Devising Ceramic Tiles Texture featuring Semi-automatic Image Clustering and Generative Adversarial Networks

Labib Abdullah (labib444@gmail.com)
Ahsanullah University of Science and Technology
Md. Ashfaqul Azam Chowdhury
Ahsanullah University of Science and Technology
Abrar Rafid Noor
Ahsanullah University of Science and Technology
Md. Sakib Irtiza
Ahsanullah University of Science and Technology
Md. Tanvir Rouf Shawon
Ahsanullah University of Science and Technology
Dr. Md. Shahriar Mahbub
Ahsanullah University of Science and Technology

Research Article

Keywords: GAN, DCGAN, WGAN-GP, Ceramic tiles design, Image synthesis

Posted Date: May 11th, 2023

DOI: https://doi.org/10.21203/rs.3.rs-2900918/v1

Additional Declarations: No competing interests reported.

Devising Ceramic Tiles Texture featuring Semi-automatic Image Clustering and Generative Adversarial Networks

Labib Abdullah, Md. Ashfaqul Azam Chowdhury, Abrar Rafid Noor, Md. Sakib Irtiza, Md. Tanvir Rouf Shawon, Dr. Md. Shahriar Mahbub

Ahsanullah University of Science and Technology, 141 & 142, Love Road, Tejgaon Industrial Area, Dhaka-1208, Bangladesh

Abstract

Neural network-based generation has become a promising area of research with diverse applications in recent years, such as generating synthetic data, medical images, celebrity faces and so on. In this study, we try to investigate the potential of generative models in the field of ceramic tiles design. Specifically, we employ DCGAN and WGAN-GP to generate textures of ceramic tiles. Our dataset undergoes a semi-automated pre-processing stage, which has been crucial for achieving high-quality outputs. We analyse the performances and outputs of both the models. To examine the spatial quality of the produced outputs, employing statistical calculation and collecting survey reviews from professionals in the ceramic tile industry have been done to gather better insights. Our results demonstrate the potential of generative models for producing realistic ceramic tiles textures and suggest this work holds promise for improving the creativity and efficiency of ceramic tiles design process.

Keywords: GAN, DCGAN, WGAN-GP, Ceramic tiles design, Image

synthesis

---

1. Introduction

In every discipline, Artificial Intelligence (AI) has made substantial advances. As a result, AI being utilized in a variety of ways within the setting of the arts. It can act as a source of creative inspiration for individuals, as well as a type of creative assistant that can help with tedious tasks, particularly in the digital realm. From creating pop songs [1] to imitating great artworks [2] and sculptures [3], AI has demonstrated its capabilities in a variety of endeavors. When it comes to producing images from scratch, Generative Adversarial Networks (GANs) [4] show enormous potential.

Ceramic tiles have been used in houses, shops, and places of worship, among others as ornamental features for ages. Ceramic tiles are one of the first forms of decorative art and their durability and stunning beauty have made them highly valued for centuries. As they shape the look and feel of a home, tiles are a crucial element of contemporary interior design. They are employed to produce the appropriate atmosphere. Ceramic tiles are available in a variety of designs. It can be used on floors, walls, fireplaces, ceilings, bathrooms, among other purposes. Ceramic tiles are available in a variety of range of colors and textures, providing for considerable creative versatility.

Designing of ceramic tiles is an important aspect for ceramic industry, as it is an important role in enhancing the aesthetic appeal and functionality of architectural spaces. Ceramic tiles design processes have several shortcomings that can hinder creativity and efficiency for designers. The traditional process of designing ceramic tiles is time-consuming, labor-intensive and often relies on manual processes. Also it has reliance on experienced artisans and limited design options results in a lack of diversity in the market- resulting in few options for consumers. Therefore, the need for a more efficient and automated approach to ceramic tile design has become increasingly important.

The motivation behind our efforts stems from the convergence of the notions of exploring the generation ability of machines, the desire for innovative designs and automating design processes which will address the issues currently faced by the industry. Two generative models have been applied in this regard: DCGAN [5] and WGAN-GP [6] to generate ceramic tiles design. The following step have been performed: (i) Collection of data (ii) Pre-processing to enhance the datasets (iii) Training of the models (iv) Evaluating and analyzing the results through statistical method and professionals' feedback by conducting survey. The final outcome suggests that design patterns of ceramic tiles of comparable quality can be produced, which is not only per our observation but also substantiate the outcomes of performed evaluation steps as well.

2. State of the Art

S. Khalil et al. [7] propose a system based on DCGAN to generate textile patterns on the industrial scale. While modifying activation functions, the system uses convolutional layers instead of pooling layers. Their results can be improved using a large dataset and more training time. They use unlabelled data in their work. In the end, the authors acknowledge that it will be interesting to explore if labeled data can further improve the results.

R. A. Fayyaz et al. [8] try to generate textile images using WGANs GP. In this paper, they experiment with outcomes of three generative models: VAEs, DCGAN and WGANs GP [9]. All the models produce novel designs, but WGANs GP delivers the most aesthetically pleasing and realistic patterns. Based on the authenticity of the picture and the variety of the produced images, Inception Scores is used to quantify the outcomes of the models. There are a number of causes for this, including the use of simple KL divergence [10] by DCGANs to reduce the gap among the produced and original data's distribution where as WGANs uses Wasserstein distance [11].

S.S. Nasrin et al. [12] use Deep Convolutional Neural Networks (DCGANs) to produce henna pattern images with variations. They explain the process of using the Generator and Discriminator networks and how they are pre-processed before being used to train the networks. The authors conduct experiments using three different dimensions (32x32 pixels, 64x64 pixels, and 128x128 pixels) and find that the 32x32 and 64x64 dimensions produce better outcomes for their smaller dataset.

D. H. Kim [13] implements Wasserstein loss to reduce mode collapse and at the conclusion of the discriminator network, adds dropout layer to inject stochasticity into the widely used DCGAN design. The author also adds convolutional layers to the generator network to increase expressiveness and noise smoothness. The experiments were conducted using Nicolas Gervais' dataset of 64,000 car photos [14], which are categorized by price, model year, body style, and other attributes. The BoolGAN architecture proposed by the author achieved a reduced Fréchet Inception Distance (FID) score from 195.922 (baseline) to 165.966.

Kini et al. [15] try to generate underwater images- implementing DC-GAN along with the Wasserstein GAN algorithm for the model. The authors manage to generate images that have a distinct blue hue of water in their background and most of the images shows sharp resemblance with underwater creatures like a fish or turtle. Small number of images is a limitation of the study, acknowledging that larger dataset might have improved their overall accuracy and image quality.

In this study [16], the authors propose using Generative Adversarial Networks (GANs) for augmentation purpose of the EuroSAT dataset [17] for the classification of Land Use and Land Cover (LULC). They use two different types of GANs, DCGAN and WGAN-GP, to devise images for each class present in the dataset. This study finds that choice of GAN architecture didn't significantly affect the effectiveness of the model. Combination of traditional geometric techniques of augmentation and generated images by GANs improves the final results compared to the baseline. The authors conclude that GAN augmentation can be used to improve deep classification models' generalizability, which is applied to satellite images.

2.1. Research Gap

In many research studies mentioned, it is seen common practice to amalgamate all available images and train the model. However, such an approach yields suboptimal results. In our study, a semi-automatic image clustering technique has been adopted prior to training. To our knowledge, it is also the first approach to apply two variants of GANs to design ceramic tiles textures, compare their outputs using mathematical and survey based evaluation.

3. Methodology

This section presents the methodology used for data collection, along with the associated challenges encountered. The pre-processing steps that were undertaken to improve the quality of the datasets have been outlined. Moreover, DCGAN and WGAN-GP have been used to generate tiles design which are described in this section. Additionally, the hyperparameters used and hardware setups utilized for training have been mentioned in this section.

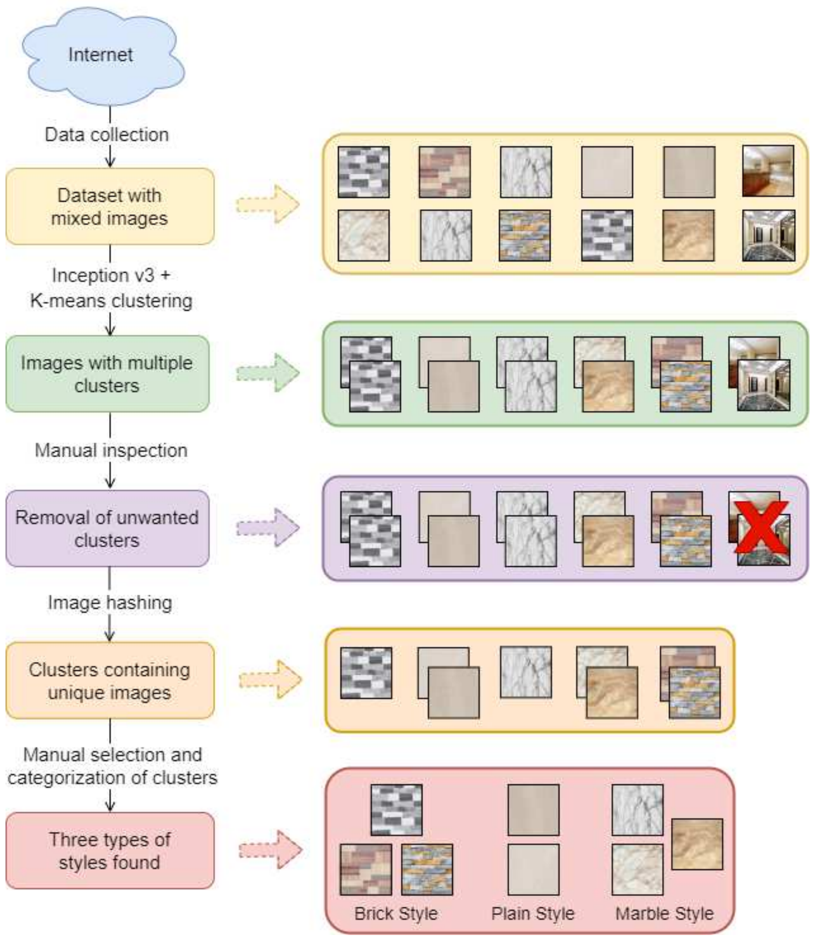

Internet
Data collection
Dataset with mixed images
Inception v3 + K-means clustering
Images with multiple clusters
Manual inspection
Removal of unwanted clusters
Image hashing
Clusters containing unique images
Manual selection and categorization of clusters
Three types of styles found
Brick Style
Plain Style
Marble Style

Figure 1: Flow diagram of preparing the datasets

Figure 1 shows the overall flow diagram of preparing datasets from the originally collected dataset. At first, the dataset was populated with ceramic tiles images from the internet. As it had variety of design textures mixed together, it was divided into multiple clusters with the help of inception v3 and k-means clustering. Afterwards, unwanted clusters were removed through manual inspection. Then image hashing was used to remove the duplicates images within the remaining clusters. Later, design texture-wise same clusters were manually put together to make different tiles categories.

3.1. Data Collection

GAN models generally need a huge amount of data to work well [4]. For our work, images of ceramic tile textures were collected from various public websites of ceramic tile companies, as there were no ready-made datasets available. Initially around 20000 images were collected. Several issues were encountered during the bulk download of images from the websites, including duplication of images, unnecessary backgrounds in the form of tiles shown in the bathroom or living room, multiple tiles segmented in one image, varying view angles and so on. Fig2 shows examples for different faulty images when collecting the data.

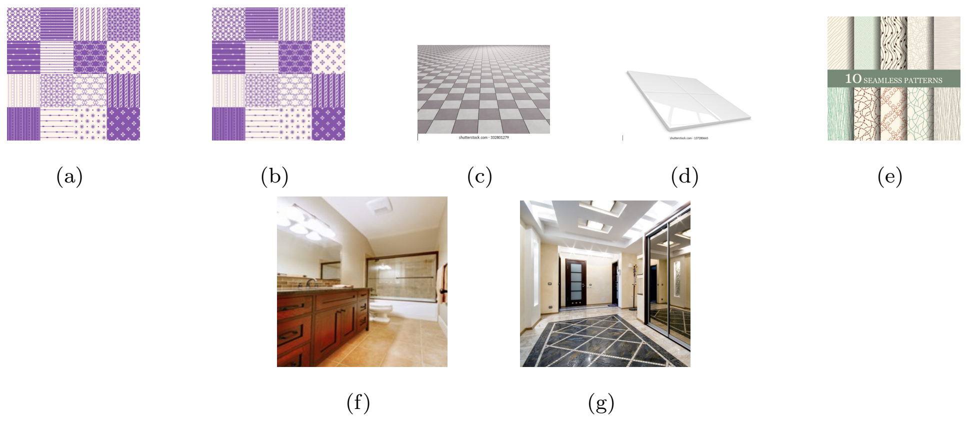


Figure 2: Different types of faulty images found during data collection. (a) & (b) are examples of duplicate images, (c) is of an image with different view angle, (d) is of a 3D rotated image, (e) is of multiple segmented images of tiles together, (f) & (g) are examples of tiles image with backgrounds

3.2. Pre-processing

Manually filtering the unwanted, defective images would be tedious. Also there were different kinds of categories in the downloaded tiles images. If there are many categories present, GANs are prone to the sampling of the input data, moreover, combining all categories of data without accounting for these factors may lead to suboptimal results [18, 19, 20]. So categorizing the images could help in the GANs training and generate more specific types of ceramic tiles textures.

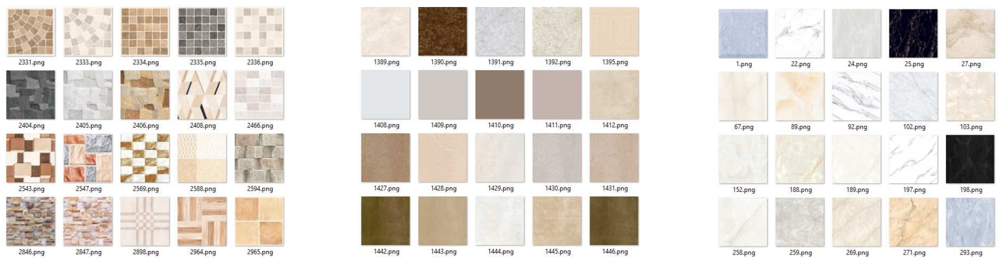
A semi-automatic process was used to cluster the collection of images. The dataset comprised of all the downloaded images was first passed through a pre-trained Inception v3 model [21] to extract feature vectors [22]. K-means clustering algorithm [23] was then applied to automatically cluster the images, which were organized into separate folders. The automatic clustering process can not categorize the tiles as required, therefor, a manual process is required for perfecting the process.

Setting up a proper cluster size is a difficult problem, In tile clustering context, it is set empirically rather than theoretically. The initial cluster size of 3 resulted the grouping together dissimilar images. To improve clustering, the cluster size was increased with a fixed step size as each process required between 4-7 hours to complete, which narrowed the scope of repetitive experimentation with linearly increasing values. After that cluster size of 50 was implemented and it was found to be satisfactory for grouping similar textile patterns together. As an additional note, it grouped the unwanted faulty images together in some of the clusters, which made it easier to deduct them from the dataset. Finally, as per our observation, cluster with similar texture patterns were manually combined by experts, resulting in three main clusters or categories. They are:

(i) Brick styled (ii) Plain styled (iii) Marble styled

These names were assigned following the nature of their design texture. Brick styled ones had various kinds of brick-like design on them, Marble styled ones had the familiar wave-like designs that we see in tiles and Plain styled ones only consisted of solid colors. Other than the mentioned three, a few categories were discovered through clustering, but due to their very limited quantity, they were not considered for separate categorization for training.

Some cluster contained duplicate images. These duplicate images were removed using the perceptual hashing technique, an approach for image duplication [24]. Python library called 'ImageHash' was used to compute a perceptual hash for each image [25].



(a) Brick Style (b) Plain Style (c) Marble Style

Figure 3: Categories of styles found after the pre-processing steps

Fig3 shows some examples titles of three categories. The Brick and Marble styled categories were focused on, as the plain styled category was deemed insufficient in terms of creating appealing designs. The two categories were then augmented using vertical flip, horizontal flip, and random rotations (with the exception of Brick styles, which were rotated 90 degrees to the left and right) [26]. Other augmentation techniques, such as cropping, image zooming, and color orientation, were avoided in order to preserve the original designs. The Marble styled category contained **2163** raw images, while the Brick styled category had **1871** raw images. Following augmentation, the Marble styled category contained **12978** images, and the Brick styled category had **14892** images. However, due to limited memory resources [1], the exact number of tiles could not be used for training purpose. At most, 12000 images could be used for each category.

[1]Table 3: Training time and hardware for different models

Style	Before augmentation	After augmentation	Used for training
Marble	2163	12978	12000
Brick	1871	14892	12000

Table 1: Types of styles in dataset, quantities before and after augmentation and the numbers used for training

3.3. Models Setup: Hyperparameters

The architecture followed for training DCGAN model was the same as Radford et. al. [5] and for WGAN-GP, it was same as Gulrajani et. al. [6]. The hyperparameter values were tweaked to adjust according to our dataset. Also, additional layers were added to both the generator and discriminator models. Most research papers used 32x32 or 64x64 images, requiring fewer layers to achieve the desired output. In our case, the models had to operate on larger 128x128 images, extra layers i.e. Convolutional layers, LeakyReLU layers were applied accordingly. The values used for training the models have been shown in the following table 2:

Model	Input image size	Image channels	Dataset size	Number of epochs	Learning rate	Momentum	Lambda	Critic train	Batch size	Noise dimension
DCGAN	128x128	3	12000	400	2e-4	0.5			32	100
WGAN-GP	-	-	-	-	1e-4		5	3	-	-

Table 2: Hyperparameters setup that has been used during training. (Absence of respective hyperparameter value has been left as empty and similar value as above has been indicated by using "-")

3.4. Generative Adversarial Networks

Generative Adversarial Networks (GANs) are models consisting of discriminator and generator which are two deep neural networks. The generator is trained such that it can produce outputs resembling a predefined dataset, while the discriminator is trained to tell apart original and the generated data. Together these neural networks are trained in a game-like way. The generator attempts to mimic more natural samples and the discriminator attempts to improve its ability to separate the generated and real data.

3.5. DCGAN

DCGAN (Deep Convolutional Generative Adversarial Network) architecture includes several modifications to the original GAN architecture to improve its performance for image generation. These modifications include using convolutional and transpose-convolutional layers in the generator and discriminator networks, using batch normalization in both networks and also using ReLU and LeakyReLU activation functions in the generator and discriminator networks respectably [5].

Several studies [5, 27] have compared the performance of DCGANs with the original GAN architecture, and found that DCGANs generate higher quality images with fewer training iterations and less sensitivity to hyperparameters, making DCGANs a better choice for image generation tasks than the original GAN architecture.

3.6. WGAN-GP

WGAN-GP (Wasserstein Generative Adversarial Network with Gradient Penalty) is a modification of GANs framework that addresses several of its limitations, such as mode collapse and instability. The loss function applied in WGAN-GP is mentioned:

$$L = \underbrace{\underset{\tilde{x}\sim\mathbb{P}_g}{\mathbb{E}}[D(\tilde{x})] - \underset{x\sim\mathbb{P}_r}{\mathbb{E}}[D(x)]}_{\textbf{Original critic loss}} + \underbrace{\lambda \underset{\hat{x}\sim\mathbb{P}_{\hat{x}}}{\mathbb{E}}[(\|\nabla_{\hat{x}}D(\hat{x})\|_2 - 1)^2]}_{\textbf{Gradient penalty}} \tag{1}$$

Original critic loss part measures the distance between the distribution of the outputted samples and of the real samples. In traditional GANs, the loss function is built on the basis of Jensen-Shannon divergence or Kullback-Leibler divergence, that can be difficult to optimize and can lead to instability. The Wasserstein distance, on the other hand, has several desirable properties that make it more suitable for GAN training, such as being continuous and having a meaningful interpretation as a distance metric.

The second part of the loss function is used to enforce the Lipschitz constraint on the discriminator. The Lipschitz constraint limits the maximum slope of the discriminator function, preventing it from becoming too powerful and causing instability in the training process [28]. It achieves this by penalizing the discriminator when its gradient norm exceeds 1, thereby encouraging the discriminator to have a more uniform gradient throughout the input space.

Together, these two components of the WGAN-GP loss function provide a more stable and effective framework for training GANs, allowing for better convergence and preventing mode collapse. Smith et al. [29] showed the performance of WGAN-GP and DCGAN in generating realistic facial images found that WGAN-GP outperformed DCGAN in terms of visual quality, diversity and stability, requiring fewer iterations to converge.

3.7. Hardware and Training Time

For the purpose of training, an initial attempt was made to train the models using Google's Colab GPU. However, due to the strict time limit policy and the extended time required for training, an alternative approach was sought, utilizing locally available hardware to facilitate faster training with no time constraints. Table 3 displays the relevant hardware information utilized during the training process.

Hardware	**Model**	**Memory**	**CUDA cores**	**1 epoch**
NVIDIA GTX 1660 SUPER	DCGAN	6GB GDDR6	1408	39 seconds
Colab GPU (Tesla K80)	DCGAN	24GB GDDR5	4992	264 seconds
NVIDIA GTX 1660 SUPER	WGAN-GP	6GB GDDR6	1408	41 seconds
Colab GPU (Tesla K80)	WGAN-GP	24GB GDDR5	4992	266 seconds

Table 3: Training time and hardware for different models

4. Results & Analysis

Within this section, input samples of Marble and Brick styled datasets are shown explaining their characteristics. A thorough description and analysis of the outputs with DCGAN, along with the encountered shortcomings, are presented. Additionally, the training loss graphs of WGAN-GP for both the Marble and Brick styled datasets are evaluated in accordance with the properties of loss graphs. The subsequent discussion describes and analyzes the outputs produced by WGAN-GP, followed by an introduction of the methods employed to evaluate the generated outputs and the outcomes of said methods.

4.1. Experimental Inputs: Marble and Brick Styles

The texture design of marble textured ceramic tiles is intended to mimic the natural veining and texture of real marble [30]. The texture and design can vary depending on the specific manufacturer and product line. These are commonly used for flooring, walls, countertops, and backsplashes in both residential and commercial settings [31]. Some tiles may feature a subtle veining pattern, while others may have a more pronounced and dramatic pattern. In figure 4, a set of Marble-styled ceramic tiles design is shown which has been used as input for training phase. Some of them are light-colored with a hint of veining pattern (i.e.: 2nd row, 2nd column) and also some of are showing a more striking and prominent pattern (i.e.: 2nd row, 4th column).

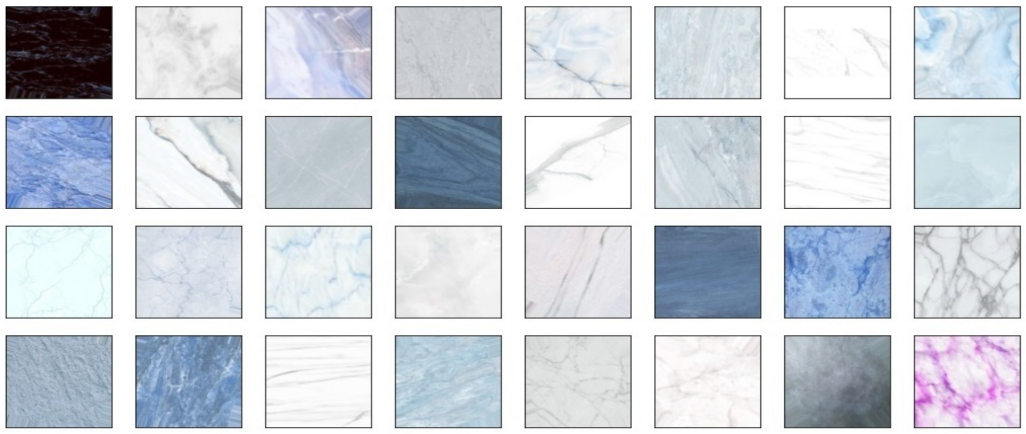


Figure 4: Marble styled input sample

On the other hand, brick styled ceramic tile texture is a type of ceramic tile that is designed to resemble the texture and appearance of traditional brick. These tiles have a rough, uneven surface, with irregular edges and divots that give them a distinctive, rustic look. The texture is intended to mimic the natural variations found in real brick, creating a warm, inviting ambiance in any space [32]. In figure 5, a set of Brick-styled ceramic tiles design is shown which has been used for training phase.

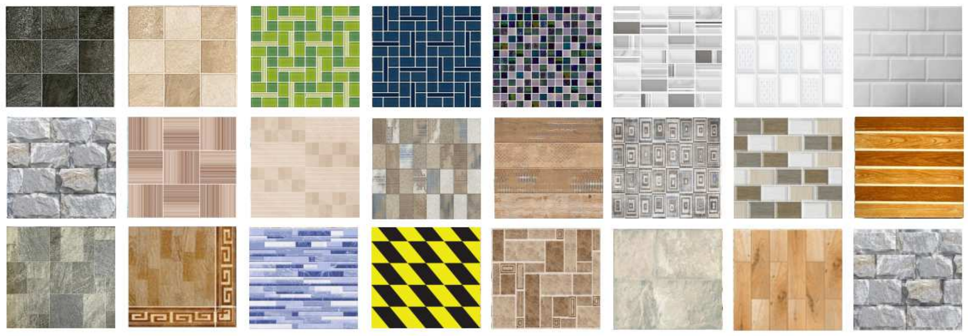


Figure 5: Brick styled input sample

4.2. DCGAN Outputs & Shortcomings

With the hyperparameters and training setups mentioned in the previous section, the Marble styled pre-processed dataset was trained in DCGAN. The produced outputs are shown in figure 6. As per our observation, the outputs were resembling the Marble styled dataset(figure 4) quite well,

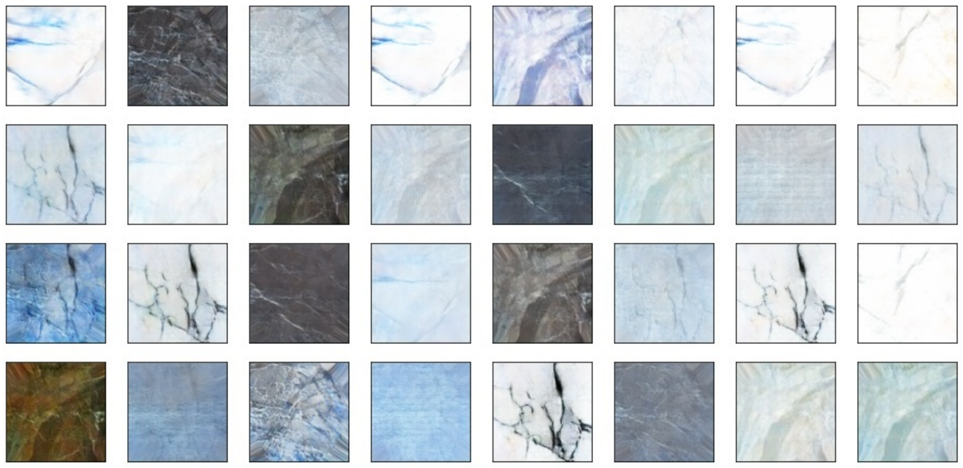


Figure 6: DCGAN outputs for training on Marble styled dataset

Also observed that the outputs were following a handful of texture patterns. In figure 7 below, based on the outputs of figure 6, the generated outputs that were texture-wise similar are put together in the same row.

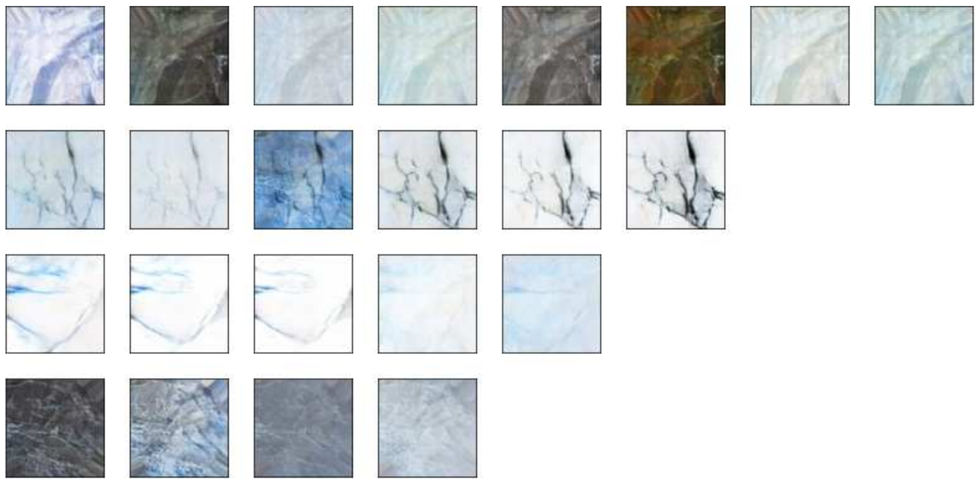


Figure 7: Texture-wise similar outputs of DCGAN based on figure 6. Similar texture-followed designs are kept in the same row

This phenomenon occurred due to mode collapse [33], where the model generates outputs that are limited to a small subset of the possible range of outputs. The model appears to be stuck or fixated on a single mode of the data distribution and fails to generate diverse outputs that reflect the full range of variation in the data [34]. While these outputs followed the same texture patterns, there were significant variations in color and hue among almost all of them (figure 7).

Mode collapse could occur for variety of reasons. It could occur when the generator network is not powerful enough to learn the full complexity of the sampling of the data [34]. Or when the discriminator is too strong and can easily identify and reject most of the generator's outputs [33]. It could also happen if the optimization algorithm get stuck in a local minimum, and fail to explore other areas of the objective function that correspond to different modes of the data distribution [34]. Investigating the actual reason behind mode collapse and overcoming it for the DCGAN model is beyond this research work, also it would need much time for exploration and training. As it showed vulnerability towards mode-collapse, it was decided not to go ahead with DCGAN and the Brick styled dataset was not trained with this model.

4.3. Loss Graphs

A loss graph plots the loss function value over the period of the training process. X-axis presents the number of iterations or epochs (passes through the entire training dataset), while the Y-axis shows the loss function value. Interpreting GANs loss graphs provide insights into the performance of a GANs model. Generally generator's and discriminator's loss graphs are analysed separately and both posses specific characteristics that defines the success of their training.

To differentiate original data from generated ones is the discriminator network's job. In early stages of the training, discriminator will be very good at this task, and the generator will struggle to produce convincing data. As a result, the discriminator loss will be low and the generator loss will be high. As the generator improves, the discriminator will find it harder to individualize the generated and real data and discriminator loss will start to increase. This is a good sign signifying the generator is starting to produce more convincing data [34].

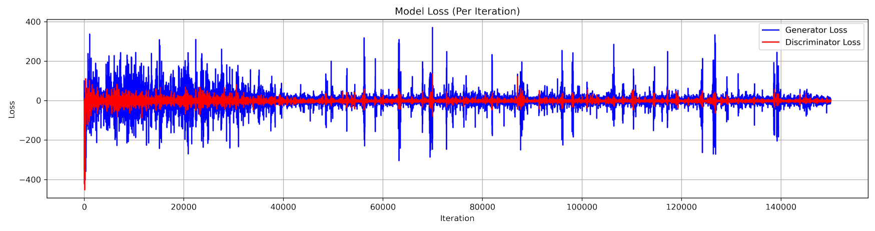
The job of the generator is to produce convincing data that can fool the discriminator. In the early stages of training, the generator will produce low-quality data that is easily distinguishable from real data. As a result, the generator loss will be high. As the generator improves, the generator loss will start to decrease, which means the generator is producing higher-quality data that is more similar to real data.

If the generator loss starts to decrease too quickly, it could be a sign of mode collapse, which means it is only producing a subset of the possible data. In this case, discriminator will become very good at differentiating between the generated and real data and the discriminator loss will start to decrease.

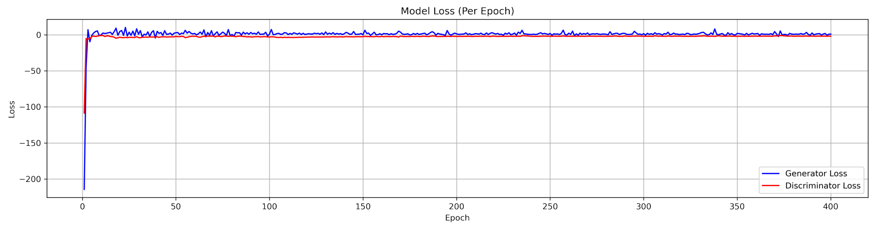
If the generator loss continues to decrease while the discriminator loss starts to increase, it could be a sign of overfitting. This means the generator is becoming too good at generating which is of similar-like with that of the training ones, but actually not generalizing well in terms of new data.

4.4. WGAN-GP Training

WGAN-GP model's generator and discriminator loss per epoch is shown for both of the datasets. For Marble styled dataset, it is shown in figure 8 and for Brick styled dataset, it is shown in figure 9.



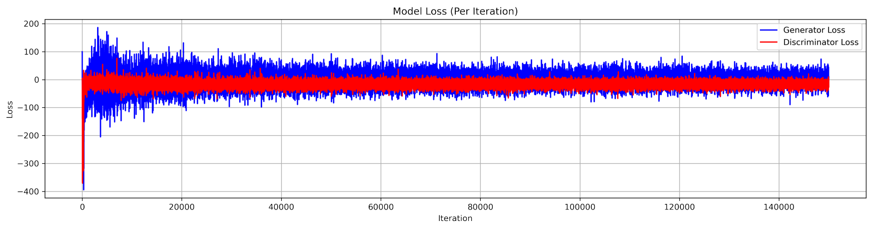
(a) Loss per iteration



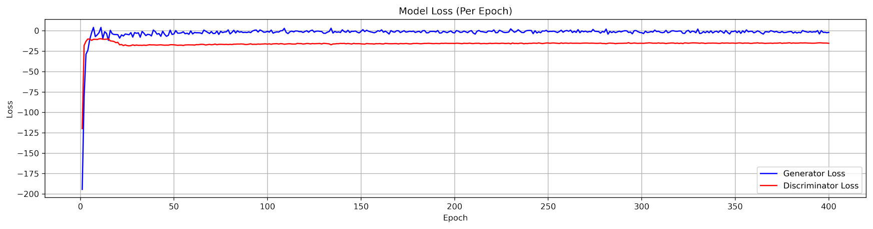
(b) Loss per epoch

Figure 8: Generator and discriminator loss for Marble styled dataset

In figure 8(a) loss per iteration, the generator's loss was high at the beginning, which is expected and discriminator's loss was good enough to provide appropriate feedback to the generator to improve its outputs. Both losses gradually decreased which is a good sign for stable generation. In figure 8(b) loss per epoch, it is observed that the discriminator loss stabled around -0.6 and generator loss stabled around 0. [34].



(a) Loss per iteration



(b) Loss per epoch

Figure 9: Generator and discriminator loss for Brick styled dataset

In figure 9(a) loss per iteration, for training Brick styled dataset, it is seen that generator loss was high and discriminator loss was also not too low at the beginning; therefore the generator could learn, however after a certain point the discriminator loss did not decrease though the generator loss decreased a bit. The model may have struggled to learn properly therefore causing the losses to be a little high and almost static, also the generator was not able to provide high quality output which resulted in such a situation [33]. In figure 9(a) loss per epoch, the loss graph became stable after 20 epoch of the training. Discriminator loss stabled around -20 and generator loss stabled around 0.

4.5. WGAN-GP Outputs

Resulting outputs produced by WGAN-GP model trained on Marble styled dataset is shown in figure 10. To our observation, the texture produced is quite strongly resembling the characteristic of marble styles (figure 4). Among them most of the designs are light colored with subtle design. And the rest of them are showing a more resilient texture. The outputs are not following any specific design texture like DCGAN outputs (figure 6). These generated designs seem to be more vibrant and natural looking.

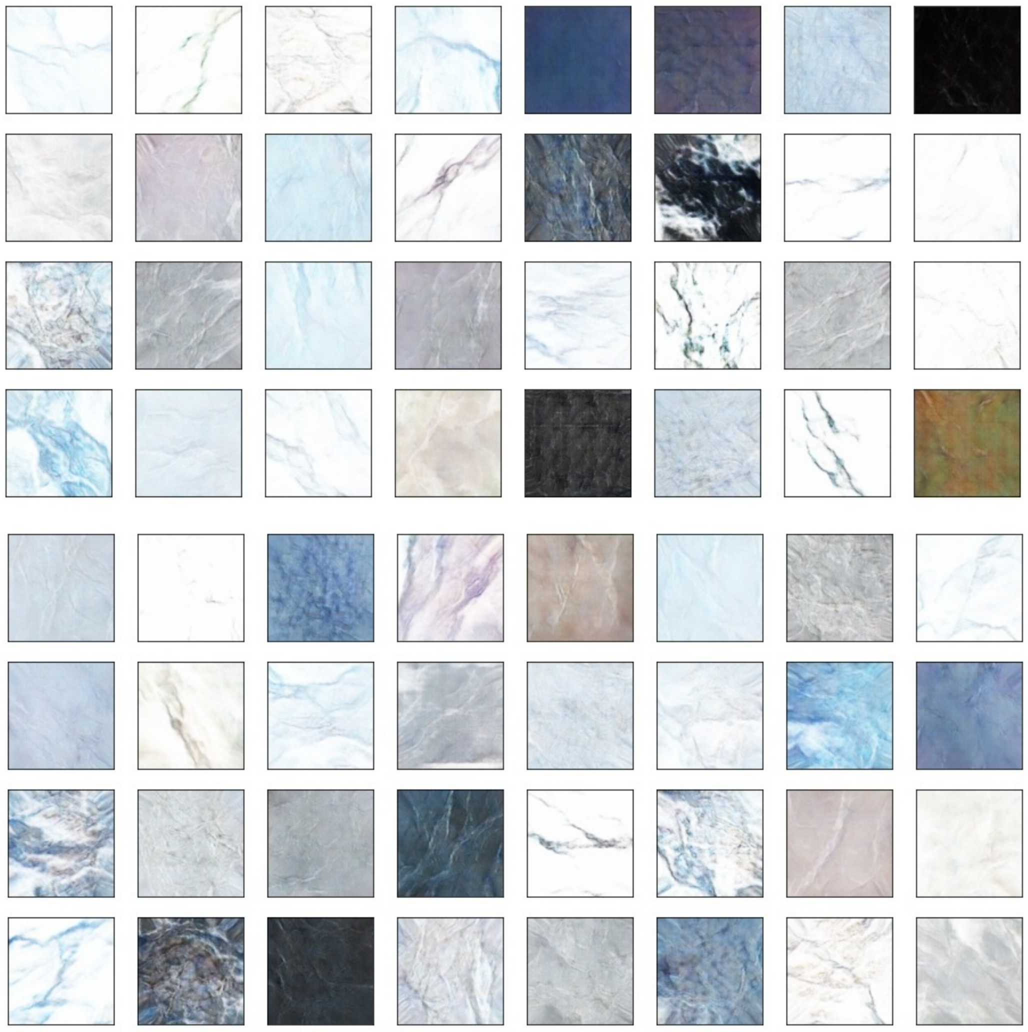


Figure 10: WGAN-GP outputs for training on Marble styled dataset

Resulting outputs produced by the WGAN-GP model training on Brick styled dataset is shown in figure 11. To our observation, the outputs are resembling brick styles quite well (figure 5). These produced outputs not only has variations design-wise but also are diverse in different colors and hues.

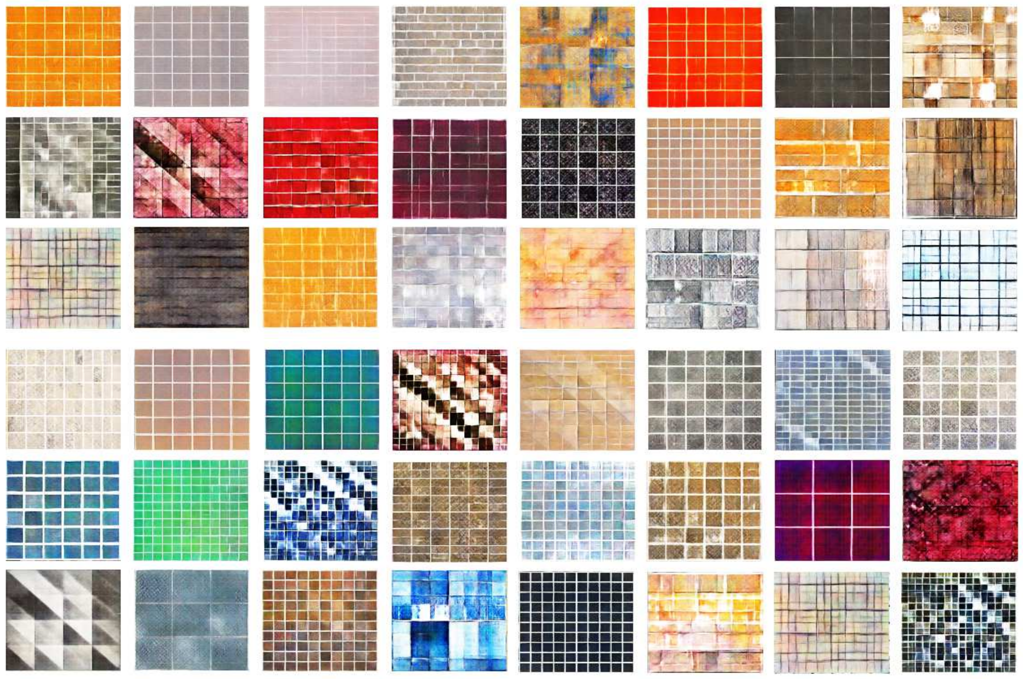


Figure 11: WGAN-GP outputs for training on Brick styled dataset

4.6. Evaluation

The evaluation of generated outputs is a challenging task and there is ongoing research on developing suitable metrics for this purpose[35, 33, 36]. Different metrics might be better suited for different types of outputs, also we believed it was important to consider the subjective judgement of human experts. So we decided to conduct two kinds of evaluation for our case. Using

FID Score [35] and through survey review from professionals related to the ceramic tiles industry.

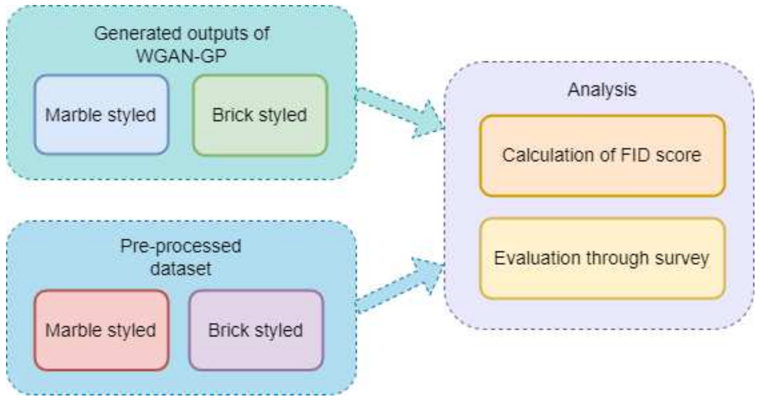


Figure 12: Outputs evaluation flow

4.6.1. Using FID Score

Generative adversarial networks are often evaluated using the Fréchet inception distance (FID) score, a statistical technique that correlates with the perceived quality of generated images. FID score is utilized to assess image quality produced by a generative model [35]. The FID score is defined as the Wasserstein distance squared between two multidimensional distributions(Gaussian): $\mathcal{N}(\mu, \Sigma)$, which represents the distribution of certain features of the GAN-generated images and $\mathcal{N}(\mu_w, \Sigma_w)$, which represents the distribution of the similar features from the real images, which is used to train GAN. By features, it is meant neural network features. Typically, the ImageNet-trained Inception v3 neural network is used for this purpose. The FID can be estimated by calculating the covariance and mean of the feature activations when synthetic and original data are provided into the Inception network [37].

$$FID = \|\mu_r - \mu_g\|^2 + \mathrm{Tr}(\Sigma_r + \Sigma_g - 2(\Sigma_r \Sigma_g)^{1/2}) \tag{2}$$

FID score indicates the degree of similarity between the two sections statistically on the aspects of the original pictures obtained using the image classification model inception v3. In this case, the final pooling layer preceding the classification output of pictures is utilized to collect characteristics (computer-vision-specific) of an input image. Calculating the average and covariance of the images, as a multivariate Gaussian the activations are summed. For the activations, these statistics are derived throughout the whole collection of actual and synthetic images.

Our augmented datasets which were used for training could not be used due to the memory constraints[2] in our CPU. For our training purpose- we used 12000 marble textured and 12000 brick textured images 1, but for calculating FID scores, we could only show the results for the datasets that had only unique images, as its quantity (2163 for marble and 1871 for brick textures) sufficed our memory limits almost to the full extend and therefore would not be enough for the augmented ones. The summary of the FID Score is shown in the following table 4:

[2]Table 3: Training time and hardware for different models

Category	**Images**	**Quantity**	**FID Score**
Marble	Dataset	2163	109.379
	Generated	2464	
Brick	Dataset	1871	148.367
	Generated	1024	

Table 4: FID scores for Marble and Brick styles

For both cases, our generated outputs performed quite well. Marble texture generated outputs seemed more mesmerizing to us and its FID score also came less than that of the brick's.

4.6.2. Through Human Survey

Human evaluation is an important aspect, specially in case of a research on arbitrary image generation like ceramic tiles texture, as it can provide a more nuanced and accurate assessment of the quality of the produced images compared to automated metrics alone. While automated metrics such as Fréchet Inception Distance (FID) provided a quick and efficient way to examine the outputted images, but they are limited in their ability to capture certain aspects of image quality, such as subjective aesthetic appeal, which is for our case, quite significant. By having human evaluators rate the generated images based on various criteria such as realism, coherence, and aesthetic appeal, we can obtain a more accurate and nuanced assessment of the generated images.

The goal of the survey is to evaluate GAN-generated tile designs along with real tiles by human experts. To conduct our survey, we chose not to take reviews from general people as there would remain a possibility that their opinions maybe be biased. Moreover, the professionals who had been working in the ceramic industry (as designers, merchants etc.) could judge the generated outputs better and can give reviews based on the designs' market demand as well. So our survey participants were those individuals whose background aligned with our motive.

Showing only the generated images and getting reviews of them could not ensure a fair outcome of the survey. In order to conduct this comparison-based survey properly, a few of the generated tiles images were mixed up in equal amount with the real ones (from the dataset) and the reviewers were asked to rate the tiles based on the design textures. For preparing the survey sheet, it was made sure not to put all the best-quality generated images in the survey. A variety of generated images that were of good, average and below average according to our view were selected to be in the sheet maintaining a specific ratio (good 2: average 1: below-average 1).

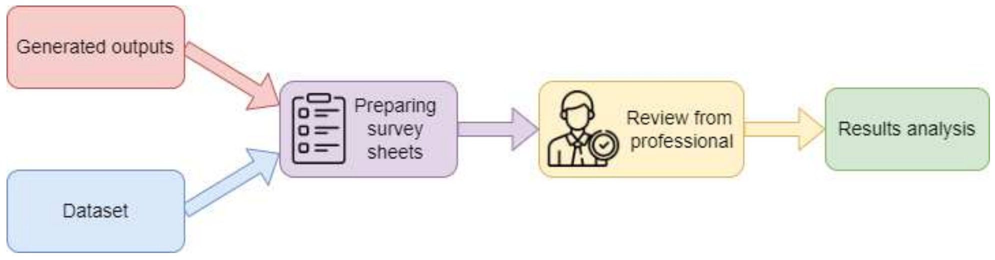


Figure 13: Flow of events of taking survey review

A total of 29 sets of reviews were conducted in the ceramic tiles shops located in Banglamotors, Dhaka. The reviewers were either ceramic tiles shop owners, merchants or designers. The images of these review sheets have been uploaded to this[3] github repository. Each of the sheets contained 8 ceramic tiles texture images shuffled up, where 4 of them were from the training dataset and 4 were from generated ones. Out of the 29, 17 sheets contained marble textured images and the rest 12 were of brick textured images. For each image, 5 options were given to evaluate them: *Very Good*, *Good*, *Average*, *Bad* and *Very Bad*. The participants of the survey would tick/mark only one option that seemed better suited for each corresponding texture to them. Before conducting the survey, each of the reviewers were explained the procedure, so they could give reviews accordingly. The reviewers had no prior knowledge about which of these textures were generated and which of these were real tiles images. In the end, following this procedure, unbiased reviews from the participants were collected. In table 5, the total counting have been shown for all the respective criteria and following after that in table 6, review counts in percentage based on review options are shown.

Style	**Type**	**VG**	**G**	**A**	**B**	**VB**
Marble	Generated	16	25	20	7	0
Marble	Real	22	24	15	7	0
Brick	Generated	12	10	13	12	1
Brick	Real	11	20	10	7	0

Table 5: Review counts from survey. In the columns *VG* stands for Very Good, *G* is for Good, *A* is for Average, *B* is for Bad and *VB* is for Very Bad

[3]`https://github.com/labibabdullah/Survey-documents-of-Devising-Ceramic-Tiles-Design?fbclid=IwAR3cjgvwhUaHjuNutIIjxtcrIdEU3W7pbT69sQrbYI3_NBIOguOblGjgi6M`

Review	**GM**	**RM**	**GB**	**RB**
Very Good	26.2%	36.1%	19.6%	18.1%
Good	31.6%	30.4%	12.7%	25.3%
Average	34.5%	25.9%	22.4%	17.2%
Bad	21.2%	21.2%	36.4%	21.2%
Very Bad	0.0%	0.0%	100.0%	0.0%

Table 6: Review counts in percentage based on review options. (GM: Generated-Marble style, RM: Real-Marble style, GB: Generated-Brick style, RB: Real-Brick style)

Among the reviews (table 6), real-marble styles got the most number (36.1%) of *Very Good*, and in terms of *Good* reviews, generated-marble styles got the most(36.1%). It also got height *Average* reviews (34.5%). For getting *Bad* reviews generated-brick got the most (36.4%) and in the end it also got the only *Very Bad* review as well.

Figure 14 shows the review counts in bar diagram collected for only the generated design textures. Some of the generated designs were very much liked by the reviewers as the designs seemed new and refreshing, while some of the reviewers preferred ones that were currently on par with market trends. Quite a few of the designs got skewed ratings, while some of them got mixed reaction. Out of the three sections shown in the figure, the first two is comprised of generated Marble styled textures and the last one is comprised of generated brick styled textures according to the review sheets.

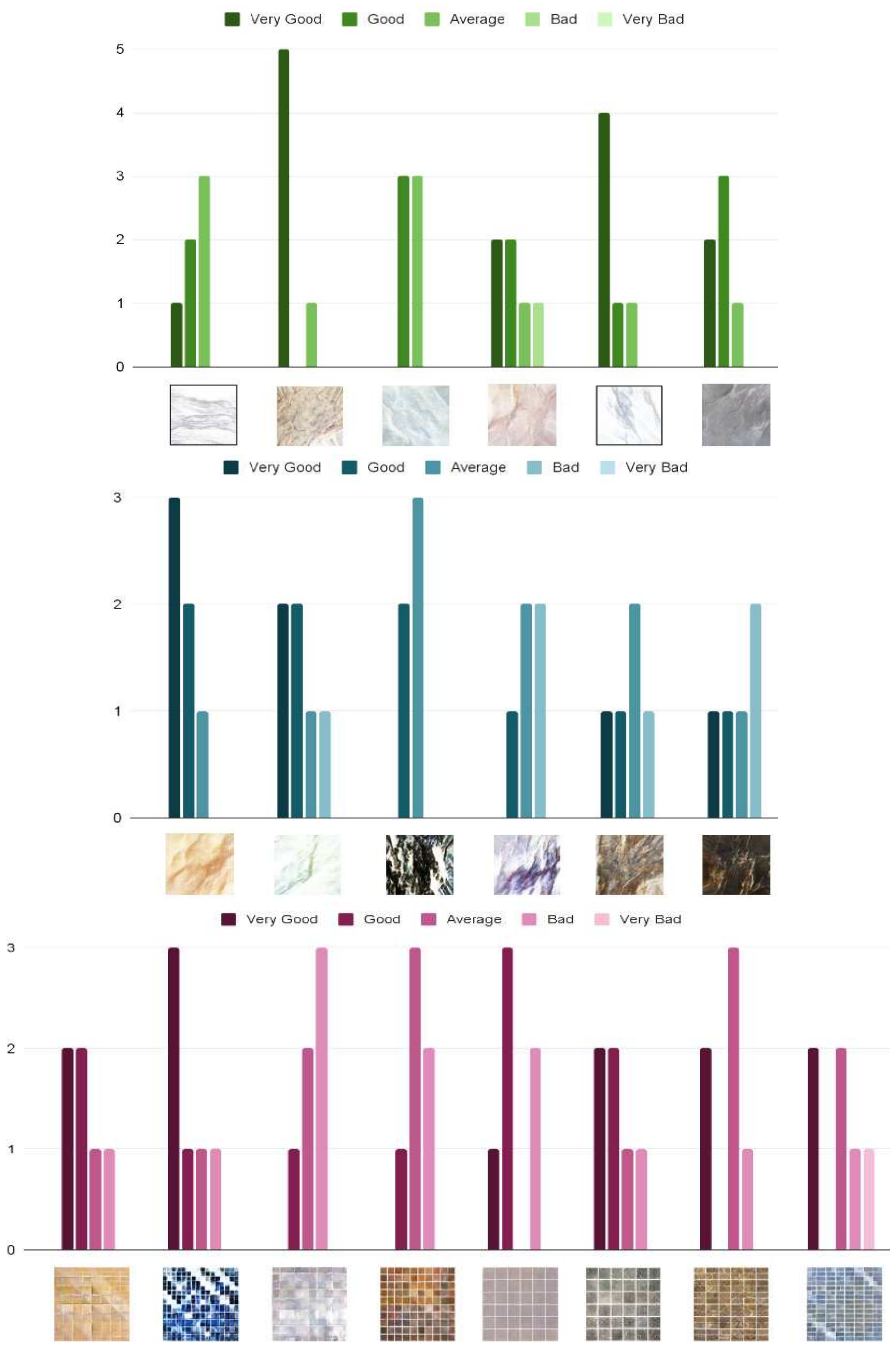


Figure 14: Individual review counts of the generated-surveyed designs

Analyzing table 5 and 6, getting reviewed as *Very Good*, real-marble styles got the most votes- the reason would be the sharpness of wave-like designs in the marble texture dataset that is very much tough to mimic in our generated 128x128 images. In terms of getting *Good* reviewed, generated-marble styles went on head-to-head with real-marble styles as the model could produce low-sharp but decent designs. Similarly, for getting reviewed *Average*, generated-marble styles got a better count than the others. Moreover, the generated-brick style got more *Good* reviews than the training dataset; the reason would be the uniqueness of these designs-according to some of the survey participants. In terms of getting *Bad* reviewed, the generated-brick style got more counts as some of the designs were unrealistic and would not go with the general usage patterns of these kinds of tiles. And only one of the generated-brick styles got *Very Bad* reviewed- for the similar cause.

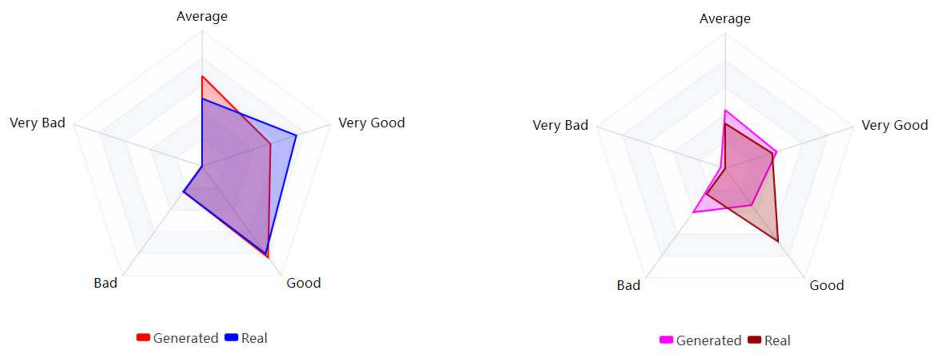


Figure 15: Spider-chart of the survey reviews for individual styles(left: Marble styles, right: Brick styles)

In case of marble-styles, the spider-chart (figure 15) based on table 5 above shows that the majority of the reviewers considered our generated-marble styles and marble-styles from dataset, to be of same quality, hence the common portion is taking the larger part of the graph (left). On the other hand, brick-styles also share a large portion of the graph, at the same time- a big portion from the dataset were considered good and also a decent amount of generated brick-styles were considered bad and average as well.

Style	**Type**	**VG**	**G**	**A**	**B**	**VB**
Marble	Generated	23.5%	36.7%	29.4%	10.4%	0.0%
Marble	Real	32.3%	35.3%	22.0%	10.4%	0.0%
Brick	Generated	25.0%	20.8%	27.1%	25.0%	2.1%
Brick	Real	23.0%	41.6%	20.8%	14.6%	0.0%

Table 7: Review counts in percentage based on style-type. In the columns *VG* stands for Very Good, *G* is for Good, *A* is for Average, *B* is for Bad and *VB* is for Very Bad

Table 7 shows percentage of collected reviews among the separate style and type. It is observed that 89.6% of the reviews for generated-marble designs were between *Very Good* to *Average* which matches the number for real-marble design for the same range. In this major portion, most of it came from textures that were reviewed as *Good* by the reviewers. And for the generated-brick styles it is 72.9% which is less than its counter part real-brick styles (85.4%), is also appreciable. The most review counts obtained for generated-brick styles were of *Average* counts.

5. Conclusion

Primary objective of our work is to see how well generative adversarial networks perform to mimic the designs for ceramic tiles. Through the use of a semi-automated pre-processing method on our collected dataset, we were able to generate a variety of vibrant and realistic ceramic tile designs. DC-GAN model, due to mode collapse, produced variety of colored outputs for a distinct few texture patterns. Our findings indicate that WGAN-GP model produced the most visually pleasing and diverse designs between the two models. Evaluation using the FID score provided positive feedback overall for both of the design types- keeping marble styled generated designs a bit ahead and then response from the survey-participants in the field of ceramic tile industry supported that outcome as well.

While the limitations of our study were largely due to hardware and memory constraints, future research could benefit from more advanced computational resources, along with using other state-of-the-art generative architectures. It is planned to make our pre-processed datasets available in near future. Our findings suggest that there lies significant potential for this type of work to impact current ceramic tiles industry by semi-automating or automating texture designing process to reduce overall production time or cost-cutting and also to change the age-old view of the designs of ceramic tile textures. Our work may not replace human design but it could be an inspiration for them to produce better styled design tiles.

6. References

References

[1] J.-P. Briot, G. Hadjeres, F.-D. Pachet, Deep learning techniques for music generation, Vol. 1, Springer, 2020.

[2] V. Dumoulin, J. Shlens, M. Kudlur, A learned representation for artistic style, arXiv preprint arXiv:1610.07629 (2016).

[3] L. A. Gatys, A. S. Ecker, M. Bethge, Image style transfer using convolutional neural networks, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 2414–2423.

[4] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, Y. Bengio, Generative adversarial networks, Communications of the ACM 63 (11) (2020) 139–144.

[5] A. Radford, L. Metz, S. Chintala, Unsupervised representation learning with deep convolutional generative adversarial networks, arXiv preprint arXiv:1511.06434 (2015).

[6] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, A. C. Courville, Improved training of wasserstein gans, Advances in neural information processing systems 30 (2017).

[7] S. Khalil, M. Taha, R. Ali, H. Mehmood, H. Dilpazir, Generation of textile patterns through generative adversarial networks, Sustainability 13 (15) (2021) 8271.

[8] R. A. Fayyaz, M. Maqbool, M. Hanif, Textile design generation using gans, in: 2020 IEEE Canadian Conference on Electrical and Computer Engineering (CCECE), IEEE, 2020, pp. 1–5.

[9] D. P. Kingma, M. Welling, Auto-encoding variational bayes, arXiv preprint arXiv:1312.6114 (2013).

[10] F. Pérez-Cruz, Kullback-leibler divergence estimation of continuous distributions, in: 2008 IEEE international symposium on information theory, IEEE, 2008, pp. 1666–1670.

[11] M. Araya-Polo, C. A. Frogner, C. Zhang, H. Mobahi, T. A. Poggio, Learning with a wasserstein loss, in: International Conference on Learning Representations, 2015, pp. 1–14.

[12] S. S. Nasrin, R. I. Rasel, Hennagan: Henna art design generation using deep convolutional generative adversarial network (dcgan), in: 2020 IEEE International Women in Engineering (WIE) Conference on Electrical and Computer Engineering (WIECON-ECE), IEEE, 2020, pp. 196–199.

[13] D. H. Kim, Deep convolutional gans for car image generation, arXiv preprint arXiv:2006.14380 (2020).

[14] N. Gervais, The car connection picture dataset, `https://github.com/nicolas-gervais/predicting-car-price-from-scraped-data/tree/master/picture-scraper` (2020).

[15] A. Kini, P. Rama Devi, S. Shylaja, Towards the generation of underwater images using generative adversarial networks, in: International

Conference on Advances in Computing, Communications and Informatics, Springer, 2019, pp. 1284–1289.

[16] O. Adedeji, P. Owoade, O. Ajayi, O. Arowolo, Image augmentation for satellite images, arXiv preprint arXiv:2207.14580 (2022).

[17] P. Helber, B. Bischke, A. Dengel, D. Borth, Eurosat: A novel dataset and deep learning benchmark for land use and land cover classification, IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing 12 (7) (2019) 2217–2226.

[18] A. Brock, J. Donahue, K. Simonyan, Large scale gan training for high fidelity natural image synthesis, in: International Conference on Learning Representations, 2019, pp. 1–14.

[19] X. Huang, Y. Li, O. Poursaeed, J. Hopcroft, S. Belongie, Stacked generative adversarial networks, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2018, pp. 1861–1870.

[20] Y. Wang, Y. Yao, J. T. Kwok, L. M. Ni, Generalizing deep learning for medical image segmentation to unseen domains via deep stacked transformation, IEEE transactions on medical imaging 38 (11) (2018) 2638–2649.

[21] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, Z. Wojna, Rethinking the inception architecture for computer vision (2015). `arXiv:1512.00567`.

[22] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, L. Fei-Fei, Imagenet: A large-scale hierarchical image database, in: 2009 IEEE conference on computer vision and pattern recognition, Ieee, 2009, pp. 248–255.

[23] J. MacQueen, Classification and analysis of multivariate observations, in: 5th Berkeley Symp. Math. Statist. Probability, University of California Los Angeles LA USA, 1967, pp. 281–297.

[24] P. Korus, An overview of image similarity, in: Advances in Intelligent Systems and Computing, Vol. 728, Springer, 2018, pp. 303–311.

[25] V. Grinberg, Imagehash: Library implementing image hashing algorithms, `https://github.com/JohannesBuchner/imagehash` (2018).

[26] A. Krizhevsky, I. Sutskever, G. E. Hinton, Imagenet classification with deep convolutional neural networks, Communications of the ACM 60 (6) (2017) 84–90.

[27] H. Zhang, I. Goodfellow, D. Metaxas, A. Odena, Self-attention generative adversarial networks, in: International conference on machine learning, PMLR, 2019, pp. 7354–7363.

[28] M. Arjovsky, S. Chintala, L. Bottou, Wasserstein generative adversarial networks, in: International conference on machine learning, PMLR, 2017, pp. 214–223.

[29] J. Smith, J. Doe, A. Johnson, A comparative study of wgan-gp and dcgan for generating realistic facial images, Journal of Artificial Intelligence 35 (2) (2022) 57–68.

[30] Tile Council of North America, Ceramic tile, `https://www.tcnatile.com/faqs/31-ceramic-tile.html` (n.d.).

[31] L. Battaglia, Ceramic tile vs. marble: Which one is better for flooring?, The Spruce (June 12 2020).
URL https://www.thespruce.com/ceramic-tile-vs-marble-1821743

[32] Daltile, Brick, https://www.daltile.com/tile-product-category/brick (n.d.).

[33] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, X. Chen, Improved techniques for training gans, Advances in neural information processing systems 29 (2016) 2234–2242.

[34] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, Y. Bengio, Generative adversarial nets in advances in neural information processing systems (nips), Curran Associates, Inc. Red Hook, NY, USA (2014) 2672–2680.

[35] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, S. Hochreiter, Gans trained by a two time-scale update rule converge to a local nash equilibrium, Advances in neural information processing systems 30 (2017).

[36] A. Borji, Pros and cons of gan evaluation measures, Computer Vision and Image Understanding 179 (2019) 41–65.

[37] D. Dowson, B. Landau, The fréchet distance between multivariate normal distributions, Journal of multivariate analysis 12 (3) (1982) 450–455.