國立台灣師範大學理學院
資訊工程學研究所
碩士論文
Department of Computer Science and Information Engineering
College of Science
National Taiwan Normal University
Master's Thesis

3D-GANTex: 3D Face Reconstruction with StyleGAN3-based Multi-View Images and 3DDFA based Mesh Generation

Rohit Das

指導教授：王科植 博士
Advisor：Dr. Ko-Chih Wang, Ph.D.
共同指導教授：林宗翰 博士
Co-Advisor：Dr. Tzung-Han Lin, Ph.D.

Acknowledgement

I would like to express my heartfelt gratitude to all those who have contributed to the completion of this master's thesis. Without their support, guidance, and encouragement, this endeavor would not have been possible.

First and foremost, I am deeply thankful to my supervisor, Prof. Tzung-Han Lin and Prof. Ko-Chih Wang, for their unwavering guidance throughout this research. Their expertise, insightful feedback, and continuous encouragement have been invaluable in shaping the direction of this thesis. I am truly grateful for their dedication and commitment.

I extend my sincere appreciation to the faculty members of National Taiwan Normal University for their valuable insights and support during my academic journey. Their commitment to excellence and passion for knowledge have been a constant source of inspiration.

I am indebted to the participants of this study who generously shared their time and insights. Their contribution has been fundamental in enriching the research findings and ensuring its relevance to real-world contexts.

My heartfelt thanks go to my family and friends for their unconditional love, encouragement, and understanding throughout this demanding journey. Their unwavering support and belief in my abilities have been my motivation during challenging times.

Lastly, I express my gratitude to all the researchers, scholars, and authors whose work has laid the foundation for this study. Their groundbreaking research and dedication to advancing knowledge have been an inspiration to me.
Thank you all for your unwavering support and belief in my abilities.

Abstract

Texture estimation from a single image is a challenging task due to the lack of texture information available and limited training data. This thesis proposes a novel approach for texture estimation from a single in the wild image using a Generative Adversarial Network (GAN) and 3D Dense Face Alignment (3DDFA). The method begins by generating multi-view faces using the latent space of GAN. Then 3DDFA generates a 3D face mesh as well as a high-resolution texture map that is consistent with the estimated face shape. The generated texture map is later refined using an iterative process that incorporates information from both the input image and the estimated 3D face shape.

Studies have been conducted to investigate the contributions of different components of the mentioned method, and show that:

1. Use of the GAN latent space can be a critical benchmark for achieving high-quality results.
2. Editing the latent space can generate high quality multi-view images.
3. Generating 3D mesh and texture map estimation from a single image is possible with a very high accuracy.

To evaluate the effectiveness of this approach, experiments were conducted on in-the-wild images and the results were compared with state-of-the-art 3D Scanner. To verify that, subjective valuation has been performed on 16 participants. The results prove that the mentioned method outperforms existing method in terms of performance, demonstrating the effectiveness of this approach.

Results generated from the aforementioned method are very accurate and has the potential to serve as an important contribution in avatar creation as well as 3D Face Reconstruction.

In summary, the proposed method for texture estimation from a single image using GAN latent space and 3DDFA represents a significant advancement in the field of computer vision and has potential applications

in a wide range of fields, including virtual try-on, facial recognition, beauty industry as well as metaverse.

Keywords: 3D Face Reconstruction, Generative Adversarial Network (GAN), Latent Space, StyleGAN3, Texture Map, Multi-View Generation.

Table of Contents

List of Figures

List of Table

1. Introduction

The last few years have witnessed tremendous growth in generative models for synthesizing high-quality photorealistic face images. However, multi-view synthesis from a single image remains a significant challenge. Generating multiple views of a human face from a single image is a complex task that involves recovering the 3D geometry and texture of the face from a 2D representation. This challenge is further amplified when the 2D image includes a specific pose, and generating texture for the unseen parts presents an additional obstacle.

While there have been notable advancements in computer vision and graphics techniques for 3D face reconstruction, several challenges persist, making this task difficult. These challenges encompass from multi-view images to 3D face model with texture map.

1.1 Objective

One of the main challenges for 3D face reconstruction is the non-rigid nature of human faces, which means that the shape and appearance of the face can vary significantly depending on factors such as facial expression, lighting, and pose. This makes it difficult to model the face using a single 3D template or shape model, and requires more complex representations that can capture the subtle variations in the face across different views. Another challenge is the limited amount of information that can be obtained from a single image, especially if the face is partially occluded or blurred. This can make it difficult to recover the fine details of the face, such as wrinkles, pores, or texture, which are important for realistic face synthesis. Despite these challenges, there have been several approaches proposed for generating multi-view images of human faces from a single image, such as using deep learning techniques, shape-from-shading methods, or face-specific prior models. However, each approach has its own limitations and trade-offs, and the performance may depend on factors such as the quality of the input image, the degree of pose variation, and the desired level of realism.

1.2 Face Frontalization

Multi-view generation from a single image, particularly for the front face, involves the task of synthesizing multiple views of a human face from a single 2D image. The goal is to generate different views of the face as if it were observed from different angles or poses.

However, there are several challenges in multi-view generation from a single front face image:

1. 3D geometry recovery: Generating multiple views requires recovering the underlying 3D geometry of the face from the 2D image. Estimating the precise shape, structure, and pose of the face from a single image is a challenging task, as it involves solving the correspondence problem and inferring the 3D information from a 2D projection.
2. Texture extrapolation: In multi-view generation, the texture of the face needs to be extrapolated or synthesized for the unseen parts of the face in the generated views. This requires accurately inferring and extending the texture information from the visible regions of the face in the input image to the occluded or unseen regions in the generated views.
3. Viewpoint consistency: Maintaining viewpoint consistency across the generated views is crucial for producing visually coherent and realistic results. The challenge lies in ensuring that the generated views align with the given front face image and have consistent facial features, proportions, and pose.
4. Facial expression and non-rigid deformations: Capturing and reproducing facial expressions and non-rigid deformations across multiple views is challenging. Faces can exhibit various expressions and deformations, and accurately modeling and transferring these variations to the generated views requires robust techniques that account for facial dynamics and articulations.
5. Occlusions and self-occlusions: Occlusions occur when certain parts of the face are obscured or hidden from view, either by other facial components or external objects. Handling occlusions and self-occlusions in multi-view generation is a complex problem, as it

requires estimating and inpainting the missing or occluded regions in the generated views.

6. Realism and quality: Generating high-quality and visually realistic views that closely resemble the appearance of real faces is a key challenge. Ensuring the generated views exhibit accurate lighting, shading, texture details, and overall facial realism is crucial for applications such as virtual reality, computer graphics, and human-computer interaction.

This is an ill-posed problem as there are multiple possible solutions, and the optimal solution may not be unique or easy to find. There are several approaches to generating multi-view from single-shot images, such as using deep learning techniques, image-based rendering, or 3D shape recovery.

1.3 Texture Generation

Texture generation from a single image involves the process of synthesizing realistic and detailed texture information for a 3D model, particularly for the surface of a human face, using only a 2D image as input. The aim is to create a texture map that captures the appearance and visual details of the object accurately.

However, there are several challenges in texture generation from a single image:

1. Limited texture information: Single images often provide limited texture details, especially when the image resolution is low or when certain parts of the object are occluded or not visible in the image. The challenge lies in extrapolating and inferring missing or unseen texture information to generate a complete and coherent texture map.
2. Perspective distortion: The perspective distortion introduced by the viewpoint from which the 2D image was captured can impact the accuracy of texture generation. Texture details may appear stretched, compressed, or distorted, requiring techniques to account for and correct these distortions to ensure a realistic texture map.
3. Illumination variations: Lighting conditions in the 2D image can affect the appearance of the object's surface and introduce variations in color, shading, and highlights. Handling illumination variations

and accurately incorporating lighting effects into the texture generation process is a challenge.

4. Surface deformations and non-rigid objects: Generating textures for non-rigid objects, such as faces with different expressions or deformations, presents additional challenges. Capturing and representing the texture variations caused by surface deformations in a realistic and coherent manner is a complex task.
5. Texture resolution and detail: Generating high-resolution and detailed textures from a single image is challenging, particularly when the image has low resolution or when the object contains fine-scale details. Preserving and enhancing the texture details during the generation process is crucial for creating visually convincing and realistic texture maps.
6. Artifacts and visual inconsistencies: Texture generation algorithms may produce artifacts, such as blurring, discontinuities, or inconsistent patterns in the texture map. Overcoming these artifacts and ensuring smooth transitions, consistent patterns, and visually plausible textures are ongoing challenges.

Recently there has been significant progress in generating texture by using GAN[1] . There has been some research[2, 3] that use the concept of rotating the 3D mesh of an image to an arbitrary pose and rendering the image back in 2D space to get a 2D rendered image. These methods though generate sublime results, needs a lot of computational resources.

1.4 3D Model Generation

3D face reconstruction from a single image involves the process of recovering the three-dimensional geometry and texture of a human face using only a two-dimensional image as input. The goal is to recreate a realistic 3D representation of the face, capturing its shape and appearance accurately.

However, there are several challenges in this task:

1. Ambiguity: A single 2D image provides limited information about the 3D structure of the face, resulting in inherent ambiguity. Multiple 3D configurations can potentially produce the same 2D projection, making it challenging to determine the true underlying 3D shape.

2. Viewpoint dependence: The accurate reconstruction of a face is highly dependent on the viewpoint or pose from which the 2D image was captured. Different poses introduce variations in the face's shape and appearance, making it difficult to generalize the reconstruction to other viewpoints.
3. Illumination and shading variations: Lighting conditions and shadows in the 2D image can obscure or distort facial features, making it challenging to accurately estimate the 3D shape and texture. Handling variations in lighting conditions and compensating for shadows is a crucial aspect of 3D face reconstruction.
4. Limited texture information: Single images often do not provide sufficient texture information for reconstructing the detailed appearance of a face. Unseen parts of the face, occlusions, or image resolution limitations can hinder the accurate generation of textures, leading to incomplete or unrealistic reconstructions.
5. Facial expressions and non-rigid deformations: Faces are highly deformable, and capturing non-rigid deformations caused by facial expressions or natural movements is challenging from a single image. Accurately modeling and reconstructing these deformations adds another layer of complexity to the process.
6. Computational complexity: Achieving real-time or efficient 3D face reconstruction from a single image is computationally demanding due to the complex optimization and inference processes involved. Balancing accuracy and efficiency remain a challenge, especially for practical applications.

Addressing these challenges requires advancements in computer vision, graphics techniques, and machine learning algorithms. Researchers are actively exploring novel approaches, such as deep learning-based methods, to tackle these difficulties and improve the accuracy and robustness of 3D face reconstruction from single images.

1.5 Challenges for 3D Model Generation

Given the recent development of generative models, arises the following challenges for 3D face generation:

1. How to synthesize multi-view of human face?
2. How to estimate near to accurate texture from a single image?

3. How to estimate near to accurate 3D mesh?
4. How to acquire the proper training dataset?

1.6 Proposed Framework

The proposed thesis 3D-GANTex addresses these issues and propose the use of Generative Adversarial Network (GAN)[1] to generate multi-view from a single image by manipulating the latent space. The generated images post processed, will be passed through an UV map model. After collecting the UV map from these multi-view images and taking the center view map, the final frontal image with a near to accurate texture map is developed.

The proposed framework does not need to rely on paired data or any kind of label. Thus, with unlimited training data this model can be used from large-scale face recognition to creating avatars in metaverse. In near future where augmented reality and virtual reality will be a very common medium, creating our own avatars from a single selfie picture will definitely help to immerse more in human computer interaction. The film industry can also benefit from this technology, especially in scenarios where it may be expensive or impractical to film at a particular location. Additionally, the technology can be used to save time and costs by utilizing a stand-in actor whose facial features can later be replaced with those of the main actor.

The thesis contributes to the following:

1. Proposing multi-view synthesis using latent space.
2. Proposing 3D face model generation with a texture map.
3. Requiring no labeled data.

The proposed model in this thesis is a motivation from FFHQ-UV[4], Image editing with StyleGAN3[5] and 3DDFA [6, 7] respectively.

2. Literature Review

This section introduces about the various methods used and implemented to achieve the results mentioned in this thesis.

2.1 Generative Adversarial Networks (GAN)

GAN[1] stands for Generative Adversarial Network. It is a class of machine learning models that consists of two neural networks, a generator and a discriminator. The generator learns to create new samples that resemble the training data, while the discriminator learns to distinguish between the real samples from the training data and the fake samples generated by the generator. During training, the generator produces fake samples and tries to fool the discriminator into thinking they are real, while the discriminator tries to correctly identify whether each sample is real or fake. As the generator and discriminator are trained together, they both improve over time. The generator learns to produce more realistic samples that are harder to distinguish from the real data, while the discriminator becomes better at identifying fake samples. Once trained, the generator can be used to generate new samples that are similar to the training data, allowing for the creation of new and novel data. GANs have been used to generate realistic images, music, and even video game levels. GANs are a powerful and versatile tool in the field of machine learning and have numerous applications in fields such as computer vision, natural language processing, and robotics. Figure 2.1 shows the architecture of GAN.

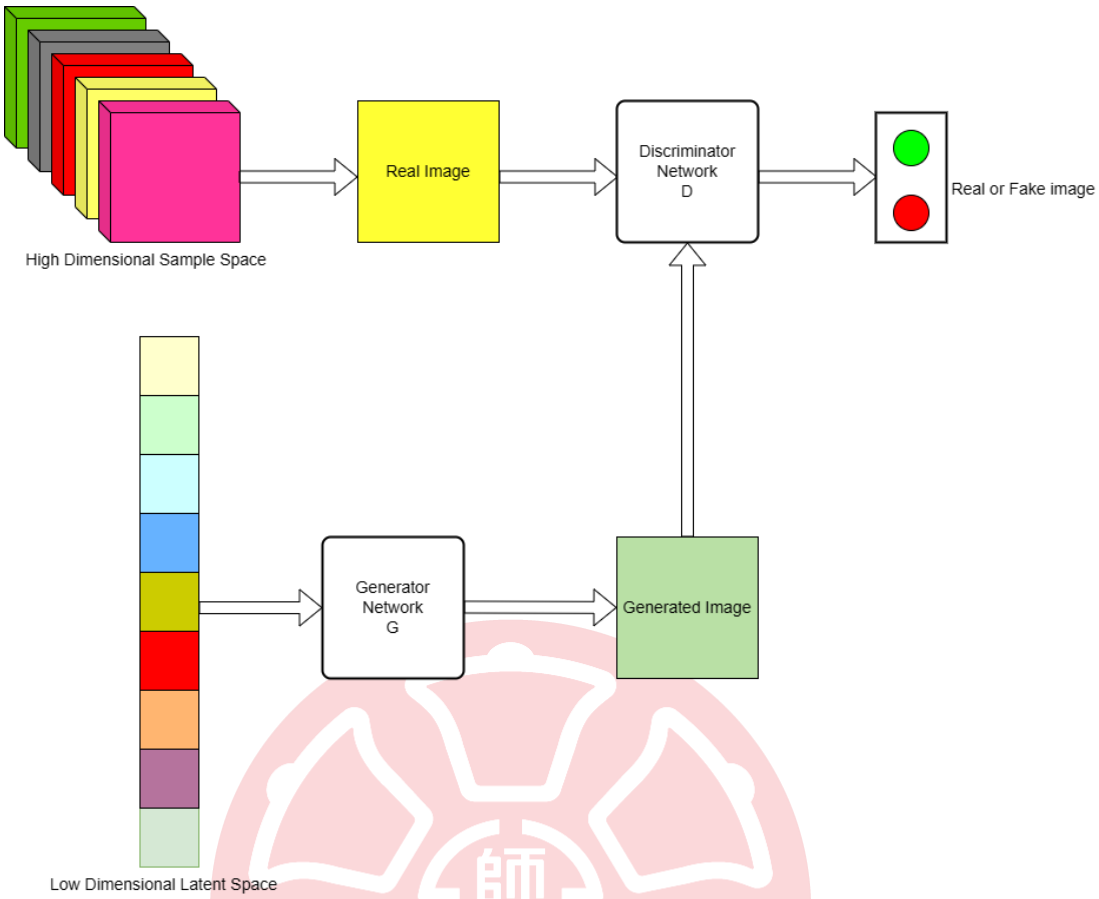


Figure 2.1: Architecture of vanilla generative adversarial network

2.2 Latent Space

Reconstruction based models like GAN have revolutionized facial data synthesis[8-10]. In GAN, the latent space refers to the high-dimensional space of random noise vectors that are fed into the generator network to produce realistic-looking images. The generator network maps the random noise vectors from the latent space to the image space, where it generates synthetic images. The latent space can be considered as a compressed representation of the distribution of real images that the generator aims to replicate.

Embedding images in latent space involves representing images as points in a high-dimensional space, where each point corresponds to a unique set of parameters or codes defining the visual features of the image. This process enables operations such as image synthesis, manipulation, and

interpolation by manipulating the corresponding codes or vectors in latent space. Different types of latent spaces are used to represent and manipulate the generative process in the context of generative models such as Variational Autoencoders (VAE) [11], GAN, and Auto-Regressive models. For generative models such as StyleGAN [8-10], the latent space refers to the parameter space used to generate images with a particular style or distribution. By manipulating the corresponding codes or vectors in latent space, operations such as image synthesis, manipulation, and interpolation become possible. Figure 2.2 illustrates the conceptual representation of Latent Space. Some of the most common types of latent spaces:

1. Continuous latent space: In a continuous latent space, the latent variables are continuous and can take any value within a certain range or distribution. This allows for smooth and continuous variations in the generated output, such as changing the color or position of an object in an image.
2. Discrete latent space: In a discrete latent space, the latent variables can only take a finite number of values, usually represented as integers or one-hot vectors. This allows for explicit control over the generative process, such as selecting a specific class or attribute for the generated output.
3. Conditional latent space: In a conditional latent space, the latent variables are conditioned on additional inputs such as labels or attributes, which can influence the generative process. This allows for conditional generation of output, such as generating an image of a specific object given its category.
4. Hierarchical latent space: In a hierarchical latent space, the latent variables are organized in a hierarchical structure, with higher-level variables controlling more abstract features and lower-level variables controlling more detailed features. This allows for disentangling of the latent factors and more fine-grained control over the generative process.
5. $\boldsymbol{W+}$ latent space: $\boldsymbol{W+}$ latent space is an extension of the $\boldsymbol{W}$ latent space used in StyleGAN and StyleGAN2, which adds a set of "style vectors" to control the fine-grained details and variations of the generated output.

6. $\boldsymbol{S}$ latent space: $\boldsymbol{S}$ latent space also known as style space refers to the space of style vectors that control the appearance and fine-grained details of the generated images. The style space is typically represented as a vector of numbers, where each dimension corresponds to a specific aspect of the image, such as color, texture, and lighting. $\boldsymbol{W}$ space is a part of style space.

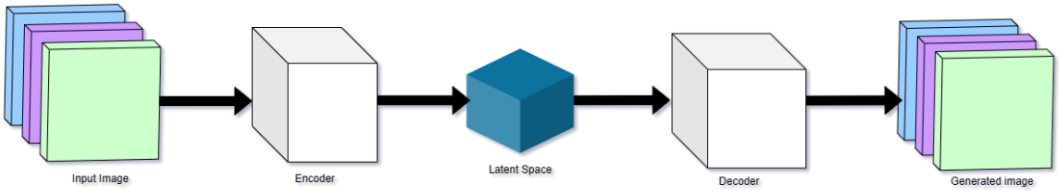


Figure 2.2: Conceptual representation of latent space

In W latent space, each image is represented as a point in a high-dimensional space, where each dimension corresponds to a parameter that controls different aspects of the generated image, such as its color, texture, and structure. $W+$ latent space (18×512 dimensions), introduced in StyleGAN2-ADA[12], extends the W space (512 dimensions) by adding a set of "style vectors"(18 layers) that can be used to control the fine-grained details and variations of the generated image. Each image is represented as a pair of a "style vector" and a "latent vector". The "style vector" captures the high-level features of the image such as pose, expression, and identity, while the "latent vector" controls the lower-level details such as texture, color, and shape. The addition of "style vectors" in $W+$ space provides several benefits over the original W space, such as more fine-grained control over the generated images, improved disentanglement of the latent factors, and better interpolation and mixing of styles. These benefits have been shown to lead to significant improvements in the quality and diversity of the generated images, as well as better generalization and robustness of the model to different datasets and tasks. The foundation of GAN gave rise to StyleGAN[10] which uses style transfer literature to generate high quality images. The recently proposed StyleGANv3 [8] has shown high-quality 2D face synthesis up to 1024×1024 resolution and fixed the previous underlying issue of unwanted

information leak in the synthesis process Table 2.1 shows DCI(Disentanglement/Completeness/Informativeness)[5, 13] metrics for StyleGAN2 and StyleGAN3 of each latent space.

Table 2.1: DCI metrics for StyleGAN2 and StyleGAN3 modified from [5]

Generator	Space	D	C	I
StyleGAN2	$\mathcal{Z}$	0.3	0.2	0.7
StyleGAN2	$\mathcal{W}$	**0.5**	**0.6**	**1.0**
StyleGAN2	$\mathcal{S}$	0.8	0.9	1.0
StyleGAN3	$\mathcal{Z}$	0.4	0.3	0.8
StyleGAN3	$\mathcal{W}$	**0.5**	**0.4**	**0.95**
StyleGAN3	$\mathcal{S}$	0.8	0.9	1.0

To compute the above metrics, Wu *et al.* [14] utilized pre-trained attribute regressors for different attributes. The DCI scores for unaligned datasets showed a consistent improvement from the initial Gaussian noise $\mathcal{Z}$, to the intermediate space W, and then to the style parameter S for both StyleGAN architecture.

2.3 Multi-View Face Generation

Multi-view face generation is the process of synthesizing multiple views of a human face from a single input image. This is typically done using deep learning models such as Generative Adversarial Networks (GANs)[1] or Variational Autoencoders (VAEs) [11]. The goal of multi-view face generation is to create realistic and diverse images of a person's face from different viewpoints, which can be useful in various applications such as virtual reality, gaming, and security systems. One approach to multi-view face generation is to use a GAN model that is trained on a dataset of 3D face models, which contain different views of a face. The generator network of the GAN takes as input a low-dimensional latent code, which

is mapped to a high-dimensional image space using a series of convolutional layers. The discriminator network of the GAN is trained to distinguish between real and generated images. During training, the generator network learns to map the latent code to different views of the face, while the discriminator network learns to provide feedback to the generator network to improve the quality of the generated images. Once the GAN model is trained, it can be used to generate multiple views of a face from a single input image by first mapping the image to the latent space using an encoder network and then sampling different points in the latent space to generate different views of the face.

2.4 Face Rotation

Face rotation (commonly known as “pose”) involves the generation of multi-view from a single image. Among those views, the frontal view attracts a lot of interest. Face frontalization as mentioned in 1.2 is a computer vision technique used to manipulate images of faces to make them appear as if they are directly facing the camera. It involves transforming an image of a face that is taken at an angle or in profile view to look as if it is taken from a frontal view. This technique is commonly used in face recognition systems, where the performance of the algorithm is improved by aligning the face images to a common orientation. It is also used in entertainment industry especially in movie sets to replace the face of actors. Face frontalization can also be used in other applications, such as in virtual try-on systems, where users can see how clothes or accessories would look on a frontal view of their face. The process of face frontalization typically involves detecting facial landmarks or key points in the image, such as the eyes, nose, and mouth, and using them to estimate the pose of the face. Then, a transformation is applied to the image to adjust the pose of the face to a frontal view. This transformation can be done using techniques such as 3D face modeling, affine transformations, or image warping. Traditional way of solving the problem includes molding a single reference model and warping the image around it[15, 16]. There is also 3DMM[17] which derive a face model by transforming the shape and texture of 2D images into a vector space representation from a set of face 3D model. However, the synthesized results are not photorealistic and does

not provide perfect texture. Fast forward a few years and the introduction of deep learning solve problems with higher accuracy than traditional methods.

Many follow up works in face frontalization[6, 7, 18-24] tackles the above problem. The results though acceptable needs paired dataset in a supervised environment. Most of the training data used above are constraint to controlled environments such as Multi-PIE[25] or synthetic dataset such as 300W-LP[6]. Models trained on controlled datasets tend to overfit and lack the desired generalization ability for in the wild images. Also, the generated image resolution is not suitable enough to generate good texture. The concept of embedding images in latent space of StyleGAN which enables semantic image editing operations indicates that StyleGAN generator can be used for multi-view face synthesis from a single image.

2.5 Encoder

Encoder also plays an important role for generating high quality texture. An encoder is a neural network architecture that takes an input and maps it to a latent representation, often with a lower dimensionality than the input. The purpose of the encoder is to learn a compressed representation of the input that retains the most important features of the data. In machine learning, encoders are commonly used in autoencoder architectures, where the encoder maps the input data to a lower-dimensional latent space, and the decoder reconstructs the original data from the latent code. Autoencoders are often used for tasks such as data compression, feature extraction, and generative modeling. In the context of GANs (Generative Adversarial Networks), an encoder can also be used to map an input image to a latent space that is compatible with the generator network. The encoder is trained to map input images to a latent code that, when fed into the generator, produces high-quality output images. This technique is used in StyleGAN, where the encoder is used to map an input image to the latent space of the generator, enabling fine-grained editing and control of the generated images. There are many types of StyleGAN encoders:

1. Standard encoder: Traditional encoder used in StyleGAN and StyleGAN2, which maps an input image to the $\boldsymbol{W}$ latent space of the generator.
2. e4e encoder: e4e (Encoder4Editing) [26] is a method that learns an encoder network to map an input image to the $\boldsymbol{W+}$ latent space of StyleGAN. The $\boldsymbol{W+}$ latent space includes additional style vectors that control the fine-grained details and variations of the generated images, and can be used for more fine-grained editing and control. Figure 2.3 shows post embedding results using e4e encoding method. The generated results produce quality texture and features but fails to preserve the identity.
3. pSp encoder: pSp (prioritized StyleGAN) [27] is another method that uses a pre-trained encoder network to project an input image to the $\boldsymbol{W+}$ latent space of StyleGAN, but with a prioritized set of style vectors that are optimized for specific editing tasks. Figure 2.4 shows post embedding results using pSp encoding method. The generated results retain good facial identity but fails to preserve the quality features.
4. Avatar encoder: An encoder[28] used for the creation of avatars, which maps an input image of a person to a StyleGAN latent space that is specifically designed for avatars. The avatar encoder takes into account the specific characteristics and features of avatars, such as hair, clothing, and accessories, and produces high-quality avatar images that can be used in various applications.

Figure 2.3: Generated results after embedding image into latent space using e4e encoder. The output generated normalizes the texture and lighting in (a) as well as (b)

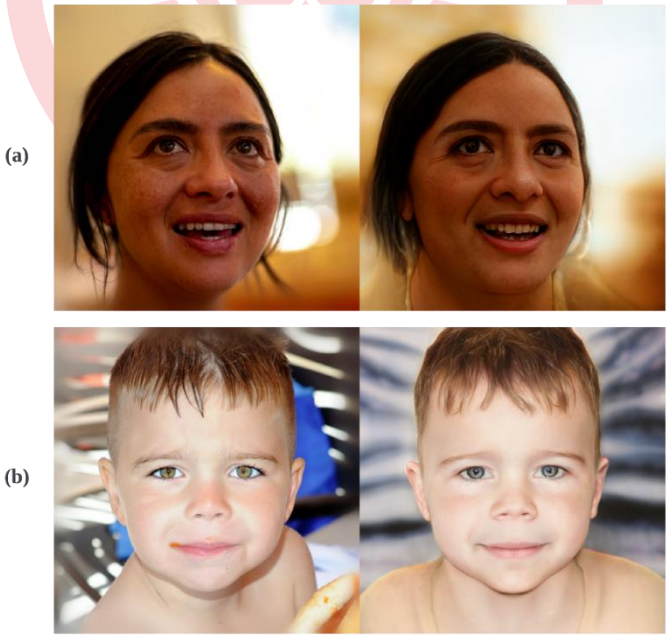


Figure 2.4: Generated results after embedding image into latent space using pSp encoder. The output generated preserves the identity of the image in (a) as well as (b)

2.6 Texture

In computer graphics, a texture map is a 2D image that is applied to the surface of a 3D object to give it the appearance of a particular material or surface texture as shown in Figure 2.5 . Texture maps are commonly used in 3D modeling and rendering to add details and realism to the surfaces of 3D objects, such as skin, wood, metal, or fabric. Texture maps contain information that describes the surface properties of an object, such as color, reflectivity, transparency, or bumpiness. The texture map is applied to the surface of the 3D object by mapping each pixel of the texture map to a corresponding point on the surface of the object. This process is called texture mapping. Texture maps can be created in various ways, such as by photographing real materials, painting textures by hand, or generating them procedurally using computer algorithms. Modelling and synthesis of faces have been studied in 3D as well [17, 29-31].

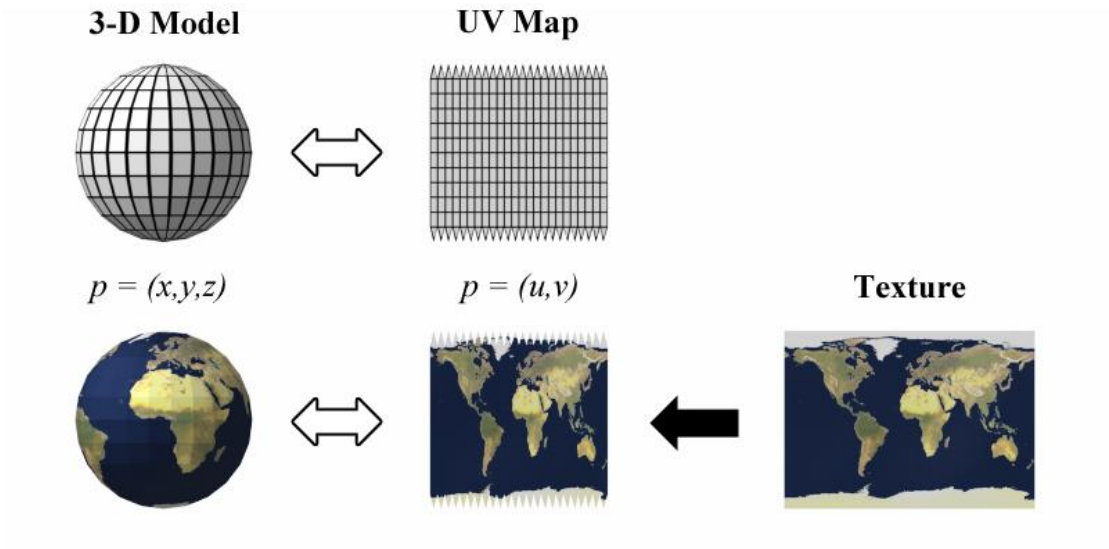


Figure 2.5: The application of a texture in the UV space related to the effect in 3D.[1]

[1] UV mapping - Wikipedia

2.6.1 Research based on 3D Morphable Model (3DMM)

Introduction to 3DMM[17] opened the door to many researches in facial geometry representation. 3DMM stands for 3D Morphable Model, which is a statistical model of 3D facial shape and texture that can be used to generate realistic 3D models of human faces. A 3DMM is typically created by analyzing a large dataset of 3D scans of human faces, and then using statistical methods to identify the most important variations in facial shape and texture across the dataset. The resulting 3DMM consists of a set of principal components that describe the variations in facial shape and texture, which can be used to generate a wide range of realistic 3D face models. By adjusting the weights of these principal components, it is possible to generate new 3D faces that have similar characteristics to the faces in the original dataset, but with different shapes or textures. Figure 2.6 shows the use of 3DMM. Derived from a dataset of prototypical 3D scans of faces, the morphable face model contributes to two main steps in face manipulation:

1. Deriving a 3D face model from a novel image.
2. Modifying shape and texture in a natural way.

The usage of Principal Component Analysis (PCA) which statistically estimates the prior distribution in the space of colored 3D scans is the one of the key factors that 3DMM is used still now for face model synthesis. However, generation of textures from the above-mentioned models[4, 17, 29-32] are far from realistic as seen in Figure 2.7. Some methods depend on global generalization function like PCA.

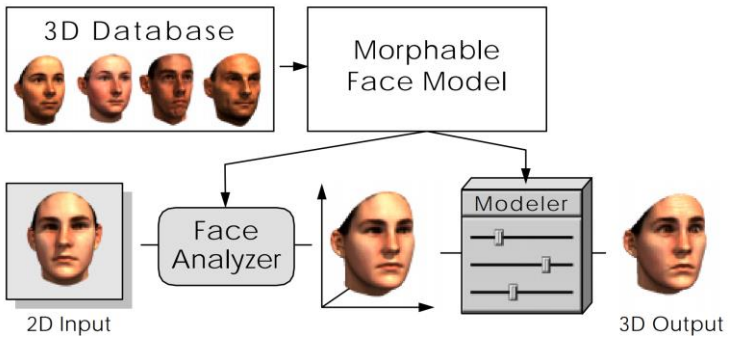


Figure 2.6: How to use 3DMM as described in [17]

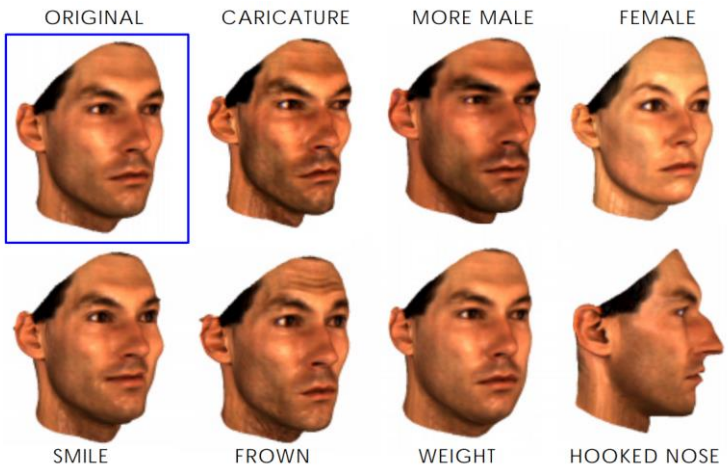


Figure 2.7:Variation of facial attributes of a single face generated from 3DMM as described in [17]

2.6.2 3D Dense Face Alignment 3DDFA

3DDFA (3D Dense Face Alignment) [6, 33] is a deep learning-based method for dense 3D face alignment and reconstruction. The goal of 3DDFA is to accurately reconstruct the 3D geometry of a face from a single 2D image, and to align a dense set of facial landmarks to the reconstructed 3D face model. This can be useful in various applications such as face recognition, facial expression analysis, and virtual reality. The 3DDFA method consists of several stages. First, a 2D facial landmark detector is used to detect a set of sparse landmarks on the face. These landmarks are then used to estimate the pose and camera parameters of the face. Next, a 3D face model is constructed using 3DMM and a set of 3D landmarks that are manually annotated on a small set of training images. The template model is then aligned to the estimated pose and camera parameters using a Procrustes analysis. Finally, a dense set of 3D facial landmarks is aligned to the reconstructed 3D face model using a cascaded regression[34-36] approach. The texture of the face is then mapped onto the 3D model using a texture mapping technique, such as UV mapping. This involves training a set of regression models that predict the offsets of the 3D landmarks from the initial estimate, and then iteratively refining the estimate using the predicted offsets. Figure 2.8 shows an overview of 3DDFA model. In the

first stream, a novel Projected Normalized Coordinate Code (PNCC) is stacked with the input image and sent to the CNN along with an intermediate parameter p_k. Meanwhile, the second stream utilizes Pose Adaptive Convolution (PAC) on feature anchors with consistent semantics. The output of both streams is merged using an additional fully connected layer to predict the parameter update Δp_k. The 3DDFA method achieves state-of-the-art performance on several benchmark datasets for 3D face reconstruction and dense face alignment, and has been used in various applications such as virtual try-on and facial expression transfer.

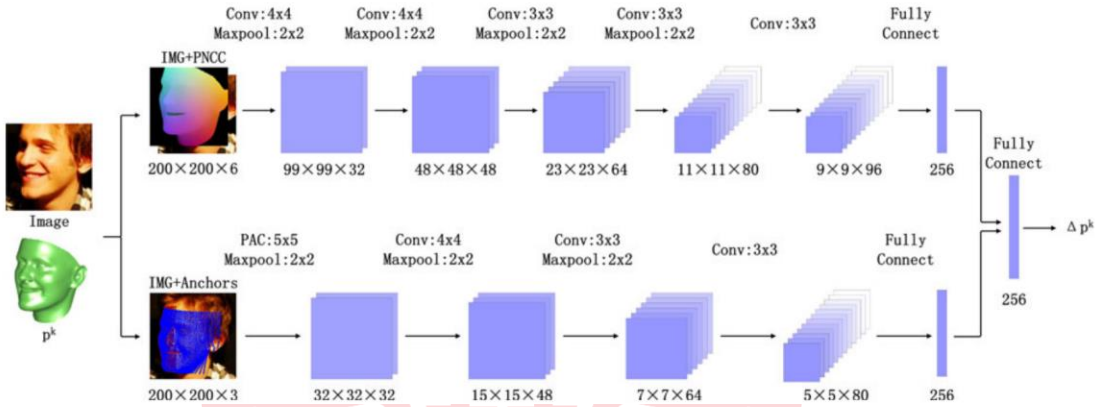


Figure 2.8: Overview of 3DDFA as described in [6]

2.7 Dataset

There are many UV datasets available to train a model. But paired dataset and high-quality texture map is difficult to procure. The reason being:

1. Privacy concerns: Many applications that require paired datasets or high-quality texture maps involve human faces or other personal information, which raises privacy concerns. It can be difficult to obtain consent from individuals to use their images or data for research or commercial purposes.
2. Cost: Collecting and curating large datasets and creating high-quality texture maps can be expensive, especially if it involves specialized equipment or expertise.
3. Complexity: Creating a paired dataset or high-quality texture map can be a complex process that requires a significant amount of time

and effort. It may involve multiple steps, such as data collection, cleaning, labeling, and processing, which can introduce errors or biases.

4. Intellectual property: In some cases, paired datasets or high-quality texture maps may be subjected to intellectual property rights, which can make it difficult or expensive to obtain the necessary permissions or licenses.

FFHQ-UV[4] needs to be trained on 10,000 high quality texture map which is not available to public. LSFM[37], AvatarMe[38, 39], HiFi3DFace[40], Facescape[41] provides good resolution but data is captured under controlled environments. And besides FFHQ-UV [4], there are few texture dataset like UV-GAN[42] and NormAvatar [28] but these are not available to the public. Table 2.2 shows the above data for better reference.

Table 2.2: Different types of datasets and their availability. Data modified from [4]

Dataset	Size	Resolution	Even Illumination	Available to Public
LSFM [37]	10,000	512 × 512	×	×
AvatarMe [38]	200	6144 × 4096	✓	×
HiFi3DFace [40]	200	2048 × 2048	✓	×
Facescape [41]	847	4096 × 4096	✓	✓
UV-GAN [42]	5,638	377 × 595	×	×
NormAvatar [28]	5,601	256 × 256	✓	×
FFHQ-UV [4]	54,165	1024 × 1024	✓	✓

3. Methodology

The methodology of this thesis postulates three parts:

1. Latent Space Embedding
2. Generate Multi-View faces
3. Inverse rendering to generate texture maps

Figure 3.1 shows a brief overview of 3D-GANTex.

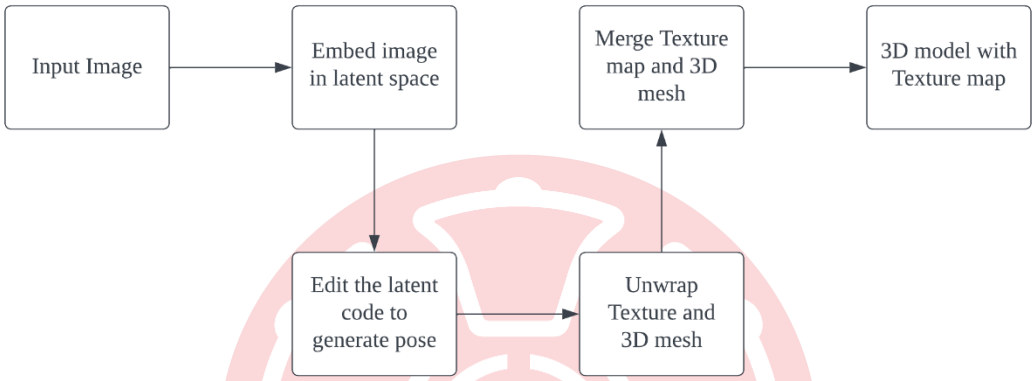


Figure 3.1: Overview of 3D-GANTex pipeline

3.1 Latent Space Embedding

Since this thesis is using StyleGAN3, $W+$ latent space will be used for embedding an image into latent space.

In general, there are two existing approaches to embed instances from the image space i to the latent space $W+$.

1. Training an encoder that maps a given image to the latent space like Variational Auto-Encoder.
2. Select a random initial latent code and use an optimizer to find the local minima like Gradient Descent [43, 44]

Between these two, the first one has more problems generalizing the dataset, and the second one may not always produce the best results. However, it offers a practical way to embed arbitrary images into an existing model without the need for extensive training data or computing resources. This thesis follows the second approach on StyleGAN3.

Given a target image x, the latent code $\hat{w}$ that optimally reconstructs it:

$$\widehat{w} = arg_w \min \mathcal{L}\left(x, G\left(w; (r, t_x, t_y)\right)\right) \quad (1)$$

Where $\mathcal{L}$ is the $\mathcal{L}_2$ or LPIPS[45] reconstruction loss, G is the generator, r is the rotation, t_x and t_y are the translation parameters.
By default, $t_x = t_y = 0$ and $r = 0$.

3.2 Encoder

This thesis uses the concept of Restyle[46] e4e encoder[26]as well as pSp encoder[27] scheme for inverting StyleGAN3.

3.2.1 Restyle Encoder

Restyle is built on top of the StyleGAN architecture, which is a generative model that can produce high-quality synthetic images. Restyle leverages the powerful latent space of StyleGAN to achieve high-quality and flexible image-to-image translation. The key idea behind Restyle is to decompose an input image into a content code and a style code. The content code represents the underlying structure and identity of the face, while the style code represents the fine-grained details such as color, texture, and lighting. By manipulating the style code while keeping the content code fixed, Restyle can generate a variety of stylized versions of the input image. Restyle achieves this by using a modified version of the StyleGAN generator network, where the style code is fed through a separate style modulation network before being applied to each layer of the generator. This allows for fine-grained control over the style of the output image. One of the key advantages of Restyle is its flexibility and speed. Unlike traditional image-to-image translation methods that require paired training data, Restyle can be trained on unpaired datasets, making it more scalable and easier to use in practice. Additionally, Restyle is fast and can perform image-to-image translation in real-time on modern GPUs.

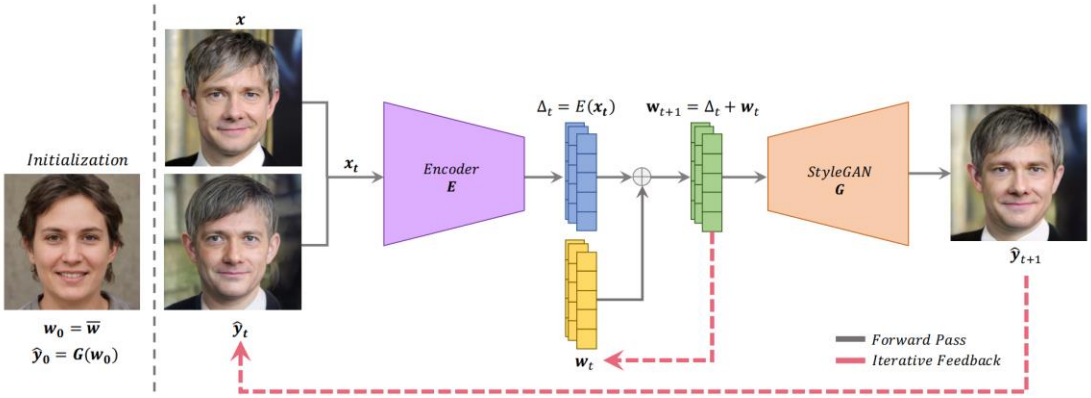


Figure 3.2: ReStyle iterative inversion scheme[46]

Figure 3.2 shows the architecture of Restyle[46]. The scheme begins with initializing the ReStyle model using an input image x and the average latent code w_0, along with its corresponding image $\hat{y}_0$. At step t, the model operates on an extended input obtained by concatenating x with the current inversion prediction $w_0 \in W+$ (yellow). The encoder E is used to predict a residual latent code, $\Delta_t \in W+$ (blue), which is added to the previous latent code w_t to obtain the updated latent code prediction w_{t+1} (green). This updated latent code is then passed to the generator G to obtain an updated reconstruction $\hat{y}_{t+1}$, which is used as input for the following step. This multi-step process is also performed during inference.

3.3 Encoder4Editing(e4e) Encoder

The e4e (encoding for editing) encoding scheme is a method for image editing that uses an encoder-decoder neural network architecture to manipulate the latent code of an input image. The e4e encoding scheme is designed to work with the StyleGAN[10] architecture. The e4e encoding scheme leverages the StyleGAN architecture by using an encoder network to map an input image into the latent space of StyleGAN. This is done by first passing the input image through a convolutional encoder network, which generates a low-dimensional feature vector that encodes the key visual attributes of the input image. The feature vector is then fed through a mapping network that transforms it into a high-dimensional latent code that can be used as input to the StyleGAN generator network. This allows for fine-grained control over the generated image, as changes to the latent

code can result in different visual attributes and styles. One of the key advantages of the e4e encoding scheme is its ability to perform image editing without the need for paired training data. Traditional image editing methods require paired data, which can be difficult and expensive to obtain. The e4e encoding scheme, on the other hand, can be trained on unpaired datasets, making it more scalable and easier to use in practice.

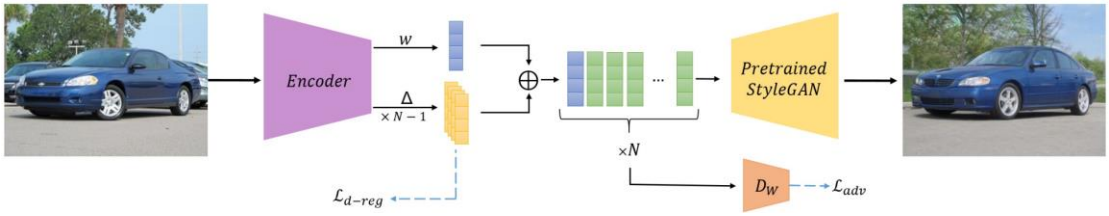


Figure 3.3: e4e encoding scheme[26]

Figure 3.3 portrays the e4e network architecture. The encoder receives an input image and outputs a single style code w together with a set of offsets $\Delta_1 .. \Delta_{N-1}$, where N denotes the number of StyleGAN's style modulation layers. The final latent representation is obtained by replicating the w vector N times and adding each Δ_i to its corresponding entry. During training, the L_{d-reg} regularization encourages small variance between different entries of the final representation, thereby remaining close to W_*. L_{adv} guides each latent code towards the range of StyleGAN's mapping network, resulting in a final representation closer to W_k. As a result of applying both regularization terms, the encoder's final learned representation lies close to W

Restyle scheme can be applied to different encoder architectures and loss objectives because it is a modular framework that separates the encoder network, the style mapping network, and the synthesis network as seen on Figure 3.2. The encoder network E can be any neural network architecture that takes an input image and outputs a feature vector that captures its key visual attributes. This feature vector is then passed through the style mapping network, which transforms it into a high-dimensional latent code that can be used as input to the synthesis network G. The synthesis network is responsible for generating the output image from the latent code. The modular architecture of the Restyle scheme makes it flexible and adaptable to different encoder architectures and loss objectives.

The style mapping network and the synthesis network can be optimized separately, allowing for fine-grained control over the visual attributes of the output image. Furthermore, the Restyle scheme can be trained with different loss objectives, depending on the specific application. For example, it can be trained with adversarial loss to ensure that the generated images are visually indistinguishable from the real images, or it can be trained with reconstruction loss to ensure that the generated images preserve the key visual attributes of the input images.

3.3.1 ReStyle-e4e Encoder

While hierarchical encoders are well-suited for structured domains like facial domain where style inputs can be divided into three levels of detail, they have little impact on less-structured, multimodal domains, and can add unnecessary overhead. In contrast, the multi-step nature of ReStyle simplifies the encoder architecture, making it more efficient. To this end, a simpler variant of the pSp[27] and e4e[26] encoders is used in which all style vectors are extracted from the final 16 × 16 feature maps. Using k different map2style blocks, which are small convolutional networks containing a series of 2-strided convolutions with LeakyReLU[47] activations, the feature map is down-sampled to obtain the corresponding 512-dimensional style input. Figure 3.4 in the paper provides a high-level overview of this encoder architecture.

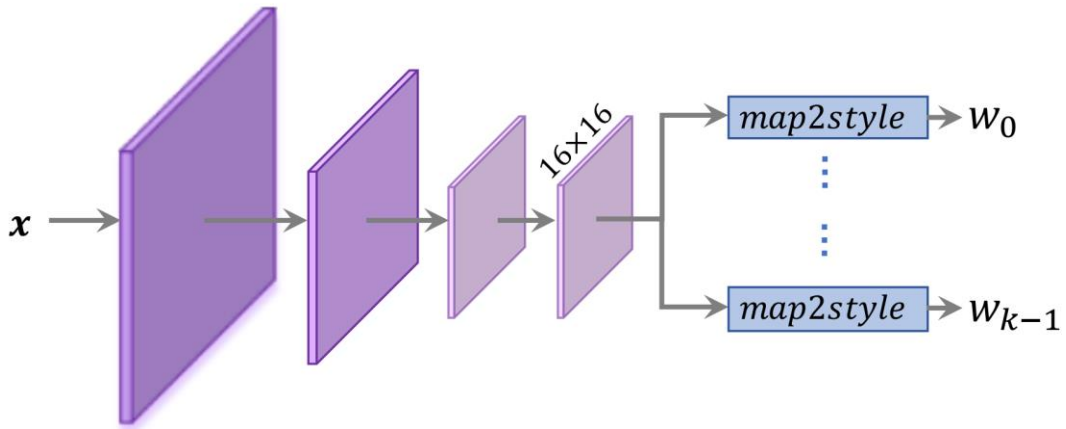


Figure 3.4: Simplified encoder architecture[46]

One way to combine the Restyle SG3 encoder and e4e encoder is to use the Restyle SG3 encoder to obtain an initial estimate of the latent code for an input image, and then refine it using the e4e encoder. This can be done by feeding the input image through the Restyle SG3 encoder to obtain an initial estimate of the latent code, and then using the e4e encoder to refine the latent code by minimizing the reconstruction error between the original image and the synthesized image from the refined latent code.

Another way to combine the two encoders is to use them in a cascaded architecture, where the Restyle SG3 encoder is used to obtain an initial estimate of the latent code, which is then fed as input to the e4e encoder to obtain a refined latent code. This refined latent code can then be used as input to the synthesis network to generate the edited image.

The combination of Restyle SG3 encoder and e4e encoder provides a powerful and flexible framework for image editing, as it leverages the strengths of both encoders to obtain high-quality and accurate latent codes for image synthesis. However, it is important to note that combining different encoder architectures and loss objectives requires careful tuning and experimentation to ensure optimal performance and avoid potential issues such as overfitting and instability.

3.4 InterFaceGAN

InterFaceGAN[48] is a framework that allows for the exploration and manipulation of facial attributes in images using Generative Adversarial Networks (GANs) and latent space editing techniques. The model works by using the pre-trained StyleGAN generator network to generate images of faces. The generated images are then mapped to a lower-dimensional latent space using an encoder network, which is trained to invert the generator. This latent space representation can be manipulated to change specific facial attributes, such as gender, age, or expression, by adding or subtracting specific vectors in the latent space. The manipulation of facial attributes is achieved through the use of a direction vector d, which represents the difference between the latent codes of two images (z_1, z_2) with different attribute values. Here, z_1 represents the input image and z_2 represents the image generated from latent space. InterFaceGAN mentions

two properties:

1. Given $\boldsymbol{n} \in \mathbb{R}^d$ with $\boldsymbol{n} \neq 0$, the set $\{\boldsymbol{z} \in \mathbb{R}^d : \boldsymbol{n}^T\boldsymbol{z} = 0\}$ defines a hyperplane in $\mathbb{R}^d$, and $\boldsymbol{n}$ is called the normal vector. All vectors $\boldsymbol{z} \in \mathbb{R}^d$ satisfying $\boldsymbol{n}^T\boldsymbol{z} > 0$ locate from the same side of the hyperplane.
2. Given $n \in \mathbb{R}^d$ with $\boldsymbol{n}^T\boldsymbol{n} = 1$, which defines a hyperplane, and a multivariate random variable $\boldsymbol{z} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}_d), we\ have\ P(|\boldsymbol{n}^T\boldsymbol{z}| \leq 2\alpha\sqrt{\frac{d}{d-2}}) \geq (1 - 3_e{}^{-cd})(1 - \frac{2}{\alpha}e^{-\alpha^2/2})$ for any α ≥ 1 and d ≥ 4. Here, $P(\cdot)$ stands for probability and c is a fixed positive constant.

From the 1st property, given a hyperplane with a unit normal vector $\boldsymbol{n} \in \mathbb{R}^d$ the distance from a sample **z** to this hyperplane is defined as:

$$d(\boldsymbol{n}, \boldsymbol{z}) = \boldsymbol{n}^T\boldsymbol{z} \qquad (2)$$

The variable d in this context is not a conventional distance measure as it can take negative values. As the input vector z approaches and crosses the hyperplane boundary, both the semantic score and the value of d fluctuate. The semantic attribute changes its value only when the sign of the "distance" flips from positive to negative or vice versa. This behavior is a result of the non-linear mapping between the input and output spaces of the neural network model. So, expecting that the d and z remains linearly dependent, the following equation can be resolved:

$$f(g(\boldsymbol{z})) = \lambda d(\boldsymbol{n}, \boldsymbol{z}) \qquad (3)$$

where $f(.)$ is the scoring function for a particular semantic and $\lambda > 0$ is a scalar to measure how fast the semantic varies along the change of distance d.

This direction vector d can be added or subtracted from the latent code of a face to manipulate its attributes.

When the case involves m different semantics, the equation (3) gets modified to:

$$s \equiv f_s(g(\boldsymbol{z})) = \Lambda\boldsymbol{n}^T\boldsymbol{z} \qquad (4)$$

where $s = [s_{1,}s_2 \dots \dots s_m]^T$ denotes the semantic scores, $\Lambda = (\lambda_{1,}\lambda_2 \dots \dots \lambda_m)$ is a diagonal matrix containing the linear co-efficient and N = $[n_{1,}n_2 \dots \dots n_m]$ indicates the separation boundaries.

The authors of InterFaceGAN also introduced the concept of semantic boundary, which is a set of vectors in the latent space that correspond to specific facial attribute changes. These semantic boundaries (pose, smile, age, gender, eyeglass) can be used to explore and manipulate specific attributes in a more intuitive way. The main advantage of this model is easy exploration and manipulation of facial attributes without requiring any labeled training data.

3.5 Loss Functions

Loss functions are a key component of deep learning models that are used to quantify the error or difference between the predicted output of the model and the true output or ground truth. The goal of training a deep learning model is to minimize the value of the loss function, which indicates how well the model is able to make predictions on the training data. There are several types of loss functions that are commonly used in deep learning models, including:

1. Mean Squared Error (MSE) Loss: This loss function measures the average squared difference between the predicted output and the true output. It is commonly used for regression tasks where the output is a continuous variable.
2. Binary Cross-Entropy Loss: This loss function is used for binary classification tasks where the output is a binary value (0 or 1). It measures the difference between the predicted probability of the positive class and the true label.
3. Categorical Cross-Entropy Loss: This loss function is used for multi-class classification tasks where the output is a categorical variable with more than two classes. It measures the difference between the predicted probability distribution over the classes and the true label.

4. KL Divergence Loss: This loss function measures the difference between two probability distributions. It is commonly used in variational autoencoders and other generative models.
5. Adversarial Loss: This loss function is used in generative adversarial networks (GANs) to train the generator and discriminator networks. The generator tries to minimize the loss while the discriminator tries to maximize it. In addition to these commonly used loss functions, there are many other loss functions that are specific to certain types of models and tasks. The choice of loss function depends on the specific problem being addressed and the type of output that the model is expected to produce.

The thesis will discuss about different loss function. Namely:

1. Pixel-Wise Loss
2. LPIPS Loss
3. Identity Based Reconstruction

3.5.1 Pixel-Wise Loss

Pixel-wise loss is a type of loss function that is commonly used in image processing and computer vision tasks, such as image reconstruction, image restoration, and image super-resolution. The goal of pixel-wise loss is to measure the difference between the target image and the generated image at the pixel level, where each pixel represents a small unit of the image. Pixel-wise loss is calculated by computing the element-wise difference between the target image and the generated image, and then summing the absolute or squared values of these differences over all pixels in the image. The resulting value represents the overall pixel-wise difference between the target and generated images. The most commonly used pixel-wise loss functions are mean squared error (MSE) and mean absolute error (MAE). MSE is defined as the average squared difference between the target and generated pixel values, while MAE is defined as the average absolute difference between the target and generated pixel values. Other variants of pixel-wise loss include structural similarity (SSIM) and perceptual loss, which take into account the perceptual similarity between images rather than just pixel-wise differences. Pixel-wise loss is a simple and effective

way to measure the quality of generated images and to optimize the parameters of the generator network during training. However, it has some limitations, such as being sensitive to noise and artifacts in the images, and not capturing higher-level features or structures in the images. As a result, more advanced loss functions that incorporate perceptual and semantic information have been proposed to overcome these limitations and improve the quality of generated images.

$$L2 = \frac{1}{n}\sum_{i=1}^{n}\left(Y_i - \hat{Y}_i\right)^2 \qquad (5)$$

The error is defined as the difference between the predicted value $\hat{Y}$ and some ground-truth value Y.

3.5.2 LPIPS Loss

LPIPS[45] stands for Learned Perceptual Image Patch Similarity, which is a perceptual distance metric for images that is based on deep learning. LPIPS measures the perceptual similarity between two images by computing the distance between their feature representations learned by a pre-trained deep neural network, such as a VGG[49] network. LPIPS is a more advanced perceptual loss function compared to traditional perceptual loss functions, as it takes into account not only the global features of the images, but also the local features, such as edges, textures, and patterns. It does this by computing the distance between patches of the images, rather than the images as a whole. LPIPS has been shown to be more effective than traditional perceptual loss functions in various computer vision tasks, such as image super-resolution, style transfer, and image-to-image translation. It has also been used as an evaluation metric for generative models, such as GANs, to measure the perceptual quality of their generated images. Overall, LPIPS is a powerful tool for measuring the perceptual similarity between images, and it has the potential to improve the quality and realism of generated images in various applications.

3.5.3 Identity Based Reconstruction

Identity-Based Reconstruction[27, 50] is a technique used in image processing and computer vision to generate an output image that is similar to a given input image, while preserving the identity or appearance of the input image. The goal of identity-based reconstruction is to generate a new image that has the same underlying structure and features as the input image, while making changes to its appearance, such as color, texture, and lighting. Identity-based reconstruction is often used in image editing and enhancement tasks, where the goal is to modify an image in some way while preserving its underlying structure and content. For example, in face editing tasks, identity-based reconstruction can be used to modify the appearance of a person's face, such as changing their hair color or adding makeup, while ensuring that the resulting image still looks like the original person. There are several approaches to implementing identity-based reconstruction, including using neural networks and optimization-based methods. Neural network-based methods typically involve training a deep neural network to learn a mapping between input and output images, where the network is trained to minimize a loss function that measures the difference between the input and output images. Optimization-based methods, on the other hand, involve solving an optimization problem to find the output image that best matches the input image while satisfying certain constraints. Identity-based reconstruction is a challenging task, as it requires balancing the need to preserve the identity of the input image with the desire to make changes to its appearance. However, with the recent advances in deep learning and computer vision, identity-based reconstruction has become an increasingly important tool in image processing and editing applications.

The Restyle-e4e encoder leverages a similar set of loss functions as those used for training StyleGAN2 encoders, consisting of a weighted combination of the L_2 pixel-wise loss, the LPIPS[45] perceptual loss, and an identity-based reconstruction[27, 50]
The overall loss objective is given by:

$$L(x) = \lambda_{l2} L_2(x) + \lambda_{lpips} L_{LPIPS}(x) + \lambda_{id} L_{id}(x) \quad (6)$$

Where $\lambda_{l2} = 1$, $\lambda_{lpips} = 0.8$ and $\lambda_{id} = 0.1$

3.6 Generate 3D Face Model and Texture Maps using 3DDFA

To generate 3D model and texture in 3DDFA, the texture mapping process goes through the following steps:

1. NCC - Normalized Coordinate Code
2. PNCC - Projected Normalized Coordinate Code
3. PAC – Pose Adaptive Convolution

3.6.1 Normalized Coordinate Code (NCC)

NCC is a measure of similarity between two images, computed by comparing their pixel values using a sliding window approach. It is often used in template matching and feature tracking tasks, where the goal is to find a specific pattern or object in an image. NCC has shown results when introduced robust to illumination changes and geometric transformations, but it can be computationally expensive and sensitive to noise

The novel type of vertex index is a crucial factor for introducing the image-view feature. Normalizing the 3D mean face to 0-1 in x, y, z axis gives:

$$NCC_d = \frac{\bar{S}_d - \min(\bar{S}_d)}{\max(\bar{S}_d) - \min(\bar{S}_d)} \quad (d = x, y, z) \qquad (7)$$

The Normalized Coordinate Code (NCC), shown in Figure 3.5, is a vertex index that allows each vertex's 3D coordinate, after normalization, to be uniquely distributed between [0,0,0] and [1,1,1]. This code is derived from the mean shape of the 3DMM, denoted as $\bar{S}$. NCC has three channels as RGB and can be treated as face texture ($NCC_x = R, NCC_y = G, NCC_z = B$).

3.6.2 Projected Normalized Coordinate Code (PNCC)

PNCC stands for Projected Normalized Coordinate Code, and it is a texture representation used in the 3DDFA (3D Dense Face Alignment) method for generating 3D face models. It encodes the normalized 3D coordinates of each pixel on the face surface in a compact form, which can be used to

reconstruct the 3D face shape and texture. PNCC is computed by projecting the 3D face model onto a 2D plane and then normalizing the coordinates with respect to the face size and orientation.

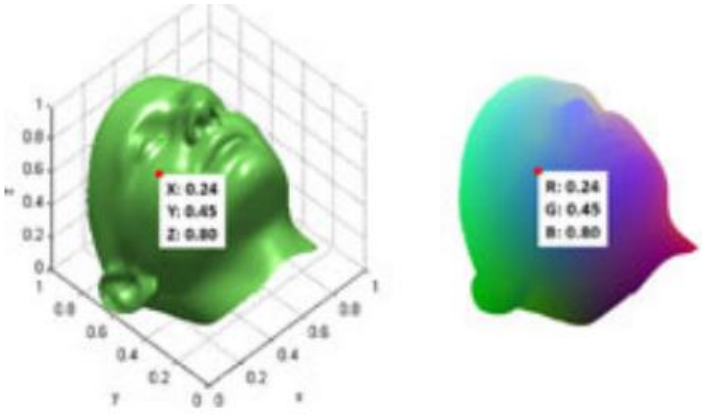


Figure 3.5: Normalized coordinate code (NCC) as described in [6]

During the fitting process, the projected 3D mesh $\boldsymbol{v}$ is rendered using $Z-Buffer(\boldsymbol{v},\tau)$ colored by τ with a model parameter $\boldsymbol{p}$ as in

$$PNCC = Z-Buffer(V_{3d}(\boldsymbol{p}), NCC) \quad (8)$$

$$V_{3d}(\boldsymbol{p}) = R * \left(\bar{S} + A_{id}\alpha_{id} + A_{exp}\alpha_{exp}\right) + [t_{2d}, 0]^t \quad (9)$$

Where, $V_{3d}(\boldsymbol{p})$ is the projected 3D face, $\bar{S}$ is the mean face, principal axes A_{id} and shape parameter α_{id} which are trained on 3D facial scans. The principal axes A_{exp} trained on offset between expression and neutral scan and expression parameter a_{exp}.The rendered image is known as Projected Normalized Code (PNCC), as shown in Figure 3.6.

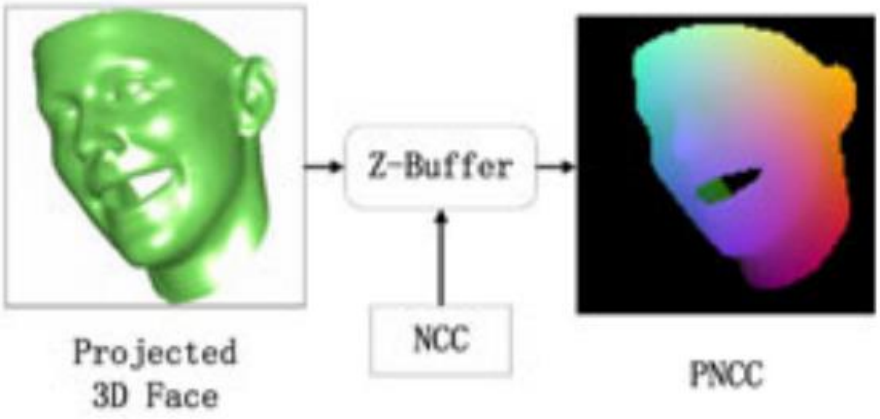


Figure 3.6: Projected normalized coordinate code (PNCC) as described in [6]

PNCC has been shown to be more effective than other texture representations such as PCA-SIFT [51] and LBP [52] in various experiments.

3.6.3 Pose Adaptive Convolutions

PAC, on the other hand, is a technique used to improve the quality of the texture generated by PNCC. PAC works by adding an additional projection step to the PNCC pipeline. Taking into account that the shape of a human face can be approximated to that of a cylinder[53], computing the cylindrical coordinates of each vertex and then sample 1024×1024 feature anchors using constant azimuth and height intervals as shown in Figure 3.7(a).

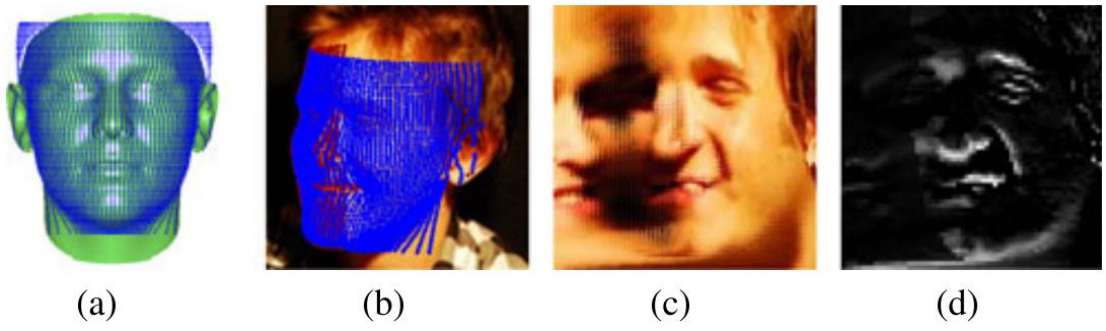


Figure 3.7: Pose adaptive convolution (PAC) as described in [6]

The feature anchors are projected and sampled on the image plane, resulting in 1024×1024×2 feature anchors represented by $V(\boldsymbol{p})_{anchor}$, as

shown in Figure 3.7(b). A patch of size $d \times d$ (taken d=5) is cropped at each feature anchor and the patches are concatenated into a patch map with dimensions$(1024 * d) \times (1024 * d)$, based on their respective cylindrical coordinates, as illustrated in Figure 3.7(c). Finally, a convolution filter of size $d \times d$ and stride d is applied to the patch map, generating 1024x1024 response maps, as depicted in Figure 3.7(d).

4. Experiments and Evaluation

In the evaluation and experiment section of this thesis, diverse range of images employed to facilitate the generation of multi-view images using StyleGAN3 and the subsequent generation of UV maps and 3D models utilizing 3DDFA. This section encompasses a comprehensive exploration of the data utilized for our research, highlighting the methodologies employed to attain meaningful insights and results.

Furthermore, the experimentation entails leveraging the capabilities of StyleGAN3, a state-of-the-art generative model, to generate multi-view images. These images, synthesized by StyleGAN3, offer a unique perspective and enable us to explore the interplay between visual aesthetics and the generation of multi-view representations.

Additionally, the power of 3DDFA, shows a highly advanced 3D Dense Face Alignment model, to estimate accurate 3D face shapes from the generated images. By harnessing the capabilities of 3DDFA, precise UV maps and construct detailed 3D models can be generated, thereby enhancing our ability to analyze and understand the underlying facial geometry and texture representation.

Through this evaluation and experimentation, this thesis aims to provide a comprehensive analysis of the performance and capabilities of the proposed methodology.

This section is divided in the following parts:

1. Hardware and Environment
2. Dataset- Flickr-Faces High Quality Dataset (FFHQ)
3. Embedding the Image to Latent Space
4. Multi-View Synthesis using InterFaceGAN
5. Generate Texture Map and 3D Model using 3DDFA
6. Evaluation Metric for StyleGAN3 Generated Images
7. Evaluation Metric for Generated UV Map
8. Evaluation 3D Face Mesh and Texture using Hardware
9. Subjective Evaluation

Figure 4.1 shows the 3D-GANTex pipeline that surrounds the proposed methodology.

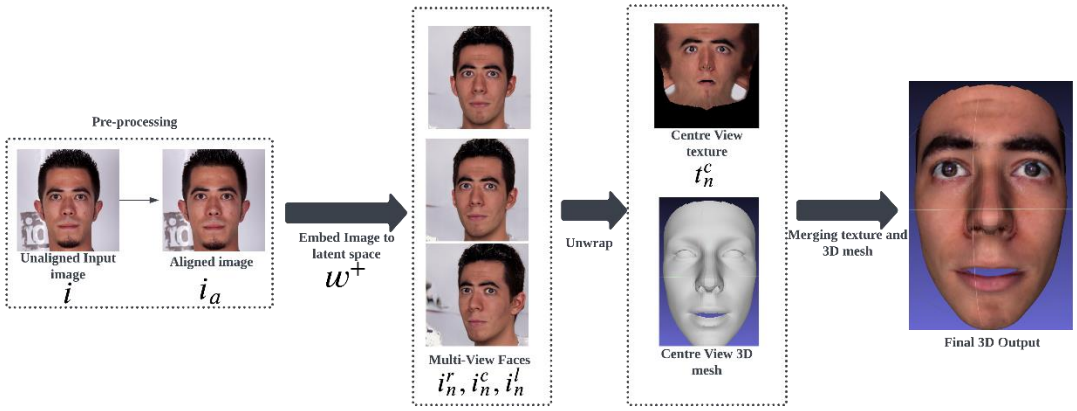


Figure 4.1: The proposed pipeline for 3D-GANTex producing frontalized 3D mesh with UV texture from a single in the wild image

4.1 Hardware and Environment

The proposed architecture uses the following hardware and software dependencies:

1. X64 based operating system and architecture
2. 12th Gen Intel(R) Core (TM) i7-12700F (20 CPUs) ~2.1 GHz
3. 64.0 GB RAM
4. NVIDIA GeForce RTX 4090 (24 GB VRAM)

4.2 Dataset- Flickr-Faces High Quality Dataset (FFHQ)

FFHQ[10] is a high-quality dataset of human faces created by scraping images from Flickr and filtering them for quality and diversity. The dataset includes over 70,000 images of faces with high-resolution and high-quality annotations as shown in Figure 4.2. It is commonly used in computer vision and deep learning research, particularly in the field of generative models such as GANs for generating realistic human faces. The dataset has become a benchmark for evaluating the performance of generative models and has led to the development of several state-of-the-art models such as StyleGAN[10], StyleGAN2[9] and StyleGAN3[8].

The thesis ignores baby face.

Figure 4.2: FFHQ Dataset[2]

4.3 Embedding the Image to Latent Space

As discussed in Section 3.2.2 since e4e[26] allows for more precise control over the generated images in latent space $W+$ and Restyle encoder[46] which provides improved image quality and flexibility.
Following are the procedures of embedding $Restyle_{e4e}$ encoder in StyleGAN3:

1. Install the required libraries and dependencies, including PyTorch, torchvision, and other necessary packages.
2. Download and extract the StyleGAN3 model and the pre-trained Restyle-e4e encoder from their respective repositories.
3. Load the pre-trained Restyle-e4e encoder and the StyleGAN3 generator model into Python script.
4. Load the input image to embed into the latent space of StyleGAN3 using the Pillow or OpenCV library.
5. Align the image using dlib to ensure that it is in the correct size and format required by the Restyle-e4e encoder (1024×1024).

[2] https://github.com/NVlabs/ffhq-dataset

6. Get the facial landmarks from dlib.
7. Pass the preprocessed image through the Restyle-e4e encoder to obtain the latent code that represents the image in the latent space.
8. Generate an image from the obtained latent code using the StyleGAN3 generator model.

Figure 4.3 shows the process for further reference.

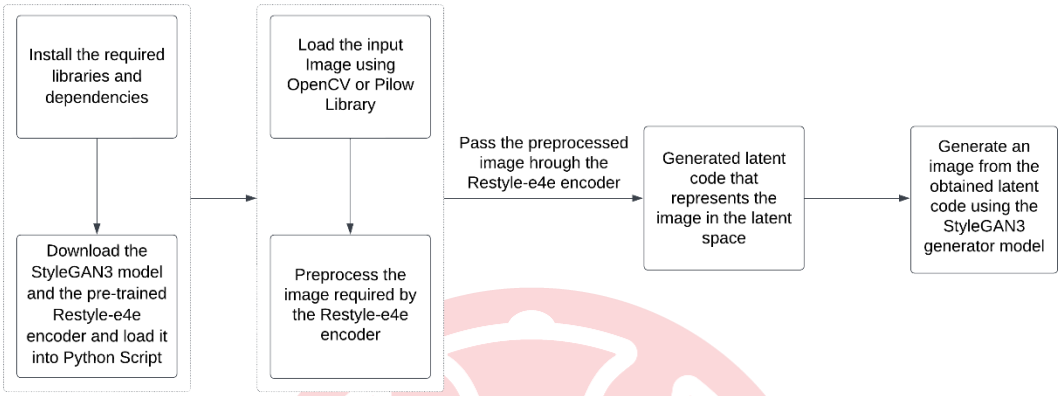


Figure 4.3: Embedding Image in latent space using Restyle-e4e encoder and StyleGAN3

4.4 Multi-View Synthesis using InterFaceGAN

InterFaceGAN[54] is a technique that allows manipulating specific features of an image by modifying the corresponding dimensions in the latent space of a pre-trained StyleGAN model . To generate poses using InterFaceGAN in StyleGAN3, the following steps are followed:

1. Load the pre-trained InterFaceGAN model: We can download the pre-trained InterFaceGAN model from the official GitHub repository. We need to load the model into memory and set it to evaluation mode.
2. Load the latent code from Restyle or the e4e encoder
3. Select the dimension to modify: select the dimension(s) in the latent space that corresponds to the pose feature we want to modify.
4. Modify the dimension by changing their values: The magnitude of the adjustment will affect the extent of the pose change.

5. Generate the output image: We can generate the output image by passing the modified latent code to the StyleGAN3 model.

Figure 4.4 shows the architecture of using InterFaceGAN to generate Multi-View Images.

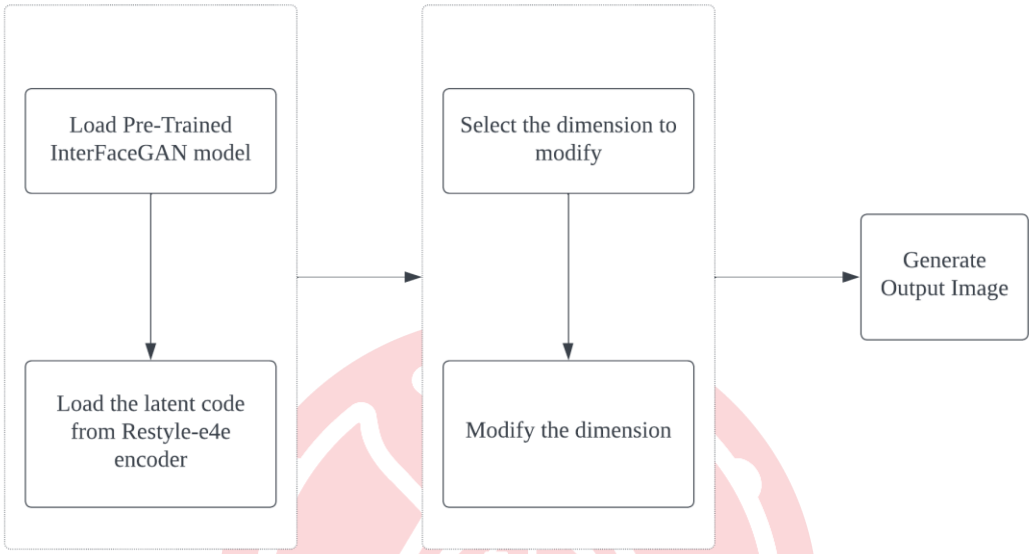


Figure 4.4: Multi-view synthesis using InterFaceGAN

4.5 Generate Texture Map and 3D Model using 3DDFA

To generate a texture map from 3DDFA, the thesis follows these below mentioned steps as shown in Figure 4.5 :

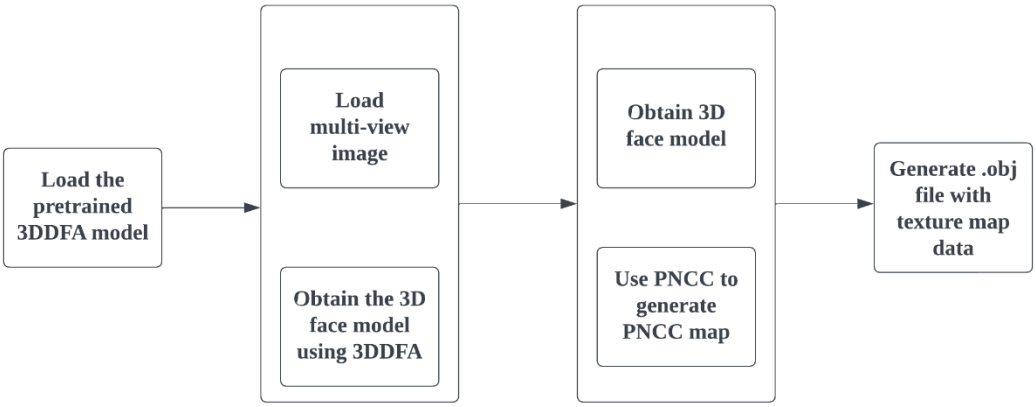


Figure 4.5: Texture generation using 3DDFA

Here are the steps in more detail:

1. Install the 3DDFA library and download the pre-trained model.
2. Load the input image and use the 3DDFA library to extract the 3D face model.
3. The 3D face model consists of 3D coordinates of the facial landmarks and the mesh connectivity.
4. Compute the face normal for each triangle in the mesh using the cross product of the triangle edges.
5. Compute the normal consistency for each vertex by averaging the normals of the triangles that share the vertex.
6. Project the normals to the image plane and normalize them to obtain the PNCC map.
7. Load the input image and the PNCC map.
8. Create a mask of the face region using the PNCC map.
9. Compute the gradient of the input image and the PNCC map.
10. Solve a Poisson equation using the gradient and the mask to obtain the texture map.
11. Merge Texture map and .obj file together
12. After generating 3D face model, add the texture map x and y value simultaneously
13. Save the.obj file

4.6 Evaluation Metric for StyleGAN3 Generated Images

There are several metrics that can be used to compare the structure and features between two images:

A. Structural Similarity Index (SSIM)

B.

C. Feature Similarity Index (FSIM)

D. Perceptual Loss

E. Multiscale Structural Similarity Index (MS-SSIM)

4.6.1 Structural Similarity Index (SSIM)

SSIM [55] stands for Structural Similarity Index. It is a widely used metric to measure the similarity between two images. The SSIM index takes into

account three factors: luminance, contrast, and structural content.
Luminance refers to the brightness of the image. Contrast measures the difference between the bright and dark regions of the image. Structural content refers to the patterns and textures in the image. The SSIM index calculates the similarity between the two images by comparing these three factors.
SSIM index produces a value between -1 and 1, where a value of 1 means the two images are identical, and a value of -1 means the two images are completely dissimilar. In practice, the SSIM index is often used to compare a generated image with a reference (ground truth) image, where a higher SSIM score indicates that the generated image is more similar to the reference image. The SSIM index has been shown to correlate well with human perception of image quality, and therefore, it is widely used in image and video processing applications.

4.6.2 Feature Similarity Index (FSIM)

Feature Similarity Index (FSIM) [56] is a metric for evaluating the quality of images or image processing algorithms. It is designed to capture the perceptual similarity between two images by measuring the similarity of their feature representations. Specifically, it computes the similarity between the local luminance, contrast, and structural information of the two images, using a set of pre-defined feature maps. The feature maps are obtained by applying a bank of filters to the input images, followed by a pooling operation to reduce the dimensionality of the feature space.
FSIM index produces a value between 0 and 1, where a value of 1 indicates that the two images are identical in terms of their feature representations. The FSIM index has been shown to outperform several other image quality metrics, including the popular Structural Similarity Index (SSIM), in terms of its correlation with human perception of image quality.

4.6.3 Perceptual Loss

Perceptual loss [57] is a type of loss function used in machine learning and deep learning models to measure the difference or dissimilarity between two images based on their perceptual qualities. It aims to capture the perceptual similarity or dissimilarity between images by considering

higher-level features and characteristics rather than pixel-level differences. The key idea behind perceptual loss is to use a pre-trained neural network, often a convolutional neural network (CNN), as a feature extractor. The network is typically trained on a large dataset for a specific task, such as image classification or object recognition. By using this pre-trained network, the loss function can measure the perceptual difference between images based on their shared high-level features.

The unique aspect of perceptual loss is that it considers the semantic content, structure, and overall appearance of the images, rather than pixel-level discrepancies. By focusing on higher-level features, perceptual loss can capture more meaningful and perceptually relevant differences between images. This makes it well-suited for tasks like image style transfer, image generation, and image-to-image translation, where capturing perceptual similarity or dissimilarity is crucial.

In summary, perceptual loss measures the perceptual dissimilarity between images by leveraging pre-trained deep neural networks to extract high-level features. It provides a more perceptually meaningful and context-aware measure of difference compared to traditional pixel-level loss functions.

4.6.4 Multiscale Structural Similarity Index (MS-SSIM)

Multiscale Structural Similarity Index (MS-SSIM) [58] is an extension of the SSIM for image quality assessment. The MS-SSIM index is designed to address some of the limitations of the SSIM index, such as its sensitivity to scaling and the inability to capture the structural information at different scales.

The MS-SSIM index decomposes an image into multiple scales and measures the similarity between the corresponding scales of two images. The index is calculated by first applying a set of Gaussian filters to the images to generate a set of scales. Then, for each scale, the SSIM index is computed between the corresponding regions of the two images. Finally, the MS-SSIM index is calculated as the weighted geometric mean of the SSIM index values for all scales.

MS-SSIM index has been shown to outperform the SSIM index in terms of its correlation with human perception of image quality, particularly for

images that are compressed or distorted. The index is widely used in image and video processing applications, such as image compression, image restoration, and video quality assessment.

4.6.5 Evaluation with No Background

To ensure normalized evaluation and improve the understanding of how background data/noise affects the evaluation score, the background was manually removed using GIMP[3]. The results as shown in Figure 4.14 shows a negligible difference compared to Figure 4.8. So, to conclude background information doesn't affect the evaluation metric between 2 images.

shows the evaluation metric. The assessment is based on FFHQ Dataset, Gender (M/F), No Glasses (NG), In-The-Wild Images (ITW).

4.7 Evaluation Metric for Generated UV Map

There are several metrics that can be used to verify the quality of a single UV map image. Here are a few commonly used ones:

1. Pixel Density
2. Texture Density

4.7.1 Pixel Density

Pixel density of a UV map refers to the number of texels (texture pixels) per unit area of the UV map. The pixel density can have a significant impact on the quality of the final 3D model, as it affects the level of detail and texture resolution.

To calculate the pixel density of a UV map, the size of the UV map is determined in pixels and the size of the texture that it will be mapped to in the 3D model. The pixel density is then calculated by dividing the number of texels in the UV map by the area of the map in square pixels.

4.7.2 Texture Density

Texture density can be an important consideration when creating high-

[3] GIMP - GNU Image Manipulation Program

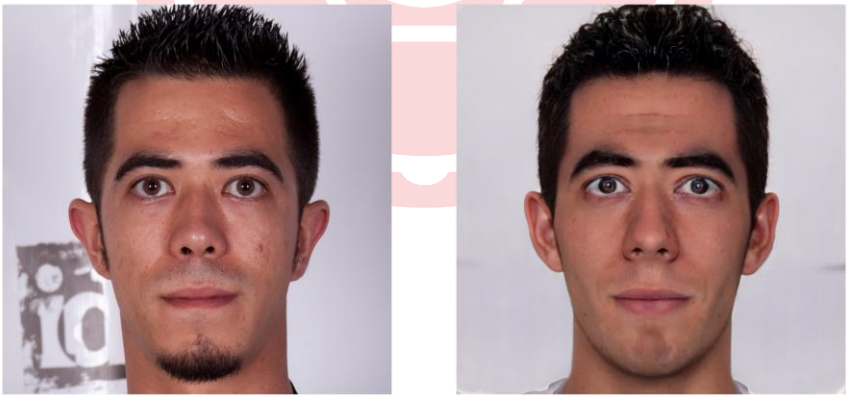
quality 3D models with detailed textures. It refers to the amount of texture detail applied to a specific region of the 3D model surface. A higher texture density means that there is more texture detail applied to a smaller surface area of the 3D model, resulting in a more detailed and realistic appearance. A balance needs to be struck between achieving high texture density in important areas such as the face or hands while not overwhelming the system with too much detail in less important regions.

To calculate the texture density of a UV map image, the Local Binary Patterns (LBP) feature can be used. LBP feature computes a texture descriptor for each pixel in the image based on the patterns of neighboring pixels, which can be used to measure the texture density of the image.

Table 4.1: Evaluation metric of UV-map generated from 3DDFA

	Pixel Density	Texture Density
UV-Map from 3DDFA	0.70	0.99

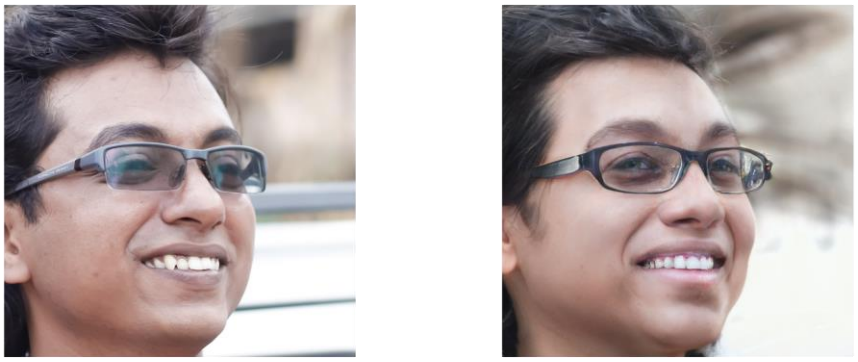
The range of value is [0, 1], and the higher the better.



Input Image Generated Image

SSIM-0.67
MS-SSIM-0.66
Perceptual Index-0.33
FSIM-0.72

Figure 4.6: Generated image from FFHQ (Male-No Glasses)



Input Image | Generated Image

SSIM-0.67
MS-SSIM-0.62
Perceptual Index-0.36
FSIM-0.78

Figure 4.7: Generated image from FFHQ (Male-Glasses)

Input Image | Generated Image

SSIM-0.60
MS-SSIM-0.62
Perceptual Loss-0.24
FSIM-0.75

Figure 4.8: Generated image from FFHQ (Female-No Glasses)

Input Image | Generated Image

SSIM-0.57
MS-SSIM-0.64
Perceptual Index-0.50
FSIM-0.74

Figure 4.9: Generated image from FFHQ (Female-Glasses)

Input Image | Generated Image

SSIM-0.50
MS-SSIM-0.27
Perceptual Index-0.84
FSIM-0.68

Figure 4.10: Generated image from In the Wild (Male-No Glasses)

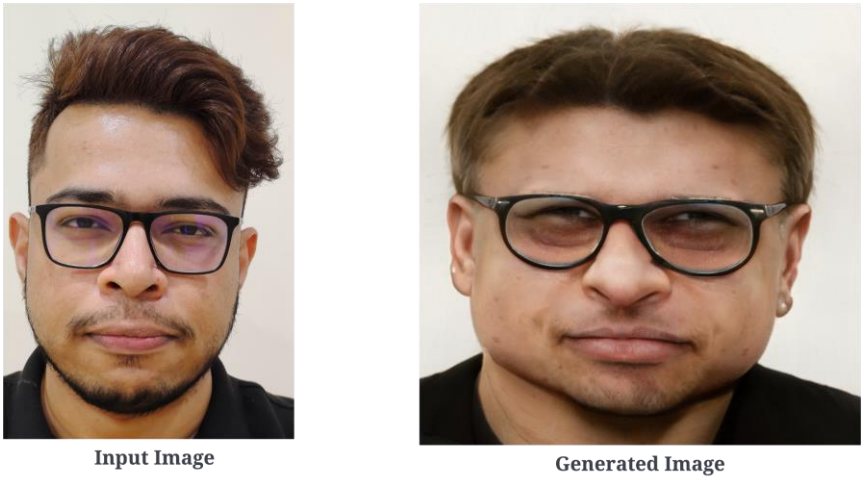


Figure 4.11: Generated image from In the Wild (Male-Glasses)

Figure 4.12: Generated image from In the Wild (Female-No Glasses)

Figure 4.13: Generated image from In the Wild (Female-Glasses)

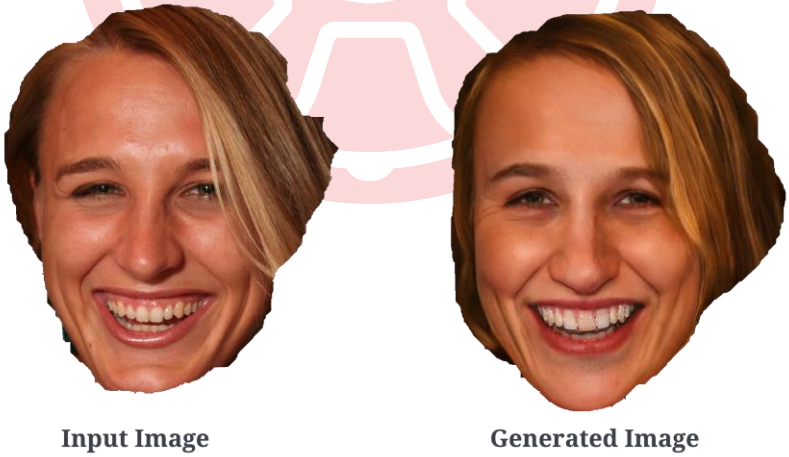
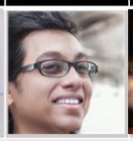
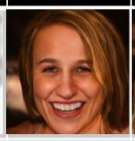
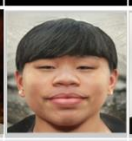
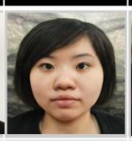


Figure 4.14: Generated image from In the Wild (Female-Glasses-No Background)

Table 4.2: Evaluation metric for images generated from StyleGAN3 using e4e encoder

	FFHQ M - NG	FFHQ M	FFHQ F - NG	FFHQ F	ITW M -NG	ITW M	ITW F - NG	ITW F
Figure								
SSIM ↑	**0.67**	**0.67**	0.60	0.57	**0.50**	0.49	**0.50**	0.44
MS-SSIM ↑	**0.66**	0.62	0.62	0.64	0.27	0.23	**0.36**	0.18
FSIM ↑	0.72	**0.78**	0.75	0.74	0.68	0.65	0.69	**0.71**
PI ↓	0.39	0.36	**0.24**	0.50	0.84	**0.48**	0.71	0.67

The range of value is [-1, +1]

4.8 Evaluation 3D Face Mesh and Texture using Hardware

Creality CR-Scan 01 as shown in Figure 4.15 is a 3D scanning device used to generate a 3D face model from a single 2D image. It employs structured light technology, where a pattern of light is projected onto the face, and the deformation of the pattern on the face's surface is captured by the scanner's cameras. This information is then used to reconstruct a 3D model of the face.

Advantages of using Creality CR-Scan 01 for 3D face modeling:

1. Single Image Input: Unlike some other methods that require multiple images or specialized equipment, Creality CR-Scan 01 can generate a 3D face model from just a single 2D image, making it convenient and accessible just like Error! Reference source not found..
2. Portable and Easy to Use: It is a handheld device, which makes it portable and suitable for various applications. It is user-friendly, allowing for easy scanning without requiring extensive technical expertise.
3. Time and Cost Efficient: With its efficient scanning process, Creality CR-Scan 01 can quickly capture facial details and generate

a 3D model, reducing the time and cost compared to more complex scanning methods.

Disadvantages of using Creality CR-Scan 01 for 3D face modeling:

1. Limited Accuracy: The accuracy of the generated 3D face model may be lower compared to more advanced and precise scanning technologies. Fine details and subtle facial features may not be captured as accurately.
2. Lighting and Environmental Dependencies: The quality of the 3D scan can be influenced by lighting conditions and the environment. Factors like shadows, reflections, or uneven lighting can affect the accuracy and reliability of the scan.
3. Limited Application Scope: It primarily designed for face scanning and may not be suitable for capturing other objects or larger scenes. It is optimized for facial scanning purposes and may not offer the versatility of specialized 3D scanners.

Figure 4.17 shows the 3D face mesh generated from 3DGANTex and Creality CR-Scan 01. Since the thesis emphasizes the use of in the wild images, the results are generated ignoring the lightning, depth and every other environmental factors. It is important to note that while CR-Scan 01 is capable of generating realistic texture as shown in Figure 4.18, the process can be time-consuming, taking approximately 20 minutes to complete. In contrast, the 3DGANTex model is significantly faster, requiring less than 45 seconds to generate comparable texture.

Figure 4.15: Creality CR-Scan 01[4]

Figure 4.16: Practical usage scenario of Creality CR-Scan 01

Figure 4.19 shows generated example of a female subject with eyes closed. The result though favors the high texture as seen in Figure 4.19, it is to be noted that the generated results from Creality took <10 minutes and more than 2000 images whereas 3DGANTex has done the same with near to high texture accuracy in <45 seconds.

[4] CR-Scan 01 3D Scanner Upgraded Combo (creality.com)

Table *4.3* shows the qualitative analysis between 3DGANTex with Creality CR-Scan 01. Although Creality CR-Scan has generated very high results, the process came with some sacrifices.

1. High Processing Time.
2. Does not work with Single Image.
3. The subject has a hard time to open their eyes due to huge strobe lights.
4. Need better illumination to get the intricate details.

Table 4.3: Qualitative analysis of 3DGANTex v/s Creality CR-Scan 01

	Visual Accuracy	Fast	Single - Image	Processing time	Texture Information
3DGANTex	✓	✓	✓	<45 seconds	High
Creality CR-Scan 01	✓	×	×	<10 minutes	Very High

Table 4.4 shows quantitative analysis between 3D mesh generated from 3DGANtex and Creality CR-Scan 01. The evaluation takes 2 metrics for analysis.

1. Triangles- In the context of a 3D mesh, triangles are the basic building blocks used to represent the surface geometry of a 3D object. It is defined by three vertices that are connected by three edges. They are often preferred in 3D modeling because they are simple and have desirable properties for rendering and computational operations.

   The use of triangles in 3D meshes allows for the efficient representation and manipulation of complex 3D surfaces. When a 3D model is rendered for display or used in simulations, the rendering engine or algorithms can work with the triangular mesh to calculate lighting, shading, and other visual effects accurately.

   A higher quantity of triangles means better quality 3D mesh and more realistic information.

2. Average Triangle Area - The average triangle area can be used as a metric to assess the quality of a 3D mesh. It influences the level of detail, surface accuracy, computational efficiency, and the ability to capture intricate surface features. Finding an optimal balance in triangle area is important to achieve a high-quality mesh that accurately represents the underlying object.

   Optimally, lower triangle area can be considered high quality 3D mesh.

Table 4.4: Quantitative Analysis of 3DGANTex v/s Creality CR-Scan 01

3D Mesh	No. of Triangles	Average Triangle Area from 50 Vertices (Lower the better)
3DGANTex	76,073	3.47
Creality CR-Scan 01	1,628,146	0.05

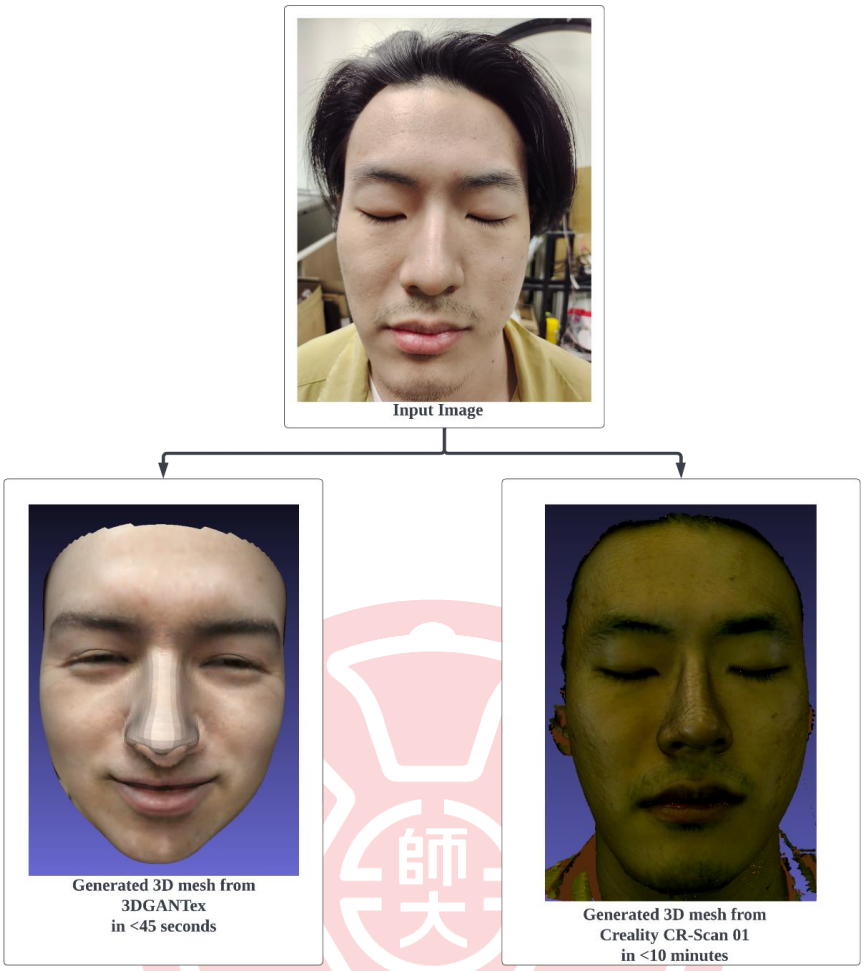


Figure 4.17: 3DGANTex v/s Creality CR-Scan 01(Male-Closed Eyes)

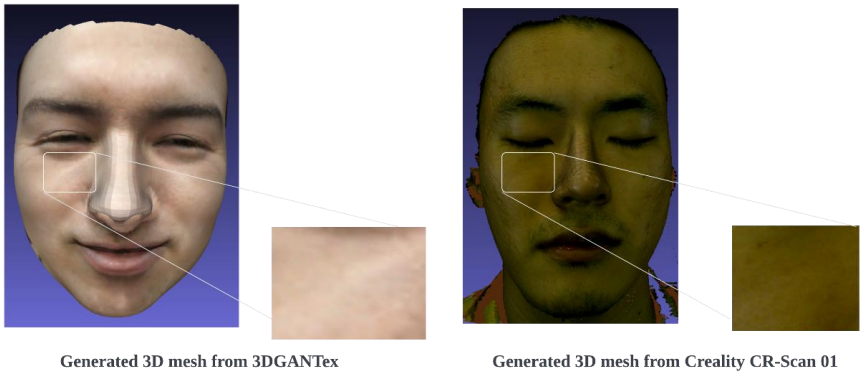


Figure 4.18: Texture difference between 3DGANTex v/s Creality CR-Scan 01 (Male-Closed Eyes)

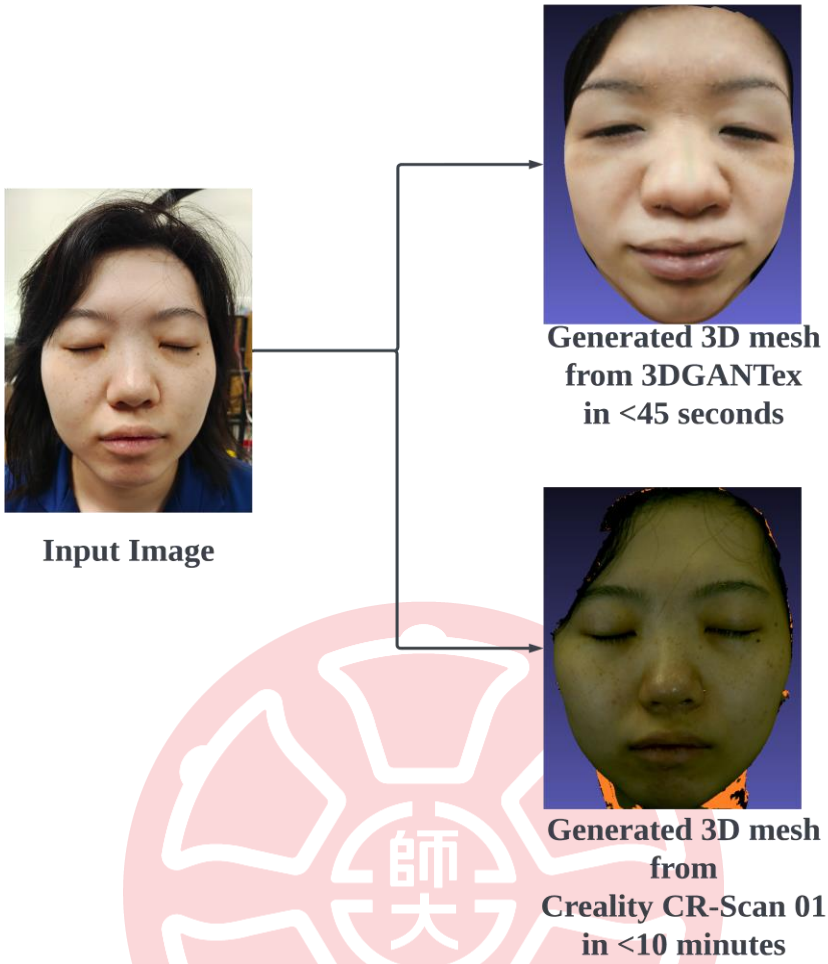


Figure 4.19: 3DGANTex v/s Creality CR-Scan 01(Female-Closed Eyes)

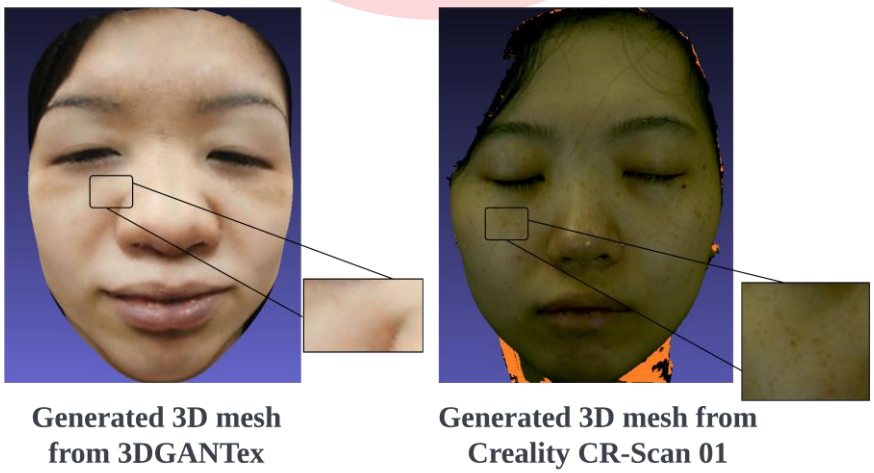


Figure 4.20: Texture difference between 3DGANTex v/s Creality CR-Scan 01 (Female-Closed Eyes)

4.9 Subjective Evaluation

Subjective evaluation refers to the assessment or judgment of something based on personal opinions, feelings, or experiences, rather than objective criteria or measurable data. It involves evaluating the quality, value, or performance of a particular subject from an individual's subjective perspective.

It is highly influenced by personal preferences, biases, and individual interpretations. It is often used in fields where subjective factors play a significant role, such as art, literature, music, film, and qualitative research. Subjective evaluations are typically contrasted with objective evaluations, which rely on measurable criteria, facts, or quantitative data. Objective evaluation aims to eliminate personal biases and provide an unbiased assessment of the subject matter.

Subjective evaluation of 3D mesh and texture maps involves assessing their quality, visual appeal, and overall effectiveness in conveying the desired aesthetics or realism in a three-dimensional model. This evaluation is based on subjective factors such as personal preferences, visual perception, and artistic judgment. Here are some aspects that may be considered during subjective evaluation:

1. Visual Quality: Evaluating the visual fidelity and level of detail in the 3D mesh and texture maps. This includes assessing the clarity of the geometry, smoothness of the surfaces, and the resolution of the textures.
2. Realism: Judging how effectively the mesh and texture maps represent real-world objects or scenes. This involves considering factors like accurate proportions, lifelike textures, and realistic lighting/shading effects.
3. Aesthetics: Assessing the overall artistic quality and visual appeal of the 3D model. This includes evaluating the design choices, color schemes, composition, and any artistic elements incorporated into the mesh or textures.
4. Consistency: Evaluating the consistency of the 3D mesh and texture maps. This involves examining whether the texture maps align

correctly with the mesh, maintain consistent levels of detail, and avoid noticeable distortions or seams.

5. Subjective Interpretation: Considering the subjective interpretation of the model, such as how well it conveys the intended message, evokes emotions, or tells a story. This aspect may vary greatly depending on the context and purpose of the 3D model.

Based on these principles, a short evaluation has been performed on verifying the quality of mesh and texture generated from 3DGANTex. The evaluation included 17 participants between the age of 23-45.

Table 4.5: Subjective evaluation questionnaire

Question No.	Question	Evaluation Metric	Evaluation Score
1	Do you think A and B are the same person?	Similarity	1: Strongly Disagree 2: Disagree 3: Neither Agree nor Disagree 4: Agree 5: Strongly Agree
2	C is the 3D model of B. Do you think C resembles the same person as B?	Similarity in 3D mesh with Generated Image	
3	Do you think A and C are the same person?	Similarity in 3D mesh with Ground Truth	
4	C is a 3D model. What do you feel about the texture of this image? Does it look realistic?	Texture	
5	D is a multi-view GIF of C. How closely does it resemble A	Multi-View	

Table 4.5 shows the evaluation criteria, used for the subjective evaluation as we can see in the sample images Figure 4.21 and Figure 4.22 .

Short Evaluation for 3DGANTex

Total points 0/0

First of all thank you for taking part in this experiment. This is a short assessment of the efficacy of 3DGANTex.
We have 8 sets of images and 5 Questions
Every example will have:
A= Input Image
B= Generated Image
C= 3D model of Generated Image (Image Format)
D= Multi-View GIF of Generated Image
Please go through the following questions after each set and answer them respectively.
The range is between 1(Strongly Agree) to 5(Strongly Disagree)
Questions can't be skipped
Thank you.

Name *

Rohit Das

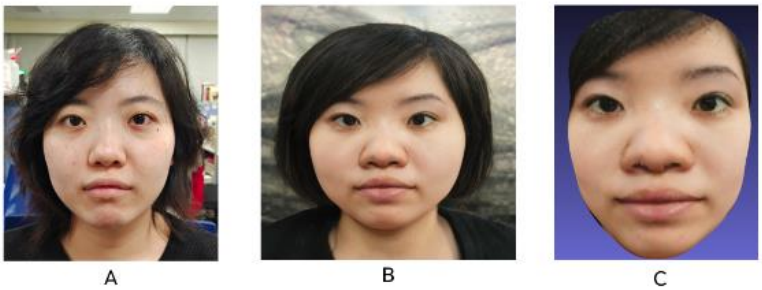
Email *

rdas.879@gmail.com

Age *

27

Female-No Glass-ITW



A B C

Figure 4.21: Subjective Evaluation for 3DGANTex (Sample)

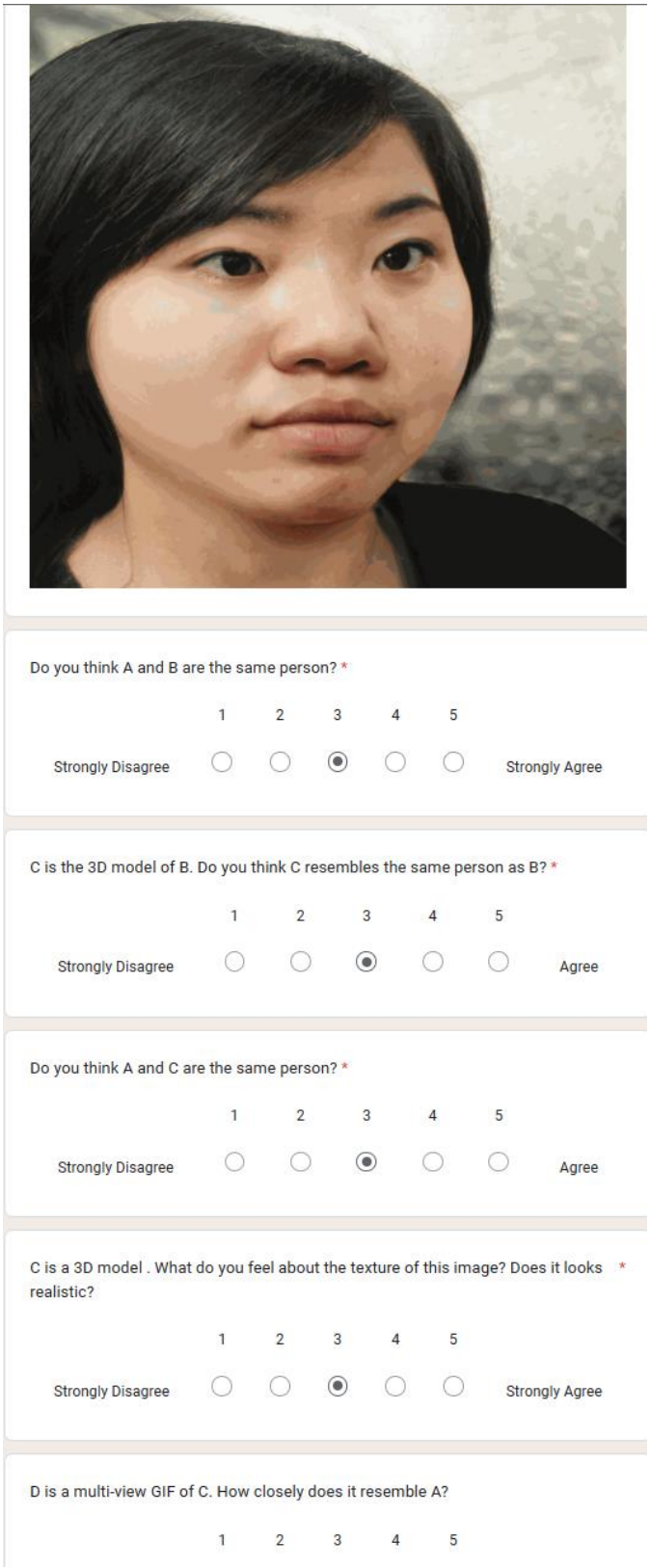


Figure 4.22: Subjective evaluation for 3DGANTex (Sample) Cont.

D is basically multi-view generated from the model and concatenating in

GIF format. So, it was not possible to insert into word format in this thesis for reference.
The evaluation metric is linear scale ranging from 1(Strongly Disagree) to 5(Strongly Agree).
Following are the insights of the data gathered from this evaluation:

A. Female -In-the-Wild No Glass

Figure 4.23 and Figure 4.24 shows the sample image for Female In-the-Wild no glasses and the evaluation results data visualization respectively.

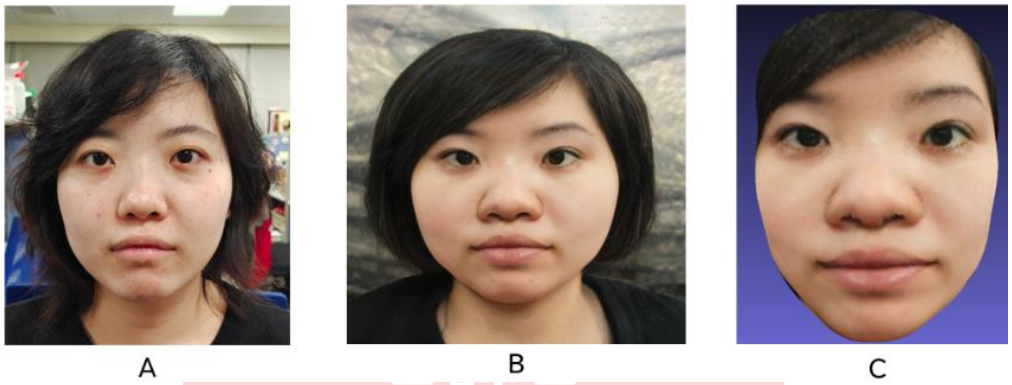


Figure 4.23: Female-In-the-Wild- No-Glass sample image for subjective evaluation

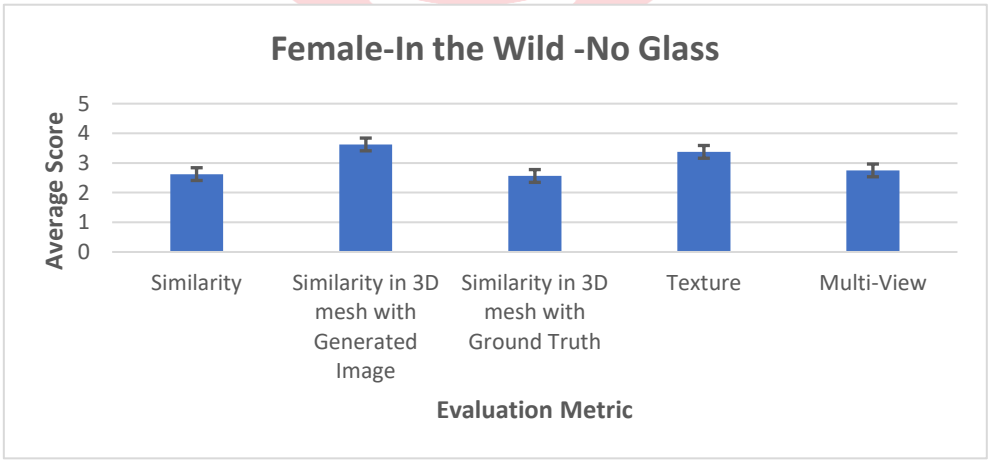


Figure 4.24: Subjective evaluation data visualization for Female-In the Wild-No Glass

B. Female-In-the-Wild-Glass

Figure 4.25 and Figure 4.26 shows the sample image for female in-the-wild with glasses and the evaluation results data visualization respectively.

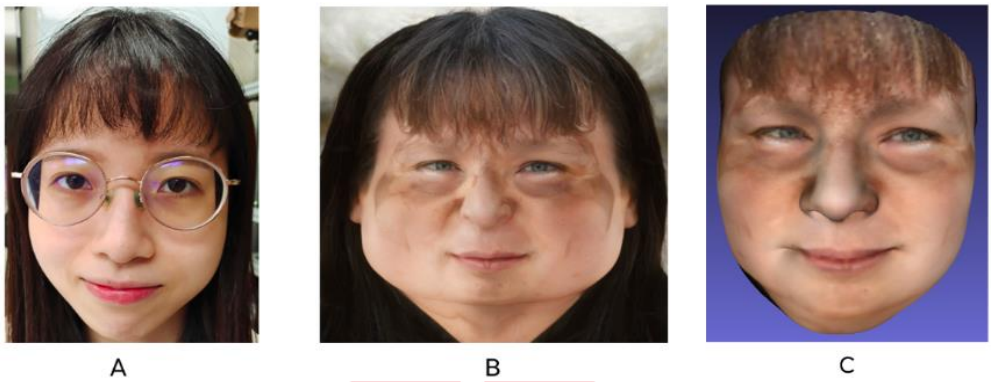


Figure 4.25: Female-In-the-Wild-Glass sample image for subjective evaluation

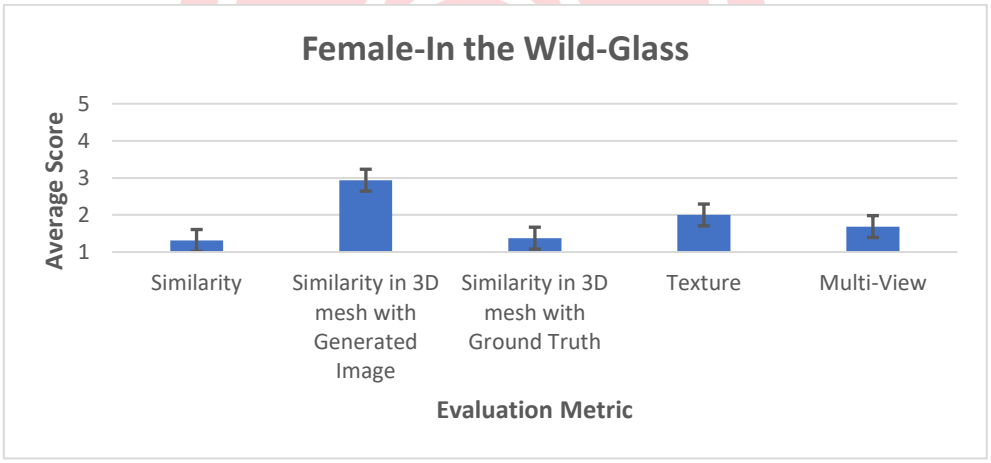


Figure 4.26: Subjective evaluation data visualization for Female-In the Wild-Glass

C. Female-FFHQ-Glass

Figure 4.27 and Figure 4.28 shows the sample image for female FFHQ with glasses and the evaluation results data visualization respectively.

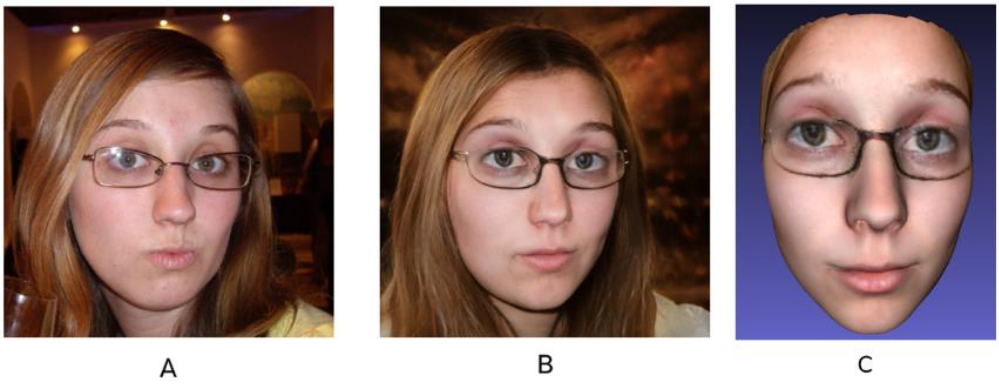


Figure 4.27: Female-FFHQ-Glass sample image for subjective evaluation

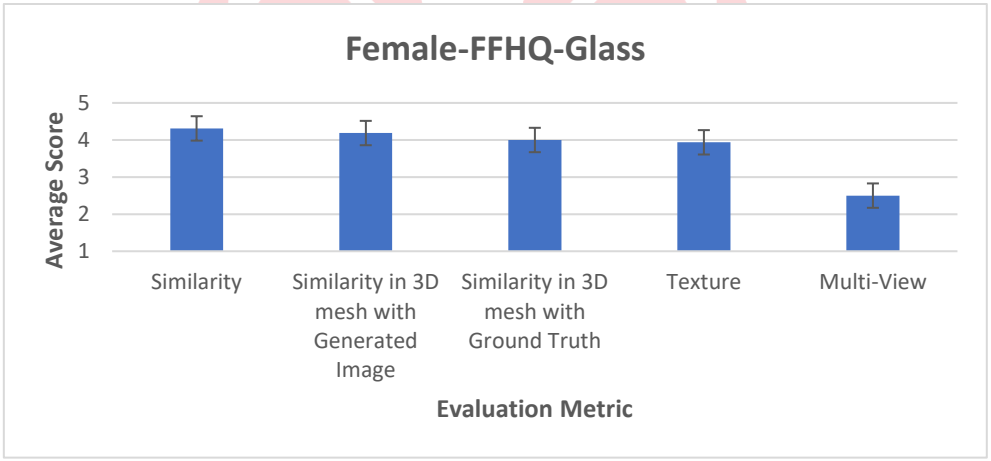


Figure 4.28: Subjective evaluation data visualization for Female-FFHQ-Glass

D. Female-FFHQ-No-Glass

Figure 4.29 and Figure 4.30 shows the sample image for female FFHQ with no glasses and the evaluation results data visualization respectively.

Figure 4.29: Female-FFHQ-No-Glass sample image for subjective evaluation

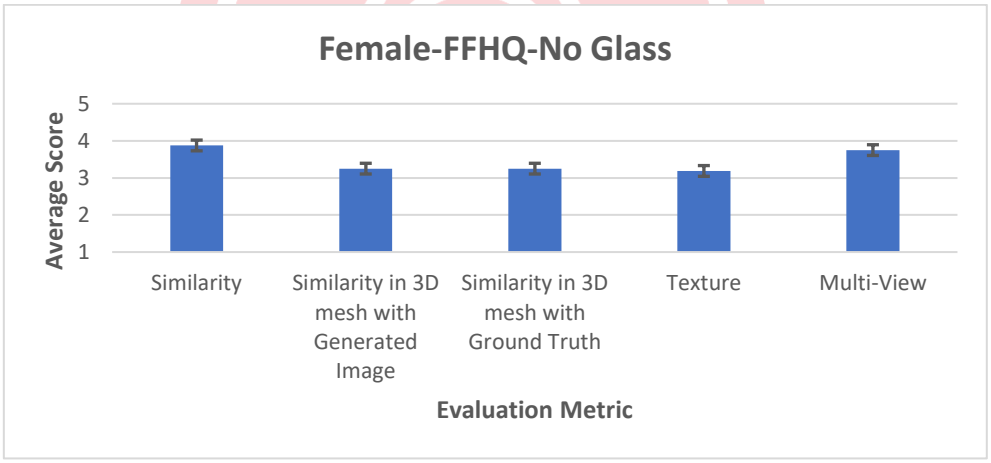


Figure 4.30: Subjective evaluation data visualization for Female-FFHQ-No-Glass

E. Male-FFHQ-Glass

Figure 4.31 and Figure 4.32 shows the sample image for male FFHQ with glasses and the evaluation results data visualization respectively.

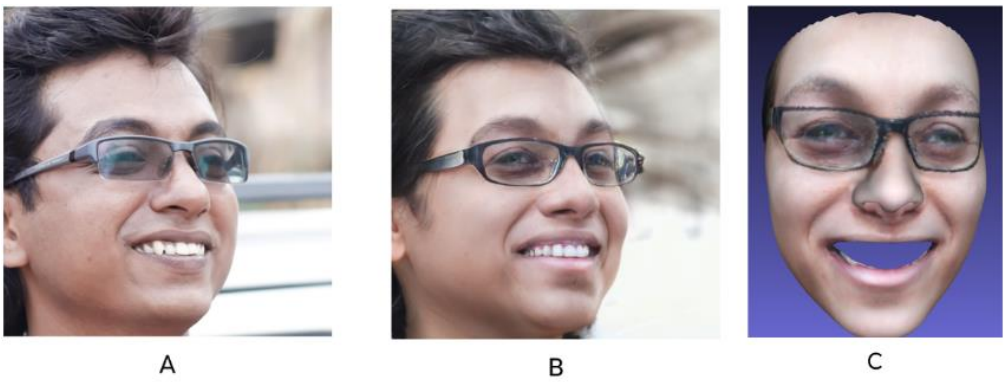


Figure 4.31: Male-FFHQ-Glass Sample Image for Subjective Evaluation

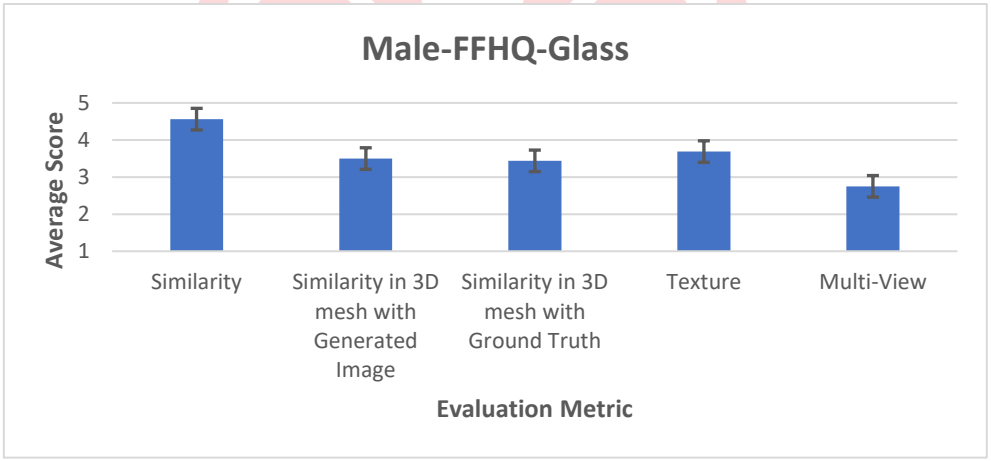


Figure 4.32: Subjective evaluation data visualization for Male-FFHQ-Glass

F. FFHQ-Male-No-Glass

Figure 4.33 and Figure 4.34 shows the sample image for male FFHQ with no glasses and the evaluation results data visualization respectively.

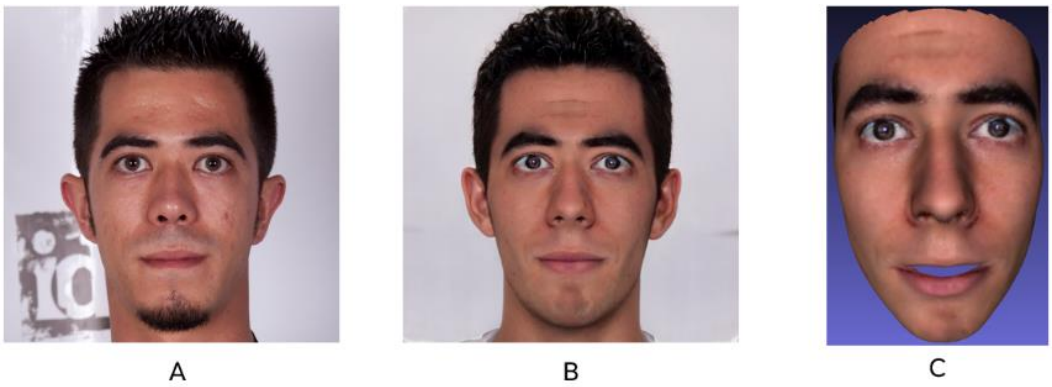


Figure 4.33: Male-FFHQ-No-Glass sample image for subjective evaluation

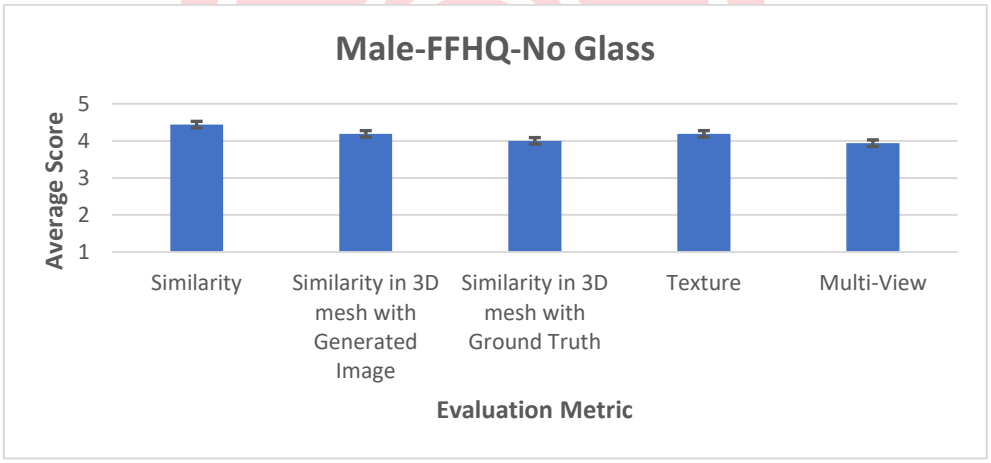


Figure 4.34: Subjective evaluation data visualization for Male-FFHQ-No-Glass

G. Male-In-the-Wild-Glass

Figure 4.35 and Figure 4.36 shows the sample image for male in-the-wild with glass and the evaluation results data visualization respectively.

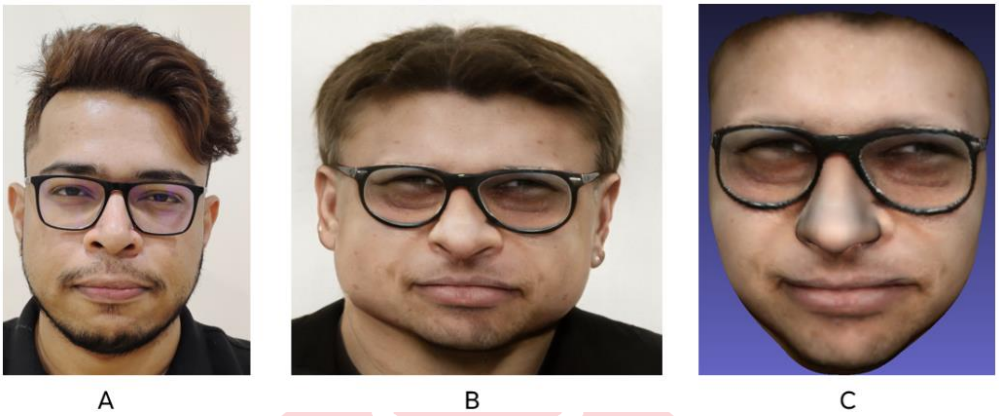


Figure 4.35: Male-In-the-Wild-Glass sample image for subjective evaluation

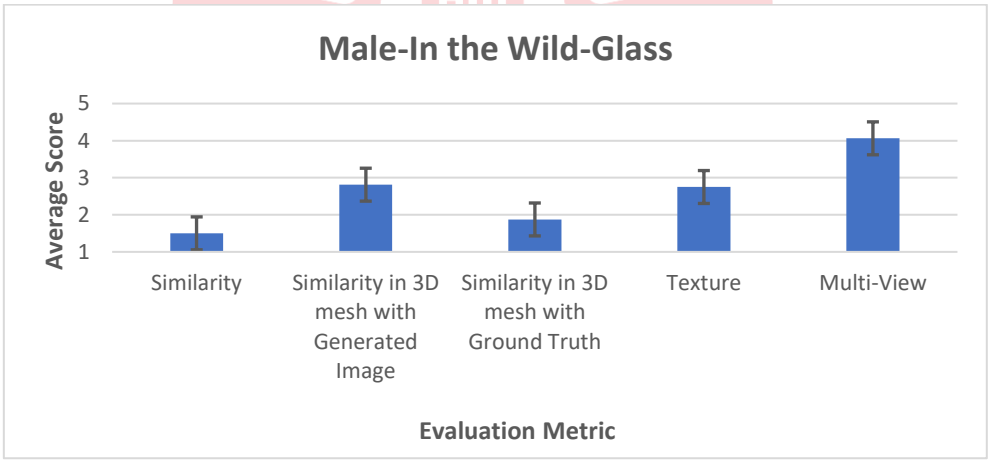


Figure 4.36: Subjective evaluation data visualization for Male-In-the-Wild-Glass

H. Male-In-the-Wild-No-Glass

Figure 4.37 and Figure 4.38 shows the sample image for male in-the-wild with no glass and the evaluation results data visualization respectively.

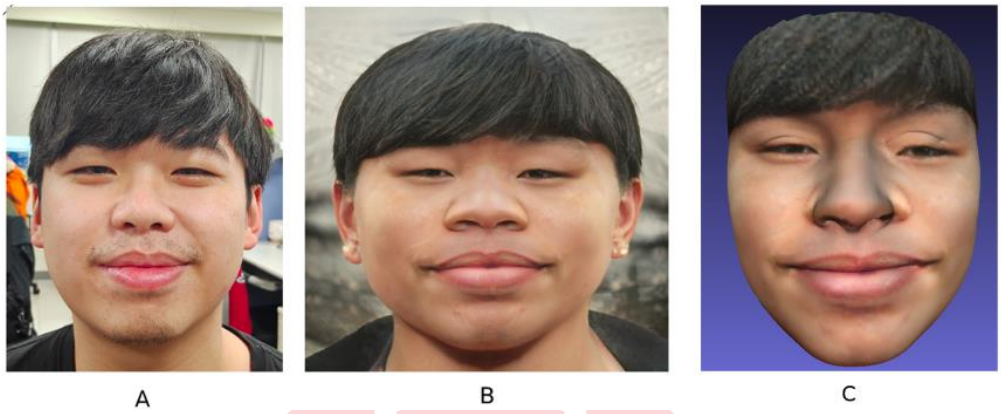


Figure 4.37: Male-In-the-Wild-No-Glass sample image for subjective evaluation

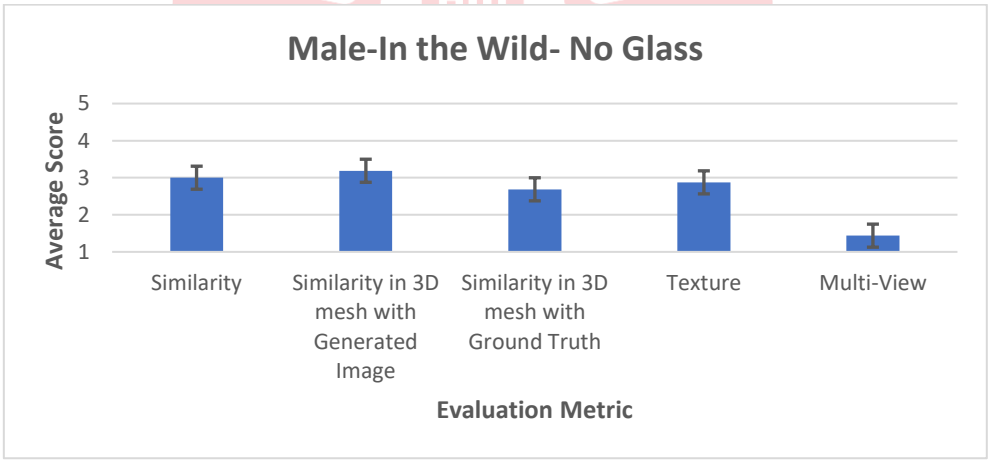


Figure 4.38: Subjective evaluation data visualization for Male-In-the-Wild-No-Glass

The evaluation results indicate that the generated results using FFHQ images were highly accurate and received positive support from the participants. This demonstrates the effectiveness of the proposed method in producing realistic and high-quality outputs. However, it is important to address the significant negative bias observed in the generated results for In-the-Wild images with glasses. This presents an opportunity for

further improvement in handling such scenarios and reducing the bias. On the other hand, when dealing with In-the-Wild images without glasses, the results showcased a more neutral and satisfactory performance. This suggests that the proposed approach is effective in generating realistic results for individuals without eyewear. It highlights the potential of our method to excel in scenarios where glasses are not present.

When evaluating the efficacy of 3DGANTex, the results as shown in Figure 4.39 and Figure 4.40 consistently favored the use of FFHQ sample images regardless of gender. This can be attributed to the training of StyleGAN3[8] on the FFHQ dataset, which provides a strong foundation for generating high-quality outputs.

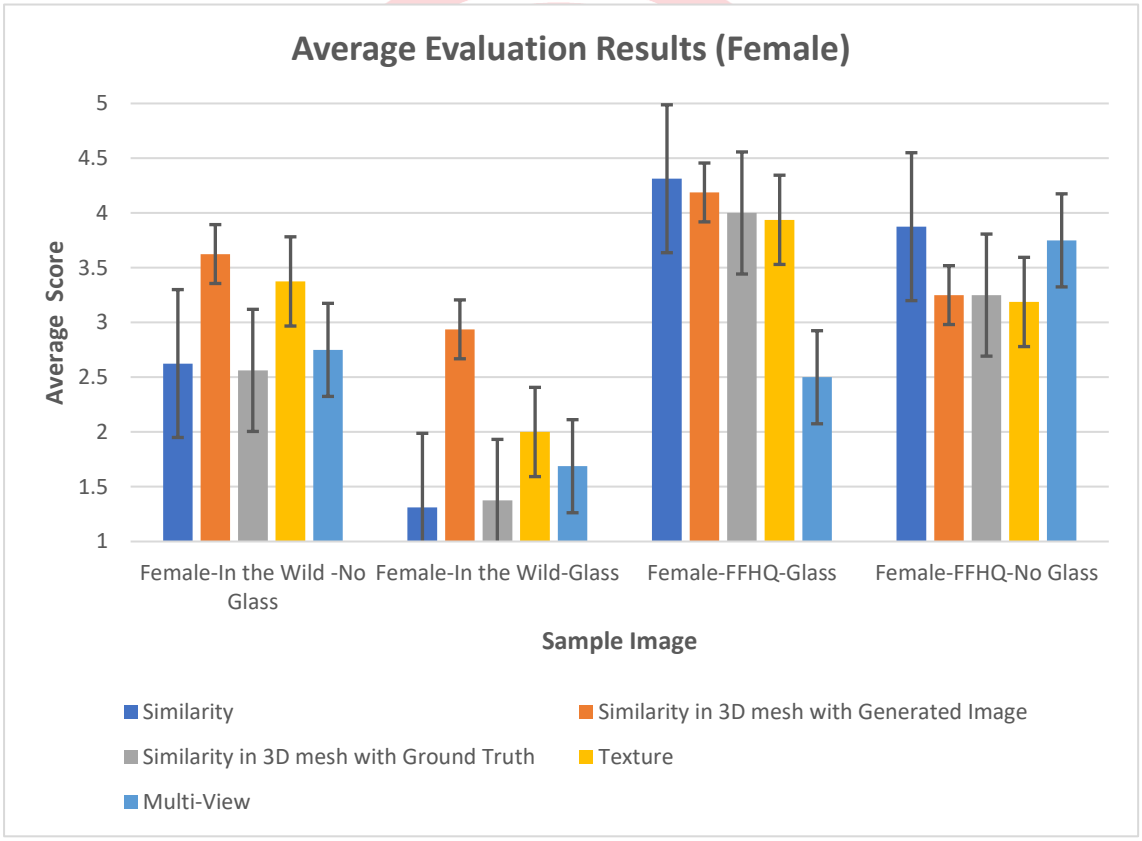


Figure 4.39: Subjective evaluation data visualization based on gender (Female)

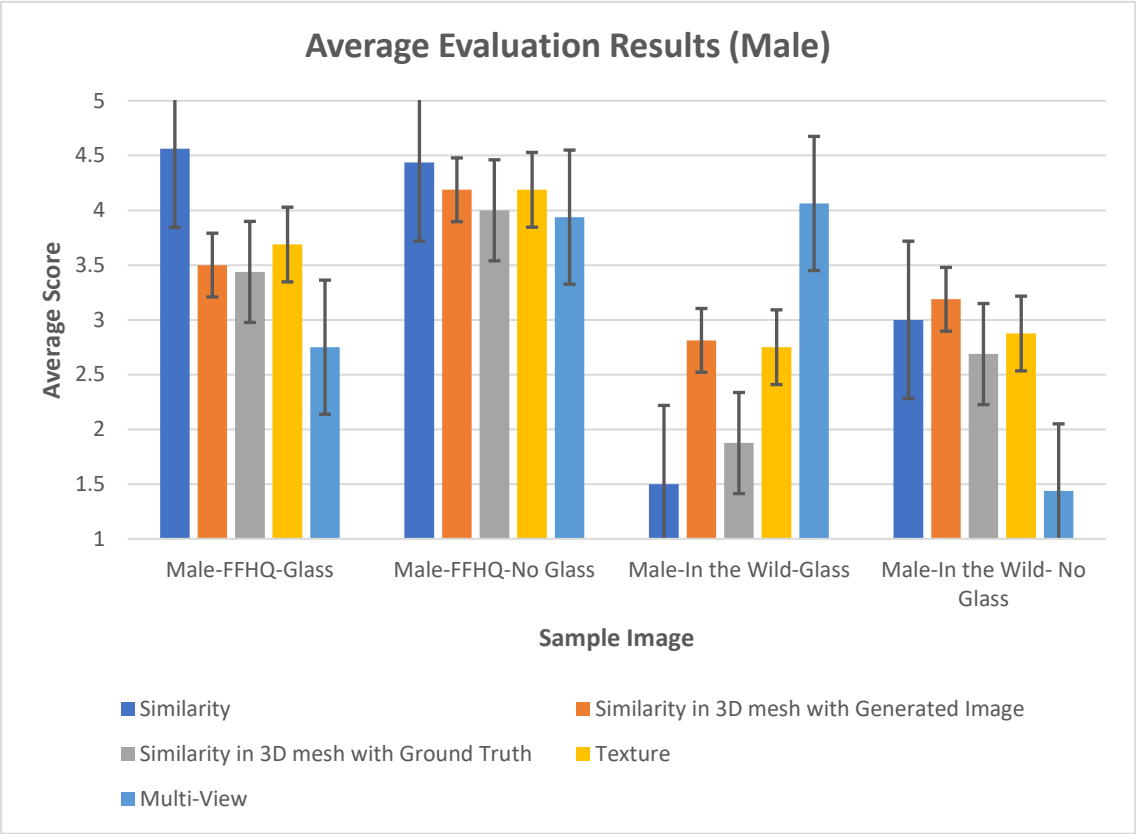


Figure 4.40: Subjective evaluation data visualization based on gender (Male)

5. Results and Discussions

The following section presents the findings obtained through the implementation of the methodology outlined in this thesis. It primarily focuses on three significant aspects:

1. Generating of multi-view images
2. Generating UV map
3. Creating 3D model utilizing 3DDFA

Within the results section, a detailed analysis and evaluation of the generated multi-view images, UV maps, and 3D models are presented. This assessment encompasses considerations of quality, accuracy, and practicality, accompanied by insightful discussions regarding the strengths, limitations, and potential applications of these generated outputs. In essence, this section provides a comprehensive examination and evaluation of the outcomes achieved through the application of StyleGAN3 for multi-view image generation, as well as the utilization of 3DDFA for UV map and 3D model generation.

5.1 Generation of Multi-View Images

Firstly, StyleGAN3, a state-of-the-art generative adversarial network, is used to generate "multi-view images" based on a single input image. This technique allowed to synthesize a set of diverse views of the subject, providing a comprehensive representation of the object from various perspectives. These multi-view images serve as a valuable resource for subsequent analysis and visualization. By observing Figure 5.1, it is evident that the e4e encoder exhibits superior feature representation capabilities compared to pSp. The e4e encoder effectively captures and preserves important visual attributes and details of the input images which is beneficial to us to generate multi-view as seen in Figure 5.2 . On the other hand, the pSp encoder excels at preserving the identity of the subjects, ensuring that the generated images maintain the distinct characteristics of the individuals but loses a lot of features as seen in Figure 5.3. The comparison between these two encoders highlights their respective

strengths and indicates their suitability for different purposes in image generation and manipulation tasks. Since the end results of our method is 3D model, this thesis prioritizes e4e encoder.

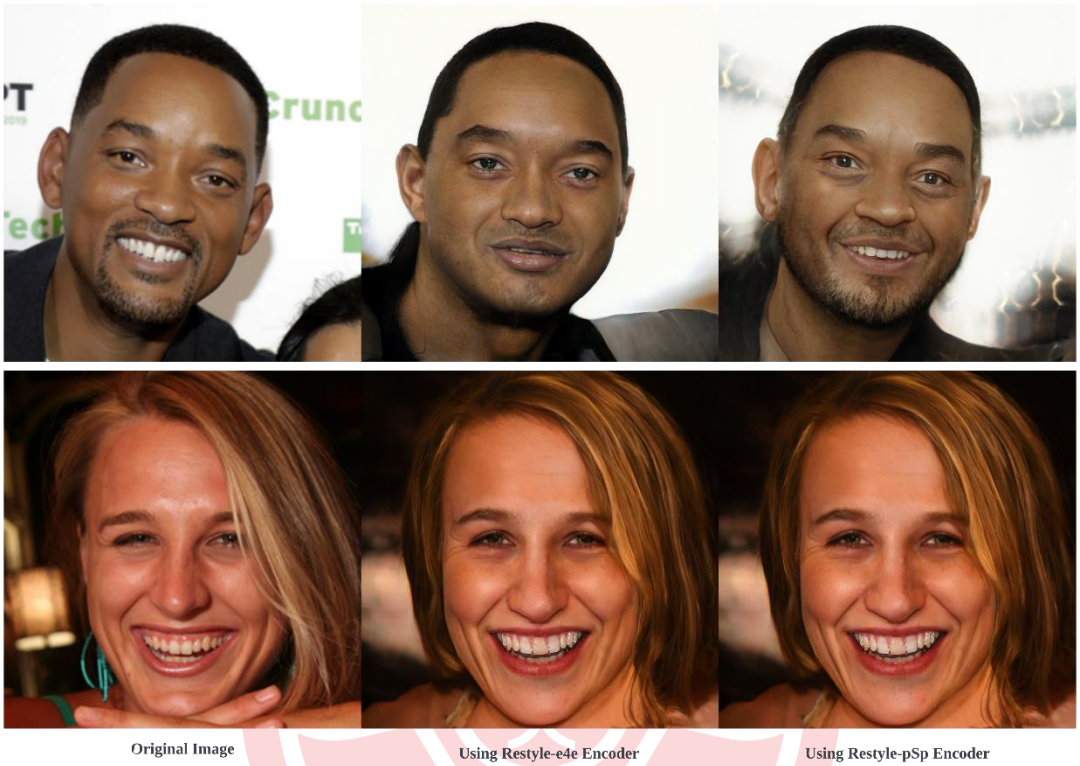


Figure 5.1: Embedding image to latent space using Restyle-e4e and Restyle-pSp encoder on StyleGAN3

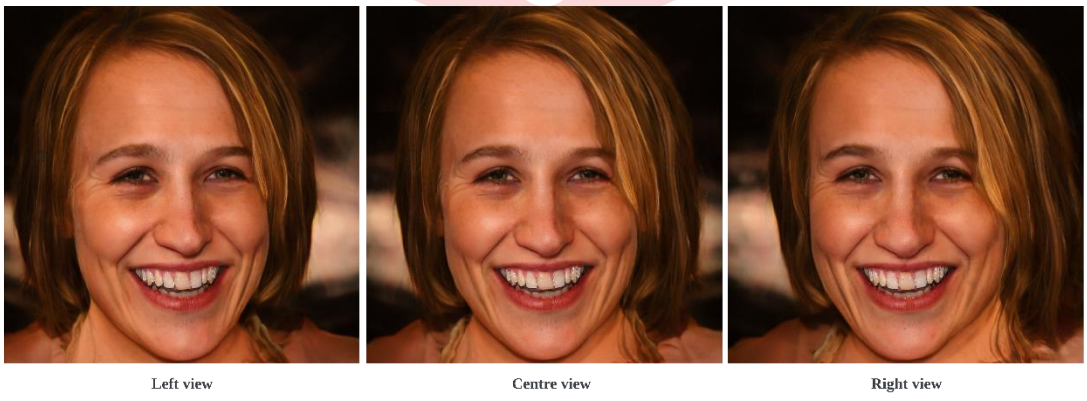


Figure 5.2: Generated multi-view using Restyle-e4e encoder

Figure 5.3: Generated multi-view using Restyle-pSp encoder

Figure 5.4: Input image (male)

Figure 5.5: Generated pose from StyleGAN3

Figure 5.6: Input image (female)

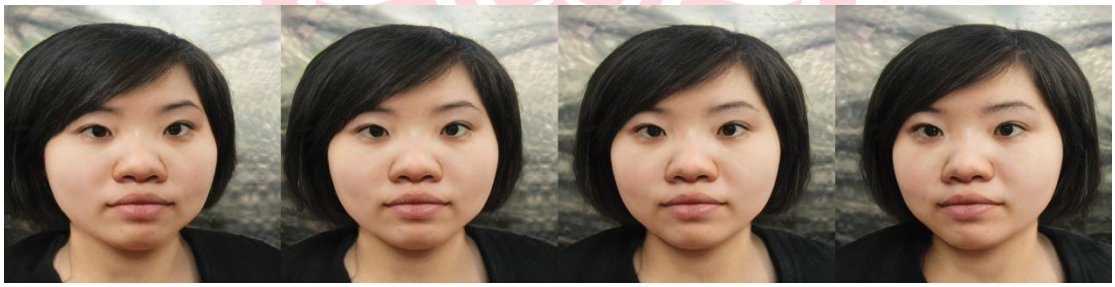


Figure 5.7: Pose generated from 3DDFA

5.2 Generating UV Map

Additionally, 3D Dense Face Alignment (3DDFA) technique is utilized to estimate the facial shape and structure from the input image. This enabled generation of "UV map" The UV map serves as a bridge between the 2D image and the subsequent 3D model generation.

As seen in Figure 5.8, texture map generated is only the front face and doesn’t include the hair, neck and other parts.

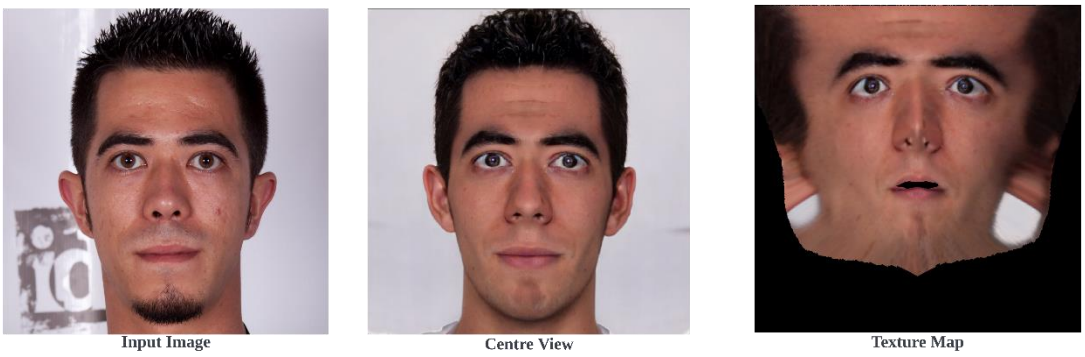


Figure 5.8: Generated UV map from 3DDFA

Also, given Figure 5.4, StyleGAN3 can be seen generating even better results by rotating the unseen parts of the image generating pose like Figure 5.5 giving a very good result for generating UV Map like Figure 5.9 .

Figure 5.9: Texture map generated from 3DDFA

Figure 5.10: UV map generated from 3DDFA

5.3 Creating 3D Model using 3DDFA

Furthermore, using the estimated facial shape from 3DDFA, the thesis proceeds to construct a "3D model" of the subject's face. This model captures the geometric information and spatial arrangement of facial features in a three-dimensional space, facilitating a more comprehensive understanding of the subject's facial structure.

Figure 5.11 shows the final result after applying to the 3D model from the given input image Figure 5.8.

Figure 5.11: UV map applied to a 3D model

The most unique result is Figure 5.12 generated from input image Figure 5.4. The reason being since the 3D model is generating from unseen part of the input image.

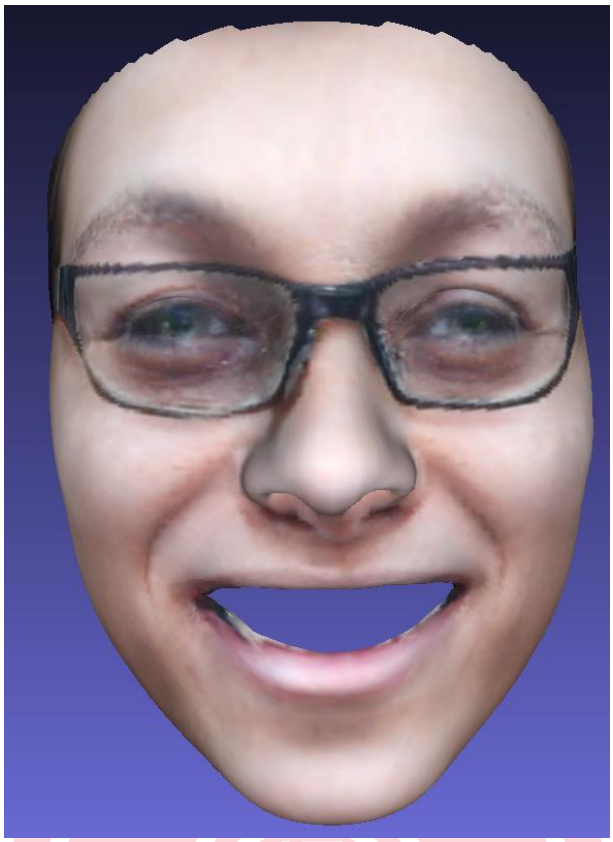


Figure 5.12: Generated 3D model from 3DDFA

When tested on Figure 5.6 which is an in-the-wild image, the model generates near to accurate pose and UV-map but loses identity of the user as it can be seen on Figure 5.7, Figure 5.10 and Figure 5.13 respectively.

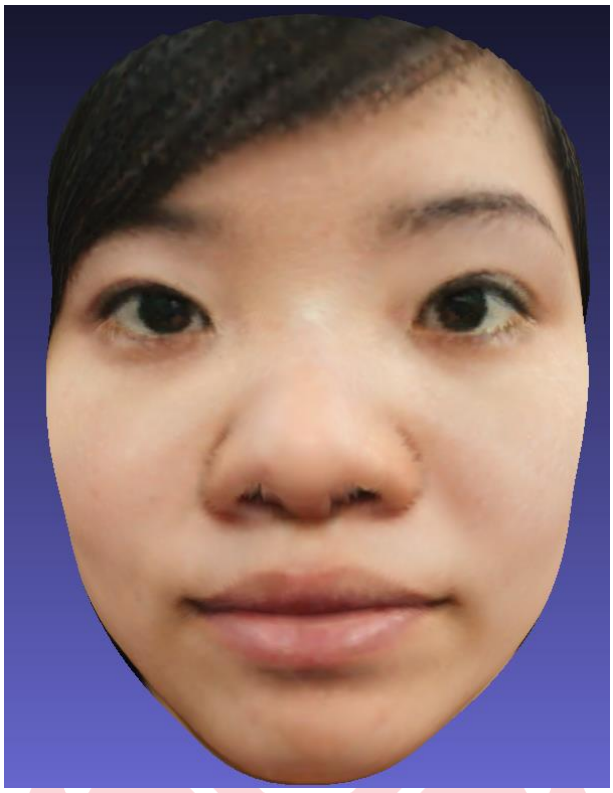


Figure 5.13: Generated 3D model from 3DDFA

6. Conclusion

The thesis thus concludes the following postulates:

1. Given a single image, the proposed architecture can embed images in latent space using Restyle-e4e encoder in StyleGAN3.
2. After embedding, given the latent code, multi-view images can be generated using InterFaceGAN.
3. Taking the center view images as the target image, generate texture map and 3D model using 3DDFA.
4. Wrap the texture map on the 3D model.
5. Subjective Evaluation shows the efficacy of 3DGANTex.

However, the results clearly show the following drawbacks:

1. FFHQ though provides good results, the images are mostly focused on the person. Whereas in real-world it is very difficult to capture images focused on human faces. Sometimes the images are captured in wide angle which can jeopardize the architecture.
2. Latent Space embedding and editing though shows commendable results, still fails to retain the identity and facial features depending on which encoder is being used.
3. 3DDFA though generates satisfactory texture map. However, it assumes that the texture of the face is largely determined by the color information in the 2D image, and does not take into account other factors such as lighting conditions and shadows. As a result, the generated texture may not always be accurate and may require further refinement or correction.

Finally, the author is intended to make it an end-to-end product later by creating a mobile application that can do the aforementioned real-world applications like generating Avatars, help in better facial recognition and can be used in various ways in entertainment industry.

References

[1] M. Mirza *et al.*, "Generative Adversarial Nets," *Advances in neural Information Processing Systems*, vol. 27, p. 2672—2680, 2014, https://doi.org/10.48550/arXiv.1406.2661.

[2] B. Gecer, J. Deng, and S. Zafeiriou, "OSTeC: One-Shot Texture Completion," in *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021-06-01 2021: IEEE, https://doi.org/10.1109/cvpr46437.2021.00754.

[3] H. Zhou *et al.*, "Rotate-and-Render: Unsupervised Photorealistic Face Rotation from Single-View Images," *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, p. 5911—5920, 2020, https://doi.org/10.48550/ARXIV.2003.08124.

[4] H. Bai *et al.*, "FFHQ-UV: Normalized Facial UV-Texture Dataset for 3D Face Reconstruction," *arXiv preprint arXiv:2211.13874*, 2022, https://doi.org/10.48550/ARXIV.2211.13874.

[5] Y. Alaluf *et al.*, "Third Time's the Charm? Image and Video Editing with StyleGAN3," in *Computer Vision–ECCV 2022 Workshops: Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part II*, 2023: Springer, p. 204—220, https://doi.org/10.1007/978-3-031-25063-7_13.

[6] X. Zhu, X. Liu, Z. Lei, and S. Z. Li, "Face Alignment in Full Pose Range: A 3D Total Solution," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 41, no. 1, p. 78—92, 2019-01-01 2019, https://doi.org/10.1109/tpami.2017.2778152.

[7] X. Yin *et al.*, "Towards Large-Pose Face Frontalization in the Wild," in *2017 IEEE International Conference on Computer Vision (ICCV)*, 2017-10-01 2017: IEEE, p. 3990—3999, https://doi.org/10.1109/iccv.2017.430.

[8] T. Karras *et al.*, "Alias-Free Generative Adversarial Networks," *Advances in Neural Information Processing Systems*, vol. 34, p. 852—863, 2021, https://doi.org/10.48550/arXiv.2106.12423.

[9] T. Karras *et al.*, "Analyzing and Improving the Image Quality of StyleGAN," in *Proceedings of the IEEE/CVF Conference on*

Computer Vision and Pattern Recognition, 2020, p. 8110—8119, https://doi.org/10.48550/arXiv.1912.04958.

[10] T. Karras, S. Laine, and T. Aila, "A Style-Based Generator Architecture for Generative Adversarial Networks," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, p. 4401—4410, https://doi.org/10.48550/arXiv.1812.04948.

[11] D. P. Kingma and M. Welling, "Auto-Encoding Variational Bayes," *arXiv preprint arXiv:1312.6114*, 2013, https://doi.org/10.48550/arXiv.1312.6114.

[12] T. Karras *et al.*, "Training Generative Adversarial Networks with Limited Data," *Advances in Neural Information Processing Systems*, vol. 33, p. 12104—12114, 2020.

[13] C. Eastwood and C. K. Williams, "A Framework for the Quantitative Evaluation of Disentangled Representations," in *International Conference on Learning Representations*, 2018.

[14] Z. Wu, D. Lischinski, and E. Shechtman, "Stylespace Analysis: Disentangled Controls for StyleGAN Image Generation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, p. 12863—12872, https://doi.org/10.48550/arXiv.2011.12799.

[15] I. Kemelmacher-Shlizerman and R. Basri, "3D Face Reconstruction from a Single Image Using a Single Reference Face Shape," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, no. 2, p. 394—405, 2011-02-01 2011, https://doi.org/10.1109/tpami.2010.63.

[16] J. J. Atick, P. A. Griffin, and A. N. Redlich, "Statistical Approach to Shape from Shading: Reconstruction of Three-Dimensional Face Surfaces from Single Two-Dimensional Images," *Neural Computation*, vol. 8, p. 1321—1340, 1996, https://doi.org/10.1162/neco.1996.8.6.1321.

[17] V. Blanz and T. Vetter, "A Morphable Model for the Synthesis of 3D Faces," in *Proceedings of the 26th Annual Conference on Computer Graphics and Interactive Techniques*, 1999, p. 187—194, https://doi.org/10.1145/311535.311556.

[18] Y. Tian *et al.*, "CR-GAN: Learning Complete Representations for Multi-View Generation," *arXiv preprint arXiv:1806.11191*, 2018, https://doi.org/10.48550/arXiv.1806.11191.

[19] L. Tran, X. Yin, and X. Liu, "Disentangled Representation Learning GAN for Pose-Invariant Face Recognition," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017-07-01 2017: IEEE, p. 1415—1424, https://doi.org/10.1109/cvpr.2017.141.

[20] X. Zhu *et al.*, "High-Fidelity Pose and Expression Normalization for Face Recognition in the Wild," in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015-06-01 2015: IEEE, https://doi.org/10.1109/cvpr.2015.7298679.

[21] Y. Hu *et al.*, "Pose-Guided Photorealistic Face Rotation," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2018-06-01 2018: IEEE, p. 8398—8406, https://doi.org/10.1109/cvpr.2018.00876.

[22] L. Tran, X. Yin, and X. Liu, "Representation Learning by Rotating Your Faces," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 41, no. 12, p. 3007—3021, 2019-12-01 2019, https://doi.org/10.1109/tpami.2018.2868350.

[23] J. Yim *et al.*, "Rotating your Face using Multi-Task Deep Neural Network," in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015-06-01 2015: IEEE, p. 676—684, https://doi.org/10.1109/cvpr.2015.7298667.

[24] Y. Qian, W. Deng, and J. Hu, "Unsupervised Face Normalization With Extreme Pose and Expression in the Wild," in *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019-06-01 2019: IEEE, p. 9851—9858, https://doi.org/10.1109/cvpr.2019.01008.

[25] R. Gross *et al.*, "Multi-PIE," *Image and Vision Computing*, vol. 28, no. 5, p. 807—813, 2010-05-01 2010, https://doi.org/10.1016/j.imavis.2009.08.002.

[26] O. Tov *et al.*, "Designing an Encoder for StyleGAN Image Manipulation," *ACM Transactions on Graphics (TOG)*, vol. 40, no. 4, p. 1—14, 2021, https://doi.org/10.1145/3450626.3459838.

[27] E. Richardson *et al.*, "Encoding in Style: A Stylegan Encoder for

Image-to-Image Translation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, p. 2287—2296, https://doi.org/10.48550/arXiv.2008.00951.

[28] H. Luo *et al.*, "Normalized Avatar Synthesis using Stylegan and Perceptual Refinement," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, p. 11662—11672, https://doi.org/10.1109/CVPR46437.2021.01149.

[29] B. Egger *et al.*, "3D Morphable Face Models—Past, Present, and Future," *ACM Transactions on Graphics*, vol. 39, no. 5, p. 1—38, 2020-10-31 2020, https://doi.org/10.1145/3395208.

[30] S. J. Garbin, M. Kowalski, M. Johnson, and J. Shotton, "High Resolution Zero-Shot Domain Adaptation of Synthetically Rendered Face Images," presented at the Computer Vision – ECCV 2020, 2020-01-01, 2020. https://doi.org/10.1007/978-3-030-58604-1_14.

[31] B. Gecer *et al.*, "Synthesizing Coupled 3D Face Modalities by Trunk-Branch Generative Adversarial Networks," presented at the Computer Vision – ECCV 2020, 2020-01-01, 2020. https://doi.org/10.1007/978-3-030-58526-6_25.

[32] B. Gecer, S. Ploumpis, I. Kotsia, and S. Zafeiriou, "GANFIT: Generative Adversarial Network Fitting for High Fidelity 3D Face Reconstruction," in *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019-06-01 2019: IEEE, p. 1155—1164, https://doi.org/10.1109/cvpr.2019.00125.

[33] J. Guo *et al.*, "Towards Fast, Accurate and Stable 3D Dense Face Alignment," in *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XIX*, 2020: Springer, p. 152—168, https://doi.org/10.1007/978-3-030-58529-7_10.

[34] P. Dollár, P. Welinder, and P. Perona, "Cascaded Pose Regression," in *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2010: IEEE, p. 1078—1085, https://doi.org/10.1109/CVPR.2010.5540094.

[35] X. Cao, Y. Wei, F. Wen, and J. Sun, "Face Alignment by Explicit Shape Regression," *International Journal of Computer Vision*, vol.

107, p. 177—190, 2014, https://doi.org/10.1109/CVPR.2012.6248015.

[36] X. Xiong and F. De la Torre, "Supervised Descent Method and its Applications to Face Alignment," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2013, p. 532—539, https://doi.org/10.1109/CVPR.2013.75.

[37] J. Booth *et al.*, "A 3D Morphable Model Learnt from 10,000 Faces," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016-06-01 2016: IEEE, p. 5543—5552, https://doi.org/10.1109/cvpr.2016.598.

[38] A. Lattas *et al.*, "AvatarMe: Realistically Renderable 3D Facial Reconstruction" In-The-Wild"," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, p. 760—769, https://doi.org/10.48550/arXiv.2003.13845.

[39] A. Lattas *et al.*, "AvatarMe++: Facial Shape and BRDF Inference With Photorealistic Rendering-Aware GANs," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 44, no. 12, p. 9269—9284, 2022-12-01 2022, https://doi.org/10.1109/tpami.2021.3125598.

[40] L. Bao *et al.*, "High-Fidelity 3D Digital Human Head Creation from RGB-D Selfies," *ACM Transactions on Graphics*, vol. 41, no. 1, p. 1—21, 2022-02-28 2022, https://doi.org/10.1145/3472954.

[41] H. Yang *et al.*, "Facescape: A Large-Scale High Quality 3D Face Dataset and Detailed Riggable 3D Face Prediction," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, p. 601—610, https://doi.org/10.48550/arXiv.2003.13989.

[42] J. Deng *et al.*, "UV-GAN: Adversarial Facial UV Map Completion for Pose-invariant Face Recognition," *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, p. 7093—7102, 2018, https://doi.org/10.48550/ARXIV.1712.04695.

[43] J.-Y. Zhu, P. Krähenbühl, E. Shechtman, and A. A. Efros, "Generative Visual Manipulation on the Natural Image Manifold," in *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings,*

Part V 14, 2016: Springer, p. 597—613, https://doi.org/10.48550/arXiv.1609.03552.

[44] A. Creswell and A. A. Bharath, "Inverting the Generator of a Generative Adversarial Network," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 30, no. 7, p. 1967—1974, 2018, https://doi.org/10.1109/TNNLS.2018.2875194.

[45] R. Zhang *et al.*, "The Unreasonable Effectiveness of Deep Features as a Perceptual Metric," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, p. 586—595, https://doi.org/10.1109/CVPR.2018.00068.

[46] Y. Alaluf, O. Patashnik, and D. Cohen-Or, "Restyle: A Residual-Based Stylegan Encoder via Iterative Refinement," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, p. 6711—6720, https://doi.org/10.48550/arXiv.2104.02699.

[47] B. Xu, N. Wang, T. Chen, and M. Li, "Empirical Evaluation of Rectified Activations in Convolutional Network," *arXiv preprint arXiv:1505.00853*, 2015, https://doi.org/10.48550/arXiv.1505.00853.

[48] Y. Shen, J. Gu, X. Tang, and B. Zhou, "Interpreting the Latent Space of GANs for Semantic Face Editing," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, p. 9243—9252, https://doi.org/10.48550/arXiv.1907.10786.

[49] K. Simonyan and A. Zisserman, "Very Deep Convolutional Networks for large-Scale Image Recognition," *arXiv preprint arXiv:1409.1556*, 2014, https://doi.org/10.48550/arXiv.1409.1556.

[50] J. Deng, J. Guo, N. Xue, and S. Zafeiriou, "Arcface: Additive Angular Margin Loss for Deep Face Recognition," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, p. 4690—4699, https://doi.org/10.1109/TPAMI.2021.3087709.

[51] Y. Ke and R. Sukthankar, "PCA-SIFT: A more Distinctive Representation for Local Image Descriptors," in *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2004. CVPR 2004.*, 2004, vol. 2: IEEE, p. II—

II, https://doi.org/10.1109/CVPR.2004.1315206.

[52] A. Hadid, "The Local Binary Pattern Approach and its Applications to Face Analysis," in *2008 First Workshops on Image Processing Theory, Tools and Applications*, 2008: IEEE, p. 1—9, https://doi.org/10.1109/IPTA.2008.4743795.

[53] L. Spreeuwers, "Fast and Accurate 3D Face Recognition: Using Registration to an Intrinsic Coordinate System and Fusion of Multiple Region Classifiers," *International Journal of Computer Vision*, vol. 93, no. 3, p. 389—414, 2011, https://doi.org/10.1007/s11263-011-0426-2.

[54] Y. Shen, C. Yang, X. Tang, and B. Zhou, "Interfacegan: Interpreting the Disentangled Face Representation Learned by GANs," *IEEE transactions on Pattern Analysis and Machine Intelligence*, vol. 44, no. 4, p. 2004—2018, 2020, https://doi.org/10.48550/arXiv.2005.09635.

[55] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image Quality Assessment: From Error Visibility to Structural Similarity," *IEEE transactions on Image Processing*, vol. 13, no. 4, p. 600—612, 2004, https://doi.org/10.1109/TIP.2003.819861.

[56] L. Zhang, L. Zhang, X. Mou, and D. Zhang, "FSIM: A Feature Similarity Index for Image Quality Assessment," *IEEE transactions on Image Processing*, vol. 20, no. 8, p. 2378—2386, 2011, https://doi.org/10.1109/TIP.2011.2109730.

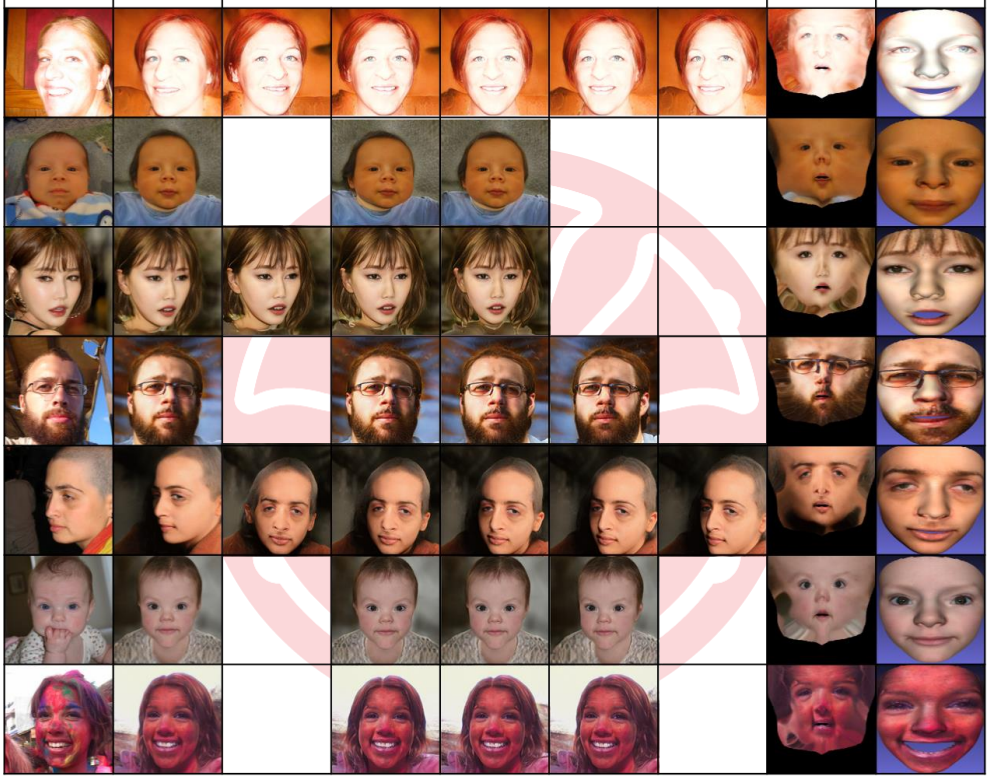
[57] J. Johnson, A. Alahi, and L. Fei-Fei, "Perceptual Losses for Real-Time Style Transfer and Super-Resolution," in *Computer Vision–ECCV 2016*, 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part II 14, 2016: Springer International Publishing, p. 694—711, https://doi.org/10.1007/978-3-319-46475-6_43.

[58] Z. Wang, E. P. Simoncelli, and A. C. Bovik, "Multiscale Structural Similarity for Image Quality Assessment," in *The Thrity-Seventh Asilomar Conference on Signals, Systems & Computers, 2003*, 2003, vol. 2: IEEE, p. 1398—1402, https://doi.org/10.1109/ACSSC.2003.1292216.

Appendix

Appendix 1

This table shows some results of using 3DGANTex under various condition.

Input Image	Inverse Image	Multi-View Images [-3,+3]	Texture Map	3D Mesh
				

Appendix 2

This table continues the above table and shows some results of using 3DGANTex under various condition.

Input Image	Inverse Image	Multi-View Images [-3,+3]	Texture Map	3D Mesh