An automated Pipeline to bring NeRFs to the Industrial Metaverse

Sabine Schleise
SICK AG
79183 Waldkirch,
Germany
sabine.schleise@sick.de

Uwe Hahne
Hochschule Furtwangen
78120, Furtwangen,
Germany
uwe.hahne@hfu.eu

Jakob Lindinger
SICK AG
79183 Waldkirch,
Germany
jakob.lindinger@sick.de

ABSTRACT

The Industrial Metaverse offers new opportunities in various industrial sectors, such as product development, collaboration, sales and marketing. It relies on rendering 3D scenes of industrial plants, factories, and machines that are viewed by multiple human observers in various industrial applications. Creating these 3D scenes requires either elaborate 3D modeling efforts or complex 3D reconstruction methods. Both approaches require expensive hardware and highly trained individuals to create photo-realistic results. To overcome these limitations, we offer an end-to-end pipeline that generates photo-realistic 3D scene renderings from video input using NeRF (neural radiance fields). Our tool automates the entire process, resulting in remarkable efficiency gains. It reduces the creation time from days to minutes, even for untrained users.

Keywords

3D Vision, NeRF, Industrial Metaverse, immersion, neural rendering

1 INTRODUCTION

Rapid advancements in the availability of graphics processing unit (GPU) technology allow new opportunities across various industrial sectors, including training, product development, and collaboration that are often described with the term *Industrial Metaverse*. At its core, the Industrial Metaverse relies on rendering three-dimensional (3D) scenes that are viewed by multiple human observers in various industrial applications. For instance, in automation, engineers and technicians can design, test, and implement robotic automation systems. By creating virtual simulations of automated workflows, also known as digital twins, the efficiency of robots in manufacturing lines or logistics operations can be optimized by human operators via remote access. It becomes also possible to predict potential issues before they arise and plan maintenance without disrupting the production process.

Currently, these 3D scenes (as well as some digital twins) are usually rendered from explicit representations such as meshes, point clouds, or volumetric grids. Those are known for their structural integrity and compatibility with modern rendering systems. However, in 2020, a novel approach called Neural Radiance Fields (NeRF) [MST*20] emerged. It diverges from these conventional methods as it uses neural networks and volume rendering methods for novel view synthesis. It creates visually impressive results from only a few photos shot by regular consumer cameras.

NeRFs are implicit scene representations, typically employing a Multi-Layer Perceptron (MLP) that is trained by comparing volumetric renderings with captured photos. Instead of reconstructing the 3D scene geometry, NeRF generates a volumetric representation called a *radiance field*. This radiance field is stored in a neural network which returns color and density values for any given point and viewing direction within the relevant 3D space. Due to this novel representation, there is currently no existing framework to generate a neural radiance field from real world data for novice users. To address this, we introduce a user-friendly software tool called *NeRF Trainer* that simplifies the creation of neural radiance fields. Our tool provides an end-to-end solution that streamlines the training of NeRFs. It automates the entire process including image selection, extraction of camera orientation using COLMAP [SZPF16, SF16], training of NeRF models and export of resulting new images, videos and 3D meshes. By leveraging the improved NeRF algorithm *Instant-NGP* [MESK22] our tool achieves short training times of a mere 20 seconds. We aim to democratize the use of NeRF technology, making it accessible to a broader audience without the barriers of technical

expertise or hardware limitations. We believe that this innovation represents a significant step forward in realizing the potential of NeRF in industrial settings and beyond.

2 BACKGROUND

Before we delve into the steps required to create NeRFs in the upcoming sections, we present fundamental findings below.

2.1 NeRF

NeRF is a method for synthesizing novel views of complex scenes by optimizing a continuous volumetric function using a sparse set of 2D images. Given a set of images showing a static scene from multiple view angles, along with their corresponding camera poses, the neural network learns to represent that scene in a way that allows new views to be synthesized. The NeRF algorithm represents a continuous scene as a vector-valued function F_Θ whose input is a 3D spatial location $\mathbf{x} = (x, y, z)$ and a 2D viewing direction $\mathbf{d} = (\phi, \theta)$. Its output is the emitted color $\mathbf{c} = (r, g, b)$ and a volume density σ, that ranges from $[0, \infty)$. To approximate this continuous 5D scene representation, an MLP network F_Θ is utilized, where $F_\Theta : (\mathbf{x}, \mathbf{d}) \longrightarrow (\mathbf{c}, \sigma)$. The weights Θ of the network are optimized to map each $5D$ input vector to its corresponding volume density and directional emitted color. The algorithm enables view synthesis through a three-step process: first, sampling 5D coordinates along camera rays and processing them through an MLP to produce both color and volume density; second, using the MLP's output to generate individual pixels of an image using classical volume rendering techniques; and finally, leveraging the differentiability of the rendering function to optimize the MLP's weights by minimizing the loss function defined as the total squared error between the rendered and true pixel colors.

The NeRF algorithm encounters difficulties when rendering scenes with high-frequency variations in color and geometry due to its direct operation on 5D input coordinates. This inefficiency is caused by the inherent bias of deep neural networks towards learning lower-frequency functions, a limitation that Rahaman et al. [RBA*19] have highlighted. To address this, NeRF uses a positional encoding that leverages sine and cosine functions to improve its representation of geometry. However, this approach can complicate training and affect GPU performance due to the high cost of complex control flow and memory operations.

These challenges are addressed by Müller et al. in "Instant Neural Graphics Primitives" (Instant-NGP) [MESK22] which introduces a novel input encoding method that significantly streamlines the training process. By employing a smaller network integrated with a multi-resolution hash table, this method reduces computational demands and optimizes memory usage. The result is a speed-up of several orders of magnitude in training time for neural graphics components, enabling high-quality renderings to be trained in seconds, even on consumer hardware. This breakthrough demonstrates the potential for rapid, efficient training of neural scene representations, as illustrated in Figure 1, which shows a scene reaching convergence in training within just 20 seconds. The scene was captured from a 180-degree perspective, utilizing 145 frames for training, and trained using a laptop with an NVIDIA A3000 GPU.

Figure 1: Training of a neural radiance field using the Instant-NGP algorithm, displayed in the Instant-NGP GUI [MESK22]. Rendering at initialization (top) and after 20 seconds of training (bottom). The object was captured from a 180-degree perspective, utilizing 145 images for training.

2.2 Manual pipeline

After laying the conceptual groundwork for NeRF and its advances in handling high-frequency scene variations, we delve into the practical methodology for training a NeRF model, focusing on the seamless integration of theory and application.

The pipeline typically starts with capturing a short video of a scene or object with a smartphone. This is followed by frame extraction, which is the process of assembling a set of images that capture the static scene from different angles. These images, whether extracted from video sequences or collected as single images, form the backbone of the dataset, capturing the spatial intricacies and visual essence of the scene from multiple perspectives.

Once the frames have been extracted, the focus shifts to determining the camera positions for each frame in the

dataset. This is achieved by using COLMAP [SF16], a structure-from-motion (SfM) software, which reconstructs the scene in 3D by identifying common points across the image set and inferring the camera positions and orientations from which each image was captured. This process lays the foundation for a spatial understanding of the scene, allowing the NeRF algorithm to accurately interpret and reconstruct the 3D environment.

With the frames extracted and the camera poses computed, the dataset is prepared for the next phase: training the NeRF model. The model, represented by a multi-layer perceptron (MLP), is tasked with learning a continuous volumetric scene function. It takes as input the 3D coordinates and 2D viewing directions, and outputs the color and volume density for each point in space. Training involves feeding the model with 5D coordinates derived from the camera rays intersecting the scene, and adjusting the network weights to minimize the discrepancy between the synthesized and actual image pixels. This optimization process uses gradient descent to iteratively refine the model to improve its ability to render novel views with high fidelity.

3 RELATED WORK

The impact of NeRF on the computer vision community has been profound and widespread. Since its publication in 2020 numerous papers and preprints emerging on platforms, reflecting the intense interest and rapid development within this area. We refer to the recent survey papers [GGH*22, TTM*22] to get an impression about the broad evolution of NeRF-based methods in the last years.

Recognizing its significance, developers have created frameworks that addressed a variety of needs, from simplifying the integration of different NeRF techniques to adapting complex algorithms for broader use. For instance, the introduction of frameworks with modular architectures [YL22, HLY*20, XC22, YSK22] have made it easier for users to work with various NeRF models within a single application. This integration supports experimentation and refinement of NeRF models, benefiting the research community. In this frameworks, new methods can be easily integrated without having to rewrite encodings, network architectures or volume rendering functions. This is particularly useful as the field of NeRF is constantly evolving, with new methods and approaches emerging regularly. The framework NVIDIA Kaolin Wisp [TPT*22] was designed as a dynamic, research-focused library for neural fields to help researchers address the challenges of this discipline. Notably, efforts have transitioned from reimplementing existing NeRF methods in different programming environments [Bha22, YC20] to offering comprehensive solutions such as Nerfstudio [TWN*23] that span the entire spectrum of NeRF technology, from data preprocessing to model training and visualization.

Despite NeRF technology's growing adoption across various fields, there are significant challenges with the current frameworks that hinder its broader accessibility, especially for non-technical users. The support for development remains basic, with a proliferation of research papers and a lack of code consolidation making it difficult to track advancements. Many researchers publish their work in isolated repositories, complicating the transfer of features and contributions between different NeRF implementations. Furthermore, the existing tools for running NeRFs on real-world data are limited. In particular, the data preparation step involving COLMAP [SF16] requires command line knowledge, which is a barrier for those unfamiliar with such interfaces. In addition, the plethora of features and NeRF models offered by most frameworks can overwhelm users new to the technology, making it difficult for them to understand its capabilities and effectiveness. These issues underscore the need for more intuitive and consolidated frameworks that can meet the needs of users who want to explore NeRF technology without deep technical expertise.

A notable advancement in this field is the startup *Luma AI*[1], which has significantly simplified the process of creating NeRFs. By releasing an iPhone app, *Luma AI* enables users to effortlessly produce videos that can then be transformed into a NeRF. The app guides users through the entire process, from capturing video to generating the NeRF, thereby making this advanced technology accessible to a wider audience. In addition, companies like *Dromni*[2] are also working to democratize the process of generating NeRFs by developing platforms that are easy to use. Nonetheless, a significant barrier to their widespread adoption in sensitive sectors is the necessity for data upload to cloud services. This is particularly critical in various industrial domains where safeguarding data confidentiality is essential.

4 NERF TRAINER

In this chapter, we present a comprehensive exploration of our innovative NeRF Trainer, a tool designed to democratize the creation of NeRFs by simplifying its process into a user-friendly application.

4.1 Requirements and design choices

Our framework is designed with specific functional requirements to facilitate the generation and exploration of neural radiance fields from video inputs. In the following, we provide a breakdown of these key requirements.

[1] https://lumalabs.ai
[2] https://www.veovid.com

4.1.1 Automated Pipeline

From a user perspective, the main feature of our framework is its capability to process video uploads. It automatically executes the entire pipeline described in Section 2.2 to generate the radiance field. This process involves converting the video into a series of images that capture the scene from multiple angles, then run COLMAP to extract the camera calibration and finally train the radiance field to construct a 3D representation of the scene. The video processing capability is designed with flexibility in mind, allowing the user to specify optional parameters to fine-tune the output. Through this automated pipeline, our technology simplifies the traditionally complex process of generating NeRF models. It allows users to obtain high-quality 3D representations from standard video footage, bypassing the need for manual image selection and camera calibration.

4.1.2 Advanced Algorithm Utilization

The framework's effectiveness hinges on incorporating an advanced, improved radiance field algorithm. The selected algorithm must strike a delicate balance between rapid processing and the rendering of scenes with high fidelity. The ultimate goal is to achieve a reconstruction that faithfully mirrors the real-world scene's complexity and subtleties, ensuring an authentic and immersive exploration experience.

4.1.3 Enhanced Output Options

Beyond the generation of neural radiance fields, the framework offers advanced functionalities for rendering videos and exporting meshes. This broadens the scope of exploration and utilization of the constructed 3D scene representations.

Snapshots: Save model snapshots as .ingp file for analysis and further exploration within the Instant-NGP's framework.

Video Rendering: Users have the option to render videos directly from the synthesized neural radiance fields, enabling dynamic visualization of the scene from various perspectives. This feature is instrumental in presenting and reviewing the photorealistic outputs in a more interactive and engaging format.

Mesh Exportation: The technology also supports the exportation of meshes. This feature allows for seamless integration into existing workflows, providing users with the tools to incorporate 3D models into their projects.

4.1.4 Graphical User Interface (GUI)

To ensure accessibility and ease of use, the application features a GUI. This interface will support informative interactions, data uploads, and parameter configurations, tailored to optimize training outcomes. The GUI is envisioned as a bridge between the user and the complex processes underlying neural radiance field generation, making the technology approachable for users with varying levels of expertise.

4.1.5 Screenshots as Training Verification

A key requirement is the ability to produce photorealistic renderings of the trained model's output. These renderings are essential for verifying the model's accuracy in recreating the scene. They act as a proof of concept, demonstrating the model's ability to replicate the real-world environment with precision and detail.

4.2 The framework

In the pursuit of democratizing advanced 3D visualization technologies, we introduce a web-based platform designed to simplify the training of NeRFs. This platform is tailored for both novice and experienced users aiming to generate high-fidelity 3D scenes from conventional images or videos. The user interface is crafted to guide users through the data upload, parameter adjustment, and visualization stages with ease, resulting in the generation of an .ingp file compatible with the Instant-NGP framework [MESK22] for immersive exploration. The back-end, built in Python, can run on any server equipped with an NVIDIA RTX GPU, eliminating the need for a local graphics card. This combination of a user-friendly interface and a separate back-end makes it simpler for users to create high-quality neural radiance fields. As illustrated in Figure 2, the front-end start screen provides the user with insights into NeRFs, results from previously trained models, and the platform's capabilities. This introductory segment aims to acquaint users with the potential and applications of NeRF technology.

4.3 Workflow

The workflow begins when users upload scene data to the platform, followed by an opportunity to modify parameters before the pipeline process begins. If users choose not to make any adjustments, the pipeline will use the default parameters by default. These defaults are designed to provide a balance between performance and quality, ensuring that the NeRF model is trained efficiently and effectively under general conditions. The pipeline process is schematically depicted in Figure 3. Our pipeline is designed to process video, images and RGB-D data. The prepocessing steps for these formats differ: When processing video data, we start by extracting individual frames. Next, camera parameters are determined using COLMAP [SF16]. Once these preliminary steps have been completed, the derived data is transformed into JSON format. This JSON-formatted data, together with the corresponding images, forms the training data set for the neural radiance field.

Figure 2: Rendered view of the background information section on the website. This section includes an introduction to what functionalities the website offers, followed by examples of previously trained NeRF models for qualitative evaluation, and concludes with an outline of the NeRF training pipeline process.

For iPhone users, especially the newer models such as iPhone 12 and its successors, we offer an alternative option to create a NeRF from RGB-D data. The Record3D[3] application allows the creation of recordings including the camera poses and parameters. This data is then read from the file and can be converted directly into the required JSON format, eliminating the need for camera calibration via COLMAP. This eases the transition to the training phase and makes the whole process more efficient.

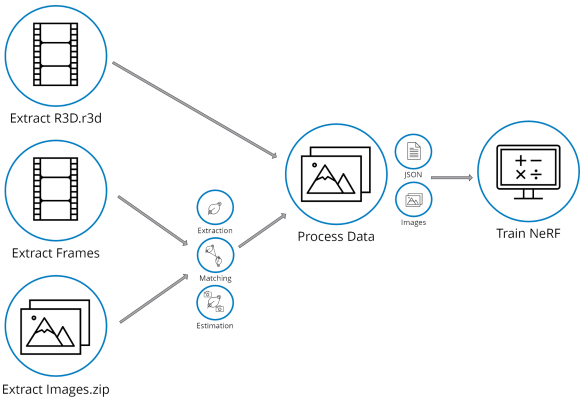

Figure 3: Comprehensive overview of our automated pipeline: The diagram delineates the distinct phases involved when utilizing either video or R3D files (the output format from the iPhone App *Record3D*).

[3] https://record3d.app/

Following the comprehensive overview of our automated pipeline, we transition to a detailed examination of each constituent step.

4.3.1 *Extract Videos*

This stage involves extracting frames from the video, a process critical for preparing the input data for subsequent NeRF model training. The *Extract Videos* component is designed with flexibility in mind, allowing users to customize the extraction process based on specific requirements. This customization includes the ability to adjust the frequency of frame extraction - such as selecting every second or third frame - and setting a maximum limit on the number of frames to be extracted. This flexibility is particularly beneficial for managing computational resources and avoiding memory access errors, which can occur when processing large video files or when operating within the constraints of limited GPU memory. This tailored approach to frame extraction ensures that the pipeline can accommodate videos of varying lengths and complexities, making it possible to generate NeRFs from a wide range of source materials without compromising on the quality or completeness of the data.

4.3.2 *Estimate camera poses from images*

Integrating COLMAP [SF16] for estimating camera poses in the process of creating NeRFs is a step that enables the successful translation of 2D images into their 3D counterparts. It is important to accurately position each frame extracted from the video within a 3D context to achieve precise 3D scene reconstruction, which is a crucial task for NeRF. The accuracy of camera pose estimation directly affects the NeRF model's ability to render new, unseen views of the scene that maintain the original photometric properties. Our framework automatically incorporates COLMAP to simplify the user experience and keep the focus on the NeRF process. This abstraction of technical intricacies allows users to engage with NeRF creation without needing to delve into complex technical details.

4.3.3 *Training Process*

For the training phase, the COLMAP data is converted to a JSON format, encapsulating both intrinsic and extrinsic camera parameters for each image. The neural radiance field is then trained using the Instant-NGP algorithm [MESK22] over a predefined, adjustable number of steps. Upon completing the training, screenshots from the trained model are rendered to evaluate its performance and visual output.

4.3.4 *Qualitative Evaluation*

Upon completion of the training process, the quality of the generated NeRF model is evaluated based on its

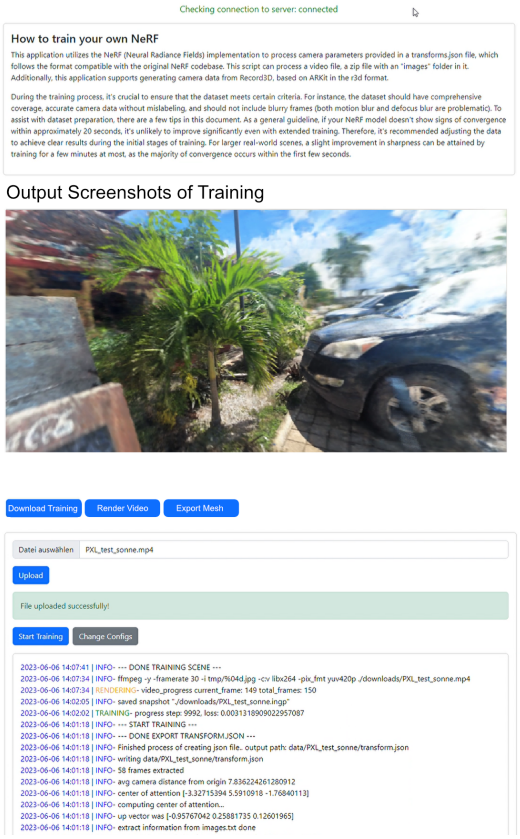

Figure 4: The user interface on the webpage after a radiance field training, encompassing back-end logs (cropped).

ability to produce photorealistic renderings of the scene from novel viewpoints. Therefore we render screenshots from new viewpoints that were not part of the initial dataset used for training. By generating these images, we provide tangible evidence of the model's ability to synthesize new perspectives with high fidelity.

Figure 4 showcases the user interface of the website upon the completion of a NeRF training session. It features a verification screenshot of the trained radiance field, alongside interactive buttons that allow the user to download the resulting data, render a video or export the trained model as a mesh. Additionally, the interface provides logs that detail the training process, offering users insights into the progression and outcomes of their session. Through a socket connection, users receive ongoing updates ensuring that they remain updated about the training's progression and outcome. Such transparent feedback and live interactions elevate the user experience, making the entire training journey more intuitive. This feature is particularly valuable during the execution of computational-intensive processes, such as the COLMAP phase. A specialized logger continually updates the user on the current state of the pipeline, providing an informed overview of the ongoing activities.

4.4 Implementation aspects

To develop and operationalize our NeRF framework, we addressed the challenges outlined in the previous sections, including the fragmented development landscape of NeRF technologies and the obstacles faced by non-technical users. Our approach has led to the creation of a robust, user-friendly framework, underpinned by two Docker containers designed for seamless communication - one dedicated to the front-end and the other to the back-end. This modular architecture not only simplifies deployment but also ensures data privacy by enabling on-premises hosting within a company's network, leveraging Kubernetes clusters for scalability and GitLab CI/CD for continuous integration and deployment. The use of GitLab CI/CD underscores our commitment to maintaining a robust, up-to-date application capable of meeting the evolving needs of our users.

Poetry is employed as a dependency management and packaging tool, which simplifies the process of managing python packages and their versions, ensuring a consistent development environment. Furthermore, Docker is utilized to containerize the back-end, allowing it to run on any server equipped with an NVIDIA RTX GPU. This eliminates the need for users to possess local graphics cards and reduces the setup complexity, offering a plug-and-play solution. By leveraging Docker, we encapsulate the NeRF application within a controlled environment, ensuring that it can be deployed and run with ease, irrespective of the underlying operating system or hardware configuration.

The splitting of the framework into a back-end, developed in Python, and a front-end, crafted with React, allows us to offer a powerful yet user-friendly tool. The React-based content management system (CMS) enables users to interact with the NeRF application without needing to navigate the command line or understand the back-end processes.

To enhance the user experience further, we employ a YAML-based configuration file, making it straightforward for users to interact with the application. YAML's human-readable format ensures that configuring the application is accessible to all, regardless of technical background. This choice underscores our commitment to making NeRF technology as user-friendly as possible.

The strategic division between front-end and back-end, combined with our deployment strategy, empowers organizations to integrate our framework into their networks seamlessly. This paves the way for a diverse range of users, including those without technical expertise, to create content for Industrial Metaverse applications. Our framework's architecture is designed not just for accessibility but also for scalability, making it

an ideal solution for businesses looking to explore and expand their capabilities within the Metaverse.

5 NERF IN THE INDUSTRIAL METAVERSE

The introduction of NeRF marked the birth of the domain of Neural Scene Rendering, a subfield within computer graphics focused on generating 3D scenes using neural networks [TTM*22]. In general, neural scene rendering is a rapidly growing field with the potential to revolutionize the way we generate 3D content, offering data-driven approaches that make the process more accessible and efficient than traditional methods.

NeRF represents a significant step forward in the realm of 3D content creation, offering substantial benefits to the growth of the Industrial Metaverse. Beyond its technical achievements, what sets NeRF apart is its potential to make high-quality 3D rendering more accessible and user-friendly. This accessibility paves the way for a wider range of applications, from enhancing product development processes to enriching educational content. By simplifying the creation of detailed, realistic scenes, NeRF democratizes 3D modeling, making it possible for a broader audience to contribute to and benefit from advancements in virtual environments.

Figure 5 showcases a series of images rendered from neural radiance fields that are generated with our NeRF Trainer. These NeRFs underwent a rapid training process, completed within just one minute, demonstrating the efficiency and capability of the utilized algorithms to generate high-fidelity 3D scenes. The use of advanced optimization techniques and computing power has enabled NeRFs to efficiently process and interpret large amounts of visual data, surpassing traditional methods like photogrammetry.

Additionally, these neural radiance fields can be explored in Virtual Reality (VR) using the Instant-NGP GUI [MESK22], offering an immersive experience that has numerous advantages in the Industrial Metaverse context. The integration of VR technology marks a significant advancement in industrial operations and planning. Through VR (e.g. the production facility displayed in Figure 6), stakeholders can now explore and interact with a fully immersive 3D model of their production environments. This capability not only enhances the understanding of spatial relationships and operational flows within the facility but also facilitates a more intuitive and effective collaboration among team members. Whether for training purposes, safety protocol reviews, or optimization of manufacturing processes, the ability to virtually walk through and examine every aspect of a production line in detail offers unparalleled benefits. These include improved design accuracy, faster decision-making, and a reduction in costly physical prototypes.

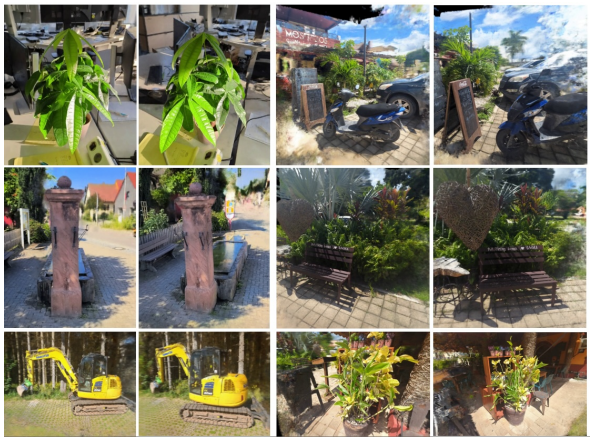

Figure 5: Rendered images from NeRFs trained within 1 minute, showcasing impressive detail and realism despite the constrained training period. This demonstrates the advancements in optimization techniques and computing power, enabling rapid processing and interpretation of visual data.

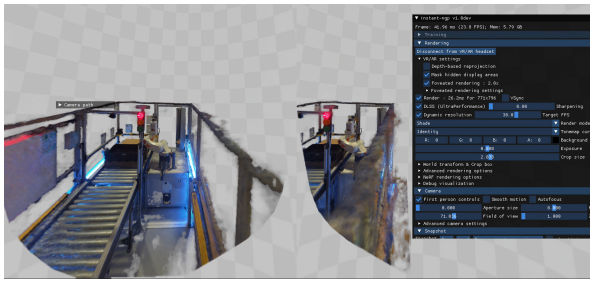

Figure 6: A screenshot showing the dual perspectives as viewed through VR lenses.

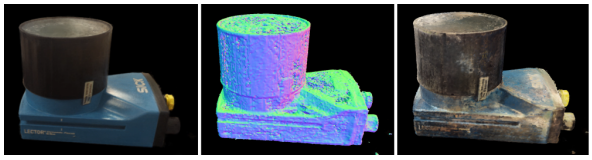

Figure 7: Comparison of renderings and meshes obtained from a NeRF. The first image shows a rendering from the neural radiance field, the second image shows the mesh with vertex normals, and the third image displays the mesh with vertex colors.

Beyond the capability to visualize NeRFs in VR, the method also allows for the option to export these 3D models as meshes, utilizing the Marching Cubes [LC87] algorithm. This process translates the dense, volumetric data produced by NeRFs into a polygonal mesh, which is more compatible with traditional 3D modeling and CAD software. In Figure 7, we provide a visual comparison highlighting the transition from NeRF to an exported mesh using the Marching Cubes algorithm. However, it is important to note that NeRFs, by their nature, are not ideally suited for direct mesh exportation and use in certain applications. NeRFs are remarkable for their ability to capture and display complex light interactions within a scene and

create photorealistic images from a variety of angles. When converted into a mesh, this lighting information and the ability to seamlessly render from arbitrary viewpoints will be lost. Yet, this limitation is contrasted by the significant advantage NeRFs offer in quickly and easily capturing detailed 3D representations of scenes using a smartphone.

6 DISCUSSION

Our framework aims to reduce some of the cumbersome manual steps required for generating NeRFs. However, it comes with its own set of minor limitations: When used as intended, our framework needs, for every group of users (e.g. within a company), a maintainer who is skilled in deploying the front- and backend services. All other users then require internet (or corporate network) access, which is not the case for most other NeRF tools (but ubiquituous nowadays). When used locally (which is not intended but possible), our framework assumes a familiarity of the user with command line tools similar to other NeRF frameworks. Further limitations are inherited from relying on existing NeRF technology. We discuss this in the following, especially when considering NeRF's application within the dynamic and demanding realms of the Industrial Metaverse.

NeRFs have demonstrated remarkable proficiency in rendering static scenes, showcasing an ability to capture intricate details with high fidelity. The performance of NeRFs in representing static scenes is impressive, but standard NeRFs reach their limits when it comes to dynamic environments. In such scenarios, objects may move or change, requiring time-dependent reconstruction beyond the capabilities of the original NeRFs and their improvements. There is much ongoing research focused on filling this gap, particularly through the use of MLP-based methods. A detailed overview of these methods can be found in Section 4.3 of the article "Advances in Neural Rendering" [TTM*22]. By incorporating dynamic aspects, NeRFs could be used in a wider range of applications, from real-time entertainment to complex simulations in industry. The NeRF algorithm also lacks integrated physics for simulating the behavior of models in industrial settings. Without this, NeRF remains a somewhat superficial tool for visual representation but not for behavioral simulation.

Another feature that traditional (digital) 3D tools offer is the possibility to easily modify the scene, e.g. add/modify objects or change certain colors. For NeRFs this is inherently complicated caused by the significant computational intensity required due to the dense radiance field representation. Tools like NeRFShop and NeuralEditor are pioneering solutions to this issue, showcasing the potential for further advancements [JKK*23, CLW23].

Moreover, the lack of standardization and fragmentation within NeRF developments presents hurdles to widespread adoption. Current models' specificity to particular tasks complicates the creation of a unified platform. Additionally, the opaque nature of neural networks underlying NeRFs poses challenges for fine-tuning and integration across diverse models. However, new approaches such as Gaussian Splatting are emerging, leveraging classical machine learning techniques to generate new views more efficiently [KKLD23].

The integration of NeRF with widely used game engines and virtual environments, including Unity, Omniverse, and Unreal Engine, as well as the process of converting NeRF's output into usable meshes, remains a complex challenge. These integration issues highlight the necessity for ongoing research and development efforts aimed at enhancing NeRF's compatibility and functionality across different digital creation platforms.

Despite these limitations, the potential for reality capture through neural radiance fields remains significant. The ongoing research in this domain points to the development of increasingly efficient and versatile methods. Adaptations that integrate dynamic scenarios, real-time interactivity, and physics-based simulation are not beyond reach, and several works in the current literature are already advancing in these directions. The fast-paced growth of research in this area indicates that many of the challenges presented in this section could be overcome in the near future.

7 CONCLUSION

In this work, we have developed a framework to make training NeRFs from user content more accessible. We overcome key challenges by providing an easy to use and system-independent setup. Our approach makes NeRF accessible to a broader audience, including non-developers in various industries, democratizing this advanced 3D rendering technology.

An important next step is to validate this claim in applied user studies. We envision a comparison of the applicability of different NeRF creation tools (e.g. ours but also research code on GitHub as well as commercially available solutions) in different scenarios. At the same time, user studies should be conducted to evaluate preferences for different audiences.

Finally, by enhancing NeRF's approachability, we open up industrial applications like virtual prototyping and immersive training, transforming virtual engagement and fostering innovation. Looking ahead, integrating NeRF with the Industrial Metaverse promises to merge physical and digital worlds. It allows powerful visualization and interaction tools that could revolutionize industrial operations.

8 REFERENCES

[Bha22] BHALGAT Y.: Hashnerf-pytorch. https://github.com/yashbhalgat/HashNeRF-pytorch/, 2022.

[CLW23] CHEN J.-K., LYU J., WANG Y.-X.: NeuralEditor: Editing Neural Radiance Fields via Manipulating Point Clouds. In *CVPR* (2023).

[GGH*22] GAO K., GAO Y., HE H., LU D., XU L., LI J.: NeRF: Neural Radiance Field in 3D Vision, A Comprehensive Review, Nov. 2022. arXiv:2210.00379 [cs].

[HLY*20] HU S.-M., LIANG D., YANG G.-Y., YANG G.-W., ZHOU W.-Y.: Jittor: a novel deep learning framework with meta-operators and unified graph execution. *Science China Information Sciences 63*, 222103 (2020), 1–21.

[JKK*23] JAMBON C., KERBL B., KOPANAS G., DIOLATZIS S., LEIMKÜHLER T., DRETTAKIS G.: NeRFshop: Interactive Editing of Neural Radiance Fields". *Proceedings of the ACM on Computer Graphics and Interactive Techniques 6*, 1 (5 2023).

[KKLD23] KERBL B., KOPANAS G., LEIMKÜHLER T., DRETTAKIS G.: 3d gaussian splatting for real-time radiance field rendering. *ACM Transactions on Graphics 42*, 4 (2023).

[LC87] LORENSEN W. E., CLINE H. E.: Marching cubes: A high resolution 3D surface construction algorithm. *ACM SIGGRAPH Computer Graphics 21*, 4 (1987), 163–169.

[MESK22] MÜLLER T., EVANS A., SCHIED C., KELLER A.: Instant neural graphics primitives with a multiresolution hash encoding. *ACM Trans. Graph. 41*, 4 (July 2022), 102:1–102:15.

[MST*20] MILDENHALL B., SRINIVASAN P., TANCIK M., BARRON J., RAMAMOORTHI R., NG R.: NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis, 2020. arXiv:2003.08934 [cs].

[RBA*19] RAHAMAN N., BARATIN A., ARPIT D., DRAXLER F., LIN M., HAMPRECHT F. A., BENGIO Y., COURVILLE A.: On the spectral bias of neural networks, 2019.

[SF16] SCHÖNBERGER J. L., FRAHM J.-M.: Structure-from-motion revisited. In *Conference on Computer Vision and Pattern Recognition (CVPR)* (2016).

[SZPF16] SCHÖNBERGER J. L., ZHENG E., POLLEFEYS M., FRAHM J.-M.: Pixelwise view selection for unstructured multi-view stereo. In *European Conference on Computer Vision (ECCV)* (2016).

[TCY*22] TANCIK M., CASSER V., YAN X., PRADHAN S., MILDENHALL B., SRINIVASAN P. P., BARRON J. T., KRETZSCHMAR H.: Block-NeRF: Scalable Large Scene Neural View Synthesis, 2022.

[TPT*22] TAKIKAWA T., PEREL O., TSANG C. F., LOOP C., LITALIEN J., TREMBLAY J., FIDLER S., SHUGRINA M.: Kaolin Wisp: A PyTorch Library and Engine for Neural Fields Research. https://github.com/NVIDIAGameWorks/kaolin-wisp, 2022.

[TTM*22] TEWARI A., THIES J., MILDENHALL B., SRINIVASAN P., TRETSCHK E., YIFAN W., LASSNER C., SITZMANN V., MARTIN-BRUALLA R., LOMBARDI S., SIMON T., THEOBALT C., NIESSNER M., BARRON J. T., WETZSTEIN G., ZOLLHOEFER M., GOLYANIK V.: Advances in neural rendering. *Computer Graphics Forum 41*, 2 (2022), 703–735.

[TWN*23] TANCIK M., WEBER E., NG E., LI R., YI B., WANG T., KRISTOFFERSEN A., AUSTIN J., SALAHI K., AHUJA A., MCALLISTER D., KERR J., KANAZAWA A.: Nerfstudio: A Modular Framework for Neural Radiance Field Development. In *Special Interest Group on Computer Graphics and Interactive Techniques Conference Conference Proceedings* (2023), ACM.

[XC22] XRNERF-CONTRIBUTORS: Openxrlab neural radiance field toolbox and benchmark. https://github.com/openxrlab/xrnerf, 2022.

[YC20] YEN-CHEN L.: Nerf-pytorch. https://github.com/yenchenlin/nerf-pytorch/, 2020.

[YL22] YUE LUO Y.-P. C.: arcnerf: nerf-based object/scene rendering and extraction framework, 2022.

[YSK22] YOONWOO J., SEUNGJOO S., KIBAEK P.: Kakaobrain/nerf-factory: An awesome pytorch nerf library, 2022.

[ZSD*21] ZHANG X., SRINIVASAN P. P., DENG B., DEBEVEC P., FREEMAN W. T., BARRON J. T.: NeRFactor: neural factorization of shape and reflectance under an unknown illumination. *ACM Transactions on Graphics 40*, 6 (2021), 1–18.