AutoNeRF: Training Implicit Scene Representations with Autonomous Agents

Pierre Marza [1]* Laetitia Matignon[2] Olivier Simonin[1] Dhruv Batra[3,5]
Christian Wolf[4] Devendra Singh Chaplot[3]
[1]INSA Lyon [2]UCBL [3]Meta AI [4]Naver Labs Europe [5]Georgia Tech
Project Page: https://pierremarza.github.io/projects/autonerf/

Abstract

Implicit representations such as Neural Radiance Fields (NeRF) have been shown to be very effective at novel view synthesis. However, these models typically require manual and careful human data collection for training. In this paper, we present AutoNeRF, a method to collect data required to train NeRFs using autonomous embodied agents. Our method allows an agent to explore an unseen environment efficiently and use the experience to build an implicit map representation autonomously. We compare the impact of different exploration strategies including handcrafted frontier-based exploration, end-to-end and modular approaches composed of trained high-level planners and classical low-level path followers. We train these models with different reward functions tailored to this problem and evaluate the quality of the learned representations on four different downstream tasks: classical viewpoint rendering, map reconstruction, planning, and pose refinement. Empirical results show that NeRFs can be trained on actively collected data using just a single episode of experience in an unseen environment, and can be used for several downstream robotic tasks, and that modular trained exploration models outperform other classical and end-to-end baselines.

1. Introduction

Exploration is a key challenge in building autonomous navigation agents that operate in unseen environments. In the last few years, there has been a significant amount of work on training exploration policies to maximize coverage [6, 9, 37], find goals specified by object categories [17, 5, 25, 34, 36], images [56, 8, 18, 28] or language [3, 21, 30] and for embodied active learning [7, 4]. Among these methods, modular learning methods have shown to be very effective at various embodied tasks [6, 5, 11, 15]. These methods learn an exploration policy that can build an explicit semantic map of the environment which is then used for planning and downstream embodied AI tasks such as Object Goal or Image Goal Navigation.

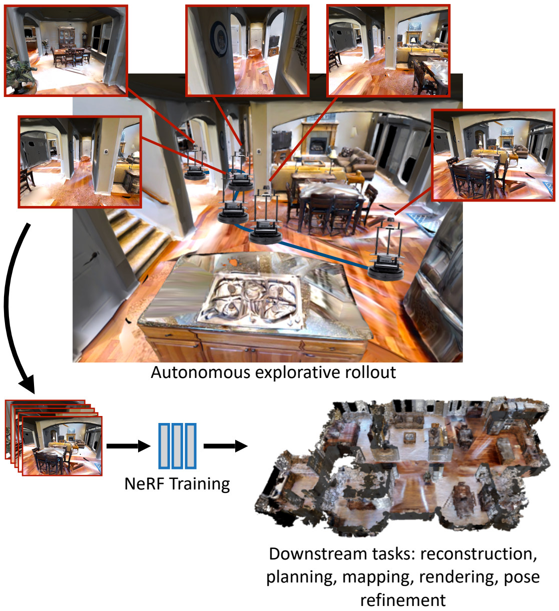
Autonomous explorative rollout
NeRF Training
Downstream tasks: reconstruction, planning, mapping, rendering, pose refinement

Figure 1: We propose a method for automatically generating 3D models of a scene by training NeRFs from data collected by autonomous agents. We compare classical and RL-trained exploration policies, with different reward definitions and evaluate the implicit representations on reconstruction, planning, mapping, rendering, and pose refinement.

Concurrently, in the computer graphics and vision communities, there has been a recent but large body of work on learning implicit map representations, particularly based on Neural Radiance Fields (NeRF) [29, 31, 14, 51, 47]. Prior methods [43, 45, 54, 55] demonstrate strong performance in novel view synthesis and are appealing from a scene understanding point of view as a compact and continuous representation of appearance and semantics in a 3D scene. However, most approaches building implicit representations require data collected by humans [29, 43, 55]. Can we train embodied agents to explore an unseen environment efficiently to collect data that can be used to create implicit map representations or NeRFs autonomously?

*Work done during an internship at Meta AI
Correspondence: pierre.marza@insa-lyon.fr

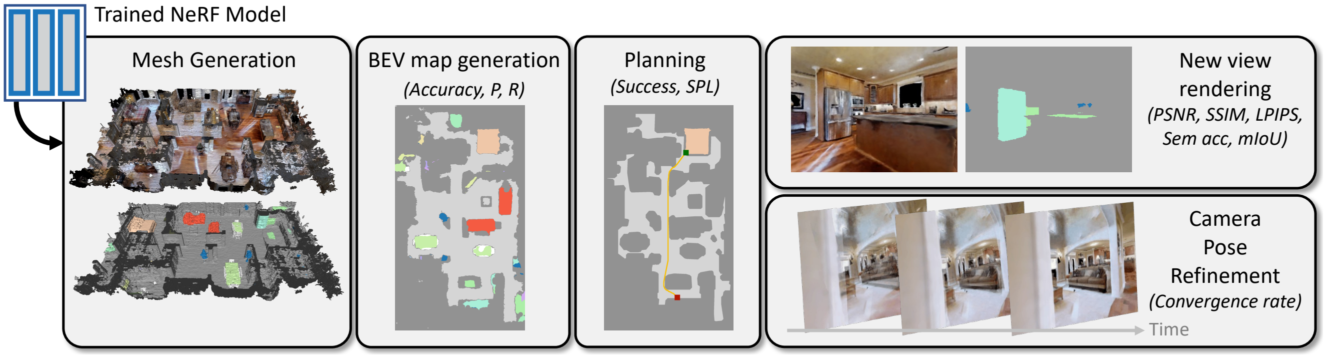

Figure 2: Downstream tasks — the model trained from autonomously collected data is used for several downstream tasks related to robotics: Mesh generation for the covered scene (color or semantic mesh); Birds-eye-view map generation and navigation/planning on this map; new view generation of RGB and semantic frames; camera pose refinement (visual servoing).

In this paper, our objective is to tackle this problem of active exploration for autonomous NeRF construction. If an embodied agent is able to build an implicit map representation autonomously, it can then use it for a variety of downstream tasks such as planning, pose estimation, and navigation. Just a single episode or a few minutes of exploration in an unseen environment can be sufficient to build an implicit representation that can be utilized for improving the performance of the agent in that environment for several tasks without any additional supervision.

In this work, we introduce AutoNeRF, a modular policy trained with Reinforcement Learning (RL) that can explore an unseen 3D scene to collect data for training a NeRF model autonomously (Figure 1). While most prior work evaluates NeRFs on rendering quality, we propose a range of downstream tasks to evaluate them (and indirectly, the exploration policies used to gather data for training these representations) for Embodied AI applications. Specifically, we use geometric and semantic map prediction accuracy, planning accuracy for Object Goal and Point Goal navigation and camera pose refinement (Figure 2). We show that AutoNeRF outperforms the well-known frontier exploration algorithm as well as state-of-the-art end-to-end learnt policies, and also study the impact of different reward functions on the downstream performance of the NeRF model.

2. Related Work

Neural fields — represent the structure of a 3D scene with a neural network. They were initially introduced in [27, 33, 10] as an alternative to discrete representations such as voxels [26], point clouds [13] or meshes [16]. Neural Radiance Fields (NeRF) [29] then introduced a differentiable volume rendering loss allowing to supervise 3D scene reconstruction from 2D supervision, achieving state-of-the-art performance on novel view synthesis. Follow-up work has addressed faster training and inference [31, 14], or training from few images [51]. [47] references advances in this growing field.

Neural fields in robotics — implicit representations are not limited to novel view synthesis, they have also been proposed for real-time SLAM [41, 58, 57]. Initial work [41] required RGB-D input and was deployed on limited-size environments. [58] introduced a hierarchical implicit representation to represent larger scenes, and [57] now only requires RGB input. Extensions incorporate semantics: [54] augmented NERFs with a semantic head trained from sparse and noisy 2D semantic maps. Implicit representations have also been used to represent occupancy, explored area, and semantic objects to navigate towards [24], or as a representation of the density of a scene for drone obstacle avoidance [1]. They have been used for camera pose refinement through SGD directly on a loss in rendered pixel space [50]. In contrast to the literature, we investigate training these representations from data collected by autonomous agents directly and explore the effect of the choice of policy on downstream robotics tasks.

Active learning for neural fields — has not yet been extensively studied. Most works focus on fixed datasets of 2D frames and tackle the active selection of training data. ActiveNeRF [32] estimates the uncertainty of a NeRF model by expressing radiance values as Gaussian distributions. ActiveRMAP [52] minimizes collisions and maximizes an entropy-based information gain metric. These methods target rather small scenes in non-robotic scenarios, either single objects or forward-facing only. In contrast, we start from unknown environments and actively explore large indoor scenes requiring robotic exploration policies capable of handling complex scene understanding and navigation.

Autonomous scene exploration — visual navigation and exploration are well-studied topics in robotics. It is generally defined as a coverage maximization problem, a well-known baseline being Frontier Based Exploration (FBE) [49]. Different variants exist [12, 20, 48] but the core principle is to maintain a frontier between explored and unexplored space and to sample points on the frontier. Several learning-based exploration approaches are also explored in recent work [6, 9, 37, 35]. We target scene exploration with

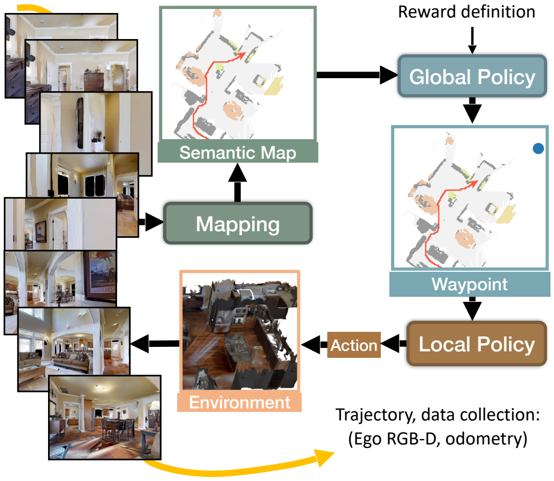

Figure 3: We adapted the modular policy in [5]: a mapping module generates a semantic and occupancy top-down map from egocentric RGB-D observations and sensor pose. A high-level policy trained with RL predicts global waypoints, which are followed by a handcrafted low-level policy (fast marching). The sequence of observations comprises the data input to NeRF training.

a different goal than maximizing vanilla coverage: we study how different definitions of exploration impact the quality of an implicit scene representation.

3. Background

To make the paper self-contained, we first briefly recall relevant background on modular exploration policies, and neural radiance fields.

3.1. Modular exploration policies

The trained policy aims to allow an agent to explore a 3D scene to collect a sequence of 2D RGB and semantic frames as well as camera poses, that will be used to train the continuous scene representation. Following [5, 6], we adapt a modular policy composed of a *Mapping* process that builds a semantic map, a *Global Policy* that outputs a global waypoint from the semantic map as input, and finally, a *Local Policy* that navigates towards the global goal, see Figure 3.

Semantic Map — a 2D top-down map is maintained at each time step t, with several components: (i) an occupancy component $\mathbf{m}_t^{occ} \in \mathbb{R}^{M\times M}$ stores information on free navigable space; (ii) an exploration component $\mathbf{m}_t^{exp} \in \mathbb{R}^{M\times M}$ sets to 1 all cells which have been within the agent's field of view since the beginning of the episode; (iii) a semantic component $\mathbf{m}_t^{sem} \in \mathbb{R}^{S\times M\times M}$, where $M\times M$ is the spatial size and S denotes the number of channels storing information about the scene. Additional maps store the current and previous agent locations. All maps are updated at each timestep from sensor information. Structural components are updated by inverse projection of the current depth observation and pooling to the ground plane, a similar computation is done for the exploration component. The semantic maps additionally use predictions obtained with Mask R-CNN [19]. Egocentric maps are integrated over time taking into account agent poses estimated from sensor information.

Policies — intermediate waypoints are predicted by the *Global Policy*, a convolutional neural network taking as input the stacked maps (we follow [5]) and is trained with RL/PPO [39]. A *Local Policy* navigates towards the waypoint taking discrete actions for 25 steps following the path planned using the *Fast Marching Method* [40].

3.2. Neural radiance fields

Vanilla Semantic NeRF — Neural Radiance Fields [29] are composed of MLPs predicting the density σ, color c and, eventually as in [54], the semantic class s of a particular $3D$ position in space $\mathbf{x} \in \mathbb{R}^3$, given a 2D camera viewing direction $\phi \in \mathbb{R}^2$. NeRFs have been designed to render new views of a scene provided a camera position and viewing direction. The color of a pixel is computed by performing an approximation of volumetric rendering, sampling N quadrature points along the ray. Given multiple images of a scene along with associated camera poses, a NeRF is trained with Stochastic Gradient Descent minimizing the difference between rendered and ground-truth images.

Semantic Nerfacto — we leverage recent advances to train NeRF models faster while maintaining high rendering quality and follow what is done in the Nerfacto model from [44], that we augment with a semantic head. The inputs $\mathbf{x}$ and ϕ are augmented with a learned appearance embedding $\mathbf{e} \in \mathbb{R}^{32}$. Both $\mathbf{x}$ and ϕ are first encoded using respectively a hash encoding function h as $\tilde{\mathbf{x}} = h(\mathbf{x})$ and a spherical harmonics encoding function sh as $\tilde{\phi} = sh(\phi)$. $\tilde{\mathbf{x}}$ is fed to an MLP f_d predicting the density at the given 3D position, yielding $(\sigma, \mathbf{h_d}) = f_d(\tilde{\mathbf{x}}; \Theta_d)$, where $\mathbf{h_d}$ is a latent representation. $\mathbf{h_d}$ is fed to another MLP model f_s that outputs a softmax distribution over the S considered semantic classes as $\mathbf{s} = f_s(\mathbf{h_d}; \Theta_s)$ where $\mathbf{s} \in \mathbb{R}^S$. Finally, $\mathbf{h_d}$, $\tilde{\phi}$ and $\mathbf{e}$ are the inputs to f_c that predicts the RGB value at the given 3D location, $\mathbf{c} = f_c(\mathbf{h_d}, \tilde{\phi}, \mathbf{e}; \Theta_c)$ where $\mathbf{c} \in \mathbb{R}^3$.

4. AutoNeRF

We present AutoNeRF, a method to collect data required to train NeRFs using autonomous embodied agents. In our task setup, the agent is initialized in an unseen environment and is tasked with gathering data in a single episode with a fixed time budget. The observations collected by the agent in this single trajectory are used to train a neural implicit representation of the scene, which will serve as a compact and continuous representation of the density, the RGB appearance, and the semantics of the considered scene. Finally, the trained scene model is evaluated on several downstream tasks in robotics: new view rendering, mapping, planning and pose refinement.

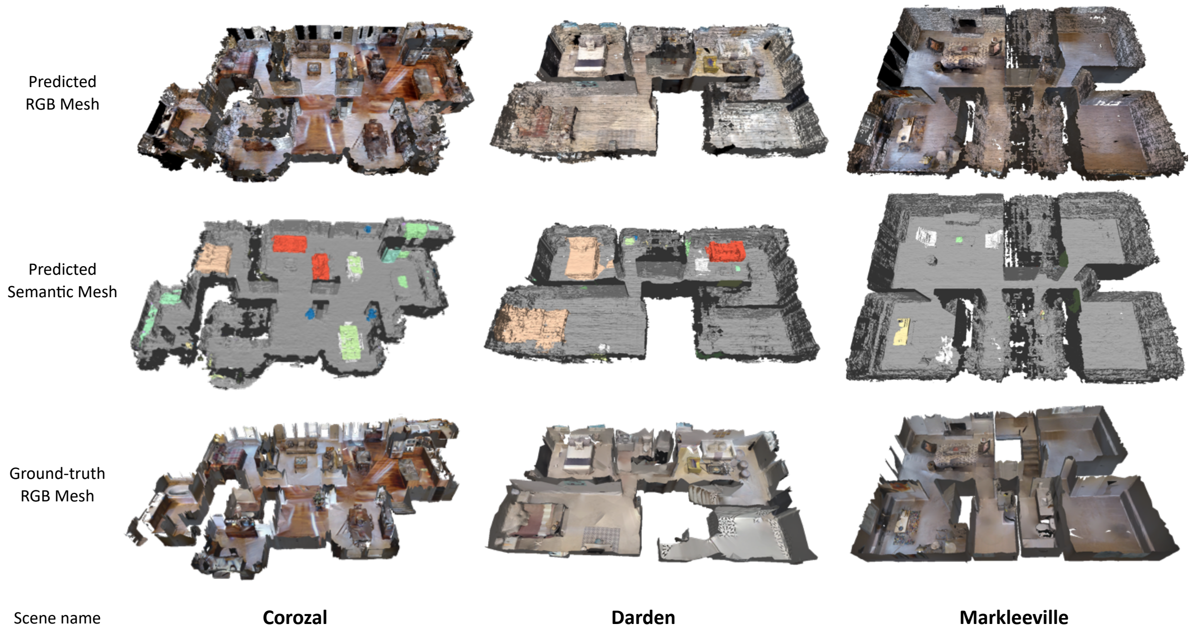

Figure 4: Mesh reconstruction: reconstruction of 3 Gibson val scenes extracted from a NeRF model trained on data gathered by our modular policy. Both geometry, semantics, and appearance are satisfying. Exploration with the modular policy, *Ours (obs).*

Task Specification — The agent is initialized at a random location in an unknown scene and at each timestep t can execute discrete actions in the space $\Lambda = \{$`FORWARD 25cm`, `TURN_LEFT 30°`, `TURN_RIGHT 30°`$\}$. At each step, the agent receives an observation $\mathbf{o}_t$ composed of an egocentric RGB frame and a depth map. The field of view of the agent is $90°$. It also has access to odometry information. The agent can navigate for a limited number of 1500 discrete steps.

AutoNeRF can be broken down into two phases: Exploration Policy Training and NeRF Training. In the first phase, we train an exploration policy to collect the observations. The policy is self-supervised, it is trained in a set of training environments using intrinsic rewards. In the second phase, we use the trained exploration policy to collect data in unseen test scenes, one trajectory per scene, and train a NeRF model using this data. The trained NeRF model is then evaluated on the set of downstream tasks.

4.1. Exploration Policy Training

As described in Section 3.1, we use a modular exploration policy architecture with the *Global Policy* primarily responsible for exploration. We consider different reward signals for training the *Global Policy* tailored to our task of scene reconstruction, and which differ in the importance they give to different aspects of the scene. All these signals are computed in a self-supervised fashion using the metric map representations built by the exploration policy.

Explored area — (*Ours (cov.)*) optimizes the coverage of the scene, i.e. the size of the explored area, and has been proposed in the literature, e.g. in [5, 6]. It accumulates differences in the exploration component $\mathbf{m}_t^{exp}$,

$$r_t^{cov} = \sum_{i=0}^{M-1} \sum_{j=0}^{M-1} \mathbf{m}_t^{exp}[i,j] - \mathbf{m}_{t-1}^{exp}[i,j]$$

Obstacle coverage — (*Ours (obs.)*) optimizes the coverage of obstacles in the scene, and accumulates differences in the corresponding component $\mathbf{m}_{t-1}^{occ}[i,j]$. It targets tasks where obstacles are considered more important than navigable floor space, which is arguably the case when viewing is less important than navigating.

$$r_t^{obs} = \sum_{i=0}^{M-1} \sum_{j=0}^{M-1} \mathbf{m}_t^{occ}[i,j] - \mathbf{m}_{t-1}^{occ}[i,j]$$

Semantic object coverage — (*Ours (sem.)*) optimizes the coverage of the S semantic classes detected and segmented in the semantic metric map $\mathbf{m}_t^{sem}$. This reward removes obstacles that are not explicitly identified as a notable semantic class — see section 5 for their definition.

$$r_t^{sem} = \sum_{i=0}^{M-1} \sum_{j=0}^{M-1} \sum_{k=0}^{S-1} \mathbf{m}_t^{sem}[i,j,k] - \mathbf{m}_{t-1}^{sem}[i,j,k]$$

Viewpoints coverage — (*Ours (view.)*) optimizes for the usage of the trained implicit representation as a dense and continuous representation of the scene usable to render arbitrary new viewpoints, either for later visualization as its own downstream task or for training new agents in simulation. To this end, we propose to maximize coverage not only in terms of agent positions but also in terms of agent viewpoints. Compared to [5], we introduce an additional 3D map $\mathbf{m}^{view}[i, j, k]$, where the first two dimensions correspond to spatial 2D positions in the scene and the third dimension corresponds to a floor plane angle of the given cell discretized into V=12 bins. A value of $\mathbf{m}_t^{view}[i, j, k] = 1$ indicates that cell (i, j) has been seen by the agent from a (discretized) angle k. The reward maximizes its changes,

$$r_t^{view} = \sum_{i=0}^{M-1} \sum_{j=0}^{M-1} \sum_{k=0}^{V-1} \mathbf{m}_t^{view}[i, j, k] - \mathbf{m}_{t-1}^{view}[i, j, k]$$

4.2. NeRF training

The sequence of observations collected by the agent comprises egocentric RGB frames $\{\mathbf{o}_t\}_{t=1...T}$, first-person semantic segmentations $\{\mathbf{s}_t\}_{t=1...T}$ and associated poses $\{\mathbf{p}_t\}_{t=1...T}$ in a reference frame, which we define as the starting position t=0 of each episode. In our experiments, we leverage privileged pose and semantics information from simulation. We also conduct an experiment showcasing the difference between using GT semantics from a simulator and a Mask R-CNN [19] model.

An important property of this procedure is that no depth information is required for reconstruction. The implicit representation is trained by mapping pixel coordinates $\mathbf{x}_i$ for each pixel i to RGB values and semantic values with the volume rendering loss described in Section 3.2. The input coordinates $\mathbf{x}_i$ are obtained using the global poses $\mathbf{p}_t$ and intrinsics from calibrated cameras.

4.3. Downstream tasks

Prior work on implicit representations generally focused on two different settings: (i) evaluating the quality of a neural field based on its new view rendering abilities given a dataset of (carefully selected) training views, and (ii) evaluating the quality of a scene representation in robotics conditioned on given (constant) trajectories, evaluated as reconstruction accuracy. We cast this task in a more holistic way and more aligned with our scene understanding objective. We evaluate the impact of trajectory generation (through exploration policies) directly on the quality of the representation, which we evaluate in a goal-oriented way through multiple tasks related to robotics (cf. Figure 2).

Task 1: Rendering — This task is the closest to the evaluation methodology prevalent in the neural field literature. We evaluate the rendering of RGB and semantic frames as proposed in [54]. Unlike the common method of evaluating an implicit representation on a subset of frames within the trajectory, we evaluate it on a set of uniformly sampled camera poses within the scene, independently of the trajectory taken by the policy. This allows us to evaluate the representation of the complete scene and not just its interpolation ability.

We render ground-truth images and semantic masks associated with sampled camera poses using the Habitat [38, 42] simulator and compare them against the NeRF rendering. RGB rendering metrics are PSNR (*Peak Signal-to-Noise Ratio*), SSIM (*Structural Similarity Index Measure*) and LPIPS [53]. Rendering of semantics is evaluated in terms of average per-class accuracy and mean intersection over union (mIoU).

Task 2: Metric Map Estimation — While rendering quality is linked to the perception of the scene, it is not necessarily a good indicator of its structural content, which is crucial for robotic downstream tasks. We evaluate the quality of the estimated structure by translating the continuous representation into a format, which is very widely used in map-and-plan baselines for navigation, a top-down (bird's-eye-view=BEV) map storing occupancy and semantic category information and compare it with the ground-truth from the simulator. We evaluate obstacle and semantic maps using accuracy, precision, and recall.

Task 3: Planning — Using maps for navigation, it is difficult to pinpoint the exact precision required for successful planning, as certain artifacts and noises might not have a strong impact on reconstruction metrics, but could lead to navigation problems, and vice-versa. We perform goal-oriented evaluation and measure to what extent path planning can be done on the obtained top-down maps.

We sample 100 points on each scene and plan from those starting points to two different types of goals: to selected end positions, *PointGoal* planning, and to objects categories, *ObjectGoal* planning. The latter, *ObjectGoal*, requires planning the shortest path from the given starting point to the closest object of each semantic class available on the given scene. For both tasks, we plan with the *Fast Marching Method* and report both mean *Success* and *SPL* as introduced in [2]. For a given episode, *Success* is 1 if planning stops less than 1m from from the goal, and *SPL* measures path efficiency.

Task 4: Pose Refinement — This task introduced in [50] involves correcting an initial noisy camera position and associated rendered view and optimizing the position until a given ground-truth position is reached, which is given through its associated rendered view only. The optimization process therefore leads to a trajectory in camera pose space. This task is closely linked to visual servoing with a "eye-in-hand" configuration, a standard problem in robotics, in particular in its "direct" variant [23], where the optimization is directly performed over losses on the observed pixel space.

We address this problem by taking the trained NeRF model f and freezing its weights θ. In what follows, we will denote the function rendering a full image $\mathbf{o}$ given camera pose c and viewing direction ϕ as $\mathbf{o} = \mathcal{R}(c, \phi)$. Then, given

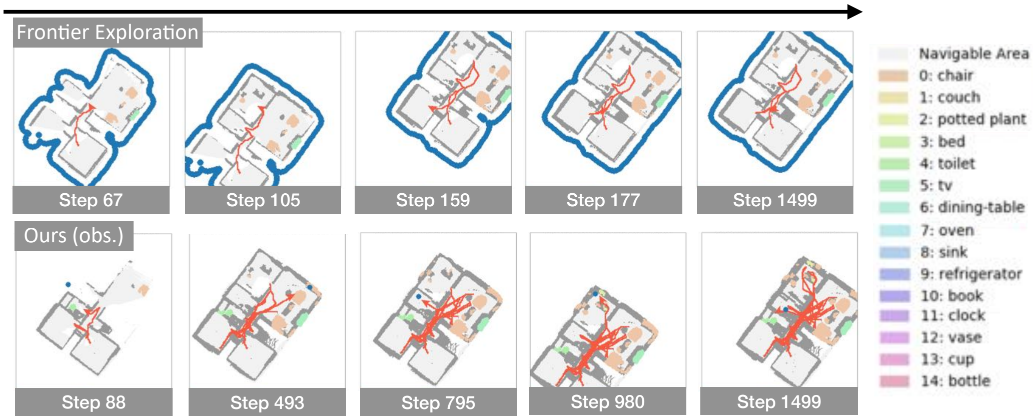

Figure 5: Rollouts by Frontier Based exploration vs. Modular policy (obs cov): FBE properly covers the scene, but does not collect a large diversity of viewpoints, while the modular policy enters the rooms and thus provides richer training data for the neural field.

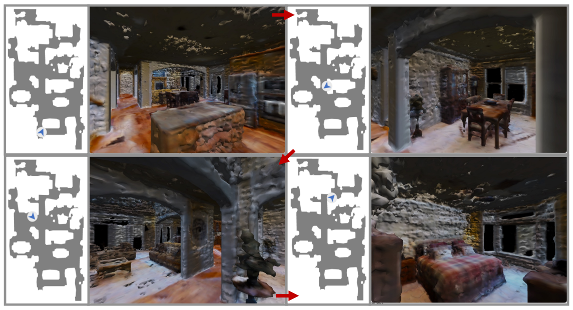

Figure 6: Navigating in the Habitat simulator: the underlying mesh was extracted from the trained NeRF, *Ours (cov)*. Rendering quality and the generated BEV map are correct, as are free navigable space and collision handling. Temporal order indicated by →.

a ground truth view $\mathbf{o}^*$, the camera position and direction can be directly optimized from a starting position $(c,\phi)^{[0]}$ with gradient descent as

$$(c,\phi)^{[t+1]} = (c,\phi)^{[t]} + \nu \left[\frac{\partial \mathcal{L}(\mathbf{o}^*, \mathcal{R}(c,\phi)}{\partial c, \phi} \right],$$

where $\mathcal{L}$ is the MSE (Mean Squared Error) loss and ν is a learning rate.

To generate episodes of starting and end positions, we take 100 sampled camera poses in each scene and apply a random transformation to generate noisy poses. The model is evaluated in terms of rotation and translation convergence rate, i.e. percentage of samples where the final difference with ground truth is less than 3° in rotation and $2cm$ in translation. We also report the mean translation and rotation errors for the converged samples.

5. Experimental Results

Modular Policy training — is performed on one V100 GPU for 7 days. All modular policies are trained on the 25 scenes of the Gibson [46]-tiny training set. The used Mask R-CNN model is pre-trained on the MS COCO dataset [22] and finetuned on Gibson train scenes. We consider $S{=}15$ semantic categories: {*chair*, *couch*, *potted plant*, *bed*, *toilet*, *tv*, *dining table*, *oven*, *sink*, *refrigerator*, *book*, *clock*, *vase*, *cup*, *bottle*}.

External baselines — We compare our trained modular policies against the classical frontier-based exploration algorithm (*Frontier*), as well as end-to-end policies trained with RL. More specifically, we consider 4 end-to-end policies from [35], that all share the same architecture but were trained with different exploration-related reward functions: coverage (*E2E (cov.)*), curiosity (*E2E (cur.)*), novelty (*E2E (nov.)*), reconstruction (*E2E (rec.)*). Reward functions are presented in [35].

Evaluation — consists in running 5 rollouts with different start positions in each of the 5 Gibson-tiny val scenes for each policy, always on the first house floor. A NeRF model is then trained on each trajectory data.

NeRF models — In our experiments, we consider two different NeRF variants presented in 3.2. Most experiments are conducted with *Semantic Nerfacto*, as it provides a great trade-off between training speed and quality of representation. *Semantic Nerfacto* is built on top of the *Nerfacto* model from the nerfstudio [44] library. We augment the model with a semantic head and implement evaluation on test camera poses independently from the collected trajectory. Only the next subsection (5.2) will involve training a *vanilla Semantic NeRF* model, more precisely the one introduced in [54] that also contains a semantic head. We chose this variant for these specific experiments to illustrate the possibility of providing high-fidelity representations of complex scenes, and show that a *vanilla Semantic NeRF* model trained for a longer time (12h) leads to better-estimated geometry. Results from *Semantic Nerfacto* are still very good (see Figures 7 and 8) but we found meshes to be higher quality with a vanilla NeRF model.

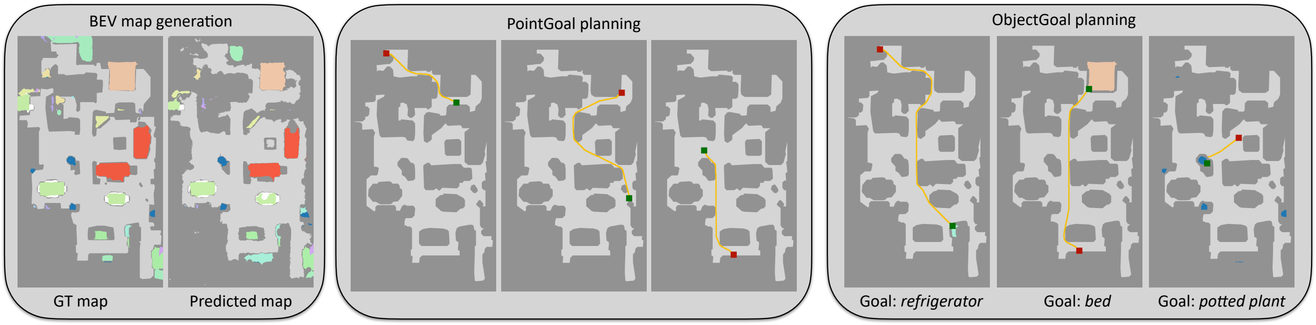

Figure 7: BEV map tasks: Generation of semantic BEV maps (Left), *PointGoal* (Middle) and *ObjectGoal* planning (Right).

Policy	Success ↑	SPL ↑
Finetuned on Gibson meshes (not comparable)†	99.7	97.9
Pre-trained (no finetuning)	90.2	82.9
Finetuned on AutoNeRF meshes	**92.9**	**86.7**

Table 1: PointGoal Finetuning: finetuning a PointGoal agent on a mesh automatically collected from a rollout and a NERF with AutoNERF improves mean performance over a pre-trained policy on a specific scene. † an upper bound which finetunes on the original mesh. In a real use case involving a robot automatically collecting data, this mesh would not be available (not comparable).

	RGB			Semantics	
Policy	PSNR ↑	SSIM ↑	LPIPS ↓	Per-class acc ↑	mIoU ↑
Frontier	19.75	0.743	0.343	81.4	65.7
E2E (cov.)	20.94	0.750	0.332	80.1	63.9
E2E (cur.)	20.60	0.747	0.338	78.7	61.9
E2E (nov.)	23.36	0.801	0.268	84.6	71.4
E2E (rec.)	23.17	0.797	0.270	84.1	70.5
Ours (cov.)	24.89	0.837	0.218	90.2	81.2
Ours (sem.)	25.34	0.843	0.207	**91.9**	81.8
Ours (obs.)	**25.56**	**0.846**	**0.203**	**91.8**	**83.2**
Ours (view.)	25.17	0.842	0.211	91.3	82.0

Table 2: Rendering performance on uniformed sampled viewpoints of the full scene after training on a single trajectory.

	Occupancy			Semantics		
Policy	Acc. ↑	Prec. ↑	Rec. ↑	Acc ↑	Prec. ↑	Rec. ↑
Frontier	81.2	86.9	49.9	99.7	26.6	21.0
E2E (cov.)	77.1	86.2	50.4	99.7	22.1	16.1
E2E (cur.)	81.8	90.3	50.7	99.7	19.2	12.5
E2E (nov.)	83.1	88.7	61.3	99.7	25.5	18.3
E2E (rec.)	81.6	87.6	60.0	99.7	26.2	18.0
Ours (cov.)	86.8	89.1	74.7	99.8	35.1	27.1
Ours (sem.)	86.6	88.3	76.5	99.8	35.7	29.8
Ours (obs.)	86.4	89.4	76.5	99.8	36.2	29.8
Ours (view.)	**88.1**	**90.9**	**77.0**	99.8	**37.4**	**30.2**

Table 3: Map Estimation Performance: comparison of BEV maps estimated from the NeRF.

5.1. Reconstructing house-scale scenes

We illustrate the possibility of autonomously reconstructing complex large-scale environments such as apartments or houses from the continuous representations trained on data collected by agents exploring the scene using the modular policy. Figure 4 shows RGB and semantic meshes for 3 Gibson val scenes. Geometry, appearance, and semantics are satisfying. In Figure 6 we show that such meshes can be loaded into the Habitat simulator and allow proper navigation and collision computations. Both occupancy top-down map generation and RGB renderings are performed by the Habitat simulator from the generated mesh.

5.2. Autonomous adaptation to a new scene

A long-term goal of Embodied AI is to train general policies that can be deployed on any new scene. Even such agents will likely struggle with some specificities of a given environment, and a scene-specific adaptation thus appears as a relevant solution. We explore the usage of AutoNERF to explore an environment to build a 3D representation, which is then loaded into a simulator to safely finetune a policy of interest. More specifically, we consider a depth-only PointGoal navigation policy pre-trained on Gibson. It is finetuned on 4 Gibson val scenes, using meshes generated with AutoNeRF, before being evaluated on the original Gibson meshes. Details about episodes sampling and training hyperparameters are given in the Supplementary Material.

As shown in Table 1, scene-specific finetuning on autonomously reconstructed 3D meshes allows to improve both Success and SPL. We also compare with finetuning directly on the Gibson mesh, which provides a non-comparable soft upper bound — in a real robotics scenario, these meshes would not be accessible. This shows that performance could still be improved, but it is important to note that reaching the performance of the upper bound might be about reconstructing fine details.

5.3. Quantitative results

Frontier Exploration vs Modular Policy — as can be seen from the quantitative comparisons on the different downstream tasks (Tables 2, 3, 4, 5), RL-trained modular policies outperform frontier exploration on all metrics and should thus be considered as the preferred means of collecting NeRF data. This is a somewhat surprising result, since Frontier Based Exploration generally performs satisfying visual coverage of the scene, even though it can sometimes get stuck because of map inaccuracies. This

Policy	PointGoal		ObjectGoal	
	Succ. ↑	SPL ↑	Succ. ↑	SPL ↑
Frontier	22.4	21.4	9.6	9.1
E2E (cov.)	30.0	29.3	8.9	8.3
E2E (cur.)	29.8	29.2	8.5	8.0
E2E (nov.)	32.3	31.9	11.4	10.8
E2E (rec.)	32.8	32.6	10.5	10.0
Ours (cov.)	**39.5**	**39.0**	14.8	14.3
Ours (sem.)	37.7	37.4	**16.0**	**15.4**
Ours (obs.)	38.2	37.8	**15.8**	**15.3**
Ours (view.)	39.0	38.6	**15.9**	**15.3**

Table 4: Planning performance on the BEV maps estimated from the NeRF obtained with the *Fast Marching* method.

Policy	Conv. rate ↑	Rot. Error (°) ↓	Trans. Error (m)↓
Frontier	7.2	0.383	0.00955
E2E (cov.)	15.4	0.319	0.00775
E2E (cur.)	12.5	0.325	0.00799
E2E (nov.)	19.4	0.315	0.00774
E2E (rec.)	19.3	0.292	0.00734
Ours (cov.)	20.2	**0.283**	**0.00734**
Ours (sem.)	**23.0**	0.319	0.00784
Ours (obs.)	22.5	0.305	0.00765
Ours (view.)	21.1	0.316	0.00769

Table 5: Pose Refinement: optimizing camera viewpoints given a rendered target viewpoint.

Task	Metrics	Sim.	Mask R-CNN
Rendering	Per-class acc.	91.8	65.4
	mIoU	83.2	61.1
Map comparison	Sem acc.	99.8	99.7
	Sem prec.	36.2	14.1
	Sem rec.	29.8	8.5
Planning	ObjGoal Succ.	15.8	6.8
	ObjGoal SPL	15.3	6.5

Table 6: NeRF semantic maps: impact of the choice of ground-truth semantics vs. semantics estimated by Mask R-CNN when data is collected by *Ours (obs.)*.

shows that vanilla visual coverage, the optimized metrics in many exploration-oriented tasks, is not a sufficient criterion to collect data for NeRF training. Figure 5 illustrates this point with rollouts from FBE and a modular policy trained to maximize obstacle coverage. FBE properly covers the scene but does not necessarily cover a large diversity of viewpoints, while the modular policy provides richer training data to the NeRF.

End-to-end Policy vs Modular Policy — Tables 2, 3, 4, 5 also show that the modular policies outperform end-to-end RL policies on all considered metrics. Interestingly, *novelty* and *reconstruction* seem to be the best reward functions when training end-to-end policies if the final goal is to autonomously collect data to build a NeRF model.

Comparing trained policies — Rewarding modular policies with obstacles (*Ours (obs.)*) and viewpoints (*Ours (view.)*) coverage appears to lead to the best overall performance when we consider the different metrics. Explored area coverage (*Ours (cov.)*) leads to highest *PointNav* performance, corroborating its importance for geometric tasks, whereas other semantic reward functions lead to higher *ObjectNav* performance, again corroborating its importance for semantic understanding of the scene.

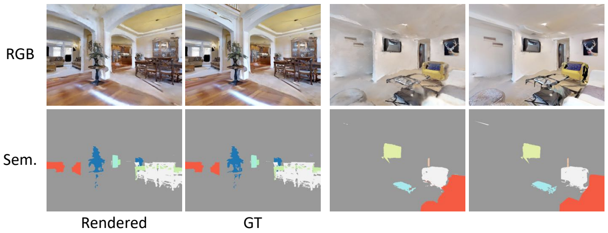

Figure 8: Quality of semantic rendering on pairs of images of different scenes, compared with GT from Sim. *Ours (obs)*.

Semantics from Mask R-CNN — Table 6 shows the impact of using Mask R-CNN to compute the semantics training data of the NeRF model vs semantics from simulation. As expected, performance drops because Mask R-CNN provides a much noisier training signal, which could partly be explained by the visual domain gap between the real world and simulators. However, performance on the different downstream tasks is still reasonable, showing that one could autonomously collect data and generate semantics training signal without requiring additional annotation.

5.4. Qualitative results

BEV maps — Figure 7 gives examples of the BEV maps generated from the continuous representation: structural details and dense semantic information are nicely recovered (Left). Planned trajectories are close to the shortest paths, for both PointGoal tasks (Middle) and ObjectGoal (Right).

Semantic rendering — Figure 8 compares the segmentation maps and RGB frames rendered with the continuous representation (trained with semantic masks from simulation) to the GT maps from the simulator. Again, the structure of the objects and even fine details are well recovered, and only very local noise is visible in certain areas. The semantic reconstruction is satisfying.

6. Conclusion

This work introduces a task involving navigating in a 3D environment to collect NeRF training data. We show that RL-trained modular policies outperform classic Frontier Based Exploration as well as other end-to-end RL baselines on this task, and compare different training reward functions. We also suggest evaluating NeRF from a scene-understanding point of view and with robotics-oriented tasks: BEV map generation, planning, rendering, and camera pose refinement. Finally, we show that it is possible with the considered method to reconstruct house-scale scenes. Interesting future work could target fine-tuning navigation models automatically on a scene.

References

[1] Michal Adamkiewicz, Timothy Chen, Adam Caccavale, Rachel Gardner, Preston Culbertson, Jeannette Bohg, and Mac Schwager. Vision-only robot navigation in a neural radiance world. *RA-L*, 2022. 2

[2] Peter Anderson, Angel X. Chang, Devendra Singh Chaplot, Alexey Dosovitskiy, Saurabh Gupta, Vladlen Koltun, Jana Kosecka, Jitendra Malik, Roozbeh Mottaghi, Manolis Savva, and Amir Roshan Zamir. On evaluation of embodied navigation agents. *arXiv preprint*, 2018. 5

[3] Peter Anderson, Qi Wu, Damien Teney, Jake Bruce, Mark Johnson, Niko Sünderhauf, Ian Reid, Stephen Gould, and Anton Van Den Hengel. Vision-and-language navigation: Interpreting visually-grounded navigation instructions in real environments. In *CVPR*, 2018. 1

[4] Devendra Singh Chaplot, Murtaza Dalal, Saurabh Gupta, Jitendra Malik, and Russ R Salakhutdinov. Seal: Self-supervised embodied active learning using exploration and 3d consistency. In *NeurIPS*, 2021. 1

[5] Devendra Singh Chaplot, Dhiraj Gandhi, Abhinav Gupta, and Ruslan Salakhutdinov. Object goal navigation using goal-oriented semantic exploration. In *NeurIPS*, 2020. 1, 3, 4, 5

[6] Devendra Singh Chaplot, Dhiraj Gandhi, Saurabh Gupta, Abhinav Gupta, and Ruslan Salakhutdinov. Learning to explore using active neural slam. In *ICLR*, 2020. 1, 2, 3, 4

[7] Devendra Singh Chaplot, Helen Jiang, Saurabh Gupta, and Abhinav Gupta. Semantic curiosity for active visual learning. In *ECCV*, 2020. 1

[8] Devendra Singh Chaplot, Ruslan Salakhutdinov, Abhinav Gupta, and Saurabh Gupta. Neural topological slam for visual navigation. In *CVPR*, 2020. 1

[9] Tao Chen, Saurabh Gupta, and Abhinav Gupta. Learning exploration policies for navigation. In *ICLR*, 2019. 1, 2

[10] Zhiqin Chen and Hao Zhang. Learning implicit fields for generative shape modeling. In *CVPR*, 2019. 2

[11] Matt Deitke, Dhruv Batra, Yonatan Bisk, Tommaso Campari, Angel X Chang, Devendra Singh Chaplot, Changan Chen, Claudia Pérez D'Arpino, Kiana Ehsani, Ali Farhadi, et al. Retrospectives on the embodied ai workshop. *arXiv preprint*, 2022. 1

[12] Christian Dornhege and Alexander Kleiner. A frontier-void-based approach for autonomous exploration in 3d. *Advanced Robotics*, 2013. 2

[13] Haoqiang Fan, Hao Su, and Leonidas J Guibas. A point set generation network for 3d object reconstruction from a single image. In *CVPR*, 2017. 2

[14] Stephan J Garbin, Marek Kowalski, Matthew Johnson, Jamie Shotton, and Julien Valentin. Fastnerf: High-fidelity neural rendering at 200fps. In *ICCV*, 2021. 1, 2

[15] Theophile Gervet, Soumith Chintala, Dhruv Batra, Jitendra Malik, and Devendra Singh Chaplot. Navigating to objects in the real world. *arXiv preprint*, 2022. 1

[16] Thibault Groueix, Matthew Fisher, Vladimir G Kim, Bryan C Russell, and Mathieu Aubry. A papier-mâché approach to learning 3d surface generation. In *CVPR*, 2018. 2

[17] Saurabh Gupta, James Davidson, Sergey Levine, Rahul Sukthankar, and Jitendra Malik. Cognitive mapping and planning for visual navigation. In *CVPR*, 2017. 1

[18] Meera Hahn, Devendra Singh Chaplot, Shubham Tulsiani, Mustafa Mukadam, James M Rehg, and Abhinav Gupta. No rl, no simulation: Learning to navigate without navigating. In *NeurIPS*, 2021. 1

[19] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask R-CNN. In *ICCV*, 2017. 3, 5

[20] Dirk Holz, Nicola Basilico, Francesco Amigoni, and Sven Behnke. Evaluating the efficiency of frontier-based exploration strategies. In *ISR*, 2010. 2

[21] Jacob Krantz, Erik Wijmans, Arjun Majumdar, Dhruv Batra, and Stefan Lee. Beyond the nav-graph: Vision-and-language navigation in continuous environments. In *ECCV*, 2020. 1

[22] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *ECCV*, 2014. 6

[23] Eric Marchand. Direct visual servoing in the frequency domain. *RA-L*, 2020. 5

[24] Pierre Marza, Laetitia Matignon, Olivier Simonin, and Christian Wolf. Multi-object navigation with dynamically learned neural implicit representations. *arXiv preprint arXiv:2210.05129*, 2022. 2

[25] Pierre Marza, Laetitia Matignon, Olivier Simonin, and Christian Wolf. Teaching agents how to map: Spatial reasoning for multi-object navigation. In *IROS*, 2022. 1

[26] Daniel Maturana and Sebastian Scherer. Voxnet: A 3d convolutional neural network for real-time object recognition. In *IROS*, 2015. 2

[27] Lars Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger. Occupancy networks: Learning 3d reconstruction in function space. In *CVPR*, 2019. 2

[28] Lina Mezghan, Sainbayar Sukhbaatar, Thibaut Lavril, Oleksandr Maksymets, Dhruv Batra, Piotr Bojanowski, and Karteek Alahari. Memory-augmented reinforcement learning for image-goal navigation. In *IROS*, 2022. 1

[29] Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In *ECCV*, 2020. 1, 2, 3

[30] So Yeon Min, Devendra Singh Chaplot, Pradeep Ravikumar, Yonatan Bisk, and Ruslan Salakhutdinov. Film: Following instructions in language with modular methods. In *ICLR*, 2022. 1

[31] Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. Instant neural graphics primitives with a multiresolution hash encoding. 2022. 1, 2

[32] Xuran Pan, Zihang Lai, Shiji Song, and Gao Huang. Activenerf: Learning where to see with uncertainty estimation. In *ECCV*, 2022. 2

[33] Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. Deepsdf: Learning continuous signed distance functions for shape representation. In *CVPR*, 2019. 2

[34] Santhosh K. Ramakrishnan, Devendra Singh Chaplot, Ziad Al-Halah, Jitendra Malik, and Kristen Grauman. Poni: Potential functions for objectgoal navigation with interaction-free learning. In *CVPR*, 2022. 1

[35] Santhosh K Ramakrishnan, Dinesh Jayaraman, and Kristen Grauman. An exploration of embodied visual exploration. *International Journal of Computer Vision*, 2021. 2, 6

[36] Ram Ramrakhya, Eric Undersander, Dhruv Batra, and Abhishek Das. Habitat-web: Learning embodied object-search strategies from human demonstrations at scale. In *CVPR*, 2022. 1

[37] Nikolay Savinov, Anton Raichuk, Damien Vincent, Raphael Marinier, Marc Pollefeys, Timothy Lillicrap, and Sylvain Gelly. Episodic curiosity through reachability. In *ICLR*, 2018. 1, 2

[38] Manolis Savva, Abhishek Kadian, Oleksandr Maksymets, Yili Zhao, Erik Wijmans, Bhavana Jain, Julian Straub, Jia Liu, Vladlen Koltun, Jitendra Malik, Devi Parikh, and Dhruv Batra. Habitat: A platform for embodied ai research. In *ICCV*, 2019. 5

[39] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint*, 2017. 3

[40] James A Sethian. A fast marching level set method for monotonically advancing fronts. 1996. 3

[41] Edgar Sucar, Shikun Liu, Joseph Ortiz, and Andrew J Davison. imap: Implicit mapping and positioning in real-time. In *ICCV*, 2021. 2

[42] Andrew Szot, Alex Clegg, Eric Undersander, Erik Wijmans, Yili Zhao, John Turner, Noah Maestre, Mustafa Mukadam, Devendra Chaplot, Oleksandr Maksymets, Aaron Gokaslan, Vladimir Vondrus, Sameer Dharur, Franziska Meier, Wojciech Galuba, Angel Chang, Zsolt Kira, Vladlen Koltun, Jitendra Malik, Manolis Savva, and Dhruv Batra. Habitat 2.0: Training home assistants to rearrange their habitat. In *NeurIPS*, 2021. 5

[43] Matthew Tancik, Vincent Casser, Xinchen Yan, Sabeek Pradhan, Ben Mildenhall, Pratul P Srinivasan, Jonathan T Barron, and Henrik Kretzschmar. Block-nerf: Scalable large scene neural view synthesis. In *CVPR*, 2022. 1

[44] Matthew Tancik, Ethan Weber, Evonne Ng, Ruilong Li, Brent Yi, Justin Kerr, Terrance Wang, Alexander Kristoffersen, Jake Austin, Kamyar Salahi, et al. Nerfstudio: A modular framework for neural radiance field development. *arXiv preprint*, 2023. 3, 6

[45] Suhani Vora, Noha Radwan, Klaus Greff, Henning Meyer, Kyle Genova, Mehdi SM Sajjadi, Etienne Pot, Andrea Tagliasacchi, and Daniel Duckworth. Nesf: Neural semantic fields for generalizable semantic segmentation of 3d scenes. *arXiv preprint*, 2021. 1

[46] Fei Xia, Amir R Zamir, Zhiyang He, Alexander Sax, Jitendra Malik, and Silvio Savarese. Gibson env: Real-world perception for embodied agents. In *CVPR*, 2018. 6

[47] Yiheng Xie, Towaki Takikawa, Shunsuke Saito, Or Litany, Shiqin Yan, Numair Khan, Federico Tombari, James Tompkin, Vincent Sitzmann, and Srinath Sridhar. Neural fields in visual computing and beyond. *arXiv preprint*, 2021. 1, 2

[48] Kai Xu, Lintao Zheng, Zihao Yan, Guohang Yan, Eugene Zhang, Matthias Niessner, Oliver Deussen, Daniel Cohen-Or, and Hui Huang. Autonomous reconstruction of unknown indoor scenes guided by time-varying tensor fields. 2017. 2

[49] Brian Yamauchi. A frontier-based approach for autonomous exploration. In *CIRA*, 1997. 2

[50] Lin Yen-Chen, Pete Florence, Jonathan T Barron, Alberto Rodriguez, Phillip Isola, and Tsung-Yi Lin. inerf: Inverting neural radiance fields for pose estimation. In *IROS*, 2021. 2, 5

[51] Alex Yu, Vickie Ye, Matthew Tancik, and Angjoo Kanazawa. pixelnerf: Neural radiance fields from one or few images. In *CVPR*, 2021. 1, 2

[52] Huangying Zhan, Jiyang Zheng, Yi Xu, Ian Reid, and Hamid Rezatofighi. Activermap: Radiance field for active mapping and planning. 2022. 2

[53] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *CVPR*, 2018. 5

[54] Shuaifeng Zhi, Tristan Laidlow, Stefan Leutenegger, and Andrew J Davison. In-place scene labelling and understanding with implicit scene representation. In *ICCV*, 2021. 1, 2, 3, 5, 6

[55] Shuaifeng Zhi, Edgar Sucar, Andre Mouton, Iain Haughton, Tristan Laidlow, and Andrew J Davison. ilabel: Interactive neural scene labelling. *arXiv preprint*, 2021. 1

[56] Yuke Zhu, Roozbeh Mottaghi, Eric Kolve, Joseph J Lim, Abhinav Gupta, Li Fei-Fei, and Ali Farhadi. Target-driven visual navigation in indoor scenes using deep reinforcement learning. In *ICRA*, 2017. 1

[57] Zihan Zhu, Songyou Peng, Viktor Larsson, Zhaopeng Cui, Martin R Oswald, Andreas Geiger, and Marc Pollefeys. Nicer-slam: Neural implicit scene encoding for rgb slam. *arXiv preprint*, 2023. 2

[58] Zihan Zhu, Songyou Peng, Viktor Larsson, Weiwei Xu, Hujun Bao, Zhaopeng Cui, Martin R Oswald, and Marc Pollefeys. Nice-slam: Neural implicit scalable encoding for slam. *CVPR*, 2022. 2