FFmpeg Aufbaukurs

Reto Kromer, Eléonore Bernard und Kamilla Ødegård
AV Preservation by reto.ch und Ødegård & Bernard Restaurierung
Bern, 13. März 2025

Einführung in den Kursablauf

- Ziele des Kurses
- Vorstellen der mitgebrachten Problemstellungen
- Kurze Repetition zu FFmpeg
- Input zu Workflow, Qualitätskontrolle und Konservierung
- Gruppenarbeit
- Erkenntnisse und Lösungen präsentieren
- FFmpeg in anderen Tools nutzen
- Schlussbesprechung

Ziele des Kurses

- Anwenden von FFmpeg für verschiedene Ton- und Bildumwandlungen
- Ressourcen und Hilfsmittel zu FFmpeg nutzen und geeignete Befehle finden
- Anwenden von FFmpeg und weiteren Programmen zur Extraktion und Interpretation von Metadaten
- Möglichkeiten der Qualitätskontrolle
- Nutzen von FFmpeg Befehlen für Stapelverarbeitung

Link zum FFmpeg-Aufbaukurs (D)
https://reto.ch/training/2025/2025-03-13/

Lien vers le cours avancé de FFmpeg (F)
https://reto.ch/training/2025/2025-03-14/

Anwendungsbeispiele

Bern, 13. März 2025

Beispiele

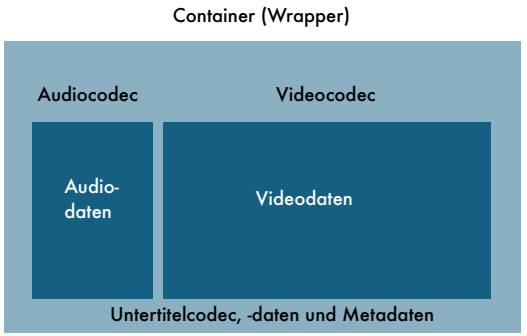
- Dateien aus DVD „rippen“ und in Sichtungskopien umwandeln
- Umwandlung von DPX-Einzelbilder in Matroska/FFV1-Bildstrom
- Umwandlung von DPX-Einzelbilder mittels RAWcooked
- Uncompressed/Archivmaster in H.264/Sichtungskopie umwandeln
- ProRes in Sichtungskopie H.264 umwandeln
- Wasserzeichen, Timecode, Logo oder Vorspann hinzufügen

Zur Erinnerung

Bern, 13. März 2025

Dateiaufbau

- Container (Wrapper)
- Audiocodec
- Audiodaten
- Videocodec
- Videodaten



Die FFmpeg Familie

Programme
- ffmpeg
- ffplay
- ffprobe

Bibliotheken
- libavutil
- libavcodec
- libavformat
- libavdevice
- libavfilter
- libswscale
- libswresample
- libpostrpoc



FFmpeg-Befehlsstruktur

$0	**$1**		**${n}**
command	*argument_1*	*...*	*argument_n*

FFmpeg-Syntax der Argumente:

```
-parameter
-parameter value
-p
-p value
```


FFmpeg-Syntax

```
ffmpeg
  [global_options]
  [input_options_n] -i input_file_n
  [output_options_n] output_file_n

ffprobe [input_options] input_file

ffplay [input_options] input_file
```

Ressourcen, Hilfestellungen

FFmpeg Cookbook for Archivists
→ avpres.net/FFmpeg/

ffmprovisr
→ amiaopensource.github.io/ffmprovisr/

ffmpeg -h
ffmpeg -codecs

Input zu Workflow

Bern, 13. März 2025

Möglicher Workflow

- Die akzeptierten Dateiformate definieren
- Qualitätskontrolle der Dateien
- Derivate erzeugen (z.B. Sichtungskopie)
- Archivpakete erstellen
- Sicherung auf digitaler Archivlösung (z.B. LTO-Magnetbänder, Serverinfrastruktur usw.)

Input zu Qualitätskontrolle

Bern, 13. März 2025

Fallbeispiel

Datei zuschneiden

- MKV/FFV1 in MP4/H.264 umgewandelt
- MP4 zugeschnitten mit folgendem Command:

➔ `ffmpeg -i inputfile.mp4 -ss hh:mm:ss -to hh:mm:ss -c:v copy -c:a copy outputfile.mp4`

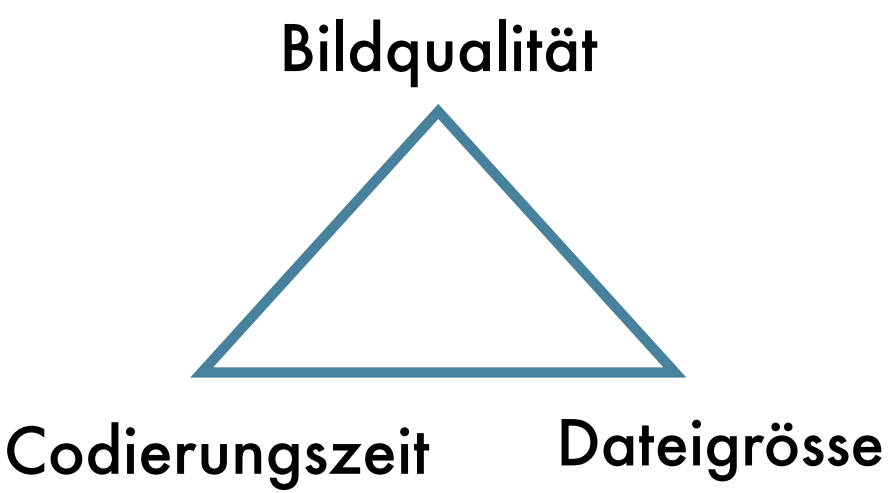
- Qualitätskontrolle durch Sichtung in VLC/QuickTime/Gridplayer
  - ➔ Video wird unterschiedlich abgespielt/interpretiert
- Gleicher Command bei MKV/FFV1 Datei funktioniert gut

Fallbeispiel

Datei zuschneiden

- Qualitätskontrolle in VLC/QuickTime/Gridplayer
  - ➔ Video wird unterschiedlich abgespielt/interpretiert
- Erklärung: Unterschiedliche Programme interpretieren Kompression unterschiedlich Audio/Video Synchronität ist deshalb nicht gegeben.
- Lösung: Command präzisieren

➔ `ffmpeg` **`-accurate_seek`** `-i inputfile.mp4 -ss hh:mm:ss -to hh:mm:ss` **`-c:v libx264 -preset veryslow -crf 18 -pix_fmt yuv420p -c:a aac`** `outputfile.mp4`



Qualitätskontrolle

Vorschlag eines Workflows

- Ordner- und Dateinamen und Struktur
- Prüfsummen, Datenintergrität
- Technische Metadaten prüfen
  - Dateiformat: Container, Codec, Untertitel, Timecode usw.
- Signale analysieren
- Datei sichten und/oder anhören
  - Bild- und Toninhalt

Dateiname und Struktur

- Prüfen der Datenablage und der Benennung nach der internen Namenskonvention
  - Dies ist vom Archiv und internen Prozessen abhängig!
- Sind Dateien eindeutig zu zuordnen?
- Wird anhand der Datenablage und Dokumentation ersichtlich, woher die Dateien stammen und welchen Status sie haben (Archivmaster, Sichtungskopie etc.)?

Prüfsummen

Kryptografisch

- MD5
- SHA-1
- SHA-256
- SHA-512

Nicht-kryptografisch

- CRC-32
- xxHash 32
- xxHash 64
- xxHash 128

Prüfsummen

Tools zur Erstellung von MD5 Prüfsummen

- MD5 (CLI, Mac)
- Checksum+ (GUI, Mac)

- MD5Checker (Windows): http://getmd5checker.com/

Prüfsummen

Commands

- MD5

→ Mac: `md5 input_file`

→ Windows: `certutil -hashfile input_file MD5`

→ `md5sum`

- SHA

→ `shasum /sha1sum / sha256sum / sha512sum`

- Framemd5

→`ffmpeg -i input_file -f framemd5 output_file_framemd5.txt`

Command zum Vergleich von Dateien

- Auf Linux/Mac/Windows Terminal oder WSL

→ `diff -s file01.txt file02.txt > file_diff.txt`

- Auf Windows

→ `fc file01.txt file02.txt > file_diff.txt`

Kann z.B. zum Vergleichen von MediaInfo.txt oder Framemd5.txt Dateien verwendet werden.

Programme zur Analyse

Analyse von Metadaten

- FFprobe (CLI)
- MediaInfo (GUI, CLI)

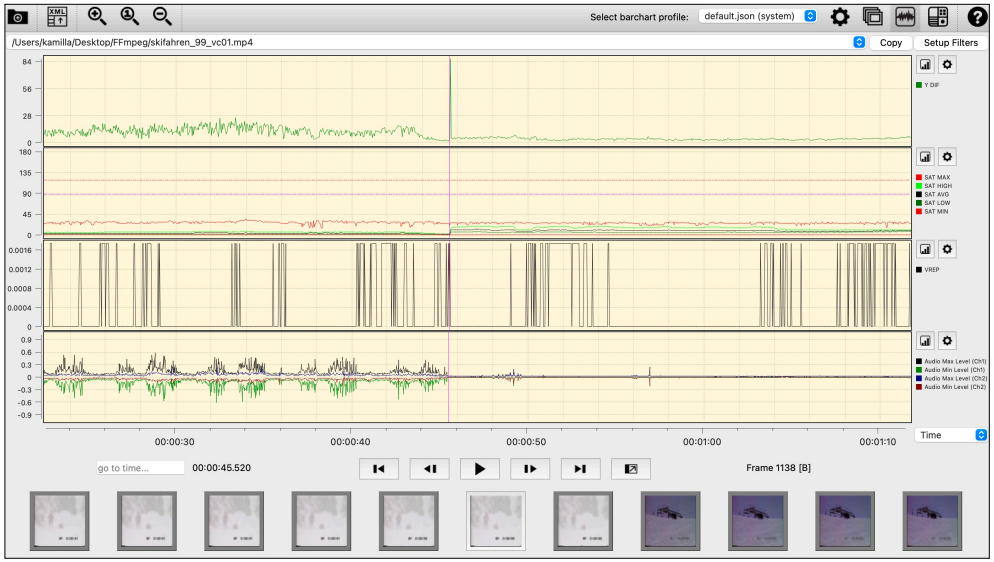
Analyse des Signal

- QCTools (GUI, CLI)
- qcli (CLI)
- SignalServer (Web App)

```
Allgemein
Vollständiger Name                       : skifahren_1999.mov
Format                                   : MPEG-4
Format-Profil                            : QuickTime
Codec-ID                                 : qt   0000.02 (qt  )
Dateigröße                               : 2,37 GiB
Dauer                                    : 1 min 31s
Gesamte Bitrate                          : 223 Mb/s
Bildwiederholungsrate                    : 25,000 FPS
Kodierendes Programm                     : Lavf61.7.100

Video
ID                                       : 1
Format                                   : YUV
Codec-ID                                 : v210
Codec-ID/Hinweis                         : AJA Video Systems Xena
Dauer                                    : 1 min 31s
Bitraten-Modus                           : konstant
Bitrate                                  : 221 Mb/s
Breite                                   : 720 Pixel
Clean aperture width                     : 703 Pixel
Höhe                                     : 576 Pixel
Clean aperture height                    : 576 Pixel
Bildseitenverhältnis                     : 4:3
Clean aperture display aspect ratio      : 4:3
Modus der Bildwiederholungsrate          : konstant
Bildwiederholungsrate                    : 25,000 FPS
Standard                                 : PAL
Color space                              : YUV
Chroma subsampling                       : 4:2:2
Bit depth                                : 10 bits
Scantyp                                  : Interlaced
Scan type, store method                  : Interleaved fields
Scanreihenfolge                          : oberes Feld zuerst
Compression mode                         : Lossless
Bits/(Pixel*Frame)                       : 21.333
Stream-Größe                             : 2,34 GiB (99%)
Sprache                                  : Englisch
Color primaries                          : BT.601 PAL
Transfer characteristics                 : BT.709
Matrix coefficients                      : BT.601

Audio
ID                                       : 2
Format                                   : PCM
```

FFprobe und MediaInfo zur Qualitätskontrolle

Commands

- FFprobe (CLI)

→ `ffprobe -show_format -show_streams -print_format json input_file > input_file.txt`

- MediaInfo (CLI)

→ `mediainfo file.ext`

→ `mediainfo --Details=1 file.ext`

→ `mediainfo --Output=JSON file.ext`

Programme zum Sichten von AV-Dateien

Mediaplayer basierend auf FFmpeg

- FFplay (CLI)
  → ffmpeg.org
- MPV (GUI, CLI)
  → mpv.io
- VLC (GUI)
  → www.videolan.org/vlc
- Gridplayer (GUI)
  → https://github.com/vzhd1701/gridplayer

Commands zum Analysieren von AV-Dateien

Commands

- Splitscreen (https://avpres.net/FFmpeg/split.html)

```
→ ffmpeg -i input_file_1 -i input_file_2 -filter_complex "[0]
crop=iw/2:ih:0:0 [left]; [1] crop=iw/2:ih:iw/2:0 [right];
[left][right] hstack [out]" -map "[out]" output_file
```

- Differenzsignal (https://avpres.net/FFmpeg/delta.html)

```
→ ffmpeg -i input_file_1 -i input_file_2 -filter_complex "[1]
format=yuva444p, lut=c3=128, negate [1_with_alpha]; [0]
[1_with_alpha] overlay [out]" -map "[out]" output_file
```


Input zur Konservierung und Dateiformate

Bern, 13. März 2025

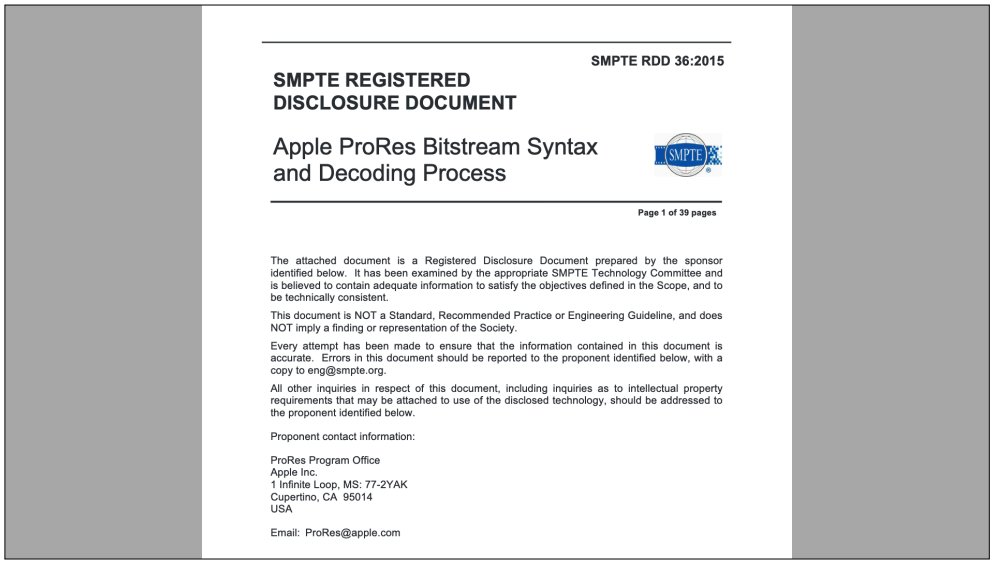
Konservierung

Analoge Elemente

- In bestmöglicher Qualität digitalisieren
- Das analoge Element weiter konservieren
- Bittiefe ist wichtiger als die Auflösung

Digitale Elemente

- Möglichst das native Format konservieren
- Nicht in ein „höheres“ Format konvertieren
- Natives ProRes ist archivtauglich



SMPTE RDD 36:2015

SMPTE REGISTERED DISCLOSURE DOCUMENT

Apple ProRes Bitstream Syntax and Decoding Process

Page 1 of 39 pages

The attached document is a Registered Disclosure Document prepared by the sponsor identified below. It has been examined by the appropriate SMPTE Technology Committee and is believed to contain adequate information to satisfy the objectives defined in the Scope, and to be technically consistent.

This document is NOT a Standard, Recommended Practice or Engineering Guideline, and does NOT imply a finding or representation of the Society.

Every attempt has been made to ensure that the information contained in this document is accurate. Errors in this document should be reported to the proponent identified below, with a copy to eng@smpte.org.

All other inquiries in respect of this document, including inquiries as to intellectual property requirements that may be attached to use of the disclosed technology, should be addressed to the proponent identified below.

Proponent contact information:

ProRes Program Office
Apple Inc.
1 Infinite Loop, MS: 77-2YAK
Cupertino, CA 95014
USA

Email: ProRes@apple.com

Einzelbild-Archivformate

- Ordner, TIFF, 2K/4K, RGB, 4:4:4:4, 16 bit
- MXF (OP 1a), DPX, 2K/4K, R'G'B', 4:4:4:4, 12/10 bit
- MKV, FFV1, 2K/4K, R'G'B', 4:4:4:4, 12/10 bit

Bildstrom-Archivformate

- MKV, uncompressed, SD/HD, $Y'C_BC_R$ 4:2:2, 10 bit
- MKV, FFV1, SD/HD, $Y'C_BC_R$ 4:2:2, 10 bit

Digitale Fernseh-Archivformate

SD-Produktionen

- MXF (OP 1a), MPEG IMX, PAL, 50 Mbit/s

HD-Produktionen

- MXF (OP 1a), XDCAM HD 422, PAL, 50 Mbit/s

Weitere digitale Bandformate sollten ebenfalls im nativen Codec eingelesen werden (z.B. Digital Betacam, DV)

Gruppenarbeit

Bern, 13. März 2025

Beispiele

- Dateien aus DVD „rippen“ und in Sichtungskopien umwandeln
- Umwandlung von DPX-Einzelbilder in Matroska/FFV1-Bildstrom
- Umwandlung von DPX-Einzelbilder mittels RAWcooked
- Uncompressed/Archivmaster in H.264/Sichtungskopie umwandeln

Weitere Commands zum Testen

- ProRes in Sichtungskopie H.264 umwandeln
- Wasserzeichen, Timecode, Vorspann oder Logo hinzufügen

Gruppenarbeit

- Fragestellung und Ziel der Übung wählen, festlegen
- Commands suchen, zusammenstellen und testen
- Qualitätskontrolle der Resultate

Themen

- Selbst mitgebrachtes Thema

- DVD
- DPX umwandeln in MKV
- Umwandeln von uncompressed, SD, 422, interlaced in Sichtungskopie
- Umwandeln von ProRes in Sichtungskopie

Ressourcen

FFmpeg Cookbook for Archivists
→ avpres.net/FFmpeg/

ffmprovisr
→ amiaopensource.github.io/ffmprovisr/

Liste von möglichen Parameter anzeigen

```
ffmpeg -h
ffmpeg -codecs
ffmpeg -decoders
ffmpeg -h decoder=flac
ffmpeg -encoders
ffmpeg -h encoder=ffv1
ffmpeg -filters
ffmpeg -formats
ffmpeg -layouts
ffmpeg -sample_fmts
ffmpeg -pix_fmts
ffmpeg -bsfs
```

Input zu DVDs /optische Datenträger

- Was sind optische Datenträger?
  - CD (CD-DA, CD-ROM, CD-R, CD-RW)
  - DVD (DVD-Video, DVD-Audio, DVD-ROM, DVD+R, DVD-R, DVD+RW, DVD-RW..)
  - Laserdisc, WORM, Blu-ray, M-Disc, MiniDisc usw.
- Identifikation ist wichtig, um den richtigen Workflow zur Sicherung wählen zu können

Input zu DVDs /optische Datenträger

- Analyse eines optischen Datenträgers
  - Handelt es sich um eine Video DVD? Was ist die Datenstruktur?
  - Analyse der Metadaten anhand von Programmen
- DVD Inhalte rippen oder/und imagen?
  - Disk Image (AV-Inhalte inklusive DVD Struktur archivieren)
  - AV-Dateien extrahieren und bei Bedarf zu einer Datei umcodieren

DVD - Aufbau

Dateien
- BUP - Backup Datei
- IFO - Informationen
- VOB - AV-Datei!

- AV-Inhalte werden in mehrere VOB Dateien aufgeteilt

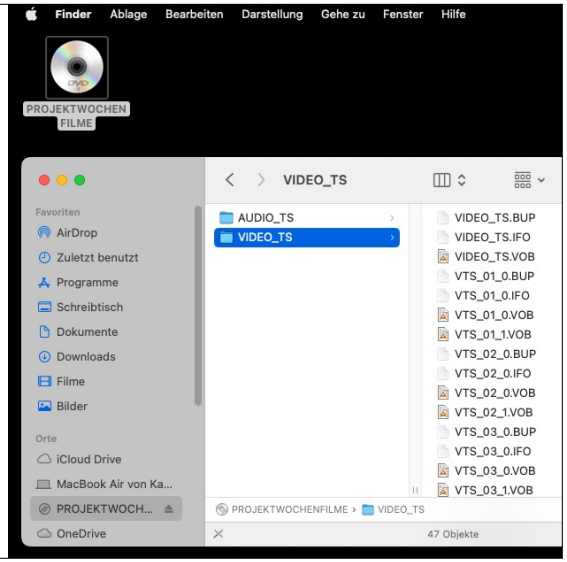


Abbildung: Datenablage einer selbstgebrannte DVD

DVD - Aufbau

Kommerzielle DVDs
- Können geschützt sein
- Umgang mit Copyright
- Enthalten gleiche Art von Dateien

- AV-Inhalte werden in mehrere VOB Dateien aufgeteilt

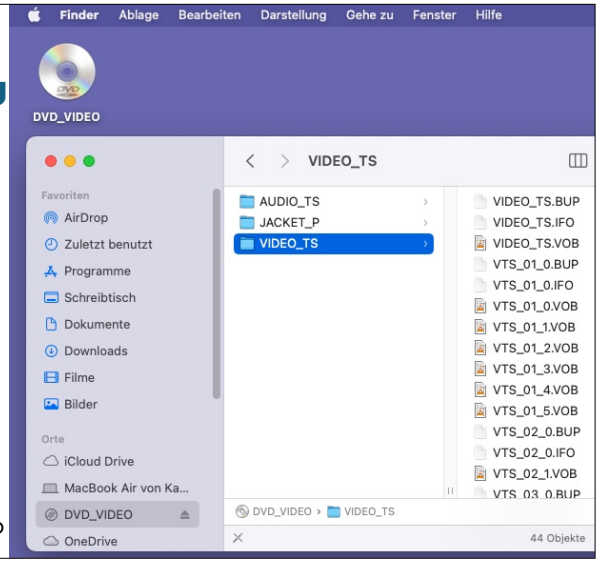


Abbildung: Datenablage einer kommerzielle DVD

Aufgabe: DVD Dateien umcodieren

- AV-Dateien extrahieren und zu einer Datei umcodieren
  - Welche Dateien sind fürs Umcodieren zu berücksichtigen?
  - Wie kann aus mehreren Dateien eine Video-Datei erstellt werden?
  - Wie können die Dateien korrekt in MP4/H.264 umgewandelt werden?

Commands: DVD

- AV-Inhalte einer DVD in eine MP4 Datei umwandeln

➔ **ffmpeg** -i "concat:VTS_01_1.VOB|VTS_01_2.VOB|VTS_01_3.VOB" -b:v 1500k -r 30 -vcodec h264 -strict -2 -acodec aac -ar 44100 -f mp4 output_file.mp4

➔ **ffmpeg** -i "concat:*input_file_1*|*input_file_2*|*input_file_3*" -c:v libx264 -c:a aac *output_file*.mp4

➔ alle Audio und Video Kanäle berücksichtigen: **ffmpeg** -i "concat:*input_file_1*|*input_file_2*" **-map 0:v -map 0:a** -c:v libx264 -c:a aac *output_file*.mp4

Commands: DVD

- AV-Inhalte einer DVD in eine MP4 Datei umwandeln

➔ deinterlacing:

ffmpeg -i "concat:VTS_02_1.VOB|VTS_03_1.VOB|VTS_04_1.VOB" -filter:v "idet, bwdif" -c:v libx264 -c:a aac output_H264.mp4

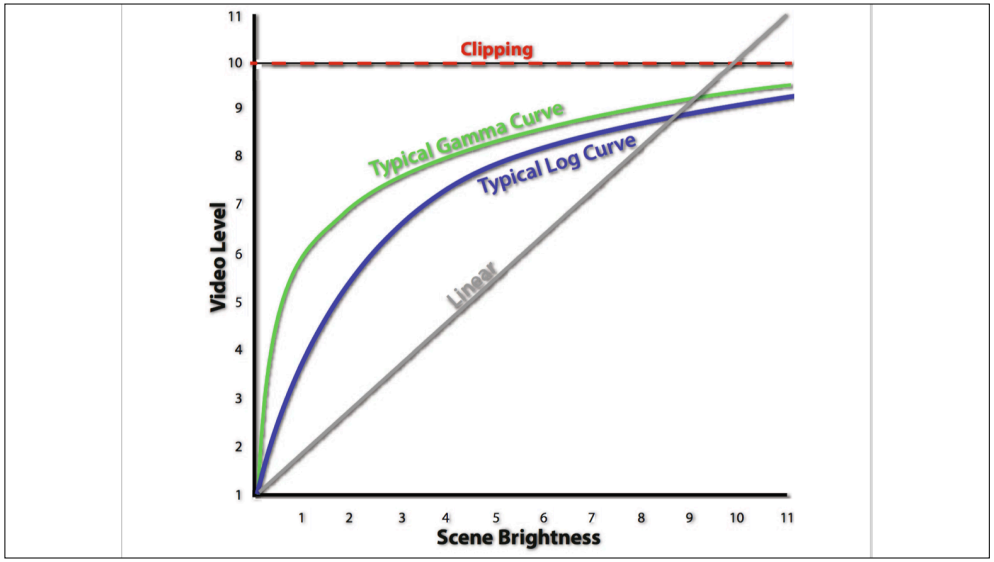
➔ deinterlacing und auf HD skaliert:

ffmpeg -i "concat:VTS_01_1.VOB|VTS_01_2.VOB|VTS_01_3.VOB" -filter:v "idet, bwdif, **scale=1440:1080:flags=lanczos, pad=1920:1080:240:0**" -c:v libx264 -c:a aac output_H264.mp4

Input zu DPX-Dateien

- DPX-Dateien (Digital Picture Exchange) werden von high-end Kameras oder Scannern bei der Aufnahme oder der Digitalisierung generiert

- Was befindet sich in einer DPX?
  - Log neg encoding
  - Log RGB encoding oder quasi-log encoding
  - Gamma encoding oder power function encoding
  - Scene-linear encoding




```
-bash
Licensed DPX flavors:
Yes DPX/Raw/RGB/8bit/U/LE
Yes DPX/Raw/RGB/10bit/U/LE/FilledA
Yes DPX/Raw/RGB/10bit/U/BE/FilledA
Yes DPX/Raw/RGB/12bit/U/LE/FilledA
Yes DPX/Raw/RGB/12bit/U/BE/Packed
Yes DPX/Raw/RGB/12bit/U/BE/FilledA
Yes DPX/Raw/RGB/16bit/U/LE
Yes DPX/Raw/RGB/16bit/U/BE
Yes DPX/Raw/RGBA/8bit/U/LE
Yes DPX/Raw/RGBA/10bit/U/LE/FilledA
Yes DPX/Raw/RGBA/10bit/U/BE/FilledA
Yes DPX/Raw/RGBA/12bit/U/LE/FilledA
Yes DPX/Raw/RGBA/12bit/U/BE/Packed
Yes DPX/Raw/RGBA/12bit/U/BE/FilledA
Yes DPX/Raw/RGBA/16bit/U/LE
Yes DPX/Raw/RGBA/16bit/U/BE
Yes DPX/Raw/Y/8bit/U/LE
Yes DPX/Raw/Y/10bit/U/BE/FilledA
Yes DPX/Raw/Y/10bit/U/BE/FilledB
Yes DPX/Raw/Y/12bit/U/BE/Packed
Yes DPX/Raw/Y/16bit/U/LE
Yes DPX/Raw/Y/16bit/U/BE

Licensed TIFF flavors:
```


Aufgabe: DPX-Dateien

- DPX Einzelbilder in Videodatei (MKV) umwandeln
- Übungsdateien Einzelbilder:
  - DPX_10-bit/Test_000nnnnn.dpx oder DPX_12-bit/ Test_000nnnnn.dpx
  - Quelle:16-mm-Filmszene, generiert in DaVinci Resolve
- Beispieldateien Bildstrom:
  - Test_DNxHR_444_12-bit.mxf

Commands: DPX

- FFmpeg Befehle:

➔ Für Einzelbilder: **`ffmpeg -f image2 -framerate 24`** `-i` *`input_file_%08d.dpx`* `-c:v ffv1 -level 3 -threads 8 -coder 1 -context 1 -g 1 -slices 24 -slicecrc 1 -start_number 00086400 -c:a copy` *`output_file.mkv`*

➔ Für Bildstrom: **`ffmpeg`** `-i` *`input_file.mxf`* `-c:v ffv1 -level 3 -threads 8 -coder 1 -context 1 -g 1 -slices 24 -slicecrc 1 -c:a copy` *`output_file.mkv`*

DPX - RAWcooked

- Mit RAWcooked können DPX in Matroska mit FFV1/FLAC enkodiert werden
- Möglichkeit eingebetteter Metadaten als sidecar files bei RAWcooked

- RAWcooked: https://mediaarea.net/RAWcooked

Input zu ProRes

- ProRes Dateien werden von vielen Kameras erzeugt und haben in diesem Fall die höchstmögliche Qualität
- Auch wenn digital bearbeiteten Dateien (FinalCut, Adobe Premiere, DaVinci Resolve) als ProRes exportiert werden, ist dies die höchstmögliche Qualität
- ProRes codecs können in MOV, MXF oder MKV Container vorhanden sein.
- Was gibt es zu berücksichtigen?

Input zu ProRes

ProRes-422-Familie
- ProRes 422 HQ (high quality)
- ProRes 422 (standard)
- ProRes 422 LT (light)
- ProRes Proxy

ProRes-4444-Familie
- ProRes 4444 XQ
- ProRes 4444

ProRes-RAW-Familie
- ProRes RAW HQ
- ProRes RAW

Commands: ProRes Datei in Sichtungskopie umcodieren

- Beispiel : selbst mitgebrachte ProRes Datei
- Metadaten der Datei analysieren vor Umcodierung

- FFmpeg command:

```
→ ffmpeg -i inputfile.mov -pix_fmt yuv420p -c:v
libx264 -preset veryslow -crf 30 -movflags
+faststart+write_colr outputfile_ProRes_H264.mp4
```

- Quelle: https://avpres.net/FFmpeg/im_H264.html

Aufgabe: uncompressed in Sichtungskopie umcodieren

- Beispiel 1: skifahren_99.mov
  - Quelle: Video8 Kassette, digitalisiert
  - Format: MOV/V210; uncompressed, YUV, 4:2:2, interlaced, PAL, 10 bit
- Was gibt es zu berücksichtigen?
  - Interlaced zu progressive / deinterlacing erwünscht?
  - 4:3 beibehalten oder in 16:9 umcodieren?

Commands: uncompressed in Sichtungskopie

Beispiel 1: uncompressed, V210, 4:2:2, interlaced, PAL, 10 bit

- FFmpeg commands:

➔ **ffmpeg** -i input_file.MOV -c:v libx264 -preset veryslow -crf 18 -pix_fmt yuv420p -c:a aac output_file.MP4

➔ deinterlacing: **ffmpeg** -i input_file -c:v libx264 -pix_fmt yuv420p **-filter:v "idet, bwdif"** output_file

➔ 4:3 in 16:9 mit Pillarbox: **ffmpeg** -i *input_file* -c:v libx264 -filter:v **"yadif, scale=1440:1080:flags=lanczos, pad=1920:1080:(ow-iw)/2:(oh-ih)/2, format=yuv420p"** *output_file*

Weitere Möglichkeiten mit FFmpeg

- Scanmodus ändern - deinterlacing (interlaced > progressive)
- Datei zuschneiden (siehe Fallbeispiel)
- Farbraum ändern (wann wäre das sinnvoll?)
- Einfügen von Wasserzeichen / Vorspann / Abspann / Logo / Timecode

Scanmodus ändern

- Scanmodus ändern

➔ deinterlacing **ffmpeg** -i input_file -c:v libx264 -pix_fmt yuv420p **-filter:v "idet, bwdif"** output_file

➔ deinterlacing **ffmpeg** -i input_file -c:v libx264 **-vf "yadif,format=yuv420p"** output_file

Input - Farbraum ändern

- Colormatrix ändern
- Wann sollte Colormatrix geändert werden?
- Wie erkenne ich aktuelle Colormatrix in den Metadaten

Farbräume ändern
Wann sollten Farbräume geändert werden?

Farbraum ändern - Commands

- Colormatrix ändern:

➔ **ffmpeg** -i *input_file* -c:v libx264 -vf colormatrix=src:dst *output_file*

➔ Beispiel Rec. 601 zu Rec. 709 ändern: **ffmpeg** -i *input_file* -c:v libx264 -vf colormatrix=bt601:bt709 *output_file*

Farbräume für PAL:

➔ **ffmpeg** -i *input_file* -c:v libx264 -color_primaries bt470bg -color_trc bt709 -colorspace bt470bg *output_file*

Farbräume für NTSC:

➔ **ffmpeg** -i *input_file* -c:v libx264 -color_primaries smpte170m -color_trc bt709 -colorspace smpte170m *output_file*

Dateieigenschaften ändern

- Wasserzeichen, Timecode, Vorspann und Logo hinzufügen

➔ Im Vorfeld muss geklärt werden, welche Fonts auf dem Computer installiert sind und der Pfad der gewünschten Font muss man kennen und einsetzen

➔ Befehl zum prüfen der installierten fonts

Auf macOS:

```
> ls /Library/Fonts
```

Auf Windows:

```
> dir \Windows\Fonts
```


Wasserzeichen

- Wasserzeichen hinzufügen
- https://amiaopensource.github.io/ffmprovisr/#text_watermark

➔ **ffmpeg** -i *input_file* -vf drawtext="fontfile=*font_path*:fontsize=*font_size*:text=*watermark_text*:fontcolor=*font_color*:alpha=0.4:x=(w-text_w)/2:y=(h-text_h)/2" *output_file*

➔ Beispiel: **ffmpeg** -i *input_file* -filter:v "drawtext=text='watermark':fontfile='/Library/Fonts/Arial.ttf':fontsize=35:fontcolor=white:alpha=0.25:x=(w-text_w)/2:y=(h-text_h)/2" *output_file*

Logo einfügen

Logo hinzufügen

- Logo muss als PNG Datei vorhanden sein
- Logo sollte wesentlich kleiner als das Video-Bild sein

→ **ffmpeg** -i input_file.mp4 -i logo.png -filter_complex "overlay=10:main_h-overlay_h-10" with_logo.mp4

Timecode einfügen

- Timecode hinzufügen

→ **ffmpeg** -i *input_file* -filter:v drawtext="timecode=starting_timecode:rate=timecode_rate :fontfile=*font_path*:fontsize=*font_size*:fontcolor=*font_colour*:box=1:boxcolor=*box_colour*:x=(w-text_w)/2:y=h/1.2" *output_file*

→ Beispiel: **ffmpeg** -i input_file.mp4 -filter:v drawtext="timecode='01\:00\:00\:00':rate=25:fontfile='/Library/Fonts/Arial.ttf':fontsize=35:fontcolor=white:x=(w-text_w)/2:y=h/1.2" with_timecode.mp4

Vorspann / Abspann

- Vorspann oder Abspann hinzufügen
  - Muss als Videodatei vorhanden sein und dieselben Eigenschaften besitzen wie die zu bearbeitende Videodatei

Dateieigenschaften ändern

- Untertitel im SRT Format hinzufügen

→ **ffmpeg** -i *output_archive.mkv* -filter:v **"subtitles=subtitles.srt"** -c:v libx264 -preset veryslow -crf 18 -pix_fmt yuv420p -c:a aac -ar 48k -movflags +faststart+write_colr *output_streaming.mp4*

Input zum SRT-Format

- SubRip-Untertiteldatei
- Textdatei beinhaltet Text der Untertitel in der richtigen Reihenfolgen mit Start- und End-Timecode
- Es gibt Tools zum erstellen von Untertitel. Man kann sie auch mit einem Texteditor erstellen

Das SRT-Format:

- Fortlaufende ganze Zahl
- Start Timecode –> Stopp Timecode
- Text des Zwischentitels
- Leerzeile

Beispiel:

- 5
- 00:01:28,250 –> 00:01:30,500
- Das ist ein Untertitel
- auf zwei Zeilen

Beobachtungen

- Was haben Sie getestet?
- Was haben Sie dabei beobachtet, gelernt?
- Inwiefern verändern sich die Dateien (Metadaten und optisch)?
- Welche Commands eigenen sich wofür am besten?

Input zu Stapelverarbeitung

Bern, 13. März 2025

Stapelverarbeitung

Möglichkeiten für Stapelverarbeitung mit FFmpeg commands

- FFCommand Engine
- HandBrake
- Audacity
- ShutterEncoder
- MediaConch (ffprobe integrieren)
- Scripts (Bash, Python usw.)

Stapelverarbeitung

- Beispiel Anwendung eines Commands auf mehrere Dateien
  - 1 Command soll bei allen Dateien, die sich in einem Ordner befinden, durchgeführt werden
- Beispiel Kombination von Commands
  - Mehrere Commands in einer bestimmten Reihenfolge bei einer Datei ausführen

Workflow für Stapelverarbeitung

Anwendung eines Commands auf mehrere Dateien

- Ordner mit mehreren gleichen Dateien vorbereiten
- Welcher Command soll ausgeführt werden?

Anwendung mehrerer Commands in bestimmter Reihenfolge bei einer Datei

- Workflow überlegen
- Commands zusammenstellen

FFcommand Engine

- FFCommand Engine installieren
- Link: https://github.com/ColorlabMD/FFCommand_Engine
- Mehrere FFmpeg-Befehle können importiert, in queue gespeichert und nacheinander durchgeführt werden.

Programme

- HandBrake installieren
  - Link: https://handbrake.fr/
  - FFmpeg Commands in HandBrake einbauen
- Audacity installieren
  - Link: https://www.audacityteam.org/download/
  - Gewünschte FFmpeg Commands in Audacity integrieren
- ShutterEncoder
  - https://www.shutterencoder.com/

Beobachtungen

- Welche Programme haben Sie getestet?
- Welche Commands haben Sie kombiniert?

Danke für Ihre Aufmerksamkeit!

- Feedback
- Was hat gefehlt?

AV Preservation by reto.ch Ødegård & Bernard Restaurierung

Reto Kromer:
reto.ch / info@reto.ch
Eléonore Bernard und Kamilla Ødegård:
https://atelier40a.ch/odegard-bernard-restaurierung-klg/
eleonore.bernard@atelier40a.ch /
kamilla.oedegard@atelier40a.ch