C-NAV: Towards Self-Evolving Continual Object Navigation in Open World

Ming-Ming Yu[1], **Fei Zhu**[2†], **Wenzhuo Liu**[3,4], **Yirong Yang**[1],
Qunbo Wang[6*], **Wenjun Wu**[1,5], **Jing Liu**[3,4]
[1]Beihang University, [2]Centre for Artificial Intelligence and Robotics, HKISI-CAS
[3]Institute of Automation, Chinese Academy of Sciences
[4]University of Chinese Academy of Sciences
[5]Hangzhou International Innovation Institute, Beihang University, [6]Beijing Jiaotong University
mingmingyu@buaa.edu.cn wangqb6@outlook.com

Abstract

Embodied agents are expected to perform object navigation in dynamic, open-world environments. However, existing approaches typically rely on static trajectories and a fixed set of object categories during training, overlooking the real-world requirement for continual adaptation to evolving scenarios. To facilitate related studies, we introduce the continual object navigation benchmark, which requires agents to acquire navigation skills for new object categories while avoiding catastrophic forgetting of previously learned knowledge. To tackle this challenge, we propose C-Nav, a continual visual navigation framework that integrates two key innovations: (1) A dual-path anti-forgetting mechanism, which comprises feature distillation that aligns multi-modal inputs into a consistent representation space to ensure representation consistency, and feature replay that retains temporal features within the action decoder to ensure policy consistency. (2) An adaptive sampling strategy that selects diverse and informative experiences, thereby reducing redundancy and minimizing memory overhead. Extensive experiments across multiple model architectures demonstrate that C-Nav consistently outperforms existing approaches, achieving superior performance even compared to baselines with full trajectory retention, while significantly lowering memory requirements. The code will be available at https://bigtree765.github.io/C-Nav-project.

1 Introduction

Successful navigation to a target object [1, 2, 3, 4, 5] is a fundamental capability for embodied agents and serves as a prerequisite for any meaningful interaction, making it a central topic in recent research. Current state-of-the-art methods [6, 7, 8, 9, 10] typically depend on pre-trained models [11, 12, 13] and large-scale demonstration trajectories [7, 10], operating under the assumption of complete access to the training dataset and a fixed set of object categories. However, these approaches generally lack continual learning ability [14] and are vulnerable to catastrophic forgetting, which restricts their applicability in dynamic, open-world applications [15]. In contrast, practical deployment demands that agents adapt continuously to new object categories and evolving user instructions. Addressing this need requires agents to incrementally refine their navigation capabilities by learning from new data, without revisiting all previously seen data or retraining the model from scratch.

To enable the acquisition of new skills without forgetting previously learned knowledge, numerous studies have achieved promising results in uni-modal settings like image classification [16, 17].

†Project lead. *Corresponding author.

39th Conference on Neural Information Processing Systems (NeurIPS 2025).

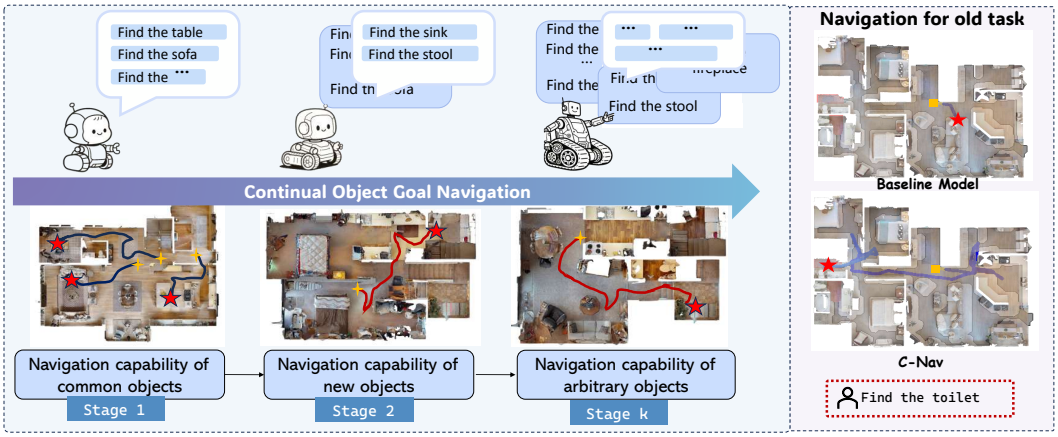

Figure 1: Continual Object Navigation: The robot must continually learn from new data while retaining its ability to navigate to previously seen object goals. After training across multiple phases, the baseline model suffers from *representation drift* and *policy degradation*, leading to a loss of navigation capability on previously learned tasks, whereas C-Nav enables cumulative knowledge retention across phases.

However, their application to embodied decision-making tasks, particularly object-goal navigation, remains largely unexplored. This might be because the task requires long-horizon decision-making and the integration of diverse multi-modal inputs, including RGB images, depth images, pose information, and textual instructions. Moreover, a single training trajectory can span hundreds of steps, which significantly increases the complexity of the learning process. To the best of our knowledge, continual learning in the context of object navigation has not been previously studied. To advance this important yet under-explored area, it is necessary to propose a comprehensive evaluation framework to assess the continual learning capabilities of object navigation models.

In related embodied navigation systems [6, 7, 8, 2, 18, 9, 10], the multi-modal encoder is often used for processing egocentric visual inputs, pose information, and other sensory signals to build a semantic representation of the environment. Simultaneously, the action decoder integrates temporal observations to predict navigation actions. As the agent is exposed to new tasks over time, distributional shifts in sensory inputs and action patterns introduce biases into the model components, leading to *representation drift* and *policy degradation*. This results in catastrophic forgetting of previously acquired knowledge, as illustrated in Figure 1. The agent often becomes capable of semantic exploration only for objects in the current task, losing its ability to navigate to goals learned earlier. Although naive data replay [19] has demonstrated potential in mitigating catastrophic forgetting, they face several notable limitations when applied to navigation tasks characterized by long sequences of temporally and spatially correlated observations: (1) storage overhead scales quickly with the number of tasks, making memory management increasingly costly; (2) retaining continuous scene observations raises privacy concerns, as these may inadvertently reveal sensitive spatial information; and (3) the reliance on fine-grained, atomic-level action representations results in substantial redundancy across adjacent frames, leading to inefficient memory usage and the replay of repetitive experiences.

To advance embodied AI research in realistic settings, we introduce a continual object navigation benchmark built upon the HM3D [20] and MP3D [21] datasets using the Habitat simulator [22]. This benchmark enables a comprehensive evaluation of mainstream navigation model architectures (including Bev-based [9], RNN-based [23], Transformer-based [18], and LLM-based models [8]) and several widely adopted continual learning techniques such as LoRA [24], model merge [25], LwF [26], and naive data replay [19]. Further, we propose C-Nav, a continual object navigation framework that incorporates a dual-path forgetting mitigation strategy. Specifically, a feature distillation path is employed to ensure representational consistency across multi-task, multi-modal encoders, while a feature replay path is designed to preserve decision stability within the policy decoder. This architectural design addresses the representational drift that occurs in both the encoder and policy modules as the agent encounters new tasks, while avoiding the storage overhead and privacy risks associated with raw trajectory retention. To mitigate the high visual redundancy inherent in navigation trajectories, C-Nav introduces an adaptive experience selection module. This module treats keyframe

selection as an outlier detection problem in the learned representation space and leverages the Local Outlier Factor (LOF) [27] algorithm to automatically identify critical navigation moments, such as "goal discovery" or "spatial transitions." In summary, our contributions include:

- To evaluate the continual learning ability of agents in dynamic task settings, we establish a continual object goal navigation benchmark, enabling a systematic assessment of state-of-the-art navigation architectures and continual learning strategies.
- We propose C-Nav, a continual Object Navigation framework that mitigates catastrophic forgetting via a dual-path strategy, enforcing representation consistency through feature distillation and policy consistency through feature replay, without relying on extensive raw trajectories.
- We develop an adaptive experience selection mechanism that identifies and retains informative features of navigation frames by leveraging representation-space outlier detection, significantly reducing memory redundancy while maintaining policy performance.
- Extensive experiments verify the effectiveness of C-Nav across architectures and continual learning methods, demonstrating superior performance in both accuracy and efficiency.

2 Related Work

2.1 Object Goal Navigation

The task of locating a specified object in an unknown environment is fundamental to robotic systems. Current approaches to object-goal navigation can be categorized into two types. (1) The first category focuses on Zero-Shot Object Navigation [28, 29, 4, 3, 30]. These methods explicitly construct maps and utilize VLMs [31, 32, 33, 34] or LLMs [35, 36, 37] for reasoning to select the most valuable frontier or waypoint for exploration. For instance, in Cow [29], the robot explores the nearest frontier point until the target is detected using CLIP features [11] and open-vocabulary object detectors [13]. ESC [5], L3MVN [38], and Voronav [3] leverage LLMs for reasoning and decision-making to enhance performance. VLFM [4] employs a VLM to assign semantic values to the map based on first-person observations and textual prompts, selecting the highest-scoring frontier. These approaches exhibit strong zero-shot generalization and avoid catastrophic forgetting, as they do not update model parameters. However, they rely on complex reasoning pipelines and manually designed exploration rules, making them cumbersome and computationally costly. Moreover, their inability to fine-tune on newly collected navigation data limits task-specific adaptation, often resulting in suboptimal performance in complex or novel environments. (2) The second category involves using pre-trained visual encoders to transform first-person observations into visual vectors, which are then processed by a navigation policy trained via large-scale imitation or reinforcement learning [6, 7, 8, 2, 18, 9, 10, 39, 40], thereby equipping agents with semantic navigation capabilities.

These approaches, often trained via imitation or supervised learning, provide robust performance for object-goal navigation tasks by incorporating large-scale datasets. However, these methods typically assume fixed environments and task categories, lacking the ability to learn continually. As a result, they struggle to adapt to dynamic settings where new objects and tasks emerge over time and are prone to catastrophic forgetting, which limits their generalization to open-world scenarios. Recent advances such as FSTTA [41] and NavMorph [42] enhance test-time adaptability through fast–slow gradient updates or self-evolving world models, but primarily focus on fine-grained, step-by-step navigation. In contrast, our work addresses coarse-grained, long-horizon navigation, where the agent interprets high-level object-goal instructions to perform long-range exploration. Moreover, unlike prior methods that train on static datasets, our framework progressively introduces new object categories across multiple training stages, enabling continual skill acquisition. To this end, we propose C-Nav, a continual navigation framework that equips agents with the ability to incrementally learn new capabilities and adapt to novel settings without forgetting previously acquired knowledge.

2.2 Continual Learning

Continual learning [14, 43, 44], also known as incremental learning or lifelong learning, has attracted significant attention in recent years. Existing methods mainly focus on image classification. Regularization methods [26, 43] aim to prevent catastrophic forgetting by constraining the changes in model parameters during learning. Data replay methods [45, 46], which store and reuse old data, can retain

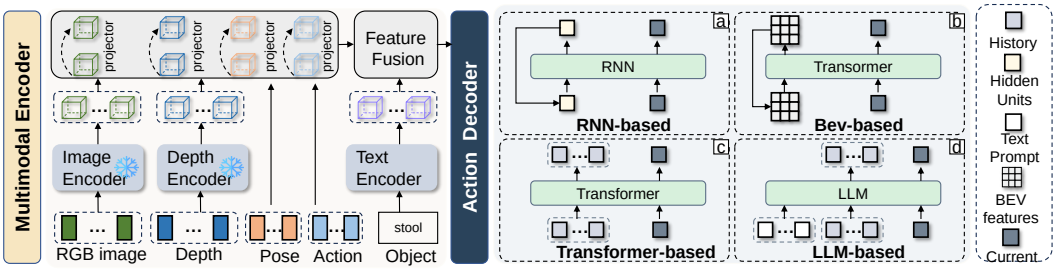

Figure 2: Different navigation model architectures. According to the action decoder, they can be categorized into a) RNN-Based, b) Bev-Based, c) Transformer-Based, and d) LLM-Based.

prior knowledge but introduce additional storage and privacy concerns [47]. Architecture-based methods [48] add new submodules for each task, leading to an increase in model parameters as the learning progresses. Recently, some studies [49, 50, 51, 52] have explored continual learning of multimodal large language models with parameter-efficient fine-tuning techniques (e.g., LoRA [24, 53], prompt tuning [54]) to facilitate continual fine-tuning without significant computational overhead. However, these methods do not extend well to embodied intelligence, especially in long-range, multimodal ObjectNav tasks. In contrast, embodied navigation poses greater challenges, requiring agents to process multimodal inputs and adapt to evolving environments and goals. Existing continual learning techniques fall short in addressing such complexity, particularly in long-horizon, multimodal tasks demanding both spatial and semantic reasoning. This gap motivates our C-Nav framework, which enables continual learning for visual navigation, with a focus on long-horizon tasks, multimodal integration, and dynamic adaptation in real-world settings.

3 Problem Definition and Benchmark

Problem Setting. We propose *Continual-ObjectNav*, where an embodied agent must incrementally master navigation skills through K sequential tasks. Each task $k \in \{1, \ldots, K\}$ introduces a distinct set of goal object categories $\mathcal{C}_k$, with strict disjointness between tasks ($\mathcal{C}_i \cap \mathcal{C}_j = \emptyset$ for all $i \neq j$). The agent receives training data $\mathcal{D}_k = \{(s_i^k, c_i^k, \tau_i^k)\}_{i=1}^{N_k}$ at each stage, where s_i^k denotes a 3D indoor environment, $c_i^k \in \mathcal{C}_k$ specifies the target object category, and $\tau_i^k = \{(o_{i,t}^k, a_{i,t}^k)\}_{t=1}^{L_i}$ provides expert demonstration trajectories. The observation $o_{i,t}^k$ integrates egocentric RGB-D sensing, agent pose (position/orientation), action history, and object name. The action $a_{i,t}^k$ is selected from a discrete action space $\mathcal{A} = \{\texttt{move\_forward}, \texttt{turn\_left}, \texttt{turn\_right}, \texttt{look\_up}, \texttt{look\_down}, \texttt{stop}\}$.

Evaluation. We evaluate the agent at the end of each training stage k, using a test set of unseen environments $\mathcal{S}_k$ and object goals sampled from the cumulative category set $\mathcal{C}_{1:k} = \bigcup_{j=1}^{k} \mathcal{C}_j$. This setup requires the agent to retain and apply previously learned navigation skills while adapting to new categories. Each episode starts from a random position, and the agent must navigate to a target object category $c \in \mathcal{C}_{1:k}$. A success is recorded if the agent issues a `stop` action within a specified distance to a valid instance of the target object within 500 time steps. We report the most widely used metrics: Success Rate (SR) and Success weighted by Path Length (SPL), averaged across all categories in $\mathcal{C}_{1:k}$. SPL measures the efficiency of the agent's trajectory by comparing it to the shortest possible path from the starting position to the nearest valid instance of the target object category.

Model Architecture Notation. As illustrated in Figure 2, the agent policy, shared across different architectures, can be parameterized as a composition of two learnable modules: a multimodal encoder $f : \mathcal{O} \to \mathbb{R}^d$, which maps observations to compact feature representations $h_t = f(o_t)$, and an action decoder π that predicts navigation actions conditioned on both the current encoded feature and the trajectory history. At each time step t, the decoder receives $h_{1:t} = \{f(o_1), \ldots, f(o_t)\}$ and produces a distribution over actions $\pi(a_t | h_{1:t})$.

Dataset. We adopt two widely used object goal navigation datasets: ObjectNav (HM3D) consists of 2,000 episodes sampled from 20 validation scenes in the HM3D dataset, covering 6 object categories. ObjectNav (MP3D), introduced in the Habitat 2020 Challenge, contains 2,195 episodes from 11 MP3D validation scenes, spanning 21 object categories. For model training, we utilize human demonstration trajectories collected via Habitat-Web [10] and PIRL [7]. The HM3D training set contains 75,488 trajectories, while the MP3D training set includes 59,604 trajectories after filtering

Table 1: Continual-ObjectNav splits for MP3D and HM3D.

Type	MP3D					HM3D				
	Stage1	Stage2	Stage3	Stage4	Total	Stage1	Stage2	Stage3	Stage4	Total
Category	12	3	3	3	21	3	1	1	1	6
Trajectory (training data)	40,608	7,306	7,484	4,206	59,604	36,667	13,640	13,182	11,999	75,488
Episode (evaluation data)	1,783	1,934	2,039	2,195	2,195	945	1,343	1,624	2,000	2,000

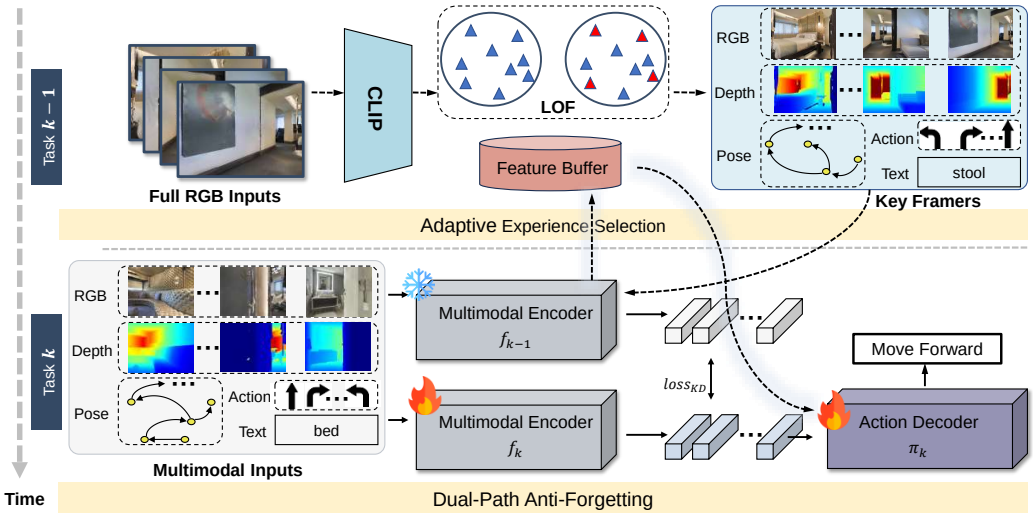

Figure 3: Overview of the proposed **C-Nav** framework for continual object navigation. It consists of two key components: (1) adaptive experience selection, which identifies semantically meaningful keyframes via LOF in the representation space, reducing storage and redundancy; and (2) dual-path anti-forgetting, which mitigates representation and policy drift through a feature distillation path (bottom left) and a feature replay path (bottom right), ensuring stability across encoder and decoder modules. For each task, C-Nav retains p sparse trajectory features in the deep feature space.

out samples involving 6 synthetic objects. To adapt to the continual Object Navigation setting, we divide the object categories and corresponding trajectories into four incremental learning stages. The detailed splits are shown in Table 1, and a full list of object category assignments per stage can be found in the supplementary materials.

4 The Proposed Method: C-Nav

To address the continual object navigation task more effectively, we propose a new method described in this section. The overall framework of our approach is illustrated in Figure. 3.

4.1 Dual-Path Anti-Forgetting

In continual learning of object navigation, catastrophic forgetting can be attributed to representational drift in both the multimodal encoder and the action decoder. As the model sequentially learns new tasks, the distribution of learned features may shift, resulting in degraded performance on earlier tasks. To address this, we propose a dual-path anti-forgetting strategy that constrains both the feature extraction and policy prediction stages.

Representation Consistency with Feature Distillation. The multimodal encoder integrates visual and goal-related information. To prevent drift in the learned feature representation, we apply feature-level distillation. Specifically, we enforce the consistency between the old and current encoder outputs by minimizing the ℓ_2 distance between their extracted features across time:

$$\mathcal{L}_{\text{KD}} = \sum_{t=1}^{L} \| f_{k-1}(o_t) - f_k(o_t) \|_2^2 , \tag{1}$$

where f_{k-1} denotes the frozen encoder from a previous task, and f_k is the current encoder. Both take as input the observation o_t, ensuring multimodal alignment is preserved during continual learning.

Policy Consistency with Feature Replay. To mitigate the policy forgetting during continual learning, we replay stored trajectory features of previous tasks along with their corresponding action labels to maintain policy consistency. The feature replay loss is defined as:

$$\mathcal{L}_{\text{FR}} = \frac{1}{|L|} \sum_{t=1}^{L} -w_t \log \pi_k(a_t|h_{1:t}), \quad w_t = 1 + \gamma \cdot \mathbb{1}_{a_t \neq a_{t-1}}, \tag{2}$$

where $h_t \in \mathbb{R}^d$ represents the t-th frame feature stored in the feature buffer, and π_k denotes the current action decoder. The weighting factor γ [10] emphasizes action transitions by assigning higher importance to time steps where the expert action differs from the previous one. This objective function ensures that the policy maintains consistent interpretations of previously learned features while adapting to new tasks. The gradient updates through this loss term preserve the mapping between historical observations and their originally learned actions, thereby mitigating catastrophic forgetting in the continual learning scenario.

4.2 Adaptive Experience Selection

In continual object navigation, replaying previously collected trajectories is critical for mitigating catastrophic forgetting. However, storing all frames in a trajectory is memory-intensive and often wasteful, as many frames are visually redundant and contain little novel information. From a data distribution perspective, frames that exhibit significant semantic shifts tend to be rare and thus can be considered outliers. These frames often correspond to moments such as entering new spaces, approaching target objects, or reaching decision points. Therefore, we propose to transform the task of selecting salient experience points into the detection of outliers within the feature space of visual observations. Specifically, given a trajectory $\tau = \{o_1, o_2, \ldots, o_L\}$, we first encode visual observations using a pretrained CLIP model to obtain feature representations: $\mathbf{v}_i = \text{CLIP}(\text{RGB}(o_i))$ for $i = \{1, \ldots, L\}$. Let $\text{dist}(\mathbf{v}_i, \mathbf{v}_j)$ denote the Cosine distance in feature space:

$$d(\mathbf{v}_i, \mathbf{v}_j) = 1 - \frac{\mathbf{v}_i^\top \mathbf{v}_j}{\|\mathbf{v}_i\|_2 \cdot \|\mathbf{v}_j\|_2}. \tag{3}$$

Base that, we define the k-distance neighborhood: $N_k(\mathbf{v}_i) = \{\mathbf{v}_j \mid d(\mathbf{v}_i, \mathbf{v}_j) \leq d_k(\mathbf{v}_i)\}$, where $d_k(\mathbf{v}_i)$ is the distance to the k-th nearest neighbor. For each neighbor $\mathbf{v}_j \in N_k(\mathbf{v}_i)$, the reachability distance is defined as $\text{reach-dist}_k(\mathbf{v}_i, \mathbf{v}_j) = \max\{d_k(\mathbf{v}_j), d(\mathbf{v}_i, \mathbf{v}_j)\}$. Using this, the local reachability density (LRD) of $\mathbf{v}_i$ is the inverse of its mean reachability distance:

$$\text{LRD}_k(\mathbf{v}_i) = \left(\frac{1}{|N_k(\mathbf{v}_i)|} \sum_{\mathbf{v}_j \in N_k(\mathbf{v}_i)} \text{reach-dist}_k(\mathbf{v}_i, \mathbf{v}_j) \right)^{-1}. \tag{4}$$

Then, the LOF is calculated as the relative density of $\mathbf{v}_i$ compared to its neighbors:

$$\text{LOF}_k(\mathbf{v}_i) = \frac{1}{|N_k(\mathbf{v}_i)|} \sum_{\mathbf{v}_j \in N_k(\mathbf{v}_i)} \frac{\text{LRD}_k(\mathbf{v}_j)}{\text{LRD}_k(\mathbf{v}_i)}. \tag{5}$$

According to this formulation, the set of selected keyframe indices is defined as $\mathcal{I} = \{i \mid \text{LOF}_k(\mathbf{v}_i) > 1\}$, where a higher $\text{LOF}_k(\mathbf{v}_i)$ indicates that the frame is considered as an outlier, typically corresponding to a navigation state with significant semantic change. These keyframes are subsequently encoded through a multimodal encoder f_k and stored in a feature buffer to mitigate catastrophic forgetting in the action decoder during training.

4.3 Overall Learning Objective of C-Nav

Our continual navigation policy is optimized using a composite loss function that integrates supervision from the current task, feature-space knowledge distillation, and feature replay consistency:

$$\mathcal{L} = \mathcal{L}_{\text{Curr}} + \lambda_{\text{KD}} \cdot \mathcal{L}_{\text{KD}} + \lambda_{\text{FR}} \cdot \mathcal{L}_{\text{FR}}, \tag{6}$$

where λ_{KD} and λ_{FR} are weighting coefficients that balance the influence of feature-level knowledge distillation and feature replay loss, respectively. The current task k using behavior cloning with inflection weighting [10]: $\mathcal{L}_{\text{curr}} = \frac{1}{L} \sum_{t=1}^{L} -w_t \log \pi_k \left(a_t \middle| f_k(o_1), \ldots, f_k(o_t)\right)$.

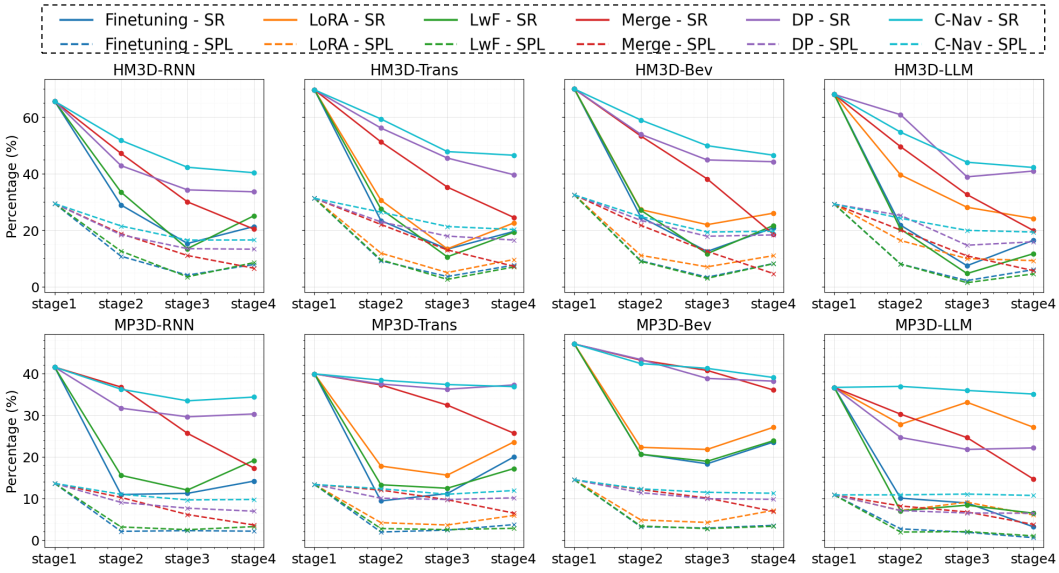

Figure 4: Results of SR and SPL on the MP3D and HM3D datasets. Solid lines represent SR, while dashed lines represent SPL.

Table 2: Performance comparison on the HM3D benchmark. *DP* indicates data replay, and *Merge* denotes model merging. *Avg* reports the average performance across all stages, while *Last* reflects the performance at the final stage.

Methods	HM3D-RNN				HM3D-Trans				HM3D-Bev				HM3D-LLM			
	Avg		Last		Avg		Last		Avg		Last		Avg		Last	
	SR	SPL	SR	SPL	SR	SPL	SR	SPL	SR	SPL	SR	SPL	SR	SPL	SR	SPL
Finetuning	32.8	13.0	21.3	7.8	31.4	12.9	19.5	7.6	32.0	13.3	20.8	8.2	28.4	11.3	16.4	6.0
LoRA	-	-	-	-	34.0	14.4	22.5	9.6	36.3	15.4	24.1	9.2	39.9	16.2	24.1	9.2
LwF	34.4	13.5	25.1	8.5	31.7	12.6	19.2	6.9	32.6	13.1	21.7	8.1	26.2	10.8	11.7	4.5
Merge	40.8	16.4	20.4	6.5	45.1	18.4	24.5	7.2	45.0	17.8	18.5	4.6	42.5	16.4	19.9	5.6
DP	44.1	18.6	33.6	13.2	52.7	22.1	39.6	16.5	53.2	23.0	44.2	18.3	52.2	21.2	40.9	15.9
C-Nav	**50.0**	**21.0**	**40.3**	**16.5**	**55.8**	**24.8**	**46.5**	**20.2**	**56.3**	**24.0**	**46.5**	**19.6**	**52.2**	**23.1**	**42.2**	**19.3**

5 Experiments

5.1 Experimental Setup

Implementation Details. We implement our multi-modal encoder using CLIP-ResNet50 [11] for visual encoding and a PointNav-pretrained ResNet-50 [55] for depth encoding. Following previous work [23, 9], we keep both encoders frozen during training. The feature fusion is implemented by the feature concatenation. We train our model using AdamW optimization with a linear warmup over 1,000 steps to reach our initial learning rate of 3×10^{-4}, followed by linear decay. We train each task stage for 25 epochs with a batch size of 32. For action prediction, the RNN-based model employs a 2-layer LSTM [56] architecture. The transformer-based and Bev-based [9] decoders utilize a 4-layer transformer [57] as the action decoder with a dropout rate of 0.1. For the LLM-based approach, we adopt Qwen2-0.5B [37] as the action decoder, incorporating six special tokens to represent atomic actions. We set the inflection weight γ to 3.48 and configure our loss balance weights λ_{KD} and λ_{FP} to 5. All experiments are conducted using two NVIDIA A6000 GPUs. Additional implementation details are provided in the supplementary materials.

Comparison Methods. We compare our method against several representative continual learning strategies across four backbone architectures. Specifically, we evaluate the following baselines. Naive Fine-tuning serves as the simplest approach, where models are sequentially fine-tuned on each stage without any mechanism to mitigate forgetting. LoRA [24] introduces low-rank adaptation modules during training; after each stage, the LoRA branches are merged into the main model for inference. LwF [26] applies knowledge distillation by computing the KL divergence between the logits of the

Table 3: Performance comparison of different methods on MP3D benchmark.

Methods	MP3D-RNN				MP3D-Trans				MP3D-Bev				MP3D-LLM			
	Avg		Last		Avg		Last		Avg		Last		Avg		Last	
	SR	SPL	SR	SPL	SR	SPL	SR	SPL	SR	SPL	SR	SPL	SR	SPL	SR	SPL
Finetuning	19.4	5	14.1	2.1	20.1	5.3	19.9	3.7	27.4	6.0	23.5	3.5	14.7	4.0	3.1	0.5
LoRA	-	-	-	-	24.2	6.7	23.5	5.9	29.5	7.6	27.1	7.1	31.1	8.2	27.1	6.0
LwF	22	5.6	19.1	3.2	20.7	5.3	17.1	2.8	27.6	5.9	23.8	3.3	14.6	3.9	6.4	1.0
Merge	30.3	8.4	17.3	3.6	33.8	10.3	25.7	6.5	41.7	10.9	36.0	6.9	26.5	7.4	14.6	3.7
DP	33.8	9.3	30.3	6.9	37.7	10.8	37.2	10.1	41.8	11.4	38.1	9.8	26.3	7.7	22.1	6.4
C-Nav	**36.4**	**11.0**	**34.3**	**9.7**	**38.1**	**12.1**	**36.8**	**11.9**	**42.4**	**12.3**	**39.0**	**11.2**	**36.1**	**10.8**	**35.0**	**10.7**

current model and those of the previous one, encouraging the model to preserve past knowledge. Model merge preserves prior knowledge by linearly interpolating the weights of the current and previous models using a 0.7/0.3 ratio. Lastly, data replay stores $p = 80$ original trajectories per object category, which are reused in subsequent training stages to mitigate forgetting. Additional implementation details are provided in the supplementary materials.

5.2 Experimental Results and Analysis

Main Results. We compare several baseline methods and model architectures for Continual-ObjectNav task. As shown in Figure 4, naive fine-tuning leads to severe catastrophic forgetting across all four architectures, with an average decline of approximately 40% in navigation success rate on previously seen categories by the final stage. Additionally, the results show that the Bev-based architecture consistently outperforms others. Meanwhile, LLM-based models do not exhibit a clear performance advantage, likely due to limited training trajectories. The LwF method demonstrates limited effectiveness across all models. In contrast, LoRA and model merge help mitigate forgetting by either reducing the number of trainable parameters or integrating knowledge from previous models. Notably, LoRA shows a significant advantage when applied to LLM-based models compared to other model architectures. Data replay achieves effective performance retention, but it incurs substantial storage overhead and raises privacy concerns. As shown in Table 2 and Table 3, the proposed C-Nav generalizes well across various architectures and datasets. Specifically, compared to data replay, C-Nav achieves an average SR improvement of 3.35% on MP3D and 2.75% on HM3D across four model architectures, while requiring lower storage costs. A detailed breakdown of performance on old versus new tasks at each training stage is provided in Appendix C.1, illustrating how C-Nav balances retention of prior knowledge with adaptation to new tasks. These results demonstrate the effectiveness of our dual-path forgetting mitigation mechanism, which ensures consistent representation and policy learning as the agent encounters new tasks.

Table 4: Ablation studies of dual-path anti-forgetting. KD denotes feature distillation, and FP denotes feature replay.

Methods	HM3D-RNN				HM3D-Trans				HM3D-Bev				HM3D-LLM			
	Avg		Last		Avg		Last		Avg		Last		Avg		Last	
	SR	SPL	SR	SPL	SR	SPL	SR	SPL	SR	SPL	SR	SPL	SR	SPL	SR	SPL
w/o KD	28.2	11.7	16.9	6.6	31.4	13.2	20.2	8.1	32.5	13.9	21.9	8.9	33.6	13.4	19.9	7.3
w/o FP	37.9	15.6	27.9	10.7	45.9	21.9	32.6	14.5	38.9	16.5	26.7	10.3	42.7	17.2	30.2	11.6
All	**50.0**	**21.0**	**40.3**	**16.5**	**55.8**	**24.8**	**46.5**	**20.2**	**56.3**	**24.0**	**46.5**	**19.6**	**52.2**	**23.1**	**42.2**	**19.3**
Methods	MP3D-RNN				MP3D-Trans				MP3D-Bev				MP3D-LLM			
	Avg		Last		Avg		Last		Avg		Last		Avg		Last	
	SR	SPL	SR	SPL	SR	SPL	SR	SPL	SR	SPL	SR	SPL	SR	SPL	SR	SPL
w/o KD	16.9	4.7	10.4	2.0	20.6	5.8	21.2	4.6	26.1	6.6	23.5	5.2	25.7	6.4	24.0	4.8
w/o FP	25.9	8.1	22.0	7.3	27.2	8.9	22.7	8.5	31.3	9.6	29.2	9.95	28.9	9.0	24.7	8.0
All	**36.4**	**11.0**	**34.3**	**9.7**	**38.1**	**12.1**	**36.8**	**11.9**	**42.4**	**12.3**	**39.0**	**11.2**	**36.1**	**10.8**	**35.0**	**10.7**

Ablation Studies of Dual-Path Anti-forgetting. To assess the contribution of each component in our proposed dual-path anti-forgetting mechanism, we conduct ablation studies by separately removing

the feature-space alignment loss and the feature replay loss in the action decoder. As presented in Table 4 and Appendix C.4, the removal of either component leads to significant performance degradation. Specifically, eliminating the feature-space KD loss results in an average SR drop of 22% and 16% across four model architectures on the HM3D and MP3D datasets, respectively. Similarly, removing the feature replay loss from the action decoder causes a reduction of 12% and 10% on the same datasets. These results highlight the complementary roles of the two components: the feature-space constraint stabilizes representation learning, while feature replay ensures consistent decision-making under distributional shifts. Additionally, we further examine representation consistency in continual learning. As detailed in Appendix C.2, we compare fully freezing the multimodal encoder against selectively fine-tuning only the modality-specific projectors with consistency regularization to softly align old and new feature spaces. The results show that fully freezing the encoder degrades performance, whereas selective fine-tuning of projectors under consistency constraints enables the model to learn representations that remain effective for both previous and new tasks.

Ablation Studies of Adaptive Experience Selection. To validate the effectiveness of the proposed adaptive experience selection in C-Nav, we compare it with a uniform sampling baseline that assumes redundancy among adjacent frames. For a fair comparison, both methods replay trajectories truncated to 50% of their original length. As shown in Table 5 and Appendix C.5, our method consistently outperforms uniform sampling across model architectures and datasets, achieving SR improvements of 3.65% on HM3D and 3.2% on MP3D. Notably, compared to full-length feature replay in C-Nav, SR drops only marginally by 1.9% and 1.3%, despite using half the data. In addition, compared to full data replay, our adaptive method achieves an average SR improvement of 0.8% on HM3D and 2.0% on MP3D across four model architectures. These results demonstrate the efficiency and effectiveness of our adaptive strategy in preserving critical experiences for the Continual-ObjectNav task.

Table 5: Ablation studies of adaptive experience selection. "Uniform", "Adaptive", and "Full" refer to uniform sampling, adaptive sampling, and no sampling, respectively.

Methods	HM3D-RNN				HM3D-Trans				HM3D-Bev				HM3D-LLM			
	Avg		Last		Avg		Last		Avg		Last		Avg		Last	
	SR	SPL	SR	SPL	SR	SPL	SR	SPL	SR	SPL	SR	SPL	SR	SPL	SR	SPL
Uniform	43.2	19.7	32.2	15.0	50.5	22.0	40.5	17.6	49.4	21.7	37.6	16.5	47.7	22.4	37.0	18.4
DP (Full)	44.1	18.6	33.6	13.2	52.7	22.1	39.6	16.5	53.2	23.0	44.2	18.3	**52.2**	21.2	40.9	15.9
Adaptive	47.3	20.7	36.1	16.0	53.7	23.2	42.5	18.4	52.8	22.4	42.1	18.0	51.6	**23.8**	40.1	**19.4**
Full	**50.0**	**21.0**	**40.3**	**16.5**	**54.7**	**24.3**	**46.5**	20.2	**56.3**	**24.0**	**46.5**	**19.6**	**52.2**	23.1	**42.2**	19.3

Methods	MP3D-RNN				MP3D-Trans				MP3D-Bev				MP3D-LLM			
	Avg		Last		Avg		Last		Avg		Last		Avg		Last	
	SR	SPL	SR	SPL	SR	SPL	SR	SPL	SR	SPL	SR	SPL	SR	SPL	SR	SPL
Uniform	30.5	10.3	24.5	8.2	35.2	11.4	31.9	9.7	35.0	11.0	29.2	9.7	34.0	11.1	31.4	**10.9**
DP (Full)	33.8	9.3	30.3	6.9	37.7	10.8	37.2	10.1	41.8	11.4	38.1	9.8	26.3	7.7	22.1	6.4
Adaptive	34.1	**11.4**	**38.4**	**10.1**	**38.4**	**12.6**	36.6	11.7	40.9	**12.3**	38.8	**11.3**	34.2	**11.2**	32.4	10.7
Full	**36.4**	11.0	34.3	9.7	38.1	12.1	**36.8**	**11.9**	**42.4**	**12.3**	**39.0**	11.2	**36.1**	10.8	**35.0**	10.7

Moreover, Figure 5 presents an ablation study of different experience selection strategies on MP3D-Trans across all training stages. First, we compare our method against uniform sampling and clustering-based sampling, both using half-length trajectories. Our approach consistently outperforms both baselines because it explicitly preserves semantically salient frames, such as those captured at decision points or near target objects. These frames often reside in outlier regions of the feature space and are typically discarded by conventional samplers. Second, we reduce the key-frame selection ratio. Remarkably, our method

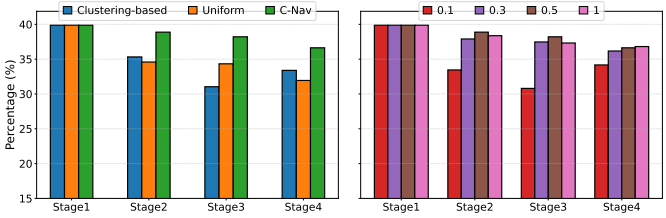

Figure 5: Ablation on experience selection and retention ratio on MP3D with a Transformer-based navigation architecture.

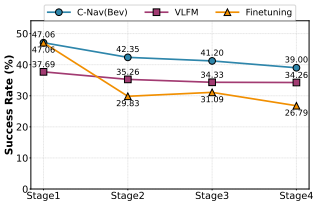

Figure 6: Comparison with the Zero-Shot Method.

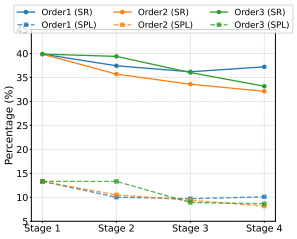

Figure 7: Sensitivity to stage order on MP3D.

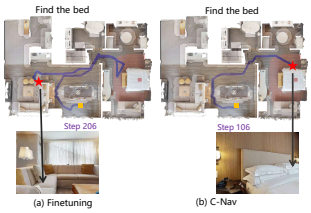

Figure 8: Qualitative navigation results.

maintains strong performance even at very low retention rates, achieving comparable or better results while using only 20% of the memory required by the clustering-based or uniform sampling baselines. This demonstrates the high efficiency and robustness of our approach.

Comparison with the Zero-Shot Method. We reproduce VLFM as a zero-shot baseline and evaluate it against C-Nav over four continual learning stages in Figure 6. VLFM selects waypoints using a pretrained vision-language model on an explicit map and executes navigation with a fixed reinforcement learning policy. Because its parameters are frozen, it avoids catastrophic forgetting and maintains stable but limited performance. This limitation stems from its inability to learn from new trajectories, leaving it constrained by the static priors of the large model. In contrast, C-Nav is an end-to-end trainable system that continuously improves from experience, adapts to new tasks, and mitigates forgetting.

Ablation Study on Stage Order Sensitivity. To evaluate the robustness of C-Nav to variations in task sequence during continual learning, we conduct ablation studies under three different stage orderings. As shown in Figure 7, the minimal performance variation across orderings further validates the effectiveness of our Dual-Path Anti-Forgetting mechanism. This insensitivity to task sequence is crucial for real-world deployment, where object categories may be encountered in arbitrary order.

Case Study. Figure 8 shows navigation trajectories of C-Nav and the Finetuning baseline in an unseen HM3D scene, where the target object (bed) was introduced in an earlier training stage. Both models were trained sequentially over four stages, with the sofa added as the target in the final stage. The baseline suffers from catastrophic forgetting and navigates toward a sofa despite the goal being a bed. In contrast, C-Nav successfully reaches the bed, demonstrating its ability to retain knowledge of previously learned categories. This highlights C-Nav's superior stability-plasticity balance in continual learning.

6 Conclusion

In this work, we establish a benchmark for evaluating continual learning in the context of object navigation. We assess various model architectures and mainstream continual learning methods. Building on this benchmark, we propose C-Nav for continual object navigation, a dual-path framework that mitigates catastrophic forgetting by jointly enforcing representation consistency through feature distillation and preserving policy consistency via feature replay. Additionally, we introduce an adaptive experience selection method to further reduce memory usage, ensuring efficient knowledge retention. Extensive experiments demonstrate that our approach outperforms existing methods, achieving superior performance across different architectures and tasks, while reducing memory overhead and mitigating forgetting more effectively than previous solutions.

Limitations. Despite the strong performance of C-Nav on the Continual-ObjectNav task, several limitations remain. First, real-world factors such as dynamic lighting and sensor noise may affect generalization, highlighting the need for validation on physical robots to assess robustness in real environments. Second, although C-Nav significantly reduces memory usage, its storage demands still grow linearly with task complexity. Future work may explore generative replay or trajectory-free approaches to further reduce memory overhead. Addressing these limitations is crucial for deploying continual navigation systems in practical applications.

Acknowledgments and Disclosure of Funding

This research is supported by the Artificial Intelligence National Science and Technology Major Project (2023ZD0121200), the National Natural Science Foundation of China (62437001, 62436001, 62206279), the Key Research and Development Program of Jiangsu Province (BE2023016-3), the Beijing Natural Science Foundation (L252146), and the InnoHK program.

References

[1] Devendra Singh Chaplot, Dhiraj Prakashchand Gandhi, Abhinav Gupta, and Russ R Salakhutdinov. Object goal navigation using goal-oriented semantic exploration. *Advances in Neural Information Processing Systems*, 33:4247–4258, 2020.

[2] Dhruv Batra, Aaron Gokaslan, Aniruddha Kembhavi, Oleksandr Maksymets, Roozbeh Mottaghi, Manolis Savva, Alexander Toshev, and Erik Wijmans. Objectnav revisited: On evaluation of embodied agents navigating to objects. *arXiv preprint arXiv:2006.13171*, 2020.

[3] Pengying Wu, Yao Mu, Bingxian Wu, Yi Hou, Ji Ma, Shanghang Zhang, and Chang Liu. Voronav: Voronoi-based zero-shot object navigation with large language model. *International Conference on Machine Learning*, 2024.

[4] Naoki Yokoyama, Sehoon Ha, Dhruv Batra, Jiuguang Wang, and Bernadette Bucher. Vlfm: Vision-language frontier maps for zero-shot semantic navigation. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 42–48, 2024.

[5] Kaiwen Zhou, Kaizhi Zheng, Connor Pryor, Yilin Shen, Hongxia Jin, Lise Getoor, and Xin Eric Wang. Esc: Exploration with soft commonsense constraints for zero-shot object navigation. In *International Conference on Machine Learning*, pages 42829–42842, 2023.

[6] Karmesh Yadav, Arjun Majumdar, Ram Ramrakhya, Naoki Yokoyama, Alexei Baevski, Zsolt Kira, Oleksandr Maksymets, and Dhruv Batra. Ovrl-v2: A simple state-of-art baseline for imagenav and objectnav. *arXiv preprint arXiv:2303.07798*, 2023.

[7] Ram Ramrakhya, Dhruv Batra, Erik Wijmans, and Abhishek Das. Pirlnav: Pretraining with imitation and rl finetuning for objectnav. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 17896–17906, 2023.

[8] Jiazhao Zhang, Kunyu Wang, Rongtao Xu, Gengze Zhou, Yicong Hong, Xiaomeng Fang, Qi Wu, Zhizheng Zhang, and He Wang. Navid: Video-based vlm plans the next step for vision-and-language navigation. *arXiv preprint arXiv:2402.15852*, 2024.

[9] Shizhe Chen, Thomas Chabal, Ivan Laptev, and Cordelia Schmid. Object goal navigation with recursive implicit maps. In *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 7089–7096. IEEE, 2023.

[10] Ram Ramrakhya, Eric Undersander, Dhruv Batra, and Abhishek Das. Habitat-web: Learning embodied object-search strategies from human demonstrations at scale. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 5173–5183, 2022.

[11] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763, 2021.

[12] Xiaohua Zhai, Basil Mustafa, Alexander Kolesnikov, and Lucas Beyer. Sigmoid loss for language image pre-training. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 11975–11986, 2023.

[13] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 9650–9660, 2021.

[14] Haiyang Guo, Fanhu Zeng, Fei Zhu, Jiayi Wang, Xukai Wang, Jingang Zhou, Hongbo Zhao, Wenzhuo Liu, Shijie Ma, Xu-Yao Zhang, et al. A comprehensive survey on continual learning in generative models. *arXiv preprint arXiv:2506.13045*, 2025.

[15] Fei Zhu, Shijie Ma, Zhen Cheng, Xu-Yao Zhang, Zhaoxiang Zhang, and Cheng-Lin Liu. Open-world machine learning: A review and new outlooks. *arXiv e-prints*, pages arXiv–2403, 2024.

[16] Marc Masana, Xialei Liu, Bartłomiej Twardowski, Mikel Menta, Andrew D. Bagdanov, and Joost Van De Weijer. Class-incremental learning: Survey and performance evaluation on image classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(5):5513–5533, 2022.

[17] Xinyuan Gao, Yuhang He, Songlin Dong, Jie Cheng, Xing Wei, and Yihong Gong. Dkt: Diverse knowledge transfer transformer for class incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 24236–24245, 2023.

[18] Kiana Ehsani, Tanmay Gupta, Rose Hendrix, Jordi Salvador, Luca Weihs, Kuo-Hao Zeng, Kunal Pratap Singh, Yejin Kim, Winson Han, Alvaro Herrasti, et al. Spoc: Imitating shortest paths in simulation enables effective navigation and manipulation in the real world. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16238–16250, 2024.

[19] Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H Lampert. icarl: Incremental classifier and representation learning. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 2001–2010, 2017.

[20] Karmesh Yadav, Ram Ramrakhya, Santhosh Kumar Ramakrishnan, Theo Gervet, John Turner, Aaron Gokaslan, Noah Maestre, Angel Xuan Chang, Dhruv Batra, Manolis Savva, and et al. Habitat-matterport 3d semantics dataset. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4927–4936, 2023.

[21] Angel Chang, Angela Dai, Thomas Funkhouser, Maciej Halber, Matthias Niessner, Manolis Savva, Shuran Song, Andy Zeng, and Yinda Zhang. Matterport3d: Learning from rgb-d data in indoor environments. *arXiv preprint arXiv:1709.06158*, 2017.

[22] Manolis Savva, Abhishek Kadian, Oleksandr Maksymets, Yili Zhao, Erik Wijmans, Bhavana Jain, Julian Straub, Jia Liu, Vladlen Koltun, Jitendra Malik, et al. Habitat: A platform for embodied ai research. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 9339–9347, 2019.

[23] Apoorv Khandelwal, Luca Weihs, Roozbeh Mottaghi, and Aniruddha Kembhavi. Simple but effective: Clip embeddings for embodied ai. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14829–14838, 2022.

[24] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, et al. Lora: Low-rank adaptation of large language models. *ICLR*, 1(2):3, 2022.

[25] Daniel Marczak, Bartłomiej Twardowski, Tomasz Trzciński, and Sebastian Cygert. Magmax: Leveraging model merging for seamless continual learning. In *European Conference on Computer Vision*, pages 379–395, 2024.

[26] Zhizhong Li and Derek Hoiem. Learning without forgetting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(12):2935–2947, 2017.

[27] Markus M Breunig, Hans-Peter Kriegel, Raymond T Ng, and Jörg Sander. Lof: identifying density-based local outliers. In *Proceedings of the 2000 ACM SIGMOD international conference on Management of data*, pages 93–104, 2000.

[28] Yuxuan Kuang, Hai Lin, and Meng Jiang. Openfmnav: Towards open-set zero-shot object navigation via vision-language foundation models. *arXiv preprint arXiv:2402.10670*, 2024.

[29] Samir Yitzhak Gadre, Mitchell Wortsman, Gabriel Ilharco, Ludwig Schmidt, and Shuran Song. Cows on pasture: Baselines and benchmarks for language-driven zero-shot object navigation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 23171–23181, 2023.

[30] Hang Yin, Xiuwei Xu, Linqing Zhao, Ziwei Wang, Jie Zhou, and Jiwen Lu. Unigoal: Towards universal zero-shot goal-oriented navigation. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 19057–19066, 2025.

[31] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. *Advances in neural information processing systems*, 36:34892–34916, 2023.

[32] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

[33] Zhijian Liu, Ligeng Zhu, Baifeng Shi, Zhuoyang Zhang, Yuming Lou, Shang Yang, Haocheng Xi, Shiyi Cao, Yuxian Gu, Dacheng Li, and et al. Nvila: Efficient frontier visual language models. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 4122–4134, 2025.

[34] Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, and et al. Qwen2. 5-vl technical report. *arXiv preprint arXiv:2502.13923*, 2025.

[35] Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

[36] Gemini Team, Rohan Anil, Sebastian Borgeaud, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M Dai, Anja Hauth, Katie Millican, et al. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*, 2023.

[37] Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, et al. Qwen technical report. *arXiv preprint arXiv:2309.16609*, 2023.

[38] Bangguo Yu, Hamidreza Kasaei, and Ming Cao. L3mvn: Leveraging large language models for visual target navigation. In *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3554–3560, 2023.

[39] Jiangyong Huang, Silong Yong, Xiaojian Ma, Xiongkun Linghu, Puhao Li, Yan Wang, Qing Li, Song-Chun Zhu, Baoxiong Jia, and Siyuan Huang. An embodied generalist agent in 3d world. *arXiv preprint arXiv:2311.12871*, 2023.

[40] Pu Feng, Junkang Liang, Size Wang, Xin Yu, Xin Ji, Yiting Chen, Kui Zhang, Rongye Shi, and Wenjun Wu. Hierarchical consensus-based multi-agent reinforcement learning for multi-robot cooperation tasks. In *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 642–649, 2024.

[41] Junyu Gao, Xuan Yao, and Changsheng Xu. Fast-slow test-time adaptation for online vision-and-language navigation. *arXiv preprint arXiv:2311.13209*, 2023.

[42] Xuan Yao, Junyu Gao, and Changsheng Xu. Navmorph: A self-evolving world model for vision-and-language navigation in continuous environments. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2025.

[43] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences*, 114(13):3521–3526, 2017.

[44] Wenzhuo Liu, Xin-Jian Wu, Fei Zhu, Ming-Ming Yu, Chuang Wang, and Cheng-Lin Liu. Class incremental learning with self-supervised pre-training and prototype learning. *Pattern Recognition*, 157:110943, 2025.

[45] Francisco M Castro, Manuel J Marín-Jiménez, Nicolás Guil, Cordelia Schmid, and Karteek Alahari. End-to-end incremental learning. In *Proceedings of the European conference on computer vision (ECCV)*, pages 233–248, 2018.

[46] Saihui Hou, Xinyu Pan, Chen Change Loy, Zilei Wang, and Dahua Lin. Learning a unified classifier incrementally via rebalancing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 831–839, 2019.

[47] Fei Zhu, Xu-Yao Zhang, Chuang Wang, Fei Yin, and Cheng-Lin Liu. Prototype augmentation and self-supervision for incremental learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 5871–5880, 2021.

[48] Shipeng Yan, Jiangwei Xie, and Xuming He. Der: Dynamically expandable representation for class incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3014–3023, 2021.

[49] Wenzhuo Liu, Fei Zhu, Haiyang Guo, Longhui Wei, and Cheng-Lin Liu. Llava-c: Continual improved visual instruction tuning. *arXiv preprint arXiv:2506.08666*, 2025.

[50] Hongbo Zhao, Fei Zhu, Haiyang Guo, Meng Wang, Rundong Wang, Gaofeng Meng, and Zhaoxiang Zhang. Mllm-cl: Continual learning for multimodal large language models. *arXiv preprint arXiv:2506.05453*, 2025.

[51] Haiyang Guo, Fanhu Zeng, Ziwei Xiang, Fei Zhu, Da-Han Wang, Xu-Yao Zhang, and Cheng-Lin Liu. Hide-llava: Hierarchical decoupling for continual instruction tuning of multimodal large language model. *arXiv preprint arXiv:2503.12941*, 2025.

[52] Fanhu Zeng, Fei Zhu, Haiyang Guo, Xu-Yao Zhang, and Cheng-Lin Liu. Modalprompt: Towards efficient multimodal continual instruction tuning with dual-modality guided prompt. *arXiv preprint arXiv:2410.05849*, 2024.

[53] Haiyang Guo, Fei Zhu, Wenzhuo Liu, Xu-Yao Zhang, and Cheng-Lin Liu. Pilora: Prototype guided incremental lora for federated class-incremental learning. In *European Conference on Computer Vision*, pages 141–159, 2024.

[54] Brian Lester, Rami Al-Rfou, and Noah Constant. The power of scale for parameter-efficient prompt tuning. *arXiv preprint arXiv:2104.08691*, April 2021.

[55] Erik Wijmans, Abhishek Kadian, Ari Morcos, Stefan Lee, Irfan Essa, Devi Parikh, Manolis Savva, and Dhruv Batra. Dd-ppo: Learning near-perfect pointgoal navigators from 2.5 billion frames. *arXiv preprint arXiv:1911.00357*, 2019.

[56] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[57] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: Please refer the abstract and introduction section.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Please refer the Conclusions section.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [NA]

Justification: Our paper does not include theoretical results.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: Please refer to the Experimental Setting section.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: We will provide complete code and documentation

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: Please refer to the Experimental Setting section. 876 Guidelines:

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [NA]

   Justification: We confirm that the large-scale dataset and fixed random seeds employed in our experiments are sufficient to guarantee the reproducibility of results. In our experiments, the variation in accuracy across repeated tests is minimal.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).

- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: answerYes

   Justification: Please refer to the Experimental Setting section.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: We comply with the NeurIPS Code of Ethics.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: Please refer to the Conclusions section.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA] .

    Justification: The paper poses no such risks.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: The navigation model and the data all open-source by the author.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.

- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: We provide the details of our dataset in the Datasets section.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [Yes]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

Answer: [Yes]

Justification: In evaluating continuous object navigation performance, we tested a model that employs an LLM as its decoder, and we have described both the methodology and purpose in our paper.

Guidelines:

- The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.

A Dataset Details

A.1 Category Splits for Continual ObjectNav

HM3D Dataset. The object categories are split into four stages as follows:

- **Stage 1:** *bed*, *chair*, *plant*
- **Stage 2:** *toilet*
- **Stage 3:** *tv_monitor*
- **Stage 4:** *sofa*

MP3D Dataset. The object categories are divided into four continual learning stages:

- **Stage 1:** *bed*, *cabinet*, *chair*, *chest_of_drawers*, *cushion*, *picture*, *plant*, *sink*, *sofa*, *stool*, *table*, *toilet*
- **Stage 2:** *shower*, *towel*, *tv_monitor*
- **Stage 3:** *bathtub*, *counter*, *fireplace*
- **Stage 4:** *clothes*, *gym_equipment*, *seating*

A.2 Demonstration Trajectory Length Distribution for Continual ObjectNav

We visualize the trajectory length distributions across different training stages in Continual ObjectNav, as shown in Figure 9 and Figure 10. For the HM3D dataset, the average trajectory lengths for Stage 1 to Stage 4 are 127.51, 142.53, 168.53, and 115.77, respectively. In contrast, the MP3D dataset exhibits longer trajectories, with average lengths of 182.79, 247.91, 227.10, and 315.75 across the four stages. The longer demonstration sequences in MP3D highlight the increased complexity of decision-making, making the Continual ObjectNav task more challenging.

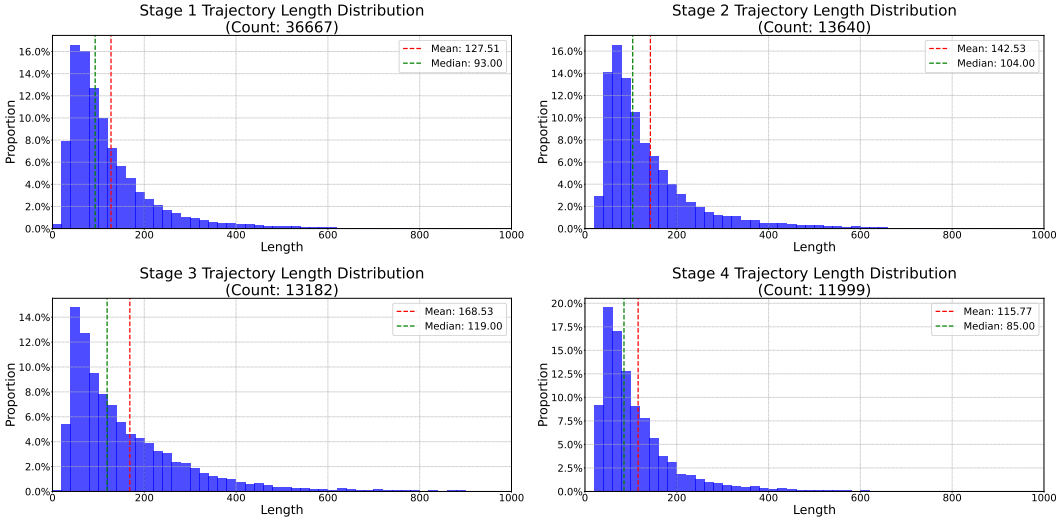

Figure 9: Demonstration trajectory length distribution per stage in HM3D dataset.

B Implementation Details

B.1 Details about the Model Architecture

For the BEV-based, transformer-based, and RNN-based models, the RGB, depth, pose, previous action, and object name inputs are transformed by linear projection layers into feature vectors with dimensionalities of 256, 128, 64, 32, and 32, respectively. In the Qwen 0.5B LLM-based architecture,

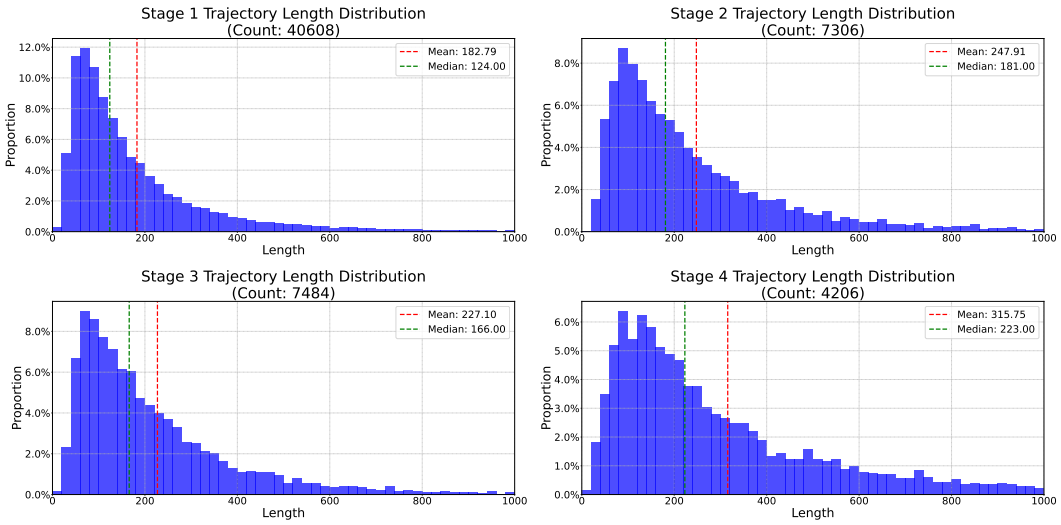

Figure 10: Demonstration trajectory length distribution per stage in MP3D Dataset.

the RGB, depth, pose, and previous action inputs are projected into feature vectors of size 400, 400, 64, and 32, respectively. These vectors are concatenated to produce a unified representation with a total dimensionality of 896. The task prompt used for the LLM-based model is formatted as follows:

```
<|im_start|>system You are a smart navigation assistant.<|im_end|>
<|im_start|> Your goal is to find the {object_name}.  What is the next
action?  <|im_end|>
```

The learning rate for the LLM-based model is set to 0.0001. `bfloat16` precision is adopted for efficient inference and reduced memory usage in the LLM-based navigation models. The weight decay is set to 0.1. Additionally, several special tokens are defined. Specifically, the action tokens include `<STOP>`, `<MOVE_FORWARD>`, `<TURN_LEFT>`, `<TURN_RIGHT>`, `<LOOK_UP>`, and `<LOOK_DOWN>`. The observation token, `<|observation|>`, is used to represent the environment's state in the model's input. All other hyperparameters are kept consistent with those used in the BEV-based, transformer-based, and RNN-based models. For the representation consistency loss in our Dual-Path Anti-Forgetting mechanism, we use the ℓ_2 distance between old and current features on HM3D, and the squared ℓ_2 distance on MP3D.

B.2 Details about the Compared Continual Learning Methods

- **LoRA**: Uses a low-rank decomposition with dimension $R = 16$, a scaling factor of $2 \times R$, and a dropout rate of 0.1.
- **LwF**: Applies a KL divergence loss between the current and previous model logits with a coefficient of 0.2.
- **Model Merge**: Linearly interpolates the weights of the current and previous models using a 0.7/0.3 ratio.
- **Data Replay**: Stores $p = 80$ original trajectories per object category for experience replay.
- **Proposed C-Nav**: Stores $p = 80$ trajectory features and samples 50% of each trajectory for adaptive experience selection, ensuring a fair comparison with data replay and uniform sampling methods.

B.3 The Pseudocode for C-Nav

The pseudocode for C-Nav is illustrated in Algorithm 1. During the initial stage, the model is trained on trajectories from the base object navigation task. Adaptive experience selection is then applied to identify important experiences, which are stored in the feature buffer $\mathcal{B}$. In subsequent learning stages, the model weights from the previous phase are loaded, and the model is optimized by combining

the current task's behavior cloning loss with feature space distillation and feature replay losses. This approach ensures the consistency of the multimodal feature encoder and action decoder across different tasks.

Algorithm 1 C-Nav for Continual ObjectNav with Adaptive Sampling

Initialize: Multimodal encoder f_0, decoder π_0, feature buffer $\mathcal{B} \leftarrow \emptyset$
for task $k = 1$ to K **do**
  Input: Dataset $\mathcal{D}_k = \{(s_i^k, c_i^k, \tau_i^k)\}_{i=1}^{N_k}$
  if $k = 1$ **then**
    # perform behavior cloning for initial task
    Compute $\mathcal{L}_{\text{Curr}}$
    Update f_k, π_k by minimizing $\mathcal{L}_{\text{Curr}}$
  else
    # perform feature distillation
    Compute $\mathcal{L}_{\text{KD}}$ (Eq. 1) between f_{k-1} and f_k
    # perform feature replay
    Compute $\mathcal{L}_{\text{FR}}$ with $\mathcal{B}$ (Eq. 2)
    Update f_k, π_k by minimizing $\mathcal{L} = \mathcal{L}_{\text{Curr}} + \lambda_{\text{KD}}\mathcal{L}_{\text{KD}} + \lambda_{\text{FR}}\mathcal{L}_{\text{FR}}$ (Eq. 10)
  end if
  # perform adaptive experience selection in deep feature space
  Randomly sample p trajectories from $\mathcal{D}_k$: $\mathcal{T}_p = \{\tau_1^k, \ldots, \tau_p^k\}$
  for each trajectory τ in $\mathcal{T}_p$ **do**
    Encode frames $\{\boldsymbol{v}_t\}$ via CLIP
    Compute LOF scores (Eq. 3–5)
    Select keyframes $\mathcal{I} = \{t \mid \text{LOF}_k(\boldsymbol{v}_t) > 1\}$
    for each $t \in \mathcal{I}$ **do**
      Store $(f_k(o_t), a_t)$ in buffer $\mathcal{B}$
    end for
  end for
end for

C More Results

C.1 Old vs. New Task Performance per Stage

To provide a more intuitive understanding of how the model adapts to new tasks and the extent of forgetting over time, we report results on the HM3D dataset by separately evaluating performance on new tasks (i.e., target objects in the current stage) and old tasks (i.e., all previously seen target objects, averaged) in Table 6 and Table 7. These results reveal two key trends. First, Finetuning suffers from severe catastrophic forgetting. Although it achieves reasonable performance on new tasks (e.g., 64.63% in Stage 4), its success rate on old tasks drops below 10% after Stage 2. Second, C-Nav achieves a superior trade-off between stability and plasticity. While its new-task performance is slightly lower than that of Data Replay in later stages, it substantially outperforms all baselines on old tasks. Notably, in Stage 4, C-Nav attains an old-task success rate of 42.61%, which is 9.7 percentage points higher than the best baseline (Data Replay at 32.9%). This demonstrates that C-Nav effectively mitigates catastrophic forgetting while preserving the ability to learn new navigation tasks.

Table 6: SR on *new tasks* across stages (HM3D).

Method	Stage 1	Stage 2	Stage 3	Stage 4
Finetuning	69.63	60.55	29.54	64.63
LoRA	69.63	60.55	38.08	69.15
LwF	69.63	54.27	19.57	57.98
Merge	69.63	16.08	4.63	21.81
Data Replay	69.63	59.80	29.18	68.35
C-Nav	69.63	54.52	26.33	63.30

Table 7: SR on *old tasks* across stages (HM3D).

Method	Stage 1	Stage 2	Stage 3	Stage 4
Finetuning	69.63	7.72	9.83	9.05
LoRA	69.63	17.99	8.19	11.70
LwF	69.63	16.19	8.64	10.22
Merge	69.63	66.03	41.62	25.12
Data Replay	69.63	54.60	48.92	32.94
C-Nav	69.63	61.38	52.27	42.61

C.2 Additional Analysis on Feature Consistency Loss

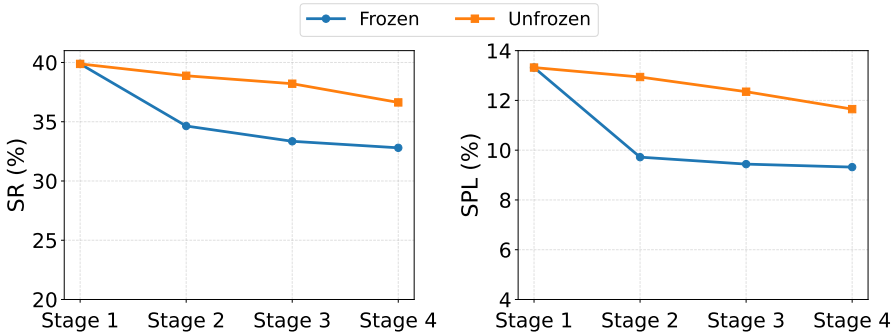

Figure 11: Ablation study: Freezing vs. unfreezing the multimodal encoder on MP3D Continual ObjectNav. We conduct experiments using a transformer-based architecture.

To validate this design choice, we conducted an ablation study on the MP3D Continual ObjectNav dataset by fully freezing the entire multimodal encoder. The results showed a performance drop across learning stages, confirming that full freezing harms generalization and underscoring the necessity of our consistency loss. Our feature consistency loss is designed to learn a shared feature space that remains effective across both past and new tasks. Simply freezing the entire multimodal encoder 11 may preserve old knowledge but limits the model's adaptability, especially when the original feature space does not sufficiently capture new task-specific semantics. Instead, we selectively fine-tune the modality-specific projectors within the encoder while applying consistency regularization to softly align the old and new feature spaces. This design allows the representations to stay connected, yet remain flexible enough to adapt to new tasks. It strikes a balance between preserving past knowledge and enabling plasticity for future learning.

C.3 Ablation on Loss Weighting Coefficient λ (MP3D)

Table 8: Ablation on λ for loss weighting (MP3D).

λ	Stage 1		Stage 2		Stage 3		Stage 4	
	SR	SPL	SR	SPL	SR	SPL	SR	SPL
1	39.88	13.32	38.00	12.25	35.41	11.00	36.22	10.85
5	39.88	13.32	38.37	12.29	37.32	10.97	36.81	11.90
10	39.88	13.32	36.66	11.80	36.68	11.45	25.29	9.51

We study the sensitivity of C-Nav to the loss weighting coefficient λ, which balances the replay loss (for retaining old knowledge) and the current-task loss (for learning new tasks). Table 8 reports success rate (SR) and SPL across four sequential training stages on MP3D, using $\lambda \in \{1, 5, 10\}$; all other model configurations are identical. As λ increases, the influence of replayed data as a regularizer grows stronger in later stages. While this helps preserve prior knowledge, an excessively large value (e.g., $\lambda = 10$) over-constrains learning and hinders adaptation to new tasks, resulting in a sharp performance drop in Stage 4 (SR = 25.29%). On the other hand, $\lambda = 1$ offers too little regularization, causing gradual forgetting of previously acquired skills. The setting $\lambda = 5$ strikes an effective balance: it yields slightly better overall performance than $\lambda = 1$ while maintaining strong retention of old tasks. Consequently, $\lambda = 5$ achieves the best trade-off between stability and plasticity and is adopted in all main experiments.

C.4 Results for the Dual-Path Anti-Forgetting Performance at Each Stage

To intuitively analyze the contribution of each component in C-Nav, we visualize the SR and SPL curves for each stage. As shown in the Figure 12, removing any component leads to a significant drop in both SR and SPL across all four model architectures and two datasets. Additionally, we report the average success rate for each component across the four architectures in Figure 13. The results

reveal that the multimodal encoder is prone to bias across tasks, leading to more severe forgetting. Specifically, removing the KD component causes substantial performance degradation: SR drops by 22.15 and 15.92 on HM3D-SR-Avg and MP3D-SR-Avg, respectively, and by 24.15 and 16.50 on HM3D-SR-Last and MP3D-SR-Last.

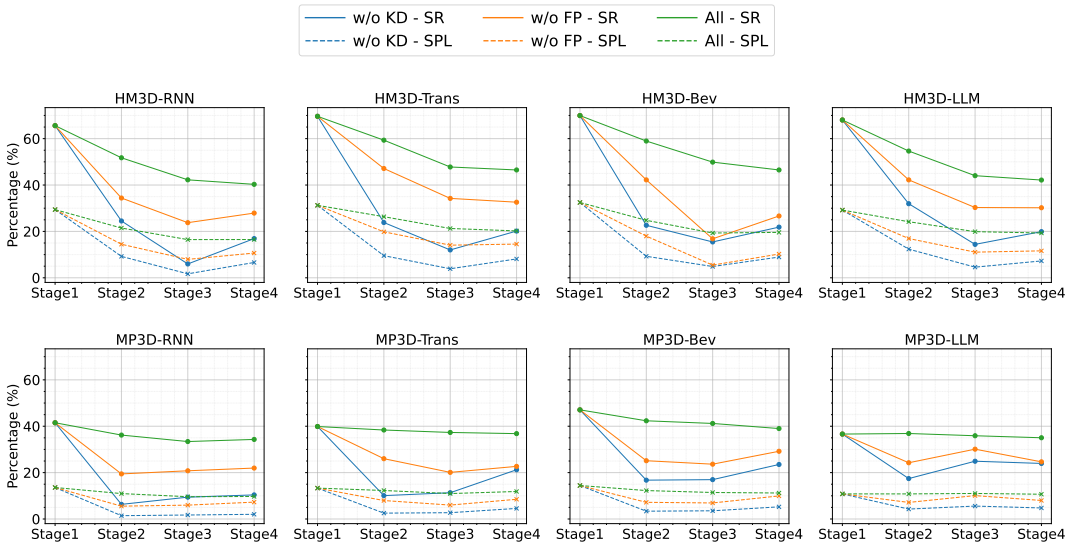

Figure 12: Results for the dual-path anti-forgetting performance at each stage.

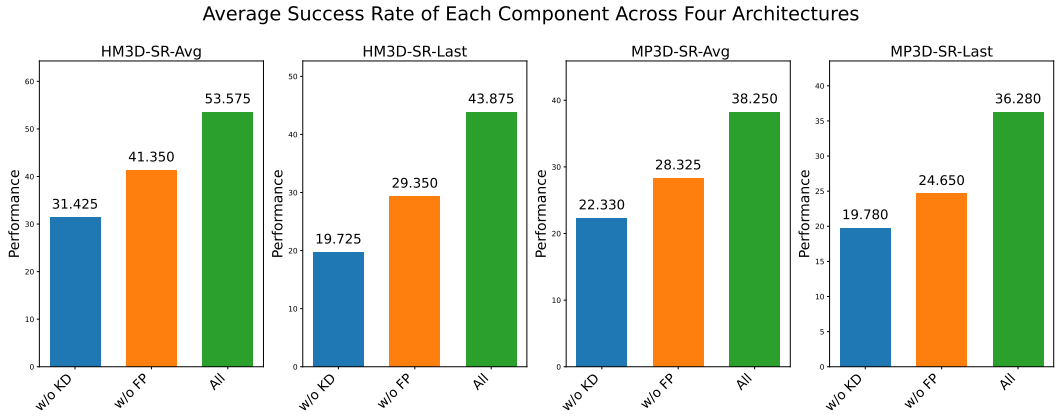

Figure 13: Average success rate for each component across four architectures. HM3D-SR-Avg and MP3D-SR-Avg denote the mean SR over all stages, while HM3D-SR-Last and MP3D-SR-Last indicate the mean SR in the final stage, averaged across architectures.

C.5 Results for the Adaptive Experience Selection at Each Stage

In Figure 14, we visualize the SR and SPL curves across different training stages under various sampling strategies. Our adaptive sampling method achieves performance comparable to training without any sampling, while clearly outperforming both uniform interval sampling and naive data replay approaches. As shown in Figure 15, the gains are particularly evident when compared to data replay: our method improves the average success rate by 0.8 and 2.0 on HM3D-SR-Avg and MP3D-SR-Avg, respectively, and by 0.625 and 4.625 on HM3D-SR-Last and MP3D-SR-Last. Compared to uniform sampling, our approach yields even larger improvements, boosting HM3D-SR-Last and MP3D-SR-Last by 3.375 and 7.3, respectively. These results demonstrate that our adaptive sampling strategy not only maintains competitive overall performance but also significantly enhances learning stability and final-stage effectiveness. By selective revisitation of information-rich trajectory frame features, the model better retains knowledge across tasks, leading to more robust continual learning in navigation scenarios

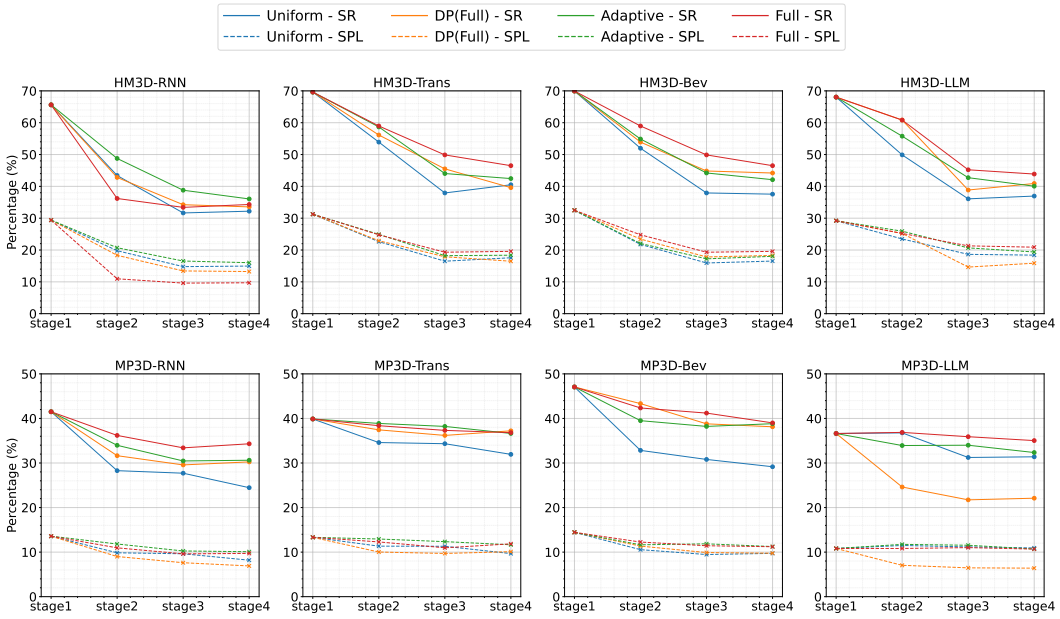

Figure 14: Results for the adaptive experience selection at each stage.

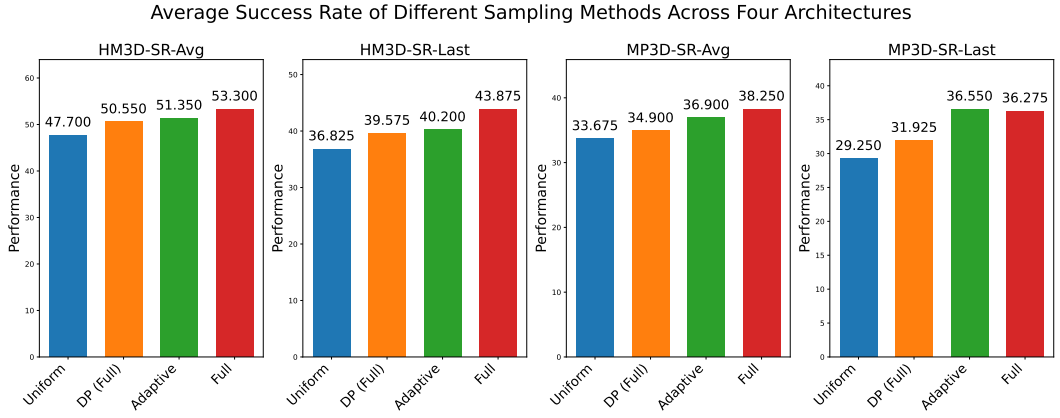

Figure 15: Average success rate for different sample methods across four architectures.