Archives of Photogrammetry, Cartography and Remote Sensing, vol. 35, pp. 39-66
ISSN 2083-2214, eISSN 2391-9477
DOI: http://doi.org/10.14681/apcrs-2023-002

RESEARCH ARTICLE

APPLICATION OF NEURAL NETWORK MODELS IN THE MAPPING OF TELECOMMUNICATIONS INFRASTRUCTURE OBJECTS

ZASTOSOWANIE MODELI SIECI NEURONOWYCH W ODWZOROWANIU OBIEKTÓW INFRASTRUKTURY TELEKOMUNIKACYJNEJ

Bartosz Lewandowski, Łukasz Wilk, Paulina Zachar

Department of Photogrammetry, Remote Sensing and Spatial Information Systems,
Faculty of Geodesy and Cartography, Warsaw University of Technology

KEYWORDS: Nerf, neural networks, artificial intelligence, telecommunications masts, 3D models

ABSTRACT: Neural Radiance Fields (NeRF) is another 3D reconstruction method developed in recent years using artificial intelligence. This paper focuses on the study of object reconstruction using NeRF in the representation of objects such as telecommunication masts. Experiments were conducted using the Mega-NeRF model and two models (Nerfacto and Nerfacto-big) provided by the Nerfstudio framework on a UAV dataset. Various models and training parameters were tested, and the results were compared with reference data obtained from UAV photogrammetry and TLS laser scanning. The final analysis of the accuracy of the point clouds generated by the NeRF models indicated that they were of similar quality to the reference data, with slight differences in density and accuracy for different models and settings. The potential of NeRF methods for reconstructing 3D objects was demonstrated, especially in the context of mapping telecommunications masts, while noting the challenges associated with training parameters and the specifics of the analyzed object.

1. INTRODUCTION

Advances in artificial intelligence-based technologies have been significant in many fields in recent years. Photogrammetry, too, is supported by various modern solutions to develop 3D reconstruction methods. The first paper in which the Neural Radiance Field (NeRF) concept was presented is from 2020, in which it was shown how photorealistic views of scenes with complex geometry and appearance can be rendered using neural networks (Mildenhall et al., 2021). Since the authors proposed the concept, the technology has been continuously developed, and as a consequence, new models with better rendering accuracy are being created. The development also goes toward reducing the time needed to train the model and adapting the training process to larger datasets of images. These aspects are addressed in the work (Tancik et al., 2023), where the authors focused on the modularity of the method, allowing, for example, real-time visualisation or export of the resulting products. Such

solutions make it easy to modify and incorporate NeRF into their projects. The idea of mapping objects using NeRF is based on the basic assumptions presented in the first paper on the subject (Mildenhall et al., 2021). The creators of NeRF compare this method to volumetric rendering while noting the main difference between the two. Traditional rendering results in a model that is comprehensible and subject to optimization capabilities, however, it has a significant disadvantage in terms of the amount of disk space used (order of magnitude GB). The model generated by NeRF is a neural network model, due to which it is difficult to understand and optimize. However, its big advantage is that it takes up small amounts of data (orders of MB). Due to this fact, one can see the possibility of transferring and mapping scenes using NeRF in a much easier way without the need for huge memory resources and a high-speed Internet connection.

It is not uncommon for a trained model to take up less space than the images it was trained on, and it not only contains images of all the photos but also can predict what an object looks like from a different perspective that was not captured during the photographic process. To realise why this is the case, it is worth reviewing the basis of the network's learning. As input, it takes five parameters in the form of spatial location x, y, z and two angles that determine the direction of observation. The output of the model is colour in the form of r, g, b and density. According to Jon Barron, one of the authors of the publication Mildenhall et al. (2021), this can be compared to the operation of oculography (eye tracking). A ray comes out from a certain place and hits an obstacle, thus the network model learns what the world should look like from a given perspective. When the trained model is visualized, it tries to reproduce reality according to the learned rules. With this approach, it is possible not only to reproduce an object from a place from which it was not seen but also to create a depth map. This is presented in Figure 1.

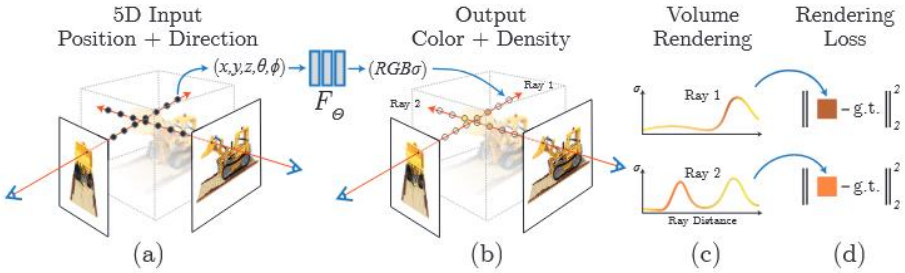

Figure 1 Diagram of the NeRF training process (Mildenhall *et al*., 2021).

It should also be mentioned that unlike the traditional application of machine learning, where the model should not be over-fitted to the input data in this case overfitting is desirable. When the fit is too poor the resulting view will be blurry. On the other hand, if the effect of overfitting the model is greater than desired, a graining effect will appear.

Initially, NeRF was developed to generate new views in 3D space. However, quite quickly, the solution began to be used to represent small objects in close proximity photography and create 3D models (Palestini et al., 2022; Murtiyoso & Grussenmeyer, 2023;

Martin-Brualla et al., 2021; Croce et al, 2023). An area in which NeRF is quite often used is the documentation of cultural heritage (Murtiyoso et al., 2023; Balloni et al., 2023; Mazzacca et al., 2023; Murtiyoso & Grussenmeyer, 2023). In view of such possibilities, it is not surprising that this type of space modelling method began to be compared with photogrammetric methods (Condorelli *et al*., 2021; Martiyoso *et al*., 2021, Vandenabeele *et al*., 2023). Moreover, with the development of new models, the possibility of training with a larger dataset for UAV (*unmanned aerial vehicle*) imagery has also been taken into account (Nex *et al*., 2023).

Furthermore, an advantage mentioned in the work Palestini *et al.* (2022), where the Instant Neural Graphics Primitives model (Müller *et al*, 2022) was used, is that it can be helpful when an object is exposed to strong sunlight and its surface is highly glossy. This paper points out that the technology is very young and the rate of development is exponential.

The apparent potential of NeRF-based methods was the motivation for conducting NeRF training experiments on a large UAV dataset using telecom mast modelling as an example. Due to the wide availability of different types of such neural networks, each of which is assumed to represent certain unique properties, it was decided to test several of them and verify their applicability to the case of telecom masts inventory.

2. ANALYSIS OF THE AVAILABLE SOLUTIONS

As mentioned, the study involved various NeRF models. However, due to the fairly large number of available solutions, the work initially consisted of analysing existing models. Both the purpose of a given model and the ease of implementation were taken into account.

The prototype on which subsequent versions are built is the aforementioned NeRF (Mildenhall *et al*., 2021). Compared to newer models, it takes longer to train and is less accurate. Instant Neural Graphics Primitives (Müller *et al*, 2022) is a model that stands out for its ability to learn networks in about 5 seconds and for its simple installation. It is adjusted to be optimized for NVIDIA graphics cards. This model needs a large amount of VRAM to train the model, its use depends on the size of the dataset. NeRF in the Wild (Martin-Brualla *et al*., 2021) is a NeRF-based model creation method using a collection of unstructured sets of disordered images. It provides the ability to learn a model from images under different conditions such as varying illumination or occlusion. It was created by Google Research and its code is not publicly available. Block-NeRF (Tancik *et al*., 2022) is a NeRF model that allows for the representation of a large-scale environment. The name comes from a method that divides the model into blocks so that the model can be divided into individually trained chunks. In the work, the training dataset consisted of 2.8 million images. Waymo and Google Research are responsible for the model; like NeRF in the Wild, its implementation is not open source. Mega-NeRF (Turki *et al*., 2022) is used to render urban-scale scenes using UAV imagery. It divides the model into several sub-modules, each of which can be trained separately, thus providing the opportunity to train a neural network in a distributed manner on the basis of thousands of images processed on multiple image cards. Different from the NeRF in the Wild and Block-NeRF models, which also provide the possibility to train on

a large dataset, this model is open source and can be tested for its own collection of images. Nerfstudio (Tancik *et al.*, 2023) to is a solution (framework) created using the Python language to help develop NeRF-type networks. It allows for building the basic components of the network and integrates with real-time visualisation in the browser, as well as with various export methods. In addition, Nerfstudio provides a custom network model, which does not come in a separate form and is modelled according to the MipNeRF-360 model.

Based on a review of the literature and available NeRF solutions, the Mega-NeRF model and two models available in Nerfstudio - Nerfacto and Nerfacto-big - were selected for testing. The models were trained with different parameter settings. The results of training the models are described. Nerfstudio allows exporting the reconstructed geometry to a point cloud, so it was possible to compare the results with reference data acquired using UAV photogrammetry and terrestrial laser scanning (TLS) techniques.

3. NERF WITH UAV DATA FOR TELECOM MASTS

3.1. DATA

The experiments were performed using a dataset of 393 UAV images of a telecommunications mast with a resolution of 5280x3956 pixels (an average of 4.34 MB per image). The images were taken with a Mavic 3 Enterprise drone equipped with a wide-angle 4/3 CMOS camera with a 20MP sensor 24mm focal length, and a built-in mechanical shutter.

3.2. PERFORMED EXPERIMENTS

The first step of the experiment was to test the possibility of using the Mega-NeRF model. Besides the possibility of training the network along with saving the model's parameters (training batches), it provides wide possibilities of use for visualization purposes as well as evaluation of results. However, these are not available seamlessly as soon as all libraries are installed. In the case of models such as NVIDIA Instant NeRFs or the Nerfstudio project, training and visualization were based on pointing to two directories, one with images and the other with parameters of extrinsic orientation. Mega-NeRF is more complex in this regard. It adopts the possibility of using its own data and information about the external camera's orientation elements for training processes, however, these must be files exported using COLMAP software processed by the script into a special format saved using PyTorch library files. Once such files are exported, the script responsible for partitioning the data and the associated masking of the images for training must be run. This approach is far different from analogous solutions due to the fact that the model can be divided into parts that will be trained at the same time on several different graphics cards. This idea is represented by a visualization of the developers' work (Figure 2).

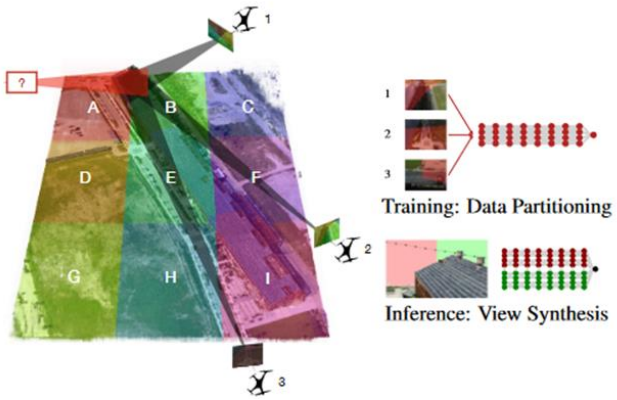

Figure 2 Mega-NeRF concept with splitting the scene into a set of spatial cells (left), learning a separate submodule for each. A discrete training dataset is created for the submodules, and as a result, each module is trained completely separately from its neighbouring cells (Turki *et al*., 2022).

The set of images was divided into training and validation, in the proportion of 372 to 21. Training of the Mega-NeRF model was carried out using the defined parameters (Table 1).

Table 1 Mega-NeRF model's basic training parameters.

num_chunks	200
batch_size	1024
image_pixel_batch_size	65536
train_iterations	500000
fine_samples	512
layers	8

On hardware equipped with an Intel Core i9-12900KF processor, 128 GB RAM clocked at 2400 MHz and a GeForce RTX 3080 Ti graphics card, the process lasted about 35 hours. The results of the model evaluation are presented in the following Table 2.

Table 2 Results of model evaluation on a dataset of UAV images for a telecommunications mast.

Average val/psnr	19.439
Average val/ssim	0.369
Average val/lpips/vgg	0.567
Average val/lpips/alex	0.573
Average val/lpips/squeeze	0.420

For comparison, a training dataset prepared by the developers of the model was carried out, for which the evaluation results are shown below (Table 3).

Table 3 Results of model evaluation on test data provided by Mega-NeRF developers.

Average val/psnr	19.503
Average val/ssim	0.452
Average val/lpips/vgg	0.615
Average val/lpips/alex	0.650
Average val/lpips/squeeze	0.519

The metrics seen in the tables above are often used when evaluating model quality and are well explained in the paper (Kniaz *et al*., 2023). PSNR (Peak Signal-to-Noise Ratio) is a quality indicator that tells about the influence of noise. The higher the PSNR ratio, the better the quality of the reconstruction. To assess the quality of NeRF models, it is worth considering the LPIPS (Learned Perceptual Image Patch Similarity) and SSIM (Structural Similarity Index Measure) indices, which focus on the accuracy of 3D scene representation. The SSIM index ranges from -1 to 1, where 1 indicates perfect similarity between the compared images. Higher SSIM values generally correspond to better image quality. The LPIPS metric calculates the distance between feature representations extracted from images. The lower the LPIPS value, the more perceptually similar the images are. The values of the metrics for the training of the test set and the set of UAV images for the mast were formed at similar levels. Visualization of the results is possible from the viewer and is as follows (Figure 3).

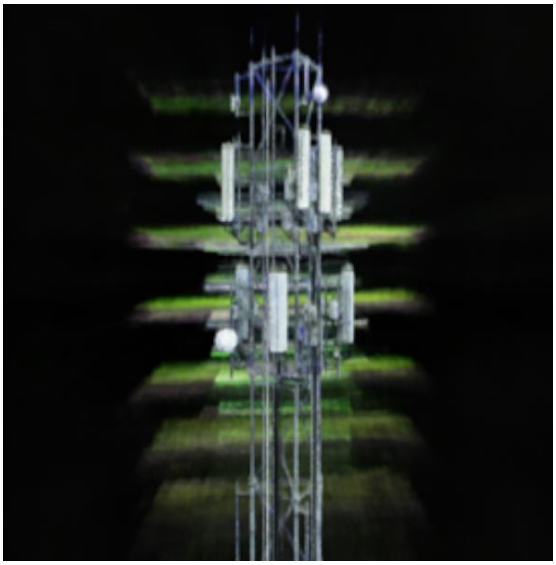

Figure 3 Visualization of the telecom mast - viewer view.

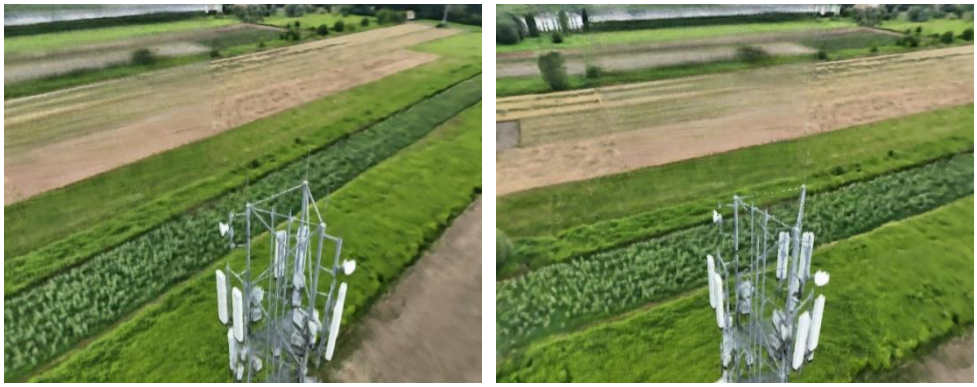

Figure 4 Example visualizations of rendered views from the model.

Figure 5 Comparison of the rendered model (left) with the UAV image (right).

In the previously presented version of the experiment, the test area was not divided into several submodels. Since this is one of the characteristics of the Mega-NeRF model this was checked in the next experiment. It should be noted that the dataset provided and tested by the model developers is significantly different from our case. In the original idea, the images were taken at a fixed altitude, and the horizontal position changed. The characteristics of a drone flight around a tall, slender object such as a telecommunications mast are significantly different from a typical flight over a city, for example, or ground photos taken for a close-range object. Considering these aspects, it could not be guaranteed that the default approach would improve the quality of the trained model, but the attempt was made. The

dataset was divided into 4 parts with the parameter --grid_dim 2 2. This resulted in the need to train 4 submodels, each of which took about 35 hours.

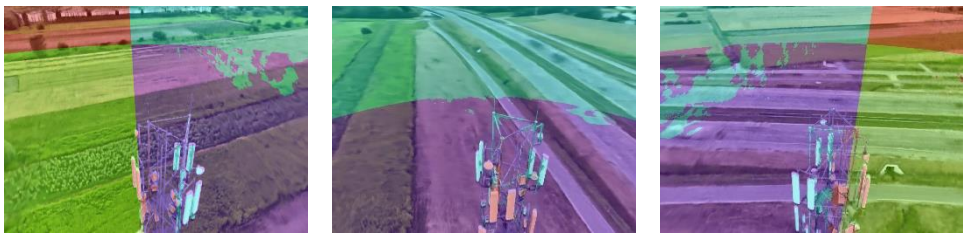

Figure 6 Visualisation of the division of the model into sub-modules for training.

The above visualizations show the result of a script tasked with dividing one large model into submodules (Figure 6). The training of each sub-module is performed taking into account the geometry information of the pixels (Tancik *et al*., 2023).

Figure 7 Comparison of an image exported from a Mega-NeRF training model without sub-model splitting (right) with one split into 4 (left).

Analysing the results visually, after having a look at the details, one can see that the image exported as a result of training the model with sub-models is more detailed (Figure 7). Unfortunately, it turns out that this method works when the visible image is trained with several perspectives corresponding to submodels. When the horizon was no longer visible, and thus 2 of the 4 submodels for processing were not available, the quality of the exported images dropped significantly, as can be seen below (Figure 8).

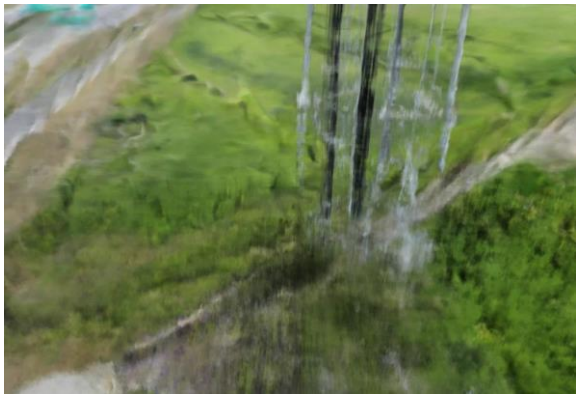

Figure 8 Issues of Mega-NeRF model trained with sub-models.

The visual assessment and the problems noted were also confirmed by the values of the metrics used to evaluate the model (Table 4). The value of the PSNR index dropped significantly relative to the first iteration of the model. Similarly, the value of the SSIM index decreased almost twice.

Table 4 Mega-NeRF model evaluation results on a dataset of UAV images for a telecommunications mast divided into four submodels.

Average val/psnr	15.096
Average val/ssim	0.247
Average val/lpips/vgg	0.678
Average val/lpips/alex	0.723
Average val/lpips/squeeze	0.510

Another run of training the model was carried out for an increased number of iterations to 800,000 using a single slice split. The results are shown in the table below (
Table 5).

Table 5 Mega-NeRF model evaluation results on a dataset of UAV images for a telecommunications mast for an increased number of iterations without sub-models.

Average val/psnr	19.547
Average val/ssim	0.381
Average val/lpips/vgg	0.552
Average val/lpips/alex	0.545
Average val/lpips/squeeze	0.395

The noticeable increase in accuracy encouraged another iteration of model training. In order to test the effect of increasing the number of epochs on the accuracy of the model, a test was conducted on 900,000 epochs. For such data, the process took about 65 hours, rendering 372 views corresponding to the camera position when the image was taken took 15 hours. An example view corresponding to the orientation elements from the first image is shown below (Figure 9).

Figure 9 Example visualization of a rendered view from a Mega-NeRF model trained with 900,000 epochs set without sub-models.

Table 6 Evaluation results of Mega-NeRF model trained with 900,000 epochs set without sub-models.

Average val/psnr	19.353
Average val/ssim	0.362
Average val/lpips/vgg	0.566
Average val/lpips/alex	0.570
Average val/lpips/squeeze	0.412

Analysing the results, it can be said that for this type of data, dividing the dataset into submodels decreases the accuracy. On the other hand, increasing the number of epochs had a positive effect on the results. Taking into account the conducted iterations of training the Mega-NeRF model, one could see the potential of using such a solution to create a 3D model for an object such as a telecommunications mast using UAV imagery. The problems seen during the evaluation of the results were motivation to test other models as well.

The next phase of the research was to test the capabilities of Nerfstudio for the same dataset. This framework was used because it is designed to create a user-friendly experience for using NeRF models. It offers its own model as well as implementations of others. It also provides the ability to easily visualise the trained model and export it to a point cloud in *.pcd and *.ply format. Thus, the generated point cloud that results from training the model can be

compared with a reference point cloud. In contrast to Instant Neural Graphics Primitives, it does not store all loaded images in the memory of the graphics card. Therefore, thanks to this, there is no problem related to the amount of VRAM. However, the problem arises when loading the model into RAM, because the implemented method takes all the images, converts them into PyTorch library-compatible objects and stores them in the cache. As a result of this approach, more than 128 GB of temporary memory is required to train the model on 393 images with a total size of 1.5 GB. A workaround for this problem that was used for the study was to assign an appropriately sized SWAP exchange partition. An increase of 200 GB allowed the model to be trained on the full set of images without having to reduce their resolution.

Nerfstudio provides the ability to train data on various models such as Instant-NGP, Instruct-NeRF2NeRF, K-Planes, LERF, Mip-NeRF, NeRF, Nerfacto, Nerfbusters, NeRFPlayer, Tetra-NeRF, TensoRF, Generfacto. The default model developed by the developers of Nerfstudio is the Nerfacto model. It is a combination of many different published models. It comes in several variants: Nerfacto, Nerfacto-big, Nerfacto-huge, Depth-nerfacto. For the purposes of the study, the Nerfacto and Nerfacto-big models were tested. The Nerfacto-big model is slower when it comes to training the model, but by design, it should give better results.

Visually comparing the results from Nerfacto (Figure 10) and Nerfacto-big (Figure 11) with the previously trained Mega-NeRF model, there are no big differences in the representation of telecommunications mast elements. However, the values of model evaluation metrics in both cases are higher than for Mega-NeRF (Table 7). Thus, it can be said that the Nerfacto model is more suitable for this type of object.

Figure 10 An example view from the trained nerfacto model.

Figure 11 An example view from the trained nerfacto-big model.

Table 7 Evaluation results for the Nerfacto and Nerfacto-big model.

	Nerfacto	Nerfacto-big
Average val/psnr	19.611	19.792
Average val/psnr_std	1.227	1.238
Average val/ssim	0.404	0.421
Average val/ssim_std	0.080	0.078
Average val/lpips	0.757	0.688
Average val/lpips_std	0.073	0.082

4. ANALYSIS OF THE ACCURACY OF THE NERF POINT

Accuracy analysis was carried out with reference data. UAV images were acquired for the mast, which were used to train the models and generate the point cloud. The same data set was used to create a point cloud from dense image matching as a product of photogrammetric processing. In addition, a point cloud was acquired from a Leica RTC360 ground-based laser scanner. The point clouds exported from Nerfstudio were compared in CloudCompare. The distances between the reference data and the point cloud from Nerfstudio were calculated. The results are shown below (Figure 12).

Figure 12 Comparison of the point cloud from Nerfstudio (Nerfacto model) with the TLS (Terrestrial Laser Scanning) point cloud and DIM (dense image matching) for the mast.

The average difference in distance between the resulting point cloud from the Nerfacto model and the point cloud from dense image matching was 1.3cm, and the standard deviation formed 1.6cm (Figure 13). However, it can be seen that the biggest differences were in the areas where there are gaps in the point cloud from image matching, which also has its disadvantages (Figure 14).

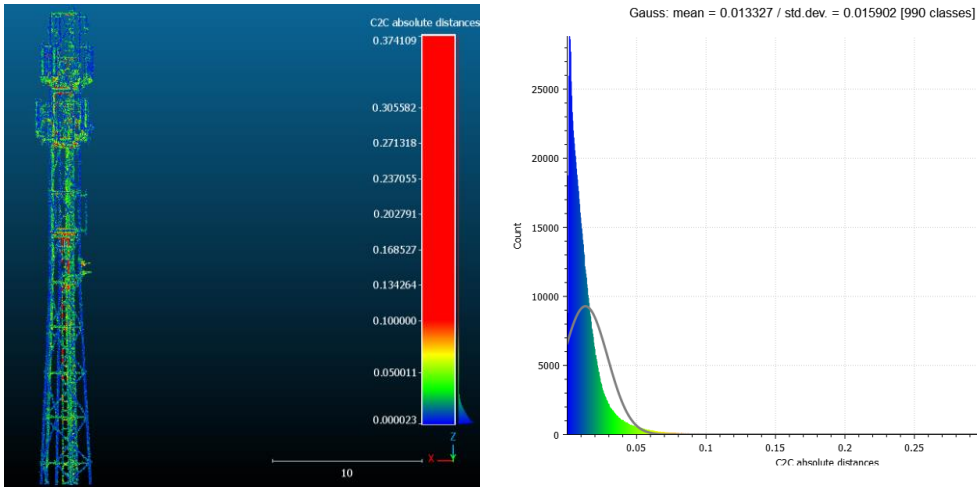

Figure 13 Differences between the point cloud from Nerfstudio (Nerfacto model) and the DIM (dense image matching) point cloud and the histogram with distribution.

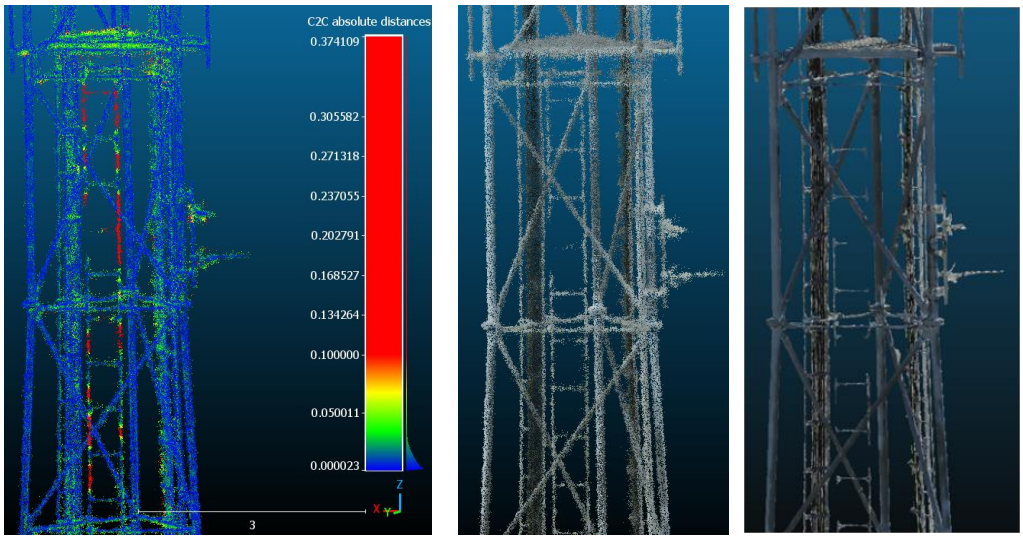

Figure 14 Differences (left) between the point cloud from Nerfstudio (Nerfacto model) (middle) and the DIM (dense image matching) point cloud (right).

Much better reference data for telecommunications masts are point clouds from terrestrial laser scanning, as they have far fewer "blind spots" and do not have as many data

completeness issues. Thus, a point cloud from Nerfacto was compared with a TLS. The results obtained as far as numerical values were on a similar level. The average distance between the point clouds 1.6 cm with a standard deviation of 1.5 cm (Figure 15). Moreover, in both examples of comparison, the highest values of differences did not exceed 40 cm.

When comparing relative to the TLS point cloud, it is apparent that the biggest differences were not for the not fully mapped mast ladder, but for the area of sector antennas (Figure 16).

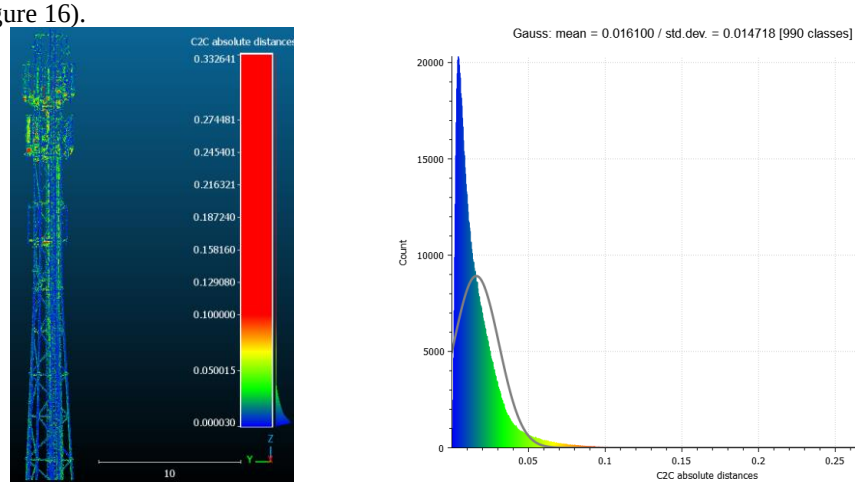

Figure 15 Differences between point cloud from Nerfstudio (Nerfacto model) and TLS (Terrestrial Laser Scanning) point cloud and histogram with distribution.

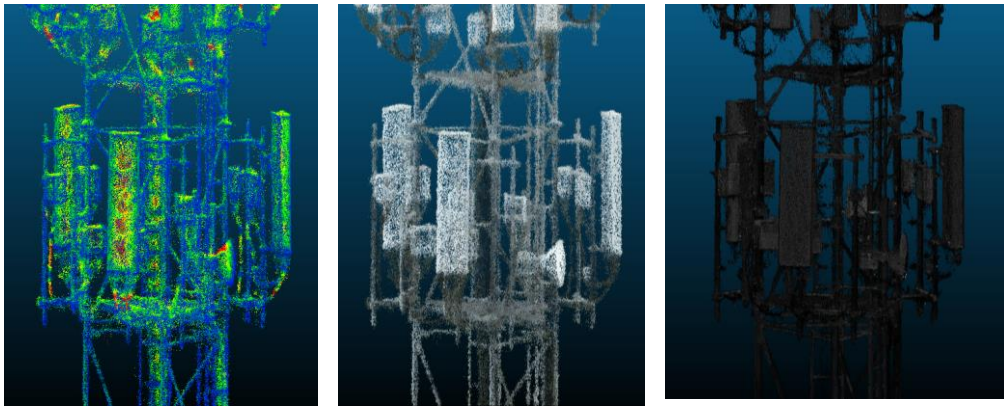

Figure 16 Differences (left) between the point cloud from Nerfstudio (Nerfacto model) (middle) and the TLS (Terrestrial Laser Scanning) point cloud (right).

Figure 17 Comparison of the point cloud from Nerfstudio (Nerfacto-big model) with the TLS (Terrestrial Laser Scanning) point cloud and DIM (dense image matching) for the mast.

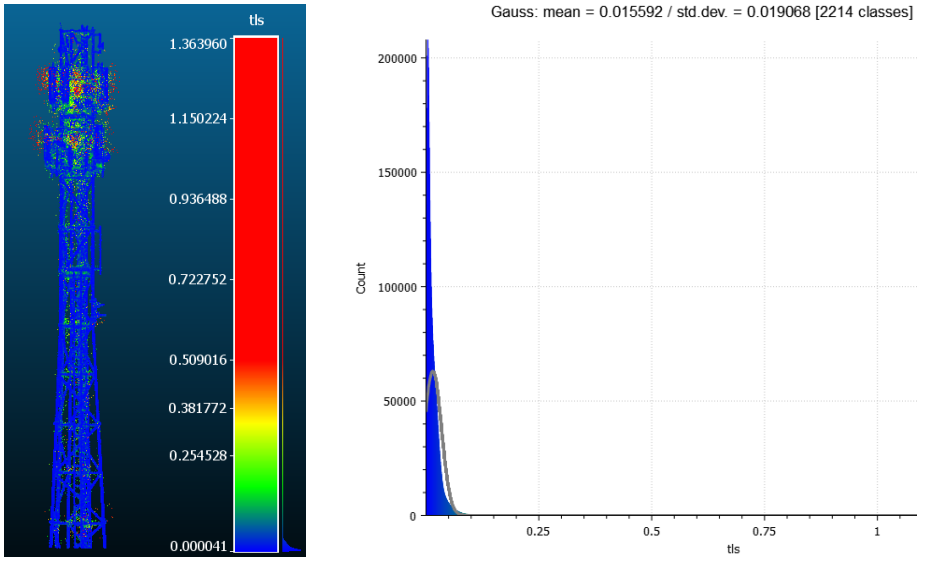

Figure 18 Differences between the point cloud from Nerfstudio (Nerfacto-big model) and the TLS (Terrestrial Laser Scanning) point cloud and the histogram with distribution.

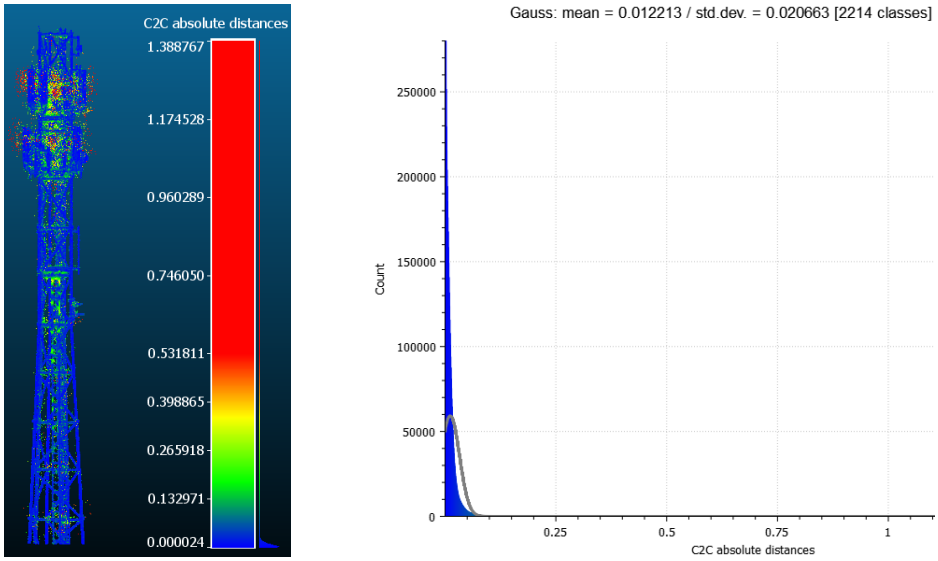

Figure 19 Differences between the point cloud from Nerfstudio (Nerfacto-big model) and the DIM (dense image matching) point cloud and the histogram with distribution.

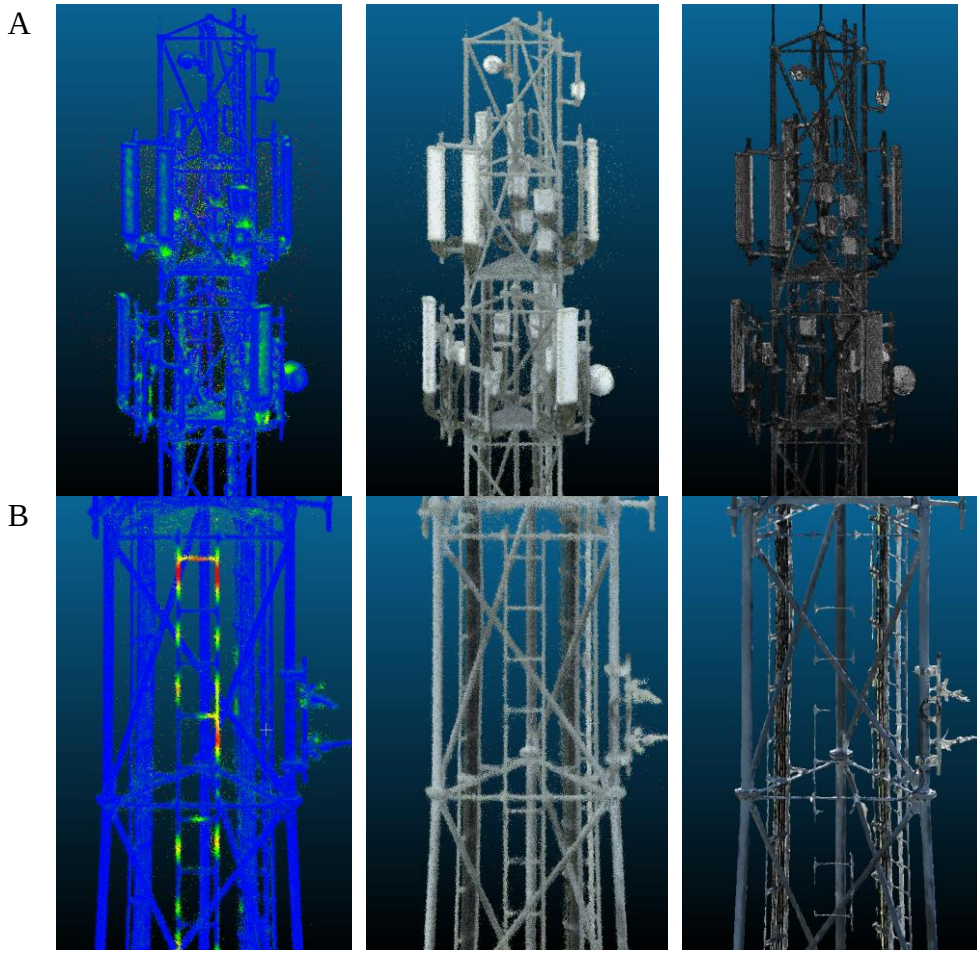

Figure 20 Differences (left) between point cloud from Nerfstudio (Nerfacto-big model) (middle) and TLS (Terrestrial Laser Scanning) point cloud (right) - A) comparison with TLS (Terrestrial Laser Scanning), B) comparison with DIM (dense image matching).

The average distance difference between the resulting point cloud from the Nerfacto-big model and the point cloud from the dense image matching was 1.2cm, and the standard deviation was 2.1cm (Figure 19). So, it can be said that the results are very similar to the Nerfacto model, which in theory should be less accurate. As can be seen in the figures above (Figure 17, Figure 20), the point cloud from the Nerfacto-big model is noisier than the cloud from the Nerfacto model. However, the density of this cloud is much higher. It is likely that the use of methods that would remove noise from the resulting cloud would improve the statistics. The average distance between the Nerfacto-big point cloud and the TLS point cloud

was 1.6 cm with a standard deviation of 1.9 cm (Figure 18). As with the comparison with point clouds from DIM (dense image matching), it can be noticed that the results are not very different for the two models.

5. CONCLUSIONS

Among NeRF neural network models, there are two basic types regarding what kind of object is to be imaged. There are 360-degree models that assume the camera will move around the object in question, and spatial models. The assumption of MegaNerf is to map a large area using images taken from drones. In our case, the mapping area is not large in the sense of the model creators. The maximum differences in the x and y coordinates of the camera differ by several meters. Taking this into account, it is not surprising that the model is not suitable for precise visualization of relatively small elements. Another drawback of this model in photogrammetric applications is the inability to export the model in either point cloud or mesh format, the only export that is supported in the original is to an octree file that can be read by a dedicated viewer. These problems are solved by Nerfstudio which gives the ability to load different models while providing visualization of the learning effects for each one. Loading the data is a limitation that becomes apparent when there is a need to use a larger dataset for training. Expanding the temporary memory with a hard drive solves the problem for moderately large data however, in extreme cases this approach may also not be sufficient. To prevent such cases, it would be necessary to change the way data is loaded into the model. A direct comparison of accuracy measures for the same images shows that Nerfstudio performs better than Mega-NeRF in the test case.

Summarizing, NeRF-based models, such as Mega-NeRF, Nerfacto and Nerfacto-big, show potential for reconstructing objects using drone data. However, there are many factors, such as model limitations, quality and quantity of training data, training parameters, etc., that can affect the quality and accuracy of reconstruction. Therefore, it is necessary to thoroughly understand the limitations of these models and adapt them to a specific use case before applying them in practice.

ACKNOWLEDGMENTS

The research was carried out as part of the project "MAST - Digital twin of mast objects as an innovative inventory service using unmanned aerial vehicles and artificial intelligence", financed by the National Centre for Research and Development, grant no POIR.01.01.01-00-0426/22.

LITERATURE

Balloni, E., Gorgoglione, L., Paolanti, M., Mancini, A., & Pierdicca, R. (2023). Few shot photogrammetry: a comparison between NeRF and MVS-SfM for the documentation of cultural heritage. *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences, 48*, 155-162.

Condorelli, F., Rinaudo, F., Salvadore, F., & Tagliaventi, S. (2021). A comparison between 3D reconstruction using nerf neural networks and MVS algorithms on cultural heritage images. *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, *43*, 565-570.

Croce, V., Caroti, G., De Luca, L., Piemonte, A., & Véron, P. (2023). Neural radiance fields (NeRF): review and potential applications to digital cultural heritage. *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, *48*, 453-460.

Kniaz, V. V., Knyaz, V. A., Bordodymov, A., Moshkantsev, P., Novikov, D., & Barylnik, S. (2023). Double Nerf: Representing Dynamic Scenes as Neural Radiance Fields. *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, *48*, 115-120.

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

Mazzacca, G., Karami, A., Rigon, S., Farella, E. M., Trybala, P., & Remondino, F. (2023). NeRF for heritage 3D reconstruction. *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, *48*, 1051-1058.

Mildenhall, B., Srinivasan, P. P., Tancik, M., Barron, J. T., Ramamoorthi, R., & Ng, R. (2021). Nerf: Representing scenes as neural radiance fields for view synthesis. *Communications of the ACM*, 65(1), 99-106.

Murtiyoso, A., & Grussenmeyer, P. (2023). Initial assessment on the use of state-of-the-art NeRF neural network 3d reconstruction for heritage documentation. *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, *48*, 1113-1118.

Murtiyoso, A., Markiewicz, J., Karwel, A. K., & Kot, P. (2023). Investigation on the Use of NeRF for Heritage 3D Dense Reconstruction for Interior Spaces. International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences-ISPRS Archives, 48(1/W3-2023), 115-121.

Müller, T., Evans, A., Schied, C., & Keller, A. (2022). Instant neural graphics primitives with a multiresolution hash encoding. *ACM Transactions on Graphics*, 41(4), 1-15.

Nex, F., Zhang, N., Remondino, F., Farella, E. M., Qin, R., & Zhang, C. (2023). Benchmarking the extraction of 3D geometry from UAV images with deep learning methods. *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, *48*, 123-130.

Palestini, C., Basso, A., & Perticarini, M. (2022). Machine Learning as AN Alternative to 3D Photomodeling Employed in Architectural Survey and Automatic Design Modelling. *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, 48, 191-197.

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

Tancik, M., Weber, E., Ng, E., Li, R., Yi, B., Wang, T., ... & Kanazawa, A. (2023). Nerfstudio: A modular framework for neural radiance field development. In ACM SIGGRAPH 2023 Conference Proceedings (pp. 1-12).

Turki, H., Ramanan, D., & Satyanarayanan, M. (2022). Mega-nerf: Scalable construction of large-scale nerfs for virtual fly-throughs. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 12922-12931.

Vandenabeele, L., Häcki, M., & Pfister, M. (2023). Crowd-sourced surveying for building archaeology: the potential of structure from motion (SfM) and neural radiance fields (NERF). *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, *48*, 1599-1605.

ZASTOSOWANIE MODELI SIECI NEURONOWYCH W ODWZOROWANIU OBIEKTÓW INFRASTRUKTURY TELEKOMUNIKACYJNEJ

SŁOWA KLUCZOWE: NeRF, sieci neuronowe, sztuczna inteligencja, maszty telekomunikacyjne, modele 3D

STRESZCZENIE: Neural Radiance Fields (NeRF) to kolejna rozwijana w ostatnich latach metoda rekonstrukcji 3D wykorzystująca sztuczną inteligencję. W artykule skupiono się na badaniach odwzorowania obiektów za pomocą NeRF w reprezentacji obiektów takich jak maszty telekomunikacyjne. Przeprowadzono eksperymenty z wykorzystaniem modelu Mega-NeRF oraz dwóch modeli (Nerfacto i Nerfacto-big) udostępnionych przez Nerfstudio na zbiorze danych UAV. Przetestowano różne modele i parametry treningowe, a wyniki były porównywane z danymi referencyjnymi pozyskanymi z fotogrametrii UAV oraz skaningu laserowego TLS. Ostateczna analiza dokładności chmur punktów wygenerowanych przez modele NeRF wskazała na ich zbliżoną jakość do danych referencyjnych, z niewielkimi różnicami gęstości i dokładności dla różnych modeli i ustawień. Wykazano potencjał metod NeRF do rekonstrukcji obiektów 3D, zwłaszcza w kontekście odwzorowania masztów telekomunikacyjnych, jednocześnie zauważając wyzwania związane z parametrami treningowymi i specyfiką analizowanego obiektu.

1. WPROWADZENIE

Postęp technologii opartych na sztucznej inteligencji jest znaczący w ostatnich latach w wielu dziedzinach. Fotogrametria również jest wspierana przez różne nowoczesne rozwiązania w celu rozwoju metod rekonstrukcji 3D. Pierwszą pracą, w której została przedstawiona koncepcja NeRF (Neural Radiance Field) jest praca z 2020 roku, w której został przedstawiony sposób jak za pomocą sieci neuronowych można renderować fotorealistyczne widoki scen o skomplikowanej geometrii i wyglądzie (Mildenhall *et al.*, 2021). Od momentu zaproponowania koncepcji przez autorów technologia ta jest stale rozwijana, w konsekwencji czego powstają nowe modele cechujące się lepszą dokładnością odwzorowania. Rozwój idzie również w kierunku redukcji czasu potrzebnego do wytrenowania modelu oraz dostosowania procesu uczenia do większych zbiorów zdjęć. Aspekty te poruszone zostały w pracy (Tancik *et al.*, 2023), gdzie autorzy skupili się na modułowości metody, umożliwiając na przykład wizualizację w czasie rzeczywistym czy eksport powstałych produktów. Takie rozwiązania pozwalają w łatwy sposób modyfikować i włączać NeRF do swoich projektów.

Idea odwzorowania obiektów za pomocą NeRF opiera się na podstawowych założeniach przedstawionych w pierwszej pracy na ten temat (Mildenhall *et al.*, 2021). Twórcy NeRF porównują tę metodę do renderowania wolumetrycznego, zwracając jednocześnie uwagę na główną różnicę między nimi. W wyniku tradycyjnego renderowania powstaje model, który jest zrozumiały oraz podlega on możliwości optymalizacji, jednakże posiada znaczącą wadę w postaci ilości zajmowanego miejsca na dysku (rząd wielkości GB). Model generowany przez NeRF jest modelem sieci neuronowej, z uwagi na co jest on trudny do zrozumienia oraz optymalizacji. Jednak dużą zaletą tego rozwiązania jest to, że zajmuje

małe ilości danych (rzędy MB). Z uwagi na ten fakt widać możliwość przy przesyłaniu i odwzorowaniu scen za pomocą NeRF w dużo łatwiejszy sposób bez konieczności posiadania ogromnych zasobów pamięci oraz wysokiej przepustowości połączenia internetowego. Nierzadko wyuczony model zajmuje mniej miejsca niż zdjęcia, na podstawie których został wytrenowany, a zawiera nie tylko obrazy wszystkich zdjęć, ale również potrafi przewidywać, jak wygląda dany przedmiot z innej perspektywy, która nie została uchwycona podczas procesu fotografowania. Żeby uświadomić sobie, dlaczego tak się dzieje warto zapoznać się z podstawą nauki sieci. Na wejściu pobiera ona pięć parametrów w postaci lokalizacji przestrzennej x, y, z oraz dwóch kątów określających kierunek obserwacji. Wyjściem modelu jest kolor w postaci r, g, b oraz gęstość. Według Jona Barrona jednego z autorów publikacji Mildenhall *et al*., (2021) można porównać to do działania okulografii (*ang. eye tracking*). Promień wychodzi z pewnego miejsca i trafia na przeszkodę, w ten sposób model sieci uczy się jak powinien wyglądać świat z danej perspektywy. W przypadku wizualizacji wytrenowanego modelu stara się odtworzyć rzeczywistość zgodnie z wyuczonymi zasadami. Dzięki takiemu podejściu możliwe jest nie tylko odwzorowanie obiektu z miejsca, z którego nie był widziany, ale również stworzyć mapę głębi. Można to zaprezentować w następujący sposób (Rysunek 1Figure 1).

Należy również wspomnieć o tym, że w przeciwieństwie do tradycyjnego zastosowania uczenia maszynowego, gdzie nie powinno się zbyt mocno dopasowywać modelu do danych wejściowych w tym wypadku przeuczenie (*ang. overfitting*) jest pożądane. Kiedy dopasowanie będzie zbyt słabe wynikowy widok będzie nieostry. Natomiast jeśli efekt nadmiernego dopasowania modelu będzie większy niż pożądany pojawi się efekt ziarnistości.

Pierwotnie NeRF został opracowany do generowania nowych widoków w przestrzeni 3D. Jednak dość szybko rozwiązanie zaczęto wykorzystywać do reprezentacji małych obiektów w bliskiej odległości fotografowania i tworzenia modeli 3D (Palestini *et al*., 2022; Murtiyoso & Grussenmeyer, 2023; Martin-Brualla *et al*., 2021; Croce *et al*, 2023). Obszar, w którym NeRF jest dość często wykorzystywany to dokumentacja dziedzictwa kulturowego (Murtiyoso *et al*., 2023; Balloni *et al*., 2023; Mazzacca *et al*., 2023; Murtiyoso & Grussenmeyer, 2023). W związku z takimi możliwościami nie dziwnym jest, że zaczęto porównywać tego typu metody modelowania przestrzeni z metodami fotogrametrycznymi (Condorelli *et al*., 2021; Martiyoso *et al*., 2021, Vandenabeele *et al*., 2023). Ponadto wraz z powstaniem nowych modeli została uwzględniona również możliwość uczenia na większym zbiorze danych dla zdjęć UAV (*ang. unmanned aerial vehicle*) (Nex *et al*., 2023).

Dodatkowo, zaletą, o której wspomniano w pracy Palestini *et al.* (2022), w której wykorzystano model Instant Neural Graphics Primitives (Müller *et al*, 2022) jest to, że potrafi być pomocny w przypadku, kiedy obiekt jest wystawiony na silne działanie światła słonecznego, a jego powierzchnia jest silnie połyskliwa. W pracy tej zwrócono uwagę na fakt, że technologia ta jest bardzo młoda oraz tempo jej rozwoju jest wręcz wykładnicze.

Widoczny potencjał metod z wykorzystaniem NeRF był motywacją do przeprowadzenia eksperymentów dotyczących nauki NeRF na dużym zbiorze danych UAV na przykładzie modelowania masztów telekomunikacyjnych. Ze względu na dużą dostępność różnych rodzajów tego typu sieci neuronowych, z których każdy w założeniu powinien reprezentować

pewne unikalne właściwości zdecydowano się na przetestowanie kilku z nich i weryfikację możliwości zastosowania w przypadku inwentaryzacji masztów telekomunikacyjnych.

2. ANALIZA DOSTĘPNYCH ROZWIĄZAŃ

Jak zostało wspomniane w ramach badań przetestowane zostały różne modele NeRF. Jednakże ze względu na dość dużą liczbę dostępnych rozwiązań, początkowo prace polegały na analizie istniejących modeli. Brano pod uwagę zarówno przeznaczenie danego modelu, a także łatwość implementacji.

Pierwowzór, na podstawie którego są budowane kolejne wersje to wspomniany NeRF (Mildenhall *et al*., 2021). W porównaniu do nowszych modeli uczy się od nich dłużej i jest mniej dokładny. Instant Neural Graphics Primitives (Müller *et al*, 2022) to model, który wyróżnia się możliwością nauki sieci w około 5 sekund oraz prostą instalacją. Dostosowany pod względem optymalizacji pod karty graficzne NVIDIA. Model ten potrzebuje dużej ilości VRam do nauki modelu, jego użycie zależy od wielkości zbioru uczącego. NeRF in the Wild (Martin-Brualla *et al*., 2021) to metoda tworzenia modelu oparta na NeRF za pomocą zbioru nieustrukturyzowanych zbiorów nieuporządkowanych zdjęć. Daje możliwość nauki modelu na zdjęciach w różnych warunkach takich, jak zmienne oświetlenie czy przejściowa okluzja (*ang. transient occluders*). Stworzony został przez Google Research i jego kod nie jest publicznie dostępny. Block-NeRF (Tancik *et al*., 2022) jest modelem NeRF pozwalającym na reprezentację wielkoskalowego środowiska. Nazwa pochodzi od metody, która dzieli model na bloki, dzięki czemu można podzielić model na indywidualnie szkolone fragmenty. W pracy zbiór uczący składał się z 2.8 miliona obrazów. Za model odpowiedzialne jest Waymo oraz Google Research, tak samo jak NeRF in the Wild implementacja jego nie jest otwarto źródłowa. Mega-NeRF (Turki *et al*., 2022) jest wykorzystywany do renderowania scen o skali miejskiej z użyciem zdjęć UAV. Dzieli on model na kilka podmodułów, z których każdy może być trenowany oddzielnie, dzięki czemu daje możliwość wytrenowania sieci neuronowej w manierze rozproszonej na podstawie tysięcy obrazów przetwarzanych na wielu kartach graficznych. W przeciwieństwie do modeli NeRF in the Wild oraz Block-NeRF, które również dają możliwość szkolenia na dużym zbiorze danych, ten model jest otwarto źródłowy i można go przetestować dla własnym zbiorze zdjęć. Nerfstudio (Tancik *et al*., 2023) to rozwiązanie (framework) stworzone z wykorzystaniem języka Python pomagające w rozwoju sieci typu NeRF. Umożliwia on zbudowanie podstawowych komponentów sieci i zintegrowanie z wizualizacją w czasie rzeczywistym w przeglądarce, jak i z rożnymi metodami eksportu. Dodatkowo Nerfstudio dostarcza autorski model sieci, który nie występuje w oddzielnej formie i jest wzorowany na modelu MipNeRF-360.

Na podstawie przeglądu literatury i dostępnych rozwiązań NeRF wybrano do badań model Mega-NeRF oraz dwa modele dostępne w Nerfstudio – Nerfacto i Nerfacto-big. Modele zostały wytrenowane przy różnych ustawieniach parametrów. Opisane zostały wyniki uczenia modeli. Nerfstudio umożliwia eksport zrekonstruowanej geometrii do postaci chmury punktów, dzięki czemu możliwe było porównanie wyników z referencyjnymi

danymi pozyskanych z wykorzystaniem fotogrametrii UAV oraz techniki naziemnego skaningu laserowego (TLS).

3. NERF Z WYKORZYSTANIEM DANYCH UAV DLA MASZTÓW TELEKOMUNIKACYJNCYH

3.1. DANE

Eksperymenty zostały wykonane z wykorzystaniem zbioru 393 zdjęć UAV masztu telekomunikacyjnego o rozdzielczości 5280x3956 pikseli (średnio 4.34 MB per obraz). Zdjęcia zostały wykonane za pomocą drona Mavic 3 Enterprise wyposażonego w szerokokątną kamerę 4/3 CMOS z sensorem 20MP i ogniskową 24mm oraz wbudowaną mechaniczną migawką.

3.2. WYKONANE EKSPERYMENTY

Pierwszym etapem eksperymentu było przetestowanie możliwości wykorzystania modelu Mega-NeRF. Oprócz możliwości trenowania sieci z zapisywaniem parametrów uczenia modelu (*ang. training batches*) daje szerokie możliwości wykorzystania w celach wizualizacji jak i ewaluacji wyników. Nie są one jednak dostępne w bezproblemowy sposób zaraz po zainstalowaniu wszystkich bibliotek. W przypadku takich modeli, jak NVIDIA Instant NeRFs czy projektu Nerfstudio, szkolenie i wizualizacja opierała się na wskazaniu dwóch katalogów, jednego ze zdjęciami a drugiego z elementami orientacji. Mega-NeRF pod tym względem jest bardziej złożony. Przyjmuje on możliwość korzystania z własnych danych i informacji o elementach orientacji zewnętrznej kamery do nauki, jednakże muszą to być pliki wyeksportowane za pomocą oprogramowania COLMAP przetworzone przez skrypt do specjalnego formatu zapisanego za pomocą plików biblioteki PyTorch. Po wyeksportowaniu tego typu plików należy uruchomić skrypt odpowiedzialny za partycjonowanie danych oraz związane z tym maskowanie obrazów do nauki. Jest to podejście daleko inne od analogicznych rozwiązań z uwagi na fakt, że można podzielić model na części, które będą szkolone w tym samym czasie na kilku różnych kartach graficznych. Ideę tę reprezentuje wizualizacja z pracy twórców (Rysunek 2).

Zbiór zdjęć został podzielony na treningowy i walidacyjny, w proporcji 372 do 21. Trening modelu Mega-NeRF został przeprowadzony z wykorzystaniem zdefiniowanych parametrów (Tabela 1).

Na sprzęcie wyposażonym w procesor Intel Core i9-12900KF, 128 GB RAM o taktowaniu 2400 MHz oraz karcie graficznej GeForce RTX 3080 Ti, trening trwał około 35 godzin. Wyniki ewaluacji modelu prezentuje poniższa tabela 2. Dla porównania przeprowadzono trening zbioru przygotowanego przez twórców modelu, dla którego wyniki ewaluacji przedstawiono poniżej (Table 3).

Wskaźniki widoczne w tabelach powyżej są często używane w przypadku ewaluacji jakości modelu i zostały dobrze wytłumaczone w pracy (Kniaz *et al.*, 2023). PSNR (*Peak Signal-to-Noise Ratio*) jest to wskaźnik jakości, który mówi o wpływie szumu. Im

wyższy współczynnik PSNR, tym lepsza jakość rekonstrukcji. Aby ocenić jakość modeli NeRF, warto uwzględnić wskaźniki LPIPS (*Learned Perceptual Image Patch Similarity*) oraz SSIM (*Structural Similarity Index Measure*), które koncentrują się na dokładności reprezentacji sceny 3D. Indeks SSIM waha się od -1 do 1, gdzie 1 oznacza idealne podobieństwo między porównywanymi obrazami. Wyższe wartości SSIM generalnie odpowiadają lepszej jakości obrazu. Metryka LPIPS oblicza odległość między reprezentacjami cech wyodrębnionymi z obrazów. Im niższa wartość LPIPS, tym bardziej percepcyjnie podobne są obrazy. Wartości wskaźników dla treningu zbioru testowego i zbioru zdjęć UAV dla masztu ukształtowały się na podobnym poziomie. Wizualizacja wyników możliwa jest z poziomu viewera i prezentuje się następująco (Tabela 3).

We wcześniej przedstawionej wersji eksperymentu obszar testowy nie był dzielony na kilka podmodeli. Z racji, że jest to jedna z cech charakterystycznych modelu Mega-NeRF zostało to sprawdzone w kolejnym eksperymencie. Należy zauważyć, że zbiór dostarczony i testowany przez twórców modelu znacząco się różni od naszego przypadku. W pierwotnym zamyśle zdjęcia były robione na stałej wysokości, a zmieniała się ich pozycja pozioma. Charakterystyka nalotu dronem wokół wysokiego, smukłego obiektu, jakim jest maszt telekomunikacyjny różni się znacząco od typowego nalotu np. nad miastem czy zdjęć naziemnych wykonywanych dla obiektu z bliskiego zasięgu. Biorąc pod uwagę te aspekty nie można było zagwarantować, że domyślne podejście przyczyni się do poprawy jakości wytrenowanego modelu, jednak próba została podjęta. Zbiór został podzielony na 4 części z parametrem --grid_dim 2 2. Spowodowało to konieczność wyszkolenia 4 podmodeli, z których każdy zajmował około 35 h.

Powyższe wizualizacje przedstawiają wynik działania skryptu, którego zadaniem jest podzielenie jednego dużego modelu na podmoduły (Rysunek 6). Trening każdego z nich następuje z uwzględnieniem informacji o geometrii pikseli (Tancik *et al.*, 2023).

Analizując wyniki wizualnie, po przyjrzeniu się szczegółom można stwierdzić, że zdjęcie wyeksportowane jako wynik uczenia modelu z podziałem na podmodele jest bardziej szczegółowe (Rysunek 7). Niestety okazuje się, że ta metoda działa wtedy, kiedy widoczny obraz został wyszkolony z kilku perspektyw odpowiadających podmodelom. W przypadku, kiedy przestał być widoczny horyzont, a co za tym idzie nie były dostępne 2 z 4 podmodeli do nauki, jakość eksportowanych obrazów drastycznie spadła, co widoczne na rysunku 8.

Ocenę wizualną i zauważone problemy potwierdziły również wartości metryk wykorzystanych do ewaluacji model (Tabela 4). Wartość wskaźnika PSNR znacząco spadła względem pierwszej iteracji modelu. Podobnie wartość wskaźnika SSIM spadła prawie dwukrotnie.

Kolejna próba treningu modelu została przeprowadzona dla zwiększonej do 800000 liczby iteracji przy zastosowaniu podziału na jeden kawałek. Wyniki przedstawiono w tabeli poniżej (Tabela 5). Widoczny wzrost dokładności skłonił do przeprowadzenia kolejnej iteracji trenowania modelu. W celu sprawdzenia wpływu zwiększenia ilości epok nauki na dokładność modelu został przeprowadzony test na 900000 epokach. Dla takich danych nauka trwała około 65 godzin, rendering 372 widoków odpowiadających pozycji kamery podczas wykonywania zdjęcia 15 godzin. Poniżej przedstawiono przykładowy widok odpowiadający elementom orientacji z pierwszego zdjęcia (Rysunek 9).

Analizując uzyskane wyniki można powiedzieć, że dla tego typu danych podział zbioru na podmodele obniża dokładność. Natomiast zwiększenie liczby epok pozytywnie wpłynęło na wyniki. Biorąc pod uwagę przeprowadzone iteracje trenowania modelu Mega-NeRF można było zobaczyć potencjał wykorzystywania takiego rozwiązania do tworzenia modelu 3D dla obiektu jakim jest maszt telekomunikacyjny z wykorzystaniem zdjęć UAV. Problemy widoczne podczas oceny wyników były motywacją do tego, by przetestować również inne modele.

Kolejnym etapem badań było sprawdzenie możliwości Nerfstudio dla tego samego zbioru danych. Framework ten został wykorzystany dlatego, że ma on zadanie stworzenia przyjaznego dla użytkownika doświadczenia z wykorzystywaniem modeli NeRF. Oferuje swój własny model jak i implementacje innych. Daje również możliwość łatwej wizualizacji wyuczonego modelu oraz jego eksportu do chmury punktów w formacie *.pcd oraz *.ply. Dzięki czemu wygenerowana chmura punktów, która jest wynikiem trenowania modelu może być porównana z referencyjną chmurą punktów. W przeciwieństwie do Instant Neural Graphics Primitives nie przechowuje on wszystkich wczytanych zdjęć w pamięci karty graficznej. Dlatego też dzięki temu nie występuje problem związany z ilością VRAM. Problem jednak pojawia się podczas wczytywania modelu do pamięci RAM, ponieważ zaimplementowana metoda pobiera wszystkie zdjęcia, zamienia je na obiekty zgodne z biblioteką PyTorch i przechowuje w pamięci podręcznej. W związku z takim podejściem do wyuczenia modelu na 393 zdjęciach o łącznej wielkości 1.5 GB konieczne jest wykorzystanie ponad 128 GB pamięci tymczasowej. Obejściem tego problemu, które zostało wykorzystane na potrzeby badań było przypisanie odpowiedniej wielkości partycji wymiany SWAP. Zwiększenie o 200 GB pozwoliło na wyuczenie modelu na pełnym zbiorze zdjęć bez potrzeby zmniejszania ich rozdzielczości.

Nerfstudio daje możliwość szkolenia danych na różnych modelach takich jak Instant-NGP, Instruct-NeRF2NeRF, K-Planes, LERF, Mip-NeRF, NeRF, Nerfacto, Nerfbusters, NeRFPlayer, Tetra-NeRF, TensoRF, Generfacto. Domyślnym modelem rozwijanym przez twórców Nerfstudio jest model Nerfacto. Jest to kombinacja wielu różnych opublikowanych modeli. Występuje on w kilku wariantach: Nerfacto, Nerfacto-big, Nerfacto-huge, Depth-nerfacto. Na potrzeby badań przetestowano model Nerfacto oraz Nerfacto-big. Model Nerfacto-big jest wolniejszy, jeśli chodzi o trening modelu, ale z założenia powinien dawać lepsze wyniki.

Porównując wizualnie wyniki z Nerfacto (Rysunek 10) i Nerfacto-big (Rysunek 11) z wytrenowanym wcześniej modelem Mega-NeRF nie widać dużych różnic w odwzorowaniu elementów masztu telekomunikacyjnego. Jednakże wartości wskaźników ewaluacji modelu w obu przypadkach są wyższe niż dla Mega-NeRF (Tabela 7). Można więc powiedzieć, że model Nerfacto jest bardziej dostosowany do takiego typu obiektów.

4. ANALIZA DOKŁADNOŚCI CHMURY PUNKTÓW NERF

Analiza dokładności została przeprowadzona względem danych referencyjnych. Dla masztu zostały pozyskane zdjęcia UAV, które posłużyły do treningu modeli i wygenerowania chmury punktów. Ten sam zestaw danych został wykorzystany do utworzenia chmury

punktów z dopasowania obrazów jako produkt przetwarzania fotogrametrycznego. Oprócz tego pozyskana została chmura punktów z naziemnego skanera laserowego Leica RTC360. Chmury punktów wyeksportowane z Nerfstudio zostały porównane w programie CloudCompare. Obliczone zostały odległości między chmurami referencyjnymi a chmurą z Nerfstudio. Poniżej przedstawione zostały wyniki (Rysunek 12)).

Średnia różnica odległości między chmurą punktów wynikową z modelu Nerfacto a chmurą punktów z gęstego dopasowania obrazów wyniosła 1.3 cm, a odchylenie standardowe ukształtowało się na poziomie 1.6 cm (Rysunek 13). Widać natomiast, że największe różnice były w miejscach, gdzie występują braki w chmurze punktów z dopasowania obrazów, która również ma swoje wady (Rysunek 14).

Znacznie lepszymi danymi referencyjnymi dla masztów telekomunikacyjnych są chmury punktów z naziemnego skanowania laserowego, gdyż w ich przypadku występuje znacznie mniej „martwych pól" i nie występuje tak dużo problemów związanych z kompletnością danych. Porównano zatem chmurę punktów z Nerfacto z chmurą TLS. Uzyskane wyniki, jeśli chodzi o wartości liczbowe ukształtowały się na podobnym poziomie. Średnia odległość między chmurą 1.6 cm z odchyleniem standardowym 1.5 cm (Rysunek 15). Ponadto, w obu przykładach porównania najwyższe wartości różnic nie przekroczyły 40 cm.

W przypadku analizy porównawczej względem chmury punktów TLS widoczne jest, że największe różnice nie występowały dla nieodwzorowanej w pełni drabinki masztu, a dla powierzchni anten sektorowych (Rysunek 16).

Średnia różnica odległości między chmurą punktów wynikową z modelu Nerfacto-big a chmurą punktów z gęstego dopasowania obrazów wyniosła 1.2 cm, a odchylenie standardowe ukształtowało się na poziomie 2.1 cm (Rysunek 19). Można więc powiedzieć, że wyniki są bardzo podobne do modelu Nerfacto, który w teorii powinien być mniej dokładny. Jak widać na powyższych rysunkach (Rysunek 17, Rysunek 20) chmura punktów z modelu Nerfacto-big jest bardziej zaszumiona niż chmura z modelu Nerfacto. Jednakże gęstość tej chmury jest znacznie wyższa. Prawdopodobnie wykorzystanie metod, które pozwoliłyby na usunięcie szumów z wynikowej chmury poprawiłoby statystyki. Średnia odległość między chmurą punktów z Nerfacto-big a chmurą TLS wyniosła 1.6 cm z odchyleniem standardowym 1.9 cm (Rysunek 18). Podobnie jak w przypadku porównania z chmurami z gęstego dopasowania obrazów można zobaczyć, że wyniki nie różnią się bardzo dla obu modeli.

5. PODSUMOWANIE

Wśród modeli sieci neuronowych NeRF można wyróżnić dwa podstawowe rodzaje odnośnie tego, jaki obiekt ma być odwzorowywany. Istnieją modele 360 stopniowe zakładające, że kamera będzie poruszać się dookoła danego obiektu oraz modele przestrzenne. Założeniem MegaNerf jest odwzorowanie dużej powierzchni za pomocą zdjęć zrobionych z dronów. W naszym wypadku powierzchnia odwzorowania nie jest duża w rozumieniu twórców modelu. Maksymalne różnice współrzędnych x oraz y kamery różnią się o kilkanaście metrów. Biorąc to pod uwagę nie dziwi fakt, że model ten nie jest przystosowany do

precyzyjnego wizualizowania względnie drobnych elementów. Kolejną wadą tego modelu w zastosowaniach fotogrametrycznych jest brak możliwości wyeksportowania modelu w formacie chmury punktów albo mesha, jedyny eksport jaki jest obsługiwany w oryginale to do pliku Octree który może zostać odczytany przez dedykowany viewer. Problemy te rozwiązuje Nerfstudio które daje możliwość wczytania różnych modeli przy jednoczesnym zapewnieniu wizualizacji efektów nauki dla każdego z nich. Wczytywanie danych stanowi pewne ograniczenie, które ujawnia się przy potrzebie wykorzystania większego zbioru danych do nauki. Rozszerzenie pamięci tymczasowej o dysk twardy rozwiązuje problem dla umiarkowanie dużych danych, jednakże w skrajnych wypadkach takie podejście może również okazać się nie wystarczające. W celu zapobiegania takim przypadkom należałoby zmienić sposób wczytywania danych do modelu. Bezpośrednie porównanie miar dokładności dla tych samych zdjęć pokazuje, że w testowanym przypadku Nerfstudio osiąga lepsze efekty niż Mega-NeRF.

Podsumowując, modele oparte na technologii NeRF, takie jak Mega-NeRF, Nerfacto i Nerfacto-big, wykazują potencjał do rekonstrukcji obiektów z użyciem danych z dronów. Niemniej jednak, istnieje wiele czynników, takich jak ograniczenia modelu, jakość i ilość danych treningowych, parametry treningowe itp., które mogą wpływać na jakość i dokładność rekonstrukcji. W związku z tym, przed zastosowaniem tych modeli w praktyce, konieczne jest dokładne zrozumienie ich ograniczeń i dostosowanie do konkretnego przypadku użycia.

Details of authors:

Eng. Bartosz Lewandowski
e-mail: barteklewandowski9@gmail.com

MSc. Eng. Łukasz Wilk
e-mail: lukasz.wilk@pw.edu.pl

MSc. Eng. Paulina Zachar
e-mail: paulina.zachar@pw.edu.pl

Submitted 23.11.2023
Accepted 31.12.2023