



INTRODUCING FFMPEG

IT'S NOT THAT SCARY ONCE YOU GET TO KNOW IT!

Viski Barbora
bashynx@bashynx.com

25. 03. 2026

TABLE OF CONTENTS

- 1 WHAT IS FFMPEG
- 2 USING FFMPEG
- 3 MY PROJECT - VERTCLIP





What is FFmpeg

WHAT IS FFMPEG



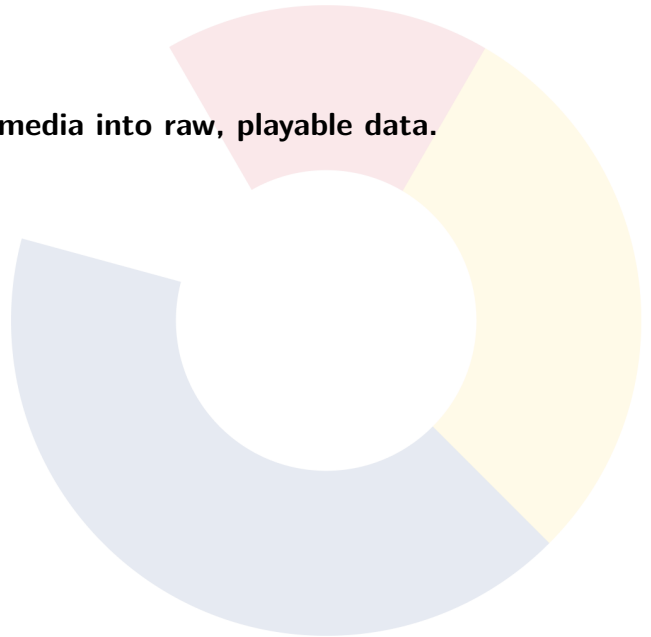
FIGURE 1: FFmpeg logo

It is a:

- free, open-source command-line framework.
- used to decode, encode, transcode, mux, demux, stream, filter and play media.
- multimedia framework for handling a wide range of formats, codecs and protocols.
- able to transcode, resize, watermark media and perform screen capture.

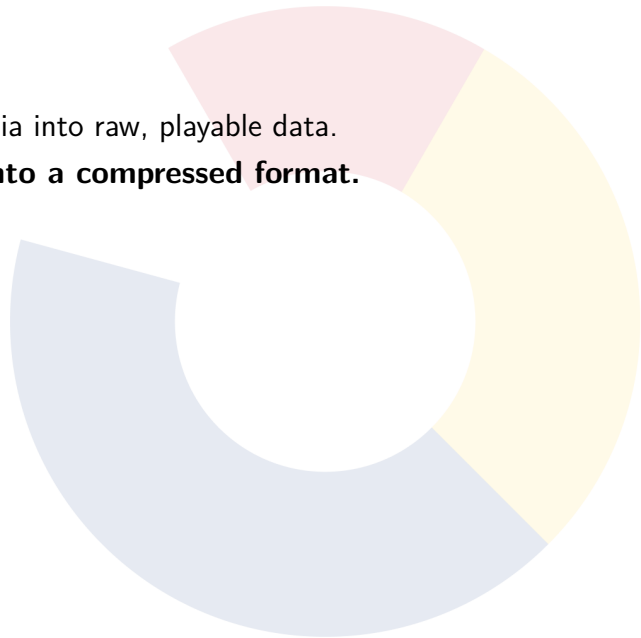
DEFINING TERMS

- **decode:** Convert compressed media into raw, playable data.



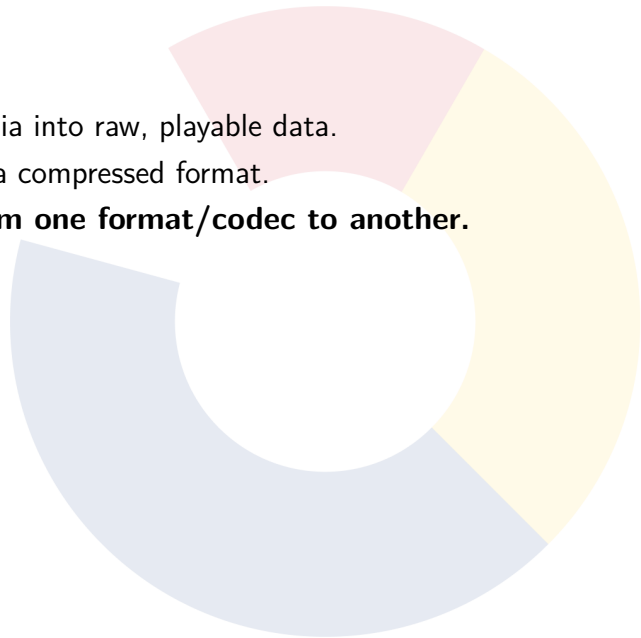
DEFINING TERMS

- decode: Convert compressed media into raw, playable data.
- **encode: Convert raw media into a compressed format.**



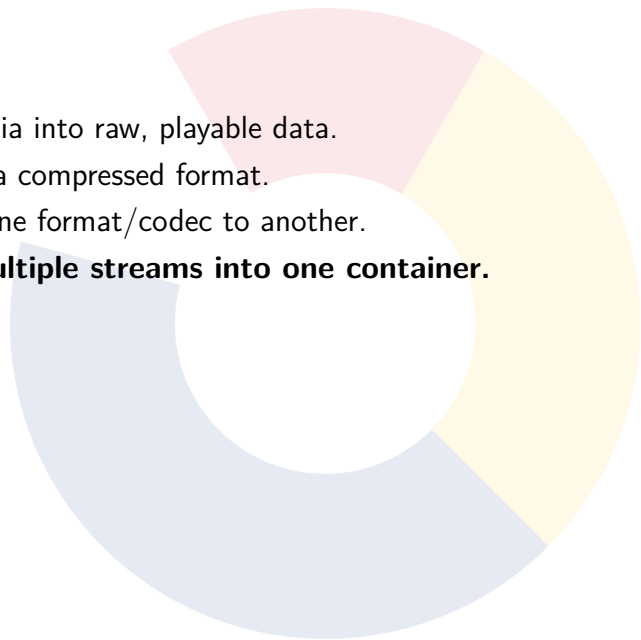
DEFINING TERMS

- decode: Convert compressed media into raw, playable data.
- encode: Convert raw media into a compressed format.
- **transcode: Convert media from one format/codec to another.**



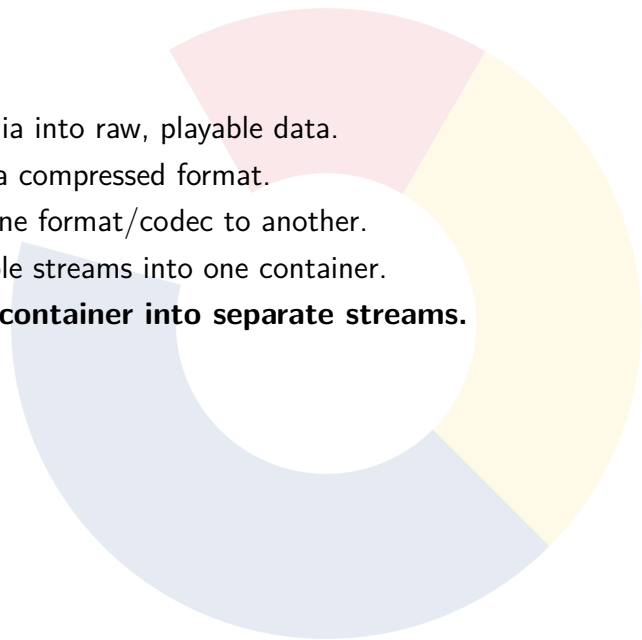
DEFINING TERMS

- decode: Convert compressed media into raw, playable data.
- encode: Convert raw media into a compressed format.
- transcode: Convert media from one format/codec to another.
- **mux (multiplex): Combine multiple streams into one container.**



DEFINING TERMS

- decode: Convert compressed media into raw, playable data.
- encode: Convert raw media into a compressed format.
- transcode: Convert media from one format/codec to another.
- mux (multiplex): Combine multiple streams into one container.
- **demux (demultiplex): Split a container into separate streams.**



DEFINING TERMS

- decode: Convert compressed media into raw, playable data.
- encode: Convert raw media into a compressed format.
- transcode: Convert media from one format/codec to another.
- mux (multiplex): Combine multiple streams into one container.
- demux (demultiplex): Split a container into separate streams.
- **codec: Algorithm/software that encodes and decodes media.**

DEFINING TERMS

- decode: Convert compressed media into raw, playable data.
- encode: Convert raw media into a compressed format.
- transcode: Convert media from one format/codec to another.
- mux (multiplex): Combine multiple streams into one container.
- demux (demultiplex): Split a container into separate streams.
- codec: Algorithm/software that encodes and decodes media.
- **protocol: Rules for transmitting data between systems.**

DEFINING TERMS

- decode: Convert compressed media into raw, playable data.
- encode: Convert raw media into a compressed format.
- transcode: Convert media from one format/codec to another.
- mux (multiplex): Combine multiple streams into one container.
- demux (demultiplex): Split a container into separate streams.
- codec: Algorithm/software that encodes and decodes media.
- protocol: Rules for transmitting data between systems.
- **container: A wrapper that holds streams, e.g., MP4, MKV.**

DEFINING TERMS

- decode: Convert compressed media into raw, playable data.
- encode: Convert raw media into a compressed format.
- transcode: Convert media from one format/codec to another.
- mux (multiplex): Combine multiple streams into one container.
- demux (demultiplex): Split a container into separate streams.
- codec: Algorithm/software that encodes and decodes media.
- protocol: Rules for transmitting data between systems.
- container: A wrapper that holds streams, e.g., MP4, MKV.
- **filter: Modify audio/video data, e.g., resize, overlay, adjust color.**

DEFINING TERMS

- decode: Convert compressed media into raw, playable data.
- encode: Convert raw media into a compressed format.
- transcode: Convert media from one format/codec to another.
- mux (multiplex): Combine multiple streams into one container.
- demux (demultiplex): Split a container into separate streams.
- codec: Algorithm/software that encodes and decodes media.
- protocol: Rules for transmitting data between systems.
- container: A wrapper that holds streams, e.g., MP4, MKV.
- filter: Modify audio/video data, e.g., resize, overlay, adjust color.
- **stream: Continuous flow of audio/video over a protocol.**

STREAM

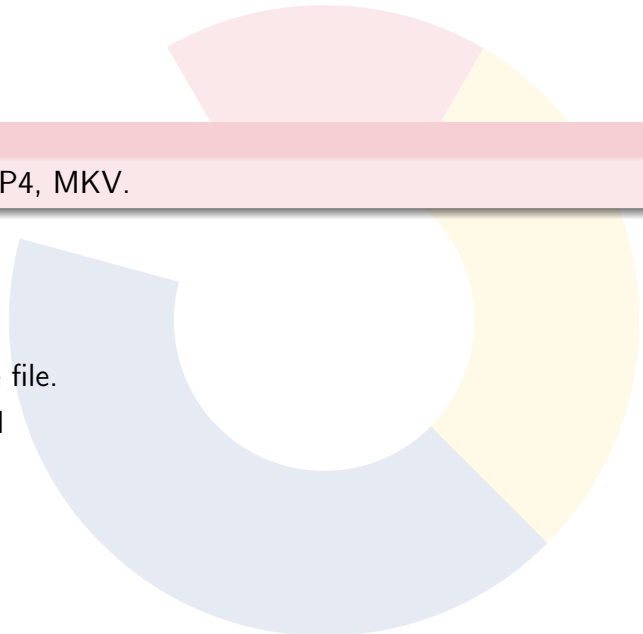
Continuous flow of audio/video over a protocol.

- Individual pieces of data inside of a container.
- A container can combine (mux) or split (demux) streams.
- Video, audio, subtitles and metadata are considered streams

CODEC

Algorithm/software that encodes and decodes media.

- Compresses or decompresses the streams.
- Single container can hold streams encoded with different codecs.
- Video codecs: H.264, H.265
- Audio codecs: AAC, MP3



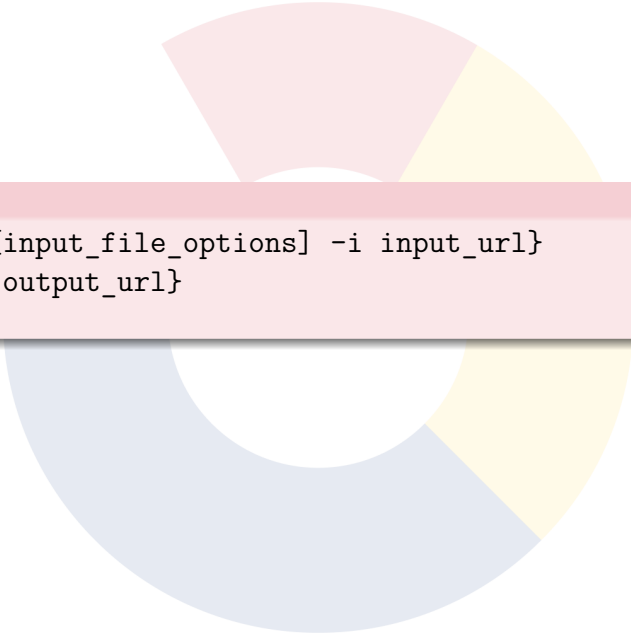
CONTAINER

A wrapper that holds streams, e.g., MP4, MKV.

- Holds multiple streams in a single file.
- Container types: MP4, MKV, AVI



Using FFmpeg



FFMPEG COMMAND

```
$ ffmpeg [global_options] {[input_file_options] -i input_url}  
... {[output_file_options] output_url}
```



CHANGE THE CONTAINER (FORMAT):

```
$ ffmpeg -i input.mp4 output.avi
```

CHANGE THE CONTAINER (FORMAT):

```
$ ffmpeg -i input.mp4 output.avi
```

EXTRACT AUDIO:

```
$ ffmpeg -i video.mp4 -vn -acodec mp3 audio.mp3
```

-**vn** ignore the video stream

-**acodec mp3** audio codec - mp3

EXTRACT A FRAME:

```
$ ffmpeg -i input.mp4 -ss 00:00:01 -frames:v 1 output.png
```

-**ss** start time for processing

-**frames:v 1** number of video frames to extract

EXTRACT A FRAME:

```
$ ffmpeg -i input.mp4 -ss 00:00:01 -frames:v 1 output.png
```

-ss start time for processing

-frames:v 1 number of video frames to extract

CUT A CLIP:

```
$ ffmpeg -ss 00:00:10 -to 00:00:20 -i input.mp4 -c copy clip.mp4
```

-to end time for processing

-c copy copy stream without re-encoding

RESIZE

```
$ ffmpeg -i input.mp4 -vf scale=1280:720 output.mp4
```

-vf scale video filter - scale

RESIZE

```
$ ffmpeg -i input.mp4 -vf scale=1280:720 output.mp4
```

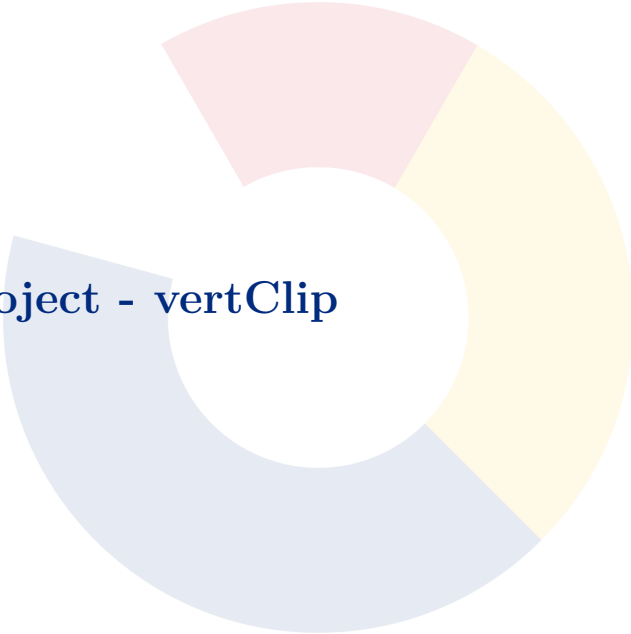
-vf scale video filter - scale

COMPRESS

```
$ ffmpeg -i input.mp4 -vcodec libx264 -crf 28 output.mp4
```

-vcodec libx264 use H.264 video codec
-crf 28 Constant Rate Factor

Lower CRF: better quality, bigger file
Higher CRF: worse quality, smaller file



My project - vertClip

WHY VERTCLIP

- I wanted a simple way of making a good looking vertical clip.
- Tools enabling this were either too complex or pricey.
- Using CLI is cool, and it's automizable.

Github repository: <https://github.com/HackmanClub/vertClip>

THE PROBLEM

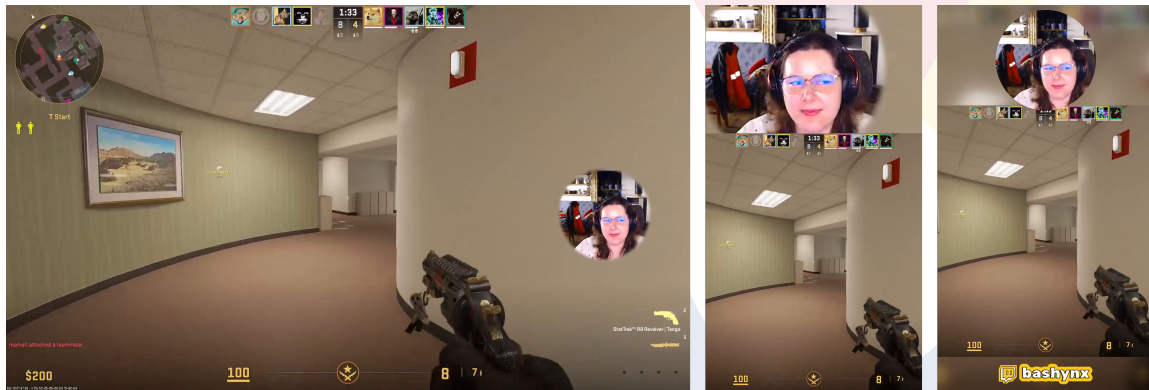


FIGURE 2: Left to right: Horizontal video, vertical video from Twitch and vertical video from vertClip

HOW IT WORKS

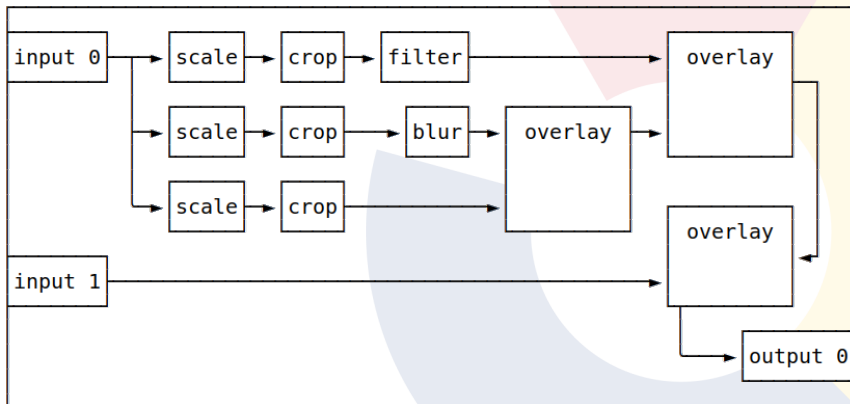
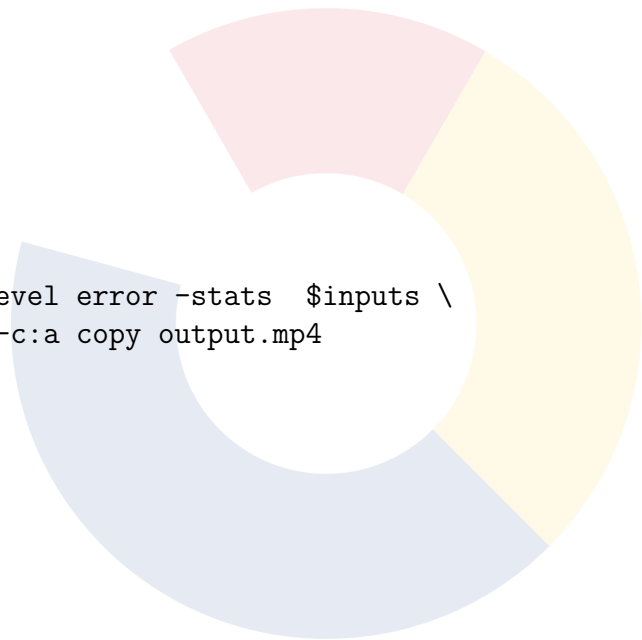


FIGURE 3: vertClip diagram explanation

HOW IT WORKS

The Command:

```
$ ffmpeg -hide_banner -loglevel error -stats $inputs \  
-filter_complex "$filter" -c:a copy output.mp4
```



HOW IT WORKS

The Command:

```
$ ffmpeg -hide_banner -loglevel error -stats $inputs \  
-filter_complex "$filter" -c:a copy output.mp4
```

The Filter:

```
filter="[0:v]format=rgba,crop=$cam_crop,scale=$cam_w:$cam_h, \  
geq=r='r(X,Y)':g='g(X,Y)':b='b(X,Y)':a='255*clip(1 - \  
(sqrt(pow(X-$(cam_w/2)),2)+pow(Y-$(cam_h/2)),2)) - \  
$(((cam_h/2)-cam_feather)))/$cam_feather, 0, 1)'[camera];
```

HOW IT WORKS

The Command:

```
$ ffmpeg -hide_banner -loglevel error -stats $inputs \  
-filter_complex "$filter" -c:a copy output.mp4
```

The Filter:

```
filter="[0:v]format=rgba,crop=$cam_crop,scale=$cam_w:$cam_h, \  
geq=r='r(X,Y)':g='g(X,Y)':b='b(X,Y)':a='255*clip(1 - \  
(sqrt(pow(X-$(cam_w/2)),2)+pow(Y-$(cam_h/2)),2)) - \  
$(((cam_h/2)-cam_feather)))/$cam_feather, 0, 1)'[camera]; \  
> [0:v]scale=-1:1920,crop=1080:1920,gblur=sigma=50[background];
```

HOW IT WORKS

The Command:

```
$ ffmpeg -hide_banner -loglevel error -stats $inputs \  
-filter_complex "$filter" -c:a copy output.mp4
```

The Filter:

```
filter="[0:v]format=rgba,crop=$cam_crop,scale=$cam_w:$cam_h, \  
geq=r='r(X,Y)':g='g(X,Y)':b='b(X,Y)':a='255*clip(1 - \  
(sqrt(pow(X-$(cam_w/2)),2)+pow(Y-$(cam_h/2)),2)) - \  
$(((cam_h/2)-cam_feather)))/$cam_feather, 0, 1)'[camera]; \  
[0:v]scale=-1:1920,crop=1080:1920,gblur=sigma=50[background]; \  
> [0:v]scale=-1:$main_height[game];
```

HOW IT WORKS

The Command:

```
$ ffmpeg -hide_banner -loglevel error -stats $inputs \  
-filter_complex "$filter" -c:a copy output.mp4
```

The Filter:

```
filter="[0:v]format=rgba,crop=$cam_crop,scale=$cam_w:$cam_h, \  
geq=r='r(X,Y)':g='g(X,Y)':b='b(X,Y)':a='255*clip(1 - \  
(sqrt(pow(X-$(cam_w/2)),2)+pow(Y-$(cam_h/2)),2)) - \  
$((cam_h/2)-cam_feather)))/$cam_feather, 0, 1)'[camera]; \  
[0:v]scale=-1:1920,crop=1080:1920,gblur=sigma=50[background]; \  
[0:v]scale=-1:$main_height[game]; \  
> [background][game]overlay=(W-w)/2:$cam_h:format=auto[tmp1];
```

HOW IT WORKS

The Command:

```
$ ffmpeg -hide_banner -loglevel error -stats $inputs \  
-filter_complex "$filter" -c:a copy output.mp4
```

The Filter:

```
filter="[0:v]format=rgba,crop=$cam_crop,scale=$cam_w:$cam_h, \  
geq=r='r(X,Y)':g='g(X,Y)':b='b(X,Y)':a='255*clip(1 - \  
(sqrt(pow(X-$(cam_w/2)),2)+pow(Y-$(cam_h/2)),2)) - \  
$(((cam_h/2)-cam_feather)))/$cam_feather, 0, 1)'[camera]; \  
[0:v]scale=-1:1920,crop=1080:1920,gblur=sigma=50[background]; \  
[0:v]scale=-1:$main_height[game]; \  
[background][game]overlay=(W-w)/2:$cam_h:format=auto[tmp1]; \  
> [tmp1][camera]overlay=(W-w)/2:25:format=auto[tmp2];
```

HOW IT WORKS

The Command:

```
$ ffmpeg -hide_banner -loglevel error -stats $inputs \  
-filter_complex "$filter" -c:a copy output.mp4
```

The Filter:

```
filter="[0:v]format=rgba,crop=$cam_crop,scale=$cam_w:$cam_h, \  
geq=r='r(X,Y)':g='g(X,Y)':b='b(X,Y)':a='255*clip(1 - \  
(sqrt(pow(X-$(cam_w/2)),2)+pow(Y-$(cam_h/2)),2)) - \  
$(((cam_h/2)-cam_feather)))/$cam_feather, 0, 1)'[camera]; \  
[0:v]scale=-1:1920,crop=1080:1920,gblur=sigma=50[background]; \  
[0:v]scale=-1:$main_height[game]; \  
[background][game]overlay=(W-w)/2:$cam_h:format=auto[tmp1]; \  
[tmp1][camera]overlay=(W-w)/2:25:format=auto[tmp2]; \  
> [tmp2][1:v]overlay=(W-w)/2:$logo_place:format=auto"
```



Thank You!

Any questions?

Viski Barbora mail: bashynx@bashynx.com
web: bashynx.com

REFERENCES

- 
-  FFmpeg, *ffmpeg Documentation*, <https://ffmpeg.org/ffmpeg.html>.
 -  Frantisek Korbel, *FFmpeg Basics*, 2012, ISBN 1-479-32783-2.