Eine Einführung in FFmpeg

Zeitraffer, Video-Bearbeitung, Vertonung, Effekte, Video-Schnitt

oder

Wie mache ich einen Film für Fulldome-Projektion?

Michael Koch, Sternwarte St. Andreasberg

Version vom 2.2. 2019

FFmpeg ist eine kostenlose Software. Die Windows-Version können wir hier runterladen:

https://ffmpeg.zeranoe.com/builds/

- **Die aktuelle Version auswählen, 64-bit oder 32-bit auswählen, "Static" auswählen, dann auf "Download Build" klicken.**
- **Die ca. 65MB große ZIP-Datei öffnen, und aus dem Ordner "bin" die beiden Dateien "ffmpeg.exe" und "ffprobe.exe" in einen neuen Ordner c:\ffmpeg kopieren. Es ist keine weitere Installation erforderlich.**
- **"ffmpeg.exe" ist das Programm für die Bearbeitung von Videos, mit dem wir uns in folgenden näher beschäftigen werden.**
- **"ffprobe.exe" ist ein Programm, mit dem man sich die Eigenschaften von Videos, Bildern oder Audio-Dateien anzeigen lassen kann. Es ist nützlich bei der Fehlersuche, falls es Probleme beim Zusammenfügen von mehreren Videos gibt.**
- **"ffplay.exe" ist ein Programm zum Abspielen von Videos. Wir brauchen es nicht wenn wir stattdessen den VLC-Player verwenden.**
- **Außerdem speichern wir aus dem Ordner "doc" die Datei "ffmpeg-all.html" irgendwo ab, wo wir sie jederzeit wiederfinden (z.B. auf den Desktop legen). Diese Datei enthält die komplette englische Dokumentation für FFmpeg. Hier sind alle Funktionen mehr oder weniger ausführlich beschrieben. Wichtig sind insbesondere die Kapitel "Audio Filter" und "Video Filter".**

Welche Vor- und Nachteile hat FFmpeg gegenüber anderen Videobearbeitungs-Programmen?

- **Sehr mächtiger Funktionsumfang, wird ständig weiterentwickelt**
- **Konvertierung von fast jedem Format in fast jedes andere Format**
- **Keine Beschränkung auf bestimmte Abmessungen (Breite * Höhe), beliebige Auflösungen sind möglich**
- **Es gibt eine Mailing-Liste wo man (auf englisch) Fragen stellen kann. Fragen sollten sich immer auf die letzte verfügbare Version beziehen, und es sollte immer die Kommandozeile und die komplette FFmpeg-Ausgabe mit angegeben werden.**
- **FFmpeg ist ein Kommandozeilen-Programm und hat keine grafische Oberfläche. Das hört sich zunächst nach einem schweren Nachteil an. Aber es hat auch einen großen Vorteil:**

  Bei einem Programm mit grafischer Windows-Oberfläche entsteht das fertige Video durch eine Abfolge von vielen Mausklicks. Wenn man zum Schluss merkt, das man ganz am Anfang irgendwas falsch gemacht hat, dann muss man wieder von vorne anfangen. Bei FFmpeg geht das viel einfacher: Man ändert einfach die Kommandozeile (die man als Batch-File abgespeichert hat) und lässt FFmpeg nochmal durchlaufen.
- **Man braucht eine gewisse Zeit für die Einarbeitung**
- **Die Dokumentation ist leider nicht immer aktuell, und oft wünscht man sich dass mehr Beispiele angegeben wären.**
- **Die Beispiele in diesem Vortrag sollen helfen, die Einarbeitungszeit etwas zu verkürzen. Alle Beispiele sind getestet und stehen zum Download zur Verfügung.**

Was kann man mit FFmpeg machen? Eine kleine Auswahl der wichtigsten Funktionen:

- **Videos von einem Format in ein anderes Format konvertieren**
- **Aus vielen Einzelbildern ein Video machen**
- **Ein Video in viele Einzelbilder zerlegen**
- **Teile aus einem Video herausschneiden**
- **Ton hinzufügen oder entfernen**
- **Lautstärke verändern**
- **Die Größe (Breite * Höhe) verändern**
- **Ausschnitte herausvergrößern oder Ränder abschneiden, z.B. ein rechteckiges Bild quadratisch machen**
- **Zeitlupe, Zeitraffer**
- **Drehen, Spiegeln**
- **Texte ein- und ausblenden**
- **Korrektur von Helligkeit, Kontrast, Gamma, Farbsättigung, Farbtemperatur, auch mit Look-Up-Tables**
- **Maskierung (Fenster-Funktion)**
- **Ein- und Ausblendungen, Überblendungen für Bild und Ton**
- **Verzerrungen oder Entzerrungen, z.B. gebogene Texte für Fulldome, oder Simulation der Raumzeitkrümmung in der Nähe von schwarzen Löchern**
- **Stabilisierung von verwackelten Videos**
- **Deflicker zur Beseitigung von Helligkeits-Sprüngen in Zeitraffer-Videos**
- **Komprimierungsgrad von Videos verändern, um die Datei kleiner zu machen**
- **und noch vieles mehr...**

Wenn FFmpeg keine grafische Bedienoberfläche hat, wie wird es dann bedient?

Es gibt zwei Möglichkeiten:

1. **Alle_Programme / Zubehör / Eingabeaufforderung**

   Das ist aber nicht empfehlenswert, denn dann müssten wir die lange Kommandozeile jedesmal von Hand eingeben. Das ist viel zu mühsam und fehleranfällig.
2. **Wir schreiben ein Batch-File, wo die FFmpeg Kommandozeile drinsteht.**

Ein Batch-File hat die Extension ".bat" und kann mit einem Text-Editor erzeugt und bearbeitet werden. Dabei sind einige Dinge zu beachten:

- **Am einfachsten ist es, wenn man mit einem bereits vorhandenen funktionsfähigen Batch-File anfängt und dann Änderungen darin vornimmt. Am Anfang ist es ratsam immer nur wenig zu verändern, und dann laufen lassen und schauen ob es noch funktioniert.**
- **Das % Zeichen hat innerhalb des Batch-Files eine andere Bedeutung. Wenn wir in der FFmpeg Kommandozeile ein % Zeichen brauchen, dann müssen wir im Batch-File zwei %% Zeichen hinschreiben.**
- **Es ist sinnvoll am Ende des Batch-Files das Kommando "pause" einzufügen. Das bewirkt, dass auf einen Tastendruck gewartet wird. Ohne dieses Kommando würde sich das Fenster sofort schließen wenn FFmpeg fertig ist, und wir würden eventuelle Fehlermeldungen nicht sehen.**
- **Mit dem Kommando "set" kann man im Batch-File Variablen definieren**
- **Mit dem Kommando "rem" kann man Kommentare einfügen, damit man später noch versteht wie das ganze funktioniert. Kommentare können auch innerhalb einer Zeile mit einem doppelten Doppelpunkt beginnen und gehen dann bis zum Ende der Zeile.**
- **Falls die Kommandozeile zu lang wird, kann man am Ende der Zeile ein ^ Zeichen setzen und dann in der nächsten Zeile weiterschreiben.**
- **Im Texteditor muss beim Abspeichern als Dateityp "Alle Dateien" ausgewählt werden. Falls man aus Versehen "Textdateien" auswählt, dann würde "Dateiname.bat.txt" erzeugt werden. Richtig wäre "Dateiname.bat".**
- **Wenn man den Inhalt des CMD-Fensters kopieren möchte, geht man so vor: Rechtsklick auf die Titelleiste des Fensters, Bearbeiten --> Alles auswählen. Dann mit Control-C markieren und mit Control-V irgendwo anders einfügen.**

Ein einfaches Batch-File kann zum Beispiel so aussehen:

```
rem  Ein einfaches Batch-File, um viele Einzelbilder zu einem Video zu kombinieren

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     :: Warte auf Tastendruck
```

Was bedeuten die einzelnen Teile?

`rem  Ein einfaches ...`	**Das ist ein Kommentar**
`c:\ffmpeg\ffmpeg`	**Das ist der Pfad zu FFmpeg.exe**
`-framerate 5`	**Gibt an, mit welcher Framerate die Bilder eingelesen werden, in diesem Fall 5 Bilder pro Sekunde**
`-start_number 3551`	**Gibt an, dass das erste Bild die Nummer 3551 hat**
`-i IMG_%%4d.jpg`	**Das ist die erste Eingabe-Datei, in diesem Fall viele Bilder, wobei %%4d für eine vierstellige Zahl steht, d.h. das erste Bild heisst IMG_3551.jpg und die Nummer wird sooft um 1 erhöht, bis kein Bild mehr gefunden wird. Bei dreistelligen Zahlen würde man %%3d schreiben.**
`-i birds.mp3`	**Das ist die zweite Eingabe-Datei, in diesem Fall eine Audio-Datei**
`^`	**Wenn eine Kommandozeile sehr lang ist, kann man sie mit diesem Zeichen unterbrechen und in der nächsten Zeile weiterschreiben. An FFmpeg wird die gesamte Kommandozeile so übergeben, als ob alles in einer Zeile stehen würde, also ohne dieses ^ Zeichen.**
`-shortest`	**Bewirkt, dass die Länge des erzeugten Videos von der kürzeren der beiden Eingabe-Dateien bestimmt wird**
`-codec:v mpeg4`	**Bewirkt, dass ein MP4 Video erzeugt wird**
`-q:v 2`	**Gibt die Qualität des erzeugten Videos an, 0 ist die beste Qualität, 2 ist normal, 9 ist stärkste Kompression**
`out.mp4`	**Dateiname des erzeugten Videos**
`pause`	**Dieses Batch-Kommando wartet auf einen Tastendruck. Das ist sinnvoll, damit man Zeit hat eventuelle Fehlermeldungen zu lesen.**
`:: Warte auf ...`	**Alles was hinter dem doppelten Doppelpunkt steht ist ein Kommentar**

Wichtig: Parameter werden immer vor der Datei angegeben, auf die sich sich beziehen.

Die Parameter "-framerate 5" und "-start_number 3551" beziehen sich hier auf die erste Eingabe-Datei (IMG_%%4d.jpg).

Die zweite Eingabe-Datei (birds.mp3) hat in diesem Fall keine Parameter.

Die Parameter "-shortest -codec:v mpeg4 -q:v 2" beziehen sich auf das erzeugte Video (out.mp4).

Es ist besserer Programmierstil, wenn wir mit Variablen arbeiten. Dann würde das gleiche Batch-File so aussehen:

```
rem  Beispiel 1   Ein einfaches Batch-File, um viele Einzelbilder zu einem Video zu kombinieren

set "FF=c:\ffmpeg\ffmpeg"    :: Pfad zu ffmpeg.exe
set "FR=5"                   :: Framerate, mit der die Bilder eingelesen werden (Bilder pro Sekunde)
set "SN=3551"                :: Nummer des ersten Bildes
set "IN=IMG_%%4d.jpg"        :: Dateiname der Bilder
set "AUDIO=birds.mp3"        :: Audiodatei
set "QU=2"                   :: MP4 Qualität, 0 ist beste Qualität, 2 ist normal, 9 ist stärkste Kompression
set "OUT=out.mp4"            :: Erzeugte Datei

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

pause                        :: Warte auf Tastendruck
```

Das ist viel übersichtlicher, weil jede Variable in einer neuen Zeile steht und mit einem Kommentar versehen werden kann.

Es ist empfehlenswert alle Variablen mit Großbuchstaben zu schreiben, damit man sie in der Kommandozeile leichter von den FFmpeg-Optionen unterscheiden kann.

Die Variablen kann man nennen wie man möchte. Es dürfen aber keine Umlaute ÄÖÜ verwendet werden.

Man kann so ein Batch-File als Vorlage für ein neues Projekt kopieren und muss dann nur noch die Variablen-Definitionen anpassen. Die eigentliche Kommandozeile muss nicht mehr bearbeitet werden.

Warum steht die Variablen-Definition in " " Anführungszeichen? Das ist nur notwendig, damit man dahinter noch einen Kommentar schreiben kann. Wenn kein Kommentar in der gleichen Zeile steht, kann man die Anführungszeichen weglassen.

Probleme bei der Durchnummerierung von Bildern:

Kameras erzeugen üblicherweise Bilder mit 4-stelliger Nummerierung. Dabei kann es passieren, dass der Zähler von 9999 nach 0001 überläuft. Somit ist die Nummerierung nicht mehr fortlaufend, wie es von FFmpeg erwartet wird. Dieses Problem kann man lösen indem man ein Fenster mit der Eingabeaufforderung öffnet, zum aktuellen Verzeichnis wechselt und dann folgende Befehle ausführt:

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

Hierdurch werden die Bilder in eine fortlaufende Reihenfolge gebracht.

Die ersten beiden Zeilen bewirken, dass auf die mit 0 oder 1 beginnenden Nummern jeweils 1000 addiert wird.

Die dritte Zeile bewirkt, dass bei den mit 9 beginnenden Nummern 9000 subtrahiert wird.

Zwischen Bild 0999 und Bild 1001 fehlt ein Bild. Das ist aber kein Problem, weil FFmpeg in so einem Fall mehrfach versucht, das nächste Bild zu laden. In der Nummerierung dürfen sich also Lücken befinden. Die zulässige Größe der Lücken ist standardmäßig auf 5 eingestellt, kann aber bei Bedarf mit dem Parameter -start_number_range auch verändert werden.

Kostenlose Gema-freie Musik kann man zum Beispiel hier finden: http://opsound.org

Kostenlos und Gema-frei bedeutet aber nicht, dass man damit machen kann was man will. Man sollte sich die Lizenzbedingungen schon genau durchlesen. Beispielsweise wird oft gefordert, dass eine angemessene Anerkennung gegeben wird (d.h. der Name des Interpreten und der Musiktitel wird genannt), und ein Link zu der Lizenz muss im Video sichtbar sein.

Wie kriegt man es hin, dass ein Zeitraffer-Video genauso lang wird wie eine vorhandene Audio-Datei?

Wenn man Musik verwendet, möchte man ja nicht einfach das Ende abschneiden.

Lösung: Wir lesen die Bilder genau mit der Geschwindigkeit ein, dass das Video die richtige Länge bekommt.

Framerate = Anzahl_der_Bilder / Zeit_in_Sekunden

In diesem Beispiel haben wir 30 Bilder und die Audio-Datei ist 20 Sekunden lang, also muss die Framerate 1.5 sein.

```
rem  Beispiel 2   Ein einfaches Batch-File, um viele Einzelbilder zu einem Video zu kombinieren

set "FF=c:\ffmpeg\ffmpeg"    :: Pfad zu ffmpeg.exe
set "FR=1.5"                 :: Framerate, mit der die Bilder eingelesen werden (Bilder pro Sekunde)
set "RATE=30"                :: Erzeugte Framerate
set "SN=3551"                :: Nummer des ersten Bildes
set "IN=IMG_%%4d.jpg"        :: Dateiname der Bilder
set "AUDIO=birds.mp3"        :: Audiodatei
set "QU=2"                   :: MP4 Qualität, 0 ist beste Qualität, 2 ist normal, 9 ist stärkste Kompression
set "OUT=out.mp4"            :: Erzeugte Datei

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

pause                        :: Warte auf Tastendruck
```

In diesem Beispiel haben wir zwei verschiedene Framerates, die unterschiedliche Funktionen haben:

- **-framerate %FR% bezieht sich auf das Einlesen der Bilder.**
- **-r %RATE% gibt die Framerate des erzeugten Videos an.**

Diese beiden Framerates sind völlig unabhängig voneinander und müssen nicht gleich sein.

Wenn die Bilder langsamer eingelesen werden als es der Ausgangs-Framerate entspricht, dann werden automatisch Bilder dupliziert.

Wenn die Bilder schneller eingelesen werden als es der Ausgangs-Framerate entspricht, dann werden automatisch Bilder übersprungen.

Wie kann man einen Überblendungs-Effekt zwischen den Bildern erreichen?

```
rem  Beispiel 3   Ein einfaches Batch-File, um viele Einzelbilder zu einem Video zu kombinieren, mit Überblendung

set "FF=c:\ffmpeg\ffmpeg"    :: Pfad zu ffmpeg.exe
set "RATE=30"                :: Erzeugte Framerate
set "SN=3551"                :: Nummer des ersten Bildes
set "IN=IMG_%%4d.jpg"        :: Dateiname der Bilder
set "QU=2"                   :: MP4 Qualität, 0 ist beste Qualität, 2 ist normal, 9 ist stärkste Kompression
set "OUT=out.mp4"            :: Erzeugte Datei
                             :: A ist die Dauer wie lange jedes Bild gezeigt wird (ohne Überblendung), hier 1.0 sec
                             :: B ist die Dauer der Überblendung, hier 0.5 sec
set "C=3"                    :: setze C = (A+B)/B (von Hand berechnen, dies muss ein Integer sein)
set "D=2"                    :: setze D = 1/B     (von Hand berechnen, dies kann eine Fliesskomma-Zahl sein)

%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                        :: Warte auf Tastendruck
```

Innerhalb des Video-Filters (beginnend mit -vf) haben wir in diesem Beispiel zwei Filter, die nacheinander angewendet werden. Der erste heisst "zoompan", und der zweite heisst "framerate".

Die Variablen C und D müssen vorher von Hand berechnet werden, weil innerhalb des "zoompan" Filters leider keine Ausdrücke berechnet werden können.

Detaillierte Erläuterungen zu diesem Beispiel folgen auf der nächsten Seite.

Detaillierte Erläuterungen zum vorigen Beispiel:

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

In diesem Beispiel werden zwei Video-Filter nacheinander angewendet, die durch ein Komma voneinander getrennt sind.

1. "zoompan", mit den Parametern "d" und "fps"

2. "framerate", mit den Parametern "fps" (muss hier nicht explizit angegeben werden), "interp_start", "interp_end", und "scene"

https://www.ffmpeg.org/ffmpeg-all.html#zoompan

Der zoompan Filter wird hier nicht zum reinzoomen verwendet, sondern nur um die Bilder zu duplizieren und mit einer bestimmten Framerate an den nachfolgenden Filter weiterzugeben.

"d" gibt an, wie oft jedes Einzelbild wiederholt wird.

"fps" ist die Framerate, die von diesem Filter erzeugt wird.

https://www.ffmpeg.org/ffmpeg-all.html#framerate

Der Framerate Filter kann Zwischenbilder aus den benachbarten Bildern interpolieren. Das ist keine Bewegungs-Interpolation, sondern nur ein Crossfade.

"fps" ist die erzeugte Framerate, dieser Parameter muss nicht explizit angegeben werden (Man hätte auch schreiben können framerate=fps=%RATE%:...)

Die übrigen drei Parameter "interp_start", "interp_end", und "scene" geben an, wann interpoliert wird und wann nicht. Die Werte die ich verwendet habe (0, 255, 100) bewirken, dass immer interpoliert wird.

Im Zusammenspiel erzeugen diese beiden Filter eine Aneinanderreihung von Bildern, wobei jedes Bild für eine gewisse Zeit angezeigt wird, gefolgt von einer Überblendung zum nächsten Bild die ebenfalls eine gewisse Zeit dauert. Beide Zeiten sind frei wählbar, das sind die Werte A und B in den Kommentaren. Aus diesen Werten muss man von Hand die Werte C und D berechnen, die dann in der Kommandozeile verwendet werden. Ich habe noch keinen Weg gefunden wie man diese Berechnung automatisieren könnte. Man kann zwar im Batch-File auch Berechnungen durchführen, aber das geht nur für Integer und nicht für Fliesskomma-Zahlen.

Wenn man den zoompan Filter weglassen und nur den framerate Filter verwenden würde, dann würde auf eine Überblendung sofort die nächste Überblendung folgen, ohne dass das Bild zwischenzeitlich für eine gewisse Zeit angezeigt wird. Daher wird der Trick mit dem zoompan Filter verwendet. Jetzt ist es zwar immer noch so dass auf eine Überblendung sofort die nächste Überblendung folgt, aber weil die Bilder zuvor dupliziert wurden, haben wir jetzt auch Überblendungen zwischen gleichen Bildern, und die sehen logischerweise so aus, dass sich das Bild nicht verändert.

Bilder mit unterschiedlicher Anzeige-Dauer zu einem Video zusammenfügen:

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

Im diesem Beispiel wird jedes Bild eine Sekunde lang gezeigt (25 Frames), ausgenommen das dritte Bild 3 Sekunden lang (25+50 Frames) und das fünfte Bild 5 Sekunden lang (25+100 Frames).

Ein Video in viele Einzelbilder zerlegen:

```
rem  Beispiel 4   Ein Video in viele Einzelbilder zerlegen

c:\ffmpeg\ffmpeg -i out.mp4 -vf fps=0.2 -y Bild%%4d.jpg

pause     :: Warte auf Tastendruck
```

Dieses Batch-File liest die Datei out.mp4 ein und erzeugt Einzelbilder mit den Dateinamen

Bild0000.jpg, Bild0001.jpg, Bild0002.jpg, u.s.w.

-vf fps=0.2 bewirkt, dass die Bilder mit einer Framerate von 0.2 erzeugt werden, d.h. alle 5 Sekunden ein Bild

-y bewirkt, dass die Ausgangs-Dateien ohne vorherige Rückfrage überschrieben werden, falls schon ältere Versionen mit den gleichen Dateinamen existieren.

Erzeugung von JPG Testbildern:

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

Wie man das Pixelformat von JPG Bildern von 4:2:0 nach 4:2:2 ändern kann:

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

Korrektur von Helligkeit / Kontrast / Farbsättigung / Gamma / Farbtemperatur:

```
rem  Beispiel 5   Batchfile zur Korrektur von Helligkeit, Kontrast, Farbsättigung, Gamma und Farbtemperatur

set "FF=c:\ffmpeg\ffmpeg" :: Pfad zu ffmpeg
set "INPUT=PanoView.mp4"  :: zu bearbeitendes Video
set "OUTPUT=out.mp4"      :: erzeugtes Video
set "CONTRAST=1.0"        :: Kontrast im Bereich -2.0 bis 2.0, normal 1.0
set "BRIGHT=0.0"          :: Helligkeit im Bereich -1.0 bis 1.0, normal 0.0
set "SATUR=1.2"           :: Farbsättigung im Bereich 0.0 bis 3.0, normal 1.0
set "GAMMA=1.0"           :: Gamma im Bereich 0.1 bis 10.0, normal 1.0
set "HUE=20"              :: Farbkorrektur, negativ in Richtung rot, positiv in Richtung blau, normal 0
                          ::  Typische Werte liegen im Bereich -30...+30
set "QU=2"                :: MP4 Qualität, 0 ist beste Qualität, 2 ist normal, 9 ist stärkste Kompression

%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 steht für "Video Filter". Es gibt sehr viele verschiedene Filter, siehe Kaptiel "Video Filter" in der FFmpeg Dokumentation.

In diesem Fall werden zwei Filter verwendet, die mit einem Komma hintereinandergereiht werden:

- **Der erste Filter heisst "hue" und bewirkt eine Rotation des Farb-Kreises.**
- **Der zweite Filter heisst "eq" und erlaubt Korrekturen für Kontrast, Helligkeit, Farbsättigung und Gamma.**

Aus mathematischer Sicht verhalten sich diese Funktionen so:

- **Kontrast ist eine Multiplikation mit einer Konstanten.**
- **Helligkeit ist die Addition einer Konstanten.**
- **Farbsättigung ist schwieriger zu beschreiben. Farbsättigung = 0 würde ein schwarz/weiss Bild erzeugen.**
- **Gamma ist eine nichtlineare Verzerrung der Übertragungsfunktion.**
  Wenn man den Gamma-Wert erhöht, werden Details in den dunklen Bildbereichen besser sichtbar.

Einen bestimmten Bereich aus einem Video herausschneiden:

Wenn die Fulldome-Kamera auf dem Boden liegt und ein Video aufnimmt, dann ist es kaum zu vermeiden dass man am Anfang des Videos selber zu sehen ist -- weil man die Kamera ja irgendwie starten muss.

Wie kann man den Anfang eines Videos wegschneiden und das Video auf eine bestimmte Länge begrenzen? Das geht so:

```
rem  Beispiel 6   Batchfile um einen bestimmten Bereich aus einem Video herauszuschneiden.
rem               Der Anfang und das Ende werden weggeschnitten.

set "FF=c:\ffmpeg\ffmpeg" :: Pfad zu ffmpeg
set "INPUT=PanoView.mp4"  :: zu bearbeitendes Video
set "OUTPUT=out.mp4"      :: erzeugtes Video
set "START=2.0"           :: Startzeitpunkt in Sekunden
set "LENGTH=3.0"          :: Länge des erzeugten Videos in Sekunden
set "QU=2"                :: MP4 Qualität, 0 ist beste Qualität, 2 ist normal, 9 ist stärkste Kompression

%FF% -ss %START% -t %LENGTH% -i %INPUT% -q:v %QU% -codec:v mpeg4 %OUTPUT%

pause
```

Einen Ausschnitt herausvergrößern (mit anderen Worten: Ränder wegschneiden) und die Größe (Breite * Höhe) des Ausgangs-Videos festlegen:

```
rem  Beispiel 7   Batchfile für eine Auschnitt-Vergrößerung und Änderung von Breite und Höhe des Videos

set "FF=c:\ffmpeg\ffmpeg" :: Pfad zu ffmpeg
set "INPUT=PanoView.mp4"  :: zu bearbeitendes Video
set "OUTPUT=out.mp4"      :: erzeugtes Video
set "CROP=1224:1224:0:0"  :: Den Ausschnitt im Eingangs-Video festlegen: Breite, Höhe, linker Rand, oberer Rand
set "SIZE=800x800"        :: Breite und Höhe des erzeugten Videos (kann kleiner oder größer als das Eingangs-Video sein)
                          ::  Das Breite/Höhe Verhältnis sollte sinnvoll gewählt werden.
                          ::  Sonst würde das Bild verzerrt, d.h. ein Kreis würde nicht mehr rund aussehen
set "QU=2"                :: MP4 Qualität, 0 ist beste Qualität, 2 ist normal, 9 ist stärkste Kompression

%FF% -i %INPUT% -vf crop=%CROP% -s %SIZE% -q:v %QU% -codec:v mpeg4 %OUTPUT%

pause
```

Im crop Filter kann man auch die Variablen "iw" und "ih" verwenden, "iw" ist die Breite und "ih" die Höhe des Eingangs-Videos.

Wenn der dritte und vierte Parameter (Koordinaten der linken oberen Ecke) nicht angegeben ist, dann wird der Ausschnitt automatisch zentriert.

crop=ih:ih ergibt einen zentrierten quadratischen Ausschnitt, sinnvoll für Fulldome-Projektion

crop=iw/2:ih:0 ergibt die linke Hälfte des Eingangs-Videos

crop=iw/2:ih:iw/2 ergibt die rechte Hälfte des Eingangs-Videos

crop=iw/4:ih/4 starke Ausschnitt-Vergrößerung (Faktor 4) auf die Mitte des Videos

Änderung der Ablauf-Geschwindigkeit (Zeitlupe, Zeitraffer)

```
rem  Beispiel 8   Batchfile für die Änderung der Ablauf-Geschwindigkeit (Zeitlupe oder Zeitraffer)

set "FF=c:\ffmpeg\ffmpeg" :: Pfad zu ffmpeg
set "INPUT=PanoView.mp4"  :: zu bearbeitendes Video
set "OUTPUT=out.mp4"      :: erzeugtes Video
set "RATE=30"             :: Erzeugte Framerate
set "SPEED=3.0"           :: Geschwindigkeitsfaktor, kleiner 1 = Zeitraffer, 1 = Echtzeit, grösser 1 = Zeitlupe
set "QU=2"                :: MP4 Qualität, 0 ist beste Qualität, 2 ist normal, 9 ist stärkste Kompression

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

pause
```

Die Einstellungen "RATE" und "SPEED" sind hier völlig unabhängig voneinander. FFmpeg überspringt bei Bedarf automatisch Bilder oder dupliziert Bilder.

Beispiel:

Wenn sowohl das Eingangs- wie auch das Ausgangs-Video 30 Bilder pro Sekunde haben, und wenn SPEED = 3 ist, dann wird automatisch jedes Bild zweimal dupliziert, so dass im Ausgangs-Video jedes Bild dreimal vorkommt.

Wenn SPEED = 0.5 eingestellt wird, dann würde jedes zweite Bild übersprungen werden.

In diesem Beispiel wurde der Video-Filter mit Namen "setpts" verwendet. Näheres dazu findet man in der FFmpeg Dokumentation in Kapitel "Video Filter" unter "setpts".

Der Zeitlupen- oder Zeitraffer Effekt wirkt nur auf das Video und nicht auf den Ton.

-an bewirkt dass der Audio-Kanal entfernt wird

Zeitlupen- oder Zeitraffer-Effekt nur für einen bestimmten Teil eines Videos:

```
rem Zeitlupen-Effekt nur für einen bestimmten Teil eines Videos

set "FF=c:\ffmpeg\ffmpeg" :: Pfad zu FFmpeg
set "IN=7Z7A2089.mov"      :: Quell-Video
set "T1=5"                 :: Startzeitpunkt T1
set "T2=8.5"               :: Zeitpunkt T2 wenn die Zeitlupe beginnen soll
set "T3=9.7"               :: Zeitpunkt T3 wenn die Zeitlupe enden soll
set "T4=11"                :: Endzeitpunkt T4
set "SPEED=5"              :: Zeitlupen-Faktor, kleiner 1 = Zeitraffer, grösser 1 = Zeitlupe
set "FR=30"                :: Erzeugte Framerate
set "OUT=out.mp4"          :: Erzeugtes 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
```

Wie man einen Text einfügt (der während der gesamten Dauer des Videos sichtbar ist):

```
set "FF=c://ffmpeg/ffmpeg"   :: Pfad zu FFmpeg
set "IN=input.mov"           :: Eingangs-Video
set "OUT=output.mp4"         :: erzeugtes Video
set "FONT=arial.ttf"         :: Zeichensatz
set "TEXT=Hello_World"       :: Eingeblendeter Text (darf keine Leerzeichen enthalten, sonst mit Textfile arbeiten,
                             ::  siehe nächstes Beispiel)
set "COLOR=yellow"           :: Textfarbe
set "SIZE=20"                :: Schriftgröße
set "POS_X=(w-tw)/2"         :: X-Textposition, für mittige Anordnung: (w-tw)/2
set "POS_Y=(h-th)/2"         :: Y-Textposition, für mittige Anordnung: (h-th)/2

%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%
pause
```

Wie man Texte langsam ein- und ausblendet:

```
rem  Beispiel 9   Batchfile für Text-Einblendung

set "FF=c:\ffmpeg\ffmpeg"   :: Pfad zu ffmpeg
set "INPUT=PanoView.mp4"    :: zu bearbeitendes Video
set "OUTPUT=out.mp4"        :: erzeugtes Video
set "QU=2"                  :: MP4 Qualität, 0 ist beste Qualität, 2 ist normal, 9 ist stärkste Kompression

set "NAME=TEXT1"            :: Eindeutiger Name für diese Texteinblendung
set "FONT=arial.ttf"        :: Zeichensatz
set "TEXT=MeinText.txt"     :: Dateiname (Muss UTF-8 codiert sein falls Umlaute enthalten sind. Sonst genügt ASCII.)
set "COLOR=yellow"          :: Textfarbe
set "SIZE=250"              :: Schriftgröße
set "POS_X=(w-tw)/2"        :: X-Textposition, für mittige Anordnung: (w-tw)/2
set "POS_Y=(h-th)/2"        :: Y-Textposition, für mittige Anordnung: (h-th)/2
set "DS=0.5"                :: Startzeitpunkt
set "DE=4.5"                :: Endzeitpunkt
set "FID=1.0"               :: Fade-in Dauer (darf klein sein, aber nicht null)
set "FOD=1.0"               :: Fade-out Dauer (darf klein sein, aber nicht null)

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"            :: Eindeutiger Name für diese Texteinblendung
set "FONT=arial.ttf"        :: Zeichensatz
set "TEXT=MeinText.txt"     :: Dateiname (Muss UTF-8 codiert sein falls Umlaute enthalten sind. Sonst genügt ASCII.)
set "COLOR=red"             :: Textfarbe
set "SIZE=250"              :: Schriftgröße
set "POS_X=(w-tw)/2"        :: X-Textposition, für mittige Anordnung: (w-tw)/2
set "POS_Y=(h-th)/3"        :: Y-Textposition, für mittige Anordnung: (h-th)/2
set "DS=0.0"                :: Startzeitpunkt
set "DE=3.0"                :: Endzeitpunkt
set "FID=0.5"               :: Fade-in Dauer (darf klein sein, aber nicht null)
set "FOD=0.5"               :: Fade-out Dauer (darf klein sein, aber nicht null)

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
```

Texte müssen in einer *.txt Datei abgespeichert werden. Falls im Text Umlaute enthalten sind, muss zwingend die UTF-8 Codierung verwendet werden.

Falls im Video am Anfang des Textes ein nicht druckbares Zeichen angezeigt wird (eine leere rechteckige Umrandung), dann muss man die *.txt Datei in einem Hex-Editor öffnen und die ersten drei Zeichen entfernen (EF_{hex} BB_{hex} BF_{hex}).

So kann man eine mitlaufende Uhr in ein Video einblenden:

```
rem  Zeit-Einblendung

set "FONTFILE=courbd.ttf" :: Zeichensatz
set "COLOR=white"         :: Schriftfarbe
set "BOXCOLOR=black"      :: Hintergrundfarbe
set "SIZE=30"             :: Schriftgröße
set "POSITION_X=0"        :: X-Textposition
set "POSITION_Y=(h-th)"   :: Y-Textposition
set "S=72060"             :: Aufnahme-Uhrzeit des ersten Bildes in Sekunden
set "I=8"                 :: Intervall in Sekunden (von einem Bild zum nächsten)

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

Man kann mit FFmpeg auch Audio-Dateien bearbeiten:

```
rem  Erzeuge aus drei Sound-Dateien eine einzige Sound-Datei, mit Überblendungen

set "FF=c:\ffmpeg\ffmpeg"    :: Pfad zu FFmpeg

set "FILE1=Sound_Tag.wav"    :: Dateiname der ersten Quelle
set "V1=1.0"                 :: Lautstärke
set "S1=0"                   :: Startzeitpunkt
set "L1=14"                  :: Länge die verwendet werden soll

set "FILE2=Sound_Nacht.wav"  :: Dateiname der zweiten Quelle
set "V2=0.2"                 :: Lautstärke
set "S2=20"                  :: Startzeitpunkt
set "L2=55"                  :: Länge die verwendet werden soll

set "FILE3=Sound_Tag.wav"    :: Dateiname der dritten Quelle
set "V3=1.0"                 :: Lautstärke
set "S3=20"                  :: Startzeitpunkt
set "L3=30"                  :: Länge die verwendet werden soll

set "DUR=5"                  :: Dauer der Überblendung
set "OUT=sound.mp3"          :: Erzeugte Audio-Datei

%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%

pause
```

In diesem Beispiel werden drei Audio-Dateien mit Kreuz-Überblendung hintereinander gehängt. Für jede der drei Eingangs-Dateien kann die Lautstärke, der Startzeitpunkt und die Länge festgelegt werden.

Es werden zunächst drei temporäre Hilfs-Dateien erzeugt, dann werden die ersten beiden kombiniert, und im letzten Schritt wird der dritte Teil angehängt.

Durch die mehrfache Bearbeitung entsteht kein Qualitätsverlust, weil *.wav ein unkomprimiertes Audio-Format ist.

Ändern von Audio-Abtastrate und der Anzahl von Kanälen:

```
rem  Ändern von Audio-Abtastrate und der Anzahl von Kanälen

set "FF=c:\ffmpeg\ffmpeg"    :: Pfad zu FFmpeg
set "IN=PanoView.mp4"        :: Eingangs-Video
set "START=5"                :: Startzeitpunkt
set "LEN=5"                  :: Länge
set "OUT=out.mp4"            :: Erzeugtes Video

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

pause
```

- **-ac 2 setzt die Anzahl der Audio-Kanäle auf 2. Falls man innerhalb einer Filterkette ein Mono-Signal zu Stereo erweitern möchte, kann man aeval=val(0)|val(0) verwenden.**
- **-af aresample=44100 ändert die Audio-Abtastrate auf 44100 Hz**

Wie wählt man bei einem Video den richtigen Lautstärke-Pegel?

- **Musik ist normalerweise auf den Maximalpegel normalisiert. Das heisst die lauteste Stelle nutzt den zur Verfügung stehenden Dynamikbereich voll aus, ohne zu übersteuern. In diesem Fall kann man die Musik entweder unverändert übernehmen, oder man kann sie leiser machen.**
- **Bei eigenen Tonaufnahmen, z.B. von Naturgeräuschen, ist die Sache nicht so einfach. Bevor man anfängt, den Lautstärkepegel einer eigenen Aufnahme festzulegen, muss man erst einmal den Lautstärkeregler am Verstärker "kalibrieren". Das mache ich so: Zuerst schaue ich mir mehrere Videos aus unterschiedlichen Quellen an (nicht meine eigenen selbstgemachten Videos), und stelle den Lautstärkeregler am Verstärker so ein, dass sich alle Videos gut anhören, d.h. sie sollen sich genau so laut anhören, wie ich sie später im Planetarium hören würde. Um sicherzustellen dass der Frequenzgang halbwegs brauchbar ist, sollten gute 3-Wege Boxen verwendet werden. Der Lautstärkeregler bleibt jetzt in dieser Stellung stehen und wird nicht mehr verändert. Jetzt wird der Lautstärke-Pegel der eigenen Videos so bearbeitet, dass sie sich ebenfalls gut anhören. Damit ist sichergestellt, dass man alle Videos (eigene und fremde) nacheinander zeigen kann, ohne dass zwischendurch Korrekturen bei der Lautstärke notwenig sind.**

Wie man störende tiefe Frequenzen (z.B. Windgeräusche) aus einer Audio-Datei herausfiltert:

```
rem  Audio Hochpass-Filterung und Lautstärke-Anpassung

set "FF=c:\ffmpeg\ffmpeg"    :: Pfad zu FFmpeg
set "IN=sound.wav"           :: Eingangs-Audio-Datei
set "AS=20"                  :: Startzeitpunkt
set "LEN=60"                 :: Länge
set "HP=500"                 :: Grenzfrequenz des Hochpassfilters
set "VOL=10"                 :: Lautstärke-Faktor
set "OUT=out.mp3"            :: Erzeugte Audio-Datei

%FF% -ss %AS% -i %IN% -af highpass=f=%HP%,highpass=f=%HP%,highpass=f=%HP%,volume=%VOL% -t %LEN% -y %OUT%

pause
```

Der Hochpass-Filter schwächt tiefe Frequenzen um 6dB pro Oktave ab. Bei der angegebenen Grenzfrequenz hat der Filter 3dB Abschwächung. In diesem Beispiel wird der gleiche Filter dreimal hintereinander angewendet, das ergibt 18dB pro Oktave.

Mal angenommen eine Audio-Datei soll bei Facebook gezeigt werden. Das geht aber nicht, weil dort nur Bilder oder Videos gezeigt werden können. Lösung: Die Audio-Datei wird mit einem einfarbigen Bild zu einem Video erweitert.

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

pause
```

Wie kann man Ultraschall in den hörbaren Bereich konvertieren, um z.B. Fledermäuse hören zu können?

Es gibt zwei grundverschiedene Methoden, wie man eine hohe Frequenz in eine tiefere Frequenz umsetzen kann:

Entweder man dividiert die Frequenz durch eine Konstante, in diesem Fall würde z.B. der Frequenzbereich von 0 bis 25kHz umgesetzt werden in den Bereich von 0 bis 12.5kHz.

```
rem  Erzeuge 2 Sekunden 1kHz Sinus-Ton, gefolgt von 2 Sekunden Stille:
c:\ffmpeg\ffmpeg -f lavfi -i "sine=frequency=1000:sample_rate=48000:duration=2" -af apad -t 4 sine.wav

rem  Eine Halbierung der Abtastrate bewirkt dass sich die Dauer verdoppelt und die Tonhöhe halbiert:
c:\ffmpeg\ffmpeg -i sine.wav -af asetrate=24000 out1.mp3

rem  Der Tempo=2 Filter bewirkt dass sich die Dauer halbiert und die Tonhöhe unverändert bleibt:
rem  (Der Faktor muss im Bereich 0.5 - 2.0 liegen, bei Bedarf mann man den Filter mehrfach hintereinander anwenden)
c:\ffmpeg\ffmpeg -i sine.wav -af atempo=2.0 out2.mp3

rem  Eine Kombination dieser beiden Effekte bewirkt, dass die Dauer unverändert bleibt und die Tonhöhe halbiert wird:
c:\ffmpeg\ffmpeg -i sine.wav -af asetrate=24000,atempo=2.0 out3.mp3

pause
```

Die zweite Methode besteht darin, eine konstante Frequenz zu subtrahieren. In diesem Fall würde z.B. der Frequenzbereich von 15kHz bis 25kHz umgesetzt werden in den Bereich von 0 bis 10kHz. Vorteil: Der Frequenzbereich von 0 bis 15kHz wird völlig unterdrückt.

```
rem  Ultraschall-Konverter durch Mischung (Subtraktion der Mischfrequenz)

set "IN=Fledermaus_44100.wav"     :: Ultraschall Datei
set "SR=44100"                    :: Sample Rate der Ultraschall Datei
set "MF=15000"                    :: Mischfrequenz (das ist die zu subtrahierende Frequenz)
set "BB=10000"                    :: Bandbreite
                                  :: Der Frequenzbereich von MF bis MF+BB wird umgesetzt in den Bereich von 0Hz bis BB
set "VOL=3"                       :: Lautstärke-Faktor
set "OUT=out.wav"                 :: erzeugte Audio-Datei

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
```

In diesem Beispiel wird der Ultraschall-Ton von einem Video durch Subtraktion in den hörbaren Bereich heruntergesetzt, und das Video wird unverändert übernommen:

```
rem  Ultraschall-Konverter durch Mischung (Subtraktion der Mischfrequenz)

set "IN=7Z7A1699.MOV"             :: Eingangs-Video
set "SR=48000"                    :: Sample Rate der Ultraschall Datei
set "MF=12000"                    :: Mischfrequenz (das ist die zu subtrahierende Frequenz)
set "BB=10000"                    :: Bandbreite
                                  :: Der Frequenzbereich von MF bis MF+BB wird umgesetzt in den Bereich 0Hz bis BB
set "VOL=40"                      :: Lautstärke
set "OUT=699.mp4"                 :: erzeugtes 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%" -y %OUT%
pause
```

Wie man Tonaufnahmen mit dem Mikrofon macht, das im Computer eingebaut ist:

Mit diesem Batch-File kann man sich anzeigen lassen, welche Mikrofone zur Verfügung stehen:

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

Mit diesem Batch-File kann man sich die Eigenschaften eines bestimmten Mikrofons anzeigen lassen:

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

Mit diesem Batch-File kann man eine Tonaufnahme mit dem internen Mikrofon machen:

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

Wie man die Ausgabe von FFmpeg an FFplay weiterleitet:

```
c:\ffmpeg\ffmpeg -i in.mp4 (hier irgendwelche Filter einfügen) -f nut - | c:\ffmpeg\ffplay -
```

Live-Ultraschall-Konvertierung, Eingang über das Computer-Mikrofon (oder eine Eingangs-Buchse), Ausgabe über die Computer-Lautsprecher:

Nachteil: Hat ca. 1-2 Sekunden Verzögerungszeit.

```
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-Ultraschall-Konverter durch Mischung (Subtraktion der Mischfrequenz)

set "SR=44100"        :: Sample Rate der Ultraschall Datei
set "MF=10000"        :: Mischfrequenz (das ist die zu subtrahierende Frequenz)
set "BB=10000"        :: Bandbreite
                      :: Der Frequenzbereich von MF bis MF+BB wird umgesetzt in den Bereich von 0Hz bis BB
set "VOL=30"          :: Lautstärke-Faktor

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 -
```

Zweite Variante mit FFT-Filter:

```
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-Ultraschall-Konverter durch Verschiebung im Frequenzbereich

set "SR=44100"             :: Sample Rate
set "F=4000"               :: Subtracted Frequency
set "VOL=30"               :: Volume Factor
set /a "N=4096*%F%/%SR%"   :: N = 4096 * F / SR

c:\ffmpeg\ffmpeg -f dshow -channels 2 -i audio="Mikrofon (Realtek High Definiti" -af volume=%VOL
%,afftfilt='real=if(lt(b+%N%,nb),real(b+%N%,ch),0)':'imag=if(lt(b+%N%,nb),imag(b+%N%,ch),0)' -f nut - | c:\ffmpeg\ffplay
-
```

Wichtig: In der Systemsteuerung bei den Eigenschaften des Mikrofons unter "Mikrofonerweiterungen" einstellen, dass kein Filter verwendet werden soll.

Womit kann man eigene Tonaufnahmen machen?

Ich verwende die folgende technische Ausstattung mit der ich sehr zufrieden bin:

- **TASCAM DR-70D Recorder, hat 4 Eingangskanäle, Abtastrate wahlweise 44100, 48000 oder 96000 Hz, 16 oder 24 Bit**
- **Das Nachfolgemodell DR-701D hat einige Verbesserungen: Die Eingangsverstärker sind etwas rauschärmer, die 4 Pegelregler können elektronisch miteinander gekoppelt werden, und als Abtastrate ist auch 192kHz möglich.**
- **2 Mikrofone RODE NT1 mit Fell-Windschutz, sehr rauscharm und hervorragend geeignet für leise Naturgeräusche**
- **Mikrofon-Kabel in 5m Länge, damit man bei der Aufnahme ein paar Meter vom Mikrofon entfernt stehen kann. Wenn man zu dicht am Mikrofon steht, hat man die eigenen Störgeräusche in der Aufnahme (man hört jede Bewegung, jedes Schlucken, jedes Magenknurren...)**
- **HAMA Joy Powerbank 10400mAh, als zusätzliche Stromversorgung für den Recorder, weil die eingebauten Batterien nur eine sehr kurze Aufnahmezeit ermöglichen, wenn die Phantomspeisung für die Mikrofone aktiviert ist.**

Wo kann man gute Tonaufnahmen von Naturgeräuschen machen?

Das ist schwieriger als man zunächst denkt. Denn man muss einen Ort finden, an dem es keine Störgeräusche gibt:

- **Flugzeuge bewirken, dass ca. 50% der Tonaufnahmen unbrauchbar sind (im Harz, wo es im Umkreis von 100km keinen großen Flugplatz gibt).**
- **Von Straßen- und Bahnverkehr braucht man mehrere Kilometer Abstand. Es ist hilfreich wenn keine direkte Sichtverbindung besteht, d.h. dazwischenliegende Berge vorteilhaft.**
- **Falls man keine Geräusche von fliessendem Wasser in der Aufnahme haben möchte, muss man Täler vermeiden.**
- **Windgeräusche sind trotz Fell-Windschutz schon bei relativ kleinen Windgeschwindigkeiten störend. Sie können eventuell nachträglich mit einem Hochpassfilter abgeschwächt werden. An Tagen mit starkem Wind braucht man gar nicht erst anfangen.**

Man kann natürlich auch beliebig viele Optionen und Filter in einem einzigen Batch-File kombinieren, wie in diesem Beispiel:

Mit diesem Batch-File kann man aus einem Video einen zeitlichen Bereich herausschneiden, Breite und Höhe verändern, die Framerate einstellen, die Geschwindigkeit verändern (Zeitlupe oder Zeitraffer), bei Bedarf auf quadratisches Format zuschneiden, bei Bedarf den Original-Ton entfernen, sowie Kontrast, Helligkeit, Farbsättigung und Gamma verändern.

```
REM  Beispiel 10   Batchfile für Video-Bearbeitung

set "FF=c:\ffmpeg\ffmpeg" :: Pfad zu ffmpeg
set "INPUT=PanoView.mp4"  :: zu bearbeitendes Video
set "OUTPUT=out.mp4"      :: erzeugtes Video
set "SIZE=800x800"        :: Breite und Höhe des erzeugten Videos, das Verhältnis sollte mit dem
                          ::  Eingangs-Video übereinstimmen, oder quadratisch falls QUAD=yes gewählt wurde
set "RATE=30"             :: Erzeugte Framerate
set "START=1.0"           :: Startzeitpunkt in Sekunden (im Eingangs-Video)
set "LENGTH=3"            :: Länge in Sekunden (im Eingangs-Video)
set "SPEED=3.0"           :: Geschwindigkeitsfaktor
                          ::  < 1 Zeitraffer, 1 Echtzeit, > 1 Zeitlupe
set "QUAD=no"             :: no:  Bilder werden nicht zugeschnitten
                          :: yes: Bilder werden auf quadratisches Format zugeschnitten
set "AUDIO=no"            :: no:  kein Ton
                          :: yes: Original-Ton wird übernommen (mit unveränderter Geschwindigkeit)
set "CONTRAST=1.0"        :: Kontrast im Bereich -2.0 bis 2.0, normal 1.0
set "BRIGHT=0.0"          :: Helligkeit im Bereich -1.0 bis 1.0, normal 0.0
set "SATUR=1.0"           :: Farbsättigung im Bereich 0.0 bis 3.0, normal 1.0
set "GAMMA=1.0"           :: Gamma im Bereich 0.1 bis 10.0, normal 1.0
set "QU=2"                :: MP4 Qualität, 0 ist beste Qualität, 2 ist normal, 9 ist stärkste Kompression

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
```

Wie man viele einzelne Szenen zu einem Video hintereinander fügt:

Das entsprechende Batch-File ist sehr einfach:

```
rem   Final cut with concat demuxer

c:\ffmpeg\ffmpeg -f concat -i concat_list.txt -c copy -y MeinVideo.mp4

pause
```

Man schreibt einfach alle vorhandenen Szenen in eine Text-Datei (hier: concat_list.txt), die z.B. so aussehen kann:

```
file text1.mp4              :: 10  Titel: Ein Jahr im Wald
file text2.mp4              :: 10  Wann und wo entstanden
file Videos/scene20.mp4     :: 12  Wald Live-Video
file text22.mp4             :: 10  Im Laufe von 15 Monaten...
file Videos/scene22.mp4     :: 52  Live Video, Kamera
file text98.mp4             :: 10  Ende
```

Rechts von den doppelten Doppelpunkten stehen optionale Kommentare (z.B. die Länge der Szenen und eine kurze Beschreibung).

Diese Methode erfordert aber zwingend, dass alle Szenen

- **die gleiche Breite und Höhe haben**
- **den gleichen Video-Codec verwenden**
- **die gleiche Framerate haben**
- **den gleichen Audio-Codec verwenden**
- **die gleiche Anzahl von Audio-Kanälen haben (Vorsicht bei Kameras die nur eine Mono-Tonspur haben!)**
- **die gleiche Audio-Abtastrate haben**

Falls eine dieser Bedingungen nicht erfüllt ist, dann gibt es eine Fehlermeldung. Man kann sich dann die Eigenschaften der Dateien mit FFprobe anschauen, um herauszufinden worin sich die Dateien unterscheiden.

Wie kann man eine Video-Datei mit FFprobe untersuchen, ohne jedesmal den Namen der Video-Datei in ein Batch-File schreiben zu müssen?

Man erzeugt einmalig dieses Batch-File und legt es auf dem Desktop ab:

```
c:\ffmpeg\ffprobe %1
pause
```

Nun kann man das zu untersuchende Video einfach mit der Maus auf das Icon dieses Batch-Files ziehen, und dann sieht man sofort das Ergebnis, ohne auch nur eine einzige Taste gedrückt zu haben. Der Parameter %1 bewirkt, dass der Dateiname an FFprobe übergeben wird.

So kann man zwei Videos nebeneinander (oder übereinander) zusammenfügen:

```
set "FF=c://ffmpeg/ffmpeg"   :: Pfad zu FFmpeg
set "IN1=links.mp4"
set "IN2=rechts.mp4"
set "OUT=out.mp4"
rem  nebeneinander mit "hstack", oder übereinander mit "vstack"

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

So kann man vier Videos zu einem 2x2 Mosaik zusammenfügen:

```
set "FF=c://ffmpeg/ffmpeg"   :: Pfad zu FFmpeg
set "IN1=linksoben.mp4"
set "IN2=rechtsoben.mp4"
set "IN3=linksunten.mp4"
set "IN4=rechtsunten.mp4"
set "OUT=mosaik.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
```

Mit welchen Kameras kann man Fulldome-Bilder oder Videos machen?

	Canon 6D	Panasonic LUMIX GH5S	PanoView XDV360	Kodak SP360_4K
				
Auflösung für Fulldome:	**180°: 3648 x 3648 (Bilder)** **180°: 1080 x 1080 (Video)**	**180°: 2880 x 2880 (Bilder)** **180°: 2496 x 2496 (Video)**	**220°: 2448 x 2448** **180°: 2104 x 2104**	**235°: 2880 x 2880** **180°: 2456 x 2456**
Tonaufzeichnung bei Video	**stereo 48000 Hz (aber beide Kanäle identisch, wenn kein externes Mikrofon angeschlossen wird)**	**stereo 48000 Hz (aber beide Kanäle identisch, wenn kein externes Mikrofon angeschlossen wird)**	**mono 8000 Hz**	**stereo 48000 Hz (aber beide Kanäle identisch, weil nur ein Mikrofon eingebaut ist, kein externes Mikrofon anschließbar)**
Geeignet für Fulldome Video?	**geht, wenn ein Fisheye verwendet wird das einen Bildkreis mit maximal 20.2mm Durchmesser ausleuchtet.**	**geht, wenn ein Fisheye verwendet wird das einen Bildkreis mit maximal 13.0mm Durchmesser ausleuchtet.**	**ja**	**ja**
Geeignet für Fulldome Nachtaufnahmen?	**ja, sehr gut**	**ja, sehr gut**	**nein**	**nein**
Geeignet für Fulldome Zeitraffer?	**ja, beliebige Intervall-Zeiten mit externem Timer**	**ja, beliebige Intervall-Zeiten mit externem Timer**	**ja, mit internem Timer**	**ja, mit internem Timer**

Fulldome-Video mit Vollformat-, APS-C- oder MFT-Kameras?

Problem: Ein Vollformat-Chip hat die Größe 36mm x 24mm und somit das Format 3:2. Bei Video-Aufzeichnung wird aber das Format 16:9 verwendet, so dass nur ein Teilbereich mit den Abmessungen 36mm x 20.25mm ausgelesen wird. Da ein Vollformat-Fisheye aber einen Bildkreis mit 24mm Durchmesser ausleuchtet, fehlt im Video oben und unten jeweils ein Streifen.

Wenn der gesamte Bildkreis des Fisheye-Objektivs im Video aufgezeichnet werden soll, dann darf der Bildkreis-Durchmesser des Objektivs nicht größer sein als die Höhe des Teilbereich des Chips, der bei der eingestellten Video-Auflösung ausgelesen wird.

Kamera	Chipgröße	Pixel	Video-Auflösung	Teilbereich des Chips, der ausgelesen wird, Breite x Höhe
Canon 6D	35.8mm x 23.9mm	5472 x 3648	640 x 480 (4:3)	31.87mm x 23.9mm
Canon 6D	35.8mm x 23.9mm	5472 x 3648	1920 x 1080 Full HD (16:9)	35.9mm x 20.19mm
Canon 5D MK4	36mm x 24mm	6720 x 4480	1920 x 1080 Full HD (16:9)	36mm x 20.25mm
Canon 5D MK4	36mm x 24mm	6720 x 4480	4096 x 2160 C4K (17:9)	21.94mm x 11.57mm (Es wird nicht die gesamte Chip-Breite verwendet)
Canon 7D	22.3mm x 14.9mm	5184 x 3456	1920 x 1080 Full HD (16:9)	22.30mm x 12.54mm
Canon EOS R	36mm x 24mm	6720 x 4480	1920 x 1080 Full HD (16:9)	36mm x 20.25mm
Canon EOS R	36mm x 24mm	6720 x 4480	3846 x 2160 4K (16:9)	20.57mm x 11.57mm (Es wird nicht die gesamte Chip-Breite verwendet)
Sony A7S II	35.6mm x 23.8mm	4240 x 2832	1920 x 1080 Full HD (16:9)	35.6mm x 20.0mm
Sony A7S II	35.6mm x 23.8mm	4240 x 2832	3840 x 2160 4K (16:9)	35.6mm x 20.0mm (Es wird die gesamte Chip-Breite verwendet)
Panasonic LUMIX DC-GH5S	19.2mm x 13.0mm	4096 x 2760	1920 x 1080 Full HD (16:9)	18.8mm x 10.6mm (muss noch geprüft werden)
Panasonic LUMIX DC-GH5S	19.2mm x 13.0mm	4096 x 2760	3846 x 2160 4K (16:9)	18.8mm x 10.6mm (muss noch geprüft werden)
Panasonic LUMIX DC-GH5S	19.2mm x 13.0mm	4096 x 2760	4096 x 2160 C4K (17:9)	19.2mm x 10.12mm (Es wird die gesamte Chip-Breite verwendet)
Panasonic LUMIX DC-GH5S	19.2mm x 13.0mm	4096 x 2760	3328 x 2496 Anamorphic (4:3)	17.3mm x 13.0mm (Es wird die gesamte Chip-Höhe verwendet)
Nikon D800	35.9mm x 24.0mm	7360 x 4912	1920 x 1080 Full HD	32.0mm x 18.0mm (Es wird nicht die gesamte Chip-Breite verwendet)

Übersicht über verfügbare Fisheye-Objektive

Objektiv	Anschluss	Blende	Bildwinkel und Bildkreis-Durchmesser	Bemerkungen
Canon EF 8-15mm bei 8mm	Canon EF	f/4.0	180° 22.9mm (selber gemessen)	Sehr gute Abbildungsqualität
Sigma EX DG 8mm	Canon EF u.a.	f/3.5	180° 22.7mm (selber gemessen)	Mittelmäßige Abbildungsqualität
Nippon Kogaku 8mm	M42 / Canon EF	f/2.8	180° 23.0mm (selber gemessen)	M42 Anschluss mit Adapter auf Canon EF
Sigma EX DG 4.5mm	Canon EF u.a.	f/2.8	180° 12.3mm (selber gemessen)	Mittelmäßige Abbildungsqualität
Meike 6-11mm bei 6mm	Canon EF u.a.	f/3.5	180° 15.1mm (selber gemessen)	Gute Abbildungsqualität
Meike 6-11mm bei 7.5mm	Canon EF u.a.	f/3.5	180° 18.4mm (selber gemessen)	Gute Abbildungsqualität
Meike 6-11mm bei 9.5mm	Canon EF u.a.	f/3.5	180° 23.7mm (selber gemessen)	Gute Abbildungsqualität
Meike 6-11mm bei 11mm	Canon EF u.a.	f/3.5	180° 28.7mm (selber gemessen)	Gute Abbildungsqualität
Meike 8mm	Canon EF u.a.	f/3.5	180° ca. 26.9mm 200° ca. 29.9mm	
Opteka 6.5mm	Canon EF	f/3.5	180° ca. 30mm	Schlechte Abbildungsqualität, Brennweite ist ca. 9mm
Entaniya HAL250 6.0mm	Canon EF u.a.	f/5.6	180° 18.2mm 250° 23.7mm	Nur für spiegellose Kameras geeignet, sehr teuer
Entaniya HAL250 4.3mm	Canon EF u.a.	f/4.0	180° 13.1mm 250° 17.0mm	Nur für spiegellose Kameras geeignet, sehr teuer
Entaniya HAL250 3.6mm	Canon EF u.a.	f/2.8	180° 11.0mm 250° 14.25mm	Nur für spiegellose Kameras geeignet, sehr teuer
Entaniya HAL250 3.0mm	Canon EF u.a.	f/2.8	180° 9.2mm 250° 11.9mm	Nur für spiegellose Kameras geeignet, sehr teuer
Entaniya HAL200 6.0mm	Canon EF u.a.	f/4.0	180° 18.2mm 200° 19.9mm	Nur für spiegellose Kameras geeignet, sehr teuer
Entaniya HAL200 5.0mm	Canon EF u.a.	f/5.6	180° 15.2mm 200° 16.6mm	Nur für spiegellose Kameras geeignet, sehr teuer
Samyang 8mm Fisheye II	EF-M, Sony E	f/2.8	180° ca. 29.7mm 188° ca. 31mm	Nur für spiegellose Kameras geeignet, kurzes Auflagemaß
Meike 6.5mm	MFT	f/2.0	180° 15.4mm 190° 15.85mm (selber gemessen)	Nur für spiegellose Kameras geeignet, kurzes Auflagemaß
Olympus M.Zuiko 8mm	MFT	f/1.8	180° ca. 22mm	Nur für spiegellose Kameras geeignet, kurzes Auflagemaß
7artisans (Viltrox) 7.5mm	MFT u.a.	f/2.8	ca. 27mm (APS-C ohne Vignettierung)	Streulicht-Blende muss mechanisch entfernt werden
ZLKC (OCDAY) 7.5mm	MFT u.a.	f/2.8	ca. 27mm (APS-C ohne Vignettierung)	
Laowa 4mm	MFT	f/2.8	180° ca. ?mm 210° ca. ?mm	Nur für spiegellose Kameras geeignet, kurzes Auflagemaß
iZugar MKX22 3.25mm	MFT	f/2.5	180° ca. 8.2mm 220° ca. 10mm	Nur für spiegellose Kameras geeignet, kurzes Auflagemaß
Yumiki 2.5mm	CS-Mount	f/1.6	180° ca. 6.1mm 190° ca. 6.4mm	
SMTSEC 2.27mm	CS-Mount	f/1.4	185° 7.2mm	
Fujinon 1.8mm	C-Mount	f/1.4	180° 5.5mm 185° 5.7mm	
Fujinon 2.7mm	C-Mount	f/1.8	180° 8.4mm 185° 8.6mm	

Welche Kamera/Fisheye Kombinationen bieten sich an?

Kamera	Video-Auflösung	Objektiv	Blende	Fulldome-Bildwinkel	Effektive Pixelanzahl
Canon 6D	640 x 480 (4:3)	Canon EF 8-15mm bei 8mm	f/4.0	180°	460 Pixel
	1920 x 1080 Full HD (16:9)	Sigma EX DG 4.5mm	f/2.8	180°	656 Pixel
	1920 x 1080 Full HD (16:9)	Meike 6-11mm bei 8.2mm	f/3.5	180°	1080 Pixel
	1920 x 1080 Full HD (16:9)	Canon EF 8-15mm bei 8mm	f/4.0	159°	952 Pixel
Canon 5D MK4	1920 x 1080 Full HD (16:9)	Sigma EX DG 4.5mm	f/2.8	180°	654 Pixel
	1920 x 1080 Full HD (16:9)	Meike 6-11mm bei 8.2mm	f/3.5	180°	1080 Pixel
	1920 x 1080 Full HD (16:9)	Canon EF 8-15mm bei 8mm	f/4.0	159°	955 Pixel
	4096 x 2160 C4K (17:9)	Sigma EX DG 4.5mm	f/2.8	170°	2160 Pixel
Canon EOS R	3840 x 2160 4K (16:9)	Entaniya HAL250 3.6mm	f/2.8	180°	2054 Pixel
Sony A7S II	1920 x 1080 Full HD (16:9)	Sigma EX DG 4.5mm	f/2.8	180°	663 Pixel
	3840 x 2160 4K (16:9)	Sigma EX DG 4.5mm	f/2.8	180°	1325 Pixel
	1920 x 1080 Full HD (16:9)	Meike 6.5mm	f/2.0	180°	832 Pixel
	3840 x 2160 4K (16:9)	Meike 6.5mm	f/2.0	180°	1663 Pixel
	1920 x 1080 Full HD (16:9)	Meike 6-11mm bei 8.2mm	f/3.5	180°	1080 Pixel
	3840 x 2160 4K (16:9)	Meike 6-11mm bei 8.2mm	f/3.5	180°	2160 Pixel
	3840 x 2160 4K (16:9)	Olympus M.Zuiko 8mm	f/1.8	ca. 164°	2160 Pixel
Sony A7S II mit externem Recorder	3840 x 2160 4K (16:9)	Sigma EX DG 8mm	f/3.5	180°	2060 Pixel
	3840 x 2160 4K (16:9)	Olympus M.Zuiko 8mm	f/1.8	180°	ca. 1996 Pixel
Panasonic LUMIX GH5S	3328 x 2496 Anamorphic (4:3)	Sigma EX DG 4.5mm	f/2.8	180°	2356 Pixel
	1920 x 1080 Full HD (16:9)	Sigma EX DG 4.5mm, Speedbooster 0.71x	f/2.0	180°	888 Pixel
	3840 x 2160 4K (16:9)	Sigma EX DG 4.5mm, Speedbooster 0.71x	f/2.0	180°	1775 Pixel
	1920 x 1080 Full HD (16:9)	Sigma EX DG 4.5mm, Speedbooster 0.64x	f/1.8	180°	800 Pixel
	3840 x 2160 4K (16:9)	Sigma EX DG 4.5mm, Speedbooster 0.64x	f/1.8	180°	1600 Pixel
	3328 x 2496 Anamorphic (4:3)	Nippon Kogaku 8mm, Speedbooster 0.64x	f/1.8	159°	2496 Pixel
	3328 x 2496 Anamorphic (4:3)	Meike 6.5mm	f/2.0	152°	2496 Pixel
	3328 x 2496 Anamorphic (4:3)	Meike 6-11mm bei 7.5mm, Speedbooster 0.71x	f/2.5	180°	2496 Pixel
	3328 x 2496 Anamorphic (4:3)	Meike 6-11mm bei 8.2mm, Speedbooster 0.64x	f/2.2	180°	2496 Pixel
	1920 x 1080 Full HD (16:9)	Meike 6-11mm bei 6.8mm, Speedbooster 0.64x	f/2.2	180°	1080 Pixel
	3840 x 2160 4K (16:9)	Meike 6-11mm bei 6.8mm, Speedbooster 0.64x	f/2.2	180°	2160 Pixel
Kodak SP360_4K	(kein Vollformat, 2880 x 2880)	0.85mm (eingebaut)	f/2.8	180°	2456 Pixel, ungeeignet für Nachtaufnahmen
Nikon D800	1920 x 1080 Full HD (16:9)	Meike 6-11mm bei 7.3mm	f/3.5	180°	1080 Pixel

Bearbeitung von Videos von der chinesischen PanoView XDV360 Fulldome-Kamera:

```
rem  Beispiel 11    Manipulate a video from PanoViev XDV360 camera (size 2448x2448)

set "FF=c:\ffmpeg\ffmpeg"      :: Path to FFmpeg
set "SIZE=1200"                :: Video size (quadratic)
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)
set "CR=2280:2280:88:88"       :: 180° field of view: 2104:2104:176:176
                               :: 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 [-2.0 ... 2.0], 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% -i Circle_3648.png -i %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
```

Das Bild "Circle_3648.png" hat die Größe 3648x3648 und enthält eine kreisförmige Maske. Die Farbe des Kreisinhalts ist als transparent deklariert, und die Umrandung ist schwarz.

"-ac 2" erweitert die Tonspur auf Stereo, und "-af aresample=44100" setzt die Audio-Abtastrate auf 44100 hoch.

Die Kodak SP360_4K Kamera ist besser als die billige chinesische PanoView XDV360, weil sie eine etwas höhere Auflösung hat und weil die Videos nicht so stark komprimiert werden.

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

set "FF=c:\ffmpeg\ffmpeg"    :: Path to FFmpeg
set "SIZE=1200"              :: Video size (quadratic)
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)
set "CROP=2728:2728:74:74"   :: 180° field of view: 2456:2456:210:210
                             :: 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 [-2.0 ... 2.0], 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
```

Erzeugung von gekrümmtem Text für Fulldome-Projektion

```
rem  Create a video with curved text fade-in fade-out, silent audio

set "FF=c:\ffmpeg\ffmpeg"  :: Path to FFmpeg
set "SIZE=1200"            :: Video size (quadratic)
set "QU=2"                 :: MP4 quality level, 0 is best quality, 2 is normal, 9 is strong compression
set "FPS=30"               :: Output Framerate
set "FONT=arial.ttf"       :: font
set "FSIZE=60"             :: font size
set "COLOR=white"          :: text color
set "BACK=black"           :: background color
set "DUR=10"               :: duration of video
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*0.9"          :: Y text position
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 "OUTPUT=text13.mp4"    :: Output filename

%FF% -f lavfi -i color=c=%BACK% -i xmap_3648.pgm -i ymap_3648.pgm -f lavfi -i anullsrc -r %FPS% -t %DUR% -aspect "1:1" ^
-lavfi scale=3648:3648,drawtext='fontfile=%FONT%:textfile=%TEXT%:^
fontcolor_expr=%COLOR%@%%{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%',format=pix_fmts=rgb24,remap ^
-s %SIZE%x%SIZE% -c:v mpeg4 -c:a aac -shortest -q:v %QU% -y %OUTPUT%

pause
```

Ich muss zugeben dass dies hier eine komplizierte Kommandozeile ist. Der eigentliche Kern ist der "remap" Filter, mit dem man beliebige Verzerrungen erzeugen kann. Die Verzerrung wird in den beiden Dateien xmap_3648.pgm und ymap_3648.pgm beschrieben. In diesen Dateien steht für jedes Pixel drin, von welchem Pixel im Eingangs-Video es hergeholt wird. Man muss sich ein (C#) Programm schreiben, das diese Dateien erzeugen kann.

-i color=c=black erzeugt ein schwarzes Bild

-i anullsrc erzeugt eine leere Tonspur

Dieses Batch-File erzeugt ein Zeitraffer aus vielen Bildern, mit Maskierung und Deflicker-Filter, mit langsamer Rotation des Bildes, Ein- und Ausblendung am Anfang und Ende:

```
rem  Beispiel 12

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 (quadratic)
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 im range [-2.0 ... 2.0], 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"               :: Defflicker 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%,^
overlay,zoompan=s=%SIZE%x%SIZE%:d=%D%+%E%*'eq(in,1)'+%E%*'eq(in,%L%)':fps=%F%,^
framerate=%FPS%:interp_start=0:interp_end=255:scene=100,rotate=%ROT%,defflicker=size=%DEF%,^
fade=in:d=%FADE%,fade=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
```

Dies ist ein Beispiel für einen Blink-Komparator. Es wird ein Animated GIF erzeugt, das dauernd zwischen zwei (oder mehr) Bildern hin- und her wechselt.

```
rem  Blinkkomparator, wechselt zwischen zwei oder mehr Bildern

set "FF=c:\ffmpeg\ffmpeg"  :: Pfad zu FFmpeg
set "IN=pluto_%%1d.jpg"    :: Dateiname der Bilder
set "FR=2.0"               :: Framerate
set "OUT=out.gif"          :: Erzeugte animated GIF Datei

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

pause     :: Warte auf Tastendruck
```

Wenn wir stattdessen ein MP4 erzeugen wollen, dann müssen wir zusätzlich noch angeben wie lang es werden soll und mit welcher Framerate es ausgegeben werden soll:

```
rem  Blinkkomparator, wechselt zwischen zwei oder mehr Bildern

set "FF=c:\ffmpeg\ffmpeg"  :: Pfad zu FFmpeg
set "IN=pluto_%%1d.jpg"    :: Dateiname der Bilder
set "FI=2.0"               :: Framerate, mit der die Bilder eingelesen werden
set "T=10"                 :: Länge in Sekunden
set "FO=25"                :: Ausgangs-Framerate
set "OUT=out.mp4"          :: Erzeugte MP4 Datei

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

pause     :: Warte auf Tastendruck
```

Der Parameter "-loop 1" bewirkt, dass die gleichen Bilder immer wieder eingelesen werden. Wenn man das macht, muss man die Länge des Videos irgendwie begrenzen, in diesem Fall mit "-t 10".

Dieses Beispiel zeigt, wie man ein einzelnes Bild in einem Video gegen ein anderes Bild austauschen kann.

Ihr habt vielleicht schon mal davon gehört, dass man zu Werbezwecken ein Produkt-Bild in einen Film einblenden kann, und zwar nur ganz kurz. Wenn die Framerate beispielsweise 25 Bilder pro Sekunde ist, dann wird ein einzelnes Bild 40ms lang gezeigt. Das ist zu kurz um das Produkt deutlich erkennen zu können, aber es genügt um bei den Zuschauern ein Gefühl dafür zu erzeugen, dass sie dieses Produkt haben wollen. Wenn z.B. im Film für 40ms eine Bratwurst oder Popcorn eingeblendet wird, dann steigen nach Ende des Films die Verkaufszahlen für genau diese Produkte. Obwohl dem Zuschauer nicht bewusst ist, warum er jetzt Appetit auf eine Bratwurst oder Popcorn bekommen hat.

```
set "FF=c:\ffmpeg\ffmpeg"   :: Pfad zu FFmpeg
set "IN=scene8.mp4"          :: Eingangs-Video
set "BW=bratwurst.jpg"       :: Bild einer Bratwurst
set "W=1920"                 :: Breite des Eingangs-Videos
set "H=1080"                 :: Höhe des Eingangs-Videos
set "T=3.0"                  :: Zeitpunkt wann das Bild eingeblendet werden soll
set "OUT=out.mp4"            :: Ausgangs-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
```

Mit dem "scale" Filter wird das Bild auf die gleiche Größe wie das Eingangs-Video skaliert. Falls das Bild schon die richtige Größe hat, kann man diesen Filter auch weglassen.

Der "setpts" Filter legt den Zeitpunkt für das Bild fest.

Der "overlay" Filter kombiniert dann die beiden Quellen.

Die Tonspur wird unverändert vom Eingangs-Video übernommen.

Gradations-Kurven und Vignettierung:

Hinweis: Dies würde ich heute nicht mehr so machen. Besser geht es mit Color-look-up-Table, siehe nächstes Kapitel.

-- Ein Bild wird in GIMP geöffnet.
-- Farben::Werte --> Im Bild geeignete Punkte für Schwarz- Weiss- und Grau auswählen.
-- Klick auf "Diese Einstellungen als Kurven bearbeiten"
-- Feinkorrekturen an den Kurven machen
-- Dabei möglichst viele Punkte auf die Kurven setzen, weil sie später durch Geraden interpoliert werden.
-- Klick auf "Einstellungen als Datei exportieren"
-- Häkchen setzen bei "Altes Dateiformat für Kurven verwenden"
-- Dateiname: curves.gimp
-- Abspeichern
-- Dann den Konverter GIMP2ACV (1) aufrufen. Dieser Konverter liest die Datei curves.gimp ein, konvertiert sie und speichert sie als curves.acv ab. Die Datei curves.gimp muss in gleichen Ordner stehen wo der Konverter aufgerufen wird.
-- Im Batch-File für die FFmpeg Bearbeitung wird u.a. der entsprechende Video-Filter ausgerufen: -vf curves=psfile='curves.acv'

Vignettierung am Bildrand kann automatisch ausgeglichen werden: -vf vignette=0.5:mode=backward
mode=backward macht die Bildescken heller, mode=forward würde sie dunkler machen. Die Zahl 0.5 muss so eingestellt werden, dass die Bildecken weder zu hell noch zu dunkel werden.

Die beiden Filter können so zusammengefasst werden:

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

(1) Quelle: http://www.astro-electronic.de/GIMP2ACV.EXE

Farbtransformationen mit Color-look-up-Tables:

Eine Color-look-up-Table (CLUT) ist eine mathematische Vorschrift, nach der jede beliebige Farbe durch eine andere Farbe ersetzt wird.

Methode 1:

Es gibt verschiedene Dateiformate für die CLUT, wir verwenden hier das *.cube Format. Bei diesem Format wird ein Farbraum mit üblicherweise 25 * 25 * 25 Einträgen verwendet, so dass die Tabelle 25^3 = 15625 verschiedene Farben enthält. Farben die zwischen den festgelegten Tabelleneinträgen liegen werden interpoliert. Man kann auch Tabellen mit 64^3 Einträgen erzeugen, aber für die meisten Anwendungen sind 25^3 Einträge völlig ausreichend.

Zunächst müssen wir eine CLUT erzeugen:

Mit dem "IWLTBAP LUT Generator" (1), wird eine sogenannte HALD Datei erzeugt. Das ist ein PNG Testbild, das von jeder Farbe mindestens ein Pixel enthält. Dieses Bild wird nun mit einem beliebigen Bildverarbeitungsprogramm genau so bearbeitet, wie man auch das Video bearbeiten möchte. Erlaubt sind Änderungen bei Helligkeit, Kontrast, Gamma, Farbsättigung, Farbtemperatur, Vertauschung und sonstige Beeinflussung der Farben. Nicht erlaubt sind beispielsweise Rauschreduzierung, Schärfung oder Vignettierung. Es sind also nur solche Transformationen erlaubt, bei denen die RGB Werte jedes Pixels eindeutig in einen anderen RGB Wert transformiert werden können.

Das fertig bearbeitete Bild wird als PNG gespeichert und vom "IWLTBAP LUT Generator" eingelesen, der daraufhin die CLUT im *.cube Format erzeugt.

Diese CLUT kann nun in FFmpeg für die Transformation eines Videos verwendet werden:

```
set "FF=c:\ffmpeg\ffmpeg"   :: Pfad zu FFmpeg
set "IN=scene8.mp4"          :: Eingangs-Video
set "LUT=myLUT.cube"         :: Look-Up-Table
set "OUT=out.mp4"            :: Ausgangs-Video

%FF% -i %IN% -vf lut3d="%LUT%" %OUT%
pause
```

(1) Quelle: https://generator.iwltbap.com/

Methode 2, der Arbeitsablauf wird hier Schritt für Schritt für ein 10-bit Video beschrieben:

Bei dieser Methode braucht man keine *.cube Datei.

Schritt 1: Mit diesem Batch-File wird aus dem 10-bit Video an einer geeigneten Stelle ein einzelnes Bild extrahiert und verlustlos als 16-bit PNG abgespeichert:

```
set "FF=c:\ffmpeg\ffmpeg"    :: Pfad zu FFmpeg
set "IN=Video_62.mov"        :: Eingangs-Video
set "T=35"                   :: Zeitpunkt
%FF% -ss %T% -i %IN% -frames 1 -y Bild.png
pause
```

Schritt 2: Mit diesem Batch-File wird eine Farbtabelle (CLUT = Color-look-up-Table) erzeugt. Das ist ein PNG Bild mit 512x512 Pixeln, das von jeder möglichen Farbe genau ein Pixel enthält. Ich bin mir noch nicht sicher, ob das Bild an dieser Stelle schon 16 Bit Auflösung haben muss. Zumindest schadet es nicht. Falls 8 Bit genügen, würde man "-pix_fmt rgb48be" weglassen.

Der Parameter LEVEL legt fest, wie viele verschiedene Farben in der CLUT enthalten sind. Die Höhe und Breite des quadratischen Bildes ist LEVEL*LEVEL*LEVEL, bei LEVEL=8 gibt es 64*64*64=262144 Farben und das Bild hat 512*512=262144 Pixel. Wichtig ist, dass die Datei in einem unkomprimierten oder verlustlos komprimierten Format gespeichert wird, PNG ist daher gut geeignet.

```
set "FF=c:\ffmpeg\ffmpeg"   :: Pfad zu FFmpeg
set "LEVEL=8"
%FF% -f lavfi -i haldclutsrc=%LEVEL% -frames 1 -pix_fmt rgb48be clut.png
pause
```

Schritt 3: Das extrahierte Bild wird in GIMP geöffnet.

Schritt 4: Die Farbtabelle wird in GIMP geöffnet, mit "Select all" ausgewählt und mit ctrl-c kopiert.

Schritt 5: Das erste Bild wird angeklickt und dann wird mit "Paste in Place" die Farbtabelle links oben eingefügt. Da das erste Bild viel größer als die Farbtabelle ist, stört die Tabelle an dieser Stelle nicht.

Schritt 6: Rechtsklick auf "Floating Selection" und dann wird "To New Layer" ausgewählt.

Schritt 7: Rechtsklick auf den neu entstandenen "Pasted Layer" und dann wird "Merge Down" ausgewählt.

Schritt 8: Jetzt wird das Bild so bearbeitet, wie es später im Video aussehen soll. Und die Farbtabelle in der linken oberen Ecke wird natürlich mit bearbeitet. Farbkorrekturen, Farbtemperatur, Farbsättigung, Gradationskurve, Helligkeit, Kontrast. Das Bild darf durchaus sichtbares Rauschen enthalten. Später im Video fällt das Rauschen nicht so stark auf, weil es sich durch die schnelle Abfolge von Bildern teilweise wegmittelt. Nicht erlaubt sind Operationen, die sich nicht durch eine Color-look-up-Table beschreiben lassen, wie z.B. Rauschreduzierung, Weichzeichner oder Schärfung.

Schritt 9: Das fertig bearbeitete Bild wird auf die Größe 512x512 Pixel beschnitten, so dass nur noch die Farbtabelle in der linken oberen Ecke übrig

bleibt. Image > Canvas Size > Width=512, Height=512, dann auf "Resize" klicken.

Schritt 10: Das Bild unter dem Namen clut2.png als 16-bit PNG exportieren und dabei als Pixelformat "16bpc RGB" auswählen. GIMP kann jetzt geschlossen werden.

Schritt 11: Diese Color-look-up-Table wird jetzt mit FFmpeg auf das ganze Video angewendet. Die Anwendung der Farbtabelle erfolgt mit 10 Bit Genauigkeit. Farben, die in der Tabelle nicht enthalten sind, werden interpoliert. Erst danach wird auf 8 Bit Genauigkeit konvertiert und ein MP4 erzeugt:

```
set "FF=c:\ffmpeg\ffmpeg"   :: Pfad zu FFmpeg
set "IN=Video_62.mov"       :: Eingangs-Video

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

Eine Vereinfachung des Arbeitsablaufs ist mit diesem Batch-File möglich, das die Schritte 1 bis 7 zusammenfasst. Die CLUT wird gleich von FFmpeg mit dem extrahierten Bild kombiniert:

```
Set "FF=c://ffmpeg/ffmpeg"      :: Pfad zu FFmpeg
set "IN=P1000099.mov"           :: Eingangs-Video
set "T=5"                       :: Zeitpunkt

%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
pause
```

Dieses Bild wird dann in GIMP bearbeitet und dann unter dem gleichen Namen als 16-bit PNG exportiert. Auf die Änderung der Bildgröße in Schritt 9 kann verzichtet werden, wenn man in FFmpeg einen crop-Filter auf das Bild anwendet. Warum die kleine Helligkeits-Korrektur vor Anwendung des haldclut-Filters notwendig ist, ist noch unklar.

```
set "FF=c://ffmpeg/ffmpeg"   :: Pfad zu FFmpeg
set "IN=P1000099.mov"        :: Eingangs-Video
set "BR=0.06"                :: Kleine Helligkeits-Feinkorrektur vor Anwendung der 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
```

Beim haldclut-Filter hängt die Größe der Color-look-up-Table vom "Level" Parameter ab:

Level	Größe der CLUT	Größe der CLUT als 16-bit PNG	Kantenlänge des RGB-Würfels	Anzahl der Stützpunkte	Abstand der Stützpunkte (8-bit / 16-bit)
4	64x64 Pixel	16.4kB	16	4096	16 / 4096
5	125x125 Pixel	61.9kB	25	15625	10.2 / 2621
6	216x216 Pixel	179kB	36	46656	7.1 / 1820
7	343x343 Pixel	436kB	49	117649	5.2 / 1337
8	512x512 Pixel	0.97MB	64	262144	4 / 1024

Workflow für Videos vom Sternenhimmel mit Panasonic GH5S und Nippon Kogaku 8mm Fisheye:

Schritt 1, Gradationskurve anwenden und dann ein Bild extrahieren und die CLUT einfügen:

```
set "FF=c:\ffmpeg\ffmpeg"     :: Pfad zu FFmpeg
set "IN=P1000128.mov"         :: Eingangs-Video
set "T=1"                     :: Zeitpunkt des extrahierten Bildes

%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

pause
```

Schritt 2, nach der Bearbeitung mit GIMP wird die CLUT extrahiert und auf das Video angewendet:

```
set "FF=c://ffmpeg/ffmpeg"    :: Pfad zu FFmpeg
set "IN=P1000128.mov"         :: Eingangs-Video
set "BR=0.04"                 :: Leichte Helligkeits-Korrektur vor Anwendung des Haldclut-Filters
set "OUT=128.mp4"             :: erzeugte Datei

%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
```

Dieses Batch-File erzeugt aus einem Video ein Histogramm für die R,G,B Komponenten:

```
set "FF=c:\ffmpeg\ffmpeg"   :: Pfad zu FFmpeg
set "IN=MVI_2562.mov"        :: Eingangs-Video

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

Dieses Batch-File leitet die Ausgabe von FFmpeg an FFplay weiter:

```
c:\ffmpeg\ffmpeg -i in.mp4 -f nut - | c:\ffmpeg\ffplay -
pause
```

Diverse unsortierte Sachen:

ffmpeg -y -f mov -i input.MOV -vcodec prores_ks -pix_fmt yuva444p10le -profile:v 4444 -bits_per_mb 8000 -s 1920x1080 output-file.mov

c:\ffmpeg\ffmpeg -h encoder=png

c:\ffmpeg\ffmpeg -i P1000063.MOV -pix_fmt yuv444p -s 1920x1080 -t 10 out.mov

profile=low-latency

Good parameters for encoding high quality MPEG-4

'-mbd rd -flags +mv4+aic -trellis 2 -cmp 2 -subcmp 2 -g 300 -pass 1/2', things to try: '-bf 2', '-mpv_flags qp_rd', '-mpv_flags mv0', '-mpv_flags skip_rd'.

There's a wiki page covering this:

https://trac.ffmpeg.org/wiki/How%20to%20speed%20up%20/%20slow%20down%20a%20video

(There's a part saying "Using a complex filtergraph, you can speed up

video and audio at the same time:". IMO, you can also do this with

separate "simple" filters, using "-vf" and "-af".)

... setpts filter doesn't change the
audio speed. The audio example in the wiki, using "atempo", changes the
speed while _maintaining_ the pitch. To stretch the audio _including_
pitch change, you need to instead change the sample speed with the
"asetrate" filter, and re-adapt the output sample rate (because your
output file will require a sane sample rate such as 44.1k or 48k) with
"-ar".

E.g. to reduce the speed of the audio to 0.8x of the given input,
assuming an input sample rate of 48000k:

$ ffmpeg -i input -af asetrate=38400 -ar 48000 output

(38400 is 0.8 * 48000. I wish asetrate would understand an "input
sample rate" variable.)

Mit diesem Tool kann man sich sehr viele EXIF-Daten von Bildern oder Videos anzeigen lassen:
https://www.sno.phy.queensu.ca/~phil/exiftool/

Weitere interessante Video-Filter in FFmpeg:

- **amplify Verstärkt Änderungen von einem Bild zum nächsten.**
- **colorbalance Farbkorrektur für die drei Grundfarben, jeweils getrennt für dunkle, mittlere und helle Pixel**
- **curves Einstellung der Gradations-Kurve, gut geeignet um schwache Objekte (Galaxien) hervorzuheben**
- **deblock Entfernt Block-Artefakte aus einem Video**
- **deshake, vidstabdetect, vidstabperform Stabilisierung von verwackelten Videos**
- **dilation Ersetzt jedes Pixel durch das hellste Pixel innerhalb der 3x3 Umgebung, sinnvoll bevor man ein Video auf kleinere Auflösung runterskaliert, ohne dabei die schwächeren Sterne zu verlieren.**
- **fftdnoiz Eine Video-Rauschunterdrückung die im Frequenzbereich arbeitet.**
- **fillborders Beseitigt fehlerhafte Ränder an Videos, indem es die am Rand liegenden Pixel mit einer bestimmten Farbe einfärbt, oder die weiter innen liegenden Pixel dupliziert. Die Größe des Videos bleibt unverändert.**
- **gradfun Entfernt Streifen in Regionen mit Helligkeits-Gradient (sinnvoll bei Fulldome-Videos in der Dämmerung)**
- **haldclut Anwendung einer Color-look-up-Table**
- **hflip, vflip Horizontale und vertikale Spiegelung**
- **hstack, vstack, xstack Legt zwei oder mehr Videos neben- oder übereinander**
- **lenscorrection, lensfun Korrigiert die geometrische Verzeichnung und andere Fehler eines Objektivs**
- **loop Wiederholt einzelne Bilder oder Teile eines Videos mehrfach**
- **lut3d Dreidimensionale Look-Up-Table für Farbkorrekturen**
- **minterpolate Berechnet Zwischenbilder mit Bewegungs-Interpolation**
- **pad Fügt Ränder hinzu**
- **reverse Lässt ein Video rückwärts laufen , braucht aber viel Speicher!**
- **rotate Drehung um beliebige Winkel, auch dynamisch**
- **selectivecolor Selektive Farbkorrektur in bestimmten Farb-Bereichen**
- **tmix Zeitliche gewichtete Mittelung aufeinanderfolgender Bilder, gut geeignet zur Rauschreduzierung**
- **tpad Fügt am Anfang oder Ende eines Videos Bilder hinzu, die wahlweise einfarbig sind oder eine Kopie des ersten oder letzten Bildes.**
- **trim Anfang und Ende eines Videos wegschneiden**
- **vignette Korrektur oder künstliche Erzeugung von Vignettierung (Helligkeitsabfall zum Rand hin)**
- **zoompan Dynamisches Zoomen oder Schwenken innerhalb eines Videos oder Einzelbildes, hat sehr viele Einstellmöglichkeiten!**

Man kann FFmpeg nicht lernen, indem man nur die Anleitung liest.

Man muss es selber ausprobieren, und mit einfachen Beispielen anfangen.

Ich bedanke mich für eure Aufmerksamkeit.