International Journal of Trend in Scientific Research and Development (IJTSRD)
Volume 5 Issue 4, May-June 2021 Available Online: www.ijtsrd.com e-ISSN: 2456 – 6470

A Study on FFmpeg Multimedia Framework

H. Sumesh Singha[1], Dr. Bhuvana J[2]

[1]Master of Computer Science Applications (SCT), [2]Associate Professor,
[1,2]Jain Deemed-to-be University, Bengaluru, Karnataka, India

ABSTRACT

With the advancement in internet technology, everyone has access to the internet. After google, YouTube is the second largest search engine and approximately 1 billion hours are consumed by people to watch video contents on YouTube. Editing the video and processing is not very easy. Network also plays an important role. With an unsteady network it can cause video to buffer which can reduce the streaming experience of users. Many people don't even have a good computer which can handle the editing of large video files as editing and processing the video utilizes hardware, software and both. Many video editing software are available on the internet. Either it can be paid or open-source software. One of the most popular open-source software available on the internet is FFmpeg (Fast Forward Moving Picture Expert Group). FFmpeg with other various software together can be used for video forensic to find traces in videos. It becomes very difficult to find traces from videos that are highly compressed or the video has low resolution. In earlier times, fetching data from camera of robots and encoding the data with software generates an issue. JNI,NDK, FFmpeg, researching about these video annotations a video player was created to examine video of sports so that user can see the how player evaluates the action practically with efficiently. Demand of multimedia increase as times goes on. Today in this global pandemic, everyone has move to digitalization. From studies to working everything has been digitalized. In this paper we are going to study about FFmpeg, how it benefits user with its features. Combining this highly popular multimedia framework with other software can create some useful technologies. Well, FFmpeg is mostly known for its memory efficiency and time efficiency. From processing image to editing videos everything can be acquired from FFmpeg.

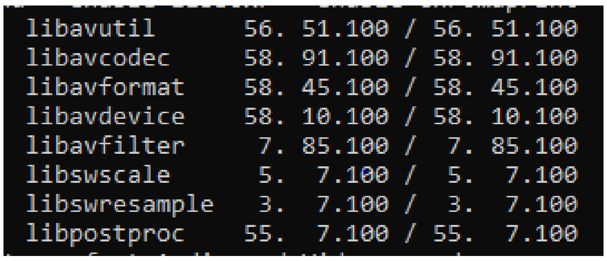
KEYWORDS: *FFmpeg, forensic, JNI, NDK*

How to cite this paper: H. Sumesh Singha | Dr. Bhuvana J "A Study on FFmpeg Multimedia Framework" Published in International Journal of Trend in Scientific Research and Development (ijtsrd), ISSN: 2456-6470, Volume-5 | Issue-4, June 2021, pp.580-584, URL: www.ijtsrd.com/papers/ijtsrd42362.pdf

IJTSRD42362

1. INTRODUCTION

Innovations in multimedia technology has modified the method of creating videos. A good video should have better quality with lower video size. There are abundant number of tools for manipulating the video such as transcoding the video, encoding the video and audio video broadcasting. This is where FFmpeg comes into play. Fast Forwarding Moving Picture Expert Group (FFmpeg) is a leading open-source multimedia framework available on the internet. It can perform all the task that human have created till now such as Multiplexing video, De-multiplexing video, video filtering and many others. If you look at the logo of FFmpeg on its official website, you will find that it has a zigzag pattern. This zigzag pattern represents Entropy Encoding that is handled by MPEG video codecs. FFmpeg provides three types of tool i.e., FFmpeg tool, FFplay tool, FFprob tool. The user interface of FFmpeg is not very user friendly because it is completely based on command line tool. So, in order to edit or manipulate a video, one has to remember the command to be executed for the required video. FFplay is the player which is used for playing the files. Whereas FFprob tool is a stream analyzer tool. It is also known as FFprobe Stream Analyzer. This software is available for Linux, Mac OS, Windows. The usage of FFmpeg is ffmpeg [options] [[infile options] -iinfile]... {[outfile options] outfile}. FFmpeg has different types libraries that are LibavDevice, LibavCodec, LibavFormat, LibavSample, LibavScale, LibavFilter, LibavUtil. In Figure 1. Also, you can see some of the libraries using the Command prompt. From simple video editing task to complex task everything can be done using FFmpeg. The main advantage of FFmpeg is that it is lightweight and it doesn't consume too much memory for processing the video. You don't even need to use any third-party software to manipulate your videos. With only FFmpeg you can edit and process the video. Many of the website and applications software uses FFmpeg such as VLC, Google chrome etc.



```
libavutil      56. 51.100 / 56. 51.100
libavcodec     58. 91.100 / 58. 91.100
libavformat    58. 45.100 / 58. 45.100
libavdevice    58. 10.100 / 58. 10.100
libavfilter     7. 85.100 /  7. 85.100
libswscale      5.  7.100 /  5.  7.100
libswresample   3.  7.100 /  3.  7.100
libpostproc    55.  7.100 / 55.  7.100
```

Figure 1. FFmpeg Libraries

2. Methodologies

Encoding video is a method where a video is transformed from raw video to digital form which can be supported in many devices. Codec is a compression technology used to compress a video into a desired size for convenient storage and delivery of video. Some of the most common encoding technologies are H.264/AVC, H.265/HEVC, AV1, VP9, H.266/VVC. If see the structure of H.264 then you will find that it has hybrid structure. [7] Using H.264 has some major benefits over H.263 codec. Over half of the rate is saved than H.263 which enhance the anti-inference ability. H.264 can be split into six encoding technologies.

A. *Inter prediction encoding*–It is that to minimize the data redundancy it image information between frames which can be achieved by the usage of by using Motion Estimation and Compensation.
B. *Intra prediction encoding* – this encoding technique is used in image frame sequence to remove Spatial redundancy from it.
C. *Transform Compression* – Using this technique the prediction residual can be converted into frequency coefficient.
D. *Quantization*- this technique is broadly utilized in Simple to-Digital Converter (ADC) framework where data are mapped within a predetermined bit of number and range.
E. *Loop Filter* – this eliminates the blocking effect which results in enhanced quality of image.
F. *Entropy* – to attain data that is compressed the probability distribution of video information is used along with VLC and CABAC. Entropy coding is also called Statistical coding.

FFmpeg can manipulate the codecs of video. You just need to mention the codec type on your script. For example - ffmpeg -i input.mov -c:v libx264 -preset slow -crf 22 -c:a copy output.mp4. Changing one video format to another is knows as video transcoding. In the above command the -c:v represent the mode of video coding and -c:a represents the the mode of audio coding. Libx264 is the video codec format. -preset slow represent the encoding speed which will generate low volume size video. [6]There are different types of protocols for transmission such as HTTP, RTMP, RTP etc. Some media container formats are FLV, MP4, AVI, MKV. So, system that relies on specific transmission protocol is not very appropriate. Reference [6] discussed how they created a real time video stream analysis system that uses FFmpeg as their kernel. Making it adaptable to different scope of media data formats. The system has two section – decoding section and analysis section. The decoding section has decoding protocol that are URL Protocol, URL Context where different types of input media protocol are stored. Next it has decoding fromat which has AV Format Context which is used to reserve media container format. Next it has AV stream that has information about the audio and video stream. AVPacket is where audio and video that are encoded is stored. Analysis section gathers the parameter generated from the decoding section and analyses them. This section can be divided into two different modules. One is Simple analysis module and the other is detail analysis module.

Reference [10] introduced a video player for sports based on android platform. This player was created using different technologies i.e., JNI, NDK, SQLite, FFmpeg. JNI stands for Java Native Interface. JNI help code that are written using Java can interact with the code written in C/C++. This can be achieved by loading code form dynamic shared libraries. NDK stands for Native Development Kit. Using NDK partial function can be implemented using native code such as C++ and C. This helps developers in code reusability and the development is improved efficiently. SQLite is type of database for android. It is a relational database and it is lightweight in nature. This video player plays the video along side it also displays the tactical information of the video. With the use of FFmpeg results shows that the player consumes very less resources of the system. Reference [7] introduce a process of an image that is based on H.264 protocol. It includes transmission mode such as RTP/RTCP. RTP stands for Real-Time Transport Protocol. Based on Unicast and Multicast service end-to-end real-time data is delivered by RTP. Whereas RTCP stands for Real-Time Transport Control Protocol. RTP is used with RTCP. Stream of media are carried by RTP and quality of service is measured by RTCP. H.264 protocol compressed the video stream then the information of the video stream is delivered via internet by using the RTP/RTCP. The H.264 format video stream is then decoded by using FFmpeg and to display the video image on the device screen multimedia development kit SDL is applied on it. Reference [8] introduced HLS player using the FFmpeg and S3 bucket. This player allows user to have better streaming service. User uploads various videos of different file format which is then delivered over HTTP. These videos are stored at the database in multiple chunks and managing these chunks is not an easy job. these chunks are stored in AWS S3 bucket which is storage service provided by Amazon Web Service. To manage these chunks, they used an automation tool called Ansible.

3. Algorithm

In FFmpeg, transformation of format and scaling of image are acquired with the use of sws_scale algorithm. Sws _scale uses algorithms of different flavours for image processing. Following options are available for the sws_scale algorithm.

#defineSWS_FAST_BILINEAR 1#define SWS_BILINEAR 2#define SWS_BICUBIC 4#define SWS_X 8#define SWS_POINT 0x10#define SWS_AREA 0x20#define SWS_BICUBLIN 0x40#define SWS_GAUSS 0x80#define SWS_SINC 0x100#define SWS_LANCZOS 0x200#define SWS_SPLINE 0x400

For the experiment, I used my Asus device with i3 3rd Gen processor and has a 12 gigabyte of RAM. I used an image in landscape form which has a scaling of 1920 * 1080. Then the image was scaled to 400 * 300 of 24-bit RGB. At the end of the test, I found that the time taken for rendering the image was almost negligible. Most of the time was used for scaling algorithm.

Algorithm	Image Subjective Feeling	Frame Rate
SWS_FAST_BILINEAR	Effect of the image was found to be very good with no distortion.	228
SWS_BILINEAR	This image was also good than the above algorithm and the image edge was much smoother.	95
SWS_BICUBIC	Results was found to be same but edge was much smooth than the e above and image was much sharper than the first algorithm.	80
SWS_X	No difference was found.	91

SWS_POINT	Comparing with the above picture effect of image was poor but the details had better sharpness.	427
SWS_AREA	No difference was found	116
SWS_BICUBLIN	Ibid	87
SWS_GAUSS	Smoothness can be done correlated to above algorithm (even blur).	80
SWS_SINC	Clear detail was found than the above algorithm.	30
SWS_LANCZOS	Has better smoothness than the above one (also can be made blur), very much to be identical.	70
SWS_SPLINE	No difference was found	47

As the table with different algorithms shows that, the quality of the image was good after the effect. If you ignore the image contrast, scaling effect can be seen hardly. The sharpness and blurriness shown in the above are known as objective feeling. Result shows that efficiency was high and the effect on the image was good.

Moving to the second experiment, here I used image in landscape of scaling 1024 * 768. This image was then zoomed to 1920 * 1080. We could say that the rendering time was not fully negligible. There was a rendering time of less than 5ms. This experiment was opposite to the above experiment.

Algorithm	Image Subjective Feeling	Frame Rate
SWS_FAST_BILINEAR	Effect of the image was found to be very good with no distortion.	103
SWS_BILINEAR	No difference was found.	100
SWS_BICUBIC	Clear details had found correlated to above algorithm.	78
SWS_X	No difference was found.	106
SWS_POINT	Jaggery edges.	112
SWS_AREA	No jaggery edges.	114
SWS_BICUBLIN	No difference found	95
SWS_GAUSS	Image seemed to be slightly more cleared compared to the above.	86
SWS_SINC	No difference found.	20
SWS_LANCZOS	No difference found.	64
SWS_SPLINE	No difference was found	40

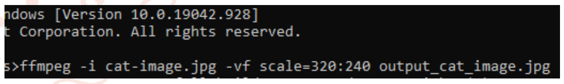
Overall, if we see the result there was no major difference can be seen with the naked human eye. Clear saw tooth was found with Point algorithm whereas no clear saw tooth found with area algorithm. When CImage is used for rendering, you can see a peak value in the frame rate of value 105. Which was same as point effect.

As per my recommendation, for image scaling, efficiency should be high. Let's take an example if you are confused whether the image should be zoomed in or zoomed out then usage of SWS_FAST_BILINEAR algorithm is a better choice. If you want the image to be zoomed out then and later to be displayed then I highly suggest you to go with the point algorithm. If you want image to be zoomed in explicitly then using CImage seemed to be much more efficient. If you do not care about the speed to achieve a better quality then selecting the frame rate with the lowest value shows good result. Since, we used scaling algorithm of FFmpeg, the result was still very fast even if we used 1920 * 1080 high-definition image for the experiment.

FFmpeghas got may features one of them is scale filter. Different types of task can be achieved by using the scale filter. For the experiment we are going to use an image of resolution 1680 * 1080 pixels.

Figure 2 Cat image.

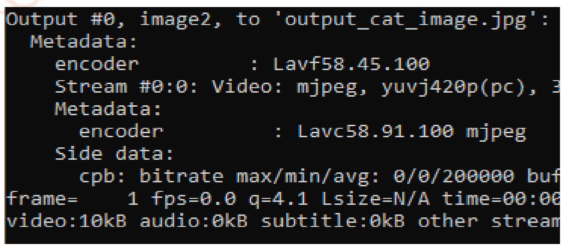
First let's see about simple rescaling. If you want your video to be rescaled to a particular size let say in 320 * 240 then most basic form of scale filter will be "*ffmpeg -i video.mp4 -vf scale=320:240 output.mp4*". If you are working with image then the above script will look like FFmpeg -i input.jpg -vf scale=320:240 output.jpg. only you need to change the input and output file.



```
ndows [Version 10.0.19042.928]
t Corporation. All rights reserved.

s>ffmpeg -i cat-image.jpg -vf scale=320:240 output_cat_image.jpg
```

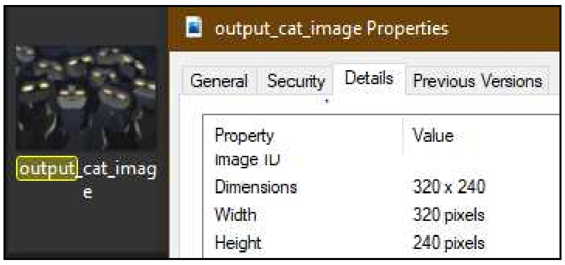
Figure 3 command line



```
Output #0, image2, to 'output_cat_image.jpg':
  Metadata:
    encoder         : Lavf58.45.100
    Stream #0:0: Video: mjpeg, yuvj420p(pc), 3
    Metadata:
      encoder         : Lavc58.91.100 mjpeg
    Side data:
      cpb: bitrate max/min/avg: 0/0/200000 buf
frame=    1 fps=0.0 q=4.1 Lsize=N/A time=00:00
video:10kB audio:0kB subtitle:0kB other stream
```

Figure 4 processing image

After running the script in the command prompt, you can observe that the input image pixel was changed from 1680 * 1080 to 320 * 240 pixels as shown in the below figure.



Property	Value
Image ID	
Dimensions	320 x 240
Width	320 pixels
Height	240 pixels

Figure 5 Output cat-image

If you want to maintain the aspect ratio of the image then add one extra component that can either be height or width. Also place -1 to the above script. For example –*"ffmpeg -i cat-image.jpg -vf scale=320:-1 output_cat_image.jpg".*The above script set the aspect ratio of the result image width to 320 pixels. According to the input image aspect ratio the height of the output image is computed. FFmpeg provide variables that can be useful instead of using numbers. These variables can be used to define the output image height and width. Let' take an example that you want to increase the width twice the value of the input image then use the iw(input width) and ih(input height). *"ffmpeg -icat-image.jpg -vf scale=iw*2:ih input_double_width.jpg".*

If an image has low dimension but you want that image to be scaled up. An expression called min can be used to avoid the upscaling. The command will look something like this –

"ffmpeg -icat-imaget.jpg -vf "scale='min(320,iw)':'min(240,ih)'" no_upscaling_input.jpg".

From different sources if you are consolidating material like fitting the input image inside a rectangle of particular size then *force_original_aspect_ratio* can be used to achieved this. This option has two values:

Decrease: If needed then the dimension of the output video can be decreased impulsively.

Increase: If needed then the dimension of the output video can be increased impulsively.

These two values can make the image fit into a 320 * 240 box. For example – *"ffmpeg -I cat-image.jpg -vf scale=w=320:h=240:force_original_aspect_ratio=decrease output_cat_image_320.jpg".*

For resizing the image user can define what algorithm to use together with the *-sws_flags* option. Let's take an example – the default bicubic scaling can be replaced with the bilinear:

"ffmpeg -iexample.tif -vf scale=504:376 -sws_flags bilinear output.bmp".

4. Scopes

FFmpeg a solution for converting and streaming audios and videos is cross platform as their developer have mention about this. Algorithm such as compressing and decompressing envelops in FFmpeg. These can be aggregated and runs on assorted guidance sets. There are many Application-Specific Integrated Circuits (ASICs) available for video and audio compression as well as for decompression. AMD firm has two ACIS Supported by FFmpeg which are UVD and VCE. UVD stands for Unified Video Decoder which is used for decoding purpose. Earlier UVD was known as Universal Video Decoder. VCE stands for Video Code Engine. Its purpose is encoding video. Amlogic firm has one ASIC which is Amlogic Video Engine that is used for decoding purpose but this firm is not supported by FFmpeg. Blackmagic ASIC known as DeckLink encodes and decode the video. Qualcomm Hexagon also supports both encoding and decoding similarly intel too. And at last, Nvidia PureVideo/ NVDEC support decoding and NVENC support encoding. Using *-hwaccel*you can activate the acceleration of decoding of internal hardware. Using a solitary script multiple files can be encoded and packaged with the use of Wildcards. Performance of a device are impacted by the thread commands as well as the quality. The thread commands allow users to control the thread that is used by the FFmpeg. Quality is an important factor of a video. If a video has large volume size, then the video will have better quality. If there are two videos of same codec then the video with larger volume and size and has more bit rate then that video will be better than the other video. FFmpeg can do pretty much everything from changing the bit rate of a video to manipulating the resolution of video. You just need to find the correct script or command for your required video. FFmpeg is best for large collection batch processing but work such as restoration detail oriented digital is not very efficient. A user with little programming experience to no programming experience can make use of FFmpeg. Since editing and processing of video utilizes software and hardware or both of them. As per my research I found that using FFmpeg save your device memory very efficiently.

5. Conclusion

The purpose of this study was to review and analyse different aspect of FFmpeg. From tis study we got to know about different FFmpeglibrabries available for processing and editing images and audios recorded in a video file. FFmpeg can be use with different programming language to create a powerful software. As it helps users to manipulate the video very easily.

I found that while executing some of the FFmpeg commands the results was genereated in an instant because it doesn't re-encode the video files while processing it. Basically, it contains all the commands from basic editing to complex editing. Everything is available in this multimedia framework. You just need to select the right command for manipulating the video. But during the research I found that FFmpeg was not user friendly. As it is completely based on command line utility. But this doesn't cause any major issue because FFmpeg has the documentation of how to use this software. The most important thing about this software is that it doesn't utilize the memory of the device also the results are generated in very less amount of time. making it highly efficient both for memory and time.

References

[1] "FFmpeg," [Online]. Available: https://ffmpeg.org/.

[2] "programmersought," [Online]. Available: https://www.programmersought.com.

[3] Y. Xu and S. Cao, "Design and Implementation of a Multi Video Transcoding Queue Based on MySQL and FFMPEG," *6th IEEE International Conference on Software Engineering and Service Science (ICSESS)*, pp. 629-632, 2015.

[4] R. Safin, E. Garipova, R. Lavrenov, H. Li, M. Svinin and E. Magid, "Hardware and software video encoding comparison," *59th Annual Conference of the Society of Instrument and Control Engineers of Japan (SICE)*, pp. 924-929, 2020.

[5] F. Bai, L. Song, Y. Tao and Y. Liu, "Video Image Restoration Based on H.264 Protocol," *Third International Conference on Instrumentation, Measurement, Computer, Communication and Control*, pp. 500-503, 2013.

[6] L. Xiaohua, J. Xiuhua and W. Caihong, "Design and Implementation of a Real-time Video Stream Analysis System Based on FFMPEG," *2013 Fourth World Congress on Software Engineering*, pp. 212-216, 2013.

[7] H. ZENG, Z. ZHANG and L. SHI, "Research and Implementation of Video Codec Based on FFmpeg,"

International Conference on Network and Information Systems for Computers, pp. 184-188, 2016.

[8] N. Jain, H. Shrivastava and A. A. Moghe, "Production-ready environment for HLS Player using FFmpeg with automation on S3 Bucket using Ansible," *2nd International Conference on Data, Engineering and Applications (IDEA),* 2020.

[9] P. Yang, D. Baracchi, M. Iuliani, D. Shullani, R. Ni, Y. Zhao and A. Piva, "Efficient video integrity analysis through container characterization," *IEEE Journal of Selected Topics in Signal Processing,* pp. 1-8, 2020.

[10] J. Sun, W. Zhang and H.-q. Zhao, "Design and Implementation of Sports Video Player Based on Android," *2014 IEEE/ACIS 13th International Conference on Computer and Information Science (ICIS),* 2014.

[11] M.-H. Chen, F.-C. Chang and H.-M. Hang, "Design and Implementation of an MHP Video and Graphics Subsystem on Xilinx ML310 Platform," *2006 International Conference on Intelligent Information Hiding and Multimedia,* 2006.

[12] C. Ryu, D. Lee, M. Jang, C. Kim and E. Seo, "Extensible Video Processing Framework in Apache Hadoop," *2013 IEEE International Conference on Cloud Computing Technology and Science,* pp. 305-308, 2013.