VIDEO CONCATINATION

PEGHEADS.ORG
GLOBAL DIGITAL AI EDUCATIONAL ORGANIZATION

A **FIVE** STEP PROCESS

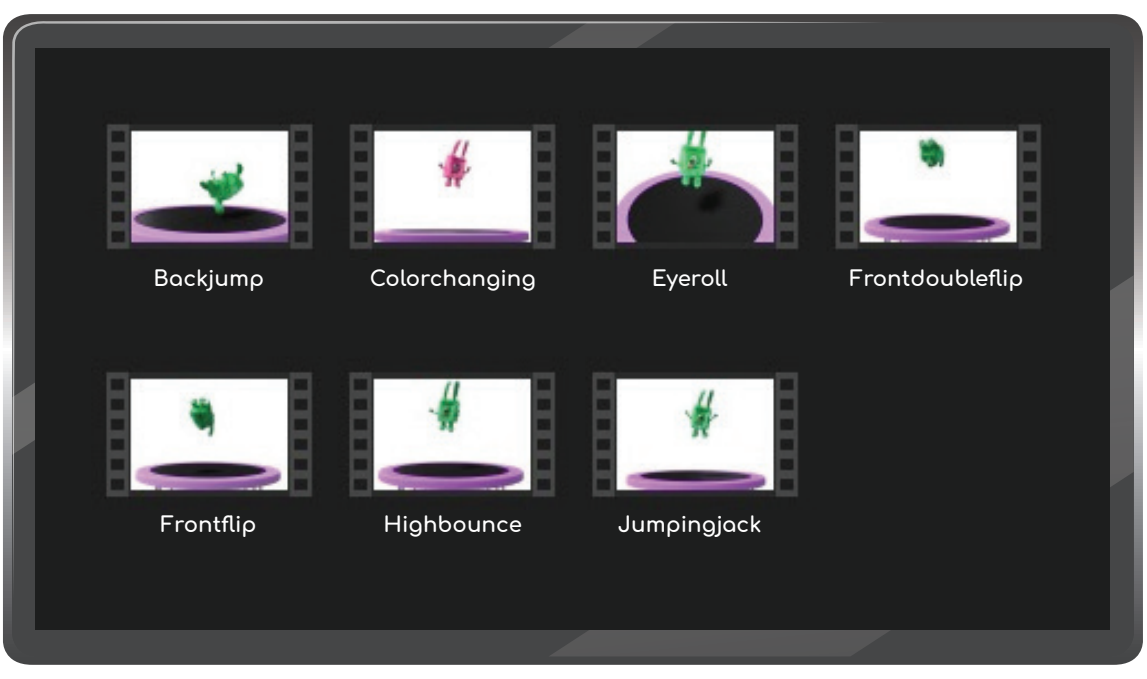
1. Get the different video clips that you want to merge
   (C1.mp4, C2.mp4, ... ,Cn.mp4) : Here 1 n is the no of video clips
2. Import MoviePy.editor library from Python
3. Save the video in different variables (C1, C2, ... , Cn)
4. Use the concatinate videoclips function from the MoviePy.editor library to merge the clips
5. Final merged video is generated and saved as (final.mp4)

GOAL

To combine video clips of different sizes into a single video file.

STEP 1

Gather all the video clips that you want to merge together. In our example we have used the following mp4 files.



STEP 2

MoviePy (full documentation) is a Python library for video editing: cutting, concatenations, title insertions, video compositing (a.k.a. non-linear editing), video processing, and creation of custom effects.

MoviePy can read and write all the most common audio and video formats, including GIF, and runs on Windows/Mac/Linux, with Python 2.7+ and 3 (or only Python 3.4+ from v.1.0).

Installation

MoviePy depends on the Python modules Numpy, imageio, Decorator, and tqdm, which will be automatically installed during MoviePy's installation. The software FFMPEG should be automatically downloaded/installed (by imageio) during your first use of MoviePy (installation will take a few seconds). If you want to use a specific version of FFMPEG, follow the instructions in config_defaults.py. In case of trouble, provide feedback.

Installation by hand: download the sources, either from PyPI or, if you want the development version, from GitHub, unzip everything into one folder, open a terminal and type:

```
$ (sudo) python setup.py install
```

Installation with pip: if you have pip installed, just type this in a terminal:

```
$ (sudo) pip install moviepy
```

Once installed just import moviepy.editor to your code as shown below:

```
$ from moviepy.editor import *
```

STEP 3

Create variables to store the video clips. To do this we are going to use the VideoFileClip function.

A VideoFileClip is a clip read from a video file (most formats are supported) or a GIF file. You load the video as follows:

```
myclip = VideoFileClip("some_video.avi")
myclip = VideoFileClip("some_animation.gif")
```

Note that these clips will have an fps (frame per second) attribute, which will be transmitted if you do small modifications of the clip, and will be used by default in write_videofile, write_gif, etc. For instance:

```
myclip = VideoFileClip("some_video.avi")
print (myclip.fps) # prints for instance '30'
# Now cut the clip between t=10 and 25 secs. This conserves the fps.
myclip2 = myclip.subclip(10, 25)
myclip2.write_gif("test.gif") # the gif will have 30 fps
```

STEP 4

Concatination or merge, we will be using the concatenate_videoclips which is a function built in moviepy.editor that does the concatenation for you.

```
$ final_clip = concatenate_videoclips([clip1, clip2, clip3,
clip4])
```

Concatenated result is now stored in the variable final_clip

STEP 5

Write the contents of final_clip to a mp4 file and get the final vedio which you can run and check. We use the write_videofile function to do this.

For example:

```
from moviepy.editor import *

# clip is the video from 00:56 to 01:06
clip = VideoFileClip("BackJumping.mp4")
clip2 = VideoFileClip("colorchange.mp4")
final_clip = concatenate_videoclips([clip, clip2])

final_clip.write_videofile("final.mp4")
```

Save and run, it will generate final.mp4 which is the end result.

STAY TUNED FOR OUR NEXT TUTORIAL

where we show an example of how to select required video clips from a dropdown menu.

Find the source code in our github page.

PEGHEADS.ORG
GLOBAL DIGITAL AI EDUCATIONAL ORGANIZATION