INTERNATIONAL JOURNAL OF SCIENTIFIC RESEARCH IN ENGINEERING AND MANAGEMENT (IJSREM)
VOLUME: 07 ISSUE: 04 | APRIL - 2023 IMPACT FACTOR: 8.176 ISSN: 2582-3930

POINT CLOUD IMAGE GENERATION USING GENERATIVE ADVERSARIAL NETWORKS (GAN) AND REINFORCEMENT LEARNING

B Yashaswi[1], B Jashwanth[2], S Vamsi Krishna[3] V Rishikesh[4] , R Krishnamoorthy 5

[1]Computer Science and Engineering, Bharath Institute of Higher Education and Research, Chennai-73
[2] Computer Science and Engineering, Bharath Institute of Higher Education and Research, Chennai-73
[3] Computer Science and Engineering, Bharath Institute of Higher Education and Research, Chennai-73
[4] Computer Science and Engineering, Bharath Institute of Higher Education and Research, Chennai-73
5 Assistant Professor, Computer Science and Engineering, Bharath Institute of Higher Education and Research, Chennai-73

---***---

Abstract - Reinforcement learning (RL) agent controls a generative adversarial network in RL-2GAN-Net in a quick and reliable manner. The framework is used for point cloud form generation, which uses the GAN to control 3D meshes into high-fidelity finished shapes. Although a GAN is unstable and difficult to train, the issue is resolved by training the GAN on a latent space representation whose dimension is reduced in comparison to the raw input of a point cloud and using an RL agent to determine the proper input to the GAN to produce the latent space representation of the shape that best fits the current input of a 3D mesh. The point cloud is reliably produced by the recommended pipeline. An RL agent is being trained in this effort to manage the GAN.

Keywords— GAN, Reinforcement Learning, Point Cloud

1.INTRODUCTION

A collection of data elements in three dimensions is referred to as a point cloud. A 3-D shape or item is represented by the points combined. An x, y, and z geometric coordinate serves as a representation for each point in the data collection. Point clouds offer a way to compile a lot of discrete spatial measurements into a dataset that can be visualized as an entity. Robot navigation and perception, depth estimation, stereo vision, visual registration, and sophisticated driver aid systems all utilize point cloud processing. (ADAS).

When 3D data is acquired, it is done so in the form of a point cloud, which is a collection of Cartesian values. Point clouds can be created using laser scanners, stereo reconstruction, or RGB-D cameras. Due to restricted viewing angles, occlusions, sensor resolution, or unstable measurement in the texture-less region (stereo reconstruction) or specular materials, the raw output typically has a big missing region. In order to make use of the data, additional post-processing is required, including registration, denoising, resampling, semantic understanding, and ultimately reconstructing the 3D mesh model.

Using the Point Cloud GAN, point clouds are produced from 3D models. The training sample is the ModelNet dataset. The initial point cloud image serves as the basis for training the RL-2GAN net so it can generate the points necessary to create a complete image on its own. The form is finished by the RL-2GAN network in a matter of seconds. Despite the input being distorted, the method produces the shape with higher accuracy than the prior method, which used an auto-encoder.

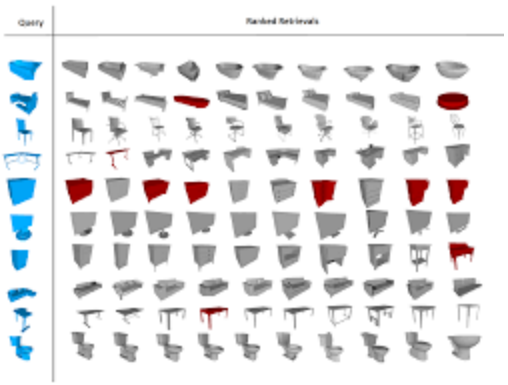


Figure 1: The ModelNet dataset of the 3D objects

The primary form of acquired measurements is the 3D point cloud which is unstructured and unordered. Therefore, it is not possible to directly apply conventional convolutional neural networks (CNN) approaches which work nicely for structured data e.g., for 2D grids of pixels. The extensions of CNN in 3D have been shown to work well with 3D voxel grid. However, the computing cost grows drastically with voxel resolution due to the cubic nature of 3D space. Recently PointNet has made it possible to directly process point cloud data despite its unstructured and permutation invariant nature. This has opened

new avenues for employing point cloud data, instead of voxels, to contemporary computer-vision applications, e.g., segmentation, classification and shape completion.

A reinforcement learning agent-controlled GAN (generative adversarial network) (fig.4) based network which can predict complete point cloud from incomplete data. As a pre-processing step, we train a normal GAN to get the latent space representation of the point cloud and we further use this representation to train the discriminator for GAN-2. Our agent is then trained to take an 'action' by selecting an appropriate z vector for the generator of the pre-trained GAN to synthesize the latent space representation of the complete point cloud. Unlike the previous approaches which use back-propagation to find the correct z vector of the GAN, our approach based on an RL agent is real time and also robust to complex data points. Therefore, we use the help of a pre-trained discriminator of GAN to decide the winner between the generated output of the GAN and the output of the RL-GAN. The final choice of completed shape preserves the global structure of the shape and is consistent with the partial observation.

2. LITERATURE REVIEW

Maciej Zamorski, Maciej Zi˛eba, Piotr Klukowski, Rafał Nowak, Karol Kurach, Wojciech Stokowiec, and **Tomasz Trzcinski** introduced the novel singlestage end-to-end 3D point cloud probabilistic model called 3D Adversarial Autoencoder (3dAAE). This generative model is capable of representing 3D point clouds with compact continuous or binary embeddings. Additionally, those representations follow an arbitrary prior distribution, such as Gaussian Mixture Model (continuous embeddings), Bernoulli or Beta (binary representations). The PointNet architecture allows handling unordered sets of real-valued points, by introducing permutation-invariant feature aggregating function. This approach has constituted a basis for numerous extensions, involving training hierarchical features, point cloud retrieval or point cloud generation.

Stefan Milz, **Martin Simon**, **Kai Fischer, Maximillian Popperl,** and **Horst-Michael Gross** presented the first approach for 3D point-cloud to image translation based on conditional Generative Adversarial Networks (cGAN). The model handles multi-modal information sources from different domains, i.e. raw point-sets and images. The generator is capable of processing three conditions, whereas the point-cloud is encoded as raw point-set and camera projection. An image background patch is used as constraint to bias environmental texturing. A global approximation function within the generator is directly applied on the point-cloud (PointNet). Hence, the representative learning model incorporates global 3D characteristics directly at the latent feature space. Conditions are used to bias the background and the viewpoint of the generated image. This opens up new ways in augmenting or texturing 3D data to aim the generation of fully individual images. We successfully evaluated our method on the KITTI and SunRGBD dataset with an outstanding object detection inception score.

Chen-Hsuan Lin, Chen Kong and **Simon Lucey** presented the Conventional methods of 3D object generative modeling learn volumetric predictions using deep networks with 3D convolutional operations, which are direct analogies to classical 2D ones. However, these methods are computationally wasteful in attempt to predict 3D shapes, where information is rich only on the surfaces. In this paper, we propose a novel 3D generative modeling framework to efficiently generate object shapes in the form of dense point clouds. We use 2D convolutional operations to predict the 3D structure from multiple viewpoints and jointly apply geometric reasoning with 2D projection optimization. They introduced the pseudo-renderer, a differentiable module to approximate the true rendering operation, to synthesize novel depth maps for optimization. Experimental results for single-image 3D object reconstruction tasks show that we outperforms state-of-the-art methods in terms of shape similarity and prediction density.

Dong Wook Shu, **Sung Woo Park,** and **Junseok Kwon** had proposed a novel generative adversarial network (GAN) for 3D point clouds generation, which is called tree-GAN. To achieve state-of-the-art performance for multi-class 3D point cloud generation, a tree-structured graph convolution network (TreeGCN) is introduced as a generator for tree-GAN. Because TreeGCN performs graph convolutions within a tree, it can use ancestor information to boost the representation power for features. To evaluate GANs for 3D point clouds accurately, we develop a novel evaluation metric called Frechet point cloud dis-tance (FPD). Experimental results demonstrate that the proposed tree-GAN outperforms state-of-the-art GANs in ∗Authors contributed equally terms of both conventional metrics and FPD, and can generate point clouds for different semantic parts without prior knowledge.

Xiaoshui Huang, **Guofeng Mei**, **Jian Zhang** and **Rana Abbas** presented the registration is a transformation estimation problem between two point clouds, which has a unique and critical role in numerous computer vision applications. The developments of optimization-based methods and deep learning methods have improved registration robustness and efficiency. Recently, the combinations of optimization-based and deep learning methods have further improved performance. However, the connections between optimization-based and deep learning methods are still unclear. Moreover, with the recent development of 3D sensors and 3D reconstruction techniques, a new research direction emerges to align cross-source point clouds. This survey conducts a comprehensive survey, including both same-source and cross-source registration methods, and summarize the connections between

optimization-based and deep learning methods, to provide further research insight. This survey also builds a new benchmark to evaluate the state-of-the-art registration algorithms in solving cross-source challenges. Besides, this survey summarizes the benchmark data sets and discusses point cloud registration applications across various domains. Finally, this survey proposes potential research directions in this rapidly growing field.

3.SYSTEM ANALYSIS

➢ Existing System

Autoencoders:

Autoencoding has gained popularity in a variety of fields and has only lately become relevant to the 3D industry. But unlike other data modalities, many 3D depictions (like point clouds) are discrete samples of the underlying continuous 3D surface. The underlying 3D shapes will unavoidably experience sampling variations as a result of this process. A preferable objective in learning 3D representation is to ignore such sampling variations and concentrate on acquiring transferrable knowledge of the underlying 3D shape. For current representation learning paradigms, this goal represents a significant task. Because the decoder must recreate the original point cloud, the standard autoencoding paradigm, for instance, compels the encoder to capture such sampling variations.

Numerous disciplines have found autoencoding to be of interest, and there different point cloud samplings of the same model can share a continuous depiction that the implicit decoder can produce. Reconstructing using the implicit representation can give priority to having the encoder ignore sampling differences and introduce the proper inductive bias to learn more broadly applicable feature representations.

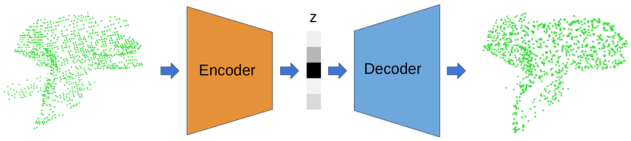


Figure 2: Point cloud image generation using autoencoder and decoder.

The RL-GAN-Net process, as depicted in fig. 3. It is a GAN (generative adversarial network)-based network controlled by reinforcement learning agents that can forecast a full point cloud from incomplete data. We train an autoencoder (AE) as a pre-processing stage to obtain the point cloud's latent space representation, and we then use this representation to train a GAN. The generator of the pre-trained GAN is then trained to perform a "action" by choosing a suitable z vector to synthesize the latent space representation of the entire point cloud. Our approach based on an RL agent is real time and robust to large missing regions, in contrast to earlier methods that use back-propagation to determine the right z vector of the GAN.

However, a straightforward AE can successfully restore the original shape for data with minor missing areas. Therefore, to choose between the decrypted output of the GAN and the output of the AE, we employ a pre-trained discriminator of GAN. The completed shape of choice maintains the shape's overall structure and is consistent with the partial observation.

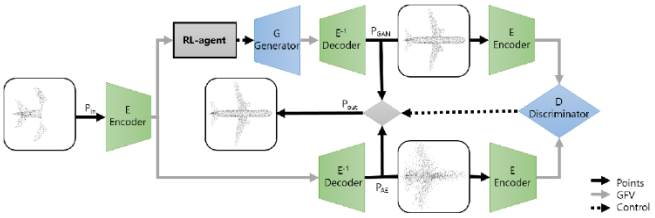


Fig3: RL-GAN

➢ Proposed System

The Rl-2GAN network is a two-fold GAN (generative adversarial network) based network controlled by a reinforcement learning agent that can forecast an entire point cloud from 3D data. We first train a PC-GAN to obtain the point cloud's latent space representation, and then we use this representation to train a GAN. The generator of the pre-trained GAN is then trained to perform a "action" by choosing a suitable z vector to synthesize the latent space representation of the entire point cloud. Our method based on an RL agent is real time and robust for producing complex data, in contrast to prior approaches that use back-propagation to determine the right z vector of the GAN. To choose the winner between the decoded output of the GAN, a pre-trained discriminator of GAN is used. The completed shape of choice maintains the shape's overall structure and is consistent with the partial observation.

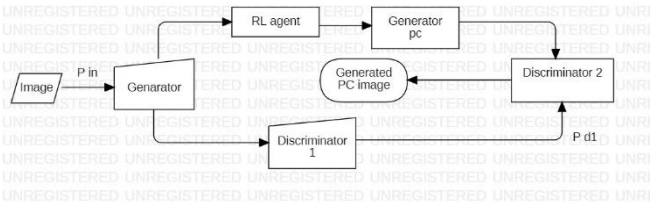


Figure 4: Architecture of the Rl-2GAN

4. WORKING THEORY

A simple GAN network to produce noisy point clouds, a latent-space generative adversarial network (l-GAN), and a reinforcement learning (RL) agent make up the generation pipeline. A distinct deep neural network must be trained for each of the components. Using the produced data, we train l-GAN after first training a GAN. A pre-trained two-fold GAN and GAN-2 are used in conjunction with the RL agent during training.

The generator of the trained GAN generates the noisy and incomplete point cloud to a noisy Global Feature Vector (GFV). Given this noisy GFV, our trained RL agent selects the correct seed for the GAN's generator. The generator produces the clean GFV which is finally passed through the discriminator of l-GAN to get the completed point cloud representation of the clean GFV. A discriminator observes the GFV of the generated shape and the one processed by GAN and selects the more plausible shape. In the following subsections, we explain the three fundamental building blocks of our approach, then describe the combined architecture.

➢ GENERATIVE ADVERSARIAL NETWORK:

GAN generates realistic data by jointly training a pair of generator and discriminator. GAN demonstrated its success in image generation tasks, in practice, training a GAN tends to be unstable and suffer from mode collapse.

➢ *GAN-2:*

The secondary GAN, which uses the data generated by the primary GAN, it is trained with the combination of generator data of GAN as noise and is controlled by the RL-agent to generate better and high fidelity point clouds.

➢ REINFORCEMENT LEARNING:

In a RL-based framework, an agent acts in an environment. Given an observation xt at each time step t, the agent performs an action at and receives a reward rt. The agent network learns a policy π which maps states to the action with some probability. The environment can be modeled as a Markov decision process, i.e., the current state and action only depend on the previous state and action. The reward at any given state is the discounted future reward.

$$Rt = PT\, i = t\, \gamma\, (i - t)\, r(si, ai).$$

The final objective is to find a policy which provides the maximum reward.

For our problem, the environment is the combination of RL and GAN, and resulting losses that are calculated as intermediate results of various networks in addition to the discrepancy between the input and the predicted shape. The observed state 'st' is the initial noisy GFV generated from the input point cloud. We assume that the environment is Markov and fully observed; i.e., the recent most observation xt is enough to define the state st. The agent takes an action at to pick the correct seed for the z-space input of the generator. The synthesized GFV is then passed through the decoder to obtain the completed point cloud shape.

One of the major tasks in training an RL agent is the correct formulation of the reward function. Depending on the quality of the action, the environment gives a reward r back to the agent. In RL-2GAN network, the right decision equates to the correct seed selection for the generator. We use the combination of negated loss functions as a reward for shape completion task that represent losses in all of Cartesian coordinate
(r CH = −LCH), latent space (r GFV = −L GFV), and in the view of the discriminator (r D = −LD). The final reward term is given as follows:

$$r = w\,CH \cdot r\,CH + w\,GFV \cdot r\,GFV + w\,D \cdot r\,D$$

here w CH, w GFV, and w D are the corresponding weights assigned to each loss function. We explain the selection of weights in the supplementary material.

5.SYSTEM REQUIREMENTS

➢ Software Requirements

- Python 3 - We have used Python which is a statistical mathematical programming language like R instead of MATLAB.
- Pytorch is used for building the Reinforcement learning agent.
- Python packages like NumPy, Matplotlib, Pandas for mathematical computation and plotting graphs.
- GitHub and Stack overflow was used for reference in case of programming syntax errors.
- GitHub and Stack overflow was used for reference in case of programming syntax errors.
- Google Collaboratory (open-source Jupyter Notebook interface with high GPU facility) - Google Colab/Colaboratory is a free Jupyter notebook environment that requires no setup and runs entirely on cloud. With Colab, one can write and execute code, save and share analyses, access powerful computing resources, all for free from browser. [Jupyter Notebook is a powerful way to iterate and write on your Python code for data analysis. Rather than writing and rewriting an entire code, one can write lines of code and run them at a time. It is built off of iPython which is an interactive way of running Python code. It allows Jupyter notebook to support multiple languages as well as storing the code and writing own markdown.

➢ *Requirement Specification*

1) **Software Requirements:**

Languages : Python,
Software : Google Colab, Blender

Operating System: Windows 10 and above, Linux

2) **Hardware Requirements:**

System : Intel Core i7
Hard Disk : 256gb and above
RAM : 16GB
Processor : Intel Core i7 @ 2.8GHz

➢ ***Dataset***:

Point Cloud image data:

The point cloud data is acquired from ModelNet dataset.

The dataset consists 50,000 point clouds approximately, each object is categorized into respective complete and in complete point clouds.

6.SYSTEM DESIGN

➢ **System Architecture:**

The system architecture for the project involves the steps, through which we train the model for our data. It represents the layer wise procedure to achieve the required algorithm.

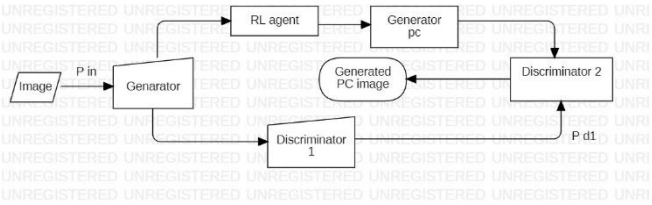


Figure 4: Architecture of the Hybrid RL-2GAN network

RL-agent utilizes Generator and discriminator. The output is decoded and completed as shown at the bottom, and does not affect the training. By employing an RL agent, our pipeline is capable of real-time shape generation.

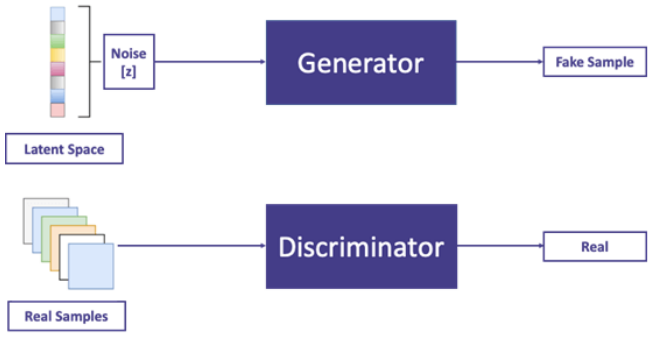


Figure 5: Architecture diagram of Generative Adversarial Network

A generative adversarial network (GAN) has two parts:

- The **generator** learns to generate plausible data. The generated instances become negative training examples for the discriminator.
- The **discriminator** learns to distinguish the generator's fake data from real data. The discriminator penalizes the generator for producing implausible results.

The last layer of the proposed network is the RL agent, it observes the state and environment of the output and performs the necessary action on the model and rewards the model.
The flow diagram of the RL agent is shown as below:

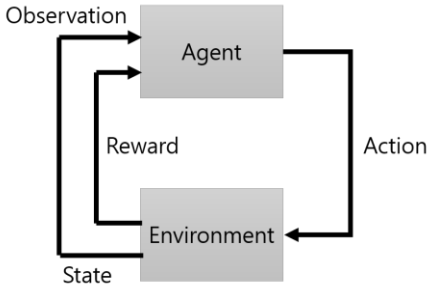


Figure 9: Flow of the RL agent

7. IMPLEMENTATION

GAN:

GAN is composed of a generator and a discriminator. For the generator and the discriminator pair, we adapted the main architecture of the GAN and applied in the latent space acquired by the GAN-1. We trained the GAN-2 using WGAN-GP adversarial loss with $\lambda gp = 10$. The total number of iterations was one million. As a typical GAN training, we updated discriminator 3 times for every update of the generator. We used Adam optimizer with $\beta 1=0.5$ and $\beta 2=0.9$. The learning rate for both generator and discriminator was set to 0.0001. Batch size was set to 32 and number of workers were set to 2.

We selected the dimension of z-vector to be 1. We did this to limit the dimensions of action space for the agent. All the experiments conducted were with a single dimension. We also tested with 6 and 32 dimensions but there was no change in the performance of the GAN or the agent in either case. Therefore we kept the dimension of the z-vector to 1.
We trained the l-GAN using the dataset generated by passing the ShapeNet point cloud dataset through the generator of GAN.

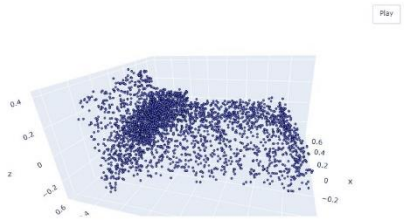


Fig : Output of the noisy GAN

RL Agent

The third element of the basic architecture is RL. The basic RL framework is composed of an agent and the environment. Among many possible variations of the RL agent, we used the actor-critic architecture to enable continuous control of the l-GAN.

Actor and Critic Architecture The actor and critic networks are chosen to be fully connected (FC) layers. The actor has four FC layers with 400, 400, 300, 300 neurons with ReLu activation for the first three layers and tanh for the last layer respectively. The input to the actor is a 128-dimensional GFV. The output is a single dimension z-vector. The critic also has four FC layers with 400, 432, 300, 300 neurons with ReLu activation for the first two layers respectively.

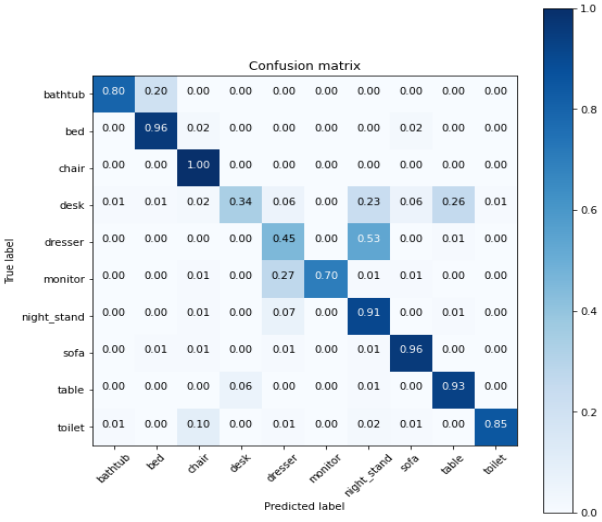
Reward Function Hyper-parameter of the main article, the multiplicative weights of wCH, wGF V , and wD are assigned to the corresponding loss functions. The weight values are chosen such that, when combined, the effects of individual terms are not out of proportion or dominant in any way. In other words, the total loss is within range for the RL agent to learn useful information for all of the terms of LCH, LGF V , and LD. For example, if the value for the Chamfer loss was approximately 1000 and the GFV loss was 10, then they are normalized by dividing by 100 and 1 respectively. After consulting the range of raw loss values of multiple trials, we set wCH = 100, wGF V = 10.0, and wD = 0.01 for all our experiments. The RL agent was adopted from the open source implementation of the DDPG algorithm.

Training Details

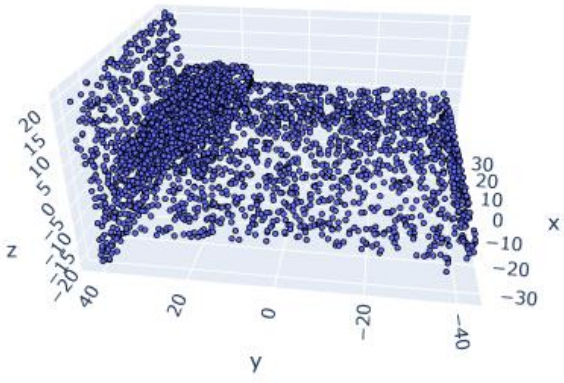
The training of the agent can be divided into two parts. The first part is the collection of experience. The second part is the training of the actor and critic network in accordance with the DDPG algorithm as outlined in the previous work. It shows the mechanism by which the replay buffer R is filled continuously with useful experiences. We fill the memory with one input at a time. This implies that the batch size for this case is one. Our task is episodic, which means that after each episode we collect a reward. The number of episodes is equal to the maximum number of allowed iterations. In each episode, the agent is allowed to take a single action after which the episode terminates.

The sequences of state, action and reward tuples are then stored in the replay buffer. The second part, i.e., training the actor and critic in accordance with DDPG, is performed by keeping the batch size equal to one hundred. This means that a batch of a 64 memories from the replay buffer is picked randomly to train the actor and critic networks according to the DDPG algorithm. The evaluation of the policy was carried out after 5000 iterations. The action dimension is determined by the dimension of the GAN's z-space, which is 1. The action space is kept to unity to achieve better performance by the agent. We also tested with 32 dimensions for z space but it did not have any noticeable effect on the performance of GAN or the agent. Parameters for the RL agent are given below:

Training metrics of the GAN:



Final high-fidelity Dense point cloud:



8 . REFERENCES

[1] Panos Achlioptas, Olga Diamanti, Ioannis Mitliagkas, and Leonidas J. Guibas. Representation learning and adversarial generation of 3d point clouds. CoRR, abs/1707.02392, 2017.

[2] Martin Arjovsky, Soumith Chintala, and Leon Bottou. ´ Wasserstein generative adversarial networks. In Doina Precup and Yee Whye Teh, editors, Proceedings of the 34th International Conference on Machine Learning, volume 70 of Proceedings of Machine Learning Research, pages 214–223, International Convention Centre, Sydney, Australia, 06–11 Aug 2017. PMLR.

[3] Alfredo Canziani, Adam Paszke, and Eugenio Culurciello. An analysis of deep neural network models for practical applications. CoRR, abs/1605.07678, 2016

[4] Miriam Bellver, Xavier Giro-i Nieto, Ferran Marques, and Jordi Torres. Hierarchical object detection with deep reinforcement learning. In Deep Reinforcement Learning Workshop, NIPS, December 2016.

[5] Juan C. Caicedo and Svetlana Lazebnik. Active object localization with deep reinforcement learning. In Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV), ICCV '15, pages 2488–2496, Washington, DC, USA, 2015. IEEE Computer Society.

[6] Alfredo Canziani, Adam Paszke, and Eugenio Culurciello. An analysis of deep neural network models for practical applications. CoRR, abs/1605.07678, 2016.

[7] Angel X. Chang, Thomas A. Funkhouser, Leonidas J. Guibas, Pat Hanrahan, Qi-Xing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, Jianxiong Xiao, Li Yi, and Fisher Yu. Shapenet: An information-rich 3d model repository. CoRR, abs/1512.03012, 2015.

[8] Angela Dai, Charles Ruizhongtai Qi, and Matthias Nießner. Shape completion using 3d-encoder-predictor cnns and shape synthesis. CoRR, abs/1612.00101, 2016.

[9] Angela Dai, Daniel Ritchie, Martin Bokeloh, Scott Reed, Jurgen Sturm, and Matthias Nießner. Scancomplete: Large- ¨ scale scene completion and semantic segmentation for 3d scans. CoRR, abs/1712.10215, 2017.

[10] Haoqiang Fan, Hao Su, and Leonidas J. Guibas. A point set generation network for 3d object reconstruction from a single image. CoRR, abs/1612.00603, 2016.

[11] Scott Fujimoto. Addressing function approximation error in actor-critic methods. https://github.com/sfujim/ TD3, 2018