Using convmlv 2.0.3

Developing Magic Lantern footage with glee

by Sofus Rose

November 24, 2018

Contents

1 Introduction

Processing the output of Magic Lantern for maximum quality is a technical balancing act. *Especially* on Linux, many neccesary programs simply don't exist for open source development of raw video, never mind MLVs!

convmlv aims to change that, and to make the development of MLVs, RAWs, and plain DNG sequences as simple and as featureful as possible in the same program, with equal amounts of help and technical depth.

I personally use it! Earlier versions of convmlv feature in my own short films: `https://www.youtube.com/watch?v=a7iSjEfch5s&t=5s` and `https://www.youtube.com/watch?v=yi-G7sXHB1M&t=1s`.

1.1 Important Links

The Forum Post: `http://www.magiclantern.fm/forum/index.php?topic=16799.0`
The Git Repository: `https://github.com/so-rose/convmlv`
The Dev Repository: `https://git.sofusrose.com/so-rose/convmlv`
Releases: `https://github.com/so-rose/convmlv/releases`

1.2 What can it do?

This entire document is dedicated to the answer! In short, it is a program **allowing you to develop .RAW, .MLV, or sequences of .DNGs into the highest quality image and video formats**. Many useful options are exposed to facilitate that, and a good editing/grading experience afterwards.

It can be as simple - 'convmlv -m footage.mlv' to develop the MLV to an MOV - or as complex as you need it to be!

1.3 Terminal: Not So Scary! I swear! (A Crash Course)

I swear, it's actually quite intuitive! Here's a crash course. First, open it up. You'll see a prompt, something like this:

```
user@computer:~
```

You're in your **Home Folder**, often abbreviated by a . Let's look at what files and folders there are:

```
ls #List - Press enter!
```

Look at all those files! Let's add a couple of "flag" to get a better idea:

```
ls -l #The l flag lists things much nicer.
```

But let's say I wanted to go to Desktop:

```
cd Desktop #Change Directory, with one option: Desktop!
```

Notice the prompt has changed:

```
user@computer:~/Desktop
```

You can type ls again to list stuff on your Desktop. If you want to go back up to home, type 'cd ..' (.. means up, . means current directory). And, you're set!

If you want to learn more, there's a lot of reading material out there. Start here: https://www.digitalocean.com/community/tutorials/an-introduction-to-the-linux-termina

2 Installation

2.1 The Easy Way

If you have a Debian, Ubuntu, Fedora, or Mac OS, the installation is easy:

1. Download the latest .tar.gz release from `https://github.com/so-rose/convmlv/releases`, and put it in its own folder.

2. Open up a Terminal, and cd to that folder containing only the downloaded .tar.gz file.

3. Extract the release using your favorite utility. In the terminal you can do:

   ```
   tar -xvzf *.tar.gz
   ```

4. Now, if you're on a Mac (**ONLY IF YOU'RE ON A MAC**), you need a package manager called "Homebrew". Installation of it is easy; run the one-line command over at `https://brew.sh/`.

5. Install all the distribution dependencies. The commands for Ubuntu, Debian, Fedora, and Mac can be found in the DEPENDENCIES file. On Debian you can run the following; replace the 0 with a 1 for Ubuntu, 2 for Fedora:

   ```
   sudo apt-get install $(convmlv -K 0) #$Only for Debian.
   sudo apt-get install $(convmlv -K 1) #$Only for Ubuntu.
   sudo apt-get install $(convmlv -K 2) #$Only for Fedora.
   sudo brew install $(convmlv -K 3) #$Only for Mac.
   ```

6. Install Python dependencies. The command is the same everywhere (make sure you're using Python 3.X!)

```
sudo python3 -m pip install numpy #Make sure numpy is installed
sudo python3 -m pip install $(./convmlv -Y) #$Use pip to install things!
```

7. Finally, to be able to use convmlv as a commmand by itself, you can type this:

```
ln -s $(pwd)/convmlv.sh /usr/local/bin/convmlv #$Can call it normally now.
```

3 Manual Installation

3.1 Introduction

convmlv comes in the form of a ".sh" script. Installation is a little hairy, but bear with me, and you'll be done soon.

Besides the script itself, several **dependencies** are required. How to best get these varies by system.

To begin, download or clone the source code from `https://github.com/so-rose/convmlv`.

3.2 Linux

On Linux systems, everything is easy. There is certain setup that must be done, however. Open up a terminal, navigate to the source code (containing convmlv and balance.py), and type each line:

```
chmod +x convmlv.sh balance.py sRange.py #Allow the scripts to execute.
```

Check that you can load the help page with './convmlv.sh -h'.

Now, you must install the dependencies. First off, these are the package dependencies, which your system has a unique way of providing. Type './convmlv.sh -h' to access the help, then type '/-K' and Enter to search for '-K'. Next to each supported distribution is a command; copy this command with Shift+Ctrl+C !

```
./convmlv.sh -h #The help page
```

Find and install the corresponding packages on your machine by typing the command you just copied. Note that you must have installed Homebrew on your Mac for this to work (See Mac instructions below):

```
sudo apt-get install $(convmlv -K 0) #$Only for Debian.
sudo apt-get install $(convmlv -K 1) #$Only for Ubuntu.
sudo apt-get install $(convmlv -K 2) #$Only for Fedora.
sudo brew install $(convmlv -K 3) #$Only for Mac.
```

Next, you must install the Python dependencies. This is easy, and works the same everywhere:

```
sudo python3 -m pip install numpy #Make sure numpy is installed
sudo python3 -m pip install $(./convmlv -Y) #$Python 3 version of pip.
```

Finally, you must install the manual dependencies made by the talented Magic Lantern community. You can list these by typing './convmlv.sh -M'.

'convmlv -h' lists all the URLs that you need.

How To Do It: Download each item in the list, making sure you put it in the same folder as convmlv.sh. 'chmod +x' each file by itself, or just run this after downloading them all:

```
chmod +x $(./convmlv.sh -N) #$Batch chmod
```

If you want to be able to type convmlv anywhere, you need to link it to your PATH:

```
sudo ln -s $(pwd)/convmlv.sh /usr/local/bin/convmlv #$ Link for execution.
```

That's it! You're all ready to go!

3.3 Mac

Follow the above UNIX directions, using the Mac Terminal. However, you must install Homebrew first.

See `http://brew.sh/` for more instructions. Otherwise, there are no differences.

4 Output Options

4.1 The Basics

The development of an mlv named 'footage.mlv' in the current directory into a ProRes video is very, very simple:

```
convmlv -m footage.mlv
```

A folder named 'raw-conv' will be created, inside of which a folder named 'footage' will be created, inside of which you'll find your high-quality, ready to edit ProRes video! Feature-wise, however, this is just the tip of the iceberg.

Image sequences, used for the highest quality color grading and VFX, can be outputted like so (compressed losslessly automatically):

```
convmlv -i footage.mlv
```

In the same 'footage' folder, you'll find a folder named exr-footage containing a sequence of OpenEXR files! Other formats are available; refer to 'convmlv -h'.

4.2 Proxies

With such big files, slowdowns often happen during editing. As such, we can use a lighter, lower-quality representation for fast editing. These are 'proxies': convmlv can make them for you in the form of either an MP4 (mode 1), a JPG sequence (mode 2) or both (mode 3). Specify that you want a proxy, along with which mode, with the -p option as such:

```
convmlv -i -p 3 footage.mlv #Will generate mp4/jpg proxy.
```

The proxy will pop up beside your final footage.

By default, the proxies are 1/2 the scale of the original footage, so that editing them is faster. Specify the proxy scale with -s, in percentages, if you want to change this:

```
convmlv -i -s 75% -p 3 footage.mlv #75% proxy scale.
```

When selecting JPG proxy, it **won't** be generated without -i. When selecting MP4 proxy **will**, it be generated **no matter what**.

Remember - you can mix & match! Specifying both -i and -m will make both video and image sequences.

4.3 Frame Range

In going over your footage with a tool like MLRawViewer, you may discover that you don't want the entire thing - especially when dealing with limited disk space! You can specify a frame range to develop:

```
convmlv -i -r 100-500 footage.mlv #From Frame 100 to 500.
```

The characters 's' and 'e' can be used instead of numbers. They represent "start frame" and "end frame".

If you just want a fast preview of the work convmlv is doing, you can develop a single frame very quickly:

```
convmlv -i -r 105 footage.mlv #Develop frame 105!
```

DO NOT try to reuse DNG sequences when using frame ranges! It will break!

5 RAW Development

5.1 General

convmlv relies on dcraw, a powerful RAW development tool, and as such inherits its features! I'll go through them in detail here:

- Demosaicing, -d: RAW images represent data the sensor spits out. As such, it's neccessary to process this data. Algorithms that do this are called demosaicing

algorithms, and you have choices! Lower numbers are faster, higher numbers are higher quality. For example, to generate the best quality compressd images, specify:

```
convmlv -i -d 3 footage.mlv #High quality demosaicing.
```

- Four Color Mode, -r: When the VNG and AHD demosaicing modes get strange, this option tends to fix things.
- Highlight Reconstruction, -H: Highlights are values too bright for detail. A mode of 2 will attempt to fix them up a bit, while modes of 3 to 9 tries to regain the detail using varying color tones. A mode of 1 will let colored highlights through; this will usually look nasty unless you tune the Saturation Point (see below). Mode 2 is almost always the best option.

5.2 Noise Reduction

convmlv includes no less than 6 ways (including Darkframe Subtraction below) to reduce noise in your footage! See 'convmlv -h' for a very thorough explanation of each option. Combining them will either condemn you or unblind you :) .

- Chroma Smooth, -c: Color noise can often be twarted just by blurring the color channels. Numbers from 0 to 3 make this blur stronger. *Note this only works on MLV input footage.*
- Wavelet Denoise, -n: RAW processing is a great point in the pipeline to do some wavelet denoising, which reduces noise at an acceptable detail loss. A setting of 50 can take the edge off.
- Temporal Denoise, -N: This denoising engine is best played with - see convmlv -h. A good starting point is '-N 0.03-0.04'.
- HQ Denoise, -Q: This is Handbrake's denoising filter, explained here in great detail: `https://mattgadient.com/2013/06/29/in-depth-look-at-de-noising-in-handbrake-with` A good starting point is '-Q 3-2:2-3'.
- Removegrain, -O: This one is very weird. You can choose 4 combinations of 24 modes, listed here: `https://ffmpeg.org/ffmpeg-filters.html#removegrain`.

6 Color

6.1 White Balance

In convmlv, the White Balance you shot with is used by default. You can also do Auto White Balance, or ignore it entirely, with modes of the -w option:

```
convmlv -i -w 0 footage.mlv #Auto White Balance
```

0 does AWB, 1 does Camera WB, 2 ignores WB. Camera WB is default

When using AWB, it averages the white balance of 15 frames spread evenly throughout the footage. You can change the amount of frames used with the –white-speed option, if AWB seems off; this will slow it down, however:

```
convmlv -i --white-speed 50 footage.mlv #50 AWB samples.
```

6.2 Color Management

convmlv is completely color managed. What does this mean for you? When moving between software, the form your color is in is very important.

Images aren't simple. They are often stored, processed, and viewed as a result of complex transformations usually called, in applications, color management. Understanding it is required as a colourist, and encouraged for DPs and Cinematographers.

- Intro: `http://www.cambridgeincolour.com/tutorials/color-management1.htm`
- Understanding Gamma: `http://www.cambridgeincolour.com/tutorials/gamma-correction.htm`
- Color Spaces: `http://www.cambridgeincolour.com/tutorials/color-spaces.htm`
- Conversions: `http://www.cambridgeincolour.com/tutorials/color-space-conversion.htm`
- Monitor Calibration: `http://www.cambridgeincolour.com/tutorials/monitor-calibration.htm`

Long story short, convmlv uses LUTs for everything. This may change in the future, as I'm working on a project called openlut (`https://github.com/so-rose/openlut`), but for now, these LUTs are part of the convmlv installation (in color-core and color-ext folders).

The -g option lets you choose a gamma, while the -G option lets you choose a gamut. You'll notice staples like sRGB, XYZ, etc. . The fancier gamma/gamuts require you to have the 'color-ext' folder in your installation - you should have this, unless you manually omitted it.

```
convmlv -i -g 2 -G 5 footage.mlv #Cineon Log in DCI-P3 Gamut!
```

6.3 LUT Grading

LUTs are really cool. Once created in a piece of color grading software, you can apply it to any footage to replicate any style. Using 3D LUTs, every color maps to a different color!

convmlv can apply any number of 3D LUTs up to 64x64x64 resolution, in cube, 3dl, dat, and m3d formats! Simply use the -l option to do so:

```
convmlv -i -l lut1.cube -l lut2.cube footage.mlv #Two LUTs applied.
```

If you want to grade to your own gamma/gamut, I suggest using LUTCalc (https://cameramanben.gi to generate a corresponding LUT.

Currently lacking is 1D LUT support, and x65 3D LUT support. Once openlut is integrated, these problems will go away.

6.4 Fixing Saturation Point

Sometimes, dcraw gets something called the "saturation point" of your camera wrong. This will manifest itself as nasty, purple highlights when using -H 1, and will screw up highlight reconstruction (-H 3 to 9) and possibly -H 2 as well when it happens.

To fix it, lower it incrementally from 15000 until the highlights turn white. Trial and error is the easy way to do this:

```
convmlv -i -S 15000 footage.mlv #Setting the sat point.
```

If you want a more scientific solution, then develop one DNG yourself using 'dcraw -D -j -4 -T'. The largest 0 to 1 pixel value, multiplied by 2$\hat{1}$4 (14 bits), is the theoretical optimal number.

I don't find this to be a problem at all in my everyday life, because most DNGs have this data properly set. However, some don't, in which case you want to fix it!

7 Features

7.1 Dual ISO

convmlv can processing Dual ISO footage! Simply specify -u:

```
convmlv -i -u footage.mlv #Dual ISO - the easy way!
```

Dual ISO is a method of increasing the dynamic range of your shots, at the cost of shadow/highlight resolution. It can create some absolutely **stunning** footage.

7.2 Bad/Focus Pixels

On some cameras, focus pixel issues are rampant. These will be static, colored dots, that end up ruining footage. To fix it, simply specify -b:

```
convmlv -i -b footage.mlv #Fixing focus pixels.
```

Note that, due to a dcraw bug, processing Dual ISO footage won't remove all the focus pixels. I suggest you use MLVFS (see the tip) in conjunction with convmlv, which solves this issue.

In the same vein, but still different, is the issue of bad camera pixels. As your camera ages, pixels will die; as such, you want to interpolate around them. This must be done manually, as described here: `http://www.dl-c.com/board/viewtopic.php?f=4&t=686`.

You end up with a .badpixels file, which you want to apply to your footage to clean these dead pixels up. This is simple to do:

```
convmlv -i -a mycamera.badpixels footage.mlv #Fix bad pixels.
```

You can freely combine the focus pixel fix with or without the bad pixels fix.

7.3 Darkframe Subtraction

In reducing noise, you can do a very sneaky technique called darkframe subtraction. This does wonders for removing shadow noise, and generally cleaning footage up a bunch.

1. Dark Footage: With your lens cap on, using the same settings as your footage, take 5 seconds of dark RAW footage.
2. Use Dark Footage: Specify -F, then the path to the dark footage, to automatically average the footage + reduce noise! Example:

   ```
   convmlv -i -F ./dark.mlv footage.mlv #Using dark.mlv as darkframe.
   ```

For speed & convenience, **you want to create** a darkframe from your dark footage. convmlv can do this for you:

```
convmlv -R f2-8.iso1600 dark.mlv #Makes "f2-8.iso1600.darkframe"f.
```

To use the averaged file as an averaged file, make sure it has the .darkframe extension, and then just specify it in the -F path:

```
convmlv -F f2-8.iso1600.darkframe footage.mlv #Makes f2-8.iso1600.darkframe.
```

If you're clever, you'll make a library of darkframes for all sorts of shooting situations, so that you'll never be without this powerful noise reduction technique.

7.4 Misc. Filters

convmlv comes with a few other filters (from FFMPEG), for convenience:

- Sharpen, -A: This filter lets you sharpen or blur your footage. A medium sharpen is something like 5:1:3:0, but see 'convmlv -h' for the full settings.
- Deshake, -D: This filter automatically stabilizes the footage, with mixed results. Typically, you want to crop/scale the output afterwards.

Let me know if you want more!

7.5 Custom Paths

By default, all developed footage will be placed in folders in './raw-conv'. You can change this:

```
convmlv -i -o differentoutdir footage.mlv #Change output directory.
```

You can also supply a custom path for any or all of the dependencies, if you wish. See convmlv -h for the full list; here's an example:

```
convmlv --dcraw ./new-dcraw-binary footage.mlv #Use custom dcraw.
```

8 Config Files

Everything that can be typed at the command line, can also be specified in config files. This means you don't have to type it, and is essential for production (for example, you can make a "production config", input all your footage, and step away while it all develops for you!). They're powerful, but also more complex than just typing in options.

As always, see convmlv -h for more info! There are also numerous examples bundled with the release, in "configs".

8.1 Specifying Options

There are three ways to specify options in convmlv. Each overwrites the last one:

- Global Config: Found in *homedir*/convmlv.conf, if it exists.
- Local Config: Found at the path specified by -C or –config.
- Command Line Options: Passed when calling convmlv.

So, if I specified 'GAMMA 2' in the Global, but then specified '-g 3' on the command line, then the output gamma would be set to mode 3.

8.2 Syntax

All valid config files begin the same way:

```
CONFIG\_NAME <name>
```

Comments start with a #. They are ignored.

You specify options like so:

```
<VARNAME> <VALUE>
```

If VARNAME is a true/false flag, then specifying it is enough - there's no VALUE.

Each command line option has a corresponding VARNAME. The VALUE is what you'd type into the terminal.

8.3 File-Specific Blocks

You can even specify options for specific files! That is to say, these options will only be triggered if the name of the file you're developing matches. This lets you essentially program how all the footage of your production is developed, all in one config file.

This only works for LOCAL config files. The way to use this feature is the File-Specific Block:

```
File-Specific Block: Config per specific filename.

/ <TRUNCATED INPUTNAME>
    ...options here will only be used for the specified filename.
*
```

You must use the truncated (no .mlv or .raw) input name after the /. Nested blocks will fail.

9 Tips and Tricks

Using this in real situations, I've come up with some time savers you might find useful!

9.1 Batch Processing

Batch processing files can be done with a *. For example, to process all .MLV files in a directory:

```
convmlv -i *.MLV #Develops all MLV files in the current directory.
```

Combine this with config files and the file-specific blocks, and you may only have to run a single command to develop an entire project of footage in a completely customized way.

9.2 Develop Anywhere Without Dependency Error

Something you should absolutely have in your Global Config is RES_PATH. This is the folder where all your manual dependencies (mlv-dump, etc.) are looked for. By default it's the current directory, but point it at the folder containing mlv-dump, raw2dng, etc. in your Global Config, and you'll be able to develop MLVs anywhere on the system! F

9.3 Thread Count

Though the script auto-detects available threads, you can of course customize this:

```
convmlv --threads 2 footage.mlv #Develop with 2 threads.
```

9.4 Read MLV Properties

convmlv can read settings from any MLV file without developing it, like camera name, aperture, ISO, White Balance, focal length, etc. using the -q option:

```
convmlv -q footage.mlv #Develop with 2 threads.
```

9.5 Use MLVFS instead of mlv_dump for Speed and Quality

MLVFS allows you to "mount" an MLV file, gaining "instant" access to the DNG files with live (website) configurable options! I won't go much more into it; you can get it here: `http://www.magiclantern.fm/forum/index.php?topic=9335.0`.

To use it with convmlv, first mount the MLVs as normal. Simply use the mounted path to the folder containing a DNG sequence as your input, and watch it skip MLV conversion entirely (it's done fast, and on the fly), jumping straight to the EXR/ProRes creation!

In my limited tests, the quality of footage has been very, very good. Note that some features, like chroma smoothing, need to be manually triggered in MLVFS; convmlv relies on mlv-dump there, which is no longer being used! Also, darkframe averaging is currently impossible this way.

In the future I'll integrate it more easily with convmlv. Stay tuned!

9.6 Stay Up to Date

Make sure to check the forum post, and/or the repository, for updates! They fix bugs, add features, and more!

9.7 I'm Here to Help!

If you have any bug reports, feedback or feature requests, I'd be happy to hear it! Just reply to the thread at `http://www.magiclantern.fm/forum/index.php?topic=16799.0`, or PM me on the forums, or write a comment, or create an issue on github!