Proceedings of the 3rd International Conference on Signal Processing and Machine Learning
DOI: 10.54254/2755-2721/6/20230957

Performance of NeRF and its optimized models on generating images from 2D to 3D of animation or game character

Yuchen Xie

Department of Mathematics, University of California, Los Angeles, Los Angeles, CA 90024, USA

Yuchen724@ucla.edu

Abstract. Neural Radiance Fields (NeRF) has recently emerged as a promising approach for single-image 3D scene reconstruction. Despite this, most studies on NeRF have focused on real-world data or items with established camera projection connections, rather than snapshots of video game characters that may not have a distinct camera projection connection. Although some research has been conducted on the use of animatable mannequins, the majority of these studies have used photographs of real people as their data source. To date, there has been limited research into the use of NeRF for animation or game characters. In this paper, we compare and evaluate a range of NeRF models on a dataset of animated game characters. The models considered include original NeRF, Block-NeRF, Mip-NeRF, and Pixel-NeRF, each of which has its own design choices and trade-offs. Our results show that Mip-NeRF models are effective at generating high-quality images of game characters on our dataset, while Pixel-NeRF performs poorly on our dataset. Despite these findings, our research demonstrates the feasibility of using NeRF-based models to create realistic character models or animations for video games, with potential applications in game development and virtual reality.

Keywords: Neural Radiance Fields, Volume Rendering, 3D Scene Representation, Animation and Game, Image-based Machine Learning.

1. Introduction

NeRF is a machine learning model that uses a neural network to represent a complex static scene. The neural network encodes all information about the lighting, geometry, and material properties of objects in the scene [1, 2]. After training, the model can generate highly realistic images of the scene from any angle, accurately representing the 3D scene. Unlike traditional machine learning models, NeRF can learn directly from images, representing complex 3D scenes with multiple objects and lighting conditions through a single, compact, fully-connected neural network without the need for convolutional layers [1]. This ability makes it more efficient than other approaches that require multiple models to handle different aspects of a 3D scene. Despite its simple structure, NeRF is able to generate new views from various 3D scenes obtained from 2D image data, including the possibility to define 3D shapes by optimizing continuous volume scenes [1]. This approach overcomes the limitations of traditional 3D modeling and data representation techniques [3], which can introduce errors and negatively impact the accuracy of 3D reconstruction results [4].

© 2023 The Authors. This is an open access article distributed under the terms of the Creative Commons Attribution License 4.0 (https://creativecommons.org/licenses/by/4.0/).

DOI: 10.54254/2755-2721/6/20230957

We investigate the behavior of screenshots of comic/game characters in the context of NeRF, a model that reconstructs surface points of objects and generates 2D images of arbitrary views based on camera projection relations. While the original NeRF dataset includes clearly defined camera projection relationships [1], this is not necessarily the case for screenshots of game characters. Previous research on animatable mannequins has used real-life photos as data [5], but our focus is on exploring how screenshots of comic/game characters perform in the NeRF framework. To do this, we will compare several variations of NeRF, including Mip-NeRF, Block-NeRF, and Pixel-NeRF, which are optimized for learning 3D scenes from series of 2D images and rendering out 2D images. By studying these models theoretically and experimentally on comic/game character datasets, we aim to better understand their capabilities and limitations.

2. Methods

2.1. Models

In this study, we conducted a comparative analysis of several variations of the NeRF model, a state-of-the-art approach for synthesizing high-quality 3D images from 2D views. Specifically, we evaluated the performance of four different NeRF models: the original NeRF model, as well as three modified versions: Mip-NeRF, Block-NeRF, and Pixel-NeRF. Each of these models was designed to address specific challenges or limitations of the original NeRF model, and our aim was to understand the trade-offs and benefits of each variation.

2.1.1. NeRF. The NeRF model uses a multi-layer perceptron (MLP) neural network to implicitly learn a static 3D scene, which is expressed and stored in 3D by the weights of the MLP [1]. It is defined as a continuous function that describes the light field in the scene [6]. Specifically, it maps a set of 3D scene coordinates $\boldsymbol{x} = (x, y, z)$ and a 2D view direction $\boldsymbol{d} = (\theta, \phi)$ to a set of RGB colors and visibility $(\boldsymbol{c}, \sigma)$. This function is called the neural radiance field and is represented by the following equation:

$$F_\theta : (\boldsymbol{x}, \boldsymbol{d}) \rightarrow (\boldsymbol{c}, \sigma) \tag{1}$$

The NeRF model is a machine learning method for volume rendering, which is a technique for visualizing 3D datasets by constructing 2D images from data [7]. It predicts the RGB color and visibility of each pixel in an image through volume rendering by emitting light pixel by pixel according to the camera position, sampling the light in the direction of the light to obtain sample points, performing positional encoding of the sample points, and feeding the encoded sample points into a MLP [8]. The corresponding (r, g, b) and visibility are then calculated. Specifically, the prediction of visibility (σ) is not dependent on the viewing direction and only considers the 3D coordinates. In contrast, the prediction of RGB color is influenced by the viewing direction.

The NeRF model employs a technique known as discrete approximate volume rendering, which relies on segmented random sampling [9]. In detail, the model divides the region where the rays need to be integrated into N segments, and then performs uniform random sampling in each small region [1]. This approach allows for positional continuity while just sampling discrete locations. The resulting color and visibility values are then combined to generate the final image.

The model utilizes a variant of stochastic gradient descent as a loss function to calculate the discrepancy between the synthetic image and the ground truth image, and to optimize the model's parameters. The loss function is defined as follows:

$$L = \frac{1}{N} \sum \left\| I_{gt} - I_{pred} \right\|^2 \tag{2}$$

where N is the number of pixels in the image, I_{gt} is the ground truth image, and I_{pred} is the synthesized image [1]. The NeRF model is trained to minimize the loss function, resulting in the ability to synthesize highly realistic 3D images that capture fine details and complex illumination effects.

DOI: 10.54254/2755-2721/6/20230957

2.1.2. Mip-NeRF. A Multi-scale Representation for Anti-Aliasing Neural Radiance Fields (Mip-NeRF) is an extension of the original NeRF method. Similar to NeRF, Mip-NeRF learns the basic 3D structure of a scene from a single 2D image and generates a 3D rendering of the scene [10]. NeRF rendering involves determining sampling points on rays emanating from a viewpoint d and a starting point o, and integrating over the rays. However, there is only one ray per pixel, and when the grid of this pixel is large or consists of different colors, a single ray may not accurately represent the pixels of the grid [1]. This can result in over-blurred or out-of-sync renderings when training or testing images with scenes of different resolutions.

To reduce blurring artifacts, Mip-NeRF replaces the pixel points in the initial NeRF with the circular part of a cone [11]. Mip-NeRF divides the scene into a hierarchy of blocks, each representing a different level of detail. For each level of detail, Mip-NeRF trains a separate NeRF model using the input image and the corresponding 3D points [10]. This allows for efficient and scalable training, enabling NeRF to capture the fine details of the scene. Training is performed using a differential rendering technique that computes the gradient of the reconstruction error relative to the NeRF parameters [12]. The resulting NeRF model can be used to reconstruct scenes with improved quality and reduced blending artifacts. The training of the original NeRF model combines supervision from ground truth 3D reconstructions with self-supervision from different scales of the input image [10], enabling Mip-NeRF to capture fine-scale details in 3D reconstructions more accurately and to process scenes with a wide range of depth scales.

Being a model that incorporates scales into the network structure, Mip-NeRF can be mathematically represented as follows: Given N neural radiance fields $F_1, F_2, \dots, F_n$, each representing a 3D scene at a different scale, Mip-NeRF combines these fields into a single multi-scale representation F using a weighting function W. Specifically, the i th neural radiance field F_i is a continuous function $F_i(x, y, z) = \big(I_i(x, y, z), D_i(x, y, z)\big)$ that maps 3D positions (x, y, z) to RGB colors and depths, and the multi-scale representation F is given by the sum of the fields weighted by W : $F(x, y, z) = \sum F_i(x, y, z) \times W(x, y, z, i)$. The weighting function $W(x, y, z, i)$ is defined as $W(x, y, z, i) = max\left(0, 1 - \frac{d(x,y,z)}{d_i}\right)$, where $d(x, y, z)$ is the distance between the input position (x, y, z) and the nearest sample in the ith field, and d_i is the sampling density of the ith field [10]. This definition of W ensures that positions closer to a sample in the ith field receive a higher weight, while positions farther away receive a lower weight, allowing Mip-NeRF to smoothly interpolate between the different fields and produce a smooth and accurate representation of the 3D scene.

2.1.3. Block-NeRF. Block NeRF is also based on the original NeRF and borrows its structure from Mip-NeRF, introducing a new block structure to the neural network that allows it to handle large and complex 3D scenes better. Specifically, the neural network is divided into blocks, each responsible for modeling a different part of the scene and training a separate NeRF model. For a given viewpoint, only the blocks associated with it are rendered by considering distance and visibility. This process is done by setting a distance radius and discarding blocks that are too far from the viewpoint location or have visibility below a set threshold [13, 14, 15, 16, 17, 18]. By using the selected blocks from the previous stage of filtering and interpolating them according to the inverse distance weighting, the whole neural network is able to generate an image with a specific viewpoint [19]. The combination of the rendering results of the relevant blocks based on their distance from the viewpoint enables the network to scale more efficiently and better capture fine details in the scene, as each block can be updated individually.

As an extension of Mip-NeRF, the Block NeRF model uses a combination of two neural networks. The first network, a MLP f_σ, predicts the density σ at location x in 3D space. The network also outputs feature vectors associated with 2D viewing directions d, exposure levels, and appearance embeddings. These are input to a second MLP f_c, which outputs the color of the points. The model also includes a visibility network f_v, which is trained to predict whether a point in space is visible in the training view and is used to reject non-conforming Block-NeRF during inference [13]. As the network progresses

DOI: 10.54254/2755-2721/6/20230957

from the first to the last block, the feature map is able to capture increasingly complex details in the scene.

2.1.4. Pixel-NeRF. Pixel-NeRF is a variant of the original NeRF algorithm that composes high-quality 3D renderings of scenes from one or several input 2D images [20]. It produces high-quality reconstructions even with limited input data [21, 22]. While the original NeRF algorithm synthesizes the entire scene simultaneously, Pixel-NeRF synthesizes each pixel of the output image independently [20]. This allows Pixel-NeRF to handle scenes with significant variations in lighting and reflectance, as well as occlusion and destruction, more efficiently than the original NeRF algorithm.

A critical difference between Pixel-NeRF and the original NeRF is the way they model the radiation field of a scene. The original NeRF model represents the radiance field as a continuous function of 3D position and viewing direction [1], while Pixel-NeRF represents a discrete function of pixel position and viewing direction [20]. This allows Pixel-NeRF to better handle high-frequency details in the scene, such as textures and refined geometry, as well as to better preserve the appearance of the scene from different viewpoints.

The images in the original NeRF dataset are only used for supervised training [1]; these images are not only used as supervisory tools but also as input information to guide the learning process of pixel NeRF [23]. Precisely, Pixel-NeRF uses image information from known viewpoints to learn neural representations in 3D space. To synthesize a single-view image, a feature vector W is extracted from the input image I. Then, for each point x on the camera ray, the coordinates $\pi(x)$ are obtained by projecting x onto the image plane and interpolating the pixel values using bilinear interpolation [20]. This process is called ray sampling and is used to obtain the features corresponding to the points on the target view [24]. The feature vector $W(\pi(x))$ is then fed into the original NeRF network f, along with the position vector and view orientation, as expressed in the equation $f\left(\gamma(x), d; W(\pi(x))\right) = (\sigma, c)$, where $\gamma(\cdot)$ represents the encoding of the position of x. Based on these features and poses, the network can then generate the final image by rendering the body-based view image.

2.2. Evaluation & Metrics

In this study, we will evaluate the performance of various NeRF models using three widely used image quality metrics: peak signal-to-noise ratio (PSNR), structural similarity index (SSIM), and learned perceptual image patch similarity (LPIPS).

PSNR is calculated by dividing the reference image's maximum power by the noise's power in the reconstructed image, and the result is expressed in decibels (dB). Higher PSNR values indicate a higher-quality image reconstruction [25].

SSIM is a metric that quantifies the structural similarity between two images. It is based on the theory of human visual perception and is sensitive to local structural changes in the image. SSIM values range from 0 to 1, with higher values indicating higher similarity. If two images are identical, the SSIM value is 1 [26].

LPIPS is a metric that measures the similarity between small patches of pixels within an image, as human perception of image quality is determined mainly by the similarity of these patches. LPIPS calculates the similarity between these patches and their corresponding human-rated quality scores by training a deep neural network on a large image dataset. LPIPS scores range from 0 to 1, with lower values indicating higher similarity [27].

PSNR and SSIM are standard image quality metrics. LPIPS can more accurately reflect human perception. These metrics will be computed on a set of test images from our dataset to provide a comprehensive and representative sample and an overall assessment of the model's capabilities.

DOI: 10.54254/2755-2721/6/20230957

3. Results

3.1. Dataset

The dataset used in this study is a collection of 300 images of a single game character in a static environment. To obtain these images, we recorded a 300-degree video of the character through a computer screen and then split the video into individual images taken from different angles. To ensure that the results of our analysis could be reliably compared, we took care to record and segment all of the images under the same computer and software conditions.

To prepare the dataset for model training, we applied the Colmap pipeline to extract position and direction information from the images [28]. In addition, we preprocessed the data to remove any blurry or fragmented images that might introduce noise into our analysis, and we aligned the camera positions to ensure that the scenes in the dataset were consistent. As a result, the dataset represents a wide range of lighting conditions, viewing angles, and object categories. Figure 1 is an example of an image from the dataset. The images in the dataset have a resolution of 720×720 pixels.



Figure 1. Examples of character images in our dataset are shown. In order to properly evaluate the performance of specific NeRF models and compare them, we have chosen images with distinct scene backgrounds rather than characters with a black background for training, which may not accurately reflect the model's capabilities in generating full 3D scene images.

For the different NeRF models that we compared in this study, we represented and used the information contained in the 300 images and locations in different ways. However, the overall amount of information in the dataset remained the same for all models.

3.2. Comparisons

To further evaluate the rendering effect of the NeRF model on game characters, a comprehensive comparison of the training results was performed on a dataset of 300 images using different models. The image results after training are shown in Fig. 2, and the results of the data are presented in Table 1. As can be seen from the table, Mip-NeRF outperforms the other models in all metrics, with the highest PSNR and SSIM scores and the lowest LPIPS score. In contrast, Pixel-NeRF scored lower than the other models in PSNR and SSIM metrics and was outscored by the original unoptimized NeRF model.

DOI: 10.54254/2755-2721/6/20230957

Table 1. Comparison of different NeRF models in terms of PSNR, SSIM, and LPIPS metrics. Higher values of PSNR and SSIM indicate an elevated level of quality. SSIM has a range of 0 - 1. A smaller value of LPIPS is preferred, with a minimum of 0.

	PSNR (↑)	SSIM (↑)	LPIPS (↓)
NeRF	26.60	0.827	0.149
Mip-NeRF	29.83	0.892	0.117
Block-NeRF	27.21	0.840	0.136
Pixel-NeRF	19.18	0.679	0.271

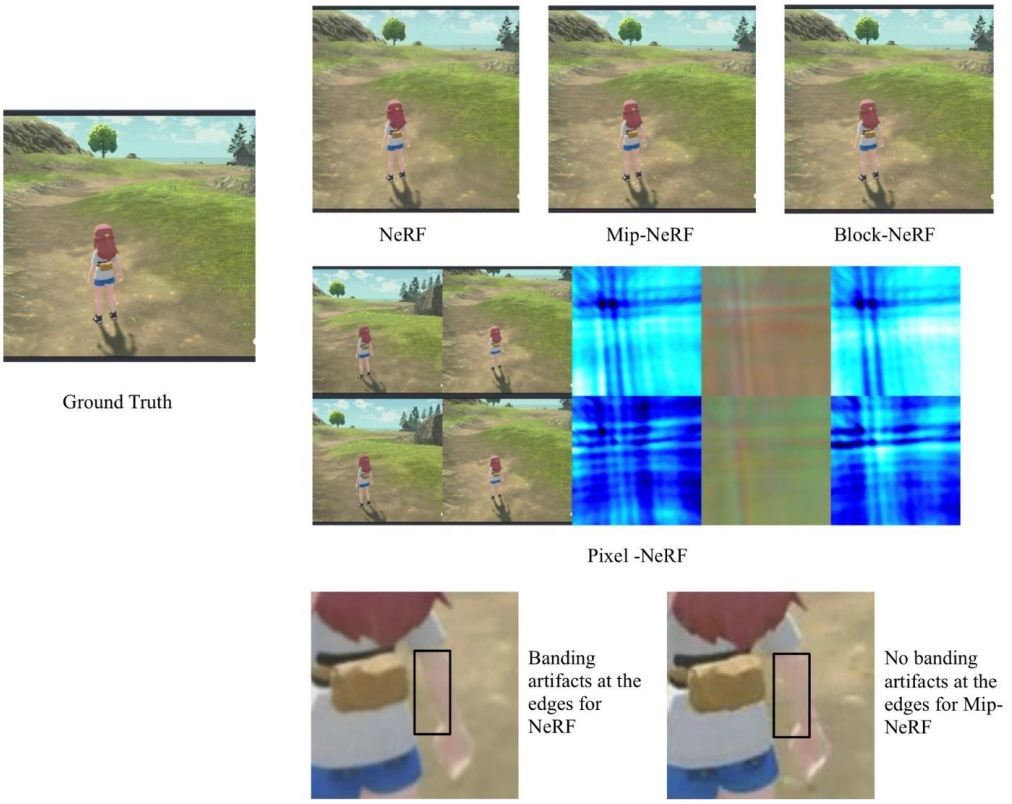


Figure 2. Comparison of scenes in our dataset on the test set view using different NeRF-based models.

The Original NeRF model exhibits banding artifacts at the edges of the figure, while Pixel-NeRF produces blurred and distorted renderings that fail to capture the figure's details accurately.

In addition to the quantitative results, we also performed a qualitative analysis of the model performance. We observe that for character images with fine details and complex backgrounds, Mip-NeRF achieves comparable or superior performance on all metrics, effectively composing 3D scenes of anime or game characters. These findings suggest that NeRF-based models can generate realistic character models or animations for video games and create immersive character avatars for virtual reality applications, which is expected to be helpful in the field of anime or game character design.

Overall, our comparison results suggest that NeRF is a powerful method for rendering game characters, and the Mip-NeRF variant is particularly effective in this regard. However, it is essential to note that further testing on larger and more diverse datasets may be necessary to fully understand the capabilities and limitations of these models and to improve their generalizability.

DOI: 10.54254/2755-2721/6/20230957

3.3. Discussion

Our experimental results demonstrate the effectiveness of NeRF for game character rendering, specifically in generating high-quality, photorealistic images in real-time. The model captures intricate details and nuances in a character's appearance, such as wrinkles and folds in clothing, and reflections and highlights on skin and hair. Furthermore, the real-time rendering capabilities of NeRF allow for smooth and seamless animation of character models.

Utilizing NeRF for game character rendering offers a significant advantage in producing high-quality images with realistic lighting and shadows without needing an expensive and time-consuming offline rendering process. This characteristic makes it possible to create more realistic and immersive game environments, with improved character models and animations that are more believable. In addition, NeRF's real-time rendering features can assist game designers and artists in accelerating iteration and development, making it a valuable tool for game development.

4. Conclusion

In this study, we compared the performance of four variants of the NeRF approach for training game/anime characters: NeRF, Mip-NeRF, Block-NeRF, and Pixel-NeRF. We used the PSNR, SSIM, and LPIPS as our main performance metrics to evaluate the training performance of these models on our dataset. Additionally, we compared the rendering details and visual quality of the images generated by these models.

Mip-NeRF achieved the highest PSNR score in our experiments, with a value of 29.83. However, original NeRF can obtain high PSNR and SSIM scores in a relatively short training time due to NVIDIA's transformation into instant-NeRF, which allows for efficient training while preserving the core principles of the original NeRF approach. This made NeRF a good trade-off between performance and efficiency.

Our findings demonstrate that NeRF is powerful for generating highly detailed and accurate representations of game/anime characters, including their surface textures and lighting effects. The ability of NeRF to generate high-quality, photo-realistic images of game characters in real time can significantly enhance the visual experience for players and inform the selection of models for similar tasks in the future.

However, one potential limitation of using NeRF for rendering game characters is the complexity of the scene being rendered. NeRF is a powerful method for generating detailed and accurate 3D representations, but it can be computationally intensive and may require extensive hardware and software resources to train and use effectively. This may make it challenging to use NeRF to render complex game characters with a large number of vertices, high-resolution textures, and intricate surface details. While this research provides valuable insights into the performance of different machine learning models on specific datasets and informs future research in the field, further optimization and investigation will be necessary as datasets and tasks become more complex to enable NeRF-based models to handle a broader range of data and challenges.

References

[1] Mildenhall B, Srinivasan PP, Tancik M, Barron JT, Ramamoorthi R, Ng R. Nerf: Representing scenes as neural radiance fields for view synthesis. Communications of the ACM. 2021 Dec 17;65(1):99-106.

[2] Gao K, Gao Y, He H, Lu D, Xu L, Li J. Nerf: Neural radiance field in 3d vision, a comprehensive review. arXiv preprint arXiv:2210.00379. 2022 Oct 1.

[3] Chang AX, Funkhouser T, Guibas L, Hanrahan P, Huang Q, Li Z, Savarese S, Savva M, Song S, Su H, Xiao J. Shapenet: An information-rich 3d model repository. arXiv preprint arXiv:1512.03012. 2015 Dec 9.

[4] Hou Y, Chen M, Volk R, Soibelman L. Investigation on performance of RGB point cloud and thermal information data fusion for 3D building thermal map modeling using aerial images under different experimental conditions. Journal of Building Engineering. 2022 Jan

DOI: 10.54254/2755-2721/6/20230957

1;45:103380.

[5] Peng S, Dong J, Wang Q, Zhang S, Shuai Q, Zhou X, Bao H. Animatable neural radiance fields for modeling dynamic human bodies. In Proceedings of the IEEE/CVF International Conference on Computer Vision. 2021 (pp. 14314-14323).

[6] Levoy M, Hanrahan P. Light field rendering. In Proceedings of the 23rd annual conference on Computer graphics and interactive techniques. 1996 Aug 1 (pp. 31-42).

[7] Drebin RA, Carpenter L, Hanrahan P. Volume rendering. ACM Siggraph Computer Graphics. 1988 Jun 1;22(4):65-74.

[8] Kaufman AE, Mueller K. Overview of volume rendering. The visualization handbook. 2005 Jan 1; 7:127-74.

[9] Chen W, Ling H, Gao J, Smith E, Lehtinen J, Jacobson A, Fidler S. Learning to predict 3d objects with an interpolation-based differentiable renderer. Advances in Neural Information Processing Systems. 2019; 32.

[10] Barron JT, Mildenhall B, Tancik M, Hedman P, Martin-Brualla R, Srinivasan PP. Mip-nerf: A multiscale representation for anti-aliasing neural radiance fields. In Proceedings of the IEEE/CVF International Conference on Computer Vision. 2021 (pp. 5855-5864).

[11] Amanatides J. Ray tracing with cones. ACM SIGGRAPH Computer Graphics. 1984 Jan 1;18(3):129-35.

[12] Niemeyer M, Mescheder L, Oechsle M, Geiger A. Differentiable volumetric rendering: Learning implicit 3d representations without 3d supervision. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2020 (pp. 3504-3515).

[13] Tancik M, Casser V, Yan X, Pradhan S, Mildenhall B, Srinivasan PP, Barron JT, Kretzschmar H. Block-nerf: Scalable large scene neural view synthesis. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2022 (pp. 8248-8258).

[14] Lin CH, Ma WC, Torralba A, Lucey S. Barf: Bundle-adjusting neural radiance fields. In Proceedings of the IEEE/CVF International Conference on Computer Vision. 2021 (pp. 5741-5751)

[15] Su SY, Yu F, Zollhoefer M, Rhodin H. A-nerf: Surface-free human 3d pose refinement via neural rendering. arXiv preprint arXiv:2102.06199. 2021 Feb 11.

[16] Wang Z, Wu S, Xie W, Chen M, Prisacariu VA. NeRF--: Neural radiance fields without known camera parameters. arXiv preprint arXiv:2102.07064. 2021 Feb 14.

[17] Yariv L, Kasten Y, Moran D, Galun M, Atzmon M, Ronen B, Lipman Y. Multiview neural surface reconstruction by disentangling geometry and appearance. Advances in Neural Information Processing Systems. 2020;33:2492-502.

[18] Yen-Chen L, Florence P, Barron JT, Rodriguez A, Isola P, Lin TY. inerf: Inverting neural radiance fields for pose estimation. In 2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). 2021 Apr 1 (pp. 1323-1330). IEEE.

[19] Martin-Brualla R, Radwan N, Sajjadi MS, Barron JT, Dosovitskiy A, Duckworth D. Nerf in the wild: Neural radiance fields for unconstrained photo collections. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2021 (pp. 7210-7219).

[20] Yu A, Ye V, Tancik M, Kanazawa A. Pixelnerf: Neural radiance fields from one or few images. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2021 (pp. 4578-4587).

[21] Shin D, Fowlkes CC, Hoiem D. Pixels, voxels, and views: A study of shape representations for single view 3d object shape prediction. In Proceedings of the IEEE conference on computer vision and pattern recognition. 2018 (pp. 3061-3069).

[22] Bautista MA, Talbott W, Zhai S, Srivastava N, Susskind JM. On the generalization of learning-based 3d reconstruction. In Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision. 2021 (pp. 2180-2189).

[23] Tatarchenko M, Richter SR, Ranftl R, Li Z, Koltun V, Brox T. What do single-view 3d reconstruction networks learn?. In Proceedings of the IEEE/CVF conference on computer

DOI: 10.54254/2755-2721/6/20230957

vision and pattern recognition. 2019 (pp. 3405-3414).
[24] Tulsiani S, Zhou T, Efros AA, Malik J. Multi-view supervision for single-view reconstruction via differentiable ray consistency. In Proceedings of the IEEE conference on computer vision and pattern recognition. 2017 (pp. 2626-2634).
[25] Huynh-Thu Q, Ghanbari M. Scope of validity of PSNR in image/video quality assessment. Electronics letters. 2008 Jun 19;44(13):800-1.
[26] Wang Z, Bovik AC, Sheikh HR, Simoncelli EP. Image quality assessment: from error visibility to structural similarity. IEEE transactions on image processing. 2004 Apr 13;13(4):600-12.
[27] Zhang R, Isola P, Efros AA, Shechtman E, Wang O. The unreasonable effectiveness of deep features as a perceptual metric. In Proceedings of the IEEE conference on computer vision and pattern recognition. 2018 (pp. 586-595).
[28] Schonberger JL, Frahm JM. Structure-from-motion revisited. In Proceedings of the IEEE conference on computer vision and pattern recognition. 2016 (pp. 4104-4113).