Neural Scanning: Rendering and determining geometry of household objects using Neural Radiance Fields

Floris Erich[1], Baptiste Bourreau[1], Chun Kwang Tan[1], Guillaume Caron[2], Yusuke Yoshiyasu[2], Noriaki Ando[2]

***Abstract*— In this paper we present a hardware and software framework for Neural Scanning of household objects using Neural Radiance Fields (NeRF). NeRF learn a probabilistic representation of radiance and density, thus they can be used to render objects and to export objects' geometry. Our framework allows for easy scanning of the objects by rotating the object while using cameras in a static position. The objects we scan are mostly taken from the Yale-CMU-Berkeley (YCB) object set, and we release our scans as part of a public dataset. Supplementary URL: `https://robocip-aist.github.io/ycb_nerf_scans`.**

I. INTRODUCTION

In recent years, the Neural Radiance Fields (NeRF) technique has shown state of the art results on rendering 3D volumes [1]. Neural radiance fields represent a volume as a function that maps position and viewing direction to radiance (RGB) and density. Recent research has directly applied NeRF for 6-degrees-of-freedom (6DOF) object localization and manipulation with a robot arm [2, 3]. NeRF can also be combined with classical meshing techniques such as Marching Cubes in order to generate 3D meshes [4]. The main challenge for using NeRF for Neural Scanning is that NeRF assumes the camera is moved around the scene, while in our case, in order to allow for easy scanning, we want to rotate the object without moving the cameras.

In this paper we discuss a hardware framework, a software framework and a scanning process for creating Neural Scans of household objects, mostly originating from the YCB Object Set [5, 6]. We release our datasets and source code as open source software to facilitate future research.

II. BACKGROUND

Spatial representations of objects are useful for various tasks, including robot manipulation. There are various types of spatial representation, including 3D point clouds and meshes. However, in this paper we focus on representing objects as a neural volume. Neural volumes are a view synthesis technique, allowing for the creation of views of an object from positions and angles that were not originally captured. The rendering quality of neural volumes is typically higher compared to techniques that require an intermediate representation such as a point cloud or mesh, because novel views are directly synthesized from previous views.

[1]Equal contribution.

[2]Equal supervision.

All authors are with National Institute of Advanced Industrial Science and Technology, Japan. G. Caron is also with Centre National de la Recherche Scientifique (CNRS-AIST Joint Robotics Lab, IRL) and Universite de Picardie Jules Verne (MIS lab), France.

NeRF [1] is a view synthesis technique that models the problem as learning a function that maps position and viewing direction to radiance and density, significantly improving qualitatively and quantitatively upon previous state of the art works such as Scene Representation Networks (SRN) [7], Neural Volumes (NV) [8] and Local Light Field Fusion (LLFF) [9]. Commonly used quantitative methods to measure image generation quality are Peak Signal-to-Noise Ratio (PSNR), Structural Similarity Index Measure (SSIM) [10] and Learned Perceptual Image Patch Similarity (LPIPS) [11].

Some studies directly use neural volumes to estimate the 6DOF position of objects in real scenes or keypoints for manipulation [2]. One of the limitations of the original implementation of NeRF is the long training times required, often taking hours to days to train, but recently many research results have enabled near instantaneous training and rendering [12]. Neural volume techniques have also been applied before to directly obtain a 3D mesh of objects found "in the wild", however the result of these studies are still of low geometry and texture quality [13]. While NeRF generates high quality renderings of objects, mesh generation can be optimized by using neural signed distance functions [14].

Other works directly use an RGBD sensor to produce a textured mesh of objects [15, 16, 17]. Such methods can produce fairly good results, however suffer from the limitations of RGBD sensors, such as noisy depth estimates and low spatial resolution. Other works have focused on creating object datasets using neural rendering techniques, such as Common Objects in 3D (CO3D) [18]. Unlike CO3D, our work uses a hardware framework that can be used to collect input images without manually moving around the object. In addition, our work reuses the YCB object set [6], which advantage is that it can be purchased and used directly in robotics experiments.

Because our cameras are simple DSLR cameras, we consider this rig to be a passive scanning system, in contrast to active scanning systems that use visible or invisible light projection, such as the Google Scanner [19], RGBD sensor based scanning [15, 16, 17] and various commercial scanners. The main benefit of using passive scanning is that the system is simpler to reproduce. Compared to commercial scanners, our system allows more control over the scanning process, but the system comes at a similar cost and produces lower quality geometry than current commercial alternatives.

A closely related technique to NeRF is photogrammetry, which is also a passive method for constructing a virtual representation of a scene. Indeed, the photogrammetry process and NeRF share the same first step, to estimate the camera

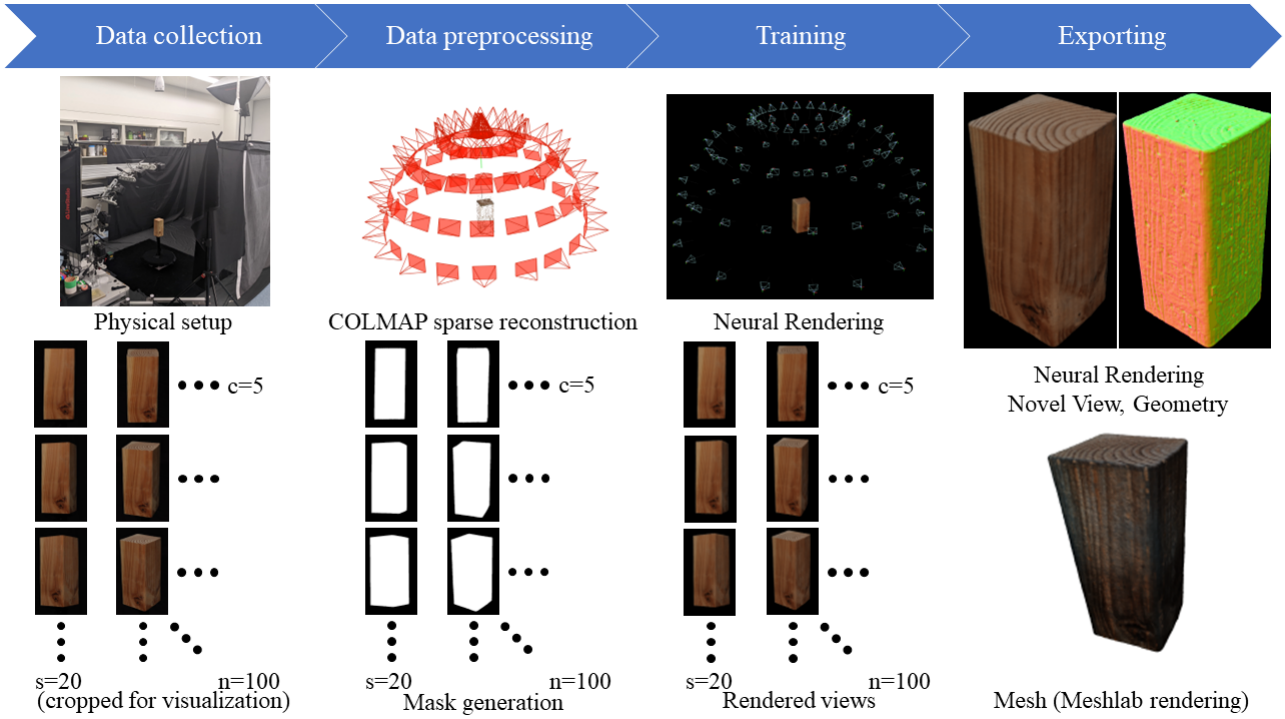


Fig. 1. Overview of our neural scanning framework. Data collection column shows our physical setup and an example of data collected of the YCB log. Data preprocessing column shows the result of intrinsic and extrinsic camera calibration using the YCB log and the result of running our mask generation tool. Training column shows the Neural Rendering of the combination of masks, original photos, camera alignment and shows the rendered views from the perspective of the training cameras. Exporting column shows a novel view of the neural scan, visualization of the surface normals (geometry), and a textured mesh exported through the neural scanning process.

intrinsics and extrinsics using multi-view geometry [20]. The main benefit of our Neural Scanning framework compared photogrammetry is the speed at which we can collect object geometry, with photogrammetry this takes minutes to hours, while our neural scanner can produce decent geometry within seconds.

III. METHOD

Our neural scanning framework consists of a hardware framework, a software framework and a scanning process. Our hardware framework consists of a five-camera hemispherical camera rig, a motorized rotation table, studio lights and background cloths and curtains. Our software framework consists of picture preprocessing scripts and integration scripts with *NVIDIA Instant neural graphics primitives (instant-ngp)* [12] for rendering and exporting 3D meshes. Fig. 1 gives an overview of our neural scanning framework. Our scanning process consists of four phases:

1) Data collection: Automated collection of hemispheric photographs.
2) Data preprocessing: COLMAP dense reconstruction and mask generation.
3) Training: Neural rendering and training using instant-ngp.
4) Exporting: Exporting a mesh using marching cubes.

A. Data collection

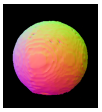
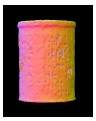
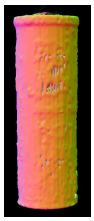
In the data collection phase, we collect high resolution color images of the object that is being scanned. Objects are placed on a rotation table (Ortery PhotoScan 360) with an extension pole attached to the center. We use five cameras (CANON EOS 90D) pointed towards the object in a 90 degree vertical arc, regularly spaced (using Ortery 3D MultiArm 2000). Objects are lit from the front and the back side using Ortery LiveStudio lights. We use Ortery Capture to control the cameras, rotation table and lights. Our hardware setup is similar to BigBIRD, except that we only use DSLR cameras, without depth sensors [5].

B. Data preprocessing

In the data preprocessing phase, we generate masks for the collected images and assign the camera position to each image. Masks are generated by manually selecting a rough region of interest for each camera in which the object is located and selecting a binary threshold value on the grayscale version of the images for filtering out background (black) pixels. The above process causes holes to appear in the masks for areas on the scanned objects with black texture patches, to remove these holes we apply contour selection for non-convex objects, and convex hull for convex objects. COLMAP is used for determining the camera position per image [21, 22]. For objects in which COLMAP alignment fails (e.g. due to lack of features on the object or repetitive patterns), we reuse camera positions previously determined

TABLE I

COMPARISON OF NEURAL SCANNING RESULTS OF YCB OBJECTS, OUR NEURAL SCANS VERSUS ORIGINAL YCB SCANS. ZOOM IN FOR MORE DETAILS.

Label	Ours (Rendering)	Ours (Geometry)	YCB (Rendering)	YCB (Geometry)	Label	Ours (Rendering)	Ours (Geometry)	YCB (Rendering)	YCB (Geometry)
Baseball					Bowl				
Coffee					Cheezit				
Drill					Hammer				
Log					Pringles				
Soft Scrub					Windex				

using a calibration object (an object that does not exhibit any problematic aspects, we typically use the YCB log for this). In our experience, using camera alignments from COLMAP is more accurate than using explicit camera alignments generated from an AR marker.

C. Training

In the training phase, we train a fully fused multi-layer perceptron (MLP) [23] to render views of the neural volume based on the predetermined camera positions. Depending on the object, some fine tuning is performed by also training the camera intrinsics and extrinsics. We use *instant-ngp* for training and previewing the neural scan [12]. We have added some modifications to *instant-ngp*, such as adding keys for stepping through each training camera and exporting training camera positions and rotations for animation, these changes have been merged with the project[1].

[1]Our fork: `https://github.com/FlorisE/instant-ngp`. Main project: `https://github.com/NVlabs/instant-ngp`.

D. Exporting

In the exporting phase, we export the trained weights of the MLP and export a mesh of the object. The marching cubes algorithm is used for generating the mesh geometry [4]. The radiance of the neural rendering is sampled for texturing the mesh, however the mesh texture quality is lower than the neural rendering quality, as the mesh texture is not directly sampled from training views. In future works we will explore how to improve the texture quality of exported meshes.

IV. RESULTS AND DISCUSSION

In this section we show qualitative results of using Neural Scanning. First, we show the result of scanning various items from the YCB object set. Second we show the results of neural scanning on a small set of objects of types commonly found in Japan. For each dataset we show the neural rendering and mesh geometry of the objects, as well as some specific data from each dataset.

We tested our neural scanner using the following NVIDIA GPU's: RTX 3060Ti, RTX 3090, RTX A5000. GPU memory

requirements depend on the resolution of the input images. RTX 3060Ti is equipped with 8GB of GPU memory, which is the least of the GPU's tested. It was possible to load 100 images with $W \times H$ of 1920×1080 on the RTX 3060Ti.

A. YCB Object Set Neural Scans

In Table I we show some examples of neural scans of objects from the YCB set, in comparison with the original YCB models. We aimed to test neural scanning on a variety of objects from the YCB set. In addition to the neural rendering and mesh geometry of the objects, we show the mesh rendering and mesh geometry of the scans (Poisson version) in the original YCB dataset using Meshlab [24]. Even though we used the official YCB object set, some items do not have the exact same appearance compared to the original scans. Each rendering was trained on 80 images (20 rotation steps, 4 cameras) for 5000 training steps[2]. More objects can be found in the supplemental material.

Our Neural Scanner can easily scan objects that exhibit Lambertian reflectance such as the coffee can and Soft Scrub. In some cases, our Neural Scanner produces better looking results than contained in the YCB model set, for example in the case of Soft Scrub, where part of the cap is missing in the YCB model, whereas our method succeeds in capturing this part, or Windex, in which our method produces more geometry for transparent areas.

Our method has some problems with specular reflectance, as is demonstrated by Bowl, Cheezit, Hammer and Windex. For Bowl, the geometry determined has some noise. For Cheezit, Hammer and Windex, specular reflectance is reproduced when rendering the neural scan. In a traditional photogrammetry setup, polarization properties of light would be used to remove specular reflectance. In our setup it is not possible to apply polarization to every camera as we have only two light sources and take pictures simultaneously. Applying a polarizing filter to only a single camera would lead to the lighting to be significantly different between this camera and the other cameras, causing a sudden change in lighting conditions between different perspectives of the neural rendering. Instead, algorithms such as a Joint Bilateral Filter could be used to reduce specular reflectance, however we leave the further exploration of this to future work.

The neural scan of Windex is much more complete than the original YCB model, even though the transparent geometry has noise. It should be noted that we scanned the emptied spray bottle, whereas the original YCB model was scanned with liquid in the bottle. One might wonder how the transparency is colored, so we have included a separate rendering of Windex with the rendering background set to white, see Fig. 2.

For dark objects, the main problem is separating areas of the object from the black background color. In case of the drill, our mask generation tool was unable to create masks to separate the drill from the background due to the object having black parts. To compensate for this, we scanned the drill using a white background instead. Due to the lack of active lighting, our Neural Scanner seems susceptible to the same kinds of visual issues as other passive scanning techniques such as photogrammetry are, e.g. transparency, reflectiveness, dark and textureless areas.

[2]We did not use the fifth camera beyond data collection, as it was hard to estimate its position. The object rotates in a plane parallel to the camera and the camera depth axis is similar to the rotation axis, which is a case in which multi-view stereo algorithms, used to determine the camera extrinsic parameters, often break down.

Fig. 2. Windex rendering and geometry with white rendering background (and black scanning background) shows how scanning background color is assigned to transparent surfaces.

B. Japanese Household Objects Neural Scans

In Table II we show some examples of neural scans of a small set of objects that can be found in Japanese offices and homes, and in Table III we calculate the Hausdorff distance bidirectionally between the meshes. We compare the results captured using our Neural Scanner – *Ours (Rendering)* and *Ours (Geometry)*, a recent RGBD-sensor-based scanner [17] – *Previous (P)* – and a commercial 3D scanner – Artec Space Spider (*GT*). Note that the Sparkling Lemon bottle was spray painted to ensure it was scannable, without spray paint none of the scanners could obtain accurate geometry. In the direction from *GT*, *Ours* has the lowest error, but in the direction to *GT*, *Previous* actually got a lower error. This is caused by some geometry being present inside *Ours*, whereas *Previous* and *GT* are hollow meshes. In addition to the neural rendering and mesh geometry of the objects, we show the colored mesh of the objects obtained using *Previous*, and mesh rendering and mesh geometry from the commercial 3D scanner. As can be seen from Table II, our neural scanner is clearly superior to *Previous*, while being slightly inferior to the commercial 3D scanner. *Previous* produced a mushroom effect on objects such as Cup Noodle, which our new system does not suffer from. In the case of the mug, *Ours (Geometry)* includes some extra particles on the inside of mug, which seems to originate from a lack of texture details (the mug is white from inside). *Previous* had a lot of issues with the mug due to the thinness of the walls of the mug, as well as the thinness of the ear. *Previous* also had trouble with shiny, textureless surfaces, such as Tissues, whereas our new system can handle this object well.

One benefit of *Previous* was the total cost of the system, only requiring a rotation table, a commercial-of-the-shelf

TABLE II

QUALITATIVE COMPARISON ON OBJECTS THAT WE HAVE SCANNED IN A PREVIOUS WORK. ZOOM IN FOR MORE DETAILS.

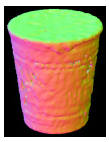
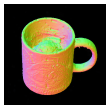
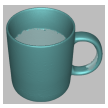
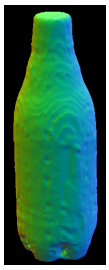
Label	Ours (Rendering)	Ours (Geometry)	Previous (RGBD)	GT (Rendering)	GT (Geometry)
Cup Noodle					
Mug					
Sparkling Lemon					
Tissues					

TABLE III

QUANTITATIVE COMPARISON ON OBJECTS THAT WE HAVE SCANNED IN A PREVIOUS WORK. MEAN ERROR IN CM.

Label	GT—Ours	GT—P	Ours—GT	P—GT
CupNoodle	0.11	0.25	0.83	0.19
Mug	0.16	0.47	0.75	0.52
SparklingLemon	0.07	0.34	0.59	0.23
Tissues	0.14	0.49	1.15	0.25

RGBD sensor and optionally a photo studio box. The total cost of the Neural Scanner presented in this paper is multiple times this cost, as in addition to the turntable used in *Previous*, we use five DSLR cameras, a mechanical structure for attaching the cameras, a camera multiplexer and studio lighting setup.

Another limitation of the neural scanner is that the mesh texture quality is not very high, as could be seen in the bottom right mesh rendering in Fig 1. While this is a significant limitation when the goal is to create an output mesh, this does not limit the usage of Neural Radiance Fields for viewing objects or even for manipulation [2, 3]. In this paper we have used the rendering by *instant-ngp* for *Ours (Rendering)*, as this is the common mode in which we expect neural scans to be used. Improving the texture quality of exported meshes is left as a future work.

V. CONCLUSIONS

In this paper we have shown that photographs captured using a setup consisting of DSLR cameras, a rotation table and studio lighting setup can be used to collect the training dataset for generating neural radiance fields. The neural radiance fields can subsequently be used to achieve high quality scanning of objects, almost rivalling commercial active 3D scanning technologies. In the future we want to improve our scanning system and results by combining high resolution camera images and depth maps captured using an additional depth sensor. We also want to further explore how to scan objects that exhibit problematic aspects, such as lack of texture, reflection/transparency, etc.

ACKNOWLEDGMENT

This research is subsidized by New Energy and Industrial Technology Development Organization (NEDO) under a project JPNP20016. This paper is one of the achievements of joint research with and is jointly owned copyrighted material of ROBOT Industrial Basic Technology Collaborative Innovation Partnership.

REFERENCES

[1] Ben Mildenhall et al. “NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis”. In: *European Conference on Computer Vision (ECCV)*. 2020 (cit. on p. 1).

[2] Lin Yen-Chen et al. “NeRF-Supervision: Learning Dense Object Descriptors from Neural Radiance Fields”. In: *IEEE Conference on Robotics and Automation (ICRA)*. 2022 (cit. on pp. 1, 5).

[3] Jeffrey Ichnowski et al. “Dex-NeRF: Using a Neural Radiance field to Grasp Transparent Objects”. In: *Conference on Robot Learning (CoRL)*. 2020 (cit. on pp. 1, 5).

[4] William E. Lorensen and Harvey E. Cline. “Marching cubes: A high resolution 3D surface construction algorithm”. In: *ACM SIGGRAPH Computer Graphics* 21.4 (Aug. 1987), pp. 163–169. ISSN: 0097-8930 (cit. on pp. 1, 3).

[5] Arjun Singh et al. “BigBIRD: A large-scale 3D database of object instances”. In: *2014 IEEE International Conference on Robotics and Automation (ICRA)*. 2014, pp. 509–516. DOI: 10.1109/ICRA.2014.6906903 (cit. on pp. 1, 2).

[6] Berk Calli et al. “The YCB object and Model set: Towards common benchmarks for manipulation research”. In: *2015 International Conference on Advanced Robotics (ICAR)*. Istanbul, Turkey: IEEE, July 2015, pp. 510–517. ISBN: 978-1-4673-7509-2. DOI: 10.1109/ICAR.2015.7251504 (cit. on p. 1).

[7] Vincent Sitzmann, Michael Zollhoefer, and Gordon Wetzstein. “Scene Representation Networks: Continuous 3D-Structure-Aware Neural Scene Representations”. In: *Advances in Neural Information Processing Systems*. Vol. 32. Curran Associates, Inc., 2019 (cit. on p. 1).

[8] Stephen Lombardi et al. “Neural Volumes: Learning Dynamic Renderable Volumes from Images”. In: *ACM Transactions on Graphics* 38.4 (July 2019), pp. 1–14. ISSN: 0730-0301, 1557-7368. DOI: 10.1145/3306346.3323020 (cit. on p. 1).

[9] Ben Mildenhall et al. “Local Light Field Fusion: Practical View Synthesis with Prescriptive Sampling Guidelines”. In: *ACM Transactions on Graphics (TOG)* (2019) (cit. on p. 1).

[10] Zhou Wang et al. “Image quality assessment: from error visibility to structural similarity”. In: *IEEE Transactions on Image Processing* 13.4 (2004), pp. 600–612. DOI: 10.1109/TIP.2003.819861 (cit. on p. 1).

[11] Richard Zhang et al. “The Unreasonable Effectiveness of Deep Features as a Perceptual Metric”. In: *CVPR*. 2018 (cit. on p. 1).

[12] Thomas Müller et al. “Instant Neural Graphics Primitives with a Multiresolution Hash Encoding”. In: *ACM Trans. Graph.* 41.4 (July 2022), 102:1–102:15. DOI: 10.1145/3528223.3530127 (cit. on pp. 1–3).

[13] Mark Boss et al. *SAMURAI: Shape And Material from Unconstrained Real-world Arbitrary Image collections*. Tech. rep. arXiv:2205.15768. May 2022 (cit. on p. 1).

[14] Jacob Munkberg et al. “Extracting Triangular 3D Models, Materials, and Lighting From Images”. In: *CVPR* (2022), p. 21 (cit. on p. 1).

[15] Richard A. Newcombe et al. “KinectFusion: Real-time dense surface mapping and tracking”. In: *2011 10th IEEE International Symposium on Mixed and Augmented Reality*. IEEE, Oct. 2011, pp. 127–136. ISBN: 978-1-4577-2183-0 978-1-4577-2185-4. DOI: 10.1109/ISMAR.2011.6092378 (cit. on p. 1).

[16] Rohan P. Singh et al. “Rapid Pose Label Generation through Sparse Representation of Unknown Objects”. In: *2021 IEEE International Conference on Robotics and Automation (ICRA)*. 2021, pp. 10287–10293. DOI: 10.1109/ICRA48506.2021.9561277 (cit. on p. 1).

[17] Floris Erich and Noriaki Ando. “A Framework for 3D Scanning using RGB-D Cameras and an Automated Rotary Table”. In: *2022 IEEE/SICE International Symposium on System Integration (SII)*. IEEE. 2022, pp. 614–619 (cit. on pp. 1, 4).

[18] Jeremy Reizenstein et al. “Common Objects in 3D: Large-Scale Learning and Evaluation of Real-life 3D Category Reconstruction”. In: *2021 International Conference on Computer Vision (ICCV)*. 2021 (cit. on p. 1).

[19] Laura Downs et al. “Google Scanned Objects: A High-Quality Dataset of 3D Scanned Household Items”. In: *2022 International Conference on Robotics and Automation (ICRA)*. 2022, pp. 2553–2560. DOI: 10.1109/ICRA46639.2022.9811809 (cit. on p. 1).

[20] Richard Hartley and Andrew Zisserman. *Multiple View Geometry in Computer Vision*. 2nd ed. New York, NY, USA: Cambridge University Press, 2003. ISBN: 0521540518 (cit. on p. 2).

[21] Johannes Lutz Schönberger and Jan-Michael Frahm. “Structure-from-Motion Revisited”. In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016 (cit. on p. 2).

[22] Johannes Lutz Schönberger et al. “Pixelwise View Selection for Unstructured Multi-View Stereo”. In: *European Conference on Computer Vision (ECCV)*. 2016 (cit. on p. 2).

[23] Thomas Müller et al. “Real-time neural radiance caching for path tracing”. en. In: *ACM Transactions on Graphics* 40.4 (Aug. 2021), pp. 1–16. ISSN: 0730-0301, 1557-7368. DOI: 10.1145/3450626.3459812 (cit. on p. 3).

[24] Paolo Cignoni et al. “MeshLab: an Open-Source Mesh Processing Tool”. In: *Eurographics Italian Chapter Conference*. Ed. by Vittorio Scarano, Rosario De Chiara, and Ugo Erra. The Eurographics Association, 2008. ISBN: 978-3-905673-68-5. DOI: 10.2312/LocalChapterEvents/ItalChap/ItalianChapConf2008/129-136 (cit. on p. 4).