2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 14-18, 2024. Abu Dhabi, UAE

NeuralLabeling: A versatile toolset for labeling vision datasets using Neural Radiance Fields

Floris Erich[1*], Naoya Chiba[2], Abdullah Mustafa[1], Yusuke Yoshiyasu[1],
Noriaki Ando[1], Ryo Hanai[1], Yukiyasu Domae[1]

***Abstract*— We present *NeuralLabeling*, a labeling approach and toolset for annotating 3D scenes using either bounding boxes or meshes and generating segmentation masks, affordance maps, 2D bounding boxes, 3D bounding boxes, 6DOF object poses, depth maps, and object meshes. *NeuralLabeling* uses Neural Radiance Fields (NeRF) as a renderer, allowing labeling to be performed using 3D spatial tools while incorporating geometric clues such as occlusions, relying only on images captured from multiple viewpoints as input. To demonstrate the applicability of *NeuralLabeling* to a practical problem in robotics, we added ground truth depth maps to 30000 frames of transparent object RGB and noisy depth maps of glasses placed in a dishwasher captured using an RGBD sensor, yielding the Dishwasher30k dataset. We show that training a simple deep neural network with supervision using the annotated depth maps yields a higher reconstruction performance than training with the previously applied weakly supervised approach. We also show how instance segmentation and depth completion datasets generated using NeuralLabeling can be incorporated into a robot application for grasping transparent objects placed in a dishwasher with an accuracy of 83.3%, compared to 16.3% without depth completion. Supplementary URI: https://florise.github.io/neural_labeling_web/.**

I. INTRODUCTION

Deep learning requires large datasets, which are time-intensive and expensive to create. There are various approaches to avoid this, such as using foundation models or weakly supervised training methods like cyclic adversarial learning [1]. However, despite being trained on massive datasets, foundation models such as Segment Anything [2] and CLIP [3] still rely on inference data to be similar to the training data, which is not always the case. Models trained using weakly supervised learning might outperform state-of-the-art models when the SOTA models are not trained on task-specific data, but their performance is lower than SOTA models evaluated on evaluation data more similar to their training data. Thus there is a need for tools that can support large dataset creation in a time-efficient low-cost manner. We hope to contribute to solving this problem by introducing a labeling tool for computer vision datasets that uses the power of Neural Radiance Fields (NeRF) [4] for photorealistic rendering and geometric understanding. Because 3D Vision can take advantage of 3D consistency, labels on a single scene can be applied to images from multiple viewpoints. This property works particularly well with photorealistic renderings such as NeRF, where richly annotated data with many views is available with only simple manual 3D labeling. This not only saves significant labeling time but is also useful in automatically generating a consistent dataset.

Fig. 1. *NeuralLabeling* supports two pipelines for labeling NeRFs: Bounding-box-based labeling for uncluttered scenes and mesh-based labeling for cluttered scenes.

Specialized labeling tools are essential for labeling vision datasets, and both academic researchers and commercial entities have released such tools. Most existing labeling tools (such as Segment Anything Labeling Tool [5] and Roboflow [6]) use single images and therefore require significant human effort to annotate long sequences, use sequential data but have no geometric understanding so they cannot be used for annotating 6DOF poses [7], or require depth data to obtain geometric information [8, 9, 10]. Our toolkit, *NeuralLabeling*, operates on sequences of images and can thus be used to more rapidly label large datasets. By using manual scaling and NeRF depth reconstruction [4], NeuralLabeling does not rely on input depth data except when used for generating datasets for depth completion tasks. Due to improvements in the training time of NeRFs [11], *NeuralLabeling* does not rely on slow dense mesh reconstruction and instead only requires camera pose estimation, which takes around an hour per scene of approximately 500 images, and could be further reduced by selecting key frames and interpolating camera poses between them [12] or avoided using NeRF recording applications such as NeRFCapture [13].

This paper has two main contributions: (1) We present *NeuralLabeling*, a novel labeling system that is deeply integrated into a NeRF-based photorealistic rendering system (Section III). (2) We construct the Dishwasher30k dataset, which can be used for NeRF-based transparent object depth completion research, and release it on our web page. Furthermore, we perform the following experiments to validate our approach: (1) We evaluate the accuracy of *NeuralLabeling* for generating transparent object datasets for depth completion (Section IV-A). (2) We evaluate the accuracy of

* Corresponding author, reachable at firstname.lastname@aist.go.jp.
[1]National Institute of Advanced Industrial Science and Technology, Tokyo, Japan.
[2]Tohoku University, Sendai, Japan.

TABLE I

COMPARING UNIQUE ASPECTS OF LABELING TOOLS. ALL TOOLS SUPPORT SEGMENTATION MASKS. NDR = NO INPUT DEPTH REQUIRED, G = GEOMETRY, M = MESH, 6D = 6DOF POSES, O = OCCLUSION MASKS, A = AFFORDANCE MAPS, OD = OBJECT DEPTH

Tool	Inputs	Selection		Outputs			
	NDR	G	M	6D	O	A	OD
ProgressLabeller [14]	✓	✗	✓	✓	✗	✗	✗
3D-DAT [15]	✓	✗	✓	✓	✗	✗	✗
Nerfing It [16]	✓	✓	✗	✗	✗	✗	✗
RapidPoseLabels [10]	✗	✓	✓	✓	✗	✗	✗
HANDAL [17]	✗	✓	✓	✓	✓	✓	✗
NeuralLabeling (Ours)	✓	✓	✓	✓	✓	✓	✓

NeuralLabeling for generating datasets for object segmentation, taking into account occlusions (Section IV-B). (3) We demonstrate how training a transparent object depth completion network using a dataset generated by *NeuralLabeling* leads to improved performance compared to unsupervised datasets (Section IV-C). (4) We show that networks trained using datasets generated by *NeuralLabeling* can be integrated into a robot manipulation system (Section IV-D).

II. BACKGROUND

A. Vision data labeling tools

NeuralLabeling was inspired by various recent tools for creating labeled datasets but qualitatively improves upon each of them. ProgressLabeller [14] is a state-of-the-art labeling tool that uses mesh alignment and posed camera images. RapidPoseLabels [10] is an RGBD-based labeling tool, allowing for labeling objects with pose annotations. Because it uses RGBD data as input it cannot be used if depth data is unavailable or unreliable. 3D-DAT [15] is a mesh-based labeling tool implemented as a Blender plugin. It uses NeRF for automated alignment of objects with NeRF geometry, but it requires meshes to be provided as input. It also does not support NeRF-to-mesh occlusions. Nerfing It [16] is a NeRF-based labeling tool, but it does not support mesh-based labeling. It also uses a vanilla NeRF implementation that is not optimized for speed, and thus requires long training times to prepare scenes for labeling. Table I compares *NeuralLabeling* with various state-of-the-art labeling tools.

Our work resembles the pipeline used for preparing the HANDAL dataset [17]. Their work uses a bi-methodical 3D-bounding-box-based and mesh-based labeling approach, similar to what we present in this paper. An advantage of HANDAL is that it also supports labeling dynamic scenes. An advantage of our tool is that it can be used to generate depth maps for transparent objects. NeuralLabeling can generate segmentation masks that can be used for training neural networks to perform object segmentation, whereas the HANDAL pipeline relies on segmentation masks generated using a pre-trained tracker [7]. Their work uses automatic scaling based on depth input, whereas our work relies on manual scaling using a scaling tool. Inspired by their work, we added an affordance labeling tool to *NeuralLabeling*.

NeuralLabeling enables the labeling of existing scenes using NeRF, however in the parallel work PEGASUS [18] we allow generating datasets by inserting objects into an existing scene and rendering them using 3D Gaussian Splatting. By inserting custom objects into a scene, a wider variety of object configurations can be generated, thus leading to more variety in the generated datasets. However, the PEGASUS renderer is unaware of scene-specific lighting, whereas for NeuralLabeling the objects inherit natural scene lighting.

B. Transparent Object Depth Completion

NeuralLabeling started as a tool to label transparent objects with accurate depth estimates to enable robots to estimate depth and shape of glasses and cups, without relying on expensive photorealistic simulations. Deep learning approaches have greatly contributed to solving the problem of transparent object depth completion [19], however most existing datasets consist of glasses placed in simple environments such as on tables and floors [19, 20, 21]. State-of-the-art pretrained models underperform when applied to more complex environments such as a dishwasher [22]. Weakly supervised training methods can outperform state-of-the-art supervised models, but still underperform compared to the performance of the state-of-the-art models on data that is more similar to their training data. We show that *NeuralLabeling* can be used to easily create supervised datasets for a complex environment such as a dishwasher, and that a network trained on such a supervised dataset can outperform a network trained on a weakly supervised dataset. Using *NeuralLabeling*, it took roughly one workweek to construct this dataset, which contains NeRFs, mesh models, alignment configurations of the meshes with the NeRF, generated depth, and generated segmentation masks. We release this dataset, which we name *Dishwasher30k*.

III. METHODOLOGY

We support labeling using either 3D bounding-boxes or meshes (Fig. 1). 3D-bounding-box-based labeling is useful when scenes are uncluttered and/or high quality object meshes for applying labels to the scene are not available. Mesh-based labeling is useful when scenes are cluttered or if we already have object meshes available. We support mesh extraction using bounding-boxes, which enables a novel pipeline where we obtain mesh models for objects placed in an uncluttered manner, and then reuse these models in a cluttered scene. Fig. 2 gives a more detailed overview of the combined labeling pipeline.

We aim to generate semantic segmentation masks, 2D and 3D bounding boxes, 6DOF object poses, depth maps and object meshes for each frame in a RGB image sequence (Fig. 3). Segmentation masks are further classified into binary, instance and class segmentation masks. 2D bounding boxes are defined by the lower left corner and upper right corner. 3D bounding boxes are defined by the lower left front corner, upper right back corner and an orientation. When

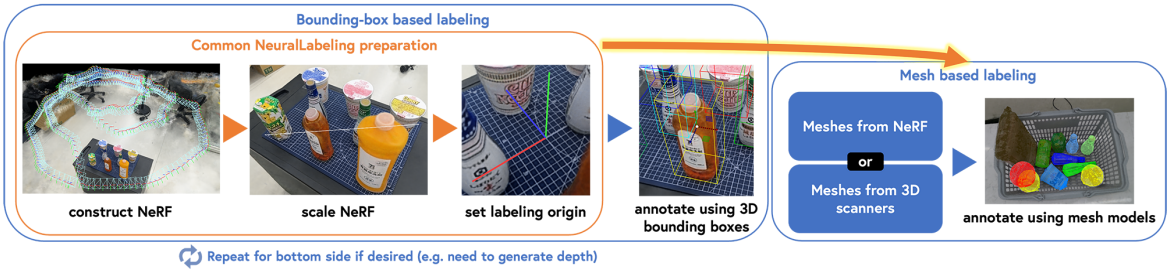


Fig. 2. A scene can be labeled using either bounding-boxes or using meshes. Bounding boxes can be used to extract meshes from a scene.

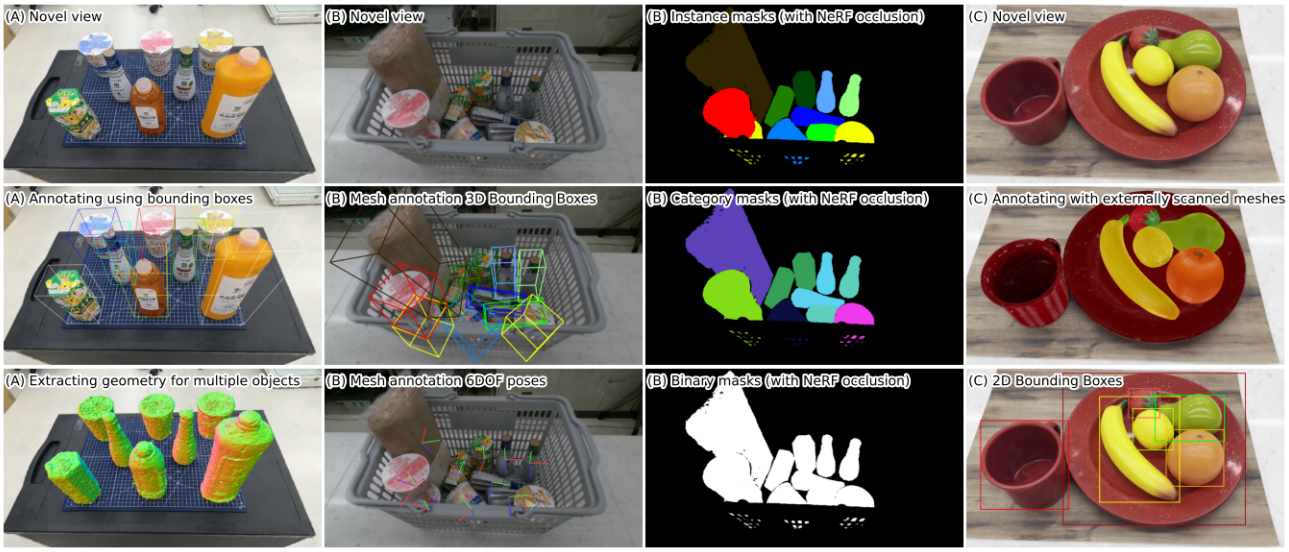


Fig. 3. *NeuralLabeling* supports a wide variety of outputs. Circled letter references the scene: (A) Mostly Lambertian objects placed upright for mesh extraction, second row shows the annotated bounding boxes, third row shows the geometry generated using the bounding boxes. (B) Most of the objects from (A) placed in a shopping basket and annotated using the meshes generated from (A), towel was captured separately, second row shows 3D bounding boxes based on the mesh annotations, third row shows 6DOF poses based on the mesh annotations. Second column of (B) shows instance masks, category masks and binary masks, each using NeRF-to-mesh occlusions rendered directly by *NeuralLabeling* to improve segmentation accuracy. (C) Lambertian objects placed on a lunch plate. We use YCB objects for which we use openly available meshes based on 3D scans using the Google Scanner, second row shows the meshes rendered directly in the scene, third row shows 2D bounding boxes generated based on mesh geometry.

using the 3D-bounding-box-based labeling workflow, we can either directly use the labeled bounding-boxes or we can optimize the bounding-boxes to tightly fit their geometry. 6DOF object poses are defined by translation and rotation of objects relative to the camera pose. Depth maps are defined by depth elements of rays cast perpendicular from the camera plane to the nearest surface, or 0 if no nearest surface exists for a depth element. Object meshes are defined using the common Wavefront OBJ format [23]. In the downstream tasks presented in this paper we use object meshes, semantic segmentation masks and depth maps. The other output types were added to increase the flexibility of the toolset. To annotate an object with affordances, sub-bounding-boxes can be added, which are stored as a JSON file alongside the exported geometry and automatically loaded when inserting exported meshes in new scenes. Per-object affordance maps can be exported in a similar way as segmentation masks.

A. *Uncluttered scene pipeline*

In this pipeline an uncluttered scene is annotated using bounding-boxes.

1) Record RGB frames of a scene containing objects to label: $\mathbf{I} \in \mathbb{R}^{N \times W \times H \times 3}$, where N is number of frames, W is width and H is height.
2) Obtain camera extrinsics $\mathbf{T} \in \mathbb{R}^{3\times4} = [\mathbf{R} \,|\, \vec{t}]$ and intrinsics for each frame using Structure-from-Motion algorithms such as COLMAP [24, 25] or hloc [26], where $\mathbf{R}$ is camera rotation matrix and $\vec{t}$ is camera translation vector.
3) Determine scale s by comparing keypoints or using AR marker [27], and rescale positions $\mathbf{T}_s = [\mathbf{R} \,|\, s \cdot \vec{t}]$.
4) Render NeRF using $\mathbf{T}_s$ and $\mathbf{I}$.
5) Label objects using bounding-boxes, by inserting boxes, translating and rotating them to surround target objects.
6) Export geometry contained in bounding-boxes by querying density of NeRF in bounding-box areas,

apply density filter and run marching cubes [28].

B. Cluttered scene pipeline

In this pipeline, a cluttered scene is labeled using polygonal meshes. If we have access to the physical objects in the scene, meshes can be obtained through the uncluttered scene pipeline. This pipeline repeats steps 1-4 from the uncluttered scene pipeline but replaces steps 5 and 6 with the following:

5) Label objects using mesh models, by inserting meshes, translating and rotating them to be aligned with NeRF rendering of objects.
6) Export semantic segmentation masks, 2D and 3D bounding boxes, 6DOF object poses and depth maps.

C. Implementation details

Because our labeling functionality is specialized, we implemented *NeuralLabeling* as a fork of *instant-ngp* [11] instead of merging our changes into the main project. *instant-ngp* allows for parallel training and rendering, and with our fork also for labeling. Rendering of geometry extracted using marching cubes and rendering of (re)inserted meshes is implemented using OpenGL. In the bounding-box-based pipeline, we support real-time geometry previews from NeRF. Rendering of overlay effects such as 2D and 3D bounding boxes is implemented using ImGui[1]. Manipulating objects (translation, rotation, scaling) is implemented using ImGuizmo[2]. We implemented an accelerated algorithm for object alignment, using multi-threading and CUDA kernels. NeRF-to-mesh occlusions are handled by comparing estimated depth of rays traced in the NeRF rendering with depth of the rendered meshes fragments. We integrate improved transparent object depth estimation via Dex-NeRF [29]. We enable scripting of *NeuralLabeling* through Python bindings, which is useful for automated dataset generation.

IV. EVALUATION

Our toolkit can be used to easily and quickly label photorealistic scenes that would be hard to manually model and generate various useful outputs for downstream deep learning tasks. We demonstrate this by (1) evaluating base performance of depth generation using object annotations in Section IV-A, (2) evaluating segmentation performance of annotating opaque objects with a high degree of environment occlusion in Section IV-B, (3) annotating scenes containing transparent objects in a complex environment and training neural networks using generated data in Section IV-C and (4) evaluating the performance of a holistic robotic transparent object manipulation system using neural networks trained or fine-tuned on generated datasets in Section IV-D.

A. Ground truth depth label accuracy

To evaluate the optimal performance of using NeuralLabeling, we recorded 30 samples of color and depth data collected from glasses placed into a scene. The glass is then replaced with an opaque clone placed into the same position and the depth data is recaptured. To place the opaque clone into the same position as the original glass, we take a picture of the original scene using a camera and render an overlay image. This is a typical approach for creating real-world validation data for transparent object depth completion [19]. In addition to capturing ground truth depth by manually aligning opaque clones, we also captured a NeRF scene recording. We create meshes of the glasses by recording two environments with opaque clones of the glasses placed facing upwards and downwards (Fig. 4). We used opaque clones instead of the original glasses for creating meshes, as this produced higher quality meshes due to NeRF not being able to correctly estimate the inner surface of the original glasses. We applied our cluttered scene pipeline to this dataset to generate experimental data by aligning the scanned meshes with the original glasses. To evaluate the accuracy of the pipeline, we compare the generated depth of the labels applied to the transparent scene with the ground truth depth generated from aligning opaque clones. This experiment resulted in a median error of 4mm and an MAE (mean absolute error) of 9mm at a mean working distance of 649mm (1.4% relative error), which is similar to the stated depth estimation error of the depth sensor (2% at 2 meters working distance). Exporting depth using NeuralLabeling without mesh annotations, but using Dex-NeRF-like [29] transparent object depth estimation resulted in a median error of 5mm and an MAE of 16mm (2.4% relative error). We can conclude that our method for labeling transparent object depth is at least as accurate as the applied depth sensor is on opaque objects, and more accurate compared to Dex-NeRF.

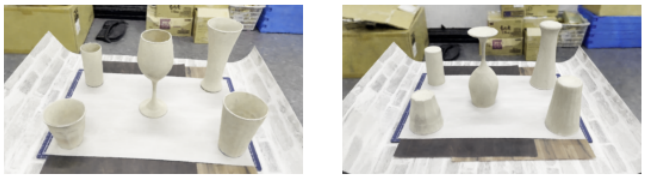


Fig. 4. Opaque clones of glasses placed up- and down-facing, rendered using NeRF. Using the bounding-box labeling pipeline we extract meshes that are used for annotating the dishwasher scenes.

B. NeRF occlusion for generating segmentation masks

One of the unique functions of *NeuralLabeling* is to use NeRF occlusions to generate accurate segmentation masks. We performed a small experiment to measure the effectiveness. We labeled a sequence of three heavily occluded frames of the basket scene (scene B of Fig. 3) with ground truth segmentation masks, then calculated F1-score, Intersection-over-Union (IoU), accuracy, precision and recall. We compare our method with Segment Anything (SAM) [2] labels using 2D bounding boxes and XMem [7] by using the first frame as input. Quantitative and qualitative results can be found in Table II and in the supplemental materials respectively. Our method outperforms SAM in almost every metric, while performing similar to XMem. Some qualitative benefits of our approach are that *NeuralLabeling* does not

[1]Online: `https://github.com/ocornut/imgui`
[2]Online: `https://github.com/CedricGuillemet/ImGuizmo`

TABLE II

QUANTITATIVE RESULTS OF MASKING USING NERF OCCLUSION. HIGHER SCORE IS BETTER.

Method	Binary					Category				
	F1-score	IoU	Accuracy	Precision	Recall	F1-score	IoU	Accuracy	Precision	Recall
SAM	0.80	0.67	0.97	0.71	0.90	0.74	0.61	1.00	0.68	0.85
XMem	0.85	0.74	0.98	0.82	0.88	0.80	0.68	1.00	0.77	0.84
Ours	0.83	0.70	0.98	0.95	0.73	0.80	0.68	1.00	0.93	0.71

require all objects to be visible in the first frame of a sequence such as with XMem and does not need per frame 2D bounding boxes such as with SAM. For generating NeRF occlusions we rely on extracting an accurate depth estimate from NeRF, which is difficult for objects with highlights and reflections. Our method for example struggles to generate accurate segmentation masks of the towel from scene B, which is wrapped in plastic. Compared to the other methods, the segmentation masks generated by NeuralLabeling are more conservative, which decreases the recall score.

C. Training networks for depth completion

In a previous study [22] we evaluated the usage of unpaired training data with a cyclic adversarial training approach [1] for transparent object depth completion. We use the same dataset and network design from the previous study but added supervised ground truth depth maps and instance segmentation masks using *NeuralLabeling*. The RGB images from the original dataset were used for determining camera poses and NeRF rendering.

1) Dataset preparation: For transparent objects a marching cubes threshold can be used to tune the mesh geometry similar to Dex-NeRF [29], however the observed mesh quality was still lower than using opaque clones. We merged the upwards and downwards facing meshes using MeshLab [30] to produce complete meshes of the glasses. We want to show that good results can be obtained using low cost methods, so we avoided using more advanced techniques such as using expensive camera setups [31] or commercial 3D scanners.

The meshes are manually aligned with the NeRF rendering. We generated camera pose estimates for 59 out of 60 scenes, camera pose estimation failed on one scene. We calibrated the camera pose scales for each scene by measuring the distance between two points where the real world distance was known, taking about a minute per scene. It then took two working days to label the 59 scenes with the meshes. An automated process generated the depth maps for the 59 scenes, which took around three minutes per scene.

NeRF-to-mesh occlusions could not reliably be generated due to the difficulty in estimating the depth elements of inner surfaces of glasses using NeRF. Instead, we use sensor depth elements to occlude the generated depth elements. Sensor depth elements for transparent objects are inaccurate due to missing elements ($depth = 0$), background depth elements and noisy surface depth elements. By using sensor depth elements for calculating occlusions, we can fill in missing depth elements and correct background depth elements to be on the object surface, but some noisy surface depth elements that were inaccurately estimated as being too close to the camera might remain. Fig. 5 shows a sample from the dishwasher dataset, with the original depth recorded by the depth sensor, with generated depth estimate from mesh annotations, and finally, the combined sensor and mesh depth that is used as ground truth for training our network.

We reuse the validation set of the original paper [22] ($N = 26$), containing scenes in which glasses were manually aligned with opaque clones (using the same process described in Section IV-A). All evaluation samples are patches with dimensions $512 \times 512 \times C$ extracted from the center of the sensor frame with dimensions $1280 \times 720 \times C$, where depth is clipped to the $[450, 2000]$ mm range. For training we first random crop frames horizontally to dimensions $720 \times 720 \times C$ and then resize to dimensions $512 \times 512 \times C$ using nearest neighbor interpolation. For evaluation we center crop the frames horizontally before resizing. C is the number of channels for the modality: 1 for depth-only, 3 for RGB only, 4 for RGBD. For each channel we map the values to the domain $[-1, 1]$ from the original domains $[0, 255]$ from RGB and $[450, 2000]$ for depth.

2) Networks and training: Whereas in the previous study we used two generator networks and two discriminator networks, in this study we use only a single generator network for each evaluated modality. The generator network in the previous study and the current study is a simple U-Net, based on Pix2pix [32]. We evaluated three modalities: RGBD2Depth, Depth2Depth and RGB2Depth. In the previous study we evaluated Depth2Depth and RGBD2RGBD modalities (the method required input and output type to be symmetric, so the target output was RGBD data of scenes containing opaque clones of the original glasses by spray painting them). In both the original study and the current study we trained for 2×10^6 iterations.

3) Results: Table III and Table IV contain quantitative and qualitative results. Cyclic adversarial measurements are sourced from our previous paper [22]. Metrics used are Root Mean Square Error (RMSE), Mean Absolute Error (MAE), Relative error (Rel), proportion of depth elements with less than 5% error (1.05), less than 10% error (1.10) and less than 25% error (1.25). We apply the metrics to the depth elements covered by transparent objects. Regardless of the modality used, we could obtain a significant improvement by using supervised data created using *NeuralLabeling*.

The weakly supervised approach required recording a separate dataset containing 60 scenes of opaque clones, which took about 4 hours to collect, but our current method

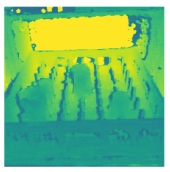
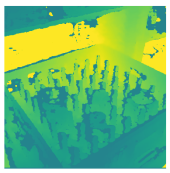
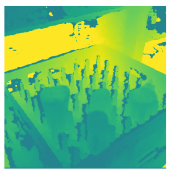
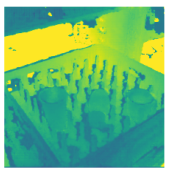
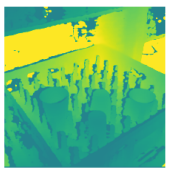
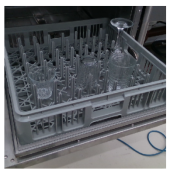
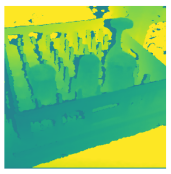
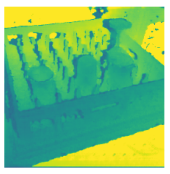
TABLE III

TRANSPARENT OBJECT DEPTH COMPLETION USING WEAKLY SUPERVISED METHODS VERSUS STRONGLY SUPERVISED METHODS

Training regime	Modality	RMSE (m) ↓	MAE (m) ↓	Rel ↓	1.05 ↑	1.10 ↑	1.25 ↑
Joint Bilateral Filter	RGBD2Depth	0.067	0.048	0.083	0.477	0.688	0.950
ClearGrasp	RGBD2Depth	0.090	0.057	0.120	0.404	0.555	0.840
Cyclic adversarial	RGBD2RGBD	0.061	0.040	0.072	0.528	0.767	0.940
Cyclic adversarial	Depth2Depth	0.058	0.035	0.061	0.589	0.861	0.954
Dishwasher30k supervised	RGBD2Depth	*0.037*	0.023	0.039	0.725	0.880	*0.959*
Dishwasher30k supervised	Depth2Depth	0.043	*0.021*	*0.038*	*0.800*	*0.895*	0.955
Dishwasher30k supervised	RGB2Depth	0.045	0.028	0.049	0.676	0.861	0.948

TABLE IV

QUALITATIVE RESULTS OF OUR SUPERVISED METHOD AND PREVIOUS BEST CYCLIC ADVERSARIAL METHOD.

Captured Color	Captured Depth	Our result	CycleGAN result	Ground truth depth
Three samples with the lowest MAE using our method				
		 0.012	 0.032	
		 0.013	 0.025	
		 0.014	 0.033	
Three samples with the highest MAE using our method				
		 0.034	 0.066	
		 0.035	 0.054	
		 0.036	 0.050	

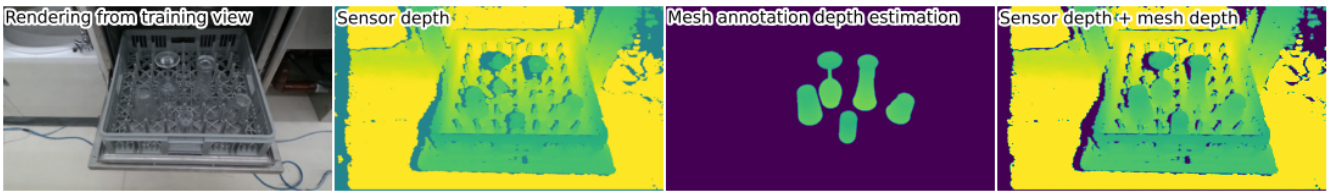


Fig. 5. Non-Lambertian objects in a complicated environment, annotated using opaque clone NeRF meshes, second column shows sensor depth estimate using RealSense D415, third column shows estimated object depth based on mesh annotations, fourth column shows the combination of generated depth with noisy sensor depth, which can be used as ground truth data for training a deep neural network.

does not use this. For the current supervised approach, we had to record two scenes of opaque clones to extract meshes, and then label the original transparent objects in the dishwasher scenes. Creating opaque meshes took around 8 hours. Aligning the opaque meshes with the transparent scenes took around 16 hours, the time per scene varying based on the amount of objects in the scene. Training time for the original approach was around 4 times longer, as four networks had to be trained instead of a single network. NeuralLabeling requires COLMAP camera estimates taking around an hour per scene and we pretrained the NeRFs to allow for faster labeling for around an hour per scene. Predictions using the newly trained networks are slightly more blurry than the CycleGAN approach due to not using a discriminator, but because the depth maps are for robot consumption this was not considered an issue. We conclude that the *NeuralLabeling* approach requires more time to prepare the dataset but allows for more accurate depth estimates and efficient training for downstream tasks.

D. Robot experiment and demonstration

We implemented ROS nodes for transparent object depth completion using the depth-to-depth network and a Detectron2 [33] instance segmentation network fine-tuned on transparent object data generated using NeuralLabeling. The robot that we used is RT Corporation Sciurus17. Grasps are evaluated on two objects, a tall glass and a wine glass, which were part of the dataset for training the depth completion network and fine-tuning the segmentation network. We placed the objects in 9 positions inside the dishwasher, and performed 3 trials per position, for a total of 54 trials. The overall grasp success rate using the system is 83.3%. Wine glass grasp success rate was 92.3% and tall glass grasp success rate was 75%. We performed the same experiment with our prediction segmentation masks but without using depth completion (i.e. the original sensor depth). The overall grasp success rate without depth completion was 16.3%. Wine glass grasp success rate without depth completion was 29.6% and tall glass grasp success rate without depth completion was 0%. In future work, we plan to explore more advanced neural network designs for more accurate depth completion, as well as mechanical improvements to the gripper to allow for a larger error tolerance. As shown in the supplemental material, our robot system can also perform sequential grasping of transparent objects placed in a dishwasher environment.

V. DISCUSSION AND CONCLUSION

We presented *NeuralLabeling*, a labeling approach and toolset for annotating NeRF renderings and generating datasets for downstream deep learning applications. With *NeuralLabeling* we were able to rapidly create datasets of transparent objects in a complex environment and use the datasets to greatly improve the performance of transparent object depth completion and to perform instance segmentation in a transparent object manipulation example. The main limitation of *NeuralLabeling* is the significant time required to record scenes and generate camera extrinsics for each captured frame, however this is mostly automated and could be further automated in the future. In future we plan to apply *NeuralLabeling* to larger scenes such as supermarkets and convenience stores for generating datasets to fine-tune vision-language models. We also plan to investigate how *NeuralLabeling* can be applied to dynamic scenes and how high-quality object meshes can be used to insert objects into scenes where the objects were not originally located.

ACKNOWLEDGMENT

The authors would like to thank Lukas Meyer for the fruitful discussions. This work was supported by JST [Moonshot R&D][Grant Number JPMJMS2031]. 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] Jun-Yan Zhu et al. “Unpaired Image-to-Image Translation Using Cycle-Consistent Adversarial Networks”. In: *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*. Oct. 2017.

[2] Alexander Kirillov et al. *Segment Anything*. 2023. arXiv: 2304.02643 [cs].

[3] Alec Radford et al. “Learning Transferable Visual Models From Natural Language Supervision”. In: *Proceedings of the 38th International Conference on Machine Learning*. International Conference on Machine Learning. PMLR, July 1, 2021, pp. 8748–8763.

[4] Ben Mildenhall et al. “NeRF: representing scenes as neural radiance fields for view synthesis”. In: *Commun. ACM* 65.1 (Dec. 2021), pp. 99–106. ISSN: 0001-0782. DOI: 10.1145/3503250.

[5] *Segment Anything Labeling Tool*. URL: https://github.com/anuragxel/salt.

[6] *Roboflow*. URL: https://www.roboflow.com.

[7] Ho Kei Cheng and Alexander G. Schwing. "XMem: Long-term Video Object Segmentation with an Atkinson-Shiffrin Memory Model". In: *European Conference on Computer Vision (ECCV)*. 2022.

[8] Kevin Lai et al. "Detection-Based Object Labeling in 3D Scenes". In: *2012 IEEE International Conference on Robotics and Automation*. 2012, pp. 1330–1337. DOI: 10.1109/ICRA.2012.6225316.

[9] Walter Zimmer, Akshay Rangesh, and Mohan Trivedi. "3D BAT: A Semi-Automatic, Web-Based 3D Annotation Toolbox for Full-Surround, Multi-Modal Data Streams". In: *2019 IEEE Intelligent Vehicles Symposium (IV)*. 2019, pp. 1816–1821. DOI: 10.1109/IVS.2019.8814071.

[10] 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.

[11] 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.

[12] Tsukasa Takeda et al. "Efficient 3D Reconstruction of NeRF Using Camera Pose Interpolation and Photometric Bundle Adjustment". In: *ACM SIGGRAPH 2023 Posters*. 2023. DOI: 10.1145/3588028.3603691.

[13] *NeRFCapture*. URL: https://github.com/jc211/NeRFCapture.

[14] Xiaotong Chen et al. "ProgressLabeller: Visual Data Stream Annotation for Training Object-Centric 3D Perception". In: *International Conference on Intelligent Robots and Systems (IROS)* (2022).

[15] Markus Suchi et al. "3D-DAT: 3D-Dataset Annotation Toolkit for Robotic Vision". In: *2023 IEEE International Conference on Robotics and Automation (ICRA)* (2023).

[16] Kenneth Blomqvist et al. "NeRFing It: Offline Object Segmentation Through Implicit Modeling". In: *2023 IEEE International Conference on Robotics and Automation (ICRA)* (2023).

[17] Andrew Guo et al. "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.

[18] Lukas Meyer et al. *PEGASUS: Physically Enhanced Gaussian Splatting Simulation System for 6DOF Object Pose Dataset Generation*. 2024. arXiv: 2401.02281 [cs].

[19] Shreeyak Sajjan et al. "Clear Grasp: 3d Shape Estimation of Transparent Objects for Manipulation". In: *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 3634–3642.

[20] Xiaotong Chen et al. "Clearpose: Large-scale Transparent Object Dataset and Benchmark". In: *European Conference on Computer Vision*. Springer, 2022, pp. 381–396.

[21] Luyang Zhu et al. "RGB-D Local Implicit Function for Depth Completion of Transparent Objects". In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2021.

[22] Floris Erich et al. "Learning Depth Completion of Transparent Objects Using Augmented Unpaired Data". In: *2023 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2023.

[23] *Wavefront OBJ File Format*. Format Description. URL: https://www.loc.gov/preservation/digital/formats/fdd/fdd000507.shtml.

[24] Johannes Lutz Schönberger and Jan-Michael Frahm. "Structure-from-Motion Revisited". In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016.

[25] Johannes Lutz Schönberger et al. "Pixelwise View Selection for Unstructured Multi-View Stereo". In: *European Conference on Computer Vision (ECCV)*. 2016.

[26] Paul-Edouard Sarlin et al. "From Coarse to Fine: Robust Hierarchical Localization at Large Scale". In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. 2019.

[27] Lukas Meyer et al. "CherryPicker: Semantic Skeletonization and Topological Reconstruction of Cherry Trees". In: *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*. 2023, pp. 6244–6253. DOI: 10.1109/CVPRW59228.2023.00664.

[28] William E. Lorensen and Harvey E. Cline. "Marching cubes: A high resolution 3D surface construction algorithm". In: *ACM Trans. Graph.* (1987), pp. 163–169. DOI: 10.1145/37401.37422.

[29] Jeffrey Ichnowski et al. "Dex-NeRF: Using a Neural Radiance Field to Grasp Transparent Objects". In: *5th Conference on Robot Learning (CoRL 2021)* (2021).

[30] Paolo Cignoni, Massimiliano Corsini, and Guido Ranzuglia. "MeshLab: An Open-Source 3D Mesh Processing System." In: *ERCIM News* 2008.73 (2008).

[31] Floris Erich et al. "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. DOI: 10.1109/SII55687.2023.10039147.

[32] Phillip Isola et al. "Image-to-Image Translation with Conditional Adversarial Networks". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2017, pp. 1125–1134.

[33] Yuxin Wu et al. *Detectron2*. https://github.com/facebookresearch/detectron2. 2019.