Journal of Electronics and Information Science (2023)
Clausius Scientific Press, Canada

DOI: 10.23977/jeis.2023.080606
ISSN 2371-9524 Vol. 8 Num. 6

Research of 3D Virtual Characters Reconstructions Based on NeRF

Lingyi Song

Chongqing No.8 Secondary School, No.8 Gongyuanbei Road, Yubei District, Chongqing, China

Keywords: Neuron Radiance Field; 3D-Reconstruction; Human Reconstruction

Abstract: With the development of the Internet, Metauniverse, and graphics processing technique, video games and immersive virtual social contacting service are largely propelled. Therefore, the quality of 3D virtual character models is getting more and more important—higher fidelities of screens have largely promoted the demand for finer 3D character models. However, the cost and time efficiency of traditional modeling relied on human artist can hardly support such a great demand, and will potentially slow down the development of video game and metauniverse industry. In an effort to improve this situation, this paper conducted research about 3D virtual characters reconstructions through NeRF and illustrated the principals and functions of NeRF. Using two different datasets (Doll Photos Dataset and Real-Human Photos Datasets), this paper evaluated the NeRF model and provided researching advice for future research about dataset building and possible directions.

1. Introduction

1.1 Research Backgrounds

Within the last decade, a considerable number of innovations and breakthroughs related to the internet occurred. For example, in regards to computer hardware, corporations such as Intel, AMD, and Nvidia Corporations significantly thrusted the upgrading of CPUs and GPUs, greatly increasing the processing efficiency of terminal devices. Among all, Nvidia Corporation has developed 2 generations of high-performance GPUs, enabling PCs and other terminal devices to cope with much more sophisticated Graphics and Deep Learning tasks. Besides, Nvidia has also developed a lot of corresponding techniques, such as new Ray Tracing technique in RTX GPUs and DLSS, which elevated the graphic performance and fluency of visual processing tasks. In regards to the Internet, 5G technique development leaded by Huawei Corporation has raised another technical revolution of the Internet. The 5G technique increased data transmitting efficiency, assisting massive amount of data to be transmitted in a short period of time and with little delay, and thus making the solutions of tasks that require high data transmitting speed achievable.

1.2. Market Demands

By the help of such technique, a lot of innovations in Computer Science has been proposed and

achieved. New techniques such as VR and AR have walk into the daily life of individuals, providing a better experience of interacting with information for the users who have increasing appetites for such interaction. Techniques including Block Chain, Digital Currency, Artificial Intelligence, Human-Computer Interaction, etc. has becoming more and more mature and have combined together. Under this intense stimulation, the concept of "Metauniverse" was proposed, owning great potential value. The term, "Metauniverse", refers to the virtual world composed of digital systems mapping and interacting the reality. Through mapping the reality, Metauniverse can satisfy personal social contacting demands, promote economy, and even raise the working efficiency. Hence, research about supportive techniques for the Metauniverse is entrusted with great values.

In regards to the entertainment industry, the occurrence of novel graphics techniques and transmitting techniques have laid bedrocks for the innovations in video game industries. Currently, the game industry has made eminent contribution to the development of economy and science. According to *Chinese Gaming Industry Report 2022* [1], the Chinese video game market is immense and economy-contributive: in 2022, the number of Chinese game customers is about 664 billion, and the total income has reached about 265.884 billion CNY. Additionally, according to the report from Yanyu Wang and his team in Chinses Academy of Science [2], the game industry has contributed 14.9%, 46.3%, and 71.6% TFP to chip, fast transmitting network, AR and VR industry, respectively. From the perspective of the trend of the industry, with higher data processing rate, the quality of video games will inevitably increase, and thus customers are more likely to purchase games that can provide them experiences closer to the reality.

Reviewing the whole history of the evolution of video games, it is not difficult to summarize that players are always pursuing games that are more immersive and resemble to the reality. In a game called *Tennis for Two* created in 1958 by physicist Willam Higinbotham, two players can play simulated tennis in an oscillograph. The whole vision is totally abstract—all elements in the game are represented only by simple lines and points. The visual effect of later games developed by the Atari corporation are quite similar to that of *Tennis for Two*.

The later game corporations including Nintendo and SONY developed many gaming devices, such as Game Boy series developed by Nintendo corporation, PS series developed by SONY corporation. These devices have opened a thoroughly new era for electronic games. Comparing to the games earlier, the visions of games at that time began to be more vivid—games are usually colored, and some games even adopted pseudo-3D effects to make their games more interesting.

Once when the computing efficiency of PCs has been largely ameliorated, on 22 June 1996, a revolutionary game *Quake III* occurred. *Quake III* is the first real-3D game, and it introduced the concept of 3D engine to the whole game world. Its occurrence has laid a milestone for the great leap in visual effects of game industry. In later games, such as *Half-Life* and *Counter-Strike: Global Offense* (also known as CS: GO), it is not difficult to discover that the vision provided by video games are becoming closer and closer to the reality.

1.3 3D Character Modeling

It is getting more and more difficult and resource taking to develop a game since the requirements of the market to the overall quality is getting increasingly high. Currently, modeling tasks in game industry is mainly done by professional 3D artists, but it costs a relatively long period of time to produce every model. Usually, during the game production, a character model will experience the following 9 steps from being designed to finished.

1): Raw Model production (generally 2 days);

2): Production of the character's clothes (7 days);

3): Detail processing, which is a relatively time taking step (5 days);

4): Production of a less precise model (to increase the fluency of games) (3 days);
5): Separating UV map (1 day);
6): Baking normal map to the less precise model (1 day);
7): Texture producing (5 days);
8): Skeleton bonding (1 day);
9): Model rendering (3 days);

After experience the aforementioned steps, the final product cannot be guaranteed to be selected as the final version in the game, so each artists need to repeat the process for many times, greatly increasing the developing cost of video games. Besides, the skills required by character modeling can hardly be mastered by ordinary users, and therefore in scenes such as the metauniverse, common users struggle to create a model for themselves.

1.4 Research Direction

Therefore, if an easy modeling method is developed, the resource costed and time spent on characters modeling will be significantly decreased, and ordinary user can finish their modeling by themselves, apparently improving the possibilities of character modeling and game developing.

3D reconstruction is a field that combined multiple other fields including Computer Graphics, Computer Vision, Deep Learning targeting to construct models of real-world objects automatically. The application range of 3D reconstruction is broad. In scene reconstruction task, Werner, Ayoub Al-Hamadi, and Philipp Werner proposed an algorithm called TSDF [3]. TSDF algorithm used a series of depth camera to catch 2D information from the reality, and then calculate the locations of each voxel through geometric models and reconstruct the whole scene. TSDF has high performance in terms of reconstructing scenes. In 3D human reconstruction, Zhenbao Liu and his team [4] proposed a method in 2015 to generate human model through pictures captured by multiple depth cameras. However, both methods mentioned above depend on depths cameras such as Kinect, which might cause extra cost to individuals and organizations. Other than these, methods for 3D human reconstructions such as SMPL [5], which uses parameters to fit captured photo data with well-adjusted human model, demonstrated good performances in recognizing human body, but it is difficult for these algorithms to satisfy high-fidelity requirements of current character modeling.

Unlike the aforementioned algorithms, the recently developed NeRF technique can complete the task of high-quality character model reconstruction through a series of photos from different angles taken by phone camera, and thus can decrease the cost of modeling to an accessible degree that individuals can accept while not losing any model details. Hence, in this paper, 3D human reconstruction task will be accomplished based on NeRF.

2. Related Works

2.1 3D Rendering

3D rendering is an important research field in Computer Graphics. The main topic of it is to display various 3D models on the screens quickly and precisely though methods involving perspective, rasterization, color rendering, and ray tracing. A 3D model can be represented in two ways in computers: explicit representation and implicit representation. Explicit representation expresses the model through mapping the models by parameters, while implicit representation expresses the model through describing the model by mathematical or other relationships. Commonly used methods of explicit representation are point cloud, mesh, and voxel representation. Comparing to explicit representation, it is simple for implicit representation to cope with structures such as liquid and fire. However, it is difficult for implicit representation to handle compound-and-complex 3D bodies such

as human. Rendering can be achieved by sampling these 3D models and convert them into discrete 2D matrix for displaying.

As a branch of 3D rendering, Volume Rendering focuses on the rendering of non-rigid bodies such as smoke, cloud, and fire. Volume Rendering is tightly connected with Optics. The fundamental principal of volume rendering is promoted by Nelson Max in 1995 [6]—by simulating the absorption, emission, in-scattering, and out-scattering of light by the objects, the ray passed through the voxel of computed. By adding up the aforementioned three values and then sampling equidistantly, every value of pixels on the screen can be calculated.

2.2. Neuron Radiance Field (NeRF)

The Neuron Radiance Field (NeRF) algorithm was promoted int 2020 by Ben Mildenhall and his team [7]. It was inspired by the reverse-operation of volume rendering. The 3D information, including color and the 3D coordinate of the bodies can be deducted by photos from different angles, which equals to the outcome of volume rendering. Then, through the deducted 3D information, 3D model can be constructed. NeRF is widely applied to tasks including Novel View Synthesis, 360 Degree Reconstruction, Scene Reconstruction, Human Reconstruction, and 3D Style Transferring. NeRF converted a 3D scene into a 5D vector (x, y, z, θ,Φ, where x, y, z represents the coordinate, θ and Φ represents the direction of camera), and then the density (the possibility of light being blocked by a voxel when passing through it) of a voxel can be obtained. Afterward, a ray with color will be ejected from the camera point. The ray will be sampled first coarsely and then finely to approximate the data of a space point. Repeating for a several times and send these space point data into a fully-connected neuron network, the implicit representation of a model can be learned (as shown in figure 1).

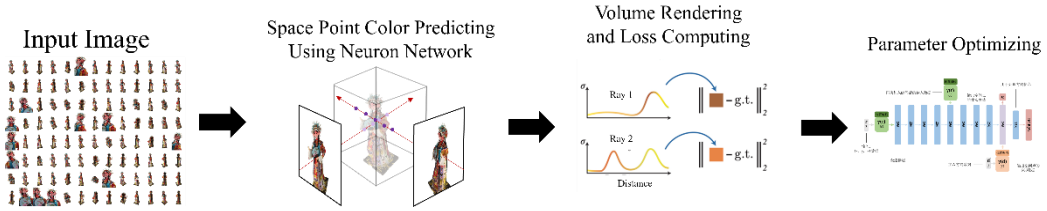

Figure 1: NeRF Learning Implicit Expression of Models

3. Data Collection and Pre-processing

For NeRF requires photo datasets with sufficient angles and high resolution, all datasets used in this paper is composed of multiple pictures taken around the character. There are 3 datasets used in the experiment: a doll dataset and two real-human photo datasets.

3.1 Collecting Data Pictures

3.1.1 Doll Dataset

The Doll Dataset was collected from the dataset “Character” in paper *Blended MVS: A Large-scale Dataset for Generalized Multi-view Stereo* [8] by Yao Yao et al. in 2019. As shown in the figure 2, the 110 pictures in the dataset contains photos taken around a doll from three different heights (top, face, and overview), each of which are sized 576*576 pixels. In each picture, the background is pure white, and the proportion in the picture the character occupied is varying.

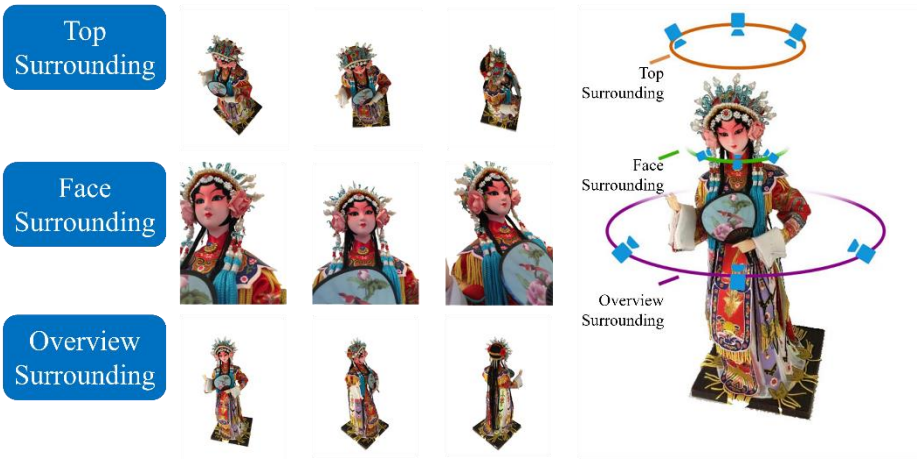

Figure 2: Sample Data in the Doll Dataset and Camera Motion Trajectory Demonstration

3.1.2 Real-Human Photo Dataset

Real-Human Dataset I is composed of pictures captured at different time in a video taken by a phone. After filtering pictures in which human figure are not intact, there are 90 pictures from different angles surrounding the volunteer from high to low (shown in figure 3).

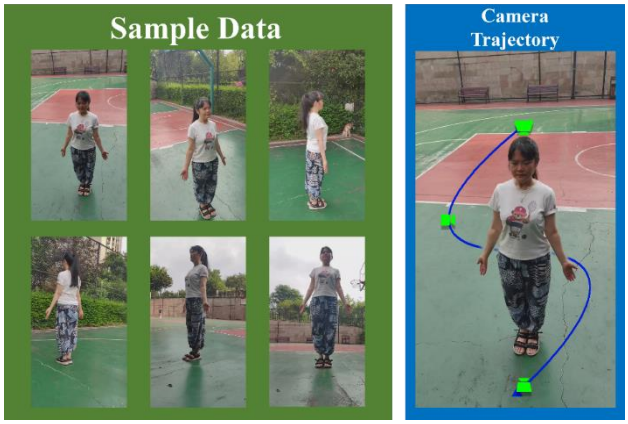

Figure 3: Camera Trajectory and Sample Data of Real-Human Photo Dataset I

As shown in figure 4, Real-Human Dataset II is composed of pictures captured at different time in a video taken by a phone. It contains 150 pictures, all of which sized 2160*3840 pixels and are pre-processed: they were resized uniformly, centered, incline and lens distortion corrected, and background-removed. The photos are taken in three distinct views: top, lower top, and eye level.

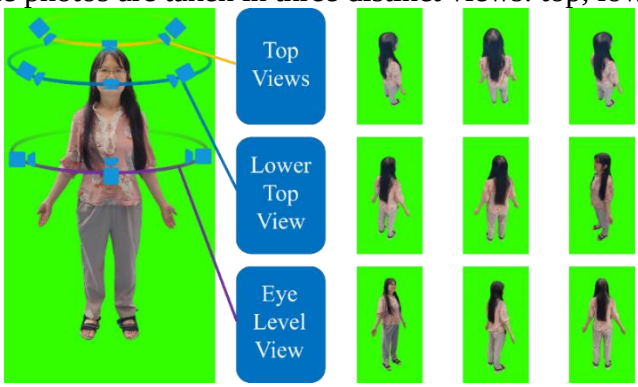

Figure 4: Camera Trajectory and Sample Data of Real-Human Photo Dataset II

3.2 Data Processing and Dataset Generating

After successfully collected all raw images, the data needs to be processed to generate LLFF data for NeRF to read. When LLFF data is read by NeRF, since the axis it used is not congruent with that is used in NeRF, LLFF data needs to be transformed. Other than LLFF data, what is also required by NeRF to complete its training is the images down sampled by a factor of 8 of original images.

3.2.1 Generating LLFF Data

Before generating LLFF data, the camera parameters, camera poses, key points, and coarse point cloud data need to be extracted from the images, each of which, in this paper, are produced by the SfM (Structure from Motion) process using COLMAP.

3.2.2 Generating Images Down Sampled By 8 of The Original Images

To generate down sampled image according to the original image, a coevolution process with the following steps needs to be operated:

1) Define a coevolutionary kernel of n*n.

2) Slide window with the same size as the kernel from left to right and top to bottom by stride s each time and dot product the data in the window with the coevolutionary kernel.

3) Out put the average product of the operation to a new matrix.

As shown in figure 5, to generate an image down sampled by a factor of 2, a 2*2 coevolutionary kernel and corresponding stride of should be defined. Then, operate the same process again on the result, the image of the original image down sampled by a factor of 4 could be generated, and similarly the image down sampled by a factor of 8.

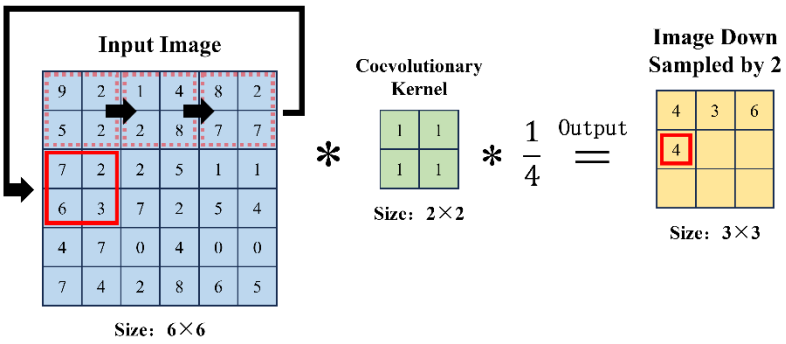

Figure 5: Down Sampling Through Coevolution

After gaining required LLFF data and images down sampled by a factor of 4 and 8, NeRF training can be started with the completed datasets.

4. Network Designing

The training process of NeRF was separated to the following step: Data Input, MLP Network Initializing, Staring Training, Voxel Rendering, Loss Computing, and Network Optimizing, each of which will be detailed later.

4.1 Data Input

The Data Input process is separated into three steps: Original Data Inputting, Poses Recentering, and Spiral Trajectory Generating.

4.1.1 Original Data Inputting

In this part, NeRF will read the processed LLFF data and the images down sampled by a factor of 8. Since using different standards of dbs information and axis system, the program will firstly convert them to the standard used in NeRF, and store them into the memory for further use. Then, the program compresses the RGB value between 0 and 1 to prevent data overflow.

4.1.2 Poses Recentering

Finishing reading the original data, NeRF processed the poses data in the memory. First, NeRF calculated an average value of all poses as the center of world coordinate. Then, it rotated all poses to keep it congruent with the X, Y, and Z axis of the world coordinate.

4.1.3 Spiral Trajectory Generating

After unifying the world coordinate, NeRF generated a series of camera poses spiraling and always-facing the origin point of the world coordinate. When outputting video, NeRF will use these poses to capture different frames of the video.

Completing all of the aforementioned parts, these data could be used to train and generate the character model.

4.2 MLP Neuron Network

Before starting training the network and generating 3D model, the MLP network used for sample points' RGB value and density prediction for further use. The MLP network is composed of four parts: Position-Encoding Function, MLP Neuron Network, Model Training Function, and Optimizer.

4.2.1. Position-Encoding Function

Since MLP network is better fitted to low-frequency functions, using x, y, z and view-direction data, which are all high-frequency functions, directly as the input data might cause problems with resolution rate and model detail in the final product, the input data need to be processed by Position-Encoding Function, which increases the input vector's dimensionality and decreases its frequency to assist the training of the MLP through calculation in formula (1). In formula (1), $\gamma(p)$ represents the Position-Encoding Function, p represents the input vector, and L represents dimensionality (larger L enables Position-Encoding Function to fit vectors with higher dimensionality). The input value is the x, y, z vector, and the output value is a 63-dimension vector.

$$\gamma(p) = (\sin(2^0\pi p), \cos(2^0\pi p), \ldots, \sin(2^{L-1}\pi p), \cos(2^{L-1}\pi p)) \tag{1}$$

4.2.2. MLP Neuron Network

NeRF separated MLP networks of coarse rendering and fine rendering to prevent co-interruption between. Coarse model network is used for prediction of RGB and density values in equidistantly sampled points, while fine model network is used for prediction of RGB and density values in finely sampled points. As shown in figure 6, the structure of both MLP function, in sequence, is:

1) The first 3-dimension input layer for x, y, z axis of predicted points;

2) The first position-encoding function with L value of 10 to increase the dimensionality of the input vector, outputting a 63-dimension vector;

3) 8 full-connection layers with 256 dimensions.

4) A 256-dimension feature layer composed of the outputs of full-connecting layers before,

outputting predicted density α. Position-encoded (with L valued 4) direction data is added to this layer to this layer;

5) A 128-dimension full connection layer connected to the feature layer for direction data processing;

6) A 5-dimension output layer, outputting predicted RGB data (3 dimensions) and view-direction (2 dimensions).

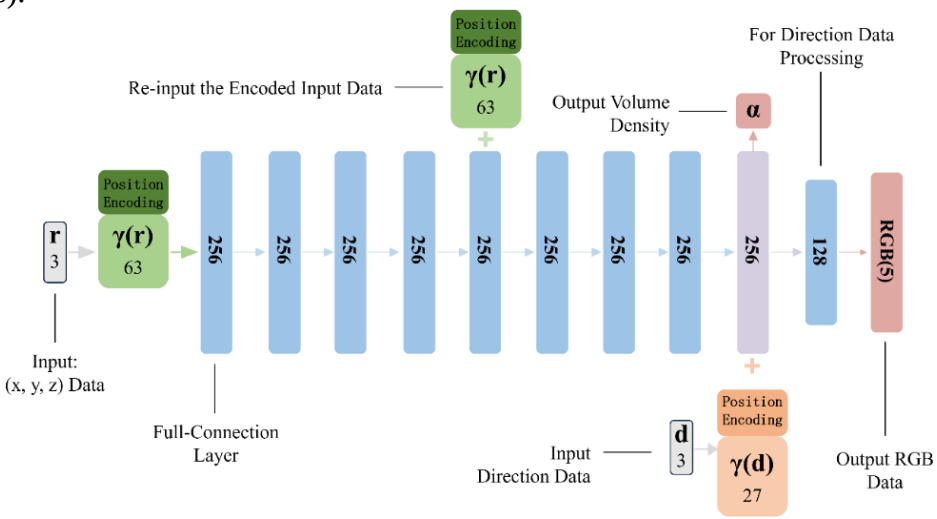

Figure 6: Structure of the Neuron Network

4.2.3 Model Training Function

The Model Training Function is used to define the operations when training. It encoded the input data and sent it into the full-connection layer, defining the forward-propagation method of all outputs simultaneously.

4.2.4 Optimizer

The Optimizer of NeRF is Adam Optimizer proposed firstly in 2017 by Diederik P. Kingma and Jimmy Lei Ba [9]. Adam Optimizer is widely used in deep learning tasks for its high computing efficiency, interpretability, and utility.

4.3 Volume Rendering

With the MLP network Successfully defined, the rendering of 3D model can be started. In NeRF, there are two steps for volume rendering: Ray Generating and Volume Rendering Based on Generated Rays.

When generating rays, NeRF firstly selected the origin point of the camera as starting points stored as O of all rays, and then mapped all pixels p to the coordinate of the camera through formula (3) (to be used as the direction of the rays). After, using camera poses, NeRF mapped the origin O and coordinate of the pixels P into the world coordinate to define a ray through connecting O and all P. In formula (3), u represents the column number of a pixel, v represents the row number of a pixel, c_x and c_y are the optical center of the camera, and f_x and f_y are the focal length of the camera. To fit the coordinate system used in NeRF, Y axis and Z axis are inverted.

$$[x, y, z] = [(u - c_x)/f_x, -(-v - c_y)/f_y, -1] \tag{2}$$

Finishing generating rays of all images, NeRF will separate all rays into multiple ray batches each

contains a relatively smaller number of rays to prevent memory overflow when rendering.

Then, volume rendering will be started. First, the program will sample each ray equidistantly, generating a series of sample points. Then send the coordinate of these sample points and view-direction into the defined MLP network for prediction of their density (σ) and color (c). After, the predicted color c, density σ, distance r and direction d on the camera ray, and distance between sample point and camera ray t to the volume rendering function for volume rendering. The Volume Rendering Function can be represented by formula (4), where the upper limit t_f and lower limit t_n in the integration represents the closest and farthest distance in the world respectively. Since data such as voxels and pixels are discrete and cannot be integrated, the aforementioned formula needs to be discretized.

$$C(r) = \int_{t_n}^{t_f} T(t)\sigma(r(t))c(r(t),d)dt$$
$$in\ Which\ T(t) = exp\left(-\int_{t_n}^{t}\sigma(r(s))ds\right) \tag{3}$$

Completing coarse-network rendering, NeRF will compute the contribution of each sample point to the result of coarse-rendering, and generate a series of fine sample points according to it. Then, the fine sample points will be send into the MLP network for prediction and be volume-rendered. An epoch of training is over when finishing fine-network volume rendering. After, NeRF will compute losses of this epoch, optimize the parameters, and start the next epoch.

4.4 Loss Computing

After an epoch of 3D rendering, NeRF will used the rendered model to generate an image with the same view direction as the original and compute their PSNR (Peak Signal-to-Noise Ratio) value as the loss to be sent into the optimizer. PSNR is a significant evaluating indicator of image reconstruction with the unit of decibel, and it is widely applied in image compressing and reconstruction. PSNR can be calculated by formula (5):

$$PSNR = 10 \cdot \log_{10}\left(\frac{MAX_I^2}{MSE}\right)$$
$$in\ which\ MSE = \frac{1}{mn}\sum_{i=0}^{m-1}\sum_{j=0}^{n-1}[I(i,j) - K(i,j)]^2 \tag{4}$$

In the formula above, I and K refers to two monochrome images, and (i, j) represents pixel located at the i^{th} row and j^{th} column in the image. MSE is the mean square error of the two images, and MAX_I is the maximum value of color, which, in 8-bit images, is 255. The minimum value of PSNR is 0, and the larger the value is, the closer the two images are. For 8-bit images, a PSNR between 30db and 50db is considered as idea.

5. Model Training

Once when the designing of NeRF model is finished, the training and reconstruction of character models can be started.

5.1 Information of The Device Used

In this paper, a high-performance cloud server is used for the training in this paper. The server used

is located in Zhenzhou Province, China, with 12-core CPU with 92GB memory and a NVIDIA V100 GPU with 32GB memory. The operating system used in this paper is 64-digit Alibaba Cloud Linux 3.2104 LTS.

5.2 Virtual Environment Setting

Because the training of NeRF is relatively complex, it depended on a number of virtual environments shown in table 1. All of the depending environments need to be installed.

Table 1: Depending Environments and Their Versions

Depending Environments	Version
Python	3.7.0
CUDA	10.1
Anaconda	23.1
PyTorch	1.11.0
Torchvision	0.12.3
Torchaudio	0.11.0
Imageio	2.28.1
Matplotlib	3.5.3
ConfigArgParse	1.5.3
Tensorboard	2.11.2
Tqdm	4.65.0
OpenCV-Python	4.7.0.72

5.3 Training Configuration

Because cloud sever is used in this paper, the complete file of NeRF, datasets to be read, and the training configuration text are needed to be uploaded to the server first. Then, run_nerf.py file can be operated with the corresponding configuration to start the training.

5.4 Downloading Results

It takes for about 6 hours for NeRF to generate the final outcome. After NeRF finished training, the result will be downloaded and saved. The training results included parameters of the neuron network in different epochs, disparity map videos spiraling around the model from different epochs, RGB videos spiraling around the model from different epochs (which are the demanded results in this paper), and the rendered images from different epochs (which will be used to evaluate the results).

6. Results

When the trainings of 200,000 epochs are over, NeRF generated the spiraling videos and images of the Doll Dataset and the Real-Human Photo Dataset. In this part, the outcome of the experiments will be demonstrated and evaluated.

6.1. Outcome of the Doll Dataset

In the video output, the spiraling view of the doll shows an intact doll model without distortion or deforming. Shown in figure 7, NeRF's output is very similar to the original data down sampled by 8, indicating that the experiment is correct in principals of setting the configurations and collecting data.

The output model of the doll is a little vague, which is possibly caused by the inadequacy of resolution of the original data and training device's performance.

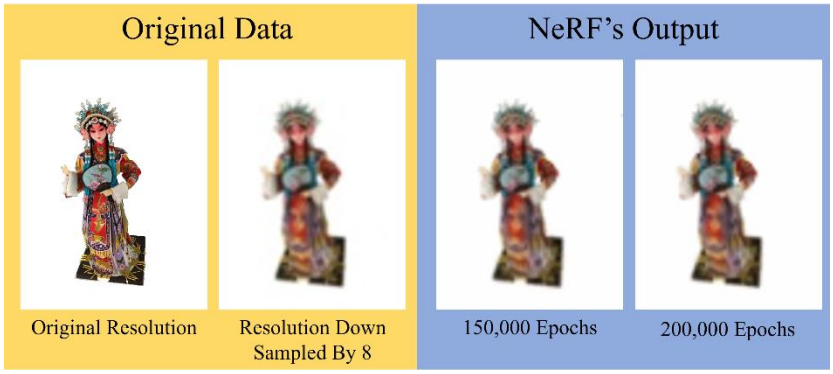

Figure 7: Comparison between NeRF Output and Original Data of The Doll Dataset

However, as shown in figure 8, there is an obvious strip-shaped defect passing through the neck of the doll. The reasons for the occurring error might be: (1) The down sample used might not fitted to NeRF well; (2) Some rays are faultily rendered; (3) The super parameters of training need to be adjusted.

It seems less possibly that the error is caused by reason 3, because the error has already occurred in epochs as early as 50,000 epoch and existed to the last (200,000 epoch).

Figure 8: Errors in the Output Model

Referring to the down-sampled-by-8 images, the PSNR distribution of the doll dataset's outcome is shown in table 2. In the training, the overall average PSNR and standard deviation of PSNR are always increasing. However, the highest average PSNR is still lower than 30db, which means that the artifact of the image is still apparent.

Table 2: PSNR Distribution of The Doll Dataset

Epoch(thousand)\PSNR(db)	Avg.	SD	Max	Min
50	26.4265	27.2775	33.4066	17.5609
100	27.0774	30.8427	34.4565	17.5336
150	27.1426	32.3479	34.8853	17.5490
200	27.1553	34.0592	35.1891	16.9960

6.2. Outcome of the Real-Human Photo Datasets

As demonstrated in figure 9 and 10, in the outcome of Real-Human Photo Dataset I, the scene and character are severely distorted to the degree that the image is unrecognizable from some angles. Such terrible quality of reconstruction is unexpected. It might be caused by the wrong way of gathering data: in the Real-Human Photo Dataset I, the location and size of the character in the images are not fixed and varying apparently. Additionally, the background of the character is sophisticated, affecting the experiment.

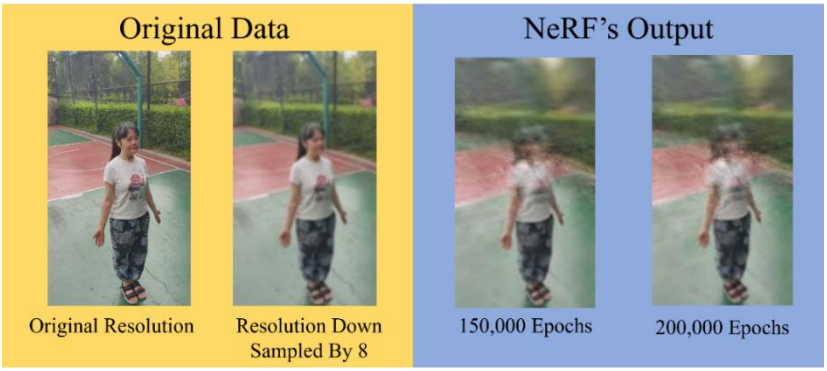

Figure 9: Comparison Between NeRF Output and Original Data of The Real-Human Photo Dataset I

Figure 10: Severe Distortion

Referring to the down-sampled-by-8 images, the outcoming PSNR distribution of Real-Human Photo Dataset I is shown below in table 3.

Table 3: PSNR Distribution of The Real-Human Photo Dataset I

Epoch (thousand)\PSNR(db)	Avg.	SD	Max	Min
50	18.6016	21.5258	27.4949	13.2895
100	18.5187	22.6042	18.2595	13.1981
150	18.4685	22.7167	28.2590	13.2201
200	18.4149	22.8209	28.2109	13.1508

The PSNR value of Real-Human Photo Dataset I is relatively stable, maintaining at the level of about 18.5, meaning that the artifact of the images is serious.

Because Real-Human Photo Dataset II is processed specially, the outcome of its experiment is much better (as shown in figure 11). Except for a little chromatism, ghosting, character disappearing when looking up (as shown in figure 12), the outcome is idea. According to the experience of improving the dataset, these problems might be caused by: 1. Inadequacy in perspective angles (lack of bottom view), causing the character to disappear when looking up; 2. Background-color mixed with character to lead to chromatism; 3. Lens distortion is not corrected completely, which may cause some degree of distortion and dislocation; 4. Objects in the images are larger getting closer, and it might cause errors in rendering.

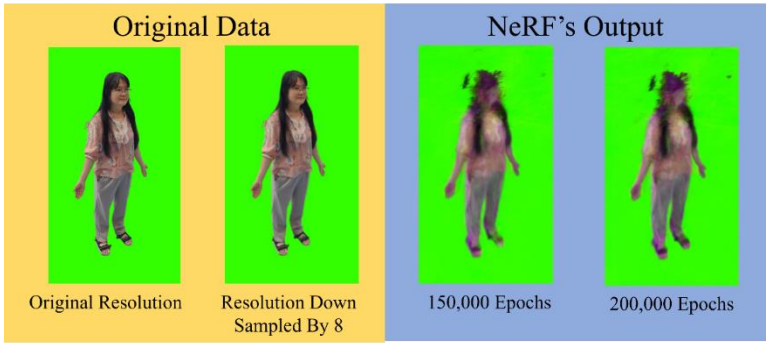

Figure 11: Comparison between NeRF Output and Original Data of The Real-Human Photo Dataset II

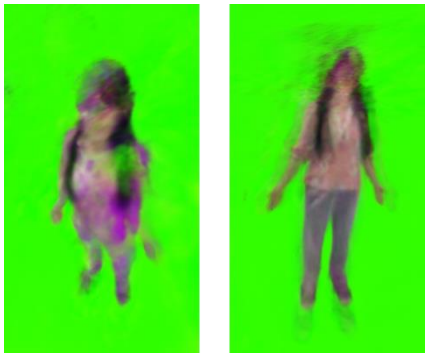

Figure 12: A Little Chromatism and Ghosting

Referring to the down-sampled-by-8 image, the outcome of Real-Human Photo Dataset is shown in table 4. The average PSNR maintained at approximately 19.7, but since the sample output available for PSNR computing are the samples with lower quality, the computed PSNR value might be lower than the actual value. Comparing to the result of Real-Human Photo Dataset I, that of Real-Human Photo Dataset II has lower SDs, meaning that the output is more stable.

Table 4: PSNR Distribution of the Real-Human Photo Dataset II

Epoch (thousand)\PSNR(db)	Avg.	SD	Max	Min
50	19.7205	11.2564	24.4945	14.6017
100	19.7750	13.7090	25.6575	14.1260
150	19.7795	14.0945	25.5775	14.0878
200	19.8419	15.0143	25.7517	14.0809

7. Conclusion

In this paper, research to reconstruction of virtual characters based on NeRF technique is conducted. Through pictures from multi-views, NeRF ejects rays and predicts the color and density of the sample points. Adding up the predicted color and density, a video spiraling character model can be rendered.

The principal and structure of NeRF is feasible and reasonable—the high fitting ability of neuron network enables NeRF to readily adapt model reconstructing in different situations. It is easy to conclude from the outcome of the Doll Dataset that NeRF can reconstruct complex models such as human-shape characters well when not specially adjusted. Although, due to the unfavorable video taking condition, NeRF performed terribly coping with Real-Human Photo Dataset I, NeRF demonstrated its ability to provide individual users virtual image creating service when the dataset is

improved. Experiencing a number of times ameliorating the datasets, this paper advices further research in terms of data collecting and processing in the following points:

1) When collecting human-shape images, try best to increase the angle-diversity, resolution of the images, and always pay attention to control the degree of perspective-distortion.

2) When processing the data images, replace the original background with pure-white background. Doing so is to prevent possible chromatics. If the color of the character is also white, select another color with higher contrasting rate.

3) When processing the data images, place the character close to the center of the image, and ensure that the sizes of characters in the images are roughly the same. When processing the top-view or bottom-view of the character, the character's size can be reduced to lower rendering errors cause by perspective distortion.

4) Always remember to correct lens distortion.

Because the Doll Dataset simulated the situation that is similar to the situation faced in the game corporations, there are several possible solutions follow: (1) Use devices with higher performance for training. (2) Collect data pictures with devices that support higher resolution (4K, 8K, or even 16K) to assist NeRF to cope with model details.

Finally, for future research associated with NeRF, the following advices are provided:

1) Specialize NeRF in terms of human-shape character modeling to improve its ability to process and reconstruct characters with higher efficiency and precision.

2) Optimize NeRF's ability to resist disruptions such as severe dislocation. Doing so can increase the overall utility of NeRF, enabling it to handle more sophisticated tasks, and ameliorate users' experience at the same time.

3) Optimize the training rate of NeRF. NeRF still takes a long period of time to complete training currently. Considering the time taken in data collection, NeRF did not show enough acceleration to the process compared to the traditional modeling method.

4) Add recognition function to NeRF to enable it model multiple objects in one scene, increasing the efficiency of large-scale modeling.

References

[1] China Audio-video and Digital Publishing Association (GPC). Chinese Gaming Industry Report 2022. February 2, 2023.

[2] China Audio-video and Digital Publishing Association (GPC). Gametech – A New Technology Cluster in The Process of Digital-real Integration. July 21, 2022.

[3] Diana Werner, Ayoub Al-Hamadi, Philipp Werner. Truncated Signed Distance Function: Experiments on Voxel Size. University of Magdeburg. 2014.

[4] Zhenbao Liu, Hongliang Qin, Shuhui Bu, Meng Yan, Jinxin Huang, Xiaojun Tang, Junwei Han. 3D Real Human Reconstruction Via Multiple Low-cost Depth Cameras. Northwestern Polytechnical University. 2017.

[5] Matthew Loper, Naureen Mahmood, Javier Romero, Gerard Pons-Moll, Michael J. Black. SMPL: A Skinned Multi-person Linear Model. 2015.

[6] Nelson Max. Optical Models for Direct Volume Rendering. University of California, Davis, and Lawrence Livermore National Laboratory. 1995.

[7] Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, Ren Ng. NeRF: Representing Scenes as Neuron Radiance Fields for View Synthesis. UC Berkeley, Google Research, and UC San Diego. 2020.

[8] Yao Yao, Zixin Luo, Shiwei Li, Jingyang Zhang, Yufan Ren, Lei Zhou, Tian Fang, Long Quan. BlendedMVS: A Large-scale Dataset for Generalized Multi-view Stereo Networks. The Hongkong University of Science and Technology, Everest Innovation Technology, and Zhejiang University. 2019.

[9] Diederik P. Kingma, Jimmy Lei Ba. Adam: A Method for Stochastic Optimization. 2017.