

NeuralMeshing: Complete Object Mesh Extraction from Casual Captures

Floris Erich^{*1}, Naoya Chiba², Abdullah Mustafa¹, Ryo Hanai¹,
Noriaki Ando¹, Yusuke Yoshiyasu¹ and Yukiyasu Domae¹

Abstract—How can we extract complete geometric models of objects that we encounter in our daily life, without having access to commercial 3D scanners? In this paper we present an automated system for generating geometric models of objects from two or more videos. Our system requires the specification of one known point in at least one frame of each video, which can be automatically determined using a fiducial marker such as a checkerboard or Augmented Reality (AR) marker. The remaining frames are automatically positioned in world space by using Structure-from-Motion techniques. By using multiple videos and merging results, a complete object mesh can be generated, without having to rely on hole filling. Code for our system is available from <https://github.com/FlorisE/NeuralMeshing>.

I. INTRODUCTION

Triangle meshes are used in various fields such as computer vision, computer graphics and robotics. They can be obtained through manual 3D modeling or through automation, known as 3D scanning. 3D scanning equipment is expensive, with prices ranging from thousands to tens of thousands of dollars. Open source software such as COLMAP and GLOMAP offer a free alternative, but they are complicated to use when the goal is to scan objects instead of environments.

We present a system that integrates various state-of-the-art techniques into an automated pipeline for generating a complete object mesh with realistic proportions. Our system only requires a video camera (e.g. smartphone or USB camera) and a surface to place the object and fiducial marker on. For best mesh generation results, a camera without autofocus and with a wide depth of field should be used. Fiducial markers can be printed on paper and are thus easy to create.

Our method allows for generating complete object models of objects with a diffuse texture. The method can also be applied on reflective or transparent objects, with some degree of success. Merging of multiple videos is achieved by the operator manually selecting matching points, however this requires that the object has at least three easily identifiable points shared between videos. Such points have unique geometry or colors allowing the operator to easily identify them in multiple videos.

We improve upon state-of-the-art methods such as NeuS2 [1] by integrating it in a pipeline for complete mesh

generation from video, as well as improving the texture quality. NeuS2 can automatically construct a neural representation from video data by integrating COLMAP and can export meshes, but the meshes will be unscaled and NeuS2 cannot merge multiple videos of an object. The texture quality of meshes exported using NeuS2 is often poor due to using an inaccurate camera direction when estimating mesh colors. By sampling from every direction, the logical direction vector for estimating the color for a vertex is opposite of the surface normal vector.

II. BACKGROUND

3D reconstruction from images is a core topic of the computer vision field, with various methods proposed to create mesh representations [2], [3] or novel view renderings [4]. Such techniques can be integrated with hardware systems such as rotation stages [5], camera rigs [6] or robots [7], however the cost of such equipment is excessive for non-commercial applications. Smartphone-based scanning applications have been released, but these have limitations such as requiring the smartphone to be equipped with a depth sensor or rely on AI techniques that do not generalize well to novel objects. Extremely sparse view mesh reconstruction generative AI methods have been released [8]–[10], but these do not reproduce the same amount of detail (such as readable text) as we aim for in this research. Dataset annotation tools such as HANDAL [11] or NeuralLabeling [12] also generate meshes, however these are often untextured or have incomplete geometry.

We present a cost-effective method for creating meshes by integrating marker-based scale estimation and state-of-the-art mesh generation techniques.

III. METHOD

An overview of each step in our method is shown in Fig. 2. Object meshes are defined as sets of vertices $V \in \mathbb{R}^3$, edges $E = V \times V$, faces $F = V \times V \times V$ and vertex colors $C \in \mathbb{R}^3$. Our method takes as input videos $X \in \{X_0, \dots, X_N\}$ consisting of frames $I_X \in \mathbb{R}^{H \times W \times 3 \times F}$, where W , H , F are the frame width, frame height and number of frames, respectively. A frame rate is specified by the operator during recording, such that there is enough overlap between frames, but not too many frames to significantly affect the processing time or to exhaust computational resources such as GPU memory. During recording, the operator moves the camera around the object carefully to avoid motion blur while capturing all the relevant sides of the object in its current position,

^{*}Corresponding author, reachable at floris.erich@aist.go.jp

¹Floris Erich, Abdullah Mustafa, Ryo Hanai, Noriaki Ando, Yusuke Yoshiyasu and Yukiyasu Domae are with the National Institute of Advanced Industrial Science and Technology (AIST), Japan.

²Naoya Chiba is with University of Osaka.

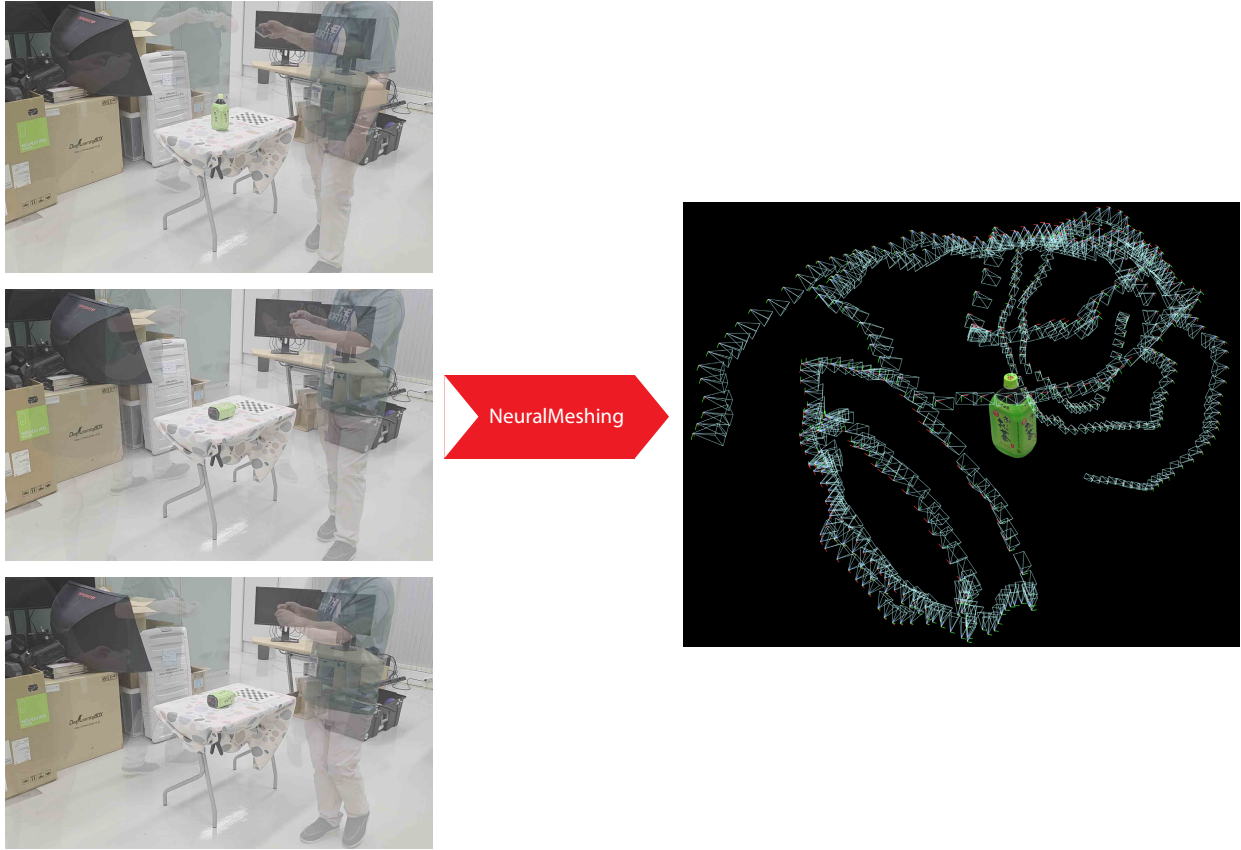


Fig. 1: Example of a neural reconstruction of a partially translucent object based on three videos (object standing upright, object placed on its back side and object placed on its front side).

while keeping the object stationary. Recording is repeated to capture sides that were occluded in other videos. For objects that can be easily flipped, two recordings suffice. For objects that cannot easily be flipped, it might be necessary to capture 3 videos or more of the object.

We use Structure-from-Motion (SfM) algorithms [2], [3] to estimate the camera intrinsics C_X and extrinsics T_X for all I_X , both being relative to a yet unknown scale factor S_X . We assume camera intrinsics are static within a video and enforce this in the SfM algorithms by detecting a single camera model for each video. We determine and store camera intrinsics using the OpenCV camera model, consisting of 9 parameters: $k_1, k_2, k_3, p_1, p_2, f_x, f_y, c_x, c_y$. We store camera extrinsics as a transformation matrix $T \in \mathbb{R}^{4 \times 4}$ consisting of a rotation matrix $T[:3, :3] = R$, a translation vector $T[:3, 3] = t$ and $T[3, :] = [0, 0, 0, 1]$.

To segment the object from its background we use Segment Anything 2 (SAM2) [13], [14] in bounding-box mode. The operator draws a 2D bounding box around the target object, after which the object is tracked through all video frames and produces $M \in \mathbb{Z}^{H \times W \times F}$. Broadcasting combines I_X and M to generate masked images $I_{X,M} \in \mathbb{R}^{H \times W \times 3 \times F}$.

We construct a neural field [4] by inputting $I_{X,M}$, C_X and

T_X using *instant-ngp* [15], which allows for fast training by using a multi-level hash encoding. A surface loss term is used in order to produce a neural field in which the density of the neural field matches the surface of the object [1]. We then extract a partial object mesh M_X by applying the marching cubes (MC) algorithm [16] on the neural field using a resolution of height H_{MC} , width W_{MC} and depth Z_{MC} . We use an implementation of NeuS2 that uses opposite surface normals as MLP query direction for determining vertex color.

For each video X , the operator selects in at least one frame one point for which the ground truth position $P_X \in \mathbb{R}^3$ is known. In practice, we place a marker and detect its position, and the operator only has to select which frames to use for marker detection. To determine the ground truth position, we use marker detection with the known marker properties such as radius for a circle or square side length for a checkerboard. To determine the position $P_{X,NeRF} \in \mathbb{R}^3$ of the marker in the reconstruction, which is subject to a scale factor, we render the neural volume with the determined camera intrinsics and use the extrinsics of the manually selected frames by the operator. We can now calculate the scale factor $S_X = \|P_X\| / \|P_{X,NeRF}\|$. In case the operator selected multiple frames to use for scaling, we simply repeat the scaling procedures for each selected frame and take the

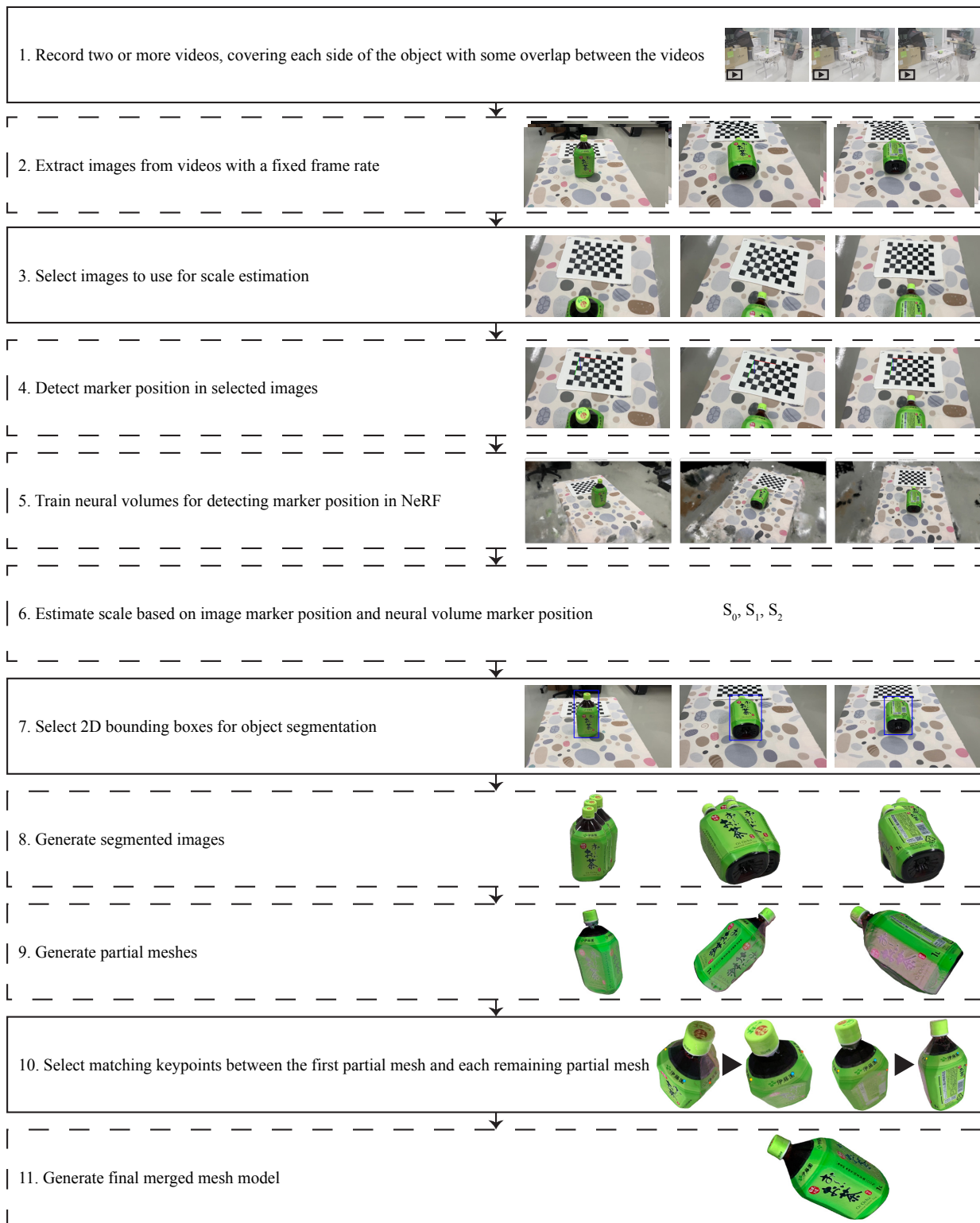


Fig. 2: Flowchart of steps executed to generate a mesh. Manual steps have a solid line while completely automated steps have a dashed line. Zoom in for more details.

mean of the individually determined scaling factors.

We scale the partial object meshes $M_{X,0}, \dots, M_{X,N}$ to a uniform scale. The operator then selects at least three matching points on each partial object mesh to perform coarse registration via point correspondences. After coarse registration, the system performs fine registration using Iterative Closest Points (ICP) [17]. This procedure results in the construction of a joint neural field. We repeat the same mesh extraction technique discussed earlier to produce the object mesh M with per-vertex coloring. Because the object meshes will have a large file size due to a high vertex count, techniques for mesh decimation and texture extraction can be automatically applied.

IV. EVALUATION

We recorded videos of various objects and extracted their meshes using an Apple iPhone SE. A frame rate of 4 frames per second was used. A checkerboard was used for automated scale estimation. Processing was done on a PC equipped with Intel Xeon Silver 4110, 48 GB of RAM and a NVIDIA RTX A6000 GPU. Our pipeline uses OpenCV [18] and Open3D [19] in various steps.

We report the quality of the exported meshes for five objects: A bottle of green tea (GreenTea), a cup of instant noodles (Noodles), a can of alcohol free beer (BeerCan), a bottle of mustard (Mustard) and a can of preground coffee (Coffee). We compare methods for extracting the final mesh, either using instant-ngp, NeuS2 or our implementation with opposite surface normal sampling. Error is calculated by re-rendering the extracted meshes and comparing with the original video frames, for 20 randomly selected frames, excluding frames in which the object was significantly out of frame. We apply the following metrics: Pixel rendering mean absolute error (MAE, lower is better), root mean square error (RMSE, lower is better) and Peak Signal to Noise Ratio (PSNR, higher is better). Table I shows the results of this experiment. Meshes extracted using NeuralMeshing have significantly higher similarity to the video frames, except for *Beer Can* in which the result was similar to *NeuS2*.

We report the time required for generating each mesh for the five objects above plus one additional object based on the video durations and processing time in Table II. The data was collected while the operator was allowed to multitask, an operator focusing on only the mesh generation process would be able to process the meshes faster. Processing time was always within an hour, 34 minutes on average.

For qualitative evaluation, we include an extra item that would be difficult to quantitatively evaluate due to the lack of ground truth data, namely a plastic bottle with an almost completely transparent label (FruitTea). Fig. 3 shows qualitative results of our method applied to the six objects. For each object we show the rendered mesh from a viewpoint matching a frame in two different videos used to generate the mesh. It can be seen that our method (*NeuralMeshing*) performs extremely well on three out of five objects that have limited to no reflectiveness (GreenTea, Noodles, Mustard). On reflective objects such as Coffee

(the bottom is reflective) and BeerCan, performance drops significantly for all methods, including NeuralMeshing. On FruitTea, the performance of NeuralMeshing and NeuS2 are similar, though both significantly outperform NeRF. FruitTea specifically shows the benefit of using neural fields based techniques for extracting meshes, as we can reasonably estimate the geometry of the object with all methods, even though automatically estimating transparency is still an open issue.

V. DISCUSSION AND CONCLUSION

One limitation of our system is that processing is done offline. It takes up to an hour to generate a mesh for an object, mostly due to training and retraining neural fields at various steps in the process. In the future we will try to reduce the processing time.

We also noticed only limited success with processing reflective or transparent objects, with errors occurring in the generated object geometry and texture. To improve these areas in the future, we plan to explore methods to preprocess the data to generate depth [20], normals, albedo and roughness [21] and methods for detecting the opacity of pixels [22], [23]. Dedicated techniques for metallic object mesh extraction using neural representations exist [24], however these typically require a few hours of training time and are thus infeasible to be integrated into our approach. In our quantitative evaluation we excluded the FruitTea object. Ground truth data for largely transparent objects cannot easily be obtained as transparent surfaces transmit the background color of the surface behind the transparent surface, which depends on the viewpoint, while triangular mesh vertex color is viewpoint independent. Even if we could encode a viewpoint dependent background color into the final mesh, it could be argued that it is more desirable to have a neutral color and a per vertex transparency value. Transparency estimation can be evaluated by re-rendering the object in the same environment but with the physical object removed, as we did in an earlier study [22], however such evaluation requires a more controlled environment (e.g. camera pose should be static).

We tested various cameras for recording the datasets. Cheap USB cameras such as Sanwa CMS-V45S struggle with high resolution image capture due to image warping. More expensive USB cameras such as Logitech BRIO might still struggle due to autofocus causing some images to become blurry. Best results were obtained with the Apple iPhone SE's camera and with cameras embedded in RealSense sensors.

Even though we used a checkerboard for automated scale estimation, we did not use it to input intrinsics in the Structure-from-Motion algorithm or for generating camera poses. The checkerboard is only visible in a subset of the frames, thus it is not possible to detect all the camera poses using it. Other markers such as AR markers or a ChArUco board might be easier to embed in each frame, which would allow skipping the SfM step.

TABLE I: Quantitatively evaluating our method compared to ablations

Method	Green Tea Bottle			Noodles			Beer Can			Mustard Bottle			Coffee Ground			Mean		
	MAE	RMSE	PSNR	MAE	RMSE	PSNR	MAE	RMSE	PSNR	MAE	RMSE	PSNR	MAE	RMSE	PSNR	MAE	RMSE	PSNR
NeRF	0.22	0.26	11.72	0.17	0.22	13.23	0.31	0.37	8.54	0.16	0.21	13.67	0.25	0.29	10.65	0.22	0.27	11.56
NeuS2	0.26	0.30	10.66	0.17	0.21	13.77	0.19	0.24	12.29	0.18	0.23	12.79	0.23	0.29	11.12	0.20	0.25	12.13
NeuralMeshing	0.10	0.14	17.27	0.09	0.13	18.28	0.17	0.22	13.12	0.07	0.10	20.46	0.14	0.19	14.86	0.11	0.16	16.80



Fig. 3: Qualitative results for 6 household objects captured using our NeuralMeshing system. Our method produces textures that more closely match the geometry and coloration of the ground truth data.

TABLE II: Video duration and processing time (from when the video data was copied to the final mesh being exported).

Object	Video durations (seconds)	Processing time (minutes)
GreenTea	39, 37, 40	59
Noodles	36, 36	28
Coffee	34, 34	26
Mustard	39, 39	26
BeerCan	35, 37	25
FruitTea	50, 51	41

In this paper we adopt NeuS2 for meshing, however there are other techniques for generating meshes from volumetric data such as nerf2mesh [25] and SuGaR [26]. In the future we will investigate integrating such alternative mesh generation components into our system.

Our pipeline heavily relies on COLMAP [2], [3] and GLOMAP [27] for estimating the camera poses. These methods use a multi-stage computational process to convert

images into camera poses. Recently various new machine learned methods for camera pose estimation have appeared, such as DUST3R [28] and VGGT [29]. DUST3R offers similar performance as COLMAP and GLOMAP, but can be used with significantly fewer cameras. However, as our end goal is mesh generation, having fewer frames negatively affects other steps such as neural field construction. VGGT can outperform COLMAP and GLOMAP in certain scenarios, so in the future we will try to embed it into our pipeline to evaluate its effectiveness for our application. MUST3R [30] extends DUST3R to enable rapid camera pose estimation for a large number of views, and is thus also interesting to consider as an alternative to VGGT.

Even though data collection can be done with low cost hardware, processing the data to generate a mesh still requires a powerful GPU. To make the system more widely available, future research could focus on introducing a cloud-based architecture which would allow for amortizing the

cost. In addition, evaluating the tradeoff between powerful hardware and model generation time and quality is an interesting future research direction.

ACKNOWLEDGMENT

This work was supported by JST [Moonshot R&D][Grant Number JPMJMS2031].

REFERENCES

- [1] Y. Wang, Q. Han, M. Habermann, K. Daniilidis, C. Theobalt, and L. Liu, "NeuS2: Fast learning of neural implicit surfaces for multi-view reconstruction," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023.
- [2] J. L. Schönberger, E. Zheng, M. Pollefeys, and J.-M. Frahm, "Pixel-wise view selection for unstructured multi-view stereo," in *European Conference on Computer Vision (ECCV)*, 2016.
- [3] J. L. Schönberger and J.-M. Frahm, "Structure-from-motion revisited," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.
- [4] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng, "NeRF: Representing scenes as neural radiance fields for view synthesis," *Communications of the ACM*, vol. 65, no. 1, pp. 99–106, Dec. 2021.
- [5] F. Erich and N. 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)*, Jan. 2022, pp. 614–619.
- [6] F. Erich, B. Bourreau, C. K. Tan, G. Caron, Y. Yoshiyasu, and N. Ando, "Neural Scanning: Rendering and determining geometry of household objects using Neural Radiance Fields," in *2023 IEEE/SICE International Symposium on System Integration (SII)*, 2023, pp. 1–6.
- [7] T. Qiu, Z. Ma, K. El-Refai, H. Shah, C. M. Kim, J. Kerr, and K. Goldberg, "Omni-Scan: Creating Visually-Accurate Digital Twin Object Models Using a Bimanual Robot with Handover and Gaussian Splat Merging," Aug. 2025.
- [8] J. Xiang, Z. Lv, S. Xu, Y. Deng, R. Wang, B. Zhang, D. Chen, X. Tong, and J. Yang, "Structured 3D Latents for Scalable and Versatile 3D Generation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2025, pp. 21 469–21 480.
- [9] R. Shi, X. Wei, C. Wang, and H. Su, "ZeroRF: Fast Sparse View 360° Reconstruction with Zero Pretraining," in *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. Seattle, WA, USA: IEEE, June 2024, pp. 21 114–21 124.
- [10] R. Liu, R. Wu, B. V. Hoorick, P. Tokmakov, S. Zakharov, and C. Vondrick, "Zero-1-to-3: Zero-shot one image to 3D object," *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 9264–9275, 2023. [Online]. Available: <https://api.semanticscholar.org/CorpusID:257631738>
- [11] A. Guo, B. Wen, J. Yuan, J. Tremblay, S. Tyree, J. Smith, and S. Birchfield, "HANDAL: A Dataset of Real-World Manipulable Object Categories with Pose Annotations, Affordances, and Reconstructions," in *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Detroit: IEEE, 2023.
- [12] F. Erich, N. Chiba, A. Mustafa, Y. Yoshiyasu, N. Ando, R. Hanai, and Y. Domae, "NeuralLabeling: A versatile toolset for labeling vision datasets using Neural Radiance Fields," in *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2024.
- [13] A. Kirillov, E. Mintun, N. Ravi, H. Mao, C. Rolland, L. Gustafson, T. Xiao, S. Whitehead, A. C. Berg, W.-Y. Lo, P. Dollár, and R. Girshick, "Segment Anything," in *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*. Paris, France: IEEE, Oct. 2023, pp. 3992–4003.
- [14] N. Ravi, V. Gabeur, Y.-T. Hu, R. Hu, C. Ryali, T. Ma, H. Khedr, R. Rädle, C. Rolland, L. Gustafson, E. Mintun, J. Pan, K. V. Alwala, N. Carion, C.-Y. Wu, R. Girshick, P. Dollár, and C. Feichtenhofer, "SAM 2: Segment Anything in Images and Videos," in *The Thirteenth International Conference on Learning Representations*, Jan. 2025. [Online]. Available: <https://openreview.net/forum?id=Ha6RTeWmD0>
- [15] T. Müller, A. Evans, C. Schied, and A. Keller, "Instant Neural Graphics Primitives with a Multiresolution Hash Encoding," *ACM Trans. Graph.*, vol. 41, no. 4, pp. 102:1–102:15, July 2022.
- [16] W. E. Lorensen and H. E. Cline, "Marching cubes: A high resolution 3D surface construction algorithm," in *Proceedings of the 14th Annual Conference on Computer Graphics and Interactive Techniques - SIGGRAPH '87*. ACM Press, 1987, pp. 163–169.
- [17] Z. Zhang, "Iterative point matching for registration of free-form curves and surfaces," *International Journal of Computer Vision*, vol. 13, no. 2, pp. 119–152, Oct. 1994.
- [18] A. Kaehler and G. Bradski, *Learning OpenCV 3: Computer Vision in C++ with the OpenCV Library*, 2nd ed. O'Reilly Media, 2014.
- [19] Q.-Y. Zhou, J. Park, and V. Koltun, "Open3D: A Modern Library for 3D Data Processing," Jan. 2018.
- [20] F. Erich, B. Leme, N. Ando, R. Hanai, and Y. Domae, "Learning depth completion of transparent objects using augmented unpaired data," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2023.
- [21] R. Liang, Z. Gojcic, H. Ling, J. Munkberg, J. Hasselgren, C.-H. Lin, J. Gao, A. Keller, N. Vijaykumar, S. Fidler, and Z. Wang, "Diffusion renderer: Neural inverse and forward rendering with video diffusion models," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2025, pp. 26 069–26 080.
- [22] F. Erich, J. Susgin, N. Ando, and Y. Yoshiyasu, "A Pipeline for Transparency Estimation of Glass and Plastic Bottle Images for Neural Scanning," in *2025 IEEE/SICE International Symposium on System Integration (SII)*, Jan. 2025, pp. 653–657.
- [23] J. Yao, X. Wang, S. Yang, and B. Wang, "ViTMatte: Boosting image matting with pre-trained plain vision transformers," *Information Fusion*, vol. 103, p. 102091, 2024.
- [24] Y. Liu, P. Wang, C. Lin, X. Long, J. Wang, L. Liu, T. Komura, and W. Wang, "NeRO: Neural geometry and BRDF reconstruction of reflective objects from multiview images," *ACM Trans. Graph.*, vol. 42, no. 4, July 2023.
- [25] J. Tang, H. Zhou, X. Chen, T. Hu, E. Ding, J. Wang, and G. Zeng, "Delicate Textured Mesh Recovery from NeRF via Adaptive Surface Refinement," in *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, Oct. 2023, pp. 17 693–17 703.
- [26] A. Guédon and V. Lepetit, "SuGaR: Surface-Aligned Gaussian Splatting for Efficient 3D Mesh Reconstruction and High-Quality Mesh Rendering," in *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. Seattle, WA, USA: IEEE, June 2024, pp. 5354–5363.
- [27] L. Pan, D. Baráth, M. Pollefeys, and J. L. Schönberger, "Global structure-from-motion revisited," in *Computer Vision – ECCV 2024*, A. Leonardis, E. Ricci, S. Roth, O. Russakovsky, T. Sattler, and G. Varol, Eds. Cham: Springer Nature Switzerland, 2025, pp. 58–77.
- [28] S. Wang, V. Leroy, Y. Cabon, B. Chidlovskii, and J. Revaud, "DUST3R: Geometric 3D vision made easy," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024.
- [29] J. Wang, M. Chen, N. Karaev, A. Vedaldi, C. Ruppert, and D. Novotny, "VGGT: Visual geometry grounded transformer," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2025.
- [30] Y. Cabon, L. Stoffl, L. Antsfeld, G. Csurka, B. Chidlovskii, J. Revaud, and V. Leroy, "MUST3R: Multi-view network for stereo 3D reconstruction," in *Proceedings of the Computer Vision and Pattern Recognition Conference (CVPR)*, June 2025, pp. 1050–1060.