Bachelor thesis

Artificial Intelligence

Radboud University 

NeRF as super-resolution method

Author:
Sven Berberich
s1006248

First supervisor:
dr. R.S. van Bergen
AI department
rubenvanbergen@gmail.com

Second supervisor:
dr. T.C. Kietzmann
AI department
tim.kietzmann@donders.ru.nl

June 18, 2021

Abstract

This thesis is concerning itself with the question whether Neural radiance fields (NeRF) can be used to perform super-resolution. The NeRF method learns a scene representation by receiving images of said scene to train on. As this representation receives information from multiple images we hypothesise that the learned representation contains more information than a single image does and could thus create accurate high-resolution outputs while only training on low-resolution images.
We observe that the creation of HR images is quite possible, the quality of these is however, only in very limited situations, comparable with interpolation based super-resolution methods and significantly worse then state-of-the-art methods.

Contents

Chapter 1

Introduction

The idea of increasing the resolution of an image while accurately filling in the missing details exists since a long time in fiction. A famous example of this are scenes in criminal shows in which a technician is asked to zoom in and enhance a picture of a CCTV camera which reveals a crystal clear image of the zoomed-in area.

This concept however is more and more becoming reality in recent years, as improvements in AI have brought up multiple promising algorithms in the field of super-resolution and image processing in general.

In this thesis we inspect the super-resolution capability of NeRF [6] a neural network based view synthesis algorithm. NeRF is a method that takes multiple images of a scene to learn a volumetric representation of the given scene. This representation is embodied by a dense feed-forward neural network that fits a high-order function to the provided image data. A volumetric rendering method can then be used to recreate images from the volumetric representation by querying the neural network about information at certain locations with specific viewing angles.

The NeRF model can render images at any desired resolution as it simply uses its volumetric rendering method. This also means that it can render images with a much higher resolution than the images used to train it. The quality of these higher resolution images however, depends on how accurate the high-order function, that is embodied by the neural network, is at estimating the intermediate points for which the training data was not precise enough.

Since this method is trained on images taken from multiple angles it is likely that the resulting representation is more accurate than the representations

of conventional interpolation based methods [2] which only have access to information from a single image.

This thesis will therefore investigate how the NeRF method compares to conventional interpolation methods as well as how it compares to state-of-the-art, super-resolution methods.

Chapter 2

Related Work

The problem that super-resolution (SR) tries to solve is to create or reconstruct high-resolution (HR) images from low-resolution (LR) images. Many different methods using different approaches have been explored to solve this problem. Here we will shortly review a couple of them to gain a better understanding of the problem.

2.1 Interpolation methods

A commonly used approach to solve this problem are interpolation based methods. These methods scale up the LR image and fill in the missing information via interpolation between the known pixels. Examples of these methods are:

- **Nearest-neighbor interpolation** [2]
  This method assigns each unknown pixel the value of the closest known pixel. The resulting HR image is usually very similar to the LR image since all new pixels have values that already existed in the image.
- **Bilinear interpolation** [2]
  This method assigns new unknown pixels a value that is a weighted average of the closest known pixels. The weight of each pixel is based on the linear distance of the unknown pixel towards each known pixel so that a unknown pixel right next to pixel A receives an almost identical value to pixel A and a pixel in the middle between A and B receives a value that is exactly in between the values of A and B.

- **Bicubic interpolation** [2][3]
  In contrast to the other two methods that only take the closest known pixels into account this method considers the 16 closest pixels and applies a much more complex interpolation operation to them. This results in this method producing smother images.

2.2 Neural network based methods

Although the mentioned interpolation based methods work, there is still a lot of room for improvement. Via training on data sets, neural network based approaches can implicitly learn patterns in up-scaling that are not yet explicitly known.
In recent years such neural network based methods have started to outperform the more conventional interpolation based methods. The majority of these methods, including the ones we compare our method with, are neural networks that are pre-trained on large data sets with LR input and corresponding HR output images. These networks are trained to minimize the loss between their output and the HR images provided by the data set. The resulting scale factor of the super-resolution is therefore for most models specific to the data set they are trained on.

Some of these Methods are namely:

- **SRResNet** [4]
  This is a deep convolutional neural network structure that uses multiple residual blocks to create HR images from its LR input images.
- **EDSR** [5]
  This method is a improvement on the on SRResNet[4] which improves the original performance by a changed structure of the the residual blocks.
- **WDSR** [8]
  Yet another iteration of the process, this model improves EDSR [5] by allowing wider features within the residual blocks.

2.3 Encoding of images in weights

Another recent trend that shows promise in accurately presenting 3D objects is the representation of these objects within the weights of a network. Any

3D object as well as any image can be represented by a high order function which can be embedded in a neural network, so that the inputs to this network would be the coordinates of a pixel while the output is the RGB value of that pixel.

Following this general idea NeRF [6] was created. When this model renders an image it takes a relative camera position and angle as input. Using this position and angle it then casts, for every pixel, a ray through space and samples values along this ray, from the volumetric representation embodied by its neural network. These values are then combined to calculate the color value of the pixel corresponding to that ray.
The neural network of the NeRF method expects x, y and z- coordinates as well as two angles describing the viewing angle (only two angles are necessary as the third rotation axis is fixed) as input. Based on this input the network returns a RGBσ value that describes the color and density at that location. This entire process is differentiable which enables us to optimize the weights of the neural network via gradient descend, and thus learn the volumetric representation of a scene by only providing images and camera angles.

Chapter 3

Method

To test the performance of NeRF as a super-resolution method we have designed an experiment. We create LR and HR images from a synthetic scene. The LR images are used to train the NeRF model while the HR images are used to evaluate the quality of the NeRF output. Figure 3.1 displays the general experiment setup.

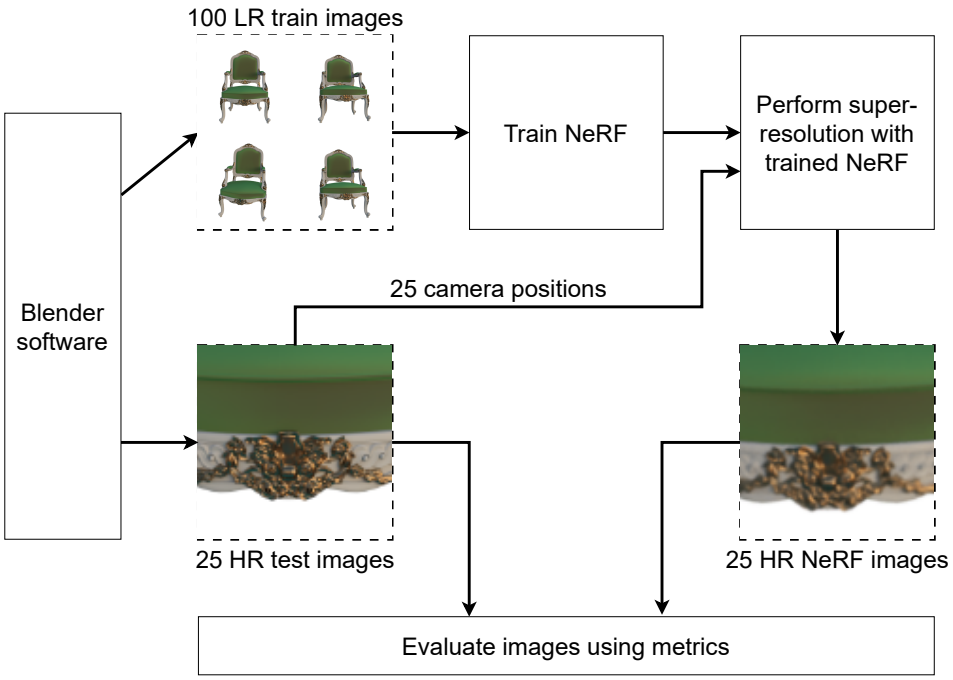

Figure 3.1: Experiment outline

3.1 Gathering Images

Evaluating a super-resolution task requires LR images as well as HR images of the exact same angle and scene. Since the NeRF model additionally needs multiple images from different angles of the same scenes, it proves quite effective to use synthetic scenes for the generation of these images, as results from rendering engines are reliably reproducible and consistent. For these reasons all the scenes that are used to evaluate the performance of NeRF as super-resolution method are synthetic and rendered using the Cycles render engine of the Blender software. We use 4 different scenes to evaluate our model that are displayed in Figure 3.2.

Figure 3.2: The four scenes used to test our method. The lego and chair scenes were taken from the original NeRF paper and the chalice and censer scenes are taken from sketchfab. For the artists names refer to the acknowledgements.

For each of our 4 scenes we create 100 LR images, with dimensions 400x400 pixels, that will be used to train the NeRF model, as well as 25 HR images, with dimensions 1600x1600 pixels, that are used as the ground truth when evaluating our results. All these images are taken from a slightly different camera angle which are evenly distributed over a certain area that is specified by the FOV angle that defines the maximum degree that a single camera angle can deviate from the central camera angle.
We additionally create 25 LR images from the same angles as the 25 HR images so that these 25 LR images can be used as input for the other super-resolution methods we compare our method with. By doing this we ensure that both, our method as well as the other methods, try to recreate the same 25 HR images that act as our ground truth.

Figure 3.3: Example of the LR training images that are taken from different angles. Show are training images of the Lego and Chair scene.

3.2 Train NeRF

For every single scene we train a new NeRF model by giving it the 100 LR images of the corresponding scene and train it for 200.000 iterations. This has to be done for every scene separately since the model is encoding the scene specific representation within its parameters. The value of 200.000 iterations was chosen based on the original NeRF paper [6], in which this amount is suggested for the trained model to perform well.

For the training of all scenes, we sample 64 points for every ray we cast through the scene. In every training step a batch of 1024 rays is processed which correspond to 1024 random pixels of one random LR training image. The network is optimized by using the Adam optimizer and is minimizing the mean squared error across the batched pixels.

Although adjusting some of these hyper parameters could improve the performance of NeRF, in regards to super-resolution, for the evaluation in this thesis the model was used as described in the original paper.

3.3 Super-resolution with NeRF

We can now use the trained NeRF models to perform our super-resolution task. To evaluate these models we will compare their output with the 25 HR ground truth images that were created as described in 3.1. We instruct

each, scene specific, NeRF model to create images from the same camera positions as the ground truth images.

To render an image of a embedded scene from a NeRF model, four parameters have to be provided. The height and width of the image that should be rendered, the focal length and the camera position. The focal length is set to be the same focal length as was used when rendering all the training images, while the camera position is set to the corresponding camera position of the 25 HR ground truth images. For the original task of NeRF, view-synthesis, the height and width would be set to the same values as the training data, namely 400x400. However since we want to perform super-resolution we set these values to 1600x1600 for all of the 25 images we render with the NeRF model per scene.

3.4 Evaluate images via metrics

To evaluate the performance of our method we then compare the output of the trained NeRF model with the 25 HR ground truth images that were rendered in 3.1. The way we compare them is using three different metrics that are commonly used to evaluate super-resolution tasks. The same methods where used in the original NeRF paper [6] as well.

- ***PSNR*** (Peak signal-to-noise ratio)
  This metric takes two images of the same dimensions and compares it using the mean squared error over all pixels. The mean squared error is then normalized by the maximum value a pixel can have (MAX in the formula) and then converted into a logarithmic scale.

$$\text{MSE} = \frac{1}{mn} \sum_{i=0}^{m-1} \sum_{j=0}^{n-1} (I(i,j) - K(i,j))^2$$

$$\text{PSNR} = 10 \cdot \log_{10} \left(\frac{MAX_I^2}{\text{MSE}} \right)$$

- ***SSIM*** [7] (Structural similarity)
  This metric compares two images using the averages, variances and covariance over the whole image. By doing this SSIM reflects correlations within the two images instead of simply the absolute error like the PSNR or MSE do. The values SSIM can have are between 0.0 and 1.0 where a 0.0 indicates no similarity at all and a 1.0 indicates that the

two images are identical.

$$\begin{aligned}
\text{SSIM(x,y)} &= \frac{(2\mu_x\mu_y + c_1)(2\sigma_{xy} + c_2)}{(\mu_x^2\mu_y^2 + c_1)(\sigma_x^2\sigma_y^2 + c_2)} \\
\mu_i &= \text{average of } i \\
\sigma_i^2 &= \text{variance of } i \\
\sigma_{ij} &= \text{covariance of } i \text{ and } j \\
c_i &= (k_i L)^2 \\
L &= \text{maximum value a pixel can have} \\
k_1 &= 0.01 \quad k_2 = 0.03
\end{aligned}$$

- ***LPIPS*** [9] (Learned Perceptual Image Patch Similarity)
  This metric compares two images by calculating their "perceptual similarity" which is doing a better job at representing human perception about the similarity of two images rather than methods like PSNR and SSIM. The way LPIPS does this is by harvesting the functionality of convolutional neural networks.

3.5 Super-resolution with other Methods

The methods we will be comparing with NeRF were already briefly introduced in 2 and are namely:

- Nearest-neighbor interpolation
- Bilinear interpolation
- Bicubic interpolation
- EDSR (state-of-the-art)
- WDSR (state-of-the-art

All these methods expect a single LR images as input to produce a single HR image as output. So to properly compare them to our NeRF model we provide each one of them with with the 25 LR images that we created in 3.1 as input. These images have the exact same camera positions as our 25 HR ground truth images so the resulting output are the best attempt of these methods to estimate the ground truth. Just like for our NeRF method we evaluate all of our results using the three metrics described in 3.4.

Chapter 4

Results

For all the results in this section it stands that the NeRF scenes where trained for 200k training iterations with a FOV angle of 20° unless otherwise specified.

4.1 General results

Comparing the HR images the NeRF model creates with the human eye it is quite apparent, that the images are not as clear as the images created by the state of the art models EDSR and WDSR. However no clear decision can be made, only using the human eye, regarding the differences between NeRF and interpolation based methods. Example outputs of this can be seen in Figure 4.1 (More output images can be seen in Appendix A).

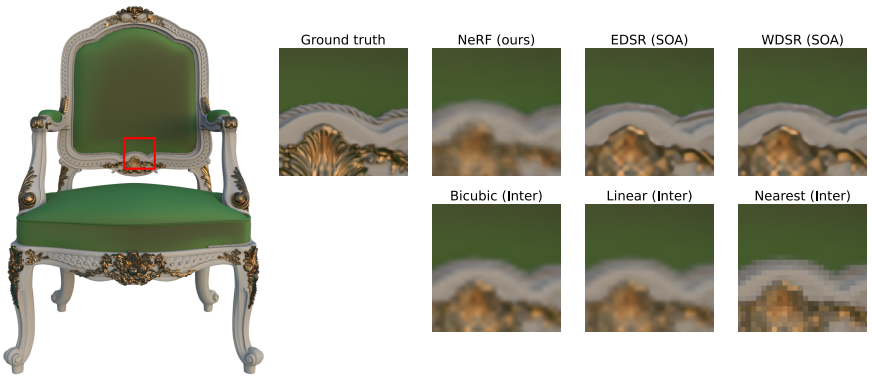

Figure 4.1: Output of all compared methods of the in red highlighted area of the 4x up-scaled Chair scene. Both "SOA" methods are neural network based state-of-the-art methods and all methods indicated by the "Inter" are conventional interpolation based methods.

The metric results that are displayed in Figure 4.2 indicate that the NeRF model performs worse than any other method in the PSNR and SSIM metric. NeRF seems to be roughly comparable to the interpolation based methods while still being considerably worse then the state of the art models in regards of the LPIPS metric.
In regards of statistical significance we find that all methods are performing significantly better in regards of PSNR and SSIM. In terms of the LPIPS metric we also see that NeRF is performing significantly worse than the state-of-the-art methods as well as the nearest neighbor interpolation method. NeRF is however performing significantly better than the linear and bicubic interpolation methods in regards of LPIPS. (This can be seen in Table 4.1)

More output images and more detailed values can be found in Appendix A and Appendix B.

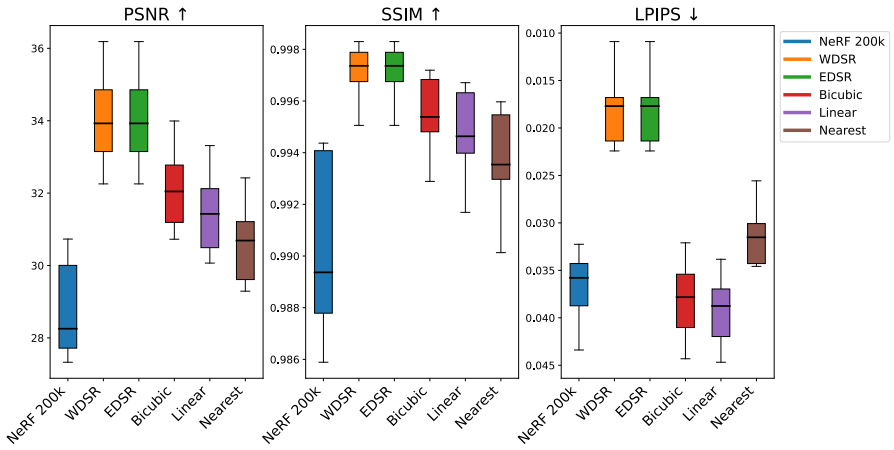

Figure 4.2: Box plot displaying the performance of all the different methods for each of the metrics. Every single box represents the averaged results from 8 different scenes. In this and all following box plots, the boxes represent the 1/4 and 3/4 quartiles and the median is represented by the orange line in the boxes. The whiskers span the entire range of all data points. Outliers are defined as being more or less then 3/2 times larger/smaller then the corresponding quartile. (Note that the y-axis of the LPIPS graph is inverted. In LPIPS lower scores indicate better performance and the inversion is done to keep the interpretation, that a value higher in the graph represents better performance, consistent.)

	Metric values			Significance vs NeRF		
	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR	SSIM	LPIPS
Nearest	30.62	0.994	0.032	$7.8 \cdot 10^{-3}$	$7.8 \cdot 10^{-3}$	0.015
Linear	31.45	0.995	0.039	$7.8 \cdot 10^{-3}$	$7.8 \cdot 10^{-3}$	$7.8 \cdot 10^{-3}$
Bi-cubic	32.10	0.996	0.038	$7.8 \cdot 10^{-3}$	$7.8 \cdot 10^{-3}$	0.015
EDSR	34.10	0.997	0.018	$7.8 \cdot 10^{-3}$	$7.8 \cdot 10^{-3}$	$7.8 \cdot 10^{-3}$
WDSR	34.10	0.997	0.018	$7.8 \cdot 10^{-3}$	$7.8 \cdot 10^{-3}$	$7.8 \cdot 10^{-3}$
NeRF	28.75	0.988	0.038	-	-	-

Table 4.1: This table contains the metric values that are averaged over all images of 8 scenes for all methods in addition to the p-values of a Wilcoxon signed-rank test that compared the results per scene of each method with the the results per scene of the NeRF method. Statistically significant p-values are highlighted in red. (Threshold for significance is 0.05)

4.2 Improvement through training

As described in 4.1 the NeRF model is not performing very well. Considering options to improve the performance of the NeRF model, the amount of training iterations comes quickly to mind.

All the NeRF models mentioned before were all trained for 200.000 training iterations. The original NeRF paper states this amount as a good "time to performance" ration but it is still the case that more training time improved performance of NeRF for its original task. Thus it is likely that the same would be the case for the task of super-resolution.

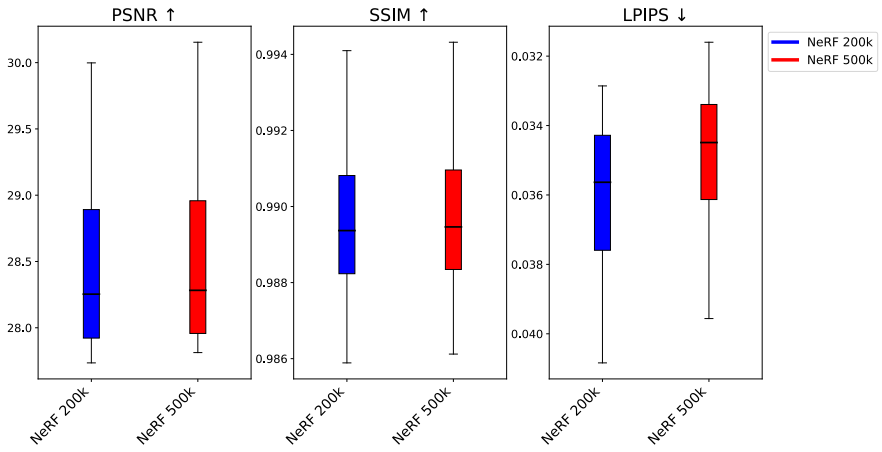

Figure 4.3: Plotted here are the averaged results from 4 different scenes given by a NeRF model trained for 200.000 iterations and one that was trained for 500.000 iterations (Note that LPIPS y-axis is inverted).

In Figure 4.3 we see that PSNR and SSIM only indicate a minimal improvement of performance while LPIPS shows a notable improvement. In regards of statistical significance we see that the 500.000 iterations model performs significantly better across all scenes in the LPIPS metric. For the PSNR and SSIM metrics the results show a significant difference in all but one scene. The values and statistical significance can be seen in Table 4.2.

The increase in training time however does not change anything regarding how NeRF compares to the other methods as depicted in Figure 4.4, NeRF is still last in PSNR and SSIM and around the same as the other interpolation based methods in LPIPS.

Scenes	Iterations	Metric values			Significance		
		PSNR ↑	SSIM ↑	LPIPS ↓	PSNR	SSIM	LPIPS
Lego	200k	27.74	0.986	0.033	$3.3 \cdot 10^{-4}$	$5.6 \cdot 10^{-4}$	$6.0 \cdot 10^{-8}$
	500k	27.81	0.986	0.032			
Chair	200k	28.52	0.990	0.035	$1.2 \cdot 10^{-3}$	$2.7 \cdot 10^{-5}$	$6.0 \cdot 10^{-8}$
	500k	28.56	0.990	0.034			
Chalice	200k	30.00	0.994	0.037	$6.0 \cdot 10^{-8}$	$6.0 \cdot 10^{-8}$	$6.0 \cdot 10^{-8}$
	500k	30.15	0.994	0.035			
Censer	200k	27.98	0.989	0.041	0.101	0.055	$6.0 \cdot 10^{-8}$
	500k	28.01	0.989	0.040			

Table 4.2: This table shows the over images averaged results per scene for both a NeRF model trained for 200.000 interations as well as one trained for 500.000 iterations. Additionally show are the p-values of Wilcoxon signed-rank tests that indicate whether there was a significant difference between the two models. Statistically significant p-values are highlighted in red. (Threshold for significance is 0.05)

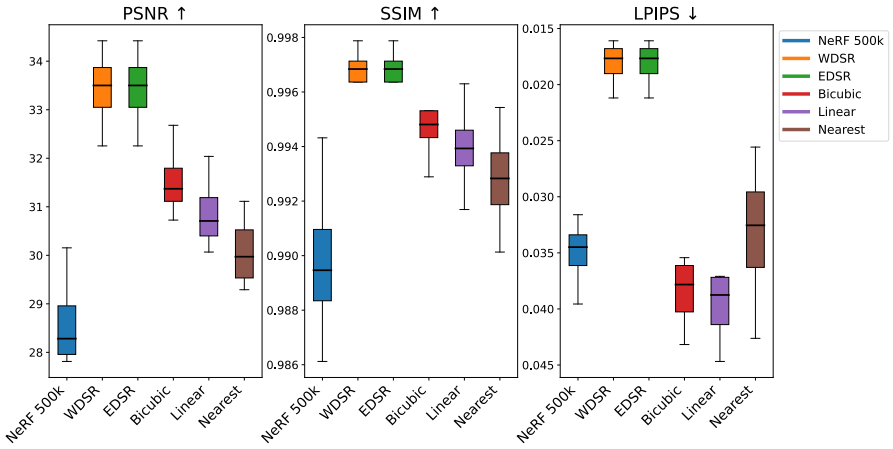

Figure 4.4: Performance of NeRF trained for 500.000 iterations vs other methods. All boxes represent the averaged results form 4 different scenes (Note that LPIPS y-axis is inverted).

4.3 Improvement through different FOV angle

Another factor that can be changed is the Field of View (FOV) angle which, as described in 3.1, defines how far apart the camera angles for the LR and HR images that are being created from the scene can be.

In all the results shown so far this FOV angle was 20°. Now the question is how the super-resolution performance is impacted should we decrease this angle while still providing the same number of training images. Presumably this could increase performance since all the image data will be closer together and thus more dense.

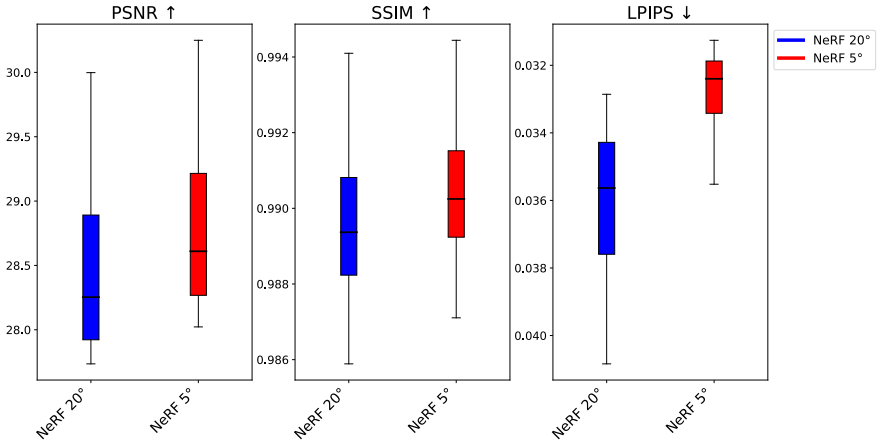

Figure 4.5: Plotted here are the averaged results from 4 different scenes given by a NeRF model trained with images that have a FOV angle of 20° and another NeRF model that was trained with images that have a FOV angle of 5° instead (Note that LPIPS y-axis is inverted).

We compared the performance of the NeRF model that was trained with a FOV angle of 20° with another NeRF model that was instead trained with a FOV angle of 5°. The results of this comparison can be seen in Figure 4.5. The 5° version of the NeRF model is performing notably better with a strong statistical significance in every of the three metrics. As can be seen in Table 4.3, all the results per scene have very low p-values and thus show a very significant difference in all metrics, with exception of the Lego scene, where the significance is not as strong.

Scenes	FOV	Metric values			Significance		
		PSNR ↑	SSIM ↑	LPIPS ↓	PSNR	SSIM	LPIPS
Lego	20°	27.74	0.986	0.033	$5.6 \cdot 10^{-3}$	0.012	0.011
	5°	28.02	0.987	0.031			
Chair	20°	28.52	0.990	0.035	$6.0 \cdot 10^{-8}$	$6.0 \cdot 10^{-8}$	$6.0 \cdot 10^{-8}$
	5°	28.87	0.991	0.032			
Chalice	20°	30.00	0.994	0.037	$6.0 \cdot 10^{-8}$	$6.0 \cdot 10^{-8}$	$6.0 \cdot 10^{-8}$
	5°	30.25	0.994	0.033			
Censer	20°	27.98	0.989	0.041	$6.0 \cdot 10^{-8}$	$6.0 \cdot 10^{-8}$	$6.0 \cdot 10^{-8}$
	5°	28.35	0.990	0.036			

Table 4.3: This table shows the over images averaged results per scene for both a NeRF model trained with a FOV angle of 20° as well as one trained with a FOV angle of 5°. Additionally show are the p-values of Wilcoxon signed-rank tests that indicate whether there was a significant difference between the two models. Statistically significant p-values are highlighted in red. (Threshold for significance is 0.05)

In Figure 4.6 we can also see that the NeRF model now always outperforms the bicubic and bilinear method in the LPIPS metric. Meaning that to human perception, the NeRF method with a FOV of 5° produces images that are closer to the ground truth. However in regards of PSNR and SSIM, although better than before, the NeRF method is still outperformed by all other methods.

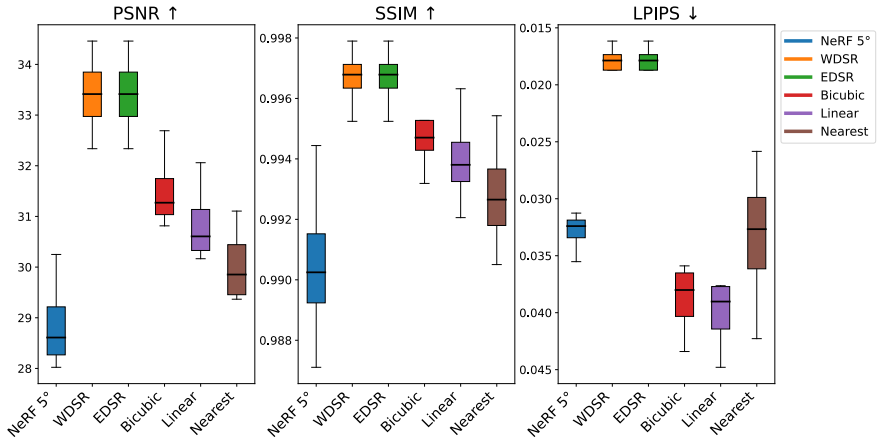

Figure 4.6: Performance of NeRF trained with a FOV of 5° vs other methods. All boxes represent the averaged results form 4 different scenes (Note that LPIPS y-axis is inverted).

Chapter 5

Discussion

We have found that the NeRF model can indeed be used as a super-resolution method. The performance of this method however is worse in the PSNR and SSIM metric than all other compared methods. The NeRF method also performs worse than the state-of-the-art methods in the LPIPS metric, while performing similar or better (given a small enough FOV angle), than the interpolation based methods, in this metric.

It is surprising to see that the NeRF model receives a worse PSNR and SSIM score than nearest neighbor interpolation since the trained NeRF model in essence is just a complex multi dimensional function which by definition is performing some kind of interpolation. Thus it is somewhat unexpected to see that nearest neighbor interpolation is outperforming NeRF.

5.1 Shortcomings of the experiment

A potential reason for this could be the fact that the 25 HR images that are being used to evaluate performance have mostly different camera angles than the 100 LR training images the NeRF was trained on (Only 4 of the 25 HR images align with angles of the 100 LR training images).
This means that the NeRF model is effectively performing two tasks for most of the 25 HR images. Firstly it is performing novel view-synthesis, the task NeRF was originally designed for, followed by the super-resolution task. Each task likely introduces its own amount of errors which could accumulate and could be the cause of why NeRF is outperformed in PSNR and SSIM by nearest neighbor interpolation.

We have performed an analysis on one of the scenes (Lego scene) whether there is a significant difference between the metric values of, by NeRF produced, HR images that are taken from camera positions that the NeRF model was trained on and, also by NeRF produced, HR images with camera positions that where not trained on. The results of this can be seen in Figure 5.1. Using a independent T-test, we could not find a significant difference in any of the metrics between training and testing images (p-values of 0.42 for PSNR, 0.48 for SSIM and 0.69 for LPIPS). Thus we have found no evidence that proved this shortcoming, however before being able to conclude that there is indeed no shortcoming, additional tests would have to be performed.

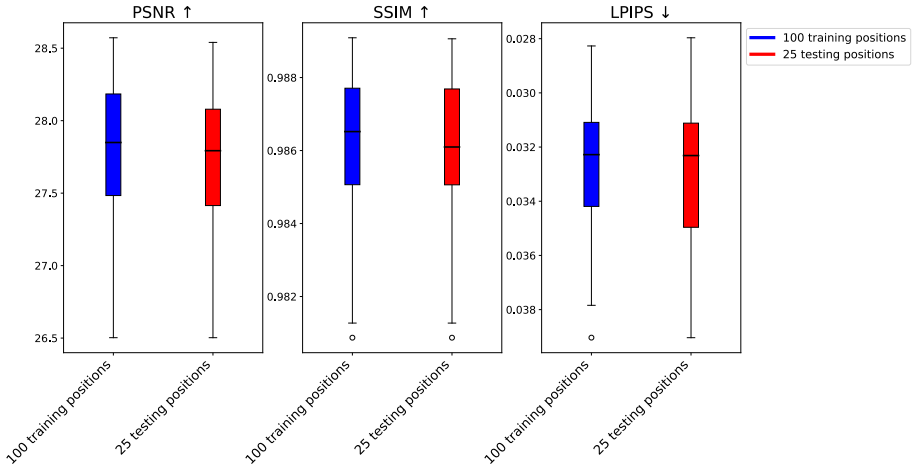

Figure 5.1: Box plot displaying the metric results of HR images created by NeRF method trained on the Lego scene. Shown are both the 100 camera positions that where used for training and the 25 camera positions that where used for testing (Note that LPIPS y-axis is inverted).

A solution to this potential shortcoming would be to repeat the experiment where the camera positions of the 25 HR images for each scene are a subset of the camera positions of the 100 LR training images.

5.2 Shortcomings of NeRF model

Another problem is that the internal representation of NeRF renders pixels based on single rays that it casts through 3D space, this shortcoming is described in detail and addressed in the Mip-NeRF paper [1]. The problem is that the pixel of an image does not represent a concrete ray of light but rather the average of rays that fall on this specific pixel. The NeRF model

can therefore not properly dissect the averaged information that a single pixel contains. This fact likely hinders NeRF from fully integrating the detail that multiple images from different angles provide. This is depicted in Figure 5.2.

A solution to this problem was already introduced in the Mip-NeRF paper [1] as also briefly explained in Figure 5.2. Therefore the only adjustment that has to be done to the experiment is to use the Mip-NeRF method as base instead of the original NeRF method.

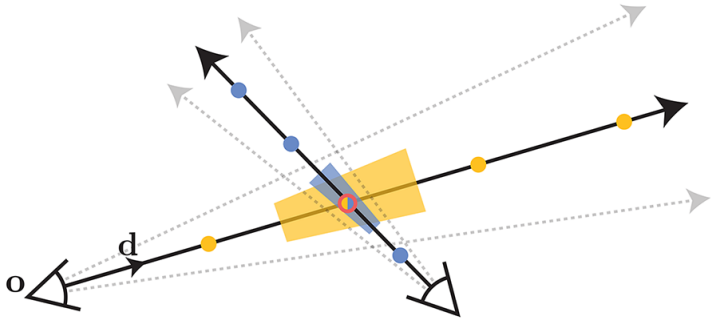

Figure 5.2: NeRF works by extracting point-sampled positional encoding features (shown here as dots) along each pixel's ray. Those point-sampled features ignore the shape and size of the volume viewed by each ray, so two different cameras imaging the same position at different scales may produce the same ambiguous point-sampled feature, thereby significantly degrading NeRF's performance. In contrast, Mip-NeRF casts cones instead of rays and explicitly models the volume of each sampled conical frustum (shown here as trapezoids), thus resolving this ambiguity. (This figure was taken from the Mip-NeRF paper [1])

5.3 Hyper-parameter optimization

Aside from fixing the shortcomings mentioned above, experimenting with different hyper-parameters for the NeRF model could also have a significant impact on performance. We already touched on changing the amount of training iterations and different FOV angles, however the experimentation on these was not exhaustive. There are also still a lot of parameters that we have not experimented with, for example:

- The Number of LR training images (100 were used in this thesis)
- The Super-resolution scale factor (4x was used in this thesis)

- The number of points sampled along a ray cast by the NeRF per pixel. (64 were used in this thesis)

Chapter 6

Conclusions

Although the NeRF model can be used as super-resolution method it is outperformed by conventional interpolation based and state-of-the-art methods in regards of PSNR and SSIM. Concerning LPIPS the NeRF method is capable of outperforming interpolation based methods given a narrow FOV angle. State-of-the-art methods however are still significantly better regarding the LPIPS metric regardless of any investigated FOV angle.
There are however multiple known potential shortcomings, with known possible solutions, to the NeRF model and the experiment that should first be fixed before a conclusive decision about the super-resolution performance of the NeRF model can be made.

Acknowledgements

I would like to thank Ruben van Bergen for the support he has given me as supervisor. Additionally I would like to thank the Blend Swap users Heintelnisse (lego) and 1DInc (chair) as well as the sketchfab users re1monsen (chalice) and cyberdan (censer) for the 3D models that where used to train and test the NeRF model.

Bibliography

[1] Jonathan T. Barron, Ben Mildenhall, Matthew Tancik, Peter Hedman, Ricardo Martin-Brualla, and Pratul P. Srinivasan, *Mip-nerf: A multi-scale representation for anti-aliasing neural radiance fields*, 2021.

[2] Cambridge in Colour, *Digital image interpolation.*

[3] Robert G. Keys, *Cubic convolution interpolation for digital image processing*, IEEE Transactions on Acoustics, Speech, and Signal Processing vol. 29, no. 6, pp. 1153–1160 (1981).

[4] Christian Ledig, Lucas Theis, Ferenc Huszár, Jose Caballero, Andrew Cunningham, Alejandro Acosta, Andrew Aitken, Alykhan Tejani, Johannes Totz, Zehan Wang, and Wenzhe Shi, *Photo-realistic single image super-resolution using a generative adversarial network*, 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017, pp. 105–114.

[5] Bee Lim, Sanghyun Son, Heewon Kim, Seungjun Nah, and Kyoung Mu Lee, *Enhanced deep residual networks for single image super-resolution*, The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops, July 2017.

[6] Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng, *Nerf: Representing scenes as neural radiance fields for view synthesis*, ECCV, 2020.

[7] Zhou Wang, A.C. Bovik, H.R. Sheikh, and E.P. Simoncelli, *Image quality assessment: from error visibility to structural similarity*, IEEE Transactions on Image Processing **13** (2004), no. 4, 600–612.

[8] Jiahui Yu, Yuchen Fan, Jianchao Yang, Ning Xu, Xinchao Wang, and Thomas S Huang, *Wide activation for efficient and accurate image super-resolution*, arXiv preprint arXiv:1808.08718 (2018).

[9] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang, *The unreasonable effectiveness of deep features as a perceptual metric*, CVPR, 2018.

Appendix A

More output images

The following figures display highlighted areas of the scenes for our the different scenes.

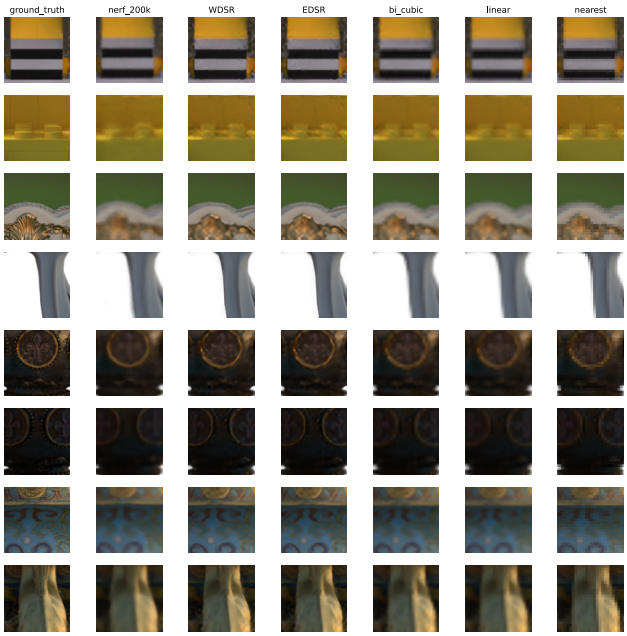

Figure A.1: Examples of output for all 8 scenes that were used to evaluate the NeRF model trained for 200.000 iterations with a FOV angle of 20°.

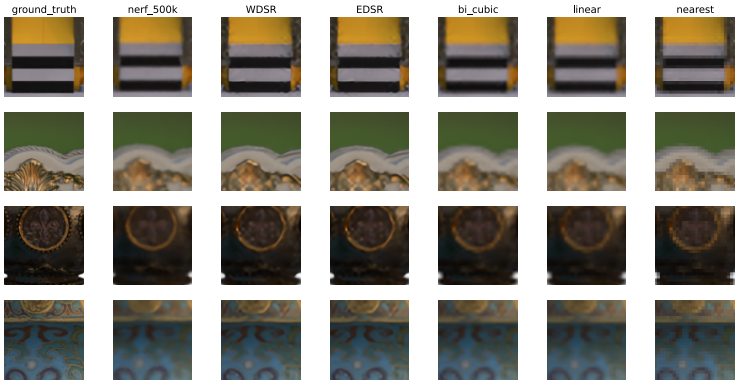

Figure A.2: Examples of output for all 4 scenes that were used to evaluate the NeRF model trained for 500.000 iterations with a FOV angle of 20°.

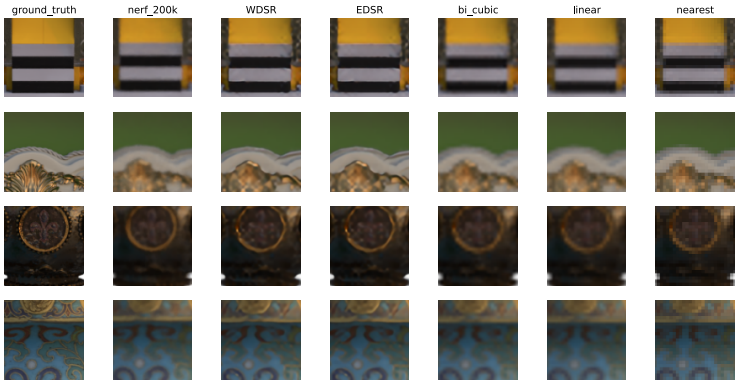

Figure A.3: Examples of output for all 4 scenes that were used to evaluate the NeRF model trained for 200.000 iterations with a FOV angle of 5°.

Appendix B

Detailed results

In the following tables the per scene results are documented each table contains the results for one or the three metrics used to evaluate the methods.

	front facing Scenes (20°)				back facing Scenes (20°)				front facing Scenes (5°)			
	Lego	Chair	Chalice	Censer	Lego	Chair	Chalice	Censer	Lego	Chair	Chalice	Censer
nearest	29.21	30.33	31.11	29.61	29.60	31.51	31.05	32.42	29.37	30.22	31.11	29.48
linear	30.011	30.91	32.04	30.51	30.44	32.38	31.95	33.31	30.17	30.83	32.06	30.38
bi-cubic	30.64	31.50	32.68	31.24	31.04	33.06	32.59	33.99	30.81	31.43	32.69	31.11
EDSR	31.90	33.69	34.42	33.32	32.65	36.19	34.17	36.15	32.34	33.64	34.46	33.18
WDSR	31.90	33.69	34.42	33.32	32.65	36.19	34.17	36.15	32.34	33.64	34.46	33.18
NeRF 200k	18.65	28.52	30.00	27.98	27.33	29.71	30.02	30.73	28.02	28.87	30.25	28.35
NeRF 500k	18.66	28.56	30.15	28.01	-	-	-	-	-	-	-	-

Table B.1: This table contains all results for the PSNR metric averaged over all images of the specific scene.

	front facing Scenes (20°)				back facing Scenes (20°)				front facing Scenes (5°)			
	Lego	Chair	Chalice	Censer	Lego	Chair	Chalice	Censer	Lego	Chair	Chalice	Censer
nearest	0.990	0.993	0.995	0.992	0.993	0.994	0.996	0.996	0.990	0.993	0.995	0.992
linear	0.992	0.994	0.996	0.994	0.994	0.995	0.996	0.997	0.992	0.994	0.996	0.994
bi-cubic	0.993	0.995	0.997	0.995	0.995	0.996	0.997	0.997	0.993	0.995	0.997	0.995
EDSR	0.995	0.997	0.998	0.997	0.997	0.998	0.998	0.998	0.995	0.997	0.998	0.997
WDSR	0.995	0.997	0.998	0.997	0.997	0.998	0.998	0.998	0.995	0.997	0.998	0.997
NeRF 200k	0.986	0.990	0.994	0.989	0.988	0.991	0.994	0.994	0.987	0.991	0.994	0.990
NeRF 500k	0.986	0.990	0.994	0.989	-	-	-	-	-	-	-	-

Table B.2: This table contains all results for the SSIM metric averaged over all images of the specific scene.

	front facing Scenes (20°)				back facing Scenes (20°)				front facing Scenes (5°)			
	Lego	Chair	Chalice	Censer	Lego	Chair	Chalice	Censer	Lego	Chair	Chalice	Censer
nearest	0.026	0.031	0.034	0.043	0.030	0.029	0.035	0.032	0.026	0.031	0.034	0.042
linear	0.037	0.037	0.040	0.045	0.044	0.034	0.041	0.037	0.038	0.038	0.040	0.045
bi-cubic	0.036	0.035	0.039	0.043	0.044	0.032	0.040	0.035	0.037	0.036	0.039	0.043
EDSR	0.017	0.016	0.021	0.018	0.022	0.011	0.022	0.017	0.018	0.016	0.021	0.018
WDSR	0.017	0.016	0.021	0.018	0.022	0.011	0.022	0.017	0.018	0.016	0.021	0.018
NeRF 200k	0.033	0.035	0.037	0.041	0.043	0.030	0.038	0.035	0.031	0.032	0.033	0.036
NeRF 500k	0.032	0.034	0.035	0.040	-	-	-	-	-	-	-	-

Table B.3: This table contains all results for the LPIPS metric averaged over all images of the specific scene.