Mkvfix Guide

Author: Kim Miikki
Date: 19.9.2021

1 Introduction

A video created with *gtlvideo.py* may not be playable on a Raspberry Pi, if the program version is less than 1.2. The mkv video can be fixed by running this program which creates a playable x264 codec video file. It can then be converted to mp4 video container by running *vid2mp4.py*.

2 System Requirements

Operating System: ALL

Program: ffmpeg

Python 3

3 Program Usage

This program requires only one argument which is the name of the mkv video file in current directory.

```
$ mkvfix.py -h
usage: mkvfix.py [-h] file

positional arguments:
  file        mkv video file name

optional arguments:
  -h, --help  show this help message and exit
```

4 Use Case

A time-lapse video was created with *gtlvideo.py* version 1.1, and to following method shows how to make it playable on a Raspberry Pi.

```
$ mkvfix.py video-rgb.mkv
Fix mkv playback issue on Raspberry Pi

Current directory:
/home/pi/python/20210919-gtlvideo_vid2mp4

ffmpeg version 4.1.6-1~deb10u1+rpt2 Copyright (c) 2000-2020 the FFmpeg
developers
...

frame=  860 fps=9.0 q=28.0 size=    2816kB time=00:00:32.32 bitrate=
713.8kbits/frame=  861 fps=8.8 q=-1.0 Lsize=    2957kB time=00:00:34.36 bitrate=
704.9kbits/s speed=0.35x
video:2957kB audio:0kB subtitle:0kB other streams:0kB global headers:0kB muxing
overhead: 0.000000%
...
[libx264 @ 0x1edb560] kb/s:703.27
```

The fixed video is stored in file with same stem as the original video, but now with h264 extension. Video properties of the new file can examined with mediainfo:

```
$ mediainfo video-rgb.h264
General
Complete name                            : video-rgb.h264
Format                                   : AVC
Format/Info                              : Advanced Video Codec
File size                                : 2.89 MiB
Writing library                          : x264 core 155 r2917 0a84d98
Encoding settings                        : cabac=1 / ref=3 / deblock=1:0:0 /
analyse=0x3:0x113 / me=hex / subme=7 / psy=1 / psy_rd=1.00:0.00 / mixed_ref=1 /
me_range=16 / chroma_me=1 / trellis=1 / 8x8dct=1 / cqm=0 / deadzone=21,11 /
fast_pskip=1 / chroma_qp_offset=-2 / threads=6 / lookahead_threads=1 /
sliced_threads=0 / nr=0 / decimate=1 / interlaced=0 / bluray_compat=0 /
constrained_intra=0 / bframes=3 / b_pyramid=2 / b_adapt=1 / b_bias=0 /
direct=1 / weightb=1 / open_gop=0 / weightp=2 / keyint=250 / keyint_min=25 /
scenecut=40 / intra_refresh=0 / rc_lookahead=40 / rc=crf / mbtree=1 / crf=23.0 /
qcomp=0.60 / qpmin=0 / qpmax=69 / qpstep=4 / ip_ratio=1.40 / aq=1:1.00

Video
Format                                   : AVC
Format/Info                              : Advanced Video Codec
Format profile                           : High@L4
Format settings                          : CABAC / 4 Ref Frames
Format settings, CABAC                   : Yes
Format settings, ReFrames                : 4 frames
Width                                    : 1 704 pixels
Height                                   : 1 068 pixels
Display aspect ratio                     : 16:10
Frame rate mode                          : Variable
Color space                              : YUV
Chroma subsampling                       : 4:2:0
Bit depth                                : 8 bits
Scan type                                : Progressive
Writing library                          : x264 core 155 r2917 0a84d98
Encoding settings                        : cabac=1 / ref=3 / deblock=1:0:0 /
analyse=0x3:0x113 / me=hex / subme=7 / psy=1 / psy_rd=1.00:0.00 / mixed_ref=1 /
me_range=16 / chroma_me=1 / trellis=1 / 8x8dct=1 / cqm=0 / deadzone=21,11 /
fast_pskip=1 / chroma_qp_offset=-2 / threads=6 / lookahead_threads=1 /
sliced_threads=0 / nr=0 / decimate=1 / interlaced=0 / bluray_compat=0 /
constrained_intra=0 / bframes=3 / b_pyramid=2 / b_adapt=1 / b_bias=0 /
direct=1 / weightb=1 / open_gop=0 / weightp=2 / keyint=250 / keyint_min=25 /
scenecut=40 / intra_refresh=0 / rc_lookahead=40 / rc=crf / mbtree=1 / crf=23.0 /
qcomp=0.60 / qpmin=0 / qpmax=69 / qpstep=4 / ip_ratio=1.40 / aq=1:1.00
```

The h264 video can be played back with omxplayer (Raspberry Pi) or VLC. If a mp4 format container is required, it can be produced with *vid2mp4.py*:

```
$ vid2mp4.py video-rgb.h264
Convert video to mp4 format

Current directory:
/home/pi/python/20210919-gtlvideo_vid2mp4

AVC-H264 import - frame size 1704 x 1068 at 25.000 FPS
AVC Import results: 861 samples - Slices: 4 I 220 P 637 B - 1 SEI - 4 IDR
Stream uses forward prediction - stream CTS offset: 2 frames
Saving video-rgb.mp4: 0.500 secs Interleaving
```