Video TIPS

Getting INFO

```
avprobe <file>
```

Convert to webm

```
ffmpeg -i input.video -threads 4 -b:v 1M -crf 10 output.webm
```

Repair index

```
mencoder -idx input.video -ovc copy -oac copy -o output.video
```

from images to video

```
# 5 images per seconds
ffmpeg -y -framerate 5 -pattern_type glob -i '*.jpg' -c:v libx264 -vf
"fps=25,scale=720:-1" out.mp4
```

MKV

convert to mkv

```
mkvmerge -o output.mkv <inputfile>
```

convert to mkv and downgrade quality

```
avconv -i <inputfile> -map 0 -c:v libx264 -crf 20 -c:a copy -c:s copy
<outfile>.mkv
```

concatenate all *.mp4 files to output.mkv

```
mkvmerge -o output.mkv $(echo *.mp4 | sed "s| | +|g")
```

x264

From x265 to x264

```
ffmpeg -xerror -i input.mkv -hide_banner -threads 0 -map 0 -c:a copy -c:s
copy -c:v libx264 -pix_fmt yuv420p output.mkv
```

script

convert files to mkv and downgrade them if greater than specific value

tomkv (chmod +x)

```
#!/bin/bash

OUTDIR=out
LIMIT_MBYTE=1500
QUALITY=20 # lower values are better

mkdir -p $OUTDIR
set -x
for f in "$@"
do
    [ -f "$f" ] || continue

    SIZE=$(stat -c%s "$f")
    OUTNAME=$OUTDIR/${f%.*}.mkv
    [ -f "$OUTNAME" ] && ( echo "Skip $OUTNAME"; continue )

    if [ $SIZE -gt $(($LIMIT_MBYTE*1000000)) ]; then
        echo avconv $f
        avconv -i "$f" -map 0 -c:v libx264 -crf $QUALITY -c:a copy -c:s copy
"$OUTNAME"
    else
        echo mkvmerge $f
        mkvmerge "$f" -o "$OUTNAME"
    fi
done
```

Usage to convert into ./out folder

```
tomkv file1 file2 ...
```

on motion

```
#!/bin/bash

CHROOT=/media/camere/

FILENAME=$1
```

```
# name of the first folder level inside CHROOT
INPUT=$(cut -d'/' -f1 <<<"${FILENAME/$CHROOT/}")

# lowercase
INPUT=${INPUT,,}

LOCK=${CHROOT}${INPUT}.lock
[ -f $FILENAME ] && ln -sf $FILENAME ${CHROOT}${INPUT}.jpg


[ -f $LOCK ] && exit 0

touch $LOCK
name="`date +%Y-%m-%d_%H.%M`"
i=${INPUT/camera/}
video=$CHROOT/camera$i/$name.mp4

echo "acquire from camera $i"
( ffmpeg -i rtsp://admin:test@192.168.0.9$i/12 -timeout 10 -stimeout 5000000
-c:v copy -an -t 300 -y $video </dev/null >/dev/null 2>/tmp/camera$i.log ;
rm -f $LOCK ) &
```

telegram

```
#!/bin/sh

set -x

caption=$1
video=$2
CHATID=$3

#
TOKEN="xxxx:xxxx"

[ -z "$CHATID" ] && exit 1


curl -F chat_id="$CHATID" \
    -F document=@"${video}" \
    -F caption="$caption" https://api.telegram.org/bot${TOKEN}/sendDocument
```

preview day

```
#!/bin/bash

set -x
```

```
CHATID=$1
CHATID=${CHATID:=xxxxxx}
i=$2
i=${i:=1}
SPEED=${3}
SPEED=${SPEED:=50}

INVSPEED=$(echo "1/$SPEED" | bc -l)

[ -z "$CHATID" ] && exit 1

BASE=/media/camere

DAY=$(date +"%Y-%m-%d")
video=$BASE/day-$i-$DAY.mp4

echo "create $video"

# 0.02: 20 volte la velocita

if [ ! -f $video ]; then
    ffmpeg -f concat -safe 0 \
        -i <(find "$BASE/camera$i" -mmin -1440 -name "*mp4" -printf "%AT
file '%p'\n" | sort -k1.1n | cut -c 21- ) \
        -c:v libx264 -filter:v "setpts=$INVSPEED*PTS" -an -y $video
        # -r 0.1 -c:v libx264 -an -y $video
fi


~/telegram.sh "Camera $i" "${video}" $CHATID
```

camera motion

```
INPUT=rtsp://foscam:foscam1@192.168.2.14/videoMain
ffprobe -i $INPUT
```

```
ffmpeg -rtsp_transport tcp -i $INPUT -c:v libx264 -an -y test.mp4
```

shinobi

```
ffmpeg -loglevel warning -analyzeduration 5000000 -probesize 5000000 \
 -rtsp_transport tcp -i rtsp://admin:admin@192.168.2.29:554/12 \
 -preset ultrafast -crf 15 -an -c:v libx264 -r 2 -f hls -s 640x380 \
 -max_muxing_queue_size 1024 \
 -tune zerolatency -g 1 -hls_time 2 -hls_list_size 3 -start_number 0 -
hls_allow_cache 0 \
 -hls_flags +delete_segments+omit_endlist
"/dev/shm/streams/8Pkk5cE2xY/LIBm503cxy/s.m3u8" \
 -f singlejpeg -vf fps=0.5 -s 640x380 pipe:0 \
```

```
 -update 1 -r 1 "/dev/shm/streams/8Pkk5cE2xY/LIBm503cxy/s.jpg" \
 -y -f mpegts -c:v libx264
http://127.0.0.1:8080/streamIn/8Pkk5cE2xY/LIBm503cxy/1
```