Art of Panda

Photo Mosaic

Zhidi Zhang
COMP 572

Agenda

- Project Goal
- Image Preparation
  - Stable Diffusion Model
  - Image generation
  - Example of Tile Images
- Photo Mosaic Algorithm Implementation
- Tuning and Output

Project Goal

Automatically create a photo mosaic with given photo

- Build an image library with google's stable diffusion model v1.5
- Implement photo mosaic algorithm in Matlab

Tech Stack: Python, Matlab

Stable Diffusion Model

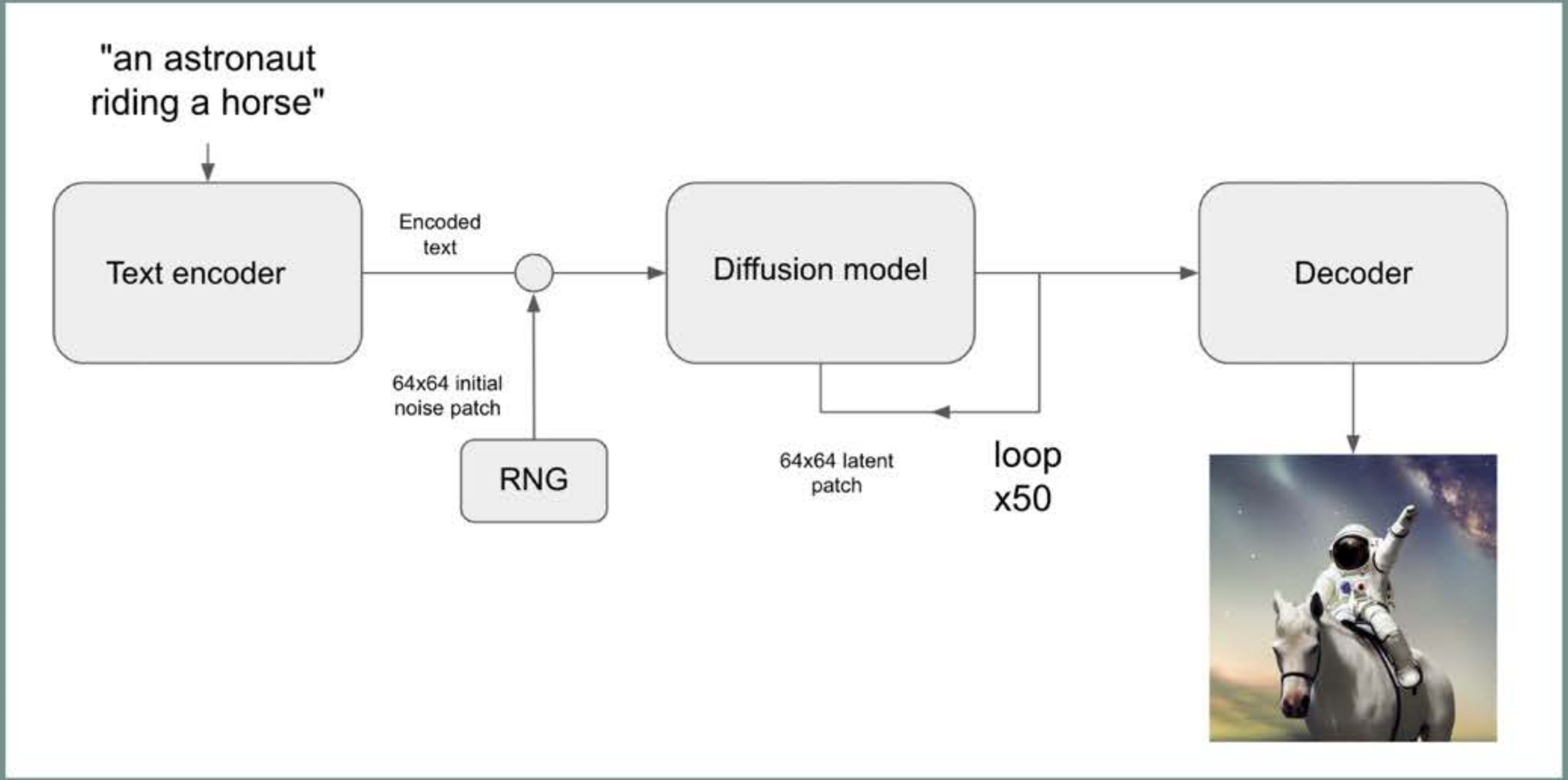


Image Source: https://keras.io/examples/generative/random_walks_with_stable_diffusion/

Build Image Library

Generate prompt message with 'Panda by (a famous artist)'

Input the prompt message to stable diffusion model

Artist Name:

Vincent van Gogh

Claude Monet

Pablo Picasso

Sandro Botticelli

Leonardo da Vinci

.....

Generated 5000 images, and saved them on my Github

Three days hardwork of my poor GPU...


```
PLMS Sampler:  92%|          | 154/167 [00:24<00:01,  6.84it/s][A[A
PLMS Sampler:  93%|          | 155/167 [00:24<00:01,  6.82it/s][A[A
PLMS Sampler:  93%|          | 156/167 [00:25<00:01,  6.84it/s][A[A
PLMS Sampler:  94%|          | 157/167 [00:25<00:01,  6.76it/s][A[A
PLMS Sampler:  95%|          | 158/167 [00:25<00:01,  6.80it/s][A[A
PLMS Sampler:  95%|          | 159/167 [00:25<00:01,  6.90it/s][A[A
PLMS Sampler:  96%|          | 160/167 [00:25<00:01,  6.79it/s][A[A
PLMS Sampler:  96%|          | 161/167 [00:25<00:00,  6.78it/s][A[A
PLMS Sampler:  97%|          | 162/167 [00:25<00:00,  6.77it/s][A[A
PLMS Sampler:  98%|          | 163/167 [00:26<00:00,  6.78it/s][A[A
PLMS Sampler:  98%|          | 164/167 [00:26<00:00,  6.74it/s][A[A
PLMS Sampler:  99%|          | 165/167 [00:26<00:00,  6.83it/s][A[A
PLMS Sampler:  99%|          | 166/167 [00:26<00:00,  6.81it/s][A[A
PLMS Sampler: 100%|          | 167/167 [00:26<00:00,  6.84it/s][A[A
PLMS Sampler: 100%|          | 167/167 [00:26<00:00,  6.27it/s]
data: 100%|          | 1/1 [00:31<00:00, 31.50s/it][A
data: 100%|          | 1/1 [00:31<00:00, 31.50s/it]
Sampling: 100%|          | 1/1 [00:31<00:00, 31.50s/it]
Sampling: 100%|          | 1/1 [00:31<00:00, 31.50s/it]
Your samples are ready and waiting for you here:
outputs/txt2img-samples/4999_René Magritte

Enjoy.
```

Inspiration by Utagawa Hiroshige

Inspiration by Vincent van Gogh

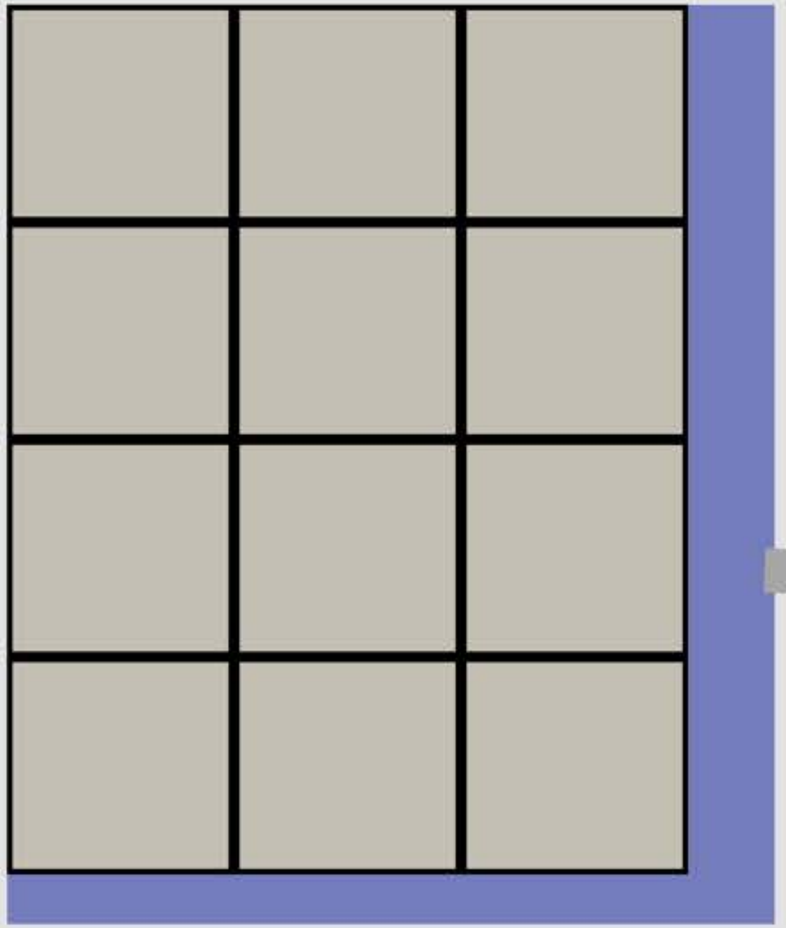
Inspiration by Claude Monet

Other Examples of Synthetic Images

Brief Explanation: Photo Mosaic Algorithm

- Divided the base image into thousands of grid
- Find the average RGB value of each grid
- Search the RGB value from image library that close to average RGB of each grid
- Replace the grid of base image with one of the closest tile images that were randomly selected to avoid the repetitions

Divided to grids and remove uncovered boundaries



Because the grid can not perfectly occupy the entire image, I used a logic to remove the uncovered boundary for the better result.

Save the average RGB value of all 5,000 tile images

Calculating the average RGB values

Calculate the average color for every tile and save it in a **KD tree** instead of an array.

- Challenge: Use an array to save the RGB values of tile images, searching will cost O(n) time complexity;
  - # of grids: 10 * 10, # of tiles: 10 -> 1.5 mins to create a photo mosaic;
  - # of grids: 100 * 100, # of tiles: 5000 -> roughly 7500 mins (125 hours, five days) to create a photo mosaic, almost infeasible in practice
- Solution: Use KD tree, searching time complexity will reduce to O(log(n)). It took around 1 hour to generate photomosaic (with # of grids 150 * 150, # of tiles 4500)

Find the closest image and draw the tiles

I utilized Euclidean distance as a similarity metric to measure the distance between average RGB value of grid and tile image

$$d = \sqrt{(R_1 - R_2)^2 + (G_1 - G_2)^2 + (B_1 - B_2)^2}$$

To find the closest distance between two images, I used the KNN search and kept the top 3 closest tile images

Then using the 'randi' function to generate a random integer between 1 and 3, and ***used that random number as an index to pick one tile image from the top 3 closest results to reduce the repetition***

Handle the 'Rick Roll' image generated by diffusion model

- Reason: If Stable Diffusion generates a Rick Roll image, it means that the NSFW filter flags your image

- Solution:
  - Find the the average RGB value of this image is (128, 122, 131)
  - If any image match this average RGB value, reset it to be (INF, INF, INF)
  - So this image will never show up in top k closest result

Tuning Parameters

Base image

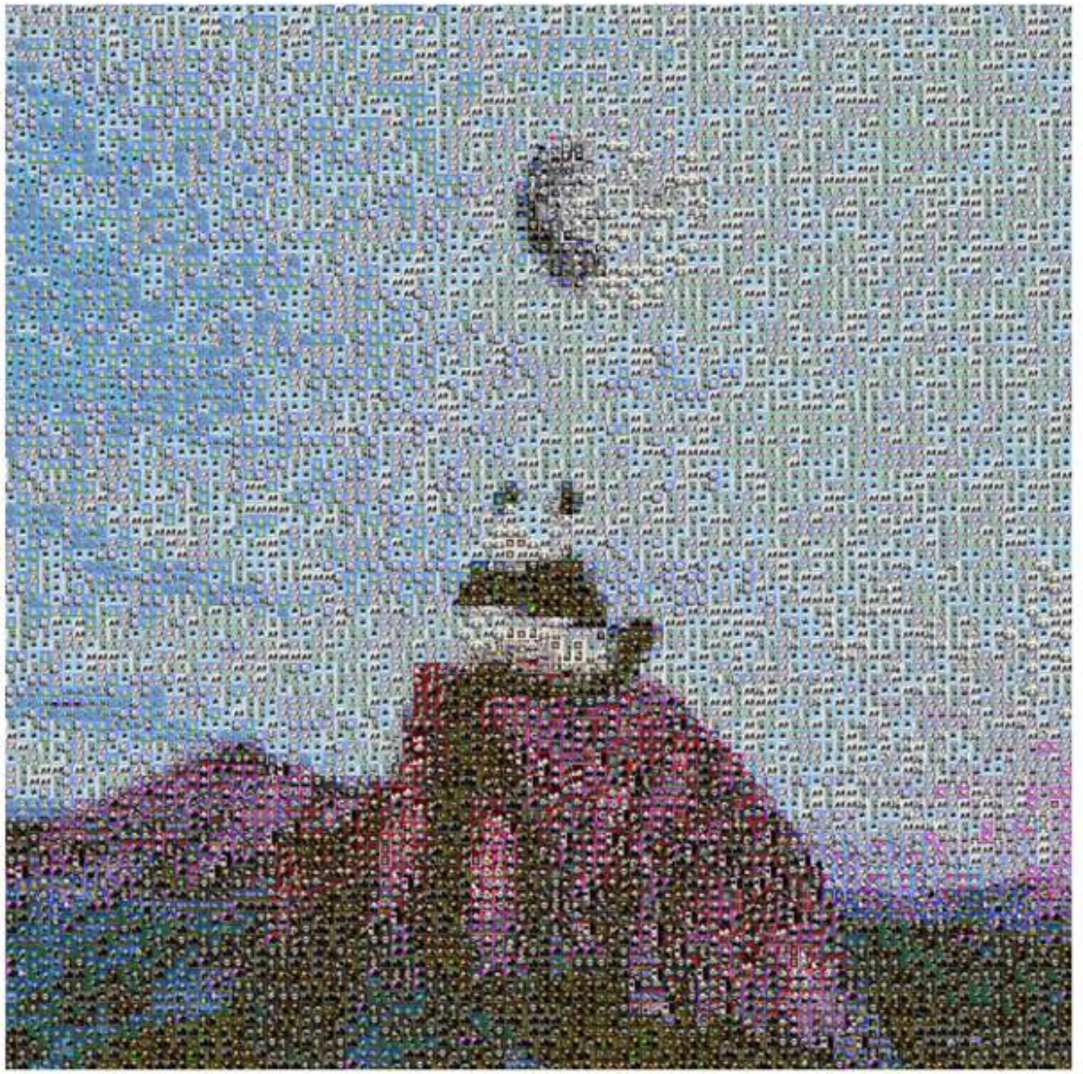
# of grids: 10 * 10
# of tiles: 500

# of grids: 100 * 100
# of tiles: 3000

# of grids: 200 * 200
# of tiles: 4500

Source: https://br.pinterest.com/pin/819795938408180286/

# of grids: 100 * 100
# of tiles: 4500



Source: https://co.pinterest.com/pin/654640495824543716/

# of grids: 100 * 100
# of tiles: 4500

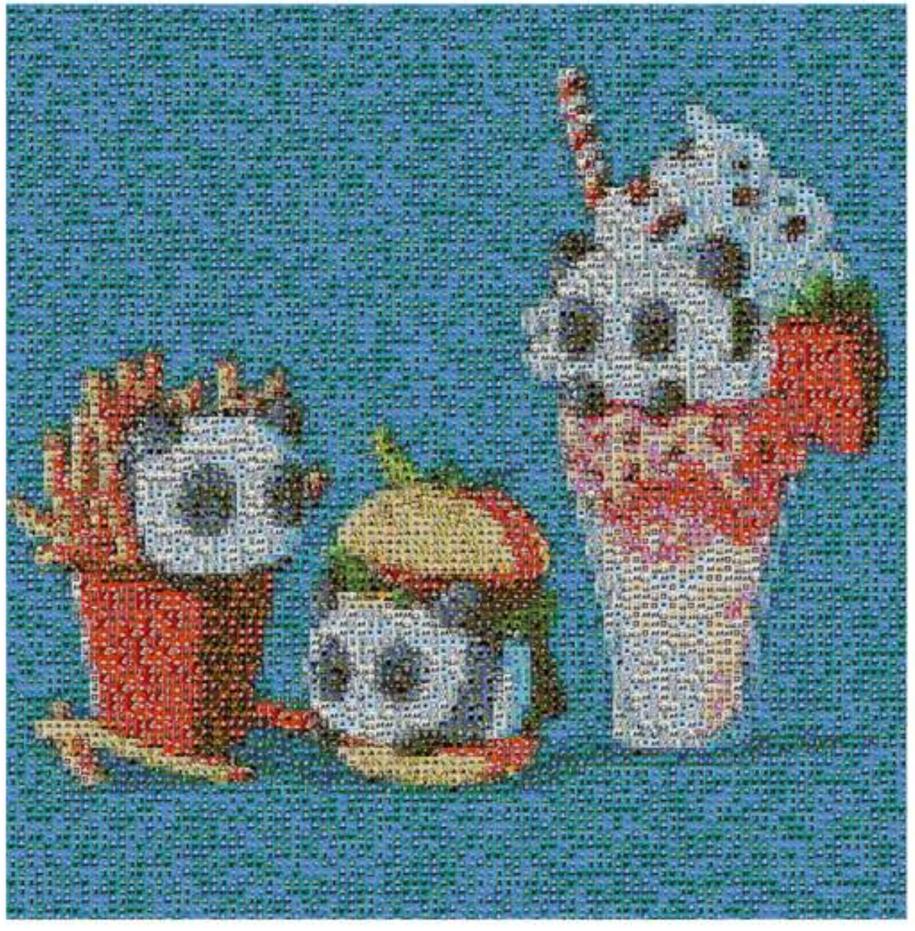
Source: https://pin.it/3z9kgLe

# of grids: 150 * 150
# of tiles: 4500

# of grids: 120 * 120
# of tiles: 4500

Source: https://www.pinterest.es/pin/847099011141351903/

Source: https://www.pinterest.es/pin/847099011141351899/



# of grids: 100 * 100
# of tiles: 4500

Source: https://www.pinterest.com/pin/299489443975453685/