Capturing and Streaming Webcam Video with the BeagleBone Green

Team: Third Eye

Baihui Zhang, Chih-Wen Chi, Chenxia Dun, Weichao Lin

2017 Fall CMPT 433

This guide was mostly based on Dr. Derek Molloy's "UDP Unicast and Multicast Streaming Video using the Beaglebone Black" Guide on http://derekmolloy.ie/udp-video-streaming-beaglebone-black/. However, it's not a trivial task to adapt Derek Molloy's method to make it work with BBG and Logitech HD C270 webcam. This guide also referred several former CMPT 433 video guides, but the former guides all have some drawbacks, which prevented them from streaming 720p video smoothly.

Also, the BeagleBone Green we used for this task came with some special features such as the FFmpeg can not be installed with little effort because of the linux system on the BBG. This guide was specifically written for BBG with Debian 8 installed and Logitech HD C270 webcam. This guide also pointed out things worth attention and possible ways to figure out how to adapt Dr. Derek Molloy's method to work with other webcams in trouble shooting parts.

1. Connect webcam to BeagleBone

We will use Logitech HD C270 webcam in this guide. Other USB webcams should work with adaption of code.

First, connect the webcam to BeagleBone's USB port. Use the following commands to check whether the webcam is connected successfully. Example output is also included.

```
#lsusb
Bus 001 Device 002: ID 046d:0825 Logitech, Inc. Webcam C270
…
```

or

```
#ls /dev/ | grep video
video0
```

Troubleshooting:

If you cannot found your webcam using lsusb or cannot found video0 in /dev,

- Double check whether you connected the webcam to BeagleBone's USB port
- Double check the USB port is working normally
- Try to unplug and re-plug in the webcam
- Unplug other devices to make sure there is enough power for the webcam

2. Capture video using v4l2

We will need video4linux command line utilities(v4l-utils) and video4linux support libraries(libv4l-dev, development files) to set up webcam and capture video.

1. BBG comes with v4l-utils, so we don't need extra steps to install it. To check, use the following command. You should be able to see v4l2-ctl directory.

```
# whereis v4l2-ctl
v4l2-ctl: /usr/bin/v4l2-ctl
```

After connecting camera to BBG, you should be able to use v4l2-ctl commands to check whether v4l2 supports our webcam and how it supports the webcam. Some useful commands are list below. You need to read the following output carefully because it determines how you should adjust your parameters for video capturing and streaming in the future.

```
# v4l2-ctl –list-device
UVC Camera (046d:0825) (usb-musb-hdrc.1.auto-1):
        /dev/video0
# v4l2-ctl --list-formats
ioctl: VIDIOC_ENUM_FMT
        Index       : 0
        Type        : Video Capture
        Pixel Format: 'YUYV'
        Name        : YUYV 4:2:2

        Index       : 1
        Type        : Video Capture
        Pixel Format: 'MJPG' (compressed)
        Name        : Motion-JPEG
```

We are able to set a variety of webcam parameters using v4l2-ctl --set… command.
2. We need to install libv4l-dev

```
# apt-get install libv4l-dev
```

3. Clone capture.c from derekmolloy's boneCV directory at https://github.com/derekmolloy/boneCV. We only need capture.c. Other files can be ignored.

4. Revise the following lines in capture.c from boneCV according to your webcam model.

```
if (force_format) {
                if (force_format==2){
                        fmt.fmt.pix.width       = 1920;
                        fmt.fmt.pix.height      = 1080;
                        fmt.fmt.pix.pixelformat = V4L2_PIX_FMT_H264;
                        fmt.fmt.pix.field       = V4L2_FIELD_INTERLACED;
                }
                else if(force_format==1){
                        fmt.fmt.pix.width       = 640;
                        fmt.fmt.pix.height      = 480;
                        fmt.fmt.pix.pixelformat = V4L2_PIX_FMT_YUYV;
                        fmt.fmt.pix.field       = V4L2_FIELD_INTERLACED;
                }
…
```

We can see from output of v4l2-ctl --list-formats that Logitech C270 supports two kinds of pixel format: YUYV and MJPEG.

Since we plan to stream video in the future, it's better to use compressed pixel format MJPEG. The following are my revisions.

```
if (force_format) {
        if (force_format==2){
                fmt.fmt.pix.width       = 1280;
                fmt.fmt.pix.height      = 720;
                fmt.fmt.pix.pixelformat = V4L2_PIX_FMT_MJPEG;
                fmt.fmt.pix.field       = V4L2_FIELD_NONE;
        }
…
```

5. Copy revised capture.c to target and compile it.

```
# gcc capture.c -lv4l2 -o capture
```

6. Alternatively, you could cross compile capture.c on host.
First, make a directory on host and copy from target /usr/lib/arm-linux-gnueabihf/libv4l2.so to that directory. For example,

```
$ mkdir ~/cmpt433/public/v4l2_lib
$ chmod 777 ~/cmpt433/public/v4l2_lib
```

When you compile, put LFLAGS = -L$(HOME)/cmpt433/public/v4l2_lib into your makefile and you should be able to cross compile capture.c on host with –lv4l2 flag.
7. With modified capture.c, we should be able to capture 720p video. The following command captures 10s video and stores it in output.raw

```
# ./capture -F -c 300 -o > output.raw
```

Argument explanation:

-F	force format to mjpeg (after revision)
-c | --count	Number of frames to grab [100] - use 0 for infinite
-o | --output	Outputs stream to stdout
-h | --help	

8. Troubleshooting:

- If you are using Beaglebone Black or your system does not come with v4l-utils, install it:

  ```
  # apt-get install v4l-utils
  ```

- If your webcam supports other pixel formats other than YUYV or MJPEG, you need to check v4l2 API to see how to adapted capture.c accordingly. boneCV also provides solution for H264 pixel format.
- If the above settings do not work with your camera, carefully check your webcam's parameters and v4l2 API and adjust your code.
- If you get errors while cross-compiling capture.c on host, check whether your have copied libv4l2.so to host and set up the directory correctly.

- If you get errors when capturing video, it may because you don't have write permission to the folder.
- If you are able to capture video, but the capturing speed is pretty slow, your pixel format setting in capture.c is most likely wrong. Read v4l2 API carefully and revise your code.

3. Transcoding raw video file using FFmpeg

1. We need to install FFmpeg to transcode the raw video. It can be done either on host or target. Because we are running debian 8 on BBG, we need extra steps to install FFmpeg on target.
First, we need to add backports to our sources.list

```
# cd /etc/apt/sources.list.d/
```

Then, modify sources.list and add the following line to the end of the file

```
deb http://ftp.debian.org/debian jessie-backports main
```

After that, FFmpeg can be installed using apt-get

```
# apt-get update
# apt-get install ffmpeg
```

2. To convert the raw file to mp4

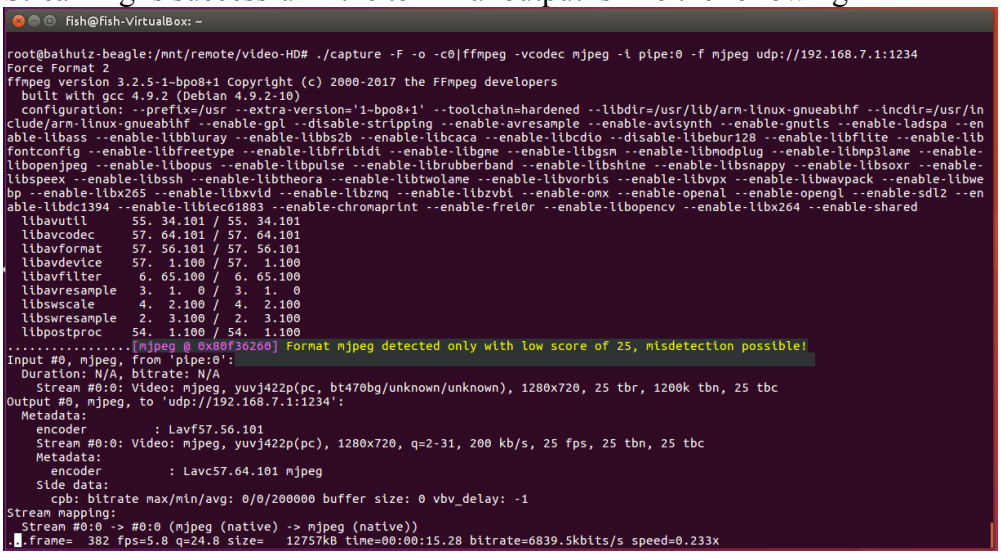
```
# ffmpeg -vcodec mjpeg -i output.raw -f mjpeg output.mp4
```

4. Stream video using FFmepg

1. Combing v4l and FFmpeg, we are able to stream video conveniently

```
# ./capture -F -o -c0|ffmpeg -vcodec mjpeg -i pipe:0 -f mjpeg udp://192.168.7.1:1234
```

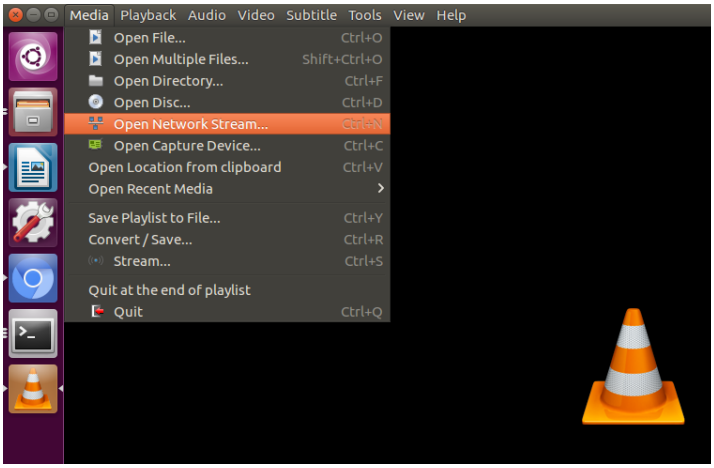
Streaming is successful if the terminal output is like the following

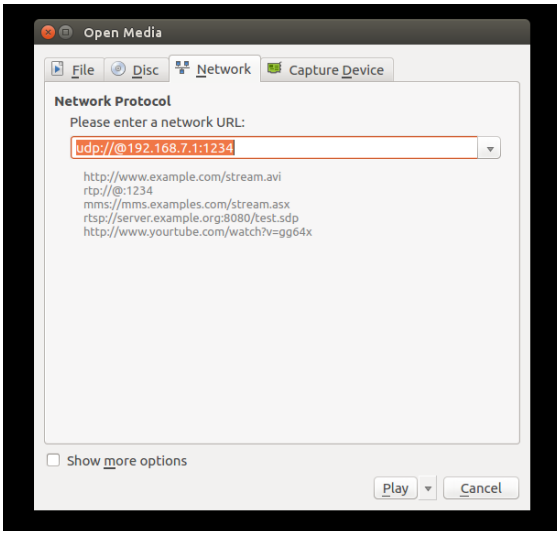


2. There are many options to accept video stream and play streamed video. One easy option on Linux is using VLC media player. To install VLC media player on host

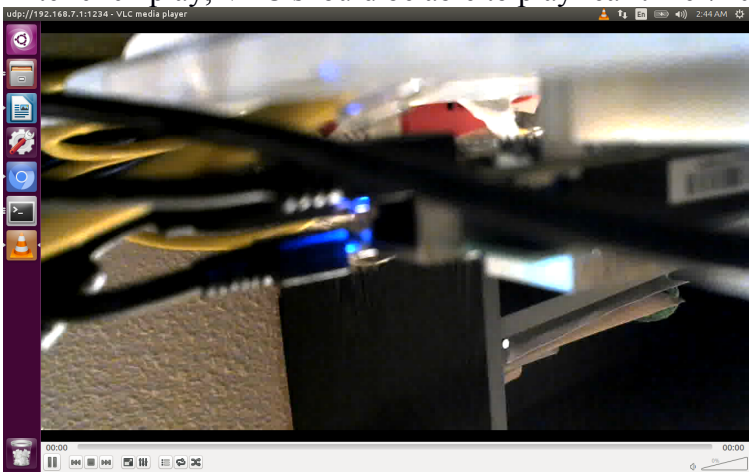
```
$ sudo apt update
$ sudo apt install vlc
```

3. Play video using VLC.
On host, open VLC, Media->open Network Stream-> Network-> Network Protocol->Please enter a network URL:
udp://@:1234





After click play, VLC should be able to play real-time 720p video as shown below.



4. Troubleshooting

- You may not need to add backports to sources.list in order to install FFmpeg. Just do that if you'd like to install FFmpeg on Debian 8 "Jessie".
- If you are not able to transcode or stream video using FFmpeg, please check FFmpeg API carefully and set up your argument accordingly. Also refer to basic multimedia transcoding knowledge if necessary.
- If you are able to transcode video but the video is of unreasonably low quality or the speed of streaming is very slow, something wrong with your capture.c pixel format or arguments for

FFmpeg are not appropriate. Check v4l2 API and FFmpeg API and check your webcam parameters using v4l2-ctl. Revise your code accordingly.

- If you started streaming successfully, but not able to play the video in VLC, double check the UDP address.

5. Reference

Method adapted from boneCV:
1. boneCV github repository: https://github.com/derekmolloy/boneCV
2. derekmolloy's blog article: http://derekmolloy.ie/udp-video-streaming-beaglebone-black/

Useful previous cmpt433 how-to guides:
1. http://www.cs.sfu.ca/CourseCentral/433/bfraser/other/2015-student-howtos/RecordingWebcamVideos.pdf
2. http://www.cs.sfu.ca/CourseCentral/433/bfraser/other/2014-student-howtos/WebCam.pdf
3. http://www.cs.sfu.ca/CourseCentral/433/bfraser/other/2016-student-howtos/WebCamVideoOpenCV.pdf