# Fastest way to extract frames using ffmpeg? [closed]

Asked 12 years, 6 months ago  Modified 9 months ago

Viewed 506k times

238

**Closed.** This question is not about programming or software development. It is not currently accepting answers.

This question does not appear to be about a specific programming problem, a software algorithm, or software tools primarily used by programmers. If you believe the question would be on-topic on another Stack Exchange site, you can leave a comment to explain where the question may be able to be answered.

Closed 10 months ago.

The community reviewed whether to reopen this question 10 months ago and left it closed:

> Original close reason(s) were not resolved

Improve this question

Hi I need to extract frames from videos using ffmpeg.. Is there a faster way to do it than this:

```
ffmpeg -i file.mpg -r 1/1 $filename%03d.jpg
```

?

video ffmpeg

Share

Improve this question

Follow

asked Jun 9, 2012 at 0:20

Stpn
6,394 ● 8 ● 48 ● 96

1 ^ updated url: trac.ffmpeg.org/wiki/Seeking – Lambart Feb 9, 2018 at 21:24

10 If you have CPU cycles to spare, you can extract from multiple videos in parallel: `parallel -i {} -r 1/1 {.}-%03d.bmp ::: *mpg` – Ole Tange Feb 19, 2018 at 0:50

Might aswell have this open. It's more useful to stackoverflow to let people find this. Trying to minimize number of questions for e.g. browsing is not really useful target. – Antti Rytsölä Aug 17 at 9:42

## 10 Answers

Sorted by: Highest score (default)

256

If the JPEG encoding step is too performance intensive, you could always store the frames uncompressed as BMP images:

```
ffmpeg -i file.mpg -r 1/1 $filename%03d.bmp
```

This also has the advantage of not incurring more quality loss through quantization by transcoding to JPEG. (PNG is also lossless but tends to take much longer than JPEG to encode.)

Share Improve this answer Follow

edited Jun 9, 2012 at 18:28

answered Jun 9, 2012 at 15:53

Multimedia Mike
13.2k ● 5 ● 48 ● 62

18 This results in a *lot* of frame dropping on my machine. Can I tell ffmpeg to render everything? – anon Jul 7, 2016 at 17:19

75 @Evi1M4chine just remove the -r parameter this will extract all frames – studioj Nov 15, 2016 at 6:14

21 I'd like to add that while JPEG isn't really hard on the CPU, uncompressed Bitmaps is really really hard on the storage, so I doubt you'll get higher throughput with BMP compared to JPEG. – Marcus Müller Apr 3, 2017 at 13:12

8 To extract all frames: `ffmpeg -r 1 file.mp4 -r 1 "$filename%03d.png"` – fiatjaf Dec 25, 2017 at 19:52

29 You mean `ffmpeg -r 1 -i file.mp4 -r 1 "$filename%03d.png`, right? (you were missing the `-i`) – Joschua Jan 23, 2020 at 19:00

102

Came across this question, so here's a quick comparison. Compare these two different ways to extract one frame per minute from a video 38m07s long:

```
time ffmpeg -i input.mp4 -filter:v fps=fps=1/60
```

```
ffmpeg_%0d.bmp
```

1m36.029s

This takes long because ffmpeg parses the entire video file to get the desired frames.

```
time for i in {0..39} ; do ffmpeg -accurate_seek -ss `echo $i*60.0 | bc` -i input.mp4   -frames:v 1 period_down_$i.bmp ; done
```

0m4.689s

This is about 20 times faster. We use fast seeking to go to the desired time index and extract a frame, then call ffmpeg several times for every time index. Note that `-accurate_seek` is the default , and make sure you add `-ss` before the input video `-i` option.

Note that it's better to use `-filter:v -fps=fps=...` instead of `-r` as the latter may be inaccurate. Although the ticket is marked as fixed, I still did experience some issues, so better play it safe.

Share Improve this answer Follow

answered Feb 4, 2015 at 12:56

community wiki
blutorange

10 Feb 2016: as of ffmpeg 2.1, the accurate seek option is now default - trac.ffmpeg.org/wiki/Seeking – mafrosis Feb 7, 2016

at 7:19

4 `bc` is not a native Ubuntu package, instead one can use bash: `let "i = $i * 60"`. BTW - excellent idea – gilad905 Dec 12, 2017 at 16:01

6 Good tip adding `-ss` before `-i`. Otherwise, the whole video will be decoded and the unrequired frames will be discarded – MoustafaAAtta Jan 16, 2018 at 1:42

2 Since this is top of google, I'd like to note that in 2018 this is still an approach that yields dividends. The best result seems to be running one `ffmpeg` per core of your host - which (for bmp) yields near-linear improvements in speed (until you hit some other bottleneck, like disk). – Knetic Apr 11, 2018 at 23:06

1 @Franva should be smt like this (UNTESTED): ``-ss `let "j = $i * 60" && echo $j` `` – gilad905 Aug 10, 2021 at 14:06

71

This is simpler than all the other commands so far:

```
ffmpeg -i input.mp4 '%04d.png'
```

Change `04` to however many digits you need to hold all frames. Make sure to always have a `0` before the number so output frame names are zero-padded.

Share Improve this answer Follow

edited Sep 13, 2021 at 0:59

answered Mar 8, 2021 at 3:54

makeworld
1,683 ● 17 ● 18

No need for `-pix_fmt rgba`. The PNG encoder will automatically choose the appropriate pixel format. – llogan Mar 8, 2021 at 17:49

1 @llogan thanks, removed it with an edit. Got a source for that though? – makeworld Mar 8, 2021 at 19:27

You can view a list of supported pixel formats for the PNG encoder with `ffmpeg -h encoder=png`. See `avcodec_find_best_pix_fmt_of_list` documentation. – llogan Mar 8, 2021 at 19:52

1 Needed to add an additional `d` as `ffmpeg -i input.mp4 '%04d.png'` – kungphil Aug 1, 2021 at 21:55

1 On Windows I had to use double quotes `ffmpeg -i input.mp4 "%04d.png"` – Jeremy Thompson Jun 6 at 8:23

22

Output one image every minute, named img001.jpg, img002.jpg, img003.jpg, etc. The %03d dictates that the ordinal number of each output image will be formatted using 3 digits.

```
ffmpeg -i myvideo.avi -vf fps=1/60 img%03d.jpg
```

Change the `fps=1/60` to `fps=1/30` to capture a image every 30 seconds. Similarly if you want to capture a image every 5 seconds then change `fps=1/60` to `fps=1/5`

SOURCE: https://trac.ffmpeg.org/wiki/Create a thumbnail image every X seconds of the video

Share Improve this answer Follow

answered Oct 8, 2020 at 10:27
mouse_freak
321 ● 5 ● 6

---

20

If you know exactly which frames to extract, eg 1, 200, 400, 600, 800, 1000, try using:

```
select='eq(n\,1)+eq(n\,200)+eq(n\,400)+eq(n\,600)+ec
\
       -vsync vfr -q:v 2
```

I'm using this with a pipe to Imagemagick's montage to get 10 frames preview from any videos. Obviously the frame numbers you'll need to figure out using `ffprobe`

```
ffmpeg -i myVideo.mov -vf \

select='eq(n\,1)+eq(n\,200)+eq(n\,400)+eq(n\,600)+ec
\
    -vsync vfr -q:v 2 -f image2pipe -vcodec ppm -
\
  | montage -tile x1 -geometry "1x1+0+0<" -quality
100 -frame 1 - output.png
```

.

**Little explanation:**

1. In ffmpeg expressions `+` stands for OR and `*` for AND

2. `\,` is simply escaping the `,` character

3. Without `-vsync vfr -q:v 2` it doesn't seem to work but I don't know why - anyone?

Share Improve this answer Follow

edited Apr 11, 2020 at 2:28

answered Mar 23, 2016 at 22:25

Voy
6,214 ● 3 ● 54 ● 66

---

`eq(n\,1)` would extract 2nd frame (order starts at 0)
– Zimba Feb 15, 2021 at 4:59

---

2 `-q:v` is an alias for `-qscale:v` ([trac.ffmpeg.org/wiki/Encode/MPEG-4](trac.ffmpeg.org/wiki/Encode/MPEG-4)) and controls image quality. `-vsync vfr` is the video sync method (you first need to understand `-vf` which is a filtergraph) . According to the docs `vfr` means `Frames are passed through with their timestamp or dropped so as to prevent 2 frames from having the same timestamp.` *I think* this is to avoid the default option `cfr` as stated here [ffmpeg.org/ffmpeg.html](ffmpeg.org/ffmpeg.html) – Antonio Gomez Alvarado Mar 2, 2021 at 8:59

---

9

This worked for me

```
ffmpeg -i file.mp4 -vf fps=1 %d.jpg
```

Share Improve this answer Follow

answered Jul 13, 2020 at 10:32

Kishan Vaghela
7,898 ● 5 ● 44 ● 70

This extracts one frame per second, not every frame according to the original video FPS value.
– Andrea Lazzarotto Jul 15, 2022 at 14:22

You can get framerate using `ffprobe -v 0 -of csv=p=0 -select_streams v:0 -show_entries stream=r_frame_rate infile` and they use fps=<value>
– Kishan Vaghela Jul 15, 2022 at 14:59

You don't need to specify the fps parameter.
– Andrea Lazzarotto Jul 15, 2022 at 15:06

use `-r 1` instead of `-vf fps=1` – a55 Feb 22, 2023 at 12:43

5

I tried it. 3600 frame in 32 seconds. your method is really slow. You should try this.

```
ffmpeg -i file.mpg -s 240x135 -vf fps=1 %d.jpg
```

Share Improve this answer Follow

edited Nov 27, 2019 at 14:19

answered Jan 30, 2019 at 14:28
Kübra
197 ● 3 ● 10

1 I am trying `ffmpeg -i "input URL" -vf fps=1/5 out%d.png` where the input URL has to be an https link.

– feed_me_pi Jun 28, 2020 at 6:59

2 Sure, scaling them to a tiny size makes it faster, but that's not what he asked for. – PRMan Dec 1, 2021 at 18:38

2

In my case I need frames at least every second. I used the 'seek to' approach above but wondered if I could parallelize the task. I used the N processes with FIFO approach here: https://unix.stackexchange.com/questions/103920/parallelize-a-bash-for-loop/216475#216475

```
open_sem(){
  mkfifo /tmp/pipe-$$
  exec 3<>/tmp/pipe-$$
  rm /tmp/pipe-$$
  local i=$1
  for((;i>0;i--)); do
    printf %s 000 >&3
  done
}
run_with_lock(){
    local x
    read -u 3 -n 3 x && ((0==x)) || exit $x
    (
    "$@"
    printf '%.3d' $? >&3
    )&
}
N=16
open_sem $N
time for i in {0..39} ; do run_with_lock ffmpeg -
ss `echo $i` -i /tmp/input/GOPR1456.MP4  -frames:v
1 /tmp/output/period_down_$i.jpg  & done
```

Essentially I forked the process with & but limited the number of concurrent threads to N.

This improved the 'seek to' approach from 26 seconds to 16 seconds in my case. The only problem is the main thread does not exit cleanly back to the terminal since stdout gets flooded.

Share Improve this answer Follow

edited Apr 11, 2020 at 0:31
Ondra Žižka
**46.7k** ● 47 ● 226 ● 297

answered Feb 15, 2018 at 15:10
Tycon
**133** ● 2 ● 12

2

Same as @makeworld's answer but also addresses issues mentioned here regarding frame count inconsistencies and need for vsync and here regarding use of vsync:

```
// reliably works for jpg output (and probably png too)
ffmpeg -i rgb.mov -vf setpts=N/FR/TB -vsync 0 ./images/%05d.jpg

// reliably works for png output only
ffmpeg -i rgb.mov -vsync 0 ./images/%05d.png
```

Share Improve this answer Follow

answered Apr 7, 2022 at 10:07

danday74
56.7k ● 53 ● 263 ● 330

---

1

Using `ffmpeg-6.0` on a Win-11 PC, I could not get it to work until I tried:

```
ffmpeg -i input.mp4 frame_%%03d.png
```

Not sure why I needed `%%` rather than the `%` mentioned above.

Share Improve this answer Follow

edited Mar 1 at 18:51

NotTheDr01ds
20.3k ● 7 ● 57 ● 86

answered Feb 29 at 17:15

kavanrat
**23** ● 3