CMPT433
How to stream OpenCV proccessed images over the network

Date: November 25, 2019

Group: Patrol Rover

Ali Maladwala

Branko Bajcetic

Amandeep Rehal

Rehmanali Jiwani

Contents

1. Hardware and software setup 1

1.1 Troubleshooting 1

2. Building the Sample Code 2

2.1 Troubleshooting 3

3. Run the program 4

3.1 Troubleshooting 6

1. Hardware and software setup

a. Plug in your USB camera to the target device (BeagleBone Green).

b. Use the Linux command **lsusb** on the target to list all connected devices and make sure your camera is listed.

c. Run the following commands on the **target** to install the needed software(takes around 700mb of space):

   i. sudo apt-get install libv4l-dev

      sudo apt-get install libopencv-dev

      sudo apt-get install ffmpeg

      sudo apt-get install libav-tools

1.1 Troubleshooting

- If installing the software dependencies fails, run **sudo apt-get update** and try again.
- Make sure you have enough space before installing the dependencies; you may have to flash the BBG image if you run out of space.

2. Building the Sample Code

Copy the following C++ Sample Code (adapted from https://github.com/derekmolloy/boneCV) to a file named camera.cpp:

```
#include<iostream>
#include<opencv2/opencv.hpp>
#include<opencv2/imgproc.hpp>
#include<time.h>
#include<unistd.h>
using namespace std;
using namespace cv;
int main(){
    //setup the camera settings (640x480 image)
    VideoCapture capture(0);
    capture.set(CV_CAP_PROP_FRAME_WIDTH,640);
    capture.set(CV_CAP_PROP_FRAME_HEIGHT,480);
    if(!capture.isOpened()){
         cout << "Failed to connect to the camera." << endl;
    }
    Mat frame;
    while(1){
         //capture and process images from the webcam
         capture >> frame;
         if(frame.empty()){
             cout << "Failed to capture an image" << endl;
             return;
         }
         std::vector<uchar> buff;
         //encode to jpg
         cv::imencode(".jpg", frame, buff);
         //write jpg to stdout so it can be piped
         fwrite(buff.data(),buff.size(),1,stdout);
         fflush(stdout);
         }
```

Explanation:

```
std::vector<uchar> buff;
cv::imencode(".jpg", frame, buff);
```

The imencode function takes the image grabbed from the camera (stored in the frame variable) and encodes it into a jpeg.

```
fwrite(buff.data(),buff.size(),1,stdout);
fflush(stdout);
```

The jpg data is written to stdout so it can be piped to the streaming application

Build the sample code using the following command when connected to the **target** (BBG):

```
g++ -O2 `pkg-config --cflags --libs opencv` -lrt camera.cpp –o camera –lpthread
```

2.1 Troubleshooting

- This is C++ code; make sure you use g++ and not gcc to compile.
- Make sure you are running the makefile through the target BBG, not the host computer.

3. Run the program

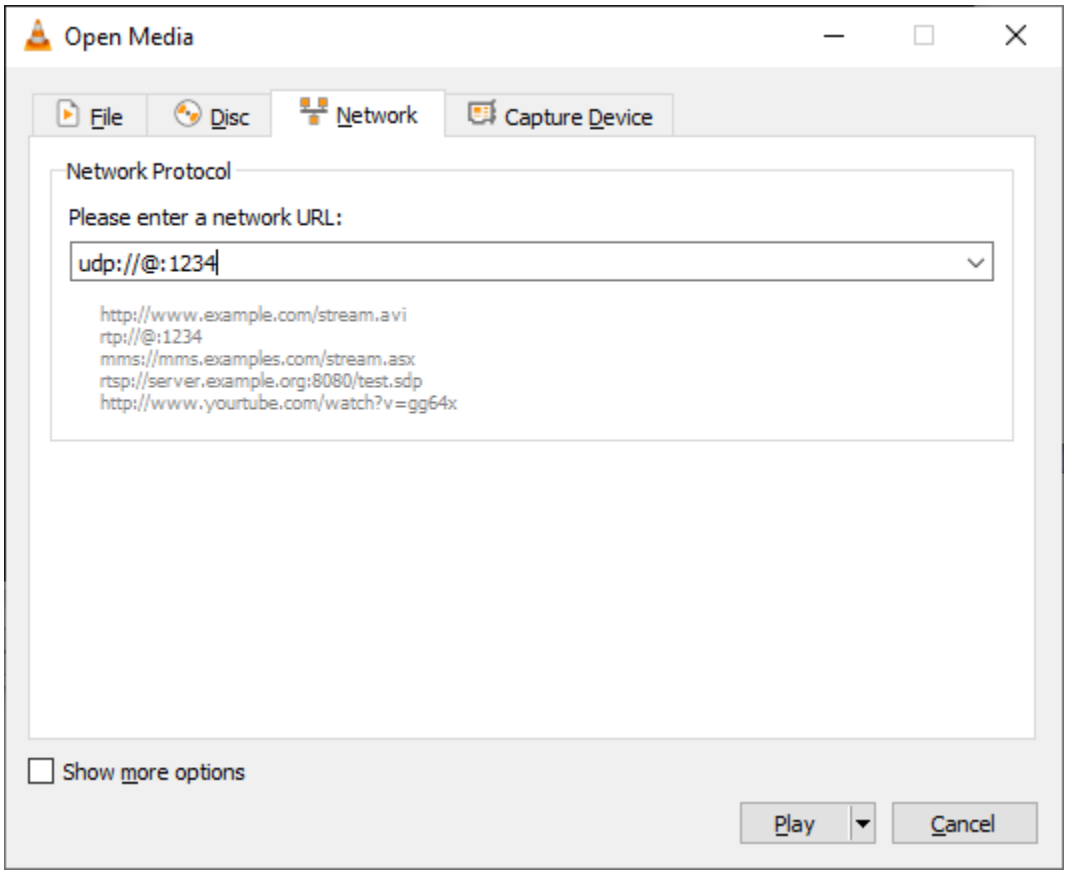
This section assumes that the BBG is on the same network as the host and that the BBG IP address is 192.168.7.2 and that the host ip is 192.168.7.1, adjust the following commands with the other ip addresses as needed.

1. Run the following command to run the test program: **./camera | avconv -vcodec mjpeg -r 5 -i pipe:0 -f mpegts udp://192.168.7.1:1234**

Explanation: The jpeg images are piped (when they are written to stdout) to avconv which runs with the following configuration to convert and stream the video:

-vcodec mjpeg	(Set the video codec to motion jpeg)
-r 5	(Set the output video framerate to 5fps, **OPTIONAL**)
pipe:0	(Read from stdin)
mpegts	(Output format: MPEG transport stream)
udp://192.168.7.1:1234	(stream over UDP to 192.168.7.1 on port 1234)

2. Open VLC on your host and press **Media -> Open Network Stream and input: udp://@:1234.**



3. Press **Play** to view the video

3.1 Troubleshooting

- If your frame rate is too low you may want to run the camera and OpenCV processing on a separate Beaglebone to improve performance.
- If you are not getting any video, run camera on its own and make sure that images are being printed out to stdout/the terminal.
- If you want to tune avconv for your use case, take a look at these documents: https://libav.org/avconv.html
- If you are not getting any video VLC, double check the hosts ip address.
- Try running the program over Ethernet via USB for debugging