Streaming Webcam Video over Beaglebone Black to Android Device

Fundamentals of Embedded Linux (Spring 2015)

By: Hassan Feroze

Overview

- Project Setup
  - Host Setup
  - Target Setup
  - Android device
- WiFi Setup
- Webcam Setup
- Video Streaming
- Camera Server
- Android Application

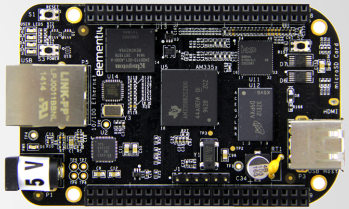
Project Setup (1/6)

Host Setup

- Oracle Virtual Box (Version 4.3.26 r98988) with 64bit Debian GNU/Linux 7
- Serial and/or USB connection to Target
- Connected to Target through Putty

Project Setup (2/6)

Target Setup



- Beaglebone Black RevC
  - Linux arm 3.14.1+ (armv71 GNU/Linux)
  - Debian GNU/Linux 8 arm
- Sabrent 4 port USB hub
- Edimax EW-7811 Wifi module
- Logitech C310 720p Webcam
- 5V 4A Mean Well AC/DC Power Supply
- Serial and/or USB connection to Host

Project Setup (3/6)

Android Device

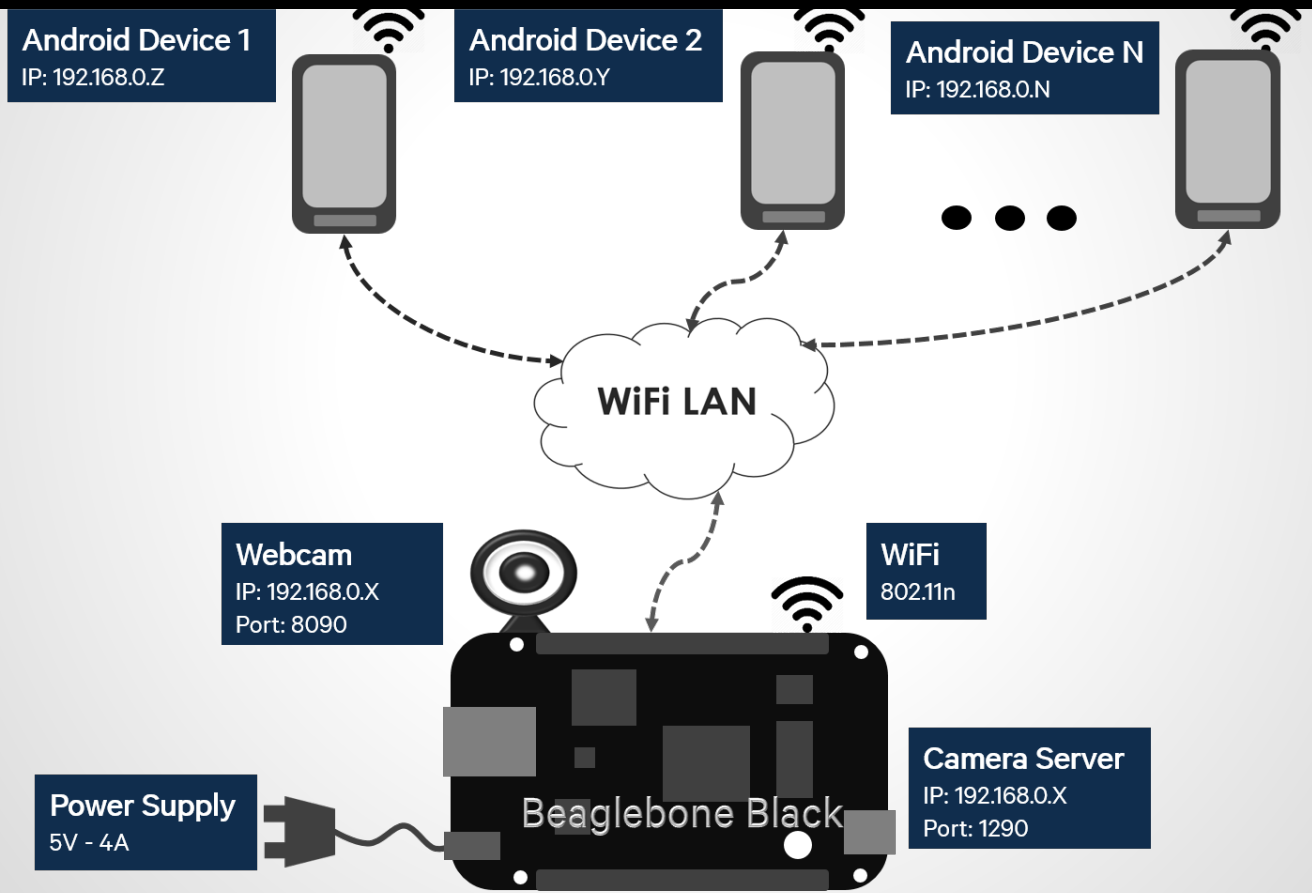
- Google Nexus 5
- OS: Android 5.1 (Lollipop)

Project Setup (54/6)

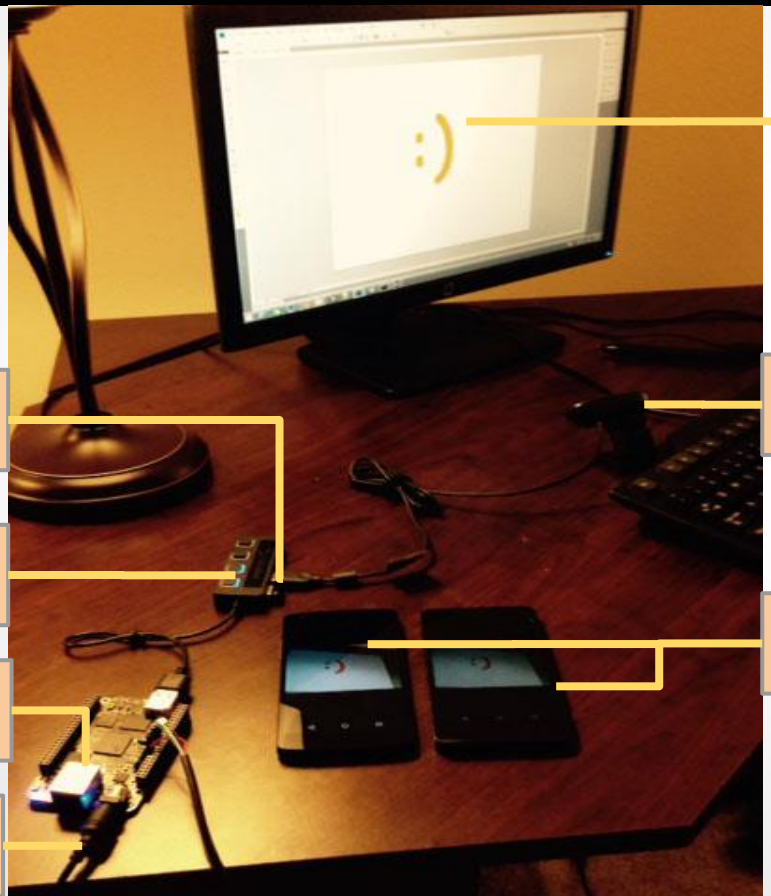
Setup Summary

- Beaglebone Black is powered using 5V-4A power supply
- ***Webcam*** and ***WiFi*** are connected to BBB through USB Hub
- BBB is equipped with a ***Camera Server***
- BBB's Camera Server interacts with Android Devices over IP network within ***WLAN***
- Android Devices send requests to BBB's Camera Server to access Webcam video stream
- Setup diagrams are presented next

Project Setup (5/6)



Project Setup (6/6)



Smiley :)

Logitech C310 Webcam

Edimax EW-7811 Wifi

USB Hub

Android Devices

Beaglebone Black RevC

5V - 4A Power Supply

WiFi Setup (1/2)

Steps to enable WiFi on BBB are:

1. Create file wpa_supplicant.conf

   nano /etc/wpa_supplicant/wpa_supplicant.conf

2. Modify wpa_supplicant.conf

COM7 - PuTTY

```
  GNU nano 2.2.6    File: /etc/wpa_supplicant/wpa_supplicant.conf

ctrl_interface=DIR=/var/run/wpa_supplicant GROUP=netdev
update_config=1
network={
 ssid="        "
 scan_ssid=1
 psk="              "
 proto=RSN
 key_mgmt=WPA-PSK
 pairwise=CCMP
 auth_alg=OPEN
}
```

WiFi SSID

WiFi Password

WiFi Setup (2/2)

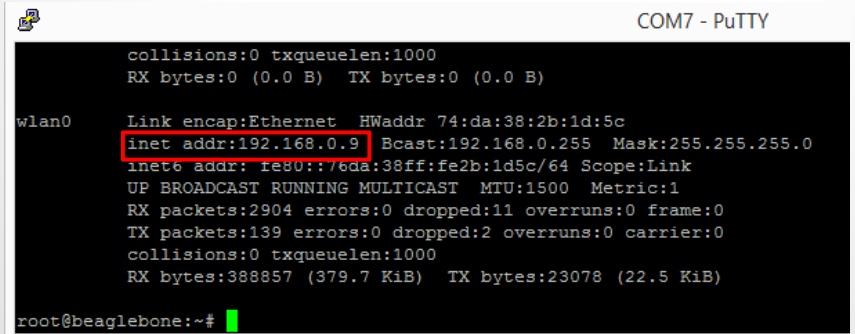
3. Execute "*ifconfig -a*". This will give the wireless LAN interface name e.g. ***wlan0***
4. Modify /etc/network/interfaces and add following lines

   allow-hotplug ***wlan0***

   iface ***wlan0*** *inet manual*

   wpa-roam /etc/wpa_supplicant/wpa_supplicant.conf

   iface default inet dhcp
5. run *ifup wlan0*
6. Turn off BBB, connect USB hub with WiFi module inserted and power on BBB
7. *ifconfig*



Webcam Setup

- Webcam drivers are plug and play on BBB
- Turn off BBB, connect webcam to USB hub and connect the USB hub to BBB. Then power on BBB
- Webcam video source shows up as /dev/video0
- ls /dev

COM7 - PuTTY

```
fd              mmcblk0boot1        shm      tty31  tty60     vcsa4
full            mmcblk0p1           snd      tty32  tty61     vcsa5
fuse            mmcblk0p2           sndstat  tty33  tty62     vcsa6
hwrng           mqueue              stderr   tty34  tty63     vcsa7
i2c-0           net                 stdin    tty35  tty7      video0
i2c-1           network_latency     stdout   tty36  tty8      watchdog
initctl         network_throughput  tty      tty37  tty9      watchdog0
input           null                tty0     tty38  ttyGS0    xconsole
kmem            ppp                 tty1     tty39  ttyO0     zero
kmsg            psaux               tty10    tty4   ttyS0
log             ptmx                tty11    tty40  ttyS1
log_events      ptp0                tty12    tty41  ttyS2
logibone_mem    pts                 tty13    tty42  ttyS3
log_main        ram0                tty14    tty43  ubi_ctrl
root@beaglebone:~#
```

Video Streaming (1/5)

- Two widely available options to stream webcam on Linux
  - ffmpeg
  - MJPG-streamer
- ffmpeg
  - More common and widely used of the two
  - Supports wide range of video formats
  - Absolute choice for H.264 format video
- MJPG-streamer
  - Supports mjpeg video streaming

Video Streaming (2/5)

- Install required components

  sudo apt-get install v4l-utils libv4l-dev ffmpeg libv4l libv4l-dev v4l-utils qv4l2 v4l2ucp

- Logitech's Webcam C310
  - supports MJPEG and YUYV
  - It does not support H.264
  - Supported formats *v4l2-ctl --list-formats*

COM7 - PuTTY

```
root@beaglebone:~# v4l2-ctl --list-formats
ioctl: VIDIOC_ENUM_FMT
        Index       : 0
        Type        : Video Capture
        Pixel Format: 'YUYV'
        Name        : YUV 4:2:2 (YUYV)

        Index       : 1
        Type        : Video Capture
        Pixel Format: 'MJPG' (compressed)
        Name        : MJPEG

root@beaglebone:~#
```

Video Streaming (3/5)

- Both formats were tried
  - ffmpeg kept dropping packets causing choppy video stream. Video got hung after playback of a minute or so. This seems to be ffmpeg's compatibility issue with MJPEG content
  - MJPG-streamer turned out to be very stable for Logitech C310

Video Streaming (4/5)

MJPG-streamer

- https://code.google.com/p/mjpg-streamer/
- Designed for embedded devices.
- Streams MJPEG video content over IP Network
- Licensed under GNU GPL v3

Video Streaming (4/5)

MJPG-streamer setup

1. *mkdir MJPG-Streamer*
2. *cd MJPG-Streamer*
3. *git clone https://github.com/jacksonliam/mjpg-streamer*
4. *cd mjpg-streamer/mjpg-streamer-experimental*
5. *make*

Video Streaming (5/5)

MJPG-streamer startup

- MJPG-streamer can be started using following command

```
# ./mjpg_streamer -i "./input_uvc.so -d /dev/video0 -n -f 15 -r 640x480" -o "./output_http.so -p 8090 -n -w ./www"
```

- MJPG-streamer has plenty of flags, the ones in this command are
  - -i --- input source (/dev/video0)
  - -f --- frame rate (15)
  - -r --- resolution (640x480)
  - -o --- output is going to the web (-w) on port 8090
  - -p --- output port (8090)
- input_uvc.so is the input module that captures JPG frames, and output_http.so is the ouput module that pushes JPG frames to the specified port. Camera stream is viewed by accessing this port.

Camera Server (1/7)

- TCP/IPV4 server
- Listens on port 1290
- Started at bootup
- Designed to listen and respond to incoming client requests in the following manner

Client Request	**Response**
Start	Turn on webcam streaming and respond with corresponding code
Shutdown	Turn off webcam streaming and respond with corresponding code
Check	Send ON/OFF status code of the webcam

Camera Server (2/7)

Server source

- Source code for camera server (*cameraServer.c*) is provided with project package
- Developed using Linux socket package
- Some details of the code's functions are given next

Camera Server (3/7)

Function	Details
createServerSocket(int port)	creates server socket on port “*port*”

Method(s) defined in createServerSocket()	**Details**
sockfd = socket(AF_INET, SOCK_STREAM, 0);	IPV4 socket created using AF_INET flag
serv_addr.sin_family = AF_INET;	IPV4 family
serv_addr.sin_addr.s_addr = inet_addr("0.0.0.0");	IP address is set to 0.0.0.0
serv_addr.sin_port = htons(1290);	port is set to 1290
*bind(sockfd, (struct sockaddr *) &serv_addr, sizeof(serv_addr))*	binds server to IP and port

Camera Server (4/7)

Function	Details
waitForClient()	wait for incoming client connection and client request

Method(s) defined in createServerSocket()	Details
listen(sockfd);	passive socket listening at port 1290
*newsock = accept(sockfd, (struct sockaddr *) &cli_addr, &clilen);*	accepts client connection at sockfd
read(newsockfd,command,255);	reads 255 bytes of client request

Camera Server (5/7)

Function	Details
checkCameraState()	Checks if webcam is currently running and responds with the correct state

Method(s) defined in checkCameraState()	**Details**
readOutput()	reads output of *netstat -tulnap | grep 8090, checks if the process MJPG-streamer is running and returns the status code* *status code: **7411** [webcam is ON]* *status code: **949** [webcam is OFF]*

Camera Server (6/7)

1. Compile server source file

2. Launch server on port 1290

3. server waits for incoming client request

4. Client sends “Start” req

5. Server complies

6. Server launches camera stream, checks status & sends status code to client

7. Client sends “check” req

8. Server checks camera status and sends status to the client

9. Client sends “shutdown” req

10. Server turns off camera stream, checks status & sends status code to client

11. server waits for incoming client request

```
root@beaglebone:~/mjpg-str/mjpg-streamer# gcc -pthread -o cameraServer cameraServer.c
root@beaglebone:~/mjpg-str/mjpg-streamer# ./cameraServer 1290
Connection parameters..AF_INET:2..SOCK_STREAM:1..portno:1290
state: 949
Waiting for client connection
 Start
 the command is Start

Executing Start

sending camera state
tcp        0      0 0.0.0.0:8090            0.0.0.0:*               LISTEN      3961/mjpg_streamer
tcp6       0      0 :::8090                 :::*                    LISTEN      3961/mjpg_streamer
state 7411

Waiting for client connection
 check
 the command is check

Executing check

Checking Camera State
tcp        0      0 0.0.0.0:8090            0.0.0.0:*               LISTEN      3961/mjpg_streamer
tcp        0 111496 192.168.0.9:8090        192.168.0.3:50751       ESTABLISHED 3961/mjpg_streamer
tcp6       0      0 :::8090                 :::*                    LISTEN      3961/mjpg_streamer
state 7411

Waiting for client connection
 Shutdown
 the command is Shutdown

Executing Shutdown

8090/tcp:             3961
tcp        0  74977 192.168.0.9:8090        192.168.0.3:50751       FIN_WAIT1   -
state 949

Waiting for client connection
```

Camera Server (7/7)

Launching camera server on bootup

- It is very inconvenient to launch server manually every time we reboot BBB
- So, functionality to launch server automatically at bootup was added
  1. sudo crontab -e
  2. add this line

@reboot /root/MJPG-Streamer/mjpg-streamer/mjpg-streamer-experimental/cameraServer 1290 &

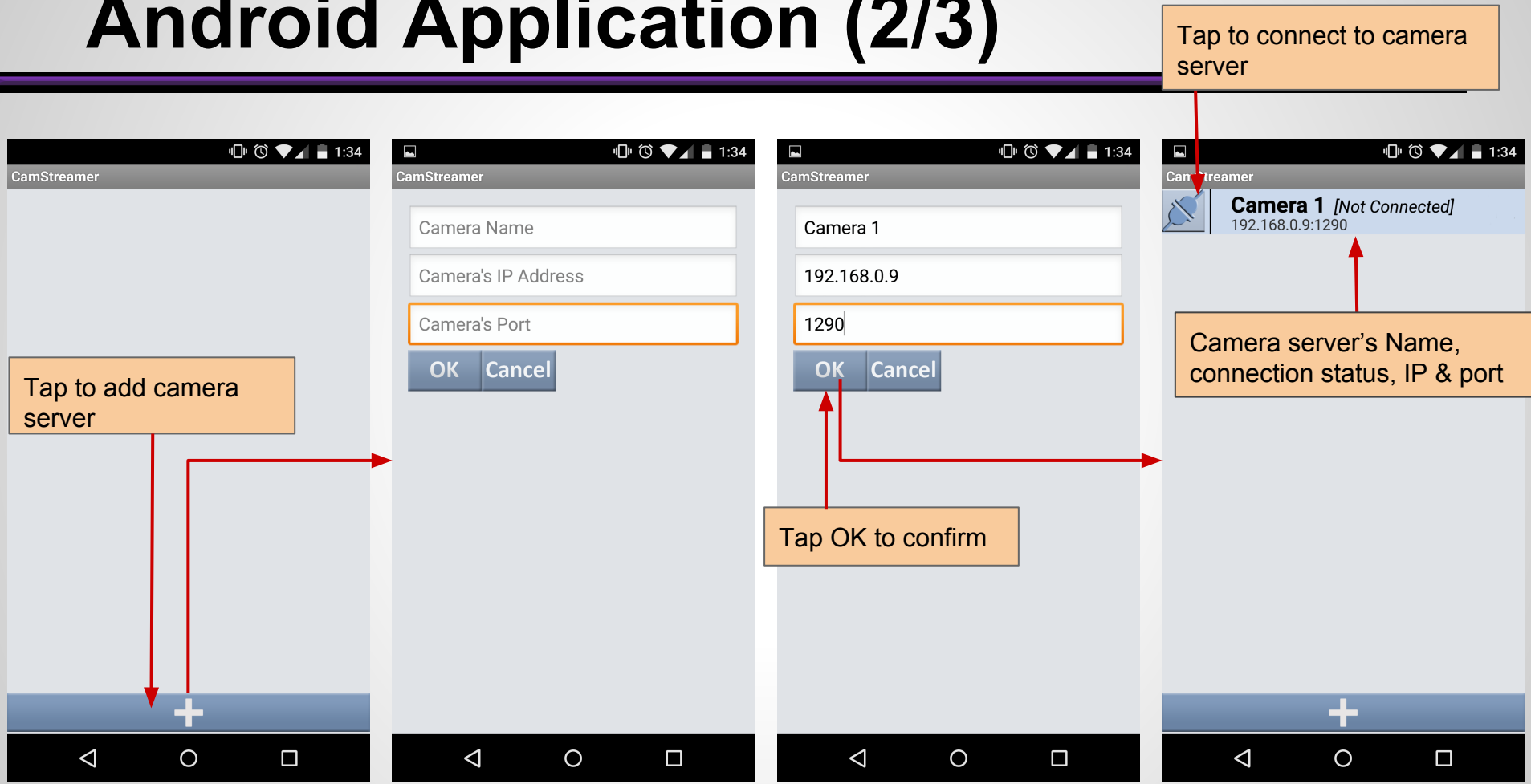
  3. save the file and reboot BBB
  4. check if server is up using "*netstat -tulnap | grep 1290*"

tcp 0 0 0.0.0.0:1290 0.0.0.0: LISTEN 890/cameraServer*

Android Application (1/3)

- Android app developed using Android SDK
- Features
  - UI to add BBB's Webcam's Name, IP and Port to SQL Database
  - Sends Webcam Start/Shutdown/Check requests over TCP/IP to the BBB camera server and listens to response code
  - Displays list of added Webcams and their corresponding Connected/Disconnected status, and
  - Shows Streaming Video

Android Application (2/3)

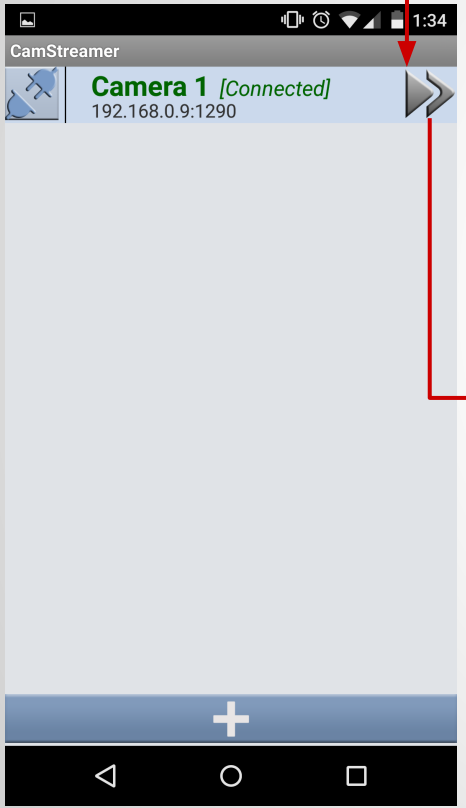


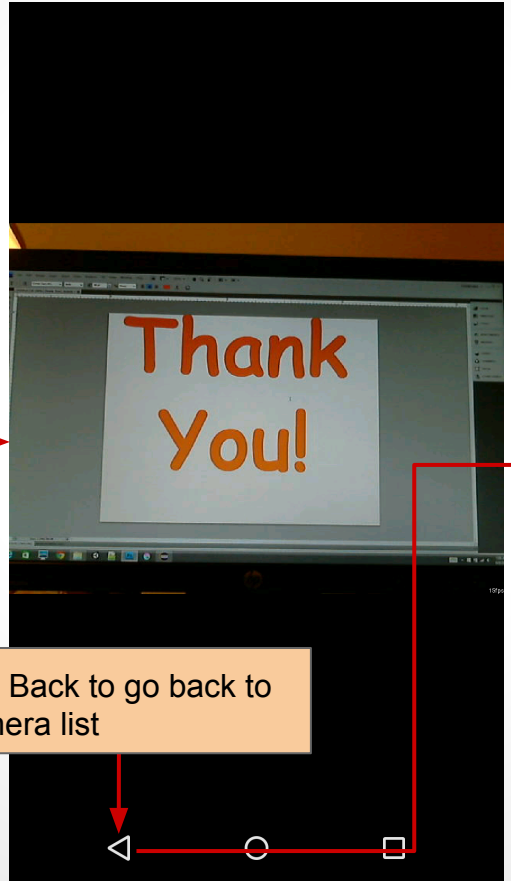
Android Application (3/3)

Tap to view Webcam stream

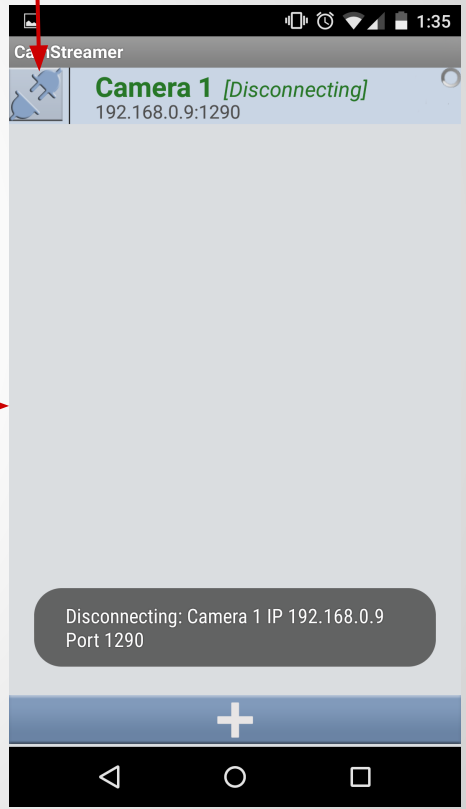
Webcam Stream view

Tap to disconnect from camera server





Tap Back to go back to camera list



Video Demo

List of Issues (1/4)

- Wifi Issues
  - HDMI module on BBB interferes with Wifi signal. To resolve this Wifi module is connected via USB hub, away from BBB's HDMI module
  - 5V Power Supply is essential to fulfill Wifi signal power requirements
- BBB won't detect WiFi or webcam if either is connected while BBB is already running. To properly have WiFi and webcam detected, they must be connected once BBB has been turned off

List of Issues (2/4)

- “/n” must be added in server response otherwise Android socket waits for response forever. /n represents the end of line
- Must use “pkill -INT mjpg_streamer” to cleanly kill MJPG-streamer process, otherwise the port might not be cleared for the next run
- Server code required -pthread flag to compile

List of Issues (3/4)

- ffmpeg along with ffserver is another way to push webcam video to IP port but the process keeps dropping packets and halts after few minutes. Several different flags and configurations were tried to optimize the process but it seems like a sort of driver/wrapper needs to be written on top of ffmpeg to work properly for MJPEG video stream.

List of Issues (4/4)

- To permanently set environment variables, changes must be made in ~./bashrc
  - cd
  - nano ~./bashrc
  - export PATH=$PATH:${new_path}
- popen() must be used to read output of a system command.

References

- http://elinux.org/BBBWiFiConfigs
- http://ffmpeg.org/ffmpeg-devices.html#video4linux2_002c-v4l2
- http://derekmolloy.ie/beaglebone-images-video-and-opencv/
- http://wolfpaulus.com/jounal/embedded/raspberrypi_webcam/
- https://code.google.com/p/mjpg-streamer/
- http://www.raspberrypi-spy.co.uk/2013/07/running-a-python-script-at-boot-using-cron/

References

- http://wiki.beyondlogic.org/index.php/BeagleBoneBlack_Building_Kernel
- http://elinux.org/Building_BBB_Kernel
- http://www.binarytides.com/server-client-example-c-sockets-linux/
- https://www.gnu.org/software/libc/manual/html_node/Signal-Handling.html
- http://www.microhowto.info/howto/ignore_sigpipe_without_affecting_other_threads_in_a_process.html