# RAWcooked

*Data is never RAW.*
*Data is always cooked in a way or another.*

Jérôme Martinez
MediaArea

FOSDEM, Brussels, February 2018

## FFV1

Lossless video compression format from FFmpeg project

Open source, patent free

Adopted by several archives

Being standardized (IETF)

Frames are divided by slices, with checksums

## About FFV1 standardization

IETF CELLAR Working Group

Well advanced

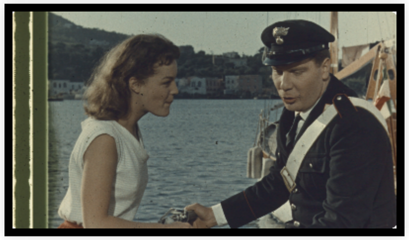
A standard this year?

We need external review

## Compression

Example with 1 second at 24 fps 10-bit HD film on a 6-core (12-thread) Skylake-X CPU:

- 24 DPX files (or in ZIP/TAR uncompressed): 189 MB
- 1 compressed ZIP file: 175 MB in 10 seconds
- 1 compressed LZMA2 file: 154 MB in 30 seconds
- 1 compressed LZMA2 solid (risky) file: 144 MB
- 1 FFV1/MKV Intra 16-slice file: 105 MB in 1.5 seconds



## FFmpeg advantages

- Open source and free
- Package is 1.5-3x smaller than DPX/TIFF
- Cheksum by "Cluster" (usually 1 second) at container level
- Cheksum by "Slice" (you choose how many per frame) at video level
- Files are natively playable by lot of tools (FFmpeg, VLC...)

# FFmpeg drawbacks

- You lose some metadata
  (DPX/TIFF header: scan software, some colorimetry info, film type, DPX time code, shutter angle, gamma...
  BWF Metadata: history of digitalization, loudness info... and opaque WAV chunks...)
- "This is not the exact source content we are requested to store" issue
- But...

## RAWcooked

- Store DPX/TIFF/WAV header/footer in Matroska elements
- Store source file names in Matroska elements
- Store other sidecar files as Matroksa attachments
- Conversion is reversible (bit-by-bit to original files, directory structure and file names)

## Usages

- Storage
  Save HDD/LTO space (less €...)
- Transport
  Encode, transport, decode; you save bandwidth (€... and transfer speed) without changing something else in your workflow (same files after revert to DPX/TIFF)

## Some drawbacks

- More CPU intensive than uncompressed ZIP
  Real time "only" with modern CPU and HD content
  (Future project: use GPU and/or SSE/AVX)
- You lose a complete slice content if there is a single corrupted bit
  (Mitigated with several slices per frame, MediaConch can fix a bitflip error and could be extended to a byte error; we could add some error-correcting codes)

## Development time line

First development snapshots available

Beta release in March 2018

Stable release in April 2018

## What is in the development repo now?

Few DPX/TIFF flavors supported

Analysis of the DPX, creation of a data file for reversability (will be changed to Matroska elements)

Provide the FFmpeg command for encoding (will be changed to something more user friendly)

Parse and uncompress natively back to DPX

## Still lot of work

All DPX/TIFF flavors (endianess, bit depth, components, padding...

Handling of a complete directory (several video and audio streams, extra files...)

Use of Matroska elements instead of attachments (current implementation, not good)

A GUI

More input formats (TIFF, EXR...)

More archiving features (error-correcting codes...)

## Initial funding

Developed by MediaArea
https://mediaarea.net

Main sponsorship by AV Preservation by reto.ch
https://reto.ch

With additional financial support from some other archives:

- CNA (National Audiovisual Centre of Luxembourg)
- Nasjonalbiblioteket (National Library of Norway)
- IFI (Irish Film Institute)

# Business model

Alway open source

But the delivered binaries have support of only few DPX flavors and few features

User need to buy a key for other flavors/features

## Stay in touch

MediaArea: https://mediaarea.net, @MediaArea_net

RAWcooked: https://MediaArea.net/RAWcooked

Jérôme Martinez: jerome@mediaarea.net

Slides: https://mediaarea.net/Events