Using Artificial Intelligence to create Digital Avatars

COMP4971C Independent Work Final Report (Fall 2022)

By PANG, Lok Chi

Supervised by Dr. David Rossiter

Abstract

This project explores the use of artificial intelligence to create digital avatars, which are graphical, virtual representations of humans used in digital entertainment or social networking.

The project involved an experiment in creating a generative adversarial network to generate digital images of virtual characters and an implementation of a automated pipeline to create a short animation, depicting a talking human, using an image generated using Stable Diffusion and an audio file.

In addition, this report will discuss the background, development, and applications of related technologies.

Contents

Figures

Tables

Introduction

Digital avatars are simply a graphical representation of a human being in a virtual world. They can refer to a wide variety of things, from user icons in traditional social media and forums, to 3D versions that represent users in video games.

While digital avatars have been in use in various forms since the dawn of digital entertainment, they are becoming more and more complex and seeing applications in an increasing number of areas in recent years.

Currently, the creation of digital avatars must be undertaken by professional artists in a laborious process. In high quality video games, player avatars are full 3D models created by 3D artists. In other products, including social media such as Snapchat, avatars are “assembled” by users from existing assets such hair, eyes, faces, etc. Still, a certain degree of manual labor is required to create art assets.

With the rise of AI art recently, the possibility of automating the creation of digital avatars has become very real.

Digital Avatars

The term “Digital Avatar” can refer to a variety of things. In general, as long as an image (or other form of multimedia) represents a person, whether in the form of a direct graphical representation of said person’s likeness, or in more abstract forms such as icons, it may be referred to as a “Digital Avatar”. Thus, it can be anything from a small image (e.g. a 32x32 .png file) to a 3D model.

For the purpose of this project, let us define Digital Avatar as a multimedia item is anthropomorphic in appearance, and can be manipulated at will to exhibit human behavior such as facial expression, speech, movement, etc.

AI in Image Generation

A variety of models have been developed for image generation. This section will give a brief introduction into two mainstream models, the generative adversarial model, and the diffusion model.

Generative Adversarial Model

The Generative Adversarial Model, pioneered by Ian Goodfellow et al. in 2014, is a framework in which two neural networks compete against each other with the goal of generating new data that resembles the training set. Applications of GANs include image generation, text-to-image translation, super-resolution, etc.

The two neural networks that comprise a GAN are the generator and the discriminator. In each iteration, the generator synthesizes new data instances which are sent to the discriminator for evaluation. The discriminator, in addition to evaluating the synthesized data, is at the same time being trained on the training set, which consists of authentic data instances. The goal of the generator is to create new data that "fools" the discriminator, while the goal of the discriminator is to correctly identify the generator data as synthesized. (Brownlee, 2019)

StyleGAN is a well-known GAN used for image synthesis.

Diffusion Model

Diffusion Models add random Gaussian noise to a sample through a Markov chain until it becomes pure noise. Then, it learns to reverse the process to denoise the sample.

Therefore, trained diffusion models can generate data from random noise, by denoising it until a clean sample is obtained. (Weng, 2021)

Building a GAN

A Generative Adversarial Network was built using Tensorflow and trained on a dataset consisting of 63,565 illustrations of human faces, obtained from Kaggle.

The model itself consists of a generator that makes use of transposed convolution to upsample random noise until a desired image size is reached, with a CNN image classifier as the discriminator.

The Dataset

The dataset is “Anime Face Dataset”, by Brian Chao. The dataset was created by scrapping www.getchu.com to obtain illustrations and cropping the faces using the face detection algorithm “lbpcascade_animeface”, by nagadomi, which is a Haar cascade classifier. (Chao, 2022)

The sizes of the images in the dataset ranges from 90x90 to 120x120. All images were resized to 128x128 before training.

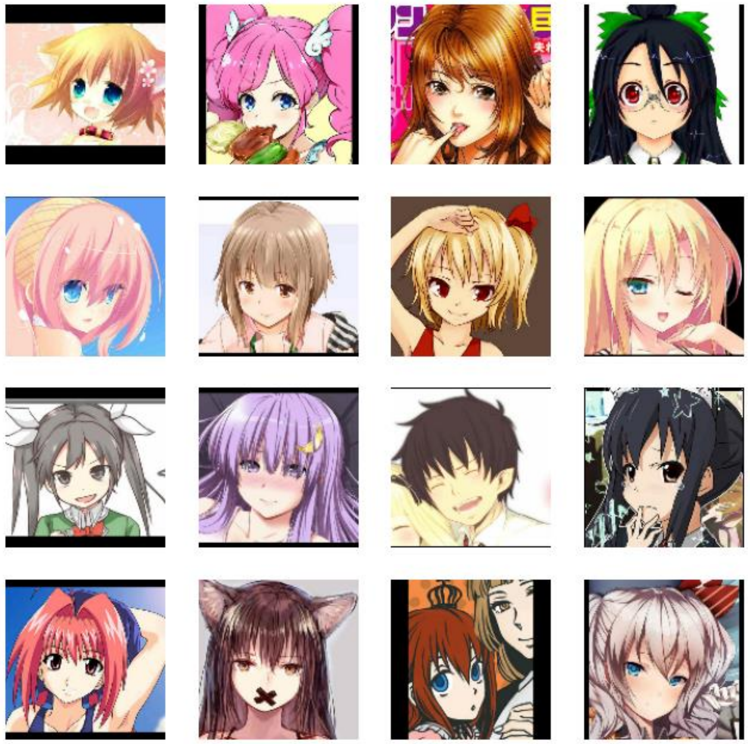


Figure 1 Images from the Dataset

The Model

The generator makes use of transposed convolution to upsample random noise until the desired image size of 128 pixels by 128 pixels is reached.

Transposed convolution is a layer that can upsample the input. As in a normal convolution layer, the input is passed through a filter. However, between each row and column of the input, a number of zeros, equal to the stride, are inserted, before the kernel is passed over the input. As a result, the dimensions of the output are essentially multiplied by a factor equal to the stride (when the padding is zero). (Anwar, 2020)

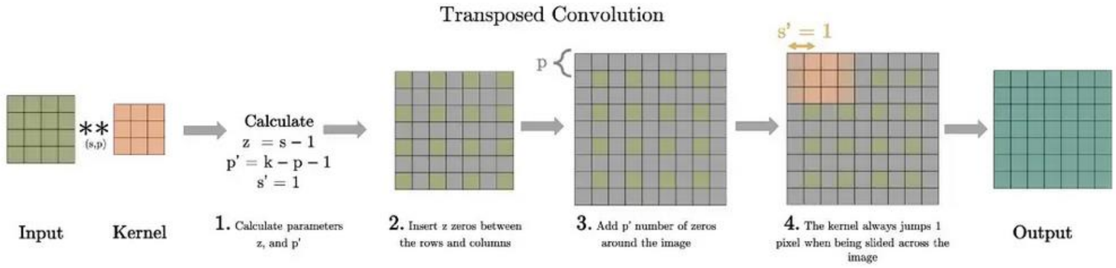

Transposed Convolution
**
(s,p)
Calculate
z = s − 1
p' = k − p − 1
s' = 1
p
s' = 1
Input
Kernel
1. Calculate parameters z, and p'
2. Insert z zeros between the rows and columns
3. Add p' number of zeros around the image
4. The kernel always jumps 1 pixel when being slided across the image
Output

Figure 2 Transposed Convolution (Anwar, 2020)

The input is a seed of 100 random numbers following the standard normal distribution, and the output is a 128x128 image with 3 channels (RGB). To reach the target output size, the model passes the random seed through a dense layer to generate 8x8 images and pass the result through 4 transposed convolution layers with 5x5 filters and stride = 2 to generate the result.

Table 1 Generator Summary

Layer	Output Shape				Number of Parameters
Dense	None	32768			3,276,800
Batch Normalization	None	32768			131,072
Leaky ReLU	None	32768			0
Reshape	None	8	8	512	0
Transposed Convolution, stride = 1	None	8	8	256	3,276,800
Batch Normalization	None	8	8	256	1,024
Leaky ReLU	None	8	8	256	0
Transposed Convolution, stride = 2	None	16	16	128	819,200
Batch Normalization	None	16	16	128	512
Leaky ReLU	None	16	16	128	0
Transposed Convolution, stride = 2	None	32	32	64	204,800
Batch Normalization	None	32	32	64	256
Leaky ReLU	None	32	32	64	0
Transposed Convolution, stride = 2	None	64	64	32	51,200
Batch Normalization	None	64	64	32	128
Leaky ReLU	None	64	64	32	0
Transposed Convolution, stride = 2	None	128	128	3	2,400

The discriminator is a regular CNN image classifier, with 2 convolution layers with 5x5 filters and 2x2 stride.

Table 2 Discriminator Summary

Layer	Output Shape				Number of Parameters
Convolution	None	64	64	64	4,864
Leaky ReLU	None	64	64	64	0
Dropout, 0.3	None	64	64	64	0
Convolution	None	32	32	128	204,928
Leaky ReLU	None	32	32	128	0
Dropout, 0.3	None	32	32	128	0
Flatten	None	131,072			0
Dense	None	1			131,073

Loss and Optimizer

Since both the generator and the discriminator have binary outputs (the binary output of the generator can be considered as true if the discriminator failed to classify the generated image as fake), binary cross-entropy is used as the loss function for both models.

Since the discriminator is being trained on the training set at the same time as it is evaluating the output from the generator, its loss function is the sum of the binary cross-entropy of the two classification tasks.

The Adam optimizer is used for both models.

Evaluation

The model was trained with batch size = 16 for 100 epochs. Some results are as follows:

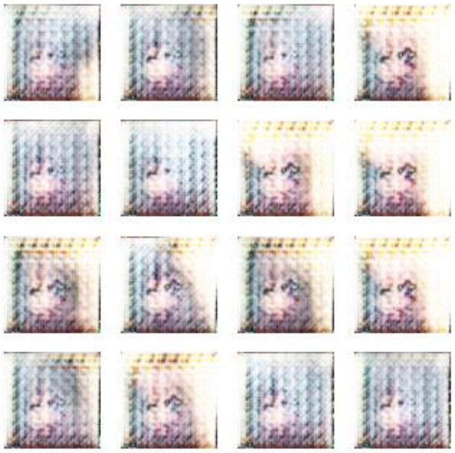


Figure 3 Epoch 1

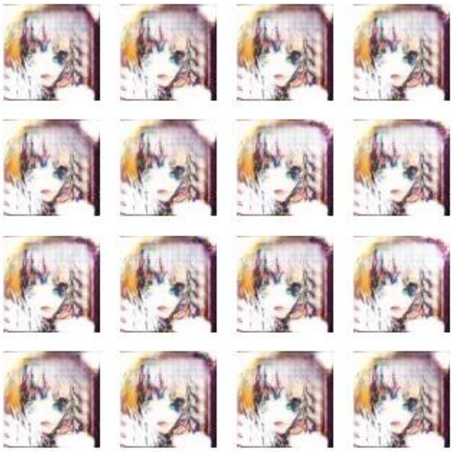


Figure 4 Epoch 54

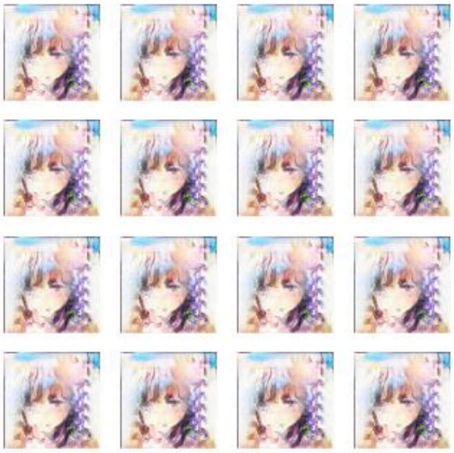


Figure 5 Epoch 81

Although the silhouette of the head and some features such as eyes and mouth started to emerge, at the end of the training, the resulting images were generally blurry and of subpar quality.

The low quality is probably due to the low number of iterations. Comparable projects by others with presentable results usually required at least thousands of iterations.

Due to time limitations, it was decided to proceed to the next part of the project using Stable Diffusion as the image generator.

Facial Animation from Still Image

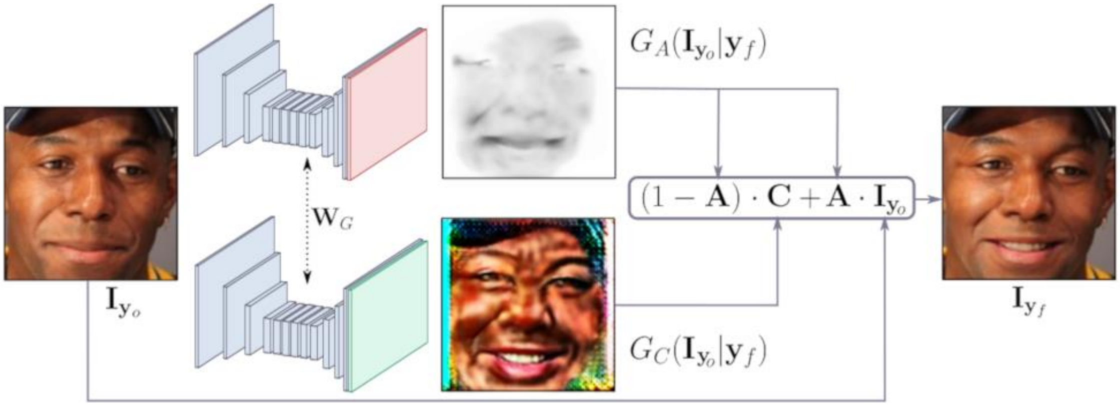
The next stage is to develop a system to animate a still image. An existing implementation by Khungurn P., "talking-head-anime" (THA) was used with slight modifications.

THA takes an input image of a human face and generates a variant with specific facial expression, such as a smile or a wink, or specific mouth positions, such as "a" or "o". It is a 2-part system. The first part is the Face Morpher which handles the facial expression, and the second is the Face Rotator, which rotates the head if needed. (Khungurn, 2019)

Face Morpher

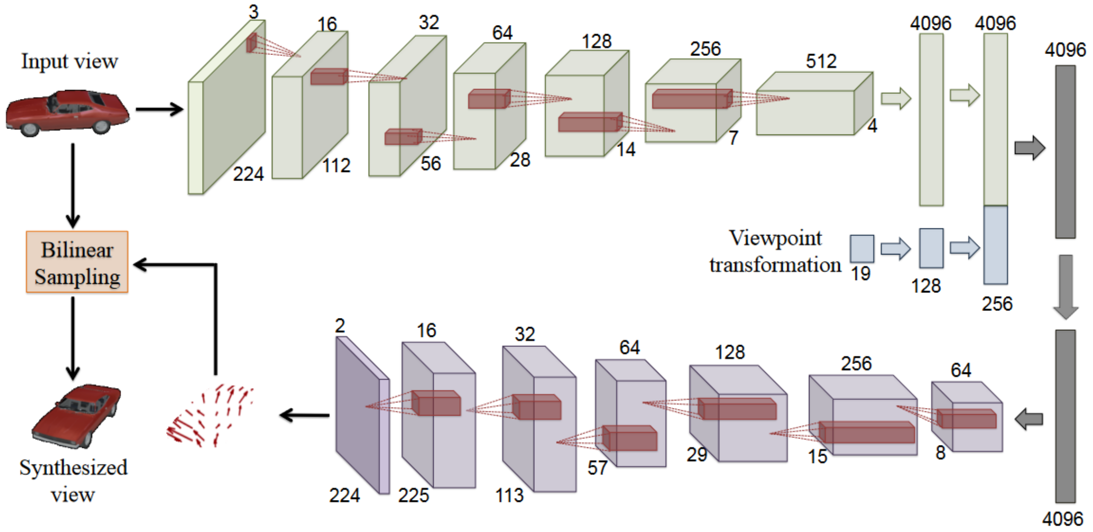
The Face Morpher model is based on a GAN model developed by Pumarola A., et al., described in "GANimation: Anatomically-aware Facial Animation from a Single Image". (Pumarola, Agudo, Martinez, Sanfeliu, & Moreno-Noguer, 2019)

In this model, the generator outputs two masks, a color mask, and an alpha mask. The color mask represents the modified facial expression, while the alpha mask contains information on how much each pixel in the color mask affects the final result. With this implementation, the generator does not need to render static elements such as the background, hair, and accessories.


$G_A(\mathbf{I}_{\mathbf{y}_o}|\mathbf{y}_f)$
$\mathbf{W}_G$
$(1-\mathbf{A})\cdot\mathbf{C}+\mathbf{A}\cdot\mathbf{I}_{\mathbf{y}_o}$
$\mathbf{I}_{\mathbf{y}_o}$
$\mathbf{I}_{\mathbf{y}_f}$
$G_C(\mathbf{I}_{\mathbf{y}_o}|\mathbf{y}_f)$

Face Rotator

The Face Rotator is based on Zhou et al.'s work "View Synthesis by Appearance Flow". It is an encoder-decoder framework in which the input image and the desired viewpoint transformation are encoded by several convolutional and fully connected layers and decoded to compute an "Appearance Flow", which is a field that represents the "movement" of pixels from the input image to the output image. (Zhou, Tulsiani, Sun, Malik, & Efros, 2016)



Training

Khungurn P. trained the model using a custom dataset. The custom dataset was created by posing 3D models of various characters into specific positions and adding facial expressions and rendering the result into images.

In this project, the publicly available weights published by Khungurn P. were used.

Evaluation

Below are some images generated from the THA model. An illustration generated by Stable Diffusion was used as the input, with the background removed using rembg.

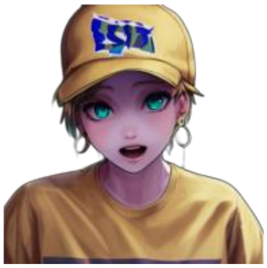


Figure 6 Image generated by Stable Diffusion

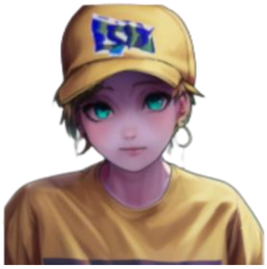


Figure 7 Mouth Closed

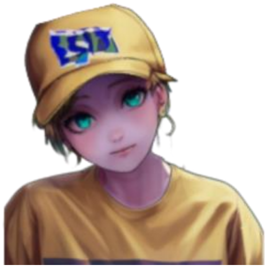


Figure 8 Head Rotated

The model is able to generate variants of the input with different facial expressions with a high degree of realism.

However, there are several limitations. First, the input image must have wide open mouths and eyes. Otherwise, the mouth and/or eye cannot be opened. Second, changing the eyebrow or eyelids may not work, especially said features are partially obscured, for example by a cap.

Lip Movements

Finally, a “talking head” is created by animating lip movements. Using THA, variants of the character with different lip positions can be created.

An audio snippet was analyzed using Rhubarb Lip Sync, a tool that scans an audio file and outputs mouth position changes and timestamps.

Rhubarb supports a total of 6 mouth positions: “a”, “e”, “i”, “o”, “u”, and “rest”. Since THA supports the 5 vowels as well, using the data generated by Rhubarb, the character portrait was animated using THA. (Wolf, 2015)

Animation Process

By changing the multiplier of the pose in THA, realistic animations could be created. For example, by setting the pose to “aaa” and having the multiplier increase from 0 to 1.0, an animation of the character opening their mouth could be created.

First, the total amount of frames is calculated by multiplying the predefined frame rate with the duration of the audio clip. Then, based on Rhubarb’s output, “keyframes” are inserted. The keyframes have multiplier = 1.0 with the corresponding mouth position.

The frames between two keyframes are filled in according to the formula below:

1. If the first keyframe is a “rest”:

   The mouth position is set to that of the second keyframe, the multiplier increases from 0 to 1.0 during the transition.

2. If the second keyframe is a “rest”:

   The mouth position is set to that of the first keyframe, the multiplier decreases from 1.0 to 0 during the transition.

3. Otherwise

   During the first half of the transition, the mouth position is set to that of the first keyframe, the multiplier decreases from 1.0 to 0.5. During the second half, the mouth position is set to that of the second keyframe, the multiplier increases from 0.5 to 1.0

Below is an illustration of the formula:

Table 3 Animation Summary

Rhubarb Output	Time	Lip Animation	Multiplier	Time
rest	0	rest	0	0
		A	0.5	0.25
A	0.5	A	1	0.5
		A	0.75	0.75
		A	0.5	1.0
		U	0.75	1.25
U	1.5	U	1.0	1.5
		U	0.5	1.75
rest	2.0	rest	0	2.0

Evaluation

Below are some images with various lip positions and multipliers:

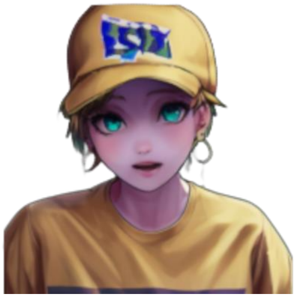


Figure 9 "aaa" 0.25

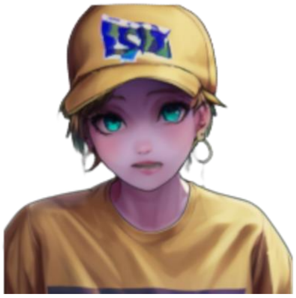


Figure 10 "iii" 0.5

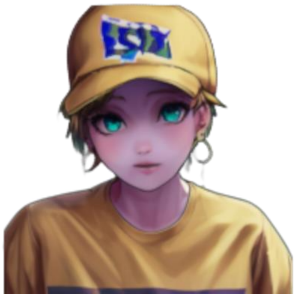


Figure 11 "uuu" 0.33

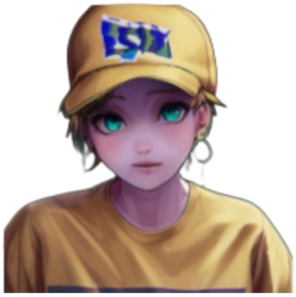


Figure 12 "eee" 0.25

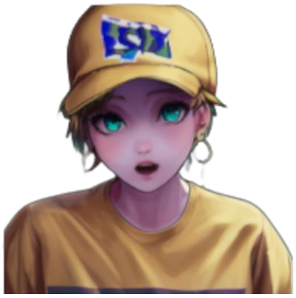


Figure 13 "ooo" 1.0

The transitions between each mouth position are generally smooth. However, there is still a sense of unnaturalness due to the linear increasing/decreasing of the multiplier.

Furthermore, it must be noted that Rhubarb is limited in that it cannot analyze audio clips with music. Therefore, in the current implementation, songs cannot be analyzed, limiting its use to regular speech.

Applications

Digital avatars are used nowadays mostly in digital entertainment. The techniques explored in this project, once refined, can no doubt see further applications in this field.

1. Gaming

   Digital avatars can be used to represent various playable or non-playable characters in games. The task of creating these avatars is often expensive, especially for 3D games. Therefore, many indie developers must shoulder the financial burden of hiring artists. A technique to automatically generate and animate avatars for use in video game development can make game development accessible to more people.

   An application for this project in the gaming field is animated character portraits.

2. Animation

   Although AI art and animation is still incapable of creating entire scenes and thus is still a long way off from emulating human animators, automating the animation of mouth movements during dialogue could streamline a mundane task for animators, allowing them to focus on other parts.

3. VTubers

   VTubers are online streamers who present themselves, and interact with their audience, in virtual avatars controlled via motion capture.

   Currently, these avatars are created in specialized software such as Live2D. Various body parts must be painstakingly illustrated and animated by professional artists. Thus, these avatars are expensive to commission, costing up to 300 USD. Full 3D avatars could cost up to 5,000 USD. (Khungurn, 2019)

   Due to the costs involved, VTuber industry is monopolized by several agencies, like Hololive, with 68 immensely popular VTubers under their banner. However, they are greatly numbered by the number of independent VTubers, which was at least 10,000 as of 2020.

   Therefore, there is a large market for VTuber avatars, and a solution to create them automatically would no doubt be popular. Furthermore, VTuber avatars only need to be capable of limited above-shoulder movement and lip syncing, which was already accomplished by this project.

Figure 14 Kizuna Ai, often credited as being the first VTuber

4. Metaverse

   With the recent development of various metaverse projects however, people will no doubt seek to represent themselves in various ways inside these virtual worlds. Techniques to automate the creation of digital avatars will again be of value.

Bibliography

Anwar, A. (7. May 2020). *Towards Data Science.* Von What is Transposed Convolutional Layer?: https://towardsdatascience.com/what-is-transposed-convolutional-layer-40e5e6e31c11 abgerufen

Brownlee, J. (17. June 2019). *Machine Learning Mastery.* Von A Gentle Introduction to Generative Adversarial Networks: https://machinelearningmastery.com/what-are-generative-adversarial-networks-gans/ abgerufen

Chao, B. (26. June 2022). *Github.* Von Anime Face Dataset: https://github.com/bchao1/Anime-Face-Dataset abgerufen

Khungurn, P. (25. 11 2019). *Talking Head Anime from a Single Image.* Von Github: https://web.archive.org/web/20220327163627/https://pkhungurn.github.io/talking-head-anime-2/ abgerufen

Pumarola, A., Agudo, A., Martinez, A. M., Sanfeliu, A., & Moreno-Noguer, F. (2019). GANimation: Anatomically-aware Facial Animation from a Single Image. *International Journal of Computer Vision (IJCV)*. Von https://arxiv.org/pdf/1605.03557.pdf abgerufen

Weng, L. (11. July 2021). *Lil'Log.* Von What are Diffusion Models?: https://lilianweng.github.io/posts/2021-07-11-diffusion-models/ abgerufen

Wolf, D. S. (2015). *Rhubarb Lip Sync*. Von Github: https://github.com/DanielSWolf/rhubarb-lip-sync abgerufen

Zhou, T., Tulsiani, S., Sun, W., Malik, J., & Efros, A. A. (2016). View Synthesis by Appearance Flow.