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

JAROSLAV SVOBODA MICHEL MUFFEI
svoboda | mmuffei @kth.se

28th November 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	500
subsampling	4 : 2 : 0	4 : 2 : 0	4 : 2 : 0
size in bytes	6 220 803 036	6 220 803 036	5 357 571 060

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.

Table 2: Used software

Name	Version
Ubuntu GNOME	16.04.1 LTS
FFmpeg	
x264	
x265	
OpenH264	
libtheora	
libvpx	
NVIDIA SMI	
Intel Power Gadget	
VMAF Development Kit	
VQMT	

Table 3: Used hardware

Part	Name
CPU	Intel Core i5-4570@3.2 GHz
RAM	DDR3 32 GB
GPU	Nvidia 960 GTX 4 GB
SSD	Samsung EVO 850 250 GB

3 Results and Analysis

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

Compile script

```
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
```

Encoding script

```
#! /bin/bash

video=("old_town_cross_2160p50" "crowd_run_2160p50" "sintel")

function press_enter
{
        echo ""
        echo -n "Press Enter to continue"
        read
        clear
}

function evaluate_x264
{
        preset=(ultrafast superfast veryfast faster fast medium slow
           slower veryslow placebo)
        bitrate=(500 1000 1500 2000 2500 3000 3500 4000 4500 5000 6000
           7000 8000 9000 10000 11000 12000 13000 14000 15000)

#       if [ -d Output/x264 ]; then
#               echo "x264 folder already exists, check for results"
#               return
#       elif [ ! -d Output/x264 ]; then
#               mkdir Output/x264
#       fi
```

```
#       if [ ! -d Output/x264/encoded ]; then
#               mkdir Output/x264/encoded
#       fi

#       if [ ! -d Output/x264/transcoded ]; then
#               mkdir Output/x264/transcoded
#       fi

#       if [ ! -d Output/x264/results ]; then
#               mkdir Output/x264/results
#       fi

#       if [ ! -d Output/x264/results/powergadget ]; then
#               mkdir Output/x264/results/powergadget
#       fi

#       if [ ! -d Output/x264/results/nvidiasmi ]; then
#               mkdir Output/x264/results/nvidiasmi
#       fi

#       if [ ! -d Output/x264/results/vqmt ]; then
#               mkdir Output/x264/results/vqmt
#       fi

#       if [ ! -d Output/x264/results/vmaf ]; then
#               mkdir Output/x264/results/vmaf
#       fi

        height=
        width=
        for v in "${video[@]}"; do
                for p in "${preset[@]}"; do
                        for b in "${bitrate[@]}"; do
                                echo -e "\e[92mStarting power
                                   consumption logging\e[0m"
                                echo "modprobe msr"
                                echo "modprobe cpuid"
                                echo "Tools/power_gadget &> Output/
                                   x264/results/powergadget/$v$p$b.csv"
                                echo "nvidia-smi -i 0 -l 1 --query-gpu=
                                   timestamp,pstate,temperature.gpu,
                                   utilization.gpu,memory.used,clocks.
                                   current.video,clocks.current.
                                   graphics,clocks.current.sm,fan.speed
                                   ,power.draw --format=csv -f Output/
                                   x264/results/nvidiasmi/$v$p$b.csv"
                                echo -e "\e[92mStarting Encoding\e[0m"
                                echo "ffmpeg -benchmark -y -i Input/$v.
                                   yuv -c:v libx264 -preset $p -b:v $b
                                   -an Output/x264/encoded/$v$p$b"
                                echo -e "\e[32mDone encoding\e[0m"
```

```
                                        echo -e "\e[32mStarting Transcoding\e[0
                                           m"
                                        echo "ffmpeg -i Output/x264/encoded/
                                           $v$p$b.mkv -c:v rawvideo -pix_fmt
                                           yuv420p Output/x264/transcoded/
                                           $v$p$b.yuv"
                                        echo -e "\e[32mDone transcoding\e[0m"
                                        echo -e "\e[32mStarting evaluation with
                                            VQMT and VMAF\e[0m"
                                        if [ $v=${video[0]} ] || [ $v=${video
                                           [1]} ]; then
                                                height=2160
                                                width=3840
                                        else
                                                height=1744
                                                width=4096
                                        fi
                                        echo $v
                                        echo "Tools/vqmt Input/$v.y4m Output/
                                           x264/transcoded/$v$p$b height width
                                           500 1 Output/x264/results/vqmt/
                                           $v$p$p PSNRHVSM MSSSIM"
                                        echo "Tools/run_vmaf yuv420p width
                                           height Input/$v.y4m Output/x264/
                                           transcoded/$v$p$b --out-fmt text >
                                           Output/x264/results/vmaf/$v$p$b"
                                        echo -e "\e[32mDone evaluating with
                                           VQMT and VMAF\e[0m"
                                        echo "rm Output/x264/encoded/$v$p$b"
                                        echo "rm Output/x264/transcoded/$v$p$b"
                                done
                        done
                done

}

function evaluate_test
{
        bitrate=(500k 1000k 1500k 2000k 2500k 3000k 3500k 4000k 4500k
           5000k 6000k 7000k 8000k 9000k 10000k 11000k 12000k 13000k
           14000k 15000k)

        if [ -d Output/test ]; then
                echo "test folder already exists, check for results"
                return
        elif [ ! -d Output/test ]; then
                mkdir Output/test
        fi

        if [ ! -d Output/test/encoded ]; then
                mkdir Output/test/encoded
        fi
```

```
if [ ! -d Output/test/transcoded ]; then
        mkdir Output/test/transcoded
fi

if [ ! -d Output/test/results ]; then
        mkdir Output/test/results
fi

if [ ! -d Output/test/results/powergadget ]; then
        mkdir Output/test/results/powergadget
fi

if [ ! -d Output/test/results/ffmpeg ]; then
        mkdir Output/test/results/ffmpeg
fi

if [ ! -d Output/test/results/nvidiasmi ]; then
        mkdir Output/test/results/nvidiasmi
fi

if [ ! -d Output/test/results/vqmt ]; then
        mkdir Output/test/results/vqmt
fi

if [ ! -d Output/test/results/vmaf ]; then
        mkdir Output/test/results/vmaf
fi

height=
width=
for v in "${video[@]}"; do

                for b in "${bitrate[@]}"; do
                        echo -e "\e[92mStarting power
                           consumption logging\e[0m"
                        modprobe msr
                        modprobe cpuid
                        Tools/power_gadget/power_gadget -e 1000
                            > Output/test/results/powergadget/
                           $v$b.csv &
                        nvidia-smi -i 0 -l 1 --query-gpu=
                           timestamp,pstate,temperature.gpu,
                           utilization.gpu,memory.used,clocks.
                           current.video,clocks.current.
                           graphics,clocks.current.sm,fan.speed
                           ,power.draw --format=csv -f Output/
                           test/results/nvidiasmi/$v$b.csv &
                        echo -e "\e[92mStarting Encoding\e[0m"
                        FFREPORT=file=Output/test/results/
                           ffmpeg/$v$b.log:level=32 Tools/
                           ffmpeg/ffmpeg -benchmark -y -i Input
```

```
                                        /y4m/$v.y4m -c:v libtheora -b:v $b -
                                        an Output/test/encoded/$v$b.mkv
                                    echo -e "\e[93mDone with encoding\e[0m"
                                    pkill -f power_gadget
                                    pkill -f nvidia-smi
                                    echo -e "\e[93mDone with power
                                        consumption logging\e[0m"
                                    echo -e "\e[92mStarting Transcoding\e[0
                                        m"
                                    FFREPORT=file=Output/test/results/
                                        ffmpeg/T$v$b.log:level=32 Tools/
                                        ffmpeg/ffmpeg -i Output/test/encoded
                                        /$v$b.mkv -c:v rawvideo -pix_fmt
                                        yuv420p Output/test/transcoded/$v$b.
                                        yuv
                                    echo -e "\e[93mDone with transcoding\e
                                        [0m"
                                    echo -e "\e[92mStarting evaluation with
                                         VQMT and VMAF\e[0m"
                                    if [ "$v" == "${video[0]}" ] || [ "$v"
                                        == "${video[1]}" ]; then
                                            height=2160
                                            width=3840
                                    elif [ "$v" == "${video[2]}" ]; then
                                            height=1744
                                            width=4096
                                    fi
                                    Tools/vqmt/vqmt Input/yuv/$v.yuv Output
                                        /test/transcoded/$v$b.yuv $height
                                        $width 500 1 Output/test/results/
                                        vqmt/$v$b PSNRHVSM MSSSIM &
                                    Tools/vmaf/run_vmaf yuv420p $width
                                        $height Input/yuv/$v.yuv Output/test
                                        /transcoded/$v$b.yuv --out-fmt text
                                        > Output/test/results/vmaf/$v$b &
                                    wait ${!}
                                    echo -e "\e[93mDone with evaluating
                                        with VQMT and VMAF\e[0m"
                                    rm Output/test/encoded/$v$b.mkv
                                    rm Output/test/transcoded/$v$b.yuv
                            done
        chmod -R 777 Output/test

        done

}


selection=

until [ "$selection" = "0" ]; do
        echo ""
```

```
        echo "SELECT AN ENCODER"
        echo "1 - x264"
        echo "2 - x265"
        echo "3 - NVENC h264"
        echo "4 - NVENC h265"
        echo "5 - QSV h264"
        echo ""
        echo "6 - Test"
        echo "0 - Exit"
        echo ""
        echo -n "Enter selection"
        read selection
        case $selection in
                1 ) evaluate_x264; press_enter ;;
                2 ) echo "evaluate_x265"; press_enter;;
                3 ) echo "evaluate_NVENCh264"; press_enter;;
                4 ) echo "evaluate_NVENCh265"; press_enter;;
                5 ) echo "evaluate_QSVh264"; press_enter;;
                6 ) evaluate_test; press_enter;;
                0 ) exit;;
                * ) echo "Selection not valid"; press_enter;
        esac
done
```