

pcDuino

Rover: A WiFi Video Surveillance Remote Control Robot Powered by pcDuino

Jingfeng Liu

Content

Introduction.....	4
Part I: Hardware	6
I. Tools Needed.....	6
II. Getting to Know Hardware.....	7
III. Assemble the PTZ Camera.....	20
IV Assembling of the Platform	40
V. Wiring Instruction.....	47
Part II: Software Part.....	48
VI. Configure pcDuino to be a WiFi Access Point.....	49
VII. Install Video Stream Server on pcDuino	59
VIII. Install Control Software	61
IX. Control Rover with a PC.....	65
X. Control Rover with an Android device	67
10.1 Introduction	67
10.2 GUI	67
10.3 Introduction to Source Code	69
10.3.1 Interface Control Module.....	69
10.3.2 Picture Format Conversion Module.....	69
10.3.3 Socket Communication Module.....	69
10.4 User Guide.....	70
XI. Download Links	73

Introduction

With the development of embedded hardware and software, various technologies are becoming more integrated. As a result, the requirement to hardware and software co-development is becoming higher and engineer's learning cycle is getting longer. In recent years, open source hardware movement is becoming increasingly popular in the world. Arduino is a leader in this movement. User groups spread from engineers to college students, then to middle school students or even primary school kids. The emergence of a variety of open-source hardware platforms greatly reduces the learning curve, stimulates innovation, and accelerates the conversion from idea to realization.

pcDuino is a super Arduino with mini PC functionality. pcDuino features onboard hardware interface fully compatible with Arduino Uno's Shield interface. Arduino Shields available in the market can be installed on pcDuino with a simple translation board (T-board). The Arduino source code is also fully compatible with pcDuino. The operation system used by pcDuino is open source Linux Ubuntu. Countless open-source Linux software packages with numerous open-source hardware boards can be combined in pcDuino to create a variety of applications. pcDuino offers a

perfect platform for development involves open source software and open source hardware.

In this article we demonstrate how to use pcDuino to build a WiFi real-time video surveillance remote control robot. On the hardware side, we use the Motor Shield for Arduino to drive the motors, and servo of the PZT camera of Rover. On the software side, we use pcDuino Linux system to achieve a WiFi AP, video streaming server and TCP / IP communication. In the following sections, we details the hardware and software implementation.

Part I: Hardware

I. Tools Needed

Diagonal pliers, needle nose pliers, 3 mm sleeve, 2 mm flathead screwdriver, Phillips screwdriver 3 mm, 5 mm Phillips screwdriver, multimeter, soldering iron, solder, tape, tweezers, wallpaper knife





tweezer



adhesive tape



soldering

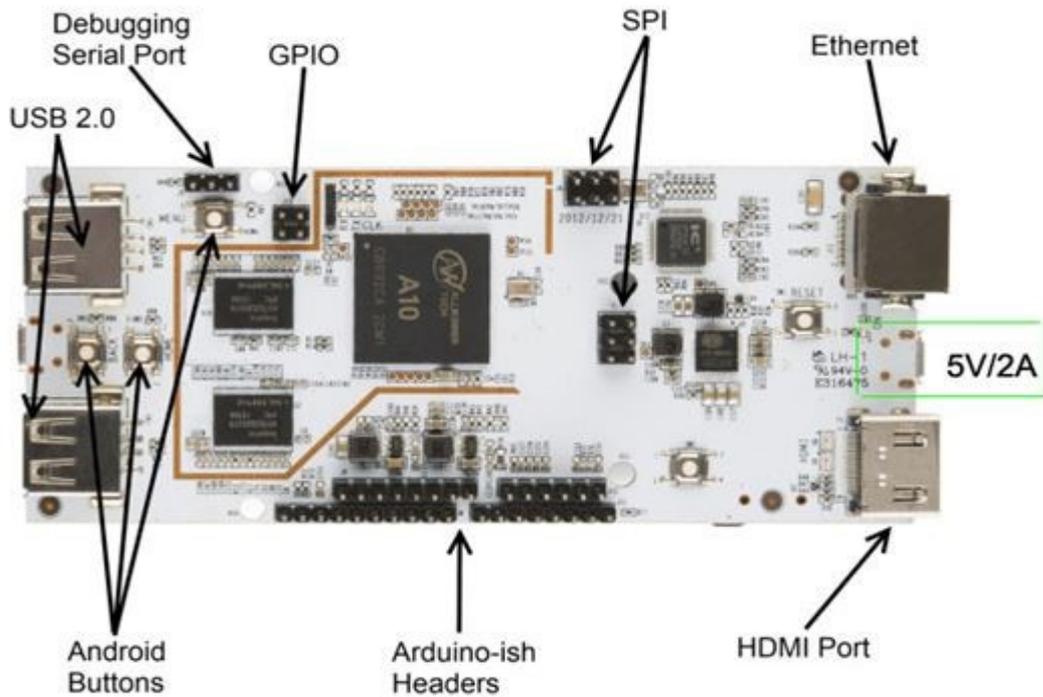


Wallpaper
cutter

II. Getting to Know Hardware

2.1 pcDuino

pcDuino is the brain of Rover. It manages the video streaming and WiFi communication and controls all the mechanical parts in Rover.



pcDuino's Hardware Features

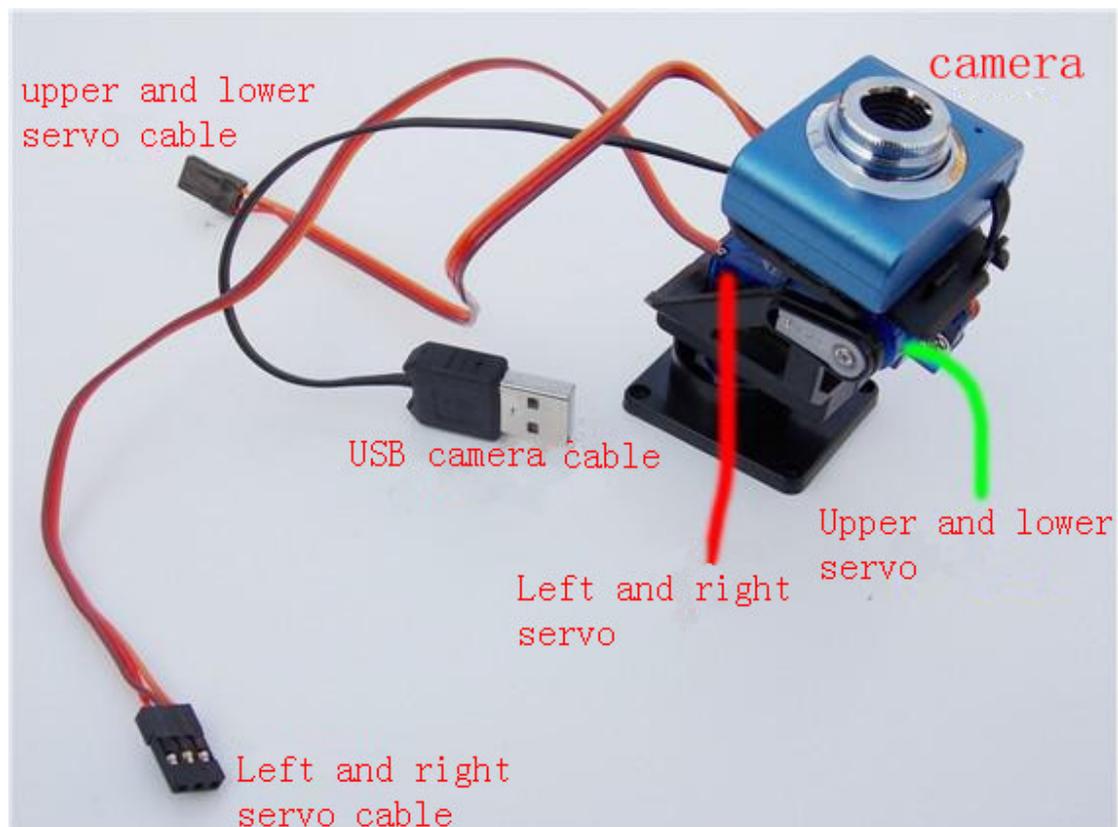
- CPU: 1GHz ARM Cortex A8 core
- GPU: OpenGL ES2.0, OpenVG 1.1 Mali 400 core
- DRAM: 1GB
- On-board storage: 2GB Flash, microSD socket expandable to 32GB
- Video output: HDMI
- Expansion interface: 2.54mm Headers compatible with Arduino
- Network: RJ45 interface and expandable USB WiFi dongle (not included)
- Power supply: 5V 2A
- Dimension: 125mm X 52mm

pcDuino's Software Features

- Operating system: Ubuntu
- API: All Arduino style expandable pins can be accessed with API, including UART, ADC, PWM, GPIO, SPI, I2C
- Programming languages: C, C++ with GNU tool chain, Python, Java

2.2 PZT Video Camera

The PTZ camera includes two servos: one with 180 degree vertical and the other one with 180 degree horizontal controls. The camera can be directly plugged into pcDuino with a USB cable.



Specification of the camera:

- Pixel: 30W
- Sensor: COMS
- Sensor size: 4386 * 3.64 mm
- Maximum resolution: 640 * 480
- Screen Format: 26bit RGB

- Interface: USB2.0 high speed transmission interface, compatible with USB1.0
- Auto white balance, color compensation

2.3 Power Supply

12V/2A power supply is used to charge the battery.

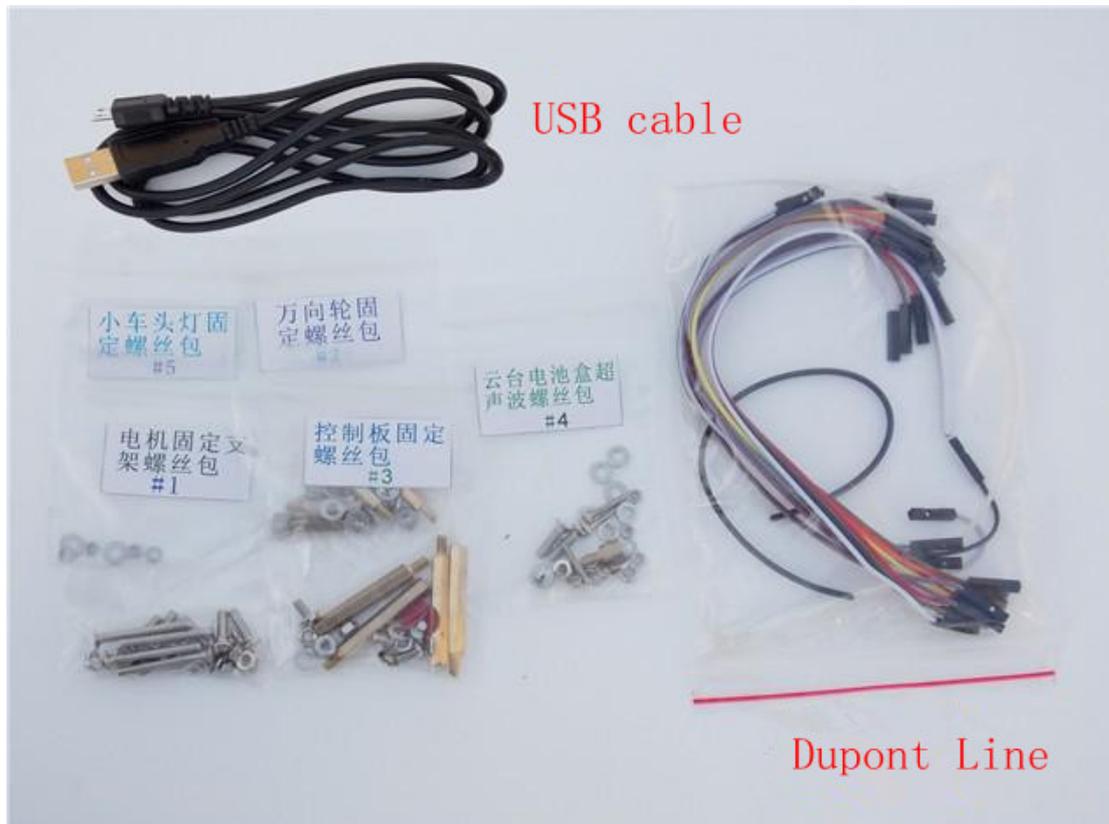


2.4 Battery

A high capacity lithium ion rechargeable battery is used. Please follow the safety issues of lithium batteries to avoid accidents. Its input voltage is 12.6VDC, and output voltage is 12.0VDC rated at 4800mAh.

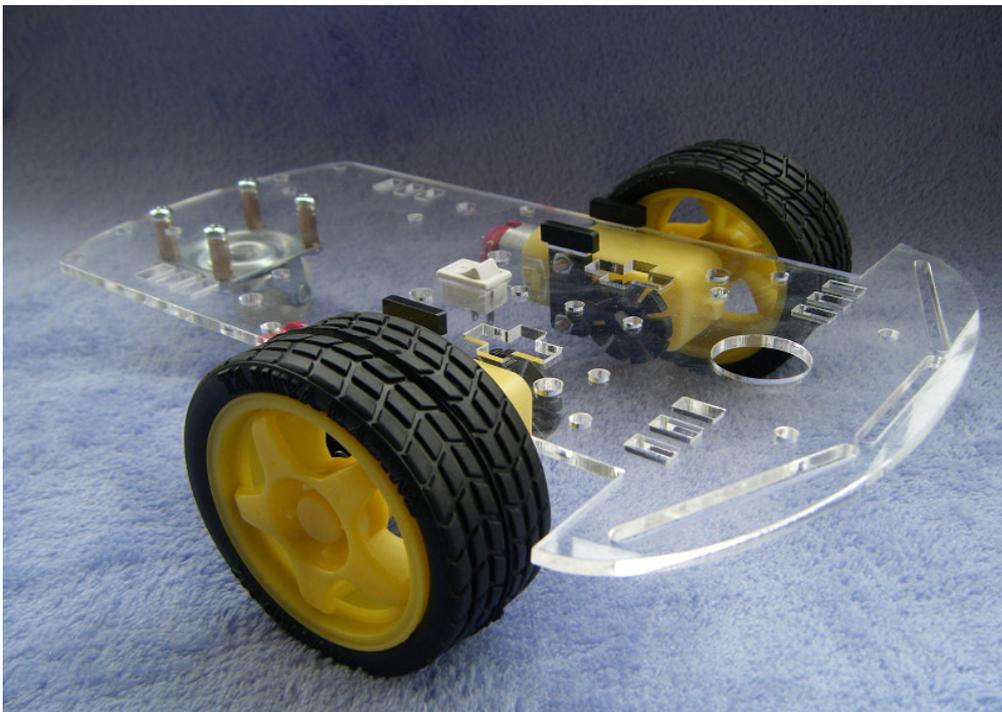


2.5 Wires, Screws, and Cables



2.6 Robot Platform

All major robot accessories are installed on the Robot platform. Please note that different models may appear different in the following pictures.



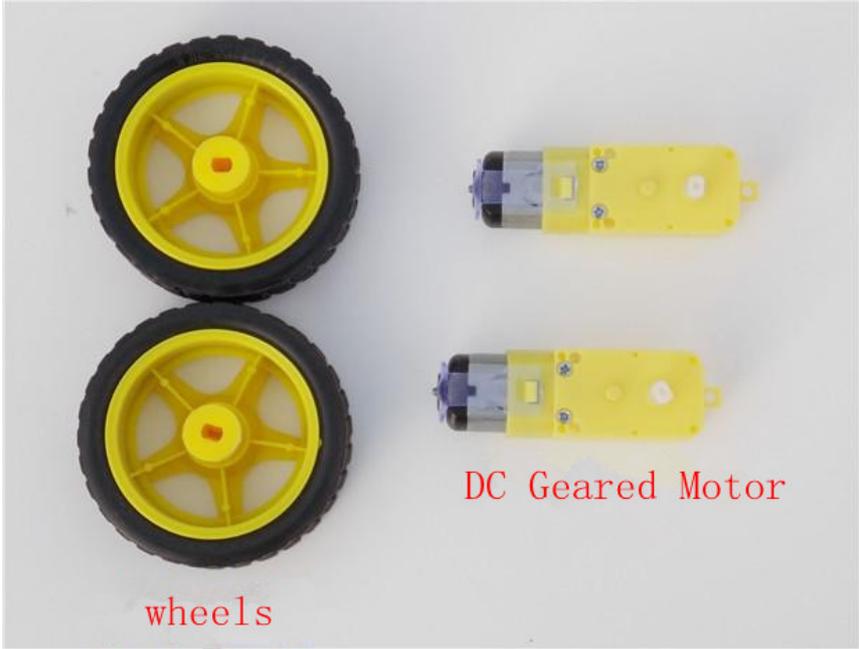
2.7 Caster

The caster supports the back of the robot platform. When mounting screws, please pay attention not to put the nut on the underneath side. It may affect the caster steering.



2.8 Trolley wheels and DC gear motors

DC gear motors drivers the robot.



2.9 DC motor mounting bracket

During installation, the side with three holes is attached to the robot platform, the side with two holes is attached to the DC gear motor.



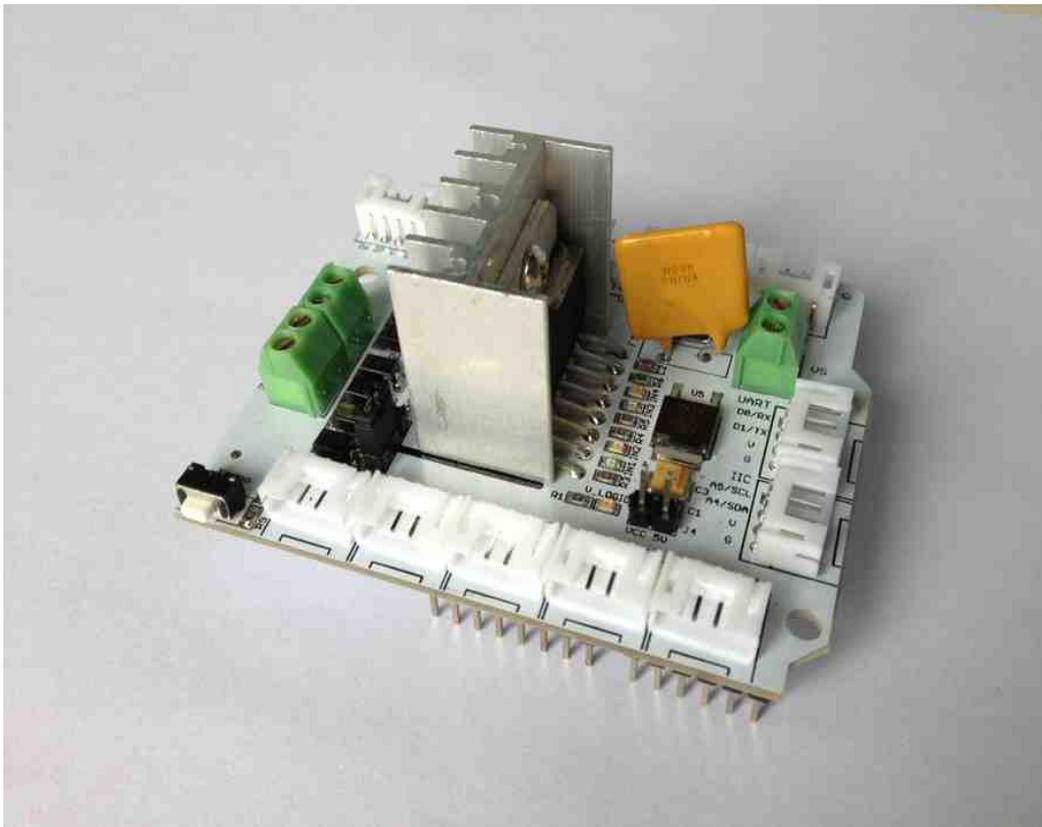
2.10 Motor shield for Arduino

Features:

- Compatible with Arduino
- PWM speed control mode
- 4 directions indicator
- Compatible with Linker Kit 2.54mm pitch Grove plug interface.

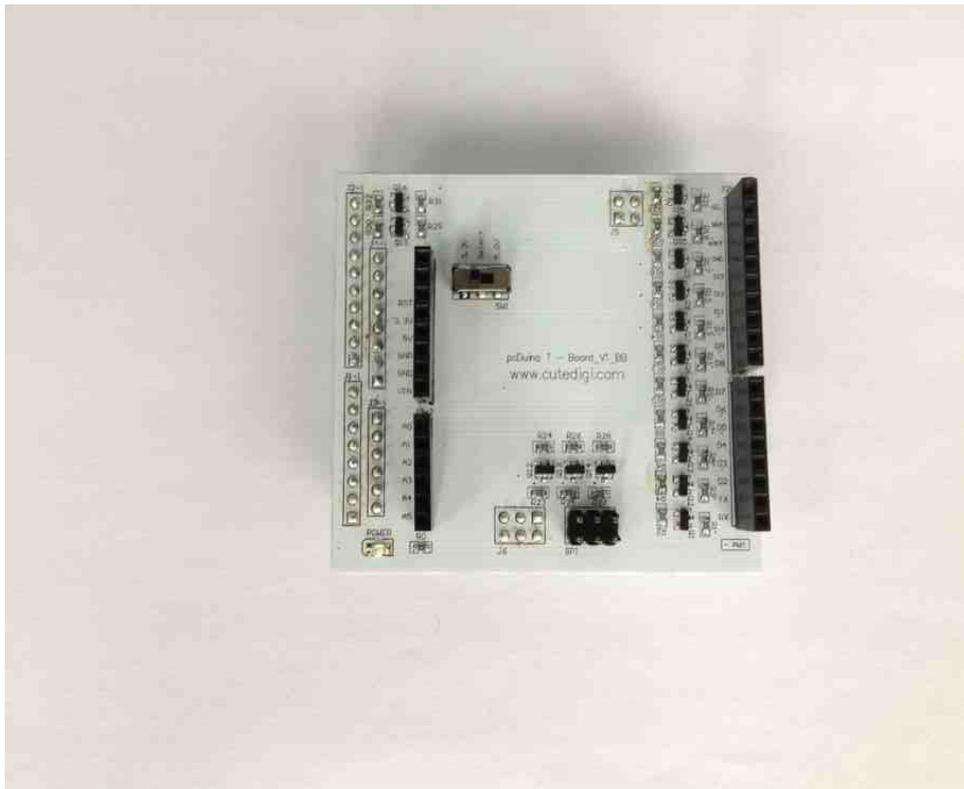
- larger models of cooling devices
- Support 14 servos
- Dimensions: 68.5x54.5x29.5 mm

Note: When the current exceeds 1000mA, the driver chip and heat sink will be very hot.



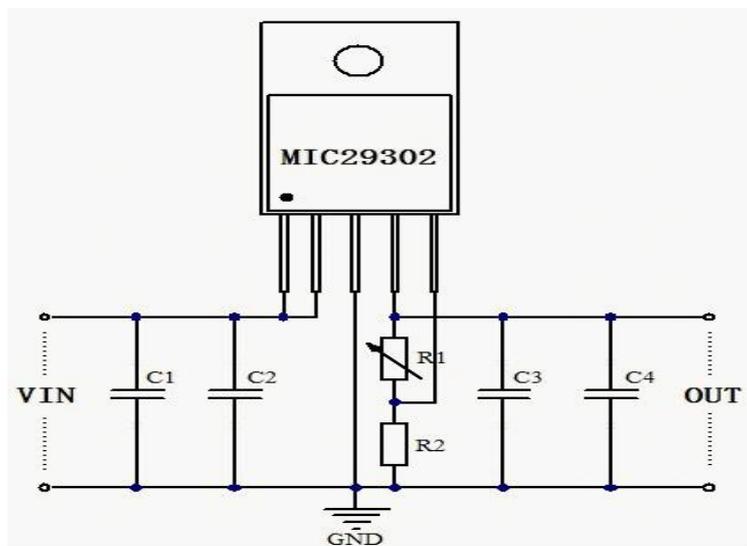
2.11 T-board

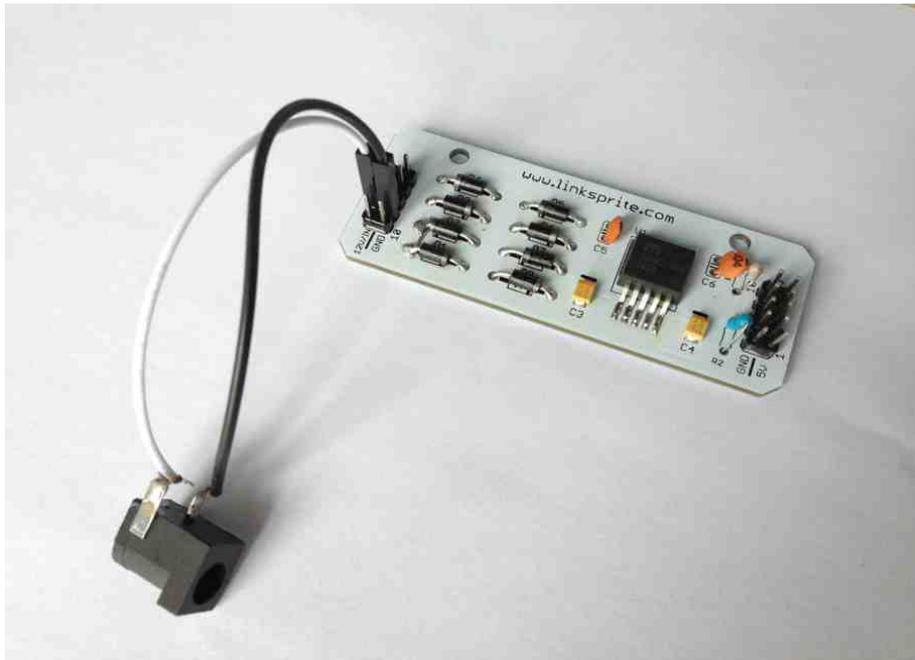
The T-board converts the 3.3V TTL signal to 5V. It is used to install the motor shield on pcDuino.



2.12 Power Module

The following is the schematic of the power module.





Dimensions: Length x width 75x29mm distance between two holes: 50mm

Input voltage: 12V

Output Voltage: 5V

Output Current: 2A

2.12 WiFi Dongle

It supports IEEE 802.11 b/g/n standards, supports USB 2.0 connections.

It is a low-cost compact WLAN module with dimensions: 32x13 mm.



III. Assemble the PTZ Camera

Before assembling the robot platform, we need to assemble and have the PTZ camera ready.

Step 1: Have the camera ready



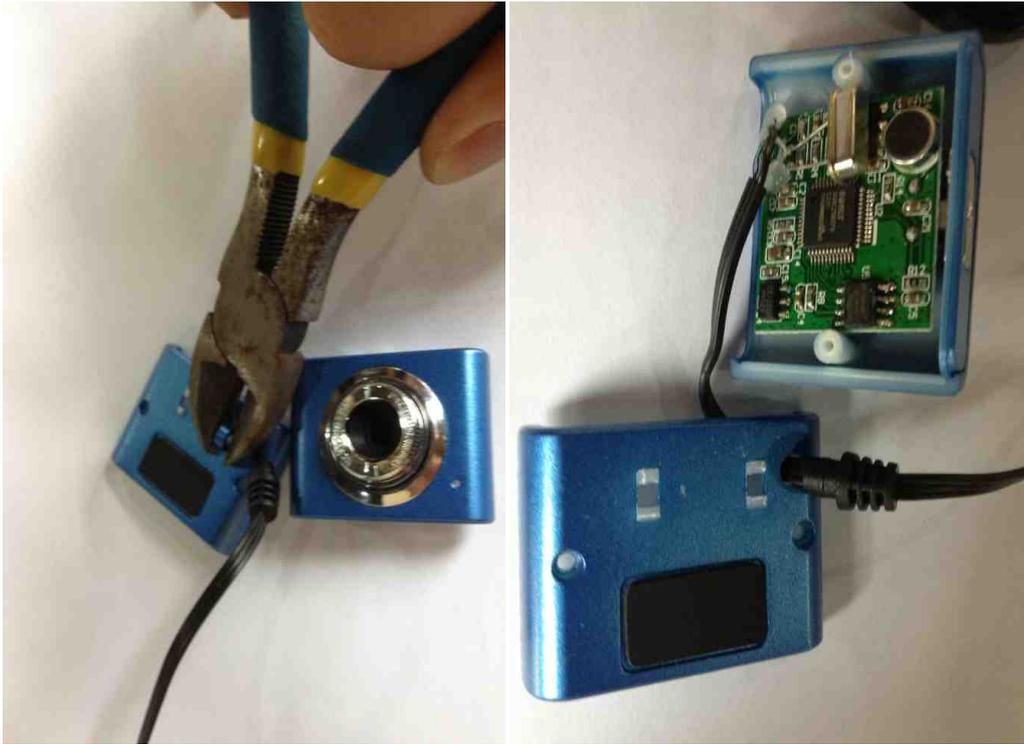
Step 2: Remove the needle



Step 3: Remove the screws on the bottom



Step 4: In order to attach the camera to the fixture, we need to drill holes and use screws. We can cut the two ears on the bottom and get two holes.



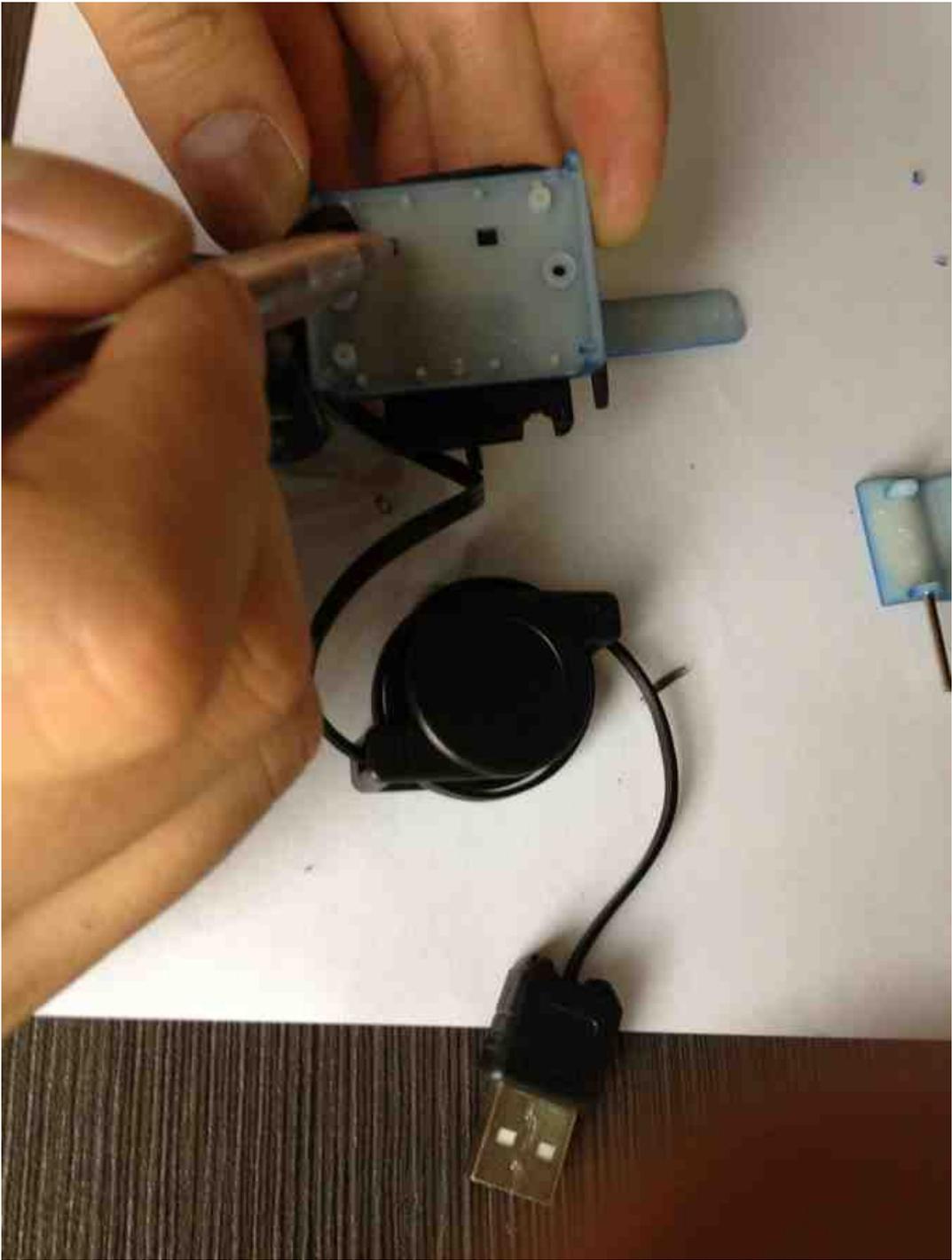
Step 5: Modify the camera fixture. Cut two sides off and smooth the surface.







Step 6: Align the fixture and bottom of the camera. Mark the hole position with pen or pencil, and drill two holes.





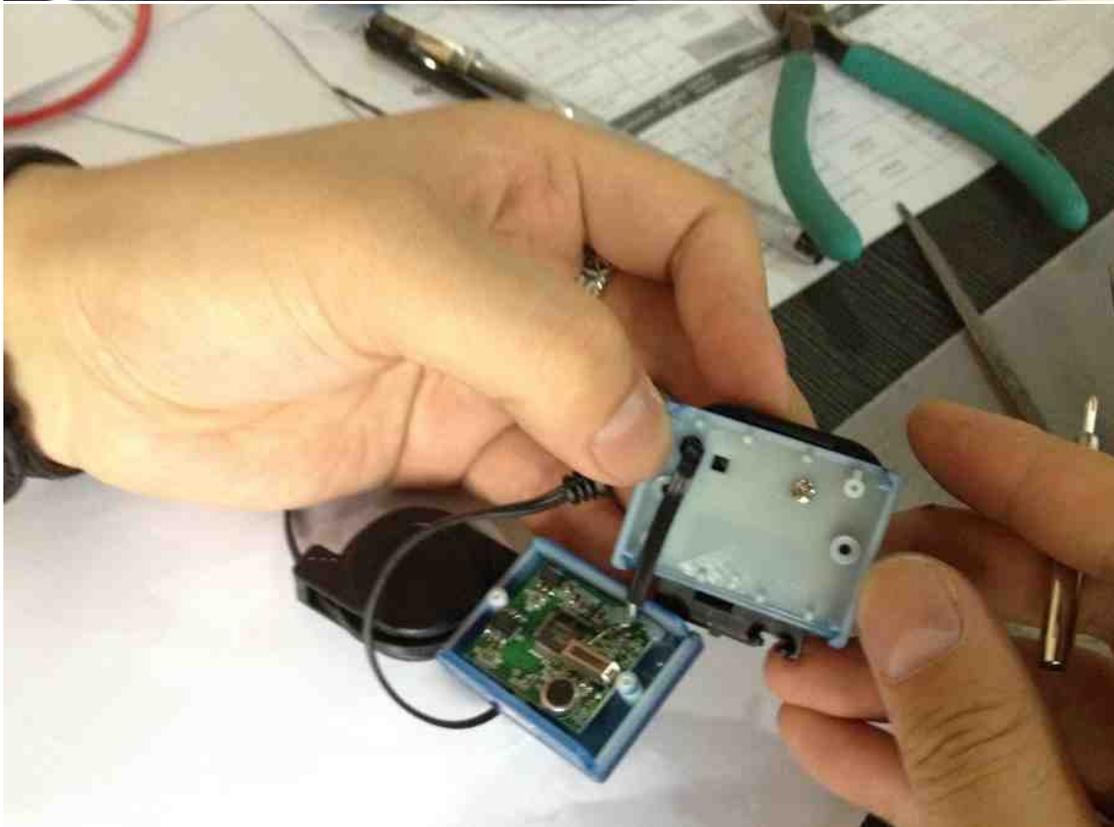
Step 7: In order not to block the two holes already on the bottom of camera, drill holes on both edges of the fixture.





Step 8: Put the bottom cover of the camera, and fixture together, insert

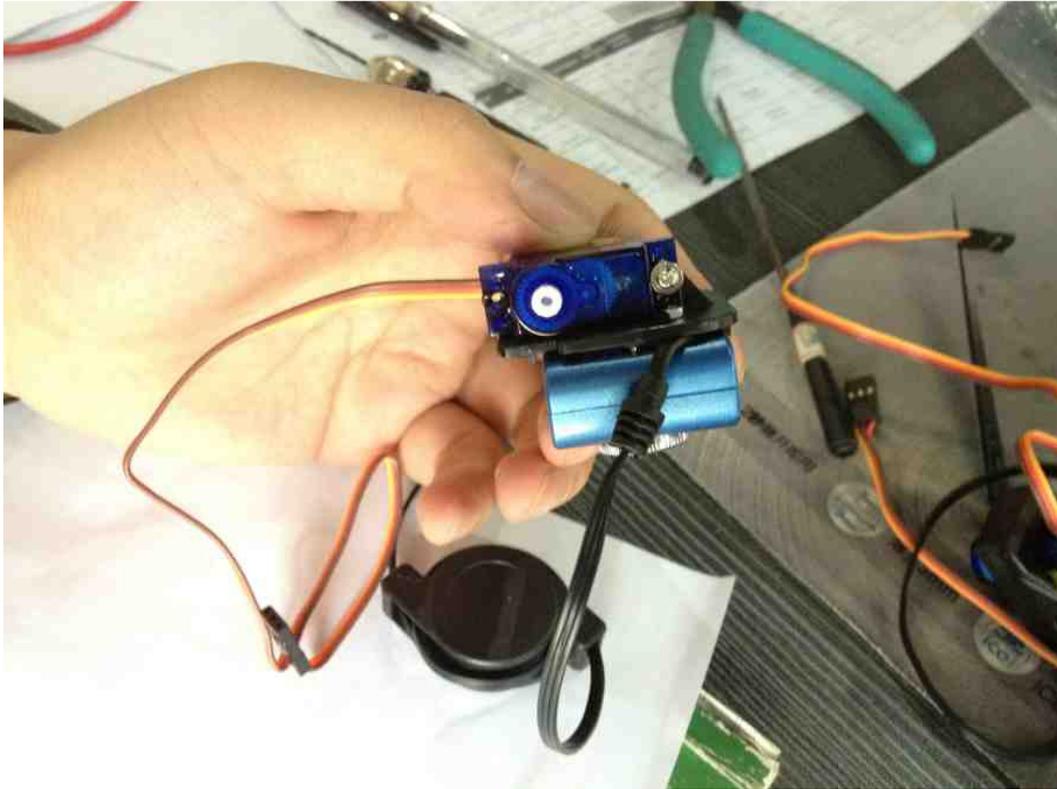
screws and tight up.



Step 9: The camera is assembled. Tight up with screws.



Step 10: Install the steering gear. Pay attention not to install in the wrong direction.

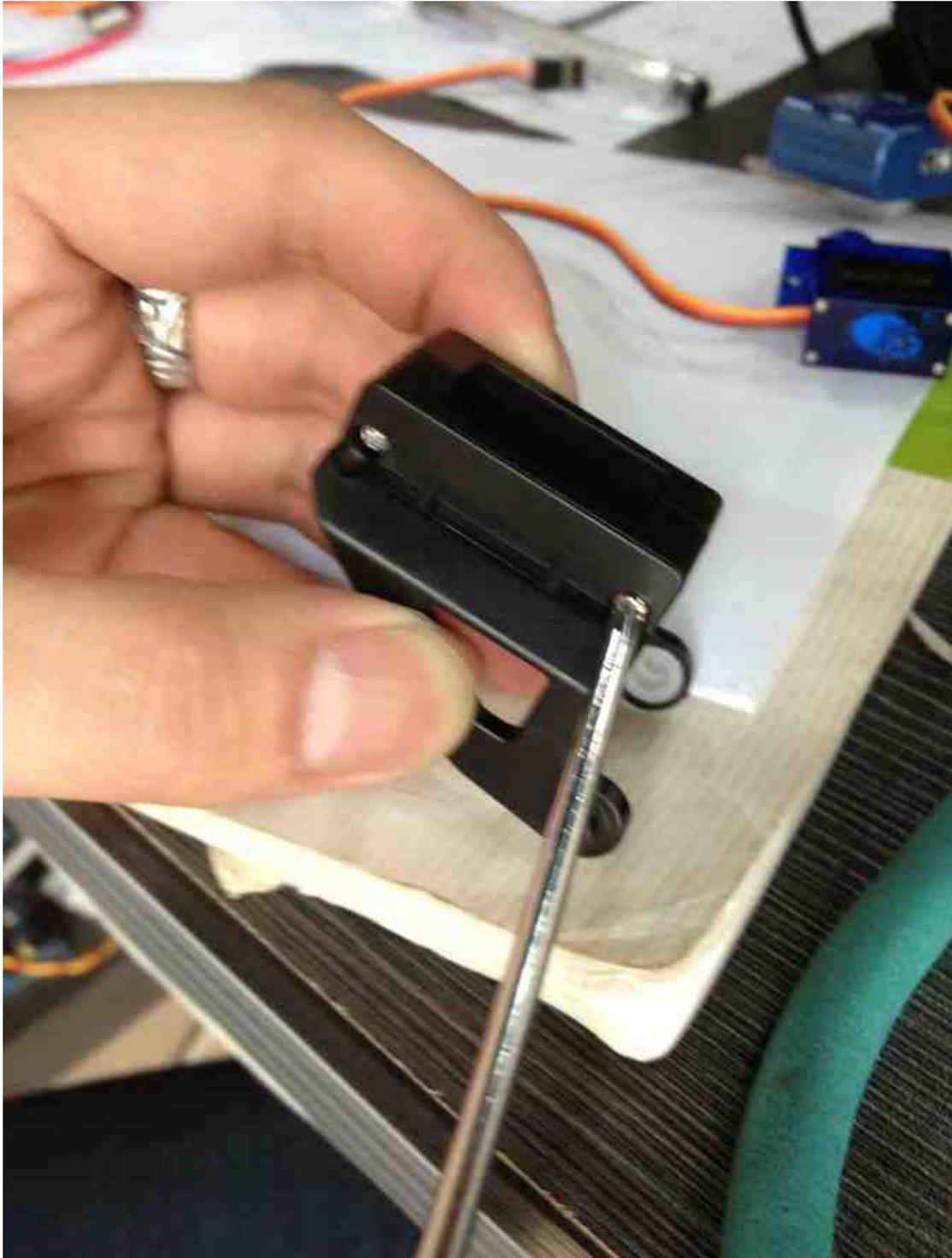


Step 11: Mount the head bracket. Compare to the opening , if the rocker arm is too long, cut off the excess part and tight up with screw.





Step 12: Disassemble the head bracket and install another steering gear.

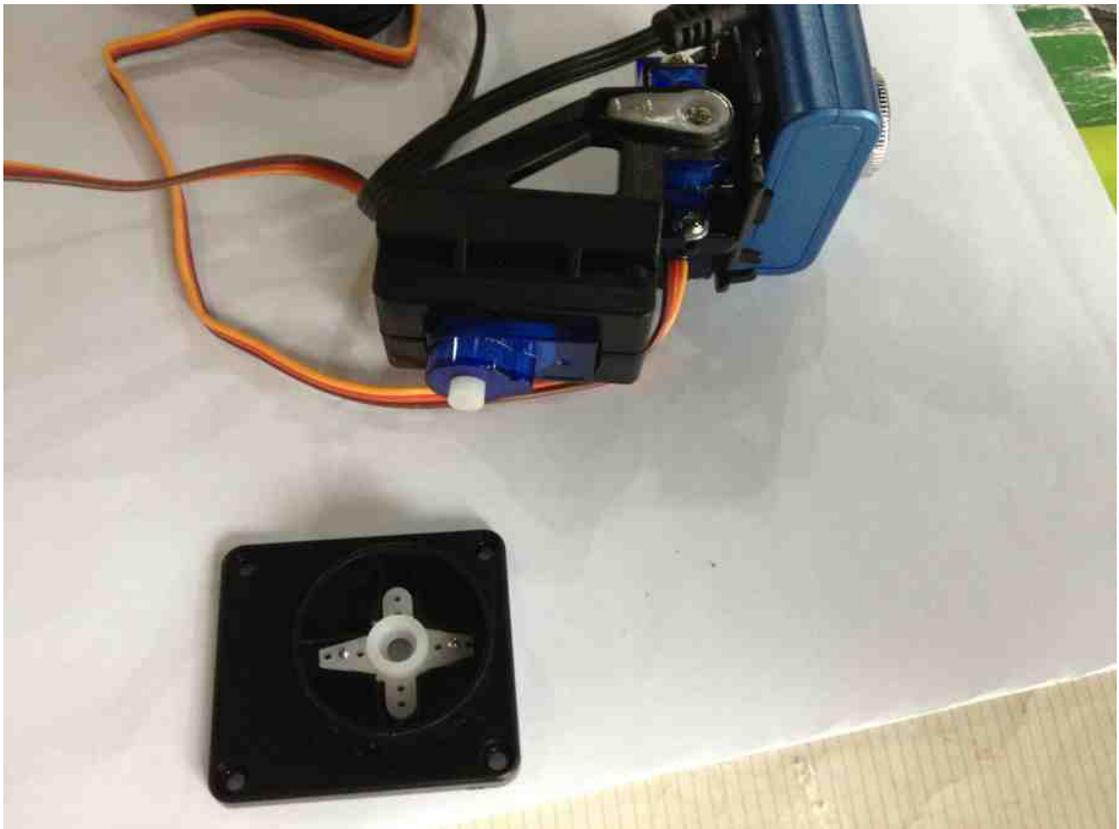


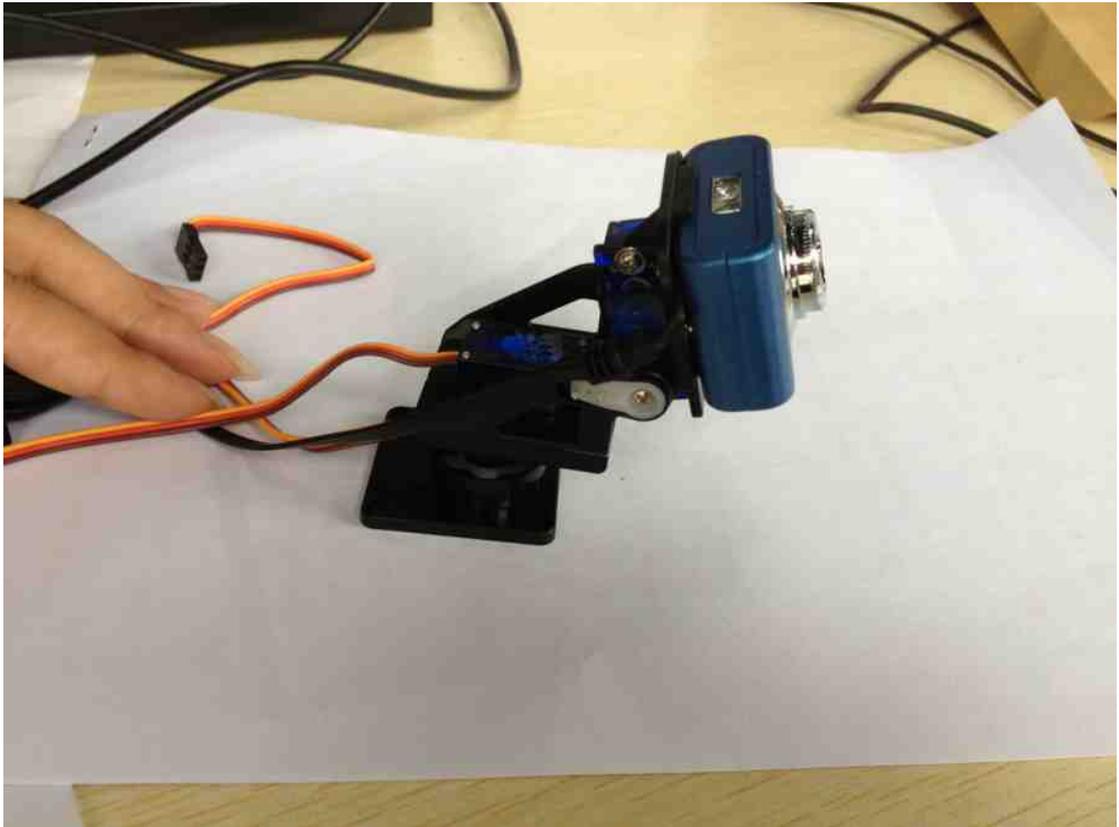


Step 13: Attach the camera to the PTZ fixture.



Step 14: Trim the multifunction steering foot and install it in the base. Insert the stand into the base, tight with screws. Then the assembling of the PTZ camera is completed.

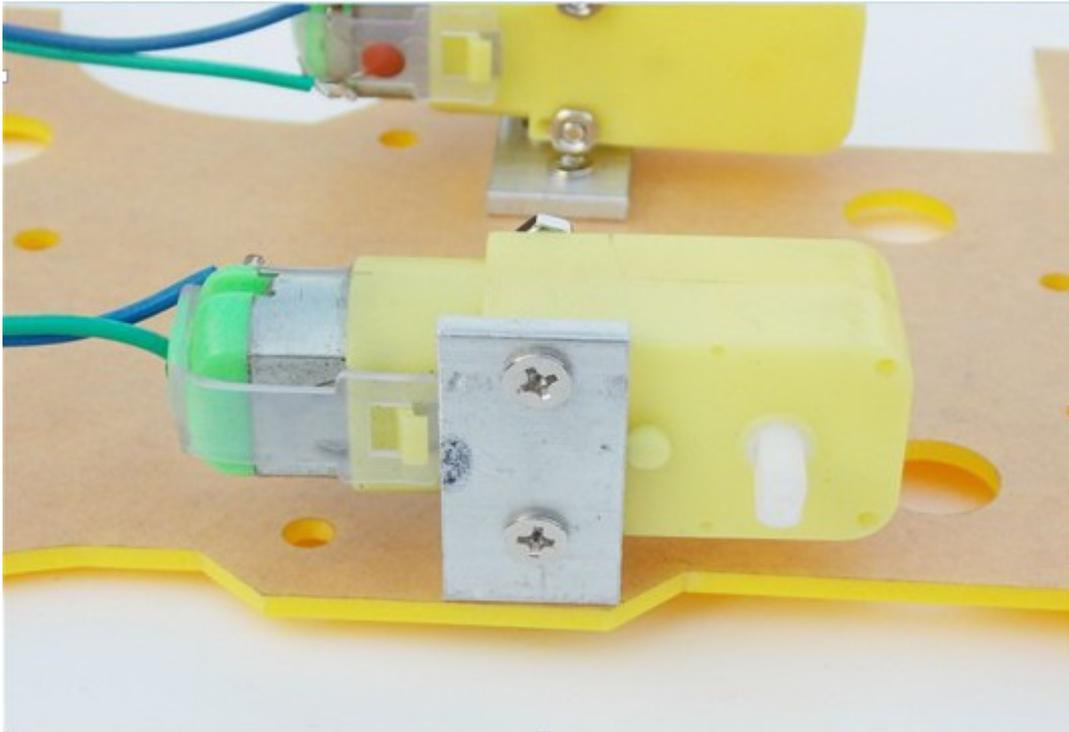




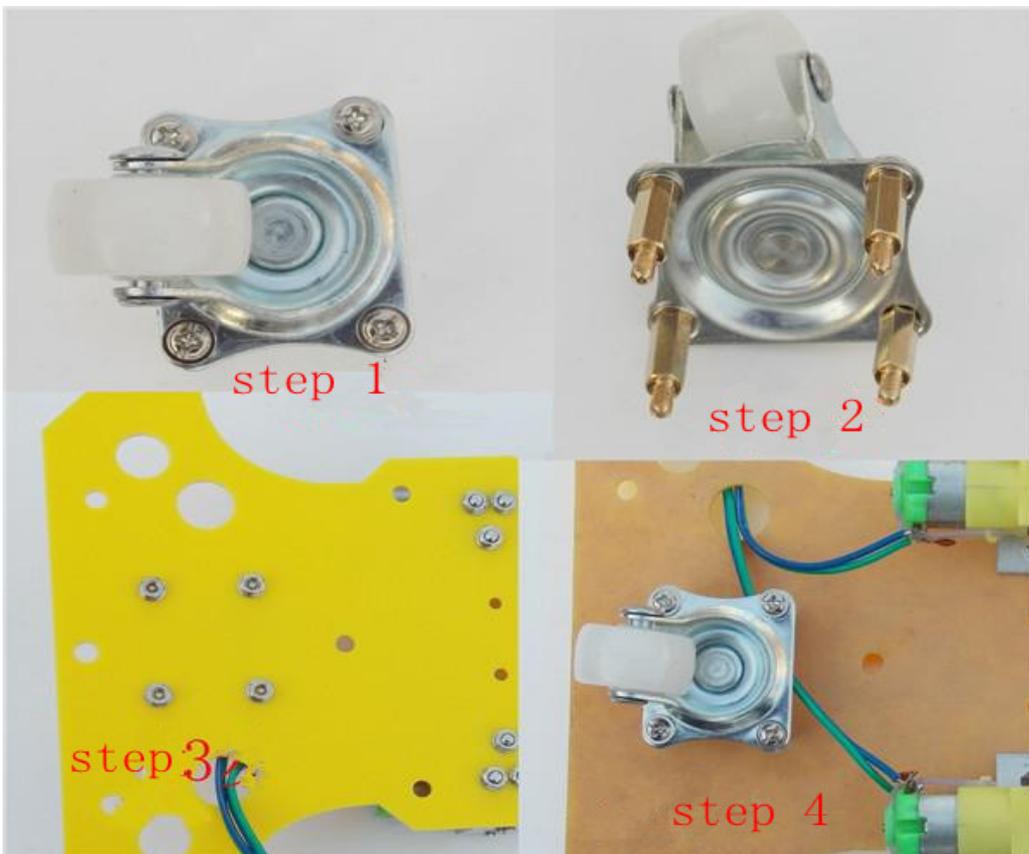
IV Assemble the Robot Platform

Step 1: Install the DC motor into position. The color of the motor may be different from the color of the chassis. First solder the wires to the motor, disregarding the polarity. A 0.1 μF ceramic capacitor can be used if available. It can prevent interference and prolog the motor's life cycle. Then install the motor bracket.

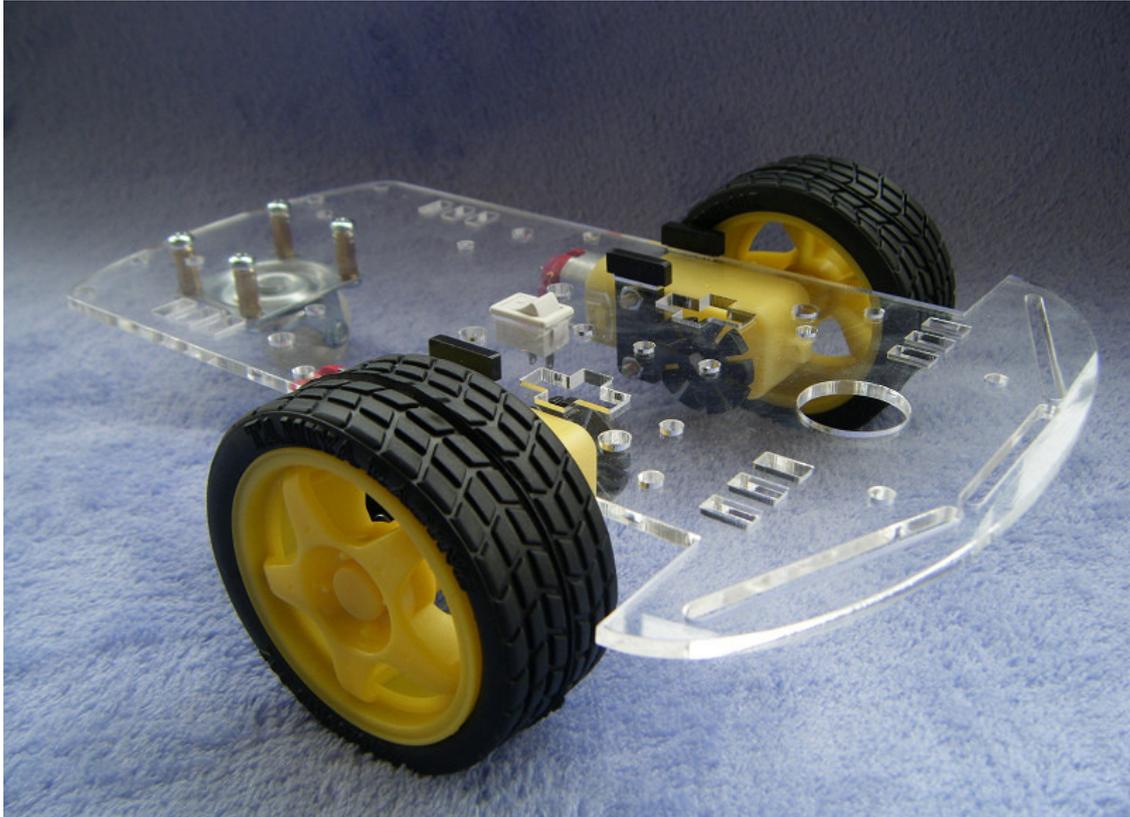




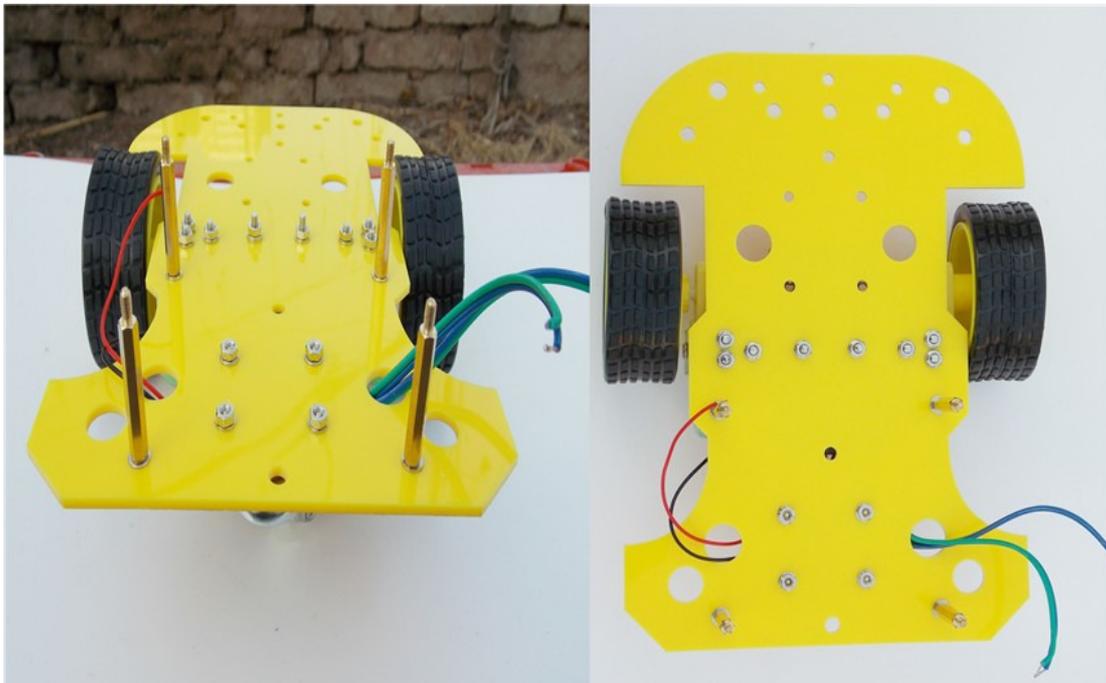
Step 2: Install casters.



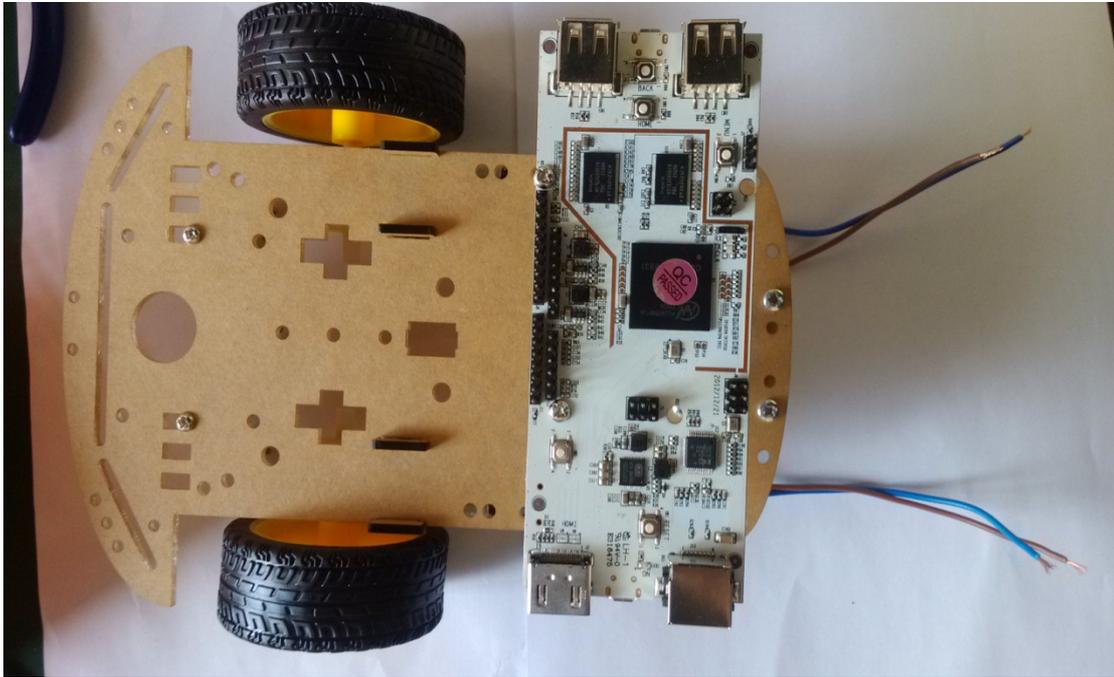
Step 3: Install wheels.



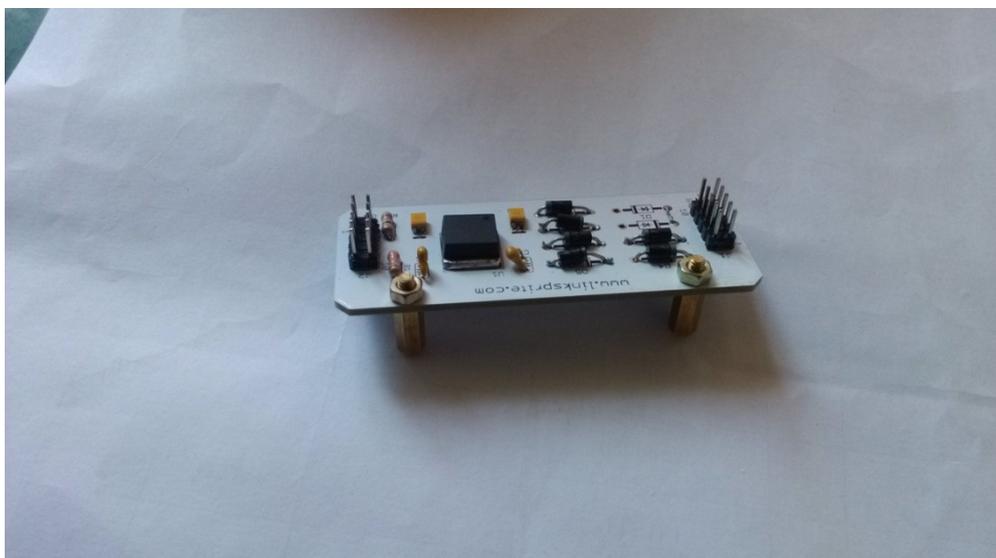
Step 4: Install copper studs that are used to hold the control board.

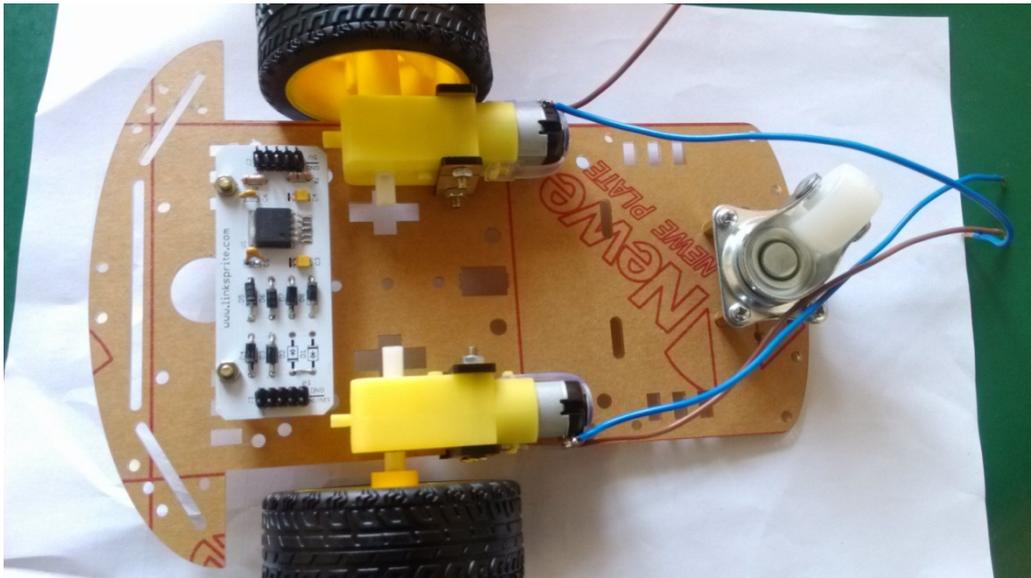


Step 5: Attach the pcDuino board to the chassis.



Step 6: Attach the power module board to the chassis. Please note that the module may be hot in operation. So be careful not to touch it with hands.

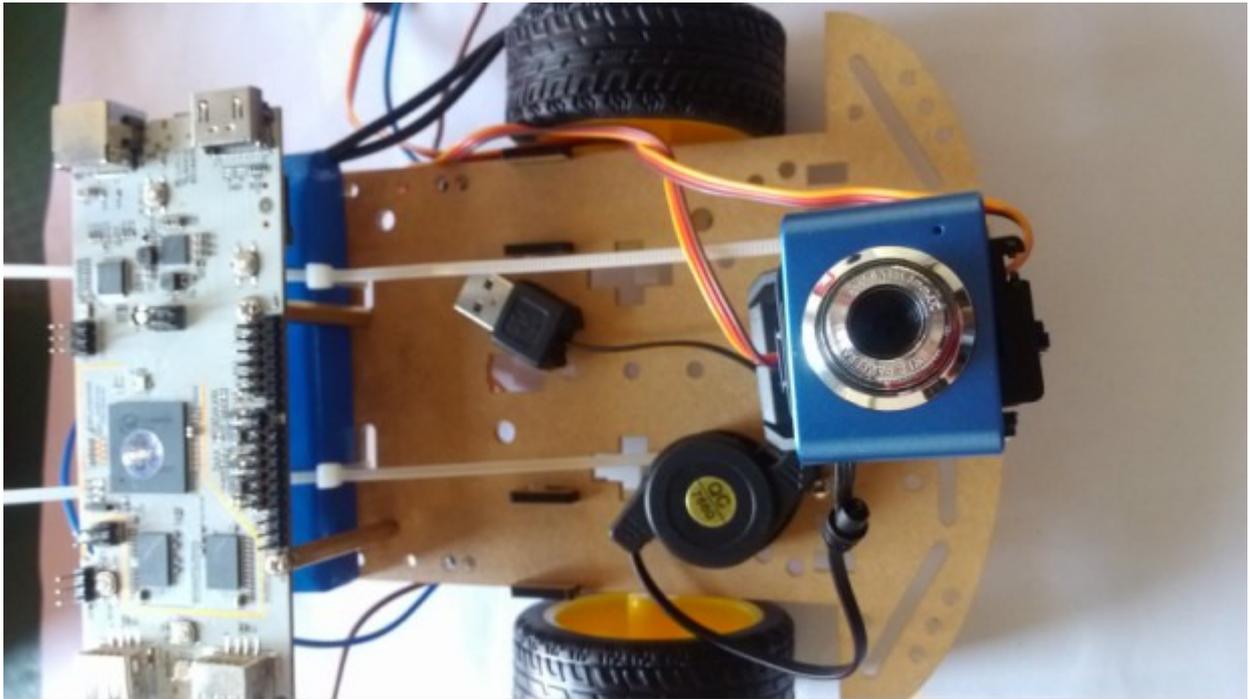




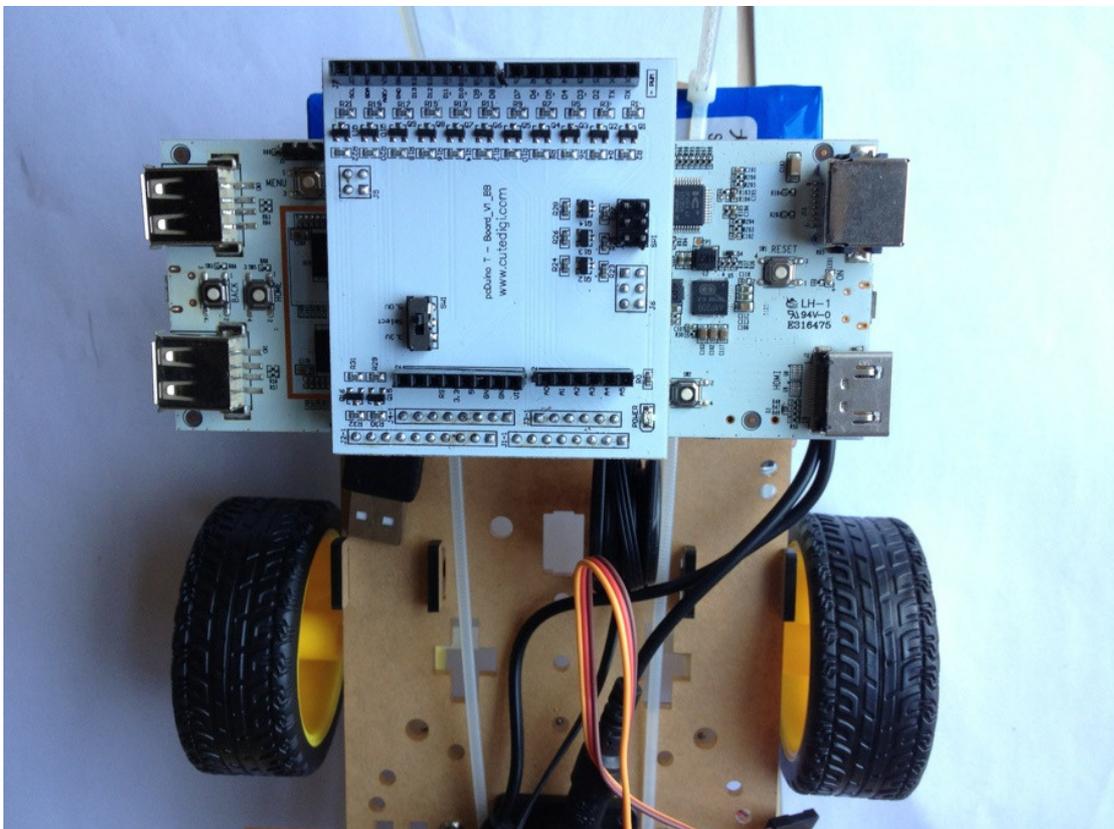
Step 7: Install battery.



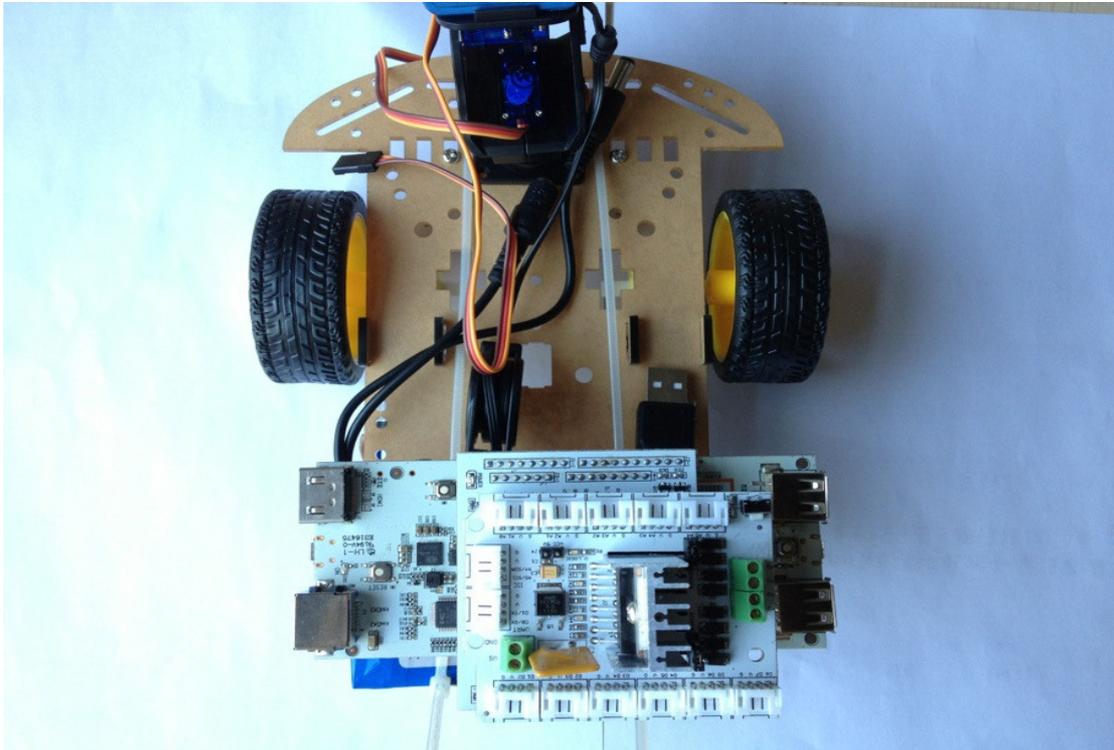
Step 8: Install the PTZ camera.



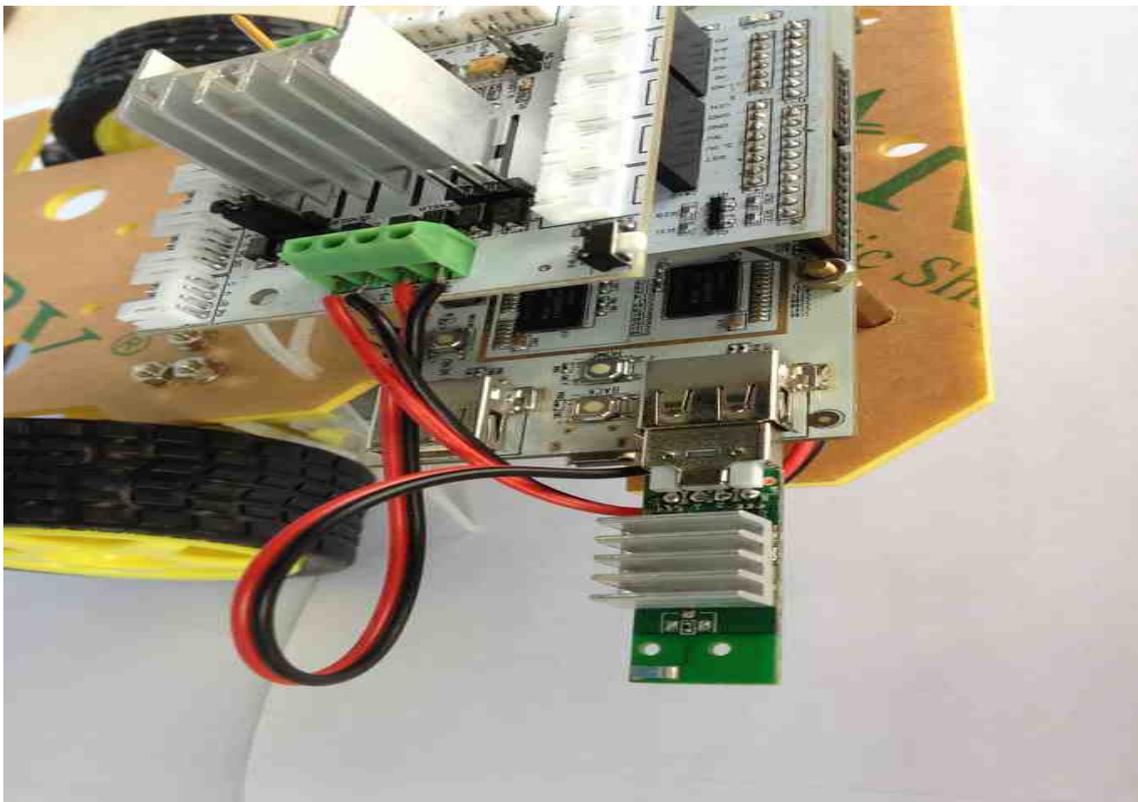
Step 9: Install T-board into pcDuino board. It is needed for the motor shield.



Step 10: Install the motor driver shield.

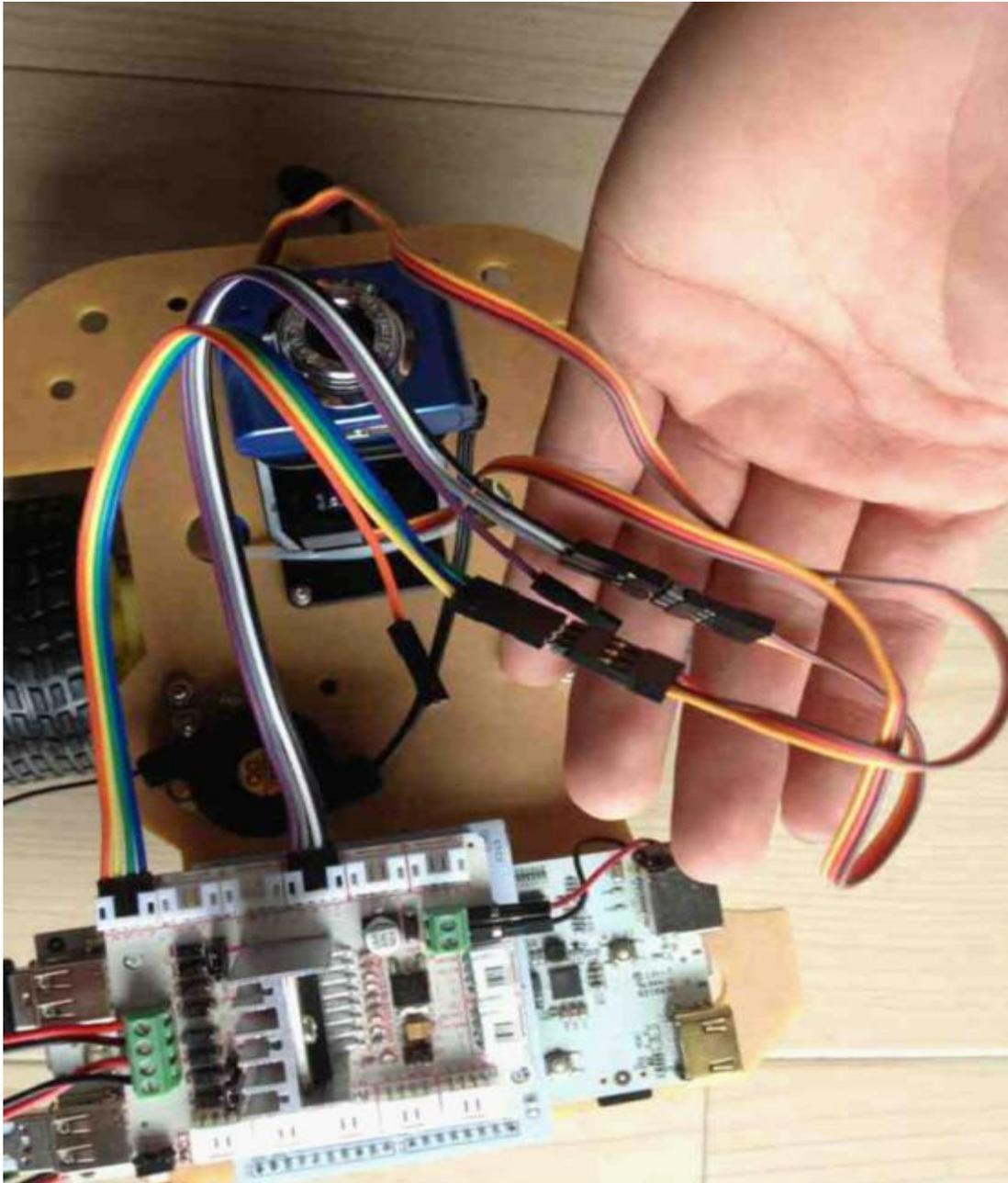


Step 11: Insert WiFi dongle into pcDuino.



We are almost there!

V. Wiring Instructions



Servo in Vertical Direction for PZT camera:

- Red (VCC) --> motor shield (VCC)

- Brown (GND) --> motor shield (GND)
- Orange (Control) --> motor shield (D7)

Servo in Horizontal Direction for PZT camera:

- ◆ Red (VCC) --> motor shield (VCC)
- ◆ Brown (GND) --> motor shield (GND)
- ◆ Orange (Control) --> motor shield (D4)

Power supply (12V) :

- (1) motor shield (VCC GND)
- (2) power adaptor (VIN GND)

pcDuino power supply:

DIY power line Red (VCC) --> OUT

Black (GND) --> GND

Part II: Software Part

The pcDuino serves as a WiFi AP, and a video stream server. It can transmit real-time video, and accept control command from client through WiFi. It controls Rover's movement and the camera's rotation.

VI. Configure pcDuino as a WiFi AP

The kernel (20130531) published on pcDuino website usually only provides WiFi STA mode driver. It does not have the iptables for IP forwarding either. So the first step is to replace the kernel Image of pcDuino.

You can download all files needed at

<https://s3.amazonaws.com/linksprite/pcduino/wifiAP.rar>.

Copy wifiAP to the directory ubuntu:

```
ubuntu@ubuntu:~$ ls
```

```
Desktop Downloads Pictures Templates c_enviroment wifiAP
```

```
Documents Music Public Videos sample
```

```
$ sudo mount /dev/nanda /boot
```

```
$ sudo cp uImage /boot -f
```

```
$ sudo cp evb.bin /boot -f
```

```
$ sudo cp 3.4.292B.tar.bz2 /lib/modules/
```

```
$ cd /lib/modules/
```

```
ubuntu@ubuntu:/lib/modules$ sudo tar -xvf 3.4.292B.tar.bz2
```

Restart pcDuino:

```
$ sudo reboot
```

Look up the software modules using the '\$lsmod' command:

```
root@ubuntu:/home/ubuntu# lsmod
```

Module	Size	Used by
rt5370sta	617141	0
8192cu	537048	0
rt2800usb	11321	0

```

rt2800lib      40721  1  rt2800usb
crc_ccitt      1094  1  rt2800lib
rt2x00usb     7245  1  rt2800usb
rt2x00lib     31040  3  rt2800usb,rt2800lib,rt2x00usb
mali_drm       2087  1
drm            157060  2  mali_drm
mac80211      257514  3  rt2800lib,rt2x00usb,rt2x00lib
cfg80211      150671  2  rt2x00lib,mac80211
mali           91258  0
disp_ump       823  0
ump            44002  4  mali,disp_ump

```

Restart and edit /etc/modules, remove rt5370sta:

```
$sudo modprobe rt2800usb
```

```
$echo 148F 5370 > /sys/bus/usb/drivers/rt2800usb/new_id
```

You can also edit the above command in a debugging serial port.

```

COM5 - PuTTY
[ 5.440000] sunxi-rtc sunxi-rtc: read time 2013-5-4 13:42:34
[ 5.450000] sunxi-rtc sunxi-rtc: setting system clock to 2013-05-04 13:42:34 UTC (1367674954)
[ 5.460000] ALSA device list:
[ 5.470000] #0: sun4i-sndhdm1
[ 5.480000] Freeing init memory: 1664K
[ 5.710000] fb_ioctl cmd=0x4600
[ 5.720000] do_fb_ioctl 1081, cmd=17920
[ 5.720000] Warning: this sunxi disp driver will see significant redesign.
[ 5.730000] Applications using /dev/disp directly will break.
[ 5.740000] For more information visit: http://linux-sunxi.org/Sunxi_disp_driver
[ 5.750000] fb_ioctl cmd=0x4700
[ 5.760000] do_fb_ioctl 1081, cmd=18176
[ 5.760000] disp: process 56 (setwindow) has skipped the version handshake.
[ 5.820000] EXT3-fs (nandd): using internal journal
[ 5.830000] kjournald starting. Commit interval 5 seconds
[ 5.840000] EXT3-fs (nandd): mounted filesystem with ordered data mode
[ 6.120000] init: plymouth main process (71) terminated with status 127
[ 6.170000] init: ureadahead main process (73) terminated with status 5
[ 11.490000] phy0 -> rt2x00lib_request_firmware: Error - Failed to request Firmware.

Welcome to Linaro 12.07 (GNU/Linux 3.4.29+ armv7l)

* Documentation: https://wiki.linaro.org/
root@ubuntu:~# [ 17.250000] disp: process 642 (setwindow) has skipped the version handshake.
[ 17.260000] disp: process 643 (setwindow) has skipped the version handshake.
[ 17.270000] disp: process 640 (setwindow) has skipped the version handshake.
[ 17.280000] disp: process 641 (setwindow) has skipped the version handshake.
[ 25.710000] disp: process 725 (setwindow) has skipped the version handshake.

root@ubuntu:~#
root@ubuntu:~# sudo modprobe rt2800usb
root@ubuntu:~# echo 148F 5370 > /sys/bus/usb/drivers/rt2800usb/new_id
root@ubuntu:~#

```

If we activate wlan, "\$ifconfig wlan3 up", we will see the following error

message:

```
root@ubuntu:~# ifconfig wlan3 up
[ 1043.640000] phy0 -> rt2x00lib_request_firmware: Error -
Failed to request Firmware.
SIOCSIFFLAGS: No such file or directory
```

It shows that there is no firmware for the USB WiFi Dongle. We need to put it into the right place.

Use command "\$modinfo rt2800usb" and find the firmware's name is

rt2870.bin.

```
root@ubuntu:~# modinfo rt2800usb
filename:
/lib/modules/3.4.29+/kernel/drivers/net/wireless/rt2x00/rt28
00usb.ko
license: GPL
firmware: rt2870.bin
description: Ralink RT2800 USB Wireless LAN driver.
version: 2.3.0
author: http://rt2x00.serialmonkey.com
```

Copy the firmware to correct directory:

```
$sudo cp rt2870.bin /lib/firmware/
```

Now we restart wlan, then we can find a WiFi connection:

```
"$sudo ifconfig wlan3 up"
```

Install iptables:

```
$sudo apt-get install iptables vim iw
```

After the installation is done, check if iptable has been installed successfully:

```
$sudo iptables -L
```

```
root@ubuntu:~# iptables -L
Chain INPUT (policy ACCEPT)
target prot opt source destination
```

```
Chain FORWARD (policy ACCEPT)
target prot opt source destination
```

```
Chain OUTPUT (policy ACCEPT)
target prot opt source destination
```

Download and edit hostapd:

In compiling we will need "libnl-dev" and "libssl-dev". So Let us install them first.

```
$sudo apt-get install libnl-dev libssl-dev
```

Hostpad 1.0 can be downloaded from <http://hostap.epitest.fi/releases/>.

A complete list of commands are:

```
ubuntu@ubuntu:~$ wget
http://hostap.epitest.fi/releases/hostapd-1.0.tar.gz
```

```
ubuntu@ubuntu:~$ tar xvf hostapd-1.0.tar.gz
ubuntu@ubuntu:~$ cd hostapd-1.0
ubuntu@ubuntu:~/hostapd-1.0$ cd hostapd/
ubuntu@ubuntu:~/hostapd-1.0/hostapd$ vim defconfig
```

Find the line with "#CONFIG_IEEE80211N=y", remove '#', save and exit.

```
ubuntu@ubuntu:~/hostapd-1.0/hostapd$ cp defconfig .config
```

```
ubuntu@ubuntu:~/hostapd-1.0/hostapd$ make
CC main.c
CC config_file.c
CC ../src/ap/hostapd.c
CC ../src/ap/wpa_auth_glue.c
CC ../src/ap/drv_callbacks.c
CC ../src/ap/ap_drv_ops.c
CC ../src/ap/utils.c
CC ../src/ap/authsrv.c
```

```
ubuntu@ubuntu:~/hostapd-1.0/hostapd$ sudo make install
mkdir -p /usr/local/bin
for i in hostapd hostapd_cli; do cp -f $i /usr/local/bin/$i; done
```

Check if hostpad has been installed successfully:

```
ubuntu@ubuntu:~/hostapd-1.0/hostapd$ hostapd -v
hostapd v1.0
User space daemon for IEEE 802.11 AP management,
IEEE 802.1X/WPA/WPA2/EAP/RADIUS Authenticator
Copyright (c) 2002-2012, Jouni Malinen <j@w1.fi> and
contributors
```

Now we begin to configure hostpad.

We hope our hot spot is configured as following,

```
SSID: ssid=pcduino
Password: wpa_pass
phrase=1234567890
Encryption: wpa_key_mgmt=WPA-PSK
```

Configure as following:

```
root@ubuntu:~# nano /etc/hostapd.conf
```

```
interface=wlan3
driver=nl80211
ssid=pcduino
hw_mode=g
channel=11
dtim_period=1
rts_threshold=2347
fragm_threshold=2346
macaddr_acl=0
auth_algs=1
ieee80211n=0
wpa=2
wpa_passphrase=1234567890
wpa_key_mgmt=WPA-PSK
wpa_pairwise=TKIP
rsn_pairwise=CCMP
```

Note: if wlan3 may change, we need to use "ipconfig" to check.

After configuration is done, we use the following commands to check the configuration.

```
root@ubuntu:~# hostapd -dd /etc/hostapd.conf
random: Trying to read entropy from /dev/random
Configuration file: /etc/hostapd.conf
nl80211: interface wlan4 in phy phy0
rfkill: initial event: idx=0 type=2 op=0 soft=0 hard=0
```

```
rfkill: initial event: idx=1 type=1 op=0 soft=0 hard=0
nl80211: Using driver-based off-channel TX
nl80211: Register frame command failed (type=208): ret=-114 (Operation
already in progress)
nl80211: Register frame match - hexdump(len=1): 06
nl80211: Failed to register Action frame processing - ignore for now
nl80211: Add own interface ifindex 5
nl80211: Set mode ifindex 5 iftype 3 (AP)
nl80211: Create interface iftype 6 (MONITOR)
nl80211: New interface mon.wlan4 created: ifindex=7
nl80211: Add own interface ifindex 7
BSS count 1, BSSID mask 00:00:00:00:00:00 (0 bits)
nl80211: Regulatory information - country=00
nl80211: 2402-2472 @ 40 MHz
nl80211: 2457-2482 @ 40 MHz
nl80211: 2474-2494 @ 20 MHz
nl80211: 5170-5250 @ 40 MHz
nl80211: 5735-5835 @ 40 MHz
nl80211: Added 802.11b mode based on 802.11g information
Allowed channel: mode=1 chan=1 freq=2412 MHz max_tx_power=20 dBm
Allowed channel: mode=1 chan=2 freq=2417 MHz max_tx_power=20 dBm
Allowed channel: mode=1 chan=3 freq=2422 MHz max_tx_power=20 dBm
Allowed channel: mode=1 chan=4 freq=2427 MHz max_tx_power=20 dBm
Allowed channel: mode=1 chan=5 freq=2432 MHz max_tx_power=20 dBm
Allowed channel: mode=1 chan=6 freq=2437 MHz max_tx_power=20 dBm
Allowed channel: mode=1 chan=7 freq=2442 MHz max_tx_power=20 dBm
Allowed channel: mode=1 chan=8 freq=2447 MHz max_tx_power=20 dBm
Allowed channel: mode=1 chan=9 freq=2452 MHz max_tx_power=20 dBm
Allowed channel: mode=1 chan=10 freq=2457 MHz max_tx_power=20 dBm
Allowed channel: mode=1 chan=11 freq=2462 MHz max_tx_power=20 dBm
Allowed channel: mode=0 chan=1 freq=2412 MHz max_tx_power=20 dBm
Allowed channel: mode=0 chan=2 freq=2417 MHz max_tx_power=20 dBm
Allowed channel: mode=0 chan=3 freq=2422 MHz max_tx_power=20 dBm
Allowed channel: mode=0 chan=4 freq=2427 MHz max_tx_power=20 dBm
Allowed channel: mode=0 chan=5 freq=2432 MHz max_tx_power=20 dBm
Allowed channel: mode=0 chan=6 freq=2437 MHz max_tx_power=20 dBm
Allowed channel: mode=0 chan=7 freq=2442 MHz max_tx_power=20 dBm
Allowed channel: mode=0 chan=8 freq=2447 MHz max_tx_power=20 dBm
Allowed channel: mode=0 chan=9 freq=2452 MHz max_tx_power=20 dBm
```

```
Allowed channel: mode=0 chan=10 freq=2457 MHz max_tx_power=20 dBm
Allowed channel: mode=0 chan=11 freq=2462 MHz max_tx_power=20 dBm
Completing interface initialization
Mode: IEEE 802.11g Channel: 11 Frequency: 2462 MHz
nl80211: Set freq 2462 (ht_enabled=0 sec_channel_offset=0)
RATE[0] rate=10 flags=0x1
RATE[1] rate=20 flags=0x1
RATE[2] rate=55 flags=0x1
RATE[3] rate=110 flags=0x1
RATE[4] rate=60 flags=0x0
RATE[5] rate=90 flags=0x0
RATE[6] rate=120 flags=0x0
RATE[7] rate=180 flags=0x0
RATE[8] rate=240 flags=0x0
RATE[9] rate=360 flags=0x0
RATE[10] rate=480 flags=0x0
RATE[11] rate=540 flags=0x0
Flushing old station entries
```

Next install the DHCP server.

```
$sudo apt-get install dhcp3-server
```

After it is done, configure the DHCP server:

```
$sudo nano /etc/dhcp/dhcpd.conf
```

Add the following lines at end of the file:

```
subnet 192.168.0.0 netmask 255.255.255.0
{
range 192.168.0.2 192.168.0.10;
option routers 192.168.0.1; #router address
option domain-name-servers 8.8.8.8;
}
```

Restart hostapd:

```
root@ubuntu:~# killall hostapd
hostapd: no process found
root@ubuntu:~# hostapd -B /etc/hostapd.conf
Configuration file: /etc/hostapd.conf
Using interface wlan4 with hwaddr 00:c3:16:a0:03:00 and ssid 'pcduino'
```

Set pcDuino's WiFi IP address:

```
root@ubuntu:~# ifconfig wlan4 192.168.0.1
```

Turn on DHCP:

```
root@ubuntu:~# dhcpd wlan4 -pf /var/run/dhcp-server/dhcpd.pid
Internet Systems Consortium DHCP Server 4.1-ESV-R4
Copyright 2004-2011 Internet Systems Consortium.
All rights reserved.
For info, please visit https://www.isc.org/software/dhcp/
Wrote 0 leases to leases file.
Listening on LPF/wlan4/00:c3:16:a0:03:00/192.168.0.0/24
Sending on LPF/wlan4/00:c3:16:a0:03:00/192.168.0.0/24
Sending on Socket/fallback/fallback-net
```

Enable IP forwarding:

```
root@ubuntu:~# echo 1 >/proc/sys/net/ipv4/ip_forward
```

Set NAT:

```
root@ubuntu:~# iptables -t nat -A POSTROUTING -o eth0 -j
MASQUERADE
```

After the above configuration is done, we should see a hot spot named "pcduino".

Note: In debugging, check your WiFi using "ipconfig" and change wlan accordingly.

VII. Install Video Streaming Server

We use the open source package mjpg-streamer to implement the video transmission server. The client can be a browser or special APP.

First install a few software packages.

```
$sudo apt-get install libv4l-dev
$sudo apt-get install libjpeg8-dev
$sudo apt-get install subversion
$sudo apt-get install imagemagick
```

The libv4l-dev and libjpeg8-dev are libraries. The Subversion and ImageMagick are needed to compile the mjpg-streamer source code.

Next download, compile and install mjpg-streamer video server software.

```
$svn co https://mjpg-streamer.svn.sourceforge.net/svnroot/mjpg-streamer mjpg-streamer
$cd mjpg-streamer/mjpg-streamer
$make USE_LIBV4L2=true clean all
$sudo make DESTDIR=/usr install
```

After installation is done, we need to open the jpg-streamer video server. The command to turn on the service is:

```
$mjpg_streamer -i "/usr/lib/input_uvc.so -d /dev/video0 -y -r
320x240 -f 10" -o "/usr/lib/output_http.so -p 8090 -w
/var/www/mjpg_streamer"
```

The parameters are:

-d: device
-r: resolution

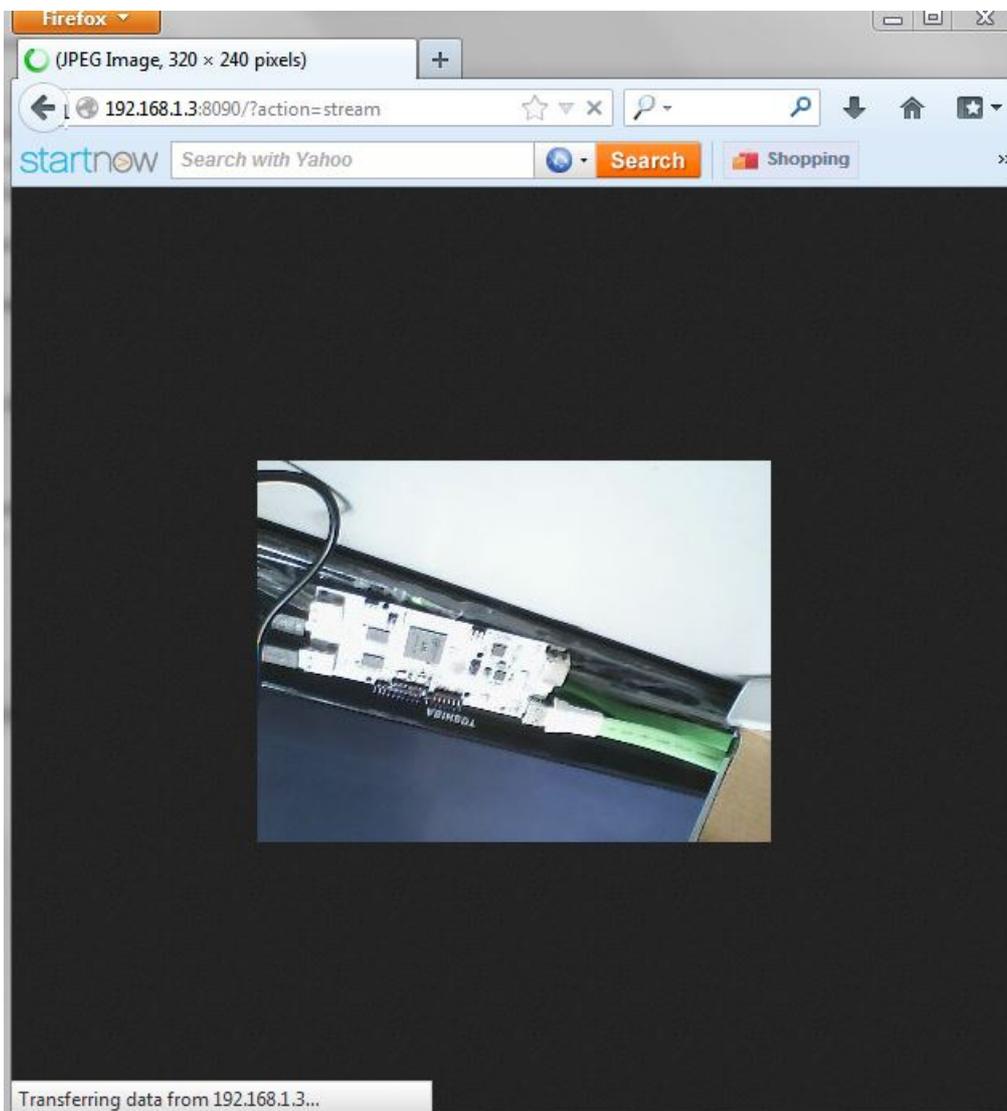
-f: frequency

If your camera does not support the MJPEG format, we can use '-y' to specify to use YUYV format instead of the MJPEG format.

-p: port

-w: web directory

On a pcDuino terminal we use the command "ipconfig" to get the IP address of the pcDuino. Then in a browser in a client, visit 192.168.1.3:8090/?action=stream to access the real-time video. The 192.168.1.3 is the IP address of the pcDuino.



VIII. Install the Control Software

1) The source code for rover control "car_test.c" can be found in appendix. Copy the source code to C_environment/sample, open makefile in and add car_test after OBJS=

Next change

```
@for i in $(OBJJS); do $(CXX) $(INCS) $$i.c -o $(TARGET)/$$i $(LIBS);  
done
```

to

```
@for i in $(OBJJS); do $(CXX) $(INCS) $$i.c -o $(TARGET)/$$i $(LIBS)  
-pthread; done
```

Save and compile. An executable file car_test will be generated in directory output/put.

2) Turn on and run corresponding software and program: car_test can be added into /etc/rc.local,

```
~$ sudo vim /etc/rc.local
```

Add the full directory name before exit(0), for example/home/ubuntu/c_environment/output/test/car_test.

Video service needs command mjpg_streamer to run. This can be edited in mjpg.sh:

```
~$ vim mjpg.sh
```

Add

```
#!/bin/bsah
```

```
mjpg_streamer -i "/usr/lib/input_uvc.so -d /dev/video0 -y -r 320x240  
-f 10" -o "/usr/lib/output_http.so -p 8090 -w /var/www/mjpg_streamer"
```

Change permission:

```
~$sudo chmod a+x mjpg.sh
```

Then add /home/ubuntu/mjpg_streamer/mjpg.sh in etc/rc.local.

The auto start of AP can be implemented by editing the script file AP.sh:

```
#!/bin/sh  
  
mypath="/sys/devices/platform/sw-ehci.2/usb3/3-1/3-1:1.0/net/wlan  
3"  
  
while [ ! -d $mypath ]  
do  
    if [ -d $mypath ] ; then  
        break  
    fi  
done  
  
service network-manager restart  
  
ifconfig wlan3 up  
  
ifconfig eth0 up  
  
killall hostapd  
  
hostapd -B /etc/hostapd.conf  
  
sleep 2  
  
ifconfig wlan3 192.168.0.1  
  
sleep 3  
  
dhcpd wlan3 -pf /var/run/dhcp-server/dhcpd.pid  
  
sleep 3  
  
echo 1 >/proc/sys/net/ipv4/ip_forward  
  
sleep 3  
  
iptables -t nat -A POSTROUTING -o eth0 -j MASQUERADE  
  
sleep 4
```

```
hostapd -B /etc/hostapd.conf  
echo setAP >/home/ubuntu/log
```

Change permission,

```
~$ sudo chmod a+x AP.sh  
~$sudo vim /etc/profile
```

In the last line add AP.sh with directory:

```
/home/ubuntu/AP.sh
```

Save and exit.

Note: Please do not change the position of the WiFi module, which may change the wlan value and the mypath in AP.sh. As a result, the pcDuino may not work properly, and a black screen will be seen when debugging.

Solution: Change the WiFi to its original position. Or find the wlan value through serial port. Find the wlan position, and edit its corresponding script file.

3) System backup:

In the first time startup, you need to follow the above procedure to make the system functions complete. To prevent a system collapse and you have to start over, you can have a system backup. With this backup, you can either start from a SD card, or use the SD card to recover if the nand system collapses. Please refer to <http://www.pcduino.com/?p=998> on how to make a SD startup disk. From SD to nand :

```
$sudo dd if=/dev/mmcblk0p2 of=/dev/nandd bs=1M && sudo sync
```

We also provide a working SD image that including everything for Rover, you can download from:

https://s3.amazonaws.com/pcduino/Images/2013-05-31/rover/rover_image.rar

IX. Control the Car with a PC

Install the client software on a PC. Run setup.exe, the following user interface shows up.



Click on Setting:



The video IP and control IP need to be changed to the ip address of the pcDuino.
Save and restart the software.

X. Control the Car with Android Device

10.1 Introduction

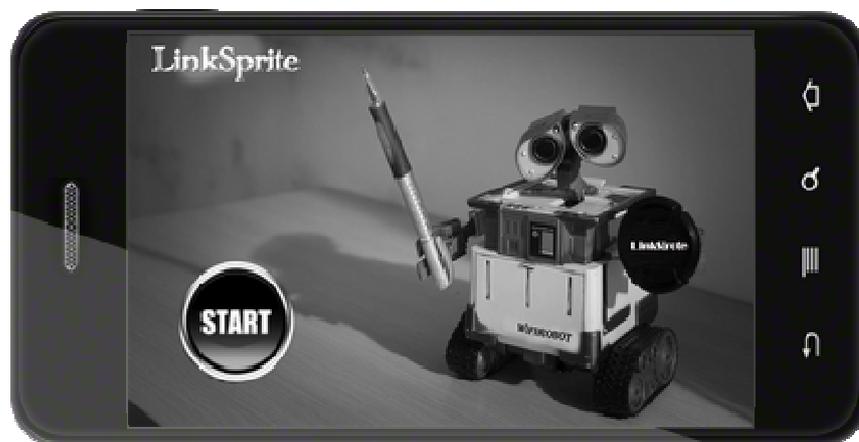
The Android client software is named WiFiRobot, for Rover powered by pcDduino. The Android app software communicates with Rover through WiFi. It can control the movement of Rover and vertical or horizontal rotation of the servo of PZT camera. It can also control the video camera, transmit and replay real time video.

There are two ways to control the movement of Rover in the Android device. One way is to use key, the other is to use gravity sensor of the phone. Use can choose by checking the corresponding selection.

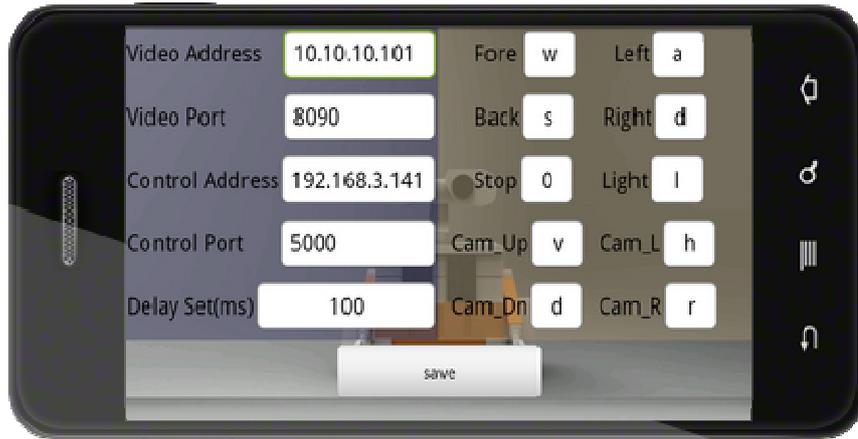
There are also two ways to control the servo. One is to use key, the other is to use scroll bar. Use can choose by checking the corresponding selection.

10.2 Interface Demo

- (1) Home interface of the Android App



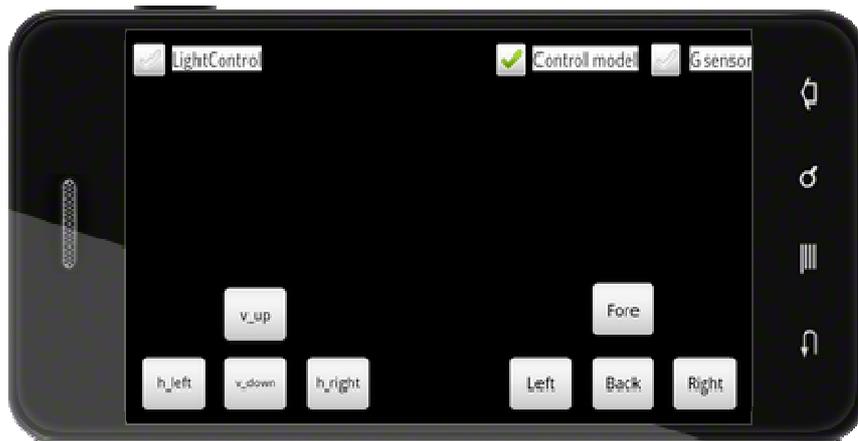
- (2) Setting page



(3) Above page



(4) Primary control interface



10.3 Introduction to Source Code

The App consists of three modules in three packages. These three modules are interface control module, picture format conversion module, and the socket communication module.

10.3.1 Interface Control Module

In the module, there are four activities: IntroductionActivity, AboutActivity, SetActivity 和 MonitorActivity.

IntroductionActivity initializes the App. Click on start to enter the home interface. The userMsgReader is used to read the pre-configured data, and initialize the App. Click on Setting and About to enter setting page, and the About page. Click on Return, a window pops up and asks the user to decide if to exit the App. This button uses onKeyDown.

AboutActivity shows the About page information. It shows the background image. A return button can be used to return to the home interface.

SetActivity sets the App parameters. User can enter this page and set his parameters. The parameters are saved in System_data. The data can be made persistent using sharedPreferences.

MonitorActivity controls the home interface. The App takes the parameters set previously, and begins to work. All work here is done using the Message mechanism. Different function module sends its message to Handler. The Handler recognizes the message and performs different operation.

10.3.2 Picture Format Conversion Module

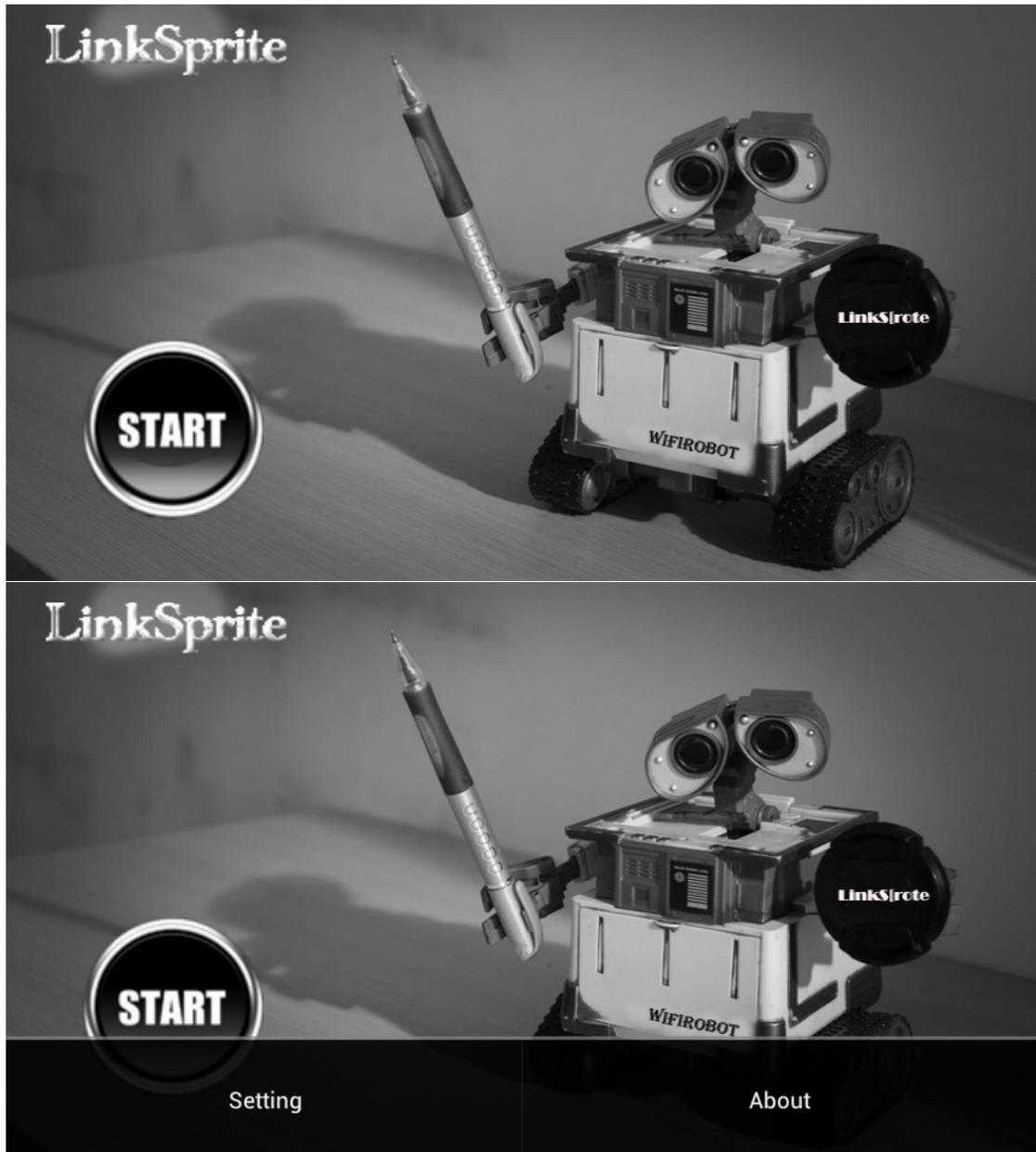
Picture format is converted in this module. Refer to the source code for details.

10.3.3 Socket Communication Module

In this module, data is sent and received in background using socket.

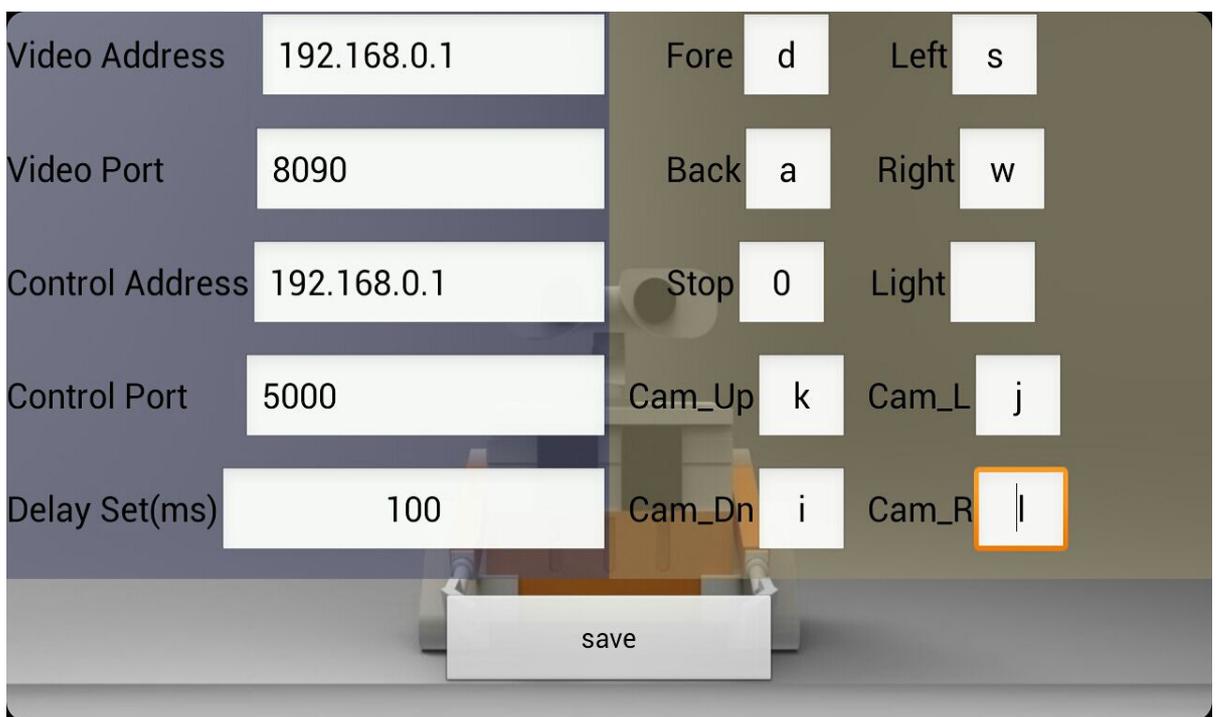
10.4 User Guide

(1) Install the software WiFiRobot in an Android phone. Run the software and click on menu, the home interface shows up:

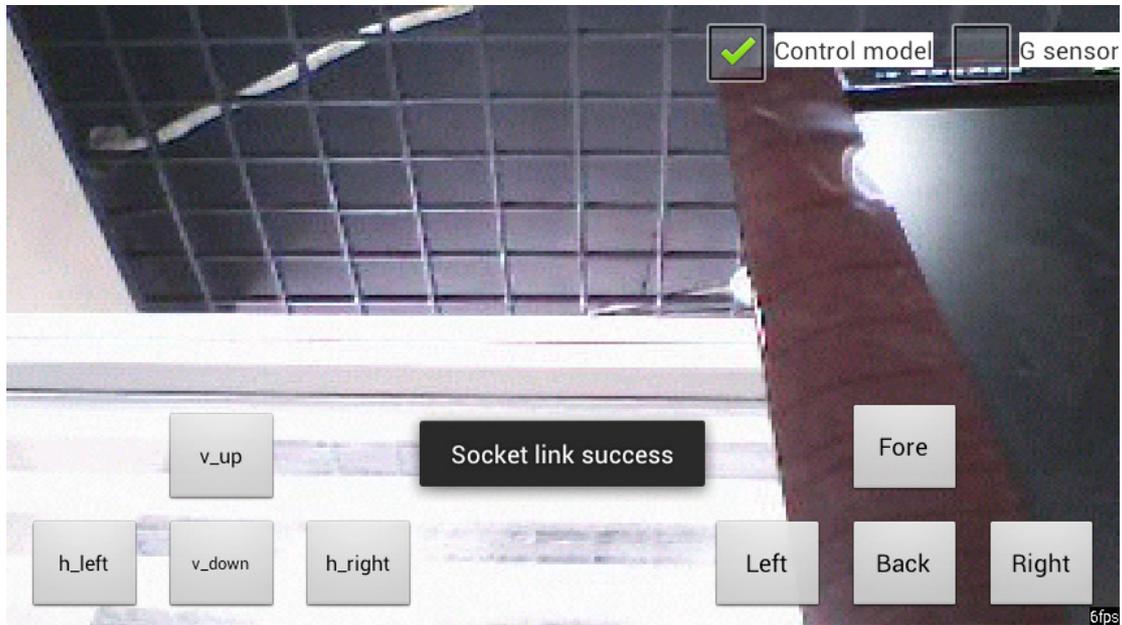


(2) Click on Setting: video ip address and control address needs to be set to the ip address of the pcduino. Set video port to 8090, Control port to 5000, Servo and camera control:

fore d
left s
back a
right w
cam_Up k
cam_L j
cam_Dn i
cam_R l



(3) Save and click on start to connect to pcdduino. When the following page shows up, the connection is successful.



XI. Download Links

1、Rover PC client:

<https://s3.amazonaws.com/linksprite/robot/Rover2/Rover-PC.rar>

2、car_test.c

3、[WiFiRobot.zip\(Android\)](#):

https://s3.amazonaws.com/linksprite/robot/Rover2/pcduino_rover/WiFiRobot.apk