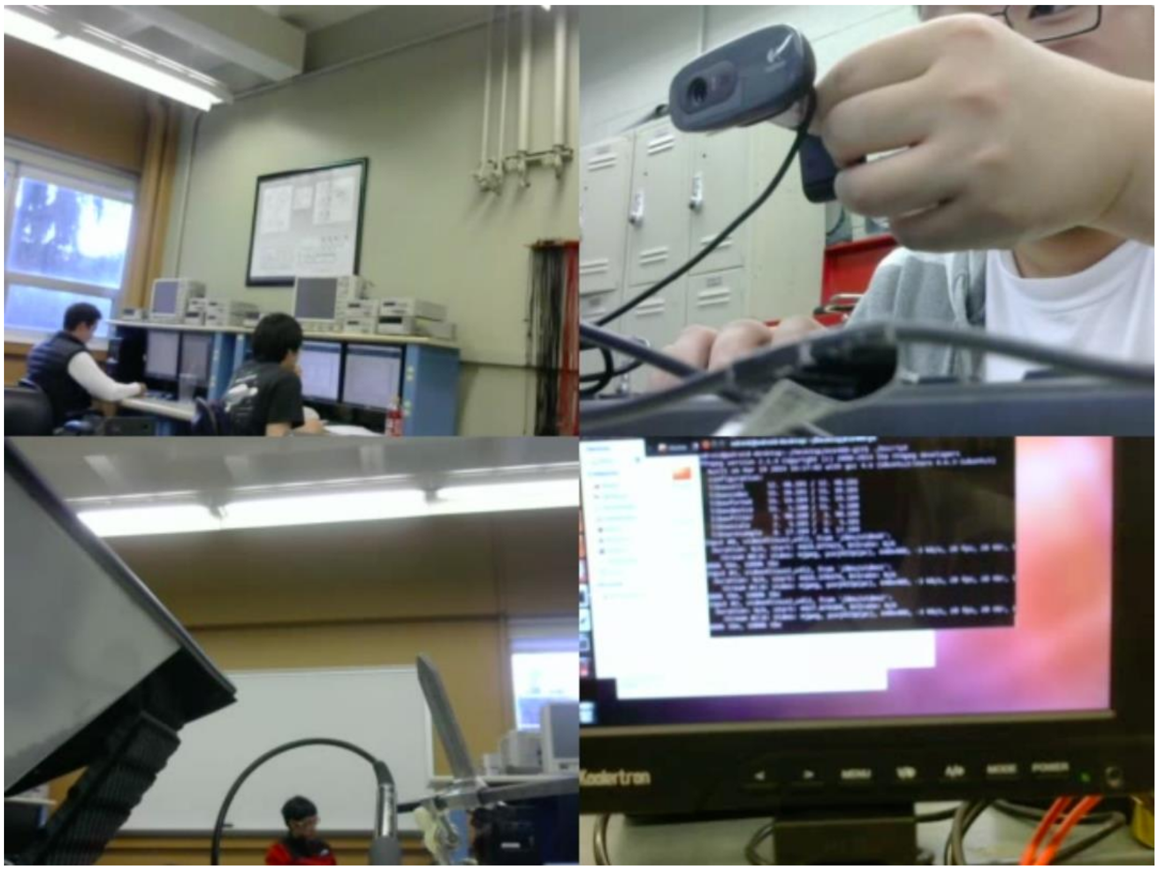
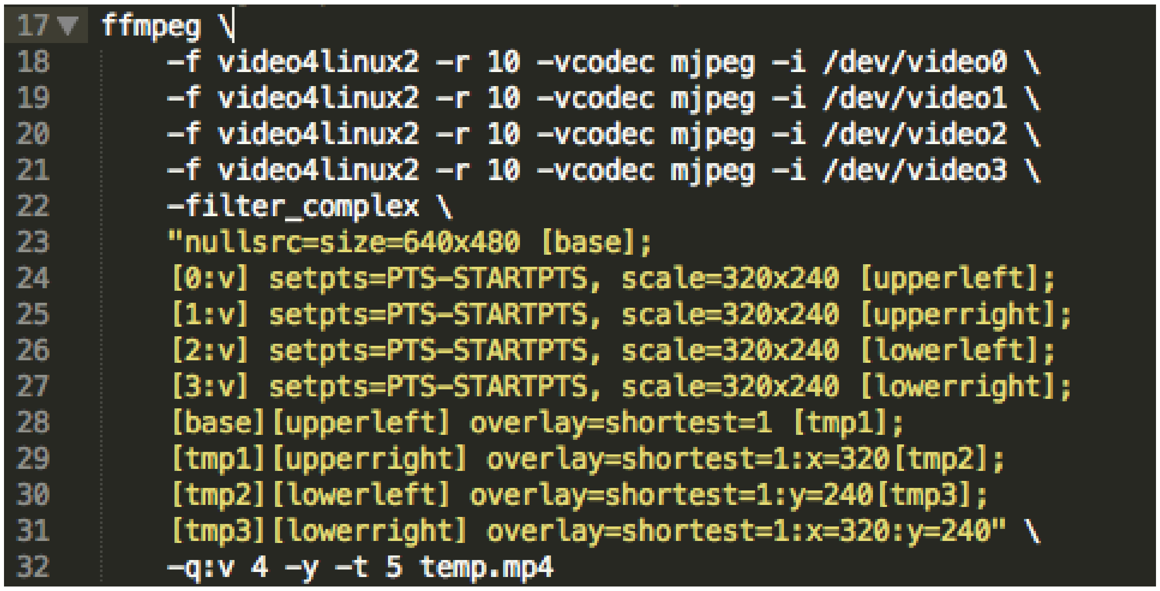
Customizing FFMPEG Commands for Mosaic View



Xinye Ji

General Overview of a base script:



```
17   ffmpeg \
18       -f video4linux2 -r 10 -vcodec mjpeg -i /dev/video0 \
19       -f video4linux2 -r 10 -vcodec mjpeg -i /dev/video1 \
20       -f video4linux2 -r 10 -vcodec mjpeg -i /dev/video2 \
21       -f video4linux2 -r 10 -vcodec mjpeg -i /dev/video3 \
22       -filter_complex \
23       "nullsrc=size=640x480 [base];
24       [0:v] setpts=PTS-STARTPTS, scale=320x240 [upperleft];
25       [1:v] setpts=PTS-STARTPTS, scale=320x240 [upperright];
26       [2:v] setpts=PTS-STARTPTS, scale=320x240 [lowerleft];
27       [3:v] setpts=PTS-STARTPTS, scale=320x240 [lowerright];
28       [base][upperleft] overlay=shortest=1 [tmp1];
29       [tmp1][upperright] overlay=shortest=1:x=320[tmp2];
30       [tmp2][lowerleft] overlay=shortest=1:y=240[tmp3];
31       [tmp3][lowerright] overlay=shortest=1:x=320:y=240" \
32       -q:v 4 -y -t 5 temp.mp4
```

In order to change certain settings while recording a live stream with FFMPEG, this base shell script must be changed. The description of each option is as follows:

Line 17: ffmpeg – telling the terminal that ffmpeg is being used.
Lines 18-21: Defining the input feeds and the settings of each feed below.

- Video4linux2 is an open source package that allows the machine to have a live feed of the webcam.
- -r 10 refers to the frame rate (in frames per second)
- -vcodec refers to the video codec that the stream is encoded in
- -i refers to the input source.

Lines 22-31: Complex Filters – This allows the user to add either color filters or overlays.

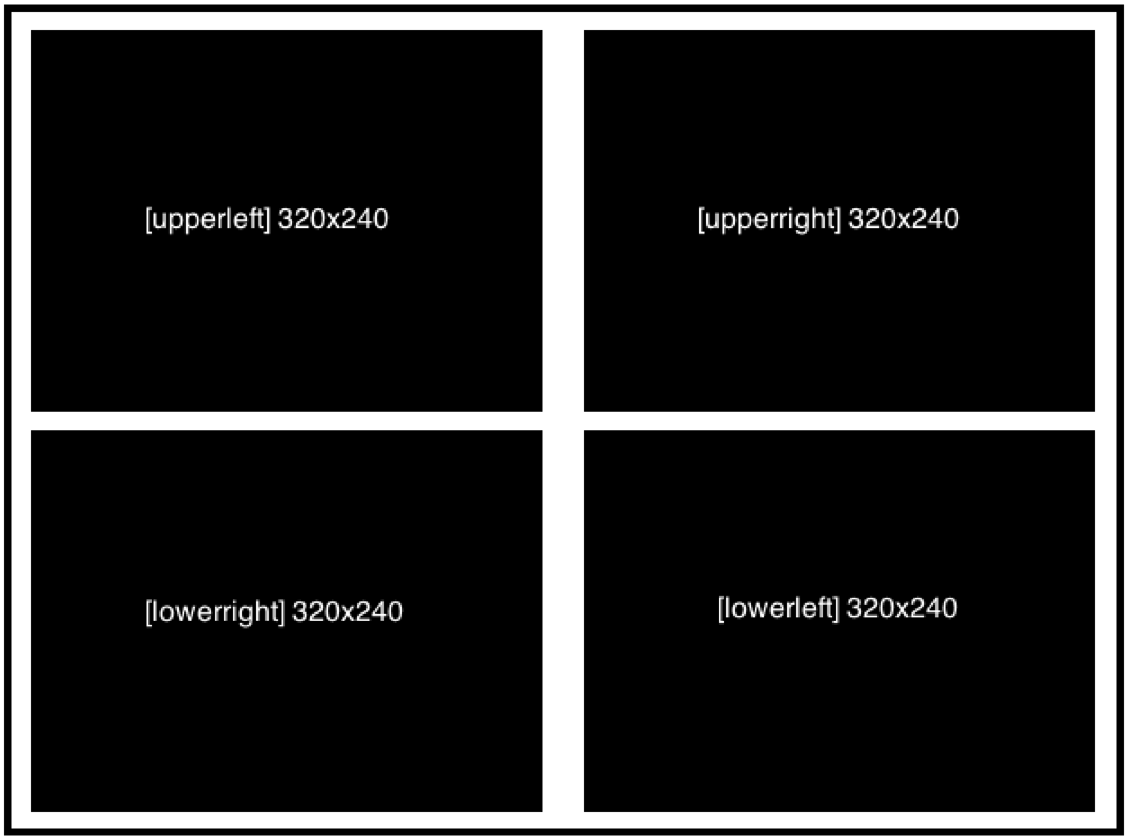
- filter_complex: the call to implement filters, this is formatted and constructed in the quoted text (highlighted yellow)
- Line 23: This refers to the base resolution of the whole video. [base] can be any name.
- Lines 24-27: Instantiating the input feeds
    - [n:v] : The numbering of the inputs, in this case 0-3
    - setpts : sets the time point. In this case, PTS-STARTPTS refers to 0, ensuring that all video streams are synchronized.
    - Scale = the individual resolution of the respective feed.
    - The last part refers to the name of this input. It can be labeled whatever is needed. In this case, it labels the four corners.
- Lines 28-31: Calls the base layer and immediately stacks the input feeds in between temporary layers.

    - Overlay=shortest=1 enables the video to end as soon as the shortest video feed ends.
    - The X and Y calls simply refer to the upper left corner of each feed. If x = 320 and y = 240, that means that the upper left corner of a video feed is located at 320 by 240.

Line 32: Settings for the output video

- -q:v – refers to the video quality. This number can range between 1-32; one (1) being the highest quality, thirty-two (32) being the lowest quality.
    - The higher the quality, the larger the file size.
- -y – This enables the script to overwrite a file.
- -t – Sets the time in seconds. Additionally, 00:00:00.00 can also be inputted.
- temp.mp4 – The output file name, this can be named everything.

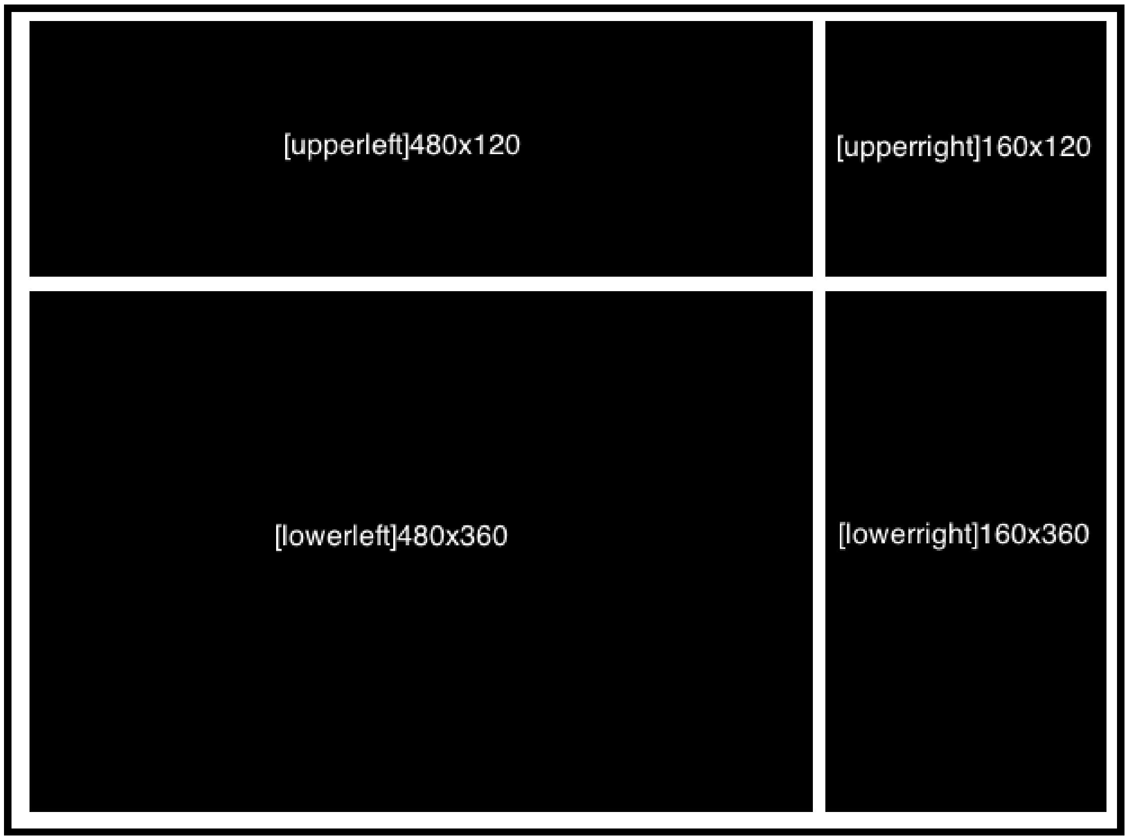
Modifying the complex filter
Being able to modify the filter is pivotal to custom tailoring the live feeds to your needs. Currently the feed has a layout similar to this.



Problem: What if one camera feed needs more detail than the rest?

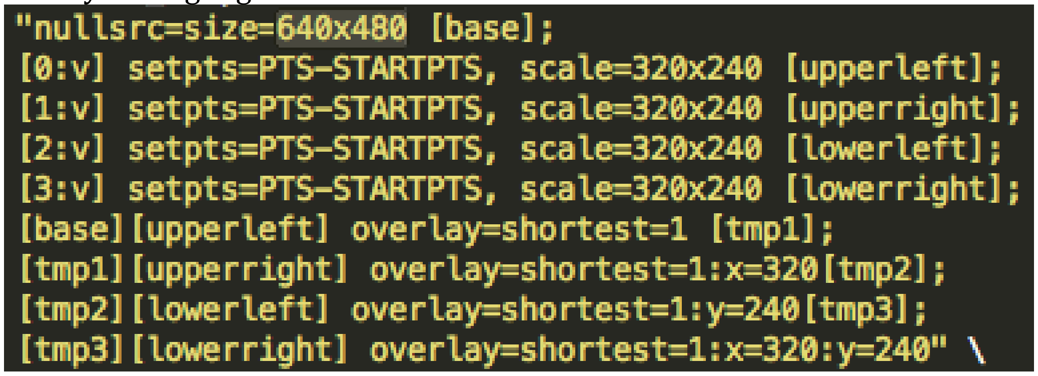
Solution: Expanding that view and cropping the others respectively.

One way to do this is:



While this isn't the most efficient method, it serves the purposes of the tutorial well.

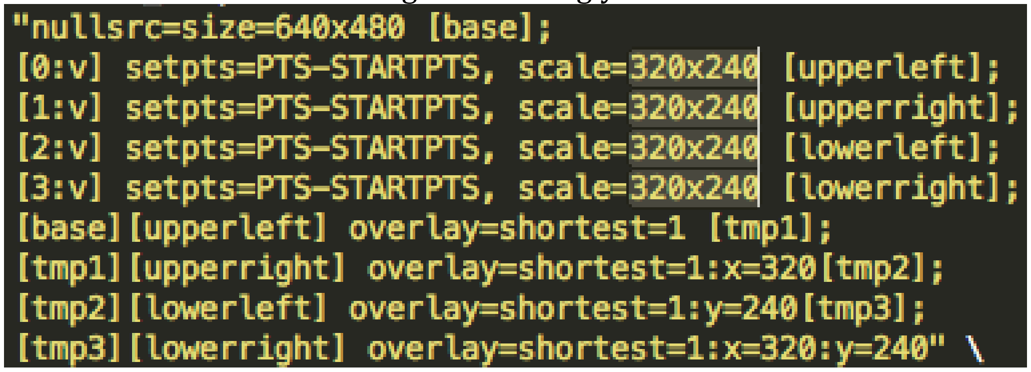
Step 1: Setting the base resolution
The total resolution of the video can be changed here. In this situation, do not modify the highlighted value.



```
"nullsrc=size=640x480 [base];
[0:v] setpts=PTS-STARTPTS, scale=320x240 [upperleft];
[1:v] setpts=PTS-STARTPTS, scale=320x240 [upperright];
[2:v] setpts=PTS-STARTPTS, scale=320x240 [lowerleft];
[3:v] setpts=PTS-STARTPTS, scale=320x240 [lowerright];
[base][upperleft] overlay=shortest=1 [tmp1];
[tmp1][upperright] overlay=shortest=1:x=320[tmp2];
[tmp2][lowerleft] overlay=shortest=1:y=240[tmp3];
[tmp3][lowerright] overlay=shortest=1:x=320:y=240" \
```

Step 2: Resizing the individual feeds.

These values need to be changed accordingly:



```
"nullsrc=size=640x480 [base];
 [0:v] setpts=PTS-STARTPTS, scale=320x240 [upperleft];
 [1:v] setpts=PTS-STARTPTS, scale=320x240 [upperright];
 [2:v] setpts=PTS-STARTPTS, scale=320x240 [lowerleft];
 [3:v] setpts=PTS-STARTPTS, scale=320x240 [lowerright];
 [base][upperleft] overlay=shortest=1 [tmp1];
 [tmp1][upperright] overlay=shortest=1:x=320[tmp2];
 [tmp2][lowerleft] overlay=shortest=1:y=240[tmp3];
 [tmp3][lowerright] overlay=shortest=1:x=320:y=240" \
```

Upper left: 480x120
Upper right: 160x120
Lower left: 480x360
Lower right: 160x360

Step 3: As this script stands, this output will not be positioned tightly, there will be odd black space between the feeds, as such we need to change the x and y coordinates of each feed.

Upper left: No changes needed, begins at (0, 0)
Upper right: (480, 0), x must be set to 480 instead of 320.
Lower left: (0, 160), y must be set to 160 instead of 240.
Lower right: (480, 360), x must be 480, and y must be 360.

INDEX
Settings table

	Valid Input	Function	Examples
-f	Live feeds	Allows FFMPEG to accept video feeds from other audio sources	video4linux2, x11grab
-i	file directory/ names	input file location	/dev/video0, /Desktop/video.mov
-r	0 < x < inf	Frame rate read in frames per second	1, 24, 29.975, 50
-vcodec	Codec libraries	Determines the codec to encode the video input or output stream	Libx264, mjpeg, mpeg, etc.
-filter_complex	Varies depending on string	Allows manipulation of complex/multiple input streams.	See "Modifying the Complex Filter"
-q:v	Number between 1-32	Determines the video quality. (Balance between file size and quality loss)	1, 4, 5, 30
-y	No input	Allows script to overwrite existing video file	N/A
-t	Any integer (in seconds) or xx:xx:xx.yy where xx is any number from 0-59, and yy is between 0-99	Determines length of the video.	5, 00:00:05.03, 50:01:00.09