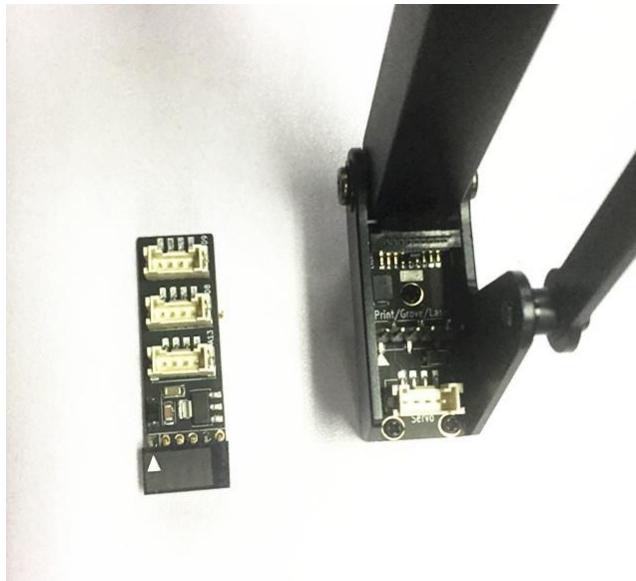
Step 2: Fix the gripper and lock the nut tightly

Step 3: Plug in the 4th axis motor

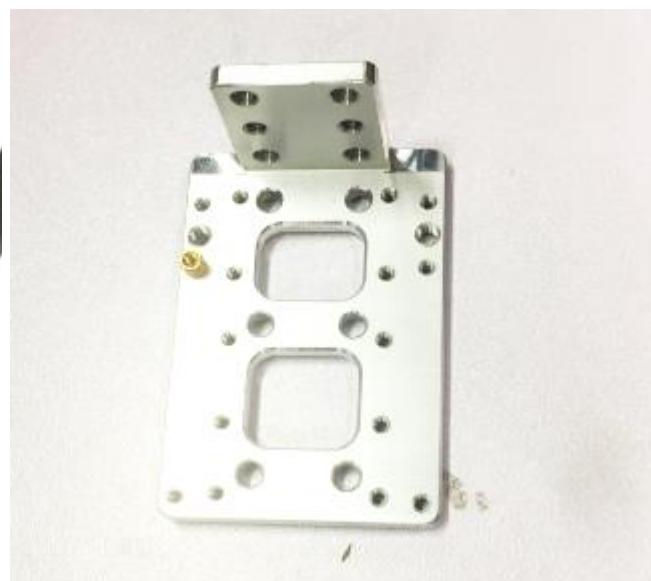


6. Seeed Grove Modules

Seeed Grove modules is a series of different sensors which helps us to extend the function of uArm to a completely new level. We are offering two parts to help you to connect the uArm with Grove much more easily.



Grove Extension



Grove mounting block

Caution:

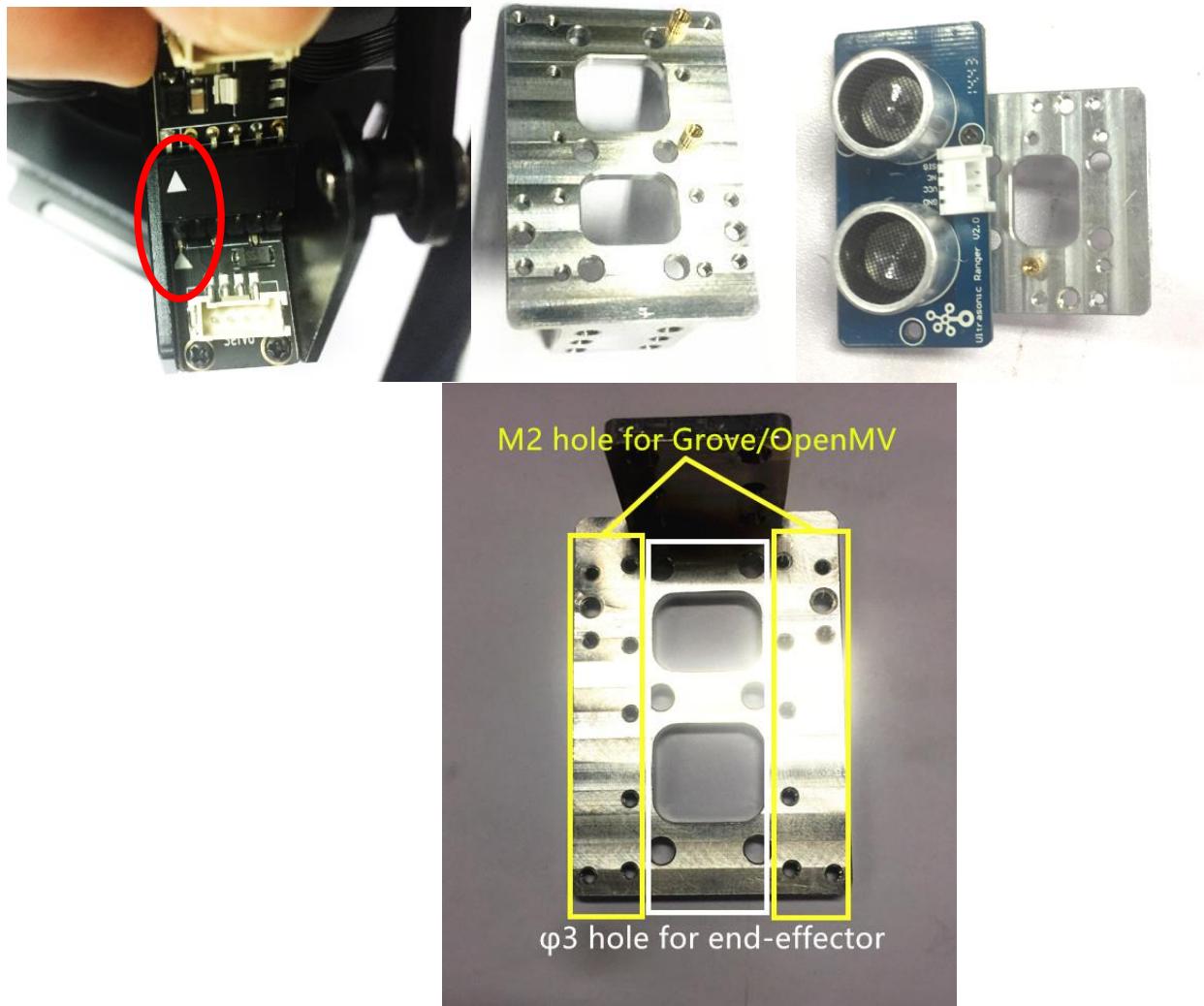
Grove extension for the uArm end-effector is just designed for(Step 1,2)

- *PIR Motion Sensor*
- *Mini Fan Module*
- *Electromagnet Module*
- *Ultrasonic Ranger*
- *Other Digital or Analog modules.*

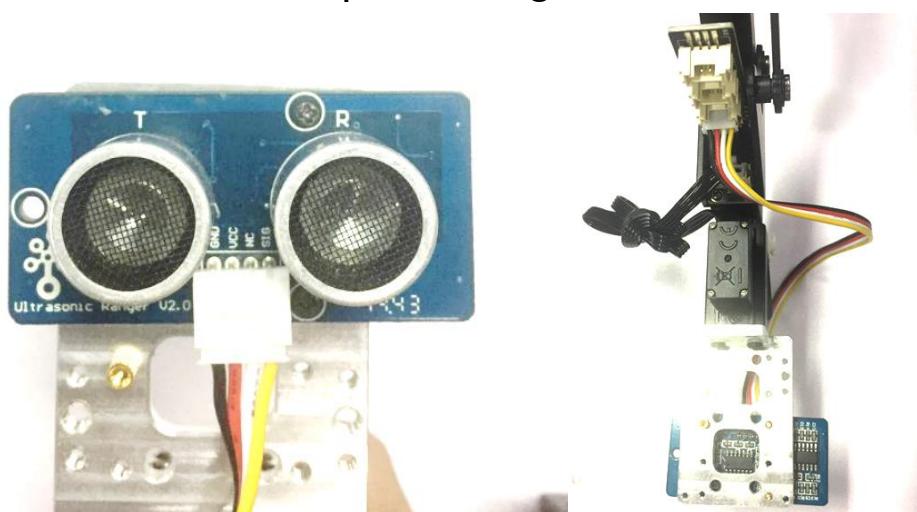
For the IIC module like: *(Step 3)*

- Temperature Sensor
- LCD RGB Backlight Module
- Color Sensor
- Gesture Sensor
- *Other Digital or Analog modules.*

Step 1 : Plug in the Grove breakout and fix the grove module to the mounting block.



Step 2 : Wiring.

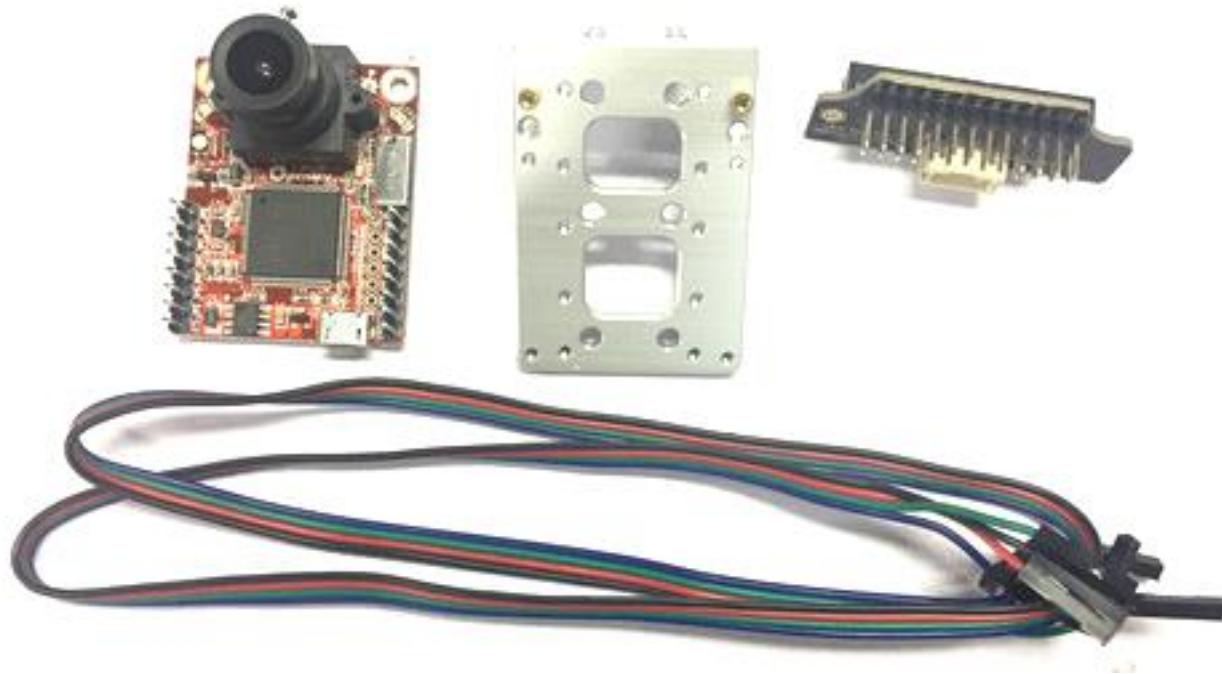


Step 3 : For the IIC modules



7. OpenMV Module (the firmware should be 3.1.9 or later)

Preparation



Step 1 : Download the latest OpenMV IDE

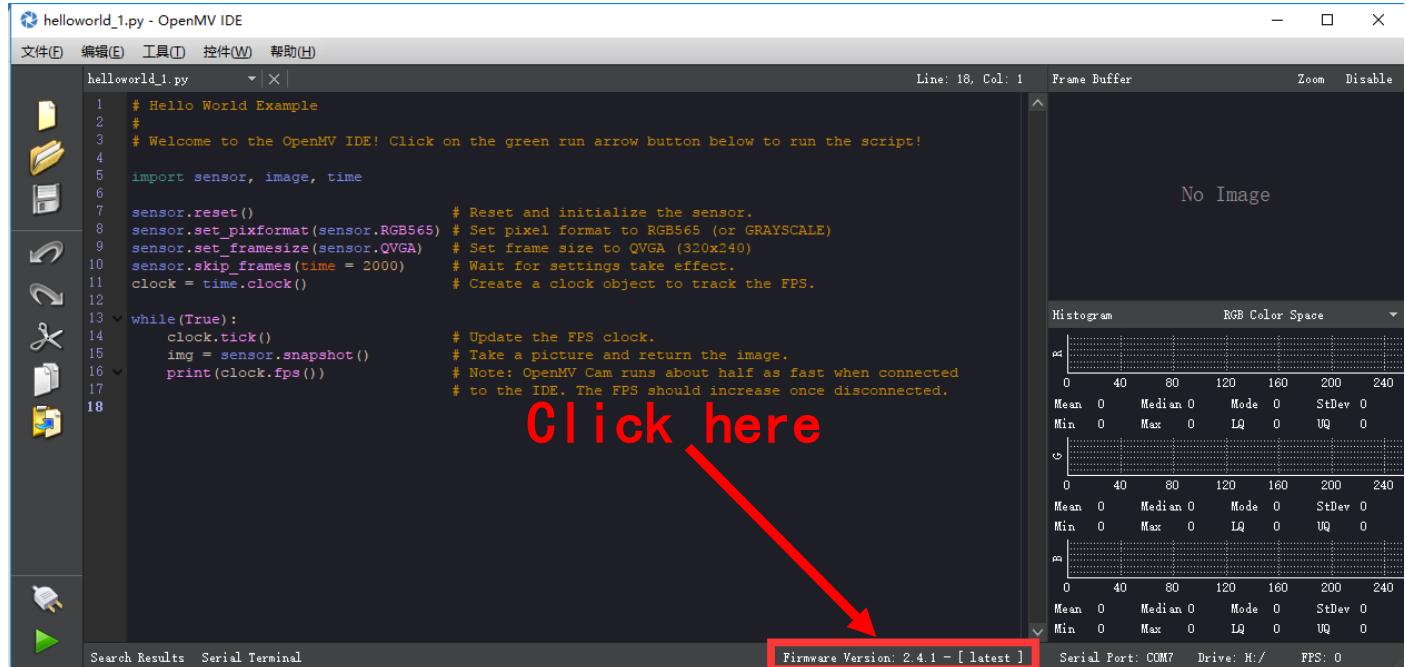
The screenshot shows the OpenMV IDE interface. On the left, the code editor displays the `helloworld_1.py` script:

```
helloworld_1.py - OpenMV IDE
文件(E) 编辑(E) 工具(I) 控件(W) 帮助(H)
helloworld_1.py Line: 18, Col: 1 Frame Buffer Zoom Disable
1 # Hello World Example
2 #
3 # Welcome to the OpenMV IDE! Click on the green run arrow button below to run the script!
4
5 import sensor, image, time
6
7 sensor.reset()          # Reset and initialize the sensor.
8 sensor.set_pixformat(sensor.RGB565) # Set pixel format to RGB565 (or GRayscale)
9 sensor.set_framesize(sensor.QVGA)   # Set frame size to QVGA (320x240)
10 sensor.skip_frames(time = 2000)    # Wait for settings take effect.
11 clock = time.clock()           # Create a clock object to track the FPS.
12
13 while(True):
14     clock.tick()               # Update the FPS clock.
15     img = sensor.snapshot()    # Take a picture and return the image.
16     print(clock.fps())        # Note: OpenMV Cam runs about half as fast when connected
                                # to the IDE. The FPS should increase once disconnected.
```

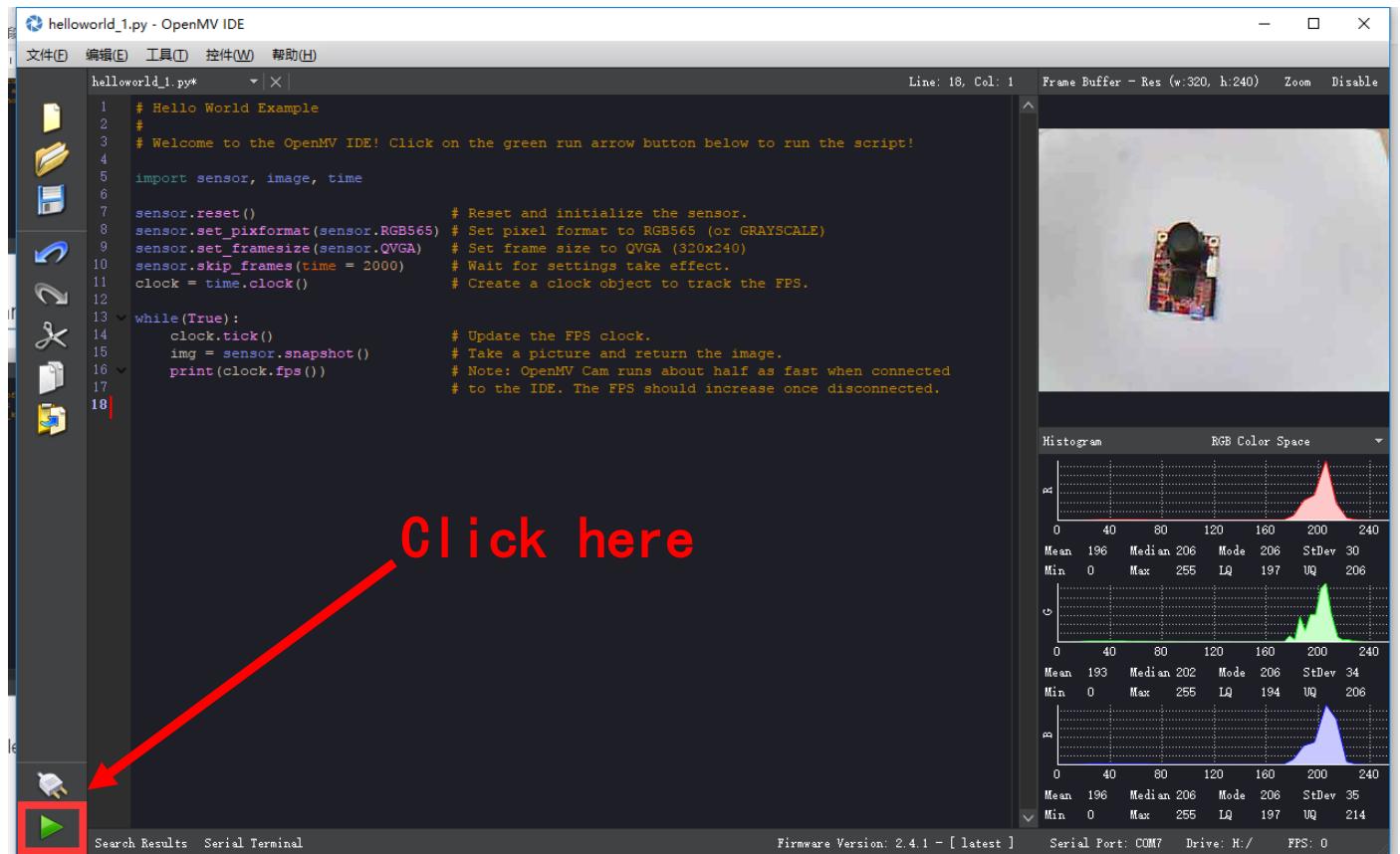
A red arrow points to the green play/run button icon at the bottom-left of the code editor. The word "Click here" is overlaid in red text above this button. To the right of the code editor is a "Frame Buffer" window showing a histogram and some text: "No Image". Below the code editor are buttons for "Search Results" and "Serial Terminal". At the bottom right are buttons for "Firmware Version", "Serial Port", "Drive", and "FPS".

(Download the latest OpenMV IDE from: <https://openmv.io/pages/download> and plug in the OpenMV camera to the computer and click Connect in the left of picture)

Step 2 : Upgrade the latest firmware to OpenMV by OpenMV IDE



Step 3 : Run the helloworld.py and focus the lens in the right window



Note: After IDE get the video, then rotate the lens to finish focusing(to see the objects 20cm away) then tight the screw.

Step 4 : Get the tracking.py code and save it to the OpenMV

The screenshot shows the OpenMV IDE interface. On the left is the code editor with the file 'tracking.py' open. A red box highlights the 'Save open script to OpenMV Cam' option in the 'File(F)' menu. The main window displays the script code, which includes initialization of a green LED and setting the arm to a default position via UART. To the right is a histogram viewer showing three histograms for 'x', 'y', and 'z' axes in 'RGB Color Space'. The status bar at the bottom indicates 'Firmware Version: 2.4.1 - [latest]', 'Serial Port: COM7', 'Drive: H:/', and 'FPS: 0'.

```
tracking.py - OpenMV IDE
File(F) 编辑(E) 工具(T) 控件(W) 帮助(H)
tracking.py Run Bootloader
Save open script to OpenMV Cam
example.
then run the script. A set of keypoints will be extracted
allowing frames. If you want a new set of keypoints re-run
for arguments to tune find_keypoints and match_keypoints.

Open Terminal
Machine Vision
选项(O)...
#initialise the alarm
10 led = LED(C) # Green led
11 led.toggle()
12 led.on()
13
14 #set the arm to the default position
15 utime.sleep_ms(3000)
16 uart = UART(3, 115200)
17 uart.write("G0 X250 Y0 Z")
18 uart.write("160 F10000\r\n")
19 utime.sleep_ms(500)
20 uart.write("G0 X250 Y0 Z")
21 uart.write("160 F10000\r\n")
22
23 utime.sleep_ms(5000)
24
25 #finish the initialization
26 led.off()
27
28 # Reset sensor
29 sensor.reset()

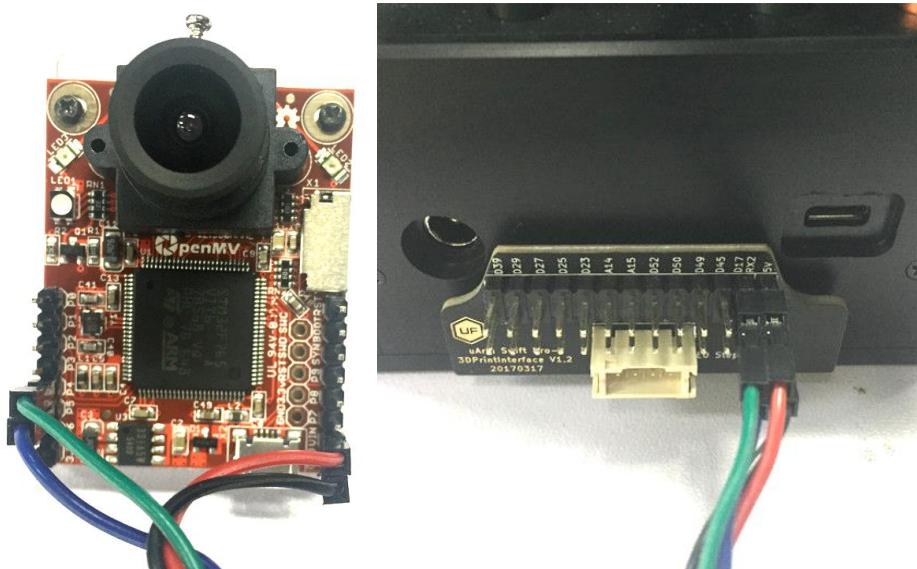
Search Results Serial Terminal
Firmware Version: 2.4.1 - [ latest ] Serial Port: COM7 Drive: H:/ FPS: 0
```

You could find the tracking.py from: <https://github.com/uArm-Developer/OpenMV-Examples>

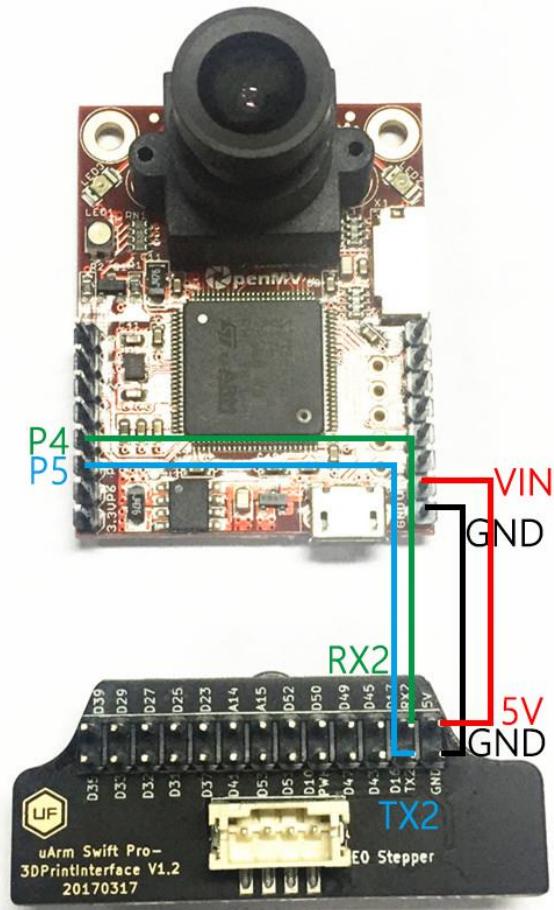
Note: The file system of OpenMV 2.4.1 is not very stable, and make sure the file has been stored into the module. Here is our steps:

- (1) Open the disk of OpenMV, and drag the tracking.py file into the disk and renamed it main.py;
- (2) If the code has been stored successfully, power on the module, the blue light turns on.

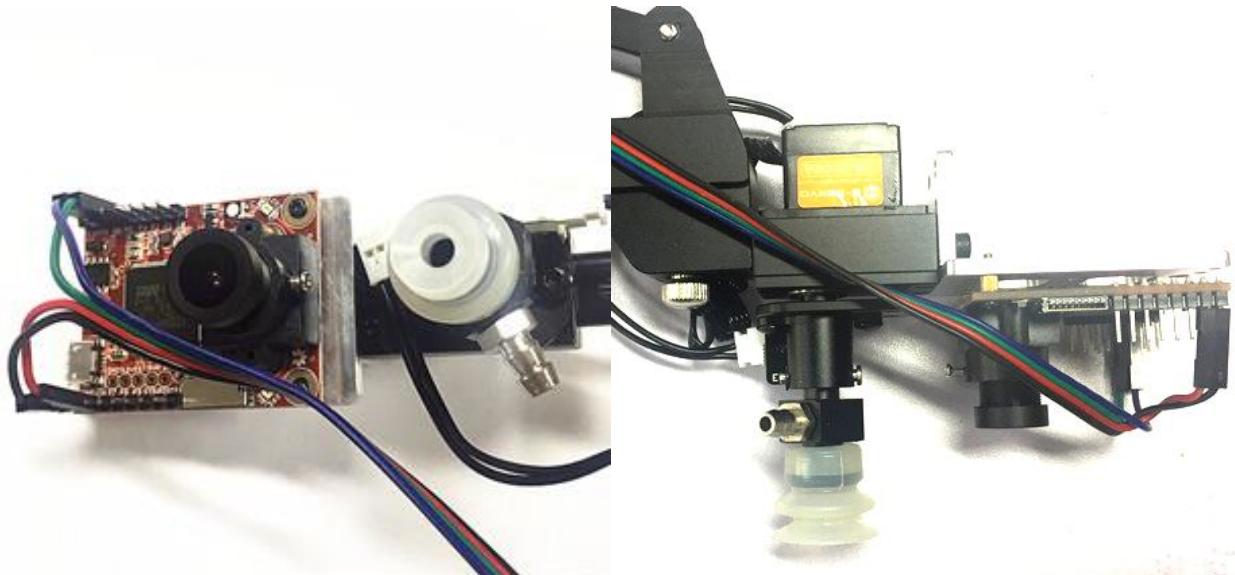
Step 5 : Unplug the OpenMV module and wiring the module



Caution: Please ensure the connection is correct. Or the computer wont recognize the uarm.



Step 6 : Install the camera module to the end-effector

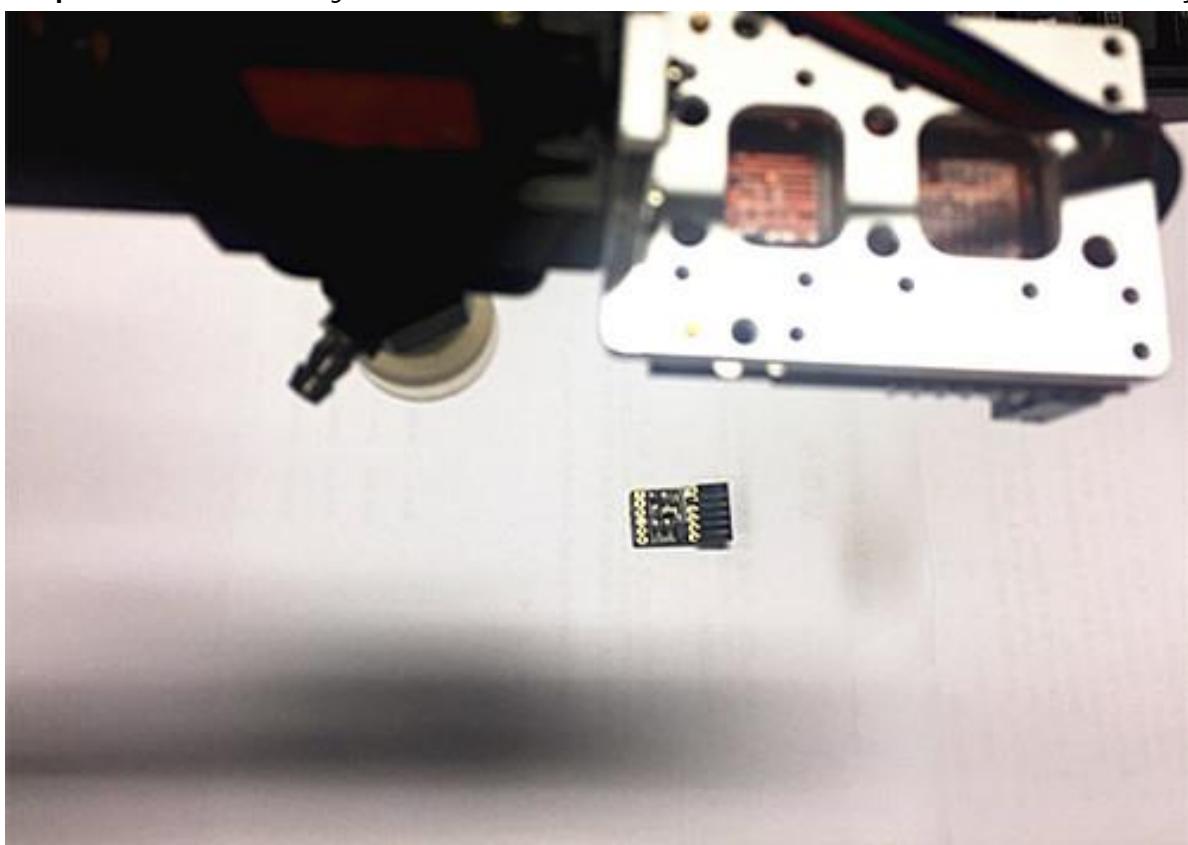


Note: Please pay attention to the assembling direction of OpenMV, or the arm will move to the opposite direction. And make sure the OpenMV is disconnected with you PC or the IDE will control the OpenMV.

Step 7 : Keep the table clean and non-reflective and get something with a lot of details like a pcb with resistors

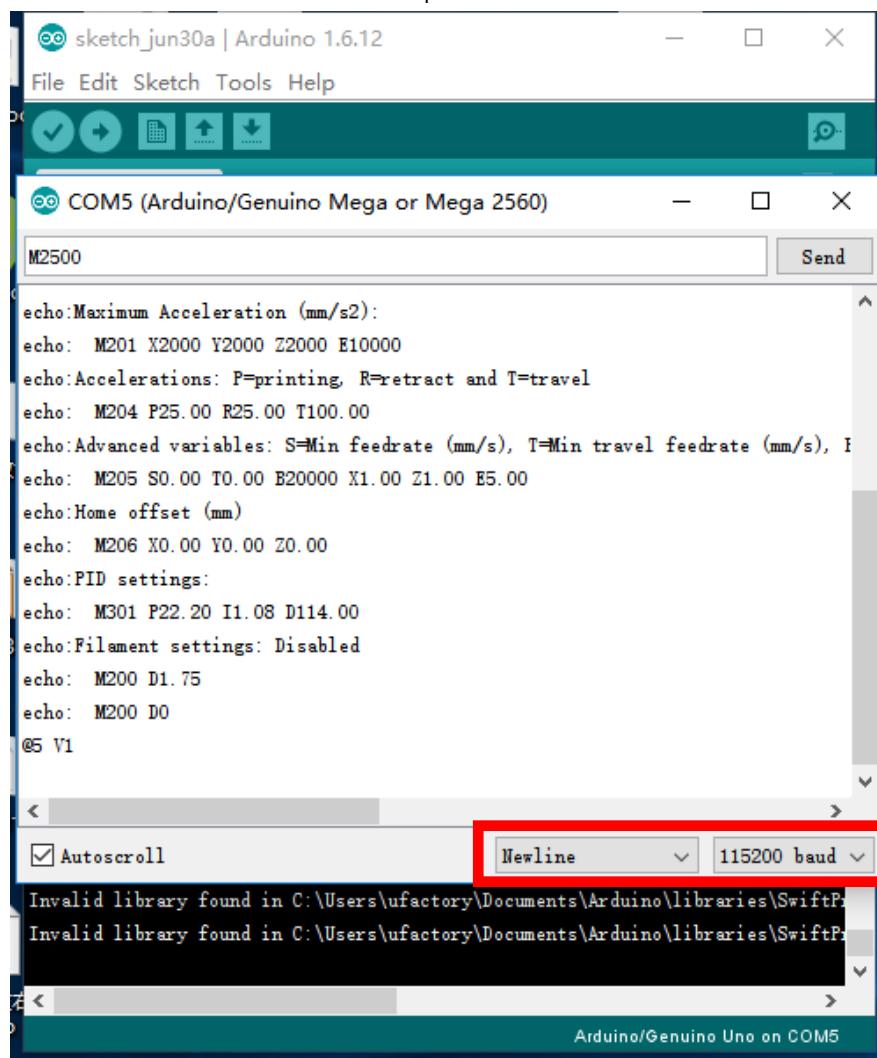


Step 8 : Put the object in front of uArm Swift Pro about 25cm away



Step 9 : Connect the USB port and power port of uArm, press the power button and open a serial monitor (for example Arduino IDE).

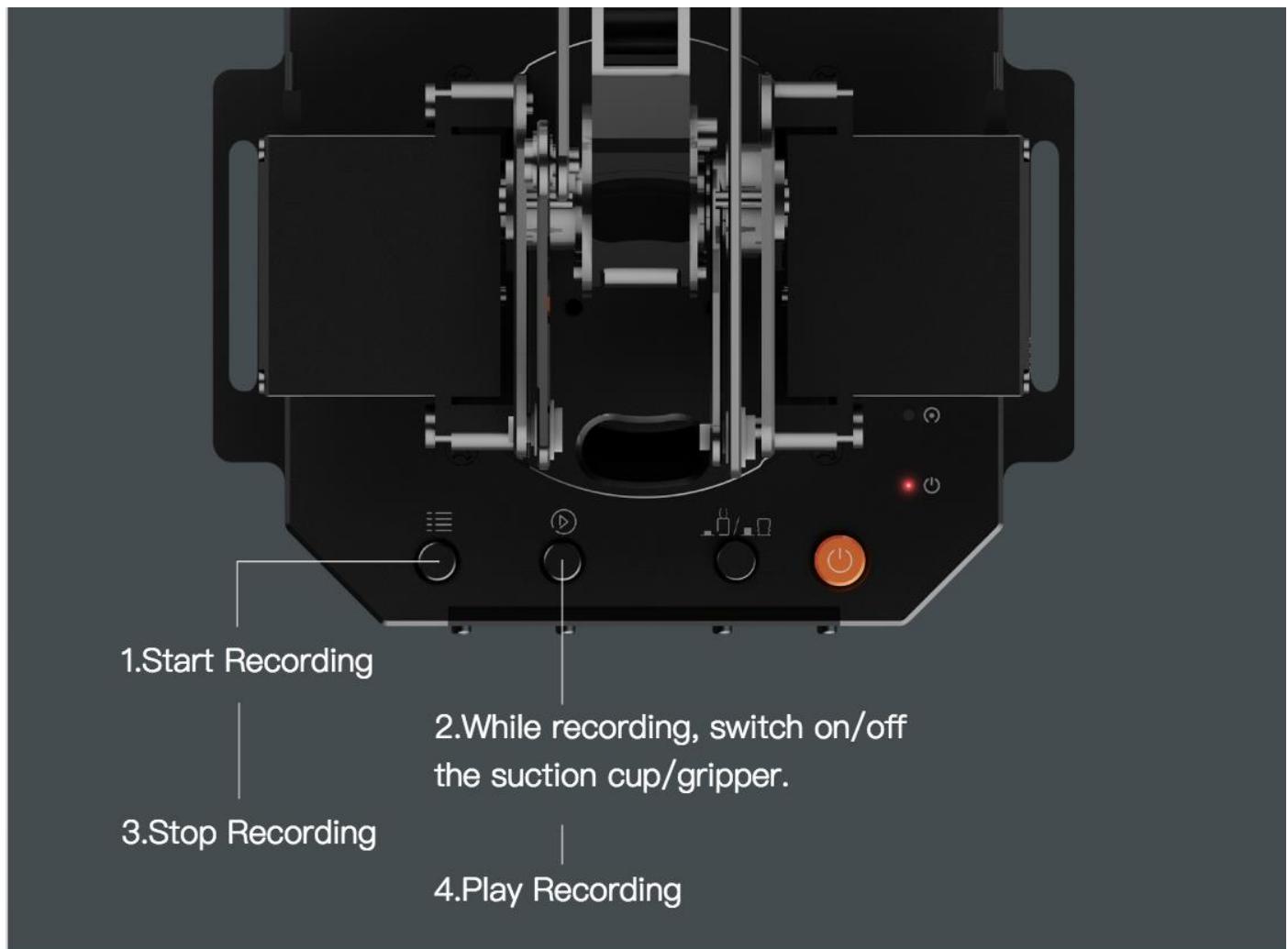
Step 10 : Adjust the settings (newline & 115200 baud) and then send the M2500 command which will switch the main UART port from USB to the port of OpenMV.



Step 11 : Move the object slowly, and the arm will follow it.

Offline Learning Mode

Use buttons on the base to “teach” uArm by hand.



SSSSSSS

Teach:

1. Start learning mode. Press the  once, and the status indicator turns green.
2. Teach the robot manually. Press the  once to turn on the end-effector, again to turn off. (If  is down end-effector is gripper, or it is pump. Please remember to keep the button up after learning or it will turn on the Bluetooth. Page 5)
3. Finish the learning process. Press  once, and the status indicator turns off.

PLAY:

1. One-time playback: Press  once, or Loop playback: press  & hold for 2 seconds.
2. The status indicator starts flashing green slowly.
3. Press  once to stop playing.

Software: uArm Studio (Win/Mac)

1. Download uArm Studio from:

<http://www.ufactory.cc/#/en/support/>

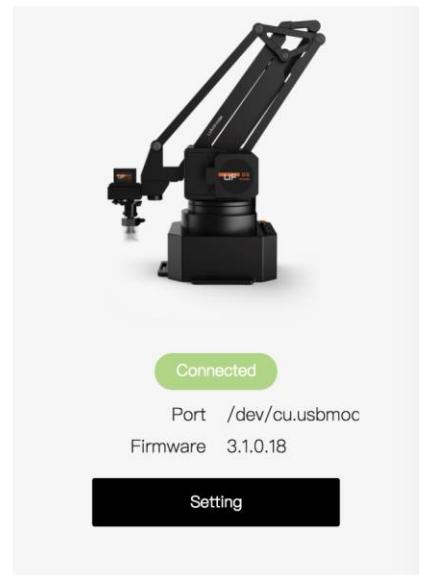
* Windows(Win7/8 or before) users will be reminded to install driver.
Simply follow the instructions to install.

2. Device Connection

- 1) Plug in the power cable.
- 2) Press down the power button.
- 3) Connect uArm to your computer via USB.

Status of device connection is displayed on home page.

More info is displayed in "Setting".



< Setting

Device

Check for Updates

Device	Swift Pro
Serial Port	/dev/cu.usbmodem1411
Firmware	3.1.0.18
Serial No.	95634303432351E01241

3. Drawing/Laser Engraving

1) Design a pattern.

Insert text/shape



Insert an image

("outline" or "black & white".)



2) Click the play button to continue.



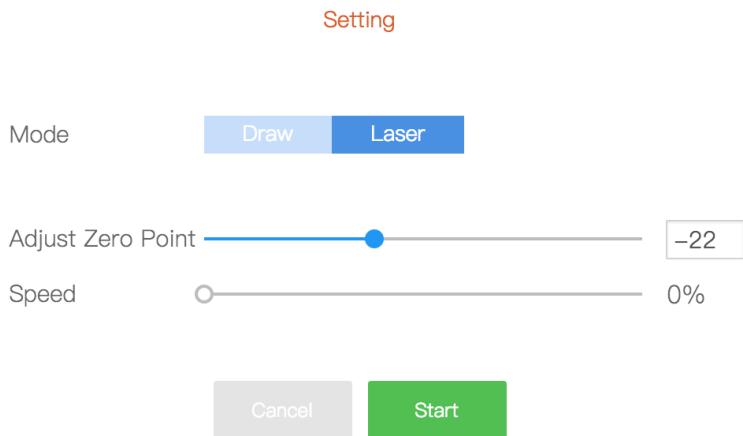
3) Adjust Zero Point



IMPORTANT:

Please adjust zero point before drawing/engraving.

Ensure the pen/laser is TOUCHING the platform.



For laser engraving, you can also adjust the speed of engraving.

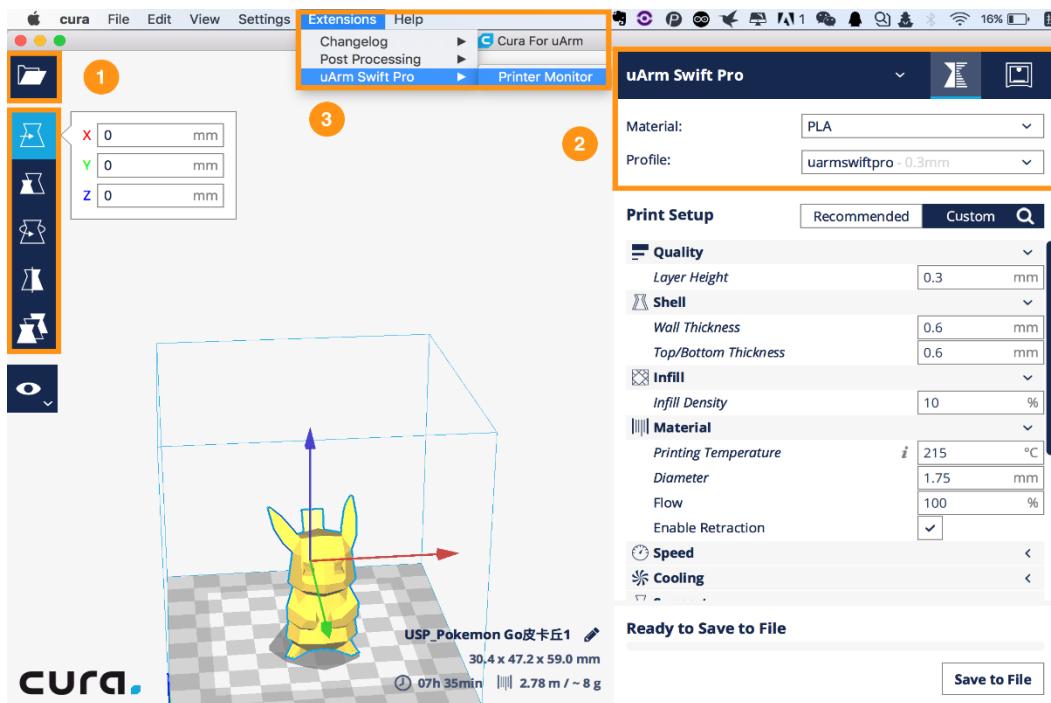
- 4) Start drawing/engraving!

4.3D Printing

Preparation

- 1) [Download CuraForuArm](#)
- 2) Double-click .dmg/.exe file to install.
- 3) Enter the 3D Printing section in Studio, and CuraForuArm window will pop up automatically. If not, click the "Open Cura" button.

CuraForuArm Interface

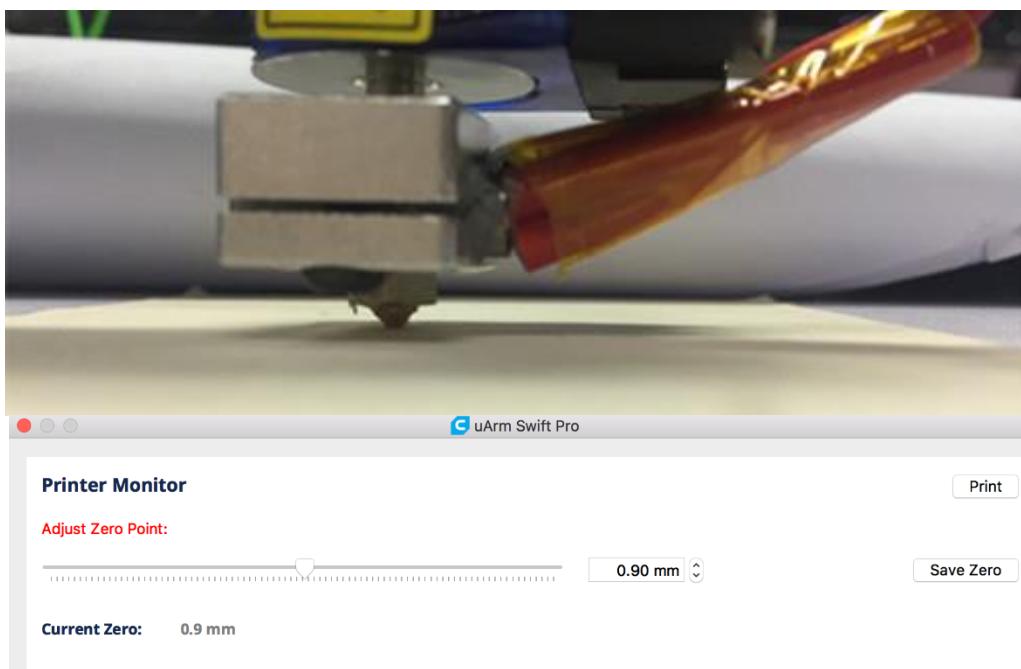


- 1) Import an .stl file, edit the size/position of the model.
- 2) Select "uArm Swift Pro" as the printer, and choose the related profile. It is recommended to keep the default settings unchanged.
- 3) Open Printer Monitor.



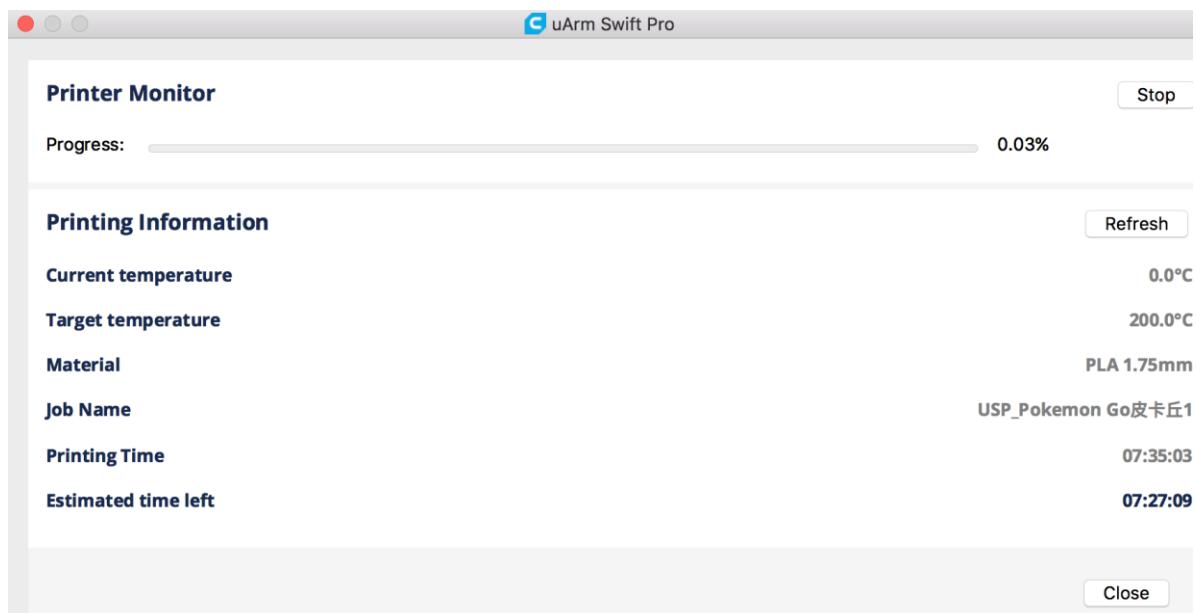
IMPORTANT: Please adjust zero point before printing.
Ensure the hot end is JUST TOUCHING the platform.
Then click "Save Zero".

(The zero point of each arm is not the same, please adjust the zero point following the step 3) before printing.)



4) Start printing!

The 3D extruder will automatically heat up to 200°C to print.
uArm will remain still during the pre-heating section.
Please don't touch the metal part of the extruder for safety reason.



5.Teach & Play: Learning Mode

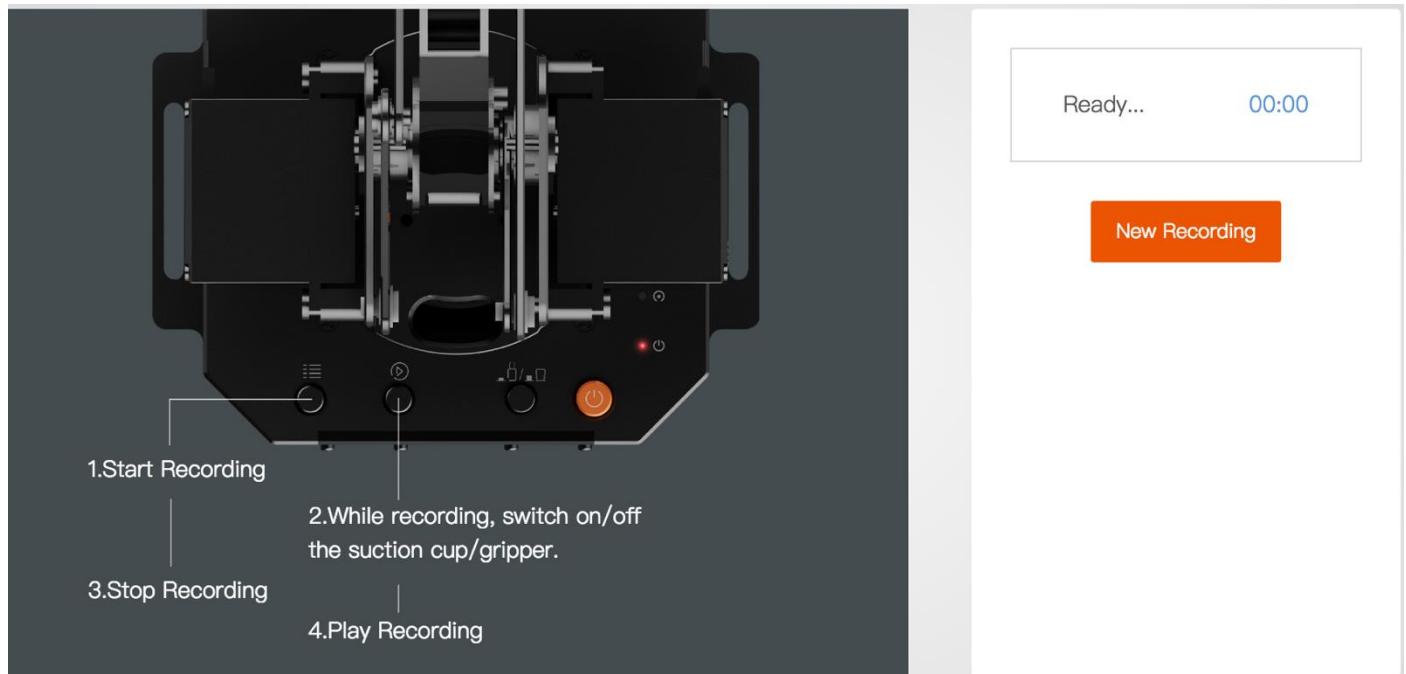
What is Teach & Play?

Teach uArm by hand, and then replay the recording anytime.

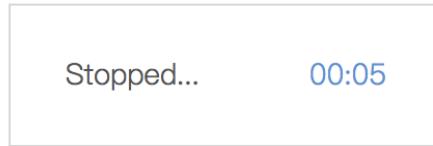
How?

1) Make a recording

- Click the “New Recording” button to start “teaching”, OR,
- Use the buttons on the base (usage of the buttons is the same as that under “*Offline Learning Mode*”).

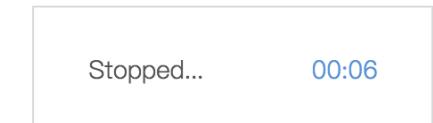


2) Save your recording



Discard **Save**

- 3) Replay the recording in different speed and times



Discard **Save**

Speed 1x

Times 1

Loop

Play

What makes “Teach & Play” different from “Offline Learning Mode”?

- 1) No time limit while “teaching” with uArm Studio.
- 2) You may save, export your recordings and import recordings made by others.
- 3) You may apply your recording in Blockly (visual programming interface, which is explained up next).

6. Blockly: Visual Programming

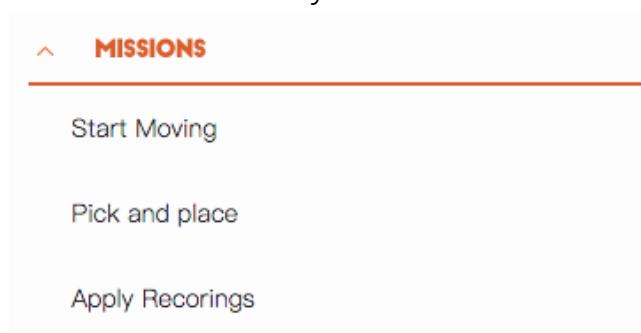
What is Blockly?

Blockly in uArm Studio is a visual programming interface specially designed for controlling uArm.

Getting Started

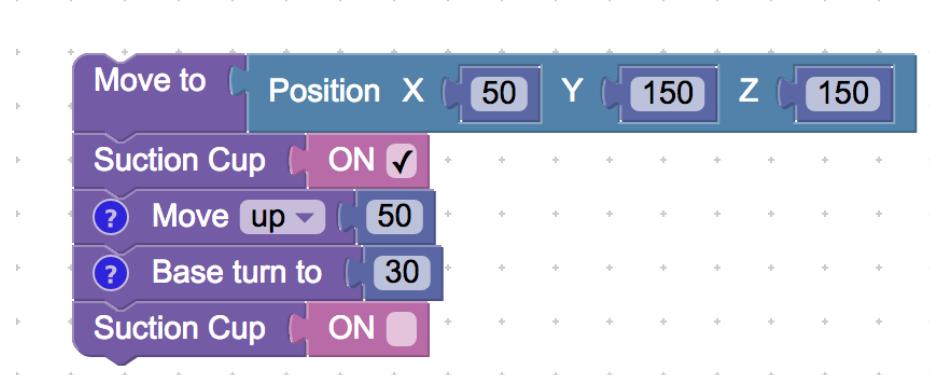
Three “missions” are prepared to get you through Blockly quickly.

Please try them out!

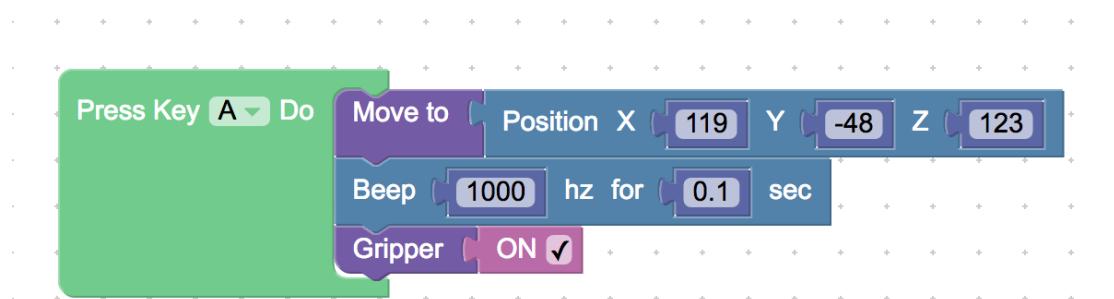


What can you do with Blockly?

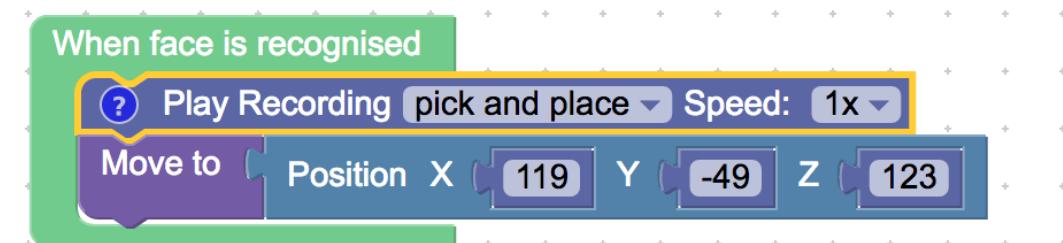
1) Control uArm's basic movements



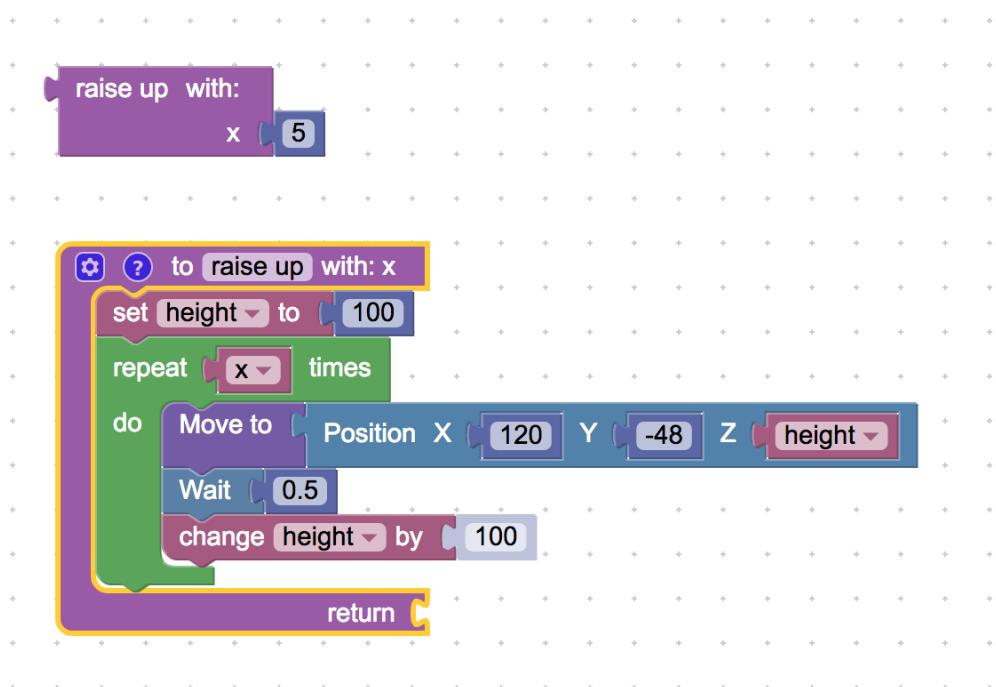
2) Change events (i.e. how you trigger commands)



3) Apply recorded movements



4) Dig deeper into programming (functions, variables, etc.)



For Developers

1. Communication Protocol

1) Introduction:

- uArm gCode is an important part of the uArm software.
- Based on the standard gCode protocol, we add a new protocol head in front of the gCode so that it can be more easily to use and debug.
- What's more, it is designed to be compatible with the standard gCode. (We offer the code of decode the standard gCode)

2) Example:

- Sending command from PC
"#25 G0 X180 Y0 Z150 F5000"
//move to [180,0,150] with the speed 5000mm/min
- Reply from uArm "\$25 ok"

3) Commands(TBD).

Command can be divided into two parts:

Command with underline: it's the new added protocol head.

- The command from PC starts with '#', while the command from uArm starts with '\$'.

- And the data following the symbol decided by the PC, and the reply from the uArm should have the same data which indicates it finish the command. (In the example above, PC sends the command with '#25' and uArm replies the command with '\$25')

Command without the underline: it's the standard gCode.

Caution

1. There should be blank space between each parameter;
2. The letters in the command should be capitalized;

GCode Command (v1.2)	Description	Feedback
1. #n is used for the debug, if you don't want to use it please remove it directly. (For Example: G2202 N <u>0</u> V <u>90</u> \n) 2. '\n' is the symbol of line feed.		
Moving Command (parameters are in underline)		
#n G0 X <u>100</u> Y <u>100</u> Z <u>100</u> F <u>1000</u> \n	Move to XYZ(mm), F is speed(mm/min)	\$ <u>n</u> ok \n or \$ <u>n</u> E <u>x</u> \n (refer to Err output)
#n G1 X <u>100</u> Y <u>100</u> Z <u>100</u> F <u>1000</u> \n	After entering the laser mode (M2400 S1) , command G1 means laser on, G0 means off.	\$ <u>n</u> ok \n or \$ <u>n</u> E <u>x</u> \n (refer to Err output)
#n G2004 P <u>1000</u> \n	Delay microsecond	\$ <u>n</u> ok \n
#n G2201 S <u>100</u> R <u>90</u> H <u>80</u> F <u>1000</u> \n	Polar coordinates, S is stretch(mm), R is rotation(degree),H is height(mm), F is speed(mm/min)	\$ <u>n</u> ok \n or \$ <u>n</u> E <u>x</u> \n (refer to Err output)
#n G2202 N <u>0</u> V <u>90</u> \n	Move the motor to the position ,N is ID of joints(0~3),V is angle(0~180)	\$ <u>n</u> ok \n or \$ <u>n</u> E <u>x</u> \n (refer to Err output)
#n G2203\n	Stop moving	\$ <u>n</u> ok \n or \$ <u>n</u> E <u>x</u> \n (refer to Err output)
#n G2204 X <u>10</u> Y <u>10</u> Z <u>10</u> F <u>1000</u> \n	Relative displacement	\$ <u>n</u> ok \n or \$ <u>n</u> E <u>x</u> \n (refer to Err output)
#n G2205 S <u>10</u> R <u>10</u> H <u>10</u> F <u>1000</u> \n	Polar coordinates for relative displacement	\$ <u>n</u> ok \n or \$ <u>n</u> E <u>x</u> \n (refer to Err output)
Setting Command (parameters are in underline)		
#n M17\n	Attach all the joint motors	\$ <u>n</u> ok \n
#n M2019\n	Detach all the joint motors	\$ <u>n</u> ok \n
#n M2120 V <u>0.2</u> \n	Set time cycle of feedback, return Cartesian coordinates, V is time(seconds)	@3 X <u>154.71</u> Y <u>194.91</u> Z <u>10.21</u> \n
#n M2200\n	Check if uArm is moving	\$ <u>n</u> ok V <u>1</u> \n (1 moving,0 stop)
#n M2201 N <u>0</u> \n	attach motor, N is ID of joints(0~3)	\$ <u>n</u> ok \n or \$ <u>n</u> E <u>x</u> \n (refer to Err output)
#n M2202 N <u>0</u> \n	Detach motor, N is ID of joints(0~3)	\$ <u>n</u> ok \n or \$ <u>n</u> E <u>x</u> \n (refer to Err output)

#n M2203 N <u>0</u> \n	Check if the motor is attached, N is ID of joints(0~3)	\$n ok V <u>1</u> \n (1 attached,0 detached)
#n M2210 F <u>1000</u> T <u>200</u> \n	buzzer,F is frequency, T is time (ms)	\$n ok \n or \$n Ex \n (refer to Err output)
#n M2211 N <u>0</u> A <u>200</u> T <u>1</u> \n	Read EEPROM N(0~2,0 is internal EEPROM,1 is USR_E2PROM, 2 is SYS_E2PROM), A is address, T is type (1 char,2 int,4 float)	\$n ok V <u>10</u> \n
#n M2212 N <u>0</u> A <u>200</u> T <u>1</u> V <u>10</u> \n	Write EEPROM N(0~2,0 is internal EEPROM,1 is USR_E2PROM, 2 is SYS_E2PROM), A is address, T is type (1 char,2 int,4 float)V is the input data	\$n ok\n
#n M2213 V <u>0</u> \n	Default function of base buttons (0 false, 1 true)	\$n ok\n
#n M2220 X <u>100</u> Y <u>100</u> Z <u>100</u> \n	Convert coordinates to angle of joints	\$n ok B <u>50</u> L <u>50</u> R <u>50</u> \n (B joint 0,L joint 1,R joints 2, 0~180)
#n M2221 B <u>0</u> L <u>50</u> R <u>50</u> \n	Convert angle of joints to coordinates	\$n ok X <u>100</u> Y <u>100</u> Z <u>100</u> \n
#n M2222 X <u>100</u> Y <u>100</u> Z <u>100</u> P <u>0</u> \n	Check if it can reach,P1 polar, P0 Cartesian coordinates	\$n ok V <u>1</u> \n (1 reachable, 0 unreachable)
#n M2231 V <u>1</u> \n	pump V1 working, V0 stop	\$n ok \n or \$n Ex \n (refer to Err output)
#n M2232 V <u>1</u> \n	gripper V1 close, V0 open	\$n ok \n or \$n Ex \n (refer to Err output)
#n M2234 V <u>1</u> \n	Enable/disable Bluetooth (1:enable, 0:disable)	\$n ok\n
#n M2240 N <u>1</u> V <u>1</u> \n	Set the digital IO output	\$n ok \n or \$n Ex \n (refer to Err output)
M2245 V <u>btname</u> \n	Set the name of Bluetooth, 11 letters limited (Do not add #n in this command)	ok \n
M2246\n	Rewrite UUID	ok\n
#n M2300 N <u>10</u> \n	Please check the Grove modules & OpenMV below	

#n M2301 N <u>10</u> V <u>1000</u> \n	Please check the Grove modules & OpenMV below	
#n M2302 N <u>10</u> V <u>1</u> \n	Please check the Grove modules & OpenMV below	
#n M2303 N <u>17</u> T <u>0</u> V <u>0</u> \n	Please check the Grove modules & OpenMV below	
#n M2400 S <u>0</u> \n	Set the mode of arm (0:Normal 1:Laser 2:3D printing 3:Universal holder)	\$n ok \n
#n M2401\n	Set the current position into the reference position	\$n ok \n
#n M2410\n	Set the height zero point	\$n ok \n
#n M2411 S <u>100</u> \n	Set the offset of end-effector (mm)	\$n ok \n
#n M2500\n	Please check the Grove modules & OpenMV below	
Querying Command (parameters are in underline)		
#n P2200\n	Get the current angle of joints	\$n ok B <u>50</u> L <u>50</u> R <u>50</u> \n
#n P2201\n	Get the device name	\$n ok V <u>3.2</u> \n
#n P2202\n	Get the hardware version	\$n ok V <u>1.2</u> \n
#n P2203\n	Get the software version	\$n ok V <u>3.2</u> \n
#n P2204\n	Get the API version	\$n ok V <u>3.2</u> \n
#n P2205\n	Get the UID	\$n ok V <u>0123456789AB</u> \n
#n P2206 N <u>0</u> \n	Get the angle of number 0 joint (0~2)	\$n ok V <u>80</u> \n
#n P2220\n	Get current coordinates	\$n ok X <u>100</u> Y <u>100</u> Z <u>100</u> \n
#n P2221\n	Get current polar coordinates	\$n ok S <u>100</u> R <u>90</u> H <u>80</u> \n
#n P2231\n	Get the status of pump	\$n ok V <u>1</u> \n (0 stop, 1 working, 2 grabbing things)
#n P2232\n	Get the status of gripper	\$n ok V <u>1</u> \n (0 stop, 1 working, 2 grabbing things)
#n P2233\n	Get the status of limited switch	\$n ok V <u>1</u> (1 triggered, 0 untriggered)

#n P2234\n	Get the status of power connection	\$n ok V <u>1</u> \n (1 connected, 0 unconnected)
#n P2240 N <u>1</u> \n	Get the status of digital IO	\$n ok V <u>1</u> \n (1 High, 0 Low)
#n P2241 N <u>1</u> \n	Get the status of analog IO	\$n ok V <u>295</u> \n (return the data of ADC)
#n P2242\n	Get the default value of AS5600 in each joint	\$n ok B <u>2401</u> L <u>344</u> R <u>1048</u> \n
#n P2400\n	Check current status	\$n ok V <u>1</u> \n (0: normal; 1: laser; 2: 3D printing; 3: Universal holder;)
Ticking feedback		
@1	Ready	
@3	Timed feedback , "M2120"	
@4 N <u>0</u> V <u>1</u> \n	Report the button event. N: 0 = Menu button, 1 = Play button V: 1 = Click, 2 = Long Press	
@5 V <u>1</u> \n	Report event of power connection	
@6 N <u>0</u> V <u>1</u> \n	Report event of limit switch in end-effector	
@7 temp error	Temperature error in 3D printing	
Err Output		
E20	Command not exist	
E21	Parameter error	
E22	Address out of range	
E23	Command buffer sssfull	
E24	Power unconnected	
E25	Operation failure	
Grove modules & OpenMV		
N is the ID of each grove modules: <u>10</u> : Color sensor; <u>11</u> : Gesture sensor; <u>12</u> : Ultrasonic; <u>13</u> : Fan; <u>14</u> : Electromagnet; <u>15</u> : Temperature & Humidity; <u>16</u> : PIR Motion; <u>17</u> : RGB LCD;		
#n M2300 N <u>10</u> \n	Initialize the Grove modules, N is the ID of each module	
#n M2301 N10 V <u>1000</u> \n	Auto report time for color	@10 N10 R <u>20</u> G <u>10</u> B <u>255</u> \n

	sensor, V is the time (microsecond)	(RGB value)
#n M2301 N11 V <u>1000</u> \n	Auto report time for gesture sensor, V is the time (microsecond)	@10 N11 V <u>16</u> \n(1: right; 2: left; 4: up; 8: down; 16: forward; 32: backward; 64: CW; 128: CCW;)
#n M2301 N12 V <u>1000</u> \n	Auto report time for ultrasonic, V is the time (microsecond)	@10 N12 V <u>27</u> \n(The distance value in cm)
#n M2301 N15 V <u>1000</u> \n	Auto report time for T&H, V is the time (microsecond)	@10 N15 T <u>32.12</u> H <u>76.5</u> \n(temperature in °C, humidity in %)
#n M2301 N16 V <u>1000</u> \n	Auto report time for PIR motion, V is the time (microsecond)	@10 N16 V <u>1</u> \n(1: motion; 0: no motion;)
#n M2302 N13 V <u>128</u> \n	Fan setting, V is the duty cycle from 0-255	\$n ok\n
#n M2302 N14 V <u>1</u> \n	Electromagnet setting, 0 is off, 1 is on	\$n ok\n
#n M2303 N17 T <u>0</u> \n	Turn off/on display (0 is off, 1 is on, 2 is clear)	\$n ok\n
#n M2303 N17 R <u>25</u> G <u>25</u> B <u>25</u> \n	Change the rgb value of backlight	\$n ok\n
#n M2303 S <u>1</u> V <u>Text</u> \n	S is the line (1 or 2), V is the text content For example: M2303 S <u>1</u> V <u>ufactory</u>	\$n ok\n
#n M2500\n	Switch the uart0 to uart2 for external TTL uart communication (For example OpenMV)	\$n ok \n

d. Different modes for uArm Swift Pro

Since different types of the end-effectors have different length and height, so we designed the command M2400, which could help us to fit the uArm into different situations easily. With this command, there is no need to concern about how to adjust the parameters for different situations.

Currently we offer 4 kinds of mode:

M2400 V0 : Normal mode (end-effector tools: suction)

M2400 V1 : Laser mode (end-effector tools: laser)

M2400 V2 : 3D printing mode (end-effector tools: extruder)

M2400 V3 : Universal holder mode (end-effector tools: universal holder)

For the gripper, there is no special mode since gripper has the fingers and can rotate horizontally.

uArm Community

[UFACTORY Official Forum](#)

[uArm User Facebook Group](#)

[Ask for Help](#)

Release Note

Version	Note	
1.0.7	Modify several steps of 3D printing and fix the misunderstanding Add the laser mode command G1	Tony
1.0.8	Add more details about OpenMV Add the note of laser focusing Add the caution of installing base extension Add the caution of user defined button	Tony
1.0.9	Modify the steps of laser focusing and grove installing	Tony
1.0.10	Add more details to OpenMV tutorial Add details to offline learning modess Add M2500 command in command list	Tony
1.0.11	Modify the OpenMV instructions Add more Gcode commands	Tony
1.0.12	Add the details of installing the tube in 3D printing mode	Tony