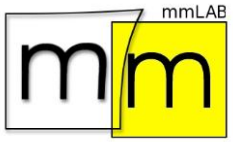
TUTORIAL FFMPEG

Multimedia Signal Processing and Understanding Lab.

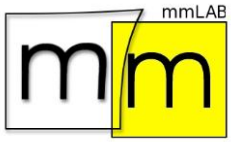
Contributors:

Mattia Daldoss



Content:

Figures:



The FFmpeg suite

FFmpeg is a complete, cross-platform solution to record, convert and stream audio and video. FFmpeg is free software licensed under the LGPL or GPL depending on your choice of configuration options.
The aim of this document is to give you a brief introduction on the FFmpeg libraries. You can find a complete feature list on the official website http://www.ffmpeg.org/documentation.html . FFmpeg has been used in several projects (http://ffmpeg.org/projects.html). We can cite some of the most popular:

- Blender
- Google Chrome
- MPlayer
- VLC

Some multimedia frameworks also use ffmpeg core, like:

- DirectShow, multimedia framework and API produced by Microsoft for software developers to perform various operations with media files or streams.
- Quicktime
- GStreamer, a pipeline-based multimedia framework written in the C programming language with the type system based on GObject
- OpenMAX, (Open Media Acceleration) is a royalty-free, cross-platform set of C-language programming interfaces that provides abstractions for routines especially useful for audio, video, and still images. It's intended for devices that process large amounts of multimedia data in predictable ways.

Implements several protocols:

- IETF standards: TCP, UDP, Gopher, HTTP, RTP, RTSP and SDP
- Apple related protocols: HTTP Live Streaming
- RealMedia related protocols: RealMedia RTSP/RDT
- Adobe related protocols: RTMP, RTMPT (via librtmp), RTMPE (via librtmp), RTMPTE (via librtmp) and RTMPS (via librtmp)
- Microsoft related protocols: MMS over TCP and MMS over HTTP

What's inside

Quoting from the website, the ffmpeg project is made of several components:

- ***ffmpeg*** is a command line tool to convert multimedia files between formats.
- ***ffserver*** is a multimedia streaming server for live broadcasts.
- ***ffplay*** is a simple media player based on SDL and the FFmpeg libraries.
- ***ffprobe*** is a is a simple multimedia stream analyzer.
- ***libavutil*** is a library containing functions for simplifying programming, including random number generators, data structures, mathematics routines, core multimedia utilities, and much more.
- ***libavcodec*** is a library containing decoders and encoders for audio/video codecs.
- ***libavformat*** is a library containing demuxers and muxers for multimedia container formats.
- ***libavdevice*** is a library containing input and output devices for grabbing from and rendering to many common multimedia input/output software frameworks, including Video4Linux, Video4Linux2, VfW, and ALSA.
- ***libavfilter*** is a library containing media filters.

- ***libswscale*** is a library performing highly optimized image scaling and color space/pixel format conversion operations.

Get FFmpeg

In most of the distros, you can directly install ffmpeg from their repositories. But in case you are looking for installing the latest version or want to customize the installation, you might need direct installation from the source code too.
Direct from repositories:

```
sudo apt-get install ffmpeg
```

From source, you have to download first the latest source code from the official website (http://www.ffmpeg.org/), then:

```
tar –xvjf ffmpeg-0.6.90-rc0.tar.bz2
./configure
make
sudo –c 'make install'
```

Basic commands

All the executables given with the ffmpeg suite are command line program, which means that giving them the correct parameters is very important
As a general rule, options are applied to the next specified file. Therefore, order is important, and you can have the same option on the command line multiple times. Each occurrence is then applied to the next input or output file. By default, FFmpeg tries to convert as losslessly as possible: It uses the same audio and video parameters for the outputs as the one specified for the inputs.
All the ff* tools share a common list of options, which can be specified after the executable. The most important are:

- ***`-h, -?, -help, --help'***, Show help.
- ***`-formats'***, Show available formats. The fields preceding the format names have the following meanings:
  - ***`D'***, Decoding available
  - ***`E'***, Encoding available
- ***`-codecs'***, Show available codecs. The fields preceding the codec names have the following meanings:
  - ***`D'***, Decoding available
  - ***`E'***, Encoding available
  - ***`V/A/S'***, Video/audio/subtitle codec
  - ***`S'***, Codec supports slices
  - ***`D'***, Codec supports direct rendering
  - ***`T'***, Codec can handle input truncated at random locations instead of only at frame boundaries

So, for example, if we want to know which codecs are available, we just have to type:

ffmpeg -codecs

We can see the result in **Error! Reference source not found.**.

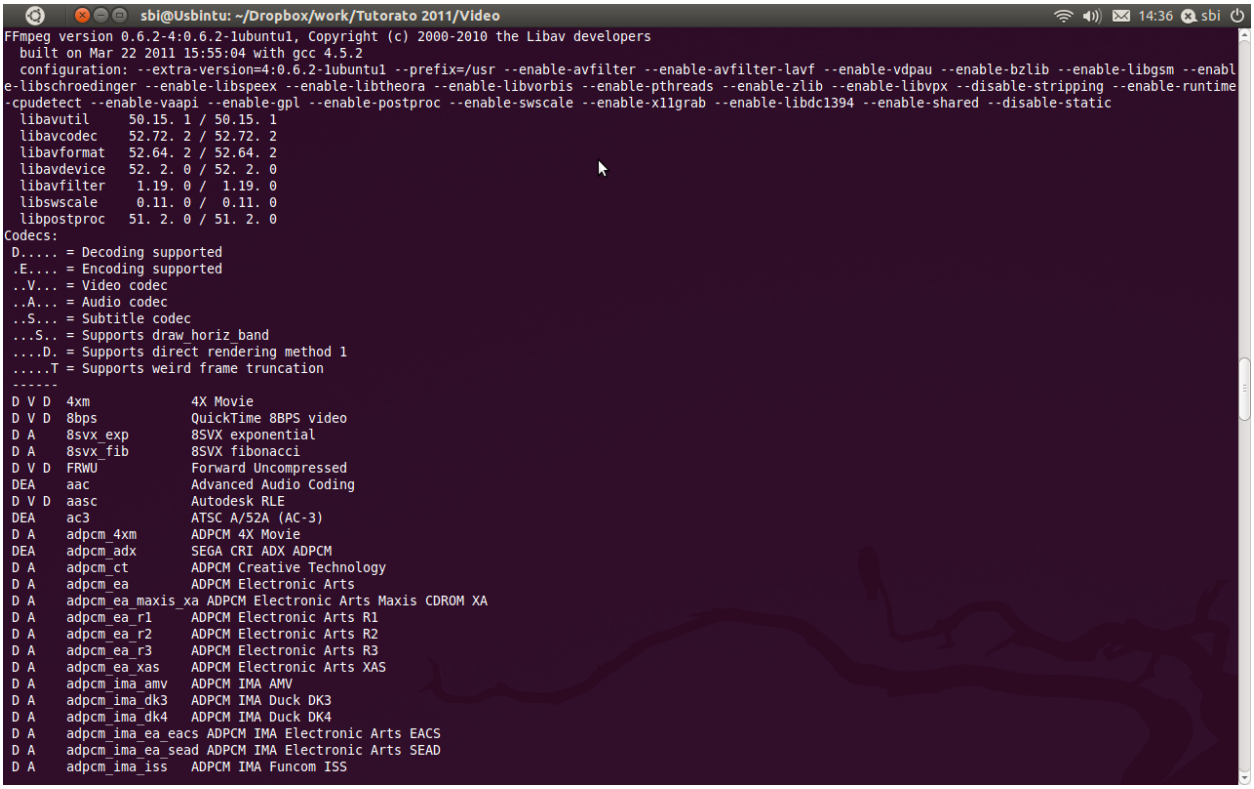


Figure 1 - Output of ffmpeg -codecs command

FFPlay

FFplay is a very simple and portable media player using the FFmpeg libraries and the SDL library. It's a powerful video/audio player, and it is mostly used as a testbed for the various FFmpeg APIs. As we pointed out before, it's a command line operator, which work as follows:

ffplay [options] `input_file'

Besides the common options previously explained, the main specific options are:

- ***`-x width'***, Force displayed width.
- ***`-y height'***, Force displayed height.
- ***`-s size'***, Set frame size (WxH or abbreviation), needed for videos which don't contain a header with the frame size like raw YUV.
- ***`-an'***, Disable audio.
- ***`-vn'***, Disable video.
- ***`-ss pos'***, Seek to a given position in seconds.
- ***`-t duration'***, play <duration> seconds of audio/video
- ***`-bytes'***, Seek by bytes.
- ***`-nodisp'*** ,Disable graphical display.
- ***`-f fmt'***, Force format.

- ***`-stats'***, Show the stream duration, the codec parameters, the current position in the stream and the audio/video synchronisation drift.

While playing, you can operate on the stream as follows:

- ***q***, ESC ,Quit.
- ***f***, Toggle full screen.
- ***p***, SPC, Pause.
- ***a***, Cycle audio channel.
- ***v***, Cycle video channel.
- ***t***, Cycle subtitle channel.
- ***w***, Show audio waves.
- ***left/right***, Seek backward/forward 10 seconds.
- ***down/up***, Seek backward/forward 1 minute.
- ***mouse click***, Seek to percentage in file corresponding to fraction of width.

For further information on FFPlay, and for specific options and operations, you can read the official manual at http://www.ffmpeg.org/ffplay.html.

FFMpeg

The ffmpeg executable is in charge of conversion, with the generic syntax:

ffmpeg [[infile options][`-i' infile]]... {[outfile options] outfile}...

The main options for this operator are, besides the one previously shown in the Basic commands section:

- ***`-f fmt'***, Force format.
- ***`-i filename'***, input file name
- ***`-y'***, Overwrite output files.
- ***`-t duration'***, Restrict the transcoded/captured video sequence to the duration specified in seconds. hh:mm:ss[.xxx] syntax is also supported.
- ***`-fs limit_size'***, Set the file size limit.
- ***`-ss position'***, Seek to given time position in seconds. hh:mm:ss[.xxx] syntax is also supported.
- ***`-itsoffset offset'***, Set the input time offset in seconds. [-]hh:mm:ss[.xxx] syntax is also supported. This option affects all the input files that follow it. The offset is added to the timestamps of the input files. Specifying a positive offset means that the corresponding streams are delayed by 'offset' seconds.
- ***`-target type'***, Specify target file type ("vcd", "svcd", "dvd", "dv", "dv50", "pal-vcd", "ntsc-svcd", ...). All the format options (bitrate, codecs, buffer sizes) are then set automatically. You can just type:

  ffmpeg -i myfile.avi -target vcd /tmp/vcd.mpg

  Nevertheless you can specify additional options as long as you know they do not conflict with the standard, as in:

  ffmpeg -i myfile.avi -target vcd -bf 2 /tmp/vcd.mpg

Video options

- ***`-b bitrate'***, Set the video bitrate in bit/s (default = 200 kb/s).

- ***`-vframes number'***, Set the number of video frames to record.
- ***`-r fps'***, Set frame rate (Hz value, fraction or abbreviation), (default = 25).
- ***`-s size'***, Set frame size. The format is `wxh' (ffserver default = 160x128). There is no default for input streams, for output streams it is set by default to the size of the source stream. If the input file has video streams with different resolutions, the behaviour is undefined. The following abbreviations are recognized:
  - `sqcif', 128x96
  - `qcif', 176x144
  - `cif', 352x288
  - `4cif', 704x576
  - `16cif', 1408x1152
  - `qqvga', 160x120
  - `qvga', 320x240
  - `vga', 640x480
  - `svga', 800x600
  - `xga', 1024x768
  - `hd480', 852x480
  - `hd720', 1280x720
  - `hd1080', 1920x1080

- ***`-aspect aspect'***, Set the video display aspect ratio specified by aspect. aspect can be a floating point number string, or a string of the form num:den, where num and den are the numerator and denominator of the aspect ratio. For example "4:3", "16:9", "1.3333", and "1.7777" are valid argument values.
- ***`-vn'***, Disable video recording.
- ***`-maxrate bitrate'***, Set max video bitrate (in bit/s). Requires -bufsize to be set.
- ***`-minrate bitrate'***, Set min video bitrate (in bit/s). Most useful in setting up a CBR encode:

```
ffmpeg -i myfile.avi -b 4000k -minrate 4000k -maxrate 4000k -bufsize 1835k out.m2v
```

- ***`-bufsize size'***, Set video buffer verifier buffer size (in bits).
- ***`-vcodec codec'***, Force video codec to codec. Use the copy special value to tell that the raw codec data must be copied as is.
- ***`-sameq'***, Use same quantizer as source (implies VBR).
- ***`-pass n'***, Select the pass number (1 or 2). It is used to do two-pass video encoding. The statistics of the video are recorded in the first pass into a log file (see also the option -passlogfile), and in the second pass that log file is used to generate the video at the exact requested bitrate. On pass 1, you may just deactivate audio and set output to null, examples for Windows and Unix:

```
ffmpeg -i foo.mov -vcodec libxvid -pass 1 -an -f rawvideo -y NUL

ffmpeg -i foo.mov -vcodec libxvid -pass 1 -an -f rawvideo -y /dev/null
```

Audio options

- ***`-aframes number'***, Set the number of audio frames to record.
- ***`-ar freq'***, Set the audio sampling frequency. For input streams it is set by default to 44100 Hz, for output streams it is set by default to the frequency of the input stream. If

the input file has audio streams with different frequencies, the behaviour is undefined.

- ***`-ab bitrate'***, Set the audio bitrate in bit/s (default = 64k).
- ***`-aq q'***, Set the audio quality (codec-specific, VBR).
- ***`-ac channels'***, Set the number of audio channels. For input streams it is set by default to 1, for output streams it is set by default to the same number of audio channels in input. If the input file has audio streams with different channel count, the behaviour is undefined.
- ***`-an'***, Disable audio recording.
- ***`-acodec codec'***, Force audio codec to codec. Use the copy special value to specify that the raw codec data must be copied as is.

Typical tasks and useful examples

Now that we listed all the major options for ffplay and ffmpeg, let's see how we can perform some useful tasks.

Obtain info about a video

ffmpeg –i video.avi

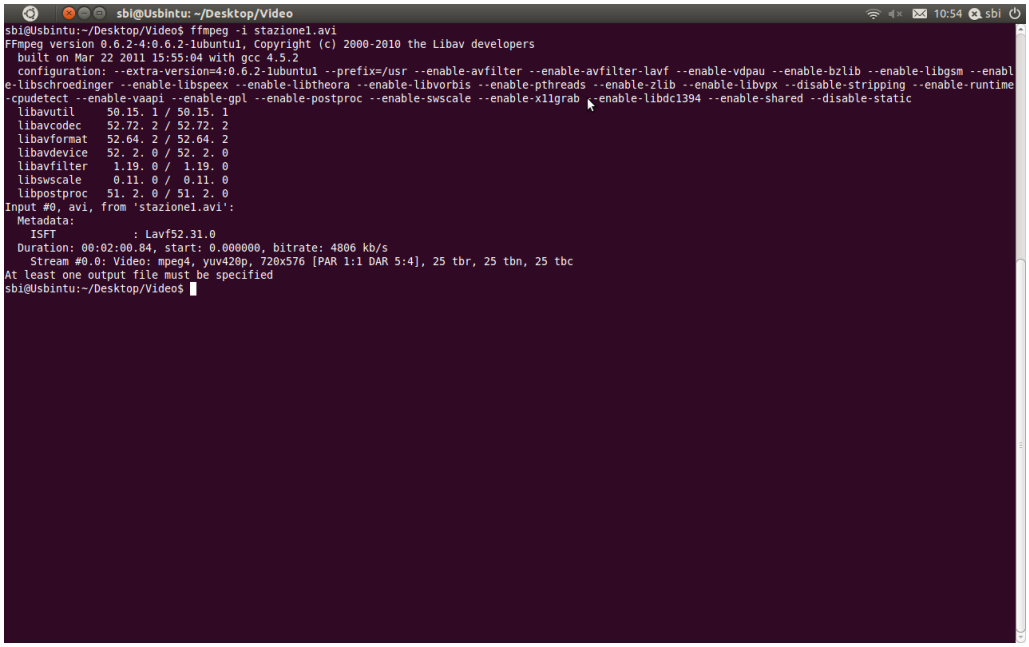


Figure 2 - Obtain information from a multimedia file

Play YUV file

Since it's a raw format, no info are is present in the header so we have to specify the dimension of the video

ffplay -s cif video_cif.yuv

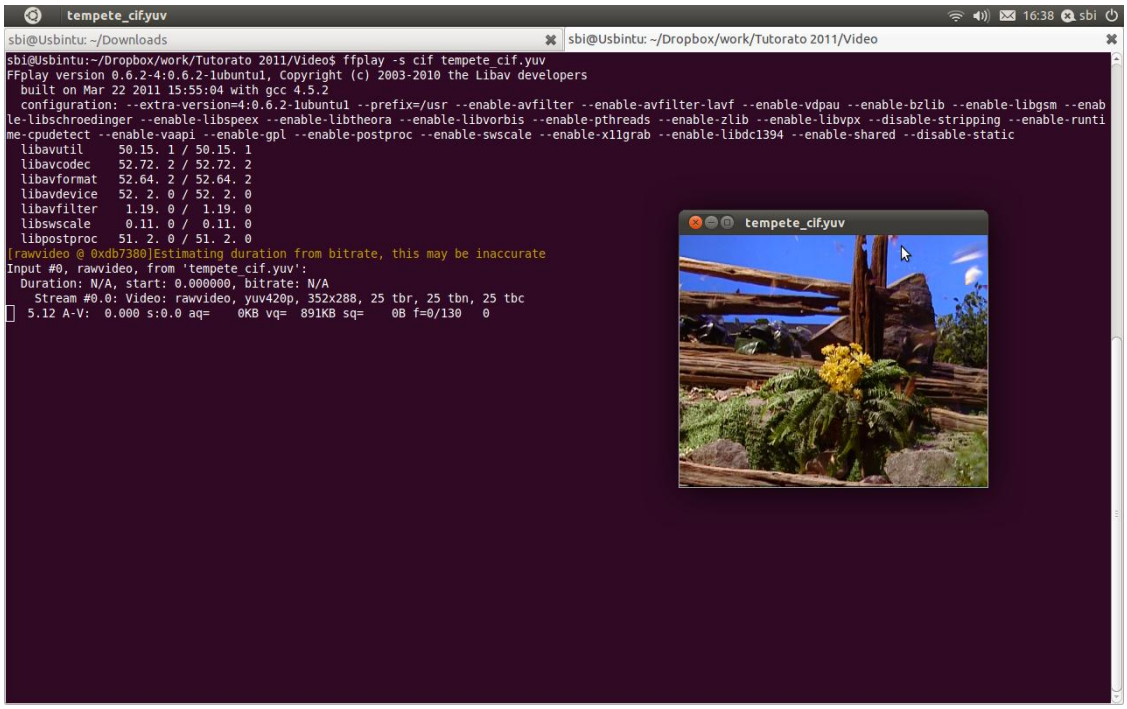


Figure 3 - Play a YUV file

Play an encoded video

Similarly, an encoded stream can be played through:

ffplay video.avi

Grab video from a camera

ffmpeg –f video4linux2 –i /dev/video0 video.mpg

Video4Linux or V4L is a video capture application programming interface for Linux. Several USB webcams, TV tuners, and other devices are supported. Video4Linux is closely integrated with the Linux kernel. It's currently in its second version. Pay attention to the ID of the webcam to attach to, typically '*video0'*.

Grab video from the desktop

ffmpeg -f x11grab -s cif -r 25 -i :0.0+10,20 out.mpg

Force video format to x11grab (grab from linux desktop), and specify:

- Dimension of the video (-s)
- Fps (-r)
- The desktop to capture from (:0.0)
- The offset of the upper left corner of the desired rectangle (+10 and +20 px)
- Output file

Convert a video in a sequence of images

ffmpeg -i input.avi -an -r 1 -y -s 320x240 video%d.jpg

It creates a sequence of screenshot in jpeg (-r 1, so every second), overwriting output files when necessary (-y), with size 320x240. The images are progressively named as video<framenumber>.jpg .

Some simple conversions

Avi to mpg:

ffmpeg –i input.avi output.mpg

Avi to flv conversion:

ffmpeg -i input.avi -sameq output.flv

Flv to mpg conversion:

ffmpeg -i input.flv -sameq output.mpg

Notes:
Flash Video is a container file format used to deliver video over the Internet using Adobe Flash Player versions 6–10. Flash Video content may also be embedded within SWF files. There are two different video file formats known as Flash Video: ***FLV*** and ***F4V***.
Audio Video Interleave (also Audio Video Interleaved), known by its acronym ***AVI***, is a multimedia container format introduced by Microsoft in November 1992 as part of its Video for Windows technology. AVI files can contain both audio and video data in a file container that allows synchronous audio-with-video playback.
Program stream (***PS*** or ***MPEG-PS***) is a container format for multiplexing digital audio, video and more. The PS format is specified in MPEG-1 Part 1 (ISO/IEC 11172-1) and MPEG-2 Part 1, Systems (ISO/IEC standard 13818-1[6]/ITU-T H.222.0[4][5]).

Cropping a Video

Removing part of the input video

ffmpeg -i inputfile.avi -croptop 88 -cropbottom 88 -cropleft 360 -cropright 360 outputfile.avi

Padding a Video

When you want to add color stripes (typically black) on the video (think about the black portion of a 16:9 DVD video)

ffmpeg -i input.avi -padtop 120 -padbottom 120 -padcolor 000000 output.avi

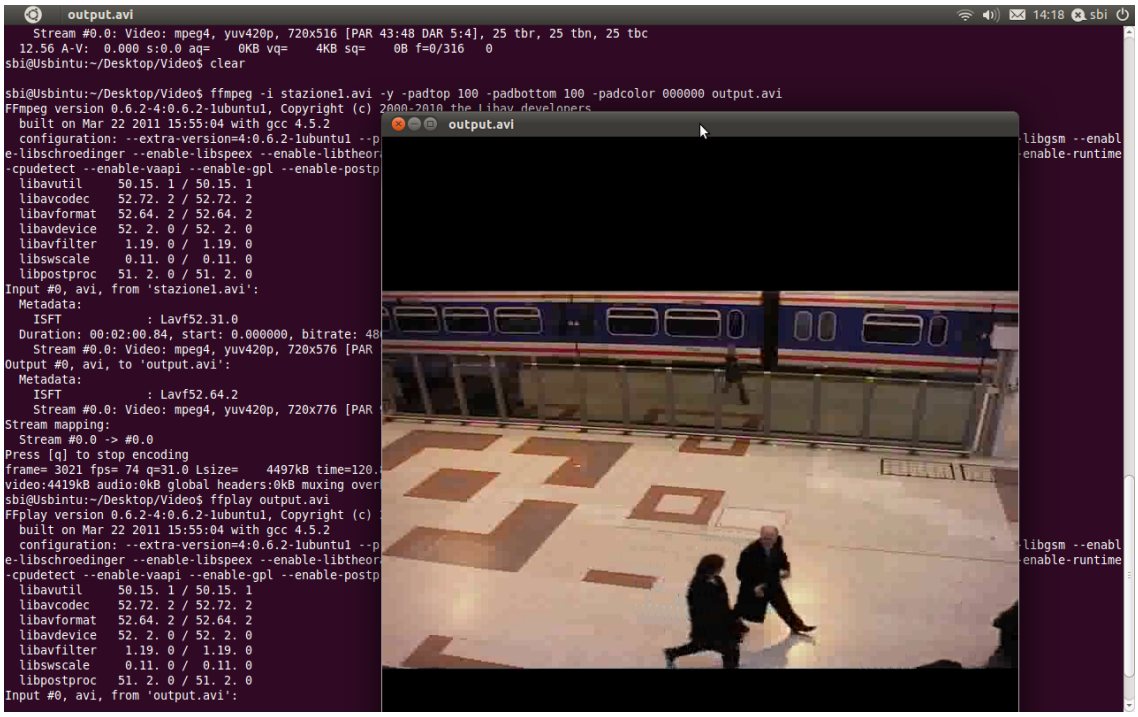


Figure 4 - Padded Video