Comparison of software and hardware video codecs from perspective of power consumption

JAROSLAV SVOBODA MICHEL MUFFEI

svoboda | mmuffei @kth.se

13th October 2016

Abstract

Your abstract here.

Contents

Acronyms

MS-SSIM Multi-Scale Structural Similarity.. 2

PSNR-HVS-M Peak Signal-to-Noise Ratio taking into account Contrast Sensitivity Function (CSF) and between-coefficient contrast masking of DCT basis functions.. 2

VMAF Video Multi-Method Assessment Fusion. 2

VQMT Video Quality Measurement Tool. 2

1 Introduction

Video encoding and decoding are processes with many variables which can influence the output of whole process of video transfer. Visual quality of video is determined by chosen coding standard, its implementation and encoding settings. All these three key elements have direct impact on energy resources we need for completing encode. Video coding standard defines complexity of algorithm and usually the more effective compression the more complex algorithm - the more power demanding. There are many types of implementations but usually the more hardwired algorithms it uses the less power demanding it is. At last, used encoding settings determine time needed for compression. That also means power necessary for encode. From this point of view, power consumption one, it is interesting to create comparison of different video codecs to see how much quality of video costs in used energy.

1.1 Theoretical framework/literature study

PSNR-HVS-MMS-SSIM

We had to compile FFmpeg with support of NVENC and QSV.[1, 2]

1.2 Research questions, hypotheses

xxxxx xxxx xxxx

2 Method

We choose three test sequences, each 10 s long. More in table 1

Table 1: Parameters of test sequences

Sequence	crowd_run_2160p50.y4m	old_town_cross_2160p50.y4m	sintel.y4m
Resolution	3840 × 2160	3840 × 2160	4096 × 1744
framerate	50p	50p	24p
# of frames	500	500	240
subsampling	4 : 2 : 0	4 : 2 : 0	4 : 2 : 0
size in bytes	6220803036	6220803036	

Whole process was done for all codecs as follows:

1. Power measuring tools NVIDIA System Management Interface and Intel Power Gadget are enabled
2. Encoding proceeds
3. Power measuring tools are disabled
4. Encoded video is trans-coded to YUV420P
5. Quality is measured by VMAF and VQMT

This is done for all three chosen sequences, all chosen codecs and all presets available in bit-rates from 500 kbit/s to 15 000 kbit/s with 500 kbit steps. Total number of encodes is xx. Information about used software are in table 2.

3 Results and Analysis

xxxxx xxxx xxxx

Table 2: Used software

Name	Version
Ubuntu GNOME	
FFmpeg	
x264	
x265	
OpenH264	
libtheora	
libvpx	
NVIDIA System Management Interface	
Intel Power Gadget	
VMAF Development Kit	
VQMT	

4 Discussion

xxxxx xxxx xxxx

References

[1] Intel Corporation. *Intel QuickSync Video and FFmpeg. Installation and Validation*. 24th Dec. 2015. URL: http://www.intel.ie/content/dam/www/public/emea/xe/en/documents/white-papers/quicksync-video-ffmpeg-install-valid.pdf (visited on 13/10/2016).

[2] NVIDIA Corporation. *FFMPEG WITH NVIDIA ACCELERATION ON UBUNTU LINUX. Installation and User Guide*. 9th Oct. 2015. URL: http://developer.download.nvidia.com/compute/redist/ffmpeg/1511-patch/FFMPEG-with-NVIDIA-Acceleration-on-Ubuntu_UG_v01.pdf (visited on 13/10/2016).

A Annex

```
sudo apt install cmake mercurial autoconf automake build-essential
   libass-dev libfreetype6-dev libsdl1.2-dev libtheora-dev libtool
   libva-dev libvdpau-dev libvorbis-dev libxcb1-dev libxcb-shm0-dev
   libxcb-xfixes0-dev pkg-config texinfo zlib1g-dev nasm libfdk-aac-dev
    libmp3lame-dev libopus-dev git yasm unzip wget sysstat libxvidcore-
   dev libfaac-dev libopencore-amrnb-dev libopencore-amrwb-dev libgsm1-
   dev zlib1g-dev libgpac1-dev
mkdir ~/ffmpeg_sources
cd ~/ffmpeg_sources
git clone https://github.com/cisco/openh264
cd openh264
make ARCH=x86_64 && sudo make install
cd ~/ffmpeg_sources
wget http://www.tortall.net/projects/yasm/releases/yasm-1.3.0.tar.gz
tar xzvf yasm-1.3.0.tar.gz
cd yasm-1.3.0
./configure --prefix=”$HOME/ffmpeg_build” --bindir=”$HOME/bin”
make
```

```
make install
make distclean
cd ~/ffmpeg_sources
wget http://download.videolan.org/pub/x264/snapshots/last_x264.tar.bz2
tar xjvf last_x264.tar.bz2
cd x264-snapshot*
PATH="$HOME/bin:$PATH" ./configure --prefix="$HOME/ffmpeg_build" --
   bindir="$HOME/bin" --enable-static
PATH="$HOME/bin:$PATH" make
make install
make distclean
cd ~/ffmpeg_sources
hg clone https://bitbucket.org/multicoreware/x265
cd ~/ffmpeg_sources/x265/build/linux
PATH="$HOME/bin:$PATH" cmake -G "Unix Makefiles" -DCMAKE_INSTALL_PREFIX
   ="$HOME/ffmpeg_build" -DENABLE_SHARED:bool=off ../../source
make
make install
make distclean
cd ~/ffmpeg_sources
git clone https://chromium.googlesource.com/webm/libvpx
cd libvpx*
PATH="$HOME/bin:$PATH" ./configure --prefix="$HOME/ffmpeg_build" --
   disable-examples --disable-unit-tests --enable-vp8 --enable-vp9
PATH="$HOME/bin:$PATH" make
make install
make clean
cd ~/ffmpeg_sources
wget http://ffmpeg.org/releases/ffmpeg-snapshot.tar.bz2
tar xjvf ffmpeg-snapshot.tar.bz2
cd ffmpeg
PATH="$HOME/bin:$PATH" PKG_CONFIG_PATH="$HOME/ffmpeg_build/lib/
   pkgconfig" ./configure \
  --prefix="$HOME/ffmpeg_build" \
  --pkg-config-flags="--static" \
  --extra-cflags="-I$HOME/ffmpeg_build/include" \
  --extra-ldflags="-L$HOME/ffmpeg_build/lib" \
  --bindir="$HOME/bin" \
  --enable-gpl \
  --enable-libass \
  --enable-libfdk-aac \
  --enable-libfreetype \
  --enable-libmp3lame \
  --enable-libopus \
  --enable-libtheora \
  --enable-libvorbis \
  --enable-libvpx \
  --enable-libx264 \
  --enable-libx265 \
  --enable-opencl \
  --enable-nvenc \
  --enable-nvresize \
  --extra-cflags=-I../cudautils \
  --extra-ldflags=-L../cudautils \
  --enable-libmfx \
  --enable-libxvid \
  --enable-libopenh264 \
  --enable-libgsm \
  --enable-libopencore-amrnb \
  --enable-nonfree
PATH="$HOME/bin:$PATH" make
make install
make distclean
hash -r
```