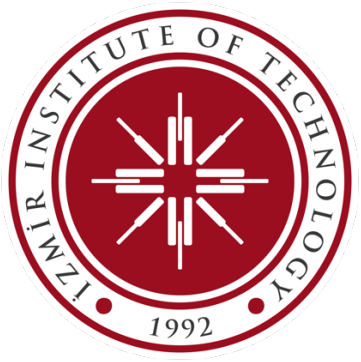


DEPARTMENT OF ELECTRICAL AND ELECTRONICS ENGINEERING

EE492 PROJECT REPORT

3D OBJECT GENERATION BY DEEP LEARNING

Behiye Erdemir 240206013

Ömer Faruk Karadaş 230206040

ADVISOR: FERİT ACAR SAVACI

DATE: 13/07/2020

ABSTRACT

Recently, innovations in artificial intelligence (AI) applications have increased rapidly, and with the effect of these improvements, AI has been used for much more than making simple predictions. Also, with the development of technology and increasing consumption, original and creative designs are needed for the industrial design sector. One of the stages of production that has been going on for years is designing, but it is difficult for designers to produce original and innovative designs that are sufficient for this consumption rate. With the developments in artificial intelligence in recent years, designs strengthened by artificial intelligence have started to be used in the industrial sector. With GAN's high amount production capability, many designs can be created instantly compared to traditional humanmade designs. Our aim in this project is to provide a modern approach to the industrial design sector by producing designs close to the industrial level with the creativity of GAN. It also includes examining the effects of Normal and Levy type Alpha stable distributions and the label smoothing method on the loss function.

TABLE OF CONTENTS

ABBREVIATIONS

GAN: Generative Adversarial Network

CNN: Convolutional Neural Network

AI: Artificial Intelligence

LS: Label Smoothing

Alpha-stable: α-stable

GPU: Graphics Processing Unit

TPU: Tensor Processing Unit

CAD: Computer Aided Design

LIST OF FIGURES

LIST OF TABLES

1. INTRODUCTION

Recently, innovations in artificial intelligence (AI) applications have increased rapidly, and with the effect of these improvements, AI has been used for much more than making simple predictions. It is now possible to see studies on artificial intelligence in many technical areas [1] [2]. One of the impressive works in this field is the processing of 3 dimensional (3D) sensor data used in many sectors such as automatic driving, medicine, and industrial design and making it meaningful for people [3] [4]. The use of 3D models produced creatively by generative adversarial networks (GAN) [5] at the industrial design level will open the door to a new era for the design industry. By interfering with the structure of the GAN's responsible for producing 3D models, designs whose structures and dimensions have been controlled, not previously designed, suitable for use in industry, and with added values can be produced. With this motivation, we aimed to design a GAN model to generate new and original objects that are important in industrial design.

1.1 Convolutional Neural Network

A Convolutional Neural Network (CNN) [6] is a Deep Learning algorithm that finds out the differences of the input by upholding spatial aspects throughout the network. Unlike regular neural networks, instead of neurons being connected to every neuron in the previous layer, they are only connected to neurons close to it and all have the same weight in CNN. A CNN is made up of multiple layers mainly are convolutional layer, pooling layer, fully connected layer.

- **Convolutional Layer:** Convolutional Layer as a keystone of CNN uses filters to capture the high-level features of the input. As a result of the convolutional operation between the filter and the input, the output matrix is called future map. [6]
- **Pooling Layer:** The Pooling Layer is a down sampling operation which applied after the convolutional layer to get more robust a feature map to changes in the position of the feature in the input. Differently in GANs, down sampling is provided by changing the stride instead of using a pooling layer. [6]
- **Fully Connected Layer:** In Fully Connected Layer, the flattened input is connected to all neurons. The classification and prediction are driven by this operation. [6]

1.2 Generative Adversarial Network

With the breakthrough of GANs [5] [1] [7], its structure is developed and optimized over the years [8] [9]. The GANs are mainly used for generation of images and a state-of-art research [10] showed that a photo-realistic output can be obtained with controllable parametrization of GANs by starting with low resolutions and increasing the resolutions gradually.

- **Deconvolution Layer:**

  Deconvolution (transpose convolution, up convolution) [11] is a mathematical approach based on doing the inverse of the convolution process. The goal is to get high dimensional output from the input, which bears the basic characteristic of the input by passing a designated input through a filter. This process is based on creating the output by passing each input value through that filter and adding the results in a row. For a better explanation, a simple illustration of the deconvolution layers is given in **Figure 1**. Using

the deconvolution layers in the generator is based on updating the deconvolution filter weights every iteration, minimizing the loss function and obtaining the desired output.

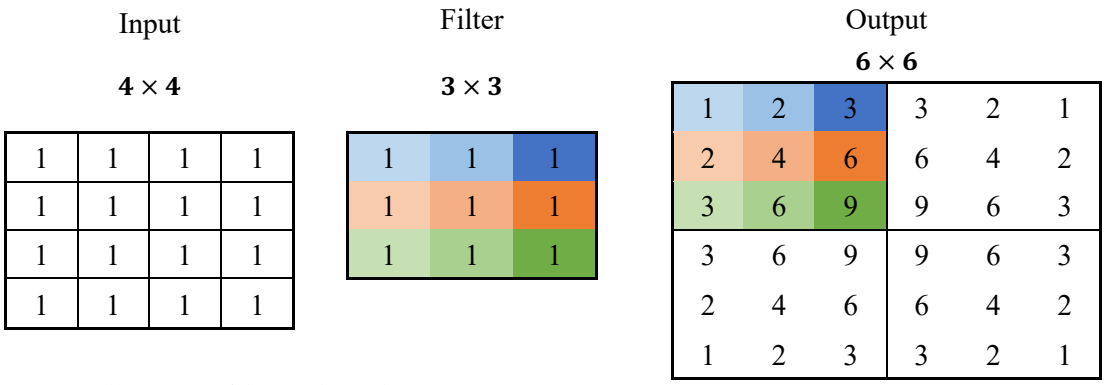


Figure 1: *Illustration of deconvolution layer*

- **Dense Layer:** In the generator model, a fully connected layer (*Dense*) [6] was used to make $\mathbb{Z}$ ($\mathbf{200 \times 1}$) in the input layer suitable for the input of the first deconvolution layer ($\mathbf{4 \times 4 \times 4 \times 512}$).
- **Batch Normalization Layer:** Performance improvement is aimed by scaling the output of deconvolution layers using a batch normalization layer [12].
- **Leaky Rectified Linear Unit (ReLU) Activation Function:** As seen in **Figure 2**, Leaky ReLU [6] allows a small negative value to pass due to the characteristic of the activation function. This is important for feature GANs, because the only way a generator has to learn is to gradient gradients from the discriminator. Consequently, Leaky Relu activation function is used in the hidden layers. In the output layer, the sigmoid activation function is used because the 3D models produced are binary.

$$y(x) = \begin{cases} 0.01x & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$$

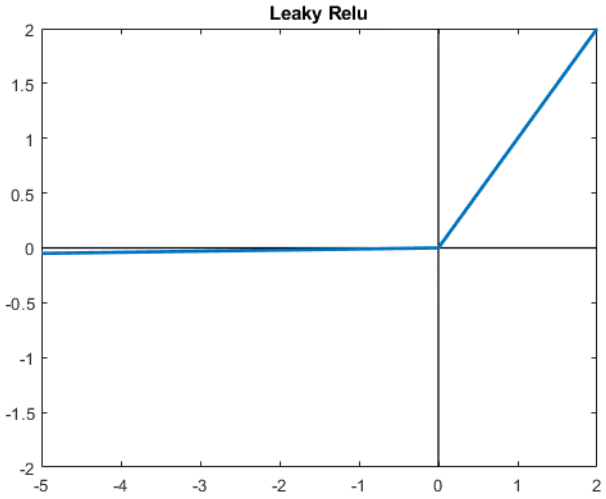


Figure 2: *Characteristic of Leaky ReLU activation function*

- **Sigmoid Activation Function:** In artificial neural networks, the activation functions define the output of the model and its accuracy. The sigmoid function [6] that is used in our model, aims to predict the probability as an output. If the output of a GAN is in range (0,1), it is convenient to use the sigmoid function. For a different GAN model, if the output is in range (-1,1), tanh activation function should be used.

$$S(x) = \frac{L}{1 + e^{-a*(x-b)}}$$

Equation 1: *L is the curve's maximum value, b is the x value of the curve's midpoint, a is the logistic growth rate*

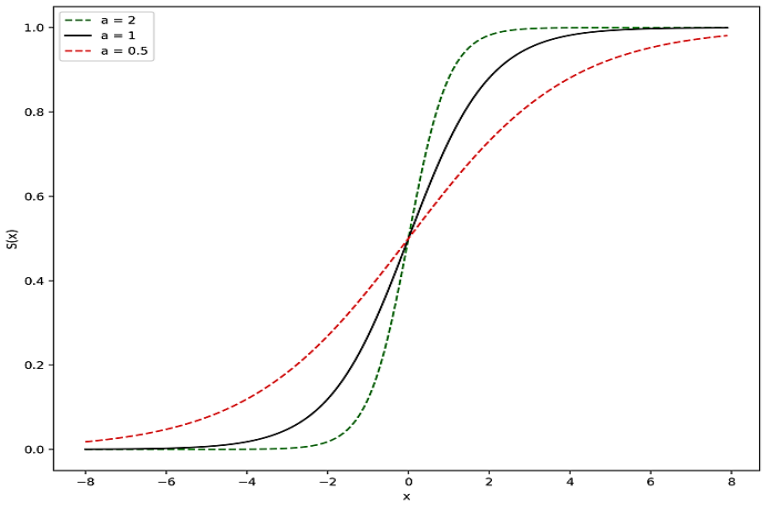


Figure 3: *The sigmoid activation function curves for L=1, b=0 and a=2, 1, 0.5 respectively*

The output matrix of our GAN model is in range of **(0,1),** hence we use the standard sigmoid function where $\boldsymbol{L = 1, b = 0}$, and $\boldsymbol{a = 1}$ in **Equation 1.** As seen in **Figure 3**, characteristic of curve varies with different values of $\boldsymbol{a}$**.**

- **Binary Cross-Entropy Loss Function:** Loss functions measure how well the model is doing on the dataset, hence control of the complexity can be observed from the value of the loss. Furthermore, the selection of the loss function is directly related to the activation function used. The way to calculate the error properly is to choose a loss function that matches the pattern in the output. For this project, our discriminator model classifies data as 1 for real and 0 for fake using the sigmoid function. We use binary cross-entropy [5] as the loss for quantifying distance distribution between the generated and the real data,

$$V(D, G) = [log\ D(x)] + \left[log\left(1 - D(G(z))\right)\right],$$

Equation 2: *x is the real data, z is the given noise randomly distributed, D(x) is discriminator's prediction on the real data, G(z) is generator's output when given noise z, D(G(z)) is the prediction of discriminator on the fake data* [5]

For the first term, discriminator aims to D(x) to be a large number because discriminator represents high confidence that a real sample is real. On the other hand, discriminator expects to D(G(z)) to be small as possible to represent a fake sample. However,

the generator aims to maximize D(G(z)) to deceive discriminator. With the inverted probability in the second term, the discriminator wants to maximize, and the generator wants to minimize the overall formula in **Equation 2**.

1.3 Point Clouds

A point cloud is the projection of a 3D object in a space consisting of only points. 3D laser scanners or sensors give coordinate points in the cartesian space as output data, and the combination of these points is called point clouds. Point clouds can be seen in applications such as scanning 3D environments such as objects or buildings, 3D object design (CAD models), interpretation of autonomous vehicle environment sensors data. The output of the generator in Generative adversarial Networks that produce 3D objects is a 3D matrix. As a result, the output produced by the generator is a point cloud. The sample output of our GAN model can be seen in **Figure 4** to be an example for Point Clouds.

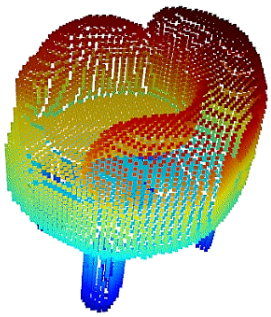


Figure 4: *3D point cloud view of example obtained by generator network*

1.4 Polygonal Mesh

Polygon meshes are one of the methods used for 3D modeling. Modeling consists of diagonals, faces and edges on the surface. The goal is to create a polynomial solid body by combining the corner points to create triangular, rectangular or more complex faces. The point cloud needs to be reconstructed to transform the point clouds into a 3D object. Approaches such as Delaunay triangulation, alpha shapes, ball pivoting, Marching cubes [13], and Non-uniform rational B-spline can be used for reconstruction. In this project, the point cloud obtained from the generator network was reconstructed with the marching cube approach for creating a 3D triangular mesh. In **Figure 5**, an example of the point cloud obtained from the generator model turned into a triangular mesh.

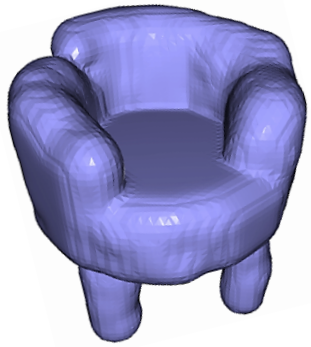


Figure 5: *3D Triangular Mesh view of example obtained by generator network*

1.5 Alpha-Stable Noise Distribution

Alpha-stable (α-stable) noise distribution [14] can be used for modeling extreme cases such as earthquakes or economic crisis. The distribution is a four-parameter family of distributions. It is denoted by S_α (β, ɤ, μ). The characteristic function [14] of α-stable Levy noise is given in Equation 3,

$$\emptyset(\theta) = \begin{cases} exp\left\{j\mu\theta - \gamma^{\alpha}|\theta|^{\alpha}(1 - j\beta sign(\theta)\tan\left(\frac{\alpha\pi}{2}\right))\right\} \; if \; a \neq 1 \\ exp\left\{j\mu\theta - \gamma|\theta|(1 + j\beta\frac{2}{\pi}sign(\theta)\ln\left(\frac{\alpha\pi}{2}\right))\right\} \; if \; a = 1 \end{cases}$$

Equation 3: *a is characteristic exponent $(0 < \alpha \leq 2)$, β is the skewness parameter $(-1 \leq \beta \leq 1)$, γ is the dispersion parameter $(\gamma \geq 0)$, μ is the location parameter $(\mu \epsilon R)$.*

where,

$$sign(x) = \begin{cases} -1 & if\ x < 1, \\ 0 & if\ x = 0, \\ 1 & if\ x > 1. \end{cases}$$

Besides, distributions dependent on the $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ parameters are shown in **Figure 6**.

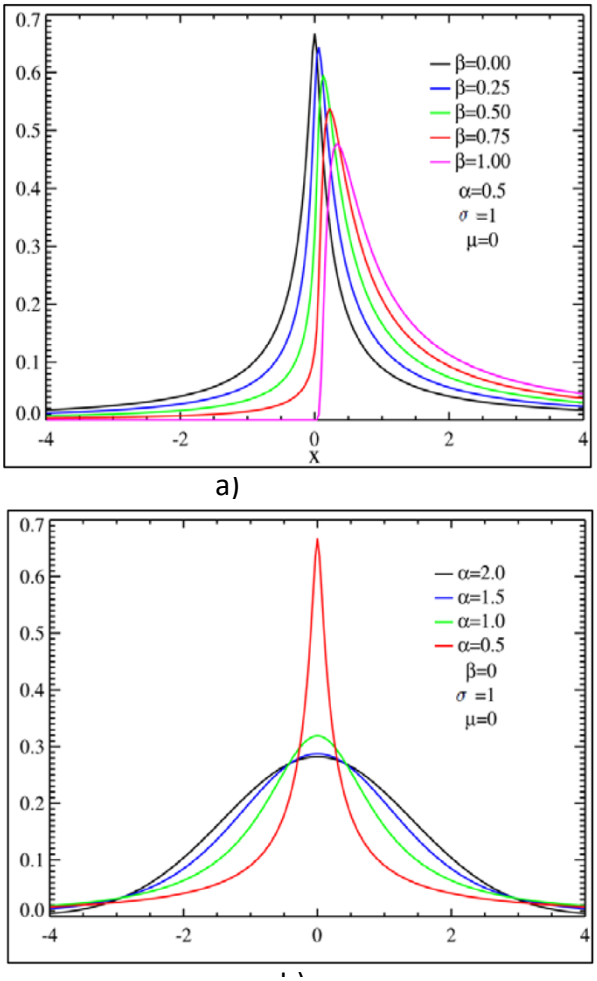


Figure 6: a) Distribution dependent on α parameter b) Distribution dependent on β parameter [14]

2 PROBLEM DEFINITON

With the development of technology and increasing consumption, original and creative designs are needed for the industrial design sector. One of the stages of production that has been going on for years is designing, but it is difficult for designers to produce original and innovative designs that are sufficient for this consumption rate. Hence, the focus of our project is to create industrial products designed with artificial intelligence instead of traditional methods.

3 PROPOSED SOLUTION

With the developments [15] in AI in recent years, designs strengthened by AI have started to be used in the industrial sector. With GAN's high amount production capability, many designs can be created instantly compared to traditional humanmade designs. Our aim in this project is to provide a modern approach to the industrial design sector by producing designs close to the industrial level with the creativity of GAN.

3.1 Implementation of GAN architecture

The first step to implement the solution of the problem is to create a GAN with the qualities suitable for the problem. Since GANs require a lot of computational costs, it is preferred to derive models that are known to work stable. For this reason, while designing a GAN compatible with 3D space, 3D-GAN [16] model structure developed by MIT Computer Science & Artificial Intelligence Lab (MIT-CSAIL) was preferred. The block diagram of our GAN is shown in **diagram 1** and detailed version in **Appendix 1**.

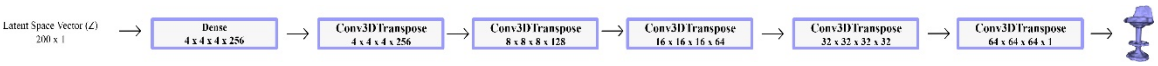


***Diagram 1**: Generator model block diagram*

In this project, it is aimed to design a GAN that works stable to produce to the targeted output and complies with the input and output specifications as shown in **Table 1**.

GAN Model Design Specifications	
Parameters	**Value**
Training dataset	800 chairs
Dimension of input vector ($\mathbb{Z}$)	200×1
Dimension of output matrix	$64 \times 64 \times 64$

***Table 1**: Our model design specifications*

In the GAN literature, the $\mathbb{Z}$ vector is usually selected as normal distribution in $\mathbb{R}^{\boldsymbol{d}}$ [9]. While choosing the dimension of $\mathbb{Z}$ **($\boldsymbol{d}$),** the preferred dimension is used in the 3D-GAN model structure. According to the studies [17, 18], Lévy type distribution used for getting stable result for neural network and spiking neuron models. In our study, it was aimed to observe the effect of the noise distribution type of $\mathbb{Z}$ by using uniform, normal and Lévy type Alpha-Stable distribution.

3.2 Label Smoothing

After the high iterations number of training, the behavior of loss values both the discriminator and the generator can be observed clearly. Label Smoothing (LS) [19] prevents the overconfident behavior caused by disruption of the balance between the generator and the discriminator. The LS proposes a robust model by changing the classification method as smooth labeling instead of binary classification.

$$\boldsymbol{y_k^{LS}} \;\; = \boldsymbol{y_k}\,(\mathbf{1} - \boldsymbol{\alpha}) + \;\frac{\boldsymbol{\alpha}}{\boldsymbol{K}},$$

Equation 3: *y_k^{LS} is the smoothed label, y_k is the one-hot encoded label vector, α is the label smoothing parameter, K is the number of label classes*

There are two label classes; y_k is '1' for real samples and '0' for fake samples in our model. In Equation 3, α was determined according to the studies [16] as **0.1** to provide that the real examples to have the largest probability like **0.95** and fake images to have the smallest probability like **0.05**.

3.3 Interpolation

Interpolation is a good starting point when it is desired to produce variations between the two objects produced. Briefly, interpolation is a form of analysis based on estimating unavailable (unknown) values using linear, circular or different mathematical methods using existing (known) values. Our goal in using interpolation is to divide the latent vectors of two objects into $\boldsymbol{n}$ (in our case $\boldsymbol{n}$ is **7)** parts and observe the transition between them. In the application of interpolation on GAN, there are options available. Interpolation can be done in the latent space of the model as in 3DGAN [16] or it can be done in the Interlayer layers of the model as in StyleGAN [10].

4 RESULTS AND DISCUSSIONS

So far, our GANs with Latent vector (ℤ) Normal and Uniform distribution have been trained for **35,000-iterations** and took approximately **140 hours**. Also, **15,000-iterations** were trained Lévy type Alpha-Stable distribution. The network must continue to be trained to achieve results at the industrial level. Sample outputs produced with our GAN can be seen in **Figures 5, 6, and 7**. Structural problems are seen in the network outputs, since the networks are not sufficiently trained yet. The examples in **Figure 6** belong to the **15000th** iterations of GAN trained with latent vectors obtained by Lévy type Alpha-Stable distribution. The examples in **Figure 7 and 8** belong to the **35000th** iteration of GAN trained with latent vectors obtained with normal and uniform distribution.

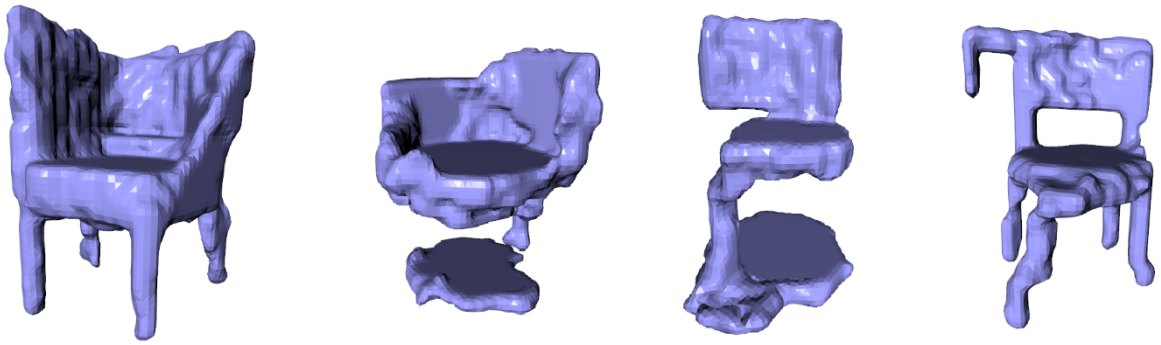


Figure 7: *Sample Network output of GAN trained with latent vectors obtained with Lévy type α-stable distribution*

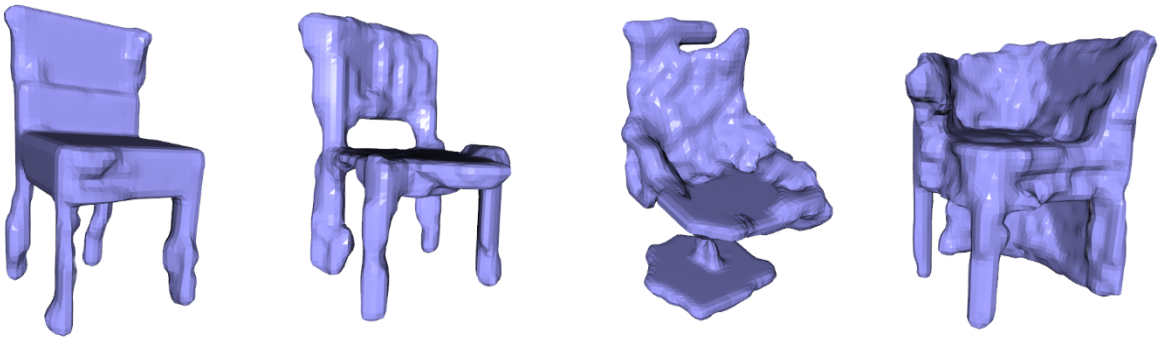


Figure 8: *Sample Network outputs of GAN trained with latent vectors obtained with normal distribution.*

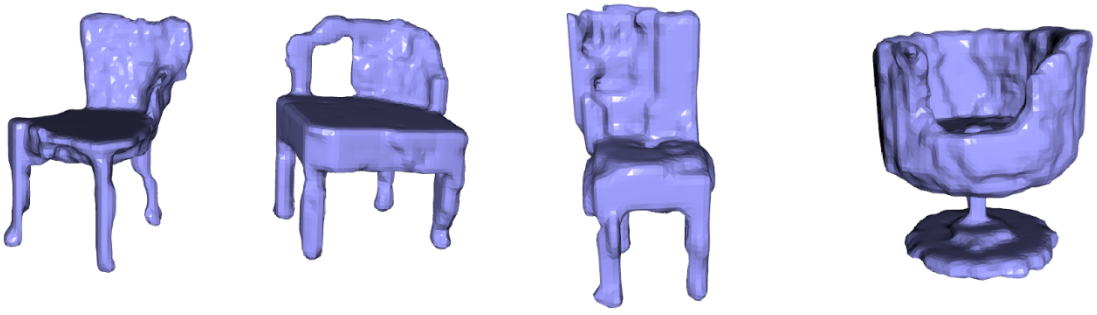


Figure 9: *Sample Network outputs of GAN trained with latent vectors obtained with uniform distribution.*

As seen in **Figure 9**, to remove unwanted noise at the output and improve model quality, the output passed on postprocessing like correlation and Gaussian smoothness filter. Marching cube algorithm [13] was used to the reconstruction of the obtained point cloud ($64 \times 64 \times 64$) into the triangular mesh structure.

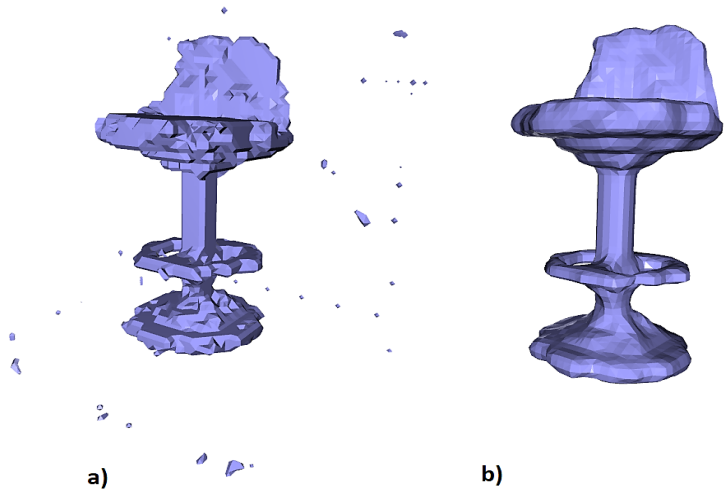


Figure 10: *Effect of postprocessing filters a) Without any filters b) After noise reduction filters*

As observed in **Figure 10**, there are unstable & high values in both generator and discriminator loss plots. Studies [19] have suggested that Label Smoothing (LS) can be used to prevent this problem. In **Figure 10**, in order to observe the effect of LS, GAN model is retrained with and without LS, and Binary Cross entropy loss graph of generator and discriminator is obtained for both cases.

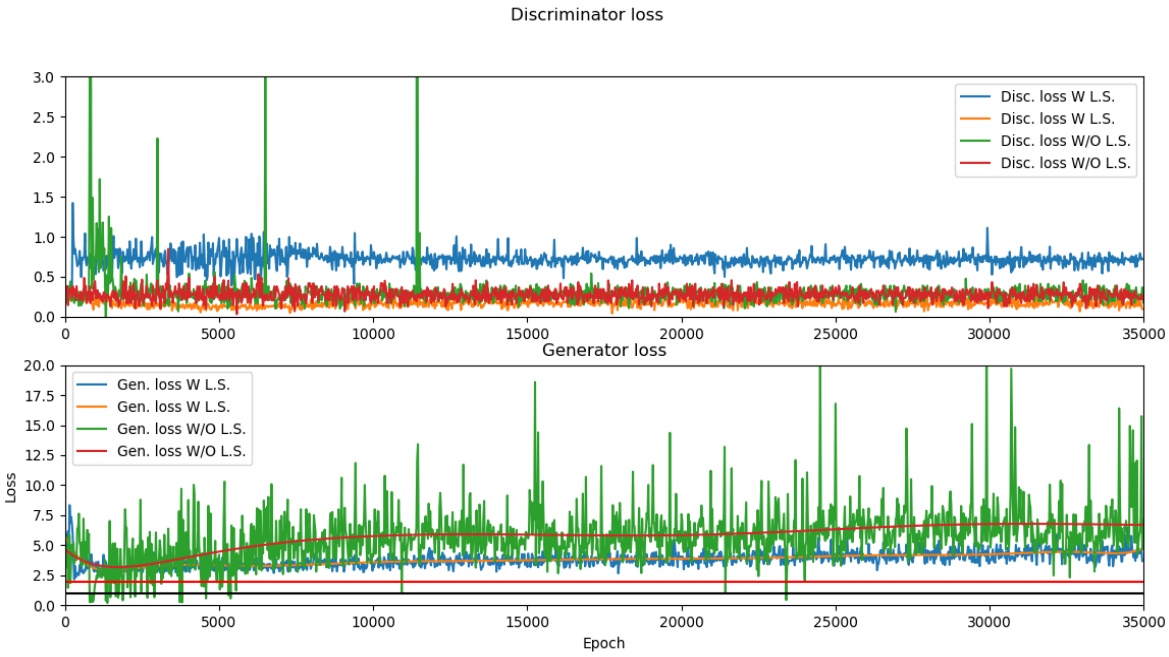


Figure 11: *Binary Cross entropy losses of both Generator and Discriminator with and without Label smoothing on Normal Distribution Latent vector*

In addition, as expected from previous studies [18, 17, 20], Lévy type Alpha-Stable distribution loss values can be observed in **Figure 11**, where it has a more stable characteristic than the normal

distribution. Thus, Lévy type Alpha-Stable distribution is observed to be more suitable for neural network structures than other distributions.

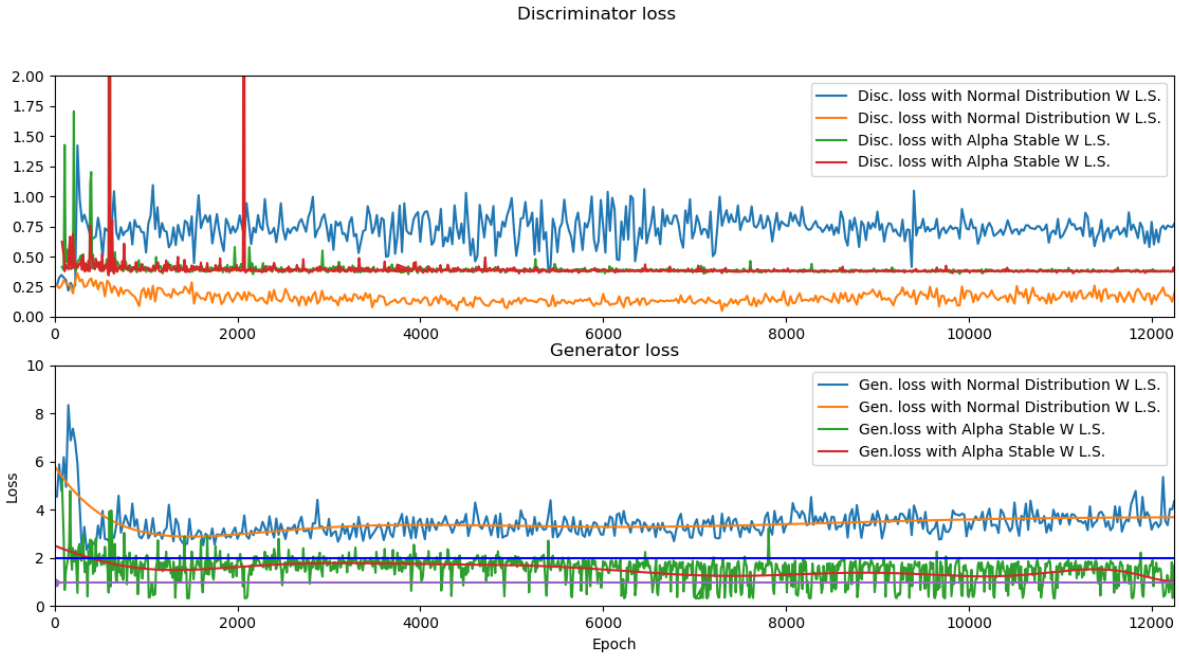


Figure 12: *Binary Cross entropy losses of both Generator and Discriminator with and without Label smoothing on Normal and Lévy type α-stable distribution latent vector*

As can be seen in **Figures 12 and 13**, variations of 3D models produced by linear interpolation can be obtained. However, variations of linear interpolation are not at the desired level in order to achieve the desired output. At this point, the desired output can be achieved with higher control by interfering with the intermediate layers of the generator network as in StyleGAN [10].

Figure 13: *Linear interpolation of Latent vector (ℤ) of two chairs*

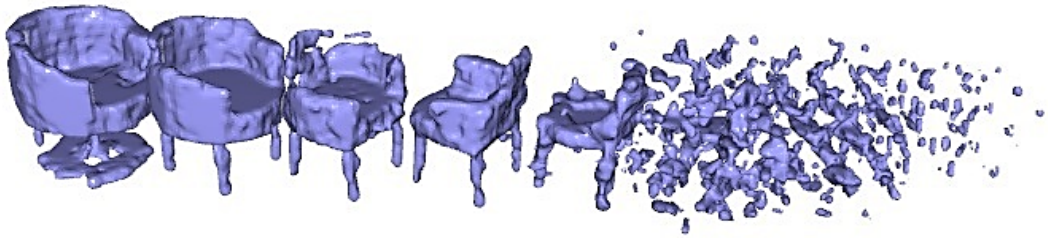


Figure 14*: Linear Interpolation between latent vector (ℤ) of chair and pure noise*

In more detail, in order to train the GAN model in the fastest way, its training was done on the supercomputers of the National Center for High-Performance Computing (UHEM)[1] and Google Collaboratory[2] using GPU. The network model is built on Python and using TensorFlow module for optimized and fast operation of GAN model. While training GAN model, ShapeNet [21] training dataset was used.

5 CONCLUSIONS

With the increasing variety of 3D modeling usage areas such as architecture, video games, engineering, and movies, fine detailed and varied modeling requests emerged. At this point, the new production methods can provide diversity outside of traditional methods. In this project, a GAN architecture was established to generate novel and original 3D objects with the motivation of contributing to the interaction of 3D modeling with AI. The success of the generated objects show that this interaction can be open to a new era for industrial design with improving the architecture of GAN. Therefore, Label Smoothing, which is the method has been successful to stabilize the network. Secondly, the training the network with various noises such as Normal Distribution and Lévy type α-stable distributions affected the stability of the network and variety of the generated samples.

In addition, it was demonstrated that the spatial transition between two objects by interpolating their latent space vectors ($\mathbb{Z}$). It is observed that various mathematical operations for interpolation have meaningful effects on the generated objects. This process allowed for the exploration of the behavior of the latent space vector.

[1] Istanbul Technical University National Center for High Performance Computing: http://en.uhem.itu.edu.tr
[2] Google Collaboratory: https://colab.research.google.com/

6 FUTURE WORK

The new approach of StyleGAN [10] offers control over the style of the generated instances by varying the style vectors and noise at different points. With the implementation of this intervention on the generator model, we can propose a new way to generate objects in the scope of desired parameters. Unlike the classical generator model, noise and latent space vector are given in different proportions to the layers in this approach. Parameters that are the detailing, interpolation proportion between samples, changing the style come with the new generator model. Thus, with the creation of these control parameters on the objects to be generated, the production and variety of objects suitable for use in the industrial area can be provided.

Another concept that can be implemented is transferring a texture to the generated objects. 3D model repositories such as ShapeNet [20] includes the 3D models without the textures. Being able to see the designs as people use them in real, gives us the opportunity to evaluate them in terms of manufacturability. Successful studies on the texture transferring [22] can be taken as a starting point to apply this concept to our project. To visualize the object with the texture, a 3D sketch was made with traditional methods as can be seen in **Figure 15**.

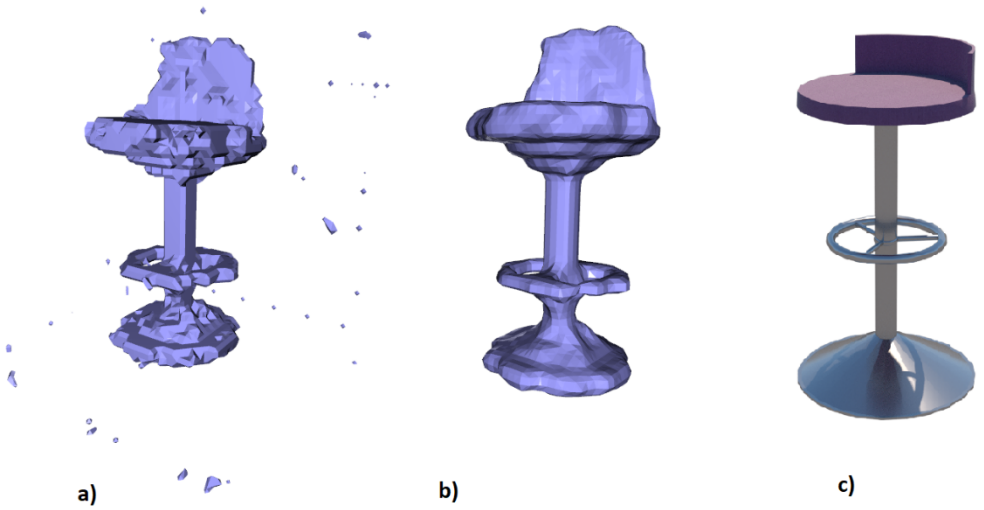


***Figure 15**: a) 3D object with noise b) 3D object after noise reduction c) 3D object with texture*

On the other hand, our network is trained on Graphics Processing Units (GPUs) that reduces the training time significantly. For increasing the number of iterations rapidly, using of multiple GPUs and Tensor Processing Units (TPUs) will be effective. Thus, with this implementation, model results can be performed more precisely for different parameters.

REFERENCES

[1] A. Radford, L. Metz and S. Chintala, *Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks,* 2015.

[2] J. Wu, C. Zhang, T. Xue, W. T. Freeman and J. B. Tenenbaum, *Learning a probabilistic latent space of object shapes via 3d generative-adversarial modeling,* 2016, p. 82–90.

[3] J. Wang, J. Zhang and Q. Xu, "Research on 3D laser scanning technology based on point cloud data acquisition," in *2014 International Conference on Audio, Language and Image Processing*, 2014.

[4] P. Jasiobedzki, M. R. M. Jenkin, E. E. Milios, B. Down, J. K. Tsotsos and T. A. Campbell, "Laser eye: a new 3D sensor for active vision," in *Sensor Fusion VI*, 1993.

[5] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville and Y. Bengio, *Generative Adversarial Networks,* 2014.

[6] I. Goodfellow, Y. Bengio and A. Courville, Deep Learning, MIT Press, 2016.

[7] M. Mirza and S. Osindero, *Conditional Generative Adversarial Nets,* 2014.

[8] T. Karras, T. Aila, S. Laine and J. Lehtinen, *Progressive Growing of GANs for Improved Quality, Stability, and Variation,* 2017.

[9] P. Bojanowski, A. Joulin, D. Lopez-Paz and A. Szlam, *Optimizing the Latent Space of Generative Networks,* 2017.

[10] T. Karras, S. Laine and T. Aila, *A Style-Based Generator Architecture for Generative Adversarial Networks,* 2018.

[11] H. Noh, S. Hong and B. Han, "Learning Deconvolution Network for Semantic Segmentation," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2015.

[12] S. Ioffe and C. Szegedy, "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift," *CoRR,* vol. abs/1502.03167, 2015.

[13] W. E. Lorensen and H. E. Cline, "Marching Cubes: A High Resolution 3D Surface Construction Algorithm," *SIGGRAPH Comput. Graph.,* vol. 21, p. 163–169, 8 1987.

[14] A. Ahmed, "Random communication systems based on alpha-stable processes," 2018.

[15] K. Shea, R. Aish and M. Gourtovaia, "Towards integrated performance-driven generative design tools," *Automation in Construction,* vol. 14, p. 253–264, 2005.

[16] J. Wu, C. Zhang, T. Xue, W. T. Freeman and J. B. Tenenbaum, *Learning a probabilistic latent space of object shapes via 3d generative-adversarial modeling,* 2016, p. 82–90.

[17] A. Patel and B. Kosko, "Stochastic resonance in continuous and spiking neuron models with Levy noise," *IEEE Transactions on Neural Networks,* vol. 19, p. 1993–2008, 2008.

[18] L. Zhou, Z. Wang, J. Zhou and W. Zhou, "Mean square synchronization of neural networks with Lévy noise via sampled-data and actuator saturating controller," *Neurocomputing,* vol. 173, p. 1235–1244, 2016.

[19] R. Müller, S. Kornblith and G. Hinton, *When Does Label Smoothing Help?,* 2019.

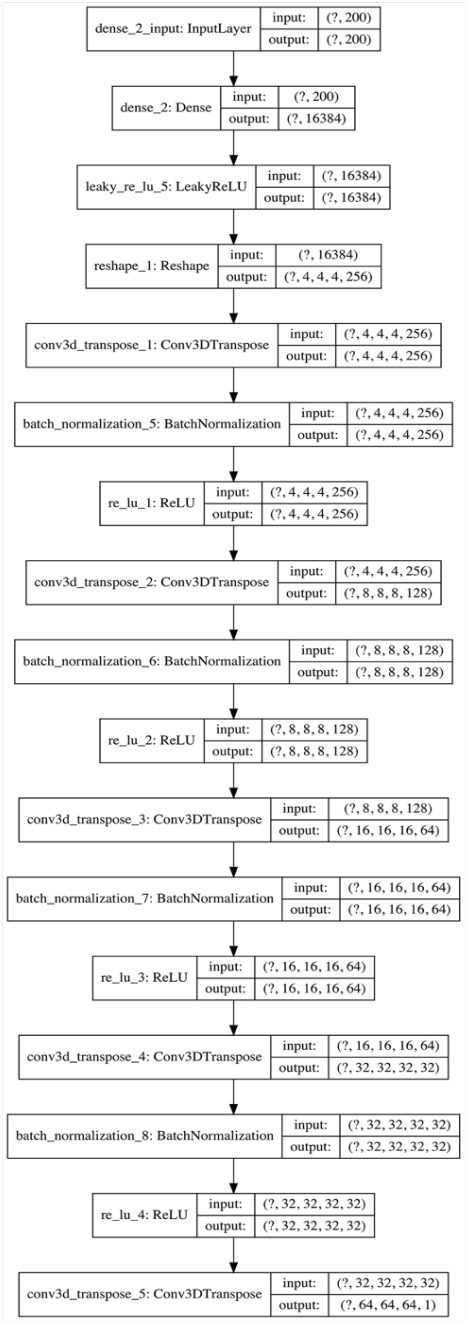
[20] M. E. Özbek, M. E. Çek and F. A. Savacı, *Skewed alpha-stable distributions for modeling and classification of musical instruments,* vol. 20, TÜBİTAK, 2012, pp. 934-947.

[21] A. X. Chang, T. Funkhouser, L. Guibas, P. Hanrahan, Q. Huang, Z. Li, S. Savarese, M. Savva, S. Song, H. Su, J. Xiao, L. Yi and F. Yu, *ShapeNet: An Information-Rich 3D Model Repository,* 2015.

[22] T. Y. Wang, H. Su, Q. Huang, J. Huang, L. J. Guibas and N. J. Mitra, "Unsupervised texture transfer from images to model collections.," *ACM Trans. Graph.,* vol. 35, p. 177–1, 2016.

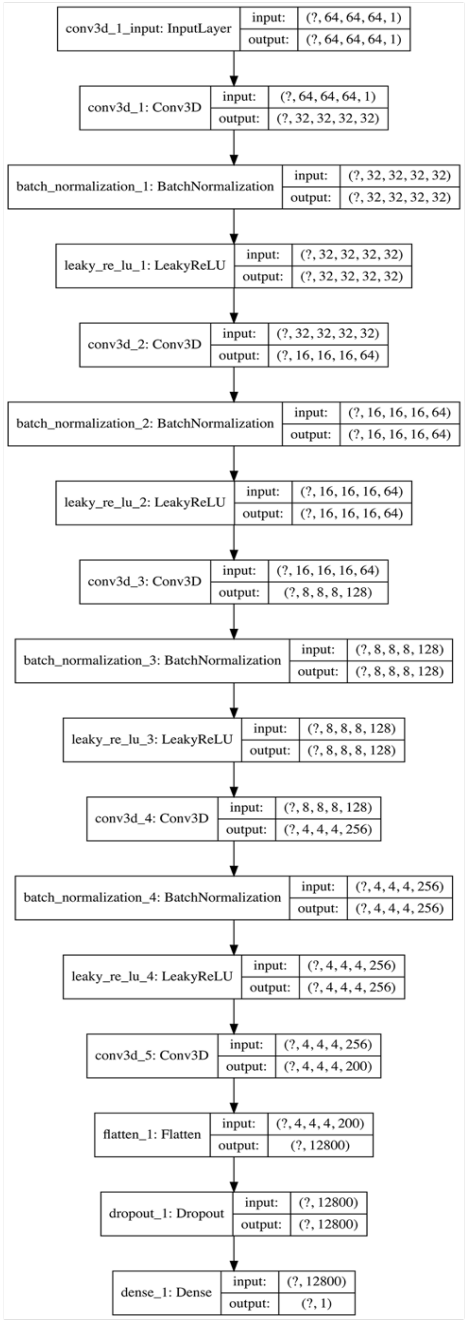
APPENDIX 1

Generator Model



a)

Discriminator Model



b)