MULTIPLE LIVE WEBCAM STREAMS USING RASPBERRY PI

dev.io

Computer Engineering Department, KNUST.

March, 8 2020

Abstract: Multiple webcams were streamed live via the use of Raspberry Pi, a credit card sized computer. To begin with, we configured the Pi to stream one camera and later upgraded to stream four cameras via a local network.

INTRODUCTION

Raspberry Pi, a credit card sized computer was originally developed to change the way children were using computers by Upton, Mullins, Lang and Mycroft, however, the Raspberry Pi is now widely used by all age groups and abilities for numerous projects. The Raspberry Pi uses Linux distributions and functions as a normal PC or Mac which can be used for numerous electronics projects, gaming or word-processing. The latter relies on BASH and Python scripts to carry out tasks. Projects carried out with the Pi include creating time-lapse videos, parent detectors and jukeboxes. Here, the Raspberry Pi was used to create time-lapse videos and live streams of multiple webcams.

SETUP

There are several models of the Pi: A, B and B+, where B+ is the most recent model. Here, the B model was used. As with any computer, the Raspberry Pi requires an operating system (OS). We used the latest version of Raspbian Buster Lite as the operating system for the Pi. The Lite version does not include a graphical user interface. We are choosing this as it would just waste system resources and it is not needed in our case. However, if you think you will want the GUI for other purposes with the Raspberry Pi, feel free to install the full version of Buster instead. It won't change the instructions below.

Download the following onto your computer

- o Raspbian Buster Lite - https://www.raspberrypi.org/downloads/raspbian/
- o Etcher - https://www.balena.io/etcher/

• Install Etcher (its Mac, Windows and Linux compatible)
• Connect an SD card reader with the SD card inside to your computer
• Open Etcher and select from your hard-drive folder the Raspbian .zip file you downloaded before
• Select the SD card
• Review your selections and click 'Flash!' to begin writing data to the SD card
• Once the process is completed, close Etcher, unplug the SD card, and plug it back in again
• A small partition of the SD card should be accessible on your computer, this is the boot partition

- Place a file in this location called ‘ssh’ with no file extension
- To create this file on Mac
  - Open terminal
  - cd /Users
  - ls
  - Note the exact (case sensitive) name of your accounts users folder
  - touch /[username]/Desktop/ssh
  - Substitute [username] for the name of your Users folder you noted
  - A file blank file should appear on your desktop call ‘ssh’
- To create this file on Windows
  - Right click on your desktop
  - Go to ‘New’ and select ‘Text Document’
  - Name the file ‘ssh’ and delete the the ‘.txt’ extension while naming it
  - Say yes when asked if you want to change the file extension of the file
- Copy the blank ‘ssh’ file into the boot partition of the SD card
  - Now eject the SD card from your computer and insert it into the Raspberry Pi.

• Connect the camera

• Plug your Raspberry Pi into power and plug it into Ethernet, you may need to temporarily set it up in a location near your router/modem in order to do this if you plan on using WIFI later on.

• If you know how to get the IP of your Pi from your network from your router or some other method, please do so and note it down. If you do not, you may need to plug a monitor and keyboard into your Pi and type the command: hostname -I

- This article may help if you are having trouble getting your Pi’s IP local address https://www.raspberrypi.org/documentation/remote-access/ip-address.md

• From here on out we will be doing everything over SSH, sending commands remotely to the Raspberry Pi from our main computer/laptop

- For Mac we use Terminal which is part of the operating system, substitute 192.168.1.7 with your Raspberry Pi’s IP address that you found as above.
  - Type: ssh pi@192.168.1.7
  - Then enter the password which is ‘raspberry’
- For Windows you will need to download, install and open Putty - https://www.putty.org/
  - Enter the IP address where is says Hostname
  - Ensure connection type SSH is selected and the Port is 22
  - Click Open and then enter the username ‘pi’ and password ‘raspberry’

• First thing we are going to do now that we are connected to our Raspberry Pi and able to send it commands is to update its firmware to make sure it has the latest version that is compatible with the camera module and the POE hat if your using it.

```
sudo rpi-update
```

WEBCAM AND LIVE STREAMS

The Pi must be updated by typing the following code

```
sudo apt-get update
sudo apt-get dist-upgrade
```

The system will reboot.

Remove existing libraries

Start by removing libraries that may conflict with the newer package. These may or may not already exist on your copy of Raspbian.

```
sudo apt-get remove libavcodec-extra-56 libavformat56 libavresample2 libavutil54
```

Now the next step need to install the following packages, we will need these as the motion software relies on them

```
sudo apt-get install curl libssl-dev libcurl4-openssl-dev libjpeg-dev libx264-142 libavcodec56 libavformat56 libmysqlclient18 libswscale3 libpq5
```

The motion package is required for creating the live streams, the following line of code allows the package to be installed onto the Pi.

```
sudo apt-get install motion
```

Once installed the motion configuration file must be edited. It can be opened using the following code

```
sudo nano /etc/motion/motion.conf
```

This will open the motion configuration file. From here, scroll down until you see daemon off, then change this to daemon on - this activates motion.

Find the following lines and change them to the following

```
stream_localhost off
```

Change the following two lines from on to off If you're having issues with stream freezing whenever motion occurs

```
ouput_pictures off
```

```
ffmpeg_output_movies off
```

`stream_maxrate 100` (This will allow for real-time streaming but requires more bandwidth and resources)

`framerate 100` (This will allow for 100 frames to be captured per second allowing for smoother video)

`width 640` (This changes the width of the image displayed)

`height 480` (This changes the height of the image displayed)

Setting up Daemon in Motion

Next need to setup the daemon, first we need to edit the motion file

```
sudo nano /etc/default/motion
```

then

```
start_motion_daemon=yes
```

Setting up the daemon to automatically run in background

Once the configuration to the file is done, save and exit by pressing CTRL+X then Y. Now make sure the camera is already connected and run the following line

```
sudo service motion start
```

If you need to stop the service, run the following

```
sudo service motion stop
```

Now you should be able to check the webcam stream at the IP address of the Pi in your browser.

For more information about motion: https://motion-project.github.io/index.html

This instruction set for setting up multiple USB cameras. Before that, the USB cameras if the generic camera will be working with this script in terminal.

```
ls /dev/vid*
```

```
/dev/video0 /dev/video1
```

First make a change direction to the folder containing the motion configuration files.

```
cd /etc/motion
```

Type in this command to view all files in the motion folder

```
ls
```

```
camera1-dist.conf  camera2-dist.conf  camera3-dist.conf  camera4-
dist.conf  motion.conf
```

To setup camera 1,

```
sudo nano camera1-dist.conf
```

Edit the file to suit your preferences and press CTRL+O to write out and CTRL+X to exit

```
videodevice /dev/video1
```

```
stream_port 8081
```

Do same for the other camera configuration files selecting different stream ports for each. One configuration file is needed for per camera for them to work.

Open the motion library by typing this `sudo nano motion.conf` on the main configuration file, most of which remains with the default values. The main change to the two thread settings pointing at the above per-camera settings:

```
streamcam_localhost off
```

```
camera /etc/motion/camera1-dist.conf
```

```
camera /etc/motion/camera2-dist.conf
```

Then press CTRL+X then press Y to save the change to the motion library.

Testing your Monitoring System:

Open your web browser to any devices to make sure that the Raspberry Pi is connected to the devices that you intended to stream on; on the same network for me I used the portable hot-spot of my mobile phone as a local network and access the port of the two USB Camera.

Example: 192.168.1.7:8081 for the Camera1 192.168.1.7:8082 for the Camera2.

TUNNELING THE STREAM

To tunnel our video stream from the pi to the backend, we used **ngrok**. **ngrok** is a secure introspectable tunnels to localhost webhook development tool and debugging tool.

To setup ngrok on the pi:

```
sudo wget https://bin.equinox.io/c/4VmDzA7iaHb/ngrok-stable-linux-arm.zip
```

```
unzip ngrok-stable-linux-arm.zip
```

Register for a free account on ngrok website, https://dashboard.ngrok.com/signup. Obtain your authorization token and run the software using the token.

```
./ngrok authtoken <your_auth_token>
```

We then configured the ngrok.yml configuration file to listen to our streaming ports. If you don't specify a location for a configuration file, ngrok tries to read one from the default location `$HOME/.ngrok2/ngrok.yml`. The configuration file is optional; no error is emitted if that path does not exist. Edit the configuration file using `nano ngrok.yml` command. The configuration file can be found in the attached files. For ngrok documentation, https://ngrok.com/docs .

Since ngrok assigns dynamic URLs to all tunnels on the free tier, we make use of the ngrok API by accessing the URLs and posting them to the backend. We wrote a python script, `postURLs.py` that does this job for us.

To make the setup work independently, we wrote a bash script that runs these subsequent scripts to make the whole system function. The `con_ngrok.sh` and `com_backend.sh` files can be found in the repository. `Con_ngrok.sh` sets up the connection to the ngrok servers and `com_backend.sh` fires up the script that posts the camera URLs to the backend.

The bash script needed to be run every time on startup. We make the bash script executable by:

```
chmod 755 con_ngrok.sh
```

```
chmod 755 com_backend.sh
```

Now test it by running:

```
sh con_ngrok.sh
```

```
sh com_backend.sh
```

To log the output from the script, navigate back to your home directory and run:

```
mkdir logs
```

We'll be using crontab. crontab is a background (daemon) process that lets you execute scripts at specific times. It's essential to Python and Raspberry Pi.

```
sudo crontab -e
```

This will bring up a crontab window. Now enter the line at the end of the file:

```
@reboot sh $HOME/con_ngrok.sh > $HOME/logs/cronlog 2>&1
```

```
@reboot sh $HOME/com_backend.sh > $HOME/logs/cronlog 2>&1
```

What this does is rather than executing the launcher script at a specific time, it will execute it once upon startup.

Reboot the device: `sudo reboot now`

To check the logs,

```
cd $HOME/logs
```

```
cat cronlog
```