NAME

openmovim − Command−line interface to encode, decode, play and analyse moving images using 'libmovim'

SYNOPSIS

openmovim (**−e** | **−d** | **−p** | **−a** | **−m**) **−i** *input_file* [**−o** *output_file*]

openmovim (**−c** | **−u**) **−i** *input_file* [**−o** *output_file*]

openmovim −h | **−v**

DESCRIPTION

MovIm is a video codec for moving images specifically designed for both conservation and restoration purposes.

libmovim is a C library implementing the **MovIm** video codec and **movimenc**, **movimdec** and **movimplay** are its associated command−line utilities.

openmovim is a Bash command−line interface to **libmovim** allowing to encode, decode, play and analyse moving images.

The **openMovIm** package includes the **libmovim** library and its associated **movimenc**, **movimdec** and **movimplay** command-line utilities, as well as the **openmovim** command−line interface.

OPTIONS

−h, −−help
display a help message

−v, −−version
display the running version

GENERAL OPTIONS

Select the mode:

−e, −−encode
encoding mode: encode an *input_file* to an *output_file* by using the **movimenc** command-line utility

The "raw" encoding is done in Gray code rather than in natural binary, in order to speed up significantly the processing time.

−d, −−decode
decoding mode: decode an *input_file* to an *output_file* by using the **movimdec** command-line utility

−p, −−play
playing mode: play an *input_file* by using the **movimplay** command-line utility

This mode is highly experimental! It is beneficial when **libmovim** is used as a standalone application rather than as an embedded library into another application, such as a restoration suite.

Please remember that, depending on the resolution, the number of channels, the bit-depth and the available computing power, the moving images may play very slowly, far below real time.

The **−−select** and **−−ignore** options allow to play only some channels, or even only some bit-planes of channels.

The author is indebted to Fabrice Bellard (and his **bpgview**) and to **mpv** for the inspiration given.

−a, −−analyse, −−analyze
analysing mode: analyse the validity of a **MovIm**−encoded *input_file* and writes a report to an *output_file* if specified or to the Terminal otherwise

−m, −−metadata
metadata mode: extract the technical metadata of an **MovIm**−encoded *input_file* (without analysing its validity) and writes a report to an *output_file* if specified or to the Terminal otherwise

or select an action:

−c, −−compress
compress an *input_file* and change **−−compression** from *no* to *yes*

If no *output_file* is specified, then the *input_file* is overwritten.

A lossless compression can be applied for conservation purposes, in order to reduce the needed storage, typically between one and two thirds, depending on the image content.

The wavelet compression used is possibly a wee bit better than HuffYUV, FFV1 or JPEG2000 in terms of both speed and compression rate.

−u, −−uncompress
expand an *input_file* and change **−−compression** from *yes* to *no*

If no *output_file* is specified, then the *input_file* is overwritten.

The "raw" format is always faster for restoration, because any compression would slow down significantly the image processing.

Select the file(s):

−i *input_file*, **−−input=***input_file*
In encoding mode, all container formats supported by FFmpeg should work.

In decoding, analysing or metadata mode, the container formats NUT (.nut), MP4 (.mp4), QuickTime (.mov), AVI (.avi) and Matroska (.mkv) have been tested as wrappers for the MovIm video codec. Yet also the uncompressed or lossless compressed data can be used directly as a file (.movim).

−o *output_file*, **−−output=***output_file*
In encoding mode, the container formats NUT (.nut), MP4 (.mp4), QuickTime (.mov), AVI (.avi) and Matroska (.mkv) have been tested as wrappers for the MovIm video codec. Yet also the uncompressed or lossless compressed data can be used directly as a file (.movim).

In decoding mode, all container formats supported by FFmpeg should work.

In analysing or metadata mode, the output file format can be plain text (.txt), JSON (.json) or XML (.xml).

ENCODING OPTIONS

The following list is not exhaustive.

−−xyz−matrix=(*x_0 y_0 z_0 . . . x_n y_n z_n***)**
defines how *input_files* should be read and how *output_files* should be written

A channel can be not only one of the current R, G, B, Y, Cb, Cr, Co, Cg or alpha, but also a Bayer-filtered channel or any band of a multispectral scan. Any number of channels is supported.

The format of the XYZ matrix is still evolving. An example of the current matrix format (CIE RGB) is:

```
0.7355  0.2645  0.0000
0.2658  0.7243  0.0099
0.1669  0.0085  0.8246
```

−−illuminant=(*x y z***)**
defines the illuminant

The default value is D65, i.e.

```
0.31271  0.32902  0.35827
```

−−bit−depth=*bit_depth*
bit_depth can be any positive integer

We have tested mainly with *10*, *12*, *16* (default) or *24* per channel. We suggest to digitise at 16−bit

per channel and to use this bit−depth for actual restoration work.

Currently, 24−bit per channel is primarily meant for research purposes on file formats for the future, because it can hardly be transcoded into current formats.

−−endian={*big*|*little*}
endianness can be *big* or *little* (default)

−−compression={*no*|*yes*}
compression can be *no* (default) or *yes*

This option is used by the **−−encode** mode. See also the **−−compress** and **−−uncompress** actions above.

OTHER OPTIONS

The following list is not exhaustive.

−−report−fmt={*json*|*plain*|*xml*}
report format can be *json*, *plain* text (default) or *xml*

−−select=*channel*[**=***bit_plane*]
In play mode, allows to select only one *channel*, or even only one single *bit_plane* of a *channel*.

This option may be repeated.

−−ignore=*channel*[**=***bit_plane*]
In play mode, allows to ignore a full *channel*, or even only one single *bit_plane* of a *channel*.

This option may be repeated.

−−lut=[*channel***=**]*path*
path to an 1D LUT to apply (default is no LUT)

A LUT can be applied in each mode to the whole input file or only to a single *channel*.

This option may be repeated.

For 1D LUT, which transforms e.g. from floating−point scene linear into camera log or a display−referred space, the maximum allowed size is currently 65’536, i.e. 16−bit precision. (24−bit precision is implemented more as a gag than for real work, because of its size of 16’777’216...)

NOTES

The current **openmovim** command−line interface is a work in progress. Foremost, no validity check of the passed parameters has been implemented so far.

Currently, all three values *x*, *y* and *z* must be given for control purposes; this may change in the future. Alternatively, the centre frequency (expressed in Hz) can be used rather than the centre wavelength (expressed in nm).

Currently, the bit-depth must be the same for all channels; this might change in the future.

Chroma subsampling is not supported.

Please note that **openMovIm** is an enhancement of **openYCoCg** and **openMSMI**, and actually it supersedes these two video codecs.

SEE ALSO

movimdec(1), **movimenc**(1) and **movimplay**(1).

COPYRIGHT

LICENSE

DISCLAIMER

The **openMovIm** package is provided "as is" without warranty or support of any kind.