Evo-NeRF: Evolving NeRF for Sequential Robot Grasping

Anonymous Author(s)
Affiliation
Address
email

Abstract: Sequential robot grasping of transparent objects is important in many industrial and household scenarios. We leverage recent speedups in NeRF training, and further extend it to progressively train on images as they are captured. We propose early training termination when a sufficient task confidence is achieved and the reusing of the NeRF weights from grasp to grasp to rapidly adapt to object removal. We propose Evo-NeRF with additional geometry regularizations improving performance in rapid capture settings. Because there can be unreliable geometry from NeRF, general purpose grasp planners such as Dex-Net struggle. To mitigate this distribution shift, we propose a grasping network Rad-Net adapted to NeRF's characteristics, and train it on NeRF models trained on synthetic photo-realistic scenes. In experiments, a physical YuMi robot using Evo-NeRF and Rad-Net achieves a 89% grasp success rate over 27 trials on single objects, with early capture termination providing a 41% speed improvement with no loss in reliability. In the sequential grasping task on 6 scenes, Evo-NeRF reusing the weight clears 72% of the objects, retaining the same performance as reconstructing the NeRF from the scratch (76%) but taking 61% of the capture time.

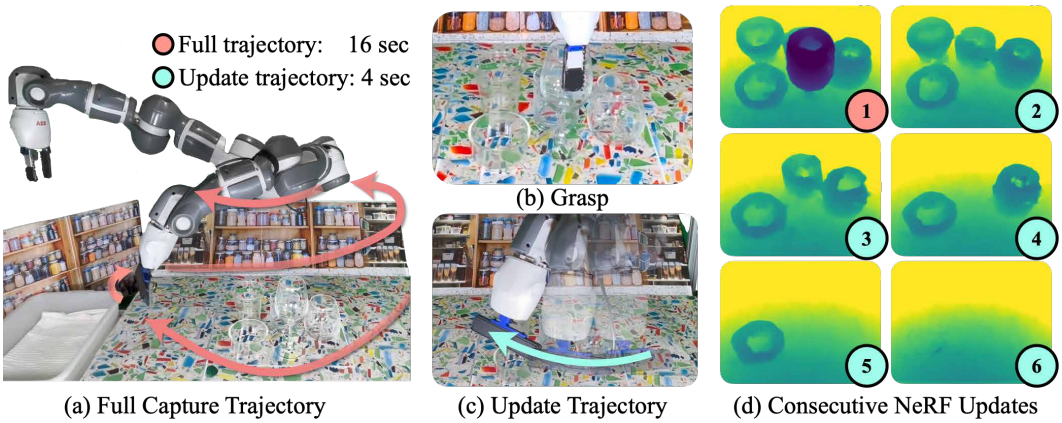
1 Introduction

Sequentially grasping transparent objects is a highly useful but difficult task for robots due to the complexity of sensing the object to grasp. It has applications in industry, pharmaceuticals, and households; but since camera-based sensors see through transparent objects from most angles, assumptions underlying traditional disparity- and structure-from-motion-based methods break. ClearGrasp [1] addresses this problem and trains a CNN to infer local surface normals on transparent objects from RGBD images based on synthetic examples from Blender. They show impressive results on 3-5 transparent objects separated by 2cm, however the system has challenges with thin walls of transparent containers and background distractors. To this end, Dex-NeRF [2] introduced a method that uses Neural Radiance Fields (NeRF) [3] to grasp transparent objects, but at the cost of hours of computation per grasp. In this paper, we propose Evo-NeRF, a method for progressively updating and rapidly training NeRF models for grasping, and Rad-Net, a neural network for robust grasping on NeRF models.

NeRFs are powerful 3D representation which can reconstruct traditionally challenging-to-model scenes that include transparent objects. NeRFs were originally designed for novel view synthesis, but their volumetric representation enables the extraction of scene geometry. The use of geometry extracted from NeRFs have been explored in robotic applications such as dynamics modeling [4, 5], localization [6] and motion planning [7], and computing grasps on transparent objects [2, 8].

In the last year, dramatic advancements in NeRF training speed have opened the door for real-world usage [9, 10, 11]. In this paper, we apply NeRF in a purely online setting to practical robotics applications which require sequentially grasping transparent objects in clutter, for example as required in dishwasher unloading, table clearing, and other household tasks.

Submitted to the 6th Conference on Robot Learning (CoRL 2022). Do not distribute.



(a) Full Capture Trajectory (b) Grasp (c) Update Trajectory (d) Consecutive NeRF Updates

Figure 1: **Sequential object removal. (a)** The YuMi moves a camera on its left hand across the full hemisphere trajectory (red arrow) to capture a scene of 5 glass objects **(b)** The robot immediately plans and executes a grasp from the NeRF **(c)** Short trajectories are used to update the NeRF between grasps **(d)** Evo-NeRF first reconstructs the whole scene (1) with the camera trajectory shown in (a), then progressively updates the scene with small camera captures shown in (c) as objects are removed.

To make NeRF practical for robotic grasping, we build on Instant-NGP [11], a fast variant of NeRF. Rather than training on a fixed set of images, we incrementally optimize over a stream of images as they are captured during a robot motion. Because of NeRF's varying convergence speed on different difficulty scenes, we also propose a method to terminate image capture upon achieving sufficient task confidence. We further adapt NeRF to sequential grasping by re-using NeRF weights from grasp to grasp and demonstrate its rapid adaptability to object removal.

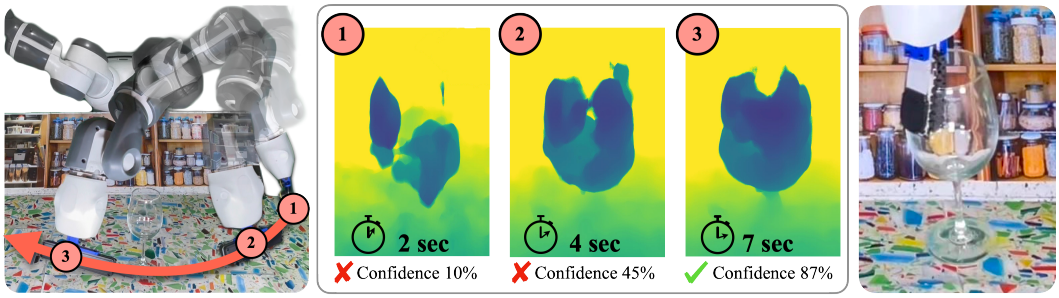
Since we propose that the robot captures images and trains a NeRF as it moves, motion blur, kinematic limitations, and speed considerations reduce the quality of the recovered geometry and introduce prominent spurious geometry known as *floaters*. We propose adding geometry regularization to the training objective, which improves the recovered geometry, but out-of-the-box grasp planners still struggle to find quality grasps due to remaining artifacts. Dex-NeRF [2] on the other hand, could use an out-of-the-box grasp sampler because it used diverse, high-quality, calibrated, *still* images captured in an offline process.

To mitigate the effect of the lower-quality NeRF reconstructions, we propose a novel Sim2Real training pipeline for a NeRF-based grasping network, which transfers well to real-world NeRF reconstructions. This pipeline takes advantage of the training speed of Instant-NGP—without it, the pipeline would be computationally infeasible. Experiments on a real robot with an actuatable camera to capture images suggest that Evo-NeRF is able to reconstruct graspable scene geometry rapidly and reliably, when combined with Rad-Net achieving a 89% success rate on single objects within *9.5* seconds of image capturing.

The contributions of this paper are: (1) novel usage of NeRF in a sequential setting, rapidly evolving the NeRF representation between grasps, (2) improvements in scene geometry reconstruction speed built on existing methods, (3) an active sensing approach to efficiently early stopping capture, (4) a novel Sim2Real training pipeline to acclimate a grasping planner to NeRF geometry characteristics, (5) a dataset of 8667 Blender rendered scenes of transparent objects with robust grasps, and (6) experimental data suggesting that Evo-NeRF enables rapid grasping on NeRF.

2 Related Work

Neural Radiance Fields (NeRF) NeRF [3] is a neural-network scene representation that enables photorealistic synthesis of novel views of a scene given a set of images and camera matrices. The representation is a function of location and view angle, and returns a density and view-dependent color. Densities and colors sampled along a camera ray are aggregated using volumetric rendering to produce a pixel color. NeRF is popular in the computer vision and graphics communities with



(a) Early Stopped Image Capture (b) Progressive NeRF Training (c) Grasp

Figure 2: **Evo-NeRF for rapid grasping: (a)** The robot begins moving along a hemisphere capture trajectory **(b)** Evo-NeRF progressively trains NeRF during arm motion, quickly building graspable geometry of the wine-glass. Grasp confidence from Rad-Net builds as NeRF learns geometry, reaching the stopping threshold at (3). **(c)** Robot execution of the grasp given by Rad-Net at the early stop point.

the applications in dynamic scene reconstruction [12, 13], image synthesis [14, 15, 16], pose estimation [17, 18, 19], and more. Optimizing NeRF to reconstruct a single scene can take hours or days—making it impractical for many robotics applications. Instant-NGP [11] and others [9, 20] speed up NeRF by using voxel feature grids instead of multi-layer perceptions to simplify or remove [10] a computational bottleneck. We build on Instant-NGP [11], which uses a learnable hash encoding and highly optimized CUDA implementation to speed up NeRF training from the order of hours to seconds. Others have also sped up NeRF by reusing computation between scenes by utilizing priors. Existing methods [21, 22, 23, 24, 25] use convolutional neural networks (CNNs) to extract image features as input to a shared network that predicts the NeRF. Tancik et al. [26] and Gao et al. [27] speed up NeRF training using meta-learning to initialize network weights to ones that converge faster for likely scenes. In this work, we propose using past reconstructions of a scene as an initialization for the current reconstruction, allowing rapid adaptation to changes in the scene.

NeRFs in Robotics Recent research has shown NeRFs to be a promising scene representation for downstream robotics tasks such as navigation and SLAM [6, 7, 28] and manipulation [5, 2, 8]. Yen-Chen et al. [17] and Tseng et al. [29] use a trained NeRF model to estimate an object's 6-DOF pose by minimizing the residuals between a rendered image and a given observed image. Driess et al. [4] train a graph neural network to learn a dynamics model in a multi-object scene represented through a NeRF model, while Li et al. [5] condition a NeRF model on a learned latent dynamics model to plan to visual goals in simulated environments. We propose building on advances in NeRF and its applications to robotics to speed up NeRF-based grasping for practical uses.

Grasping Transparent Objects: Most closely related to this paper are two recent works leveraging NeRFs to manipulate objects that cannot be detected by commodity RGBD sensors. Yen-Chen et al. [8] use a NeRF model offline to train dense object descriptors and manipulate thin and reflective objects. Ichnowski et al. [2] show that manually constructing an offline dataset of a given scene then training NeRF allows off-the-shelf grasp planners [30] to compute successful grasps on transparent objects. ClearGrasp [1] trains a Sim2Real depth prediction network on RGB images, then uses this network in real environments to estimate surface geometry for grasps. This idea has been extended to pointclouds and with more efficient real-world data collection [31, 32]. Using NeRF removes the requirement for RGB datasets which make Sim2Real challenging, and has superior performance on thin surfaces, occlusions, and complex backgrounds.

3 Problem Statement

Given a set of transparent objects resting on a planar workspace, the objective is for the robot to find, grasp, and remove each object quickly. Objects are placed close to each other (2.5 cm) and the robot has an actuatable camera and a parallel jaw gripper (Fig. 1). The focus is on finding robust grasps rapidly, with grasp success measured as transporting an object without dropping.

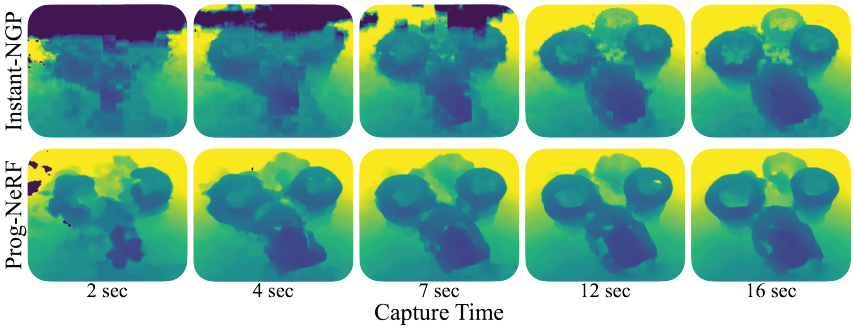


Figure 3: Comparison of Evo-NeRF's training over time vs NGP on the exact same camera trajectory. Added geometry regularization improves the speed and smoothness of geometry regularization.

We assume (1) objects rest in graspable stable poses on a flat surface, (2) objects are in the reachable workspace of the robot with a known forward kinematic model, (3) the camera-to-arm transform is known and stable, and (5) the robot can follow a known obstacle-free trajectory to capture images.

4 Method

To rapidly compute robust grasps, we propose *Evo-NeRF* and *Rad-Net*. Evo-NeRF, or *Evo*lving *NeRF*, builds on Instant-NGP [11], a fast implementation of NeRF, and adds a series of modifications to make it practical. Rad-Net, or *Rad*iance-adjusted grasp *Net*work, is a network trained to compute grasps from geometry reconstructed from a NeRF.

4.1 Evo-NeRF

To shorten the time to get a trained NeRF, we propose Evo-NeRF, a method that pipelines image capture with NeRF training, reuses weights in sequential grasping, adds regularization to counter effects from rapid capture, and includes an early stopping condition to start a grasp when the grasp network has high confidence.

Image capture: The Evo-NeRF method starts with the robot moving a camera around its workspace to capture images. Heuristically, hemispherical captures are ideal for NeRF since they maximally vary the view angles of the scene. The Evo-NeRF capture trajectory sweeps the camera through a discretized hemisphere centered at a location of interest while pointing at the center. First, the camera sweeps around the z-axis to maximize the variance of viewing angle early in the capture sequence. In experiments, we capture images every 3 cm while moving at 20 cm/s, and a full capture trajectory takes 16 seconds and includes 80 images with trajectory shown in Fig. 1.

Continual NeRF training: NeRF training, even sped up by Instant-NGP, is a bottleneck. We propose continually training NeRF from the moment the first image is captured, and incrementally adding images to the NeRF training dataset as the camera moves to new viewpoints. This effectively pipelines the image-capture and NeRF-training processes, and allows for usable NeRF representations quickly after (and sometimes before) the capture process finishes.

During each capture motion we train NeRF in batches of 48 steps, adding new images between each batch when available. This is akin to other online neural implicit methods like iMAP [6] and NICE-SLAM [33], who also update the image sets between training batches. We compute the camera frame using the forward kinematics and pair it with each image. In practice, this yields pose error around 1 cm, which NeRF accounts for by optimizing the camera extrinsics.

Reusing NeRF weights: In sequential grasping scenarios, scenes often change by only the removal of the last object grasped. To take advantage the information already trained, we use the NeRF network weights from the previous grasp in the subsequent grasp. In implementation, we remove the old images from the training dataset and start capturing and training on images for the next grasp.

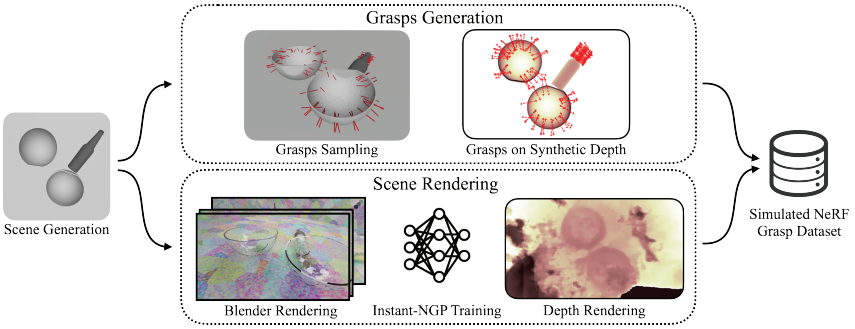


Figure 4: **Sim2Real Dataset Generation**. Each scene includes a subset of the training objects in simulation (Fig. 5). **Top:** Grasp generation starts by sampling grasps on the object meshes and projecting it to a top-down view. **Bottom:** We render multiple views of each scene using Blender, then train Instant-NGP and render a top-down depth image. We accumulate NeRF depth rendering and projected grasps into a dataset.

Geometry regularization: A well-known artifact of NeRF's volumetric rendering loss are *floaters*, spurious regions of density floating in space. When using NeRF for view synthesis, floaters can go unnoticed, but in grasping, floaters can lead to grasp failures. We apply 2 regularizations which increase the speed and smoothness of geometry reconstruction while sacrificing RGB quality.

First, we adapt the total-variation regularization loss (TV-loss) from Plenoxels [34] to discourage floaters and encourage smooth scene geometry. During training, at each step Evo-NeRF sample N random points p_i using rejection sampling to constrain samples to locations with non-trivial density values. Evo-NeRF then queries the density at all 8-connected neighbors n_j at a radius r. The final TV-loss is $L_{\text{tv}} = \sum_{i=1}^{N} \sum_{j=1}^{8} \lambda_{\text{tv}} (\sigma(p_i) - \sigma(n_j^i))^2$, where σ is the raw, pre-activation output from the density network, and λ_{tv} is a loss scaling factor.

Second, sampling along each ray more coarsely during training reduces floaters and quickly acquires meaningful geometry. In Instant-NGP the distance between samples grows proportional to distance along the ray, and we scale this distance to be 10x larger. By training with coarse samples, the NeRF is incentivized to learn a low frequency representation of the scene to minimize reconstruction error.

Efficient perception stopping: In scenes where NeRF is able to recover usable geometry before the full camera trajectory has terminated, Evo-NeRF can terminates the capture phase early to speed task completion. In Sec. 5.2 we present experiments showing this by querying grasp confidence of Rad-Net in a closed loop while the robot moves the camera and trains NeRF. When a confidence exceeds a threshold, the capture stops early and the robot executes the grasp.

4.2 Grasp Planning Network

When NeRF is trained to completion with dense camera viewpoints, grasp planners trained on ground truth depth like Dex-Net [35] produce usable grasps. However, in an online setting where viewpoints are of lower quality, depth images rendered from NeRF appears significantly different from depth images from standard RGBD cameras. To mitigate this test-time distribution shift and enable more reliable grasping from online NeRFs, we train a network directly on NeRF-rendered depth maps to predict grasps in a Sim2Real fashion.

Network Architecture: We train a *location* neural networks to predict the center of the grasp location when given a NeRF-rendered image; and we train a *rotation* network to predict the discretied grasp angle when given a cropped patch around the grasp location. We adapt the grasping architecture proposed by Zhu et al. [36], which suggests that an equivariant convolutional neural network learns to perform top-down grasps in fewer samples than standard networks. We train location and rotation networks on a static grasp dataset, in contrast to the online setting in Zhu et al. [36].

Dataset Generation: We generate the training dataset in simulation. We use 7 object meshes that are representative of the common household transparent objects which are graspable by the YuMi robot, shown in Fig. 5a. We model all objects with the same density as glass (2500 kg/m^3). Then, we assemble a set of scenes with labeled grasp qualities: we randomly place objects in stable poses

(a) Training objects (Blender) (b) In-distribution real objects (c) Out-of-dist. objects (d) Clutter

Figure 5: **Training and testing objects.** The Blender rendering in **(a)** shows the 7 objects we use in data generation for computing grasps and rendering in various stable poses. Objects in **(b)** are real objects tat are in-distribution with the training objects. We also test on out-of-distribution objects shown in **(c)**. To test grasping in clutter, we setup various testing scenes with objects in and out of distribution, with examples shown in **(d)**.

on a planar surface and analytically sample antipodal grasp closure axes based on mesh surface normals as in Dex-Net [37]. We use a soft point-contact model [38], and evaluate the probability of grasp success using wrench resistance [39], a common analytic measure for grasp success. This method is inexpensive to calculate (0.02 sec per grasp) and has high precision for measuring grasp success [40]. We densely sample 1000 collision free grasps for each stable pose and we use Blender to render the scenes.

Training: To train Rad-Net, we project sampled grasps onto the depth images and store at each pixel the maximum grasp confidence over all rotations resulting confidence heatmaps. We dilate and blur the masks with a 3x3 kernel to smooth the predictions, and randomly augment the masks with translation, shear and scale transformation for both the depth images and the confidence heatmaps. To train the rotation network, we sample crops from grasps above 0.7 quality, and use a cross-entropy loss on the output rotation probabilities.

Grasp Planning: To execute a grasp from Rad-Net we render a depth image from NeRF of size 144×256 from the camera pose used during dataset generation, using the ray transmittance truncation of Dex-NeRF [2].We query the *location* network on this depth image to obtain a heatmap over the image of grasp confidence, then crop a patch of the depth image centered at the argmax of this heatmap. The *rotation* network takes this crop and outputs 8 grasp angle probabilities, and we take the weighted average of the argmax with its neighbors to produce the final grasp angle. We determine grasp depth by analyzing a local pointcloud at the grasp location, and subtracting a static 1.5 cm grasp depth from the highest point.

5 Experiments

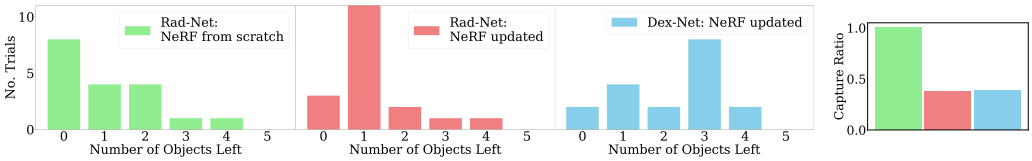
We evaluate the reliability of Evo-NeRF paired with Rad-Net vs Dex-Net [35], evaluate the speed improvements from early stopping captures and Evo-NeRF reusing representations, and ablate aspects of the system including NeRF modifications and training on NeRF depth vs ground-truth depth. We compare to Dex-Net to highlight the improvement in reliability gained from training on NeRF-rendered depth rather than ground-truth depth, and note that in Dex-NeRF [2], the NeRF model was trained for 1900x longer, with an offline, manually captured set of images with precisely calibrated poses from Colmap [41, 42]. This difference in view quality and training length from rapid capture results in a notable drop in raw Dex-Net grasp robustness because of lower quality reconstructions.

5.1 Physical Setup

We evaluate on a physical YuMi robot with a ZED Mini camera. The pose of the ZED relative to the arm that holds it is calibrated with a chessboard once before all experiments. We surround the robot with a kitchen-like workspace containing printed images of a countertop and shelves, where test objects are positioned near the center of the workspace. The workspace has 3 LED floodlights positioned across from the robot aiming at the workspace. We use one NVIDIA GeForce RTX 3080 GPU for NeRF training and grasp network inference. We evaluate on 9 different objects, both in distribution and out of distribution with respect to the train set in Fig. 5a.

Table 1: **Single objects results:** each cell shows the average over 27 different trials. Full capture compares success after a complete camera trajectory, and early stop compares grasp success at the same stop early stop point. Early stopping results in a 41 % speed improvement with no drop in success rate for Rad-Net.

Method	Full Capture Success	Early Stop Success
Dex-Net	56 %	11 %
Rad-Net	89 %	89 %
Time	16s	9.5s



(a) Number of trials with each number of objects left after extraction of each method.

(b) Capture time ratio.

Figure 6: **Decluttering results.** We report **(a)** histograms of objects remaining after each trial for Rad-Net and Dex-Net [35] and **(b)** the ratio of the capture time over the full capture time for each method. Rad-Net is able to extract more objects compared to Dex-Net. Furthermore, reusing NeRF to update scene geometry rather than recapturing the scene retains performance of Rad-Net, while reducing capture time by 61%.

5.2 Rapid single object retrieval

We apply confidence-based capture early stopping (4.1) with a threshold of 70% to execute a grasp as quickly as possible as shown in Fig. 2. We place each of the 9 test objects near the center of the workspace, and report grasp success and total time spent capturing images. We repeat each experiment 3 times and compare Rad-Net with Dex-Net [35] and evaluate with and without early capture stopping. Since Dex-Net does not output grasp confidence we use the same stopping point for both networks, as determined by Rad-Net. An experiment is successful if the robot grasps and places the object into the storage bin.

Table 1 summarizes the results. Using Rad-Net for early stopping results in a capture time reduction of 41 %, with no drop in reliability. On average, the robot grasps objects within 9.5 seconds with an 89% success rate over 54 trials. Rad-Net outperforms Dex-Net in grasp success by 1.6x even with a full capture of the scene as a result of its habituation to NeRF density. Dex-Net performs poorly overall because of its sensitivity to NeRF depth noise and Rad-Net's primary failure cases are on out-of-distribution objects, specifically missing grasps on the lightbulb and tape dispenser. This is likely because the training set has no small profile items.

5.3 Sequential decluttering

We evaluate on the decluttering task where multiple transparent objects are placed in close proximity (2cm) in stable poses on the table, and the robot must sequentially grasp and place all objects in the bin one by one as shown in Fig. 1. We consider three tiers of experiments with different difficulties and two scenes for each tier, resulting in a total of 6 different scenes (Fig. 5). We repeat each scene 3 times and compare against Dex-Net [35]. At the beginning of each experiment, the robot executes a full capture of the scene as shown in Fig. 1a. Then, after each consecutive grasp, the robot executes a much smaller hemispherical capture centered at the grasp location to update the NeRF (Fig. 1c). We allow only as many grasp attempts as objects in the scene. If a grasp planner generates a wrong grasp leading to a joint over-torque error, we terminate the experiment as a failure.

Results are summarized in Fig 6, showing the number of objects remaining after each trial terminates and the capture time ratio. Overall, Evo-NeRF with Rad-Net clears 72 % of test objects across all tiers, while Dex-Net clears 48 % of objects, and takes 39% of the capture time compared to Rad-Net using the full image capture each grasp, which has similar clear performance (76 %). This suggests Evo-NeRF retains graspable geometry over successive updates, despite their short duration.

Table 2: **Ablations of graspability regularizations.** We query Dex-Net [35] continuously through camera capture trajectories and report the percent of the trajectory needed until the highest probability grasp is on an object. We compare vanilla Instant-NGP with Evo-NeRF, as well as ablating TV-loss and coarse ray sampling.

	Instant-NGP	Evo-NeRF -TV	Evo-NeRF -Coarse	Evo-NeRF
% Capture Until Grasp ↓	80.3 %	64.8 %	62.0 %	**52.6 %**

5.4 Graspability ablation

In this experiment, we ablate the changes made to NeRF speeding geometry graspability. We capture 9 single-object and 3 multi-object scenes, then continuously train NeRF as it captures, using the same static images and holding all other hyperparameters constant. We measure the capture time needed until the first grasp output from Dex-Net [35] lands on a real object as a proxy for graspability convergence. Table 2 shows the percent of the capture trajectory needed, and Fig. 3 shows a timelapse of visual qualities over a capture. Results suggest that the proposed method produces graspable geometry faster, with a 32 % reduction in capture time needed to grasp from Dex-Net.

5.5 NeRF Depth vs Ground Truth Depth

We quantitatively compare the effect of the distribution shift between NeRF-generated and ground-truth depth images on training grasp networks. We generate ground-truth depth images with pyrender [43]. We train an additional network, GT-Net, on ground-truth depth only. We also evaluate Dex-Net [35], which is trained on a much larger dataset with ground-truth depth images. We test on the held-out test set of NeRF-rendered depth images and report average grasp confidence using the soft-point-contact model and wrench resistance. We normalize results with respect to the performance of GT-Net and evaluated on the on ground-truth depth images of the same scenes.

GT-Net achieves 0 %, Rad-Net achieves 42 %, and Dex-Net achieves 0.1%, suggesting that there is a large distribution shift from training on ground-truth depth to testing on NeRF-depth in simulation. Rad-Net performance in simulated scenes is worse than in real scenes because the synthetic dataset contains fewer camera angles than real scenes (52 vs. 80), resulting in more floaters.

6 Conclusion

In this work, we introduced Evo-NeRF, a method that progressively captures and trains NeRFs for practical robotic grasping. While its rapid image capture produces lower-quality reconstructions than prior work, we propose reusing trained weights in sequential grasping, geometry regularization, and continual training to obtain better 3D reconstructions. We further propose a novel Sim2Real training pipeline, where we train grasp networks on NeRF rendered depth images in simulated environments, and we find the networks are able to predict high quality grasps in the physical environment. In experiments, Evo-NeRF and Rad-Net can grasp transparent objects around 1900x faster than Ichnowski et al. [2] and with 89 % success on singulated objects.

6.1 Limitations and future work

Rad-Net uses on rendered depth images, throwing away much of the rich 3D NeRF information. It may missing out on more robust grasps than a network trained on 3D geometry (e.g., VGN [44]). Future work in this direction could also include optimization techniques which use the differentiability of NeRF's density to refine a grasp estimate with local gradient descent. Though we have shown that NeRF is adaptable to geometry deletion, NeRF resists adding new geometry—thus reusing weights will slow down training when objects are added between grasps. Future work in adapting NeRF to changing scenes would greatly improve the practicality of real-time usage.

References

[1] S. S. Sajjan, M. Moore, M. Pan, G. Nagaraja, J. Lee, A. Zeng, and S. Song. Cleargrasp: 3d shape estimation of transparent objects for manipulation. *CoRR*, abs/1910.02550, 2019. URL http://arxiv.org/abs/1910.02550.

[2] J. Ichnowski, Y. Avigal, J. Kerr, and K. Goldberg. Dex-nerf: Using a neural radiance field to grasp transparent objects. In *5th Annual Conference on Robot Learning*, 2021. URL https://openreview.net/forum?id=zOjU2vZzhCk.

[3] 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. In *European Conference on Computer Vision*, pages 405–421. Springer, 2020.

[4] D. Driess, Z. Huang, Y. Li, R. Tedrake, and M. Toussaint. Learning multi-object dynamics with compositional neural radiance fields. *arXiv preprint arXiv:2202.11855*, 2022.

[5] Y. Li, S. Li, V. Sitzmann, P. Agrawal, and A. Torralba. 3d neural scene representations for visuomotor control. In *Conference on Robot Learning*, pages 112–123. PMLR, 2022.

[6] E. Sucar, S. Liu, J. Ortiz, and A. J. Davison. imap: Implicit mapping and positioning in real-time. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6229–6238, 2021.

[7] M. Adamkiewicz, T. Chen, A. Caccavale, R. Gardner, P. Culbertson, J. Bohg, and M. Schwager. Vision-only robot navigation in a neural radiance world. *IEEE Robotics and Automation Letters*, 7(2):4606–4613, 2022.

[8] L. Yen-Chen, P. Florence, J. T. Barron, T.-Y. Lin, A. Rodriguez, and P. Isola. Nerf-supervision: Learning dense object descriptors from neural radiance fields. *arXiv preprint arXiv:2203.01913*, 2022.

[9] C. Sun, M. Sun, and H.-T. Chen. Direct voxel grid optimization: Super-fast convergence for radiance fields reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5459–5469, 2022.

[10] S. Fridovich-Keil, A. Yu, M. Tancik, Q. Chen, B. Recht, and A. Kanazawa. Plenoxels: Radiance fields without neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5501–5510, 2022.

[11] T. Müller, A. Evans, C. Schied, and A. Keller. Instant neural graphics primitives with a multiresolution hash encoding. *ACM Trans. Graph.*, 2022.

[12] K. Park, U. Sinha, J. T. Barron, S. Bouaziz, D. B. Goldman, S. M. Seitz, and R. Martin-Brualla. Nerfies: Deformable neural radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5865–5874, 2021.

[13] Z. Li, S. Niklaus, N. Snavely, and O. Wang. Neural scene flow fields for space-time view synthesis of dynamic scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6498–6508, 2021.

[14] K. Schwarz, Y. Liao, M. Niemeyer, and A. Geiger. Graf: Generative radiance fields for 3D-aware image synthesis. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 33, 2020.

[15] E. R. Chan, M. Monteiro, P. Kellnhofer, J. Wu, and G. Wetzstein. pi-gan: Periodic implicit generative adversarial networks for 3d-aware image synthesis. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 5799–5809, 2021.

[16] J. Gu, L. Liu, P. Wang, and C. Theobalt. Stylenerf: A style-based 3d-aware generator for high-resolution image synthesis. *arXiv preprint arXiv:2110.08985*, 2021.

[17] L. Yen-Chen, P. Florence, J. T. Barron, A. Rodriguez, P. Isola, and T.-Y. Lin. inerf: Inverting neural radiance fields for pose estimation. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1323–1330. IEEE, 2021.

[18] Q. Meng, A. Chen, H. Luo, M. Wu, H. Su, L. Xu, X. He, and J. Yu. Gnerf: Gan-based neural radiance field without posed camera. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6351–6361, 2021.

[19] C.-H. Lin, W.-C. Ma, A. Torralba, and S. Lucey. Barf: Bundle-adjusting neural radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5741–5751, 2021.

[20] A. Chen, Z. Xu, A. Geiger, J. Yu, and H. Su. Tensorf: Tensorial radiance fields. *arXiv preprint arXiv:2203.09517*, 2022.

[21] Q. Wang, Z. Wang, K. Genova, P. Srinivasan, H. Zhou, J. T. Barron, R. Martin-Brualla, N. Snavely, and T. Funkhouser. Ibrnet: Learning multi-view image-based rendering. In *CVPR*, 2021.

[22] A. Trevithick and B. Yang. Grf: Learning a general radiance field for 3d representation and rendering. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 15182–15192, 2021.

[23] A. Chen, Z. Xu, F. Zhao, X. Zhang, F. Xiang, J. Yu, and H. Su. Mvsnerf: Fast generalizable radiance field reconstruction from multi-view stereo. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 14124–14133, 2021.

[24] J. Chibane, A. Bansal, V. Lazova, and G. Pons-Moll. Stereo radiance fields (srf): Learning view synthesis for sparse views of novel scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7911–7920, 2021.

[25] A. Yu, V. Ye, M. Tancik, and A. Kanazawa. pixelnerf: Neural radiance fields from one or few images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4578–4587, 2021.

[26] M. Tancik, B. Mildenhall, T. Wang, D. Schmidt, P. P. Srinivasan, J. T. Barron, and R. Ng. Learned initializations for optimizing coordinate-based neural representations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2846–2855, 2021.

[27] C. Gao, Y. Shih, W.-S. Lai, C.-K. Liang, and J.-B. Huang. Portrait neural radiance fields from a single image. *arXiv preprint arXiv:2012.05903*, 2020.

[28] J. Abou-Chakra, F. Dayoub, and N. Sünderhauf. Implicit object mapping with noisy data. *arXiv preprint arXiv:2204.10516*, 2022.

[29] W.-C. Tseng, H.-J. Liao, L. Yen-Chen, and M. Sun. Cla-nerf: Category-level articulated neural radiance field. *arXiv preprint arXiv:2202.00181*, 2022.

[30] J. Mahler, J. Liang, S. Niyaz, M. Laskey, R. Doan, X. Liu, J. A. Ojea, and K. Goldberg. Dex-net 2.0: Deep learning to plan robust grasps with synthetic point clouds and analytic grasp metrics. 2017.

[31] H. Xu, Y. R. Wang, S. Eppel, A. Aspuru-Guzik, F. Shkurti, and A. Garg. Seeing glass: Joint point cloud and depth completion for transparent objects. *CoRR*, abs/2110.00087, 2021. URL https://arxiv.org/abs/2110.00087.

[32] T. Weng, A. Pallankize, Y. Tang, O. Kroemer, and D. Held. Multi-modal transfer learning for grasping transparent and specular objects. *CoRR*, abs/2006.00028, 2020. URL https://arxiv.org/abs/2006.00028.

[33] Z. Zhu, S. Peng, V. Larsson, W. Xu, H. Bao, Z. Cui, M. R. Oswald, and M. Pollefeys. Nice-slam: Neural implicit scalable encoding for slam. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2022.

[34] A. Yu, S. Fridovich-Keil, M. Tancik, Q. Chen, B. Recht, and A. Kanazawa. Plenoxels: Radiance fields without neural networks. *CoRR*, abs/2112.05131, 2021. URL https://arxiv.org/abs/2112.05131.

[35] V. Satish, J. Mahler, and K. Goldberg. On-policy dataset synthesis for learning robot grasping policies using fully convolutional deep networks. *IEEE Robotics and Automation Letters*, 2019.

[36] X. Zhu, D. Wang, O. Biza, G. Su, R. Walters, and R. Platt. Sample efficient grasp learning using equivariant models. *arXiv preprint arXiv:2202.09468*, 2022.

[37] J. Mahler, F. T. Pokorny, B. Hou, M. Roderick, M. Laskey, M. Aubry, K. Kohlhoff, T. Kröger, J. Kuffner, and K. Goldberg. Dex-net 1.0: A cloud-based network of 3d objects for robust grasp planning using a multi-armed bandit model with correlated rewards. In *2016 IEEE international conference on robotics and automation (ICRA)*, pages 1957–1964. IEEE, 2016.

[38] Y. Zheng and W.-H. Qian. Coping with the grasping uncertainties in force-closure analysis. *The international journal of robotics research*, 24(4):311–327, 2005.

[39] J. Mahler, S. Patil, B. Kehoe, J. Van Den Berg, M. Ciocarlie, P. Abbeel, and K. Goldberg. Gp-gpis-opt: Grasp planning with shape uncertainty using gaussian process implicit surfaces and sequential convex programming. In *2015 IEEE international conference on robotics and automation (ICRA)*, pages 4919–4926. IEEE, 2015.

[40] C. M. Kim, M. Danielczuk, I. Huang, and K. Goldberg. Simulation of parallel-jaw grasping using incremental potential contact models. *arXiv preprint arXiv:2111.01391*, 2021.

[41] J. L. Schönberger and J.-M. Frahm. Structure-from-motion revisited. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[42] J. L. Schönberger, E. Zheng, M. Pollefeys, and J.-M. Frahm. Pixelwise view selection for unstructured multi-view stereo. In *European Conference on Computer Vision (ECCV)*, 2016.

[43] M. Matl. Pyrender. https://github.com/mmatl/pyrender, 2019.

[44] M. Breyer, J. J. Chung, L. Ott, R. Siegwart, and J. I. Nieto. Volumetric grasping network: Real-time 6 DOF grasp detection in clutter. *CoRR*, abs/2101.01132, 2021. URL https://arxiv.org/abs/2101.01132.