Robust Multi-Agent Cooperative Active Sampling System for Efficient Robotic Online Learning

Anonymous Author(s)
Submission Id: 1331*

Abstract

Online training tasks deployed over mobile robots (robotic online training) have the potential to automatically collect high quality training data to improve the performance of their training Artificial Intelligence (AI) model. While active learning methods aim at finding the training data with highest quality, they fail to produce a navigation path with highest accumulated information gain for the online training tasks because they fail to model the evolution of the training AI model when new training data are collected along the navigation path. To overcome this problem, in this paper, we propose LOss-Dynamics-Aware multi-agent cooperative active sampling system for robotic online training (LODA) that leverages a dynamic perspective of the training AI model that models the evolution of the training AI model and the corresponding information gain as the dynamics of loss value rendered to a sparse 3D grid corresponding to the environment. Using this representation, we are able to recursively predict the information gain of each step taken along a candidate path, and find the optimal path with highest accumulated information gain. Our system can also be easily extended to multiple robots by recursively predicting information gain from the planned path of other robots as an impact of the decision making of the local robot. Evaluation over different sizes of environments and different numbers of robots show at most 6.7% higher final accuracy and 49.8% less training time to reach a same high accuracy over the baselines.

1 Introduction

The recent flourishing of machine learning [5, 7, 12] is emphasizing the importance of both quantity and quality (or information gain) of training data for high performance (e.g., accuracy) when training an Artificial Intelligence (AI) model [19]. For online training tasks typically (e.g., domain adaptation [1, 18], implicit rendering [9, 15, 20] deployed over mobile robots which consecutively take unlabelled sample (e.g., images, lidar) from the environment as training input for an AI model, enabling automatic acquisition of high quality training data on the robots will boost the performance of their training AI model.

The existing related domain, active learning [2, 11, 13] estimates the information gain of possible training input for the training AI model according to metrics such as gradients or uncertainty. But unfortunately, the found training data of highest estimated information gain as the acquisition target fails to form a navigation path for the robot to collect high quality training data. Specifically, the robot is moving and sampling in a continuous world and along the path the robot tends to collect similar samples as the acquisition target, which in turn causes more low quality training data being input. In our evaluation, such methods cause the similarity between consecutively collected training data to be over 84.28% (measured in SSIM). Another problem of such methods is that such methods cannot be scaled to multiple robots, because after each planning step on one robot, the information gain estimation must be re-optimized to account for newly added training input [3, 8, 13], which cannot run in parallel.

The key reason of the above problem is that when searching for navigation path (the acquisition destination), they are viewing the training AI model in a static perspective. As online training of the AI model proceeds, the parameters of the AI model keeps evolving and thus, the possible training data of highest quality is also changed, especially when the robot starts navigating and new training data is collected. Failure to model such AI model evolution in path searching leads to low quality of training data collected along the navigation path.

To tackle the above problem, a dynamic perspective of the training AI model is necessary while searching for a path, which is difficult to predict since the training data sampled in the future are not available. Instead, we observe that if we relate the training loss with the zones of the environment of their corresponding training input, its dynamics (varying trend) have the potential to predict future training loss if the same zone is sampled in the future, whose diff reflects possible information gain and AI model evolution. The predicted future training loss can recursively serve as the basis of the prediction of the training loss when a further step is taken and finally forms the prediction of the accumulated information gain along the whole path, with which we can search for an optimal path with highest accumulated information gain.

Notably, this method can easily be extended to multi-agent co-operation: an agent can directly plan its own path based on the predicted training loss from the planned path shared by other agents, which enables co-operative multi-view sampling across different agents, so as to alleviate the limitation of the continuous action and sampling space of a single robot.

Based on these ideas, we propose LOss-Dynamics-Aware multi-agent cooperative active sampling system for robotic online training (LODA). In LODA, we distribute the online training workload across all agents involved and renders the training loss to a sparse 3D grid on each agent which acts as the representation of the training loss dynamics. With this representation, we propose LOss-Dynamics-Aware Recursive Path Planning (LODA-RPP) algorithm: given candidate paths based on path searching algorithms such as Random Rapid Tree algorithm, we use an online trained small multi-layer perception model (MLP) to recursively predict the future training loss after taking each step of a candidate path, which approximates the accumulated information gain along the path. If planned paths from other robots are received, their predicted future training loss will be temporarily merged into the grid representation as the basis of path planning. In this way, each agent finds a cooperative multi-view sampling path that optimizes accumulated information gain by considering the AI model state evolution modelled by loss dynamics, leading to higher quality of collected training data and higher performance of the training AI model.

We implemented LODA based on ROS2 Galactic and PyTorch on Ubuntu20.04 and evaluated its performance across one to two robots involved. We choose a state-of-the-art (SOTA) online training task

BNV-Fusion [9] in the domain of implicit rendering as the major workload and use Replica [14], a standard computer-vision dataset, together with Habitat simulator [16] and evaluated LODA across different sizes of the simulated scenes. We select two SOTA active learning baselines [2, 13] as the major baselines. Evaluation shows that:

- LODA is accurate. Compared to baselines, LODA improved the accuracy of the training AI model being training underlying online training task by 6.7% ~ 0.7% due to the improved quality of sampled training input from the environment.
- LODA is efficient. The time reaching the same accuracy of the training AI model was at most reduced by 49.8% compared with the baselines.

The major contributions of this paper is we propose a novel dynamic perspective of training AI model modelled with the loss dynamics of the environment for automatic high quality training data acquisition in robotic online training tasks to improve the performance of training AI model. Our proposed system LODA and the LODA-RPP algorithm together model the accumulated training information gain of candidate paths and find an optimal path for robot sampling that optimizes the overall quality of collected training data. It also enables efficient multi-robot multi-view cooperation that further increases the quality of training data by simply sharing the planned path among the robots. They will nurture the development of more real-world robotic online training applications and improving their performance by enabling efficient and high performance automatic high quality training data acquisition.

2 Background & RELATED WORK

2.1 Online Training

While machine learning methods heavily rely on labeled training datasets (supervised training), it is costly to label datasets in every possible environment. Especially, real-world robots are running in non-stationary and continually changing environments where the target domain distribution can even change over time. As a result, various unsupervised training algorithms are developed to learn knowledge from unlabeled data [1, 17, 18]. For example, adversarial unsupervised domain adaptation methods [4, 10, 17] aim to adapt a source pretrained model to a target domain without using any source data. By typically training the pretrained models with both online collected data from the new environment and labels predicted with generative methods, robots can adapt to new environments with higher accuracy of the models.

Another special category of online training is implicit rendering[6, 9, 20], which aims to train an implicit representation of the 3D scene with the supervision of typically a sequence of RGBD images and camera poses. They feature the ability of preserving finer textural information, scene completion, real-time reconstruction,etc. We envision that the prosperity of these unsupervised training algorithms is making online training on online collected data on robots feasible and practical.

2.2 Active Learning

Active learning [2, 11] works in protocol where labels can be requested by the algorithm in a sequential, feedback-driven fashion, which in the field of robotics would be instructing the robot to acquire the corresponding sample. Active learning algorithms aim to identify and label only maximally-informative samples, so that a high-performing AI model can be trained with minimal labeling effort. As such, a robust active learning algorithm for deep neural networks enables the deep learning algorithm to select data for learning in order to gain accurate prediction despite fewer training labels, especially when labeling or gathering new data is costly.

They mainly inspect certain metrics of the training AI model about the unlabeled training data to estimate their information gain. For example, [2] estimates information gain of an unlabeled training data by computing the gradient intensity and diversity when the training model is back-propagated using pseudo labeling. [8, 13] model the model output with a Gaussian probability distribution and train an extra model head to predict the divergence of the distribution under a training data as a estimation of uncertainty and thus information gain when the training data is collected for training. Nevertheless, they typically rely on a static estimation about the current training model, which fail to model the evolution of the training AI model and result in inefficient acquisition path for the robot along which low quality training data are collected.

3 Overview

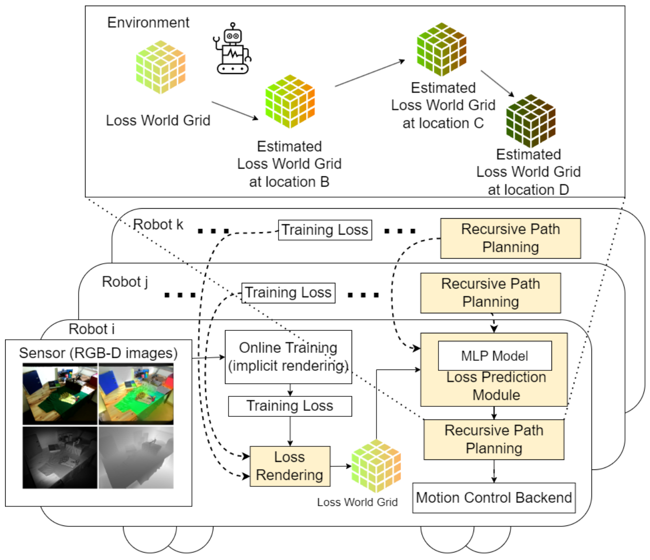


Figure 1. Overview of LODA.

As shown in Fig. 1, assume that each robot is a four-wheel robot with sensors (e.g., cameras, lidar) attached to it and it is periodically taking samples ($I_i \in \boldsymbol{I}$) from the sensors as the training input for an underlying online training task (e.g., 3D reconstruction), where $i = 1, 2, \ldots$ means the i^{th} sample and $\boldsymbol{I}$ is the sample space of the environment. We assume that the size of the environment is limited and navigable for the robot, such as a room or an office. The underlying online training task trains an AI model θ periodically on collected training input to extract knowledge (e.g., increase in accuracy of the AI model) from the environment and $L(\theta_t, I_i)$ is the training loss function on sample I_i at time t. The goal of automatic high quality training data acquisition for a such scenario is

$$\begin{aligned} \min \quad & t, m \\ s.t. \quad & L(\theta_t) \approx \frac{1}{m}\sum_{i=1}^{m} L(\theta_t, I_i) < t_{loss} \qquad (1) \\ & C(\{I_i\}) > t_{coverage} \end{aligned}$$

where $C(\cdot)$ is the coverage estimation of the samples. It means finding a minimum sequence of samples that fastest covers the environment and have the highest information gain to reduce the loss function which forms the navigation path of the robots.

Considering the evolution of θ over time, the variation of θ_t is typically impossible to predict. Instead, we predict the level of information gain when I_i is acquired exploiting loss dynamics through the following components of our system: loss rendering, loss dynamics modelling and future loss prediction, and finally forms an optimal path with recursive path planning as discussed below.

3.1 Loss Prediction Module

In LODA, we store the running statistics of training loss by rendering it to a sparse grid of tensor named World Loss Grid (WLG) in correspondence to the coordinates of the training samples in the environment. An MLP model is periodically trained based on these data to learn the dynamics of loss and predict future loss given a desired sampling position and orientation. The MLP model is online trained to adapt to possible different loss intensity and different loss dynamics in different environments or online training tasks.

3.2 Recursive Path Planning

As in Fig. 1, starting from the original WLG, we predict the training loss when a next step is taken using the loss prediction module (excluded for simplicity in the figure) and incrementally render a new sparse grid,. On the basis of the predicted new sparse grid, we continue to predict the training loss for the following steps recursively and finally at the end of the path the accumulated information gain estimation of the whole path is calculated as the difference between the final predicted sparse grid and the original to search for the path with optimal accumulated information gain.

To support multi-robot cooperation in active high quality training data acquisition, a robot should consider the plans from other robots and estimate the impact of the plans from other robots for the decision making of itself. In LODA this can be easily achieved with the above design. We recursively predict the resulting sparse grid of training loss of planned paths from other robots and merge it with the original WLG as the basis of recursive path planning of this robot, resulting in a cooperative path considering the impact from the possible actions from other robots.

4 Design

4.1 Loss Prediction Module

4.1.1 Loss rendering. We assume that the sampling environment is limited (e.g., within a room or office) which can be divided into a grid of finite 3D cells ($u_k \in \boldsymbol{u}$) and there exists a correspondence relationship that $I_i = Render(\boldsymbol{u}, p_i)$ where p_i is the sampling perspective (position and orientation) of I_i, and $\boldsymbol{u}_i$ is the set of related cells, typically regulated by depth values sensed at p_i and only a fraction of $\boldsymbol{u}$ is visible which we refer to as $\boldsymbol{u}_i$. Thus, $L(\theta_t)$ in Equa. 1 is rewritten as $L(\theta_t) \approx \sum_{i=1}^{m} L'(\theta_t, \boldsymbol{u}_i, p_i) < t_{loss}$.

While the fast coverage problem is well approximated by traditional RRT algorithms for a limited environment and the number of samples needed to train an AI model is typically much larger than the number of samples needed to cover the environment, thus we may consider the problem based on a set of $\{I_i\}$ that already satisfies the coverage constraint and finding new samples that reduce $\sum_{i=1}^{m} L'(\theta_n, \boldsymbol{u}, p_i)$ the most, and we define such diff as:

$$D(i+1) = \sum_{i=1}^{m+1} L'(\theta_t, \boldsymbol{u}_i, p_i) - \sum_{i=1}^{m+1} L'(\theta_{t+1}, \boldsymbol{u}_i, p_i) = \sum_{i=1}^{m} \Delta {L'}_t^i + \Delta {L'}_t^{m+1} \tag{2}$$

where $\Delta {L'}_t^i = L'(\theta_t, \boldsymbol{u}_i, p_i) - L'(\theta_{t+1}, \boldsymbol{u}_i, p_i)$. We assume knowledge from I_{m+1} does not conflict with the previous data, which means it will not increase loss of previous data and $\sum_{i=1}^{m} \Delta {L'}_t^i \geq 0$. We have

$$D(i+1) \geq \Delta {L'}_t^{m+1} \tag{3}$$

which means the beginning from θ_t to θ_{t+1}, the training loss reduction over $\boldsymbol{u}_{m+1}$ and p_{m+1} reflects the level of information gain can be learned from I_{m+1}. By now we have associated the variation of training loss bounded to a set of grid cell and its sampling pose with the possible information gain, whose prediction model can be online trained using existing data as shown in the following subsection. p_{m+1} (and thus $\boldsymbol{u}_{m+1}$) maximizing such loss variation is the training data with highest information gain.

We have approximated the possible information gain of an unknown sample with the difference between the loss rendered to the visible area of a 3D grid of the environment under the regulation of sampling position and orientation and depth values. Given a candidate p_{m+1}, the parameters to calculate $\Delta {L'}_t^{m+1}$, θ_{t+1} and the actual sensor input are still unknown, but their resulting loss is of low dimension and can be modelled via online training methods. In LODA, we use WLG G to represent the environment, which is a four-dimensional sparse grid with the last dimension storing information for loss prediction. Specifically, we divide the possible sampling distances and orientations to consider into different levels, n_{dist} and n_{dir}. When a loss value is rendered to a cell on the grid, we store the number of times that this loss is hit (weight), its average and the latest loss value together with the sampling position and orientation using position encoding.

4.1.2 Training and inference. Given the groundtruth loss l_u of a cell, its sampling distance and orientationi_{dist} and i_{dir} and G, a three-layer MLP model M is trained to learn and predict the training loss. The training loss function of M is:

$$L_M = \|M(i_{dist}, i_{dir}, G_{j,k,h}) - l_u\|_2 \tag{4}$$

which means modelling the dynamics (or varying trend) of loss of a cell based on history information. During inference, given a sampling pose p, we calculate the visible cells $\boldsymbol{u}_p$ and the sampling distance and orientation of each cell. For a cell u on the WLG sampled at i_{dist} and i_{dir}, its loss is predicted as $l_u = M(i_{dist}, i_{dir}, u)$. Note the inference input and inference result of $\boldsymbol{u}_p$ together are of the same form of information needed to render cells for G, forming the basis of the recursive path planning in LODA-RPP algorithm.

4.2 LODA-RPP Algorithm

In LODA, candidate paths (sequences of sampling poses.) are generated by random sampling algorithms such as RRT. While traditional active learning methods can only consider the information gain of one pose in a path, we manage to calculate the accumulated information gain along the path with LODA-RPP to get a better estimation of overall information gain, and finally find the optimal path with highest accumulated information gain.

Algorithm 1: Recursive State Prediction.

```
Input: WLG: $G$; Loss Prediction Module with MLP: $M$;
       candidate path: $P$
Output: predicted future state of World Los Grid when
        candidate path is executed: $G'$
if $len(P) = 0$ then
    return $G$
$p = P[0]$;
$\boldsymbol{u_p}, \boldsymbol{i_{dist}}, \boldsymbol{i_{dir}}$ = FrustumCalculation($p$);
$l_{\boldsymbol{u_p}} = M(\boldsymbol{i_{dist}}, \boldsymbol{i_{dir}}, \boldsymbol{u_p})$;
$G'_p$ = LossRendering($l_{\boldsymbol{u_p}}, p, G$);
$G_{next}$ = overwrite $G$ with $G'_p$;
$P_{next} = P[1:]$;
$G'$ = RecursiveStatePrediction($G_{next}, M, P_{next}$);
```

We first introduce the process of Recursive State Prediction as shown in Alg. 1. With the WLG G and MLP model M, given desired sampling pose p from a path P, We first compute the visible cells $\boldsymbol{u_p}$ on G and their sampling distances and orientations, with which we can predict their future training loss $l_{\boldsymbol{u_p}}$ from G. We then use the prediction result $l_{\boldsymbol{u_p}}$ and sampling pose p to render cells in $\boldsymbol{u_p}$ as a partial update of G, referred to as G'_p, which is the predicted future state of $\boldsymbol{u_p}$ in G when the robot navigates to p to acquire new training input. Merging G and G'_p, we will get the predicted WLG when we have arrived at p and taken samples. Recursing such process, we will get the future WLG when every step in P is taken.

Algorithm 2: Framework of LODA-RPP algorithm

```
Input: WLG: $G$; Loss Prediction Module with MLP: $M$; start
       pose: $p$; planned paths from other robots: $\boldsymbol{P_o}$
Output: Planned Path: $P$
for $P'$ in $\boldsymbol{P_o}$ do
    $G$ = RecursiveStatePrediction($G, M, P'$);
generate candidate paths $P_{candidate}$;
$e = 0$;
$P = [p]$;
for $P'$ in $P_{candidate}$ do
    $G'$ = RecursiveStatePrediction($G, M, P'$);
    $e' = G - G'$;
    if $e' > e$ then
        $P = P'$
```

Integrating Recursive State Prediction, LODA Recursive Path Planning (LODA-RPP) in Alg. 2 is able to predict the accumulated information gain of each candidate path. First, we update the WLG temporarily with the planned paths from other robots using Recursive State Prediction to consider their impact on the path planning of this robot to search for cooperative paths. Then for every candidate path, we estimate the accumulated information gain of each path by predicting their resulting WLG when poses in each path are executed with Recursive State Prediction and computing the difference between the starting WLG and resulting WLG. Finally the outputted path is the optimal path with highest estimated accumulated information gain which also cooperates with other robots.

5 Implementation

We implemented LODA based on Robot Operation System 2 (ROS2) Galactic, Python3.8 and PyTorch on Ubuntu20.04, using the default navigation stack of ROS2. We maintained the Loss World Grid on a sparse coordinated tensor using PyTorch, instead of a dense tensor, because in a room there are typically a lot of empty space that holds no useful information and will not have loss being computed. This design significantly reduces the memory and computation burden.

6 Evaluation

6.1 Evaluation Settings

6.1.1 Testbed. For reproducibility and stability, we evaluated LODA over a server with 4 2080Ti 11GB GPUs, an Intel(R) Xeon(R) Silver 4116 2.10GHz CPU and 128 GB RAM. When extending the number of robots involved, each robot is allocated a unique GPU to avoid performance bottleneck due to sharing GPU.

6.1.2 Workload. We choose a state-of-the-art implicit rendering method, BNV Fusion [9] as our workload, which reconstructs dense 3D mesh of the environment based on a sequence of depth images and their sampling poses. BNV Fusion [9] is composed of a sparse grid of latent codes which is an implicit representation of the 3D environment and a pre-trained decoder. We use 2cm voxel size in BNV Fusion, other default hyper-parameters and the default decoder parameter checkpoint officially released.

6.1.3 Dataset. We use the Habitat simulator [16] and Replica dataset [14] commonly used in computer vision tasks. We select three real-life high quality scenes from the Replica dataset of three levels of sizes: office_0 sized $22.04m^2$, hotel_0 sized $35.88m^2$ and frl_apartment_1 sized $93.10m^2$, referred to as the small scene, the medium scene and the large scene. The robots are spawned at random positions in the scenes.

The Habitat simulator broadcasts the simulated rgb samples of each robot at 10 Hz, and the online training task of BNV Fusion collects and processes the simulated output to optimize the training model at 1 Hz. Every 2 minutes a reconstructed dense 3D mesh of the scene is produced to estimate the model's performance.

6.1.4 Baselines. We choose two state of the art active learning methods as our major baselines, namely Badge [3] and ActiveNerf [13]. Badge [3] (referred to as badge) estimates the information gain of a possible input by computing the gradient magnitude and diversity with respect to parameters in the final (output) layer, which is computed using the most likely output according to the model. ActiveNerf [13] (referred to as uncertainty) is a method optimized for implicit rendering that models the training model output as a Gaussian distribution and add an extra model head that learns to predict the output variance as a measurement of uncertainty, where the zones with highest uncertainty is considered of highest information gain. To extend them to multiple robots, we duplicate these methods on each robot. The baselines all use the same navigation stack as our system, with RRT* algorithm providing candidate training input for them to select from.

6.1.5 Metrics. To estimate the quality of the reconstructed mesh, we choose the completion ratio that is commonly used in 3D reconstruction tasks [9, 20]. To calculate completion ratio, starting from vertices in the groundtruth mesh, we query for vertices in the reconstructed mesh whose distance is within a threshold and the ratio of successful query is the completion ratio, which reflects how close that the reconstructed mesh is to the groundtruth. We also used Structure Similarity Index Measure (SSIM) to estimate the similarity between consecutive samples in breakdown to show how LODA instructs the robot to collect training input with higher diversity and quality. SSIM close to 1 means that the two estimated images are identical, and SSIM smaller means less similar.

6.2 End-to-End Performance

6.2.1 Different Sizes of Scenes. We first compare the performance of three different methods (LODA referred to as loss_dynamics in the figures) with one robot exploring and acquiring training data according to their information gain for the 3D reconstruction process in scenes of different sizes. Note that the visible area of the whole scene to the four-wheel robot is limited because of the comparatively low perspective of a robot, the completion ratio in each scene is limited and lower than one hundred percent.

The experimental result in Fig 2 shows that our method outperforms the baselines across different scene sizes in terms of both time to reach the same completion ratio and the final completion ratio. Compared with badge and uncertainty, LODA saved the time by about 43.8% and 45.6% for reconstruction completion ratio reaching 54% in the office_0 case, by 21.5% and 44.6% for the completion ratio reaching 45% in the hotel_0 case and by 49.0% and 49.8% for the completion ratio reaching 8% in the frl_apartment_1 scene This can be attributed to LODA's ability to model the accumulated information gain of the whole navigation path and find the navigation path with highest accumulated information gain using the Loss Prediction Module and LODA-RRT algorithm.

Also, thanks to higher quality of training input sampled in LODA, the final completion ratio with LODA also increased across all the scenes. Specifically, compared with badge and uncertainty, LODA achieved 4.1% and 4.0% higher completion ratio in the office_0 case, 1.8% and 6.7% higher completion ratio in the hotel_0 case and 1.1% and 2.0% higher completion ratio in the frl_apartment_0.

6.2.2 Different Numbers of Robots. We also carried out experiments of two robots exploring and taking training data in the same scenes. And we found LODA still had the advantages of higher final reconstruction completion ratio: compared with badge and uncertainty, LODA achieved 0.9% and 1.0% higher completion ratio in the office_0 case, 1.7% and 1.9% higher completion ratio in the hotel_0 case and 0.7% and 2.5% higher completion ratio in the frl_apartment_0. But it can also be observed that the intensity of this advantages shrunk and the time to reach the same completion ratio was not evident. This can be due to that while the baseline methods with a single robot tend to sample similar training input of low quality, duplicating them into multiple robots does extend their searching spaces for training data of high information gain and results in more diverse samples, mitigating their limitation and limiting LODA's advantages. Nevertheless, the advantages in final completion ratio still reveal the contribution of the higher quality training data collected under the instruction of LODA.

6.3 Breakdown

In this section we inspect certain runtime statistics to better understand how LODA manages to acquire training data of higher quality. First, we sampled and evaluated a test set sized 2000 for the Loss Prediction Module at each time spot indicated on Fig. 4 in the one robot with office_0 case and calculated the variation percentage *percent* between the predicted loss l_p and the groundtruth l as $percent = \frac{|l_p - l|}{l}$. The average of the variation percentage is recorded to be 1.84% and the resulting variation range is depicted in Fig. 4, which proves the effectiveness of the Loss Prediction Module.

Table 1. Average SSIM between consecutively collected samples.

Methods	loss_dynamics	badge	uncertainty
SSIM	0.789	0.843	0.840

Second, we calculated the SSIM values between depth images consecutively sampled across all evaluated methods in the one robot with office_0 case depicted in Fig. 5, with average shown in Table. 1. We can learn that the baselines typically control the robot sampling depth images that are more similar than those collected under LODA. Although lower SSIM does not directly translate into higher information gain between consecutive samples, higher SSIM does reveal the problem of the baselines that they tend to sample similar depth images when executing the planned path due to their static perspective of the training AI model.

7 Conclusion

In this paper, we propose a novel dynamic perspective that models the evolution of AI model in robotic online training tasks and its influence on the information gain estimation of a navigation path with loss dynamics. And we propose LOss-Dynamics-Aware multi-agent cooperative active sampling system for robotic online training (LODA). Using the Loss Prediction Module and the Recursive Path Planning algorithm of LODA, we manage to model the accumulated information gain along each step of a navigation path and find the most informative path for the training AI model, which accelerates the increase of its performance. LODA will nurture the development of more real-world robotic online training applications and improving their performance by enabling efficient and high performance automatic high quality training data acquisition.

References

[1] Sk Miraj Ahmed, Dripta S. Raychaudhuri, Sujoy Paul, Samet Oymak, and Amit K. Roy-Chowdhury. 2021. Unsupervised Multi-Source Domain Adaptation Without Access to Source Data. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 10103–10112. https://openaccess.thecvf.com/content/CVPR2021/html/Ahmed_Unsupervised_Multi-Source_Domain_Adaptation_Without_Access_to_Source_Data_CVPR_2021_paper.html

[2] Jordan T. Ash, Chicheng Zhang, Akshay Krishnamurthy, John Langford, and Alekh Agarwal. 2020. Deep Batch Active Learning by Diverse, Uncertain Gradient Lower Bounds. arXiv:1906.03671 [cs.LG]

[3] Jordan T. Ash, Chicheng Zhang, Akshay Krishnamurthy, John Langford, and Alekh Agarwal. 2020. Deep Batch Active Learning by Diverse, Uncertain Gradient Lower Bounds. https://doi.org/10.48550/arXiv.1906.03671 arXiv:1906.03671 [cs, stat].

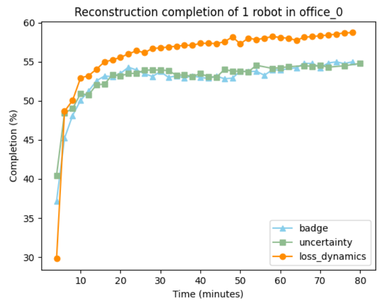
[4] Muhammad Awais, Fengwei Zhou, Hang Xu, Lanqing Hong, Ping Luo, Sung-Ho Bae, and Zhenguo Li. 2021. Adversarial Robustness for Unsupervised Domain Adaptation. In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*. 8548–8557. https://doi.org/10.1109/ICCV48922.2021.00845

[5] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Proceedings of the 34th International Conference on Neural Information Processing Systems (NIPS'20)*. Curran Associates Inc., Red Hook, NY, USA, 1877–1901.

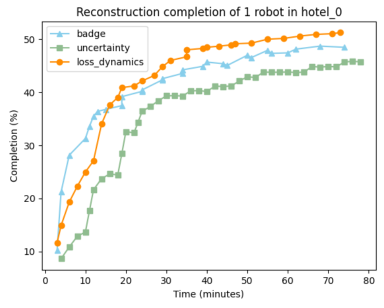
[6] Gadiel Sznaier Camps, Robert Dyro, Marco Pavone, and Mac Schwager. 2022. Learning Deep SDF Maps Online for Robot Navigation and Exploration. arXiv:2207.10782 [cs.RO]

[7] K. R. Chowdhary. 2020. Natural Language Processing. In *Fundamentals of Artificial Intelligence*, K.R. Chowdhary (Ed.). Springer India, New Delhi, 603–649. https://doi.org/10.1007/978-81-322-3972-7_19

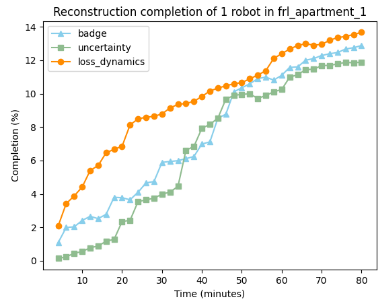
[8] Liren Jin, Xieyuanli Chen, Julius Rückin, and Marija Popović. 2023. NeU-NBV: Next Best View Planning Using Uncertainty Estimation in Image-Based Neural Rendering. https://doi.org/10.48550/arXiv.2303.01284 arXiv:2303.01284 [cs].



(a) office_0 (Small Scene)

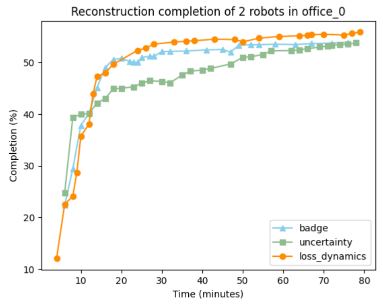


(b) hotel_0 (Medium Scene)

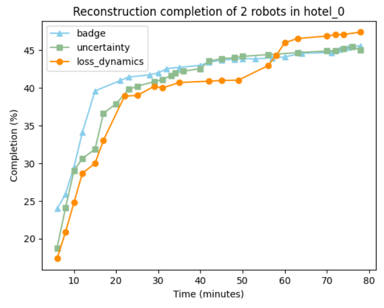


(c) frl_apartment_1 (Large Scene)

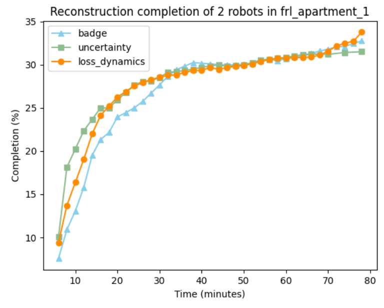
Figure 2. The completion ratio of the reconstructed mesh with one single robot in different sizes of scenes.



(a) office_0 (Small Scene)



(b) hotel_0 (Medium Scene)



(c) frl_apartment_1 (Large Scene)

Figure 3. The completion ratio of the reconstructed mesh with two robots in different sizes of scenes.

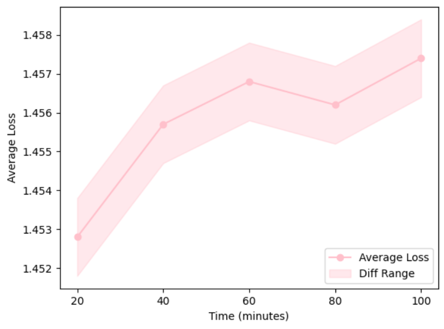


Figure 4. Varying range of predicted loss and the actual loss.

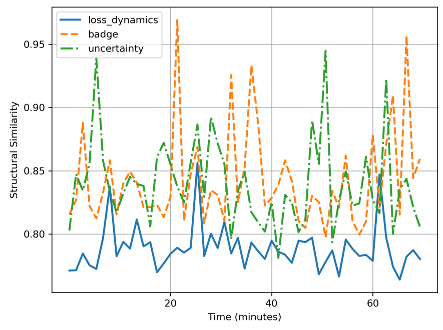


Figure 5. SSIM between consecutive collected training data.

[9] Kejie Li, Yansong Tang, Victor Adrian Prisacariu, and Philip H. S. Torr. 2022. BNV-Fusion: Dense 3D Reconstruction using Bi-level Neural Volume Fusion. arXiv:2204.01139 [cs.CV]

[10] Xiaofeng Liu, Zhenhua Guo, Site Li, Fangxu Xing, Jane You, C.-C. Jay Kuo, Georges El Fakhri, and Jonghye Woo. 2021. Adversarial Unsupervised Domain Adaptation with Conditional and Label Shift: Infer, Align and Iterate. In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*. 10347–10356. https://doi.org/10.1109/ICCV48922.2021.01020

[11] Vu-Linh Nguyen, Mohammad Hossein Shaker, and Eyke Hüllermeier. 2022. How to measure uncertainty in uncertainty sampling for active learning. *Machine Learning* 111, 1 (2022), 89–122.

[12] OpenAI. 2023. GPT-4 Technical Report. https://doi.org/10.48550/arXiv.2303.08774 arXiv:2303.08774 [cs].

[13] Xuran Pan, Zihang Lai, Shiji Song, and Gao Huang. 2022. ActiveNeRF: Learning Where to See with Uncertainty Estimation. In *Computer Vision – ECCV 2022*, Shai Avidan, Gabriel Brostow, Moustapha Cissé, Giovanni Maria Farinella, and Tal Hassner (Eds.). Vol. 13693. Springer Nature Switzerland, Cham, 230–246. https://doi.org/10.1007/978-3-031-19827-4_14 Series Title: Lecture Notes in Computer Science.

[14] Julian Straub, Thomas Whelan, Lingni Ma, Yufan Chen, Erik Wijmans, Simon Green, Jakob J. Engel, Raul Mur-Artal, Carl Ren, Shobhit Verma, Anton Clarkson, Mingfei Yan, Brian Budge, Yajie Yan, Xiaqing Pan, June Yon, Yuyang Zou, Kimberly Leon, Nigel Carter, Jesus Briales, Tyler Gillingham, Elias Mueggler, Luis Pesqueira, Manolis Savva, Dhruv Batra, Hauke M. Strasdat, Renzo De Nardi, Michael Goesele, Steven Lovegrove, and Richard Newcombe. 2019. *The Replica Dataset: A Digital Replica of Indoor Spaces.* Technical Report arXiv:1906.05797. arXiv. https://doi.org/10.48550/arXiv.1906.05797 arXiv:1906.05797 [cs, eess] type: article.

[15] Edgar Sucar, Shikun Liu, Joseph Ortiz, and Andrew J. Davison. 2021. iMAP: Implicit Mapping and Positioning in Real-Time. In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*. IEEE, Montreal, QC, Canada, 6209–6218. https://doi.org/10.1109/ICCV48922.2021.00617

[16] 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. 2021. Habitat 2.0: Training Home Assistants to Rearrange their Habitat. In *Advances in Neural Information Processing Systems (NeurIPS)*.

[17] Qin Wang, Olga Fink, Luc Van Gool, and Dengxin Dai. 2022. Continual Test-Time Domain Adaptation. arXiv:2203.13591 [cs.CV]

[18] Garrett Wilson and Diane J. Cook. 2020. A Survey of Unsupervised Deep Domain Adaptation. *ACM Transactions on Intelligent Systems and Technology* 11, 5 (Sept. 2020), 1–46. https://doi.org/10.1145/3400066

[19] Daochen Zha, Zaid Pervaiz Bhat, Kwei-Herng Lai, Fan Yang, Zhimeng Jiang, Shaochen Zhong, and Xia Hu. 2023. Data-centric Artificial Intelligence: A Survey. https://doi.org/10.48550/arXiv.2303.10158 arXiv:2303.10158 [cs].

[20] Zihan Zhu, Songyou Peng, Viktor Larsson, Weiwei Xu, Hujun Bao, Zhaopeng Cui, Martin R. Oswald, and Marc Pollefeys. 2022. NICE-SLAM: Neural Implicit Scalable Encoding for SLAM. https://doi.org/10.48550/arXiv.2112.12130 Number: arXiv:2112.12130 arXiv:2112.12130 [cs].