An Introduction to
FFmpeg, Timelapse and Fulldome Video Production,
Color Grading, Audio Processing, Panasonic LUMIX GH5S,
Image Processing and Astronomy Software
by
Michael Koch, astroelectronic@t-online.de

Version from March 14, 2020
Contents

1 Introduction to FFmpeg.................................................................5
  1.1 What can be done with FFmpeg? ..........................................7
  1.2 If FFmpeg has no graphical user interface, how do we use it? ...8
  1.3 The first example...............................................................9
  1.4 Using variables...................................................................10

2 FFmpeg in detail........................................................................11
  2.1 Convert from one video format to another video format .......11
  2.2 Fit timelapse length to music length....................................12
  2.3 Timelapse or slideshow from many images, with crossfading...13
  2.4 Slideshow with different durations.....................................14
  2.5 Extract many images from a video....................................15
  2.6 Extract the last frame from a video....................................15
  2.7 Modify brightness, contrast, saturation, gamma and hue...16
  2.8 Strong contrast enhancement............................................17
  2.9 Inverting a video or image (make a negative)...................18
  2.10 Correcting the color channels.........................................19
  2.11 Colorhold.........................................................................20
  2.12 Atmospheric dispersion correction.................................21
  2.13 Amplify filter....................................................................21
  2.14 Extract a time segment from a video.................................22
  2.15 Trim filter........................................................................22
  2.16 Tpad filter, add a few seconds black at the beginning or end.23
  2.17 Extract the last 30 seconds of a video..............................24
  2.18 Fade-in and fade-out..........................................................25
  2.19 Crossfading......................................................................26
  2.20 Crop a video......................................................................27
  2.21 Changing the speed, slow motion and timelapse...............28
  2.22 Slow motion or timelapse only for a segment of the video...29
  2.23 Time Remapping...............................................................30
  2.24 Insert a text which is visible for the whole duration........32
  2.25 Slowly fade a text in and out............................................32
  2.26 Show a running clock in the video....................................34
  2.27 Generation of curved text for fulldome projection...........36

2.28 Combine multiple videos with concat demuxer...............39
2.29 Combine multiple videos with concat filter.....................40
2.30 Switch between two cameras, using audio from camera1....41
2.31 Stack videos side by side (or on top of each other)...........42
2.32 Horizontal and vertical flipping.......................................42
2.33 Stack four videos to a 2x2 mosaic....................................43
2.34 Blink comparator..............................................................44
2.35 Animated GIF.....................................................................45
2.36 Replace one frame in a video by another........................46
2.37 Blend filter........................................................................47
2.38 Subtracting a darkframe..................................................48
2.39 Gradation curves and vignetting.......................................49
2.40 Color grading with color look-up tables, full workflow.....50
2.41 Color grading with color look-up tables, simplified workflow.52
2.42 Size of color-look-up tables.................................53
2.43 Histogram.......................................................................53
2.44 Lagfun filter.................................................................54
2.45 Deblock filter.................................................................55
2.46 Gradfun filter.................................................................55
2.47 Dilation filter.................................................................55
2.48 V360 filter for rotation of equirectangular 360° videos......56
2.49 Equirectangular images of the night sky.........................58
2.50 Remap a fisheye video to an equirectangular video...........59
2.51 Remap an equirectangular video to a fisheye video..........63
2.52 Remap an equirectangular video to a "Little planet" video....64
2.53 Remap an equirectangular video to a "Mirror sphere" video..66
2.54 Shift the viewing direction in a fisheye image or video.....68
2.55 Stitching together double-fisheye videos........................71
2.56 Remove vertical stitching artefacts................................73
2.57 Preprocessing a flat video for fulldome projection.............76
2.58 Rotating earth or planet....................................................78
2.59 Black hole simulation with remap filter............................79
2.60 Wormhole simulation........................................................87
Introduction to FFmpeg

FFmpeg is an open-source software and available for Linux, Windows and OS-X. It's a very powerful command line tool and has no graphic user interface.

Main project website:  www.ffmpeg.org
Download site for the Windows and OS-X versions:  https://ffmpeg.zeranoe.com/builds/

How to install the Windows version:
Choose 64-bit or 32-bit, "Static", and click on "Download Build". Open the ZIP file, open the "bin" folder and copy ffmpeg.exe, ffprobe.exe and ffplay.exe to a new folder, for example c:\ffmpeg\ That's all.

ffmpeg.exe is the very powerful software for manipulating videos.
ffprobe.exe is a program for viewing the properties of videos, pictures or audio files. It's useful for troubleshooting.
ffplay.exe is a video player. In most cases we don't need it if we use the VLC player instead.
It's also a good idea to save the file doc\ffmpeg-all.html somewhere on your own computer. This file contains the full documentation for FFmpeg. The most important chapters are "Audio Filters" and "Video Filters".

Additional to the official documentation, there are also two wikis available:
https://trac.ffmpeg.org/wiki

All examples in this document were tested only with Windows 7.
What are the pros and cons of FFmpeg, compared to other programs for video manipulation?

- Very powerful capabilities.
- It’s an active project, updates come almost daily.
- Conversion from almost all formats to almost all other formats.
- No restrictions for video size (width * height), as in some other programs.
- There is a mailing list where you can ask questions in English. Before you ask, make sure that the problem is still present in the latest FFmpeg version. Always include the complete FFmpeg console output, because it contains many useful informations.
- FFmpeg is a command line program and has no graphical user interface. At first glimpse this sounds like a big drawback. But it’s a nice idea to have all commands in a batch file, because later you can easily make modifications at all arbitrary steps in the workflow. Just modify the batch file and execute it again.
- You will need some time for learning FFmpeg.
- Unfortunately the documentation is the weak point of the project, and many times I wished that the documentation contained more informations and especially more examples.
1.1 What can be done with FFmpeg?

- Convert a video, picture or sound from one format to another.
- Make a (timelapse) video from many pictures.
- Make many pictures from a video.
- Cut segments from a video, for example remove the beginning or the end.
- Add or remove audio, or modify the audio volume.
- Change the video size (width * height).
- Enlarge parts of the video or cut away the borders, for example make a rectangular video square.
- Change the speed, timelapse or slow motion.
- Rotate, mirror or flip.
- Add texts to a video.
- Correct brightness, contrast, gamma, saturation, color temperature, also with look-up-tables.
- Masking, for example superimpose a circular mask over a video.
- Fade-in, fade-out and crossfade for video and audio.
- Morphing, for example curved texts for fulldome projection, or simulation of curved spacetime near block holes.
- Stabilizing of shaky videos
- Deflicker, for reducing brightness steps in timelapse.
- Change the video compression, to make the video smaller.
- and many more interesting things...
- It’s always a good idea to begin with a working example, and then modify it step by step. I hope that the examples in this book are a good starting point for you.
1.2 If FFmpeg has no graphical user interface, how do we use it?

There are two possible ways:

1. Open a console window   All_Programs / Accessories / Command_Promt   (german) Alle_Programme / Zubehör / Eingabeaufrorderung
   Another way to open the console window is to press WINDOW R and then enter "cmd".  
   But this is not recommended, because in many cases the command lines are quite long.

2. The recommended way is to write a batch file which contains the FFmpeg command line.
   - A batch file has the extension *.bat and can be created and modified with any text editor. When you save a batch file with Notepad, make sure that you choose "all files" and save it as *.bat and don't choose "text files", because then the extension would be *.bat.txt (Hint: Configure the explorer so that all file extensions are visible!)
   - You can edit a batch file by right clicking on it, and then choose "Edit".
   - You can execute a batch file by double clicking on the icon or filename.
   - It's recommended to begin with a working example, and then modify it step by step. Make small steps and always make a test run. If it fails, go back to the last working version.
   - The % character has a special meaning inside a batch file. If you need a one % character in the FFmpeg command line, you must replace it in the batch file by two %% characters.
   - It's recommended to insert the command "pause" at the end of the batch file. This means the batch file waits for a keypress. Without this command, the console window would close immediately when FFmpeg has finished, and you wouldn't see if there were any error messages.
   - With the command "set" you can define variables in the batch file.
   - With the command "rem" you can insert comments, so that you later understand how the batch file works. Comments can also begin with :: in the same line as a command. Everything from :: to the end of the line is a comment.
   - If the command line becomes too long, you can insert a ^ character at the end of the line and continue in the next line.
   - How to copy and paste the content of the console window: Right click in the title of the Command_Prompt window, Edit -> Select_All, then Edit -> Copy, then paste the content with ctrl-v somewhere else.
   - (german) Wenn man den Inhalt des CMD-Fensters kopieren möchte, geht man so vor: Rechtsklick auf die Titelleiste des Fensters, Bearbeiten --> Alles auswählen, dann Bearbeiten --> Kopieren, dann mit Control-V irgendwo anders einfügen.
1.3 The first example

This is a simple batch file:

```
rem  A simple batch file for making a video from many pictures

C:\ffmpeg\ffmpeg -framerate 5 -start_number 3551 -i IMG_%%4d.jpg -i birds.mp3 ^
-shortest -codec:v mpeg4 -q:v 2 out.mp4

pause :: wait for a keypress
```

What's the meaning of the parts?

```
rem  A simple ... This is a comment
C:\ffmpeg\ffmpeg This is the path to ffmpeg.exe
-framerate 5 This defines how fast the pictures are read in, in this case 5 pictures per second.
-start_number 3551 This is the number of the first picture, in this case 3551
-i IMG_%%4d.jpg This is the filename of the input pictures. The term %%4d stands for a 4-digit number. The filename of the first picture is IMG_3551.jpg and the number will be increased by 1, until no more picture is found. For 3-digit numbers you would write %%3d instead.
-i birds.mp3 This is the second input file, in this case an audio file.
^ If the command line becomes too long, you can break it with the ^ character and continue in the next line. FFmpeg will get the whole line without the ^ character.
-shortest This option means that the length of the output video is determined by the shortest of the two input files.
-codec:v mpeg4 This option means that a MPEG4 video will be produced.
-q:v 2 This is an option for the quality of the output video. 0 is best quality, 2 is normal, 9 is strongest compression.
out.mp4 Filename of the output video
pause This command waits for a keypress, so that you have a chance to see any error messages before the console window closes.
:: wait for ... Everything right of :: is a comment.
```

Important: Options are always written before the file they refer to. The options "-framerate 5" and "-start_number 3551" refer to the first input file "IMG_%%4d.jpg".

The second input file "birds.mp3" doesn't have any options in this case.

The options "-shortest -codec:v mpeg4 -q:v 2" refer to the output video "out.mp4".
1.4 Using variables

Using variables is much better programming style. This batch file has exactly the same function as the first example:

```bash
rem A simple batch file for making a video from many pictures

set "FF=c:\ffmpeg\ffmpeg"    :: Path to ffmpeg.exe
set "FR=5"                   :: Framerate for reading in the pictures (Frames per second)
set "SN=3551"                :: Number of the first picture
set "IN=IMG_%%4d.jpg"        :: Filename of the pictures
set "AUDIO=birds.mp3"        :: Audio filename
set "QU=2"                   :: MP4 Quality, 0 is best quality, 2 is normal, 9 is strongest compression
set "OUT=out.mp4"            :: Output filename

%FF% -framerate %FR% -start_number %SN% -i %IN% -i %AUDIO% -shortest -codec:v mpeg4 -q:v %QU% %OUT%

pause                        :: wait for a keypress
```

This is much clearer, because each variable is written in a new line and has its own comment.

It's recommended to use capital letters for the variables, so that you can easily distinguish them from command line options.

All variable names are allowed, but don't use special characters like ÄÖÜ.

You can copy a batch file and save it under a new name for a new project. Then you must only set the variables, so that they fit to the new project. There is no need to modify (or even understand) the command line.

Why are the variable definitions written in " " quotation marks? This is only necessary if you want to add a comment in the same line. Without comments, the quotation marks are unnecessary.
2 FFmpeg in detail

2.1 Convert from one video format to another video format

Some examples for format conversion:

```
rem Convert any input format to any output format
ffmpeg -i anyinput.xxx anyoutput.xxx

rem Convert MP4 to mov
ffmpeg -i in.mp4 -acodec copy -vcodec copy -f mov out.mov

rem Convert mov to MP4
ffmpeg -i in.mov -acodec copy -vcodec copy out.mp4

rem Convert mov to MP4 using h265 compression, default preset is medium, default crf is 28
ffmpeg -i in.mov -c:v libx265 -preset slow -crf 25 -acodec copy out.mp4
```
2.2 Fit timelapse length to music length

How to give a timelapse video exactly the same length as the music?

We don't want to cut off the end of the music, and we don't want to hear silence at the end of the timelapse video.

The solution is to adjust the framerate, so that the length of the timelapse becomes equal to the music length.

Framerate = Number_of_images / Time_in_seconds

In this example we have 30 images and the music is 20 seconds long, so that the framerate must be 1.5.

<table>
<thead>
<tr>
<th>rem A simple batch file for combining many images to a video</th>
</tr>
</thead>
<tbody>
<tr>
<td>set &quot;FF=c:\ffmpeg\ffmpeg&quot; :: Path to ffmpeg.exe</td>
</tr>
<tr>
<td>set &quot;FR=1.5&quot; :: Framerate for reading in the images (frames per second)</td>
</tr>
<tr>
<td>set &quot;RATE=30&quot; :: Output framerate</td>
</tr>
<tr>
<td>set &quot;SN=3551&quot; :: Number of the first image</td>
</tr>
<tr>
<td>set &quot;IN=IMG_%%4d.jpg&quot; :: Filename of the images</td>
</tr>
<tr>
<td>set &quot;AUDIO=birds.mp3&quot; :: Audio filename</td>
</tr>
<tr>
<td>set &quot;QU=2&quot; :: MP4 Quality, 0 is best Quality, 2 is normal, 9 is strongest compression</td>
</tr>
<tr>
<td>set &quot;OUT=out.mp4&quot; :: Output file</td>
</tr>
</tbody>
</table>

%FF% -framerate %FR% -start_number %SN% -i %IN% -i %AUDIO% -r %RATE% -shortest -codec:v mpeg4 -q:v %QU% %OUT%

pause :: Wait for a keypress

In this example we have two different framerates, which have different purpose:

- -framerate %FR% this is the framerate for reading in the images
- -r %RATE% this is the framerate of the output video.

These two framerates are totally independent from each other, and can be different. If the images are read in slower than the output framerate, FFmpeg will automatically duplicate images. If the images are read in faster, then FFmpeg will automatically skip images.
2.3 Timelapse or slideshow from many images, with crossfading

rem Make a timeelapse or slideshow from many images, with crossfading

set "FF=c:\ffmpeg\ffmpeg" :: Path to ffmpeg.exe
set "RATE=30" :: Output framerate
set "SN=3551" :: Number of first image
set "IN=IMG_%%4d.jpg" :: Filename of the images
set "QU=2" :: MP4 Quality, 0 is best Quality, 2 is normal, 9 is strongest compression
set "OUT=out.mp4" :: Output file

:: A is the Duration how long each image is shown (without crossfading), here 1.0 sec
:: B is the Duration of the crossfade, here 0.5 sec
set "C=3" :: set C = (A+B)/B (you must calculate this integer manually)
set "D=2" :: set D = 1/B (you must calculate this floating point value manually)

%FF% -start_number %SN% -i %IN% ^
-vf zoompan=d=%C%:fps=%D%,framerate=%RATE%:interp_start=0:interp_end=255:scene=100 ^
-codec:v mpeg4 -q:v %QU% %OUT%

pause :: Wait for a keypress

Inside the video filter (beginning with -vf) we have in this example two filters, which are applied one after the other. The first is "zoompan" and the second is "framerate".

You must calculate the variables C and D manually, because there are no expressions allowed inside the "zoompan" filter.

Detailed explanations for this example:

-vf zoompan=d=%C%:fps=%D%,framerate=%RATE%:interp_start=0:interp_end=255:scene=100

In this example two video filters are applied consecutively, separated by a (,) comma.
1. "zoompan", with the parameters "d" and "fps"
2. "framerate", with the parameters "fps" (which can be omitted in this case), "interp_start", "interp_end", and "scene"
The zoompan filter is here not used for zooming in, but for duplicating the frames and passing them to the next filter with a certain framerate.
"d" specifies how often each frame is repeated.
"fps" is the output framerate of this filter.

The framerate filter can calculate intermediate images between consecutive images. This is not a motion interpolation but a crossfade.
"fps" is the output framerate. It's not required to explicitly write this parameter; you could also write framerate=fps=%RATE%:...

The remaining three parameters "interp_start", "interp_end", and "scene" specify, when interpolation is active and when not. With those values that I used (0, 255, 100), interpolation is always active.

These two filters together produce a video in which each image is shown for a certain duration, followed by a crossfade to the next image which also has a certain duration. Both durations can be chosen freely, these are the values A and B in the comments. From these values you must manually calculate the variables C and D, which are used in the command line. I haven't yet found a way to make this calculation automatically. It's possible to make calculations in the batch file, but this works only with integer precision.

If you omit the zoompan filter and use only the framerate filter, the next crossfade would immediately follow when the previous has ended. With other words: You always have a crossfade and there is no time where the image is shown without crossfade. That's why we use the trick with the zoompan filter. Now it's still the case that one crossfade follows immediately on the previous one, but now we have crossfades between identical images, because the images were duplicated by the zoompan filter. A crossfade between identical images isn't visible, of course.

### 2.4 Slideshow with different durations

```
c:\ffmpeg\ffmpeg -i img%4d.jpg -vf zoompan=d=25+50*eq(in,3)+100*eq(in,5) out.mp4
```

In this example each frame is shown one second (25 frames), except the 4th image which is shown 3 seconds (25+50 frames) and the 6th image which is shown 5 seconds (25+100 frames). Please note that the image numbering starts with 0, if not specified differently with "-start_number".
2.5 Extract many images from a video

```
rem Extract many images from a video

c:\ffmpeg\ffmpeg -i in.mp4 -vf fps=0.2 -y image%%4d.jpg
pause :: Wait for a keypress
```

This batch file reads the file in.mp4 and produces images with the filenames
image0000.jpg, image0001.jpg, image0002.jpg, and so on.
-vf fps=0.2 this specifies that images are extracted with a framerate of 0.2, which means one frame every 5 seconds.
-\( y \) this option means that FFmpeg will overwrite any output files that already exist with the same filename, without asking. If you omit this option, FFmpeg would ask before overwriting a file.

This example will extract each \( n \)-th frame from a video, beginning with the 0-th frame:

```
set "FF=c:\ffmpeg\ffmpeg" :: Path to FFmpeg
set "IN=video.mp4" :: Input video
set "STEP=10" :: Step width
set "OUT=image%%4d.jpg" :: Output images filename

%FF% -i %IN% -vf framestep=%STEP% -y %OUT%
```

2.6 Extract the last frame from a video

```
c:\ffmpeg\ffmpeg -sseof -0.2 -i in.mp4 -q:v 1 -update 1 last_frame.jpg
pause
```
2.7 Modify brightness, contrast, saturation, gamma and hue

rem Modify brightness, contrast, saturation, gamma and hue

set "FF=c:\ffmpeg\ffmpeg" :: Path to FFmpeg
set "INPUT=Panoview.mp4" :: Input video
set "OUTPUT=out.mp4" :: Output video
set "CONTRAST=1.0" :: Contrast in range -1000 to 1000, normal is 1.0
set "BRIGHT=0.0" :: Brightness in range -1.0 bis 1.0, normal is 0.0
set "SATUR=1.2" :: Saturation in range 0.0 bis 3.0, normal is 1.0
set "GAMMA=1.0" :: Gamma in range 0.1 to 10.0, normal is 1.0
set "HUE=20" :: Color correction (hue), negative shifts towards red and positive towards blue, normal is 0
:: Typical values are in the -30...+30 range
set "QU=2" :: MP4 Quality, 0 is best Quality, 2 is normal, 9 is strongest compression

%FF% -i %INPUT% -vf hue=h=%HUE%,eq=contrast=%CONTRAST%:brightness=%BRIGHT%:saturation=%SATUR%:gamma=%GAMMA% ^
-q:v %QU% -codec:v mpeg4 %OUTPUT%
pause

-vf is the command for "Video Filter". There are many different filters, see chapter "Video Filter" in the FFmpeg documentation.

In this case we use two filters, which are separated by a (,) comma.

- The first filter is "hue" and makes a rotation of the color circle.
- The second filter is "eq" and adjusts contrast, brightness, saturation and gamma.

From a mathematically point of view these functions work as follows:

- Contrast is a multiplication by a constant. Please note that what contrast does is scale the distance of a pixel's value from the median value i.e. 128 for a 8-bit input. So, if a pixel channel has a value of 100, then a contrast of 3 results in a value of 128 + 3*(100-128) = 44.
- Brightness is the addition of a constant.
- Saturation is difficult to describe mathematically. Setting saturation to 0 would produce a black and white video.
- Gamma is a nonlinear distortion of the transfer function. When you increase the gamma value, details in dark areas become better visible.

It doesn't care in which order you list the parameters in the command line. They are always executed in the order contrast -> brightness -> gamma.
## 2.8 Strong contrast enhancement

There are several filters that can be used for a strong contrast enhancement:

<table>
<thead>
<tr>
<th>Filter</th>
<th>Example for strong contrast enhancement by a factor 5: Input range [0.1 ... 0.3], Output range [0.0 ... 1.0], Output = 5 * (Input - 0.1)</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>eq</td>
<td>eq=brightness=0.3, eq=contrast=5</td>
<td>The pivot point for contrast is always at 0.5 which means you have to adjust both brightness and contrast. The eq filter must be invoked two times, because we need first the brightness adjustment and then the contrast adjustment.</td>
</tr>
<tr>
<td>eq</td>
<td>eq=brightness=0.3:contrast=5</td>
<td>This doesn't work as expected because the eq filter is invoked only one time, which means the order is contrast before brightness and that's the wrong order in this case.</td>
</tr>
<tr>
<td>eq</td>
<td>eq=brightness=1.5:contrast=5</td>
<td>This doesn't work because the brightness value isn't in the allowed range [-1 ... +1]</td>
</tr>
<tr>
<td>geq</td>
<td>For 8-bit video: geq=lum='5*(lum(X,Y)-25.6)':cr='cr(X,Y)':cb='cb(X,Y)' For 8-bit video with limiter: geq=lum='clip(5*(lum(X,Y)-25.6),0,255)':cr='cr(X,Y)':cb='cb(X,Y)'</td>
<td>Not recommended because it's slow, has no built-in limiter and the function must be different for 8-bit and 10-bit videos.</td>
</tr>
<tr>
<td>colorlevels</td>
<td>colorlevels=rimin=0.1:gimin=0.1:bimin=0.1:rimax=0.3:ɡimax=0.3:ɡimax=0.3:ɡimax=0.3</td>
<td>Best method because you can directly set the black and white points. The only drawback is that you have to write the same values three times, but that can be done with variables in the batch file.</td>
</tr>
<tr>
<td>curves</td>
<td>curves=all='0/0 0.1/0 0.3/1 1/1'</td>
<td>This is a nonlinear transfer function because it uses a smooth curve through the defined points.</td>
</tr>
</tbody>
</table>
2.9 Inverting a video or image (make a negative)

There are several methods for inverting (which means black becomes white, and vice versa):

<table>
<thead>
<tr>
<th>Filter</th>
<th>Example</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>eq</td>
<td>eq=contrast=-1</td>
<td>This changes only bright to dark and vice versa, but keeps the colors as they are.</td>
</tr>
<tr>
<td>negate</td>
<td>negate</td>
<td>This negates all channels and changes the colors to their complementary colors.</td>
</tr>
<tr>
<td>geq</td>
<td>geq=r='255-r(X,Y)' : g='255-g(X,Y)' : b='255-b(X,Y)'</td>
<td>Same result as negate. Can also be used if only one or two channels are to be inverted. The functions must be different for 10-bit videos.</td>
</tr>
</tbody>
</table>
2.10 Correcting the color channels

Color corrections can be made with the "colorchannelmixer" filter. In this example we amplify the red channel and attenuate the blue channel. The values must be in the range [-2 ... +2].

```plaintext
set "FF=c:\ffmpeg\ffmpeg" :: Path to FFmpeg
set "INPUT=7Z7A2065.MOV" :: Input video
set "OUTPUT=out.mp4" :: Output video
set "R=1.2" :: Factor for red channel
set "G=1.0" :: Factor for green channel
set "B=0.8" :: Factor for blue channel
set "QU=2" :: MP4 Quality, 0 ist best Quality, 2 is normal, 9 is strongest compression

%FF% -i %INPUT% -vf colorchannelmixer=rr=%R%:gg=%G%:bb=%B% -q:v %QU% -codec:v mpeg4 -y %OUTPUT%
```

Exchange red and green components:

```plaintext
set "FF=c:\ffmpeg\ffmpeg" :: Path to FFmpeg
set "INPUT=7Z7A2065.MOV" :: Input video
set "OUTPUT=out.mp4" :: Output video

%FF% -i %INPUT% -vf colorchannelmixer=0:1:0:0:1:0:0:0:0:0:0:1:0 -y %OUTPUT%
```
2.11 Colorhold

This video filter removes all color informations except for one certain color.

There are three parameters:

"color" is the color to be preserved, can be specified by name or by RGB values, for example "orange" can be replaced by #FFA500

"similarity" is a percentage, 0.01 means only the specified color is preserved, 1.0 means all colors are preserved.

"blend" is a percentage, 0.0 makes pixels fully gray, higher values result in more preserved color.

This example preserves only colors from yellow to orange to light brown:

c:\ffmpeg\ffmpeg -i 7Z7A2027.jpg -filter_complex split[1][2];[1]colorhold="orange":0.5:0[3][2][3]hstack -y out.jpg

Output of this example:
2.12  Atmospheric dispersion correction

It's possible to shift the RGB channels with respect to each other:

```
set "FF=c://ffmpeg/ffmpeg" :: Path to FFmpeg ::
set "IN=P1000479.mov"      :: Input video
set "OUT=out.mp4"          :: Output video
                  ::
set "RV=5"                 :: Vertical shift of red channel
set "BV=-5"                :: Vertical shift of blue channel

%FF% -i %IN% -lavfi "rgbashift=rv=%RV%:bv=%BV%" -y %OUT%
pause
```

2.13  Amplify filfer

The "amplify" filter amplifies differences between adjacent frames. Good for motion detection, but it's also sensitive to noise.
2.14 Extract a time segment from a video

When you have a fisheye camera pointing upwards, it's unavoidable that you are visible in the video at the beginning and the end, because you must start and stop the camera. That means we must cut off the beginning and the end.

```rem Extract a time segment from a video
set "FF=c:\ffmpeg\ffmpeg" :: Path to FFmpeg
set "INPUT=PanoView.mp4" :: Input video
set "OUTPUT=out.mp4" :: Output video
set "START=2.0" :: Start time in seconds
set "LENGTH=3.0" :: Length of the segment in seconds

%FF% -ss %START% -t %LENGTH% -i %INPUT% -c copy %OUTPUT%
``` pause

The arguments for -ss and -t can also be specified in hours, minutes and seconds:

- 1:20 = 1 minute, 20 seconds
- 1:10:30 = 1 hour, 10 minutes, 30 seconds

Instead of the length it's also possible to specify the end time with the -to option.

If you want to save the output video with exactly the same quality as the input video (without re-encoding), then use the -c copy option. In this case it makes no sense to specify the output video quality.

```c:\ffmpeg\ffmpeg -ss 5 -i input.mov -t 10 -c copy output.mov
``` pause

The same thing can also be done with the "trim" filter.
2.15   Trim filter

Drop everything except the second minute of input:

```
-i INPUT -vf trim=60:120 OUTPUT
```

Keep only the first second:

```
-i INPUT -i INPUT -vf trim=duration=1 OUTPUT
```

2.16   Tpad filter, add a few seconds black at the beginning or end

Method 1, using the "tpad" filter:

```
set "FF=c:\ffmpeg\ffmpeg" :: Path to FFmpeg
set "IN=my_video.mp4"     :: Input video
set "DUR=3"               :: Duration in seconds
set "OUT=out.mp4"         :: Output video

%FF% -i %IN% -vf tpad=start_duration=%DUR% %OUT% pause
```

The "tpad" filter inserts frames at the beginning or at the end of a video. These frames contain either a uniform color or a copy of the first or last frame. The default color is black.

Method 2, using the concat filter:

```
set "FF=c:\ffmpeg\ffmpeg" :: Path to FFmpeg
set "IN=my_video.mp4"     :: Input video
set "DUR=3"               :: Duration in seconds
set "OUT=out.mp4"         :: Output video

%FF% -i %IN% -an -filter_complex 'color=black:duration=%DUR%[black];[black][0:0]concat=n=2:v=1:a=0[v]' -map [v] %OUT% pause
```
2.17 Extract the last 30 seconds of a video

When I make real-time videos of meteors, I let the Panasonic LUMIX GH5S camera record continuously. When I see a meteor, I speak to the soundtrack in which part of the sky I've seen it, and after about 10 seconds I press the REC button to stop the recording, and immediately start a new recording. That means after downloading the videos to the computer, meteors are always at the end of the videos. There is no need to watch the videos in full length (that would be boring). This batch file extracts the last 30 seconds of the video which is drag-and-dropped over it, and for the output filename the string "P1" is replaced by "CUT" (e.g. P1000336.MOV becomes CUT000336.MOV). It's lossless because the "-c copy" option is used.

```
set INPUT=%1
set OUTPUT=%INPUT:P1=CUT%
c:\ffmpeg\ffmpeg -sseof -30 -i %INPUT% -c copy %OUTPUT%
pause
```

This batch file (for Windows 7) does the same thing for all P1*.MOV files in the current folder:

```
for %f in (P1*.MOV) do call :for_body %f
goto :the_end

:for_body
    set INPUT=%1
    set OUTPUT=%INPUT:P1=CUT%
c:\ffmpeg\ffmpeg -sseof -30 -i %INPUT% -c copy -y %OUTPUT%
exit /b

:the_end
pause
```
2.18     Fade-in and fade-out

Fade-in and fade-out for a video of known length (only for video, not for audio). Here the times are expressed in frames:

```
ffmpeg -i input.mp4 -vf 'fade=in:0:30,fade=out:9650:30' output.mp4
```

Fade-in and fade-out of a video of known length (both video and audio). Here the times are in seconds:

```
ffmpeg -i input.mp4 -vf 'fade=in:st=0:f=1,fade=out:st=32:d=1' -af 'afade=in:st=0:d=1,afade=out:st=32:d=1' output.mp4
```

This is a workaround for fade in/out a video with unknown duration:

```
ffmpeg ffmpeg -i input.mp4 -sseof -1 -copyts -i input.mp4 -filter_complex "[1]fade=out:0:30[t];[0][t]overlay,fade=in:0:30[v]; anullsrc,atrim=0:2[at];[0][at]acrossfade=d=1,afade=d=1[a]",map "[v]" -map "[a]" -c:v libx264 -crf 22 -preset veryfast -shortest output.mp4
```

The trick is to feed the same input twice. From the second input only the last second is used. The timestamps are preserved. A fade-out is applied to the short second input, and then both files are combined with overlay. For audio a 2 seconds dummy with silence is created, and then crossfaded with the input audio. The -shortest option cuts the output to the same length as the input.

Another workaround for making fade-in and fade-out for audio of unknown length:

```
ffmpeg -i input.mp4 -filter_complex "afade=d=0.5, areverse, afade=d=0.5, areverse" output.mp4
```

The same thing does also work for video, but keep in mind that you need a lot of memory for the reverse filter:

```
ffmpeg -i input.mp4 -filter_complex "fade=d=0.5, reverse, fade=d=0.5, reverse" output.mp4
```

Another option is to use acrossfade with a silent track, but this works not for video because there is no crossfade filter for video:

```
ffmpeg -i input.mp4 -filter_complex "aevalsrc=0:d=0.6 [a_silence]; [0:a:0] [a_silence] acrossfade=d=0.6" output.mp4
```

A fade curves are shown on this wiki page: [https://trac.ffmpeg.org/wiki/AfadeCurves](https://trac.ffmpeg.org/wiki/AfadeCurves)
2.19 Crossfading

The different types of xfade crossfadings are shown on this wiki page:

https://trac.ffmpeg.org/wiki/Xfade
2.20 Crop a video

Cropping means to cut off the borders, and in the next step you can also set the size (width * height) of the output video:

```bash
rem Crop and set the output size
set "FF=c:\ffmpeg\ffmpeg" :: Path to FFmpeg
set "INPUT=PanoView.mp4" :: Input video
set "OUTPUT=out.mp4" :: Output video
set "CROP=1224:1224:0:0" :: Specify the visible part: Width, height, left edge, top edge
set "SIZE=800x800" :: Width and height of the output video (can be smaller or larger than the input video)
   :: Keep the width/height ratio constant, otherwise the video looks distorted,
   :: for example a circle would become an ellipse.
set "QU=2" :: MP4 Quality, 0 is best Quality, 2 is normal, 9 is strongest compression
%FF% -i %INPUT% -vf crop=%CROP% -s %SIZE% -q:v %QU% -codec:v mpeg4 %OUTPUT%

pause
```

In the crop filter you can use the variables "iw" and "ih", which are the width and height of the input video.

If the 3rd and 4th parameter (coordinates of top left corner) isn't specified, the crop will be automatically centered.

crop=iw:ih makes a centered square crop, useful for fulldome videos
crop=iw/2:ih:0 returns the left half of the input video
crop=iw/2:ih:iw/2 returns the right half of the input video
crop=iw/4:ih/4 strong enlargement by a factor 4 in the center of the video

The "pad" filter does the opposite thing, it adds paddings with a uniform color to the video.
2.21 Changing the speed, slow motion and timelapse

rem Changing the speed (slow motion or timelapse)
set "FF=c:\ffmpeg\ffmpeg" :: Path to FFmpeg
set "INPUT=PanoView.mp4" :: Input video
set "OUTPUT=out.mp4" :: Output video
set "RATE=30" :: Output framerate
set "SPEED=3.0" :: Speed factor, smaller than 1 = timelapse, 1 = real time, larger than 1 = slow motion
set "QU=2" :: MP4 Quality, 0 is best Quality, 2 is normal, 9 is strongest compression

%FF% -i %INPUT% -vf setpts=%SPEED%*PTS -r %RATE% -q:v %QU% -codec:v mpeg4 -an -y %OUTPUT%

pause

In this example the settings for "RATE" and "SPEED" are totally independant from each other. FFmpeg will automatically skip or duplicate frames, if required.

Example:
If both input and output frame rate are 30, and if SPEED = 3, then each frame will automatically duplicated 2 times, so that we see it 3 times in the output video.
If SPEED = 0.5, then each second frame is skipped.

In this example we used the video filter "setpts". For details, have a look at the "Video Filters" section in the FFmpeg documentation.

In this example the slow motion or timelapse effect affects only video and not audio. It makes sense to disable the audio channel with the -an option.

There's a wiki page covering this:
https://trac.ffmpeg.org/wiki/How%20to%20speed%20up%20or%20down%20a%20video
2.22 Slow motion or timelapse only for a segment of the video

See the comments for explanation.

```
set "FF=c:\ffmpeg\ffmpeg"  :: Path to FFmpeg
set "IN=7Z7A2089.mov"      :: Input Video
set "T1=5"                 :: Start time T1
set "T2=8.5"               :: Time T2 when slow motion begins
set "T3=9.7"               :: Time T3 when slow motion ends
set "T4=11"                :: End time T4
set "SPEED=5"              :: Speed factor, smaller than 1 = timelapse, larger than 1 = slow motion
set "FR=30"                :: Output framerate
set "OUT=out.mp4"          :: Output video

%FF% -i %IN% -filter_complex "[0:v]trim=%T1%:%T2%,setpts=PTS-STARTPTS[v1];[0:v]trim=%T2%:%T3%,setpts=%SPEED%*(PTS-STARTPTS)[v2];[0:v]trim=%T3%:%T4%,setpts=PTS-STARTPTS[v3];[v1][v2][v3]concat=n=3:v=1" -an -r %FR% -q:v 2 -y out.mp4
```

pause
2.23 Time Remapping

This is an example for a gradual ramp into and out of slow motion:

```bash
set "FF=c:\ffmpeg\ffmpeg" :: Path to FFmpeg
%FF% -f lavfi -i testsrc2=size=vga:duration=10:rate=20 -lavfi "^{[0] trim=0.0:3.2, setpts=(PTS-STARTPTS) [1];^{[0] trim=3.2:3.6, setpts=(PTS-STARTPTS)/0.80[2];^{[0] trim=3.6:4.0, setpts=(PTS-STARTPTS)/0.60[3];^{[0] trim=4.0:6.0, setpts=(PTS-STARTPTS)/0.40[4];^{[0] trim=6.0:6.4, setpts=(PTS-STARTPTS)/0.60[5];^{[0] trim=6.4:6.8, setpts=(PTS-STARTPTS)/0.80[6];^{[0] trim=6.8:10.0, setpts=(PTS-STARTPTS) [7];^{[0] concat=n=7:v=1" -y out.mp4

pause
```

This is an example for a 10s input video where the framerate changes linearly from 20 to 10:

```bash
set "FF=c:\ffmpeg\ffmpeg" :: Path to FFmpeg
%FF% -f lavfi -i testsrc2=size=vga:duration=10:rate=20 -lavfi "^{[0] trim=0:1, setpts=(PTS-STARTPTS)/0.975[1];^{[0] trim=1:2, setpts=(PTS-STARTPTS)/0.925[2];^{[0] trim=2:3, setpts=(PTS-STARTPTS)/0.875[3];^{[0] trim=3:4, setpts=(PTS-STARTPTS)/0.825[4];^{[0] trim=4:5, setpts=(PTS-STARTPTS)/0.775[5];^{[0] trim=5:6, setpts=(PTS-STARTPTS)/0.725[6];^{[0] trim=6:7, setpts=(PTS-STARTPTS)/0.675[7];^{[0] trim=7:8, setpts=(PTS-STARTPTS)/0.625[8];^{[0] trim=8:9, setpts=(PTS-STARTPTS)/0.575[9];^{[0] trim=9:10, setpts=(PTS-STARTPTS)/0.525[10];[1] [2] [3] [4] [5] [6] [7] [8] [9] [10] concat=n=10:v=1" -y out.mp4

pause
```

The length of the output video is 13.65s
Use the following example carefully, as I'm not 100% convinced that the approach is correct. This is based on an posting from Nicolas George in the FFmpeg user mailing list, September 23, 2019. In the first equation it's unclear if $t$ is the time in the input video or in the output video.

So, to compute the timestamp of a frame with variable speed:

- Express your frame rate as a complete formula: $t \rightarrow v$
- Integrate it: $t \rightarrow f$.
- Find the reciprocal: $f \rightarrow t$.

Let's assume we have a 10s video and the framerate changes linearly from 20 at the beginning to 10 at the end:

$$v = 20 - t \quad v(0) = 20 \quad v(10) = 10$$

After integrating we get:

$$f = 20 * t - 0.5 * t^2$$

The inverse function is:

$$t = 20 - \sqrt{400 - 2 * f}$$

Create a test video with framerate=20 and length=10s:

```bash
ffmpeg -f lavfi -i testsrc2=size=vga:duration=10:rate=20 -y test.mp4
```

Apply the time remapping:

```bash
ffmpeg -i test.mp4 -lavfi setpts='(20-sqrt(400-2*N))/TB' -y out.mp4
```

The resulting video gets slower towards the end (too slow, in fact), and the length is 18.95s and that seems to be wrong. With a constant framerate of 20 the length is 10s, with a constant framerate of 10 the length is 20s, and if the framerate changes from 20 to 10 the length should be about 15s. I don't fully understand what's going on here.

Keywords for searching: "Time remapping", "Time ramp", "Slow motion ramp", "Speed ramp"
2.24 Insert a text which is visible for the whole duration

```plaintext
set "FF=c://ffmpeg/ffmpeg" :: Path to FFmpeg
set "IN=input.mov" :: Input video
set "OUT=output.mp4" :: Output video
set "FONT=arial.ttf" :: Font
set "TEXT=Hello_World" :: Text (no space characters allowed, see next example)
set "COLOR=yellow" :: Text color
set "SIZE=20" :: Font size
set "POS_X=(w-tw)/2" :: X position of text, use (w-tw)/2 for centering
set "POS_Y=(h-th)/2" :: Y position of text, use (h-th)/2 for centering

%FF%  -i %IN% -vf
drawtext='fontfile=%FONT%:text=%TEXT%:fontcolor=%COLOR%:fontsize=%SIZE%:x=%POS_X%:y=%POS_Y%' -c:v mpeg4 -q:v 1 -y %OUT%
```

2.25 Slowly fade a text in and out

```plaintext
rem Slowly fade a text in and out

set "FF=c:\ffmpeg\ffmpeg" :: Path to FFmpeg
set "INPUT=PanoView.mp4" :: Input video
set "OUTPUT=out.mp4" :: Output video
set "QU=2" :: MP4 Quality, 0 is best Quality, 2 is normal, 9 is strongest compression
set "NAME=TEXT1" :: Unique name for this text
set "FONT=arial.ttf" :: Font
set "TEXT=MeinText.txt" :: Text filename (must be UTF-8 coded, if the text contains non-ASCII characters like ä, ö, ü. The text can be ASCII coded if no special characters are used.
set "COLOR=yellow" :: Text color
set "SIZE=250" :: Font size
set "POS_X=(w-tw)/2" :: X position of text, use (w-tw)/2 for centering
```
set "POS_Y=(h-th)/2"        :: Y position of text, use (h-th)/2 for centering
set "DS=0.5"                :: Start time
set "DE=4.5"                :: End time
set "FID=1.0"               :: Fade-in duration (may be small, but not zero)
set "FOD=1.0"               :: Fade-out duration (may be small, but not zero)

set %NAME%=drawtext='fontfile=%FONT%:textfile=%TEXT%:fontcolor_expr=%COLOR%@%%{e\:clip(((t-%DS%)/%FID%)*^between(t,%DS%,(%DS+%DE%)/2)+(%DE%-t)/%FOD%*between(t,(%DS+%DE%)/2,%DE%),0,1)}:fontsize=%SIZE%:x=%POS_X%:y=%POS_Y%'

set "NAME=TEXT2"            :: Unique name for this text
set "FONT=arial.ttf"        :: Font
set "TEXT=MeinText.txt"     :: Text filename (must be UTF-8 coded, if the text contains non-ASCII characters like ä, ö, ü. The text can be ASCII coded if no special characters are used.
set "COLOR=red"             :: Text color
set "SIZE=250"              :: Font size
set "POS_X=(w-tw)/2"        :: X position of text, use (w-tw)/2 for centering
set "POS_Y=(h-th)/3"        :: Y position of text, use (h-th)/2 for centering
set "DS=0.0"                :: Start time
set "DE=3.0"                :: End time
set "FID=0.5"               :: Fade-in duration (may be small, but not zero)
set "FOD=0.5"               :: Fade-out duration (may be small, but not zero)

set %NAME%=drawtext='fontfile=%FONT%:textfile=%TEXT%:fontcolor_expr=%COLOR%@\{e\:clip(((t-%DS%)/%FID%)*^between(t,%DS%,(%DS+%DE%)/2)+(%DE%-t)/%FOD%*between(t,(%DS+%DE%)/2,%DE%),0,1)}:fontsize=%SIZE%:x=%POS_X%:y=%POS_Y%'

%FF% -i %INPUT% -vf "%TEXT1%,%TEXT2%" -q:v %QU% -codec:v mpeg4 -an -y %OUTPUT%

pause

Texts must be saved in a *.txt file. If the text contains special characters like ä, ö, ü then the text must be saved with UTF-8 coding.

In some cases you may see in the video a non-printable character (shown as an empty rectangle) at the beginning of the text. In this case open the text file with a hex editor and remove the first three characters (EF\hex BB\hex BF\hex).
2.26 Show a running clock in the video

In this example a running clock is inserted in each frame of the video, in the format "hours:minutes:seconds.milliseconds"

```
set "FF=c://ffmpeg/ffmpeg" :: Path to FFmpeg
set "IN=P1000479.mov" :: Input video
set "OUT=sylvia.mp4" :: Output video
set "BP_R=0.015" :: Black point red, positive value makes background darker
set "BP_G=0.005" :: Black point green, positive value makes background darker
set "BP_B=0.015" :: Black point blue, positive value makes background darker
set "WP=0.26" :: White point
set "S=300" :: Start time
set "T=40" :: Duration
set "FONT=arial.ttf" :: Font
set "COLOR=white" :: Font color
set "BOXCOLOR=black" :: Background color
set "SIZE=30" :: Font size
set "POSITION_X=0" :: X position of clock
set "POSITION_Y=(h-th)" :: Y position of clock
set "OF=2340" :: Offset time in seconds, shown in the first frame
set "I=0.04" :: Time interval from one frame to the next = 1/framerate

set CLOCK=drawtext='fontfile=%FONT%:text=%%{eif:\mod((%OF%+%I%*n)/3600,24):'d':2}:'"%%{eif:\mod((%OF%+%I%*n)/60,60):'d':2}:'"%%{eif:\mod(%OF%+%I%*n,60):'d':2}:'"%%{eif:\mod((%OF%+%I%*n)*1000,1000):'d':3}:fontcolor=%COLOR%:boxcolor=%BOXCOLOR%:box=1:fontsize=%SIZE%:x=%POSITION_X%:y=%POSITION_Y%'

%FF% -ss %S% -i %IN% -vf "colorlevels=rimin=%BP_R%:gimin=%BP_G%:bimin=%BP_B%:rimax=%WP%:gimax=%WP%:bimax=%WP%,%CLOCK%" -pix_fmt yuv420p -t %T% -y %OUT%
```

This batch file does the same thing and is simpler:

```batch
set "FF=c://ffmpeg/ffmpeg" :: Path to FFmpeg

set "IN=P1000479.mov"      :: Input video
set "OUT=sylvia.mp4"       :: Output video

set "BP_R=0.015"           :: Black point red, positive value makes background darker
set "BP_G=0.005"           :: Black point green, positive value makes background darker
set "BP_B=0.015"           :: Black point blue, positive value makes background darker

set "WP=0.26"              :: White point

set "S=300"                :: Start time
set "T=40"                 :: Duration

set "FONT=arial.ttf"       :: Font
set "COLOR=white"          :: Font color
set "BCOLOR=black"         :: Background color
set "SIZE=30"              :: Font size
set "POS_X=0"              :: X position of clock
set "POS_Y=(h-th)"         :: Y position of clock
set "OFFSET=2340"          :: Offset time in seconds, added to the timestamp of the first frame

set CLOCK=drawtext='fontfile=%FONT%:text=%pts\:hms:\:%OFFSET%:fontcolor=%COLOR%:boxcolor=%BCOLOR%:box=1:fontsize=%SIZE\:%x=%POS_X%:y=%POS_Y%'

%FF% -ss %S% -i %IN% -vf "colorlevels=rimin=%BP_R%:gimin=%BP_G%:bimin=%BP_B%:rimax=%WP%:gimax=%WP%:bimax=%WP%,%CLOCK%" -pix_fmt yuv420p -t %T% -y %OUT%
pause
```

This is another example, using the "timecode" option of the drawtext filter:

```bash
set CLOCK=drawtext='fontfile=arial.ttf:timecode='00\:00\:00':r=25:x=20:y=40 -y out.mp4
```

```bash
c://ffmpeg/ffmpeg -f lavfi -i testsrc2=size=hd720:duration=10 -vf
drawtext=fontsize=60:fontcolor=Black:fontfile='arial.ttf':timecode='00\:00\:00':r=25:x=20:y=40 -y out.mp4
```

pause
2.27 Generation of curved text for fulldome projection

I have to admit that this is a complicated command line. The actual core is the "remap" filter, with which you can create arbitrary distortions. The distortion is described in the two files xmap_3648.pgm and ymap_3648.pgm. In these files the pixel in the input video from which it is retrieved is indicated for each pixel. You have to write a (C#) program that can create these files.

- i color=c=black creates a black image
- i anullsrc creates an empty audio track
This is the C# code for generating the xmap and ymap files:

```csharp
int a = (int)numericUpDown1.Value; // get the size of the square map
double c = (double)numericUpDown2.Value; // this is the aspect ratio of the text, normal = 1
int b = a/2;
int xx, yy;

TextWriter xmap = File.CreateText("xmap_" + a.ToString() + ".pgm");
xmap.Write("P2\n");
xmap.Write("# Xmap file for fulldome remap \n");
xmap.Write(a.ToString() + " " + a.ToString() + " \n");
xmap.Write("65535\n");

TextWriter ymap = File.CreateText("ymap_" + a.ToString() + ".pgm");
ymap.Write("P2\n");
ymap.Write("# Ymap file for fulldome remap \n");
ymap.Write(a.ToString() + " " + a.ToString() + " \n");
ymap.Write("65535\n");

for (int y = 0; y < a; y++)
{
    for (int x = 0; x < a; x++)
    {
        xx = x;
        yy = y;
        if (y > b)
        {
            xx = b + (int)(b / c * Math.Atan((double)(x - b) / (double)(y - b)));
            yy = b + (int)Math.Sqrt((x - b) * (x - b) + (y - b) * (y - b));
            if (xx < 0) xx = 0;
            if (yy < 0) yy = 0;
            if (xx > a - 1) xx = a - 1;
            if (yy > a - 1) yy = a - 1;
        }
        xmap.Write(xx + " ");
        ymap.Write(yy + " ");
    }
    xmap.Write("\n");
    ymap.Write("\n");
}
xmap.Write("\n");
ymap.Write("\n");
xmap.Close();
ymap.Close();
```
This is a simpler example for generating curved text for fulldome projection, using the v360 filter:

```
set "FF=c:\ffmpeg\ffmpeg"  :: Path to FFmpeg
set "UP=30"                :: Up-looking angle in degrees (center of the rectangular video)
set "H=64"                 :: Horizontal field of view, this is for 16:9 aspect ratio
set "V=36"                 :: Vertical field of view, this is for 16:9 aspect ratio
set "SIZE=1200"            :: Square size of the output video
set "FONT=arial.ttf"       :: font
set "FSIZE=120"            :: font size
set "COLOR=white"          :: text color
set "BACK=black"           :: background color
set "TEXT=text13.txt"      :: text file
set "POS_X=(w-tw)/2"       :: X text position, for centered text: (w-tw)/2
set "POS_Y=(h-th)/2"       :: Y text position, for centered text: (h-th)/2
set "S=1"                  :: start time for text
set "E=9"                  :: end time for text
set "FI=2"                 :: fade-in duration (may be small, but not zero)
set "FO=2"                 :: fade-out duration (may be small, but not zero)
set "DUR=10"               :: duration of video
set "OUT=out.mp4"          :: Output video

%FF% -f lavfi -i color=%BACK%:size=hd1080 -vf drawtext='fontfile=%FONT%:textfile=%TEXT%:fontcolor_expr=%COLOR%@\n{(e\:\clip((t-%S%)/%FI%*\between(t,%S%,%S%+%FI%)+(%E%-t)/%FO%*\between(t,%S%+%FI%,%E%)),0,1)):fontsize=%FSIZE%:x=%POS_X%:y=%POS_Y%',v360=input=flat:ih_fov=%H%:iv_fov=%V%:output=fisheye:h_fov=180:v_fov=180:pitch='90-%UP%':w=%SIZE%:h=%SIZE% -t %DUR% -y %OUT%
pause
```
2.28 Combine multiple videos with concat demuxer

The concat demuxer combines several videos without re-encoding. It's very fast.

```
rem   Final cut with concat demuxer
rem

rem   Final cut with concat demuxer
rem
c:\ffmpeg\ffmpeg -f concat -i concat_list.txt -c copy -y MyVideo.mp4

pause
```

You simply write all existing scenes into a text file (here: concat_list.txt), which looks like this:

```
file text1.mp4              :: 10  Title: A year in the woods
file text2.mp4              :: 10  When and where
file Videos/scene20.mp4     :: 12  Live video in the wood
# This is a comment
file text22.mp4             :: 10  In 15 months...
file Videos/scene22.mp4     :: 52  Live video, camera
file text98.mp4             :: 10  the end
```

To the right of the double colons are optional comments (e.g. the length of the scenes and a short description). Comments can also begin with #.

This method, however, requires that all scenes have

- the same width and height
- the same video codec
- the same framerate
- the same audio codec
- the same number of audio tracks (take care when you use a camera which writes only a mono soundtrack)
- the same audio sample rate

If one of these conditions isn't met, an error message is issued. You can then look at the properties of the files with FFprobe or Exiftool to find out where the files differ.
2.29 Combine multiple videos with concat filter

In this example the concat filter is used for input videos of the same size and no audio.

Each of the -ss and -t specifies the start time and length of the next input file. You can remove these options if you want to use the full videos.

The value n=3 passed to the concat filter should match the number of input files.

This filter does re-encode the videos, so the process is slow but you can also specify the encoding quality.

```verbatim
set "FF=c:\ffmpeg\ffmpeg" :: Path to FFmpeg
set "I1=my_video1.mp4"    :: Input video 1
set "S1=0"                :: Set start time 1
set "L1=4"                :: Set length 1
set "I2=my_video2.mp4"    :: Input video 2
set "S2=3"                :: Set start time 2
set "L2=3"                :: Set length 2
set "I3=my_video3.mp4"    :: Input video 3
set "S3=6"                :: Set start time 3
set "L3=2"                :: Set length 3
set "OUT=out.mp4"         :: Output video

%FF% -ss %S1% -t %L1% -i %I1% -ss %S2% -t %L2% -i %I2% -ss %S3% -t %L3% -i %I3% -filter_complex "concat=n=3:v=1:a=0" -an
%OUT%
```
2.30 Switch between two cameras, using audio from camera1

set "FF=c:\ffmpeg\ffmpeg" :: Path to FFmpeg

rem Create a 6 seconds red video with 400Hz tone
%FF% -f lavfi -i color=c=red:s=vga -f lavfi -i sine=frequency=400 -t 6 -y video1.mp4

rem Create a 6 seconds test video with 1200Hz tone
%FF% -f lavfi -i testsrc2=s=vga -f lavfi -i sine=frequency=1200 -t 6 -y video2.mp4

rem Switch to video2 from 2 to 4 seconds, but use always the audio from video1
%FF% -i video1.mp4 -i video2.mp4 -filter_complex blend=all_expr='if(between(T,2,4),B,A)' -y test.mp4

Note: In this example both videos start at the same time. The video2 segment from 2 to 4 seconds is inserted in the output video from 2 to 4 seconds. You get this output video:

<table>
<thead>
<tr>
<th></th>
<th>0 &lt; t &lt; 2</th>
<th>2 &lt; t &lt; 4</th>
<th>4 &lt; t &lt; 6</th>
</tr>
</thead>
<tbody>
<tr>
<td>Video</td>
<td>from video1 (0...2)</td>
<td>from video2 (2...4)</td>
<td>from video1 (4...6)</td>
</tr>
<tr>
<td>Audio</td>
<td>from video1 (0...2)</td>
<td>from video1 (2...4)</td>
<td>from video1 (4...6)</td>
</tr>
</tbody>
</table>

If you want to insert the video2 segment from 0 to 2 seconds in the output video from 2 to 4 seconds, use this command line instead:

%FF% -i video1.mp4 -i video2.mp4 -filter_complex [1]tpad=start_duration=2[2];[0][2]blend=all_expr='if(between (T,2,4),B,A)' -y test.mp4

In this case you get this output video:

<table>
<thead>
<tr>
<th></th>
<th>0 &lt; t &lt; 2</th>
<th>2 &lt; t &lt; 4</th>
<th>4 &lt; t &lt; 6</th>
</tr>
</thead>
<tbody>
<tr>
<td>Video</td>
<td>from video1 (0...2)</td>
<td>from video2 (0...2)</td>
<td>from video1 (4...6)</td>
</tr>
<tr>
<td>Audio</td>
<td>from video1 (0...2)</td>
<td>from video1 (2...4)</td>
<td>from video1 (4...6)</td>
</tr>
</tbody>
</table>
2.31 Stack videos side by side (or on top of each other)

```plaintext
set "FF=c://ffmpeg/ffmpeg" :: Path to FFmpeg
set "IN1=left.mp4"
set "IN2=right.mp4"
set "OUT=out.mp4"
rem use "hstack" for horizontal stacking and "vstack" for vertical stacking

%FF% -i %IN1% -i %IN2% -filter_complex hstack -an -shortest -c:v mpeg4 -y %OUT%

pause
```

Note: If the videos have different width or height, use the xstack filter instead.

2.32 Horizontal and vertical flipping

This can be done with the "hflip" and "vflip" filters.
2.33 Stack four videos to a 2x2 mosaic

```bash
set "FF=c://ffmpeg/ffmpeg"    :: Path to FFmpeg
set "IN1=topleft.mp4"
set "IN2=topright.mp4"
set "IN3=bottomleft.mp4"
set "IN4=bottomright.mp4"
set "OUT=mosaic.mp4"

%FF% -i %IN1% -i %IN2% -i %IN3% -i %IN4% -filter_complex [0:v][1:v]hstack[t];[2:v][3:v]hstack[b];[t][b]vstack -an -shortest -c:v mpeg4 -q:v 1 -y %OUT%

pause
```

Other method using xstack:

```bash
set "FF=c://ffmpeg/ffmpeg"    :: Path to FFmpeg
set "IN1=topleft.mp4"
set "IN2=topright.mp4"
set "IN3=bottomleft.mp4"
set "IN4=bottomright.mp4"
set "OUT=mosaic.mp4"

%FF% -i %IN1% -i %IN2% -i %IN3% -i %IN4% -filter_complex "xstack=inputs=4:layout=0_0|0_h0|w0_0|w0_h0" -shortest %OUT%

pause
```

Display 4 inputs into a vertical 1x4 grid, note that the input videos may have different widths (vstack can't handle this case).

```bash
%FF% -i %IN1% -i %IN2% -i %IN3% -i %IN4% -filter_complex "xstack=inputs=4:layout=0_0|0_h0|0_h0+h1|0_h0+h1+h2" %OUT%
```
This is an example of a blink comparator. It creates an animated GIF that continuously toggles between two (or more) images.

```bash
rem  Blink comparator, animated GIF

set "FF=c:\ffmpeg\ffmpeg"  :: Path to FFmpeg
set "IN=pluto_%1d.jpg"    :: Filename of the images
set "FR=2.0"               :: Frame rate
set "OUT=out.gif"          :: Animated GIF output file

%FF% -framerate %FR% -i %IN% -q:v 0 -y %OUT%

pause :: Wait for a keypress
```

Please note that there is a known problem with FFmpeg's GIF encoder which may result in wrong colors in the output file. See the next chapter for a workaround.

If you want to create an MP4 instead, then you have to specify how long it should be and the input and output framerates:

```bash
rem  Blink comparator, MP4

set "FF=c:\ffmpeg\ffmpeg"  :: Path to FFmpeg
set "IN=pluto_%1d.jpg"    :: Filename of the images
set "FI=2.0"               :: Framerate for reading in the pictures
set "T=10"                 :: Length in seconds
set "FO=25"                :: Output framerate
set "OUT=out.mp4"          :: Output MP4 file

%FF% -loop 1 -framerate %FI% -i %IN% -t %T% -r %FO% -q:v 0 -y %OUT%

pause :: Wait for a keypress
```

The parameter "-loop 1" causes the same images to be read in again and again. If you do this, you have to limit the length of the video somehow, in this case with "-t 10".
2.35 Animated GIF

There is a known problem with FFmpeg's GIF encoder which may result in wrong colors in the animated GIF output file. If you encounter this problem, you can use the following workaround which uses the palettegen and paletteuse filters. Thanks to Javier Infante Porro for posting this workaround in the FFmpeg user mailing list on September 26, 2019.

```batch
set "FF=c:\ffmpeg\ffmpeg" :: Path to FFmpeg
set "IN=in.gif" :: Input video (animated GIF)
set "COL=8" :: Number of colors (including one transparent color)
set "OUT=out.gif" :: Output video (animated GIF)

%FF% -i %IN% -lavfi "split[s0][s1];[s0]palettegen=max_colors=%COL%;[s1][p]paletteuse" -y %OUT%

pause
```

Please note that one entry in the palette is reserved for the transparent color by default. So when you set the `max_colors` parameter to 8, you have only 7 different visible colors. If you don't want a transparent color, you must disable it with the `reserve_transparent=0` option.

Much more about this subject can be found here:

http://blog.pkh.me/p/21-high-quality-gif-with-ffmpeg.html
Replace one frame in a video by another

This example shows how to replace a single image in a video with another image. You may have heard of a trick to insert a product image into a film for advertising purposes, only for the duration of a single frame. For example, if the frame rate is 25 frames per second, then a single frame will be shown for 40ms. That's too short to recognize the product clearly, but it's long enough to make viewers feel that they want this product. If, for example, a bratwurst or popcorn is shown for 40ms in the film, the sales figures for exactly these products increase after the end of the film. Although the viewer is not aware of why he has now gotten an appetite for a bratwurst or popcorn.

```
set "FF=c:\ffmpeg\ffmpeg" :: Path to FFmpeg
set "IN=scene8.mp4" :: Input video
set "BW=bratwurst.jpg" :: Image of bratwurst
set "W=1920" :: Width of input video
set "H=1080" :: Height of input video
set "T=3.0" :: Time when the image shall be insert
set "OUT=out.mp4" :: Output video
%FF% -i %IN% -i %BW% ^
-filter_complex "[1]scale=w=%W%:h=%H%,setpts=%T%/TB[im];[0][im]overlay=eof_action=pass" -c:a copy -q:v 0 %OUT%
pause
```

The "scale" filter scales the image to the same size as the input video. If the image already has the correct size, you can omit this filter. The "setpts" filter sets the time for the image. The "overlay" filter then combines the two sources. The audio track is taken unchanged from the input video.

The same thing can also be done with the freezeframes filter:

```
set "FF=c:\ffmpeg\ffmpeg" :: Path to FFmpeg
set "IN=scene8.mp4" :: Input video
set "IN2=test.mp4" :: Second input which contains the replacement frame
set "F=75" :: Number of the frame to be replaced
set "R=1" :: Number of the replacement frame from the second input
c:\ffmpeg\ffmpeg -i %IN% -i %IN2% -filter_complex freezeframes=first=%F%:last=%F%:replace=%R% out.mp4
pause
```
### 2.37 Blend filter

Unfortunately the FFmpeg documentation doesn’t explain what all the blend filters do. So you have to look it up in the source code:

```c
DEFINE_BLEND16(addition,    FFMIN(65535, A + B), 16)  // Output is (A*B) with an upper limit at white level
DEFINE_BLEND16(grainmerge,  av_clip_uint16(A + B - 32768), 16)
DEFINE_BLEND16(average,     (A + B) / 2, 16)           // Output is the arithmetic mean of A and B
DEFINE_BLEND16(subtract,    FFMAX(0, A - B), 16)        // Output is (A-B) with a lower limit at black level
DEFINE_BLEND16(multiply,    MULTIPLY(1, A, B), 16)
DEFINE_BLEND16(multiply128, av_clip_uint16((A - 32768) * B / 8192. + 32768), 16)
DEFINE_BLEND16(negation,    65535 - FFABS(65535 - A - B), 16)
DEFINE_BLEND16(extremity,   FFABS(65535 - A - B), 16)
DEFINE_BLEND16(difference, FFABS(A - B), 16)            // Output is the absolute difference of A and B
DEFINE_BLEND16(grainextract, av_clip_uint16(32768 + A - B), 16)  // Output is (A-B), shifted to 50% gray level, with limits at black and white levels
DEFINE_BLEND16(screen,      SCREEN(1, A, B), 16)
DEFINE_BLEND16(overlay,     (A < 32768) ? MULTIPLY(2, A, B) : SCREEN(2, A, B), 16)
DEFINE_BLEND16(hardlight,   (B < 32768) ? MULTIPLY(2, B, A) : SCREEN(2, B, A), 16)
DEFINE_BLEND16(hardmix,     (A < (65535 - B)) ? 0: 65535, 16)
DEFINE_BLEND16(heat,        (A == 0) ? 0 : FFMIN(((65535 - B) * (65535 - B)) / A, 65535), 16)
DEFINE_BLEND16(freeze,      (B == 0) ? 0 : FFMIN(((65535 - A) * (65535 - A)) / B, 65535), 16)
DEFINE_BLEND16(darken,      FFMIN(A, B), 16)            // Output is the minimum of A and B
DEFINE_BLEND16(lighten,     FFMAX(A, B), 16)            // Output is the maximum of A and B
DEFINE_BLEND16(divide,      av_clip_uint16(B == 0 ? 65535 : 65535 * A / B), 16)
DEFINE_BLEND16(dodge,       DODGE(A, B), 16)
DEFINE_BLEND16(burn,        BURN(A, B), 16)
DEFINE_BLEND16(softlight,   (A > 32767) ? B + (65535 - B) * (A - 32767.5) / 32767.5 * (0.5 - fabs(B - 32767.5) / 65535): B - B * ((32767.5 - A) / 32767.5) * (0.5 - fabs(B - 32767.5)/65535), 16)
DEFINE_BLEND16(exclusion,   A + B - 2 * A * B / 65535, 16)
DEFINE_BLEND16(pinlight,    (B < 32768) ? FFMIN(A, 2 * B) : FFMAX(A, 2 * (B - 32768)), 16)
DEFINE_BLEND16(phoenix,     FFMIN(A - B, A) + 65535, 16)
DEFINE_BLEND16(reflect,     (B == 65535) ? B : FFMIN(65535, (A * A / (65535 - B))), 16)
DEFINE_BLEND16(glow,        (A == 65535) ? A : FFMIN(65535, (B * B / (65535 - A))), 16)
DEFINE_BLEND16(and,         A & B, 16)
DEFINE_BLEND16(or,          A | B, 16)
DEFINE_BLEND16(xor,         A ^ B, 16)
DEFINE_BLEND16(vividlight,  BURN(2 * A, B) : DODGE(2 * (A - 32768), B), 16)
```
2.38 Subtracting a darkframe

Noise, hot pixels and amplifier glow in a low-light video can be reduced by subtracting a darkframe. Make a dark video with the same settings and at the same temperature as your main video. The only difference is that you put the cap on the lens. Then you can average many (up to 128) frames from the dark video and save the darkframe lossless as 16-bit PNG:

```bash
set "FF=c://ffmpeg/ffmpeg" :: Path to FFmpeg
set "DARKVID=Dark.mov" :: Dark video

%FF% -i %DARKVID% -vf "tmix=128,format=rgb48" -frames 1 -y dark.png
```

Pause

Now you can subtract this darkframe from all frames of your video:

```bash
set "FF=c://ffmpeg/ffmpeg" :: Path to FFmpeg
set "IN=meteor.mov" :: Input video
set "OUT=meteor-dark.mp4" :: Output video

%FF% -i %IN% -i dark.png -filter_complex "format=rgb48[a];[a][1]blend=subtract" -y %OUT%
```

Pause
2.39 Gradation curves and vignetting

Note: This is obsolete. It’s better done with a Color look-up table.

-- An image is opened in GIMP.
-- Colors::Values --> Select suitable points for black, white and gray in the image.
-- Click on "Edit these settings as curves".
-- make fine corrections on the curves
-- Set as many points as possible on the curves, because later they will be interpolated by straight lines.
-- Click on "Export settings as file".
-- Check the box "Use old file format for curves".
-- filename: curves.gimp
-- Save
-- Then call the GIMP2ACV converter (1). This converter reads the file curves.gimp, converts it and saves it as curves.acv. The file curves.gimp must be located in the same folder where the converter is called.
-- In the batch file for the FFmpeg editing the corresponding video filter is called: -vf curves=psfile='curves.acv'

Vignetting at the edge of the image can be compensated automatically: -vf vignette=0.5:mode=backward
mode=backward makes the image corners brighter, mode=forward makes them darker. The number 0.5 must be set so that the corners are neither too bright nor too dark.
The two filters can be combined in this way:

```
-vf vignette=0.5:mode=backward,curves=psfile='curves.acv'
```

(1) Source: http://www.astro-electronic.de/GIMP2ACV.EXE
2.40 Color grading with color look-up tables, full workflow

A color look-up table (CLUT) is a mathematical rule according to which any color is replaced by another color.

There are different file formats for the CLUT:

The *.cube format normally has a color space of 25 * 25 * 25 entries, so that the table contains 25^3 = 15625 different colors. Colors between the specified table entries are interpolated. You can also create tables with 64^3 entries, but for most applications 25^3 entries are sufficient.

But it's also possible to save a CLUT in any uncompressed image format.

The complete workflow is now described step by step for a 10-bit video. This workflow can be simplified, see the next chapter.

Step 1: With this batch file a single image is extracted from the 10-bit video at a suitable location and saved lossless as 16-bit PNG:

```batch
set "FF=c:\ffmpeg\ffmpeg"   :: Path to FFmpeg
set "IN=Video_62.mov"        :: Input video
set "T=35"                   :: Time where image is extracted
%FF% -ss %T% -i %IN% -frames 1 -y image.png
pause
```

Step 2: This batch file is used to create a CLUT (= Color-look-up-Table). This is a PNG image with 512x512 pixels that contains exactly one pixel of each possible color. I'm not yet sure if the image has to have 16 bit resolution at this point. At least it doesn't hurt. If 8 bits are enough, you would omit "-pix_fmt rgb48be".

The LEVEL parameter determines how many different colors are contained in the CLUT. The height and width of the square image is LEVEL*LEVEL*LEVEL, at LEVEL=8 there are 64*64*64=262144 colors and the image has 512*512=262144 pixels. It is important that the file is saved in an uncompressed or lossless compressed format, so PNG is well suited.

```batch
set "FF=c:\ffmpeg\ffmpeg"   :: Path to FFmpeg
set "LEVEL=8"
%FF% -f lavfi -i haldcluts=LEVEL% -frames 1 -pix_fmt rgb48be clut.png
pause
```

Step 3: The extracted image is opened in GIMP.
Step 4: The color table will be opened in GIMP, selected with "Select all" and copied with ctrl-c.

Step 5: The first image is clicked and then the color table is inserted in the upper left corner with "Paste in Place". Since the first image is much larger than the color table, the table does not interfere at this position.

Step 6: Right click on "Floating Selection" and select "To New Layer".

Step 7: Right click on the newly created "Pasted Layer" and select "Merge Down".

Step 8: Now the image is edited as it should look in the video. And of course the color table in the upper left corner will be edited as well. Color corrections, color temperature, color saturation, gradation curve, brightness, contrast. The image may contain visible noise. Later in the video, the noise doesn't stand out so much, because it is partly averted by the fast sequence of images. Operations that cannot be described by a color look-up table, such as noise reduction, soft focus or sharpening, are not permitted.

Step 9: The finished image is trimmed to a size of 512x512 pixels so that only the color table in the upper left corner remains. Image > Canvas Size > Width=512, Height=512, then click on "Resize".

Step 10: Export the image under the name clut2.png as 16-bit PNG and select "16bpc RGB" as pixel format. GIMP can now be closed.

Step 11: This color look-up table is now applied to the whole video with FFmpeg. The color table is applied with 10 bit accuracy. Colors not included in the table are interpolated. Only then is the color table converted to 8 bit accuracy and an MP4 generated:

```bash
set "FF=c:\ffmpeg\ffmpeg" :: Path to FFmpeg
set "IN=Video_62.mov" :: Input video

%FF% -i %IN% -i clut2.png -filter_complex [0][1]haldclut out.mp4
```

pause
2.41 Color grading with color look-up tables, simplified workflow

The above workflow can be simplified as follows:

Step 1: In this batch file FFmpeg does immediately combine the CLUT with the extracted image:

```
set "FF=c://ffmpeg/ffmpeg" :: Path to FFmpeg
set "IN=P1000099.mov" :: Input video
set "T=5" :: Time where image is extracted

%FF% -ss %T% -i %IN% -f lavfi -i haldclutsrc=8 -filter_complex "[1]format=pix_fmts=rgb48be[a];[a][0]xstack=inputs=2:layout=0_0|w0_0" -frames 1 -y Image_with_CLUT.png
```

Step 2: This image is now processed in GIMP (or any other suitable image processing software) and then exported with the same file name as 16-bit PNG. You can edit brightness, contrast, gamma, saturation and hue. You can also adjust the curves. Of course, all modifications must be applied to the whole image consisting of the video frame and the clut. Filters like noise reduction, sharpening or softening are not allowed.

Step 3: This batch file does first use the crop filter to remove the image so that only the CLUT remains. Why the small brightness correction is necessary before applying the haldclut filter isn’t yet fully understood. In the second FFmpeg run the CLUT is applied to the input video. Then the CLUT is deleted because it’s no longer required.

```
set "FF=c://ffmpeg/ffmpeg" :: Path to FFmpeg
set "IN=P1000099.mov" :: Input video
set "BR=0.06" :: Small brightness adjustment before applying the CLUT

%FF% -i Image_with_CLUT.png -vf crop=512:512:0:0 -y clut.png
%FF% -i %IN% -i CLUT.png -filter_complex [0]eq=brightness=%BR%[a];[a][1]haldclut -y out.mp4
del clut.png
```

pause
2.42 Size of color-look-up tables

The size of the Color-look-up for the haldclut filter depends on the "Level" parameter as follows:

<table>
<thead>
<tr>
<th>Level</th>
<th>Size of CLUT n^3 x n^3</th>
<th>File size of CLUT as 16-bit PNG</th>
<th>Edge length of the RGB cube n^2</th>
<th>Number of support points n^6</th>
<th>Distance of support points 8-bit 256 / n^2</th>
<th>Distance of support points 16-bit 65536 / n^2</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>64x64 Pixel</td>
<td>16.4kB</td>
<td>16</td>
<td>4096</td>
<td>16</td>
<td>4096</td>
</tr>
<tr>
<td>5</td>
<td>125x125 Pixel</td>
<td>61.9kB</td>
<td>25</td>
<td>15625</td>
<td>10.2</td>
<td>2621</td>
</tr>
<tr>
<td>6</td>
<td>216x216 Pixel</td>
<td>179kB</td>
<td>36</td>
<td>46656</td>
<td>7.11</td>
<td>1820</td>
</tr>
<tr>
<td>7</td>
<td>343x343 Pixel</td>
<td>436kB</td>
<td>49</td>
<td>117649</td>
<td>5.22</td>
<td>1337</td>
</tr>
<tr>
<td>8</td>
<td>512x512 Pixel</td>
<td>0.97MB</td>
<td>64</td>
<td>262144</td>
<td>4</td>
<td>1024</td>
</tr>
<tr>
<td>9</td>
<td>729x729 Pixel</td>
<td></td>
<td>81</td>
<td>531441</td>
<td>3.16</td>
<td>809</td>
</tr>
<tr>
<td>10</td>
<td>1000x1000 Pixel</td>
<td></td>
<td>100</td>
<td>1000000</td>
<td>2.56</td>
<td>655</td>
</tr>
<tr>
<td>11</td>
<td>1331x1331 Pixel</td>
<td></td>
<td>121</td>
<td>1771561</td>
<td>2.12</td>
<td>542</td>
</tr>
<tr>
<td>12</td>
<td>1728x1728 Pixel</td>
<td></td>
<td>144</td>
<td>2985984</td>
<td>1.78</td>
<td>455</td>
</tr>
<tr>
<td>13</td>
<td>2197x2197 Pixel</td>
<td></td>
<td>169</td>
<td>4826809</td>
<td>1.51</td>
<td>388</td>
</tr>
<tr>
<td>14</td>
<td>2744x2744 Pixel</td>
<td></td>
<td>196</td>
<td>7529536</td>
<td>1.31</td>
<td>334</td>
</tr>
<tr>
<td>15</td>
<td>3375x3375 Pixel</td>
<td></td>
<td>225</td>
<td>11390625</td>
<td>1.14</td>
<td>291</td>
</tr>
<tr>
<td>16</td>
<td>4096x4096 Pixel</td>
<td></td>
<td>256</td>
<td>16777216</td>
<td>1</td>
<td>256</td>
</tr>
</tbody>
</table>

2.43 Histogram

This batch file generates a histogram for the R,G,B components from a video:

```plaintext
set "FF=c:\ffmpeg\ffmpeg"   :: Path to FFmpeg
set "IN=MVI_2562.mov"       :: Input video

%FF% -i %IN% -vf format=pix_fmts=rgb24,histogram=levels_mode=logarithmic -y out.mp4
pause
```
2.44 Lagfun filter

The lagfun filter makes short pulses of light appear longer, with an exponential decay curve. Good for meteors in the night sky.

It works as follows:

The previous output frame is multiplied by the decay constant, which is in the range [0 ... 1] and a typical value is 0.95. This image is used as the next output frame. But if a pixel in the next input frame is brighter, then the brighter value is used. So all pixels have a fast rise time constant and a slow decay time constant. Like an oscilloscope screen with a long persistence time.

Time constant in seconds = 1 / ((1 - decay) * framerate)

The time constant is the duration during which a signal drops from level 1.0 to 1 / e = 0.368

---

rem Example for lagfun, left side of output video is without lagfun and right side is with lagfun

```bash
set "PATH=c:/ffmpeg/ffmpeg" :: Path to FFmpeg
set "SN=1400" :: Start number
set "CONTRAST=2.0" :: Contrast in range [-1000 ... 1000], normal is 1.0
set "BRIGHT=0.22" :: Brightness in range [-1.0 ... 1.0], normal is 0.0
set "GAMMA=2.5" :: Gamma in range [0.1 ... 10.0], normal is 1.0
set "DEF=10" :: Deflicker frames
set "DECAY=0.95" :: Decay factor
set "QU=2" :: MP4 quality level, 0 is best quality, 2 is normal, 9 is strong compression
set "FPS=30" :: Output framerate
set "OUT=meteors.mp4" :: Output filename

%PATH% -start_number %SN% -i IMG_%%4d.jpg ^
-filter_complex "eq=contrast=%CONTRAST%:brightness=%BRIGHT%:gamma=%GAMMA%,deflicker=size=%DEF%,split[a][b];
[b]lagfun=decay=%DECAY%[c];[a][c]hstack" -r 30 -codec:v mpeg4 -q:v %QU% -y %OUT%

pause
```

---

54
2.45 Deblock filter

This filter removes unwanted blocking artefacts from low-quality input images or videos.

2.46 Gradfun filter

This filter removes unwanted banding artefacts that appear in backgrounds with a brightness gradient, especially in the sky towards the horizon.

```
set "FF=c:\ffmpeg\ffmpeg" :: Path to FFmpeg
set "IN=MVI_2562.mov" :: Input video
set "OUT=output.mp4" :: Output video

%FF% -i %IN% -vf gradfun=3.5:8 -y %OUT%
```

The first parameter is the strength, this is the maximum amount the filter will change any one pixel. Allowed values are from 0.51 to 64, the default value is 1.2

The second parameter is the radius, which defines the neighborhood to fit the gradient to. Accepted values are from 8 to 32, the default is 16.

Don't use this filter before lossy compression.

2.47 Dilation filter

This filter replaces each pixel by the brightest pixel in the 3x3 neighborhood. It's very useful if you have fisheye images of the night sky (taken with Canon 6D, height 3648 pixels) and want to scale them down to height 1200 pixels (for projection in the planetarium). Scaling down would remove the fainter stars, because each pixel in the resulting image would be the average of 3x3 pixels in the original image. You can avoid this by using the dilation filter prior to scaling down.
2.48  V360 filter for rotation of equirectangular 360° videos

This video filter converts equirectangular 360° panoramic videos between various formats, and it can also rotate them.

The default rotation order is yaw --> pitch --> roll, but can be changed by setting the "rorder" parameter. Positive yaw moves the line of sight towards the right, positive pitch moves the line of sight up, positive roll rotates the image clockwise (or rotates the observer's head counter-clockwise).

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>set &quot;FF=c:\ffmpeg\ffmpeg&quot; :: Path to FFmpeg</td>
<td></td>
</tr>
<tr>
<td>set &quot;IN=test1.mp4&quot; :: Input video</td>
<td></td>
</tr>
<tr>
<td>%FF% -ss 10 -i %IN% -vf v360=yaw=0:output=e -frames 1 -y t_original.jpg</td>
<td><strong>t_original.jpg</strong></td>
</tr>
<tr>
<td>%FF% -ss 10 -i %IN% -vf v360=yaw=90:output=e -frames 1 -y t_yaw90.jpg</td>
<td><strong>t_yaw90.jpg</strong></td>
</tr>
<tr>
<td>%FF% -ss 10 -i %IN% -vf v360=pitch=90:output=e -frames 1 -y t_pitch90.jpg</td>
<td><strong>t_pitch90.jpg</strong></td>
</tr>
<tr>
<td>%FF% -ss 10 -i %IN% -vf v360=roll=90:output=e -frames 1 -y t_roll90.jpg</td>
<td><strong>t_roll90.jpg</strong></td>
</tr>
<tr>
<td>%FF% -ss 10 -i %IN% -vf v360=yaw=90:pitch=90:output=e -frames 1 -y t_yaw90_pitch90.jpg</td>
<td><strong>t_yaw90_pitch90.jpg</strong></td>
</tr>
<tr>
<td>%FF% -ss 10 -i %IN% -vf v360=yaw=90:roll=90:output=e -frames 1 -y t_yaw90_roll90.jpg</td>
<td><strong>t_yaw90_roll90.jpg</strong></td>
</tr>
<tr>
<td>%FF% -ss 10 -i %IN% -vf v360=pitch=90:roll=90:output=e -frames 1 -y t_pitch90_roll90.jpg</td>
<td><strong>t_pitch90_roll90.jpg</strong></td>
</tr>
</tbody>
</table>
Parameters of the v360 filter:

<table>
<thead>
<tr>
<th>input, output</th>
<th>e, equirect</th>
<th>Equirectangular format</th>
</tr>
</thead>
<tbody>
<tr>
<td>c3x2, c6x1, c1x6</td>
<td>Three different cubemap formats</td>
<td></td>
</tr>
<tr>
<td>eac</td>
<td>Equi-angular cubemap</td>
<td></td>
</tr>
<tr>
<td>flat, gnomonic, rectilinear</td>
<td>Regular video format</td>
<td></td>
</tr>
<tr>
<td>dfisheye</td>
<td>Dual fisheye format</td>
<td></td>
</tr>
<tr>
<td>barrel, fb</td>
<td>Facebook's 360 format</td>
<td></td>
</tr>
<tr>
<td>sg</td>
<td>Stereographic format</td>
<td></td>
</tr>
<tr>
<td>mercator</td>
<td>Mercator format</td>
<td></td>
</tr>
<tr>
<td>ball</td>
<td>Ball format</td>
<td></td>
</tr>
<tr>
<td>hammer</td>
<td>Hammer-Aitoff map projection format</td>
<td></td>
</tr>
<tr>
<td>sinusoidal</td>
<td>Sinusoidal map projection format</td>
<td></td>
</tr>
<tr>
<td>fisheye</td>
<td>Single fisheye format</td>
<td></td>
</tr>
<tr>
<td>pannini</td>
<td>Pannini projection (output only)</td>
<td></td>
</tr>
<tr>
<td>cylindrical</td>
<td>Cylindrical projection</td>
<td></td>
</tr>
<tr>
<td>perspective</td>
<td>Perspective projection, this is like watching a sphere from big distance (output only)</td>
<td></td>
</tr>
<tr>
<td>tetrahedron</td>
<td>Tetrahedron projection</td>
<td></td>
</tr>
<tr>
<td>interp</td>
<td>near, nearest</td>
<td>Nearest neighbour interpolation</td>
</tr>
<tr>
<td>line, linear</td>
<td>Bilinear interpolation, this is the default</td>
<td></td>
</tr>
<tr>
<td>cube, cubic</td>
<td>Bicubic interpolation</td>
<td></td>
</tr>
<tr>
<td>lanc, lanczos</td>
<td>Lanczos interpolation</td>
<td></td>
</tr>
<tr>
<td>sp16, spline16</td>
<td>Spline16 interpolation</td>
<td></td>
</tr>
<tr>
<td>gauss, gaussian</td>
<td>Gaussian interpolation</td>
<td></td>
</tr>
<tr>
<td>w, h</td>
<td>Width and height of the output video, default size depends on output format</td>
<td></td>
</tr>
<tr>
<td>yaw, pitch, roll</td>
<td>in degrees</td>
<td>Rotation angles</td>
</tr>
<tr>
<td>rorder</td>
<td>'ypr', 'yrp', 'pyr', 'rpy', 'rpy'</td>
<td>Set the rotation order, default is 'ypr'</td>
</tr>
<tr>
<td></td>
<td>0, 1</td>
<td></td>
</tr>
<tr>
<td>------------------</td>
<td>------------------------</td>
<td>-----------------------------------------------------------------</td>
</tr>
<tr>
<td>h_flip, v_flip</td>
<td>Flip the output</td>
<td>horizontally or vertically</td>
</tr>
<tr>
<td>d_flip</td>
<td>Flip the output</td>
<td>back / forward</td>
</tr>
<tr>
<td>ih_flip, iv_flip</td>
<td>Flip the input</td>
<td>horizontally or vertically</td>
</tr>
<tr>
<td>in_trans</td>
<td>Transpose the input</td>
<td></td>
</tr>
<tr>
<td>out_trans</td>
<td>Transpose the output</td>
<td></td>
</tr>
<tr>
<td>h_fov, v_fov, d_fov</td>
<td>in degrees</td>
<td>Set the horizontal, vertical or diagonal field of view for output</td>
</tr>
<tr>
<td>ih_fov, iv_fov, d_fov</td>
<td>in degrees</td>
<td>Set the horizontal, vertical or diagonal field of view for input</td>
</tr>
</tbody>
</table>

Undocumented feature of the v360 filter: The top left pixel of the input video is mapped to all those pixels in the output video, which get no input data. If you want to give the unused area a specific color, you can just fill the top left pixel of the input video with this color:

```
vf drawbox=w=1:h=1:color=green,v360=...```

## 2.49 Equirectangular images of the night sky

Equirectangular images of the night sky can be found here:

- [http://paulbourke.net/miscellaneous/astronomy/](http://paulbourke.net/miscellaneous/astronomy/)
- [https://svs.gsfc.nasa.gov/vis/a000000/a003500/a003572/](https://svs.gsfc.nasa.gov/vis/a000000/a003500/a003572/)
- [https://sci.esa.int/web/gaia/-/60196-gaia-s-sky-in-colour-equirectangular-projection](https://sci.esa.int/web/gaia/-/60196-gaia-s-sky-in-colour-equirectangular-projection)
2.50 Remap a fisheye video to an equirectangular video

In this example the xmap and ymap files for the remap filter are created by FFmpeg (no C# code required).
The size of the equirectangular video is defined by the user and can be different from 2:1.

```
set "FF=c:\ffmpeg\ffmpeg"     :: Path to FFmpeg
set "IN=110_0001.mp4"         :: Input video
set "SQ=2880"                 :: Size of square input video
set "SR=1440"                 :: Radius that is actually used from the source video, must be SQ/2 or smaller
set "PW=1920"                 :: Width of panorama video
set "PH=550"                  :: Height of panorama video
set "OUT=out.mp4"             :: Output video

rem Create the xmap file

%FF% -f lavfi -i nullsrc=size=%PW%x%PH% -vf format=pix_fmts=gray16le,geq='%SQ%/2-Y*%SR%/%PH%*sin(X*2*PI/%PW%)' -frames 1 -y xmap.pgm

rem Create the ymap file

%FF% -f lavfi -i nullsrc=size=%PW%x%PH% -vf format=pix_fmts=gray16le,geq='%SQ%/2-Y*%SR%/%PH%*cos(X*2*PI/%PW%)' -frames 1 -y ymap.pgm

rem Apply the remap filter to the video

%FF% -i %IN% -i xmap.pgm -i ymap.pgm -lavfi "format=pix_fmts=rgb24,remap" -c:v mpeg4 -q:v 2 -y %OUT%
```

If the fisheye lens has more than 180° field of view, but you want only 180° visible in the panorama, set the SR variable to a value smaller than SQ/2.

A lot of informations about fisheye projections can be found on Paul Bourke's website: [www.paulbourke.net/dome/](http://www.paulbourke.net/dome/)

More informations about the remap filter can be found here: [https://trac.ffmpeg.org/wiki/RemapFilter](https://trac.ffmpeg.org/wiki/RemapFilter)
The VLC player won't recognize the output video as a spherical equirectangular video, because some special metadata is missing. This metadata can't be inserted with FFmpeg, but it can be done with this application: 
https://github.com/google/spatial-media/releases/tag/v2.1
In this example the fisheye's field of view can be set to any value up to 360°, and the width/height ratio of the equirectangular output video is always 2:1. The lower part is filled with black if the fisheye has less than 360° field of view.

```
set "FF=c:\ffmpeg\ffmpeg"   :: Path to FFmpeg
set "IN=IMG_077.jpg"        :: Fisheye input image or video, must be square
set "SQ=3648"               :: Size of square fisheye input image
set "FOV=220"               :: Fisheye field of view in degrees
set "Q=2"                   :: Size divider for output image, use 1 for best quality, 
                             :: or a bigger value for faster computing
set /a "H=%SQ%/%Q%"         :: Height of equirectangular image
set /a "W=2*%H%"            :: Width of equirectangular image is always twice the height
set /a "A=%H%*%FOV%/360"    :: Height of equirectangular image that is actually filled with data, 
                             :: the lower part of the output image remains black
set "OUT=out.jpg"           :: Equirectangular output image or video

rem Create the xmap file for remapping from fisheye to equirectangular
%FF% -f lavfi -i nullsrc=size=%W%x%H% -vf format=pix_fmts=gray16le,^ 
geq='%SQ%/2*(1-Y/%A%*sin(X*2*PI/%W%))' -frames 1 -y xmap1.pgm

rem Create the ymap file for remapping from fisheye to equirectangular
%FF% -f lavfi -i nullsrc=size=%W%x%H% -vf format=pix_fmts=gray16le,^ 
geq='%SQ%/2*(1-Y/%A%*cos(X*2*PI/%W%))' -frames 1 -y ymap1.pgm

rem Remap from fisheye to equirectangular
%FF% -i %IN% -i xmap1.pgm -i ymap1.pgm -filter_complex "format=pix_fmts=rgb24,remap" -y %OUT%
pause
```
For a square 180° single-fisheye video the conversion to an equirectangular video can also be done with the V360 filter. The second hemisphere is filled with a user-defined color. This example is obsolete, please use the next example.

```plaintext
set "FF=c:\ffmpeg\ffmpeg"     :: Path to FFmpeg
set "IN=in.mp4"               :: Fisheye input video (square, camera pointing upwards)
set "OUT=out.mp4"             :: Equirectangular output video

%FF% -i %IN% -lavfi "pad=w=2*iw:color=darkgreen,v360=input=dfisheye:output=e:pitch=90" -y %OUT%
```

Square single-fisheye images or videos with any field of view can be converted to equirectangular images or videos:

```plaintext
set "FF=c:\ffmpeg\ffmpeg"     :: Path to FFmpeg
set "IN=1200.png"             :: Input image or video
set "FOV=180"                 :: Input field of view in degrees
set "C=green"                 :: Color for filling unused area
set "OUT=out.png"             :: Equirectangular output image or video

%FF% -i %IN% -vf drawbox=w=1:h=1:color=%C%,v360=input=fisheye:id_fov=%FOV%:output=equirect:pitch=-90 -y %OUT%
```

If required, the lower part of the equirectangular output can be cut off with the crop filter.
2.51 Remap an equirectangular video to a fisheye video

The field of view can be set between 1 and 360 degrees. The sky is in the center of the fisheye video, and the ground is at the circular edge. The input video must have 2:1 width/height ratio.

```
set "FF=c:\ffmpeg\ffmpeg"     :: Path to FFmpeg
set "IN=test1.mp4"            :: Input video
set "H=960"                   :: Height of equirectangular input video
set "S=1080"                  :: Size of square fisheye output video
set "FOV=220"                 :: Set the field of view in degrees
set "OUT=fish.mp4"            :: Output video

rem Create the xmap file
%FF% -f lavfi -i nullsrc=size=%S%x%S% -vf format=pix_fmts=gray16le,^geq='%H%*(0.9999+atan2(X-%S%/2,Y-%S%/2)/PI)' -frames 1 -y xmap.pgm

rem Create the ymap file
%FF% -f lavfi -i nullsrc=size=%S%x%S% -vf format=pix_fmts=gray16le,^geq='%H%/360*%FOV%*(hypot((2*X/%S%)-1,(2*Y/%S%)-1))/PI)' -frames 1 -y ymap.pgm

rem Apply the remap filter to the video
%FF% -i %IN% -i xmap.pgm -i ymap.pgm -lavfi "format=pix_fmts=rgb24,remap" -q:v 2 -y %OUT%
```

The same thing can also be done with the v360 filter:

```
set "FF=c:\ffmpeg\ffmpeg"     :: Path to FFmpeg
set "IN=equirectangular.png"  :: Input image or video
set "FOV=220"                 :: Output field of view in degrees
set "OUT=fish.png"            :: Output image or video

%FF% -i %IN% -vf v360=input=equirect:output=fisheye:h_fov=%FOV%:v_fov=%FOV%:pitch=90 -y %OUT%
```
2.52 Remap an equirectangular video to a "Little planet" video

Fisheye projection is used. The ground is in the center of the video, and the sky is at the circular edge. The input video must have 2:1 width/height ratio.

```bash
set "FF=c:\ffmpeg\ffmpeg" :: Path to FFmpeg
set "IN=test3.mp4" :: Equirectangular input video
set "H=960" :: Height of input video (width = 2 * height)
set "S=1080" :: Size of square little planet output video
set "OUT=out.mp4" :: Output video

rem Create the xmap file
%FF% -f lavfi -i nullsrc=size=%S%x%S% -vf format=pix_fmts=gray16le,^ 
geq='H%*(0.9999+atan2(Y-%S%/2,X-%S%/2)/PI)' -frames 1 -y xmap.pgm

rem Create the ymap file
%FF% -f lavfi -i nullsrc=size=%S%x%S% -vf format=pix_fmts=gray16le,^ 
geq='H%*(1-hypot((2*X/%S%-1),(2*Y/%S%-1)))' -frames 1 -y ymap.pgm

rem Apply the remap filter to the video
%FF% -i %IN% -i xmap.pgm -i ymap.pgm -lavfi "format=pix_fmts=rgb24,remap=fill=green" -q:v 2 -y %OUT%
```

The values in the xmap and ymap files can't be negative. If a value is larger than the size of the input image, this pixel is painted with the color that's specified by the "fill" option.
If you want the sky in the center and the ground at the circular edge, use these remap functions instead:

```bash
%FF% -f lavfi -i nullsrc=size=%S%x%S% -vf format=pix_fmts=gray16le,^'geq='H%*(0.9999.atan2(Y-%S%/2,X-%S%/2)/PI)' -frames 1 -y xmap.pgm

%FF% -f lavfi -i nullsrc=size=%S%x%S% -vf format=pix_fmts=gray16le,^'geq='H%*(hypot((2*X/%S%)-1,(2*Y/%S%)-1))' -frames 1 -y ymap.pgm
```

The same thing can also be done with the v360 filter:

```bash
set "FF=c:\ffmpeg\ffmpeg"     :: Path to FFmpeg
set "IN=test1.png"            :: Input image or video
set "FOV=360"                 :: Output field of view in degrees
set "OUT=littleplanet.png"    :: Output image or video

%FF% -i %IN% -vf v360=input=equirect:output=fisheye:h_fov=%FOV%:v_fov=%FOV%:pitch=-90 -y %OUT%
```
Remap an equirectangular video to a "Mirror sphere" video

Similar to "Little planet", but using a different projection. The 360° world is shown as a reflection on a mirror sphere. The ground is in the center of the video, and the sky is at the circular edge. The input video must have 2:1 width/height ratio.

```plaintext
set "FF=c:\ffmpeg\ffmpeg"          :: Path to FFmpeg
set "IN=equirectangular_test.png"  :: Equirectangular input video
set "H=1200"                       :: Height of input video (width = 2 * height)
set "S=900"                        :: Size of square mirror sphere output video
set "OUT=mirror.png"               :: Output video

rem Create the xmap file

%FF% -f lavfi -i nullsrc=size=%S%x%S% -vf format=pix_fmts=gray16le,^   
g eq='\%H\%*(0.9999+atan2(Y-%S%/2,X-%S%/2)/PI)' -frames 1 -y xmap.pgm

rem Create the ymap file

%FF% -f lavfi -i nullsrc=size=%S%x%S% -vf format=pix_fmts=gray16le,^   
g eq='\%H\%*(1-2/PI*asin(hypot((2*X/%S%-1,(2*Y/%S%-1)))/PI))' -frames 1 -y ymap.pgm

rem Apply the remap filter to the video

%FF% -i %IN% -i xmap.pgm -i ymap.pgm -lavfi "format=pix_fmts=rgb24,remap" -q:v 2 -y %OUT%

pause

If you want the sky in the center and the ground at the circular edge, use these remap functions instead:

```plaintext
%FF% -f lavfi -i nullsrc=size=%S%x%S% -vf format=pix_fmts=gray16le,^  
g eq='\%H\%*(0.9999+atan2(Y-%S%/2,X-%S%/2)/PI)' -frames 1 -y xmap.pgm

%FF% -f lavfi -i nullsrc=size=%S%x%S% -vf format=pix_fmts=gray16le,^  
g eq='\%H\%*(2/PI*asin(hypot((2*X/%S%-1,(2*Y/%S%-1))))/PI))' -frames 1 -y ymap.pgm
```
The same thing can also be done with the "ball" output format of the v360 filter:

```
set "FF=c:\ffmpeg\ffmpeg"
set "IN=test1.png"
set "OUT=mirror.png"

%FF% -i %IN% -lavfi "v360=input=e:output=ball:pitch=90" -q:v 2 -y mirror.png
```

Pause

Pitch=90 is for the sky in the center, pitch=-90 is for the ground in the center.
Shift the viewing direction in a fisheye image or video

When you want to create a timelapse of many fisheye images, it may happen that one of the images isn't aligned correctly because the viewing direction of the camera was off. With normal (non-fisheye) images that isn’t a big problem, because you can simply re-align the image by shifting it in x and y directions. However for fisheye images things are much more complicated. The required procedure is as follows:

1. Remap the fisheye image to an equirectangular 360° image. The lower part of the image remains black.

2. Apply two rotations to this equirectangular image.

3. Remap the equirectangular image back to a fisheye image.

```
set "FF=c:\ffmpeg\ffmpeg"   :: Path to FFmpeg
set "IN=IMG_077.jpg"        :: Input image or video
set "S=3648"                :: Size of square fisheye input image
set "FOV=180"               :: Fisheye field of view in degrees
set "X=15"                  :: Rotation angle around X axis
set "Y=0"                   :: Rotation angle around Y axis
set "Q=5"                   :: Size divider for the intermediate equirectangular image, use 1 for best quality, or a bigger value for faster computing
set /a "H=%S%/%Q%"          :: Height of equirectangular image
set /a "W=2*%H%"            :: Width of equirectangular image is always twice the height
set /a "A=%H%*%FOV%/360"    :: Height of equirectangular image that is actually filled with data, the rest remains black
set "OUT=out.jpg"           :: Output image or video

rem Create the xmap file for remapping from fisheye to equirectangular
%FF% -f lavfi -i nullsrc=size=%W%x%H% -vf format=pix_fmts=gray16le,^
geq='%S%/2*(1-Y/%A%*sin(X*2*PI/%W%))' -frames 1 -y xmap1.pgm

rem Create the ymap file for remapping from fisheye to equirectangular
%FF% -f lavfi -i nullsrc=size=%W%x%H% -vf format=pix_fmts=gray16le,^
geq='%S%/2*(1-Y/%A%*cos(X*2*PI/%W%))' -frames 1 -y ymap1.pgm
```
rem Create the xmap file for remapping from equirectangular to fisheye

%FF% -f lavfi -i nullsrc=size=%S%x%S% -vf format=pix_fmts=gray16le,^ geq='%H%*(0.9999+atan2(X-%S%/2,Y-%S%/2)/PI)' -frames 1 -y xmap2.pgm

rem Create the ymap file for remapping from equirectangular to fisheye

%FF% -f lavfi -i nullsrc=size=%S%x%S% -vf format=pix_fmts=gray16le,^ geq='%H%/360*%FOV%*(hypot((2*X/%S%)-1,(2*Y/%S%)-1))'~frames 1 -y ymap2.pgm

rem Remap from fisheye to equirectangular, apply the rotations, then remap back to fisheye

%FF% -i %IN% -i xmap1.pgm -i ymap1.pgm -i xmap2.pgm -i ymap2.pgm -filter_complex "format=pix_fmts=rgb24,remap,v360=pitch=+%Y%:roll=+%X%:output=e[5];[5][3][4]remap" -y %OUT%

pause

The same thing can also be done with the v360 filter. In this example the top left pixel of the input image or video is set to a specific color with the "drawbox" filter. This color is used for all those pixels in the output file, that aren't mapped to a pixel in the input file. Please note that this is an undocumented feature of the v360 filter and it's not guaranteed that it works in all cases.

set "FF=c:\ffmpeg\ffmpeg" :: Path to FFmpeg
set "IN=1200.png" :: Input image or video
set "FOV=180" :: Field of view in degrees
set "PITCH=0" :: Rotation angle around X axis
set "YAW=30" :: Rotation angle around Y axis
set "C=green" :: Color for filling unused area
set "OUT=out.png" :: Output image or video

%FF% -i %IN% -vf drawbox=w=1:h=1:color=%C%,v360=input=fisheye:ih_fov=%FOV%:iv_fov=%FOV%:output=fisheye:h_fov=%FOV%:v_fov=%FOV%:yaw=%YAW%:pitch=%PITCH% -y %OUT%

codepage=932 pause
The v360 filter does have the "alpha_mask" option. If this option is set, all unused pixels in the output file are set to maximum transparency, so that the overlay filter can be used for filling this area with a color. This example does exactly the same thing as the previous example. Decide yourself which one is easier or faster:

```
set "FF=c:\ffmpeg\ffmpeg" :: Path to FFmpeg
set "IN=1200.png" :: Input image or video
set "FOV=180" :: Field of view in degrees
set "PITCH=0" :: Rotation angle around X axis
set "YAW=30" :: Rotation angle around Y axis
set "C=green" :: Color for filling unused area
set "OUT=out.png" :: Output image or video

%FF% -i %IN% -f lavfi -i color=%C%:s=1200x1200 -filter_complex v360=input=fisheye:ih_fov=%FOV%:iv_fov=%FOV%::output=fisheye:h_fov=%FOV%:v_fov=%FOV%:yaw=%YAW%:pitch=%PITCH%:alpha_mask=1[a],[1][a]:overlay -frames 1 -y %OUT%
pause
```

If the input is a video, remove the -frames 1 option.

See also [www.paulbourke.net/dome/fishtilt/](http://www.paulbourke.net/dome/fishtilt/)
Stitching together double-fisheye videos

The result is an equirectangular panorama video.

```plaintext
set "FF=c:\ffmpeg\ffmpeg"     :: Path to FFmpeg
set "IN=double_fisheye.jpg"   :: Input video or picture
set "X1=198"                  :: X coordinate of center of left fisheye image
set "Y1=210"                  :: Y coordinate of center of left fisheye image
set "X2=595"                  :: X coordinate of center of right fisheye image
set "Y2=210"                  :: Y coordinate of center of right fisheye image
set "SR=192"                  :: Radius that is actually used from the source video
set "PW=1920"                 :: Width of panorama video
set "PH=960"                  :: Height of panorama video
set "OUT=out.jpg"             :: Output video or picture

rem  Create the xmap file
%FF% -f lavfi -i nullsrc=size=%PW%x%PH% -vf format=pix_fmts=gray16le,^
geq='if(lt(Y,%PH%/2),%X1%-Y*2*%SR%/%PH%*sin(X*2*PI/%PW%),%X2%+(%PH%-Y)*2*%SR%/%PH%*sin(X*2*PI/%PW%))' -frames 1 -y
xmap.pgm

rem  Create the ymap file
%FF% -f lavfi -i nullsrc=size=%PW%x%PH% -vf format=pix_fmts=gray16le,^
geq='if(lt(Y,%PH%/2),%Y1%-Y*2*%SR%/%PH%*cos(X*2*PI/%PW%),%Y2%-%(PH%-Y)*2*%SR%/%PH%*cos(X*2*PI/%PW%))' -frames 1 -y
ymap.pgm

rem  Apply the remap filter to the video
%FF% -i %IN% -i xmap.pgm -i ymap.pgm -lavfi "format=pix_fmts=rgb24,remap" -q:v 2 -y %OUT%
pause
```

The parameters X1, Y1, X2, Y2 and SR must be carefully adjusted (by try and error) to get a good stitching result. They depend on the size of the source video or picture. Use these values as a starting point: X1=width/4, Y1=height/2, X2=width*3/4, Y2=height/2, SR=height/2. The following table shows how the parameters affect the stitching.

Note: The same thing can also be done with the V360 filter.
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Result when decreasing the parameter</th>
<th>Result when increasing the parameter</th>
</tr>
</thead>
<tbody>
<tr>
<td>X1</td>
<td>+-------------------------------+</td>
<td>+-------------------------------+</td>
</tr>
<tr>
<td></td>
<td></td>
<td>upper half from left fisheye</td>
</tr>
<tr>
<td></td>
<td></td>
<td>+-------------------------------+</td>
</tr>
<tr>
<td></td>
<td></td>
<td>lower half from right fisheye</td>
</tr>
<tr>
<td></td>
<td>+-------------------------------+</td>
<td>+-------------------------------+</td>
</tr>
<tr>
<td>Y1</td>
<td>+-------------------------------+</td>
<td>+-------------------------------+</td>
</tr>
<tr>
<td></td>
<td></td>
<td>upper half from left fisheye</td>
</tr>
<tr>
<td></td>
<td></td>
<td>+-------------------------------+</td>
</tr>
<tr>
<td></td>
<td></td>
<td>lower half from right fisheye</td>
</tr>
<tr>
<td></td>
<td>+-------------------------------+</td>
<td>+-------------------------------+</td>
</tr>
<tr>
<td>X2</td>
<td>+-------------------------------+</td>
<td>+-------------------------------+</td>
</tr>
<tr>
<td></td>
<td></td>
<td>upper half from left fisheye</td>
</tr>
<tr>
<td></td>
<td></td>
<td>+-------------------------------+</td>
</tr>
<tr>
<td></td>
<td></td>
<td>lower half from right fisheye</td>
</tr>
<tr>
<td></td>
<td>+-------------------------------+</td>
<td>+-------------------------------+</td>
</tr>
<tr>
<td>Y2</td>
<td>+-------------------------------+</td>
<td>+-------------------------------+</td>
</tr>
<tr>
<td></td>
<td></td>
<td>upper half from left fisheye</td>
</tr>
<tr>
<td></td>
<td></td>
<td>+-------------------------------+</td>
</tr>
<tr>
<td></td>
<td></td>
<td>lower half from right fisheye</td>
</tr>
<tr>
<td></td>
<td>+-------------------------------+</td>
<td>+-------------------------------+</td>
</tr>
<tr>
<td>SR</td>
<td>+-------------------------------+</td>
<td>+-------------------------------+</td>
</tr>
<tr>
<td></td>
<td></td>
<td>upper half from left fisheye</td>
</tr>
<tr>
<td></td>
<td></td>
<td>+-------------------------------+</td>
</tr>
<tr>
<td></td>
<td></td>
<td>lower half from right fisheye</td>
</tr>
<tr>
<td></td>
<td>+-------------------------------+</td>
<td>+-------------------------------+</td>
</tr>
</tbody>
</table>
2.56 Remove vertical stitching artefacts

When double-fisheye images are stitched together to an equirectangular image, it’s possible that stitching artefacts are visible as two vertical lines where the luminance from the two images doesn’t fit together. These artefacts can be removed by applying a suitable luminance gradient at one or both sides of the border. This example applies the gradient to the left side of two vertical borders:

```
set "FF=c:\ffmpeg\ffmpeg"   :: Path to FFmpeg
set "IN=fli0z.png"          :: Input image
set "B1=250"                :: Right side of first vertical border, left side is at B1-1
set "B2=750"                :: Right side of second vertical border, left side is at B2-1
set "W=25"                  :: Width of interpolation area

%FF% -i %IN% -vf "geq=cb_expr='cb(X,Y)'::cr_expr='cr(X,Y)'::lum_expr='clip(lum(X,Y)+between(X,%B1%-1-%W%,%B1%-1)*lerp(0,lum(%B1%,Y)-lum(%B1%-1,Y),(X-%B1%-1+%W%)/%W%)+between(X,%B2%-1-%W%,%B2%-1)*lerp(0,lum(%B2%,Y)-lum(%B2%-1,Y),(X-%B2%-1+%W%)/%W%),0,255)',format=rgb24" -y out.png
```

How it works:

In the area of width W to the left side of the vertical border, a ramp is added to the luminance. The amplitude of this ramp equals the difference of the luminance values left and right of the border.
You have to know in advance where exactly the vertical borders are.
Same as previous example, but now applying the gradient to the left side of the first border and to the right side of the second border:

```bash
set "FF=c:\ffmpeg\ffmpeg" :: Path to FFmpeg
set "IN=fli0z.png" :: Input image
set "B1=250" :: Right side of first vertical border, left side is at B1-1
set "B2=750" :: Right side of second vertical border, left side is at B2-1
set "W=25" :: Width of interpolation area

%FF% -i %IN% -vf "geq=cb_expr='cb(X,Y)':cr_expr='cr(X,Y)':lum_expr='clip(lum(X,Y)+
between(X,%B1%-1-%W%,%B1%-1)*lerp(0.0,lum(%B1%,Y)-lum(%B1%-1,Y),(X-%B1%-1+%W%)/%W%)+
between(X,%B2%-1-%W%,%B2%-1)*lerp(lum(%B2%,Y)-lum(%B2%-1,Y),0,(X-%B2%-1+%W%)/%W%)+
between(X,%B1%,%B1%+%W%)*lerp(lum(%B1%-1,Y)-lum(%B1%,Y),0,(X-%B1%)/%W%)+
between(X,%B2%,%B2%+%W%)*lerp(lum(%B2%-1,Y)-lum(%B2%,Y),0,(X-%B2%)/%W%)),0,255)',format=rgb24" -y out.png
```

Same as previous examples, but now applying half of the gradient to the left side and the other half to the right side of both borders:

```bash
set "FF=c:\ffmpeg\ffmpeg" :: Path to FFmpeg
set "IN=fli0z.png" :: Input image
set "B1=250" :: Right side of first vertical border, left side is at B1-1
set "B2=750" :: Right side of second vertical border, left side is at B2-1
set "W=25" :: Half width of interpolation area

%FF% -i %IN% -vf "geq=cb_expr='cb(X,Y)':cr_expr='cr(X,Y)':lum_expr='clip(lum(X,Y)+0.5*(
between(X,%B1%-1-%W%,%B1%-1)*lerp(0.0,lum(%B1%,Y)-lum(%B1%-1,Y),(X-%B1%-1+%W%)/%W%)+
between(X,%B2%-1-%W%,%B2%-1)*lerp(0.0,lum(%B2%,Y)-lum(%B2%-1,Y),(X-%B2%-1+%W%)/%W%)+
between(X,%B1%,%B1%+%W%)*lerp(lum(%B1%-1,Y)-lum(%B1%,Y),0,(X-%B1%)/%W%)+
between(X,%B2%,%B2%+%W%)*lerp(lum(%B2%-1,Y)-lum(%B2%,Y),0,(X-%B2%)/%W%)),0,255)',format=rgb24" -y out.png
```

Remove the line feeds from the command line, which were only inserted for clarity.

Please note that workarounds with geq filter are quite slow.
This is an example for merging two overlapping fisheye videos, realized with the "maskedmerge" filter:

```plaintext
set "FF=c:\ffmpeg\ffmpeg" :: Path to FFmpeg
set "IN=double_fisheye.mp4" :: Input video
set "H=640" :: Height of input video
set "FOV=191.5" :: Horizontal and vertical field of view of the fisheye lenses in degrees
set "C=11.5" :: Width of interpolation band in degrees, must be smaller or equal than (FOV-180°)
set "T=10" :: Duration in seconds
set "OUT=out.mp4" :: Output video

rem Create the mergemap file

%FF% -f lavfi -i nullsrc=size=%H%x%H% -vf "format=gray8,geq='clip(128-128/%C%*(180-%FOV%/(%H%/2)*hypot(X-%H%/2,Y-%H%/2)),0,255)',v360=input=fisheye:output=e:ih_fov=%FOV%:iv_fov=%FOV%,format=rgb24" -frames 1 -y mergemap.png

rem Merge the two fisheye images from the double-fisheye input video

%FF% -i %IN% -i mergemap.png -lavfi ":0[0]format=rgb24,split[a][b];[a]crop=ih:iw/2:0:0,v360=input=fisheye:output=e:ih_fov=%FOV%:iv_fov=%FOV%[c];[b]crop=ih:iw/2:iw/2:0,v360=input=fisheye:output=e:yaw=180:ih_fov=%FOV%:iv_fov=%FOV%[d];[c][d][1]maskedmerge,format=rgb24" -t %T% -y %OUT%
```

Tested with this input video, downloaded in 1280x640 size: [https://www.youtube.com/watch?v=70Wd7Ex54jE](https://www.youtube.com/watch?v=70Wd7Ex54jE)

Note: It seems that maskedmerge has a different output format than the three inputs, even if all three inputs have the same pixel format. That's why format=rgb24 is required after maskedmerge.

Note: The FOV variable must be set to the correct field of view of the fisheye lenses. Find the best value by try and error.
2.57 Preprocessing a flat video for fulldome projection

If a flat video is to be shown in a fulldome planetarium with a fisheye projector, some preprocessing is required. The video is downscaled to a smaller size, padded with large black borders to equirectangular 2:1 format, rotated with the v360 filter, and then given out in 180° fisheye output.

```bash
set "FF=c:\ffmpeg\ffmpeg" :: Path to FFmpeg
set "IN=pk14.mp4"             :: Input video
set "UP=35"                   :: Up-looking angle in degrees (center of the rectangular video)
set "W=480"                   :: Width of input video after downsampling, this is for 16:9 aspect ratio
set "H=270"                   :: Height of input video after downsampling, this is for 16:9 aspect ratio
set "S=1200"                  :: Size of square fisheye output video
set "OUT=out.mp4"             :: Output video
%FF% -i %IN% -lavfi "scale=%W%:%H%,pad='2*%S%':%S%:-1:- 1,format=pix_fmts=rgb24,v360=input=equirect:output=fisheye:h_fov=180:v_fov=180:pitch='90-%UP%'" -y %OUT%
pause
```

It's also possible to use the flat video directly as input for the v360 filter. This has the problem that the unused area is filled with a random color (coming from the top left pixel of the input video). As a workaround, this pixel is filled with black before using the v360 filter:

```bash
set "FF=c:\ffmpeg\ffmpeg" :: Path to FFmpeg
set "IN=pk14.mp4"             :: Input video
set "UP=30"                   :: Up-looking angle in degrees (center of the rectangular video)
set "H=64"                    :: Horizontal field of view, this is for 16:9 aspect ratio
set "V=36"                    :: Vertical field of view, this is for 16:9 aspect ratio
set "OUT=out.mp4"             :: Output video
%FF% -i %IN% -vf drawbox=w=1:h=1:color=black,v360=input=flat:ih_fov=%H%:iv_fov=%V %:output=fisheye:h_fov=180:v_fov=180:pitch='90-%UP%' -y %OUT%
pause
```
With sufficient computing power live processing is possible. Just drag and drop the input video over the icon of this batch file:

```batch
set "FF=c:\ffmpeg\ffmpeg" :: Path to FFmpeg
set "UP=30"                :: Up-looking angle in degrees (center of the rectangular video)
set "H=64"                 :: Horizontal field of view, this is for 16:9 aspect ratio
set "V=36"                 :: Vertical field of view, this is for 16:9 aspect ratio

%FF% -re -i %1 -vf drawbox=w=1:h=1:color=black,v360=input=flat:ih_fov=%H%:iv_fov=%V
%:output=fisheye:h_fov=180:v_fov=180:pitch='90-%UP%' -window_fullscreen 1 -f sdl2 -
```

Please note that the sdl2 output doesn't play audio. The Windows taskbar remains visible in fullscreen mode. You can hide it as follows: Make a right click on the taskbar, click on "properties" and then select "automatically hide taskbar".

This is an example for live processing and passing the output to FFplay. Just drag and drop the input video over the icon of this batch file. FFplay has the advantage that it does also play audio, and the Windows taskbar is automatically hidden:

```batch
set "FF=c:\ffmpeg\ffmpeg" :: Path to FFmpeg
set "UP=30"                :: Up-looking angle in degrees (center of the rectangular video)
set "H=64"                 :: Horizontal field of view, this is for 16:9 aspect ratio
set "V=36"                 :: Vertical field of view, this is for 16:9 aspect ratio

%FF% -re -i %1 -vf drawbox=w=1:h=1:color=black,v360=input=flat:ih_fov=%H%:iv_fov=%V
%:output=fisheye:h_fov=180:v_fov=180:pitch='90-%UP%' -q:v 2 -c:v mpeg4 -f nut - | c:\ffmpeg
\ffplay -fs -autoexit -
```

The -fs option means full screen, and -autoexit means that FFplay closes automatically when the end of the video has been reached.
2.58 Rotating earth or planet

set "FF=c:\ffmpeg\ffmpeg"  :: Path to FFmpeg
set "IN=Earth_eq.jpg"     :: Equirectangular image of earth or planet surface, for example from:
                          ::   https://de.wikipedia.org/wiki/Datei:Nasa_land_ocean_ice_8192.jpg
set "BG=Starfield.jpg"   :: Background image
set "P=-50"              :: Pitch angle
set "R=30"               :: Roll angle
set "S=0.005"            :: Rotation speed, 1.0 means one full revolution per frame
set "D=200"              :: Diameter of planet
set "X=900"              :: X position of planet
set "Y=450"              :: y position of planet
set "T=10"               :: Length of output video

%FF% -loop 1 -i %BG% -loop 1 -i %IN% -lavfi "[1]scroll=h=%S%,v360=e:perspective:pitch=%P%:roll=%R%:alpha_mask=1, scale=%D%.%D%[a][0][a], overlay=x=%X%:y=%Y%" -t %T% -y out.mp4

pause
2.59 Black hole simulation with remap filter

FFmpeg's remap filter can be used to simulate the light deviation near black holes. When a beam of light passes near a black hole, it will be deviated by angle alpha (in Radians):

\[ \alpha = \frac{2 \times rs}{(r - rs)} \]

where \(rs\) is the Schwarzschild radius of the black hole, and \(r\) is the closest distance between the beam and the center of the black hole.

Assuming we have a 180° fisheye image, we can express the light deviation in pixels:

\[ c = \frac{\text{height}}{\pi} \times \frac{2 \times rs}{(r - rs)} \]

The values for the PGM files (which are required for the remap filter) can be calculated with these formulas:

\[ r = \sqrt{(x - xc)^2 + (y - yc)^2} \]

\[ c = \frac{\text{shape}}{(r - rs)} \]

where \(shape\) is a constant that defines the "strength" of the distortion.

If \(r > rs\):

\[ x_{\text{remap}} = x - c \times (x - xc) \]
\[ y_{\text{remap}} = y - c \times (y - yc) \]

If \(r < rs\):

\[ x_{\text{remap}} = 0 \]
\[ y_{\text{remap}} = 0 \]

where \(xc, yc\) are the pixel coordinates of the center of the black hole, \(x, y\) are the pixel coordinates in the source video and \(r\) is the distance between the source pixel and the center of the black hole.
This is the batch file for applying the black-hole-effect to a video:

```
set "FF=c:\ffmpeg\ffmpeg"  :: Path to FFmpeg
set "IN=MVI_2562.mov"       :: Input video
set "OUT=output.mp4"        :: Output video

%FF% -i %IN% -i xmap.pgm -i ymap.pgm -lavfi "format=pix_fmts=rgb24,remap" -c:v mpeg4 -q:v 2 -y %OUT%
pause
```

It's also possible to simulate moving black holes. To do this you need many xmap and ymap files (one for each frame), and loop through them.

```
set "FF=c:\ffmpeg\ffmpeg"  :: Path to FFmpeg
set "IN=MVI_2562.mov"       :: Input video
set "OUT=output.mp4"        :: Output video

%FF% -i %IN% -framerate 30 -i xmap%%4d.pgm -framerate 30 -i ymap%%4d.pgm -lavfi "format=pix_fmts=rgb24,remap" -c:v mpeg4 -q:v 2 -y %OUT%
pause
```
This is a C# program for creating the xmap and ymap files for a moving black hole:

```csharp
using System;
using System.IO;
using System.Globalization;

namespace Moving_Black_Hole
{
    class Program
    {
        static void Main(string[] args)
        {
            int width = 1920; // Width of images
            int height = 1080; // Height of images
            double shape = 10; // Strength factor of black hole effect
            int count = 250; // Number of frames
            int framerate = 25; // Framerate for conversion of frame number to time
            double ignore = 0.1; // Central fraction of radius that's not used for mapping, can be deactivated by setting to 0
            // Use this parameter to hide the small white ball on the invisible tripod.
            // Example: If the black hole radius is 50 pixels and the small ball has 5 pixels radius,
            // set ignore to 0.1
            string path = @"F:\Wormhole_2020\"; // Path for writing the xmap and ymap files

            int[,] nxyr = new int[11, 4] // This array contains the interpolation data: n, x, y, r
            { // n is the frame number beginning with 0,
                // the last entry must not be smaller than the number of frames
                // x and y are the center coordinates of the black hole
                // r is the black hole's radius
                { 0, 1735, 527, 50 },
                { 25, 1723, 529, 50 },
                { 50, 1605, 526, 50 },
                { 75, 1440, 526, 50 },
                { 100, 1266, 523, 50 },
                { 125, 1113, 523, 50 },
                { 150, 1007, 522, 50 },
                { 175,  966, 522, 70 },
                { 200,  964, 522, 90 },
                { 225,  964, 522, 1 },
                { 250,  964, 522, 1 }
            };

            int xc, yc, radius, dx, dy, xb, yb;
            CultureInfo invC = CultureInfo.InvariantCulture;
```
TextWriter pos = File.CreateText(path + "positions.cmd"); // Write positions.cmd file
for (int i = 0; i < nxyr.GetLength(0) - 1; i++)
{
    int t0 = nxyr[i,0] / framerate;
    int t1 = nxyr[i+1,0] / framerate;
    int x0 = nxyr[i,1];
    int y0 = nxyr[i,2];
    int x1 = nxyr[i+1,1];
    int y1 = nxyr[i+1,2];
    int r0 = nxyr[i,3];
    int r1 = nxyr[i+1,3];
    pos.WriteLine(t0.ToString("F2", invC) + "-" + t1.ToString("F2", invC) +
    " overlay x \'lerp(" + (x0-r0).ToString(invC).PadLeft(4) +
    "," + (x1-r1).ToString(invC).PadLeft(4) + ",t-" + i.ToString() +
    ")', overlay y \'lerp(" + (y0-r0).ToString(invC).PadLeft(4) +
    "," + (y1-r1).ToString(invC).PadLeft(4) + ",t-" + i.ToString() + ")',
    ", scale w \'lerp(" + (2 * r0 + 1).ToString(invC).PadLeft(3) + "," +
    (2 * r1 + 1).ToString(invC).PadLeft(3) + ",t-" + i.ToString() + ")',
    ", scale h \'lerp(" + (2 * r0 + 1).ToString(invC).PadLeft(3) + "," +
    (2 * r1 + 1).ToString(invC).PadLeft(3) + ",t-" + i.ToString() + ")');
}
pos.Close();
for (int n = 0; n < count; n++)
{
    int i = 0;
    while ((n < nxyr[i, 0]) || (n > nxyr[i + 1, 0]))
    i++;
    xc = nxyr[i, 1] + (nxyr[i + 1, 1] - nxyr[i, 1]) * (n - nxyr[i, 0]) / (nxyr[i + 1, 0] - nxyr[i, 0]);
    yc = nxyr[i, 2] + (nxyr[i + 1, 2] - nxyr[i, 2]) * (n - nxyr[i, 0]) / (nxyr[i + 1, 0] - nxyr[i, 0]);
    radius = nxyr[i, 3] + (nxyr[i + 1, 3] - nxyr[i, 3]) * (n - nxyr[i, 0]) / (nxyr[i + 1, 0] - nxyr[i, 0]);
    Console.WriteLine("Writing mapping files for frame " + (n + 1).ToString() + " of " + count.ToString() + 
    " x: " + xc.ToString() + " y: " + yc.ToString() + " r: " + radius.ToString());

TextWriter xmap = File.CreateText(path + "xmap" + n.ToString("0000") + ".pgm");
xmap.Write("P2
");
xmap.Write("# Xmap file for FFmpeg remap \n");
xmap.Write(width.ToString() + " " + height.ToString() + " \n");
xmap.Write("65535\n");
TextWriter ymap = File.CreateText(path + "ymap" + n.ToString("0000") + ".pgm");
ymap.Write("P2\n");
ymap.Write("# Ymap file for FFmpeg remap \n");
ymap.Write(width.ToString() + " " + height.ToString() + " \n");
ymap.Write("65535\n");

for (int y = 0; y < height; y++)
{
    dy = y - yc;
    for (int x = 0; x < width; x++)
    {
        dx = x - xc;
        double r = Math.Sqrt(dx * dx + dy * dy);
        if (r > radius) // outer area of black hole
        {
            double c = shape / (r - radius);
            if (c > 1.0 - ignore) c += 2 * ignore;
            xb = x - (int)(1.0 + dx * c);
            yb = y - (int)(1.0 + dy * c);
            if (xb < 0) xb = 0;
            if (yb < 0) yb = 0;
            if (xb >= width) xb = width - 1;
            if (yb >= height) yb = height - 1;
        }
        else
        {
            xb = 65535; // inner area of black hole is declared as "unmapped"
            yb = 65535;
        }
        xmap.Write(xb + " ");
        ymap.Write(yb + " ");
    }
    xmap.Write("\n");
    ymap.Write("\n");
}
xmap.Close();
ymap.Close();

Console.WriteLine("All done!");
Example of a simulated black hole:
Black hole simulation with FFmpeg, no C# code required:

```
set "FF=c:\ffmpeg\ffmpeg" :: Path to FFmpeg
set "W=2448" :: Width of image
set "H=2448" :: Height of image
set "CX=2000" :: X center of distortion
set "CY=1200" :: Y center of distortion
set "RS=50" :: Schwarzschild radius
set "SH=0.50" :: Shape parameter

rem Create the xmap file
%FF% -f lavfi -i nullsrc=size=%W%x%H% -vf format=pix_fmts=gray16le,geq='st(0,X-%CX%);st(1,hypot(ld(0),%CY%-Y));st(2,X-(ld(0)*%SH%*2*%RS%/(ld(1)-%RS%));if(lt(%RS%,ld(1)),clip(ld(2),0,%W%),0)' -frames 1 -y xmap.pgm

rem Create the ymap file
%FF% -f lavfi -i nullsrc=size=%W%x%H% -vf format=pix_fmts=gray16le,geq='st(0,Y-%CY%);st(1,hypot(%CX%-X,ld(0)));st(2,Y-(ld(0)*%SH%*2*%RS%/(ld(1)-%RS%));if(lt(%RS%,ld(1)),clip(ld(2),0,%H%),0)' -frames 1 -y ymap.pgm

rem Apply the displace filter to the image
%FF% -i test3.mp4 -i xmap.pgm -i ymap.pgm -lavfi format=pix_fmts=rgb24,remap -frames 1 -y out.jpg

rem Alternatively all can be written in one command line:
%FF% -i test3.mp4 -f lavfi -i nullsrc=size=%W%x%H% -f lavfi -i nullsrc=size=%W%x%H% -lavfi [0]format=pix_fmts=rgb24[v];[1]format=pix_fmts=gray16le,geq='st(0,X-%CX%);st(1,hypot(ld(0),%CY%-Y));st(2,X-(ld(0)*%SH%*2*%RS%/(ld(1)-%RS%));if(lt(%RS%,ld(1)),clip(ld(2),0,%W%),0)' [x];[2]format=pix_fmts=gray16le,geq='st(0,Y-%CY%);st(1,hypot(%CX%-X,ld(0)));st(2,Y-(ld(0)*%SH%*2*%RS%/(ld(1)-%RS%));if(lt(%RS%,ld(1)),clip(ld(2),0,%H%),0)' [y];[v][x][y]remap -frames 1 -y out.jpg

pause
```
This example is for a moving black hole, no C# code required (but unfortunately this is extremely slow):

```
set "FF=c:\ffmpeg\ffmpeg" :: Path to FFmpeg
set "IN=test3.mp4" :: Input video
set "W=2448" :: Width of video
set "H=2448" :: Height of video
set "CX0=2000" :: X center of distortion, T=0
set "CY0=1200" :: Y center of distortion, T=0
set "CX1=1900" :: X center of distortion, T=1
set "CY1=1500" :: Y center of distortion, T=1
set "CX2=1600" :: X center of distortion, T=2
set "CY2=1800" :: Y center of distortion, T=2
set "CX3=1000" :: X center of distortion, T=3
set "CY3=2000" :: Y center of distortion, T=3
set "RS=50" :: Schwarzschild radius
set "SH=0.50" :: Shape parameter
set "OUT=out.mp4" :: Output video

%FF% -i %IN% -f lavfi -i nullsrc=size=%W%x%H% -f lavfi -i nullsrc=size=%W%x%H% -lavfi ^
[0]format=pix_fmts=rgb24[v];^ 
[1]format=pix_fmts=gray16le,geq='^ 
st(0,between(T+0.001,0,1)*lerp(%CX0%,%CX1%,T)+between(T+0.001,1,2)*lerp(%CX1%,%CX2%,T-1)+between(T+0.001,2,3)*lerp(%CX2%,%CX3%,T-2));^ 

st(1,between(T+0.001,0,1)*lerp(%CY0%,%CY1%,T)+between(T+0.001,1,2)*lerp(%CY1%,%CY2%,T-1)+between(T+0.001,2,3)*lerp(%CY2%,%CY3%,T-2));^ 

st(2,X-ld(0));

st(3,hypot(1d(2),1d(1)-Y));^ 

st(4,X-ld(2)*%SH%*2*%RS%/1d(3)-%RS%));^ 
if(1t(1R%S%,1d(3)),clip(1d(4),0,%W%),0)'[x];^ 

[2]format=pix_fmts=gray16le,geq='^ 

st(0,between(T+0.001,0,1)*lerp(%CX0%,%CX1%,T)+between(T+0.001,1,2)*lerp(%CX1%,%CX2%,T-1)+between(T+0.001,2,3)*lerp(%CX2%,%CX3%,T-2));^ 

st(1,between(T+0.001,0,1)*lerp(%CY0%,%CY1%,T)+between(T+0.001,1,2)*lerp(%CY1%,%CY2%,T-1)+between(T+0.001,2,3)*lerp(%CY2%,%CY3%,T-2));^ 

st(2,Y-ld(1));

st(3,hypot(1d(0)-X,1d(2)));^ 

st(4,Y-ld(2)*%SH%*2*%RS%/1d(3)-%RS%));^ 
if(1t(1R%S%,1d(3)),clip(1d(4),0,%H%),0)'[y];^ 

[v][w][y]remap -t 3 -y %OUT%
```

"T+0.001" is a workaround to avoid the problem that at the segment borders two "between" expressions become simultaneously true.

This method is extremely slow because this expression must be evaluated four times for each pixel, although it would be sufficient to evaluate it only one time per frame:

```
st(1,between(T+0.001,0,1)*lerp(%CY0%,%CY1%,T)+between(T+0.001,1,2)*lerp(%CY1%,%CY2%,T-1)+between(T+0.001,2,3)*lerp(%CY2%,%CY3%,T-2));
```

Recommended workaround: Calculate many xmap and ymap files in advance by C# code.
2.60  Wormhole simulation

A wormhole is a hypothetical window to another place in space or time, or even in another universe. For more informations please see https://en.wikipedia.org/wiki/Wormhole


A wormhole can be simulated in a video as follows:

- In the outer area the light rays are distorted in the same way as when passing near a black hole. This can be simulated with the remap filter.
- In the inner area, another video is inserted as a 360° "little planet" video (or even better a mirror-sphere video).
This is a batch file for wormhole simulation. The xmap0000 and ymap0000 files for the black hole are created in advance by C# code.

```
set "FF=c:\ffmpeg\ffmpeg" :: Path to FFmpeg
set "IN=main.mov"             :: Main input video
set "LP=test1.mp4"            :: Equirectangular video for little planet
set "H=960"                   :: Height of equirectangular input video
set "S=1080"                  :: Size of square little planet output video
set "P=0"                     :: Pitch angle
set "Y=90"                    :: Yaw angle
set "R=-90"                   :: Roll angle
set "LPD=100"                 :: Little planet diameter
set "LPX=1500"                :: X Position of center of little planet
set "LPY=1000"                :: Y Position of center of little planet
set "T=8"                     :: Length of output video

rem Step 1: Convert the equirectangular video to a little planet video
rem Create the xmap and ymap files

%FF% -f lavfi -i nullsrc=size=%S%x%S% -vf format=pix_fmts=gray16le,^  
geq='%H%*(0.9999+atan2(Y-%S%/2,X-%S%/2)/PI)' -frames 1 -y xmap.pgm

%FF% -f lavfi -i nullsrc=size=%S%x%S% -vf format=pix_fmts=gray16le,^  
geq='%H%*(1-hypot((2*X/%S%)-1,(2*Y/%S%)-1))' -frames 1 -y ymap.pgm

rem Apply the remap filter to the video

%FF% -i %LP% -i xmap.pgm -i ymap.pgm -lavfi "v360=output=e:pitch=%P%:yaw=%Y%:roll=%R%,format=pix_fmts=rgb24,remap" -q:v 2 -t %T% -y lp.mp4

rem Step 2: Apply the black hole effect to the main video and then overlay the little planet video over the black hole

%FF% -i %IN% -i lp.mp4 -i xmap0000.pgm -i ymap0000.pgm -filter_complex "[0][2][3]remap[4];[1]scale=%LPD%:%LPD%[5];[4][5]overlay=x=%LPX%-%LPD%/2:y=%LPY%-%LPD%/2" -q:v 2 -t %T% -y out.mp4

pause
```
The same thing can be done much easier with the v360 filter and the alpha_mask option:

```bash
set "FF=c:\ffmpeg\ffmpeg" :: Path to FFmpeg
set "IN=main.mov" :: Main input video
set "LP=test1.mp4" :: Equirectangular video for mirror-sphere
set "H=960" :: Height of equirectangular input video
set "S=1080" :: Size of square mirror-sphere output video
set "P=30" :: Pitch angle
set "Y=0" :: Yaw angle
set "R=0" :: Roll angle
set "LPD=102" :: Mirror-sphere diameter
set "LPX=1800" :: X Position of center of mirror-sphere
set "LPY=1000" :: Y Position of center of mirror-sphere
set "T=8" :: Length of output video

rem Make only the mirror-sphere video
rem %FF% -i %LP% -vf v360=output=ball:pitch=%P%:yaw=%Y%:roll=%R% -q:v 2 -t %T% -y lp.mp4

%FF% -i %IN% -i xmap0000.pgm -i ymap0000.pgm -i %LP% -filter_complex "[0][1][2]remap[4];[3]v360=output=ball:pitch=%P%:yaw=%Y%:roll=%R%:alpha_mask=1, scale=%LPD%:%LPD%;[4][5]overlay=x=%LPX%-%LPD%/2:y=%LPY%-%LPD%/2" -q:v 2 -t %T% -y out.mp4

pause
```
2.61 Simulation of a moving wormhole

If the wormhole shall move in the field of view, two things must move:

1. The black hole distortion, this requires many xmap and ymap files, created by C# code (shown in the black hole chapter).
2. The inserted mirror-sphere video, this can be realized with sendcmd and overlay filters.

Step 1: Extract a suitable number of frames from the main video:

```
set "FF=c:\ffmpeg\ffmpeg"  :: Path to FFmpeg
set "IN=in.mp4"            :: Input video
set "STEP=25"              :: Step width (number of frames)
set "OUT=image%%4d.jpg"    :: Output images filename

%FF% -i %IN% -vf framestep=%STEP% -start_number 0 -y %OUT%
```

Step 2: Measure the x,y position of the small ball on the invisible tripod in all these frames. This can be done with IrfanView.

Step 3: Enter the x,y positions into the C# code and calculate the xmap and ymap files for all frames.

Step 4: The file positions.cmd was also automatically created by C# code:

```
0.00-1.00 overlay x 'lerp(1685,1673,t-0)', overlay y 'lerp( 477, 479,t-0)', scale w 'lerp(101,101,t-0)', scale h 'lerp(101,101,t-0)';
1.00-2.00 overlay x 'lerp(1673,1555,t-1)', overlay y 'lerp( 479, 476,t-1)', scale w 'lerp(101,101,t-1)', scale h 'lerp(101,101,t-1)';
2.00-3.00 overlay x 'lerp(1555,1390,t-2)', overlay y 'lerp( 476, 473,t-2)', scale w 'lerp(101,101,t-2)', scale h 'lerp(101,101,t-2)';
3.00-4.00 overlay x 'lerp(1390,1216,t-3)', overlay y 'lerp( 473, 473,t-3)', scale w 'lerp(101,101,t-3)', scale h 'lerp(101,101,t-3)';
4.00-5.00 overlay x 'lerp(1216,1063,t-4)', overlay y 'lerp( 473, 473,t-4)', scale w 'lerp(101,101,t-4)', scale h 'lerp(101,101,t-4)';
5.00-6.00 overlay x 'lerp(1063, 957,t-5)', overlay y 'lerp( 473, 472,t-5)', scale w 'lerp(101,101,t-5)', scale h 'lerp(101,101,t-5)';
6.00-7.00 overlay x 'lerp( 957, 896,t-6)', overlay y 'lerp( 472, 452,t-6)', scale w 'lerp(101,141,t-6)', scale h 'lerp(101,141,t-6)';
7.00-8.00 overlay x 'lerp( 896, 874,t-7)', overlay y 'lerp( 452, 432,t-7)', scale w 'lerp(141, 3,t-7)', scale h 'lerp(141, 3,t-7)';
8.00-9.00 overlay x 'lerp( 874, 963,t-8)', overlay y 'lerp( 432, 521,t-8)', scale w 'lerp( 181, 3,t-8)', scale h 'lerp( 181, 3,t-8)';
9.00-10.00 overlay x 'lerp( 963, 963,t-9)', overlay y 'lerp( 521, 521,t-9)', scale w 'lerp(  3, 3,t-9)', scale h 'lerp(  3, 3,t-9)';
```
Step 5: Run this batch file to create the final moving wormhole video:

```
set "FF=c:\ffmpeg\ffmpeg" :: Path to FFmpeg
set "IN=in.mp4" :: Main input video
set "LP=lp.mp4" :: Equirectangular video for mirror-sphere
set "P=30" :: Pitch angle
set "Y=0" :: Yaw angle
set "R=0" :: Roll angle
set "V=9" :: Time when wormhole vanishes
set "T=10" :: Length of output video

rem Make only the mirror-sphere video
rem %FF% -i %LP% -vf v360=output=ball:pitch=%P%:yaw=%Y%:roll=%R% -t %T% -y lp.mp4

%FF% -i %IN% -i %LP% -start_number 0 -i xmap%4d.pgm -start_number 0 -i ymap%4d.pgm -lavfi "[0]sendcmd='%V% streamselect map 1',split[4][5];[4][2][3]remap=fill=red,sendcmd=f=positions.cmd[6];[1]v360=output=ball:pitch=%P%:yaw=%Y%:roll=%R%:alpha_mask=1, scale=w=10:h=10:eval=frame[7];[6][7]overlay=x=0:y=0:format=rgb[8];[8][5]streamselect=map=0" -t %T% -y out.mp4

pause
```

overlay=format=rgb is strongly required, because the default format yuv420 allows only to set the x,y coordinates in multiples of 2.
"remap=fill=red" is used here only to make alignment errors visible, if the overlay isn't exactly at the same position as the black hole distortion. Normally there should no red pixels be visible. After this check you can replace it by "remap=fill=black".
It's also possible to let the inner area of the wormhole rotate as a function of time. Two different rotations are applied in this example. The first rotation is using the scroll filter (applied to an equirectangular video, before the v360 filter) and the other is using the rotate filter after the v360 filter:

```plaintext
set "FF=c:\ffmpeg\ffmpeg" :: Path to FFmpeg
set "IN=in.mp4" :: Main input video
set "LP=lp.mp4" :: Equirectangular video for mirror-sphere
set "P=30" :: Pitch angle
set "Y=0" :: Yaw angle
set "R=0" :: Roll angle
set "R1=0.01" :: Rotation speed before v360 filter, 1.0 means one revolution per frame
set "R2=0.00" :: Rotation speed after v360 filter, 1.0 means one revolution per second
set "V=9" :: Time when wormhole vanishes
set "T=10" :: Length of output video
rem Make only the mirror-sphere video
rem %FF% -i %LP% -vf v360=output=ball:pitch=%P%:yaw=%Y%:roll=%R% -t %T% -y lp.mp4

%FF% -i %IN% -i %LP% -start_number 0 -i xmap%%4d.pgm -start_number 0 -i ymap%%4d.pgm -lavfi "[0]sendcmd='%V
streamselect map 1',split[4][5];[4][2][3]remap=fill=black,sendcmd=f=positions.cmd[6];[1]v360=e:e:pitch=60,scroll=h=%R1%,v360=output=ball:pitch=%P%:yaw=%Y%:roll=%R
%:alpha_mask=1,rotate='%R2%*2*P*T':c=black@0.0,eval=frame[7];[6][7]overlay=x=0:y=0:format=rgb[8];[8]
[5]streamselect=map=0" -t %T% -y out.mp4

pause
```

Note for scroll filter: scroll=h=1.0 means one full horizontal revolution per frame. So you have to know the framerate to set the rotation speed.
2.62 Sendcmd and commands

- sendcmd has many pitfalls and can drive you crazy!
- The sendcmd filter sends commands to another filter. For example in the previous chapter sendcmd was used to send the x and y coordinates to the overlay filter. The commands are defined in the file positions.cmd, or could also be defined in the command line.
- Normally the sendcmd filter is inserted in the filter chain somewhere before the target filter. A problem arises when the target filter has more than one input (for example overlay has two inputs). This doesn't work, because sendcmd accepts only one input. In this case sendcmd must be inserted somewhere earlier in the filter chain, where only one input exists.
- It's important that sendcmd is inserted at a position in the filter chain that has sufficient duration. For example, if the overlaid video is shorter than the main video, and if sendcmd is inserted in the input of the shorter video, that would give unexpected behaviour, because when the shorter video has ended, sendcmd will get the wrong time (which stays then constant), and will send wrong commands to the other filters based on the wrong time. Always insert sendcmd at the longest input.
- It's also possible to have more than one sendcmd in the filter chain, for example at both inputs.
- It's also possible to insert sendcmd after the target filter, for example at the end of the filter chain. The drawback of this method is that the changes become effective with one frame delay.
- All arguments of the sendcmd target filter must be initialized with valid values, even if these values are never used because sendcmd does always overwrite them.
- It's also possible to evaluate an expression in sendcmd and send the result to the target filter. To enable expression evaluation the [expr] flag must be used instead of the default [enter] flag.
A few examples for `sendcmd` and single / double quotes:

```
ffmpeg -i in1.mp4 -i in2.mp4 -lavfi
"[0]sendcmd='3.0 streamselect map 1'"[a];[a][1]streamselect=inputs=2:map=0" out.mp4
```

sendcmd at the beginning of the filter chain, double quotes for whole filter chain. Works fine, but would give unexpected results if the first input is shorter than the second input!

```
ffmpeg -i in1.mp4 -i in2.mp4 -lavfi
"[0][1]sendcmd='3.0 streamselect map 1',streamselect=inputs=2:map=0" out.mp4
```

This doesn't work because `sendcmd` accepts only one input

```
ffmpeg -i in1.mp4 -i in2.mp4 -lavfi
sendcmd='3.0 streamselect map 1',streamselect=inputs=2:map=0 out.mp4
```

This is the example from the `streamselect` documentation, doesn't work under Windows.

```
ffmpeg -i in1.mp4 -i in2.mp4 -lavfi
sendcmd="3.0 streamselect map 1",streamselect=inputs=2:map=0 out.mp4
```

Single quotes replaced by double quotes, works under Windows.

```
ffmpeg -i in1.mp4 -i in2.mp4 -lavfi
"sendcmd='3.0 streamselect map 1',streamselect=inputs=2:map=0" out.mp4
```

Double quotes added for the whole filter chain, single quotes for `sendcmd` argument, works under Windows.

```
ffmpeg -i in1.mp4 -i in2.mp4 -lavfi
"[0][1]streamselect@my=inputs=2:map=0,sendcmd='3.0 streamselect@my map 1'" out.mp4
```

[0][1] added, `sendcmd` at the end of the filter chain, commands become effective with one frame delay. Double quotes for filter chain, single quotes for `sendcmd` argument

```
ffmpeg -i in1.mp4 -i in2.mp4 -lavfi
"[0][1]streamselect@my=inputs=2:map=0,sendcmd="3.0 streamselect@my map 1" out.mp4
```

[0][1] added, `sendcmd` at the end of the filter chain, commands become effective with one frame delay. No quotes for filter chain, double quotes for `sendcmd` argument

Note about double quotes around the filter chain:
In Windows it's not required to put the whole filter chain in double quotes, but it seems these double quotes are required on a Mac. Not tested myself.
Remap Video-in-Video with perspective filter

Suppose you have a video in which a TV or computer screen is visible, and in postprocessing you want another video to be shown on that screen. Or you have a video in which a beamer projects an image on a wall, which is almost impossible to capture flicker-free in a video. It's better to overlay the projected image in postprocessing.

The perspective filter can be used to remap a rectangular video into the distorted screen (which is an irregular quadrangle).

The coordinates of the corners of the screen are \( x_0, y_0 \) (top left), \( x_1, y_1 \) (top right), \( x_2, y_2 \) (bottom left) and \( x_3, y_3 \) (bottom right) and must me measured in the original video.

```bash
set "FF=c:\ffmpeg\ffmpeg"     :: Path to FFmpeg
set "X0=500"                  :: Top left corner
set "Y0=250"
set "X1=1250"                 :: Top right corner
set "Y1=150"
set "X2=400"                  :: Bottom left corner
set "Y2=750"
set "X3=1150"                 :: Bottom right corner
set "Y3=850"
rem  Make a color test video
%FF% -f lavfi -i testsrc2=s=hd1080 -t 5 -y video1.mp4
rem  Make a black and white test video
%FF% -f lavfi -i testsrc2=s=hd1080 -vf eq=saturation=0 -t 5 -y video2.mp4
rem  Embed the black and white video into the color video
%FF% -i video1.mp4 -i video2.mp4 -lavfi "[1]format=argb,pad=w=iw+2:h=ih+2:x=1:y=1:color=black@0.0,perspective=x0=\%X0%:y0=\%Y0%:x1=\%X1%:y1=\%Y1%:x2=\%X2%:y2=\%Y2%:x3=\%X3%:y3=\%Y3%:sense=1[2];[0][2]overlay" -q:v 2 -y out.mp4
```

95
Before I discovered the perspective filter, I thought that I had to use the remap filter for this purpose, and I figured out the formulas myself. Here they are:

The coordinates of the point to be remapped are \(x, y\).

We draw a vertical line through point \(x, y\) and calculate the intersection points with the upper and lower edge of the quadrangle:

\[
a = \frac{(x - x_0)}{(x_1 - x_0)}
\]

The parameter \(a\) describes where the line intersects the upper edge. For \(a=0\) it's at the top left corner, for \(a=1\) it's at the top right corner. For \(0 < a < 1\) the intersection point is somewhere between these two corners. But there are also cases possible \(a < 0\) or \(a > 1\) where the intersection point is outside the finite line segment.

The intersection point is \((x_4, y_4)\)

\[
x_4 = x
\]

\[
y_4 = y_0 + a \times (y_1 - y_0)
\]

We do the same thing for the lower edge:

\[
b = \frac{(x - x_2)}{(x_3 - x_2)}
\]

\[
x_5 = x
\]

\[
y_5 = y_2 + b \times (y_3 - y_2)
\]

Parameter \(c\) describes where the point \(x, y\) lies on the line segment between points 4 and 5:

\[
c = \frac{(y - y_4)}{(y_5 - y_4)}
\]

Now we remap these points into a unit quadrat with the top left corner at \(0, 0\):

Point 4 is at coordinates \((a, 0)\) and point 5 is at coordinates \((b, 1)\)

Point \(x, y\) is remapped to coordinates

\[
x_{\text{map}} = (a + c \times (b - a))
\]

\[
y_{\text{map}} = c
\]
2.64 Image warping with displace filter

```bash
set "FF=c:\ffmpeg\ffmpeg" :: Path to FFmpeg
set "W=751" :: Width of image
set "H=853" :: Height of image
set "CX=347" :: X center of distortion
set "CY=451" :: Y center of distortion
set "A=15" :: Maximum amplitude of displacement, positive displaces outwards and negative inwards, allowed range is [0..127], best values are below 20
set "D=30" :: Radius from center of distortion, where the maximum displacement occurs

rem Create the displace_x file
%FF% -f lavfi -i nullsrc=size=%W%x%H% -vf format=pix_fmts=gray8,geq='st(0,2*%A%*%D%/((%CX%-X)^2+(%CY%-Y)^2+%D%*%D%));128-ld(0)*(X-%CX%)' -frames 1 -y displace_x.pgm

rem Create the displace_y file
%FF% -f lavfi -i nullsrc=size=%W%x%H% -vf format=pix_fmts=gray8,geq='st(0,2*%A%*%D%/((%CX%-X)^2+(%CY%-Y)^2+%D%*%D%));128-ld(0)*(Y-%CY%)' -frames 1 -y displace_y.pgm

rem Apply the displace filter to the image
%FF% -i me.jpg -i displace_x.pgm -i displace_y.pgm -lavfi format=pix_fmts=rgb24,displace -frames 1 -y bigger_nose.jpg

set "A=-15" :: Now let's try the other sign

rem Create the displace_x file
%FF% -f lavfi -i nullsrc=size=%W%x%H% -vf format=pix_fmts=gray8,geq='st(0,2*%A%*%D%/((%CX%-X)^2+(%CY%-Y)^2+%D%*%D%));128-ld(0)*(X-%CX%)' -frames 1 -y displace_x.pgm

rem Create the displace_y file
%FF% -f lavfi -i nullsrc=size=%W%x%H% -vf format=pix_fmts=gray8,geq='st(0,2*%A%*%D%/((%CX%-X)^2+(%CY%-Y)^2+%D%*%D%));128-ld(0)*(Y-%CY%)' -frames 1 -y displace_y.pgm

rem Apply the displace filter to the image
%FF% -i me.jpg -i displace_x.pgm -i displace_y.pgm -lavfi format=pix_fmts=rgb24,displace -frames 1 -y smaller_nose.jpg

pause
```
Here is the input image and the two output images:

It might be dangerous to use this kind of processing for images of women without prior asking them for permission :-)

The “displace” filter expects mapping files with relative values in the range [0..255], where 128 is the neutral value for no displacement. Larger displacements than 127 pixels aren’t possible.

I recommend to set the format to rgb24 before using the displace filter.
If the output is a video, remove "-frames 1" in the last command line.
Here are the input and output images:
Mathematics for this distortion:

\[ d = \frac{2ADr}{r^2 + D^2} \]

\[ r = \sqrt{(x-cx)^2 + (y-cy)^2} \]

\[ \frac{d}{r} = \frac{2AD}{(x-cx)^2 + (y-cy)^2 + D^2} \]

\[ dx = (x-cx) \frac{d}{r} \]

\[ dy = (y-cy) \frac{d}{r} \]

with \( d \) = displacement distance

\( A \) = maximum amplitude of displacement

\( r \) = distance from pixel \( x,y \) to center of distortion \( cx, cy \)

\( D \) = distance where the largest displacement occurs

\( cx, cy \) = coordinates of center of the distortion

\( dx, dy \) = displacement values
### 2.65 Noise reduction

FFmpeg has several filters for video noise reduction:

<table>
<thead>
<tr>
<th>Filter</th>
<th>Description</th>
<th>Notes and Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>atadenoise</td>
<td>Apply an Adaptive Temporal Averaging Denoiser to the video input</td>
<td>very fast, temporal only with no motion compensation; LGPL</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Example: atadenoise=0a=0.2:1a=0.2:2a=0.2:0b=0.3:1b=0.3:2b=0.3</td>
</tr>
<tr>
<td>bm3d</td>
<td>Denoise frames using Block-Matching 3D algorithm</td>
<td>very very slow, currently implemented as spatial only, algorithm considered as one of the state of art denoisers; LGPL</td>
</tr>
<tr>
<td>dctdnoiz</td>
<td>Denoise frames using 2D DCT (frequency domain filtering)</td>
<td>very very slow: spatial only, blurs too much; LGPL</td>
</tr>
<tr>
<td>fftdenoiz</td>
<td>Denoise frames using 3D FFT (frequency domain filtering)</td>
<td>slow, spatial and limited temporal, using Fast Fourier Transform, may have introduce ringing with bad settings; LGPL</td>
</tr>
<tr>
<td>hqdn3d</td>
<td>This is a high precision/quality 3d denoise filter. It aims to reduce image noise, producing smooth images and making still images really still. It should enhance compressibility.</td>
<td>fast, both spatial and temporal, does basically lowpass by destroying high frequencies, blurs with extreme settings; GPL</td>
</tr>
<tr>
<td>nlmeans</td>
<td>Denoise frames using Non-Local means algorithm</td>
<td>very slow, currently implemented as spatial only, algorithm considered as one of the state of art denoisers; LGPL</td>
</tr>
<tr>
<td>owdenoise</td>
<td>Apply Overcomplete Wavelet denoiser</td>
<td>very very very slow, spatial only, wavelet; GPL</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Example: owdenoise=ls=25</td>
</tr>
<tr>
<td>removegrain</td>
<td>Spatial denoiser for progressive video</td>
<td>fast, spatial only, limited usecase</td>
</tr>
<tr>
<td>vagueenoiser</td>
<td>Apply a wavelet based denoiser</td>
<td>slow, spatial only, pretty good, wavelet; LGPL</td>
</tr>
<tr>
<td>tmix</td>
<td>Noise reduction by averaging up to 128 successive frames. Not suitable for moving objects, of course</td>
<td>Example: tmix=frames=20</td>
</tr>
</tbody>
</table>

Special thanks to Paul B Mahol who posted the notes in the FFmpeg-user list on October 27, 2019.
2.66 Time delay within a filter chain

This is an example for a time delay within a filter chain:

```
ffmpeg -f lavfi -i testsrc=duration=10:size=vga -filter_complex split[a][b];[a]setpts=PTS-5/TB[c];[b]
[c]hstack=shortest=1 -y out.mp4
```

Hint: Subtracting a constant from PTS works fine. However if you try to add a constant to PTS (e.g. setpts=PTS+5/TB), this may lead to the problem that the true length of the output video isn't equal to the length in the metadata.

In this example the same thing is done with tpad and trim filters:

```
ffmpeg -f lavfi -i testsrc=duration=10:size=vga -filter_complex split[a][b];[a]tpad=start_duration=5[c];[b]
[c]hstack=shortest=1,trim=start=5,setpts=PTS-5/TB -y out.mp4
```

2.67 -filter_complex_script

The complex filtergraph can be loaded from an external script file.

Line feeds and empty lines are allowed in the script file. This makes the script much more readable.

The drawback is that you can't use variables as in a batch file.
2.68 Chroma subsampling, pixel format of images or videos

When you make a video from many JPG images, all images must have the same pixel format. But sometimes they are different. For example I has many images that came from the camera with 4:2:2 pixel format. But I had to edit one of the images with IrfanView, it then it was saved with pixel format 4:2:0. This example changes the pixel format of an image from 4:2:0 to 4:2:2

```
ffmpeg -i IMG_044x.jpg -pix_fmt yuvj422p -q 0 IMG_044.jpg
```

Set the pixel format of a video to 4:4:4 and scale the video to 1920x1080

```
c:\ffmpeg\ffmpeg -i input.MOV -pix_fmt yuv444p -s 1920x1080 out.mov
```

In a filter chain the format can be set as follows:

```
c:\ffmpeg\ffmpeg -i input.MOV -lavfi "format=pix_fmts=rgb24" -y out.mp4
```

Which is the same as:

```
c:\ffmpeg\ffmpeg -i input.MOV -lavfi "format=rgb24" -y out.mp4
```

All available pixel formats can be shown with this command:

```
ffmpeg -pix_fmts
```
<table>
<thead>
<tr>
<th>Chroma subsampling</th>
<th>8-bit format</th>
<th>10-bit format</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>4:4:4</td>
<td>yuv444p</td>
<td>yuv444p10le</td>
<td>Each of the three Y'CbCr components have the same sample rate, thus there is no chroma subsampling. This scheme is sometimes used in high-end film scanners and cinematic post production. Note that &quot;4:4:4&quot; may instead be referring to R'G'B' color space, which implicitly also does not have any chroma subsampling.</td>
</tr>
<tr>
<td>4:2:2</td>
<td>yuv422p</td>
<td>yuv422p10le</td>
<td>The two chroma components are sampled at half the sample rate of luma: the horizontal chroma resolution is halved. This reduces the bandwidth of an uncompressed video signal by one-third with little to no visual difference.</td>
</tr>
<tr>
<td>4:2:0</td>
<td>yuv420p</td>
<td>yuv420p10le</td>
<td>In 4:2:0, the horizontal sampling is doubled compared to 4:1:1, but as the Cb and Cr channels are only sampled on each alternate line in this scheme, the vertical resolution is halved. The data rate is thus the same. Cb and Cr are each subsampled at a factor of 2 both horizontally and vertically.</td>
</tr>
</tbody>
</table>


RGB and gray pixel formats (this is only a subset of the available formats):

<table>
<thead>
<tr>
<th>Format</th>
<th>NB_COMPONENTS</th>
<th>BITS_PER_PIXEL</th>
</tr>
</thead>
<tbody>
<tr>
<td>rgb24, bgr24</td>
<td>3</td>
<td>24</td>
</tr>
<tr>
<td>gray</td>
<td>1</td>
<td>8</td>
</tr>
<tr>
<td>argb, rgba, abgr, bgra</td>
<td>4</td>
<td>32</td>
</tr>
<tr>
<td>gray16be, gray16le</td>
<td>1</td>
<td>16</td>
</tr>
<tr>
<td>rgb48be, rgb48le, bgr48be, bgr48le</td>
<td>3</td>
<td>48</td>
</tr>
<tr>
<td>gray14be, gray14le</td>
<td>1</td>
<td>14</td>
</tr>
</tbody>
</table>
2.69 Video Codecs

| `-c:v` mpeg4 | This is the older MP4 codec. Search for "libxvid" in the documentation. |
| `-c:v` libx264 | Newer codec with better compression than mpeg4, but it's possible that the videos don't play on older computers. |
| `-c:v` prores_ks | Apple ProRes encoder, example: ffmpeg -i input.MOV -vcodec prores_ks -pix_fmt yuva444p10le -profile:v 4444 -bits_per_mb 8000 -s 1920x1080 out.mov |

The constant rate factor (CRF) can be set with the `-crf` parameter. Use this mode if you want to keep the best quality and don't care about the file size. The CRF range 0–51 for 8-bit x264 and 0-63 for 10-bit. 0 is lossless, 23 is the default, and 51 is worst quality possible.

Use a preset with the `-preset` parameter. Possible options are ultrafast, superfast, veryfast, faster, fast, medium (this is the default), slow, slower and veryslow. A preset is a collection of options that will provide a certain encoding speed to compression ratio. A slower preset will provide better compression. Use the slowest preset that you have patience for.

The `-tune` parameter can be set to these options:

| film | use for high quality movie content; lowers deblocking |
| animation | good for cartoons; uses higher deblocking and more reference frames |
| grain | preserves the grain structure in old, grainy film material |
| stillimage | good for slideshow-like content |
| fastdecode | allows faster decoding by disabling certain filters |
| zerolatency | good for fast encoding and low-latency streaming |

List all possible internal presets and tunes:

```
ffmpeg -hide_banner -f lavfi -i nullsrc -c:v libx264 -preset help -f mp4 -
```
2.70 Metadata

Global metadata can be saved in a text file as follows:

```
ffmpeg -i input.mp4 -f ffmetadata metadata.txt
```

If you also need the metadata from the video and audio streams (which may contain more informations), use this:

```
ffmpeg -i input.mp4 -c copy -map_metadata 0 -map_metadata:s:v 0:s:v -map_metadata:s:a 0:s:a -f ffmetadata metadata.txt
```

The metadata can be re-inserted into a file as follows:

```
ffmpeg -i input.mp4 -i metadata.txt -map_metadata 1 -codec copy output.mp4
```

Write metadata "title" to mp4 video without re-encoding:

```
ffmpeg -i input.mp4 -metadata title="This is the Title" -acodec copy -codec copy -copyts output.mp4
```

Unfortunately FFmpeg can’t insert the metadata that is required for a spherical 360° video.

2.71 Video filters "copy" and "null"

These filters are only for testing, for example when you want to disable part of a filter chain.

The "null" filter does really nothing, the output is the same as the input.

The "copy" filter copies the old frame and deletes the old one. The output is the same as with the "null" filter.
2.72 Re-numbering images

Cameras do normally create images with 4-digit numbers. When the counter (in Canon cameras) overflows, the number changes from 9999 to 0001. That means the number 0000 is missing and the numbers aren’t continuously increasing, as it’s expected by FFmpeg. This problem can be solved with this sequence of console commands:

```plaintext
ren IMG_1*.* IMG_2*.*
ren IMG_0*.* IMG_1*.*
ren IMG_9*.* IMG_0*.*
```

The first two commands add 1000 to those numbers that begin with 0 or 1. The last command subtracts 9000 from those numbers that begin with 9.

Now the images are in increasing order, but one image is still missing between images 0999 and 1001. This isn’t a problem for FFmpeg because it does automatically search the next image. That means small gaps are allowed in the numbering. The maximum gap size is 5 by default. This can be changed with the parameter -start_number_range.
### 2.73 Filenames for images

Image filenames can contain variables. Please note that in a Windows batch file the % character must be replaced by two % characters.

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>%nd</td>
<td>A sequential n-digit number</td>
<td>The start number can be specified with the -start_number option. The default value is 1.</td>
</tr>
<tr>
<td>%0nd</td>
<td>A sequential number that is 0-padded to n digits</td>
<td></td>
</tr>
<tr>
<td>%Y</td>
<td>Year</td>
<td>Only available if the &quot;-strftime 1&quot; option is used.</td>
</tr>
<tr>
<td>%m</td>
<td>Month</td>
<td>Example for embedding date and time in the filename:</td>
</tr>
<tr>
<td>%d</td>
<td>Day</td>
<td>Screenshot-%Y-%m-%d-%H-%M-%S.jpg</td>
</tr>
<tr>
<td>%H</td>
<td>Hours</td>
<td></td>
</tr>
<tr>
<td>%M</td>
<td>Minutes</td>
<td></td>
</tr>
<tr>
<td>%S</td>
<td>Seconds</td>
<td></td>
</tr>
<tr>
<td>%d</td>
<td>PTS of the frame (Presentation time stamp)</td>
<td>Only available if the &quot;-frame_pts 1&quot; option is used.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Example for embedding the PTS in the filename:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Screenshot-%d.jpg</td>
</tr>
</tbody>
</table>
2.74 Make many JPG test images

```
ffmpeg -f lavfi -i testsrc2=size=5472x3648:duration=12:rate=10 test%3d.jpg
```

2.75 Make a chessboard video

```
c:\ffmpeg\ffmpeg -f lavfi -i color=black:s=vga -vf geq=lum='255*mod(floor(X/40)+floor(Y/40),2):cr=128' -t 5 -y out.mp4
```

2.76 Make a test video with audio

```
rem Make a 6 seconds video with 1kHz tone

c:\ffmpeg\ffmpeg -f lavfi -i testsrc2=size=vga -f lavfi -i sine=1000 -t 6 -y video.mp4
```

2.77 Find an object in a video and hide it

The find_rect function can find a rectangular object in an image or video, and cover_rect can then replace it by another image or by interpolating the neighbor pixels.

```
c:\ffmpeg\ffmpeg -i test1.png -vf find_rect=test2.pgm,cover_rect out.png
```

pause

If anybody knows how to write the find_rect result (x,y coordinates) to a file, please let me know.
# Image formats

FFmpeg can save images in these formats (the list isn't complete):

<table>
<thead>
<tr>
<th>Image Format</th>
<th>compressed?</th>
<th>lossless?</th>
<th>16-Bit?</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>JPG</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>recommended for 8-bit images if small file size is required</td>
</tr>
<tr>
<td>BMP</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>very big file size</td>
</tr>
<tr>
<td>GIF</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
<td>this is obsolete, use PNG instead</td>
</tr>
<tr>
<td>TGA</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
<td>this is obsolete, use PNG instead</td>
</tr>
<tr>
<td>PNG</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>recommended for lossless saving of 8-bit or 16-bit images</td>
</tr>
<tr>
<td>PGM</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
<td>&quot;Portable Graymap&quot;, these files are required for the remap filter. FFmpeg can read binary PGM (P5) files and ASCII PGM (P2) files, but for output only binary PGM (P5) is possible. PGM files contain values in the range [0..65535]. Negative values aren't possible, but FFmpeg gives no warning if a negative number is found in a P2 file.</td>
</tr>
<tr>
<td>PPM</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
<td>&quot;PortablePixmap&quot;</td>
</tr>
<tr>
<td>PAM</td>
<td>no</td>
<td>?</td>
<td>?</td>
<td>&quot;Portable Arbitrary Map&quot;</td>
</tr>
<tr>
<td>PGMYUV</td>
<td>no</td>
<td>?</td>
<td>?</td>
<td>This is a FFmpeg variant of the binary PGM format</td>
</tr>
<tr>
<td>TIFF</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
<td></td>
</tr>
<tr>
<td>SGI</td>
<td>?</td>
<td>?</td>
<td>?</td>
<td>?</td>
</tr>
</tbody>
</table>

Show all supported pixel formats of the PNG encoder:

```
c:\ffmpeg\ffmpeg -h encoder=png
```
## 2.79 Video sizes

This list contains only the most important video sizes. The full list is in the FFmpeg documentation.

<table>
<thead>
<tr>
<th>Size</th>
<th>Aspect ratio</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>ntsc</td>
<td>720x480</td>
<td>3:2</td>
</tr>
<tr>
<td>pal</td>
<td>720x576</td>
<td>5:4</td>
</tr>
<tr>
<td>vga</td>
<td>640x480</td>
<td>4:3</td>
</tr>
<tr>
<td>svga</td>
<td>800x600</td>
<td>4:3</td>
</tr>
<tr>
<td>xga</td>
<td>1024x768</td>
<td>4:3</td>
</tr>
<tr>
<td>uxga</td>
<td>1600x1200</td>
<td>4:3</td>
</tr>
<tr>
<td>wuxga</td>
<td>1920x1200</td>
<td>8:5 Beamer in the planetarium in the St. Andreasberg observatory</td>
</tr>
<tr>
<td>hd720</td>
<td>1280x720</td>
<td>16:9</td>
</tr>
<tr>
<td>hd1080</td>
<td>1920x1080</td>
<td>16:9</td>
</tr>
<tr>
<td>2k</td>
<td>2048x1080</td>
<td>256:135 = 17.066:9</td>
</tr>
<tr>
<td>2kdci</td>
<td>2048x1080</td>
<td>256:135 = 17.066:9</td>
</tr>
<tr>
<td>2kflat</td>
<td>1998x1080</td>
<td>37:20 = 16.65:9 Cinema advertising</td>
</tr>
<tr>
<td>4k</td>
<td>4096x2160</td>
<td>256:135 = 17.066:9</td>
</tr>
<tr>
<td>4kdci</td>
<td>4096x2160</td>
<td>256:135 = 17.066:9</td>
</tr>
<tr>
<td>4kflat</td>
<td>3996x2160</td>
<td>37:20 = 16.65:9</td>
</tr>
<tr>
<td>uhd2160</td>
<td>3840x2160</td>
<td>16:9</td>
</tr>
</tbody>
</table>
2.80  Editing videos from PanoView XDV360 fulldome camera

rem  Editing videos from the chinese PanoView XDV360 fulldome camera

set "FF=c:\ffmpeg\ffmpeg"   :: Path to FFmpeg
set "SIZE=1200"             :: Video size (square)
set "QU=2"                  :: MP4 quality level, 0 is best quality, 2 is normal, 9 is strong compression
set "FPS=30"                :: Output Framerate
set "IN=PanoView.mp4"       :: Input video
set "START=0.5"             :: Start time in seconds in the input video
set "LEN=4"                 :: Length in seconds in the input video
set "FADE=1"                :: Fade-In and Fade-Out duration in seconds
set "FO=3.5"                :: Start time of Fade-Out (Start + Length - Fade)
                               :: 185° field of view: 2168:2168:144:144
                               :: 190° field of view: 2224:2224:116:116
                               :: 195° field of view: 2280:2280:88:88
                               :: 200° field of view: 2336:2336:60:60
set "CON=1.0"               :: Contrast in range [-1000 ... 1000], normal is 1.0
set "BR=0.0"                :: Brightness in range [-1.0 ... 1.0], normal is 0.0
set "SA=1.0"                :: Saturation in range [0.0 ... 3.0], normal is 1.0
set "GA=1.0"                :: Gamma in range [0.1 ... 10.0], normal is 1.0
set "HUE=25"                :: Color correction, negative towards red, positive towards blue
set "SOUND=birds.mp3"       :: Audio input file
set "VOL=2.0"               :: Audio volume, normal is 1.0
set "OUT=out.mp4"           :: Output filename

%FF% -i %IN% %SOUND% -ss %START% -t %LEN% ^
-filter_complex crop=cr:scale=3648:3648,overlay,hue=h=HUE%,eq=contrast=CON%,brightness=BR%,saturation=SA% ^
-gamma=GA%,fade=in:st=%START%:d=%FADE%,fade=out:st=%FO%:d=%FADE% ^
-ac 2 -af volume=VOL%,aresample=44100,afade=in:st=%START%:d=%FADE%,afade=out:st=%FO%:d=%FADE% ^
-s %SIZE%x%SIZE% -c:v mpeg4 -q:v QU% -y OUT% ^
pause

The image "Circle_3648.png" has the size 3648x3648 and contains a circular mask. The color of the circle content is declared transparent, and the border is black. "-ac 2" expands the audio tracks to stereo, and "-af aresample=44100" increases the audio sampling rate to 44100.
2.81 Editing videos from the Kodak SP360_4K camera

The Kodak SP360_4K camera is better than the PanoView XDV360 because it has a slightly higher resolution and the videos are not compressed as much.

rem Manipulate a video from KODAK SP360_4K camera (size 2880x2880)

set "FF=c:\ffmpeg\ffmpeg" :: Path to FFmpeg
set "SIZE=1200" :: Video size (square)
set "QU=2" :: MP4 quality level, 0 is best quality, 2 is normal, 9 is strong compression
set "FPS=30" :: Output Framerate
set "INPUT=102_0001.MP4" :: Input video
set "START=5" :: Start time in seconds in the input video
set "LENGTH=28" :: Length in seconds in the input video
set "FADE=1" :: Fade-In and Fade-Out duration in seconds
set "FADEOUT=32" :: Start time of Fade-Out (Start + Length - Fade)
:: 185° field of view: 2528:2528:174:174
:: 190° field of view: 2592:2592:142:142
:: 195° field of view: 2664:2664:106:106
:: 200° field of view: 2728:2728:74:74
set "CONTRAST=1.0" :: Contrast in range [-1000 ... 1000], normal is 1.0
set "BRIGHT=0.0" :: Brightness in range [-1.0 ... 1.0], normal is 0.0
set "SATUR=1.0" :: Saturation in range [0.0 ... 3.0], normal is 1.0
set "GAMMA=1.0" :: Gamma in range [0.1 ... 10.0], normal is 1.0
set "HUE=0" :: Color correction, negative towards red, positive towards blue
set "SOUND=vogelstimmen.mp3" :: Audio file
set "VOL=0.4" :: Audio volume, normal is 1.0
set "OUTPUT=scene30.mp4" :: Output filename

%FF% -i %INPUT% -i Circle_3648.png -ss %START% -t %LENGTH% ^
-filter_complex crop=%CROP%,scale=3648:3648,overlay,hue=h=%HUE%,eq=contrast=%CONTRAST%:brightness=%BRIGHT%:^
saturation=%SATUR%:gamma=%GAMMA%,fade=in:st=%START%:d=%FADE%,fade=out:st=%FADEOUT%:d=%FADE% ^
-af volume=%VOL%,aresample=44100,afade=in:st=%START%:d=%FADE%,afade=out:st=%FO%:d=%FADE% ^
-s %SIZE%x%SIZE% -c:v mpeg4 -q:v %QU% -y %OUT%

pause
2.82 Postprocessing of real time videos of the night sky

set "FF=c://ffmpeg/ffmpeg" :: Path to FFmpeg
set "DARK=Dark.mov" :: Dark video
set "IN=sternschnuppe175.mov" :: Input video
set "AMP=0.06" :: Gain factor
set "D=0.3" :: Parameter for atadenoise filter
set "TMIX=25" :: Tmix number, more than 25 isn't required
set "BR=0.05" :: Small brightness increase after dark subtraction
set "NR=0.8" :: Noise reduction in maximum function
set "OUT=175.mp4" :: Output video

rem Create a darkframe by averaging many frames from the dark video

%FF% -i %DARK% -vf "crop=2824:ih,pad=iw:2824:-1:-1,scale=1200:1200,curves=all='0/0 %AMP%/1 1/1',
eq=saturation=0,tmix=frames=128" -frames 1 -y dark.png
rem create a video with dark subtraction, strong contrast enhancement and denoise filter

%FF% -i %IN% -i dark.png -filter_complex "crop=2824:ih,pad=iw:2824:-1:-1,scale=1200:1200,curves=all='0/0 %AMP%/1 1/1',
eq=saturation=0[0];[0][1]blend=subtract,eq=brightness=%BR%,split=2[1][2];[1]atadenoise=0a=%D%:1a=%D%:2a=%D%:0b=%D%:1b=%D%:2b=%D%:[2];[2]tmix=%TMIX%[3];[3][3]blend=all_expr='max(%NR%*A,B)'' -q:v 1 -y %OUT%

Better version:

set "FF=c://ffmpeg/ffmpeg" :: Path to FFmpeg
set "DARKVID=Dark.mov" :: Dark video
set "MAKE DARK=no" :: Make a darkframe yes / no
set "IN=sternschnuppe256.mov" :: Input video
set "OUT=meteor256bw.mp4" :: Output video

pause
set "BP_R=0.02"                 :: Black point red, positive value makes background darker
set "BP_G=0.00"                 :: Black point green, positive value makes background darker
set "BP_B=0.02"                 :: Black point blue, positive value makes background darker
::
set "WP=0.20"                   :: White point
::  Make sure that all pixel values in the dark frame
::  are below this white point
::
set "SAT=0.0"                   :: Saturation, normal = 1.0, set to 0 for monochrome
set "GAM=1.0"                   :: Gamma, normal = 1.0
::
set "D=0.3"                     :: Parameter for Atadenoise filter
set "TMIX=25"                   :: Tmix count, more than 25 isn't required
::
set "CUT=12"                    :: Duration that is cut off from the beginning
::  In the input video the duration from the beginning
::  to the first appearance of the meteor must be greater
::  than the duration of the output video
::
set "AUDIO=no"                  :: Copy audio yes / no
::
set AUD=""                     :: Copy audio yes / no
if %AUDIO%==no (set AUD=-an)
if %MAKEDARK%==no goto VIDEO

%FF% -i %DARKVID% -vf "crop=2824:ih,pad=iw:2824:-1:-1, scale=1200:1200, tmix=128, format=rgb48" -frames 1 -y dark.png

:VIDEO
%FF% -i %IN% -i dark.png -filter_complex "crop=2824:ih,pad=iw:2824:-1:-1, scale=1200:1200, format=rgb48[a];[a]
:a=gamma=GAM%, split[b][c];[b]setpts=PTS-%CUT%/TB, atadenoise=0a=%D%:1a=%D%:2a=%D%:0b=%D%:1b=%D%:2b=%D%[d];[c]tmix=TMIX% [e];[d][e]blend=lighten:shortest=1" -q:v 1 %AUD% -y %OUT%

pause
This batch file does the same postprocessing with a loop over all CUT*.MOV files in the current folder:

```batch
set "FF=c://ffmpeg/ffmpeg" :: Path to FFmpeg
set "BP_R=0.02" :: Black point red, positive value makes background darker
set "BP_G=0.00" :: Black point green, positive value makes background darker
set "BP_B=0.02" :: Black point blue, positive value makes background darker
set "WP=0.2" :: White point
          :: Make sure that all pixel values in the dark frame
          :: are below this white point
set "SAT=1.0" :: Saturation, normal = 1.0, set to 0 for monochrome
set "GAM=1.0" :: Gamma, normal = 1.0
set "D=0.3" :: Parameter for Atadenoise filter,
             :: 0.3 = strongest noise reduction
set "TMIX=25" :: Tmix count, more than 25 isn't required
set "CUT=15" :: Duration that is cut off from the beginning
                :: In the input video the duration from the beginning
                :: to the first appearance of the meteor must be greater
                :: than the duration of the output video
set "AUDIO=no" :: Copy audio yes / no

set AUD="
if %AUDIO%==no (set AUD=-an)

for %%f in (CUT*.MOV) do call :for_body %%f
goto :the_end

:for_body
%FF% -i %1 -i dark.png -filter_complex "crop=2824:ih,pad=iw:2824:-1:-1,scale=1200:1200,format=rgb48[a];[a]
[1]blend=subtract,colorlevels=rimin=%BP_R%:gimin=%BP_G%:bimin=%BP_B%:rimax=%WP%:gimax=%WP%:bimax=%WP%,eq=saturation=%SAT
%:gamma=%GAM%,split[b][c];[b]setpts=PTS-%CUT%/TB,atadenoise=0a=%D%:1a=%D%:2a=%D%:0b=%D%:1b=%D%:2b=%D%[d];[c]tmix=%TMIX%
[e];[d][e]blend=lighten:shortest=1" -q:v 1 -c:v mpeg4 %AUD% -y %~n1.mp4
exit /b
:the_end
pause
```

117
2.83 **Workflow for night sky videos with GH5S**

Workflow for videos from the starry sky with Panasonic GH5S and Nippon Kogaku 8mm Fisheye:

Step 1, apply gradation curve and then extract an image and insert the CLUT:

```
set "FF=c:\ffmpeg\ffmpeg"  :: Path to FFmpeg
set "IN=P1000128.mov"         :: Input video
set "T=1"                     :: Time where image is extracted

%FF% -ss %T% -i %IN% -f lavfi -i haldclutsrc=8 -filter_complex "[0]format=pix_fmts=rgb48be[a];[1]format=pix_fmts=rgb48be[b];[b][a]xstack=inputs=2:layout=0_0|w0_0" -frames 1 -y image_with_clut.png
```

Step 2, after editing with GIMP, the CLUT is extracted and applied to the video:

```
set "FF=c://ffmpeg/ffmpeg"    :: Path to FFmpeg
set "IN=P1000128.mov"         :: Input video
set "BR=0.04"                 :: Small brightness adjustment before applying the CLUT
set "OUT=128.mp4"             :: Output video

%FF% -i processed_image_with_clut.png -vf crop=512:512:0:0 -y clut.png

%FF% -i %IN% -i clut.png -filter_complex "[0]format=pix_fmts=rgb48be,crop=2824:ih,pad=iw:2824:-1:-1,eq=brightness=%BR %,scale=1200:1200[a],[a][1]haldclut" -an -y %OUT%

del clut.png
```

pause
2.84 Combine many options and filters

Of course, you can also combine any number of options and filters in a single batch file, as in this example. With this batch file you can cut a temporal part out of a video, change width and height, adjust the frame rate, change the speed (slow motion or time lapse), if necessary crop to square format, if necessary remove the original sound, and change contrast, brightness, saturation and gamma.

REM Video processing

set "FF=c:\ffmpeg\ffmpeg" :: Path to FFmpeg
set "INPUT=PanoView.mp4" :: Input video
set "OUTPUT=out.mp4" :: Output video
set "SIZE=800x800" :: Width and height of output video; the aspect ratio should be the same as in the
:: input video, or square if QUAD=yes was selected
set "RATE=30" :: Output framerate
set "START=1.0" :: Start time in seconds (in input video)
set "LENGTH=3" :: Length in seconds (in input video)
set "SPEED=3.0" :: Speed factor: < 1 timelapse, 1 real time, > 1 slow motion
set "QUAD=no" :: no: keep the aspect ratio as-is
:: yes: crop to square aspect ratio
set "AUDIO=no" :: no: suppress sound
:: yes: keep the original sound (with unchanged speed)
set "CONTRAST=1.0" :: Contrast in range [-1000 ... 1000], normal is 1.0
set "BRIGHT=0.0" :: Brightness in range [-1.0 ... 1.0], normal is 0.0
set "SATUR=1.0" :: Saturation in range [0.0 ... 3.0], normal is 1.0
set "GAMMA=1.0" :: Gamma in range [0.1 ... 10.0], normal is 1.0
set "QU=2" :: MP4 Quality, 0 is best Quality, 2 is normal, 9 is strongest compression

set CROP=iw:ih
if %QUAD%==yes (set CROP=ih:ih)

set SOUND=
if %AUDIO%==no (set SOUND=-an)

%FF% -ss %START% -t %LENGTH% -i %INPUT% %SOUND% ^
-vf crop=%CROP%,setpts=%SPEED%*PTS,eq=contrast=%CONTRAST%:brightness=%BRIGHT%:saturation=%SATUR%:gamma=%GAMMA% ^
-s %SIZE% -r %RATE% -q:v %QU% -codec:v mpeg4 %OUTPUT%
pause
2.85  Timelapse example with masking, deflicker, rotation, fading

This batch file creates a time lapse from many images, with masking and deflicker filter, with slow rotation of the image, fade in and fade out at the beginning and end:

```batch
set "FF=c:\ffmpeg\ffmpeg"  :: Path to FFmpeg
set "IN=IMG_%%4d.jpg"      :: Input pictures
set "SN=3551"              :: Number of first picture
set "SIZE=1200"            :: Video size (square)
set "QU=2"                 :: MP4 quality level, 0 is best quality, 2 is normal, 9 is strong compression
set "FPS=30"               :: Output Framerate
set "FADE=3"               :: Fade-In and Fade-Out duration in seconds
set "FADEOUT=11.5"         :: Start time of Fade-Out (Length - Fade)
set "CONTRAST=1.0"         :: Contrast in range [-1000 ... 1000], normal is 1.0
set "BRIGHT=0"             :: Brightness in range [-1.0 ... 1.0], normal is 0.0
set "SATUR=1.2"            :: Saturation in range [0.0 ... 3.0], normal is 1.0
set "GAMMA=1.1"            :: Gamma in range [0.1 ... 10.0], normal is 1.0
set "ROT=0.0+n*0.002"      :: Rotation angle in radians
set "DEF=20"               :: Deflicker frames
set "AUDIO=birds.mp3"      :: Audio file
set "VOL=1.5"              :: Audio volume
set "OUT=out.mp4"          :: Output filename
rem  A is the duration in seconds how long a single image is shown (without crossfade duration), here: 0.2
rem  B is the crossfade duration in seconds, here: 0.2
set "D=2"   :: enter (A+B)/B    D=1: 100% fade   D=2: 50% fade   D=4: 25% fade
set "F=5"   :: enter 1/B
set "E=13"  :: enter FADE * F - D  (longer duration for first and last picture)
set "L=30"  :: Number of pictures

%FF% -start_number %SN% -i %IN% -i Circle_3648.png -i %AUDIO% -shortest ^
-filter_complex crop=ih:ih, scale=3648:3648, eq=contrast=%CONTRAST%:brightness=%BRIGHT%:saturation=%SATUR%:gamma=%GAMMA%,^
overlapy, zoompan=x=%SIZE%:y=%SIZE%:d=%D%+%E%*eq(in,1)+%E%*eq(in,%L%):fps=%FPS%,^
framerate=FPS:interp_start=0:interp_end=255:scene=100,rotate=%ROT%,deflicker=size=%DEF%,^
fade=in:d=%FADE%,interp_start=0:interp_end=255:scene=100,rotate=%ROT%,deflicker=size=%DEF%,^
afade=in:st=0:d=%FADE%,afade=out:st=%FADEOUT%:d=%FADE% -af volume=%VOL%,afade=in:st=0:d=%FADE%,afade=out:st=%FADEOUT%:d=%FADE% -codec:v mpeg4 -q:v %QU% -y %OUT%
pause
```
Slow motion with Panasonic GH5S at 240fps

Set "Rec Format" to "MOV" and "Rec Quality" to one of these:

<table>
<thead>
<tr>
<th>System Frequency</th>
<th>Rec Quality</th>
<th>Available Framerates</th>
</tr>
</thead>
<tbody>
<tr>
<td>59.94Hz (NTSC)</td>
<td>[FHD/8bit/100M/60p]</td>
<td>2  30  56  60  62  64  90  120  150  180  210  240</td>
</tr>
<tr>
<td></td>
<td>[FHD/8bit/100M/30p]</td>
<td>2  15  26  28  30  32  34  45  60  75  90  105  120  135  150  165  180  195  210  225  240</td>
</tr>
<tr>
<td></td>
<td>[FHD/8bit/100M/24p]</td>
<td>2  12  20  22  24  26  28  36  48  60  72  84  96  108  120  132  144  156  168  180  192  204  216  228  240</td>
</tr>
<tr>
<td>50.00Hz (PAL)</td>
<td>[FHD/8bit/100M/50p]</td>
<td>2  25  46  48  50  52  54  75  100  125  150  200  240</td>
</tr>
<tr>
<td></td>
<td>[FHD/8bit/100M/25p]</td>
<td>2  12  21  23  25  27  30  37  50  62  75  87  100  112  125  137  150  175  200  225  240</td>
</tr>
<tr>
<td>24.00Hz (CINEMA)</td>
<td>[FHD/8bit/100M/24p]</td>
<td>2  12  20  22  24  26  28  36  48  60  72  84  96  108  120  132  144  156  168  180  192  204  216  228  240</td>
</tr>
</tbody>
</table>

Set "Variable Frame Rate" to "On" and set the desired framerate.

The video can be played as-is with VLC player and is already a timelapse. The speed factor can be calculated as follows:

\[
\text{Speed\_Factor} = \frac{\text{Framerate}}{\text{Base\_Framerate}}
\]

where Base_Framerate is either 24, 25, 30, 50 or 60 as specified in "Rec Quality".

If you want a higher speed factor, you have to use the setpts filter:

```
set "FF=c:\ffmpeg\ffmpeg" :: Path to FFmpeg
set "IN=P1000128.mov" :: Input video
set "S=5.0" :: Slow motion factor
%
FF% -i %IN% -vf setpts=5*PTS -y out.mp4
pause
```

Example:

The video was taken in [FHD/8bit/100M/50p] mode at 240fps. It has already a 4.8x speed factor if played as-is.

Output from FFprobe: FrameRate = 50p, VFRRatio = 50/240

With the additional factor 5 the total speed factor becomes 24x, with other words one second in reality is shown as 24 seconds in playback. The output framerate can be set with the -r option if required; this doesn't affect the speed factor.
Table for setpts value (for videos taken at 240fps):

<table>
<thead>
<tr>
<th>Base Framerate</th>
<th>Desired Slow Motion Factor</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>2x</td>
</tr>
<tr>
<td>24</td>
<td>0.2</td>
</tr>
<tr>
<td>25</td>
<td>0.20833</td>
</tr>
<tr>
<td>30</td>
<td>0.25</td>
</tr>
<tr>
<td>50</td>
<td>0.41667</td>
</tr>
<tr>
<td>60</td>
<td>0.5</td>
</tr>
<tr>
<td>Output fps</td>
<td>120</td>
</tr>
</tbody>
</table>

Setpts_Value = Desired_Slow_Motion_Factor * Base_Framerate / 240

If the output framerate is too slow, you could use the "minterpolate" filter to calculate intermediate frames with motion interpolation.
2.87 Create a light curve of a star occultation

```
set "FF=c://ffmpeg/ffmpeg" :: Path to FFmpeg

set "IN=P1000479.mov" :: Input video
set "OUT=occultation.mp4" :: Output video

set "BP_R=0.015" :: Black point red, positive value makes background darker
set "BP_G=0.005" :: Black point green, positive value makes background darker
set "BP_B=0.015" :: Black point blue, positive value makes background darker
set "WP=0.26" :: White point

set "S=300" :: Start time
set "T=40" :: Duration

set "X=926" :: X Position of star
set "Y=475" :: Y Position of star
set "B=10" :: Half of the box size for averaging the brightness, in this example the box is 20x20 pixels
set "MIN=60" :: Minimum brightness for Y axis
set "MAX=90" :: Maximum brightness for Y axis

set "FONT=arial.ttf" :: Font for the clock
set "COLOR=white" :: Font color
set "BCOLOR=black" :: Background color
set "SIZE=30" :: Font size
set "POS_X=0" :: X position of clock
set "POS_Y=(h-th)" :: Y position of clock
set "OFFSET=2340" :: Offset time in seconds, added to the timestamp of the first frame

set CLOCK=drawtext='fontfile=%FONT%:text=%{pts\:hms:%OFFSET%}:fontcolor=%COLOR%:boxcolor=%BCOLOR%:box=1:fontsize=%SIZE %:x=%POS_X%:y=%POS_Y%'

rem Create a video with contrast enhancement, with clock, but without light curve and markers:
rem %FF% -ss %S% -i %IN% -vf "colorlevels=rimin=%BP_R%:gimin=%BP_G%:bimin=%BP_B%:rimax=%WP%:gimax=%WP%:bimax=%WP%,%CLOCK %" -pix_fmt yuv420p -t %T% -y %OUT%
rem Extract the first frame, for finding the x,y coordinates of the star:
```
%FF% -ss %S% -i %IN% -vf "colorlevels=rimin=%BP_R%:gimin=%BP_G%:bimin=%BP_B%:rimax=%WP%:gimax=%WP%:bimax=%WP%" -frames 1 -y framel.png

rem Find the coordinates of the star (for example with IrfanView) and set the variables X,Y accordingly.
rem Then create a video with contrast enhancement, clock, light curve and markers at the star.
rem The light curve can be YAVG for average over the box or YMAX for the maximum in the box:

%FF% -ss %S% -i %IN% -lavfi "crop=2*%B%:2*%B%:%X%-%B%:%Y%-%B%,
signalstats,
drawgraph=m3=lavfi.signalstats.YAVG:mode=line:slide=scroll:min=%MIN%:max=%MAX%:size=1920x270:bg=0x000000@0.0[1];
[0]colorlevels=rimin=%BP_R%:gimin=%BP_G%:bimin=%BP_B%:rimax=%WP%:gimax=%WP%:bimax=%WP%,
drawbox=x=%X%+2*%B%:y=%Y%:c=yellow:t=1:w=2*%B%:h=1,
drawbox=x=%X%:y=%Y%+2*%B%:c=yellow:t=1:w=1:h=2*%B%[2];
[2][1]overlay=y=810,%CLOCK%" -pix_fmt yuv420p -t %T% -y %OUT%

pause

Please note that for better readability the command line is shown here with line feeds. These must be removed in the real batch file.

Important note for the drawgraph filter:

The colors of the curves must be specified in the non-standard format 0xAABBGGRR, however the background color must be specified in FFmpeg's normal color format 0xRRGGBBAA, for which many predefined colors are available.
This is the light curve of the star TYC1932-00469-1 which was occulted by the asteroid (87) Sylvia on October 30, 2019. The video was taken with a Canon EF 400mm f/2.8 lens, SpeedBooster 0.64x and Panasonic GH5S camera at 25600 ISO, FHD 25fps and [Ex. Tele Conv.] = 2.1x.
This is a batch file for drawing light curves of two stars simultaneously:

```
set "FF=c://ffmpeg/ffmpeg" :: Path to FFmpeg
::
set "IN=P1000479.mov"      :: Input video
set "OUT=occultation.mp4"  :: Output video
::
set "BP_R=0.015"           :: Black point red, positive value makes background darker
set "BP_G=0.005"           :: Black point green, positive value makes background darker
set "BP_B=0.015"           :: Black point blue, positive value makes background darker
set "WP=0.26"              :: White point
::
set "S=300"                :: Start time
set "T=40"                 :: Duration
::
set "X1=926"               :: X Position of star
set "Y1=475"               :: Y Position of star
set "C1=0xffffffff"        :: Color for star curve, in format 0xAABBGGR
set "X2=1054"              :: X Position of reference star
set "Y2=267"               :: Y Position of reference star
set "C2=0xffff00ff"        :: Color for reference star curve, in format 0xAABBGGR
set "BG=0x00000000"        :: Background color for curves, in format 0xRRGGBBAA
set "B=10"                 :: Half of the box size for averaging the brightness
set "MIN=60"               :: Minimum brightness for Y axis
set "MAX=90"               :: Maximum brightness for Y axis
::
set "FONT=arial.ttf"       :: Font
set "COLOR=white"          :: Font color
set "BCOLOR=black"         :: Background color
set "SIZE=30"              :: Font size
set "POS_X=0"              :: X position of clock
set "POS_Y=(h-th)"         :: Y position of clock
set "OFFSET=2340"          :: Offset time in seconds, added to the timestamp of the first frame
::
set CLOCK=drawtext='fontfile=%FONT%:text=%%{pts\:hms\:%OFFSET%}:fontcolor=%COLOR%:boxcolor=%BCOLOR%:box=1:fontsize=%SIZE %:x=%POS_X%:y=%POS_Y%'
::
rem Extract the first frame:
rem %FF% -ss %S% -i %IN% -vf "colorlevels=rimin=%BP_R%:gimin=%BP_G%:bimin=%BP_B%:rimax=%WP%:gimax=%WP%:bimax=%WP%"
```
rem  Find the coordinates of the star (with IrfanView) and set them to the variables X1,Y1 and X2,Y2.
rem  Then create a video with light curves and markers at the stars. The light curve can be YAVG for average over the
box or YMAX for maximum in the box:

%FF% -ss %S% -i %IN% -lavfi "[0]crop=2*%B%:2*%B%:%X1%-%B%:%Y1%-%B%,signalstats,
drawgraph=m1=lavfi.signalstats.YAVG:mode=line:slide=scroll:min=%MIN%:max=%MAX%:size=1920x270:fg1=%C1%:bg=%BG%[1];
[0]crop=2*%B%:2*%B%:%X2%-%B%:%Y2%-%B%,signalstats,
drawgraph=m1=lavfi.signalstats.YAVG:mode=line:slide=scroll:min=%MIN%:max=%MAX%:size=1920x270:fg1=%C2%:bg=%BG%[2];
[2][1]overlay[3];
[0]colorlevels=rimin=%BP_R%:gimin=%BP_G%:bimin=%BP_B%:rimax=%WP%:gimax=%WP%:bimax=%WP%,
drawbox=x=%X1%+2*%B%:y=%Y1%:c=White:t=1:w=2*%B%:h=1,
drawbox=x=%X1%:y=%Y1%+2*%B%:c=White:t=1:w=1:h=2*%B%,
drawbox=x=%X2%+2*%B%:y=%Y2%:c=Violet:t=1:w=2*%B%:h=1,
drawbox=x=%X2%:y=%Y2%+2*%B%:c=Violet:t=1:w=1:h=2*%B%[4];
[4][3]overlay=y=810,%CLOCK%" -pix_fmt yuv420p -t %T% -y %OUT%
pause

This is a batch file for drawing the light curve and the audio level simultaneously. But it has the problem that the two graphs aren't running exactly
synchronously. I don't know why.

set "FF=c://ffmpeg/ffmpeg" :: Path to FFmpeg
set "IN=P1000479.mov" :: Input video
set "OUT=occultation.mp4" :: Output video
set "BP_R=0.015" :: Black point red, positive value makes background darker
set "BP_G=0.005" :: Black point green, positive value makes background darker
set "BP_B=0.015" :: Black point blue, positive value makes background darker
set "WP=0.26" :: White point
set "S=300" :: Start time
set "T=40" :: Duration
set "X=926" :: X Position of star
set "Y=475" :: Y Position of star
set "C1=0xffffffff" :: Color for star curve, in format 0xAABBGGRR
set "C2=0xffff00ff"        :: Color for audio curve, in format 0xAABBGGRR
set "BG=0x00000000"        :: Background color for curves, in format 0xRRGGBBAA
set "B=10"                 :: Half of the box size for averaging the brightness
set "MAX=90"               :: Maximum brightness for Y axis
set "MIN=70"               :: Minimum brightness for Y axis
set "AMAX=0"               :: Maximum audio level
set "AMIN=-50"             :: Minimum audio level
set "FONT=arial.ttf"       :: Font
set "COLOR=white"          :: Font color
set "BCOLOR=black"         :: Background color
set "SIZE=30"              :: Font size
set "POS_X=0"              :: X position of clock
set "POS_Y=(h-th)"         :: Y position of clock
set "OFFSET=2340"          :: Offset time in seconds, added to the timestamp of the first frame

set CLOCK=drawtext='fontfile=%FONT%:text=%{pts\:hms:%OFFSET%}:fontcolor=%COLOR%:boxcolor=%BCOLOR%:box=1:fontsize=%SIZE
:x=%POS_X%:y=%POS_Y%'

rem Extract the first frame:
rem %FF% -ss %S% -i %IN% -vf "colorlevels=rimin=%BP_R%:gimin=%BP_G%:bimin=%BP_B%:rimax=%WP%:gimax=%WP%:bimax=%WP%"
"frames 1 -y frameml.png
rem Find the coordinates of the star (with IrfanView) and set them to the variables X and Y.
rem Then create a video with light curve and audio level curve. The light curve can be YAVG for average over the box or YMAX for maximum in the box.

%FF% -ss %S% -i %IN% -lavfi [0:v]crop=2*%B%:2*%B%:%X%-%B%:%Y%-%B
%signalstats,drawgraph=ml= lavfi.signalstats.YAVG:mode=line:slide=scroll:min=%MIN%:max=%MAX%:size=1920x200:fg1=%C1%:bg=%BG%[1];
[0:a]asetnsamples=n=1920,astats=metadata=1,drawgraph=ml= lavfi.astats.1.RMS_level:mode=bar:slide=scroll:min=%MIN%:max=%MAX%:size=1920x200:fg1=%C1%:bg=%BG%[2];[1][2]vstack[3];[0]colorlevels=rimin=%BP_R%:gimin=%BP_G%:bimin=%BP_B%
%:rimax=%WP%:gimax=%WP%:bimax=%WP%,drawbox=x=%X%+2*%B%:y=%Y%:c=White:t=1:w=2*%B%:h=1,drawbox=x=%X%:y=%Y%+2*%B%
%:c=White:t=1:w=1:h=2*%B%[4];[4][3]overlay=y=750,%CLOCK% -pix_fmt yuv420p -t %T% -y %OUT%

pause

The value 1920 for asetnsamples is the number of audio samples per video frame, in this case 48000 / 25 = 1920.
2.88 Oscilloscope

The oscilloscope filter can show the brightness over a video line:

```
set "FF=c://ffmpeg/ffmpeg" :: Path to FFmpeg
set "IN=P1000479.mov" :: Input video
set "OUT=out.mp4" :: Output video
set "LINE=495" :: The shown line
set "H=1080" :: Height of the video

%FF% -i %IN% -lavfi "oscilloscope=x=0.5:y=%LINE%/%H%:c=1" -t 10 -y %OUT%
pause
```

c=1 means show only the first component, in this case the luma component.
c=7 means show the first three components.
If you want to show the RGB components, add "format=rgb24," before the oscilloscope filter.
Capture a video from a webcam

List all supported, connected capture devices:

```
c:\ffmpeg\ffmpeg -list_devices 1 -f dshow -i dummy
```

List the possible video sizes, frame rates ans pixel formats for one capture device:

```
c:\ffmpeg\ffmpeg -list_options 1 -f dshow -i video="HD WebCam"
```

Capture video from the webcam:

```
c:\ffmpeg\ffmpeg -y -f dshow -video_size 1280x720 -framerate 10 -pixel_format yuyv422 -i video="HD WebCam" -t 5 out.mp4
```

See also: [https://trac.ffmpeg.org/wiki/DirectShow](https://trac.ffmpeg.org/wiki/DirectShow)
2.90 Adding subtitles to a video

Create a *.srt subtitle file with the content in this format, the text must begin in the first line:

<table>
<thead>
<tr>
<th>Time</th>
<th>Text</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 00:00:00,000 --&gt; 00:00:03,000</td>
<td>This is the title text</td>
</tr>
<tr>
<td>2 00:00:04,000 --&gt; 00:00:05,000</td>
<td>Hello !</td>
</tr>
<tr>
<td>3 00:00:06,500 --&gt; 00:00:07,000</td>
<td>This video will end soon...</td>
</tr>
</tbody>
</table>

Method 1, burning the subtitles directly into the video frames:

c:\ffmpeg\ffmpeg -i subtitle.srt -y subtitle.ass

c:\ffmpeg\ffmpeg -i test.mp4 -vf ass=subtitle.ass -y out.mp4

Method 2, the output seems to be the same as with method 1:

c:\ffmpeg\ffmpeg -i test.mp4 -vf subtitles=subtitle.srt -y out.mp4

Method 3, creating a subtitle stream that can be switched on/off in the player. Works fine with VLC player, but not with FFplay:

c:\ffmpeg\ffmpeg -i test.mp4 -i subtitle.srt -c:v copy -c:a copy -c:s mov_text -metadata:s:s:0 language=ger -y out.mp4

See also https://jacknorthrup.com/Multiple-Program-Languages-Documentation/FFMPEG.html
## 2.91 Expression evaluation

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>abs(x)</code></td>
<td>Return absolute value of x.</td>
</tr>
<tr>
<td><code>acos(x)</code></td>
<td>Return arccosine of x.</td>
</tr>
<tr>
<td><code>asin(x)</code></td>
<td>Return arcsine of x.</td>
</tr>
<tr>
<td><code>atan(x)</code></td>
<td>Return arctangent of x.</td>
</tr>
<tr>
<td><code>atan2(x, y)</code></td>
<td>Return the arc tangent of y/x in the range -PI to PI.</td>
</tr>
<tr>
<td><code>between(x, min, max)</code></td>
<td>Return 1 if x is greater than or equal to min and lesser than or equal to max, 0 otherwise.</td>
</tr>
<tr>
<td><code>bitand(x, y)</code></td>
<td>Compute bitwise and operation on x and y.</td>
</tr>
<tr>
<td><code>bitor(x, y)</code></td>
<td>Compute bitwise or operation on x and y.</td>
</tr>
<tr>
<td><code>ceil(expr)</code></td>
<td>Round the value of expression expr upwards to the nearest integer. For example, &quot;ceil(1.5)&quot; is &quot;2.0&quot;.</td>
</tr>
<tr>
<td><code>clip(x, min, max)</code></td>
<td>Return the value of x clipped between min and max.</td>
</tr>
<tr>
<td><code>cos(x)</code></td>
<td>Compute cosine of x.</td>
</tr>
<tr>
<td><code>eq(x, y)</code></td>
<td>Return 1 if x and y are equivalent, 0 otherwise.</td>
</tr>
<tr>
<td><code>exp(x)</code></td>
<td>Compute exponential of x.</td>
</tr>
<tr>
<td><code>floor(expr)</code></td>
<td>Round the value of expression expr downwards to the nearest integer. For example, “floor(-1.5)” is “-2.0”.</td>
</tr>
<tr>
<td><code>gt(x, y)</code></td>
<td>Return 1 if x is greater than y, 0 otherwise.</td>
</tr>
<tr>
<td><code>gte(x, y)</code></td>
<td>Return 1 if x is greater than or equal to y, 0 otherwise.</td>
</tr>
<tr>
<td><code>hypot(x, y)</code></td>
<td>Return sqrt(x^2 + y^2)</td>
</tr>
<tr>
<td><code>if(x, y)</code></td>
<td>Evaluate x, and if the result is non-zero return the result of the evaluation of y, return 0 otherwise.</td>
</tr>
<tr>
<td><code>if(x, y, z)</code></td>
<td>Evaluate x, and if the result is non-zero return the evaluation result of y, otherwise the evaluation result of z.</td>
</tr>
<tr>
<td><code>ld(var)</code></td>
<td>Load the value of the internal variable with number var, which was previously stored with st(var, expr). The function returns the loaded value.</td>
</tr>
<tr>
<td><code>lerp(x, y, z)</code></td>
<td>Return x if z = 0, y if z = 1 and a linear interpolation for any value of z. There is no clipping for z &lt; 0 or z &gt; 1. The return value is: x + z * (y - x)</td>
</tr>
<tr>
<td><code>log(x)</code></td>
<td>Compute natural logarithm of x.</td>
</tr>
<tr>
<td>Function</td>
<td>Description</td>
</tr>
<tr>
<td>------------</td>
<td>-------------</td>
</tr>
<tr>
<td>lt(x, y)</td>
<td>Return 1 if x is lesser than y, 0 otherwise.</td>
</tr>
<tr>
<td>lte(x, y)</td>
<td>Return 1 if x is lesser than or equal to y, 0 otherwise.</td>
</tr>
<tr>
<td>max(x, y)</td>
<td>Return the maximum between x and y.</td>
</tr>
<tr>
<td>min(x, y)</td>
<td>Return the minimum between x and y.</td>
</tr>
<tr>
<td>mod(x, y)</td>
<td>Return the remainder of division of x by y.</td>
</tr>
<tr>
<td>pow(x, y)</td>
<td>Return the power of x elevated y, it is equivalent to &quot;(x)^y&quot;.</td>
</tr>
<tr>
<td>print(t)</td>
<td>Print the value of expression t and returns the value of the expression printed.</td>
</tr>
<tr>
<td>random(x)</td>
<td>Return a pseudo random value between 0.0 and 1.0. x is the index of the internal variable which will be used to save the seed/state.</td>
</tr>
<tr>
<td>round(expr)</td>
<td>Round the value of expression expr to the nearest integer. For example, &quot;round(1.5)&quot; is &quot;2.0&quot;.</td>
</tr>
<tr>
<td>sgn(x)</td>
<td>Return the sign of x (-1, 0 or +1)</td>
</tr>
<tr>
<td>sin(x)</td>
<td>Return sine of x.</td>
</tr>
<tr>
<td>sqrt(x)</td>
<td>Return the square root of x.</td>
</tr>
<tr>
<td>st(var, expr)</td>
<td>Store the value of the expression expr in an internal variable. var specifies the number of the variable where to store the value, and it is a value ranging from 0 to 9. The function returns the value stored in the internal variable. Note, variables are currently not shared between expressions.</td>
</tr>
<tr>
<td>tan(x)</td>
<td>Compute tangent of x.</td>
</tr>
<tr>
<td>trunc(expr)</td>
<td>Round the value of expression expr towards zero to the nearest integer. For example, &quot;trunc(-1.5)&quot; is &quot;-1.0&quot;.</td>
</tr>
<tr>
<td>while(cond, expr)</td>
<td>Evaluate expression expr while the expression cond is non-zero, and returns the value of the last expr evaluation, or NAN if cond was always false.</td>
</tr>
</tbody>
</table>

PI approximately 3.1415

Two expressions expr1 and expr2 can be combined to form another expression "expr1;expr2". expr1 and expr2 are evaluated in turn, and the new expression evaluates to the value of expr2.

Workaround for segment-wise linear interpolation, one segment per second:
```
between(t,0,1) * lerp(v0,v1,t) + between(t,1,2) * lerp(v1,v2,t-1) + between(t,2,3) * lerp(v2,v3,t-2) + ...
```

Workaround for segment-wise linear interpolation, two segments per second:
```
between(t,0,0.5) * lerp(v00,v05,2*t) + between(t,0.5,1.0) * lerp(v05,v10,2*(t-0.5)) + between(t,1.0,1.5) * ...
```
The above two workarounds have a problem: if t is exactly at the border of two segments, then both "between" expressions are true. As a workaround, you can add 0.0001 to t.

If used inside the geq filter, the variable 't' must be written as a capital 'T'.

Please note that for most parameters expressions can't be used. They are only allowed for those parameters which are described as "expr" in the documentation. If in doubt, use ffmpeg -h filter= name_of_filter to get the types of the parameters.

2.92 Uploading videos to Facebook

It's possible to upload 10-bit videos to Facebook, but there may be a problem with the preview image, which is shown corrupted.

Use an 8-bit pixel format to avoid this problem, for example -pix_fmt yuv420p

2.93 Hardware acceleration

This command lists all hardware acceleration methods supported in this build of FFmpeg, regardless if the hardware is really available in this computer:

```
set "FF=c:\ffmpeg\ffmpeg" :: Path to FFmpeg

%FF% -hwaccels

pause
```

See also: https://trac.ffmpeg.org/wiki/HWAccelIntro
2.94  FFmpeg console output

| Explanation:                                                                 |
| SAR  | Sample Aspect Ratio |
| DAR  | Display Aspect Ratio |
| fps  | frames per second   |
| tbr  | This is guessed from the video stream and is the value users want to see when they look for the video frame rate, except sometimes it is twice what one would expect because of field rate versus frame rate. |
| tbn  | The time base in AVStream that has come from the container. It is used for all AVStream time stamps. |
| tbc  | The time base in AVCodecContext for the codec used for a particular stream. It is used for all AVCodecContext and related time stamps. |

2.95  My To Do List

-- find out how to compile FFmpeg with options (difficult...)
-- learn how GIT works (difficult...)
-- Find out how to process videos with DaVinci Resolve
-- Build an invisible tripod
-- I know that the Windows scripts in this book need some modifications if they are to be used on MAC or Linux computers. But I have no experience with MAC and Linux. Who wants to write a short chapter about the differences of Windows/MAC/Linux scripts? I’d like to add it to this book, of course with proper credit to the author.
-- Redshift, Blueshift
-- Stitching 360° videos from more than two cameras
2.96 FFmpeg: Suggestions for improvement

Listed in alphabetic order:

"acrossover" audio filter:
-- Documentation: Missing examples.

"afftfilt" filter:
-- In the documentation for the win_func parameter the possible options are missing.
-- Also, for the overlap parameter, what are the default values if set to 1?

"amplify" filter:
-- Many parameters are in the [0..65535] range. It's unclear how these parameters must be set for 8-bit videos. Is it [0..255] in this case, or is it always [0..65535]? Which range for a 10-bit video, [0..1023] or [0..65535]? Please add this info to the documentation.
-- Documentation for "threshold" parameter: "Any difference greater or equal to this value..." I'm not sure if this is correct, or if it should read "Any absolute value of the difference greater or equal to this value...". Same question also for "tolerance" parameter.
-- Does it have a built-in limiter?

"blend" filter:
-- Please add documentation what all the modes do. A few modes are self-explaining, but most are not.

"colorbalance" filter:
I don't understand what's written in the documentation. In the first sentence it's said the filter modifies the intensity. What's meant here by "intensity"? Brightness or saturation? However in the third sentence it's said the colors are shifted. For me it's unclear what this filter does. More examples and/or a better description is required.
"cue" filter:
-- please add some examples to the documentation

"curves" filter:
-- In the documentation for "preset", please add the coordinates of the points for all presets. For example, what does "strong_contrast" mean? How strong is it? It would really help to know the coordinates of the points. This would also be usable as a starting point for defining your own curves.
-- Documentation of "master" option: I didn't understand it, please add an example and explain it.
-- Feature request: Allow import of curves files from GIMP.
-- Feature request: Make an option for using straight lines instead of smooth curves. This would be a nice workaround for strong linear contrast enhancement, using only four points: 0/0 b/0 w/1 1/1 where parameter b is the black point and w is the white point.

"deband" filter:
-- Documentation: Missing examples. I don't understand the description of this filter. The unit of the threshold values is unclear.

"deflicker" filter:
-- Documentation: It's unclear how this filter corrects the brightness of the images. Does it add a constant, or does it multiply by a constant? With other words: Does it change the brightness or the contrast?

"derain" filter:
-- Documentation: Please add examples.

"drawgraph" filter:
-- Make clear in the documentation that the color format for the graphs is 0xAABBGGRR, however the color format for the background is 0xRRGGBBAA.

"eq" filter:
-- Please add to the documentation: "Do note that what contrast does is scale the distance of a pixel's value from the median value i.e. 128 for a 8-bit input. So, if a pixel channel has a value of 100, then a contrast of 3 results in a value of  128 + 3*(100-128) = 44. "

137
-- Please also add to the documentation that the order is always contrast -> brightness -> gamma, regardless of the order in the command line. Also fill in where saturation fits in the order (I don't know).

-- Please add this example for brightness before contrast: eq=brightness=0.3,eq=contrast=5, and explain that eq=brightness=0.3:contrast=5 will be executed in the order contrast before brightness.

-- It might be helpful to point out that this filter includes a limiter, if the result is out of range.

-- Feature request:

If we assume the video data is in the [0..1] range, the transfer function (before applying gamma) is: out = brightness + 0.5 + contrast * (in - 0.5)

This works fine for images where most pixels are in the center of the histogram. But it fails for images which consist mainly of a dark background, with only a few bright details. To amplify contrast in such images, both brightness and contrast must be adjusted, which is complicated.

I suggest to add a new parameter "pivot" in the range [0..1] which is 0.5 by default (so that it's compatible with old command lines).

out = brightness + pivot + contrast * (in - pivot)

With the p value properly set, the contrast can be adjusted without changing the brightness. This feature is already implemented in 3D_LUT_Creator.

Equirectangular 360° spherical videos:

-- Feature request: Inserting the required metadata so that VLC (and other players) recognize these videos as spherical. The same thing that the "Spatial Media Metadata Injector" does.

Expression evaluation:

-- 'lerp(x, y, z)' Documentation, better explanation: lerp(x,y,z) returns x if z=0, y if z=1 and interpolates if 0<z<1. The return value is x + z * (y - x). There is no clipping for z < 0 or z > 1.

-- 'taylor(expr, x)' 'taylor(expr, x, id)' Documentation: Better explanation and example required. I know what a taylor series is, but I don't understand how to use this FFmpeg function.

-- Feature request: A new function getval(x, y, filename) which reads a text file (or CSV file) and returns x-th value from the y-th line. Usable for many purposes where you have a parameter that can't be expressed by a simple formula, so that reading from an external file is easier. For example if you want to overlay a small video over the main video, at a variable position which is an arbitrary function of time.

-- Feature request: Share variables between expressions. This is especially required if inside the geq filter a variable is calculated by a long and complicated expression which depends only on T (with other words: a function of time). This expression will be evaluated for each pixel again and again, making the filter extremely slow. It would be much better to calculate the variable only one time per frame, and then make this variable readable inside the geq filter.

-- Feature request: Allow to set variables by reading from an input text or CSV file.
-- Feature request: A function for segment-wise linear or spline interpolation of a curve through a set of given points. Similar to this workaround for segment-wise linear interpolation: between(T+0.001,0,1)*lerp(200,400,T)+between(T+0.001,1,2)*lerp(400,500,T-1)+between(T+0.001,2,3)*lerp(500,550,T-2)

"fade" filter:
-- Feature request: Please allow a fade-out at the end of the video, where the start time is defined relative to the end of the video. With other words it should work without knowing the length of the video.

"filter_complex_script"
-- Documentation: Please add that line feeds and empty lines are allowed in the script file, which makes the content much better readable.

"find_rect" filter:
-- Feature request: Write the x,y coordinates to a text or CSV file
-- Documentation: Can find_rect find multiple instances of the same object in a frame?

"geq" filter:
-- Feature request: Allow commands, especially it should be possible to set a variable sd() inside the geq argument.
-- Please add to the documentation that this filter has no built-in limiter for overflow handling when the result is out of range, and that the functions must be different for 8-bit or 10-bit videos. This filter doesn't interpret video data in the [0..1] range, but instead [0..255] for 8-bit video and [0..1023] for 10-bit video. This isn't clear in the documentation.

"overlay" filter:
-- Feature request: "enable" option

"perspective" filter:
-- When the "sense=destination" option is used, the output size is smaller than the input size and the outer area is filled with the colors of the pixels at the edge of the video. It would be better to define a uniform color, with black as default. As a workaround I used the "pad" filter to create a black border around the video, before using the perspective filter. Please see my example in the "Video-in-Video" chapter.
"scale" filter:
-- Feature request: An option for reducing the size of an image, which calculates the brightness of the destination pixel not by averaging the source pixels, but instead by choosing the brightest of the source pixels. Usable for shrinking images of the starry night sky, without loosing faint stars.

"selectivecolor" filter:
-- I don't understand the difference between "absolute" and "relative". Please add a better explanation and an example to the documentation.

"sendcmd" filter:
-- Documentation: Please add an example for sendcmd, if the target filter has more than one input. In this case the sendcmd filter can't be inserted directly before the target filter. When choosing a suitable position for sendcmd in the filter chain, make sure that at this position the duration is sufficient. If you have signal sources of different lengths, always choose the longest one for sendcmd.
-- Documentation: All arguments of the target filter must be initialized with valid values, even if these values are never used because sendcmd does always overwrite them.
-- Feature request: Allow that the sendcmd filter accepts any number of inputs, and just pass the inputs to the next filter. This will simplify things if you need a sendcmd before a target filter which has more than one input.
-- Feature request: Add a new variable for expressions: \( Ti = \frac{(T - Ts)}{(Te - Ts)} \) where Ts and Te are the start and end times of the current interval. This variable is running from 0 to 1 in the interval and can be used for the "lerp" function. The start and end times of the interval could be changed, without changing the expression.

"streamselect" filter:
-- Documentation: The examples are misleading. In the more general case it's impossible to write sendcmd directly before streamselect, because streamselect requires at least two inputs but sendcmd accepts only one input.
-- Feature request: Please allow expressions for "map". That's easier than sendcmd.

"superequalizer" audio filter:
-- Documentation: The unit of the parameters is unclear. Example missing.

"tmix" filter:
-- Undocumented feature missing in documentation: If not specified, all weights are 1 by default.
-- Add the most simple example to the documentation: "tmix" works and is the same as "tmix=frames=3"
-- Undocumented feature missing in documentation: "tmix=frames=1" works and is a bypass mode.
-- The last sentence in the documentation is misleading and should be changed to "By default scale is set to 1 / (sum of weights)".

"trim" filter:
-- Feature request: Please allow to specify the end time relative to the end of the video, so that it's possible to trim 5 seconds from the end of the video, without having to know the video length.

"vibrato" audio filter
-- Documentation: It's unclear how a phase modulation can be expressed as a percentage. What does d=1.0 mean?

"xfade" filter:
It would be nice if it could be used similar as "acrossfade", where the crossfade is always at the end of the first video, so that you don't have to know the length of the first video. Could be used as a workaround for fade-out at the end of a video, by crossfading with a black video. I know that this requires a lot of memory, but this is acceptable as in most cases crossfades aren't longer than 1-2 seconds.

Feature request: "Black hole" filter, which simulates the bending of light rays near a black hole. There are several workarounds in this document, as C# code and also as a long and complicated expression for geq filter. The filter should accept four input parameters: x_center, y_center, radius, shape. All parameters should allow expressions (for moving black holes, and for dynamically changing the radius).

Feature request for reading PGM files (required for example for "remap" filter): These files should contain only values in the [0..65535] range. In the case of ASCII PGM (P2) files, it's theoretically possible that the file contains negative values. FFmpeg should give a warning if a negative value is found.

Feature request: Cross correlation for audio. For comparing two audio channels from the same sound source, which were recorded with two microphones at different places. Automatically find the best-fit delay time between two audio channels.

There are a few pairs of video filters which share a common description: blend/tblend, lut2/tlut2, setdar/setsar and weave/doubleweave.
The problem is that the second filter is difficult to find if you search by alphabet. I suggest to list these filters in correct alphabetic order, and the description of the second one contains only a link to the other filter.
Same problem also for "afifo", "agraphmonitor" and "astreamselect" which are listed only in "Video filters". They should be listed in "Audio filters". A link which points to the corresponding video filter would be sufficient.
3 Audio processing with FFmpeg

3.1 Combine multiple audio files with crossfadings

```
rem Combine multiple audio files with crossfadings

set "FF=c:\ffmpeg\ffmpeg" :: Path to FFmpeg
set "FILE1=Sound_Day.wav" :: First audio filename
set "V1=1.0" :: Volume
set "S1=0" :: Start time
set "L1=14" :: Length

set "FILE2=Sound_Night.wav" :: Second audio filename
set "V2=0.2" :: Volume
set "S2=20" :: Start time
set "L2=55" :: Length

set "FILE3=Sound_Day.wav" :: Third audio filename
set "V3=1.0" :: Volume
set "S3=20" :: Start time
set "L3=30" :: Length

set "DUR=5" :: Crossfade duration
set "OUT=sound.mp3" :: Output audio filename

%FF% -ss %S1% -i %FILE1% -t %L1% -af volume=%V1% -y s1.wav
%FF% -ss %S2% -i %FILE2% -t %L2% -af volume=%V2% -y s2.wav
%FF% -ss %S3% -i %FILE3% -t %L3% -af volume=%V3% -y s3.wav
%FF% -i s1.wav -i s2.wav -filter_complex acrossfade=d=%DUR% -y s12.wav
%FF% -i s12.wav -i s3.wav -filter_complex acrossfade=d=%DUR% -y %OUT%
```

In this example three audio files are concatenated with crossfadings. For each file the volume, start time and length can be specified.

At first three temporary files are created, then the first two are combined, and in the last step the third file is added.

There is no quality loss because *.wav is an uncompressed audio format.
3.2 Change audio sample rate and number of audio channels

```bash
rem Change the audio sample rate and the number of audio channels

set "FF=c:\ffmpeg\ffmpeg" :: Path to FFmpeg
set "IN=PanoView.mp4" :: Input video
set "START=5" :: Start time
set "LEN=5" :: Length
set "OUT=out.mp4" :: Output video

%FF% -ss %START% -t %LEN% -i %IN% -ac 2 -af aresample=44100 -y %OUT%

pause
```

- `-ac 2` sets the number of audio channels to 2. If you want to copy a mono channel to both stereo channels, use `aeval=val(0)|val(0)`
- `-af aresample=44100` changes the audio sample rate to 44100 Hz

3.3 Replace a segment of the audio stream by silence

```bash
set "FF=c:\ffmpeg\ffmpeg" :: Path to FFmpeg
set "B=10" :: Time where silence begins
set "E=10" :: Time where silence ends

%FF% -i in.mp4 -c:v copy -af "volume=enable='between(t,%B%,%E%)':volume=0" out.mp4

pause
```
3.4 Stereo --> mix into one mono channel

Both channels of the stereo stream will be downmixed into the stream:

```
ffmpeg -i stereo.wav -ac 1 mono.wav
```

3.5 Check if both stereo channels are equal

In this example the right audio channel is inverted and then both channels are mixed into a mono channel. If the result is silence, then both input channels were equal. The input can be a video or an audio file.

```
c:\ffmpeg\ffmpeg -i input.mp4 -af "aeval=val(0)|-val(1)" -ac 1 mono.wav
```

3.6 Extract one mono channel from stereo

```
ffmpeg -i stereo.wav -filter_complex "[0:a]channelsplit=channel_layout=stereo:channels=FR[right]" -map "[right]" front_right.wav
```

3.7 Stereo --> two mono channels

```
ffmpeg -i stereo.wav -filter_complex "[0:a]channelsplit=channel_layout=stereo[left][right]" -map "[left]" left.wav -map "[right]" right.wav
```

This command line does the same thing:

```
ffmpeg -i stereo.wav -map_channel 0.0.0 left.wav -map_channel 0.0.1 right.wav
```
3.8 Mono --> stereo

Of course both stereo channels will be identical.

```
ffmpeg -i input.wav -ac 2 output.wav
```

3.9 Two mono channels --> stereo

```
ffmpeg -i left.mp3 -i right.mp3 -filter_complex ":[0:a][1:a]join=inputs=2:channel_layout=stereo[a]" -map ":[a]" output.mp3
```

3.10 Mix two stereo channels to one stereo channel

```
ffmpeg -i input1.wav -i input2.wav -filter_complex ":[0:a][1:a]amerge=inputs=2[a]" -map ":[a]" -ac 2 output.mp3
```

3.11 How to choose the correct audio volume level

Normally music is normalized to the maximum value (+-32676 for 16-bit). That means the loudest part uses the maximum possible values, just without clipping. You can use the music for your video as-is, or you can make it quieter. If you make it louder, then it may be clipped.

Things are totally different when you make your own sound records, for example nature sounds.

As the first step, I recommend to calibrate the volume knob of your amplifier. To do this, show several videos from different sources (not your own selfmade videos), and adjust the volume knob so that all videos sound just right, with other words: Adjust the volume knob so, as you would like to hear these videos in the planetarium. To make sure that the frequency response is acceptable, use good 3-way boxes. Leave the volume knob in this position and don't change it.

Now you can adjust the volume of your own video, so that it also sounds great in the planetarium. This ensures that you can play all videos (your own and other videos) one after the other. You don't want to touch the volume knob during a presentation!
3.12  Make a video from an audio file

Suppose an audio file is to be shown on Facebook. However, this is not possible because only pictures or videos can be shown there. Solution: The audio file is extended with a monochrome picture to a video.

```
c:\ffmpeg\ffmpeg -lavfi -i color=c=black -i audio.mp3 -shortest out.mp4
```  

Do the same thing with a picture:

```
c:\ffmpeg\ffmpeg -loop 1 -i picture.jpg -i audio.mp3 -shortest out.mp4
```  

3.13  Remove low frequencies (wind noise) from an audio track

```
rem Audio high pass filtering and volume adjustment
set "FF=c:\ffmpeg\ffmpeg" :: Path to FFmpeg
set "IN=sound.wav" :: Input soundtrack
set "AS=20" :: Start time
set "LEN=60" :: Length
set "HP=500" :: Cut-off frequency of the high pass filter
set "VOL=10" :: Volume factor
set "OUT=out.mp3" :: Output soundtrack
%FF% -ss %AS% -i %IN% -af highpass=f=%HP%,highpass=f=%HP%,highpass=f=%HP%,volume=%VOL% -t %LEN% -y %OUT%
```  

The high pass filter attenuates low frequencies by 6 dB per octave. At the specified cut-off frequency, the filter has 3dB attenuation. In this example, the same filter is used three times in a row, resulting in 18dB per octave.
3.14 Convert ultrasound to the audible range, e.g. to hear bats

There are two fundamentally different methods of converting a high frequency to a lower frequency. The first method is to divide the frequency by a constant (for example 2), in this case the frequency range from 0 to 25kHz is converted to the range from 0 to 12.5kHz.

rem Make 2 seconds 1kHz sinewave, followed by 2 seconds silence:
c:\ffmpeg\ffmpeg -f lavfi -i "sine=frequency=1000:sample_rate=48000:duration=2" -af 'apad -t 4' sine.wav

rem Halving the sampling rate doubles the duration and halves the pitch
c:\ffmpeg\ffmpeg -i sine.wav -af 'asetrate=24000' out1.mp3

rem The atempo=2 filter causes the duration to be halved and the pitch to remain unchanged
rem (The factor must be in the range [0.5 .. 2.0], if necessary you can use the filter several times in a row)
c:\ffmpeg\ffmpeg -i sine.wav -af 'atempo=2.0' out2.mp3

rem A combination of these two effects causes the duration to remain unchanged and the pitch to be halved:
c:\ffmpeg\ffmpeg -i sine.wav -af 'asetrate=24000,atempo=2.0' out3.mp3

pause

The second method is to subtract a constant frequency. In this case, for example, the frequency range [15kHz ... 25kHz] is converted to the range [0kHz ... 10kHz]. The advantage is that the frequency range from 0 to 15kHz is completely suppressed.

rem Ultrasound converter by mixing (subtraction of the mixing frequency)
set "IN=Fledermaus_44100.wav"     :: Input soundtrack (containing ultrasound)
set "SR=44100"                    :: Sample rate of the input soundtrack
set "MF=15000"                    :: Mixing frequency (this is the frequency to be subtracted)
set "BB=10000"                    :: Bandwidth
    :: The frequency range [MF ... MF+BB] is converted to the range [0Hz ... BB]
set "VOL=3"                       :: Volume factor
set "OUT=out.wav"                 :: Output soundtrack
c:\ffmpeg\ffmpeg -ss 100 -i %IN% -f lavfi -i aevalsrc="sin(%MF%*2*PI*t):c=stereo:s=%SR%" ^
-filter_complex "[0]volume=%VOL%,highpass=f=%MF%,highpass=f=%MF%,highpass=f=%MF%,highpass=f=%MF%[sound];
[sound][1]amultiply,lowpass=f=%BB%,lowpass=f=%BB%,lowpass=f=%BB%,lowpass=f=%BB%" -y %OUT%

pause
The amultiply filter in the above example does multiply the input signal by a sine wave. Both inputs and the output are in the [-1...+1] range.

In this example, the ultrasound from a video is mixed to the audible range and the video is copied as-is:

```plaintext
rem Ultrasound converter by mixing (subtraction of the mixing frequency)

set "IN=7Z7A1699.MOV"            :: Input video
set "SR=48000"                    :: Sample rate of the input soundtrack
set "MF=12000"                    :: Mixing frequency (this is the frequency to be subtracted)
set "BB=10000"                    :: Bandwidth
                                :: The frequency range [MF ... MF+BB] is converted to the range [0Hz ... BB]
set "VOL=40"                      :: Volume factor
set "OUT=699.mp4"                 :: Output video

\c:\ffmpeg\ffmpeg -i %IN% -f lavfi -i aevalsrc="sin(%MF%*2*PI*t):c=stereo:s=%SR%" ^
-filter_complex "[0]volume=%VOL%,highpass=f=%MF%,highpass=f=%MF%,highpass=f=%MF%,highpass=f=%MF%[sound];
[sound][1]amultiply,lowpass=f=%BB%,lowpass=f=%BB%,lowpass=f=%BB%,lowpass=f=%BB%,lowpass=f=%BB%" -y %OUT%
```

If the output audio sample rate is specified with -ar, it must be a sane sample rate such as 44.1k or 48k.
3.15 Record sound with the computer's built-in microphone

With this batch file you can see which microphones are available:

```
c:\ffmpeg\ffmpeg -list_devices 1 -f dshow -i dummy
```

With this batch file you can display the properties of a specific microphone:

```
c:\ffmpeg\ffmpeg -list_options 1 -f dshow -i "audio=Mikrofon (Realtek High Definiti"
```

With this batch file you can record sound with the internal microphone:

```
c:\ffmpeg\ffmpeg -f dshow -sample_rate 44100 -sample_size 16 -channels 2 -i "audio=Mikrofon (Realtek High Definiti" -t 5 -f mp3 -y out.mp3
```

With this batch file you can record sound with a cheap chinese USB soundcard (Model "3D SOUND"):

```
rem c:\ffmpeg\ffmpeg -list_devices 1 -f dshow -i dummy
rem c:\ffmpeg\ffmpeg -list_options 1 -f dshow -i "audio=Mikrofon (USB Audio Device)"
c:\ffmpeg\ffmpeg -f dshow -sample_rate 44100 -sample_size 16 -channels 2 -i "audio=Mikrofon (USB Audio Device)" -t 5 -f mp3 -y out.mp3
```

```
pause
```
3.16 Record a "Voice-Over" audio track

A "Voice-Over" track is an audio track that's recorded while simultaneously a video is played. Useful also for making sound effects that fit to the video.

Note: Later I found out that it's much easier to record a Voice-Over track with Davinci Resolve. But for completeness I'm also showing here how it can be done with FFmpeg.

```
set "FF=c:\ffmpeg\ffmpeg"   :: Path to FFmpeg
set "IN=test.mp4"           :: Input video
set "AUDIO=audio.wav"       :: Output audio

%FF% -re -i %IN% -f dshow -audio_buffer_size 100 -channels 2 -i audio="Mikrofon (Realtek High Definiti" -y -map 1:a %AUDIO% -map 0:v -f sdl2 -
```

Please note that this command line has several problems:

1. It doesn't stop when the end of the video has been reached. The audio file gets longer than the video. But you can manually close the console window.
2. Video and audio are not perfectly synchrone. This depends also on the "audio_buffer_size" value.
3. If audio_buffer_size is large (or if the large default value is used), FFmpeg doesn't play the video continously. It's stop-and-go.
4. I found no documentation for -f sdl2

Alternatively it's also possible to pipe the video output to FFplay. In this case the scale filter is required to make the FFplay window smaller, so that you can still see the console window and close it manually when the video has ended.

```
set "FF=c:\ffmpeg\ffmpeg"   :: Path to FFmpeg
set "FP=c:\ffmpeg\ffplay"   :: Path to FFplay
set "IN=test.mp4"           :: Input video
set "AUDIO=audio.wav"       :: Output audio

%FF% -an -i %IN% -f dshow -audio_buffer_size 100 -channels 2 -i audio="Mikrofon (Realtek High Definiti" -y -map 1:a %AUDIO% -map 0:v -vf scale=iw/2:-1 -f nut - | %FP% -
```

This example that was proposed by Gyan Doshi in the FFmpeg user list on January 17th, 2020. It uses the mpegts format instead. It works, but unfortunately it takes about 5 seconds until the FFplay window appears. The -autoexit option should close FFplay when EOF is detected, but in this
example it doesn't work.

```
c:\ffmpeg\ffmpeg -an -i test.mp4 -f dshow -audio_buffer_size 100 -channels 2 -i audio="Mikrofon (Realtek High Definition)" -y -map 1:a audio.wav -map 0:v -vf scale=iw/2:-1 -f mpegts - | c:\ffmpeg\ffplay -f mpegts -autoexit -
```

The original video (with original audio) and the new recorded audio can now be mixed with this command line. The best value for "offset" has to be found by try and error.

```
set "FF=c:\ffmpeg\ffmpeg"   :: Path to FFmpeg
set "IN=test.mp4"           :: Input video (with audio)
set "AUDIO=audio.wav"       :: Input audio
set "OFFSET=0.35"           :: Offset time in seconds, a positive value means audio is shifted towards the beginning
set "W1=0.2"                :: Weight of original sound from the video
set "W2=2.5"                :: Weight of sound from audio file

%FF% -i %IN% -i %AUDIO% -filter_complex "[1:a]atrim=%OFFSET%[2],[0:a][2]amix=weights='%W1% %W2%'
```

Note: In the above example I did use the mpeg4 encoder, because with the default lib264 encoder there is a problem. Unexpectedly the audio in the output file is shorter than the video. The problem can be reproduced as follows:

```
c:\ffmpeg\ffmpeg -f lavfi -i testsrc2=size=vga -f lavfi -i sine=1000 -t 6 -y video.mp4
c:\ffmpeg\ffmpeg -i video.mp4 -i video.mp4 -lavfi "[0:a][1:a]amix=weights='1.0 0.1'" -y -shortest -q:v 2 -c:v mpeg4 out.mp4
```

rem 3 known workarounds:
rem c:\ffmpeg\ffmpeg -i video.mp4 -i video.mp4 -lavfi "[0:a][1:a]amix=weights='1.0 0.1'" -c:v mpeg4 -y out.mp4
rem c:\ffmpeg\ffmpeg -i video.mp4 -i video.mp4 -lavfi "[0:a]apad[];[p][1:a]amix=weights='1.0 0.1'" -shortest -y out.mp4
rem c:\ffmpeg\ffmpeg -i video.mp4 -i video.mp4 -lavfi "[0:a]apad=pad_len=1[p];[p][1:a]amix=weights='1.0 0.1'" -y out.mp4
```
3.17  Passing the FFmpeg output to FFplay

This batch file passes the video and audio output of FFmpeg to FFplay

```
c:\ffmpeg\ffmpeg -i in.mp4 (insert some filters here) -f nut - | c:\ffmpeg\ffplay -pause
```

3.18  Record sound and pass the output to FFplay

This batch file records sound from the computer's microphone (or audio input) and passes the output to FFplay

```
c:\ffmpeg\ffmpeg -f dshow -sample_rate 44100 -sample_size 16 -channels 2 -i "audio=Microphone (SoundMAX Integrated) (insert some filters here) -f wav - | c:\ffmpeg\ffplay -pause
```
3.19 Live ultrasound conversion

It's possible to make ultrasound conversion in almost real time. You can input the ultrasound via the computer's microphone (or 3.5mm input jack), and play the converted sound via the computer's speakers (or 3.5mm headphone output to an audio amplifier). The conversion has about 1-2 seconds delay. Make sure that in the Windows control panel, in the microphone properties under "Microphone Extensions", no filter should be used.

```plaintext
rem c:\ffmpeg\ffmpeg -list_devices 1 -f dshow -i dummy
rem c:\ffmpeg\ffmpeg -list_options 1 -f dshow -i "audio=Mikrofon (Realtek High Definiti"
rem Live ultrasound converter by mixing (subtraction of the mixing frequency)
set "SR=44100" :: Sample rate of the input soundtrack
set "MF=10000" :: Mixing frequency (this is the frequency to be subtracted)
set "BB=10000" :: Bandwidth
            :: The frequency range [MF ... MF+BB] is converted to the range [0Hz ... BB]
set "VOL=30"  :: Volume factor

c:\ffmpeg\ffmpeg -f dshow -channels 2 -i audio="Mikrofon (Realtek High Definiti" -f lavfi -i aevalsrc="sin (%MF%*2*PI*t):c=stereo:s=%SR%" -filter_complex "][0]volume=%VOL%,highpass=f=%MF%,highpass=f=%MF%,highpass=f= %MF%,highpass=f=%MF%[sound];[sound][1]amultiply,lowpass=f=%BB%,lowpass=f=%BB%,lowpass=f=%BB%,lowpass=f=%BB%" -f nut - | c:\ffmpeg\ffplay -
```

Pinout of 3.5mm stereo connectors: Tip contact is left channel, middle contact is right channel, outer contact is ground.

This is another method for live ultrasound conversion. It's faster and uses the FFT filter:

```plaintext
rem c:\ffmpeg\ffmpeg -list_devices 1 -f dshow -i dummy
rem c:\ffmpeg\ffmpeg -list_options 1 -f dshow -i "audio=Mikrofon (Realtek High Definiti"
rem Live ultrasound converter by mixing (subtraction of the mixing frequency)
set "SR=44100" :: Input sample rate
set "F=4000" :: Subtracted frequency in Hz
set "VOL=30" :: Volume factor
set /a "N=4096*%F%/%SR%" :: N = 4096 * F / SR
```
In this example the delay time between input and output can be minimized by setting the `audio_buffer_size` to a small value, for example with 10ms buffer size the delay is about 0.5 seconds.
3.20  Extract the audio from a video

c:\ffmpeg\ffmpeg -i video.mp4 -vn audio.mp3

3.21  Split a video into audio-only and video-only

Audio and video are saved if individual files.

c:\ffmpeg\ffmpeg -i input.mp4 -vcodec mpeg2video output_video.m2v -acodec copy output_audio.mp3

3.22  Synchronize audio with video

If you have a video with out-of-sync audio, you can synchronize it as follows. In this example a 0.5 seconds delay is added to the audio stream:

c:\ffmpeg\ffmpeg -i input.mp4 -itsoffset 0.5 -i input.mp4 -map 0:0 -map 1:1 -acodec copy -cvodec copy output.mp4

3.23  Sources for royalty-free music

Royalty-free music can be found here for example:  http://opsound.org

But royalty-free doesn't mean you can do what you want with the music. You should read the licence carefully. For example it may be required to give the artist a proper credit, and show a link to the licence in the video.
### Sound effects, from frequency domain to time domain

<table>
<thead>
<tr>
<th>Frequency domain</th>
<th>Time domain</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>freq(t)</td>
<td>y(t)=sin(2<em>pi</em>freq(t)*t)</td>
<td>This formula is wrong. It doesn’t work this way.</td>
</tr>
<tr>
<td>freq(t)</td>
<td>y(t)=sin(2<em>pi</em> [ \int_0^t \text{freq}(t) , dt ])</td>
<td>This is the correct way to calculate the signal in the time domain.</td>
</tr>
<tr>
<td>freq(t)=f0</td>
<td>y(t)=sin(2<em>pi</em>t*f0)</td>
<td>Sine tone with constant frequency</td>
</tr>
<tr>
<td>freq(t)=f0+(f1-f0)*t/p</td>
<td>y(t)=sin(2<em>pi</em>t*(f0+t*(f1-f0)/(2*p)))</td>
<td>Linear chirp from f0 to f1 in p seconds</td>
</tr>
<tr>
<td>freq(t)=f0+f1<em>sin(2</em>pi*t)</td>
<td>y(t)=sin(2<em>pi</em>t<em>f0-f1</em>cos(2<em>pi</em>t))</td>
<td>Sinusoidally rising and falling tone from f0-f1 to f0+f1 with a period of one second</td>
</tr>
<tr>
<td>freq(t)=f0+f1<em>sin(2</em>pi*t/p)</td>
<td>y(t)=sin(2<em>pi</em>t<em>f0-f1</em>p<em>cos(2</em>pi*t/p))</td>
<td>Sinusoidally rising and falling tone with a period of p seconds</td>
</tr>
</tbody>
</table>

Here are a few examples:

```bash
rem Create a sine tone
set "FF=c:\ffmpeg\ffmpeg" :: Path to FFmpeg
set "F0=1000" :: Frequency in Hz
set "T=5" :: Duration in seconds
set "OUT=out.wav" :: Output filename

%FF% -f lavfi -i sine=%F0%:d=%T% -y %OUT%
```

pause
rem Linear chirp from 1kHz to 10kHz in 9 seconds
set "FF=c:\ffmpeg\ffmpeg"  :: Path to FFmpeg
set "F0=1000"               :: Start frequency in Hz
set "F1=10000"              :: End frequency in Hz
set "P=9"                   :: Duration in seconds
set "VOL=0.1"               :: Volume
set "OUT=out.wav"           :: Output filename
%FF% -f lavfi -i aevalsrc='%VOL%*sin(2*PI*t*(%F0%+t*(%F1%-%F0%)/(2*%P%))):c=stereo:s=44100' -t %P% -y %OUT%
pause

rem Linear chirp from 20Hz to 2kHz in 10 seconds
rem This is an approximated rectangular wave consisting of the fundamental wave and three overtones
rem This is also an example for the st() and ld() functions
rem First the phase is calculated and saved in variable "0", then the saved phase is used multiple times
rem to calculate the amplitudes of the fundamental wave and its overtones
set "FF=c:\ffmpeg\ffmpeg"  :: Path to FFmpeg
set "F0=20"                 :: Start frequency in Hz
set "F1=2000"               :: End frequency in Hz
set "P=10"                  :: Duration in seconds
set "VOL=0.2"               :: Volume
set "OUT=out.wav"           :: Output filename
%FF% -f lavfi -i aevalsrc='st(0,2*PI*t*(%F0%+t*(%F1%-%F0%)/(2*%P%)))+%VOL%*(sin(ld(0))+sin(3*ld(0)))/3+sin(5*ld(0)))/5+sin(7*ld(0)))/7):c=stereo:s=48000' -t %P% -y %OUT%
pause
rem Linear chirp from 20Hz to 2kHz in 10 seconds
rem This is a rectangular wave

set "FF=c:\ffmpeg\ffmpeg"   :: Path to FFmpeg
set "F0=20"                 :: Start frequency in Hz
set "F1=2000"               :: End frequency in Hz
set "P=10"                  :: Duration in seconds
set "VOL=0.2"               :: Volume
set "OUT=out.wav"           :: Output filename

%FF% -f lavfi -i aevalsrc='st(0,'2*PI*t*(%F0%+t*(%F1%-%F0%)/(2*%P%))');%VOL%*lt(mod(ld(0),2*PI),PI):c=stereo:s=48000' -t %P% -y %OUT%
pause

rem Linear chirp from 20Hz to 10kHz in 10 seconds
rem This is an approximated triangular wave consisting of the fundamental wave and three overtones

set "FF=c:\ffmpeg\ffmpeg"   :: Path to FFmpeg
set "F0=20"                 :: Start frequency in Hz
set "F1=10000"              :: End frequency in Hz
set "P=10"                  :: Duration in seconds
set "VOL=0.2"               :: Volume
set "OUT=out.wav"           :: Output filename

%FF% -f lavfi -i aevalsrc='st(0,'2*PI*t*(%F0%+t*(%F1%-%F0%)/(2*%P%))');%VOL%*(sin(ld(0))-sin(3*ld(0))/9+sin(5*ld(0))/25-sin(7*ld(0))/49):c=stereo:s=48000' -t %P% -y %OUT%
pause
rem Linear chirp from 20Hz to 2kHz in 10 seconds
rem The waveform is a needle impulse which a width of 1/25 of the period

set "FF=c:\ffmpeg\ffmpeg"   :: Path to FFmpeg
set "F0=20"                 :: Start frequency in Hz
set "F1=2000"               :: End frequency in Hz
set "W=1/25"                :: Width of needle impulses
set "P=10"                  :: Duration in seconds
set "VOL=1"                 :: Volume
set "OUT=out.wav"           :: Output filename

%FF% -f lavfi -i aevalsrc='st(0,'2*PI*t*(%F0%+t*(%F1%-%F0%)/(2*%P%))');%VOL%*lt(mod(1d(0),2*PI),2*PI*%W %):c=stereo:s=96000' -t %P% -y %OUT%
pause

rem Sinusoidally rising and falling tone from 400Hz to 1600Hz with a period of 2 seconds

set "FF=c:\ffmpeg\ffmpeg"   :: Path to FFmpeg
set "F0=1000"               :: Mid frequency in Hz
set "F1=600"                :: Half of frequency sweep in Hz
set "P=2"                   :: Period in seconds
set "T=10"                  :: Duration in seconds
set "VOL=0.1"               :: Volume
set "OUT=out.wav"           :: Output filename

%FF% -f lavfi -i aevalsrc='%VOL%*sin(2*PI*t*%F0%-%F1%*%P%*cos(2*PI*t/%P%)):c=stereo:s=44100' -t %T% -y %OUT%
pause
rem Create band-limited noise from f0 to f1

set "FF=c:\ffmpeg\ffmpeg"  : Path to FFmpeg
set "F0=2000"             : Lower frequency in Hz
set "F1=4000"             : Upper frequency in Hz
set "SR=44100"            : Sample rate in Hz
set "WS=65536"            : Window size for FFT filter, bigger size reduces click noise
set "T=5"                 : Duration in seconds
set "VOL=.5"              : Volume
set "OUT=out.wav"          : Output filename

%FF% -f lavfi -i anoisesrc=r=%SR%:a=%VOL%:d=%T% -af afftfilt='win_size=%WS%:real=re*bitor(between(0.5*sr*b/nb,%F0%,%F1%):imag=im*bitor(between(0.5*sr*b/nb,%F0%,%F1%),' y %OUT%

pause

This example sends the sound directly to FFplay:

rem Create band-limited noise for two bands from f0 to f1 and from f2 to f3

set "FF=c:\ffmpeg\ffmpeg"  : Path to FFmpeg
set "F0=500"              : Lower band start frequency in Hz
set "F1=600"              : Lower band stop frequency in Hz
set "F2=1000"             : Upper band start frequency in Hz
set "F3=1200"             : Upper band stop frequency in Hz
set "SR=44100"            : Sample rate in Hz
set "T=5"                 : Duration in seconds
set "VOL=1"               : Volume

%FF% -f lavfi -i anoisesrc=r=%SR%:a=%VOL%:d=%T% -af afftfilt='win_size=65536:real=re*bitor(between(0.5*sr*b/nb,%F0%,%F1%),between(0.5*sr*b/nb,%F2%,%F3%)):imag=im*bitor(between(0.5*sr*b/nb,%F0%,%F1%),between(0.5*sr*b/nb,%F2%,%F3%))' -f nut - | c:\ffmpeg\ffplay -autoexit -

pause
rem Create band-limited noise for three bands from f0 to f1, from f2 to f3 and from f4 to f5

set "FF=c:\ffmpeg\ffmpeg" :: Path to FFmpeg
set "F0=500" :: Lower band start frequency in Hz
set "F1=550" :: Lower band stop frequency in Hz
set "F2=1000" :: Mid band start frequency in Hz
set "F3=1100" :: Mid band stop frequency in Hz
set "F4=1500" :: Upper band start frequency in Hz
set "F5=1650" :: Upper band stop frequency in Hz
set "SR=44100" :: Sample rate in Hz
set "T=5" :: Duration in seconds
set "VOL=1" :: Volume
set "OUT=out.wav" :: Output filename

%FF% -f lavfi -i anoisesrc=r=%SR%:a=%VOL%:d=%T% -af afftfilt='win_size=65536:real=re*bitor(bitor(bitor(between(0.5*sr*b/nb,%F0%,%F1%)),between(0.5*sr*b/nb,%F2%,%F3%)),between(0.5*sr*b/nb,%F4%,%F5%)):imag=im*bitor(bitor(bitor(between(0.5*sr*b/nb,%F0%,%F1%)),between(0.5*sr*b/nb,%F2%,%F3%)),between(0.5*sr*b/nb,%F4%,%F5%))' -y %OUT%
pause

3.25 Make an audio file with a short test tone

rem Make a 10 seconds audio file with a short 3kHz tone at t=3s

c:\ffmpeg\ffmpeg -f lavfi -i sine=3000:duration=0.1 -af adelays=3000,apad -t 10 -y audio.wav
3.26 Which equipment is useful for making sound records?

I use the following equipment:

- **TASCAM DR-70D Recorder**, has 4 input channels, sample rate 44100, 48000 or 96000 Hz, 16 or 24 Bit
- The successor DR-701D has some improvements: The input amplifiers are somewhat less noisy, the four level controls can be electronically coupled, and the sampling rate can be up to 192kHz.
- **RODE NT1 Microphone** RODE NT1 with fur windshields, very low-noise and excellent for quiet nature sounds
- Microphone cable in 5m length, so that you can stand a few meters away from the microphone when recording. If you stand too close to the microphone, you will have your own noise in the recording. You would hear every movement, every swallowing, every stomach growl...
- **HAMA Joy Powerbank 10400mAh**, as additional power supply for the recorder, because the built-in batteries only allow a very short recording time when the phantom power for the microphones is activated.
3.27  Mathematical properties of sample rates 44100 and 48000

\[ 44100 = 2^2 \times 3^2 \times 5^2 \times 7^2 \quad 48000 = 2^7 \times 3^1 \times 5^3 \]
The greatest common divisor of 44100 and 48000 is 300.

Divisors of 48000:
1, 2, 3, 4, 5, 6, 8, 10, 12, 15, 16, 20, 24, 25, 30, 32, 40, 48, 50, 60, 64, 75, 80, 96, 100, 120, 125, 128, 150, 160, 192, 200, 240, 250, 300, 320, 375, 384, 400, 480, 500, 600, 640, 750, 800, 960, 1000, 1200, 1500, 1600, 1920, 2000, 2400, 3000, 3200, 4000, 4800, 6000, 8000, 9600, 12000, 16000, 24000, 48000

Divisors of 44100:
1, 2, 3, 4, 5, 6, 7, 9, 10, 12, 14, 15, 16, 20, 24, 25, 30, 32, 40, 48, 50, 60, 64, 70, 75, 84, 90, 98, 100, 105, 126, 140, 147, 150, 175, 180, 196, 210, 225, 245, 252, 294, 300, 315, 350, 420, 441, 450, 490, 525, 588, 630, 700, 735, 882, 900, 980, 1050, 1225, 1260, 1470, 1575, 1764, 2100, 2205, 2450, 2940, 3150, 3675, 4410, 4900, 6300, 7350, 8820, 11025, 14700, 22050, 44100

Divisors of 9600:
1, 2, 3, 4, 5, 6, 8, 10, 12, 15, 16, 20, 24, 25, 30, 32, 40, 48, 50, 60, 64, 75, 80, 96, 100, 120, 125, 128, 150, 160, 192, 200, 240, 300, 320, 384, 400, 480, 600, 640, 800, 960, 1200, 1600, 1920, 2400, 3200, 4800, 9600

Divisors of 8820:
1, 2, 3, 4, 5, 6, 7, 9, 10, 12, 14, 15, 18, 20, 21, 25, 28, 30, 35, 36, 42, 45, 49, 50, 60, 63, 70, 75, 84, 90, 98, 100, 105, 126, 140, 147, 150, 175, 180, 196, 210, 225, 245, 252, 294, 300, 315, 350, 420, 441, 450, 490, 525, 588, 630, 700, 735, 882, 900, 980, 1050, 1225, 1260, 1470, 1575, 1764, 2100, 2205, 2450, 2940, 3150, 3675, 4410, 4900, 6300, 7350, 8820, 11025, 14700, 22050, 44100

Divisors of 4800:
1, 2, 3, 4, 5, 6, 8, 10, 12, 15, 16, 20, 24, 25, 30, 32, 40, 48, 50, 60, 64, 75, 80, 96, 100, 120, 125, 128, 150, 160, 192, 200, 240, 300, 320, 400, 480, 600, 800, 960, 1200, 1600, 2400, 4800

Divisors of 4410:
1, 2, 3, 5, 6, 7, 9, 10, 14, 15, 18, 21, 30, 35, 42, 45, 49, 63, 70, 90, 98, 105, 126, 147, 210, 245, 294, 315, 441, 490, 630, 735, 882, 1470, 2205, 4410
3.28 Speed of sound

Speed of sound in air at 20°C: 343.2 m/s

<table>
<thead>
<tr>
<th>Time [ms]</th>
<th>Distance [m]</th>
<th>Samples @ 44100Hz</th>
<th>Samples @ 48000Hz</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.022675</td>
<td>0.00778</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.020833</td>
<td>0.00715</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0.3432</td>
<td>44.1</td>
<td>48</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>0.6864</td>
<td>88.2</td>
<td>96</td>
<td></td>
</tr>
<tr>
<td>2.2675</td>
<td>0.7782</td>
<td>100</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2.9138</td>
<td>1</td>
<td>128.497</td>
<td>139.860</td>
<td></td>
</tr>
<tr>
<td>3.3333</td>
<td>1.144</td>
<td>147</td>
<td>160</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>This is the smallest possible time interval where the number of samples is an integer at sample rate 44100 Hz and also at 48000 Hz</td>
</tr>
<tr>
<td>5</td>
<td>1.716</td>
<td>220.5</td>
<td>240</td>
<td></td>
</tr>
<tr>
<td>5.8275</td>
<td>2</td>
<td>256.699</td>
<td>279.720</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>3.432</td>
<td>441</td>
<td>480</td>
<td>Both sample numbers are integers</td>
</tr>
<tr>
<td>14.569</td>
<td>5</td>
<td>642.483</td>
<td>699.301</td>
<td></td>
</tr>
<tr>
<td>29.138</td>
<td>10</td>
<td>1284.97</td>
<td>1398.60</td>
<td></td>
</tr>
<tr>
<td>100</td>
<td>34.32</td>
<td>4410</td>
<td>4800</td>
<td>Both sample numbers are integers</td>
</tr>
<tr>
<td>1000</td>
<td>343.2</td>
<td>44100</td>
<td>48000</td>
<td>Both sample numbers are integers</td>
</tr>
</tbody>
</table>
4 FFprobe

How to examine a video file with FFprobe without having to write the name of the video into a batch file each time?

It's very simple, just create this batch file once and put it on your desktop:

```
c:\ffmpeg\ffprobe %1
pause
```

Now you can simply drag the video you want to examine with the mouse onto the icon of this batch file, and you will immediately see the result without having pressed a single key. The parameter %1 causes the file name to be passed to FFprobe.

By the way, it's also possible to let FFmpeg examine a file.

To see whether FFmpeg recognizes the file as something:

```
c:\ffmpeg\ffmpeg -i myfile.xxx
pause
```

To see whether FFmpeg can decode the file:

```
c:\ffmpeg\ffmpeg -i myfile.xxx -f null -
pause
```
# 5 FFplay

Keyboard commands while playing:

<table>
<thead>
<tr>
<th>Key</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>q, ESC</td>
<td>Quit</td>
</tr>
<tr>
<td>f, left mouse double-click</td>
<td>Toggle full screen</td>
</tr>
<tr>
<td>p, SPACE</td>
<td>Pause</td>
</tr>
<tr>
<td>s</td>
<td>Step to the next frame. Pause if the stream is not already paused, step to the next video frame, and pause.</td>
</tr>
<tr>
<td>m</td>
<td>Toggle mute.</td>
</tr>
<tr>
<td>9, 0</td>
<td>Decrease and increase volume respectively.</td>
</tr>
<tr>
<td>/, *</td>
<td>Decrease and increase volume respectively.</td>
</tr>
<tr>
<td>a</td>
<td>Cycle audio channel in the current program.</td>
</tr>
<tr>
<td>v</td>
<td>Cycle video channel.</td>
</tr>
<tr>
<td>t</td>
<td>Cycle subtitle channel in the current program.</td>
</tr>
<tr>
<td>c</td>
<td>Cycle program.</td>
</tr>
<tr>
<td>w</td>
<td>Cycle video filters or show modes.</td>
</tr>
<tr>
<td>left/right</td>
<td>Seek backward/forward 10 seconds.</td>
</tr>
<tr>
<td>down/up</td>
<td>Seek backward/forward 1 minute.</td>
</tr>
<tr>
<td>page down/page up</td>
<td>Seek to the previous/next chapter, or if there are no chapters seek backward/forward 10 minutes.</td>
</tr>
<tr>
<td>right mouse click</td>
<td>Seek to percentage in file corresponding to fraction of width.</td>
</tr>
</tbody>
</table>

This is a batch file that you can put on your desktop, and then play a video simply by drag-and-drop:

c:\ffmpeg\ffplay %1 -autoexit
This is a batch file for playing audio files by drag-and-drop (without video output):
```
c:\ffmpeg\ffplay %1 -nodisp -autoexit
```

List of the most important FFplay options:

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-s size</td>
<td>Set frame size (WxH or abbreviation)</td>
</tr>
<tr>
<td>-fs</td>
<td>Start in fullscreen mode</td>
</tr>
<tr>
<td>-an</td>
<td>Disable audio</td>
</tr>
<tr>
<td>-vn</td>
<td>Disable video</td>
</tr>
<tr>
<td>-ss pos</td>
<td>Seek to pos</td>
</tr>
<tr>
<td>-nodisp</td>
<td>Disable graphical display</td>
</tr>
<tr>
<td>-noborder</td>
<td>Borderless window</td>
</tr>
<tr>
<td>-alwaysontop</td>
<td>Window always on top</td>
</tr>
<tr>
<td>-f fmt</td>
<td>Force format</td>
</tr>
<tr>
<td>-loop number</td>
<td>Loops movie playback (&lt;number&gt;) times. 0 means forever</td>
</tr>
<tr>
<td>-vf filtergraph</td>
<td>Create the filtergraph specified by (filtergraph) and use it to filter the video stream</td>
</tr>
<tr>
<td>-af filtergraph</td>
<td>(filtergraph) is a description of the filtergraph to apply to the input audio</td>
</tr>
<tr>
<td>-autoexit</td>
<td>Exit when video is done playing</td>
</tr>
<tr>
<td>-exitonkeydown</td>
<td>Exit if any key is pressed</td>
</tr>
<tr>
<td>-exitonmousedown</td>
<td>Exit if any mouse button is pressed</td>
</tr>
</tbody>
</table>
6 Exiftool

With this tool you can show all exif data that are contained in pictures or videos.

https://www.sno.phy.queensu.ca/~phil/exiftool/

Usage is very simple if you create this batch file once and put it on your desktop:

```
c:\ffmpeg\exiftool %1
pause
```

Now you can simply drag the picture or video you want to examine with the mouse onto the icon of this batch file, and you will immediately see the result without having pressed a single key. The parameter %1 causes the file name to be passed to Exiftool.

Exiftool can be combined with the batch command "findstr", if you want to filter only a few lines from the large Exiftool output:

```
@echo off
c:\ffmpeg\exiftool %1 | findstr /C:"File Name" /C:"File Size" /C:"Duration" /C:"Image Width" /C:"Image Height" /C:"Video Frame Rate" /C:"Exposure Time" /C:"F Number" /C:"Exposure Program" /C:"ISO" /C:"Photo Style" /B /C:"Noise Reduction" /C:"Contrast " /C:"Saturation" /C:"Sharpness" /C:"Avg Bitrate" /C:"Track Create Date"
pause
```

"findstr" is in detail explained here: https://ss64.com/nt/findstr.html
7 Batch files (Windows 7)

Some useful links for writing batch files:
https://en.wikibooks.org/wiki/Windows_Batch_Scripting (english)
https://de.wikibooks.org/wiki/Batch-Programmierung/_Druckversion (german)

7.1 Wildcards in filenames

* any sequence of one or more characters
? a single character other than a period "."

When a command-line argument contains a filename, a special syntax can be used to get various information about this file:

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Result</th>
<th>Example for F:\Meteors_2019\CUT00380.MOV</th>
</tr>
</thead>
<tbody>
<tr>
<td>%1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>%~1</td>
<td>%1 with no enclosing quotation marks</td>
<td>CUT00380.MOV</td>
</tr>
<tr>
<td>%~f1</td>
<td>Full path with a drive letter</td>
<td>F:\Meteors_2019\CUT00380.MOV</td>
</tr>
<tr>
<td>%~d1</td>
<td>Drive letter</td>
<td>F:</td>
</tr>
<tr>
<td>%~p1</td>
<td>Drive-less path with the trailing backslash</td>
<td>\Meteors_2019\</td>
</tr>
<tr>
<td>%~n1</td>
<td>For a file, the file name without path and extension</td>
<td>Meteors_2019\</td>
</tr>
<tr>
<td>%~x1</td>
<td>File name extension including the period</td>
<td>.MOV</td>
</tr>
</tbody>
</table>

The same syntax applies to single-letter variables created by the FOR command.

Change the extension of a filename in a batch file:

```
set OLDFILENAME=%1
set NEWFILENAME=%OLDFILENAME:MOV=MP4%
```
Please note that all instances of "MOV" will be replaced by "MP4". This fails if "MOV" is part of the path or filename, as in "MOVEMENT.MOV"

### 7.2 Create beeps in a batch file

```batch
@echo #
@timeout 1
@echo #
@timeout 1
@echo #
```

In this example the # stands for the non-printable character (ASCII code 7), which you can't enter with Notepad. You can type any other character instead and later use a hex editor to replace it by 0x07.

Another way for creating the ASCII 7 is to type this command line at the command prompt:

```cmd
echo @echo ^G>test33.bat
```

where ^G means typing CTRL G

This is an endless loop for beeping every 10 seconds, without any output on the screen (except a line feed):

```batch
:beep
@echo #
@timeout 10 > nul
@goto :beep
```
Loop over all files in a directory

```
for %%f in (img*.jpg) do call :for_body %%f
goto :the_end

:for_body
  c:\ffmpeg\ffmpeg -i %1 -y %~n1.png
exit /b

:the_end
pause
```
https://www.videolan.org/vlc/

My notebook doesn't have enough computing power for playing 4K videos (400Mbit/s from Panasonic GH5S) smoothly with VLC player. This batch file reduces the size to 50% and then plays the video.

Simply drag and drop the video on the batch file's icon. The parameter %1 causes the file name to be passed to FFmpeg.

In the second line the path must be written in quotation marks because there is a space character in "Program Files".

<table>
<thead>
<tr>
<th>Key</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>F</td>
<td>Toggle fullscreen</td>
</tr>
<tr>
<td>ESC</td>
<td>Leave fullscreen/close dialogue</td>
</tr>
<tr>
<td>space</td>
<td>Play/pause</td>
</tr>
<tr>
<td>E</td>
<td>Next frame</td>
</tr>
<tr>
<td>+</td>
<td>Faster</td>
</tr>
<tr>
<td>-</td>
<td>Slower</td>
</tr>
<tr>
<td>=</td>
<td>Normal rate</td>
</tr>
<tr>
<td>]</td>
<td>Faster (fine)</td>
</tr>
<tr>
<td>[</td>
<td>Slower (fine)</td>
</tr>
<tr>
<td>S</td>
<td>Stop</td>
</tr>
<tr>
<td>T</td>
<td>Position/time</td>
</tr>
<tr>
<td>Ctrl + Q</td>
<td>Quit</td>
</tr>
<tr>
<td>M</td>
<td>Mute</td>
</tr>
<tr>
<td>V</td>
<td>Cycle subtitle track</td>
</tr>
<tr>
<td>Shift - V</td>
<td>Toggle subtitles</td>
</tr>
</tbody>
</table>
Color grading with 3D LUT Creator

3D LUT Creator is a software for color grading. A free demo version is available, and the full version costs $249 (July 2019). Sometimes there seems to be 25% discount.

Website: [https://3dlutcreator.com/](https://3dlutcreator.com/)

Drawback of this software: The written manual is totally outdated, and for the latest version you have to watch many video tutorials (what I don’t like so much).

All video tutorials can be found here: [https://3dlutcreator.com/3d-lut-creator---tutorials.html](https://3dlutcreator.com/3d-lut-creator---tutorials.html)

A guide for all hotkeys in 3D LUT Creator:
[https://medium.com/@alexeyadamitsky/3d-lut-creator-ultimate-hotkeys-guide-17a32f957077](https://medium.com/@alexeyadamitsky/3d-lut-creator-ultimate-hotkeys-guide-17a32f957077)

This is a very powerful software for creating and editing color-look-up-tables. It’s also possible to match the colors to a ColorChecker. A ColorChecker is a card with 24 different colors, which is hold in the camera’s field of view.

Original X-Rite ColorChecker:
[https://www.xrite.com/categories/calibration-profiling/colorchecker-targets](https://www.xrite.com/categories/calibration-profiling/colorchecker-targets)

There are also cheap ($20-$25) chinese ColorCheckers available. Their colors may be a little bit different from the original.

Hotkeys:
CTRL N  New Preset (Reset All)
CTRL O  Load an image
CTRL E  Save the LUT
CTRL +  Zoom in
CTRL -  Zoom out
I'd like to describe the workflow for making a video with a ColorChecker, and how to create a LUT and apply that LUT to the video.

Step 1: Make a video where at some time the ColorChecker is visible, preferably at the beginning of the video. It doesn't care which file format you use, as FFmpeg can decode almost all of them.

Step 2: Open the video in a viewer (for example VLC player) and check at which time the ColorChecker is visible.

Step 3: Use this batch file to extract a single picture from the video and save it lossless as PNG. Of course, you must enter your filename and the correct time in the batch file. The picture will be automatically 8-bit or 16-bit PNG, depending on the bit depth of the input video, so that there is no loss of quality.

```batch
set "FF=c:\ffmpeg\ffmpeg"    :: Path to ffmpeg.exe
set "IN=my_video.mov"        :: Input video
set "T=3"                    :: Time in seconds, where picture shall be extracted

%FF% -ss %T% -i %IN% -frames 1 -y picture.png
pause
```

In the previous example I did use variables, because they make the batch file more readable. The following batch file is doing exactly the same thing:

```
c:\ffmpeg\ffmpeg -ss 3 -i my_video.mov -frames 1 -y picture.png
pause
```

The -y option means that the output file will be overwritten if it already exists (without asking). Without the -y option, FFmpeg would ask before overwriting.

The pause command means that you have to press a button before the window closes. Without this command the window would close immediately when FFmpeg is finished, making is impossible to see if there were any error messages.

Step 4: Drag and drop this picture into 3D LUT Creator (or alternatively press CTRL+O). In the field to the left of the "Match" button select "Curves+3DLUT". Click on the "Color Chart Grid Tool" icon (this is a small rectangle with 6 dots in it). Move the corners of the grid tool with the mouse to the corresponding ColorChecker fields in the picture. Then click on "Match". After a few seconds the picture should be shown with all colors matched to the ColorChecker. Click on "Save 3DLUT" in the bottom left corner (or alternatively press CTRL+E) to save the CLUT as my_lut.cube
Step 5: Use this batch file to apply the LUT to your video:

```plaintext
set "FF=c:\ffmpeg\ffmpeg"    :: Path to ffmpeg.exe
set "IN=my_video.mov"        :: Input video
set "LUT=my_lut.cube"        :: Look-Up-Table
set "OUT=out.mov"            :: Output video

%FF% -i %IN% -vf lut3d="%LUT%" -y %OUT%

pause
```

If you want to see only a few seconds at the beginning of the video, you can limit the length with the -t parameter.

```plaintext
set "FF=c:\ffmpeg\ffmpeg"    :: Path to ffmpeg.exe
set "IN=my_video.mov"        :: Input video
set "LUT=my_lut.cube"        :: Look-Up-Table
set "OUT=out.mov"            :: Output video
set "T=10"                   :: Length of output video

%FF% -i %IN% -vf lut3d="%LUT%" -t %T% -y %OUT%

pause
```

You can also show the video simultaneously before and after applying the LUT. The scale filter is used to reduce the size to 50%, and the hstack filter is used to combine two videos side by side.

```plaintext
set "FF=c:\ffmpeg\ffmpeg"    :: Path to ffmpeg.exe
set "IN=my_video.mov"        :: Input video
set "LUT=my_lut.cube"        :: Look-Up-Table
set "OUT=out.mov"            :: Output video
set "T=10"                   :: Length of output video

%FF% -i %IN% -filter_complex scale=iw/2:ih/2,split[a][b];[b]lut3d="%LUT%"[c];[a][c]hstack -t %T% -y %OUT%

pause
```
Here is a converter for different types of color-look-up-tables:

This software can also fit colors to a ColorChecker:
https://www.xrite.com/service-support/downloads/c/colorchecker_camera_calibration_v1_1_1

But it has two severe drawbacks:
1. It requires a DNG image as input. That's no problem for photography, but there's no way to extract a DNG from a video.
2. The output of this software is a camera profile in Adobe *.dcp format and I don't know how this can be converted into a CLUT for FFmpeg.
9.1 Beginner tutorials for 3D LUT Creator

1. What is the LUT?  
   https://www.youtube.com/watch?time_continue=2&v=3ZpbUOGDWLE
   This video explains 1D and 3D LUTs. 3D LUT Creator can save LUTs in many different formats, including *.cube and *.png

2. Working with LUTs in Photoshop and 3D LUT Creator  
   https://www.youtube.com/watch?time_continue=7&v=K0t-HSO-OUU

3. Working with Lightroom and 3D LUT Creator  
   https://www.youtube.com/watch?v=o968FH1kV3w

4. Working with the Image window  
   https://www.youtube.com/watch?v=TmZAITXStfU
   The working image can be loaded by drag and drop.
   For an optional reference image use "File / Load Reference Image".
   In the video he says you can toggle between working image and reference image by pressing the = key, but that's wrong. It's the + key.
   CTRL + Zoom in   CTRL - Zoom out
   For moving the image in the window, press the mouse wheel and move the mouse.++
   "Image / Assign Color Profile" doesn't change the image, it changes only the way how the image is displayed.
   "Image / Convert to Profile" does change the image.
   Toggle the compare mode by pressing the c key.
   By pressing the x key the image is split in the middle, one half is before and the other is after.
   By pressing the v key you can toggle between horizontal and vertical splitting.

5. Look manager  
   https://www.youtube.com/watch?v=dY_6MeE-gAg
   Window / Look Manager
   Open a folder to see all presets applied to your image.
   The size of the images can be changed with the + and - buttons in the top left corner.

6. Working principle of A/B and C/L color grids  
   https://www.youtube.com/watch?v=AlSYkjdDdq
   In the A/B grid we can change only hue (position angle) and saturation (radial distance from center). Lightness stays unchanged.
In the C/L grids we can change only saturation (left - right, neutral in center) and lightness (vertical).

7. HSP and LAB color models  [https://www.youtube.com/watch?v=mJfEgvheWeM](https://www.youtube.com/watch?v=mJfEgvheWeM)
   In this video he explains the difference between the different HSP color models, and which model to use for which purpose.

8. LXY, MXY, MABe, MXYe, SXY, SXY, CMYK and RGBW color models  [https://www.youtube.com/watch?v=7uC1vtS1BnU](https://www.youtube.com/watch?v=7uC1vtS1BnU)

9. The Luminance curve and the Brightness slider  [https://www.youtube.com/watch?v=BBjY3ivCjPg](https://www.youtube.com/watch?v=BBjY3ivCjPg)

10. Saturation curves  [https://www.youtube.com/watch?v=TnUp3Ds_b_DU](https://www.youtube.com/watch?v=TnUp3Ds_b_DU)

11. Basics of working with the A/B color grid  [https://www.youtube.com/watch?v=35EoR_c4D9w](https://www.youtube.com/watch?v=35EoR_c4D9w)

12. Practice with A/B color grid, part 1  [https://www.youtube.com/watch?v=BYe_V0UF5os](https://www.youtube.com/watch?v=BYe_V0UF5os)

13. Practice with A/B color grid, part 2  [https://www.youtube.com/watch?v=dR4pjHRpU0Y](https://www.youtube.com/watch?v=dR4pjHRpU0Y)

14. Tools for working with the A/B color grid  [https://www.youtube.com/watch?v=etIX_e8-_lk](https://www.youtube.com/watch?v=etIX_e8-_lk)

15. Batch processing in 3D LUT Creator  [https://www.youtube.com/watch?v=1wv1NqXywY](https://www.youtube.com/watch?v=1wv1NqXywY)

9.2 Advanced tutorials for 3D LUT Creator

1. Color Match with the Reference image  [https://youtu.be/k0YQNm7TINM](https://youtu.be/k0YQNm7TINM)

2. How to create 3D LUT files from Lightroom presets or Photoshop Plugins  [https://youtu.be/MIoEciU1ISU](https://youtu.be/MIoEciU1ISU)

3. RAW photo developing with 3D LUT Creator  [https://youtu.be/3Sm120XC37Q](https://youtu.be/3Sm120XC37Q)


6. How to change third party LUTs in 3D LUT Creator  [https://youtu.be/Lx6ppOmx9kCy](https://youtu.be/Lx6ppOmx9kCy)


9. New way of color toning in 3D LUT Creator  [https://youtu.be/xpoU3ZqIgLE](https://youtu.be/xpoU3ZqIgLE)
10. Color correction in game production with 3D LUT Creator & Unity  https://youtu.be/pzJXtyseARo
11. Skin tone color correction by numbers with RGB curves https://youtu.be/NYzJXdpJDPU
12. Adjusting the Skin Tone using color match tool https://youtu.be/rgVFTuu9Kls
13. Color Masks https://youtu.be/rQHooXewsN0

9.3 Working with Color Targets in for 3D LUT Creator

1. Color correction with Color Checkers in 3D LUT Creator, part 1 https://youtu.be/mZvrj8__5r0
2. Color correction with Color Checkers in 3D LUT Creator, part 2 https://youtu.be/0UALWETt1q4
3. Working with ChromaDuMonde in 3D LUT Creator and Davinci Resolve https://youtu.be/5oCS4WqPKB8

9.4 Working with video in for 3D LUT Creator

1. Working with LOG video footage in 3D LUT Creator https://youtu.be/jX3i34wFsG0
2. Using 3D LUT Creator with Davinci Resolve & Red Camera Footage https://youtu.be/4e4OrN60_wc
3. Working with ChromaDuMonde in 3D LUT Creator and Davinci Resolve https://youtu.be/5oCS4WqPKB8
This is the custom ColorChecker file for the cheap Chinese ColorChecker, using the RGB values printed on the back side and converted to LAB. Save this file as "MyColorChecker.txt". The differences to the original X-Rite ColorChecker seem to be quite small.

<table>
<thead>
<tr>
<th>SampleID</th>
<th>SAMPLE_NAME</th>
<th>RGB_R</th>
<th>RGB_G</th>
<th>RGB_B</th>
<th>LAB_L</th>
<th>LAB_A</th>
<th>LAB_B</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>A1</td>
<td>0.45</td>
<td>0.32</td>
<td>0.27</td>
<td>38.24</td>
<td>12.86</td>
<td>13.38</td>
</tr>
<tr>
<td>2</td>
<td>A2</td>
<td>0.80</td>
<td>0.63</td>
<td>0.55</td>
<td>69.89</td>
<td>14.32</td>
<td>16.84</td>
</tr>
<tr>
<td>3</td>
<td>A3</td>
<td>0.40</td>
<td>0.53</td>
<td>0.70</td>
<td>54.78</td>
<td>-2.82</td>
<td>-27.79</td>
</tr>
<tr>
<td>4</td>
<td>A4</td>
<td>0.35</td>
<td>0.43</td>
<td>0.24</td>
<td>43.46</td>
<td>-14.44</td>
<td>24.19</td>
</tr>
<tr>
<td>5</td>
<td>A5</td>
<td>0.55</td>
<td>0.54</td>
<td>0.76</td>
<td>59.09</td>
<td>11.19</td>
<td>-29.35</td>
</tr>
<tr>
<td>6</td>
<td>A6</td>
<td>0.52</td>
<td>0.89</td>
<td>0.82</td>
<td>84.26</td>
<td>-33.27</td>
<td>0.42</td>
</tr>
<tr>
<td>7</td>
<td>B1</td>
<td>0.98</td>
<td>0.46</td>
<td>0.14</td>
<td>65.10</td>
<td>48.17</td>
<td>65.24</td>
</tr>
<tr>
<td>8</td>
<td>B2</td>
<td>0.31</td>
<td>0.36</td>
<td>0.71</td>
<td>41.41</td>
<td>16.72</td>
<td>-50.62</td>
</tr>
<tr>
<td>9</td>
<td>B3</td>
<td>0.87</td>
<td>0.36</td>
<td>0.49</td>
<td>57.07</td>
<td>54.23</td>
<td>8.61</td>
</tr>
<tr>
<td>10</td>
<td>B4</td>
<td>0.36</td>
<td>0.25</td>
<td>0.48</td>
<td>32.01</td>
<td>22.42</td>
<td>-29.94</td>
</tr>
<tr>
<td>11</td>
<td>B5</td>
<td>0.68</td>
<td>0.91</td>
<td>0.36</td>
<td>85.90</td>
<td>-35.40</td>
<td>60.19</td>
</tr>
<tr>
<td>12</td>
<td>B6</td>
<td>1.00</td>
<td>0.64</td>
<td>0.10</td>
<td>75.41</td>
<td>28.21</td>
<td>75.36</td>
</tr>
<tr>
<td>13</td>
<td>C1</td>
<td>0.17</td>
<td>0.22</td>
<td>0.56</td>
<td>26.83</td>
<td>18.77</td>
<td>-50.29</td>
</tr>
<tr>
<td>14</td>
<td>C2</td>
<td>0.29</td>
<td>0.58</td>
<td>0.32</td>
<td>55.33</td>
<td>-35.21</td>
<td>27.97</td>
</tr>
<tr>
<td>15</td>
<td>C3</td>
<td>0.70</td>
<td>0.16</td>
<td>0.20</td>
<td>41.03</td>
<td>55.03</td>
<td>31.12</td>
</tr>
<tr>
<td>16</td>
<td>C4</td>
<td>0.98</td>
<td>0.89</td>
<td>0.08</td>
<td>89.78</td>
<td>-4.29</td>
<td>85.94</td>
</tr>
<tr>
<td>17</td>
<td>C5</td>
<td>0.75</td>
<td>0.32</td>
<td>0.63</td>
<td>51.41</td>
<td>51.68</td>
<td>-20.96</td>
</tr>
<tr>
<td>18</td>
<td>C6</td>
<td>0.02</td>
<td>0.56</td>
<td>0.67</td>
<td>53.96</td>
<td>-24.51</td>
<td>-25.37</td>
</tr>
<tr>
<td>19</td>
<td>D1</td>
<td>0.99</td>
<td>0.99</td>
<td>0.99</td>
<td>98.96</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>20</td>
<td>D2</td>
<td>0.90</td>
<td>0.90</td>
<td>0.90</td>
<td>91.29</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>21</td>
<td>D3</td>
<td>0.78</td>
<td>0.78</td>
<td>0.78</td>
<td>80.60</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>22</td>
<td>D4</td>
<td>0.56</td>
<td>0.56</td>
<td>0.56</td>
<td>59.38</td>
<td>-0.14</td>
<td>0.53</td>
</tr>
<tr>
<td>23</td>
<td>D5</td>
<td>0.39</td>
<td>0.39</td>
<td>0.39</td>
<td>42.37</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>24</td>
<td>D6</td>
<td>0.20</td>
<td>0.20</td>
<td>0.20</td>
<td>20.79</td>
<td>0.00</td>
<td>0.00</td>
</tr>
</tbody>
</table>

Color converter for different color spaces:

https://www.nixsensor.com/free-color-converter/
The software looks very promising, but it has so many functions and I don't know where to begin.

https://www.blackmagicdesign.com/de/products/davinciresolve/

I got this book: Paul Saccone, Dion Scoppettuolo: "Der ultimative Leitfaden zu DaVinci Resolve 15" (I got the german translation, but it's also available in english).

Please note that this book is for version 15. Version 16 seems to have a different user interface, so for learning with this book it's better to use the older version 15. So far (November 2019), there is no book for version 16. The official manual for version 16 is extremely long, more than 3000 pages.

It's also possible to match an audio track to another audio track that was recorded with the camera, see page 339.
Manually syncing is described on page 340.

Found this somewhere on Facebook:
The free DaVinci Resolve 15 version can't import 4K 4:2:2 10 bit videos from the Panasonic GH5S camera. But this FFmpeg conversion does the job:
ffmpeg -i "P1000975.MOV" -map_metadata 0 -pix_fmt yuv422p10le -c:v dnxhd -profile:v 4 -c:a pcm_s24le -color_range pc -movflags write_colr "out.mov"
Mouse buttons and keyboard shortcuts:

<table>
<thead>
<tr>
<th>Mouse Button</th>
<th>Keyboard</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt; · &gt;</td>
<td></td>
<td>(seems to be the same as moving the position with the mouse)</td>
</tr>
<tr>
<td>⏯</td>
<td>J</td>
<td>Jump to the start</td>
</tr>
<tr>
<td>⏯</td>
<td>J+K</td>
<td>Play backward with half speed</td>
</tr>
<tr>
<td>Hold K and press J once, or press arrow left</td>
<td>One frame backward</td>
<td></td>
</tr>
<tr>
<td>■</td>
<td>K</td>
<td>Stop</td>
</tr>
<tr>
<td>Hold K and press L once, or press arrow right</td>
<td>One frame forward</td>
<td></td>
</tr>
<tr>
<td>K+L</td>
<td></td>
<td>Play forward with half speed</td>
</tr>
<tr>
<td>▶</td>
<td>L</td>
<td>Play forward</td>
</tr>
<tr>
<td>press L twice</td>
<td></td>
<td>Play with double speed</td>
</tr>
<tr>
<td>▶▶</td>
<td></td>
<td>Jump to the end</td>
</tr>
<tr>
<td>▶▶</td>
<td></td>
<td>Endless loop mode</td>
</tr>
<tr>
<td>Space</td>
<td></td>
<td>Play / Pause</td>
</tr>
<tr>
<td>■</td>
<td>I</td>
<td>Set the &quot;In&quot; marker</td>
</tr>
<tr>
<td>◀</td>
<td>O</td>
<td>Set the &quot;Out&quot; marker</td>
</tr>
<tr>
<td>Q</td>
<td></td>
<td>Toggle between source and timeline viewer</td>
</tr>
<tr>
<td>F9</td>
<td></td>
<td>Insert</td>
</tr>
<tr>
<td>F10</td>
<td></td>
<td>Overwrite</td>
</tr>
<tr>
<td>F11</td>
<td></td>
<td>Replace</td>
</tr>
<tr>
<td>F12</td>
<td></td>
<td>Place on top</td>
</tr>
<tr>
<td>T</td>
<td></td>
<td>Trim edit mode</td>
</tr>
<tr>
<td>W</td>
<td></td>
<td>Dynamic trim mode (slip)</td>
</tr>
</tbody>
</table>
10.1 Recording a Voice-Over audio track

This is also explained in german in the chapter "Audio in einer Timeline aufzeichnen" in: Paul Saccone, Dion Scoppettuolo: Der Ultimative Leitfaden zu DAVINCI RESOLVE 15

- Let's assume you are using the built-in microphone in your laptop, or the audio-in connector which is connected to the same A/D converter.
- Open a project, click on File --> "Project Settings", then click on "Capture and Playback", then select in the field "Save Clips to" the folder where you want to save the new audio tracks. Close the project settings with "Save".
- Hide the level meter and show the mixer (if it's not already visible).
- Make a right click on any track headline and choose "Add Track" --> "Mono". Or "Stereo", if required. But normally a voice over is recorded in mono.
- Double click the headline of the new audio track and change the name to "VO" for voice over.
- You can also change the track color if you want.
- The following things are only possible if you switch to the "Fairlight" room, that's the ♫ icon.
- The new track is also shown in the mixer, and in the row "Input" it's marked as "No Input". Click on this field, select "Input..." and then a "Patch Input/Output" window will appear. At the left side you can select the microphone and at the right side you select the VO track. Click on "Patch" and close this window.
- All audio tracks have a "R" icon which can be activated for recording. Do this for the "VO" track. Most probably you will now hear an acoustic feedback loop. This is because the sound from the speakers is coupled back to the microphone. To avoid this acoustic feedback, either set the level for the "Main1" output to the lowest possible level (all way down), or simply activate the "Mute" icon for the "Main1" output (this is the "M" icon).
- Set the "VO" track to "Solo" by activating the "S" icon.
- Now you can start a record by clicking on the ● icon in the timeline (which is only available in the "Fairlight" room).
DaVinci Resolve is designed for Windows10. There are some problems to be expected when you run it on Windows7.

One problem is that recording sound from USB soundcards is impossible or difficult. Although the USB soundcard is shown in the "Patch Input/Output" window and a patch can be created, there comes no signal from this soundcard.

I got it working with this workaround: Go to Davinci Resolve --> Preferences --> Video and Audio I/O --> Speaker Setup and make these settings:

  - Speaker Configuration: Manual
  - Monitor Set: MAIN
  - Device: Lautsprecher (Realtek High Definition Audio)
  - Monitor Set Format: Stereo

With these settings, I got the USB soundcard working. But it seems my Windows7 computer is too slow and can't record a voice-over track in real time without disturbing artefacts.
11 Tips and tricks for video

• Learn by analyzing other videos and films
• Use a variable neutral density filter, so that you can use wide open aperture for narrow depth of field
• Always record 3 seconds before the action begins and also 3 seconds after action has ended.
• Use a good tripod for video. A recommended manufacturer is Sachtler.
• Interviewing two people: Record one from the left and the other from the right.
• Know your camera before you begin to record videos.
12 LINUX and GIT

Some experiments with Linux...

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$ apt-get install ffmpeg</td>
<td>Install FFmpeg</td>
</tr>
<tr>
<td>$ git clone <a href="http://git.ffmpeg.org/ffmpeg.git">http://git.ffmpeg.org/ffmpeg.git</a></td>
<td>Download the FFmpeg source code. Unfortunately the source code is very badly commented (at least those files that I had a look at).</td>
</tr>
<tr>
<td>$ ./configure</td>
<td>This is the procedure for building FFmpeg. Go to the FFmpeg folder and type these three command lines. For detailed instructions see <a href="https://trac.ffmpeg.org/wiki/CompilationGuide/Ubuntu">https://trac.ffmpeg.org/wiki/CompilationGuide/Ubuntu</a></td>
</tr>
<tr>
<td>$ sudo make</td>
<td>Install VLC player (or any other program)</td>
</tr>
<tr>
<td>$ chmod +x my_script</td>
<td>Makes a script file executable. In Linux script files have no extension. The first line in the script file must contain: #!/bin/bash</td>
</tr>
<tr>
<td>$ cd ..</td>
<td>Go back to the parent folder. Please note there must be a space character between cd and the two dots.</td>
</tr>
</tbody>
</table>

How to change something in the FFmpeg documentation and work with GIT, this example is from Carl Eugen Hoyos 27.9.19 in the FFmpeg user mailing list. The example works but I don’t yet understand how git works. Very complicated.

$ git clone http://git.ffmpeg.org/ffmpeg.git
$ cd ffmpeg
edit a file in the doc directory.
$ git commit doc
(I suspect this will ask you to set your name and email when running it for the first time)
$ git format-patch HEAD^ This produces a file that you can send to the mailing list after visual inspection for commit message and your name.
$ git reset HEAD^
## Cameras and lenses for fulldome video production

<table>
<thead>
<tr>
<th></th>
<th>Canon 6D</th>
<th>Panasonic LUMIX GH5S</th>
<th>PanoView XDV360</th>
<th>Kodak SP360_4K</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Cameras and lenses</strong></td>
<td><img src="image" alt="Canon 6D" /></td>
<td><img src="image" alt="Panasonic LUMIX GH5S" /></td>
<td><img src="image" alt="PanoView XDV360" /></td>
<td><img src="image" alt="Kodak SP360_4K" /></td>
</tr>
<tr>
<td><strong>Fulldome resolution</strong></td>
<td>180°: 3648 x 3648 (Pictures)</td>
<td>180°: 2880 x 2880 (Pictures)</td>
<td>220°: 2448 x 2448</td>
<td>235°: 2880 x 2880</td>
</tr>
<tr>
<td></td>
<td>180°: 1080 x 1080 (Video)</td>
<td>180°: 2496 x 2496 (Video)</td>
<td>180°: 2104 x 2104</td>
<td>180°: 2456 x 2456</td>
</tr>
<tr>
<td><strong>Sound recording</strong></td>
<td>stereo 48000 Hz, but both channels are identical, if no external microphone is connected</td>
<td>stereo 48000 Hz, but both channels are identical, if no external microphone is connected</td>
<td>mono 8000 Hz, there is no connector for an external microphone</td>
<td>stereo 48000 Hz, but both channels are almost equal because the microphones are close together; no connector for external microphones</td>
</tr>
<tr>
<td><strong>Suitable for fulldome video?</strong></td>
<td>yes, if a fisheye lens is used which has a 180° image diameter less than 20.2mm</td>
<td>yes, if a fisheye lens is used which has a 180° image diameter less than 13.0mm</td>
<td>yes</td>
<td>yes</td>
</tr>
<tr>
<td><strong>Suitable for fulldome video at night?</strong></td>
<td>yes</td>
<td>yes, very good</td>
<td>no, too much noise</td>
<td>no, too much noise</td>
</tr>
<tr>
<td><strong>Suitable for fulldome timelapse?</strong></td>
<td>yes, arbitrary interval times with external timer</td>
<td>yes, arbitrary interval times with external timer</td>
<td>yes, with internal timer</td>
<td>yes, with internal timer</td>
</tr>
</tbody>
</table>
### 13.1 Read-out chip size of cameras at different video modes

Problem: A full format chip has the size 36mm x 24mm and thus the format 3:2. For video recording, however, the format 16:9 is used, so that only a part with the dimensions 36mm x 20.25mm is read out. But as a full format fisheye normally illuminates a 24mm diameter circle, there are two strips missing at the top and bottom of the video.

If the entire image circle of the fisheye lens is to be recorded in the video, the image circle diameter of the lens must not be larger than the read-out height of the chip at the set video resolution.

<table>
<thead>
<tr>
<th>Camera</th>
<th>Chip Size</th>
<th>Pixels</th>
<th>Video Resolution</th>
<th>Read-out Part of the Chip, Width x Height</th>
</tr>
</thead>
<tbody>
<tr>
<td>Canon 6D</td>
<td>35.8mm x 23.9mm</td>
<td>5472 x 3648</td>
<td>640 x 480 (4:3)</td>
<td>31.87mm x 23.9mm</td>
</tr>
<tr>
<td>Canon 6D</td>
<td>35.8mm x 23.9mm</td>
<td>5472 x 3648</td>
<td>1920 x 1080 Full HD (16:9)</td>
<td>35.9mm x 20.19mm</td>
</tr>
<tr>
<td>Canon 5D MK4</td>
<td>36mm x 24mm</td>
<td>6720 x 4480</td>
<td>1920 x 1080 Full HD (16:9)</td>
<td>36mm x 20.25mm</td>
</tr>
<tr>
<td>Canon 5D MK4</td>
<td>36mm x 24mm</td>
<td>6720 x 4480</td>
<td>4096 x 2160 C4K (17:9)</td>
<td>21.94mm x 11.57mm (Not the whole chip width is used)</td>
</tr>
<tr>
<td>Canon 7D</td>
<td>22.3mm x 14.9mm</td>
<td>5184 x 3456</td>
<td>1920 x 1080 Full HD</td>
<td>22.30mm x 12.54mm</td>
</tr>
<tr>
<td>Canon EOS R</td>
<td>36mm x 24mm</td>
<td>6720 x 4480</td>
<td>1920 x 1080 Full HD (16:9)</td>
<td>36mm x 20.25mm</td>
</tr>
<tr>
<td>Canon EOS R</td>
<td>36mm x 24mm</td>
<td>6720 x 4480</td>
<td>3846 x 2160 4K (16:9)</td>
<td>20.57mm x 11.57mm (Not the whole chip width is used)</td>
</tr>
<tr>
<td>Sony A7S II</td>
<td>35.6mm x 23.8mm</td>
<td>4240 x 2832</td>
<td>1920 x 1080 Full HD (16:9)</td>
<td>35.6mm x 20.0mm</td>
</tr>
<tr>
<td>Sony A7S II</td>
<td>35.6mm x 23.8mm</td>
<td>4240 x 2832</td>
<td>3840 x 2160 4K (16:9)</td>
<td>35.6mm x 20.0mm (The whole chip width is used)</td>
</tr>
<tr>
<td>Panasonic LUMIX DC-GH5S</td>
<td>19.2mm x 13.0mm</td>
<td>4096 x 2760</td>
<td>1920 x 1080 Full HD (16:9)</td>
<td>18.8mm x 10.6mm (yet to be confirmed)</td>
</tr>
<tr>
<td>Panasonic LUMIX DC-GH5S</td>
<td>19.2mm x 13.0mm</td>
<td>4096 x 2760</td>
<td>3846 x 2160 4K (16:9)</td>
<td>18.8mm x 10.6mm (yet to be confirmed)</td>
</tr>
<tr>
<td>Panasonic LUMIX DC-GH5S</td>
<td>19.2mm x 13.0mm</td>
<td>4096 x 2760</td>
<td>4096 x 2160 C4K (17:9)</td>
<td>19.2mm x 10.12mm (The whole chip width is used)</td>
</tr>
<tr>
<td>Panasonic LUMIX DC-GH5S</td>
<td>19.2mm x 13.0mm</td>
<td>4096 x 2760</td>
<td>3328 x 2496 Anamorphic (4:3)</td>
<td>17.3mm x 13.0mm (The whole chip height is used)</td>
</tr>
<tr>
<td>Nikon D800</td>
<td>35.9mm x 24.0mm</td>
<td>7360 x 4912</td>
<td>1920 x 1080 Full HD</td>
<td>32.0mm x 18.0mm (Not the whole chip width is used)</td>
</tr>
<tr>
<td>ZWO ASI178MM</td>
<td>7.4mm x 5.0mm</td>
<td>3096x2080</td>
<td>3096x2080</td>
<td>7.4mm x 5.0mm (The full chip size is used)</td>
</tr>
<tr>
<td>Pulnix TM-9701</td>
<td>8.9mm x 6.6mm</td>
<td>768 x 484</td>
<td>768 x 484</td>
<td>8.9mm x 6.6mm (The full chip size is used)</td>
</tr>
</tbody>
</table>

Effective chip size of GH5S with 0.64x SpeedBooster, in FHD or 4K mode: 29.37mm x 16.56mm
## 13.2 Overview of available fisheye lenses

<table>
<thead>
<tr>
<th>Lens</th>
<th>Mount</th>
<th>Aperture</th>
<th>Image Angle and Image Circle Diameter</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>Canon EF 8-15mm at 8mm</td>
<td>Canon EF</td>
<td>f/4.0</td>
<td>180° 22.9mm (measured myself)</td>
<td>Very good image quality</td>
</tr>
<tr>
<td>Sigma EX DG 8mm</td>
<td>Canon EF</td>
<td>f/3.5</td>
<td>180° 22.7mm (measured myself)</td>
<td>Mediocre image quality</td>
</tr>
<tr>
<td>Nippon Kogaku 8mm</td>
<td>M42 / Canon EF</td>
<td>f/2.8</td>
<td>180° 23.0mm (measured myself)</td>
<td>M42 mount with adapter to Canon EF</td>
</tr>
<tr>
<td>Sigma EX DG 4.5mm</td>
<td>Canon EF</td>
<td>f/2.8</td>
<td>180° 12.3mm (measured myself)</td>
<td>Mediocre image quality</td>
</tr>
<tr>
<td>Meike 6-11mm at 6mm</td>
<td>Canon EF</td>
<td>f/3.5</td>
<td>180° 15.1mm (measured myself)</td>
<td>Good image quality</td>
</tr>
<tr>
<td>Meike 6-11mm at 7.5mm</td>
<td>Canon EF</td>
<td>f/3.5</td>
<td>180° 18.4mm (measured myself)</td>
<td>Good image quality</td>
</tr>
<tr>
<td>Meike 6-11mm at 9.5mm</td>
<td>Canon EF</td>
<td>f/3.5</td>
<td>180° 23.7mm (measured myself)</td>
<td>Good image quality</td>
</tr>
<tr>
<td>Meike 6-11mm at 11mm</td>
<td>Canon EF</td>
<td>f/3.5</td>
<td>180° 28.7mm (measured myself)</td>
<td>Good image quality</td>
</tr>
<tr>
<td>Meike 8mm</td>
<td>Canon EF</td>
<td>f/3.5</td>
<td>180° approx. 26.9mm 200° approx. 29.9mm</td>
<td></td>
</tr>
<tr>
<td>Opteka 6.5mm</td>
<td>Canon EF</td>
<td>f/3.5</td>
<td>180° approx. 30mm</td>
<td>Bad image quality, focal length is about 9mm</td>
</tr>
<tr>
<td>Entaniya HAL250 6.0mm</td>
<td>Canon EF</td>
<td>f/5.6</td>
<td>180° 18.2mm 250° 23.7mm</td>
<td>Only suitable for mirrorless cameras, very expensive</td>
</tr>
<tr>
<td>Entaniya HAL250 4.3mm</td>
<td>Canon EF</td>
<td>f/4.0</td>
<td>180° 13.1mm 250° 17.0mm</td>
<td>Only suitable for mirrorless cameras, very expensive</td>
</tr>
<tr>
<td>Entaniya HAL250 3.6mm</td>
<td>Canon EF</td>
<td>f/2.8</td>
<td>180° 11.0mm 250° 14.25mm</td>
<td>Only suitable for mirrorless cameras, very expensive</td>
</tr>
<tr>
<td>Entaniya HAL250 3.0mm</td>
<td>Canon EF</td>
<td>f/2.8</td>
<td>180° 9.2mm 250° 11.9mm</td>
<td>Only suitable for mirrorless cameras, very expensive</td>
</tr>
<tr>
<td>Entaniya HAL200 6.0mm</td>
<td>Canon EF</td>
<td>f/4.0</td>
<td>180° 18.2mm 200° 19.9mm</td>
<td>Only suitable for mirrorless cameras, very expensive</td>
</tr>
<tr>
<td>Entaniya HAL200 5.0mm</td>
<td>Canon EF</td>
<td>f/5.6</td>
<td>180° 15.2mm 200° 16.8mm</td>
<td>Only suitable for mirrorless cameras, very expensive</td>
</tr>
<tr>
<td>Samyang 8mm Fisheye II</td>
<td>EF-M, Sony E</td>
<td>f/2.8</td>
<td>180° approx. 29.7mm 180° approx. 31mm</td>
<td>Only suitable for mirrorless cameras, short flange distance</td>
</tr>
<tr>
<td>Meike 6.5mm</td>
<td>MFT</td>
<td>f/2.0</td>
<td>180° 15.4mm 190° 15.85mm (measured myself)</td>
<td>Only suitable for mirrorless cameras, short flange distance</td>
</tr>
<tr>
<td>Olympus M.Zuiko 8mm</td>
<td>MFT</td>
<td>f/1.8</td>
<td>180° approx. 22mm</td>
<td>Lens hood must be removed mechanically</td>
</tr>
<tr>
<td>7artisans (Viltrox) 7.5mm</td>
<td>MFT</td>
<td>f/2.8</td>
<td>ca. 27mm (APS-C without vignetting)</td>
<td>Lens hood must be removed mechanically</td>
</tr>
<tr>
<td>ZLKC (OCDAY) 7.5mm</td>
<td>MFT</td>
<td>f/2.8</td>
<td>ca. 27mm (APS-C without vignetting)</td>
<td></td>
</tr>
<tr>
<td>Laowa 4mm</td>
<td>MFT</td>
<td>f/2.8</td>
<td>180° 7mm 210° 7mm</td>
<td>Only suitable for mirrorless cameras, short flange distance</td>
</tr>
<tr>
<td>iZugar MKX200-ASPH 3.8mm</td>
<td>MFT</td>
<td>f/2.8</td>
<td>180° 11.7mm 200° 13.5mm</td>
<td>Only suitable for mirrorless cameras, short flange distance</td>
</tr>
<tr>
<td>iZugar MKX22 3.25mm</td>
<td>MFT</td>
<td>f/2.5</td>
<td>180° approx. 6.2mm 220° approx. 10mm</td>
<td>Only suitable for mirrorless cameras, short flange distance</td>
</tr>
<tr>
<td>Yumiki 2.5mm</td>
<td>CS-Mount</td>
<td>f/1.6</td>
<td>180° approx. 6.1mm 190° approx. 6.4mm</td>
<td></td>
</tr>
<tr>
<td>SMTSEC 2.27mm</td>
<td>CS-Mount</td>
<td>f/1.4</td>
<td>185° 7.2mm</td>
<td></td>
</tr>
<tr>
<td>Fujinon 1.8mm</td>
<td>C-Mount</td>
<td>f/1.4</td>
<td>180° 5.5mm 185° 5.7mm</td>
<td></td>
</tr>
<tr>
<td>Fujinon 2.7mm</td>
<td>C-Mount</td>
<td>f/1.8</td>
<td>180° 8.4mm 185° 8.6mm</td>
<td></td>
</tr>
</tbody>
</table>
## 13.3 Favorable camera / fisheye combinations

<table>
<thead>
<tr>
<th>Camera</th>
<th>Video resolution</th>
<th>Lens</th>
<th>Aperture</th>
<th>Fully illuminated image circle</th>
<th>Effective number of pixels</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Canon 6D</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>640 x 480 (4:3)</td>
<td>Canon EF 8-15mm at 8mm</td>
<td>f/4.0</td>
<td>180°</td>
<td>460 Pixel</td>
<td></td>
</tr>
<tr>
<td>1920 x 1080</td>
<td>Sigma EX DG 4.5mm</td>
<td>f/2.8</td>
<td>180°</td>
<td>656 Pixel</td>
<td></td>
</tr>
<tr>
<td>1920 x 1080</td>
<td>Meike 6-11mm at 8.2mm</td>
<td>f/3.5</td>
<td>180°</td>
<td>1080 Pixel</td>
<td></td>
</tr>
<tr>
<td>1920 x 1080</td>
<td>Canon EF 8-15mm at 8mm</td>
<td>f/4.0</td>
<td>159°</td>
<td>952 Pixel</td>
<td></td>
</tr>
<tr>
<td><strong>Canon 5D MK4</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1920 x 1080</td>
<td>Sigma EX DG 4.5mm</td>
<td>f/2.8</td>
<td>180°</td>
<td>654 Pixel</td>
<td></td>
</tr>
<tr>
<td>1920 x 1080</td>
<td>Meike 6-11mm at 8.2mm</td>
<td>f/3.5</td>
<td>180°</td>
<td>1080 Pixel</td>
<td></td>
</tr>
<tr>
<td>1920 x 1080</td>
<td>Canon EF 8-15mm at 8mm</td>
<td>f/4.0</td>
<td>159°</td>
<td>955 Pixel</td>
<td></td>
</tr>
<tr>
<td>4096 x 2160</td>
<td>C4K (17:9)</td>
<td>Sigma EX DG 4.5mm</td>
<td>f/2.8</td>
<td>170°</td>
<td>2160 Pixel</td>
</tr>
<tr>
<td><strong>Canon EOS R</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3840 x 2160</td>
<td>Entaniya HAL250 3.6mm</td>
<td>f/2.8</td>
<td>180°</td>
<td>2054 Pixel</td>
<td></td>
</tr>
<tr>
<td><strong>Sony A7S II</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3840 x 2160</td>
<td>Sigma EX DG 4.5mm</td>
<td>f/2.8</td>
<td>180°</td>
<td>663 Pixel</td>
<td></td>
</tr>
<tr>
<td>3840 x 2160</td>
<td>Meike 6.5mm</td>
<td>f/2.0</td>
<td>180°</td>
<td>832 Pixel</td>
<td></td>
</tr>
<tr>
<td>3840 x 2160</td>
<td>Meike 6.5mm</td>
<td>f/2.0</td>
<td>180°</td>
<td>1663 Pixel</td>
<td></td>
</tr>
<tr>
<td>3840 x 2160</td>
<td>Meike 6-11mm at 8.2mm</td>
<td>f/3.5</td>
<td>180°</td>
<td>1080 Pixel</td>
<td></td>
</tr>
<tr>
<td>3840 x 2160</td>
<td>Meike 6-11mm at 8.2mm</td>
<td>f/3.5</td>
<td>180°</td>
<td>2160 Pixel</td>
<td></td>
</tr>
<tr>
<td><strong>Sony A7S II</strong></td>
<td>with external recorder</td>
<td>Olympus M.Zuiko 8mm</td>
<td>f/1.8</td>
<td>approx. 164°</td>
<td>2160 Pixel</td>
</tr>
<tr>
<td>3840 x 2160</td>
<td>Sigma EX DG 8mm</td>
<td>f/3.5</td>
<td>180°</td>
<td>2060 Pixel</td>
<td></td>
</tr>
<tr>
<td>3840 x 2160</td>
<td>Olympus M.Zuiko 8mm</td>
<td>f/1.8</td>
<td>180°</td>
<td>ca. 1996 Pixel</td>
<td></td>
</tr>
<tr>
<td><strong>Panasonic LUMIX GH5S</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3328 x 2496</td>
<td>Sigma EX DG 4.5mm</td>
<td>f/2.8</td>
<td>180°</td>
<td>2356 Pixel</td>
<td></td>
</tr>
<tr>
<td>3840 x 2160</td>
<td>Sigma EX DG 4.5mm, Speedbooster 0.71x</td>
<td>f/2.0</td>
<td>180°</td>
<td>888 Pixel</td>
<td></td>
</tr>
<tr>
<td>3840 x 2160</td>
<td>Sigma EX DG 4.5mm, Speedbooster 0.71x</td>
<td>f/2.0</td>
<td>180°</td>
<td>1775 Pixel</td>
<td></td>
</tr>
<tr>
<td>3840 x 2160</td>
<td>Sigma EX DG 4.5mm, Speedbooster 0.64x</td>
<td>f/1.8</td>
<td>180°</td>
<td>800 Pixel</td>
<td></td>
</tr>
<tr>
<td>3840 x 2160</td>
<td>Sigma EX DG 4.5mm, Speedbooster 0.64x</td>
<td>f/1.8</td>
<td>180°</td>
<td>1600 Pixel</td>
<td></td>
</tr>
<tr>
<td><strong>Nikon D800</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1920 x 1080</td>
<td>Meike 6-11mm at 7.3mm</td>
<td>f/3.5</td>
<td>180°</td>
<td>1080 Pixel</td>
<td></td>
</tr>
</tbody>
</table>
13.4 Flange distances

<table>
<thead>
<tr>
<th>Camera Mount</th>
<th>Flange Distance</th>
</tr>
</thead>
<tbody>
<tr>
<td>MFT (Micro 4/3)</td>
<td>19.25mm</td>
</tr>
<tr>
<td>Canon EF und EF-S</td>
<td>44.0mm</td>
</tr>
<tr>
<td>Canon EF-M</td>
<td>18.0mm</td>
</tr>
<tr>
<td>Canon R</td>
<td>20.0mm</td>
</tr>
<tr>
<td>Canon FD</td>
<td>42.0mm</td>
</tr>
<tr>
<td>M42 = M42x1.0</td>
<td>45.46mm</td>
</tr>
<tr>
<td>T2 = M42x0.75</td>
<td>55.0mm</td>
</tr>
<tr>
<td>C-Mount</td>
<td>17.526mm</td>
</tr>
<tr>
<td>CS-Mount</td>
<td>12.526mm</td>
</tr>
<tr>
<td>Sony E-Mount</td>
<td>18.0mm</td>
</tr>
<tr>
<td>Nikon F</td>
<td>46.5mm</td>
</tr>
<tr>
<td>ZWO ASI178MM</td>
<td>12.5mm</td>
</tr>
</tbody>
</table>

13.5 Aperture numbers, rounded and exact

<table>
<thead>
<tr>
<th>Aperture Number</th>
<th>Exact Aperture</th>
<th>Rounded Aperture</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.8</td>
<td>0.794</td>
<td>1.0</td>
</tr>
<tr>
<td>0.9</td>
<td>0.891</td>
<td>1.0</td>
</tr>
<tr>
<td>1.0</td>
<td>1.000</td>
<td>1.0</td>
</tr>
<tr>
<td>1.1</td>
<td>1.122</td>
<td>1.1</td>
</tr>
<tr>
<td>1.2</td>
<td>1.260</td>
<td>1.3</td>
</tr>
<tr>
<td>1.4</td>
<td>1.414</td>
<td>1.4</td>
</tr>
<tr>
<td>1.6</td>
<td>1.587</td>
<td>1.6</td>
</tr>
<tr>
<td>1.8</td>
<td>1.782</td>
<td>2.0</td>
</tr>
<tr>
<td>2.0</td>
<td>2.000</td>
<td>2.0</td>
</tr>
<tr>
<td>2.2</td>
<td>2.245</td>
<td>2.5</td>
</tr>
<tr>
<td>2.5</td>
<td>2.520</td>
<td>3.0</td>
</tr>
<tr>
<td>2.8</td>
<td>2.828</td>
<td>3.5</td>
</tr>
<tr>
<td>3.2</td>
<td>3.175</td>
<td>4.0</td>
</tr>
<tr>
<td>3.5</td>
<td>3.564</td>
<td>4.5</td>
</tr>
<tr>
<td>4.0</td>
<td>4.000</td>
<td>5.0</td>
</tr>
<tr>
<td>4.5</td>
<td>4.490</td>
<td>5.5</td>
</tr>
<tr>
<td>5.0</td>
<td>5.040</td>
<td>6.0</td>
</tr>
<tr>
<td>5.6</td>
<td>5.657</td>
<td>7.0</td>
</tr>
</tbody>
</table>

Formula for exact numbers: \( f_{no} = 2^{n/6} \) with \( n = -2 \) to \( 15 \)
13.6 Test patterns for fulldome projection

Very nice fulldome test patterns:
http://www.paulbourke.net/dome/testpattern/

Make a double-fisheye test image and an equirectangular test image:

```plaintext
set "FF=c:\ffmpeg\ffmpeg" :: Path to FFmpeg
set "IN=1200.png" :: Test pattern from http://www.paulbourke.net/dome/testpattern/1200.png
set "OUT=double_fisheye_test.png" :: Double fisheye test image

%FF% -i %IN% -i %IN% -lavfi ".[0]transpose=1[left];[1]transpose=2,negate[right];[left][right]hstack" -y %OUT%

set "IN=double_fisheye_test.png"
set "OUT=equirectangular_test.png" :: Equirectangular test image

%FF% -i %IN% -lavfi "v360=input=dfisheye:output=e:pitch=90" -y %OUT%

pause
```
### 14 Canon 5D-Mark4

#### 14.1 All Canon 5D-Mark4 video modes for PAL video system

<table>
<thead>
<tr>
<th>MOV / MP4</th>
<th>Movie rec. size</th>
<th>Size</th>
<th>Frame rate</th>
<th>Bit rate</th>
<th>YUV/bit</th>
<th>Image compression</th>
</tr>
</thead>
<tbody>
<tr>
<td>MOV</td>
<td>4K 25.00P MJPG</td>
<td>4096x2160</td>
<td>25</td>
<td>480 Mbps</td>
<td>4:2:2 / 8 bit</td>
<td>MJPG yuvj422p</td>
</tr>
<tr>
<td>MOV</td>
<td>4K 24.00P MJPG</td>
<td>4096x2160</td>
<td>24</td>
<td>480 Mbps</td>
<td>4:2:2 / 8 bit</td>
<td>MJPG yuvj422p</td>
</tr>
<tr>
<td>MOV</td>
<td>FHD 50.00P ALL-I</td>
<td>1920x1080</td>
<td>50</td>
<td>174 Mbps</td>
<td>4:2:0 / 8 bit</td>
<td>ALL-I h264 yuvj420p</td>
</tr>
<tr>
<td>MOV</td>
<td>FHD 50.00P IPB</td>
<td>1920x1080</td>
<td>50</td>
<td>59 Mbps</td>
<td>4:2:0 / 8 bit</td>
<td>IPB h264 yuvj420p</td>
</tr>
<tr>
<td>MOV</td>
<td>FHD 25.00P ALL-I</td>
<td>1920x1080</td>
<td>25</td>
<td>88 Mbps</td>
<td>4:2:0 / 8 bit</td>
<td>ALL-I h264 yuvj420p</td>
</tr>
<tr>
<td>MOV</td>
<td>FHD 25.00P IPB</td>
<td>1920x1080</td>
<td>25</td>
<td>30 Mbps</td>
<td>4:2:0 / 8 bit</td>
<td>IPB h264 yuvj420p</td>
</tr>
<tr>
<td>MOV</td>
<td>FHD 24.00P ALL-I</td>
<td>1920x1080</td>
<td>24</td>
<td>88 Mbps</td>
<td>4:2:0 / 8 bit</td>
<td>IPB h264 yuvj420p</td>
</tr>
<tr>
<td>MOV</td>
<td>FHD 24.00P IPB</td>
<td>1920x1080</td>
<td>24</td>
<td>30 Mbps</td>
<td>4:2:0 / 8 bit</td>
<td>IPB h264 yuvj420p</td>
</tr>
<tr>
<td>MOV</td>
<td>HD 100.0P ALL-I</td>
<td>1280x720</td>
<td>100</td>
<td>154 Mbps</td>
<td>4:2:0 / 8 bit</td>
<td>ALL-I h264 yuvj420p</td>
</tr>
<tr>
<td>MP4</td>
<td>FHD 50.00P IPB</td>
<td>1920x1080</td>
<td>50</td>
<td>58 Mbps</td>
<td>4:2:0 / 8 bit</td>
<td>IPB h264 yuvj420p</td>
</tr>
<tr>
<td>MP4</td>
<td>FHD 25.00P IPB</td>
<td>1920x1080</td>
<td>25</td>
<td>29 Mbps</td>
<td>4:2:0 / 8 bit</td>
<td>IPB h264 yuvj420p</td>
</tr>
<tr>
<td>MP4</td>
<td>FHD 25.00P IPB</td>
<td>1920x1080</td>
<td>25</td>
<td>12 Mbps</td>
<td>4:2:0 / 8 bit</td>
<td>IPB (&quot;Light&quot;, this is a stronger compression) h264 yuvj420p</td>
</tr>
<tr>
<td>MP4</td>
<td>FHD 24.00P IPB</td>
<td>1920x1080</td>
<td>24</td>
<td>29 Mbps</td>
<td>4:2:0 / 8 bit</td>
<td>IPB h264 yuvj420p</td>
</tr>
</tbody>
</table>
## 14.2 All Canon 5D-Mark4 video modes for NTSC video system

<table>
<thead>
<tr>
<th>MOV / MP4</th>
<th>Movie rec. size</th>
<th>Size</th>
<th>Frame rate</th>
<th>Bit rate</th>
<th>YUV/bit</th>
<th>Image compression</th>
</tr>
</thead>
<tbody>
<tr>
<td>MOV</td>
<td>4K 29.97P MJPG</td>
<td>4096x2160</td>
<td>29.97</td>
<td>480 Mbps</td>
<td>4:2:2 / 8 bit</td>
<td>MJPG yuvj422p</td>
</tr>
<tr>
<td>MOV</td>
<td>4K 23.98P MJPG</td>
<td>4096x2160</td>
<td>23.98</td>
<td>480 Mbps</td>
<td>4:2:2 / 8 bit</td>
<td>MJPG yuvj422p</td>
</tr>
<tr>
<td>MOV</td>
<td>4K 24.00P MJPG</td>
<td>4096x2160</td>
<td>24</td>
<td>480 Mbps</td>
<td>4:2:2 / 8 bit</td>
<td>MJPG yuvj422p</td>
</tr>
<tr>
<td>MOV</td>
<td>FHD 59.94P ALL-I</td>
<td>1920x1080</td>
<td>59.94</td>
<td>174 Mbps</td>
<td>4:2:0 / 8 bit</td>
<td>ALL-I h264 yuvj420p</td>
</tr>
<tr>
<td>MOV</td>
<td>FHD 59.94P IPB</td>
<td>1920x1080</td>
<td>59.94</td>
<td>59 Mbps</td>
<td>4:2:0 / 8 bit</td>
<td>IPB h264 yuvj420p</td>
</tr>
<tr>
<td>MOV</td>
<td>FHD 29.97P ALL-I</td>
<td>1920x1080</td>
<td>29.97</td>
<td>88 Mbps</td>
<td>4:2:0 / 8 bit</td>
<td>ALL-I h264 yuvj420p</td>
</tr>
<tr>
<td>MOV</td>
<td>FHD 29.97P IPB</td>
<td>1920x1080</td>
<td>29.97</td>
<td>30 Mbps</td>
<td>4:2:0 / 8 bit</td>
<td>IPB h264 yuvj420p</td>
</tr>
<tr>
<td>MOV</td>
<td>FHD 23.98P ALL-I</td>
<td>1920x1080</td>
<td>23.98</td>
<td>88 Mbps</td>
<td>4:2:0 / 8 bit</td>
<td>ALL-I h264 yuvj420p</td>
</tr>
<tr>
<td>MOV</td>
<td>FHD 23.98P IPB</td>
<td>1920x1080</td>
<td>23.98</td>
<td>30 Mbps</td>
<td>4:2:0 / 8 bit</td>
<td>IPB h264 yuvj420p</td>
</tr>
<tr>
<td>MOV</td>
<td>FHD 24.00P ALL-I</td>
<td>1920x1080</td>
<td>24</td>
<td>88 Mbps</td>
<td>4:2:0 / 8 bit</td>
<td>ALL-I h264 yuvj420p</td>
</tr>
<tr>
<td>MOV</td>
<td>FHD 24.00P IPB</td>
<td>1920x1080</td>
<td>24</td>
<td>30 Mbps</td>
<td>4:2:0 / 8 bit</td>
<td>IPB h264 yuvj420p</td>
</tr>
<tr>
<td>MP4</td>
<td>HD 119.9P ALL-I</td>
<td>1280x720</td>
<td>119.9</td>
<td>154 Mbps</td>
<td>4:2:0 / 8 bit</td>
<td>ALL-I h264 yuvj420p</td>
</tr>
<tr>
<td>MP4</td>
<td>FHD 59.94P IPB</td>
<td>1920x1080</td>
<td>59.94</td>
<td>58 Mbps</td>
<td>4:2:0 / 8 bit</td>
<td>IPB h264 yuvj420p</td>
</tr>
<tr>
<td>MP4</td>
<td>FHD 29.97P IPB</td>
<td>1920x1080</td>
<td>29.97</td>
<td>29 Mbps</td>
<td>4:2:0 / 8 bit</td>
<td>IPB h264 yuvj420p</td>
</tr>
<tr>
<td>MP4</td>
<td>FHD 29.97P IPB</td>
<td>1920x1080</td>
<td>29.97</td>
<td>12 Mbps</td>
<td>4:2:0 / 8 bit</td>
<td>IPB (&quot;Light&quot;, this is a stronger compression) h264 yuvj420p</td>
</tr>
<tr>
<td>MP4</td>
<td>FHD 23.98P IPB</td>
<td>1920x1080</td>
<td>23.98</td>
<td>29 Mbps</td>
<td>4:2:0 / 8 bit</td>
<td>IPB h264 yuvj420p</td>
</tr>
<tr>
<td>MP4</td>
<td>FHD 24.00P IPB</td>
<td>1920x1080</td>
<td>24.00</td>
<td>29 Mbps</td>
<td>4:2:0 / 8 bit</td>
<td>IPB h264 yuvj420p</td>
</tr>
</tbody>
</table>

Important note: If the size of a video exceeds 4GB, it can only be downloaded to the computer with "EOS Utility" software.
14.3 Video tutorials for Canon 5D-Mark4

The Canon 5D-Mark4 has a very good autofocus and is perfect for photography of fast moving objects (e.g. wildlife, birds). I'm not a friend of video tutorials, but for the Canon 5D-Mark4 I found some tutorials that are indeed helpful. I will summarize the content below:

Grant Atkinson: Canon 5D Mk IV - Autofocus: Part 1/4 - Control Setup for Moving Subjects
https://www.youtube.com/watch?v=7iP60Np0lPW

<table>
<thead>
<tr>
<th>AF Operation</th>
<th>Notes</th>
<th>Drive Mode</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>ONE SHOT</td>
<td>For non-moving objects</td>
<td>-</td>
<td>Single shot</td>
</tr>
<tr>
<td>AI FOCUS</td>
<td>This decides automatically if the object is moving or not. Not recommended.</td>
<td>H</td>
<td>7 Pictures per second</td>
</tr>
<tr>
<td>AI SERVO</td>
<td>For moving objects, recommended as default.</td>
<td>-</td>
<td>3 Pictures per second</td>
</tr>
<tr>
<td></td>
<td></td>
<td>S</td>
<td>3 Pictures per second, silent mode</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Clock Symbol</td>
<td>10 Seconds self timer</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Clock Symbol 2</td>
<td>2 Seconds self timer</td>
</tr>
</tbody>
</table>

Orange Menu (second from right) → 3 → Custom Controls

<table>
<thead>
<tr>
<th>Shutter Button</th>
<th>leave as-is</th>
</tr>
</thead>
<tbody>
<tr>
<td>AF_ON Button</td>
<td>set to AF_OFF, that means when you are in AF_SERVO mode you can hold the focus as long as you press this button.</td>
</tr>
<tr>
<td>* Button</td>
<td>set to ONE_SHOT/SERVO, that means by pressing this button you can toggle very fast between ONE_SHOT and AF_SERVO. Additionally you must press the INFO button and select the option to the right. But this function isn't very important, because you can work without ONE_SHOT. In another video he sets the * button also to AF_OFF, which is useful if you accidentally press the wrong button.</td>
</tr>
<tr>
<td>Multi_Controller</td>
<td>Set to &quot;Direct AF point selection&quot;</td>
</tr>
<tr>
<td>AF Area Selection Button</td>
<td>Set to &quot;Direct AF area selection&quot;</td>
</tr>
<tr>
<td>SET Button</td>
<td>In another video he sets the SET button to &quot;Exposure Compensation&quot;</td>
</tr>
</tbody>
</table>
Pink AF Menü (second from left) --> 4 --> Select AF area selec. mode
Here you can select which of the 7 AF area selection modes you want to use. He chooses 2, 3 and 5.

<table>
<thead>
<tr>
<th>(1) Spot AF (Square with Point)</th>
<th>Very small, good choice if you take pictures through branches</th>
</tr>
</thead>
<tbody>
<tr>
<td>(2) Single Point AF</td>
<td>This is the default setting, very precise and fast, if you manage to hold the point on the object.</td>
</tr>
<tr>
<td>(3) Expand AF area (5 Points)</td>
<td>Recommended method for moving objects. The center point is prioritized and if this point looses focus, then one of the neighbor points is used. Place the center point on the eye of the object.</td>
</tr>
<tr>
<td>(4) AF Expand Surround (9 Points)</td>
<td>Same as (3), but 8 neighbor points.</td>
</tr>
<tr>
<td>(5) Zone AF (9 or 12 Points)</td>
<td>All selected points have the same weight. You don't know which point is actually used. Don't use this method if you want to have the focus on the eye of the object.</td>
</tr>
<tr>
<td>(6) Large Zone</td>
<td>Same as (5), but more points.</td>
</tr>
<tr>
<td>(7) Auto AF Selection (all 61 Points)</td>
<td>This may be useful for birds in the sky, if there is sufficient depth of focus. You don't know which point is actually used. Don't use this method if you want to have the focus on the eye of the object.</td>
</tr>
</tbody>
</table>

Pink AF Menü (second from left) --> 4 --> Selectable AF Point
Here you can reduce the number of selectable points. His choice: 61 or 15, because then you can choose the best point very fast.

Grant Atkinson: Canon 5D Mk IV - Autofocus: Part 3/4 - Prioritizing Your Autofocus Options
https://www.youtube.com/watch?v=VOilQs1UEl8

Pink AF Menü (second from left) --> 2
Here you can set the priorities for the first picture and for all subsequent pictures. His choice: 1th image: RELEASE, 2nd image: 0 to -2

<table>
<thead>
<tr>
<th>Focus Priority</th>
<th>This means the first picture is taken not before the focus is found. This may lead to pauses, if no focus is found.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Speed Priority</td>
<td>This means that less time is used for focusing.</td>
</tr>
</tbody>
</table>

197
Pink AF Menü (second from left) --> 1

The "cases" contain predefined settings. He doesn't use them, however he has put the three settings (Tracking Sensitivity, Accel/decel tracking and AF pt auto switching) into "MyMenu". This can be done as follows:

Press Q, until "My Menu" is selected.
Add My Menu Tab, OK
Configure MyMenu1
Select items to register.
Now select the three items that were mentioned above. They are now available in "My Menu".

<table>
<thead>
<tr>
<th>Tracking Sensitivity</th>
<th>This is by far the most important parameter! It describes how easily the focus can move away from the previously found focus.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accel/decel tracking</td>
<td>This is difficult to understand. Best if you leave it at 0.</td>
</tr>
<tr>
<td>AF pt auto switching</td>
<td>This describes, how fast the camera switches from one AF point to a neighbor AF point. He leaves it at 0, which means deactivated.</td>
</tr>
</tbody>
</table>
Grant Atkinson: Canon 5D Mark IV - Settings For Wildlife Photography
https://www.youtube.com/watch?v=yy_72JQ-QT4

Red Camera Menu (first from left)

<table>
<thead>
<tr>
<th>Page 1 --&gt; Lens aberration correction</th>
<th>He switches all options off, so that the pictures can be saved faster.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Page 2 --&gt; Auto Lighting Optimizer</td>
<td>OFF</td>
</tr>
<tr>
<td>Page 3 --&gt; High ISO speed NR</td>
<td>OFF</td>
</tr>
<tr>
<td>Page 2 --&gt; ISO speed settings</td>
<td>AUTO 100 - 12800 for both ranges</td>
</tr>
<tr>
<td>Page 1 --&gt; Release Shutter without card</td>
<td>Disable</td>
</tr>
</tbody>
</table>

Pink AF Menu (second from left)

| Page 4 --> Auto AF pt sel: EOS iTR AF | OFF                                      |

Grant Atkinson: Shooting Canon 5D Mark IV in M mode with auto ISO
https://www.youtube.com/watch?v=Xmud7-O8HNs

You can use the M mode together with "Auto ISO". Exposure compensation is also possible in M mode.

Tony & Chelsea Northrup: How to Photograph Flying Birds
https://www.youtube.com/watch?v=GFghMNX9zrl

Shutter: 1/2000s TV, Auto ISO might be useful

<table>
<thead>
<tr>
<th>Birds in front of trees or water</th>
<th>Use a single AF point and hold it on the object.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Birds in the sky</td>
<td>It's easier to use all AF points.</td>
</tr>
</tbody>
</table>
15 Panasonic LUMIX GH5S

15.1 GH5S Record formats

<table>
<thead>
<tr>
<th>Record format</th>
<th>Bits</th>
<th>Video Codec</th>
<th>Audio Codec</th>
<th>Anamorphic</th>
<th>VFR</th>
<th>HLG</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>AVCHD</td>
<td>8</td>
<td>?</td>
<td>?</td>
<td>no</td>
<td>some</td>
<td>no</td>
<td>This data format is suitable for when playing back on a high-definition TV, etc.</td>
</tr>
<tr>
<td>MP4</td>
<td>8</td>
<td>?</td>
<td>?</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>This data format is suitable for when playing back on a PC, etc.</td>
</tr>
<tr>
<td>MP4 HEVC (High Efficiency Video Coding)</td>
<td>10</td>
<td>h.265</td>
<td>?</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>This data format is for HDR motion picture and suitable for playback on a HDR (HLG format)-compatible TV or recorder.</td>
</tr>
<tr>
<td>MP4 (LPCM)</td>
<td>8 or 10</td>
<td>h.264</td>
<td>LPCM (uncompressed)</td>
<td>possible</td>
<td>some</td>
<td>only 10 bit</td>
<td>The MP4 data format for image editing.</td>
</tr>
<tr>
<td>MOV</td>
<td>8 or 10</td>
<td>h.264</td>
<td>?</td>
<td>possible</td>
<td>some</td>
<td>only 10 bit</td>
<td>Data format for image editing.</td>
</tr>
</tbody>
</table>

15.2 GH5S (and other) Abbreviations

DR = Dynamic Range
ETC = Extra Tele Conversion
ETTL = Expose To The Left
ETTR = Expose To The Right
HDR = High Dynamic Range
HLG = Hybrid Log Gamma
LUT = Look-up-Table
NR = Noise Reduction
PQ = Perceptual Quantization
SDR = Standard Dynamic Range
SNR = Signal to Noise Ratio
VFR = Variable Frame Rate
VO = Voice-Over
15.3 GH5S Recommended settings

<table>
<thead>
<tr>
<th></th>
<th>Cinelike-D</th>
<th>V-LOG L</th>
<th>HLG (Hybrid Log Gamma)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Contrast</td>
<td>0</td>
<td>[NA]</td>
<td>[NA]</td>
</tr>
<tr>
<td>Sharpness (1)</td>
<td>-5 ?</td>
<td>-5 ?</td>
<td>-5 ?</td>
</tr>
<tr>
<td>Noise Reduction (2)</td>
<td>-5</td>
<td>-5</td>
<td>-5</td>
</tr>
<tr>
<td>Saturation</td>
<td>-5</td>
<td>[NA]</td>
<td>-5</td>
</tr>
<tr>
<td>Hue</td>
<td>0</td>
<td>[NA]</td>
<td>0</td>
</tr>
<tr>
<td>Luminance Level</td>
<td>0-1023 (0-255 for 8 bit)</td>
<td>fixed at 32-200 (128-800 for 10 bit) (3)</td>
<td>fixed at 0-1023</td>
</tr>
<tr>
<td>Zebras</td>
<td>100%</td>
<td>75%</td>
<td>90%</td>
</tr>
<tr>
<td>Exposure compensation</td>
<td>+1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Possible ISO range for &quot;Dual Native ISO Settings&quot; = Low</td>
<td>80 - 800</td>
<td>320 - 1600</td>
<td>320 - 1600</td>
</tr>
<tr>
<td>Possible ISO range for &quot;Dual Native ISO Settings&quot; = High</td>
<td>800 - 204800</td>
<td>1600 - 25600</td>
<td>1600 - 204800</td>
</tr>
<tr>
<td>Dynamic range [F-Stops]</td>
<td>10.5</td>
<td>11.58</td>
<td>11.5</td>
</tr>
<tr>
<td>Notes</td>
<td></td>
<td>Best choice for video post processing?</td>
<td>Best choice for night sky!</td>
</tr>
</tbody>
</table>

(1) At higher ISO values (for example 25600), the sharpness setting is quite irrelevant, as there is no big difference in videos taken with sharpness -5 and +5. I'm unsure if negative sharpness values are a low pass filter or not.

(2) Any setting larger than -5 will suppress fainter stars in the night sky!

(3) V-LOG L uses only the range [128..800] from the possible range [0..1023], which means it's closer to 9-bit than to 10-bit
15.4  **GH5S Custom settings C1, C2, C3-1, C3-2, C3-2**

Up to 5 settings can be saved in Menu -> Settings -> Cust.Set Mem.

They can be loaded by turning the wheel to C1, C2 or C3.

In case of C3, you must additionally press the menu button and then select C3-1, C3-2 or C3-3.

My own settings:

<table>
<thead>
<tr>
<th></th>
<th>Rec Format</th>
<th>Pixel</th>
<th>fps</th>
<th>ISO, Photo Style</th>
<th>Exposure Mode, Exposure time</th>
<th>System Frequency</th>
<th>Application</th>
</tr>
</thead>
<tbody>
<tr>
<td>C1</td>
<td>[4K/10bit/150M/25p] 422 / 10Bit / Long GOP</td>
<td>3840x2160 4K</td>
<td>25</td>
<td>Auto, STD</td>
<td>M, 1/50s</td>
<td>50.00Hz (PAL)</td>
<td>For 4K videos</td>
</tr>
<tr>
<td>C2</td>
<td>[C4K/10bit/150M/25p] 422 / 10Bit / Long GOP</td>
<td>4096x2160 C4K</td>
<td>25</td>
<td>Auto, STD</td>
<td>M, 1/50s</td>
<td>50.00Hz (PAL)</td>
<td>For C4K videos</td>
</tr>
<tr>
<td>C3-1</td>
<td>[4K/A/150M/25p] 422 / 10bit / Long GOP Anamorphic</td>
<td>3328x2496 HLG, NR=-5</td>
<td>25</td>
<td>M, 1/25s</td>
<td>50.00Hz (PAL)</td>
<td>For meteor astronomy with SpeedBooster and Nippon Kogaku 8mm f/2.8 fisheye lens</td>
<td></td>
</tr>
<tr>
<td>C3-2</td>
<td>[FHD/8bit/100M/25p] 420 / 8Bit / Long GOP FHD</td>
<td>1920x1080 STD</td>
<td>125</td>
<td>51200, STD</td>
<td>M, 1/25s</td>
<td>50.00Hz (PAL)</td>
<td>For video astronomy: Variable framerate: 125 Ex.Telev Conv is OFF, but can be set to ON</td>
</tr>
<tr>
<td>C3-3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
15.5 GH5S Luminance level

Motion Picture > Luminance Level
Select the luminance range to match the use of video. Settings: [0-255]/[16-235]/[16-255]

• If you set Rec Quality to a 10bit motion picture setting, the available options change to [0-1023], [64-940], and [64-1023].
• This function works only for motion pictures. Still pictures (including those you take during motion picture recording) will be taken with [0-255].
• When Rec Format is set to AVCHD or MP4, [0-255] in Luminance Level will switch to [16-255].
• When Photo Style is set to Hybrid Log Gamma, setting is fixed to [0-1023]. The manual says [64-640], but I think this is wrong.
• When Photo Style is set to V-Log L, setting is fixed to [32-200] or [128-800]. The manual says [0-255], but I think this is wrong.

15.6 GH5S Master pedestal level

Creative Video > Master Pedestal Level

<table>
<thead>
<tr>
<th></th>
<th>This side creates a high contrast image with a crisp atmosphere.</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Standard</td>
</tr>
<tr>
<td>+</td>
<td>This side creates a slightly misty atmosphere.</td>
</tr>
</tbody>
</table>

This function is not available when Photo Style is set to V-Log L

15.7 GH5S Video size

<table>
<thead>
<tr>
<th>Mode</th>
<th>Resolution</th>
<th>Read-out Chip Size</th>
<th>Diagonal Size</th>
<th>Number of Pixels</th>
</tr>
</thead>
<tbody>
<tr>
<td>4K</td>
<td>3820 x 2160 (16:9)</td>
<td>18.8mm x 10.6mm</td>
<td>21.6mm</td>
<td>8251200</td>
</tr>
<tr>
<td>C4K</td>
<td>4096 x 2160 (17:9)</td>
<td>19.2mm x 10.12mm</td>
<td>21.7mm</td>
<td>8847360</td>
</tr>
<tr>
<td>Anamorphic</td>
<td>3328 x 2496 (4:3)</td>
<td>17.3mm x 13.0mm</td>
<td>21.6mm</td>
<td>8306688</td>
</tr>
<tr>
<td>FHD</td>
<td>1920 x 1080 (16:9)</td>
<td>18.8mm x 10.6mm (1)</td>
<td>21.6mm</td>
<td>2062800</td>
</tr>
<tr>
<td>FHD with 2.1x Extra Tele Conversion</td>
<td>1920 x 1080 (16:9)</td>
<td>8.95mm x 5.05mm</td>
<td>10.3mm</td>
<td>2062800</td>
</tr>
</tbody>
</table>

(1) Read-out chip size is smaller when frame rate is larger than 200
15.8 GH5S Mechanical / electronic shutter

Rec > Shutter Type

<table>
<thead>
<tr>
<th>Shutter Type</th>
<th>ISO</th>
<th>Exposure Time Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mechanical shutter</td>
<td>100-204800</td>
<td>60s - 1/8000s</td>
</tr>
<tr>
<td>Electronic shutter</td>
<td>204800</td>
<td>1/30s - 1/16000s</td>
</tr>
<tr>
<td></td>
<td>102400</td>
<td>1/15s - 1/16000s</td>
</tr>
<tr>
<td></td>
<td>51200</td>
<td>1/8s - 1/16000s</td>
</tr>
<tr>
<td></td>
<td>25600</td>
<td>1/4s - 1/16000s</td>
</tr>
<tr>
<td></td>
<td>12800</td>
<td>1/2s - 1/16000s</td>
</tr>
<tr>
<td></td>
<td>100 - 6400</td>
<td>1s - 1/16000s</td>
</tr>
</tbody>
</table>

15.9 GH5S Longer exposure time than framerate allows

When making a 25fps video, exposure times longer than 1/25s up to 1/2s are possible. Duplicated frames are written to the SD card.

At least these settings are required (there may be more requirements that I don't know):

-- Creative film mode
-- Exposure mode "M"
-- Autofocus must be switched off at the lens
-- "SS/Gain Operation" must be set to "SEC/ISO"
-- Not in variable framerate mode
15.10  GH5S Cable remote trigger

The cable remote trigger has a 2.5mm connector with 4 contacts:

<table>
<thead>
<tr>
<th>Contact</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tip contact</td>
<td>not connected</td>
</tr>
<tr>
<td>2nd contact</td>
<td>not connected</td>
</tr>
<tr>
<td>3rd contact</td>
<td>not pressed: 38.5 kOhm to ground (33 kOhm more than half pressed)</td>
</tr>
<tr>
<td></td>
<td>half pressed: 5.5 kOhm to ground (3.3kOhm more than full pressed)</td>
</tr>
<tr>
<td></td>
<td>full pressed: 2.2 kOhm to ground</td>
</tr>
<tr>
<td>Outer contact</td>
<td>ground</td>
</tr>
</tbody>
</table>

15.11  GH5S Variable frame rate

<table>
<thead>
<tr>
<th>System Frequency</th>
<th>Rec Quality</th>
<th>Available Framerates</th>
</tr>
</thead>
<tbody>
<tr>
<td>59.94Hz (NTSC)</td>
<td>[4K/8bit/100M/30p]</td>
<td>2 15 26 28 30 32 34 45 60</td>
</tr>
<tr>
<td></td>
<td>[FHD/24M/30p]</td>
<td>2 30 56 58 60 62 64 90 120 150 180 210 240</td>
</tr>
<tr>
<td></td>
<td>[FHD/8bit/100M/60p]</td>
<td>2 15 26 28 30 32 34 45 60 75 90 105 120 135 150 165 180 195 210 225 240</td>
</tr>
<tr>
<td></td>
<td>[FHD/8bit/100M/30p]</td>
<td>2 15 26 28 30 32 34 45 60 75 90 105 120 135 150 165 180 195 210 225 240</td>
</tr>
<tr>
<td></td>
<td>[4K/8bit/100M/24p]</td>
<td>2 12 20 22 24 26 28 36 48 60</td>
</tr>
<tr>
<td></td>
<td>[FHD/24M/24p]</td>
<td>2 12 20 22 24 26 28 36 48 60 72 84 96 108 120 132 144 156 168 180 192 204 216 228 240</td>
</tr>
<tr>
<td>50.00Hz (PAL)</td>
<td>[4K/8bit/100M/25p]</td>
<td>2 12 21 23 25 27 30 37 60</td>
</tr>
<tr>
<td></td>
<td>[FHD/24M/25p]</td>
<td>2 25 46 48 50 52 54 75 100 125 150 200 240</td>
</tr>
<tr>
<td></td>
<td>[FHD/8bit/100M/50p]</td>
<td>2 25 46 48 50 52 54 75 100 125 150 200 240</td>
</tr>
<tr>
<td></td>
<td>[FHD/8bit/100M/25p]</td>
<td>2 12 21 23 25 27 30 37 50 62 75 87 100 112 125 137 150 175 200 225 240</td>
</tr>
<tr>
<td>24.00Hz (CINEMA)</td>
<td>[4K/8bit/100M/24p]</td>
<td>2 12 20 22 24 26 28 36 48 60</td>
</tr>
<tr>
<td></td>
<td>[C4K/8bit/100M/24p]</td>
<td>2 12 20 22 24 26 28 36 48 60 72 84 96 108 120 132 144 156 168 180 192 204 216 228 240</td>
</tr>
<tr>
<td></td>
<td>[FHD/8bit/100M/24p]</td>
<td>2 12 20 22 24 26 28 36 48 60 72 84 96 108 120 132 144 156 168 180 192 204 216 228 240</td>
</tr>
</tbody>
</table>

Note: The GH5S doesn't record any audio in VFR mode.
15.12 Recording duration on SD cards

<table>
<thead>
<tr>
<th>Mbps</th>
<th>MB/s</th>
<th>MB/min</th>
<th>128GB card</th>
<th>256GB card</th>
<th>512GB card</th>
<th>640GB card (128GB + 512GB)</th>
<th>1024GB card (512GB + 512GB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>400</td>
<td>50</td>
<td>3000</td>
<td>43 min = 0.7 h</td>
<td>87 min = 1.4 h</td>
<td>174 min = 2.9 h</td>
<td>218 min = 3.6 h</td>
<td>349 min = 5.8 h</td>
</tr>
<tr>
<td>200</td>
<td>25</td>
<td>1500</td>
<td>87 min = 1.4 h</td>
<td>174 min = 2.9 h</td>
<td>349 min = 5.8 h</td>
<td>436 min = 7.2 h</td>
<td>699 min = 11.6 h</td>
</tr>
<tr>
<td>150</td>
<td>18.75</td>
<td>1125</td>
<td>116 min = 1.9 h</td>
<td>233 min = 3.9 h</td>
<td>466 min = 7.8 h</td>
<td>582 min = 9.7 h</td>
<td>932 min = 15.5 h</td>
</tr>
<tr>
<td>100</td>
<td>12.5</td>
<td>750</td>
<td>174 min = 2.9 h</td>
<td>349 min = 5.8 h</td>
<td>699 min = 11.6 h</td>
<td>873 min = 14.5 h</td>
<td>1398 min = 23.3 h</td>
</tr>
<tr>
<td>72</td>
<td>9</td>
<td>540</td>
<td>242 min = 4.0 h</td>
<td>485 min = 8.1 h</td>
<td>970 min = 16.2 h</td>
<td>1213 min = 20.2 h</td>
<td>1941 min = 32.3 h</td>
</tr>
<tr>
<td>28</td>
<td>3.5</td>
<td>210</td>
<td>624 min = 10.4 h</td>
<td>1248 min = 20.8 h</td>
<td>2496 min = 41.6 h</td>
<td>3120 min = 52.0 h</td>
<td>4993 min = 83.2 h</td>
</tr>
<tr>
<td>24</td>
<td>3</td>
<td>180</td>
<td>728 min = 12.1 h</td>
<td>1456 min = 24.3 h</td>
<td>2912 min = 48.5 h</td>
<td>3640 min = 60.6 h</td>
<td>5825 min = 97.0 h</td>
</tr>
<tr>
<td>20</td>
<td>2.5</td>
<td>150</td>
<td>873 min = 14.5 h</td>
<td>1747 min = 29.1 h</td>
<td>3495 min = 58.2 h</td>
<td>4369 min = 72.8 h</td>
<td>6990 min = 116.5 h</td>
</tr>
<tr>
<td>17</td>
<td>2.125</td>
<td>127.5</td>
<td>1028 min = 17.1 h</td>
<td>2056 min = 34.3 h</td>
<td>4112 min = 68.5 h</td>
<td>5397 min = 89.9 h</td>
<td>8224 min = 137.0 h</td>
</tr>
</tbody>
</table>

15.13 GH5S Cheap chinese battery adapters

If a 10kOhm resistor is soldered between the "-" and "T" contacts, the GH5S will accept all voltages from 6.5 to 8.5 Volts without any error messages. Without this resistance, the input voltage is much more critical. The original Panasonic DMW-AC10E power supply is rated 8.4V at 2.5A and the voltage is about 9.1V without load.

15.14 GH5S Telescopic effect

Set the [Ex. Tele Conv.] parameter to [ON] for a fixed 2.1x telescopic effect. This is on page 2/5 in the motion pictures menu. This function is not available when [HDR] is set to [ON], or when motion pictures size is set to [C4K] or [4K] in [Rec Quality], or when a frame rate of 150fps or higher is set for [Variable Frame Rate].
### 15.15  GH5S with SpeedBooster 0.64x

<table>
<thead>
<tr>
<th>Lens</th>
<th>With SpeedBooster 0.64x</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sigma EX DG 4.5mm f/2.8</td>
<td>2.9mm f/1.8</td>
</tr>
<tr>
<td>Meike 6-11mm f/3.5</td>
<td>3.8mm-7.0mm f/2.2</td>
</tr>
<tr>
<td>Nippon Kogaku 8mm f/2.8</td>
<td>5.1mm f/1.8</td>
</tr>
<tr>
<td>Sigma EX DG 8mm f/3.5</td>
<td>5.1mm f/2.2</td>
</tr>
<tr>
<td>Canon EF 8-15mm f/4.0</td>
<td>5.1mm f/2.5</td>
</tr>
<tr>
<td>Canon EF 11-24mm f/4.0</td>
<td>7.0mm-15.4mm f/2.5</td>
</tr>
<tr>
<td>Sigma 14mm f/1.8</td>
<td>9.0mm f/1.1</td>
</tr>
<tr>
<td>Canon CN-E 24mm T1.5 L F</td>
<td>15.4mm T 0.96</td>
</tr>
<tr>
<td>Sigma 24mm f/1.4</td>
<td>15.4mm f/0.9</td>
</tr>
<tr>
<td>Laowa 24mm f/14</td>
<td>15.4mm f/9.0</td>
</tr>
<tr>
<td>Canon EF 24-70mm f/4.0</td>
<td>15.4mm-44.8mm f/2.5</td>
</tr>
<tr>
<td>Canon EF 50mm f/1.4</td>
<td>32mm f/0.9</td>
</tr>
<tr>
<td>Canon EF 100mm f/2.8</td>
<td>64mm f/1.8</td>
</tr>
<tr>
<td>Canon EF 100-400mm f/4.5-5.6</td>
<td>64mm-256mm f/2.8-3.5</td>
</tr>
<tr>
<td>Canon EF 200mm f/2.0</td>
<td>128mm f/1.2</td>
</tr>
<tr>
<td>Canon EF 400mm f/2.8</td>
<td>256mm f/1.8</td>
</tr>
<tr>
<td>Canon EF 500mm f/4.0</td>
<td>320mm f/2.5</td>
</tr>
</tbody>
</table>
15.16  GH5S, all 77 video modes

<table>
<thead>
<tr>
<th>Rec Quality</th>
<th>Rec Format</th>
<th>VFR</th>
<th>HLG</th>
<th>System frequency</th>
<th>Size</th>
<th>Frame rate</th>
<th>Bit rate</th>
<th>YUV/bit</th>
<th>Image compression</th>
</tr>
</thead>
<tbody>
<tr>
<td>[C4K/8bit/150M/60p]</td>
<td>MP4 (LPCM) / MOV</td>
<td>no</td>
<td>no</td>
<td>59.94Hz (NTSC)</td>
<td>4096x2160</td>
<td>59.94p</td>
<td>150 Mbps</td>
<td>4:2:0/8 bit</td>
<td>Long GOP</td>
</tr>
<tr>
<td>[C4K/10bit/150M/30p]</td>
<td>MP4 (LPCM) / MOV</td>
<td>no</td>
<td>yes</td>
<td>59.94Hz (NTSC)</td>
<td>4096x2160</td>
<td>29.97p</td>
<td>150 Mbps</td>
<td>4:2:2/10 bit</td>
<td>Long GOP</td>
</tr>
<tr>
<td>[C4K/8bit/100M/30p]</td>
<td>MP4 (LPCM) / MOV</td>
<td>no</td>
<td>no</td>
<td>59.94Hz (NTSC)</td>
<td>4096x2160</td>
<td>29.97p</td>
<td>100 Mbps</td>
<td>4:2:0/8 bit</td>
<td>Long GOP</td>
</tr>
<tr>
<td>[C4K/ALL-I/400M/24p]</td>
<td>MP4 (LPCM) / MOV</td>
<td>yes</td>
<td>no</td>
<td>59.94Hz (NTSC)</td>
<td>4096x2160</td>
<td>23.98p</td>
<td>400 Mbps</td>
<td>4:2:2/10 bit</td>
<td>ALL-Intra</td>
</tr>
<tr>
<td>[C4K/10bit/150M/24p]</td>
<td>MP4 (LPCM) / MOV</td>
<td>no</td>
<td>no</td>
<td>59.94Hz (NTSC)</td>
<td>4096x2160</td>
<td>23.98p</td>
<td>100 Mbps</td>
<td>4:2:0/8 bit</td>
<td>Long GOP</td>
</tr>
<tr>
<td>[C4K/8bit/100M/24p]</td>
<td>MP4 (LPCM) / MOV</td>
<td>no</td>
<td>no</td>
<td>59.94Hz (NTSC)</td>
<td>4096x2160</td>
<td>23.98p</td>
<td>100 Mbps</td>
<td>4:2:0/8 bit</td>
<td>Long GOP</td>
</tr>
<tr>
<td>[4K/8bit/150M/60p]</td>
<td>MP4 (LPCM) / MOV</td>
<td>no</td>
<td>no</td>
<td>59.94Hz (NTSC)</td>
<td>3840x2160</td>
<td>59.94p</td>
<td>150 Mbps</td>
<td>4:2:0/8 bit</td>
<td>Long GOP</td>
</tr>
<tr>
<td>[4K/ALL-I/400M/30p]</td>
<td>MP4 (LPCM) / MOV</td>
<td>no</td>
<td>yes</td>
<td>59.94Hz (NTSC)</td>
<td>3840x2160</td>
<td>29.97p</td>
<td>150 Mbps</td>
<td>4:2:2/10 bit</td>
<td>Long GOP</td>
</tr>
<tr>
<td>[4K/10bit/150M/30p]</td>
<td>MP4 (LPCM) / MOV</td>
<td>no</td>
<td>yes</td>
<td>59.94Hz (NTSC)</td>
<td>3840x2160</td>
<td>29.97p</td>
<td>100 Mbps</td>
<td>4:2:2/10 bit</td>
<td>ALL-Intra</td>
</tr>
<tr>
<td>[4K/8bit/100M/30p]</td>
<td>MP4 (LPCM) / MOV</td>
<td>yes</td>
<td>no</td>
<td>59.94Hz (NTSC)</td>
<td>3840x2160</td>
<td>23.98p</td>
<td>100 Mbps</td>
<td>4:2:0/8 bit</td>
<td>Long GOP</td>
</tr>
<tr>
<td>[4K/10bit/150M/24p]</td>
<td>MP4 (LPCM) / MOV</td>
<td>no</td>
<td>yes</td>
<td>59.94Hz (NTSC)</td>
<td>3840x2160</td>
<td>23.98p</td>
<td>150 Mbps</td>
<td>4:2:2/10 bit</td>
<td>Long GOP</td>
</tr>
<tr>
<td>[4K/8bit/100M/24p]</td>
<td>MP4 (LPCM) / MOV</td>
<td>yes</td>
<td>no</td>
<td>59.94Hz (NTSC)</td>
<td>3840x2160</td>
<td>23.98p</td>
<td>100 Mbps</td>
<td>4:2:0/8 bit</td>
<td>Long GOP</td>
</tr>
<tr>
<td>[4K/72M/30p]</td>
<td>MP4 HEVC</td>
<td>no</td>
<td>yes</td>
<td>59.94Hz (NTSC)</td>
<td>3840x2160</td>
<td>29.97p</td>
<td>72 Mbps</td>
<td>4:2:0/10 bit</td>
<td>Long GOP</td>
</tr>
<tr>
<td>[4K/72M/24p]</td>
<td>MP4 HEVC</td>
<td>no</td>
<td>yes</td>
<td>59.94Hz (NTSC)</td>
<td>3840x2160</td>
<td>23.98p</td>
<td>72 Mbps</td>
<td>4:2:0/10 bit</td>
<td>Long GOP</td>
</tr>
<tr>
<td>[4K/100M/30p]</td>
<td>MP4</td>
<td>no</td>
<td>no</td>
<td>59.94Hz (NTSC)</td>
<td>3840x2160</td>
<td>29.97p</td>
<td>100 Mbps</td>
<td>4:2:0/8 bit</td>
<td>Long GOP</td>
</tr>
<tr>
<td>[4K/100M/24p]</td>
<td>MP4</td>
<td>no</td>
<td>no</td>
<td>59.94Hz (NTSC)</td>
<td>3840x2160</td>
<td>23.98p</td>
<td>100 Mbps</td>
<td>4:2:0/8 bit</td>
<td>Long GOP</td>
</tr>
<tr>
<td>[4K/A/150M/60p]</td>
<td>MP4 (LPCM) / MOV</td>
<td>no</td>
<td>no</td>
<td>59.94Hz (NTSC)</td>
<td>3328x2496</td>
<td>59.94p</td>
<td>150 Mbps</td>
<td>4:2:0/8 bit</td>
<td>Long GOP</td>
</tr>
<tr>
<td>[4K/A/400M/30p]</td>
<td>MP4 (LPCM) / MOV</td>
<td>no</td>
<td>yes</td>
<td>59.94Hz (NTSC)</td>
<td>3328x2496</td>
<td>29.97p</td>
<td>400 Mbps</td>
<td>4:2:2/10 bit</td>
<td>ALL-Intra</td>
</tr>
<tr>
<td>[4K/A/150M/30p]</td>
<td>MP4 (LPCM) / MOV</td>
<td>no</td>
<td>yes</td>
<td>59.94Hz (NTSC)</td>
<td>3328x2496</td>
<td>29.97p</td>
<td>150 Mbps</td>
<td>4:2:2/10 bit</td>
<td>Long GOP</td>
</tr>
<tr>
<td>[4K/A/100M/30p]</td>
<td>MP4 (LPCM) / MOV</td>
<td>no</td>
<td>no</td>
<td>59.94Hz (NTSC)</td>
<td>3328x2496</td>
<td>29.97p</td>
<td>100 Mbps</td>
<td>4:2:0/8 bit</td>
<td>Long GOP</td>
</tr>
<tr>
<td>[4K/A/400M/24p]</td>
<td>MP4 (LPCM) / MOV</td>
<td>no</td>
<td>yes</td>
<td>59.94Hz (NTSC)</td>
<td>3328x2496</td>
<td>23.98p</td>
<td>400 Mbps</td>
<td>4:2:2/10 bit</td>
<td>ALL-Intra</td>
</tr>
<tr>
<td>[4K/A/150M/24p]</td>
<td>MP4 (LPCM) / MOV</td>
<td>yes</td>
<td>no</td>
<td>59.94Hz (NTSC)</td>
<td>3328x2496</td>
<td>23.98p</td>
<td>150 Mbps</td>
<td>4:2:2/10 bit</td>
<td>Long GOP</td>
</tr>
<tr>
<td>File Size</td>
<td>Group of Pictures</td>
<td>Video Codec</td>
<td>Audio Codec</td>
<td>Video Size</td>
<td>Frame Rate</td>
<td>Bit Rate</td>
<td>Color Depth</td>
<td>GOP</td>
<td>Frame Rate Type</td>
</tr>
<tr>
<td>-----------</td>
<td>------------------</td>
<td>-------------</td>
<td>-------------</td>
<td>------------</td>
<td>------------</td>
<td>----------</td>
<td>-------------</td>
<td>-----</td>
<td>----------------</td>
</tr>
<tr>
<td>[4K/A/100M/24p]</td>
<td>MP4 (LPCM) / MOV no no 59.94 Hz (NTSC) 3328x2496 23.98p 100 Mbps 4:2:0/8 bit Long GOP</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>[FHD/28M/60p]</td>
<td>AVCHD no no 59.94 Hz (NTSC) 1920x1080 59.94p 28 Mbps 4:2:0/8 bit Long GOP</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>[FHD/17M/60i]</td>
<td>AVCHD no no 59.94 Hz (NTSC) 1920x1080 59.94i 17 Mbps 4:2:0/8 bit Long GOP</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>[FHD/24M/30p]</td>
<td>AVCHD yes no 59.94 Hz (NTSC) 1920x1080 59.94i 24 Mbps 4:2:0/8 bit Long GOP</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>[FHD/24M/24p]</td>
<td>AVCHD yes no 59.94 Hz (NTSC) 1920x1080 23.98p 24 Mbps 4:2:0/8 bit Long GOP</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>[FHD/ALL-I/200M/60p]</td>
<td>MP4 (LPCM) / MOV no yes 59.94 Hz (NTSC) 1920x1080 59.94p 200 Mbps 4:2:2/10 bit ALL-Intra</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>[FHD/10bit/100M/60p]</td>
<td>MP4 (LPCM) / MOV no yes 59.94 Hz (NTSC) 1920x1080 59.94p 100 Mbps 4:2:2/10 bit Long GOP</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>[FHD/ALL-I/200M/30p]</td>
<td>MP4 (LPCM) / MOV no yes 59.94 Hz (NTSC) 1920x1080 29.97p 200 Mbps 4:2:2/10 bit ALL-Intra</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>[FHD/10bit/100M/30p]</td>
<td>MP4 (LPCM) / MOV no yes 59.94 Hz (NTSC) 1920x1080 29.97p 100 Mbps 4:2:2/10 bit Long GOP</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>[FHD/ALL-I/200M/24p]</td>
<td>MP4 (LPCM) / MOV no yes 59.94 Hz (NTSC) 1920x1080 23.98p 200 Mbps 4:2:2/10 bit ALL-Intra</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>[FHD/10bit/100M/24p]</td>
<td>MP4 (LPCM) / MOV no yes 59.94 Hz (NTSC) 1920x1080 23.98p 100 Mbps 4:2:2/10 bit Long GOP</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>[C4K/8bit/100M/25p]</td>
<td>MP4 (LPCM) / MOV no no 50.00 Hz (PAL) 4096x2160 50.00p 100 Mbps 4:2:0/8 bit Long GOP</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>[C4K/10bit/150M/25p]</td>
<td>MP4 (LPCM) / MOV no yes 50.00 Hz (PAL) 4096x2160 25.00p 150 Mbps 4:2:2/10 bit Long GOP</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>[C4K/8bit/100M/25p]</td>
<td>MP4 (LPCM) / MOV no no 50.00 Hz (PAL) 4096x2160 25.00p 100 Mbps 4:2:0/8 bit Long GOP</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>[4K/8bit/150M/50p]</td>
<td>MP4 (LPCM) / MOV no no 50.00 Hz (PAL) 3840x2160 50.00p 150 Mbps 4:2:0/8 bit Long GOP</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>[4K/ALL-I/400M/25p]</td>
<td>MP4 (LPCM) / MOV no yes 50.00 Hz (PAL) 3840x2160 25.00p 400 Mbps 4:2:2/10 bit ALL-Intra</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>[4K/10bit/150M/25p]</td>
<td>MP4 (LPCM) / MOV no yes 50.00 Hz (PAL) 3840x2160 25.00p 150 Mbps 4:2:2/10 bit Long GOP</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>[4K/8bit/250M/25p]</td>
<td>MP4 (LPCM) / MOV yes no 50.00 Hz (PAL) 3840x2160 25.00p 100 Mbps 4:2:0/8 bit Long GOP</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>[4K/72M/25p]</td>
<td>MP4 HEVC no yes 50.00 Hz (PAL) 3840x2160 25.00p 72 Mbps 4:2:0/10 bit Long GOP</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>[4K/100M/25p]</td>
<td>MP4 no no 50.00 Hz (PAL) 3840x2160 25.00p 100 Mbps 4:2:0/8 bit Long GOP</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Format</td>
<td>Codec</td>
<td>Bit Rate</td>
<td>Frame Rate</td>
<td>Resolution</td>
<td>Data Rate</td>
<td>Bit Depth</td>
<td>GOP</td>
<td></td>
<td></td>
</tr>
<tr>
<td>--------</td>
<td>-------</td>
<td>----------</td>
<td>------------</td>
<td>------------</td>
<td>-----------</td>
<td>----------</td>
<td>-----</td>
<td></td>
<td></td>
</tr>
<tr>
<td>[4K/A/150M/50p]</td>
<td>MP4 (LPCM) / MOV</td>
<td>no</td>
<td>no</td>
<td>50.00Hz (PAL)</td>
<td>3328x2496</td>
<td>50.00p</td>
<td>150 Mbps</td>
<td>4:2:0/8 bit</td>
<td>Long GOP</td>
</tr>
<tr>
<td>[4K/A/400M/25p]</td>
<td>MP4 (LPCM) / MOV</td>
<td>no</td>
<td>yes</td>
<td>50.00Hz (PAL)</td>
<td>3328x2496</td>
<td>25.00p</td>
<td>400 Mbps</td>
<td>4:2:2/10 bit</td>
<td>ALL-Intra</td>
</tr>
<tr>
<td>[4K/A/150M/25p]</td>
<td>MP4 (LPCM) / MOV</td>
<td>no</td>
<td>yes</td>
<td>50.00Hz (PAL)</td>
<td>3328x2496</td>
<td>25.00p</td>
<td>150 Mbps</td>
<td>4:2:2/10 bit</td>
<td>Long GOP</td>
</tr>
<tr>
<td>[4K/A/100M/25p]</td>
<td>MP4 (LPCM) / MOV</td>
<td>no</td>
<td>no</td>
<td>50.00Hz (PAL)</td>
<td>3328x2496</td>
<td>25.00p</td>
<td>100 Mbps</td>
<td>4:2:0/8 bit</td>
<td>Long GOP</td>
</tr>
<tr>
<td>[FHD/28M/50p]</td>
<td>AVCHD</td>
<td>no</td>
<td>no</td>
<td>50.00Hz (PAL)</td>
<td>1920x1080</td>
<td>50.00p</td>
<td>28 Mbps</td>
<td>4:2:0/8 bit</td>
<td>Long GOP</td>
</tr>
<tr>
<td>[FHD/17M/50]</td>
<td>AVCHD</td>
<td>no</td>
<td>yes</td>
<td>50.00Hz (PAL)</td>
<td>1920x1080</td>
<td>50.00i</td>
<td>17 Mbps</td>
<td>4:2:0/8 bit</td>
<td>Long GOP</td>
</tr>
<tr>
<td>[FHD/24M/25p]</td>
<td>AVCHD</td>
<td>yes</td>
<td>no</td>
<td>50.00Hz (PAL)</td>
<td>1920x1080</td>
<td>50.00i</td>
<td>24 Mbps</td>
<td>4:2:0/8 bit</td>
<td>Long GOP</td>
</tr>
<tr>
<td>[FHD/ALL-I/200M/50p]</td>
<td>MP4 (LPCM) / MOV</td>
<td>no</td>
<td>yes</td>
<td>50.00Hz (PAL)</td>
<td>1920x1080</td>
<td>50.00p</td>
<td>200 Mbps</td>
<td>4:2:2/10 bit</td>
<td>ALL-Intra</td>
</tr>
<tr>
<td>[FHD/8bit/100M/50p]</td>
<td>MP4 (LPCM) / MOV</td>
<td>yes</td>
<td>no</td>
<td>50.00Hz (PAL)</td>
<td>1920x1080</td>
<td>50.00p</td>
<td>100 Mbps</td>
<td>4:2:0/8 bit</td>
<td>LongGOP</td>
</tr>
<tr>
<td>[FHD/10bit/100M/25p]</td>
<td>MP4 (LPCM) / MOV</td>
<td>no</td>
<td>yes</td>
<td>50.00Hz (PAL)</td>
<td>1920x1080</td>
<td>25.00p</td>
<td>100 Mbps</td>
<td>4:2:2/10 bit</td>
<td>Long GOP</td>
</tr>
<tr>
<td>[FHD/8bit/100M/25p]</td>
<td>MP4 (LPCM) / MOV</td>
<td>yes</td>
<td>no</td>
<td>50.00Hz (PAL)</td>
<td>1920x1080</td>
<td>25.00p</td>
<td>100 Mbps</td>
<td>4:2:0/8 bit</td>
<td>Long GOP</td>
</tr>
<tr>
<td>[FHD/28M/50p]</td>
<td>MP4</td>
<td>no</td>
<td>no</td>
<td>50.00Hz (PAL)</td>
<td>1920x1080</td>
<td>50.00p</td>
<td>28 Mbps</td>
<td>4:2:0/8 bit</td>
<td>Long GOP</td>
</tr>
<tr>
<td>[FHD/20M/25p]</td>
<td>MP4</td>
<td>no</td>
<td>no</td>
<td>50.00Hz (PAL)</td>
<td>1920x1080</td>
<td>25.00p</td>
<td>20 Mbps</td>
<td>4:2:0/8 bit</td>
<td>Long GOP</td>
</tr>
<tr>
<td>[C4K/ALL-I/400M/24p]</td>
<td>MP4 (LPCM) / MOV</td>
<td>no</td>
<td>yes</td>
<td>24.00Hz (CINEMA)</td>
<td>4096x2160</td>
<td>24.00p</td>
<td>400 Mbps</td>
<td>4:2:2/10 bit</td>
<td>ALL-Intra</td>
</tr>
<tr>
<td>[C4K/10bit/150M/24p]</td>
<td>MP4 (LPCM) / MOV</td>
<td>no</td>
<td>yes</td>
<td>24.00Hz (CINEMA)</td>
<td>4096x2160</td>
<td>24.00p</td>
<td>150 Mbps</td>
<td>4:2:2/10 bit</td>
<td>Long GOP</td>
</tr>
<tr>
<td>[C4K/8bit/100M/24p]</td>
<td>MP4 (LPCM) / MOV</td>
<td>yes</td>
<td>no</td>
<td>24.00Hz (CINEMA)</td>
<td>4096x2160</td>
<td>24.00p</td>
<td>100 Mbps</td>
<td>4:2:0/8 bit</td>
<td>Long GOP</td>
</tr>
<tr>
<td>[4K/ALL-I/400M/24p]</td>
<td>MP4 (LPCM) / MOV</td>
<td>no</td>
<td>yes</td>
<td>24.00Hz (CINEMA)</td>
<td>3840x2160</td>
<td>24.00p</td>
<td>400 Mbps</td>
<td>4:2:2/10 bit</td>
<td>ALL-Intra</td>
</tr>
<tr>
<td>[4K/10bit/150M/24p]</td>
<td>MP4 (LPCM) / MOV</td>
<td>no</td>
<td>yes</td>
<td>24.00Hz (CINEMA)</td>
<td>3840x2160</td>
<td>24.00p</td>
<td>150 Mbps</td>
<td>4:2:2/10 bit</td>
<td>Long GOP</td>
</tr>
<tr>
<td>[4K/8bit/100M/24p]</td>
<td>MP4 (LPCM) / MOV</td>
<td>yes</td>
<td>no</td>
<td>24.00Hz (CINEMA)</td>
<td>3840x2160</td>
<td>24.00p</td>
<td>100 Mbps</td>
<td>4:2:0/8 bit</td>
<td>Long GOP</td>
</tr>
<tr>
<td>[4K/A/400M/24p]</td>
<td>MP4 (LPCM) / MOV</td>
<td>no</td>
<td>yes</td>
<td>24.00Hz (CINEMA)</td>
<td>3328x2496</td>
<td>24.00p</td>
<td>400 Mbps</td>
<td>4:2:2/10 bit</td>
<td>ALL-Intra</td>
</tr>
<tr>
<td>[4K/A/150M/24p]</td>
<td>MP4 (LPCM) / MOV</td>
<td>no</td>
<td>yes</td>
<td>24.00Hz (CINEMA)</td>
<td>3328x2496</td>
<td>24.00p</td>
<td>150 Mbps</td>
<td>4:2:2/10 bit</td>
<td>Long GOP</td>
</tr>
<tr>
<td>[4K/A/100M/24p]</td>
<td>MP4 (LPCM) / MOV</td>
<td>no</td>
<td>no</td>
<td>24.00Hz (CINEMA)</td>
<td>3328x2496</td>
<td>24.00p</td>
<td>100 Mbps</td>
<td>4:2:0/8 bit</td>
<td>Long GOP</td>
</tr>
<tr>
<td>[FHD/ALL-I/200M/24p]</td>
<td>MP4 (LPCM) / MOV</td>
<td>no</td>
<td>yes</td>
<td>24.00Hz (CINEMA)</td>
<td>1920x1080</td>
<td>24.00p</td>
<td>200 Mbps</td>
<td>4:2:2/10 bit</td>
<td>ALL-Intra</td>
</tr>
<tr>
<td>[FHD/10bit/100M/24p]</td>
<td>MP4 (LPCM) / MOV</td>
<td>yes</td>
<td>no</td>
<td>24.00Hz (CINEMA)</td>
<td>1920x1080</td>
<td>24.00p</td>
<td>100 Mbps</td>
<td>4:2:2/10 bit</td>
<td>Long GOP</td>
</tr>
</tbody>
</table>

210
## 15.17 GH5S, all C4K 8 bit modes

<table>
<thead>
<tr>
<th>Rec Quality</th>
<th>Rec Format</th>
<th>VFR</th>
<th>HLG</th>
<th>System frequency</th>
<th>Size</th>
<th>Frame rate</th>
<th>Bit rate</th>
<th>YUV/bit</th>
<th>Image compression</th>
</tr>
</thead>
<tbody>
<tr>
<td>[C4K/8bit/150M/60p]</td>
<td>MP4 (LPCM) / MOV</td>
<td>no</td>
<td>no</td>
<td>59.94Hz (NTSC)</td>
<td>4096x2160</td>
<td>59.94p</td>
<td>150 Mbps</td>
<td>4:2:0/8 bit</td>
<td>Long GOP</td>
</tr>
<tr>
<td>[C4K/8bit/100M/30p]</td>
<td>MP4 (LPCM) / MOV</td>
<td>no</td>
<td>no</td>
<td>59.94Hz (NTSC)</td>
<td>4096x2160</td>
<td>29.97p</td>
<td>100 Mbps</td>
<td>4:2:0/8 bit</td>
<td>Long GOP</td>
</tr>
<tr>
<td>[C4K/8bit/100M/24p]</td>
<td>MP4 (LPCM) / MOV</td>
<td>no</td>
<td>no</td>
<td>59.94Hz (NTSC)</td>
<td>4096x2160</td>
<td>23.98p</td>
<td>100 Mbps</td>
<td>4:2:0/8 bit</td>
<td>Long GOP</td>
</tr>
<tr>
<td>[C4K/8bit/150M/50p]</td>
<td>MP4 (LPCM) / MOV</td>
<td>no</td>
<td>no</td>
<td>50.00Hz (PAL)</td>
<td>4096x2160</td>
<td>50.00p</td>
<td>150 Mbps</td>
<td>4:2:0/8 bit</td>
<td>Long GOP</td>
</tr>
<tr>
<td>[C4K/8bit/100M/25p]</td>
<td>MP4 (LPCM) / MOV</td>
<td>no</td>
<td>yes</td>
<td>50.00Hz (PAL)</td>
<td>4096x2160</td>
<td>25.00p</td>
<td>100 Mbps</td>
<td>4:2:0/8 bit</td>
<td>Long GOP</td>
</tr>
<tr>
<td>[C4K/8bit/100M/24p]</td>
<td>MP4 (LPCM) / MOV</td>
<td>yes</td>
<td>no</td>
<td>24.00Hz (CINEMA)</td>
<td>4096x2160</td>
<td>24.00p</td>
<td>100 Mbps</td>
<td>4:2:0/8 bit</td>
<td>Long GOP</td>
</tr>
</tbody>
</table>

## 15.18 GH5S, all C4K 10 bit modes

<table>
<thead>
<tr>
<th>Rec Quality</th>
<th>Rec Format</th>
<th>VFR</th>
<th>HLG</th>
<th>System frequency</th>
<th>Size</th>
<th>Frame rate</th>
<th>Bit rate</th>
<th>YUV/bit</th>
<th>Image compression</th>
</tr>
</thead>
<tbody>
<tr>
<td>[C4K/10bit/150M/30p]</td>
<td>MP4 (LPCM) / MOV</td>
<td>no</td>
<td>yes</td>
<td>59.94Hz (NTSC)</td>
<td>4096x2160</td>
<td>29.97p</td>
<td>150 Mbps</td>
<td>4:2:2/10 bit</td>
<td>Long GOP</td>
</tr>
<tr>
<td>[C4K/ALL-I/400M/24p]</td>
<td>MP4 (LPCM) / MOV</td>
<td>no</td>
<td>yes</td>
<td>59.94Hz (NTSC)</td>
<td>4096x2160</td>
<td>23.98p</td>
<td>400 Mbps</td>
<td>4:2:2/10 bit</td>
<td>ALL-Intra</td>
</tr>
<tr>
<td>[C4K/10bit/150M/24p]</td>
<td>MP4 (LPCM) / MOV</td>
<td>no</td>
<td>yes</td>
<td>59.94Hz (NTSC)</td>
<td>4096x2160</td>
<td>23.98p</td>
<td>150 Mbps</td>
<td>4:2:2/10 bit</td>
<td>Long GOP</td>
</tr>
<tr>
<td>[C4K/10bit/150M/25p]</td>
<td>MP4 (LPCM) / MOV</td>
<td>no</td>
<td>yes</td>
<td>50.00Hz (PAL)</td>
<td>4096x2160</td>
<td>25.00p</td>
<td>150 Mbps</td>
<td>4:2:2/10 bit</td>
<td>Long GOP</td>
</tr>
<tr>
<td>[C4K/ALL-I/400M/24p]</td>
<td>MP4 (LPCM) / MOV</td>
<td>no</td>
<td>yes</td>
<td>24.00Hz (CINEMA)</td>
<td>4096x2160</td>
<td>24.00p</td>
<td>400 Mbps</td>
<td>4:2:2/10 bit</td>
<td>ALL-Intra</td>
</tr>
<tr>
<td>[C4K/10bit/150M/24p]</td>
<td>MP4 (LPCM) / MOV</td>
<td>no</td>
<td>yes</td>
<td>24.00Hz (CINEMA)</td>
<td>4096x2160</td>
<td>24.00p</td>
<td>150 Mbps</td>
<td>4:2:2/10 bit</td>
<td>Long GOP</td>
</tr>
</tbody>
</table>
### GH5S, all 4K 8 bit modes

<table>
<thead>
<tr>
<th>Rec Quality</th>
<th>Rec Format</th>
<th>VFR</th>
<th>HLG</th>
<th>System frequency</th>
<th>Size</th>
<th>Frame rate</th>
<th>Bit rate</th>
<th>YUV/bit</th>
<th>Image compression</th>
</tr>
</thead>
<tbody>
<tr>
<td>[4K/8bit/150M/60p]</td>
<td>MP4 (LPCM) / MOV</td>
<td>no</td>
<td>no</td>
<td>59.94Hz (NTSC)</td>
<td>3840x2160</td>
<td>59.94p</td>
<td>150 Mbps</td>
<td>4:2:0/8 bit</td>
<td>Long GOP</td>
</tr>
<tr>
<td>[4K/8bit/100M/30p]</td>
<td>MP4 (LPCM) / MOV</td>
<td>yes</td>
<td>no</td>
<td>59.94Hz (NTSC)</td>
<td>3840x2160</td>
<td>29.97p</td>
<td>100 Mbps</td>
<td>4:2:0/8 bit</td>
<td>Long GOP</td>
</tr>
<tr>
<td>[4K/8bit/100M/24p]</td>
<td>MP4 (LPCM) / MOV</td>
<td>yes</td>
<td>no</td>
<td>59.94Hz (NTSC)</td>
<td>3840x2160</td>
<td>23.98p</td>
<td>100 Mbps</td>
<td>4:2:0/8 bit</td>
<td>Long GOP</td>
</tr>
<tr>
<td>[4K/100M/30p]</td>
<td>MP4</td>
<td>no</td>
<td>no</td>
<td>59.94Hz (NTSC)</td>
<td>3840x2160</td>
<td>29.97p</td>
<td>100 Mbps</td>
<td>4:2:0/8 bit</td>
<td>Long GOP</td>
</tr>
<tr>
<td>[4K/100M/24p]</td>
<td>MP4</td>
<td>no</td>
<td>no</td>
<td>59.94Hz (NTSC)</td>
<td>3840x2160</td>
<td>23.98p</td>
<td>100 Mbps</td>
<td>4:2:0/8 bit</td>
<td>Long GOP</td>
</tr>
<tr>
<td>[4K/8bit/150M/50p]</td>
<td>MP4 (LPCM) / MOV</td>
<td>no</td>
<td>no</td>
<td>50.00Hz (PAL)</td>
<td>3840x2160</td>
<td>50.00p</td>
<td>150 Mbps</td>
<td>4:2:0/8 bit</td>
<td>Long GOP</td>
</tr>
<tr>
<td>[4K/8bit/100M/25p]</td>
<td>MP4 (LPCM) / MOV</td>
<td>yes</td>
<td>no</td>
<td>50.00Hz (PAL)</td>
<td>3840x2160</td>
<td>25.00p</td>
<td>100 Mbps</td>
<td>4:2:0/8 bit</td>
<td>Long GOP</td>
</tr>
<tr>
<td>[4K/100M/25p]</td>
<td>MP4</td>
<td>no</td>
<td>no</td>
<td>50.00Hz (PAL)</td>
<td>3840x2160</td>
<td>25.00p</td>
<td>100 Mbps</td>
<td>4:2:0/8 bit</td>
<td>Long GOP</td>
</tr>
<tr>
<td>[4K/8bit/100M/24p]</td>
<td>MP4 (LPCM) / MOV</td>
<td>yes</td>
<td>no</td>
<td>24.00Hz (CINEMA)</td>
<td>3840x2160</td>
<td>24.00p</td>
<td>100 Mbps</td>
<td>4:2:0/8 bit</td>
<td>Long GOP</td>
</tr>
</tbody>
</table>
## 15.20 GH5S, all 4K 10 bit modes

<table>
<thead>
<tr>
<th>Rec Quality</th>
<th>Rec Format</th>
<th>VFR</th>
<th>HLG</th>
<th>System frequency</th>
<th>Size</th>
<th>Frame rate</th>
<th>Bit rate</th>
<th>YUV/bit</th>
<th>Image compression</th>
</tr>
</thead>
<tbody>
<tr>
<td>[4K/ALL-I/400M/30p]</td>
<td>MP4 (LPCM) / MOV</td>
<td>no</td>
<td>yes</td>
<td>59.94Hz (NTSC)</td>
<td>3840x2160</td>
<td>29.97p</td>
<td>400 Mbps</td>
<td>4:2:10 bit</td>
<td>ALL-Intra</td>
</tr>
<tr>
<td>[4K/10bit/150M/30p]</td>
<td>MP4 (LPCM) / MOV</td>
<td>no</td>
<td>yes</td>
<td>59.94Hz (NTSC)</td>
<td>3840x2160</td>
<td>29.97p</td>
<td>150 Mbps</td>
<td>4:2:10 bit</td>
<td>Long GOP</td>
</tr>
<tr>
<td>[4K/ALL-I/400M/24p]</td>
<td>MP4 (LPCM) / MOV</td>
<td>no</td>
<td>yes</td>
<td>59.94Hz (NTSC)</td>
<td>3840x2160</td>
<td>23.98p</td>
<td>400 Mbps</td>
<td>4:2:10 bit</td>
<td>ALL-Intra</td>
</tr>
<tr>
<td>[4K/10bit/150M/24p]</td>
<td>MP4 (LPCM) / MOV</td>
<td>no</td>
<td>yes</td>
<td>59.94Hz (NTSC)</td>
<td>3840x2160</td>
<td>23.98p</td>
<td>150 Mbps</td>
<td>4:2:10 bit</td>
<td>Long GOP</td>
</tr>
<tr>
<td>[4K/72M/30p]</td>
<td>MP4 HEVC</td>
<td>no</td>
<td>yes</td>
<td>59.94Hz (NTSC)</td>
<td>3840x2160</td>
<td>29.97p</td>
<td>72 Mbps</td>
<td>4:2:0/10 bit</td>
<td>Long GOP</td>
</tr>
<tr>
<td>[4K/72M/24p]</td>
<td>MP4 HEVC</td>
<td>no</td>
<td>yes</td>
<td>59.94Hz (NTSC)</td>
<td>3840x2160</td>
<td>23.98p</td>
<td>72 Mbps</td>
<td>4:2:0/10 bit</td>
<td>Long GOP</td>
</tr>
<tr>
<td>[4K/ALL-I/400M/25p]</td>
<td>MP4 (LPCM) / MOV</td>
<td>no</td>
<td>yes</td>
<td>50.00Hz (PAL)</td>
<td>3840x2160</td>
<td>25.00p</td>
<td>400 Mbps</td>
<td>4:2:10 bit</td>
<td>ALL-Intra</td>
</tr>
<tr>
<td>[4K/10bit/150M/25p]</td>
<td>MP4 (LPCM) / MOV</td>
<td>no</td>
<td>yes</td>
<td>50.00Hz (PAL)</td>
<td>3840x2160</td>
<td>25.00p</td>
<td>150 Mbps</td>
<td>4:2:10 bit</td>
<td>Long GOP</td>
</tr>
<tr>
<td>[4K/72M/25p]</td>
<td>MP4 HEVC</td>
<td>no</td>
<td>yes</td>
<td>50.00Hz (PAL)</td>
<td>3840x2160</td>
<td>25.00p</td>
<td>72 Mbps</td>
<td>4:2:0/10 bit</td>
<td>Long GOP</td>
</tr>
<tr>
<td>[4K/ALL-I/400M/24p]</td>
<td>MP4 (LPCM) / MOV</td>
<td>no</td>
<td>yes</td>
<td>24.00Hz (CINEMA)</td>
<td>3840x2160</td>
<td>24.00p</td>
<td>400 Mbps</td>
<td>4:2:10 bit</td>
<td>ALL-Intra</td>
</tr>
<tr>
<td>[4K/10bit/150M/24p]</td>
<td>MP4 (LPCM) / MOV</td>
<td>no</td>
<td>yes</td>
<td>24.00Hz (CINEMA)</td>
<td>3840x2160</td>
<td>24.00p</td>
<td>150 Mbps</td>
<td>4:2:10 bit</td>
<td>Long GOP</td>
</tr>
</tbody>
</table>

## 15.21 GH5S, all anamorphic 8 bit modes

<table>
<thead>
<tr>
<th>Rec Quality</th>
<th>Rec Format</th>
<th>VFR</th>
<th>HLG</th>
<th>System frequency</th>
<th>Size</th>
<th>Frame rate</th>
<th>Bit rate</th>
<th>YUV/bit</th>
<th>Image compression</th>
</tr>
</thead>
<tbody>
<tr>
<td>[4K/A/150M/60p]</td>
<td>MP4 (LPCM) / MOV</td>
<td>no</td>
<td>no</td>
<td>59.94Hz (NTSC)</td>
<td>3328x2496</td>
<td>59.94p</td>
<td>150 Mbps</td>
<td>4:2:0/8 bit</td>
<td>Long GOP</td>
</tr>
<tr>
<td>[4K/A/100M/30p]</td>
<td>MP4 (LPCM) / MOV</td>
<td>no</td>
<td>no</td>
<td>59.94Hz (NTSC)</td>
<td>3328x2496</td>
<td>29.97p</td>
<td>100 Mbps</td>
<td>4:2:0/8 bit</td>
<td>Long GOP</td>
</tr>
<tr>
<td>[4K/A/100M/24p]</td>
<td>MP4 (LPCM) / MOV</td>
<td>no</td>
<td>no</td>
<td>59.94Hz (NTSC)</td>
<td>3328x2496</td>
<td>23.98p</td>
<td>100 Mbps</td>
<td>4:2:0/8 bit</td>
<td>Long GOP</td>
</tr>
<tr>
<td>[4K/A/150M/50p]</td>
<td>MP4 (LPCM) / MOV</td>
<td>no</td>
<td>no</td>
<td>50.00Hz (PAL)</td>
<td>3328x2496</td>
<td>50.00p</td>
<td>150 Mbps</td>
<td>4:2:0/8 bit</td>
<td>Long GOP</td>
</tr>
<tr>
<td>[4K/A/100M/25p]</td>
<td>MP4 (LPCM) / MOV</td>
<td>no</td>
<td>no</td>
<td>50.00Hz (PAL)</td>
<td>3328x2496</td>
<td>25.00p</td>
<td>100 Mbps</td>
<td>4:2:0/8 bit</td>
<td>Long GOP</td>
</tr>
<tr>
<td>[4K/A/100M/24p]</td>
<td>MP4 (LPCM) / MOV</td>
<td>no</td>
<td>no</td>
<td>24.00Hz (CINEMA)</td>
<td>3328x2496</td>
<td>24.00p</td>
<td>100 Mbps</td>
<td>4:2:0/8 bit</td>
<td>Long GOP</td>
</tr>
</tbody>
</table>
### GH5S, all anamorphic 10 bit modes

<table>
<thead>
<tr>
<th>Rec Quality</th>
<th>Rec Format</th>
<th>VFR</th>
<th>HLG</th>
<th>System frequency</th>
<th>Size</th>
<th>Frame rate</th>
<th>Bit rate</th>
<th>YUV/bit</th>
<th>Image compression</th>
</tr>
</thead>
<tbody>
<tr>
<td>[4K/A/400M/30p]</td>
<td>MP4 (LPCM) / MOV</td>
<td>no</td>
<td>yes</td>
<td>59.94Hz (NTSC)</td>
<td>3328x2496</td>
<td>29.97p</td>
<td>400 Mbps</td>
<td>4:2:2/10 bit</td>
<td>ALL-Intra</td>
</tr>
<tr>
<td>[4K/A/150M/30p]</td>
<td>MP4 (LPCM) / MOV</td>
<td>no</td>
<td>yes</td>
<td>59.94Hz (NTSC)</td>
<td>3328x2496</td>
<td>29.97p</td>
<td>150 Mbps</td>
<td>4:2:2/10 bit</td>
<td>Long GOP</td>
</tr>
<tr>
<td>[4K/A/400M/24p]</td>
<td>MP4 (LPCM) / MOV</td>
<td>no</td>
<td>yes</td>
<td>59.94Hz (NTSC)</td>
<td>3328x2496</td>
<td>23.98p</td>
<td>400 Mbps</td>
<td>4:2:2/10 bit</td>
<td>ALL-Intra</td>
</tr>
<tr>
<td>[4K/A/150M/24p]</td>
<td>MP4 (LPCM) / MOV</td>
<td>no</td>
<td>yes</td>
<td>59.94Hz (NTSC)</td>
<td>3328x2496</td>
<td>23.98p</td>
<td>150 Mbps</td>
<td>4:2:2/10 bit</td>
<td>Long GOP</td>
</tr>
<tr>
<td>[4K/A/400M/25p]</td>
<td>MP4 (LPCM) / MOV</td>
<td>no</td>
<td>yes</td>
<td>50.00Hz (PAL)</td>
<td>3328x2496</td>
<td>25.00p</td>
<td>400 Mbps</td>
<td>4:2:2/10 bit</td>
<td>ALL-Intra</td>
</tr>
<tr>
<td>[4K/A/150M/25p]</td>
<td>MP4 (LPCM) / MOV</td>
<td>no</td>
<td>yes</td>
<td>50.00Hz (PAL)</td>
<td>3328x2496</td>
<td>25.00p</td>
<td>150 Mbps</td>
<td>4:2:2/10 bit</td>
<td>Long GOP</td>
</tr>
<tr>
<td>[4K/A/400M/24p]</td>
<td>MP4 (LPCM) / MOV</td>
<td>no</td>
<td>yes</td>
<td>24.00Hz (CINEMA)</td>
<td>3328x2496</td>
<td>24.00p</td>
<td>400 Mbps</td>
<td>4:2:2/10 bit</td>
<td>ALL-Intra</td>
</tr>
<tr>
<td>[4K/A/150M/24p]</td>
<td>MP4 (LPCM) / MOV</td>
<td>no</td>
<td>yes</td>
<td>24.00Hz (CINEMA)</td>
<td>3328x2496</td>
<td>24.00p</td>
<td>150 Mbps</td>
<td>4:2:2/10 bit</td>
<td>Long GOP</td>
</tr>
</tbody>
</table>
### 15.23 GH5S, all FHD 8 bit modes

<table>
<thead>
<tr>
<th>Rec Quality</th>
<th>Rec Format</th>
<th>VFR</th>
<th>HLG</th>
<th>System frequency</th>
<th>Size</th>
<th>Frame rate</th>
<th>Bit rate</th>
<th>YUV/bit</th>
<th>Image compression</th>
</tr>
</thead>
<tbody>
<tr>
<td>[FHD/28M/60p]</td>
<td>AVCHD</td>
<td>no</td>
<td>no</td>
<td>59.94Hz (NTSC)</td>
<td>1920x1080</td>
<td>59.94p</td>
<td>28 Mbps</td>
<td>4:2:0/8 bit</td>
<td>Long GOP</td>
</tr>
<tr>
<td>[FHD/24M/60]</td>
<td>AVCHD</td>
<td>no</td>
<td>no</td>
<td>59.94Hz (NTSC)</td>
<td>1920x1080</td>
<td>59.94i</td>
<td>17 Mbps</td>
<td>4:2:0/8 bit</td>
<td>Long GOP</td>
</tr>
<tr>
<td>[FHD/24M/30p]</td>
<td>AVCHD</td>
<td>yes</td>
<td>no</td>
<td>59.94Hz (NTSC)</td>
<td>1920x1080</td>
<td>59.94i</td>
<td>24 Mbps</td>
<td>4:2:0/8 bit</td>
<td>Long GOP</td>
</tr>
<tr>
<td>[FHD/24M/24p]</td>
<td>AVCHD</td>
<td>yes</td>
<td>no</td>
<td>59.94Hz (NTSC)</td>
<td>1920x1080</td>
<td>24.00p</td>
<td>24 Mbps</td>
<td>4:2:0/8 bit</td>
<td>Long GOP</td>
</tr>
<tr>
<td>[FHD/8bit/100M/60p]</td>
<td>MP4 (LPCM) / MOV</td>
<td>yes</td>
<td>no</td>
<td>59.94Hz (NTSC)</td>
<td>1920x1080</td>
<td>59.94p</td>
<td>100 Mbps</td>
<td>4:2:0/8 bit</td>
<td>Long GOP</td>
</tr>
<tr>
<td>[FHD/8bit/100M/30p]</td>
<td>MP4 (LPCM) / MOV</td>
<td>yes</td>
<td>no</td>
<td>59.94Hz (NTSC)</td>
<td>1920x1080</td>
<td>29.97p</td>
<td>100 Mbps</td>
<td>4:2:0/8 bit</td>
<td>Long GOP</td>
</tr>
<tr>
<td>[FHD/8bit/100M/25p]</td>
<td>MP4 (LPCM) / MOV</td>
<td>yes</td>
<td>no</td>
<td>59.94Hz (NTSC)</td>
<td>1920x1080</td>
<td>25.00p</td>
<td>100 Mbps</td>
<td>4:2:0/8 bit</td>
<td>Long GOP</td>
</tr>
<tr>
<td>[FHD/28M/60p]</td>
<td>MP4</td>
<td>no</td>
<td>no</td>
<td>59.94Hz (NTSC)</td>
<td>1920x1080</td>
<td>59.94p</td>
<td>28 Mbps</td>
<td>4:2:0/8 bit</td>
<td>Long GOP</td>
</tr>
<tr>
<td>[FHD/20M/30p]</td>
<td>MP4</td>
<td>no</td>
<td>no</td>
<td>59.94Hz (NTSC)</td>
<td>1920x1080</td>
<td>29.97p</td>
<td>20 Mbps</td>
<td>4:2:0/8 bit</td>
<td>Long GOP</td>
</tr>
<tr>
<td>[FHD/24M/24p]</td>
<td>MP4</td>
<td>no</td>
<td>no</td>
<td>59.94Hz (NTSC)</td>
<td>1920x1080</td>
<td>24.00p</td>
<td>24 Mbps</td>
<td>4:2:0/8 bit</td>
<td>Long GOP</td>
</tr>
<tr>
<td>[FHD/24M/50p]</td>
<td>AVCHD</td>
<td>no</td>
<td>no</td>
<td>50.00Hz (PAL)</td>
<td>1920x1080</td>
<td>50.00p</td>
<td>28 Mbps</td>
<td>4:2:0/8 bit</td>
<td>Long GOP</td>
</tr>
<tr>
<td>[FHD/24M/25p]</td>
<td>AVCHD</td>
<td>yes</td>
<td>no</td>
<td>50.00Hz (PAL)</td>
<td>1920x1080</td>
<td>50.00i</td>
<td>24 Mbps</td>
<td>4:2:0/8 bit</td>
<td>Long GOP</td>
</tr>
<tr>
<td>[FHD/8bit/100M/50p]</td>
<td>MP4 (LPCM) / MOV</td>
<td>yes</td>
<td>no</td>
<td>50.00Hz (PAL)</td>
<td>1920x1080</td>
<td>50.00p</td>
<td>100 Mbps</td>
<td>4:2:0/8 bit</td>
<td>Long GOP</td>
</tr>
<tr>
<td>[FHD/8bit/100M/25p]</td>
<td>MP4 (LPCM) / MOV</td>
<td>yes</td>
<td>no</td>
<td>50.00Hz (PAL)</td>
<td>1920x1080</td>
<td>25.00p</td>
<td>100 Mbps</td>
<td>4:2:0/8 bit</td>
<td>Long GOP</td>
</tr>
<tr>
<td>[FHD/28M/50p]</td>
<td>MP4</td>
<td>no</td>
<td>no</td>
<td>50.00Hz (PAL)</td>
<td>1920x1080</td>
<td>50.00p</td>
<td>28 Mbps</td>
<td>4:2:0/8 bit</td>
<td>Long GOP</td>
</tr>
<tr>
<td>[FHD/20M/25p]</td>
<td>MP4</td>
<td>no</td>
<td>no</td>
<td>50.00Hz (PAL)</td>
<td>1920x1080</td>
<td>25.00p</td>
<td>20 Mbps</td>
<td>4:2:0/8 bit</td>
<td>Long GOP</td>
</tr>
<tr>
<td>[FHD/8bit/100M/24p]</td>
<td>MP4 (LPCM) / MOV</td>
<td>yes</td>
<td>no</td>
<td>24.00Hz (CINEMA)</td>
<td>1920x1080</td>
<td>24.00p</td>
<td>100 Mbps</td>
<td>4:2:0/8 bit</td>
<td>Long GOP</td>
</tr>
<tr>
<td>[FHD/24M/24p]</td>
<td>MP4</td>
<td>no</td>
<td>no</td>
<td>24.00Hz (CINEMA)</td>
<td>1920x1080</td>
<td>24.00p</td>
<td>24 Mbps</td>
<td>4:2:0/8 bit</td>
<td>Long GOP</td>
</tr>
</tbody>
</table>
## GH5S, all FHD 10 bit modes

<table>
<thead>
<tr>
<th>Rec Quality</th>
<th>Rec Format</th>
<th>VFR</th>
<th>HLG</th>
<th>System frequency</th>
<th>Size</th>
<th>Frame rate</th>
<th>Bit rate</th>
<th>YUV/bit</th>
<th>Image compression</th>
</tr>
</thead>
<tbody>
<tr>
<td>[FHD/ALL-I/200M/60p]</td>
<td>MP4 (LPCM) / MOV</td>
<td>no</td>
<td>yes</td>
<td>59.94Hz (NTSC)</td>
<td>1920x1080</td>
<td>59.94p</td>
<td>200 Mbps</td>
<td>4:2:2/10 bit</td>
<td>ALL-Intra</td>
</tr>
<tr>
<td>[FHD/10bit/100M/60p]</td>
<td>MP4 (LPCM) / MOV</td>
<td>no</td>
<td>yes</td>
<td>59.94Hz (NTSC)</td>
<td>1920x1080</td>
<td>59.94p</td>
<td>100 Mbps</td>
<td>4:2:2/10 bit</td>
<td>Long GOP</td>
</tr>
<tr>
<td>[FHD/ALL-I/200M/30p]</td>
<td>MP4 (LPCM) / MOV</td>
<td>no</td>
<td>yes</td>
<td>59.94Hz (NTSC)</td>
<td>1920x1080</td>
<td>29.97p</td>
<td>200 Mbps</td>
<td>4:2:2/10 bit</td>
<td>ALL-Intra</td>
</tr>
<tr>
<td>[FHD/10bit/100M/30p]</td>
<td>MP4 (LPCM) / MOV</td>
<td>no</td>
<td>yes</td>
<td>59.94Hz (NTSC)</td>
<td>1920x1080</td>
<td>29.97p</td>
<td>100 Mbps</td>
<td>4:2:2/10 bit</td>
<td>Long GOP</td>
</tr>
<tr>
<td>[FHD/ALL-I/200M/24p]</td>
<td>MP4 (LPCM) / MOV</td>
<td>no</td>
<td>yes</td>
<td>59.94Hz (NTSC)</td>
<td>1920x1080</td>
<td>23.98p</td>
<td>200 Mbps</td>
<td>4:2:2/10 bit</td>
<td>ALL-Intra</td>
</tr>
<tr>
<td>[FHD/10bit/100M/24p]</td>
<td>MP4 (LPCM) / MOV</td>
<td>no</td>
<td>yes</td>
<td>59.94Hz (NTSC)</td>
<td>1920x1080</td>
<td>23.98p</td>
<td>100 Mbps</td>
<td>4:2:2/10 bit</td>
<td>Long GOP</td>
</tr>
<tr>
<td>[FHD/ALL-I/200M/50p]</td>
<td>MP4 (LPCM) / MOV</td>
<td>no</td>
<td>yes</td>
<td>50.00Hz (PAL)</td>
<td>1920x1080</td>
<td>50.00p</td>
<td>200 Mbps</td>
<td>4:2:2/10 bit</td>
<td>ALL-Intra</td>
</tr>
<tr>
<td>[FHD/10bit/100M/50p]</td>
<td>MP4 (LPCM) / MOV</td>
<td>no</td>
<td>yes</td>
<td>50.00Hz (PAL)</td>
<td>1920x1080</td>
<td>50.00p</td>
<td>100 Mbps</td>
<td>4:2:2/10 bit</td>
<td>Long GOP</td>
</tr>
<tr>
<td>[FHD/ALL-I/200M/25p]</td>
<td>MP4 (LPCM) / MOV</td>
<td>no</td>
<td>yes</td>
<td>50.00Hz (PAL)</td>
<td>1920x1080</td>
<td>25.00p</td>
<td>200 Mbps</td>
<td>4:2:2/10 bit</td>
<td>ALL-Intra</td>
</tr>
<tr>
<td>[FHD/10bit/100M/25p]</td>
<td>MP4 (LPCM) / MOV</td>
<td>no</td>
<td>yes</td>
<td>50.00Hz (PAL)</td>
<td>1920x1080</td>
<td>25.00p</td>
<td>100 Mbps</td>
<td>4:2:2/10 bit</td>
<td>Long GOP</td>
</tr>
<tr>
<td>[FHD/ALL-I/200M/24p]</td>
<td>MP4 (LPCM) / MOV</td>
<td>no</td>
<td>yes</td>
<td>24.00Hz (CINEMA)</td>
<td>1920x1080</td>
<td>24.00p</td>
<td>200 Mbps</td>
<td>4:2:2/10 bit</td>
<td>ALL-Intra</td>
</tr>
<tr>
<td>[FHD/10bit/100M/24p]</td>
<td>MP4 (LPCM) / MOV</td>
<td>no</td>
<td>yes</td>
<td>24.00Hz (CINEMA)</td>
<td>1920x1080</td>
<td>24.00p</td>
<td>100 Mbps</td>
<td>4:2:2/10 bit</td>
<td>Long GOP</td>
</tr>
</tbody>
</table>
16 PanoView XDV360 camera

This is a very cheap Chinese camera with a 200° 1.1mm f/2.0 fisheye lens.
Some hints for using:

Change the mode (Video / Photo / Timelapse / Settings) by short pressing the "on/off" button.
You can go directly to the settings by pressing the "arrow down" button.
Scroll in the settings with "arrow down" and "arrow up" buttons.
Switch to the right for the next menu with the "on/off" button.
Select and confirm with "start/stop" button.
Possible square video resolutions: 2448 / 2048 / 1440 / 1072 with 30 fps or 1440 / 1072 with 60 fps
Recommended exposure correction for video:
-- If the sun is in the field of view, use 0
-- In the woods, but sun is not directly visible: use 0 to +3
-- If in doubt, you aren't wrong if you use 0.

Table with crop values for different field of view:

<table>
<thead>
<tr>
<th>Field of View</th>
<th>Top and left border</th>
<th>Width and height</th>
</tr>
</thead>
<tbody>
<tr>
<td>180°</td>
<td>176</td>
<td>2104</td>
</tr>
<tr>
<td>185°</td>
<td>144</td>
<td>2168</td>
</tr>
<tr>
<td>190°</td>
<td>116</td>
<td>2224</td>
</tr>
<tr>
<td>195°</td>
<td>88</td>
<td>2280</td>
</tr>
<tr>
<td>200°</td>
<td>60</td>
<td>2336</td>
</tr>
</tbody>
</table>
This is a small camera with a 235° 0.85mm f/2.8 fisheye lens. The maximum image size is 2880 x 2880 pixels.

<table>
<thead>
<tr>
<th>Field of View</th>
<th>Top and left border</th>
<th>Width and height</th>
</tr>
</thead>
<tbody>
<tr>
<td>180°</td>
<td>210</td>
<td>2456</td>
</tr>
<tr>
<td>185°</td>
<td>174</td>
<td>2528</td>
</tr>
<tr>
<td>190°</td>
<td>142</td>
<td>2592</td>
</tr>
<tr>
<td>195°</td>
<td>106</td>
<td>2664</td>
</tr>
<tr>
<td>200°</td>
<td>74</td>
<td>2728</td>
</tr>
</tbody>
</table>

Charging the battery with the external battery charger: Lamp is red while charging and becomes green when battery is full. The charging time is at least 4 hours for a completely empty battery.

Charging the battery in the camera: Just plug in the USB cable and don’t switch the camera on. The lamp is blinking orange while charging and goes off when the battery is full.

Error in instruction manual: The lamp is not continuously orange when the battery is charging.
<table>
<thead>
<tr>
<th>18</th>
<th>Chinese 360° Panoramic camera</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image.jpg" alt="Image" /></td>
<td>This is a very cheap Chinese panoramic camera with two 220° fisheye lenses.</td>
</tr>
<tr>
<td></td>
<td>Video resolution: 1920x960@30fps, 2880x1440@25fps, 3840x1920@15fps</td>
</tr>
<tr>
<td></td>
<td>Lenses: f=0.88mm F/2.0, distance between the two lenses is about 26mm</td>
</tr>
<tr>
<td></td>
<td>Audio: 44.1kHz, 16-bit, stereo</td>
</tr>
<tr>
<td></td>
<td>After downloading the video from the Micro SD card, it is already an equirectangular video and can be viewed as-is with the VLC player. The stitching is already done in the camera and there is no postprocessing required.</td>
</tr>
<tr>
<td></td>
<td>Charging via USB: Blue lamp is on while charging, and goes off when battery is full.</td>
</tr>
<tr>
<td></td>
<td>If you mount this camera on a selfie stick, the stick itself isn't visible in the video. But its shadow is visible! So it's a good idea to choose the diameter of the stick as small as possible.</td>
</tr>
<tr>
<td></td>
<td>This camera has no exposure compensation setting.</td>
</tr>
<tr>
<td>Main Menu</td>
<td>Sub Menu</td>
</tr>
<tr>
<td>-----------</td>
<td>----------</td>
</tr>
<tr>
<td>BASIC</td>
<td>RECORD</td>
</tr>
<tr>
<td></td>
<td>PAN</td>
</tr>
<tr>
<td></td>
<td>GAIN</td>
</tr>
<tr>
<td></td>
<td>INPUT</td>
</tr>
<tr>
<td>MONITOR</td>
<td>MIX</td>
</tr>
<tr>
<td>INPUT</td>
<td>INPUT GAIN</td>
</tr>
<tr>
<td></td>
<td>LIMITER</td>
</tr>
<tr>
<td></td>
<td>LOWCUT</td>
</tr>
<tr>
<td></td>
<td>DELAY</td>
</tr>
<tr>
<td></td>
<td>PHASE</td>
</tr>
<tr>
<td>RECORD</td>
<td>FILE TYPE</td>
</tr>
<tr>
<td></td>
<td>FORMAT</td>
</tr>
<tr>
<td></td>
<td>SAMPLE</td>
</tr>
<tr>
<td></td>
<td>DUAL REC</td>
</tr>
<tr>
<td>SLATE</td>
<td></td>
</tr>
<tr>
<td>MIC</td>
<td>MS MODE 1/2</td>
</tr>
<tr>
<td></td>
<td>MS MODE 3/4</td>
</tr>
<tr>
<td></td>
<td>PHANTOM VOLT</td>
</tr>
<tr>
<td>OTHERS</td>
<td>SYSTEM --&gt; FORMAT</td>
</tr>
<tr>
<td></td>
<td>BATTERY</td>
</tr>
<tr>
<td></td>
<td>DATE / TIME</td>
</tr>
</tbody>
</table>
I typically make either 4-channel records, or 2-channel records with DUAL REC -10dB. These settings must be changed:

<table>
<thead>
<tr>
<th>Application</th>
<th>BASIC --&gt; RECORD CH3+4</th>
<th>RECORD --&gt; DUAL RECORD</th>
</tr>
</thead>
<tbody>
<tr>
<td>4-Channel Recording</td>
<td>ON</td>
<td>OFF</td>
</tr>
<tr>
<td>2-Channel Recording with DUAL REC -10dB</td>
<td>OFF</td>
<td>-10dB</td>
</tr>
</tbody>
</table>

WAV 24bit 44.1 kHz, maximum recording length with 2GB SD card: 3h 22m

Always power the recorder with an external powerbank. The internal batteries are much too small, especially if phantom voltage is used.
## TASCAM DR-701D

<table>
<thead>
<tr>
<th>Menu</th>
<th>Recommended Setting</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>1/19 INPUT</td>
<td>GAIN: LINE / LOW / MID / HI / HI+ SEL: IN 1-2 IN 3-4</td>
<td>Choose the input gain and which inputs are used</td>
</tr>
<tr>
<td></td>
<td></td>
<td>LOW: +20dB, MID: +40dB, HI: +52dB, HI+: +64dB</td>
</tr>
<tr>
<td>2/19 MIXER</td>
<td>LVL: 100, 100, 100, 100 PAN: L12, R12, L12, R12 MS: OFF</td>
<td>Important: If you set PAN to &quot;C&quot;, both stereo channels are equal!</td>
</tr>
<tr>
<td>3/19 PHASE / DELAY</td>
<td>0, OFF</td>
<td>Reverse the polarity, set a delay time</td>
</tr>
<tr>
<td>4/19 LEVEL CONTROL</td>
<td>OFF</td>
<td></td>
</tr>
<tr>
<td>5/19 TRIM GANG</td>
<td>GRP 1: all ON GRP2: all OFF</td>
<td>Adjust all channels simultaneously with channel 1 knob</td>
</tr>
<tr>
<td>6/19 OUTPUT LEVEL</td>
<td>CAMERA: 30db LINE: 0db</td>
<td>Set the output levels</td>
</tr>
<tr>
<td>7/19 MIC POWER</td>
<td>PHAN: all ON, VOLTAGE : 48V PLUGIN: OFF</td>
<td>Use 48V for RODE NT1 microphones PLUGIN is the supply voltage for microphones at the EXT IN 1/2 input</td>
</tr>
<tr>
<td>8/19 RECORD</td>
<td>CH1, CH2, CH3, CH4: ON MIX: OFF DUAL: OFF or 1-2, -12dB</td>
<td>When DUAL mode is used, channels 3 and 4 are automatically deselected</td>
</tr>
<tr>
<td>9/19 REC SETTING</td>
<td>FILE TYPE: STEREO FORMAT: WAV 24bit SAMPLE: 44.1kHz / 48kHz / 96kHz / 192kHz</td>
<td>One or two stereo files will be written Use 44.1kHz or 48kHz for normal sound, or 96kHz for ultrasound</td>
</tr>
<tr>
<td>10/19 FILE</td>
<td>NAME TYPE: DATE WORD: TASCAM</td>
<td></td>
</tr>
<tr>
<td>11/19 MEDIA</td>
<td>FORMAT</td>
<td>Here you can format the SD card</td>
</tr>
<tr>
<td>12/19 TIME CODE</td>
<td></td>
<td></td>
</tr>
<tr>
<td>13/19 SLATE TONE</td>
<td>AUTO: OFF OSCILLATOR</td>
<td>Use the OSCILLATOR feature for generating a -20dB test tone</td>
</tr>
<tr>
<td>14/19 HDMI AUDIO ASSIGN</td>
<td>OFF</td>
<td></td>
</tr>
<tr>
<td>15/19 AMBISONICS</td>
<td>OFF</td>
<td></td>
</tr>
<tr>
<td>------------------</td>
<td>-----</td>
<td></td>
</tr>
</tbody>
</table>
| 16/19 METER/TRIM | PEAK HOLD: 2sec  
TRIM MIN: MIN |
| 17/19 POWER MANAGEMENT | BATTERY TYPE: ALKALI  
AUTO PWR SAVE: 30min  
BACKLIGHT: 10sec |
| 18/19 REMOTE | DATE / TIME  
Setting date and time |
| 19/19 SYSTEM | DATE / TIME  
Setting date and time |

I typically make either 4-channel records, or 2-channel records with DUAL REC -10dB. For toggling between these modes, only one setting must be changed: Set RECORD / DUAL to OFF or 1-2.

Always power the recorder with an external powerbank. The internal batteries are much too small, especially if phantom voltage is used.

Pinout of 3.5mm stereo connectors: Tip contact is left channel, middle contact is right channel, outer contact is ground.
20.1 Matching the DR-701D's output level to the GH5S' input level

The output level of the TASCAM DR-701D camera output can be set in the menu "OUTPUT LEVEL / CAMERA" in the range -24dB to +42dB. There are hardware switches between 0dB and 1dB, between 12dB and 13dB and between 30dB and 31dB.

A 1kHz test tone can be generated in the menu "SLATE TONE / OSCILLATOR", with level -18dB or -20dB. The reference level seems to be about 62mV without load.

Output level at the TASCAM's **camera output** (measured with high impedance):

<table>
<thead>
<tr>
<th>OUTPUT LEVEL / CAMERA</th>
<th>Output voltage (OSCILLATOR = -18dB)</th>
<th>Output voltage (OSCILLATOR = -20dB)</th>
<th>Maximum 1kHz sine output voltage, just before clipping occurs in the output signal.</th>
</tr>
</thead>
<tbody>
<tr>
<td>0dB</td>
<td>7.5 mV_rms</td>
<td>6.2 mV_rms</td>
<td>62 mV_rms</td>
</tr>
<tr>
<td>12dB</td>
<td>30.3 mV_rms</td>
<td>24.0 mV_rms</td>
<td>240 mV_rms</td>
</tr>
<tr>
<td>20dB</td>
<td>79.0 mV_rms</td>
<td>62.0 mV_rms</td>
<td>620 mV_rms</td>
</tr>
<tr>
<td>30dB</td>
<td>249.6 mV_rms</td>
<td>200.0 mV_rms</td>
<td>2.0 V_rms</td>
</tr>
<tr>
<td>40dB</td>
<td>795 mV_rms</td>
<td>622 mV_rms</td>
<td>3.35 V_rms</td>
</tr>
<tr>
<td>42dB</td>
<td>993 mV_rms</td>
<td>795 mV_rms</td>
<td>3.35 V_rms</td>
</tr>
</tbody>
</table>

The output level of the TASCAM DR-701D line output can be set in the menu "OUTPUT LEVEL / LINE" in the range -12dB to +12dB. There is a hardware switch between 0dB and 1dB.

Output level at the TASCAM's **line output** (measured with high impedance):

<table>
<thead>
<tr>
<th>OUTPUT LEVEL / LINE</th>
<th>Output voltage (OSCILLATOR = -18dB)</th>
<th>Output voltage (OSCILLATOR = -20dB)</th>
<th>Maximum 1kHz sine output voltage, just before clipping occurs in the output signal.</th>
</tr>
</thead>
<tbody>
<tr>
<td>-12dB</td>
<td>62 mV_rms</td>
<td>49 mV_rms</td>
<td>0.5 V_rms</td>
</tr>
<tr>
<td>-3dB</td>
<td>175 mV_rms</td>
<td>139 mV_rms</td>
<td>1.41 V_rms</td>
</tr>
<tr>
<td>0dB</td>
<td>248 mV_rms</td>
<td>197 mV_rms</td>
<td>2.0 V_rms</td>
</tr>
<tr>
<td>12dB</td>
<td>990 mV_rms</td>
<td>785 mV_rms</td>
<td>3.27 V_rms</td>
</tr>
</tbody>
</table>
The input level of the Panasonic LUMIX GH5S can be set to "LINE" in the menu Motion_Picture --> Mic_Socket.
The Motion_Picture --> Sound_Rec_Level_Adj. parameter can be set in the -12dB to +6dB range.
For measuring the clipping voltage, make sure that Motion_Picture --> Sound_Rec_level_Limiter is OFF.

<table>
<thead>
<tr>
<th>Sound_Rec_Level_Adj.</th>
<th>Input voltage when level indicator is at -12dB mark</th>
<th>Maximum sine voltage before clipping occurs</th>
<th>Maximum peak voltage before clipping occurs</th>
</tr>
</thead>
<tbody>
<tr>
<td>-12dB</td>
<td>1050 mV_rms</td>
<td>4.88 V_rms</td>
<td>+- 6.90 V</td>
</tr>
<tr>
<td>-6dB</td>
<td>525 mV_rms</td>
<td>2.44 V_rms</td>
<td>+- 3.45 V</td>
</tr>
<tr>
<td>0dB</td>
<td>262 mV_rms</td>
<td>1.22 V_rms</td>
<td>+- 1.73 V</td>
</tr>
<tr>
<td>+6dB</td>
<td>131 mV_rms</td>
<td>0.61 V_rms</td>
<td>+- 0.86 V</td>
</tr>
</tbody>
</table>

So after all these measurements, what's a good match between the output level of the TASCAM and the input level of the GH5S?

<table>
<thead>
<tr>
<th>TASCAM DR-701D</th>
<th>Panasonic LUMIX GH5S</th>
</tr>
</thead>
<tbody>
<tr>
<td>Camera output 27dB or line output -3dB</td>
<td>Set microphone input to &quot;LINE&quot; and Sound_Rec_Level_Adj. to 0dB</td>
</tr>
</tbody>
</table>

Or alternatively:

<table>
<thead>
<tr>
<th>TASCAM DR-701D</th>
<th>Panasonic LUMIX GH5S</th>
</tr>
</thead>
<tbody>
<tr>
<td>Camera output 30dB or line output 0dB</td>
<td>Set microphone input to &quot;LINE&quot; and Sound_Rec_Level_Adj. to -3dB</td>
</tr>
</tbody>
</table>

With these settings both recorders get the same amplitude and clipping occurs at the same level.
21 The Apprehension Engine: Sound effects for horror films

This is a machine for creating sound effects for horror films. It was envisioned by movie composer Mark Korven and created by guitar maker Tony Duggan-Smith. http://apprehensionengine.com/
The apprehension engine was used to make the sounds for the movie "The Witch": https://en.wikipedia.org/wiki/The_Witch_(2015_film)

Some videos by Jakob Balogh showing what you can do with this engine:
The Apprehension Engine - First Look Part 01 (Horror Machine) https://www.youtube.com/watch?v=dSVzFD6bDwQ
The Apprehension Engine - First Look Part 02 (Horror Machine) https://www.youtube.com/watch?v=61Cw5vApw-o
The Apprehension Engine - First Look Part 03 (Horror Machine) https://www.youtube.com/watch?v=n5nAXLdBc40

Other videos showing how to use the engine and similar instruments:
The Apprehension Engine - Horror Suite Part 1 https://www.youtube.com/watch?v=QUYFMHM3wns
The Apprehension Engine - Horror Suite Part 2 https://www.youtube.com/watch?v=K9xE1UHDoLU
Apprehension Engine: Sound check in Chicago https://www.youtube.com/watch?v=cpp7SwROgXy
Horror Musical Instrument - The Apprehension Engine https://www.youtube.com/watch?v=Izk-l8Gm0MY
DIY Apprehension Engine 1 - Metallic Bones (Rulers) https://www.youtube.com/watch?v=q_yMKs1NYg
DIY Apprehension Engine 2 - Glass Shards (Wine Glass) https://www.youtube.com/watch?v=1arnzEoAuAk
DIY Apprehension Engine 3 - Gates of Hell (Spring Reverb) https://www.youtube.com/watch?v=v7yq1ToQu1Eo
DIY Apprehension Engine 4 - Heavy Metal (Guitar) https://www.youtube.com/watch?v=gh7TbusFy-4
Latest build: "Horror Box 1.0" - (demo) - spring box w/ piezo microphone https://www.youtube.com/watch?v=tEqUXjIDElg

Here is a series of "How to Build The Apprehension Engine" videos by Michael Freudenberg on Youtube:

<table>
<thead>
<tr>
<th>Chapter</th>
<th>Youtube Link</th>
<th>Material needed</th>
</tr>
</thead>
<tbody>
<tr>
<td>#1 - HISTORY</td>
<td><a href="https://www.youtube.com/watch?v=xHXUEycuAMY">https://www.youtube.com/watch?v=xHXUEycuAMY</a></td>
<td></td>
</tr>
<tr>
<td>#2 - The Base</td>
<td><a href="https://www.youtube.com/watch?v=ufzZmU4l4P10">https://www.youtube.com/watch?v=ufzZmU4l4P10</a></td>
<td>5 Ply board: (for the base) Width 40cm x Length 1.2 meters Pine wood: Width 4.2 cm x Depth 1.9 cm x Length 1.2 meters</td>
</tr>
<tr>
<td>#3 - The Sides</td>
<td><a href="https://www.youtube.com/watch?v=eZXM-Dj7Iqw">https://www.youtube.com/watch?v=eZXM-Dj7Iqw</a></td>
<td></td>
</tr>
<tr>
<td>#4 - Finishing the Sides</td>
<td><a href="https://www.youtube.com/watch?v=JEjMSTJ">https://www.youtube.com/watch?v=JEjMSTJ</a> Ts</td>
<td></td>
</tr>
<tr>
<td>#5 - Attaching rear support</td>
<td><a href="https://www.youtube.com/watch?v=SkLuiXLvTgw">https://www.youtube.com/watch?v=SkLuiXLvTgw</a></td>
<td></td>
</tr>
<tr>
<td>#6 - The Hurdy Gurdy Wheel</td>
<td><a href="https://www.youtube.com/watch?v=cIL_YBDax5s">https://www.youtube.com/watch?v=cIL_YBDax5s</a></td>
<td>MDF 16mm thick</td>
</tr>
<tr>
<td>#7 - The Hurdy Gurdy bearing</td>
<td><a href="https://www.youtube.com/watch?v=xU8ES5OLxak">https://www.youtube.com/watch?v=xU8ES5OLxak</a></td>
<td></td>
</tr>
<tr>
<td>#8 - Installing the Front Frame</td>
<td><a href="https://www.youtube.com/watch?v=3tSBeRqQSOE">https://www.youtube.com/watch?v=3tSBeRqQSOE</a></td>
<td></td>
</tr>
<tr>
<td>-----------------------------</td>
<td>---------------------------------------------</td>
<td></td>
</tr>
<tr>
<td>#9 - Installing the Rear Frame</td>
<td><a href="https://www.youtube.com/watch?v=XWddPilneco">https://www.youtube.com/watch?v=XWddPilneco</a></td>
<td></td>
</tr>
<tr>
<td>#10 - Guitar Neck Supports</td>
<td><a href="https://www.youtube.com/watch?v=az5uFla6gBg">https://www.youtube.com/watch?v=az5uFla6gBg</a></td>
<td></td>
</tr>
<tr>
<td>#11 - Left side Soundboard</td>
<td><a href="https://www.youtube.com/watch?v=HhuUxrRj5SE">https://www.youtube.com/watch?v=HhuUxrRj5SE</a></td>
<td></td>
</tr>
<tr>
<td>#12 - Front and Right side Soundboard</td>
<td><a href="https://www.youtube.com/watch?v=wlzUkMqLDhM">https://www.youtube.com/watch?v=wlzUkMqLDhM</a></td>
<td></td>
</tr>
<tr>
<td>#13 - Installing Top Soundboard</td>
<td><a href="https://www.youtube.com/watch?v=DBW_x_KKEdM">https://www.youtube.com/watch?v=DBW_x_KKEdM</a></td>
<td></td>
</tr>
<tr>
<td>#14 - Installing the Hurdy Gurdy</td>
<td><a href="https://www.youtube.com/watch?v=t1JzJRfPtW0">https://www.youtube.com/watch?v=t1JzJRfPtW0</a></td>
<td></td>
</tr>
<tr>
<td>#15 - Making the Guitar Necks</td>
<td><a href="https://www.youtube.com/watch?v=84oymSJ6L1w">https://www.youtube.com/watch?v=84oymSJ6L1w</a></td>
<td></td>
</tr>
<tr>
<td>Part II</td>
<td>Hard wood 65mm x 18mm x 1.2 meters</td>
<td></td>
</tr>
<tr>
<td>#16 - Making the Guitar Necks Part II</td>
<td><a href="https://www.youtube.com/watch?v=lrAJwj0ZpoU">https://www.youtube.com/watch?v=lrAJwj0ZpoU</a></td>
<td></td>
</tr>
<tr>
<td>#17 - FINISHING THE BOX</td>
<td><a href="https://www.youtube.com/watch?v=OJtZyos_ZaE">https://www.youtube.com/watch?v=OJtZyos_ZaE</a></td>
<td></td>
</tr>
<tr>
<td>#18 – The Electronics and parts</td>
<td><a href="https://www.youtube.com/watch?v=Kel4onBniTs">https://www.youtube.com/watch?v=Kel4onBniTs</a></td>
<td></td>
</tr>
<tr>
<td>3 mm to 5 mm Ply wood (3 ply) 18mm x 18mm square pine wood</td>
<td></td>
<td></td>
</tr>
<tr>
<td>#19 - Installing The Guitar Tuners</td>
<td><a href="https://www.youtube.com/watch?v=oNs79OUh3AU">https://www.youtube.com/watch?v=oNs79OUh3AU</a></td>
<td></td>
</tr>
<tr>
<td>#20 - The String Bridges</td>
<td><a href="https://www.youtube.com/watch?v=8h9N7T8jA50">https://www.youtube.com/watch?v=8h9N7T8jA50</a></td>
<td></td>
</tr>
<tr>
<td>#21 - Installing the Humbucker pickup</td>
<td><a href="https://www.youtube.com/watch?v=mRA0JK5aCfk">https://www.youtube.com/watch?v=mRA0JK5aCfk</a></td>
<td></td>
</tr>
<tr>
<td>#22 - Installing the Piezo Contact Pickup</td>
<td><a href="https://www.youtube.com/watch?v=N1BU6MSp8Xs">https://www.youtube.com/watch?v=N1BU6MSp8Xs</a></td>
<td></td>
</tr>
<tr>
<td>---------------------------------</td>
<td>---------------------------------------------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>#24 - The Final Tutorial</td>
<td><a href="https://www.youtube.com/watch?v=QSJOtxwgd8Y">https://www.youtube.com/watch?v=QSJOtxwgd8Y</a></td>
<td>Learn how to add the cotton to the Hurdy Gurdy strings here: <a href="https://www.youtube.com/watch?v=0TTi5FoNKw8">https://www.youtube.com/watch?v=0TTi5FoNKw8</a></td>
</tr>
</tbody>
</table>

Reverb tank 4FB3A1A: Input 1475 Ohm, Output 2250 Ohm, Long dcay 2.75s - 4s, Input grounded, Output grounded
This is the corrected block diagram of the Betagear FX82USB mixer:
# Timelapse duration table

This table lists the number of pictures and the video duration at 30fps, depending on interval and recording time.

<table>
<thead>
<tr>
<th>Interval</th>
<th>1h</th>
<th>2h</th>
<th>3h</th>
<th>4h</th>
<th>5h</th>
<th>6h</th>
<th>8h</th>
<th>12h</th>
<th>24h</th>
</tr>
</thead>
<tbody>
<tr>
<td>2s</td>
<td>1800</td>
<td>60s</td>
<td>3600</td>
<td>120s</td>
<td>5400</td>
<td>150s</td>
<td>7200</td>
<td>240s</td>
<td>9400</td>
</tr>
<tr>
<td>3s</td>
<td>1200</td>
<td>40s</td>
<td>2400</td>
<td>80s</td>
<td>3600</td>
<td>120s</td>
<td>4800</td>
<td>160s</td>
<td>6000</td>
</tr>
<tr>
<td>4s</td>
<td>900</td>
<td>30s</td>
<td>1800</td>
<td>60s</td>
<td>2700</td>
<td>90s</td>
<td>3600</td>
<td>120s</td>
<td>4500</td>
</tr>
<tr>
<td>5s</td>
<td>720</td>
<td>24s</td>
<td>1440</td>
<td>48s</td>
<td>2160</td>
<td>72s</td>
<td>2880</td>
<td>96s</td>
<td>3600</td>
</tr>
<tr>
<td>6s</td>
<td>600</td>
<td>20s</td>
<td>1200</td>
<td>40s</td>
<td>1800</td>
<td>60s</td>
<td>2400</td>
<td>80s</td>
<td>3000</td>
</tr>
<tr>
<td>8s</td>
<td>450</td>
<td>15s</td>
<td>900</td>
<td>30s</td>
<td>1350</td>
<td>45s</td>
<td>1800</td>
<td>60s</td>
<td>2350</td>
</tr>
<tr>
<td>10s</td>
<td>360</td>
<td>12s</td>
<td>720</td>
<td>24s</td>
<td>1080</td>
<td>36s</td>
<td>1440</td>
<td>48s</td>
<td>1800</td>
</tr>
<tr>
<td>12s</td>
<td>300</td>
<td>10s</td>
<td>600</td>
<td>20s</td>
<td>900</td>
<td>30s</td>
<td>1200</td>
<td>40s</td>
<td>1500</td>
</tr>
<tr>
<td>15s</td>
<td>240</td>
<td>8s</td>
<td>480</td>
<td>16s</td>
<td>720</td>
<td>24s</td>
<td>960</td>
<td>32s</td>
<td>1200</td>
</tr>
<tr>
<td>20s</td>
<td>180</td>
<td>6s</td>
<td>360</td>
<td>12s</td>
<td>540</td>
<td>18s</td>
<td>720</td>
<td>24s</td>
<td>900</td>
</tr>
<tr>
<td>24s</td>
<td>150</td>
<td>5s</td>
<td>300</td>
<td>10s</td>
<td>450</td>
<td>15s</td>
<td>600</td>
<td>20s</td>
<td>750</td>
</tr>
<tr>
<td>30s</td>
<td>120</td>
<td>4s</td>
<td>240</td>
<td>8s</td>
<td>360</td>
<td>12s</td>
<td>480</td>
<td>16s</td>
<td>600</td>
</tr>
<tr>
<td>40s</td>
<td>90</td>
<td>3s</td>
<td>180</td>
<td>6s</td>
<td>270</td>
<td>9s</td>
<td>360</td>
<td>12s</td>
<td>450</td>
</tr>
<tr>
<td>60s</td>
<td>60</td>
<td>2s</td>
<td>120</td>
<td>4s</td>
<td>180</td>
<td>6s</td>
<td>240</td>
<td>8s</td>
<td>300</td>
</tr>
<tr>
<td>120s</td>
<td>30</td>
<td>1s</td>
<td>60</td>
<td>2s</td>
<td>90</td>
<td>3s</td>
<td>120</td>
<td>4s</td>
<td>150</td>
</tr>
</tbody>
</table>
This is a small device that connects to a camera and controls the exposure time, aperture and ISO automatically, so that day-to-night or night-to-day timelapses are possible.

https://www.timelapseplus.com/

Important notes:

- Set the camera to manual (M) mode
- Use a native ISO setting (not Auto ISO)
- Save as RAW (not RAW + JPG)
- Manual focus (no autofocus)
- Disable image stabilization
- Check all parameters before using
- Don’t rely on the internal battery, use an external powerbank
- Save the images in the camera, not in the Timelapse+ View
- The "Night Exposure" parameter describes how much darker the video shall become at night. Typical values are -0.5 to -0.75. Please note that the unit of this parameter isn’t specified. These are not exposure compensation values! I did try -2 and the resulting video was much too dark in the night (about -11 exposure values).
24 Guide 9.1

This chapter may be a little bit off-topic in this document, because Guide 9.1 is an astronomy program. But I didn't want to create a new document for my notes about it.

Website: https://www.projectpluto.com/

24.1 Install Guide 9.1

- Insert the Guide 9.0 DVD and open the folder, then run "setup.exe". This will install Guide 9.0 very fast, but most of the data is still on the DVD. Which means it does only work if you let the DVD in the drive.
- If you have enough space on the harddisk, it's recommended to install all on the harddisk. Run Guide 9.0 and click on Extras / Install_on_hard_drive. It's best if you select all, except those languages that you don't need.
- It's highly recommended to install the upgrade to Guide 9.1, which is available here: https://www.projectpluto.com/ This upgrade is required for communication with a telescope over the serial port, and also for downloading the latest comet orbit data.

24.2 Control a LX200-compatible telescope

- If your computer doesn't have a RS232 port, then use a USB / RS232 adapter. Plug this adapter into a free USB port and find out which COM number was assigned to the adapter, e.g. COM10. Use always the same USB port. Otherwise the COM number will change.
- In Guide 9.1 click on Settings / Scope_Control. here you choose the COM number and as telescope type you use "LX200". Then click on "OK".
- Now there is a new menue "Scope Pad". When you click on it, a small window opens. Here you can control the telescope. It's described in the FS2 manual.
- USB adapters don't work with Guide 9.0. You must install the Guide 9.1 upgrade.
24.3  Add new comets to Guide 9.1

The "Add MPC Comets / Asteroids" function does no longer work. You can use this workaround:
Go to http://astro.vanbuitenen.nl/cometelements?format=guide

and save this file in your Guide folder as soft02cm.txt (this is used for Guide) and also as comets.dat or cometg.dat (this is used for Charon, use the filename that already exists).

The broken "Add MPC Comets / Asteroids" function in Guide9.1 can be repaired if you copy and paste the following content to the "add_mpc.hee" file. In german installations the filename may be "add_mpc.hed". This doesn't work with Guide 9.0, the upgrade to Guide 9.1 is required. (Thanks to Larry Wood who posted this in the Guide user group, September 18, 2019)

| The ^Minor Planet Center (MPC)//xhttps://www.minorplanetcenter.net/cfa/ps/mpc.html^ and the ^IMCCE//xhttp://www.imcce.fr/fr^ provide orbital elements for comets. Guide updates its list of comets using both sources: MPC gives currently-observable comets, IMCCE all comets since about 1995. (Data for historical comets is already built into the Guide DVD.) You can click on the following to download some of these files, getting orbital data for newly-found objects and improving orbits for already known objects. About 600 KBytes will be downloaded.
| ^Click to download updated comet data and add it to Guide//dhttp://astro.vanbuitenen.nl/cometelements?format=guide soft02cm.txt;dhttps://www.projectpluto.com/eltdat.txt eltdat.txt;a2789^ Guide can also import other orbital elements if they're provided in the "eight-line format", or the "one-line format" used for Daily Orbit Updates. You wouldn't normally do this, but if you have generated an orbit using Find_Orb, for example, you could import the resulting file of orbital elements using the following command.
| ^Add MPC asteroids/comets//!2052^ Please note that the long line in the middle must be written in one line and there is a space character between "guide" and "soft02cm". |
24.4 Add ephemerides to Guide 9.1

The path of those comets or asteroids which have a close encounter with other objects (e.g. planets) can't be described by orbital elements for a longer time. If you want to add the ephemeride of such an object point-wise into Guide 9.1, follow these instructions:

Go to this MPC website: http://www.minorplanetcenter.net/iau/MPEph/MPEph.html

and write the name of the object in the large white field (e.g. 2012DA14). Then fill in some more fields (use your own data, of course):

Ephemeris start date: e.g. 2013 02 15 19:00
Number of dates to output: e.g. 400
Ephemeris interval: e.g. 1 minute
Longitude: e.g. 10.3454
Latitude: e.g. 51.3829
Altitude: e.g. 257
Display R.A./Decl. positions in: full sexagesimal
Tick the box "Suppress output if sun above local horizon" if that makes sense for your object.

Then click on "Get Ephemerides/HTML page". Now copy and paste the data lines (without the header) to an editor. It should look like this:

| Date | Start Time | RA  | Dec  | Epoch  | B  | L  | A  | RA  | Dec  | Epoch  | B  | L  | A  | RA  | Dec  | Epoch  | B  | L  | A  | RA  | Dec  | Epoch  | B  | L  | A  | RA  | Dec  | Epoch  | B  | L  | A  |
|------|------------|-----|------|--------|----|----|----|-----|------|--------|----|----|----|-----|------|--------|----|----|----|-----|------|--------|----|----|----|-----|------|--------|----|----|----|-----|------|--------|----|----|----|-----|------|--------|
| 2013 02 15 | 190000 12 01 19.4 | -31 18 25 | 0.00026 | 0.988 | 127.0 | 52.9 | 8.4 | 2279.58 | 003.5 | 285 | -29 | -23 | 0.31 | 147 | +36 | 44 359.0 | Map | Offsets
| 2013 02 15 | 190100 12 01 30.2 | -30 40 19 | 0.00026 | 0.988 | 127.5 | 52.5 | 8.3 | 2300.72 | 003.5 | 284 | -29 | -24 | 0.31 | 147 | +35 | 44 359.3 | Map | Offsets
| 2013 02 15 | 190200 12 01 41.1 | -30 01 52 | 0.00025 | 0.988 | 128.0 | 52.0 | 8.3 | 2321.73 | 003.5 | 284 | -28 | -24 | 0.31 | 147 | +35 | 44 359.5 | Map | Offsets
| 2013 02 15 | 190300 12 01 51.9 | -29 23 04 | 0.00025 | 0.988 | 128.5 | 51.5 | 8.3 | 2342.59 | 003.5 | 283 | -28 | -24 | 0.31 | 148 | +35 | 44 359.8 | Map | Offsets
| 2013 02 15 | 190400 12 02 02.7 | -28 43 55 | 0.00025 | 0.988 | 128.9 | 51.1 | 8.3 | 2363.24 | 003.4 | 283 | -27 | -24 | 0.31 | 148 | +35 | 44 000.1 | Map | Offsets

and so on...

Save this file as "2012DA14.dat" to your Guide folder (e.g. C:/GUIDE9).

It's absolutely required that all data are in the correct columns, as shown above.
Now create another file "2012DA14.tdf" and save it in the same folder. This is the content:

```
file 2012DA14.dat
title Asteroid 2012DA14
RA H 19 2
RA M 22 2
RA S 25 4
de d 30 3
de m 34 2
de s 37 2
mag 70 4
text 12 4
pref 2012DA14
epoch 2000
type sc1;e0,0,30; #green circle, 30 pixels diameter
shown 1
end
```

That's all. Start Guide and the positions will be shown.

### 24.5 Update the position of Jupiter's great red spot

The longitude of Jupiter's great red spot must be updated from time to time. To do this, open the file "grs_long.txt" from the Guide folder with an editor. Then insert a new line near the top, for example:

```
2019 6 1 311 (user)
```

In this example the longitude is 311° for date 2019 June 1th.

Save the file with the same filename.
24.6 Add a user-defined horizon

Either measure the horizon heights with a azimuthal telescope, or make a 180° fisheye image of your location. The lens is pointing to the zenith. Convert this image to an equirectangular image as follows:

```plaintext
set "FF=c:\ffmpeg\ffmpeg"   :: Path to FFmpeg
set "IN=s3.jpg"             :: Input fisheye image, south at the bottom
set "SQ=3648"               :: Height of input image (width doesn't care)
set "OUT=s3.png"            :: Stereographic output image 360x90 pixel,
                             :: north at the left, south in the center, horizon at the top, zenith at the bottom

%FF% -i %IN% -lavfi "crop=ih:ih, scale=180:180, pad=w=2*iw, v360=input=dfisheye: output=e, rorder='rpy': roll=180: pitch=90, crop=iw:ih/2: y=0, vflip" -y %OUT%
```

Open the output image with IrfanView. The X coordinate is the azimuth angle in degrees, and the Y coordinate is the height over the horizon in degrees. Now you can manually read out the pixel positions of the horizon line for all azimuth angles in 5- or 10-degree steps.

Then you can insert the horizon line at the beginning of the horizon.dat file:

```
hor 32 0 0 ; these are the RGB colors
0 25
10 21
20 20
30 16
40 22
50 18
60 16
70 22
80 19
90 16
100 22
110 20
```

236
hend

- $\text{N}_{\text{for\ North}} 0 .5 .5$
- $\text{N}_{\text{for\ North}} 43 .5 .4$
- $\text{E}_{\text{for\ East}} 47 .5 .4$
- $\text{E}_{\text{for\ East}} 90 .5 .5$
- $\text{S}_{\text{for\ South}} 133 .5 .4$
- $\text{E}_{\text{for\ East}} 137 .5 .4$
- $\text{S}_{\text{for\ South}} 180 .5 .5$
- $\text{S}_{\text{for\ South}} 223 .5 .4$
- $\text{W}_{\text{for\ West}} 227 .5 .4$
- $\text{W}_{\text{for\ West}} 270 .5 .5$
- $\text{N}_{\text{for\ North}} 313 .5 .4$
- $\text{W}_{\text{for\ West}} 317 .5 .4$
24.7 Switch between several user-defined horizon files

If you have several horizon files and you want to switch between them, you can create one batch file for each horizon, with this content:

```
copy d:\guide9\my_horizon_1.dat d:\guide9\horizon.dat
```

Put the batch files on the desktop and execute one of them by double-clicking. It will automatically overwrite the horizon.dat file with your own horizon file. You can change the horizon file while Guide is running. After overwriting the horizon file, just click in the Guide window to refresh the graphics and then you see the new horizon.

24.8 Install the Gaia2 catalog for Guide 9.1

Thanks to Jost Jahn who made it possible to download the Gaia2 catalog for Guide 9.1 here:

http://www.gaia2.de/index.html

Simply follow the instructions there.

24.9 Set up Guide 9.1 to show only the Gaia2 catalog

This is useful for making realistic timelapse videos of the proper motion of stars over 10000's of years. The trick is to create a second Guide folder which contains a minimal installation of Guide 9.1, with only those files that are absolutely required. No star catalogs are present in this folder.

- Create a new folder "guide_gaia_only"
- Copy the following files and folders from the original guide folder to the new folder: cache (folder), ngcic (folder), astnum, bitfont, cometg.dat, constbnd.ove, constlab.ove, gaia-std.tdf, gscdata2.idx, guide.dat, guide9.exe, hotkey.dat, lunar.dll, marks.nam, maximum.dat, messier.hee, overlays.nam, startup.mar, strings.dat, tdf_list.dat, temp_mar.txt, vsop.bin, win_meng.dat and win_menu.dat
- Open the file "gaia-std.tdf" with an editor and search/replace "file !:\STD" to "file D:\Guide\STD" using the actual path to the Gaia catalog. You don't want to have this catalog on your harddisk twice. There are 180 instances in the file that must be changed.
- Start Guide in the new folder.
DeepSkyStacker 4.2.2 can be downloaded here: http://deepskystacker.tree.fr/german

Support group: https://groups.io/g/DeepSkyStacker

A big advantage of this version is that it can read the RAW files from the Canon 5D-MK4.

The language used by DeepSkyStacker is automatically set from the language used in the operating system. If you want to force another language you can change it from the "About" box.

Known problems:
When you open the light / dark / flat / offset images, unfortunately DSS always opens by default the folder from the last session. Same problem when you save the file list. Take care that you don't accidentally overwrite the file list from the last session! There is no known workaround to fix this problem. You have to select five times the same new folder.
25.1 How to stack on comets with known motion

Normally the comet must be marked in at least 3 images: The first, the last and the reference image. If the first or last image is the reference image, then two images are sufficient. Marking the comet is simple if the comet is clearly visible in the images.

However things are getting difficult if either the comet is invisible (because it's too faint and hidden in the noise) or if the comet is so diffuse that it's difficult to define it's center. In these cases you can proceed as follows:

- It's required that north is up in all images, and that all images are already registered.
- Use the first or the last image as reference image. Use that one with the higher score. That means you have to mark the comet only in two images.
- Mark the same star as a comet in the first and last image. It's best to choose a star near the comet.
- These two images must also be checked in the leftmost column in the file list.
- Save the file list. It's not required to close DSS.
- The motion of a comet can be found for example in Guide9.1, if you make a right click on the comet and then click on "More info". The RA and DE motions are given in the last line in degrees/day. Example: "Motion is -0.42 degrees/day in RA, -0.36 degrees/day in dec"
- Calculate the time difference between the first and last image in the unit "days". Example: 2h55m37s - 2h40m57s = 0.0102d
- Calculate how far the comet has moved in RA and De during this time. Example: -0.42°/d * 0.0102d = -0.00428°, -0.35°/d * 0.012d = -0.00367°
- Calculate the image scale in degrees/pixel. Example for a full frame camera with 36mm image width, 6744 horizontal pixels and a 400mm lens: arctan(36mm / (6744 * 400mm)) = 0.000765°/pixel
- Now you can calculate how man pixels the comet has moved between the two images. Example: X = 0.00428° / 0.000765°/pixel = -5.60 pixel (to the left), Y = -0.00367° / 0.000765°/pixel = -4.80 pixel (downwards)
- Open the file "last_image.info.txt" with an editor. In the 6th line from the top is the position of the comet, for example: Comet = 4022.09, 1957.80
- Now modify these coordinates. To the X coordinate you add the value that you calculated above 4022.09 + (-5.60) = 4016.49 and from the Y coordinate you subtract the value from above. That's because the direction of the Y axis is top down. 1957.80 - (-4.80) = 1962.60
- Save the file with the same filename.
- Open the file list in DSS.
- Check that in the first image the comet is still marked at the same star as before. However in the last image the violet circle must have moved with respect to the star. Check that it has moved in the correct direction of the comet's movement.
• Check if under Settings / Stacking_Settings / Comet the box at "Comet Stacking" is ticked, and then start stacking.

Print out this form for the calculations:

\[ \text{RA\_speed} = \text{RA comet motion in degrees per day} = \]
\[ \text{DE\_speed} = \text{DE comet motion in degrees per day} = \]
\[ \text{T1} = \text{Time of first image} = \]
\[ \text{T2} = \text{Time of last image} = \]
\[ \text{dT} = \text{Time difference in days} = \text{T2} - \text{T1} = \]
\[ \text{RA\_deg} = \text{RA comet movement in degrees} = \text{RA\_speed} \times \text{dT} = \]
\[ \text{DE\_deg} = \text{DE comet movement in degrees} = \text{DE\_speed} \times \text{dT} = \]
\[ \text{S} = \text{Image scale in degrees per pixel} = \text{For Canon 5D-MK4 with 400mm lens: 0.000765°/pixel} \]
\[ \text{RA\_pix} = \frac{\text{RA\_deg}}{\text{S}} = \]
\[ \text{DE\_pix} = \frac{\text{DE\_deg}}{\text{S}} = \]
\[ \text{X\_old} = \text{Original X value in info.txt} = \]
\[ \text{Y\_old} = \text{Original Y value in info.txt} = \]
\[ \text{X\_new} = \text{New X value in info.txt} = \text{X\_old} + \text{RA\_pix} = \]
\[ \text{Y\_new} = \text{New Y value in info.txt} = \text{Y\_old} - \text{DE\_pix} = \]
26 C# Programming project / Digital maps and elevation data

From time to time I have to deal with C# programming to stay in practice. This year’s task is: A digital topographic map is available for an area of approx. 10km x 10km. In addition, digital elevation data is available, which is to be superimposed on the map. Virtual noise sources are to be placed along the roads on the map. Then those places should be found that are affected as little as possible by the noise sources. It should be assumed that the sound propagates in a straight line and can be shadowed by mountains. We are therefore looking for places from which all noise sources are hidden behind mountains. These are the places where I want to record nature sounds.

26.1 Where is the best place for recording nature sounds?

That's more difficult than you might think, because you have to find a place without any disturbing noise:

• Road and rail traffic requires a distance of several kilometres. It's helpful if there is no direct line of sight, i.e. mountains in between are advantageous.
• If you don't want to record the sound of running water, you have to avoid valleys.
• Wind noise is disturbing already at quite small wind speeds despite fur windshield. Wind noise can be attenuated by a high pass filter. However on days with strong wind it's wasted time to make a record.
• Airplanes cause that approx. 50% of the sound recordings are unusable (even in the Harz Mountains in Germany, where there is no large airfield within a radius of 80km).
26.2 Digital topographic maps

A very good source for free digital topographic maps is OpenTopoMap, the web version is here: https://opentopomap.org/#map=13/51.66473/10.42482
On this page is briefly mentioned (in german) how tiles of size 256x256 pixels can be downloaded: https://opentopomap.org/about
This is a sample download of one tile (this is the place where I live): https://a.tile.opentopomap.org/13/4331/2718.png
In this case the zoom level is 13, the X value is 4331 and the Y value is 2718.
This page contains very detailed explanations about the folder structure and mathematics of the zoom levels and coordinates: https://wiki.openstreetmap.org/wiki/Slippy_map_tilenames

Coordinate transformation from latitude and longitude to X and Y:

\[
\begin{align*}
n &= 2^\text{zoom} \\
x &= n \times \frac{(\text{lon}_\text{deg} + 180) \, / \, 360}{360} \\
y &= n \times \frac{1 - \left(\log(\tan(\text{lat}_\text{rad}) + \sec(\text{lat}_\text{rad})) \, / \, \pi\right)}{2}
\end{align*}
\]

Coordinate transformation from X and Y to latitude and longitude:

\[
\begin{align*}
n &= 2^\text{zoom} \\
\text{lon}_\text{deg} &= \frac{X}{n} \times 360.0 \, - \, 180.0 \\
\text{lat}_\text{rad} &= \arctan(\sinh(\pi \times (1 - 2 \times Y \, / \, n))) \\
\text{lat}_\text{deg} &= \text{lat}_\text{rad} \times 180.0 \, / \, \pi
\end{align*}
\]

This returns the north west (top left) corner of the tile. Use X+1 and/or Y+1 to get the other corners. Use X+0.5 and Y+0.5 to get the coordinates of the tile's center.

Calculate the resolution:

\[
\text{resolution} = 156543.03 \, \text{meters/pixel} \, \times \, \cos(\text{latitude}) \, / \, (2 \, ^\text{zoom})
\]
26.3 Digital elevation data

I found several sources for free digital elevation data:


- NASA Earthdata Search: [https://search.earthdata.nasa.gov/search](https://search.earthdata.nasa.gov/search) After registering you can download free elevation data with 1 arcsec resolution (which is about 27m in latitude, and less than 27m in longitude). Search for the ASTER Digital Elevation Model (AST14DEM): [https://lpdaac.usgs.gov/products/ast14demv003/](https://lpdaac.usgs.gov/products/ast14demv003/) However the elevation data seems to be inaccurate, as I found some peaks in the mountains about 40m too low.

- [https://opendem.info/download_srtm.html](https://opendem.info/download_srtm.html) This is a very large 272MB GeoTiff file of Germany with 13201x10801 pixels. The resolution is 1200 pixels per degree. You can choose between surface data (including buildings and vegetation) and terrain data (without buildings and vegetation). The elevation data isn't perfect, as I found some peaks in the mountains up to 17m too low. But for my project that's good enough and I did use the terrain data file.

This really large 16-bit GeoTiff file contains Germany from longitude 5° to 16° and from latitude 47° to 56°. It covers a 11° longitude range with 13201 pixels and a 9° latitude range with 10801 pixels. The top left corner is at 5° longitude and 56° latitude. Please note that the resolution (in Meter) is different for longitude and latitude. The pixels aren't square. The pixel size is about 92.63m x 149.74m.

Of course we need only a small part of the map, so how to crop it? GeoTiff's can be read the same way as Tiff's, but the software must be able to read and write 16-bit data.

- IrfanView can read 16-bit Tiff, but converts it internally to 8-bit.
- Fitswork can read and write 16-bit Tiff and crop the region, but it can't write PGM files.
- Gimp can read and write 16-Bit Tiff. You can crop the region as follows: Make a double click on the "Rectangle Select Tool". Draw a rectangle in the picture. Now fill in the values for position and size. Then use "Image / Crop_to_Selection". Gimp can also save the output image as 16-bit ASCII PGM (P2 Portable Gray Map) file, which is easy to read by C# code.
- FFmpeg can read and write 16-bit Tiff and also crop the region. It can write binary PGM (P5) files, but unfortunately it can't write ASCII PGM (P2) files.
Here is an example for cropping a region of the GeoTiff with FFmpeg:

```plaintext
rem Crop a large GeoTiff file and save it as Tiff or binary PGM file

set "FF=c:\ffmpeg\ffmpeg" :: Path to FFmpeg
set "IN=srtm_germany_dtm.tif" :: Input GeoTiff file
set "WIDTH=264" :: Width
set "HEIGHT=131" :: Height
set "LEFT=6340" :: Left edge
set "TOP=5128" :: Top edge
set "OUT=elevation.tif" :: Output Tiff or binary PGM file; it isn't possible to write an ASCII PGM file

%FF% -i %IN% -vf format=pix_fmts=gray16le,crop=%WIDTH%:%HEIGHT%:%LEFT%:%TOP% -y %OUT%

pause
```

However, it's also possible to read the GeoTiff file directly with C# code, using the external BitMiracle.LibTiff.NET library. I did use this library in my C# code.


This is the link to the library: https://bitmiracle.com/libtiff/
26.4 Noise map (red = traffic noise, blue = running water noise)
27 Astronomy

27.1 What's the best time for moon observing?

That depends on the moon's phase as follows:

<table>
<thead>
<tr>
<th>Moon phase</th>
<th>Largest altitude:</th>
<th>When has the ecliptic the steepest angle to the horizon?</th>
</tr>
</thead>
<tbody>
<tr>
<td>New Moon</td>
<td>(not during darkness)</td>
<td></td>
</tr>
<tr>
<td>Waxing crescent</td>
<td>(not during darkness)</td>
<td>Spring, March 20, at sunrise</td>
</tr>
<tr>
<td>First quarter</td>
<td>Spring, March 20, 18:00 o'clock</td>
<td></td>
</tr>
<tr>
<td>Waxing gibbous</td>
<td>Winter, February 5, 21:00 o'clock</td>
<td></td>
</tr>
<tr>
<td>Full moon</td>
<td>Winter, December 20, 0:00 o'clock</td>
<td></td>
</tr>
<tr>
<td>Waning gibbous</td>
<td>Autumn, November 5, 3:00 o'clock</td>
<td></td>
</tr>
<tr>
<td>Last quarter</td>
<td>Autumn, September 20, 6:00 o'clock</td>
<td></td>
</tr>
<tr>
<td>Waning crescent</td>
<td>(not during darkness)</td>
<td>Autumn, September 20, at sunset</td>
</tr>
</tbody>
</table>

The given dates are a rough estimate plus or minus one month, only valid for observers on the northern hemisphere.
### 27.2 Limiting magnitude for video astronomy

<table>
<thead>
<tr>
<th>Lens</th>
<th>Camera</th>
<th>Video mode</th>
<th>ISO</th>
<th>Limiting magnitude</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Canon EF 400mm f/2.8 + SpeedBooster 0.64x</td>
<td>Panasonic GH5S</td>
<td>FHD 25fps. [Ex. Tele Conv.] = 2.1x</td>
<td>25600</td>
<td>about 12.2 mag</td>
<td>Sky wasn't perfectly clear, with 4x contrast enhancement, no noise reduction</td>
</tr>
</tbody>
</table>

### 27.3 Limiting magnitude for stacked exposures

<table>
<thead>
<tr>
<th>Lens</th>
<th>Camera</th>
<th>Exposure</th>
<th>ISO</th>
<th>Limiting magnitude</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Canon EF 400mm f/2.8</td>
<td>Canon 5D MK4</td>
<td>173 x 30s = 86.5 min</td>
<td>3200</td>
<td>about 18.5 mag</td>
<td>Stacked with DeepSkyStacker, sky wasn't perfectly clear</td>
</tr>
</tbody>
</table>

### 27.4 Useful calculations for Magnitudes

**Convert magnitude $m_V$ [mag] to illuminance $E_V$ [Lux]:**

$$E_V = 10 ^ {(-14.18 - M_V) / 2.5}$$

**Convert illuminance $E_V$ [Lux] to magnitude $m_V$ [mag]:**

$$M_V = -14.18 - 2.5 * \log_{10} E_V$$

**Convert illuminance $E_V$ [Lux] to irradiance $E_E$ [W/m^2], for wavelength 555nm:**

248
\[ E_V = \frac{E_E}{683 \text{ lx/W}} \]

Convert irradiance \( E_E \) [W/m\(^2\)] to illuminance \( E_V \) [Lux], for wavelength 555nm:
\[ E_V = E_E \times 683 \text{ lx/W} \]

## 27.5 Crab Pulsar

The crab pulsar is a supernova remnant and consists of a neutron star which is rapidly spinning with a frequency of about 30 Hz. It emits pulses in radio, visual, X-ray and gamma spectral range.

https://en.wikipedia.org/wiki/Crab_Nebula#Central_star

The frequency is slowly decreasing and the latest measurement results can be found here:

https://heasarc.gsfc.nasa.gov/W3Browse/all/crabtime.html

http://www.jb.man.ac.uk/~pulsar/crab/crab2.txt

In the book "Paul Horowitz, Winfield Hill: The Art of Electronics, Second Edition, Page 1030ff" is described how to measure the light curve with a 60" telescope, a photomultiplier and a signal averager. They used 5 million sweeps which is more than 41 hours of sampling time.

When averaging the signal, three effects must be considered:

1. The pulsar's frequency decreases by about 1.326µHz per hour.

2. The Doppler effect due to earth's rotation. The velocity of the observer is: \( V = 2 \times \pi \times 6370\text{km} / 86400\text{s} \times \cos(\text{latitude}) \)
   For 51.5° latitude the velocity is 0.288 km/s towards the east point of the local horizon.
   The Doppler frequency shift is \( f_D = f \times \frac{V}{c} \) where \( c \) is the speed of light 300000 km/s.
   For \( f = 30\text{Hz} \) the Doppler frequency shift is 28.8µHz at the east horizon and -28.8µHz at the west horizon. The frequency shift is 0 at the meridian.
   Near the meridian the Doppler frequency shift decreases by 7.5µHz per hour.

3. The Doppler effect due to earth orbiting around the sun. The velocity of the observer is: \( V = 2 \times \pi \times 149.6e6 \text{ km} / 365.25\text{d} / 86400\text{s} = 29.786 \text{ km/s} \)
   towards a point on the ecliptic which is about 90° west of the sun.
   The Doppler frequency shift is \( f_D = f \times \frac{V}{c} \) where \( c \) is the speed of light 300000 km/s.
   For \( f = 30\text{Hz} \) the Doppler frequency shift is 99.3µHz on the ecliptic 90° west of the sun and -99.3µHz on the ecliptic 90° east of the sun. At midnight in winter the pulsar is approximately 180° away from the sun, so that the Doppler frequency shift is small. Under these conditions the frequency shift decreases by 2.13µHz per hour.
Adding these three effects, the observed pulsar frequency decreases by about 11µHz per hour (valid only in winter at midnight, when the pulsar is in the south near the meridian).

An error $\Delta f$ in the reference frequency will produce after time $t$ a phase error $\Delta p$ with respect to the pulsar phase as follows: $\Delta p / 360^\circ = t \times \Delta f$

Example: If the reference frequency is off by 10µHz, after 2 hours there will be a phase error of 25.92°.

The latest exact frequency of the pulsar is from December 15, 2019:

$f = 29.6122791665 \text{ Hz} - 0.000001326 \text{ Hz / hour}$
28 DCF77 Decoding

DCF77 is a 77.5kHz longwave time signal transmitter in Germany. The time signal is coded with 59 short (100ms = "0") or long (200ms = "1") impulses as follows:

<table>
<thead>
<tr>
<th>Bit</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Start of minute, is always 0</td>
</tr>
<tr>
<td>1-16</td>
<td>Civil warning bits and other informations</td>
</tr>
<tr>
<td>17</td>
<td>CET: 0   CEST: 1</td>
</tr>
<tr>
<td>18</td>
<td>CET: 1   CEST: 0</td>
</tr>
<tr>
<td>19</td>
<td>Leap second announcement</td>
</tr>
<tr>
<td>20</td>
<td>Always 1</td>
</tr>
<tr>
<td>21</td>
<td>Minutes 1</td>
</tr>
<tr>
<td>22</td>
<td>Minutes 2</td>
</tr>
<tr>
<td>23</td>
<td>Minutes 4</td>
</tr>
<tr>
<td>24</td>
<td>Minutes 8</td>
</tr>
<tr>
<td>25</td>
<td>Minutes 10</td>
</tr>
<tr>
<td>26</td>
<td>Minutes 20</td>
</tr>
<tr>
<td>27</td>
<td>Minutes 40</td>
</tr>
<tr>
<td>28</td>
<td>Even parity over minute bits 21-2</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Bit</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>29</td>
<td>Hours 1</td>
</tr>
<tr>
<td>30</td>
<td>Hours 2</td>
</tr>
<tr>
<td>31</td>
<td>Hours 4</td>
</tr>
<tr>
<td>32</td>
<td>Hours 8</td>
</tr>
<tr>
<td>33</td>
<td>Hours 10</td>
</tr>
<tr>
<td>34</td>
<td>Hours 20</td>
</tr>
<tr>
<td>35</td>
<td>Even parity over hours bits 29-35</td>
</tr>
<tr>
<td>36-41</td>
<td>Day of month</td>
</tr>
<tr>
<td>42-44</td>
<td>Day of week</td>
</tr>
<tr>
<td>45-49</td>
<td>Month number</td>
</tr>
<tr>
<td>50-57</td>
<td>Year within century</td>
</tr>
<tr>
<td>58</td>
<td>Even parity over the date bits 36-58</td>
</tr>
<tr>
<td>59</td>
<td>No impulse</td>
</tr>
<tr>
<td>0</td>
<td>Previous defined time is valid at the beginning of this impulse (which is always 0)</td>
</tr>
</tbody>
</table>

Schematic diagram of a DCF77 receiver which produces 2kHz beeps for recording with the GH5S camera:
Acknowledgements
I found many of the FFmpeg hints and examples somewhere in the internet. Thanks to all who posted them!

Thanks to all who contributed to the great FFmpeg software, and special thanks to Carl Eugen Hoyos, Paul B Mahol, Moritz Barsnick and Gyan Doshi who answered many questions on the FFmpeg user mailing list.

Thanks to Bill Gray for writing the best astronomy software (Guide 9.1), and to Jost Jahn and Larry Wood for help with Guide 9.1.

Thanks to all who contributed to DeepSkyStacker software.

Thanks to all contributors to OpenStreetMap, OpenTopoMap, OpenDEM and Sergey Bobrovsky and Vitaliy Shibaev of BitMiracle for the LibTiff.net library.
# Index

| 0-padded | acos | asin | 109 | 66, 132 |
| 0x07     | acoust | acoustic feedback | 132 | 148 |
| 0xABB    | across | acrossfade | 141 | 143 |
| 0xGG     | crossover | 136 | 47 | 148 |
| 7artisans Viltrox 7.5mm | aheader | aheader | 162 | 148 |
| 10 bit video | addition | 128 | 162 |
| 10-bit video | adrawgraph | 128 | 128 |
| 16-Bit.. | aevalsrc | 25, 148, 158, 160 | 128 |
| 16-bit GeoTiff | adelay | 152 | 25 |
| 1kHz tone | afade | 25, 113, 120 | 152 |
| 2.1x Telescopic effect | afade curves | 25 | 152 |
| 2.5mm connector | afltt | 136, 155, 161 | 152 |
| 2k       | alpha_mask | 70, 78, 89, 91 | 152 |
| 2kdc     | amix | 152 | 152 |
| 2kflat   | Amplifier | 146 | 152 |
| 2x2 mosaic | amplify | 21, 136 | 152 |
| 3.5mm headphone output | amultiply | 148, 154 | 152 |
| 3.5mm input jack | Anamorphic | 189 | 152 |
| 3.5mm stereo connector | Anamorphic 10 bit Modes GH5S | 214 | 152 |
| 3D LUT Creator | Anamorphic 8 bit Modes GH5S | 213 | 152 |
| 3x3 neighborhood | Animated GIF | 44f. | 152 |
| 4:2:0    | animation | 106 | 152 |
| 4:2:2    | anoisesrc | 161 | 152 |
| 4:4:4    | anullsrsc | 36 | 152 |
| 4k       | apad | 148, 162 | 152 |
| 4k10 bit Modes GH5S | Aperture | 190 | 152 |
| 4k8 bit Modes GH5S | Aperture number | 192 | 152 |
| 4kdc     | Apprehension Engine | 226 | 152 |
| 4kflat   | apt-get install ffmpeg | 187 | 152 |
| 7artisans Viltrox 7.5mm | aresample | 113, 144 | 152 |
| Abbreviations, GH5S | areverse | 25 | 152 |
| abs      | Artefacts | 55 | 152 |
| Acknowledgements | ASCII PGM (P2) | 141 | 152 |
|           | asetamples | 128 | 152 |
|           | asetrate | 148 | 152 |
|           | atan | 132 | 152 |
|           | atan2 | .63f., 69, 88, 132 | 152 |
|           | atempo | 148 | 152 |
|           | Atmospheric dispersion correction | 21 | 152 |
|           | atrim | 152 | 152 |
|           | Audible range | 148 | 152 |
|           | Audio processing | 143 | 152 |
|           | Audio sample rate | 144 | 152 |
|           | Audio volume level | 146 | 152 |
|           | AVCHD | 200 | 152 |
|           | averaging | 115 | 152 |
|           | ball | 57, 67, 89, 91 | 152 |
|           | Band-limited noise | 161 | 152 |
|           | Banding artefacts | 55 | 152 |
|           | Banding artefacts | 55 | 152 |
|           | barrel | 57 | 152 |
|           | Batch file | 8 | 152 |
|           | Batch files (Windows 7) | 170 | 152 |
|           | Baths | 148 | 152 |
|           | Beginning of video | 23 | 152 |
|           | Betagear FX82USB mixer | 229 | 152 |
|           | between | 73, 86, 132f., 161 | 152 |
|           | Big eyes | 99 | 152 |
|           | Bigger nose | 99 | 152 |
|           | bitand | 132 | 152 |
|           | BitMiracle | 245 | 152 |
|           | bitor | 132, 161 | 152 |
|           | black | 23 | 152 |
|           | Black hole | 87, 141 | 152 |
Longitud...243
Look-up-Table...200
lossless...111
Low-noise...163
lowpass...148, 154
lt...133, 160
lt...133
lum_expr...73
Luminance Level GH5S...201, 203
LUT...175, 200
Lux...248
M42...192
mag...248
Magnitude...248
make...187
Mark Korven...226
maskedmerge...75
Master Pedestal Level GH5S...203
max...133
max_colors...45
MB/min...206
MB/s...206
Mechanical / Electronic Shutter GH5S...204
medium...106
Meike 6-11mm...190
Meike 6-11mm f/3.5...207
Meike 6.5mm...190
Meike 8mm...190
mercator...57
Metadata...107
MFT...190, 192
Mic_Socket...225
Microphone...150, 154
Microphone cable...163
min...133
minterpolate...122
Mirror sphere...66
Mirror-sphere video...87
Mixing frequency...148
mod...34, 133, 160
Monochrome picture...147
Moon phase...247
Motion interpolation...122
Mount...190
MOV...200
mov_text...131
Moving black hole...86
Moving wormhole...90
MP4...200
MP4 (LPCM)...200
MP4 HEVC...200
mpeg4...106
Music...146
Mute...184
Nature sounds...146, 242
Needle impulse...160
generate...18, 193
Negative...18
Neuton star...249
Night sky videos with GH5S...118
Nikon D800...189, 191
Nikon F...192
Nippon Kogaku 8 mm...190
Nippon Kogaku 8 mm f/2.8...202, 207
Nippon Kogaku 8 mm fisheye...118
nlmeans...102
Noise map...246
Noise reduction...102
Noise Reduction...200
Norse...97
NR...200
ntsc...112
null (video filter)...107
nullsrc...63, 75, 88, 97, 106
Number of Pixels GH5S...203
Olympus M.Zuiko 8mm...190
Open-source...5
OpenTopoMap...243
Opteka 6.5mm...190
OS-X...5
OSCILLATOR...224
oscilloscope...129
Overlapping fisheye videos...75
overlay...46, 70, 78, 88f., 91, 95, 113, 120, 124, 128, 139
Overtone...158
owdenoise...102
pad...27, 62, 95, 116, 236
pal...112
palette...45
paletteuse...45
PAM...111
Panasonic GH5S...125, 248
Panasonic GH5S at 240fps...121
Panasonic LUMIX DC-GH5S...189
Panasonic LUMIX GH5S...188, 191, 200, 225
pannini...57
Panorama...59
Panoramic camera...219
PanoView XD360...113, 188
PanoView XD360 camera...217
Passing the FFmpeg output to FFplay...153
Patch Input/Output...185
Paul Bourke...59
pause...8
Perceptual Quantization...200
perspective...57, 78, 95, 139
PGM...111, 141
PGMYUV...111
Phantom power...163
Pl...59, 88, 133
pitch...56, 62, 69, 88, 236
pivot...138
pix_fmts...63f., 104
Pixel format...104
Pixels...189
Remove low frequencies
Remap filter
Rectangular wave
Recording nature sounds
Record sound
Record Formats, GH5S
Read-out Chip Size
random
Proper motion of stars
prores_ks
PTS
pts
Pulnix TM-9701
random
Re-numbering images
Read-out chip size
Read-out Chip Size
Rec Quality
Record Formats, GH5S
Record sound
Recording duration on SD cards
Recording nature sounds
Rectangular wave
rem
remap
Remap filter
Remap fisheye to equirectangular
remap=fill=green
Remove low frequencies
removegrain
ren
Replace one frame by another
reserve_transparent
reverse
rgb48
rgbshift
RODE NT1 Microphone
roll
roder
rotate
Rotating earth or planet
Rotation order
round.
Royalty-free music
RS232 port.
Rudy Rucker
Running clock
Running water
Sachtler
Sample rate
Samyang 8mm Fisheye II
SAR
Saturation
scale
Schwarzschild radius
scroll
sd2
SDR
Segment
Segment-wise linear interpolation
selectivecolor
sendcmd
sendcmd examples
sense
Sequential number
Serial port
set
set /a
setpts
settings, GH5S
sg
SGI
sgn
Short test tone
shortest=1
Sigma 14mm f/1.8
Sigma 24mm f/1.4
Sigma EX DG 4.5mm
Sigma EX DG 4.5mm f/2.8
Sigma EX DG 8mm
Sigma EX DG 8mm f/3.5
Signal to Noise Ratio
sin.
sin.
sine
Sine tone
Single quotes
sinusoidal
Sinusoidally rising and falling tone
Size of color-look-up tables
slow
Slow motion
Slow motion ramp
slower
Smaller nose
SMTSEC 2.27mm
SNR
Solo
Sony A7S II
Sony E
Sony E-Mount
Sound effects
Sound effects for horror films
Sound records
Sound_Rec_Level_Adjust
Sound_Rec_Level_Limiter
Source code
Speakers
speed factor
Speed of sound
Wormhole..............................................................90
Wormhole simulation.............................................87
wuxga.................................................................112
X-Rite ColorChecker..............................................174
xfade.................................................................26, 141
xga.................................................................112
xmap......................................................................59
xstack.................................................................43, 52, 118
Y'CbCr.................................................................105
yaw.......................................................................56, 88
ymap......................................................................59
Yumiki 2.5mm.........................................................190
yuva444p...............................................................105, 134
yuva444p10le.........................................................105
yuva422p...............................................................105
yuva422p10le.........................................................105
yuva444p10le.........................................................104f.
Zebros GH5S..........................................................201
Zeranoe.....................................................................5
zerolatency............................................................106
ZLKC (OCDAY) 7.5mm.............................................190
Zoom level.............................................................243
zoompan..................................................................13, 120
ZWO ASI178MM....................................................189, 192
^ character..............................................................8
^Q.................................................................171
-ac......................................................................144
-acodec..................................................................11
-af.....................................................................120, 168
-alwaysontop..........................................................168
-an.....................................................................28, 168
-audio_buffer_size.................................................151, 155
-autoexit............................................................77, 161, 168
-bits_per_mb.........................................................106
-c copy...............................................................22, 24
-c:s.................................................................131
-c:v libx264..........................................................106
-c:v mpeg4...........................................................106, 152
-c:v prores ks.......................................................106
-channels............................................................150
-codec:v............................................................9
-crf..................................................................11, 106
-exitonclick.........................................................168
-exitonmousedown................................................168
-f.......................................................................168
-f dshow.............................................................130, 154
-f lavfi..................................................................30f.
-f nut.................................................................153f, 161
-f sdl2.................................................................77, 151
-filter_complex..................................................48, 116
-filter_complex_script..........................................103
-frame_pts.........................................................109
-framerate...........................................................9, 12, 44, 130
-fs.....................................................................77, 168
-hide_banner.......................................................106
-i color...............................................................70
-itsoffset............................................................156
-lavfi...................................................................30f.
-list_devices......................................................130, 154
-list_options.......................................................130, 150, 154
-loop.................................................................44, 147, 168
-map..................................................................23, 151
-metadata:s:s:0....................................................131
-noborder..........................................................168
-nodisp..............................................................168
-pix_fmt.........................................................34, 104, 106, 123f, 134
-pix_fmts..........................................................104
-pixel_format......................................................130
-preset.............................................................11, 106
-preset help.........................................................106
-profile:v..........................................................106
-q:v.................................................................9
-r.................................................................12, 44
-re...............................................................151
-s.................................................................168
-shortest.........................................................9, 42, 152
-ss.................................................................143, 168
-sseof..............................................................15, 24
-start_number...................................................9, 91, 109
-strftime..........................................................109
-t.................................................................143, 176
-to.................................................................22
-tune...............................................................106
-vcodec.............................................................11
-vf.....................................................................16, 168
-video_size.......................................................130
-vn.................................................................168
-y.................................................................15
-y option...........................................................175
.*.bat.................................................................8
.*.srt...............................................................143, 176
#!/bin/bash........................................................187
% character.........................................................8
%~*.................................................................170
%0nd..............................................................109
%1d...............................................................166
%d...............................................................109
%H...............................................................109
%m...............................................................109
%M..............................................................109
%nd..............................................................109
%S...............................................................109
%Y...............................................................109

261