# CSCI 1430 Final Project Report:
# StyleGAN Image Morphing using Autoencoder Networks

*Animorphs*: Kenta Yoshii, Naveen Sharma, Yuki Hayashita.
Brown University

## Abstract

*Generative models are used to create new images from scratch, by training a neural network on a large dataset of images. By using these models, computer scientists have been able to explore the upper limits of computer-generated art. In this paper, we explore one such creative use case of generative models: image morphing. The main purpose of this project was to explore the possibilities of image morphing, and compare results across numerous architectures and datasets. As a result, we have learned a great deal about autoencoders, StyleGAN, and generative models, and were able to produce some interesting results. Specifically, we experimented with both pretrained and custom designed encoders, the StyleGAN generative model, and four contrasting datasets (human faces, artwork, cars, cats).*

## 1. Introduction

This project aims to explore the creative use of generative models, specifically with image morphing. Through our prior research, we have found numerous architectures and datasets which can be used to accomplish this task. The goal we wish to accomplish in this project is to compare and contrast the variety of morphing techniques use, in order to determine which methods lead to the best results. Through this analysis, we hope to gain insight into generative models themselves, as well as learn how they perform on different types of data. Learning this new information may prove useful in future image morphing projects, and may be generalizable to a broader range of computer vision tasks.

## 2. Related Work

**StyleGAN**: Karras et al. [1]:

Developed by NVIDIA, StyleGAN is one of the leading generative models for human faces. Unlike traditional GAN models, where latent vectors are derived from a feed forward network, StyleGAN produces latent vectors using two separate networks: the mapping network and the synthesis network. The mapping network is an 8-layer perceptron. The output from the mapping network is integrated at each point in the generator model via a new layer called adaptive instance normalization(AdaIN), defined as:

$$AdaIN(x_i, y) = y_{s,i}\frac{x_i - \mu(x_i)}{\sigma(x_i)} + y_{b,i}$$

where $y_s, y_b$ are styles learned in the affine transformation. The use of this style vector gives control over the style of the generated image. The AdaIN layers involve first standardizing the output of feature map to a standard Gaussian, then adding the style vector as a bias term. This unique architecture allows us to have a control over the style of generated images at different levels of detail.

**Encoder4Editing**: Tov et al. [2]:

In this paper, Tov et al introduce a method for producing latent vectors for images in the StyleGAN latent space. The model uses an encoder paired with a pretrained StyleGAN to learn encoder weights. The result is a network that allows us to pass in custom images, and obtain their corresponding latent vectors.

## 3. Method

In this project, 4 different datasets were used. Morphing was achieved on each dataset. In each section, the method of how morphing results were obtained will be detailed.

1. **Human Faces**

   To begin, we created a Google Colab to run our morphing experiments. We decided to use this platform for its ease of use and ability to run code on Google's servers. All code in this project was written in Python. We first downloaded and extracted the pretrained StyleGAN model for human faces. With this model, latent vectors could be passed in and generated images would be produced. To obtain latent vectors for custom images, we tried two approaches. First, we used

MSE loss to approximate an image's latent vector. This involved initializing a random vector, passing it through the generator of StyleGAN, comparing the initial and final images with MSE, and updating the latent vector to reduce loss. A second approach was using the e4e model as described in [2]. The results of these initial experiments are detailed in the Results section.

To achieve morphing, we utilized a technique called linear interpolation. Given two latent vectors $w_1$ and $w_2$, corresponding to the starting and ending images respectively, we obtain a combined vector $w$ as follows:

$$\lambda w_1 + (1 - \lambda) w_2 = w \tag{1}$$

The $\lambda$ value determines how close the resulting image should be to the ending image (0 is the closest, 1 is the furthest). In our project, we created several $w$ vectors using a range of $\lambda$ values. Specifically, we produced vectors for every $\lambda$ multiple of 0.05 between 0 and 1, for a total of 20 intermediary vectors. After passing each vector into the StyleGAN generator, we produced 20 intermediary images. By stitching these image together using the Pillow libary, GIFs could be created which show smooth morphing from the starting image to the ending image.

Since e4e was used with this dataset, we also ran experiments editing certain features of the images. In the Results section, morphing results can be seen moving in the 'pose' direction, the 'age' direction, and finally both 'pose' and 'age' directions simultaneously.

2. **Artwork**

The dataset used for this portion of the project was the WikiArt Painter by Numbers dataset (Linked here). This dataset included 23817 images from a variety of artists. In contrast to the previous dataset, we did not have a pretrained generator to work with. So, our main goal in this section of the project was to train our own generative model. For this model, we used the StyleGAN architecture. In the StyleGAN official Tensorflow implementation, steps are outlined on how to utilize the architecture for custom training. Following these steps, which involved modifying the typical StyleGAN training regimen, and loading in our new dataset, we began training our WikiArt generative model. For this, we used GCP and trained the model using 4 GPUs. Sadly, after 3 days of training, the GCP console crashed and interrupted the training process. Fortunately, we were able to load a checkpoint saved during the training that had been completed.

Using our partially-trained generative model, we used the same techniques as described in the Human Faces section to achieve morphing. Since we did not have a pretrained encoder trained on artwork, we used the naive latent vector approximation approach described in the previous section to map images to the generator's latent space. Using linear interpolation, morphing was achieved. Our results from this section, including randomly generated images and morphing results are shown in the Results section.

3. **Cars**

Upon further research into StyleGAN and its uses, we discovered that StyleGAN has been pretrained on far more tasks than just human face generation. One particular pretrained StyleGAN model that stood out to us was the car model. Since most cars are relatively similar in shape, we figured morphing might work especially well with this dataset. Downloading and extracting the car StyleGAN model, we were able to achieve similar results to the previous two sections. One difference with this dataset is that the images were of a smaller resolution. For the human face data, 1024x1024 images were used. In the car dataset, 256x256 images were used. After modifying our existing architecure to accomodate for the smaller image size, we obtained morphing results. Similarly to the WikiArt dataset, since we did not have an encoder which could produce latent vectors given images, we had to use randomly generated images for our start and end points for morphing.

4. **Cats**

With the cat dataset, we wanted to run experiments on the MSE loss latent vector approximation approach to morphing. To do so, we ran our algorithm on cat images for 500, 1000, and 1500 steps each. After producing the latent vectors and passing them through StyleGAN (shown in Results section), we were able to achieve morphing. The effectiveness of morphing varied immensely with the changes in training steps. This will be discussed further later in the paper.

## 4. Results

Below, the results for each datasets will be displayed. Since we are not able to show our morphing GIFs in this format, they can be accessed online at this link.

1. **Human Faces**

Below, we have the results from our naive MSE loss encoder program. Note that the images are not vertically aligned with their generated counterparts. The correct pairings are (0, 3), (1, 0), (2, 4), (3, 1), (4, 2).

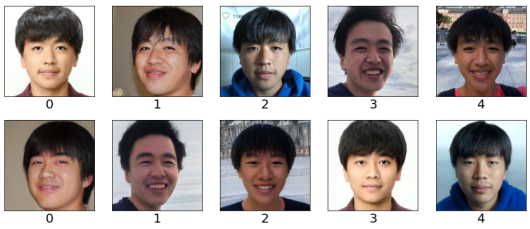


Figure 1. Naive latent vector approximation

While these results are good, the faces are clearly altered from the original images. Using e4e, we were able to produce much better results.

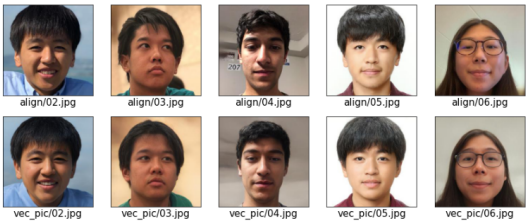


Figure 2. e4e latent vector approximation

For some of these images, the original and generated images are nearly indistinguishable.

Finally, below are some of the morphing results achieved with this dataset:

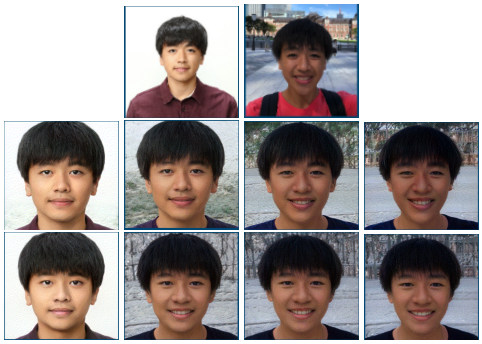

Figure 3. Morphing with human face dataset. (From top to bottom: start/end images, naive, e4e)

2. **Artwork**

Using our trained artwork generative model, here are some examples of artwork the model produced (given random latent vectors):

Figure 4. Generated Art using WikiArt model

With our naive latent vector approximation, here are some of the reconstructions produced by the generator. As seen with Hokusai's The Great Wave (image 2 in the top row), the generator does a decent job at recreating the artwork. However, with the other examples, the results are not as promising. Comparing this to the results of the naive latent vector approximation done on the human face dataset, we can see that the results achieved with human faces were far superior. Perhaps this implies that human faces are easier to represent in a latent space than artwork. Also, these sub-optimal results could be the product of our interrupted training of the generator.


Figure 5. Generated Art using WikiArt model and Naive latent vector approximation

Finally, below are some of the morphing results achieved with this dataset:

Figure 6. Morphing with WikiArt dataset

3. **Cars**

Using the pretrained StyleGAN on cars and linear interpolation, below is the result of morphing between two randomly generated cars.

4. **Cats**

Figure 7. Morphing with car dataset

Using the naive MSE loss latent vector approximation approach, below are the results for 500, 1000, and 1500 steps.

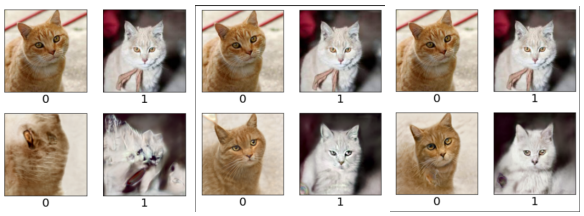

Figure 8. Cat dataset experiments (Left: 500 steps, Middle: 1000 steps, Right: 1500 steps)

### 4.1. Technical Discussion

Our experiments with image generation and morphing led to some interesting and visually appealing results. Perhaps the most intriguing aspect of the project was to see which methods worked better than others and determine why this was the case. Our team agrees that the best morphing results were achieved using the e4e encoder and StyleGAN generator on the human face dataset. Our main criteria for success was that every intermediate image look like a natural face on its own, and this was definitely the case with this approach (see Fig. 3).

On every dataset, we felt as though e4e generated better results than the naive MSE loss approach. This was as expected. We estimate that the more complex encoder architecture of e4e was responsible for the improvement.

In the experiments for the cat dataset, we see that the results for 1500 iterations are significantly better than those for 500 iterations. At 500 iterations, the reconstructed image barely looks like a cat, while at 1500 iterations, there are only small differences between the original and generated images (for instance the pink ribbon on the white cat;s neck). In the morphing results, (accessible on our README linked above), we see that this improvement is noticeable when morphing as well. This experiment proved to us the importance of training steps when approximating latent vectors using this approach. While a higher number of iterations takes longer to compute, we feel as though the benefit in results in worth it.

As for the question of which datasets worked the best with image morphing, we agree that the human face and car datasets worked especially well, where as the WikiArt and cat dataset did not. We believe this can be explained by the difference in image variation for each dataset. With categories like human faces and cars, each image will have the same components in roughly the same places. For instance, every face in the dataset has two eyes, a nose, and a mouth. There is very little variation in fact, with the human face dataset, (given that each face is centered and aligned) compared to some other categories. The WikiArt dataset contained artwork that were all completely different than one another. Since art has no constraints, like cars and faces, the variation in the dataset was far greater. Because of this, we felt as though the performance of our models on this dataset were weaker. Some of the intermediary images did not appear as convincing. In addition, when generating random artwork, we felt that portraits in particular came out slightly distorted (see Fig. 4).

### 4.2. Societal Discussion

Response to 5 critiques:

1. In our initial proposal, we highlighted a concern that generative art may someday reduce the need for real artists. The critique we received was that no matter the progress of computer generated art, there will always be a need for human artists. After seeing the results produced by our model (especially those on the WikiArt dataset), we came to agree with this critique. While the artwork created is convincing, it is very difficult for a computer to create art which captures deeper meaning than just the picture itself.

2. Another point brought up by the reviewers was that if computer generated art became more popular, it would favor those with better computational resources. We agree with this comment as well. From our results, we can clearly see that more compute leads to better results (ex. e4e and StyleGAN). To help prevent this, perhaps artists could release their pre-trained models on platforms like Google Colab, where all the code runs on Google machines.

3. For face morphing, a concern brought up was that it our training data was not sufficiently diverse, that our model would not perform in the same way for different groups of people. This is definitely a valid concern.

For our face morphing StyleGAN, the dataset used was FFHQ, a face dataset with images collected from Flickr. One issue with this is that the biases that were prevalent within Flickr were carried over into the FFHQ dataset, and thus are present in our project as well. In the future, if we were to improve upon this project, one important step would be to gather a more diverse and unbiased face dataset.

4. A question brought up in the critique was about who would own the rights to digitally created art. We believe that the creator of the code as well as the creator of the training dataset should have the rights to the artwork.

5. Another comment brought up was that this project could potentially have applications in the NFT space. We agree that generative models can definitely be used to create NFTs. A generative model could be used to create a large number of NFTs with relatively little effort, so it could be quite possible that generated NFTs take over the space.

## 5. Conclusion

This paper explored several techniques for morphing images together. Through several experiments on four different datasets, we explored latent vector approximations, autoencoders, the StyleGAN generative model, and linear interpolation. After achieving image morphing with both pretrained networks and custom trained networks, we were able to compare and contrast the different approaches and datasets. We determined that the e4e and StyleGAN architecture produced the best morphing results. We also learned that the variation in image datasets plays a crucial role in the success of generative models and latent vector approximations, and as a result affects image morphing. In the future, with the new knowledge gained from this project, perhaps ourselves or readers could train new image morphing models which improve upon the ones detailed in this paper.

## References


[1] Karras et al. A style-based generator architecture for generative adversarial networks, 2019. 1

[2] Tov et al. Designing an encoder for stylegan image manipulation, 2021. 1, 2