Welcome to Augus Video Encoding Testbed. This Testbed will lead you to recover an experiment called "The Comparison in H.264, H.265, VP9 and AV1 Encoder with Different Encoding Parameters". Please read carefully about this README and follow the instructions step by step.

I Pre-installation

Linux: A Linux operating system, Ubuntu 18.04 LTS and CentOS 7 are recommended.

ffmpeg: A static ffmpeg include popular codecs and ffprobe is recommended. You can download it by http://www.ee.ucl.ac.uk/~iandreop/ffmpeg_static_with_VMAF.zip. Also, this downloaded version is provided in the testbed by default. Thus, you don't need download it by yourself. If you download it by yourself, you need to recombine the source code with the ffmpeg folder and create sub folders inside the ffmpeg.
This version doesn't include ffprobe, if you would like to install a ffprobe, please download and install the complete virsion at http://ffmpeg.org/download.html#build-linux. However, you need rewrite the source code to let the code can run a proper ffmpeg command and combine them together.

Python: Python 2.7 or 3.7 are both fine. After python is built, please use pip to install fours libs:

1. wxPython – sudo pip install -U wxPython (if you used python3 please type pip3 instead).
2. Pandas & numPy – sudo pip install pandas && sudo pip install numpy
3. Matplotlib – sudo pip install matplotlib. There might be a problem when you install it. The error massage shows that "Cannot uninstall 'pyparsing'". To solve this problem, you need run: sudo pip install -I pyparsing==2.2.0 to reinstall the error package. However, when you do "import matplotlib.pyplot as plt". There might be an error massage say that "ImportError: No module named Tkinter". This is because your python can't run into a Linux TK interface. TK is a Linux standard GUI interface. This is because most of the Linux user run python on command line interface. Thus, a simple CLI cannot draw any figures, it needs the support of GUI. In this case you need to switch your plot project (only plot part) to some IDEs, for example Spyder in Windows Anaconda.

Other optional tools: Media-info, MKVToolnix and MP4Box

II Get start

When the basic environment is set up, unzip the 2 zip files and you can see there are three main parts:

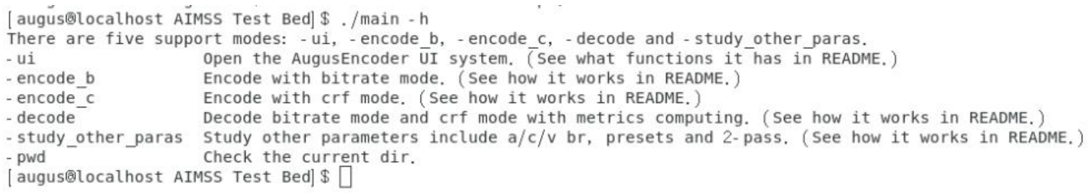
1. FFmpeg static
2. Source code
3. The main executable control program

Before you start, you should click into the ffmpeg static folder and there is an executable file called ffmpeg. Use chmod 777 (or 755) to give it the highest permission. After that, go back to the first folder to run the main control program by using ./main -para

There are five supported modes in this test bed.

1. Encode with bitrate mode.
2. Encode with crf mode.
3. Decode bitrate mode and crf mode with metrics computing
4. Study other parameters mode include a/c/v br, presets and 2-pass.
5. UI mode, this AugusEncoder is a very simple UI system to let the user start another test by them self.

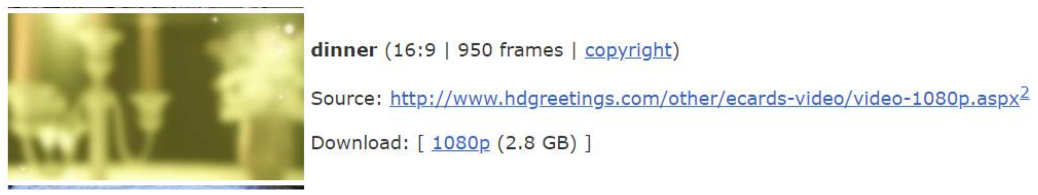
You can use ./main -h to see how to access each mode.



```
[augus@localhost AIMSS Test Bed]$ ./main -h
There are five support modes: -ui, -encode_b, -encode_c, -decode and -study_other_paras.
-ui                 Open the AugusEncoder UI system. (See what functions it has in README.)
-encode_b           Encode with bitrate mode. (See how it works in README.)
-encode_c           Encode with crf mode. (See how it works in README.)
-decode             Decode bitrate mode and crf mode with metrics computing. (See how it works in README.)
-study_other_paras  Study other parameters include a/c/v br, presets and 2-pass. (See how it works in README.)
-pwd                Check the current dir.
[augus@localhost AIMSS Test Bed]$ 
```

It’s better to do the test in this order:
Run mode1 and 2 together. After 1 and 2 are finished, use mode 3 to decode all the encoded file generated by mode 1 and 2 and assess them.
When mode 1, 2, 3 mode is finished, start a simple test on mode 4 to observe some other parameters.
At last, run the UI mode to start some other testes by yourself.

You should prepare a 1080P 30fps y4m video by yourself. There is no initial one inside the test bed. Download a good y4m video at: https://media.xiph.org/video/derf/
Find:



dinner (16:9 | 950 frames | copyright)

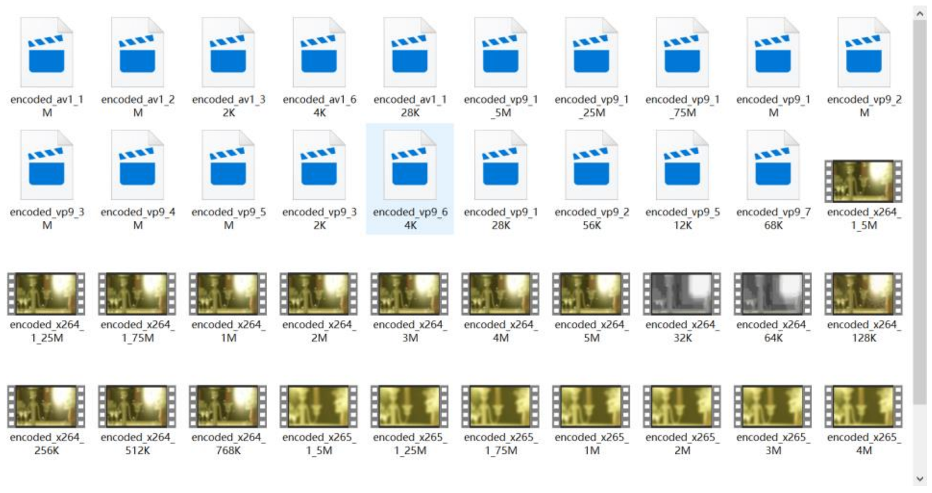
Source: http://www.hdgreetings.com/other/ecards-video/video-1080p.aspx[2]

Download: [1080p (2.8 GB)]

If you want to use your own one, please name it as source.y4m and put it to:
./ffmpeg_static_with_VMAF/ffmpeg_static/y4m/

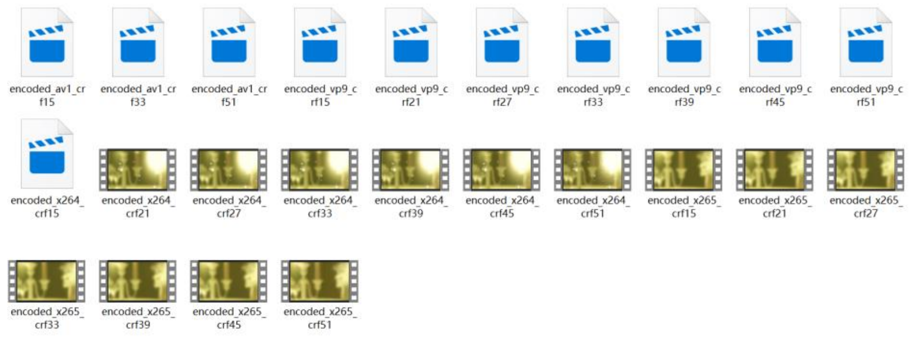
III encode with bitrate mode

This mode let you to encode a series of video by the target bitrate mode (abr mode). Before you start, it’s better to delete all the encoded exist video in:
./ffmpeg_static_with_VMAF/ffmpeg_static/encoded/
Type ./main -encode_b to start the program. When the program is finished, there will be **47** videos generated. The encoding process might take at least one day because av1 encoder is very slow for now. Thus, if you can’t bear the speed, just comment the lines for encoding av1 in ./souce_code/encode_with_bitrate.py



IV encode with crf mode

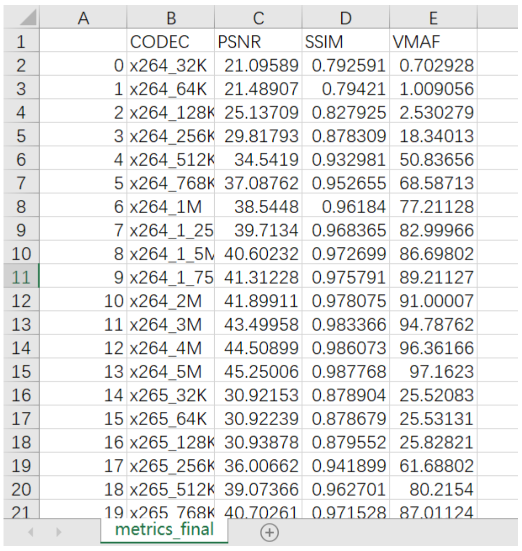
This mode let you to encode a series of video by the constant quality mode (crf mode). Before you start, it's better to delete all the encoded exist video in:
./ffmpeg_static_with_VMAF/ffmpeg_static/encoded_crf/
Type ./main -encode_c to start the program. When the program is finished, there will be **24** videos generated. The encoding process might take at least one day because av1 encoder is very slow for now. Thus, if you can't bear the speed, just comment the lines for encoding av1 in ./souce_code/encode_with_crf.py
After encoding finished, there will be some time record in:
./ffmpeg_static_with_VMAF/ffmpeg_static/ for example time_record_crf15.txt. These are going to be used at the last part.



V Decode bitrate mode and crf mode with metrics computing

This mode let you to decode all the encoded videos generated before. Type ./main -decode to start the program. When the program is finished, there will be a metrics_final.csv created in ./ffmpeg_static_with_VMAF/ffmpeg_static/. This is the assessment result. It's going to be used at the last part.
The decoding and computing process might take at least 10 hours because the assessment process is slow (PSNR, SSIM and VMAF).

The csv file looks like below:



	A	B	C	D	E
1		CODEC	PSNR	SSIM	VMAF
2	0	x264_32K	21.09589	0.792591	0.702928
3	1	x264_64K	21.48907	0.79421	1.009056
4	2	x264_128K	25.13709	0.827925	2.530279
5	3	x264_256K	29.81793	0.878309	18.34013
6	4	x264_512K	34.5419	0.932981	50.83656
7	5	x264_768K	37.08762	0.952655	68.58713
8	6	x264_1M	38.5448	0.96184	77.21128
9	7	x264_1_25	39.7134	0.968365	82.99966
10	8	x264_1_5M	40.60232	0.972699	86.69802
11	9	x264_1_75	41.31228	0.975791	89.21127
12	10	x264_2M	41.89911	0.978075	91.00007
13	11	x264_3M	43.49958	0.983366	94.78762
14	12	x264_4M	44.50899	0.986073	96.36166
15	13	x264_5M	45.25006	0.987768	97.1623
16	14	x265_32K	30.92153	0.878904	25.52083
17	15	x265_64K	30.92239	0.878679	25.53131
18	16	x265_128K	30.93878	0.879552	25.82821
19	17	x265_256K	36.00662	0.941899	61.68802
20	18	x265_512K	39.07366	0.962701	80.2154
21	19	x265_768K	40.70261	0.971528	87.01124

Hint: the decode process needs at least 5GB (relation to your video length).

VI Study other parameters mode include a/c/v br, presets and 2-pass.

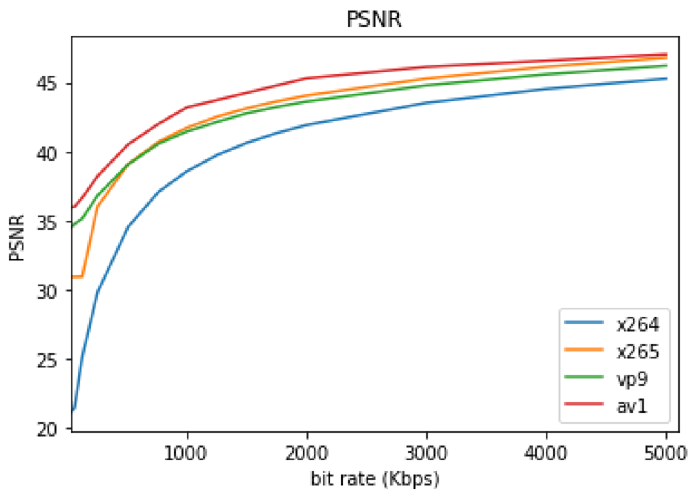
This mode is very similar to the previous three one because it's a combined of them with different encoding parameters. before you start, it's better to delete all the encoded exist video in:

./ffmpeg_static_with_VMAF/ffmpeg_static/encoded_cbr/
./ffmpeg_static_with_VMAF/ffmpeg_static/encoded_preset/
./ffmpeg_static_with_VMAF/ffmpeg_static/encoded_2pass/

Type ./main -study_other_paras to start the program. The encoding and decoding speed is fine because it's only used the x.264 encoder. There are going to output 3 files, metrics_cbr.csv, metrics_preset.csv and metrics_2pass.csv in ./ffmpeg_static_with_VMAF/ffmpeg_static/

VI Plot figures

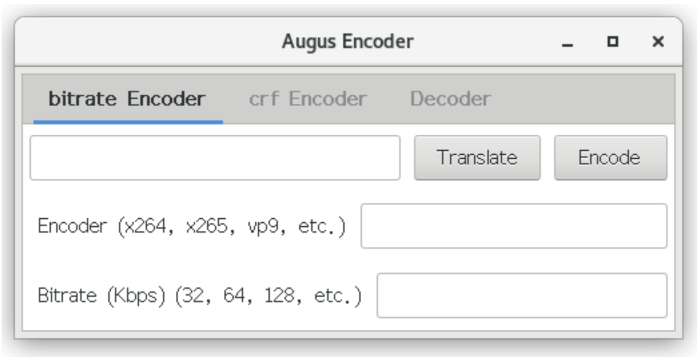
When you finish to collect all the assessment result csv file, put them into the default directory for the GUI python (Spyder). Run Vedio_Metrics_test.py to plot all the possible figures. An example figure looks like below:

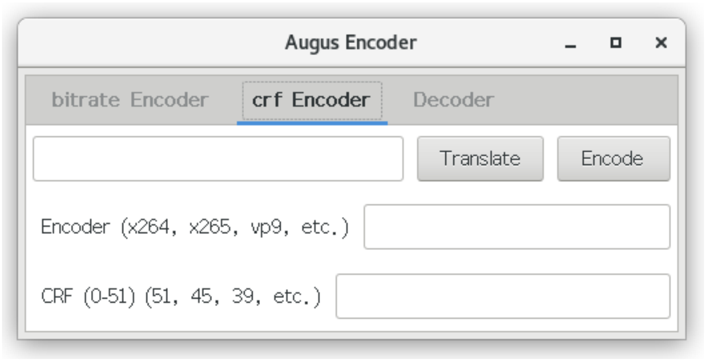


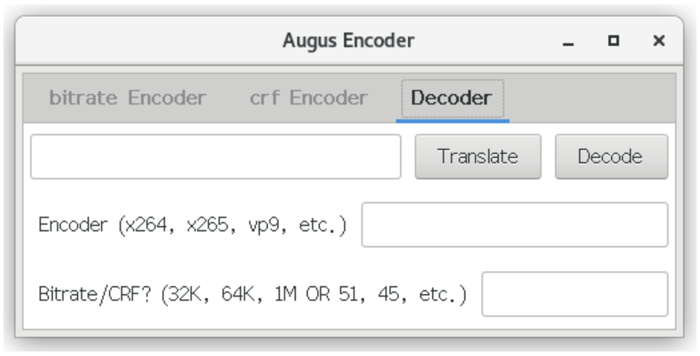
When you get these figures, the main part of this testbed is finish!

VI UI mode

This user-friendly user interface is an alternative part for this test bed. The user can use this UI to encode, decode and assess some videos. There are three modes: bitrate encoder, crf encoder and decoder. The user should type some correct parameters in the box. Translate it (Translate button) to a command and run the command (Encode/Decode button).


Augus Encoder
bitrate Encoder
crf Encoder
Decoder
Translate
Encode
Encoder (x264, x265, vp9, etc.)
Bitrate (Kbps) (32, 64, 128, etc.)




Augus Encoder
bitrate Encoder
crf Encoder
Decoder
Translate
Decode
Encoder (x264, x265, vp9, etc.)
Bitrate/CRF? (32K, 64K, 1M OR 51, 45, etc.)

Hints:

1. in bitrate encoder, x264 is the standard format for encoder parameter and **NO** K for bitrate parameter. Click Translate button and Encode buttons.
2. In crf encoder, x264 is the standard format for encoder parameter and a number for crf parameter. Click Translate button and Encode buttons.
3. In decoder, x264 is the standard format for encoder parameter. Just a number for crf encoded video or a number **+ K** for bitrate encoded video. There is a K or not decide the encoded video. Click translate and Decode buttons. **Also, a metrics computing is done here.**

The output video for encode is in: ./ffmpeg_static_with_VMAF/ffmpeg_static/encoded_ui/
The output csv for decode is in: ./ffmpeg_static_with_VMAF/ffmpeg_static/metrics_ui.csv

So far, all the functions provided by this test bed is finished. If you have further questions, please contact yuxuan.chen.18@ucl.ac.uk.