ffmpeg en CLI

Commandes

Volumétrie

Nous compterons les images et leur volumétrie avant d'encoder en h.264

```
export jour="2017-08-15"
echo "$(ls $jour | wc -l) photos" && du -sh $jour | awk '{print $1"o"}' &&
du -s $jour | awk '{print $1" octets"}'
8466 photos
40Go
41220124 octets
```

Timelapse

Cas d'usage: montage d'un timelapse de plusieurs milliers de photos dans un dossier

```
time ffmpeg -framerate 10 -f image2 -pattern_type glob -i $jour/'*.JPG' -r
10 -c:v libx264 -s hd1080 9.mp4
real    74m43,036s
user    145m8,072s
sys 1m4,452s
```

Inspiré de cet article et les options y sont expliquées, sinon man ffmpeg..
Ici le framerate est à 10 et non 24 ou plus, vu le parcours sinueux et la prise d'image à une image/seconde, trop rapide sinon.
Attention, pour Windows, le globbing n'est pas supporté, il faudra mettre les images dans le même dossier et procéder séquentiellement;

```
ffmpeg -framerate 10 -f image2 -start_number 42 -i G00%d.JPG -c:v libx264 -s
hd1080 9.mp4
```

Concaténation

Concaténation de plusieurs vidéos de même encodage:

```
# via un fichier
for f in ./*.mp4; do echo "file '$f'" >> liste.txt; done
ffmpeg -f concat -safe 0 -i liste.txt -c copy final.mp4
# directement
```

```
ffmpeg -safe 0 -f concat -i <(find . -type f -name '*.mp4' -printf "file
'$PWD/%p'\n" | sort) -c copy final.mp4
```

Voir doc et des commentaires

Crop

Truncate

Scripts

cut

```
#!/bin/bash
# This program is free software: you can redistribute it and/or modify
# it under the terms of the GNU General Public License as published by
# the Free Software Foundation, either version 3 of the License, or
# (at your option) any later version.

# This program is distributed in the hope that it will be useful,
# but WITHOUT ANY WARRANTY; without even the implied warranty of
# MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
# GNU General Public License for more details.

# You should have received a copy of the GNU General Public License
# along with this program.  If not, see <http://www.gnu.org/licenses/>.

# Auteur : Xanatos
# License : GPL-V3 - voir
https://www.gnu.org/licenses/quick-guide-gplv3.html
# Nom: duration.sh
# Version : 0.1
# Date création : 10/06/17
# Date MAJ : 10/06/17
# Description : Ce script parcrous le dossier donné en paramètre
#   utilise ffmpeg pour récupérer la durée d'un fichier vidéo
#   utilise du pour connaitre sa taille
#   en sortie nous aurons le cumul des temps, taille et quantité

# Dépendances : ffmpeg, du, find, bc, sed, grep, cut, awk
# Entrée(s) :
# Sortie(s) : console
# Usage : ./duration.sh un_dossier/

chemin="$1"
debut="$2"
```

```
fin="$3"
suffixe="$4"

fichier="$(echo "$chemin" | cut -f1 -d".")"
extension="$(echo "$chemin" | cut -f2 -d".")"
fichier_cut="$fichier$suffixe.$extension"

ffmpeg -i "$chemin" -vcodec copy -acodec copy -ss "$debut" -to "$fin"
"$fichier_cut"

exit 0
```

Exemples en sortie:

```
~$ /opt/cut.sh ~/20170818_110425.MOV 00:01:40 00:03:45 _1

ffmpeg version 3.2.5-1 Copyright (c) 2000-2017 the FFmpeg developers
  built with gcc 6.3.0 (Debian 6.3.0-18) 20170516
  configuration: --prefix=/usr --extra-version=1 --toolchain=hardened --
libdir=/usr/lib/x86_64-linux-gnu --incdir=/usr/include/x86_64-linux-gnu --
enable-gpl --disable-stripping --enable-avresample --enable-avisynth --
enable-gnutls --enable-ladspa --enable-libass --enable-libbluray --enable-
libbs2b --enable-libcaca --enable-libcdio --enable-libebur128 --enable-
libflite --enable-libfontconfig --enable-libfreetype --enable-libfribidi --
enable-libgme --enable-libgsm --enable-libmp3lame --enable-libopenjpeg --
enable-libopenmpt --enable-libopus --enable-libpulse --enable-librubberband
--enable-libshine --enable-libsnappy --enable-libsoxr --enable-libspeex --
enable-libssh --enable-libtheora --enable-libtwolame --enable-libvorbis --
enable-libvpx --enable-libwavpack --enable-libwebp --enable-libx265 --
enable-libxvid --enable-libzmq --enable-libzvbi --enable-omx --enable-openal
--enable-opengl --enable-sdl2 --enable-libdc1394 --enable-libiec61883 --
enable-chromaprint --enable-frei0r --enable-libopencv --enable-libx264 --
enable-shared
  libavutil      55. 34.101 / 55. 34.101
  libavcodec     57. 64.101 / 57. 64.101
  libavformat    57. 56.101 / 57. 56.101
  libavdevice    57.  1.100 / 57.  1.100
  libavfilter     6. 65.100 /  6. 65.100
  libavresample   3.  1.  0 /  3.  1.  0
  libswscale      4.  2.100 /  4.  2.100
  libswresample   2.  3.100 /  2.  3.100
  libpostproc    54.  1.100 / 54.  1.100
Guessed Channel Layout for Input Stream #0.1 : stereo
Input #0, mov,mp4,m4a,3gp,3g2,mj2, from '/home/USEER/20170818_110425.MOV':
  Metadata:
    major_brand     : qt
    minor_version   : 0
    compatible_brands: qt
    creation_time   : 2017-08-18T11:04:25.000000Z
  Duration: 00:04:19.00, start: 0.000000, bitrate: 31456 kb/s
    Stream #0:0(eng): Video: h264 (High) (avc1 / 0x31637661), yuv420p,
```

```
1920x1080, 30039 kb/s, 60 fps, 60 tbr, 60k tbn, 120k tbc (default)
    Metadata:
      creation_time   : 2017-08-18T11:04:25.000000Z
      handler_name    : iCatch Alias Data Handler
      encoder         : iCatch AVCC
    Stream #0:1(eng): Audio: pcm_s16le (sowt / 0x74776F73), 44100 Hz,
stereo, s16, 1411 kb/s (default)
    Metadata:
      creation_time   : 2017-08-18T11:04:25.000000Z
      handler_name    : iCat Alias Data Handler
Output #0, mov, to '/home/USER/20170818_110425_1.MOV':
  Metadata:
    major_brand     : qt
    minor_version   : 0
    compatible_brands: qt
    encoder         : Lavf57.56.101
    Stream #0:0(eng): Video: h264 (High) (avc1 / 0x31637661), yuv420p,
1920x1080, q=2-31, 30039 kb/s, 60 fps, 60 tbr, 60k tbn, 60k tbc (default)
    Metadata:
      creation_time   : 2017-08-18T11:04:25.000000Z
      handler_name    : iCatch Alias Data Handler
      encoder         : iCatch AVCC
    Stream #0:1(eng): Audio: pcm_s16le (sowt / 0x74776F73), 44100 Hz,
stereo, 1411 kb/s (default)
    Metadata:
      creation_time   : 2017-08-18T11:04:25.000000Z
      handler_name    : iCat Alias Data Handler
Stream mapping:
  Stream #0:0 -> #0:0 (copy)
  Stream #0:1 -> #0:1 (copy)
Press [q] to stop, [?] for help
frame=    0 fps=0.0 q=-1.0 size=       0kB time=00:00:00.00 bitrate=N/A
speed=  frame=    0 fps=0.0 q=-1.0 size=       0kB time=00:00:00.00
bitrate=N/A speed=  frame=  700 fps=466 q=-1.0 size=   45104kB
time=00:00:11.85 bitrate=31181.0kbitsframe= 2212 fps=1105 q=-1.0 size=
140531kB time=00:00:37.05 bitrate=31072.4kbitframe= 3658 fps=1462 q=-1.0
size=  232576kB time=00:01:01.15 bitrate=31157.2kbitframe= 4384 fps=1461
q=-1.0 size=  279122kB time=00:01:13.25 bitrate=31215.9kbitframe= 4976
fps=1420 q=-1.0 size=  317509kB time=00:01:23.11 bitrate=31293.8kbitframe=
5762 fps=1414 q=-1.0 size=  367583kB time=00:01:36.21
bitrate=31296.4kbitframe= 6538 fps=1428 q=-1.0 size=  417772kB
time=00:01:49.15 bitrate=31354.9kbitframe= 6708 fps=1321 q=-1.0 size=
428465kB time=00:01:51.99 bitrate=31341.1kbitframe= 7488 fps=1417 q=-1.0
Lsize=  478603kB time=00:02:04.99 bitrate=31367.5kbits/s speed=23.7x
video:456891kB audio:21533kB subtitle:0kB other streams:0kB global
headers:0kB muxing overhead: 0.037355%
```

duration

```
#!/bin/bash
# This program is free software: you can redistribute it and/or modify
# it under the terms of the GNU General Public License as published by
# the Free Software Foundation, either version 3 of the License, or
# (at your option) any later version.

# This program is distributed in the hope that it will be useful,
# but WITHOUT ANY WARRANTY; without even the implied warranty of
# MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
# GNU General Public License for more details.

# You should have received a copy of the GNU General Public License
# along with this program.  If not, see <http://www.gnu.org/licenses/>.

# Auteur : Xanatos
# License : GPL-V3 - voir
https://www.gnu.org/licenses/quick-guide-gplv3.html
# Nom: duration.sh
# Version : 0.1
# Date création : 10/06/17
# Date MAJ : 10/06/17
# Description : Ce script parcrous le dossier donné en paramètre
#   utilise ffmpeg pour récupérer la durée d'un fichier vidéo
#   utilise du pour connaitre sa taille
#   en sortie nous aurons le cumul des temps, taille et quantité

# Dépendances : ffmpeg, du, find, bc, sed, grep, cut, awk
# Entrée(s) :
# Sortie(s) : console
# Usage : ./duration.sh un_dossier/

# Temps total
T=0
# Nombre de fichiers
NB=0
# Taille totale
SI=0

# Cherchons tout les fichiers dont le mimetype est vidéo
find "$1" -type f -exec file -i {} + | sed -n '/video/s/:[^:]\+$//p' | \
{
# et redirection du résultat dans une boucle
while read -r chemin
do
    # ffmpeg et isolation de l'attribut Duration et convertision en secondes
    d="$(ffmpeg -i "$chemin" 2>&1 | grep Duration | cut -d ' ' -f 4 | sed
```

```
s/,// | \
        awk '{ split($1, A, ":"); print 3600*A[1] + 60*A[2] + A[3] }')"
    # bc pour additioner des float
        c="$(bc <<< "$T"+"$d")"
        T="$c"
    # du pour récupérer la taille du fichier
        ((SI+="$(du -s "$chemin" | cut -f1)"))
    # et on compte
        ((NB++))
done

# verbeux
echo -e "\
$NB fichiers
$SI octets, soit: $(bc <<< "scale=2;($SI/1000000)") Go.
Temps cumulé: $T secondes
soit: \c"

# split SSS.MMS en 2 variables
MMS="$(echo "$T" | cut -f2 -d".")"
T="$(echo "$T" | cut -f1 -d".")"

# conversion en DD:HH:MM:SS:MMS
D=$((T/60/60/24))
H=$((T/60/60%24))
M=$((T/60%60))
S=$((T%60))

(( D > 0 )) && echo -e "$D jours \c"
(( H > 0 )) && echo -e "$H heures \c"
(( M > 0 )) && echo -e "$M minutes \c"
(( D > 0 || H > 0 || M > 0 )) && echo -e "$S secondes \c"
(( D > 0 || H > 0 || M > 0 || MMS > 0)) && echo "$MMS millisecondes"
}

exit 0
```

Exemples en sortie:

```
$ ./duration.sh Nouveau\ dossier/
3 fichiers
573232 octets, soit: .57 Go.
Temps cumulé: 131.06 secondes
soit: 2 minutes 11 secondes 06 millisecondes
$ ./duration.sh 2017-06-09/
42 fichiers
23043816 octets, soit: 23.04 Go.
Temps cumulé: 5044.18 secondes
soit: 1 heure 24 minutes 4 secondes 18 millisecondes
```

Ressources

- http://lukemiller.org/index.php/2014/10/ffmpeg-time-lapse-notes/
- http://mahugh.com/2015/04/29/creating-time-lapse-videos/

Cas d'usage DVR

Je possède un casque FPV Eachine qui permet de faire des DVR (Direct Video Recorder).
En voulant partager sur Mega; j'utilise un compte gratuit de 25Go qui sert exclusivement à ce genre de partage et surtout permet le streaming; je m'apperçois que la vidéo ne se lit pas en streaming avec le lecteur intégré, seulement le son et sur un DVR le son c'est les parasite des fréquences...
Fouinons...

```
$ ffmpeg -hide_banner -i PICT0045.AVI
[avi @ 0x557325e6ac80] non-interleaved AVI
Guessed Channel Layout for Input Stream #0.1 : mono
Input #0, avi, from 'PICT0045.AVI':
  Duration: 00:02:58.39, start: 0.000000, bitrate: 7963 kb/s
    Stream #0:0: Video: mjpeg (Baseline) (MJPG / 0x47504A4D), yuvj420p(pc,
bt470bg/unknown/unknown), 720x480, 7779 kb/s, 25.06 fps, 25.06 tbr, 25.06
tbn, 25.06 tbc
    Stream #0:1: Audio: pcm_s16le ([1][0][0][0] / 0x0001), 8000 Hz, mono,
s16, 128 kb/s
At least one output file must be specified
```

MJPEG, connait pas et puis le “non-interleaved AVI”
AVI (Audio Video Interleave) est un conteneur vidéo, dedans nous avons la piste vidéo encodée en mjpeg et la piste audio en pcm_s16le voir la doc
Le non-interleaved AVI signifie, si je saisis bien, que le flux vidéo est compressé et l'audio qui fût ajouté, ne l'est pas.Si je compare avec un autre fichier issu d'une runcam split mini qui pond des .MOV:

```
$ ffmpeg -hide_banner -i RC_VID_0000.MOV
[mov,mp4,m4a,3gp,3g2,mj2 @ 0x556598d6cc80] st: 0 edit list: 1 Missing key
frame while searching for timestamp: 2000
[mov,mp4,m4a,3gp,3g2,mj2 @ 0x556598d6cc80] st: 0 edit list 1 Cannot find an
index entry before timestamp: 2000.
Guessed Channel Layout for Input Stream #0.1 : stereo
Input #0, mov,mp4,m4a,3gp,3g2,mj2, from
'../2020-07-23_mission_epave/RC_VID_0000.MOV':
  Metadata:
    major_brand     : qt
    minor_version   : 0
    compatible_brands: qt
    creation_time   : 2015-04-27T12:34:35.000000Z
```

```
  Duration: 00:03:28.00, start: 0.000000, bitrate: 21745 kb/s
    Stream #0:0(eng): Video: h264 (High) (avc1 / 0x31637661), yuv420p,
1920x1080, 20047 kb/s, 30 fps, 30 tbr, 60k tbn, 120k tbc (default)
    Metadata:
      creation_time   : 2015-04-27T12:34:35.000000Z
      handler_name    : Video Handler
      encoder         : iCatch AVCC
    Stream #0:1(eng): Audio: pcm_s16le (sowt / 0x74776F73), 44100 Hz,
stereo, s16, 1411 kb/s (default)
    Metadata:
      creation_time   : 2015-04-27T12:34:35.000000Z
      handler_name    : Audio Handler
At least one output file must be specified
```

Je n'ai pas trouvé les formats supportés par le lecteur intégré de Mega, je sais par expérience que le 2e fichier réencodé en h.264 en conteneur AVI passe.
Les 2 encodes suivantes fonctionnent:

```
$ ffmpeg -i PICT0045.AVI -c:v libx264 -an test.mp4
# encodage en 1 passe en h.264, le crf par défaut de 23 et suppression piste
audio avec -an

$ ffmpeg -i PICT0045.AVI -pix_fmt yuv420p -b:v 4000k -c:v libx264 -an
testchroma.mp4
# encodage en 1 passe en h.264, crf par défaut, suppression piste audio et
modification chroma
```

Pour la 2e solution c'est suite à une recherche

> The usual reason for this is the pixel format. The MJPEG may have 4:2:2 or 4:4:4 chroma sampling which most players don't support in H264. So try

Source
Le chrominance est la balance des couleurs et de la luminosité encodées dans une image, un bonne explication, c'est ce qui semble coincer.
La commande réduit au passge le poids des vidéos d'un facteur de 1.6-1.8 environ, la qualité semble la même, sur 720×480 cela suffit amplement, ce sont des DVR...
Et la commande qui va bien pour traiter en boucle dans le dossier courrant en concervant le nom d'origine:

```
for file in *.AVI ; do name=`echo "$file" | cut -d'.' -f1` ; echo "$name" ;
ffmpeg -loglevel panic -i "$file" -pix_fmt yuv420p -b:v 4000k -c:v libx264 -
an "${name}.mp4" ; done && notify-send 'Transcodage fini' --icon=dialog-
information
```