Latent Space Exploration

```
# Mount G-Drive
from google.colab import drive
drive.mount('/content/drive', force_remount=True)
```

```
Mounted at /content/drive
```

Clone StyleGAN2 ADA PyTorch from GitHub.

```
!git clone https://github.com/NVlabs/stylegan2-ada-pytorch.git
!pip install ninja
```

```
Cloning into 'stylegan2-ada-pytorch'...
remote: Enumerating objects: 125, done.
remote: Total 125 (delta 0), reused 0 (delta 0), pack-reused 125
Receiving objects: 100% (125/125), 1.12 MiB | 3.36 MiB/s, done.
Resolving deltas: 100% (55/55), done.
Collecting ninja
  Downloading ninja-1.10.2.2-py2.py3-none-manylinux_2_5_x86_64.manylinux1_x86_64
     |████████████████████████████████| 108 kB 8.1 MB/s
Installing collected packages: ninja
Successfully installed ninja-1.10.2.2
```

Verify that StyleGAN has been cloned.

```
!ls /content/stylegan2-ada-pytorch/
```

```
calc_metrics.py  docker_run.sh  LICENSE.txt   style_mixing.py
dataset_tool.py  docs           metrics       torch_utils
dnnlib           generate.py    projector.py  training
Dockerfile       legacy.py      README.md     train.py
```

Interpolation video

Run StyleGAN2 From Python Code

The code below is based on code from NVIDIA. This will generate the images.

```
import sys
sys.path.insert(0, "/content/stylegan2-ada-pytorch")
import pickle
import os
import numpy as np
```

```
import PIL.Image
from IPython.display import Image
import matplotlib.pyplot as plt
import IPython.display
import torch
import dnnlib
import legacy

def seed2vec(G, seed):
  return np.random.RandomState(seed).randn(1, G.z_dim)

def display_image(image):
  plt.axis('off')
  plt.imshow(image)
  plt.show()

def generate_image(G, z, truncation_psi):
    # Render images for dlatents initialized from random seeds.
    Gs_kwargs = {
        'output_transform': dict(func=tflib.convert_images_to_uint8, nchw_to_nhwc=True
        'randomize_noise': False
    }
    if truncation_psi is not None:
        Gs_kwargs['truncation_psi'] = truncation_psi

    label = np.zeros([1] + G.input_shapes[1][1:])
    images = G.run(z, label, **G_kwargs) # [minibatch, height, width, channel]
    return images[0]

def get_label(G, device, class_idx):
  label = torch.zeros([1, G.c_dim], device=device)
  if G.c_dim != 0:
      if class_idx is None:
          ctx.fail('Must specify class label with --class when using a conditional net
      label[:, class_idx] = 1
  else:
      if class_idx is not None:
          print ('warn: --class=lbl ignored when running on an unconditional network')
  return label

def generate_image(device, G, z, truncation_psi=1.0, noise_mode='const', class_idx=Non
  z = torch.from_numpy(z).to(device)
  label = get_label(G, device, class_idx)
  img = G(z, label, truncation_psi=truncation_psi, noise_mode=noise_mode)
  img = (img.permute(0, 2, 3, 1) * 127.5 + 128).clamp(0, 255).to(torch.uint8)
  #PIL.Image.fromarray(img[0].cpu().numpy(), 'RGB').save(f'{outdir}/seed{seed:04d}.png
  return PIL.Image.fromarray(img[0].cpu().numpy(), 'RGB')


URL = "/content/drive/MyDrive/malliGAN/results/00008-food-eheitner1024-mirror-auto1-ad

print(f'Loading networks from "{URL}"...')
```

```
device = torch.device('cuda')
with dnnlib.util.open_url(URL) as f:
    G = legacy.load_network_pkl(f)['G_ema'].to(device) # type: ignore
```

```
Loading networks from "/content/drive/MyDrive/malliGAN/results/00008-food-eheitn
```

Preview ranges of fakes to choose subjects for interpolation video

```
# Choose starting and ending seed.
SEED_FROM = 70000
SEED_TO = 70040

# Generate the images for the seeds.
for i in range(SEED_FROM, SEED_TO):
  print(f"Seed {i}")
  z = seed2vec(G, i)
  img = generate_image(device, G, z)
  display_image(img)
```

Seed 70000

Seed 70001

Seed 70002

Seed 70003

Seed 70004

Seed 70005

Seed 70006

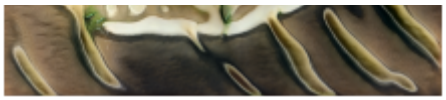
Seed 70007

Seed 70008

Seed 70009

Seed 70010

Seed 70011



Seed 70012

Seed 70013

Seed 70014

Seed 70015

Seed 70016

Seed 70017

Seed 70018

Seed 70019

Seed 70020

Seed 70021

Seed 70022

Seed 70023

Seed 70024

Seed 70025

Seed 70026

Seed 70027

Seed 70028

Seed 70029

Seed 70030

Seed 70031

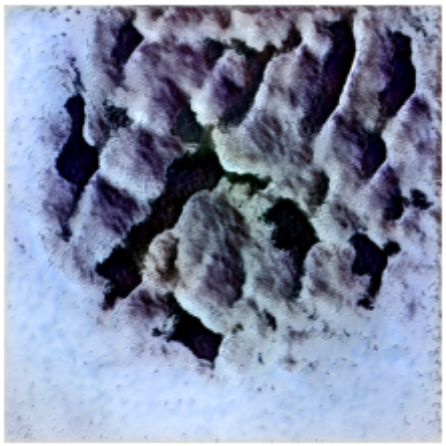
Seed 70032

Seed 70033

Seed 70034

Seed 70035

Seed 70036



Seed 70037

Preview list of chosen fakes for interpolation video

```
# Choose list of seeds
SEED_LIST = [8025, 3006, 700, 208, 201, 107, 2018, 607, 70024]

# Generate the images for the seeds.
for i in SEED_LIST:
  print(f"Seed {i}")
  z = seed2vec(G, i)
  img = generate_image(device, G, z)
  display_image(img)
```