

Copy Victor - No-Zoom VQGAN\_CLIP\_Octaves\_v2.ipynb ☆

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

Collect trash, empty cache  
Show code

604

GPU Check  
Show code

```
GPU 0: Tesla K80 (UUID: GPU-944cdd37-63df-69bb-8369-1933c50cb689)

=====NVSMI LOG=====

Timestamp                : Fri Mar  4 19:32:44 2022
Driver Version            : 460.32.03
CUDA Version              : 11.2

Attached GPUS             : 1
GPU 00000000:00:04.0
  FB Memory Usage
    Total                 : 11441 MiB
    Used                  : 1778 MiB
    Free                  : 9663 MiB
  BAR1 Memory Usage
    Total                 : 16384 MiB
    Used                  : 2 MiB
    Free                  : 16382 MiB
```

1

2

(There should be a tick)

Comment Share Editing

RAM Disk

↑ ↓ ↶ ↷ ⚙ 📄 🗑 ⋮

## STEP 1 - Starting

1 - Check if you are connected to Google Collab GPU. (If the tick is not there, run the first 2 modules)

2 - Click to open the files tab

The screenshot shows the Copy Vitor IDE interface. The top bar includes the Copy Vitor logo, the title "Copy Vitor - No-Zoom VQGAN\_CLIP\_Octaves\_v2.ipynb", and a star icon. Below the title is a menu bar with "File", "Edit", "View", "Insert", "Runtime", "Tools", and "Help". The left sidebar shows a file explorer with a folder named "sample\_data". A context menu is open over the file explorer, showing options: "Upload", "Refresh", "New file", and "New folder". The main area displays two code cells. The first cell, "Collect trash, empty cache", has a "Show code" link and a status of "604". The second cell, "GPU Check", has a "Show code" link and displays the following output:

```
GPU 0: Tesla K80 (UUID: GPU-944cdd37-63df-69bb-8369-1933c50cb689)

=====NVSMI LOG=====

Timestamp                : Fri Mar  4 19:32:44 2022
Driver Version           : 460.32.03
CUDA Version             : 11.2

Attached GPUs            : 1
GPU 00000000:00:04.0
  FB Memory Usage
    Total                : 11441 MiB
    Used                 : 1778 MiB
    Free                 : 9663 MiB
  BAR1 Memory Usage
    Total                : 16384 MiB
    Used                 : 2 MiB
    Free                 : 16382 MiB
```

A warning banner at the bottom states: "Warning: you are connected to a GPU runtime, but not utilizing the GPU. [Change to a standard runtime](#)".

## STEP 2 - Uploading the base image

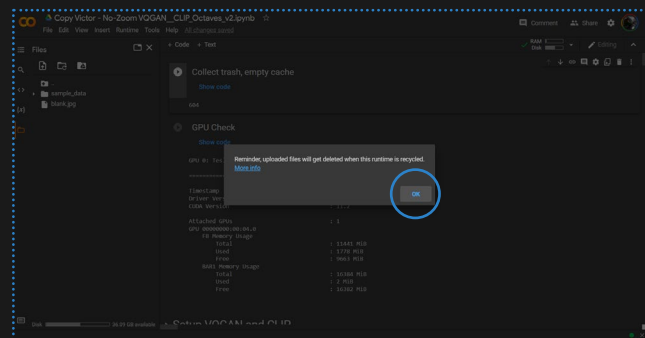
1 - Right-click on the file's tab. You should see new options (Upload, Refresh, New File, New folder)

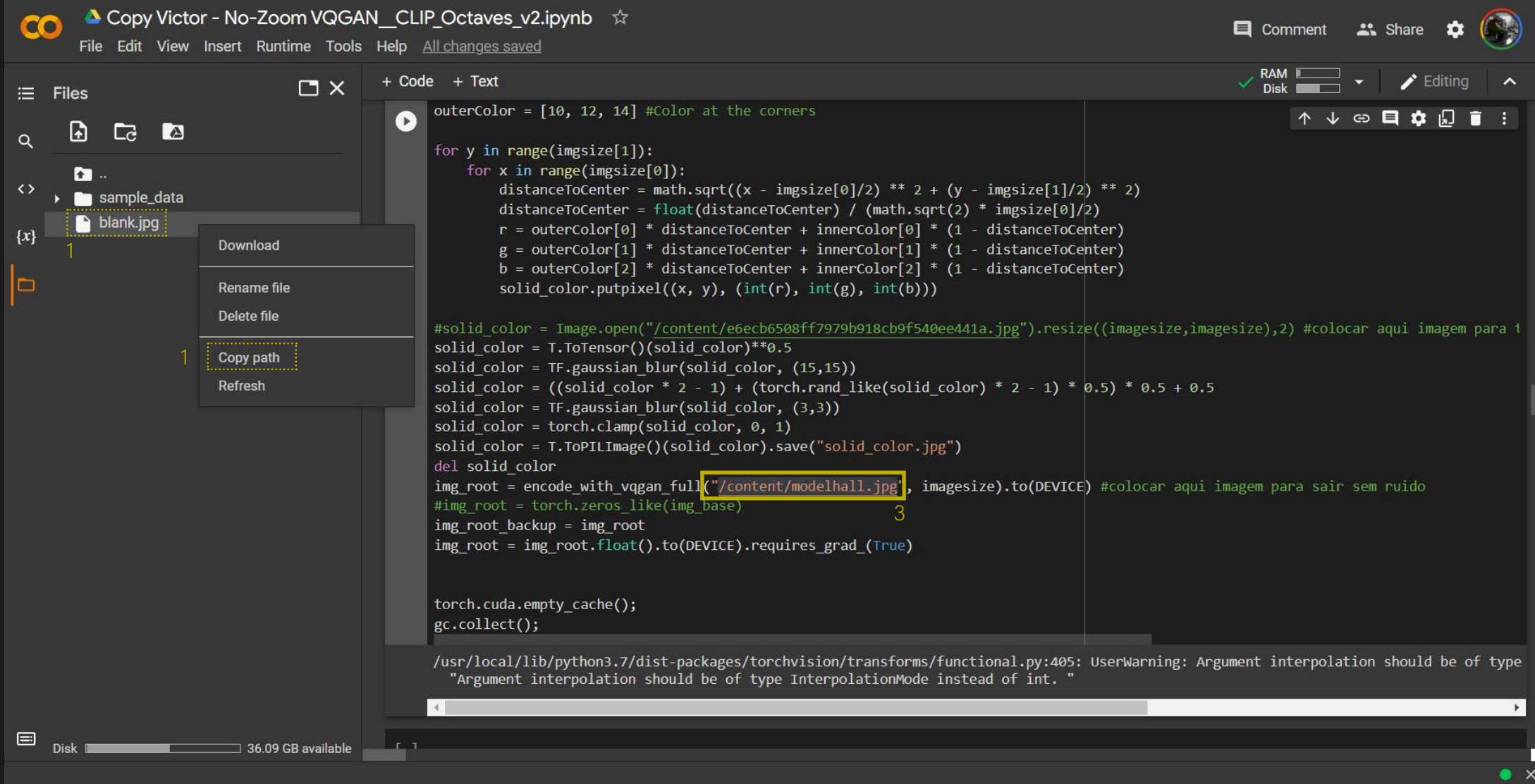
2 - Click "Upload" to insert a base image.

(This image will be background used for AI hallucinations. It could be a solid color image, a sketch or photo)

**image should be:** 500 x 500 pixels; jpg format; simple and short name

- Click "Ok" in the dial box to conclude the upload. It means that once the script is restarted all the files inserted and generated will be lost.





### STEP 3 - Setting the path

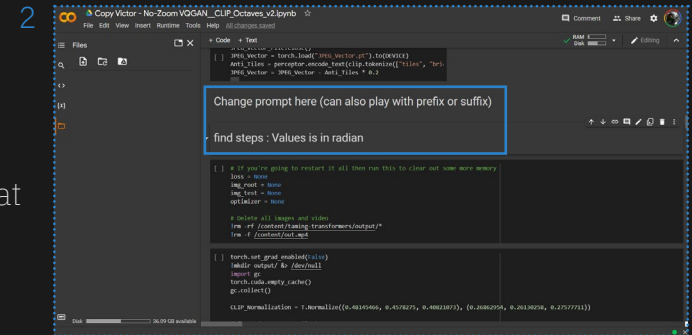
1 - Right-click on the uploaded image name. Click on the option "Copy path"

2 - Scroll down the script until you find the module "Change prompt here" - as in the screenshot in the left.

In this module you will find the line:

> `img_root = encode_with_vqgan_full("/content/modelhall.jpg", imagesize).to(DEVICE)` #this is the line that will determine which image will be used for the background

3 - Paste (CTRL + V) the copied path inside the ". In this example this line will change from "/content/modelhall.jpg" to "/content/blank.jpg"







Files



sample\_data

blank.jpg

{x}



Disk 36.09 GB available

+ Code + Text

RAM Disk Editing

```
prompt_encoded /= prompt_encoded.norm()
return prompt_encoded
if mean == False:
    return prompts_encoded

1 prompt_pre = ["Zaha hadid", "Frei Otto", "Niemeyer"]
#prompt_pre = ["trevor henderson", "andreas achenbach", "Erik Bodom," light artist "James Turrell","Dan Flavin","Tracey Emin,"
2 prompt_main = ["impossible cosmic space bridge"]
prompt_suffix1 = ["creepypasta", "scp", "ghost sto ry", "folklore", "folk tale"]
prompt_suffix2 = ["", 1831, "", 1875, "", 1847, "", 1871] #", 1847, The Walters Art Museum"
prompt_suffix3 = ["", Horror, Supernatural", "", Supernatural, Horror, ""]
#prompt = EncodeMultiPrompt(prompt_pre, prompt_main, prompt_suffix1, prompt_suffix2)
prompt = EncodeMultiPrompt(prompt_pre, prompt_main)
#This is just to put random shit in the prompt. The 2x multiplier is arbitrary and
#the hardtanh to limit it to (-1,1) is to prevent big outliers that might fuck up the gist
#of the prompt.
prompt = prompt + HardTanh(torch.randn_like(prompt) * 2.0) * 0.02

prompt_non_pre = ["words", "letters", "text", "writing", "font", "fonts", "signature", "signatures", "typeface", "typefaces", "handwri
prompt_non_main = ["text", "writing", "font", "signature", "handwriting", ""]
prompt_non = EncodeMultiPrompt(prompt_non_pre, prompt_non3main)

prompt_non2_pre = ["cartoon", "cartoons", "anime", "animes", "flash animation", "flash game", "manga"]
prompt_non2_main = ["screenshot", ""]
prompt_non2 = EncodeMultiPrompt(prompt_non2_pre, prompt_non2_main)

#prompt = prompt - prompt_non * 0.1
#prompt = prompt - prompt_non2 * 0.1
#prompt = prompt + JPEG_Vector * 0.05
#prompt = prompt / torch.linalg.norm(prompt)
#prompt = prompt + JPEG_Vector * 0.05

#228 because it's going to be cropped after transforms
```

## STEP 4 - Changing the parameters (this is where the fun begins!)

1 - In the same module, scroll up to find `> prompt_pre = ["Zaha hadid", "Frei Otto", "Niemeyer"]`

In this line you can put your own visual references (can be artists, architects or visual elements; it is a good practice to chose wide published ones, then the script can have a larger database of images to be influenced). To change/add the artists use always the following notation `["insertthere", "insertthere", "insertthere"]`. There is no limit of visual references, it is also possible to write `"vray", "render", "sci-fi", "modernism"`... instead of famous persons. The possibilities are diverse.

2 - `prompt_main = ["impossible cosmic space bridge"]` # this is the concept the script will generate. This could be an infinite variety o themes, concepts and topics. Examples: (using emotions and time inputs) `prompt_main = ["sad garden in soviet future"]` // (using several words to achieve different result) `prompt_main = ["beautiful architecture", "modernism utopic", "arctic base", "sci-fi looking"]`



## Results from example

```
prompt_pre = ["Zaha hadid", "Frei Otto", "Niemeyer"]  
prompt_main = ["impossible cosmic space bridge"] imagesize = 350  
steps = 500
```

```
learning_rate = 0.02 #1.80 #0.30 #0.40  
lr_min = 0.04  
lr_decay_steps = 25  
weight_decay = 0.30 #0.20 #0.15 #0.20
```

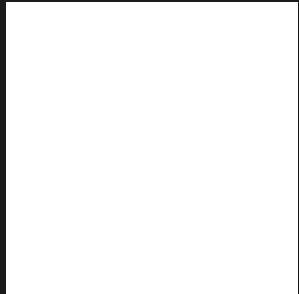
#visual references

#concept

#number of iterations (every 50 iterations is a new frame. To fast tests is better to stay between 200 - 500. Above 500 wil take longer, but the results can become really interesting, since it will get smarter with time)

#learning rate (the higher, it will make changes faster)

base image



final result - step 500 (first time running)



final result - step 500 (second time running)





## Results from example

```
prompt_pre = ["Zaha hadid", "Frei Otto", "Niemeyer"]  
prompt_main = ["impossible cosmic space bridge"] imagesize = 350  
steps = 500
```

```
learning_rate = 0.02 #1.80 #0.30 #0.40  
lr_min = 0.04  
lr_decay_steps = 25  
weight_decay = 0.30 #0.20 #0.15 #0.20
```

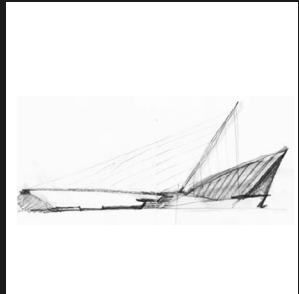
#visual references

#concept

#number of iterations (every 50 iterations is a new frame. To fast tests is better to stay between 200 - 500. Above 500 will take longer, but the results can become really interesting, since it will get smarter with time)

#learning rate (the higher, it will make changes faster)

base image



first iterations - step 50 (first time running)



final result - step 500 (second time running)



## STEP 4 - Observations

1 - The script works in two steps. First it translates the meaning of the concept and then identifies images corresponding to the visual reference (CLIP). Then the last part interprets it visually by crossing the meaning with the reference published works found in online databases, creating loopings to validate it gradually (VQGan).

2 - In this sense, the script can produce results highly unpredictable/exploratory, since the way CLIP works is partially a blackbox, where the meanings and databases used can change depending on each parameter. This script also is particularly good in doing scary/uncanny images - like when it does human/animals.

3 - Another possible interesting approaches can be:

- Using visual artists as references to render narratives and universes
- Using architects to render complex human feelings
- Explore impossible architectures and urban forms (like, floating buildings in a lava world)
- Exploring how using the words "view from above", "view from far", "view from inside" etc.. in the concept can change the camera angle
  - Write down colors to visualize how they are gonna be applied
- Visualize utopic and conceptual visions made by architects (like the invisible cities from Italo Calvino)
- Visualize the limitations of vague concepts as sustainability (usually, it starts to render grass, animals and wood everywhere)
- Use very abstract sketches to see unpredictable results
- Render realistic patterns (as metal chrome patterns, etc..)
- Visualize end of the world scenarios (cold war, far-right apocalypse, world collapse, usually renders very impressive results, which can be potentially disturbing).