

Training Efficient Agents for Long-Term Decision
Making



Gunshi Gupta

Supervisor: Prof. Yarin Gal

Lady Margaret Hall

University of Oxford

July 2025

Acknowledgements

I would like to express my sincere gratitude to everyone who has supported me throughout my D.Phil. journey at Oxford.

First and foremost, I am deeply thankful to my advisor, Yarin Gal, for fostering an open, collaborative, and exploratory environment at OATML. Yarin's guidance and trust gave me the freedom to pursue my research interests while learning from the incredible community he helped build. I especially appreciated his balance of thoughtful feedback and allowing me the space to develop my own ideas, as well as the lab culture he shaped — one that encouraged intellectual curiosity, collaboration, and exploration. I am grateful to Tim G. Rudner for his excellent mentorship and advice over the years — supervising two of my projects, Tim taught me invaluable lessons about positioning and presenting research with clarity and polish. I am fortunate to have been advised and mentored by Adrien Gaidon and Rowan McAllister from TRI on my first research project. Our discussions on robotics and related challenges were formative, and I appreciated their guidance and support throughout. I would also like to thank Rahaf Aljundi from Toyota Motors Europe for her mentorship, openness, encouragement, and collaborative spirit, which made pursuing several interesting and diverse research directions possible. I always enjoyed our brainstorming sessions — her thoughtful questions and openness to exploring different ideas made them both productive and fun.

A special thanks to my main collaborator and partner throughout this journey, my husband Karmesh Yadav — none of this would have been possible without his insight, encouragement, and shared problem-solving, and he made the doctoral journey twice as fun as it might have been if we hadn't embarked on it together.

To all my OATML labmates — Lisa, Shreshth, Kelsey, Jannik, Angus, Andrew, Mo, Joost, Tim, Milad, Angelos, Clare, Tim, Panos, Pascal, Hazel, Daniella, Kunal, Yonatan, Matt, Luke and others — thank you for all the thoughtful discussions and for being a supportive research community. Lisa, in particular, for the long coffee

University of Oxford

walks and conversations, for the many small things I learnt from her, and for being such a grounding presence throughout my time at OATML — I'm glad we'll be in the same city again soon.

I would like to thank Katrina for managing the many moving parts of OATML so smoothly and generously, all while pursuing her own studies at Oxford. Her support and involvement greatly improved our lab's functioning. And to Xiaowen Dong, for looking out for students at LMH and regularly checking in — thank you. I am grateful to my transfer and confirmation assessors, Joao Henriques and Jan-Peter Calliess, for their feedback during key milestones.

My internship at Microsoft Research was another important part of this journey. I am thankful to Katja Hofmann, Sam Devlin, Tim Pearce, Tabish Rashid, Anssi Kanervisto, Tarun Gupta, Lukas, Raluca, Udit, and the rest of the Gaming Intelligence team for the opportunity to learn and contribute during my time there. I also valued working with and learning from Cong Lu on different reinforcement learning projects; his positivity and motivation made every interaction rewarding. My thanks as well to Yusuf, with whom I collaborated on my latest project, and to Zsolt Kira and Dhruv Batra for their informal advising and perspectives.

To the AIMS CDT cohort — Kelsey, Shreshth, Mathew, Patrick, Yash, Benedetta, Seb, Aleks, and others — I appreciated sharing this experience with all of you. And to Wendy Poole, for being the ever-reliable CDT course administrator who helped navigate everything behind the scenes with kindness and humour.

On a personal note, I am endlessly grateful to my parents Anju and Rakesh, and my sister, Sachi, for their unwavering support and understanding throughout this long process. Thank you for always being home. To Leuuba and Gaurav, for being a home away from home in London — thank you for your friendship and the fun times over the years.

A heartfelt thank you to Poorva, Akshay, Yash, Nitish, and Priyansh — my very first friends in Oxford, with whom I shared countless late-night conversations and hangouts that made Oxford feel like home. Living with Poorva and Akshay in my

University of Oxford

second year especially helped shape some of my most memorable experiences here. I am grateful for the sense of belonging and friendship you all offered during this time. To friends from different walks of life in the UK — Ragini, Manav, Urvashi, Disha, Sami, Sparsh, Prakhar, Kshitij, Srishti, and many others — thank you for your companionship along the way. And finally, to Oxford and the many coffee shops around the world that served as my writing and thinking spaces, thank you for the quiet corners and steady caffeine.

My doctoral studies would not have been possible without the support of Toyota Motor Europe and the collaborative spirit of the broader Toyota Research Institute. I am grateful for their generous funding and encouragement throughout my Ph.D.

Abstract

Reinforcement learning has ventured from tabletop simulators to real robots and open-world games, but today’s agents still learn with prohibitively low sample efficiency, ignore the priors encoded in foundation models, and forget most of what they have seen after a few hundred steps. This thesis pursues a unifying agenda—*efficiently training efficient decision-making agents*—through three successive contributions.

Chapter 3 demonstrates that sample efficiency can be substantially improved by re-weighting experience toward the transitions that are most informative. An ensemble-based uncertainty criterion selectively upsamples those rare interactions that clarify causal structure, enabling offline RL to achieve safe, performant policies with far fewer gradient updates than uniform replay.

Stronger supervision is possible even when no new interaction data are collected, provided we can import structure learned elsewhere. **Chapter 4** investigates this idea by tapping the internal representations of large generative vision models. Text-to-image diffusion backbones, although trained for synthesis rather than control, accumulate multi-scale spatial and semantic cues that are difficult to rediscover from scratch in a robotics dataset. By freezing these backbones and projecting their multi-layer activations into a control-friendly embedding—what we term Stable Control Representations (SCRs)—an agent starts with a rich inductive prior over object geometry and language grounding. In manipulation and open-vocabulary navigation tasks, SCRs cut the number of gradient steps needed to reach a given return by up to an order of magnitude and consistently outperform contrastively trained encoders, all without generating a single additional pixel. This result shows that re-using pretrained knowledge can convert computationally expensive exploration into cheap representation reuse, markedly improving sample efficiency.

While these chapters focus on learning efficiently, deployed agents must also *act* efficiently by leveraging context that spans hours or days. **Chapter 5** introduces

University of Oxford

Memo, a transformer policy that interleaves periodic summary tokens with streaming observations so memory capacity grows gently with task length. To measure such long-term reasoning, **Chapter 6** contributes *FindingDory*, a procedurally extendable benchmark family whose 60 tasks probe how well embodied agents store and retrieve experience.

Together, these works chart a coherent path toward agents that **learn quickly, inherit rich priors, and remember what matters**, moving a step closer to truly lifelong, self-improving intelligence.

Contents

1	Introduction	13
2	Background	17
2.1	Reinforcement Learning	17
2.1.1	Problem Formulation	17
2.1.2	On-policy Learning	18
2.1.3	Off-policy Learning	19
2.2	Uncertainty Quantification	20
2.2.1	Deep Ensembles	20
2.2.2	Active Learning and Active Sampling	20
2.3	Transformers for Generative Modelling	22
2.3.1	Architecture and Self-Attention	22
2.3.2	Training and Inference	23
2.3.3	Context Processing and Scaling Limits	24
2.3.4	Roles in Sequential Decision Making	24
3	Leveraging Uncertainty for Sample-Efficient Policy Learning	25
3.1	Introduction	26
3.2	Preliminaries	29
3.3	Related Work	31
3.4	Alleviating Causal Confusion in Offline RL via Active Sampling	33

3.4.1	Conditional Average Treatment Effect Estimation from Causally Ambiguous Data	33
3.4.2	Active Sampling	34
3.4.3	Computing Acquisition Scores	37
3.5	Experiments	38
3.5.1	Experiment Design	38
3.6	Evaluation Metrics	39
3.6.1	Illustrative Example: Traffic-World	40
3.6.2	Assessing Generalization in Offline Reinforcement Learning: Procgen	42
3.6.3	Causal Confusion in the ALE Benchmark	44
3.6.4	Effect of Predictive Uncertainty on Active Sampling and Causal Confusion	46
3.7	Computational Cost	47
3.8	Conclusion	48
3.9	Discussion	48
4	Leveraging Existing Knowledge for Sample-Efficient Policy Learning	51
4.1	Introduction	52
4.2	Related Work	54
4.3	Background	55
4.3.1	Diffusion Models	55
4.3.2	Representation Learning for Visual Control	56
4.3.3	Policy-learning notation	57

4.4	Stable Control Representations	57
4.4.1	Layer Selection and Aggregation	58
4.4.2	Diffusion Timestep Selection	59
4.4.3	Prompt Specification	60
4.4.4	Intermediate Attention Map Selection	60
4.4.5	Using Text-to-Image Diffusion Model Representations to Learn Control Policies	60
4.4.6	Fine-Tuning on General Robotics Datasets	61
4.5	Empirical Evaluation	61
4.5.1	Baselines	62
4.5.2	Few-shot Imitation Learning	63
4.5.3	Image-Goal Navigation	64
4.5.4	Open Vocabulary Mobile Manipulation	65
4.5.5	Fine-tuning CLIP	67
4.5.6	Referring Expressions Grounding	67
4.5.7	Grasp Affordance Prediction	69
4.5.8	Comparison with LIV	69
4.5.9	Overall Ranking of Representations	70
4.6	Deconstructing Stable Control Representations	70
4.6.1	Layer Selection	71
4.6.2	Sensitivity to the Noising Timestep	73
4.6.3	How is Language Guiding the Representations?	73
4.6.4	The Effect of Spatial Aggregation	75

4.7	Discussion	76
4.8	Conclusion	77
5	Learning Policies to Leverage Experience	79
5.1	Introduction	80
5.2	Related Work	82
5.3	Methodology	83
5.3.1	Problem Setting	84
5.3.2	Context Summarization in Memo	85
5.4	Experiments	87
5.4.1	Benchmarks	87
5.4.2	Baselines	89
5.4.3	Summarization Outperforms Full In-Context Access	90
5.4.4	Advantage of Summary Accumulation Over Fixed Recurrent Memory	91
5.4.5	Importance of Long-Horizon Gradient Propagation: Compar- ison to Autocompressors	93
5.4.6	Extrapolation and Experience Subsampling During Evaluation	94
5.4.7	Finetuning	94
5.4.8	Ablations.	95
5.5	Discussion	96
5.6	Limitations	97
6	Benchmarking Multimodal Agents on Memory-Intensive Tasks	99
6.1	Introduction	99

6.2	Related Work	101
6.2.1	Video QA benchmarks	102
6.2.2	Embodied AI benchmarks	102
6.3	FINDINGDORY	104
6.3.1	Benchmark Overview	104
6.3.2	Tasks	106
6.3.3	Implementation details	107
6.4	Experiments	108
6.4.1	Evaluated Approaches	108
	High-Level Goal Selection	109
	Low-Level Navigation	110
6.4.2	Evaluation Metrics	111
6.5	Results	112
6.6	Conclusion	116
7	Discussion and Future Directions	118
7.1	Active Sampling in the Scaling Era	118
7.2	Representation Reuse in Embodied Control	119
7.3	Memory and Long-Context Reasoning	119
7.4	The Road Ahead	120
	Bibliography	122
A	Leveraging Uncertainty for Sample-Efficient Policy Learning	149
A.1	Implementation	149

University of Oxford

A.2	Further Analysis	149
A.3	Data Collection	151
A.4	Hyperparameters	152
B	Leveraging Existing Knowledge for Sample-Efficient Policy Learning	155
B.1	Task Descriptions	155
B.1.1	Few-Shot Imitation Learning	155
B.1.2	Open Vocabulary Mobile Manipulation	156
B.1.3	ImageNav	157
B.2	Implementation Details	158
B.2.1	Fine-tuning Stable Diffusion	158
B.2.2	Representation Extraction Details	161
B.3	Runtime Analysis	163
B.3.1	Hyperparameters	163
C	Learning Policies to Leverage Experience	164
C.0.1	Task Setting and Model architecture: EXT OBJNAV	164
C.0.2	Task Setting and Model architecture: Dark-Key-To-Door	165
C.0.3	Training Scenes Dataset: EXT OBJNAV	165
C.0.4	Pseudocode for Memo	166
C.0.5	Extended Result: FLOPs Comparison	166
C.0.6	Extended Result: Memory-Maze	168
C.0.7	Extended Ablation: Direct KV Cache Construction from Summary Embeddings	168

C.0.8	Training-Validation Gap	170
C.0.9	Sample Efficiency of Memo	170
C.0.10	Hyperparameters	171
D	Benchmarking Multimodal Agents on Memory-Intensive Tasks	173
D.1	Limitations	173
D.2	Benchmark Details	174
D.2.1	FINDINGDORY Episode Creation	174
D.2.2	Experience Collection Phase	176
D.2.3	Evaluations	178
D.3	Extended Related Work	180
D.3.1	An overview of approaches tackling memory in Long VQA and embodied tasks	180
D.4	Baseline Details	182
D.4.1	Gemini Agent	182
D.4.2	Qwen and Gemma Agent	182
D.4.3	GPT Agent	182
D.4.4	Text Agent	183
D.4.5	Supervised Finetuning Baseline	185
D.5	Low-level Policy Details	188
D.5.1	Image-Goal Navigation Agent	188
D.5.2	Explicit Mapping Agent	189
D.6	Qualitative Examples	190

1 | Introduction

Recent advancements in deep learning have significantly accelerated progress in artificial intelligence. Foundation models—large neural networks trained on vast web-scale datasets, typically containing on the order of 10^8 – 10^{12} samples aggregated from the web—have demonstrated remarkable capabilities in generalization, enabling effective adaptation to new tasks such as question-answering, image captioning, and code generation without additional fine-tuning. Concurrently, embodied agents, including robots, virtual assistants, and interactive game characters, are increasingly deployed to perform complex sequential decision-making tasks in dynamic environments. However, unlike foundation models, these embodied agents typically require extensive, task-specific interactions to achieve competence, often struggling to generalize across variations in tasks, instructions, or environmental conditions.

This difference highlights a critical motivation in the current state of AI research: developing intelligent agents capable of efficiently and effectively performing useful tasks in diverse real-world scenarios. Bridging the strengths of foundation models—namely, generalizability and efficient reuse of knowledge—and the requirements of embodied agents—robustness in sequential decision-making and long-term memory—has become an important objective in modern AI. While initial steps have been made toward transferring large-scale pre-trained foundational knowledge to embodied settings, substantial challenges remain in efficiently learning from relevant experiences, leveraging high-quality visual representations for fine-grained tasks, and maintaining long-term contextual memory in practical real-time settings.

Addressing these challenges requires:

- **Efficient sampling of long-tailed behavior logs.** Human-generated demonstrations naturally result in imbalanced datasets, where critical yet infrequent scenarios—such as unexpected object occlusions or uncommon interactions—form only a small portion of the total data. Treating all interactions equally can

lead to inefficient training and fragile policies. While language modeling research has begun emphasizing curated, higher-quality training data or adaptive sampling strategies, similar innovations must be translated into embodied learning pipelines.

- **Identifying effective visual representations for control tasks.** Despite continual improvements in self-supervised and multimodal visual representations, it remains unclear which existing encoders best support spatial precision required by manipulation and navigation tasks, particularly when tasks involve language-based instructions. Selecting or adapting visual backbones suitable for precise embodied tasks is crucial for transferring the representational strength of foundation models into the domain of control.
- **Scalable long-term context modeling in real-time.** Transformer models’ ability to handle longer context windows has improved dramatically, but efficiently using very large contexts in real-time scenarios remains challenging. Embodied agents often need to process and reason over extensive interaction histories, significantly exceeding typical benchmark scales, without incurring prohibitive computational latency.

These core challenges motivate the central thesis of this dissertation:

Efficiently training *efficient* decision-making agents.

We pursue this objective through the following research threads:

Focused sampling during learning. Behavioral datasets inherently exhibit long-tailed distributions, necessitating strategies to prioritize learning from rare yet highly informative events rather than indiscriminately collecting more data. Chapter 3 introduces CLEAR, an uncertainty-driven replay method that dynamically emphasizes experiences crucial for effective policy improvement. Compared to traditional uniform replay or fixed heuristic methods, CLEAR significantly reduces training costs while maintaining or improving policy robustness.

Leveraging existing models and data. At the outset of this research, CLIP representations were widely adopted in robotics, despite concerns regarding their spatial fidelity for precise tasks. Chapter 4 demonstrates that internal representations from text-to-image diffusion models contain both detailed spatial and rich semantic information beneficial for robotic manipulation and navigation. Our Stable Control Representations enable effective transfer without requiring extensive task-specific fine-tuning or large-scale new data collection.

Making use of long-term experience. Despite substantial growth in transformer context lengths over recent years, embodied agents typically utilize limited temporal contexts due to computational constraints. Chapter 5 introduces *Memo*, a transformer-based policy incorporating periodic summary tokens to achieve sub-linear memory usage growth over time, making long-term memory feasible for real-time inference. To benchmark these capabilities, we introduce *FindingDory* (Chapter 6), a procedurally extendable dataset featuring 57 scenarios explicitly designed to test long-duration memory and reasoning.

Organisation. The thesis is structured as follows: following a comprehensive background chapter, each research thread described above is detailed in a dedicated chapter. A concluding discussion explores open questions, such as scaling adaptive sampling to internet-scale behavioral logs, integrating rich 3D representations from foundation models into embodied control, and enhancing the feedback loop between memory mechanisms and exploratory behaviors.

Publications and Impact. The research presented in this dissertation is grounded in a series of both peer-reviewed and under-review publications. These include works on adaptive experience prioritization in reinforcement learning ("*Can Active Sampling Reduce Causal Confusion in Offline Reinforcement Learning?*", Clear 2025), leveraging diffusion representations for embodied control ("*Pre-trained Text-to-Image Diffusion Models Learn Versatile Representations for Control*", NeurIPS 2024 spotlight), scalable memory architectures for long-horizon agents ("*Memo: Training Memory-Efficient Embodied Agents with Reinforcement Learning*", NeurIPS

University of Oxford

2025 spotlight), and benchmarks for evaluating memory in embodied settings (*"FindingDory: A Benchmark to Evaluate Memory in Embodied Agents"*, Under Review). Together, they have appeared at leading venues, reflecting both the novelty and relevance of the contributions. Several of these papers have already been cited and extended by subsequent research indicating that the ideas explored here are beginning to influence the broader embodied AI community.

Collectively, this dissertation advocates for an integrated approach: *prioritize experiences effectively, utilize existing knowledge intelligently, and retain critical context efficiently*. This strategy provides foundational steps toward creating lifelong, continuously improving agents capable of robust and efficient performance in diverse, open-world environments.

2 | Background

Contents

2.1 Reinforcement Learning	17
2.1.1 Problem Formulation	17
2.1.2 On-policy Learning	18
2.1.3 Off-policy Learning	19
2.2 Uncertainty Quantification	20
2.2.1 Deep Ensembles	20
2.2.2 Active Learning and Active Sampling	20
2.3 Transformers for Generative Modelling	22
2.3.1 Architecture and Self-Attention	22
2.3.2 Training and Inference	23
2.3.3 Context Processing and Scaling Limits	24
2.3.4 Roles in Sequential Decision Making	24

This chapter reviews the three pillars that support the thesis: (1) reinforcement learning (RL), (2) uncertainty quantification, and (3) modern generative modelling. The treatment is intentionally tutorial in style—first building the core ideas from first principles, then positioning the specific methods (e.g. PPO, CQL, deep ensembles) that re-occur in later chapters.

2.1 Reinforcement Learning

2.1.1 Problem Formulation

We model an environment as a *Markov Decision Process* (MDP) $\mathcal{M} = (\mathcal{S}, \mathcal{A}, \mathcal{P}, r, d_0, \gamma)$ with state space \mathcal{S} , action space \mathcal{A} , transition kernel $\mathcal{P}(s' | s, a)$, reward function $r: \mathcal{S} \times \mathcal{A} \rightarrow \mathbb{R}$, initial state distribution d_0 , and discount factor $\gamma \in [0, 1)$. A *policy* $\pi_\theta(a | s)$, parameterised by $\theta \in \mathbb{R}^d$, induces a distribution over infinite horizon trajectories $\tau = (s_0, a_0, r_0, s_1, \dots)$, where $s_0 \sim d_0, a_t \sim \pi_\theta(\cdot | s_t), s_{t+1} \sim \mathcal{P}(\cdot | s_t, a_t)$.

University of Oxford

Return and Objective. The *return* of a trajectory is $R(\tau) = \sum_{t=0}^{\infty} \gamma^t r(s_t, a_t)$, and the goal of RL is to maximise the expected return

$$J(\theta) = \mathbb{E}_{\tau \sim \pi_\theta} [R(\tau)], \quad \theta^* = \arg \max_{\theta} J(\theta). \quad (2.1)$$

Value functions and Bellman operators. The state-value and action-value functions under π are

$$V^\pi(s) = \mathbb{E}_{\pi, \mathcal{P}} \left[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t) \mid s_0 = s \right], \quad Q^\pi(s, a) = \mathbb{E}_{\pi, \mathcal{P}} \left[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t) \mid s_0 = s, a_0 = a \right].$$

For many algorithms it is convenient to express value updates via the *Bellman optimality operator*

$$(\mathcal{B}Q)(s, a) = r(s, a) + \gamma \mathbb{E}_{s' \sim \mathcal{P}} \left[\max_{a'} Q(s', a') \right].$$

The optimal action-value function Q^* is defined as the unique fixed point of this operator, i.e. $Q^* = \mathcal{B}Q^*$. In practice, the expectation over next states is approximated using sampled transitions.

2.1.2 On-policy Learning

On-policy methods improve the very policy that generates the data they consume. Each gradient step changes the data-generating policy, which limits the reuse of past samples when learning the policy directly, favouring stability over raw sample reuse.

Policy gradient. The policy-gradient theorem yields

$$\nabla_{\theta} J(\theta) = \mathbb{E}_{s, a \sim \pi_{\theta}} \left[\nabla_{\theta} \log \pi_{\theta}(a \mid s) Q^{\pi_{\theta}}(s, a) \right].$$

Replacing $Q^{\pi_\theta}(s, a)$ with a Monte Carlo estimate of the return

$$\hat{Q}(s_t, a_t) = \sum_{t'=t}^T \gamma^{t'-t} r_{t'}$$

yields the REINFORCE algorithm [?]. Introducing a baseline and estimating the resulting advantage estimator defined as $\hat{A}(s, a) = \hat{Q}(s, a) - b(s)$ leads to actor-critic methods, where $b(s)$ is a baseline, commonly chosen as a learned state-value function $V_\phi(s)$ to reduce gradient variance.

Proximal Policy Optimisation. Proximal Policy Optimisation (PPO) [135] constrains each update by clipping the importance ratio $\rho_t = \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{\text{old}}}(a_t|s_t)}$ and maximising $\min(\rho_t \hat{A}_t, \text{clip}(\rho_t, 1-\varepsilon, 1+\varepsilon) \hat{A}_t)$. The clipped surrogate curbs destructive steps while retaining the simplicity of vanilla policy gradients. PPO serves as the on-policy baseline throughout this thesis.

2.1.3 Off-policy Learning

Off-policy algorithms learn from trajectories generated by *any* behaviour policy μ , making heavy use of replay buffers and logged data. Key ingredients are:

- **Value bootstrapping.** Temporal-difference (TD) methods update value estimates by minimising squared TD errors of the form $(Q_\phi(s_t, a_t) - y_t)^2$, where the target y_t bootstraps from subsequent value estimates. Because the target depends on the current parameters (or their delayed copies), learning corresponds to approximate fixed-point iteration rather than optimisation of a single static objective. Target networks or double estimators are commonly used to stabilise this process.
- **Importance sampling.** When explicit corrections for the ratio π/μ are feasible, they provide unbiased value estimates at the cost of higher variance.
- **Actor-critic methods.** Actor-critic algorithms maintain both a policy π_θ and a value function Q_ϕ or V_ϕ , using the critic as a low-variance proxy for

long-horizon returns. While actor–critic methods can be on-policy or off-policy, many modern off-policy algorithms (e.g. TD3, SAC) fall into this class.

A note on offline RL: Offline (batch) RL is characterised by a fixed dataset collected *a priori*, with no further environment interaction. While offline RL often involves learning a policy different from the behaviour policy (off-policy), the defining challenge is learning under severe distribution shift rather than only off-policy evaluation. These issues are the focus of Chapter 3.

2.2 Uncertainty Quantification

In both control and perception systems, acting without an estimate of *confidence* can lead to unsafe or sub-optimal behaviour. We distinguish two types:

- *Epistemic uncertainty* (knowledge uncertainty) stems from limited data and can, in principle, be reduced.
- *Aleatoric uncertainty* (intrinsic noise) is irreducible given the input.

2.2.1 Deep Ensembles

Training M independent models $\{f^{(m)}\}_{m=1}^M$ with different random initialisations or data resamplings yields simple yet powerful *epistemic* uncertainty estimates [83]. The predictive mean $\bar{f}(x) = \frac{1}{M} \sum_m f^{(m)}(x)$ is used for inference, while the variance

$$\text{Var}[f^{(m)}(x)] = \frac{1}{M-1} \sum_m (f^{(m)}(x) - \bar{f}(x))^2$$

serves as an epistemic uncertainty proxy.

2.2.2 Active Learning and Active Sampling

Active learning (supervised setting). Classic *active learning* assumes a large *pool* of unlabelled inputs $\mathcal{X} = \{x_i\}_{i=1}^N$ and a costly labelling oracle. The learner iteratively selects a query subset $\mathcal{Q} \subseteq \mathcal{X}$, receives ground-truth labels

$\{y_i\}_{x_i \in \mathcal{Q}}$, updates its hypothesis f_θ , and repeats. The crux is the *acquisition function* $a(x; \theta)$ —a scalar scoring rule that ranks points by their expected utility. Throughout this section, θ denotes the parameters of the predictive model being trained; acquisition functions are deterministic functions of the model’s current state and do not introduce additional learnable parameters unless stated otherwise. Popular choices include

- **Uncertainty sampling:** pick the inputs with largest predictive entropy or smallest margin [88, 138].
- **Query-by-committee:** use an ensemble; select points on which committee members disagree most [139].
- **Expected model change / error reduction:** estimate how much updating on a candidate would shift the parameters or reduce the generalisation error [25].

Active *sampling* (limited-data setting). When labels are *already* available—as in offline RL or in fully annotated vision datasets—the budget constraint shifts from *labelling cost* to *gradient-budget* (how many examples can reasonably be processed per update). Here the goal is to *re-weight* or *sub-sample* the fixed set $\mathcal{D} = \{(x_i, y_i)\}$ so that learning focuses on the most informative or mis-predicted instances. Concretely, active sampling instantiates a probability distribution

$$p_{\text{sample}}(i \mid \theta) \propto a((x_i, y_i); \theta),$$

and draws the next minibatch from that distribution. Unlike pool-based active learning, no new labels are queried; only the *presentation order* and *frequency* of existing samples change.

RL analogues to active learning and sampling.

- **Online RL.** Uncertainty-guided exploration chooses state–action pairs that maximise epistemic uncertainty (e.g. bootstrapped DQN [112], optimistic posterior sampling [130]). The acquisition function influences *which new transitions are*

collected.

- **Offline / replay-buffer RL.** Prioritised Experience Replay (PER) [134] is an early form of active sampling: transitions with large temporal-difference (TD) errors, defined as

$$\delta_t = r_t + \gamma \max_a Q(s_{t+1}, a) - Q(s_t, a_t),$$

are replayed more often. Recent work extends PER with model-based epistemic uncertainty, ensemble variance, or information-gain criteria, targeting under-represented, high-leverage parts of the dataset.

Why distinguish the two? *Active learning* asks “which **new** label should I buy?”; *active sampling* asks “which **existing** labelled example should I study next?”. Both rely on acquisition functions, but they differ in the resource that is scarce (labels vs. gradient updates) and in the impact on the data distribution (querying enlarges \mathcal{D} ; sampling re-weights it).

2.3 Transformers for Generative Modelling

Generative modelling seeks a parametrised distribution $\hat{p}_\theta(x)$ or conditional distribution $\hat{p}_\theta(x | c)$ whose samples are indistinguishable from those drawn from the true data distribution, given optional conditioning information c (e.g. text prompts, class labels, or context). Among the many families proposed over the past decade, the *transformer* has emerged as the dominant work-horse—remarkably capable of aligning to diverse modalities once the raw signal is cast into a sequence of discrete *tokens*. This section reviews the core mechanics that make transformers a natural fit for the sequential decision-making problems addressed later in the thesis.

2.3.1 Architecture and Self-Attention

A transformer layer replaces recurrence with a *self-attention* operation. Given query, key, and value matrices $Q, K, V \in \mathbb{R}^{L \times d_k}$ derived from the input sequence of length

University of Oxford

L , attention computes

$$\text{Attn}(Q, K, V) = \text{softmax}(QK^\top / \sqrt{d_k})V,$$

so every position aggregates information from all others in a single matrix multiplication. Multi-head attention repeats this operation h times with independent projections, then concatenates the heads; stacking N such blocks with feed-forward layers yields a depth- N transformer in which each layer can model relationships across the full context.

Each transformer block contains learnable projection matrices for the query, key, and value mappings, as well as output projections, followed by a position-wise feed-forward multilayer perceptron (MLP). Layer normalisation and residual connections are applied around both the attention and MLP sublayers, yielding the standard Transformer encoder architecture [159]. All such weights collectively constitute the model parameters θ .

Tokenisation. Images can be broken into patches [158], audio can be framed [11], and continuous control states can be vector-quantised [53], all producing a uniform token interface when the aforementioned quantities are passed through an embedding layer. The same architecture can therefore serve language, vision, and robotics once modality-specific front-ends perform the discretisation.

2.3.2 Training and Inference

With *causal masking*, a transformer is trained by *maximum-likelihood estimation* to predict the next token, giving the factorisation

$$p_\theta(x_{1:L}) = \prod_{t=1}^L p_\theta(x_t | x_{<t}).$$

Because all targets are known, gradients can be computed in parallel over sequence positions, making training throughput limited mainly by hardware. Generation, in contrast, is *autoregressive*: tokens are sampled one at a time and fed back into the

model, so inference latency scales linearly with output length.

2.3.3 Context Processing and Scaling Limits

Each attention layer incurs $O(L^2)$ time and memory, constraining the context length L that fits on present-day accelerators. Sparse kernels, linear attention, and recurrence relax but do not eliminate the quadratic wall; efficient context handling therefore remains a key research axis—especially for robotics, where long sensory histories matter. Despite this cost, scaling depth (N) and width (d_k) reliably improves performance, a phenomenon captured by empirical *scaling laws* that hold across modalities [69].

2.3.4 Roles in Sequential Decision Making

Given their expressive sequential modeling abilities, transformers have penetrated the RL toolkit in three complementary roles:

1. **World models** that roll out future sensory streams, giving planners a rich imagination without extra environment queries. While powerful, world models are not studied further in this thesis and are mentioned here for completeness.
2. **Policy networks** that map tokenised state sequences to action distributions, benefiting from the model’s ability to pool information over long horizons.
3. **Value critics** whose global receptive field over *time* (long histories) and, where applicable, *observation structure* (e.g. spatial or multi-modal inputs) sharpens credit assignment in delayed-reward tasks.

These roles motivate the memory-efficient transformer architecture introduced later (Chapter 5) and the long-horizon benchmarks of Chapter 6.

3 | Leveraging Uncertainty for Sample-Efficient Policy Learning

Contents

3.1	Introduction	26
3.2	Preliminaries	29
3.3	Related Work	31
3.4	Alleviating Causal Confusion in Offline RL via Active Sampling	33
3.4.1	Conditional Average Treatment Effect Estimation from Causally Ambiguous Data	33
3.4.2	Active Sampling	34
3.4.3	Computing Acquisition Scores	37
3.5	Experiments	38
3.5.1	Experiment Design	38
3.6	Evaluation Metrics	39
3.6.1	Illustrative Example: Traffic-World	40
3.6.2	Assessing Generalization in Offline Reinforcement Learning: Progen	42
3.6.3	Causal Confusion in the ALE Benchmark	44
3.6.4	Effect of Predictive Uncertainty on Active Sampling and Causal Confusion	46
3.7	Computational Cost	47
3.8	Conclusion	48
3.9	Discussion	48

3.1 Introduction

Offline learning offers opportunities to scale reinforcement learning to domains where offline data is plentiful but online interaction with the environment is costly. The central challenge of offline reinforcement learning is that the effects of actions must be inferred from a fixed dataset, without the ability to perform targeted interventions or collect counterfactual experience. As a result, learning reliable action–outcome relationships depends critically on the coverage of the dataset over relevant states and actions. In the absence of online interactions, our hope is that the dataset uniformly covers an exhaustive set of plausible scenarios. This is often not the case in datasets for robotic control, where the state–action distribution exhibits a long tail of rare but important events that are sparsely represented in the data.

This work focuses on failure modes that arise when learning from such imperfect datasets. We distinguish between spurious correlations and causal confusion, terms often used interchangeably but representing meaningfully different phenomena.

Spurious correlations arise when the agent observes variables that correlate with reward or success during training, even though these variables have no causal influence on the outcome. The misuse of these features is an optimisation artifact: models trained with gradient descent naturally exploit any predictive statistical pattern available in the data, even when the underlying mechanism does not support that dependence. This is especially pronounced in models trained on high-dimensional visual inputs, where it is far easier for neural networks to latch onto shortcuts—such as background co-occurrence cues—than to extract the true causal structure of the scene [17, 49].

In contrast, causal confusion refers to cases where the agent lacks access to the true causal variables or their interactions, typically due to unobserved confounders or unmodeled aspects of the environment. As clarified by prior work [38], causal confusion occurs when multiple causal explanations are consistent with the observations in the offline dataset, making it impossible for a policy to correctly attribute

which variables drive downstream outcomes. For example, if an unobserved variable influences both a reward and an observed feature, the agent may incorrectly treat the observed feature as causally relevant.

In practice, offline RL datasets often exhibit both issues: spurious visual features that correlate with reward, and ambiguous causal structures in regions of poor dataset coverage. By the above distinction, many of our experimental scenarios involve spurious correlations; we nevertheless adopt the term causal confusion for alignment with the established terminology in earlier work.

To illustrate, consider a case where an agent consistently observes variables X and Y together whenever reward R occurs, and R depends causally on Y but not on X . If X is observed, the agent may simply exploit the correlation between X and R (a spurious correlation). If instead a hidden confounder jointly generates X , Y , and R , the agent has no way to disambiguate the true causal mechanism, leading to causal confusion. Problematically, when the correlation between X and R breaks at deployment time, a model that relies on either form of misattribution can suffer sharp performance degradation.

Although spurious features might not *perfectly* explain all the input-label pairs in offline data, optimisation methods such as stochastic gradient descent benefit from correlations in the data when seeking to reduce the training loss. Therefore spurious correlations are unwittingly transferred from the data to the mechanisms of learned models. This phenomenon is especially pronounced in models trained on high-dimensional visual inputs, since extracting the true causal factors of an environment (and their interplay) from an image is a particularly difficult problem. On the other hand, it is noticeably easy for neural networks to find shortcuts for prediction, for instance, using co-occurrence patterns between objects and backgrounds to successfully perform object detection, as has been extensively documented in the computer vision literature [17, 49]. Several works have also reported causal confusion in control policy learning and used heuristic loss re-weighting or dataset-balancing schemes based on domain knowledge of the prediction task to reduce the amount

of causal confusion in the learned models [167, 81]. However, these heuristics require a practitioner to know the source of causal confusion *a priori* since they explicitly upweigh the loss for specific spurious-correlation-breaking samples in the dataset [166, 165]. While this may be limiting in practice, the success of these heuristics demonstrates that it is possible to recover less causally-confused models from a fixed dataset, as long as we have access to an *oracle* sampling scheme for the training data.

This insight suggests that causal confusion in learned models may primarily be an artifact of the optimisation procedure and raises the question of whether systematically providing data points that expose spurious correlations to the optimizer may allow alleviating causal confusion. Unfortunately, in practice, identifying such sets of data points for a given task is challenging. Furthermore, naive sampling schemes, such as uniform sampling from the dataset for a sufficiently long training horizon, may eventually be able to resolve causal confusion in a learned model. For instance, if a model has achieved close-to-zero loss on the dominant scenarios, high loss from the tail-cases may begin to influence optimisation. However, even if such naive approaches were successful at partially or fully resolving causal confusion, for reasons of computational efficiency and predictive performance (e.g., early stopping), their practical usefulness would be limited.

In this chapter, we investigate the efficacy of data sampling strategies in mitigating causal confusion in offline reinforcement learning from datasets exhibiting causal ambiguity. Our contributions can be summarised as follows:

1. We design three decision-making tasks to study causal confusion in offline RL from visual inputs: a grid-world simulation of traffic-light navigation; a maze navigation task with goals correlated to a fixed position in the maze; and a car-racing game with highly-correlated action sequences across time.
2. We study uncertainty-based and loss-based active sampling techniques and find that active sampling alleviates causal confusion in offline training and yields a higher reward at evaluation.

3. We further show that active sampling is able to alleviate causal confusion at a significantly higher sample efficiency than naive uniform sampling, and that the usefulness of active sampling in alleviating causal confusion is highly related to the quality of predictive uncertainty estimates used in the best-performing, uncertainty-based acquisition function.

Our work suggests that in many cases we may not have saturated our datasets and can push the capabilities of offline reinforcement learning agents by optimising on the long tail of the data. We show that simply showing the same training dataset according to a better distribution to a model during training could be a viable strategy to explore before investing efforts in collecting more high-quality data or exploring larger models.

Our results demonstrate that while dominant spurious correlations in large offline datasets may lead to poor performance, and hence to harmful decision-making at deployment, active sampling techniques can significantly reduce the negative effects of spurious correlations on offline RL agents and that they are able to do so with high sample efficiency.

The study conducted in this chapter is located at the intersection between the fields of reinforcement learning and Bayesian active learning, which we will now describe in Sections 3.2 and 3.4.

3.2 Preliminaries

The study conducted in this chapter is located at the intersection between the fields of reinforcement learning and Bayesian active learning, which we described in Chapter 2.

Offline RL algorithms aim to learn a policy by estimating a policy-dependent action-value function from a static dataset, and iteratively improving the policy under conservative value estimates. This is learnt from a static dataset of transitions $\mathcal{D} = \{(\mathbf{s}, \mathbf{a}, r, \mathbf{s}')\}$ collected by a behaviour policy π_β . Since the agent does not interact with the environment, naively using online RL techniques in the offline case

leads to value function overestimation on unseen states and actions. This happens because bootstrapping on out-of-distribution actions in the bootstrapped target can propagate erroneously high values from the next state’s value estimate into the update. Thus, algorithms based on the *pessimism principle* of underestimating the Q -values to optimise the worst-case regret bound have been successful at learning policies from datasets containing either good coverage of the state–action space or high return trajectories [80, 79, 22, 12].

Prior work categorised pessimistic offline RL algorithms into (1) *proximal* and (2) *uncertainty-aware* algorithms [22]. The former penalises action-value estimates based on deviations from the actions seen in the dataset, while the latter conservatively updates the value functions, taking the uncertainty of their targets into account. Proximal pessimistic algorithms like CQL, BEAR [79, 80] are known to work well with exactly the kind of narrow and biased data distributions that are most prone to causal confusion [48]. We, therefore, analyse the proximal class of algorithms in this work and leave the study of causal confusion in uncertainty-aware pessimistic offline RL algorithms to future work.

Conservative Q -Learning. We choose CQL [80] as an instantiation of a proximal pessimistic offline RL algorithm in our experiments, due to its simplicity and strong performance on offline RL benchmarks.

The CQL critic objective combines a policy-dependent temporal-difference (TD) term with a conservatism penalty that constrains deviations from the behaviour policy. Formally, it is written as:

$$\begin{aligned} \mathcal{L}_{\text{critic}}^{\text{CQL}}(\theta) = & \frac{1}{2} \mathbb{E}_{(s,a,s') \sim \mathcal{D}} \left[\left(Q_{\theta}(s,a) - \mathcal{B}^{\pi_{\theta}} Q_{\bar{\theta}}(s,a) \right)^2 \right] \\ & + \alpha_0 \mathbb{E}_{s \sim \mathcal{D}} \left[\log \sum_a \exp Q_{\theta}(s,a) - \mathbb{E}_{a \sim \pi_{\beta}} [Q_{\theta}(s,a)] \right], \end{aligned} \quad (3.1)$$

where:

- α_0 controls the degree of conservatism.

University of Oxford

- π_θ is the learned policy associated with Q_θ .
- $\bar{\theta}$ denotes the parameters of a *target network*, which is a lagged copy of θ used to stabilise the bootstrapped TD updates.
- \mathcal{B}^{π_θ} is the *policy-dependent Bellman backup operator*,

$$(\mathcal{B}^{\pi_\theta} Q)(s, a) = r(s, a) + \gamma \mathbb{E}_{s' \sim \mathcal{P}} \left[\mathbb{E}_{a' \sim \pi_\theta(\cdot|s')} Q(s', a') \right].$$

This distinguishes it from the classical Bellman optimality operator that defines Q^* .

The squared term in Equation (3.1) corresponds to a TD error relative to the *policy-dependent target*,

$$\delta(s, a, s') = Q_\theta(s, a) - \mathcal{B}^{\pi_\theta} Q_{\bar{\theta}}(s, a),$$

which measures the inconsistency between the current critic and its conservative backup.

The second term encourages *conservatism* by penalising Q-values for out-of-distribution actions relative to the behaviour policy π_β observed in the dataset \mathcal{D} .

3.3 Related Work

Causal Confusion in Supervised Policy Learning. Several works in offline imitation learning have proposed solutions to mitigate causal confusion. de Haan et al. [38] demonstrated causal mis-identification in models trained on expert trajectories collected in the Mountain-Car and CARLA simulators [148, 40] where inputs were augmented with the previous control command taken by the acting agent. They proposed to resolve the confusion through a scheme to query an expert or collect additional rollouts in the environment to refine a learned causal graph that conditions a learned policy. Wen et al. [165] propose adversarial training to prune out any *known* sources of spurious correlations from the policy’s representation,

for instance, the previous control commands given to a robot; Wen et al. [166] propose re-weighting the losses of data points based on the loss of a model trained with just the spurious correlates as the input; OREO [117] regularises the model’s representation to be invariant to any individual object being dropped out in a scene. Lee et al. [87] propose training a diversified policy ensemble for imitation learning in the case when *perfect spurious correlations* exist in the data and later select from these hypotheses based on validation data. Causal Confusion has also recently been studied in reward-learning from preferences [155], where spurious correlations can be drawn between a human evaluator’s preferences and certain actions or parts of the state space, for tasks in the Assistive gym [44]. For instance, a reward model trained to classify or rank trajectories for a feeding task can easily learn that higher force correlates to higher reward since a certain level of force needs to be inserted in the direction of the mouth in all preferred trajectories.

Active Sampling. The analysis in this work is inspired by the approach presented in Jesson et al. [66], which considers the problem of treatment effect estimation in settings where we wish to be sample-efficient in terms of querying for outcomes of costly experiments. Jesson et al. [66] propose several causality-inspired acquisition functions that prefer data points that have both high variance in their estimated outcomes and that correspond to covariates with considerable overlap in the dataset. In Q -learning-based RL algorithms, Prioritised Experience Replay [133] is a non-Bayesian loss-based sampling scheme proposed for off-policy learning. It computes acquisition scores based on the TD-error of transitions and has not been studied in offline RL.

Ensemble Models in RL. Ensembles have been studied extensively to guide exploration in online RL [113, 86], and recently to construct adaptive pessimism constraints in offline RL, to disincentivise uncertain actions from having high estimated returns. It was recently also shown that significantly increasing the size and diversity of the ensembled critic in Soft-Actor-Critic [55] performs competitively with state-of-the-art offline RL algorithms [8]. We are not aware of any prior work

University of Oxford

that has explored how uncertainty about the value function can be used to sample state transitions in RL.

AI Alignment. AI alignment seeks to align the behavior of agents with the intentions of their creators by investigating the incentives behind demonstrated tasks. Recent work on *Goal Mis-generalisation* [84] explores how online RL agents in Procgen [33] can get confused about the goal they are pursuing since those goals co-occur with irrelevant artifacts in the environment most of the time. In this case, the specification is correct, but the agent still pursues an unintended objective (as opposed to poor reward definitions that predictably lead to reward hacking).

3.4 Alleviating Causal Confusion in Offline RL via Active Sampling

In this section, we provide a brief introduction to terminology in the field of causal inference and discuss how active sampling can be seen as facilitating efficient causal discovery on offline (observational) data. We follow this with relevant background on active sampling and describe the active sampling schemes we use in this work in conjunction with offline RL.

3.4.1 Conditional Average Treatment Effect Estimation from Causally Ambiguous Data

In this section, we discuss a causal interpretation of our approach which employs active sampling as a tool for causal discovery. We start by describing quantities of interest in the field of causal inference and their connection to Q -values (and derived quantities) used in RL.

Treatment-effect estimation, where the goal is to estimate the effect of a treatment $T \in \mathcal{T}$ on the outcome $Y \in \mathcal{Y}$ for individuals described by covariates $X \in \mathcal{X}$, is a central problem in causal inference. In particular, we may wish to estimate the expected difference in potential outcomes for individuals when subjected ($t = 1$)

University of Oxford

or not subjected ($t = 0$) to a treatment t , measured by the Conditional Average Treatment Effect (CATE, Abrevaya et al. [1]), defined as

$$\tau(X) \equiv \mathbb{E}[Y \mid x, t = 1] - \mathbb{E}[Y \mid x, t = 0]. \quad (3.2)$$

Here the treatment t is often considered to be a binary variable, but the definition can be extended to the multivariate case with continuous or discrete values. Realizations of the random variables X, T, Y are denoted by x, t, y , respectively. The set of assumptions needed to ensure identifiability of the CATE estimator are listed in [128].

To frame the problem of disambiguating the effect of different actions (i.e., treatments $t \in T$) in a given state (i.e., a set of covariates $x \in \mathcal{X}$), we frame estimation of the advantage functions $A(s, a) = Q(s, a) - V(s)$ in reinforcement learning as CATE estimation [115]. In particular, the outcome Y corresponds to the Q -function (i.e., the expected return) for a given state–action pair, and we can express the corresponding CATE estimator as

$$\begin{aligned} \mathbb{E} \left[\sum_{t=0}^{\infty} \gamma^t r(\mathbf{s}_t, \mathbf{a}_t) \mid s_t = s, a_t = a \right] - \mathbb{E} \left[\sum_{t=0}^{\infty} \gamma^t r(\mathbf{s}_t, \mathbf{a}_t) \mid s_t = s \right] & \quad (3.3) \\ & = Q(s, a) - V(s) = A(s, a), \quad (3.4) \end{aligned}$$

which indicates the advantage of executing an action a at state s as opposed to any other alternative action. With this connection established, we can now cast active sampling of data points to estimate a value or advantage function, as a sequential process of accurately estimating treatment effects.

3.4.2 Active Sampling

Active sampling techniques are used to selectively sample from a given dataset during training to enable sample-efficient learning or to improve learning from noisy data [93, 106]. To perform active sampling, data points are scored according on a given acquisition function, and a small number of points are sampled based on

these scores using a pre-specified weighting scheme. Similar to active learning, the acquisition functions are often information-theoretic quantities and seek to reduce the uncertainty about model parameters by acquiring informative data points. In this work, we build on Jesson et al. [66] and study whether active sampling can enable sample-efficient resolution of causal confusion in models trained on long-tailed datasets.

In particular, the focus of this work is to investigate whether active sampling techniques can alleviate the negative effects of causal ambiguity in state–action trajectories used in offline RL—without necessitating any modifications to the learning objective. Algorithm 1 describes the procedure for CQL, integrated with our proposed method for active sampling of state transitions. The modifications from uniform sampling are highlighted in blue. As we noted earlier, causal confusion in policy learning has been observed to occur due to insufficient exposure of a model to the rare scenarios that make up the long tail of the dataset it is trained on. These data points are unknown in advance, and we hypothesise that a good approach to finding them is to compute the epistemic uncertainty or the loss of the training model of each point in the dataset. We will thus define a set of loss-based and uncertainty-based acquisition functions, which we will use to perform active sampling, next.

Uncertainty-based Sampling. We aim to sample state transitions for which the learned Q -network is uncertain about the value of either the action observed in the dataset, $a_\beta \sim \pi_\beta(s)$, or the greedy action under the current policy, $a^* = \arg \max_a Q_\theta(s, a)$.

To model epistemic uncertainty in the learned Q -function, we construct an ensemble of M Q -networks $\{Q_{\theta^i}\}_{i=1}^M$, each trained independently on the same dataset \mathcal{D} and equipped with its own target network $\{\bar{\theta}^i\}_i$, following prior work (e.g., Ghasemipour et al. [50]). For a given state–action pair (s, a) , the TD target for ensemble member i is

$$y^i = r(s, a) + \gamma \mathbb{E}_{s' \sim \mathcal{P}} \left[\mathbb{E}_{a' \sim \pi_{\theta^i}(\cdot | s')} Q_{\bar{\theta}^i}(s', a') \right],$$

University of Oxford

so each member bootstraps its own policy-dependent estimate $Q^{\pi_{\theta^i}}$.

A naive variance over Q -values can be misleading due to noise introduced by iterative bootstrapping. Instead, we compute *advantage estimates* to obtain a policy-relative measure of uncertainty. For ensemble member i , the advantage is defined as

$$A^{\pi_{\theta^i}}(s, a) = Q_{\theta^i}(s, a) - V_{\theta^i}(s), \quad V_{\theta^i}(s) = \sum_a \pi_{\theta^i}(a | s) Q_{\theta^i}(s, a),$$

where $V_{\theta^i}(s)$ denotes the state-value under member i 's policy. In practice, π_{θ^i} is represented by a softmax over $Q_{\theta^i}(s, \cdot)$, yielding an approximation to the policy-dependent value function.

The advantage function represents a causal quantity that captures the relative *effect* of an action on the expected return at a given state. Our acquisition score for uncertainty-based sampling is therefore

$$\text{Var}_i[A^{\pi_{\theta^i}}(s, a)],$$

the variance of the advantage estimates across ensemble members.

We consider two variants:

- **Variance-greedy:** compute the variance for the greedy action, $a = a^*$.
- **Variance-data:** compute the variance for the behaviour policy action, $a = a_\beta$.

All ensemble members share the same architecture and ensemble size across all sampling schemes, including uniform sampling baselines, to ensure a fair comparison.

Loss-based Sampling. We sample transitions based on their Temporal Difference error similar to Prioritised Experience Replay (PER; Schaul et al. [133]) and refer to this variant as *TD-Error*.

We discuss a causal interpretation of our active sampling approach in Section 3.4.1 and explain how it relates to causal discovery of accurate treatment effects from offline data.

Algorithm 1 Conservative Q-Learning (+ active sampling)

- 1: Initialise ensemble Q -function Q_θ , n_{ep} =epochs, d_{sz} =dataset size, b_{sz} =batch size, T =steps-per-epoch.
 - 2: **for** epoch e in $\{1, \dots, n_{ep}\}$ **do**
 - 3: **for** step t in $\{1, \dots, T\}$ **do**
 - 4: compute scores acq_i over $\mathcal{D}_{train} = [s_i, a_i]_{i=1}^{d_{sz}}$ according to the acquisition functions in Section 3.4.2
 - 5: $acq_i = \frac{acq_i}{\sum_{j=1}^{d_{sz}} acq_j}$ (normalise scores)
 - 6: sample batch $B = [s_i, a_i, s'_i, r_i]_{i=1}^{b_{sz}}$ from \mathcal{D}_{train} according to a categorical distribution with probabilities acq
 - 7: Train the Q -function on D_{train} using objective from Equation (3.1)
 - 8:
 - 9: **end for**
 - 10: **end for**
-

3.4.3 Computing Acquisition Scores

In practice, computing the acquisition scores over all the transitions in the dataset can be both expensive and redundant since high-error or high-uncertainty points will likely stay high over a short window of subsequent gradient steps. Thus, we only recompute all the scores after every n gradient steps and vary n as a hyper-parameter in our experiments.

Later in Section 3.6.3, we will also explore a scheme where scores are recomputed only over sampled batches, as done in Prioritised Experience Replay (PER). This is an approximation where a priority queue is maintained and the priority of every data point is derived from the TD-Error computed on it. In this case, the priorities are only updated for a small subset of points at every gradient step, since the acquisition scores are only recomputed on the data points in the sampled batch, potentially leading to many points in the replay buffer with *stale* scores. We adopt the same implementation as that of PER with the scores for the priorities coming from the TD-error and Variance estimates. We include further details about this procedure in the appendix. We refer to the above two cases by appending *-dataset* and *-batch* to the names of the sampling schemes to indicate that scores are recomputed for the entire data versus just the sampled batch respectively.

3.5 Experiments

In this section, we first describe the experimental setup and evaluation protocol (in Section 3.5.1). Next, we introduce the different benchmark domains and present the results of the corresponding empirical evaluations (in Sections 3.6.1 to 3.6.4).

3.5.1 Experiment Design

We focus on investigating the following questions through our experiments: **(1)** Can causal confusion be consistently observed in offline RL agents when training on a fixed, pre-collected dataset of transitions with long-tailed coverage?

(2) Can active sampling based on a model’s predictive uncertainty, or its loss, help in alleviating the effects of causal confusion? **(3)** To what extent does the quality of predictive uncertainty estimates affect predictive performance and sample efficiency gains under uncertainty-based active sampling?

To answer these questions, we modify three reinforcement learning environments (MiniGrid, Atari, and Procgen) to either include a realistic spurious correlate in the observations or bias the sampling of episodes. The specific modifications ensure that a demonstration dataset collected in these environments produces causally-confused agents when trained with an offline RL algorithm with uniform sampling of the data. The modifications are described and motivated further in the following sections.

Prior work on causal confusion, largely in the domain of imitation learning, has used a training protocol where training is terminated once a model reaches a sufficiently low loss [38]. The last checkpoint is then evaluated in the environment to report the achieved reward. This procedure, however, does not capture the full picture and disregards certain practical considerations of training neural networks—especially in the offline RL setting—for the following two reasons:

1. This approach cannot be adopted to offline RL as offline RL training does not lead to monotonically increasing performance, and the Q-function starts to excessively optimise the conservatism penalty after a certain point in training.

Approaches to perform early stopping in offline RL recommend termination based on statistics independent of the loss curves [80, 3].

2. Scenarios occurring in the long tail of a demonstration dataset for a task will appear less frequently when episodes are sampled uniformly from the task environment for model evaluation. In this case, computing the validation loss over a uniformly sampled test set or average evaluation reward over uniformly sampled episodes will not accurately reflect the model’s progress in resolving causal confusion in this environment.

To address these issues, we adopt two measures: First, we evaluate the reward at the end of every epoch (as commonly done in offline RL) and continue training until the reward saturates. This allows us to measure both whether active sampling can recover the rewards achieved by uniform sampling in fewer steps and also whether it achieves a higher maximum reward. Second, we design the evaluation environment episodes in a way that allows isolating the causally confused behaviour and identifying whether causal confusion has been resolved. This amounts to doing a scenario-driven evaluation for the toy environments in which we know what the spurious correlate is by constructing all possible scenarios with and without the spurious correlate present. This process is described in more concrete terms for the Traffic-World MiniGrid environment in Section 3.6.1.

3.6 Evaluation Metrics

We present the reward curves directly in most of our experimental reporting. The curves are computed by taking the inter-quartile-mean (IQM) across seeds as proposed by Agarwal et al. [5]. We report the number of seeds used for all experiments in Appendix A.4.

Offline RL training performance is known to be non-monotonic, unlike supervised learning where the accuracy (or loss) increases (decreases) and then saturates. Often the reward curves start to decrease after a period when overfitting to the data-action values (through the conservatism penalty) starts to happen. Since we

define deterministic benchmark environments testing all the tail-scenarios in the case of Traffic-World and Maze experiments, we can consider any point on the curve where the model solves the highest number of environments (achieves the highest reward) as the point of convergence (as opposed to it being a noisy spike due to stochastic evaluation). This is similar to taking the max-reward checkpoint as done in [4]. However, to ensure that the solutions learned by any method are recoverable, we want to be able to get a good checkpoint from the model without needing to evaluate it too often. Thus we consider a method as having achieved higher reward than other methods at any point in time if it maintains this gap for at least two subsequent post-epoch evaluations. This is akin to taking a windowed-max over the reward curves.

We will now describe each benchmark environment and discuss the performance of the proposed active sampling schemes in their respective sections.

3.6.1 Illustrative Example: Traffic-World

The autonomous driving literature cites many examples where models trained on large datasets are very performant but exhibit causal confusion on the tail cases of their operational domain, for instance: (1) self-driving agents stopping at pedestrian crossings regardless of whether a pedestrian is present or not since the two often co-occur; (2) agents that simply try to *cruise* if they know their current speed since expert driving datasets contain cruising behaviour in a large fraction of each trajectory.

We build on the traffic gridworld environment proposed in [95] (shown in the left panel of Figure 3.1), where an agent (red triangle) starts at the leftmost point in a row behind randomly initialised leading vehicles (blue circles), and needs to cross a traffic light to reach a goal location (green square) on the right side of the grid. We collect data such that the probability of the traffic light turning red becomes lower as the agent approaches it, and so the data distribution contains: (1) mostly episodes where the light is green throughout the episode, (2) some episodes

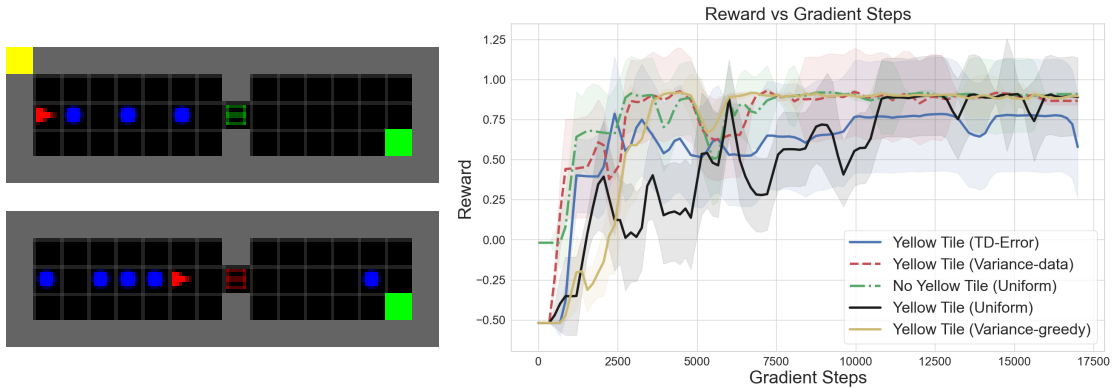


Figure 3.1: Traffic-World: *Left Top:* The leading vehicle is static causing the top-left tile to flash yellow. *Left Bottom:* The agent is in front of a red light, and the top-left tile is not yellow since the agent is not blocked by the leading vehicle. *Right:* Reward curves for agents trained on data not exhibiting causal ambiguity (green) and data exhibiting causal ambiguity (others), i.e. data without and with a spurious correlate respectively.

where the traffic light is red and the agent has to wait behind the vehicle in front (referenced here onward as the leading vehicle) before the light turns green again, and **(3)** only a couple of episodes where the light turns red with the agent at the front of the traffic queue.

In this setup, the agent could simply learn to follow the leading vehicle, instead of learning traffic light rules. To test causal confusion explicitly here, we introduce a related spurious correlate: a flashing yellow tile at the top left of the grid (emulating the brake lights on a leading vehicle), that is yellow whenever the leading vehicle is stopped or blocked, and grey otherwise. The agent could follow this as an indicator of whether to stop or go ahead, and this policy’s actions would be optimal for 98% of the data points.

We now define four instances of this environment that constitute our evaluation set, and we average the reward over these when reporting the results. The first three instances cover tail-cases, and the last one covers the majority case of scenarios that occur in our 7000-episode dataset:

1. **simple-green-with-tile:** The agent is unobstructed; the traffic light is green; the tile flashes yellow. This tests whether the agent relies on the tile to decide when to move.

2. **simple-red-no-tile**: The agent is unobstructed; the traffic light is red; the tile does not flash yellow. This tests whether the agent relies on the tile to decide when to stop.
3. **traffic-light-switches-with-tile**: The agent starts behind traffic; the traffic light switches to red for some time steps when the agent reaches it; the tile flashes according to the leading vehicle.
4. **always-green-with-tile**: This is a trivial episode with traffic where the agent can simply reach the goal without needing to navigate a red light since it is always green.

Figure 3.1 shows the evaluation curves of CQL agents trained with uniformly-sampled data, with and without the yellow tile present in images in the dataset. We see that the performance of the former agent degrades and it takes $4\times$ the number of gradient steps to converge to the solution of the latter agent which is trained without the spurious correlate present. Also shown are the active sampling variants (*TD-Error*, *Variance-greedy* and *Variance-data*) trained with the spurious yellow tile, which perform very similarly to uniform sampling without the spurious correlate present. From the plots, we see that for the TD-error variant there is high variance across seeds; thus the inter-quartile mean of the average rewards across seeds is lower than the maximum achievable reward, although good solutions were found for some random seeds very early in training. We also see that the Variance-based versions recover a good solution quickly.

3.6.2 Assessing Generalization in Offline Reinforcement Learning: Procgen

The Maze environment in Procgen [33] defines a navigation task where the agent starts at the bottom left in the maze and receives a reward of +10 upon successfully reaching the goal which is sampled at any valid location in the maze. Langosco et al. [84] recently showed that an agent trained on a series of environments with the goal always at the top-right will be causally confused about the source of the reward

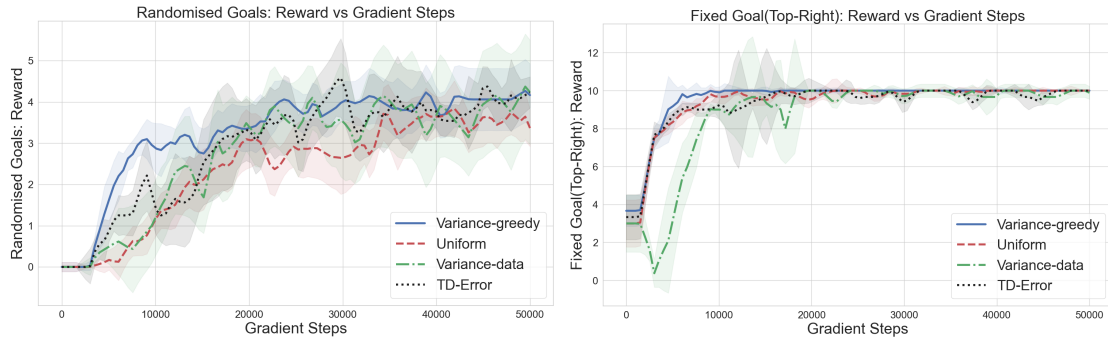


Figure 3.2: (Maze) Rewards curves of different agents trained on a skewed dataset containing 6000 episodes with a fixed goal at the top-right and 200 episodes with a randomly sampled goal in Maze. We see that the agents trained with uniform sampling and active sampling perform similarly on the fixed goal evaluation environment (right), but the active sampling variants achieve a higher reward in the environments with randomly sampled goals (left). This verifies that the model is not just performing well in one of the two kinds of environments and the reason behind the lower performance of uniform sampling, in this case, is causal confusion about the location of the goal.

and will still navigate to the top-right even when the goal is sampled elsewhere.

We generate a skewed *mixture* dataset containing mostly episodes where the goal is sampled at the top-right, and a few episodes where the goal is sampled randomly. Further details about the collection are described in the Appendix. Figure 3.2 shows the evaluation performance of random and active sampling agents trained on this *mixture* dataset. The left and right plots show the performance when the goals are sampled randomly and from the top-right in the evaluation environment, respectively. We observe that active sampling recovers the maximum performance achieved by uniform sampling in half the number of training steps. On this benchmarking task, the gains from active sampling are largely in terms of training efficiency since both active and uniform sampling variants saturate to similar final rewards in the environment with randomly sampled goals. We plot the computation time for the uniform and active sampling variants in Figure 3.5 in the appendix and note that variance-based sampling reaches the highest-score achieved by uniform sampling in lesser wall-clock time. Qualitative evaluations show that uniform sampling agents which achieve a lower reward still successfully navigate to the top-right corner of *hard* mazes, and are therefore only confused about the location of the goal.

It is important to note that if the evaluation environments considered here had

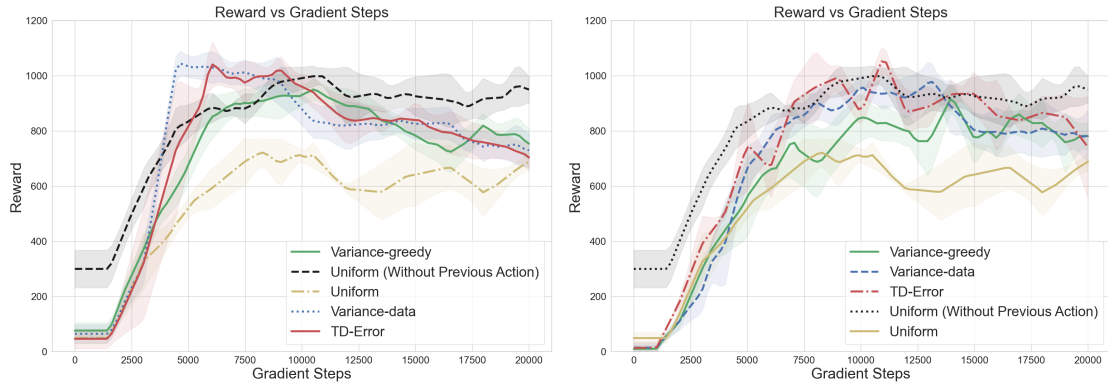


Figure 3.3: Comparison of active and uniform sampling agents on Enduro. On the left is the comparison in the *-dataset* case, and on the right is the case with *-batch* sampling where scores are incremented and updated based on the sampled batch. We see an increase in the number of gradient steps it takes the *-batch* case to reach the highest possible reward.

followed a distribution of scenarios similar to the training data distribution, then we would have observed saturation of the evaluation reward somewhere between 10,000–20,000 gradient steps. This is because the evaluation performance on the environments with the fixed goal, which occurs much more frequently, saturates at this point. An active sampling model on which early-stopping would be performed based on the validation loss, would then have fared much better on the tail case of randomly-sampled-goal episodes (and done similarly well otherwise), as compared to the model trained with uniform sampling of data. This shows why the experimental design choices we adopted in this work—scenario-driven evaluation and termination of training based on rewards—are crucial to appropriately evaluating causal confusion in offline RL agents.

3.6.3 Causal Confusion in the ALE Benchmark

We are now interested in evaluating our active sampling baselines on a larger pre-existing dataset with more realistic noise and variations. Prior work in imitation learning [117, 38] has attempted to simulate causal confusion in specific Atari game-play datasets by modifying images to display the previous action taken by the agent. This kind of causal confusion is inspired by robotics datasets that have trajectories with highly correlated (and thus predictable) sequences of actions, because embodiment dictates that an agent’s state does not change too drastically

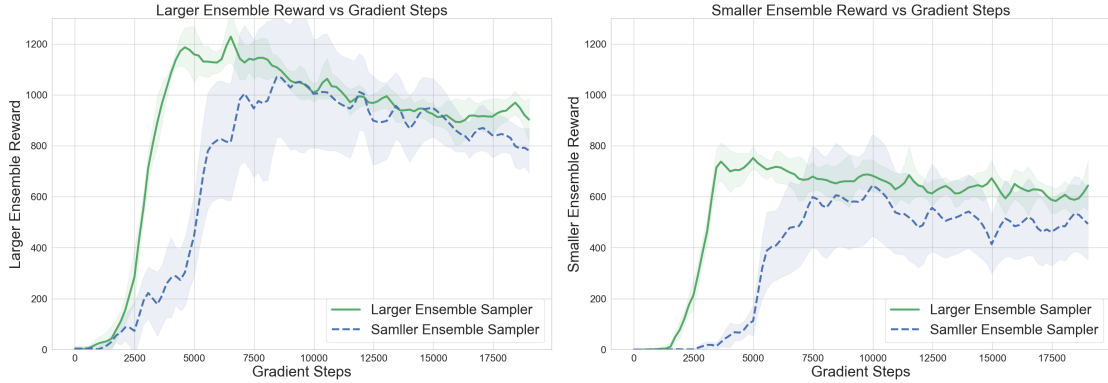


Figure 3.4: Reward curves of the larger (left) and smaller (right) ensemble training when sampling on the basis of the uncertainty of either one: of the speed of convergence depending on the quality of uncertainty estimation.

between subsequent timesteps.

Table 3.1: Sub-sampling experiment on the Enduro dataset.

Fraction of data sub-sampled	Reward (Uniform)	Reward (Variance-data)	Reward (Variance-greedy)	Reward (TD-Error)
0.5	470	780	560	560
0.7	570	850	810	820
0.9	660	1000	890	980

However, when we trained uniform sampling CQL agents with and without the previous action on the images for 20 Atari games, only a select few games actually exhibited causal confusion of this kind. We pick Enduro since we can consistently observe both a convergence speed degradation and final reward degradation upon adding the previous action to the image. Enduro is a car racing game where an agent needs to overtake cars (receiving a positive reward for each car overtaken) and drive along a winding road, with a limited number of collisions allowed.

Figure 3.3 shows our proposed active sampling baselines compared to uniform sampling in the following two cases that we previously discussed in Section 3.4.3: (1) *-dataset* case when we recompute the scores across the dataset every few gradient steps and (2) *-batch* case when the scores are only recomputed for the sampled batch and updated in a priority queue.

Also shown is the uniform sampling baseline trained without the previous action

displayed on the image (data without causal ambiguity). We note that all active sampling variants perform significantly better than their uniform sampling counterpart when both are trained with the spurious correlate (i.e., with the previous action displayed on the image). This discrepancy is observed in terms of convergence time to their best performance and in terms of the highest reward achieved, which is approximately 40 percent higher than the reward achieved by uniform sampling. While both *TD-Error* and *Variance*-based variants achieve a similar highest score, *TD-error* sampling takes twice the number of forward passes (and thus twice as long) to compute the acquisition score (i.e., the TD-error) in the *-dataset* case.

We further observe that when we go from the *-dataset* to the *-batch* setting (i.e., updating scores only for the sampled batch, Figure 3.3, right), the final reward achieved by the active variants is only slightly lower than previously. However, in this setting, all methods require twice the number of gradient steps to obtain this reward since the acquisition scores in the *-batch* setting are stale for many data points. A relevant line of future work will be to investigate how to maintain the sampling quality without needing to recompute all scores as in the *-dataset* case.

We also conducted an experiment where we uniformly sub-sample decreasing fractions of the original dataset and see that even the active sampling variants trained with 70% of the data outperform the uniform sampling variants trained on all the data. This suggests that naively acquiring more data is unlikely to resolve causal confusion if the data distribution remains skewed. We present these results in Table 3.1.

3.6.4 Effect of Predictive Uncertainty on Active Sampling and Causal Confusion

To better understand the improvement in convergence speed of active sampling vis-à-vis uniform sampling, we train CQL with a smaller ($n = 3$) and a larger ($n = 10$) ensemble of Q -networks, all else equal, and sample identical transitions based on the predictive uncertainty of only either the smaller or the larger ensemble.

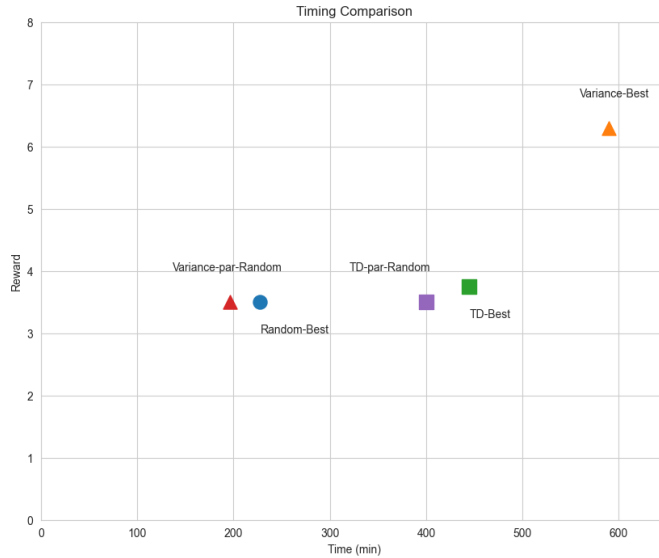


Figure 3.5: Timing Comparison for different sampling schemes on the Procgen-Maze benchmark plotted as reward achieved versus wallclock time in minutes.

This experiment setup allows us to isolate the effect of the predictive variance on active sampling and therefore on the observed reduction in causal confusion.

Figure 3.4 shows the training curves for both ensembles, and we find that when transitions are sampled according to the predictive uncertainty of the larger ensemble, even the smaller ensemble converges faster (albeit to a smaller value than that achieved by the larger ensemble). Similarly, when points are sampled according to the predictive uncertainty of the smaller ensemble, the larger ensemble converges to its highest reward more slowly. This result implies that improved predictive uncertainty estimation, for which we use the size of the ensemble as a proxy, improves the ability of active sampling to identify samples that break spurious correlations in the data and reduce causal confusion. We provide a further discussion of these observations in Appendix A.2.

3.7 Computational Cost

Figure 3.5 shows a scatter plot for the wall-clock times to achieve highest reward across different active and uniform baselines (labelled as TD-Best, Variance-Best and Random-Best) corresponding to the results for the Maze environment. It

also plots the time needed for active sampling variants to achieve the best reward that uniform sampling achieves (denoted as Variance-par-Random and TD-par-Random in Figure 3.5). We note that we use a very simple implementation of active sampling in our work which can be optimised significantly by paralleling the computation of acquisition scores over the dataset. This will make the trade-off between performance and training time even more favourable towards approaches that do active sampling.

3.8 Conclusion

In this chapter, we studied how to alleviate causal confusion in offline RL. We designed uncertainty- and loss-based data sampling baselines to selectively sample transitions for training, and found evidence that active sampling can recover a less causally-confused model in significantly fewer training steps as compared to uniform sampling. In future work, we hope to scale the analysis performed in this work to larger benchmark environments, with sources of noise in the reward and transitions, to further corroborate our findings. Such an analysis would help further distinguish the quality of solutions found through loss-based and uncertainty-based active sampling since noisy transitions can have *irreducible* loss. Additional promising avenues for future work include studying the usefulness of active sampling in the regime of offline-to-online RL-based fine-tuning, as well as extending the analysis presented in this paper to environments with continuous action spaces.

3.9 Discussion

We summarise the following key take-aways from our experimental results, which pave the way for follow-up work:

- **Long-tail leverage.** Variance-based replay surfaces counter-factual transitions sooner, erasing up to 40 % of the causal-confusion performance gap in TrafficWorld and Enduro while using $4\times$ fewer gradient steps.
- **Uncertainty quality matters.** A 10-member critic ensemble provides better

uncertainty, which in turn grants a smaller 3-member learner near-optimal sampling efficiency.

- **Sampling alone is insufficient.** If the dataset genuinely lacks counter-examples (e.g., all pedestrians cross only on green lights), no replay scheme can fix the policy—fresh or synthetic data are required. Our experiments purposefully included rare but causal-disambiguating events. In real-world robotics logs, however, those “tail” cases may be *absent*: robots seldom perform risky or failure actions, and human tele-operation traces are both costly and biased. Two complementary remedies emerge:

1. **Collect more data:** impractical for every long-tail edge case.
2. **Synthesise counter-factual data:** feasible if we can *control* powerful generative models.

Follow up: Controllable Generation of Useful Transitions Building on this work, we sought to further study cases where policy learning would benefit from directed generation of samples that were not present in the dataset and provided learning signal towards higher generalisation and robustness. We discussed in the previous section that when sampling is insufficient, one could collect more data or synthesise important counter-factual points/tail cases. Our first attempt pursued the second remedy. We asked: *Can a vision-conditional diffusion model hallucinate minority transitions that “break” spurious correlations in an offline RL dataset?* Preliminary results were mixed: while diffusion models captured scene diversity, aligning rewards and state transitions proved brittle, especially for high-dimensional robotics observations. Reward-consistent tuple generation remained an open challenge.

Using Representations from Visual Generative Models Although our early attempts to generate minority-case data with diffusion models faltered—reward consistency proved brittle—the failure analysis revealed a silver lining: the models’ intermediate features encode far richer spatial semantics than CLIP-style contrastive encoders. This insight drives the pivot in Chapter 4, which repurposes text-to-image

backbones as control representations rather than data sources, shifting the focus from **how to sample** data to **how to exploit** the knowledge already embedded in large generative models and thereby enabling policy learning that is both data-efficient and semantically grounded.

Chapter 4 therefore shifts focus from *where* data comes from to *how* to reuse the vast knowledge already embedded in generative models of vision. Concretely, we will see how **Stable Control Representations** re-purpose the encoder backbone of the Stable Diffusion model for control.

The journey from active sampling to foundation-model representation use exemplifies a broader thesis theme—**efficiency through smarter use of what we already have**, whether that is logged data (Chapter 3), pre-trained weights (Chapter 4), or long-term agent memories (Chapter 5).

4 | Leveraging Existing Knowledge for Sample-Efficient Policy Learning

Contents

4.1	Introduction	52
4.2	Related Work	54
4.3	Background	55
4.3.1	Diffusion Models	55
4.3.2	Representation Learning for Visual Control	56
4.3.3	Policy-learning notation	57
4.4	Stable Control Representations	57
4.4.1	Layer Selection and Aggregation	58
4.4.2	Diffusion Timestep Selection	59
4.4.3	Prompt Specification	60
4.4.4	Intermediate Attention Map Selection	60
4.4.5	Using Text-to-Image Diffusion Model Representations to Learn Control Policies	60
4.4.6	Fine-Tuning on General Robotics Datasets	61
4.5	Empirical Evaluation	61
4.5.1	Baselines	62
4.5.2	Few-shot Imitation Learning	63
4.5.3	Image-Goal Navigation	64
4.5.4	Open Vocabulary Mobile Manipulation	65
4.5.5	Fine-tuning CLIP	67
4.5.6	Referring Expressions Grounding	67
4.5.7	Grasp Affordance Prediction	69
4.5.8	Comparison with LIV	69
4.5.9	Overall Ranking of Representations	70
4.6	Deconstructing Stable Control Representations	70
4.6.1	Layer Selection	71

4.6.2	Sensitivity to the Noising Timestep	73
4.6.3	How is Language Guiding the Representations?	73
4.6.4	The Effect of Spatial Aggregation	75
4.7	Discussion	76
4.8	Conclusion	77

4.1 Introduction

As general-purpose, pre-trained “foundation” models [7, 20, 29, 92, 111, 126, 156] are becoming widely available, a central question in the field of embodied AI has emerged: How can foundation models be used to construct model representations that improve generalization in challenging robotic control tasks [18, 140, 191]?

Robotic control tasks often employ pixel-based visual inputs paired with a language-based goal specification, making vision-language model representations particularly well-suited for this setting. However, while vision-language representations obtained via Contrastive Language-Image Pre-training [CLIP; 121]—a state-of-the-art method—have been successfully applied to a broad range of computer vision tasks, the use of CLIP representations has been shown to lead to poor downstream performance for robotic control. This shortcoming has prompted the development of alternative, control-specific representations for embodied AI [97, 108] but has left other sources of general-purpose pre-trained vision-language representations—such as text-to-image diffusion models—largely unexplored for control applications.

In this chapter, we present **Stable Control Representations (SCR)**: pre-trained vision-language representations from text-to-image diffusion models that can capture both high and low-level details of a scene [60, 126]. While diffusion representations have seen success in downstream vision-language tasks, for example, in semantic segmentation [14, 154, 162], they have—to date—not been used for control. We perform a careful empirical analysis in which we deconstruct pre-trained text-to-image diffusion model representations to understand the impact of different design

decisions.

In our investigation, we find that diffusion representations can outperform general-purpose models like CLIP [121] across a wide variety of embodied control tasks despite not being trained for representation learning. This is the case for purely vision-based tasks as well as for settings that require task understanding through text prompts. A highlight of our results is the finding that diffusion model representations enable better generalization to unseen object categories in a challenging open-vocabulary navigation benchmark [180] and provide improved interpretability through attention maps [151].

Our key contributions are as follows:

1. In section 4.4, we introduce a multi-step approach for extracting vision-language representations for control from text-to-image diffusion models. We show that these representations are capable of capturing both the abstract high-level and fundamental low-level details of a scene, offering an alternative to models trained specifically for representation learning.
2. In section 4.5, we evaluate the representation learning capabilities of diffusion models on a broad range of embodied control tasks, ranging from purely vision-based tasks to problems that require an understanding of tasks through text prompts, thereby showcasing the versatility of diffusion model representations.
3. In section 4.6, we systematically deconstruct the key features of diffusion model representations for control, elucidating different aspects of the representation design space, such as the input selection, the aggregation of intermediate features, and the impact of fine-tuning on performance.

We have demonstrated that diffusion models learn versatile representations for control and can help drive progress in embodied AI. Figure 4.1 presents a summary of our approach and results.¹

¹Code link: <https://github.com/ykarmesh/stable-control-representations>.

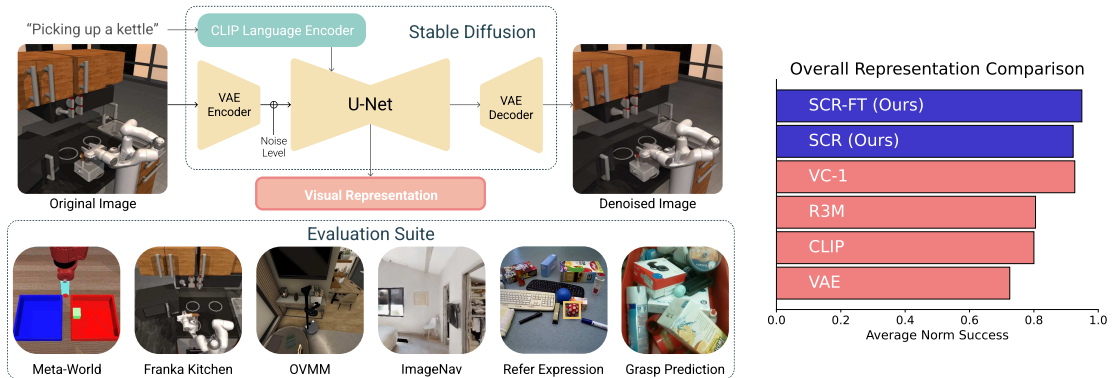


Figure 4.1: Left: Our paper proposes Stable Control Representations, which uses pre-trained text-to-image diffusion models as a source of language-guided visual representations for downstream policy learning. **Right:** Stable Control Representations enable learning control policies that achieve all-round competitive performance on a wide range of embodied control tasks, including in domains that require open-vocabulary generalization. Empirical results are provided in section 4.5.

4.2 Related Work

We begin with a review of prior work on representation learning and diffusion models for control.

Representation Learning with Diffusion Models. Diffusion models have received a lot of recent attention as flexible representation learners for computer vision tasks of varying granularity—ranging from key point detection and segmentation [154, 162] to image classification [157, 179]. Wang et al. [162] has shown that intermediate layers of a text-to-image diffusion model encode semantics and depth maps that are recoverable by training probes. These approaches similarly extract representations by considering a moderately noised input, and find that the choice of timestep can vary based on the granularity of prediction required for the task. Yang and Wang [179] train a policy to select an optimal diffusion timestep, we simply used a fixed timestep per class of task. Several works [151, 154, 162] observe that the cross-attention layers that attend over the text and image embeddings encode a lot of the spatial layout associated with an image and therefore focus their method around tuning, post-processing, or extracting information embedded within these layers.

Visual Representation Learning for Control. Over the past decade, pre-

trained representation learning approaches have been scaled for visual discrimination tasks first, and control tasks more recently. Contrastively pre-trained CLIP [121] representations were employed for embodied navigation tasks by EmbCLIP [72]. MAE representations have been used in control tasks by prior works like VC-1 [100], MVP [172] and OVRL-v2 [177]. R3M [108] and Voltron [70] leverage language supervision to learn visual representations. In contrast, we investigate if powerful text-to-image diffusion models trained for image generation can provide effective representations for control.

Diffusion Models for Control. Diffusion models have seen a wide range of uses in control aside from learning representations. These can broadly be categorized into three areas. First, diffusion models have been used as a class of expressive models for learning action distributions for policies [31, 57, 119]; They can improve model multimodality and generate richer action distributions than Gaussians. Second, off-the-shelf diffusion models have been used to augment limited robot demonstration datasets by specifying randomizations for object categories seen in the data through inpainting [68, 102, 182]. Third, planning can be cast as sequence modeling through diffusion models [6, 42, 65].

4.3 Background

This work builds on two threads of prior art: (i) diffusion models for text-conditional image generation, and (ii) representation learning for visual control. We review both in turn and close with notation for the policy-learning tasks considered in our experiments.

4.3.1 Diffusion Models

Denoising objective. Let $p(\mathbf{x}_0)$ denote the data distribution over images $\mathbf{x}_0 \in \mathbb{R}^{H \times W \times 3}$, and let $\{\sigma_t\}_{t=1}^T$ be a predefined noise schedule. A diffusion model learns a

University of Oxford

denoiser $\varepsilon_\theta(\mathbf{x}_t, t)$ by minimising

$$\mathcal{L}_{\text{DM}}(\theta) = \mathbb{E}_{\mathbf{x}_0 \sim p, \epsilon \sim \mathcal{N}(0,1), t \sim \mathcal{U}\{1, T\}} \left[\left\| \epsilon - \varepsilon_\theta(\mathbf{x}_0 + \sigma_t \epsilon, t) \right\|_2^2 \right]. \quad (4.1)$$

At inference the process is reversed: one samples $\mathbf{x}_T \sim \mathcal{N}(0, \sigma_T^2 I)$ and iteratively denoises through $p_\theta(\mathbf{x}_{t-1} | \mathbf{x}_t)$, obtaining a high-fidelity \mathbf{x}_0 after T steps.

Conditional generation. To draw from a conditional distribution $p(\mathbf{x}_0 | c)$ (e.g. images that match a text prompt c), the denoiser is augmented to $\varepsilon_\theta(\mathbf{x}_t, t, c)$ and trained jointly on (\mathbf{x}_0, c) pairs. Classifier-free guidance [59] mixes conditional and unconditional score estimates during sampling, improving visual fidelity without retraining.

Latent diffusion. High-resolution diffusion can be accelerated by operating in the latent space of a pre-trained autoencoder $\mathcal{E} : \mathbf{x} \mapsto \mathbf{z}$, $\mathcal{D} : \mathbf{z} \mapsto \mathbf{x}$ [126]. Equation (4.1) is applied to $\mathbf{z}_0 = \mathcal{E}(\mathbf{x}_0)$, giving

$$\mathcal{L}_{\text{LDM}}(\theta) = \mathbb{E} \left[\left\| \epsilon - \varepsilon_\theta(\mathbf{z}_0 + \sigma_t \epsilon, t, c) \right\|_2^2 \right], \quad (4.2)$$

after which a single decoder pass $\mathbf{x}_0 = \mathcal{D}(\mathbf{z}_0)$ recovers the image.

Stable Diffusion (SD) is a prominent example: it couples a U-Net denoiser with a CLIP text encoder and achieves sub-second 512² image generation on commodity GPUs.

4.3.2 Representation Learning for Visual Control

Control agents that perceive the world through raw pixels must compress high-dimensional observations into task-relevant features before acting. *Representation learning for control* therefore seeks an encoder f_ϕ that maps an image (and, where applicable, a language task specification) to a compact latent $\mathbf{h} = f_\phi(s_{\text{image}}, s_{\text{text}})$, such that downstream policy learning becomes data-efficient and robust.

Evaluation proceeds along two axes:

- **Few-shot imitation.** A policy head $\pi_\psi(a \mid \mathbf{h})$ is trained by behaviour cloning on tens to hundreds of expert demonstrations; success is measured by task completion rate from unseen initial states.
- **Reinforcement learning fine-tuning.** When demonstrations are absent or sparse, the same head is trained with an RL algorithm—here, a clipped PPO variant—and performance is tracked as a function of environment interactions.

Historically, control-oriented encoders were trained from scratch on the task at hand or pre-trained contrastively (e.g. CLIP). Our work explores an alternative: *reuse the spatially precise representations internal to text-to-image diffusion models*, thereby sidestepping the need to synthesise new images or to rely on purely semantic CLIP features.

4.3.3 Policy-learning notation

We model each environment as an MDP $M = (\mathcal{S}, \mathcal{A}, P, R, \gamma)$ with image-language states $s = [s_{\text{image}}, s_{\text{text}}]$. The control pipeline is

$$s \longrightarrow \mathbf{h} = f_\phi(s_{\text{image}}, s_{\text{text}}) \longrightarrow a \sim \pi_\psi(a \mid \mathbf{h}),$$

where the encoder parameters ϕ are *frozen* (pre-trained) and only the policy head ψ is task-specific. We train ψ either by supervised behaviour cloning on demonstration datasets or by PPO on interactive roll-outs, depending on the task.

4.4 Stable Control Representations

In this work, we investigate the use of language-guided visual representations from the open-source Stable Diffusion model (v1.5) and their application to language-conditioned visual control tasks. We present background on latent diffusion models and text-to-image diffusion models, along with the notation we adopt in this work, in Section 4.3.

To extract representations, we follow a similar protocol as Wang et al. [162], Traub

[157], and Yang and Wang [179]: Given an image-text prompt, $s = \{s_{\text{image}}, s_{\text{text}}\}$, associated with a particular task, we use the SD VQ-VAE model as the encoder $\mathcal{E}(\cdot)$ and partially noise the encoded latents $\mathbf{z}_0 \doteq \mathcal{E}(s_{\text{image}})$ to some diffusion timestep t , to obtain the noised latent z_t . We then extract a representation composed of the intermediate layer outputs of the U-Net ϵ_θ as it produces a denoising estimate $\epsilon_\theta(\mathbf{z}_t, t, s_{\text{text}})$. This process is illustrated in Figure 4.2. We refer to the extracted representations as **Stable Control Representations (SCR)**. In Sections 4.4.1, 4.4.2, 4.4.3, and 4.4.4, we describe the design space for extracting SCR, and in Sections 4.4.5 and 4.4.6, we explain how we use the representations for control.

4.4.1 Layer Selection and Aggregation

We are interested in evaluating the internal representations from the denoiser network, that is, the U-Net $\epsilon_\theta(\cdot)$. The first design choice we consider is which layers of ϵ_θ to aggregate intermediate outputs from. The U-Net does not have a representational bottleneck, and different layers potentially encode different levels of detail. Trading off size with fidelity, we concatenate the feature maps output from the mid and down-sampling blocks to construct the representation. This results in a representation size comparable to that of the other pre-trained models we study in section 4.5. This is shown at the bottom of Figure 4.2 and we ablate this choice in Section 4.6.1. Since outputs from different layers may have different spatial dimensions, we bilinearly interpolate them so that they are of a common spatial dimension and can be stacked together. We then pass them through a learnable convolutional layer to reduce the channel dimension before feeding them to downstream policies. The method used to spatially aggregate pre-trained representations can significantly affect their efficacy in downstream tasks, as we will discuss in Section 4.6.4. We use the best-performing spatial aggregation method for all the baselines that we re-train in section 4.5.

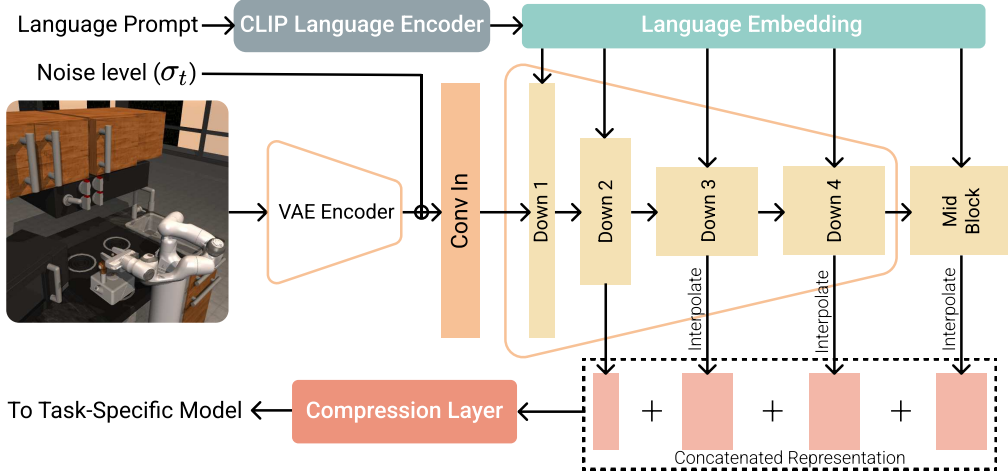


Figure 4.2: Extraction of Stable Control Representations from Stable Diffusion. Given an image-text prompt, $s = \{s_{\text{image}}, s_{\text{text}}\}$, we encode and noise the image and feed it into the U-Net together with the language prompt. We then aggregate feature maps from multiple layers within the U-Net, as described in section 4.4. Shown here are features from the mid and downsampling blocks of the U-Net.

4.4.2 Diffusion Timestep Selection

Next, we consider the choice of extraction timestep t for the denoising network (shown on the left of Figure 4.2). Recall that the images we observe in control tasks are un-noised (i.e., corresponding to \mathbf{x}_0), whereas the SD U-Net expects noised latents, corresponding to \mathbf{z}_t for $t \in [0, 1000]$. The choice of timestep t influences the fidelity of the encoded latents since a higher value means more noising of the inputs. Yang and Wang [179] have observed that there are task-dependent optimal timesteps and proposed adaptive selection of t during training, while Xu et al. [175] have used $t = 0$ to extract representations using un-noised inputs to do open-vocabulary segmentation. We hypothesize that control tasks that require a detailed spatial scene understanding would benefit from a lower diffusion timestep, corresponding to a later stage in the denoising process where the inputs have less noise. We provide evidence consistent with this hypothesis in Section 4.6.2. To illustrate the effect of the timestep, we display final denoised images for various t values in different domains in Figure B.1.

4.4.3 Prompt Specification

Since text-to-image diffusion models allow conditioning on text, we investigate if we can influence the representations to be more task-specific via this conditioning mechanism. For tasks that come with a text specifier, for example, the sentence “go to object X”, we simply encode this string and pass it to the U-Net. However, some tasks are purely vision-based and in these settings, we explore whether constructing reasonable text prompts affects downstream policy learning when using the U-Net’s language-guided visual representations. We present this analysis in Section 4.6.3.

4.4.4 Intermediate Attention Map Selection

Wang et al. [162] and Tang et al. [151] demonstrate that the Stable Diffusion model generates localized attention maps aligned with text during the combined processing of vision and language modalities. Wang et al. [162] leveraged these word-level attention maps to perform open-domain semantic segmentation. We hypothesize that these maps can also help downstream control policies to generalize to an open vocabulary of object categories by providing helpful intermediate outputs that are category-agnostic. Following Tang et al. [151], we extract the cross-attention maps between the visual features and the CLIP text embeddings within the U-Net. We test our hypothesis on an open-domain navigation task in Section 4.5.4, where we fuse the cross-attention maps with the extracted feature maps from the U-Net. We refer to this attention-map-augmented representation as **SCR-attn**.

4.4.5 Using Text-to-Image Diffusion Model Representations to Learn Control Policies

To solve visual control tasks with states given by $s = [s_{\text{image}}, s_{\text{text}}]$, where s_{text} may be used to specify the task, we wish to use pre-trained vision-language representations capable of encoding the state s as $f_{\phi}(s_{\text{image}}, s_{\text{text}})$. This encoded state is then supplied to a downstream, task-specific policy network, which is trained to predict the action a_t . Our evaluation encompasses both supervised learning and reinforcement learning

regimes for training the downstream policies. We train agents through behavior cloning on a small set of demonstrations for the few-shot manipulation tasks we study in Section 4.5.2. For the indoor navigation tasks we study in Sections 4.5.3 and 4.5.4, we use a version of the Proximal Policy Optimization [PPO, 136] algorithm for reinforcement learning.

4.4.6 Fine-Tuning on General Robotics Datasets

Finally, we consider fine-tuning strategies to better align the base Stable Diffusion model towards generating representations for control. This serves to bridge the domain gap between the diffusion model’s training data (e.g., LAION images) and robotics datasets’ visual inputs (e.g., egocentric tabletop views in manipulation tasks or indoor settings for navigation). Crucially, we do not require any task-specific losses for fine-tuning. Instead, we adopt the same text-conditioned generation objective as that of the base model for the fine-tuning phase. We use a small subset of the collection of datasets used by prior works on representation learning for embodied AI [100, 172]: we use subsets of the EpicKitchens [37], Something-Something-v2 [SS-v2; 52], and Bridge-v2 [160] datasets. As is standard, we fine-tune the denoiser U-Net ϵ_θ but not the VAE encoder or decoder. Image-text pairs are uniformly sampled from the video-text pairs present in these datasets. A possible limitation of this strategy is that text-video aligned pairs (a sequence of frames that correspond to a single language instruction) may define a many-to-one relation for image-text pairs. However, as we see in experiments in which we compare to the base Stable Diffusion model in section 4.5, this simple approach to robotics alignment is useful in most cases. Further details related to fine-tuning are provided in Appendix B.2.1. We refer to the representations from this fine-tuned model as **SCR-ft**.

4.5 Empirical Evaluation

In this work, we evaluate Stable Control Representations (SCR) on an extensive suite of tasks from 6 benchmarks covering few-shot imitation learning for manipulation

in Section 4.5.2, reinforcement learning-based indoor navigation in Sections 4.5.3 and 4.5.4, and tasks related to fine-grained visual prediction in Sections 4.5.6 and 4.5.7. Together, these tasks allow us to comprehensively evaluate whether our extracted representations can encode both high and low-level semantic understanding of a scene to aid downstream policy learning. We describe the common baselines used across tasks in section 4.5.1, and present the individual task setups and results in the following subsections.

4.5.1 Baselines

We compare SCR and its variants (i.e., SCR-FT and SCR-FT-ATTN) to the following prior work in representation learning for control:

1. **R3M** [108] pre-trains a ResNet50 encoder on video-language pairs from the Ego4D dataset using time-contrastive video-language alignment learning.
2. **MVP** [172] and **VC-1** [100] both pre-train ViT-B/L models with the masked auto-encoding (MAE) objective on egocentric data from Ego4D, Epic-Kitchens, SS-v2, and ImageNet, with VC-1 additionally pre-training on indoor navigation videos.
3. **CLIP** [121] trains text and ViT-based image encoders using contrastive learning on web-scale data.
4. **Voltron** [70] is a language-driven representation learning method that involves pre-training a ViT-B using MAE and video-captioning objectives on aligned text-video pairs from SS-v2.
5. **SD-VAE** [126] is the base VAE encoder used by SD to encode images into latents.

To assess how well the vision-only methods would do on tasks with language specification, we concatenate their visual representations with the CLIP text embeddings of the language prompts. While we are limited by the architecture designs of the released models we are studying, to ensure a more fair comparison we



Figure 4.3: Sample scenes from the Habitat environments for the ImageNav (left) and OVMM (center) tasks. Instances from training and validation datasets of the OVMM object set are shown on the right.

try to match parameter counts as much as we can. We use the ViT-Large (307M parameters) versions of CLIP, MVP, and VC-1 since extracting SCR involves a forward pass through 400M parameters.

4.5.2 Few-shot Imitation Learning

We start by evaluating SCR on commonly studied representation learning benchmarks in few-shot imitation learning. Specifically, our investigation incorporates five commonly studied tasks from Meta-World [181] (same as CORTEXBENCH [100]), which includes bin picking, assembly, pick-place, drawer opening, and hammer usage; as well as five tasks from the Franka-Kitchen environments included in the RoboHive suite [82], which entail tasks such as turning a knob or opening a door. We adhere to the training and evaluation protocols adopted in their respective prior works to ensure our results are directly comparable (detailed further in Appendix B.1.1).

Results. We report the best results of SCR and baselines in Table 4.1a. On Meta-World, we see that SCR outperforms most prior works, achieving 94.9% success rate. In comparison, VC-1, the visual foundation model for embodied AI and CLIP achieved 92.3 and 90.1% respectively. On Franka-Kitchen, SCR obtains 49.9% success rate, which is much higher than CLIP (36.3%) and again outperforms all other baselines except for R3M. We note that R3M’s sparse representations excel in few-shot manipulation with limited demos but struggle to transfer beyond this setting [100, 70]. We see that while the SD-VAE encoder performs competitively on Franka-Kitchen, it achieves a low success rate on Meta-World. This observation

Table 4.1: Average Success Rate and standard error evaluated across different representations.

(a) Meta-World & Franka-Kitchen.			(b) ImageNav	
Model	Meta-World	Franka-Kitchen	Model	Success
R3M	96.0 ± 1.1	57.6 ± 3.3	R3M	30.6
CLIP	90.1 ± 3.6	36.3 ± 3.2	CLIP-B	52.2
VC-1	92.3 ± 2.5	47.5 ± 3.4	VC-1	70.3
Voltron	72.5 ± 5.2	33.5 ± 3.2	MVP	68.1
SD-VAE	75.5 ± 5.2	43.7 ± 3.1	SD-VAE	46.6
SCR	94.4 ± 1.9	45.0 ± 3.3	SCR	73.9
SCR-FT	94.9 ± 2.0	49.9 ± 3.4	SCR-FT	69.5

(c) OVMM	
Model	Success
Oracle	77.6
Detic	36.7
CLIP	38.7 ± 1.7
VC-1	40.6 ± 2.2
SCR	38.7 ± 1.2
SCR-FT	41.9 ± 1.0
SCR-FT-ATTN	43.6 ± 2.1

allows us to gauge the improved performance of SCR from the base performance gain we may get just from operating in the latent space of this VAE. Additionally, we see that the task-agnostic fine-tuning gives SCR-FT an advantage (4%) over SCR on Franka-Kitchen while making no difference on Meta-World. Note that the other high-performing baselines (R3M and Voltron) have been developed for downstream control usage with training objectives that take temporal information into account, while VC-1 has been trained on a diverse curation of robotics-relevant data. In this context, SCR’s comparable performance shows that generative foundation models hold promise for providing useful representations for control, even with relatively minimal fine-tuning on non-task-specific data.

4.5.3 Image-Goal Navigation

We now assess SCR in more realistic visual environments, surpassing the simple table-top scenes in manipulation benchmarks. In these complex settings, the

representations derived from pre-trained foundational models are particularly effective, benefiting from their large-scale training. We study Image-Goal Navigation (ImageNav), an indoor visual navigation task that evaluates an agent’s ability to navigate to the viewpoint of a provided goal image [190]. The position reached by the agent must be within a 1-meter distance from the goal image’s camera position. This requires the ability to differentiate between nearby or similar-looking views within a home environment. This task, along with the semantic object navigation task that we study in Section 4.5.4, allows for a comprehensive evaluation of a representation’s ability to code both semantic and visual appearance-related features in completely novel evaluation environments. We follow the protocol for the ImageNav task used by Majumdar et al. [100] and input the pre-trained representations to an LSTM-based policy trained with DD-PPO [168] for 500 million steps on 16 A40 GPUs (further details in Appendix B.1.3). Given the large compute requirements for training, we directly compare SCR and SCR-FT to the results provided in Majumdar et al. [100].

Results. We evaluate our agent on 4200 episodes in 14 held-out scenes from the Gibson dataset and report the success rate in Table 4.1b. We find that SCR outperforms all other representations, while the fine-tuned version SCR-FT is almost on par with the second-best-performing VC-1 (69.5% vs 70.3%), the SOTA visual representation from prior work. This can be expected given that it was fine-tuned on images solely from table-top manipulation datasets. We also see that R3M, the best model for few-shot manipulation from Table 4.1a performs very poorly (30.6%) in this domain, showing its limited transferability to navigation tasks.

4.5.4 Open Vocabulary Mobile Manipulation

We now shift our focus to evaluating how Stable Diffusion’s web-scale training can enhance policy learning in open-ended domains. We consider the Open Vocabulary Mobile Manipulation (OVMM) benchmark [180] that requires an agent to find, pick up, and place objects in unfamiliar environments. One of the primary challenges here is locating previously unseen object categories in novel scenes

(illustrated in Figure 4.3 (left)). To manage this complex sparse-reward task, existing solutions [180] divide the problem into sub-tasks and design modular pipelines that use open-vocabulary object detectors such as Detic [188] to enable generalization to novel objects. We study a modified version of the Gaze sub-task (detailed in Appendix B.1.2), which focuses on locating a specified object category for an abstracted grasping action.

The task’s success is measured by the agent’s ability to precisely focus on the target object category. This category is provided as an input to the policy through its CLIP text encoder embedding. The evaluation environments cover both novel instances of object categories seen during policy learning, as well as entirely unseen categories. We compare to VC-1, the best model from Section 4.5.3 and CLIP, since prior work has studied it for open-vocab navigation [72, 99]. We also incorporate a baseline that trains a policy with access to ground truth object masks, evaluated using either the ground truth or Detic-generated masks at test time (labeled as Oracle/Detic).

Results. Table 4.1c shows that SCR-FT surpasses VC-1 by 1.3%, beating CLIP and SCR by 3.2%. It is surprising that VC-1’s visual representation does better than CLIP’s image encoder representation, given that the downstream policy has to use these with the CLIP text encoder’s embedding of the target object category. Comparing these to SCR-FT-ATTN, we can see the benefit of providing intermediate outputs in the form of text-aligned attention maps to the downstream policy (+1.7%). Samples of attention maps overlaid on images from an evaluation episode can be found in Appendix B.1. These word-level cross-attention maps simultaneously improve policy performance and also aid explainability, allowing us to diagnose successes and failures. Interestingly, the foundation model representations (CLIP, VC-1, SCR) perform better than Detic. While object detections serve as a category-agnostic input for downstream pick-and-place policies, noisy detections can often lead to degraded downstream performance, as we see in this case. Nonetheless, there is still a sizeable gap to ‘Oracle’ which benefits from ground truth object masks at test-time.

4.5.5 Fine-tuning CLIP

We follow the same experimental constraints that we took into account while fine-tuning the diffusion model to get SCR-FT: we trained it on the same text-image pairs from the same datasets and used CLIP’s contrastive loss to bring the visual embedding of the middle frames of a video

closer to the video caption’s text embedding. Specifically, for our experiment, we use the huggingface CLIP finetuning implementation and train the model with a batch size of 384 (the maximum number of samples we were able to fit on 8 A40 GPUs) with a learning rate of 5e-5 and a weight decay of 0.001 for 5000 update steps (same as SR-FT). We present the results in Table 4.2 for Franka-Kitchen, and note the lack of improvement on the task post-fine-tuning.

Table 4.2: Performance on Franka-Kitchen after fine-tuning CLIP.

Model	Franka-Kitchen
CLIP	36.9 \pm 3.2
CLIP (FT)	34.2 \pm 2.9

4.5.6 Referring Expressions Grounding

In Sections 4.5.2 to 4.5.4, our analysis focused on the performance of various representations across an array of control tasks. We now turn our attention to two downstream tasks involving fine-grained visual prediction. The first task, Referring Expressions Grounding, is detailed within this section, while the second task, Grasp Affordance Prediction, is discussed in Section 4.5.7. Karamcheti et al. [70] have previously examined the performance on these tasks as proxy measures to evaluate the efficacy of representations for control applications.

The **Referring Expressions Grounding** task requires the identification and bounding box prediction of an object in an image based on its textual description. Similar to Karamcheti et al. [70], we use the OCID-Ref Dataset [163] for our experiments. We show a sample image-text pair from the dataset to showcase the complexity of the task in Figure 4.4. The frozen visual representation is concatenated with a text embedding and passed to a 4-layer MLP, which predicts the bounding box coordinates.

Results. We report the bounding box accuracy at a 25% Intersection-over-Union (IoU) threshold across different scene clutter levels for SCR-variants and baselines in Table 4.3. Firstly, we present the results from [70] for CLIP, R3M and Voltron and observe that Voltron outperforms the other two baselines. We see that SCR is tied with Voltron and that VC-1 and SD-VAE perform the best with a 1.5% lead. The better performance of these vision-encoder-only methods highlights that on this task, it is not a challenge for the downstream decoder to learn to associate the visual embeddings with the (CLIP) text encoding of the language specification. Since the training budget is fixed, we observed that some of the runs could potentially improve over extended training. However, we were primarily interested in this task not just to compare the downstream visual prediction performance, but to use it as a testbed for exploring the following two questions: (1) Do the performance differences between the representations we evaluated in Sections 4.5.2 to 4.5.4, stem from the absence of fine-grained spatial information encoded within the representations? We refute this claim in Section 4.6.4, where we present the impact of the representations’ spatial aggregation method on prediction performance. (2) Additionally, we explore to what extent language prompting influences the representations from SCR on language-conditioned tasks in Section 4.6.3.



The lemon on the rear left of the instant_noodles.

Figure 4.4: Sample from the OCID-Ref dataset used for the Referring Expressions task.

Model	Average	Maximum clutter	Medium clutter	Minimum clutter
CLIP	68.1	60.3	76.6	67.0
R3M	63.3	55.3	68.3	63.3
Voltron	92.5	96.9	91.8	90.2
VC-1	94.6	93.7	96.5	93.7
SD-VAE	94.3	93.2	96.3	93.4
SCR	92.9	91.1	95.9	91.8
SCR-FT	91.8	90.1	94.8	90.8

Table 4.3: Referring Expression Grounding (Accuracy at threshold IoU of 0.25 with label.).

4.5.7 Grasp Affordance Prediction

In this section, we present our experiments on a second visual prediction task continuing from the experiments in Section 4.5.2. The **Grasp Affordance Prediction** task requires predicting per-pixel segmentation outputs for certain areas of objects in an RGB image. These areas correspond to parts of the surface that would be amenable to grasping by a suction gripper. The evaluation metric adopted in prior work is the precision of predictions corresponding to positive graspability at varying confidence levels (90, 95, and 99th percentile of the predicted per-pixel probabilities, denoted as Top90, Top95, and Top99 in Table 4.4). We refer the reader to Karamcheti et al. [70] for the complete task setup details.

We re-ran all the methods using the evaluation repository provided with the work, and obtained different results compared to the reported numbers in Karamcheti et al. [70], which we attribute to a bug that we fixed related to the computation of the precision metrics. The evaluation procedure for this task adopted in prior work involves a 5-fold cross-validation, and we observed a high variability in the results, with different runs of 5-fold cross-validation yielding different final test metrics. Our findings highlight that SCR and our adaptation of CLIP (in gray, detailed in Section 4.6.4) both excel at this task, achieving a Top99 score of 72.9. Interestingly, we see that fine-tuning did not enhance the performance of SCR on the visual prediction tasks explored in this section and Section 4.5.2, suggesting a potential disconnect between visual prediction and control task benchmarks.

4.5.8 Comparison with LIV

We include a comparison with LIV [97] on two tasks that involve manipulation and navigation. LIV is a vision-language representation learned through contrastive learning on the EpicKitchens dataset [37]. Similar to R3M results in the main paper, this representation does well on manipulation tasks but poorly on navigation tasks.

Table 4.4: Grasp Affordance Prediction: Precision on pixels corresponding to positive graspability at varying probability threshold levels.

Model	Top99	Top95	Top90
CLIP	60.3	45.0	28.6
CLIP (Comp)	72.9	55.9	36.5
Voltron	62.5	42.8	32.1
SD-VAE	55.6	41.3	33.8
SCR	72.8	55.9	54.5
SCR-FT	72.3	54.6	44.4

Table 4.5: Comparing to LIV representations on manipulation and navigation tasks.

Model	Franka-Kitchen	OVMM
SCR	45.0	38.7
SCR-FT	49.9	41.9
LIV	54.2	8.4

4.5.9 Overall Ranking of Representations

In Table 4.6, we present the consolidated scores across the four control benchmarks we study in section 4.5, for all the representations we evaluate in this work. This is to give a higher-level view of the all-round performance of the different representations on the diverse set of tasks we consider. We see that VC-1, SCR, and SCR-FT emerge as the top three visual representations overall. While VC-1 is a representation-learning foundation model trained specifically for robotics tasks, SCR and SCR-FT are the diffusion model representations that we developed in this work, confirming the potential of large pre-trained foundation generative models across a wide array of downstream robotics tasks.

4.6 Deconstructing Stable Control Representations

In this section, we deconstruct Stable Control Representations to explain which design choices are most determinative of model robustness and downstream performance.

Table 4.6: Representation Performance Comparison: Numbers in the task columns (OVMM, ImageNav, MetaWorld, Franka Kitchen) indicate relative scores of different representations (normalized by the highest score on that task), and the average normalized score column indicates the averaged scores across the task-wise relative scores where numbers are available.

Method	OVMM	ImageNav	MetaWorld	Franka Kitchen	Avg. Norm. Score
VAE	-	0.629	0.786	0.759	0.725
R3M	-	0.414	1.000	1.000	0.805
VC-1	0.969	0.951	0.961	0.825	0.927
CLIP	0.924	0.706	0.939	0.630	0.800
SR	0.924	1.000	0.983	0.781	0.922
SR-FT	1.000	0.942	0.989	0.866	0.949

Table 4.7: Ablations. We analyze the impact of varying the denoising timestep, layer selection, and input text prompt on SCR performance on the Franka-Kitchen benchmark. We report the mean and standard error over 3 random seeds.

(a) Denoising timestep.		(b) Layer selection.	
Timestep	Success Rate	Layers	Success Rate
0	49.9 ± 3.4	Down[1-3] + Mid	49.9 ± 3.4
10	48.2 ± 3.1	Down[1-3]	43.0 ± 3.4
100	42.0 ± 3.7	Mid	41.6 ± 3.3
110	42.0 ± 3.4	Mid + Up[0]	42.1 ± 3.6
200	35.1 ± 3.2	Mid + Up[0-1]	48.1 ± 3.6

(c) Input text prompt.	
Prompt Type	Success Rate
None	49.9 ± 3.4
Relevant	49.2 ± 3.5
Irrelevant	48.7 ± 3.3

4.6.1 Layer Selection

We begin our investigation by examining how the performance of SCR is influenced by the selection of layers from which we extract feature maps. We previously chose outputs from the mid and downsampling layers of the U-Net (Figure 4.2), because their aggregate size closely matches the representation sizes from the ViT-based models (VC-1, MVP, and CLIP). Appendix B.2.2 details the feature map sizes obtained for all the models we study. Table 4.7a lists the success rates achieved on the Franka-Kitchen domain when we use different sets of block outputs in SCR.

Table 4.8: Layer-selection ablations across different benchmarks.

(a) Ablations for CLIP on Franka-Kitchen.

Model	Layers	Success
CLIP-L	23 (last layer)	36.3 ± 1.7
CLIP-L	21+23	35.4 ± 2.9
CLIP-L	19+23	38.5 ± 3.2
CLIP-L	12+23	40.8 ± 2.8
CLIP-L	10+23	40.2 ± 3.2
SCR (ours)	Down[1-3] + Mid	49.9 ± 3.4

(b) Layer Ablations for SCR on Meta-World.

Layers	Noise	Success
Mid	200	94.7 ± 2.8
Down[3] + Mid	200	97.3 ± 1.4
Down[1-3]	200	94.1 ± 1.9
Down[1-3] + Mid	200	94.4 ± 1.9
Down[1-3] + Mid	100	94.4 ± 1.9
Down[1-3] + Mid	0	94.1 ± 1.9

We present similar ablations for Meta-World in the top four rows of Table 4.8b.

We observe that utilizing outputs from multiple layers is instrumental to SCR’s high performance. This finding underscores a broader principle applicable to the design of representations across different models: Leveraging a richer set of features from multi-layer outputs should enhance performance on downstream tasks. However, it is important to acknowledge the practical challenges in applying this strategy to ViT-based models due to the high dimensionality of each layer’s patch-wise embeddings ($16 \times 16 \times 1024$ for ViT-L for images of size 224×224). We present the success rates achieved on the four benchmarks when aggregating multi-layer embeddings from CLIP models in Tables 4.8a and 4.9, alongside SCR (the representation size for which is now half in comparison). In Table 4.8a, we observe that moving towards middle layers leads to higher performance indicating that CLIP layers 10-14 encode some details useful to the Franka-Kitchen benchmark. While we see benefits from including the output of certain additional layers, it still underperforms SCR.

Table 4.9: Comparison of CLIP Layer Ablations on Meta-World, OVMM, and ImageNav

(a) Meta-World and OVMM			
Model	Layers	Meta-World	OVMM
CLIP-L	23 (Last Layer)	90.1 ± 3.6	38.7 ± 1.7
CLIP-L	21+23	91.2 ± 2.3	-
CLIP-L	12+23	91.7 ± 2.6	38.6 ± 1.6
SCR	Down[1-3] + Mid	94.9 ± 2.0	43.6 ± 2.1

(b) ImageNav		
Model	Layers	ImageNav
CLIP-B	11 (Last Layer)	52.2
CLIP-B	6+11	66.6
SCR	Down[1-3] + Mid	73.9

4.6.2 Sensitivity to the Noising Timestep

Next, we characterize the sensitivity of task performance to the denoising step values chosen during representation extraction. We present results on the Franka-Kitchen tasks in Table 4.7b, and on the Meta-World tasks in the bottom three rows of Table 4.8b. We see that the performance across nearby timesteps (0 and 10 or 100 and 110) is similar, and that there is a benefit to doing a coarse grid search up to a reasonable noising level (0 vs 100 vs 200) to get the best value for a given task.

4.6.3 How is Language Guiding the Representations?

Recall that in the OVMM experiments (Section 4.5.4), we concatenated the target object’s CLIP text embedding to the visual representations before feeding it to the policy. For SCR and SCR-FT, we also provided the category as the text prompt to the U-Net, and additionally extracted the generated cross-attention maps for SCR-FT-ATTN. In this subsection, we seek to more closely understand how the text prompts impact the representations in SCR.

We first consider the Franka-Kitchen setup from Section 4.5.2, which includes manipulation tasks that do not originally come with a language specification. We experiment with providing variations of task-relevant and irrelevant prompts during

the representation extraction in SCR. Table 4.7c shows the downstream policy success rates for irrelevant (“*an elephant in the jungle*”) and relevant (“*a Franka robot arm opening a microwave door*”) prompts, compared to the default setting of not providing a text prompt. We see that providing a prompt does not help with downstream policy performance and may even degrade performance as the prompt gets more irrelevant to the visual context of the input.

We now move to the Referring Expressions Grounding task which requires predicting a bounding box for an object being referred to in a sentence, within an image depicting cluttered objects. We defer the main presentation of this task to Section 4.5.6 and use it to probe the degree of language grounding in SCR in this section. To study the role of the U-Net in shaping the visual representations guided by the text, we examine different text integration methods to generate SCR representations in Table 4.10.

We compared the following approaches for providing the task’s text specification to the task decoder (also depicted in Figure 4.5):

- (a) **No text input:** Exclude text prompt from both SCR and the task decoder by passing an empty prompt to the U-Net and using only the resulting SCR output for the decoder.
- (b) **Prompt only:** Pass text prompt only to the U-Net.
- (c) **Concat only:** Concatenate the CLIP embedding of the text prompt with the visual representation, feeding an empty prompt to the U-Net.
- (d) **Prompt + Concat:** Combine “Prompt Only” and “Concat Only”.
- (e) **Only text encoding:** Ignore the visual representation and rely only on CLIP text embeddings.

Investigating the results of (a) and (b) in Table 4.10, it is evident that incorporating the text prompt into the U-Net significantly enhances accuracy compared to ignoring the text altogether. The difference in scores between (b) and (c) indicates that directly providing text embeddings to the decoder improves performance, suggesting

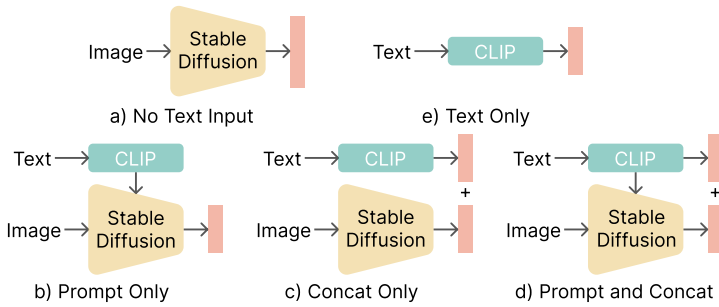


Figure 4.5: Illustration of different approaches to providing relevant vision-language inputs to a downstream task-decoder.

Table 4.10: Ablating text input to SCR on the referring expressions grounding task.

Configuration	Score
(a) No text input	14.8
(b) Prompt only	82.7
(c) Concat only	92.2
(d) Prompt + Concat	92.9
(e) Only text encoding	37.5

that certain crucial aspects of object localization are not fully captured by the representation alone. Comparing (c) to (d), we see that with concatenated text embeddings, further modulation of the visual representations does not provide significant benefits. Finally, the significant decrease in the score for (e) reveals the extent to which the task relies on text-based guesswork.

These findings align with both intuition and recent research on controllable generation with diffusion models [186] that highlights the challenges associated with using long-form text guidance. There are ongoing research efforts, focused on training models with more detailed descriptions or leveraging approaches to encode and integrate sub-phrases of long texts, that seek to address these challenges.

4.6.4 The Effect of Spatial Aggregation

In this study, we refine the approach for extracting representations by integrating a convolutional layer that downsamples the spatial grid of pre-trained representations. This adjustment, referred to as a “compression layer” by Yadav et al. [177], aims to reduce the high channel dimension of pre-trained model outputs without losing spatial details, facilitating more effective input processing by downstream task-specific decoders.

We explore the effect of spatial aggregation methods by comparing the convolutional downsampling layer method to multi-headed attention pooling (MAP) used for CLIP embeddings in Karamcheti et al. [70]. We find that using a compression

layer significantly improves performance on the fine-grained visual prediction tasks described in Sections 4.5.6 and 4.5.7 as reported in Table 4.11 (columns 3-4). This result challenges the conjecture made in prior work that CLIP representations are limited in their ability to provide accurate low-level spatial information [70] and emphasizes the critical role of appropriate representation aggregation.

Building on this result, we assess whether better spatial aggregation can improve the performance of CLIP representations on downstream control tasks. We present these results in Table 4.11 (columns 5-6) for VC-1 and CLIP on the MuJoCo tasks. We see that the compression layer often outperforms the use of CLS token embeddings (by 1-2%), but CLIP representations still fail to match the best-performing models. This result provides evidence that the underperformance of CLIP representations on control tasks is unlikely due to a lack of sufficiently fine-grained visual information. Finally, we note that the compression layer aggregation technique was used for all baselines in Tables 4.1b and 4.1c to ensure a strong baseline comparison. We recommend that future studies adopt this methodology to enable a fairer comparison of representations.

4.7 Discussion

In section 4.6, we deconstructed Stable Control Representations and highlighted techniques used in our approach that could be applied to extract representations from other foundation models. Our analysis in Sections 4.6.1 and 4.6.4 revealed that

Table 4.11: We ablate the spatial aggregation method for VC-1 and CLIP. On the fine-grained visual prediction tasks, we compare the average precision between using multi-head attention pooling (MAP) and the compression layer. On the Meta-World & Franka-Kitchen tasks, we compare the average success rates (\pm one standard error) between the CLS token and compression layer embeddings.

Model	Aggregation Method	Refer Exp. Grounding	Grasp Affordance Prediction	Meta-World	Franka-Kitchen
VC-1	MAP/CLS	93.2	24.7	88.8 \pm 2.2	52.0 \pm 3.4
VC-1	Compression	94.6	83.9	92.3 \pm 2.5	47.5 \pm 3.4
CLIP	MAP/CLS	68.1	60.3	88.8 \pm 3.9	35.3 \pm 3.4
CLIP	Compression	94.3	72.9	90.1 \pm 3.6	36.3 \pm 3.2

University of Oxford

using multi-layer features and appropriate spatial aggregation significantly affects performance, and overlooking these factors can lead to misleading conclusions about the capabilities of previously used representations.

Next, our investigation into how language prompts shape diffusion model representations uncovered nuanced results and showed that text influence on representations does not consistently increase downstream utility. This is particularly evident in tasks where text specification is not required and where training and test environments are congruent, minimizing the need for semantic generalization. In contrast, tasks like referring expressions grounding demonstrate the necessity of direct access to text embeddings for accurate object localization, even when representations are modulated to considerable success. For the OVMM task, we identified a scenario where multimodal alignment is essential and proposed a method to explicitly utilize the latent knowledge of the Stable Diffusion model through text-aligned attention maps, which is not straightforward to do for other multimodal models. Future research could design methods to derive precise text-associated attribution maps for other models.

Finally, we contrasted the simplicity of fine-tuning diffusion models with that of the contrastive learning objective required to fine-tune CLIP representations. The former only requires image-text samples for the conditional generation objective, whereas the latter requires a sophisticated negative label sampling pipeline along with large batch sizes to prevent the model from collapsing to a degenerate solution [121]. We demonstrate this phenomenon empirically on the Franka-Kitchen environment by fine-tuning CLIP similarly to SCR-FT in Section 4.5.5.

4.8 Conclusion

In this chapter, we proposed Stable Control Representations, a method for leveraging representations of general-purpose, pre-trained diffusion models for control. We showed that using representations extracted from text-to-image diffusion models for policy learning can improve generalization across a wide range of tasks including

University of Oxford

manipulation, image-goal and object-goal based navigation, grasp point prediction, and referring expressions grounding. We also demonstrated the interpretability benefits of incorporating attention maps extracted from pre-trained text-to-image diffusion models, which we showed can improve performance and help identify downstream failures of the policy during development. Finally, we discussed ways in which the insights presented in this chapter, for example, regarding feature aggregation and fine-tuning, may be applicable to other foundation models used for control. We hope that Stable Control Representations will help advance data-efficient control and enable open-vocabulary generalization in challenging control domains as the capabilities of diffusion models continue to improve.

5 | Learning Policies to Leverage Experience

Contents

5.1	Introduction	80
5.2	Related Work	82
5.3	Methodology	83
5.3.1	Problem Setting	84
5.3.2	Context Summarization in Memo	85
5.4	Experiments	87
5.4.1	Benchmarks	87
5.4.2	Baselines	89
5.4.3	Summarization Outperforms Full In-Context Access	90
5.4.4	Advantage of Summary Accumulation Over Fixed Recurrent Memory	91
5.4.5	Importance of Long-Horizon Gradient Propagation: Comparison to Autocompressors	93
5.4.6	Extrapolation and Experience Subsampling During Evaluation	94
5.4.7	Finetuning	94
5.4.8	Ablations	95
5.5	Discussion	96
5.6	Limitations	97

Beyond leveraging powerful pretrained representations, efficient agents must also act efficiently by leveraging relevant context across long timeframes. This is particularly critical in tasks where delayed rewards necessitate recalling key past events over extended horizons. Standard transformer architectures struggle with such long-horizon reasoning due to both computational and memory constraints. To address this, the following chapter introduces Memo, a memory-augmented transformer framework that enables agents to summarize and store past experience using

learned memory tokens. Memo allows agents to maintain compact, scalable memory representations while retaining the capacity for long-term decision making.

5.1 Introduction

Humans intuitively prioritize and retain memories relevant to their current tasks, filtering out irrelevant details such as the color of a house’s walls along a route unless it serves as a crucial navigation landmark. Similarly, reinforcement learning (RL) agents must learn to selectively retain and access task-specific memories guided by task objectives rather than predefined rules. This ability is especially critical for long-horizon tasks, where delayed rewards make credit assignment challenging and necessitate efficient training and inference over extended timescales to effectively capture the relationship between past and future events.

Recently, Transformers have become the go-to framework for sequence modeling due to their flexible and expressive encoding, free from the fixed-size constraints of recurrent neural networks (RNNs) [32, 61]. However, it assumes access to all past information at each step rather than selectively retaining relevant memories, which can lead to inefficiencies and make it harder to extract task-relevant information for decision-making. Their quadratic attention complexity further limits scalability, making it challenging to process long contexts efficiently, especially when gradients must propagate over large sequences. At test time, Transformers face another practical challenge: storing an increasingly large key-value (KV) cache is computationally expensive, and attending over ever-growing contexts requires the ability to generalize beyond the temporal patterns seen during training.

To address these challenges, we propose **Memo**, a novel framework that enhances transformer-based agents by enabling them to summarize and index past experiences through specialized summary tokens. Memo introduces a simple yet effective training procedure where transformers learn to compress their experience by periodically generating summary tokens that encode relevant past information. These summary tokens are stored in a dedicated memory buffer, allowing the model to attend to

condensed representations of prior states instead of maintaining a full-context cache. This enables Memo to maintain a compact memory footprint at inference, reducing computational costs while preserving long-horizon reasoning capabilities.

Inspired by advancements in extending context lengths in language models, Memo adapts these principles to reinforcement learning (RL), addressing the unique challenges presented by RL tasks. Specifically, Memo integrates with both on-policy and off-policy RL agents, leveraging learned compression mechanisms to enhance sample efficiency and generalization in sequential decision-making settings. We demonstrate its effectiveness across diverse tasks, including a grid world and a 3D indoor navigation task in Habitat. Memo matches or outperforms the naive transformer baseline, which requires storing its entire context, leading to an 8-10 \times larger cache. Despite using significantly less memory, Memo achieves better in-context learning (ICL) behavior, higher success rates, and shorter path lengths in unseen environments. On large-scale navigation tasks, we show that Memo also exhibits superior robustness in streaming settings where the cached context is truncated, demonstrating improved adaptability under limited inference budgets.

Our contributions are summarized as follows:

- We propose Memo, a framework that enables transformers to learn to summarize and store task-relevant past experiences, reducing computational costs while maintaining long-horizon reasoning capabilities.
- We evaluate Memo on sequential decision-making tasks, demonstrating its efficiency and improved in-context learning behavior compared to transformers that require full-context storage.
- We show that Memo is a general and versatile memory augmentation technique applicable to both on-policy and off-policy RL agents for long-horizon tasks.

5.2 Related Work

In this section we briefly review research on extending context limits of transformers in language modeling, as well as the state of the art in long sequence modeling and decision making in RL.

Extending Transformer Contexts in language: A significant limitation of current large language models is their restricted input context length, prompting research into methods for scaling and generalizing to larger contexts [184, 107, 58, 142]. Key-value (KV) caching can reduce recomputation at training [36] or inference time [114], by storing and reusing self-attention outputs during decoding. [122] further reduce memory by maintaining a compressed cache, though this compression is not completely task-guided.

Beyond compute efficiency, recent work explores dynamic context extension. Recurrent Memory Transformer (RMT) [23] introduces summarization tokens to periodically compress and propagate prior context while discarding older tokens. Autocompressors (AC) [30] extend this by accumulating generated summaries across context windows, avoiding RMT’s fixed-size memory but still truncating gradient propagation through summaries. We extend this context summarization approach to the more involved setting of reinforcement learning (RL), where summarization must support decision-making and handle credit assignment over long horizons. Unlike AC, which fine-tunes pretrained models on a supervised token prediction task with limited gradient propagation—and aims to match but does not surpass full-context transformer baselines—we train from scratch, integrating summarization directly into the RL optimization process and propagating gradients across all summaries. This enables more effective task-driven memory formation, surpassing full-context transformer baselines in both efficiency and scalability.

Transformers in RL: Transformers, while widely used in self-supervised language modeling [39, 19], have seen slower adoption in reinforcement learning (RL) due to challenges like sparse rewards, optimization instability, and limited batch sizes in on-policy RL. However, as we move to more complex tasks that demand long-horizon

planning and reasoning over past experiences, transformers have shown promising initial results, outperforming recurrent models in POMDPs when encoding longer temporal dependencies [109].

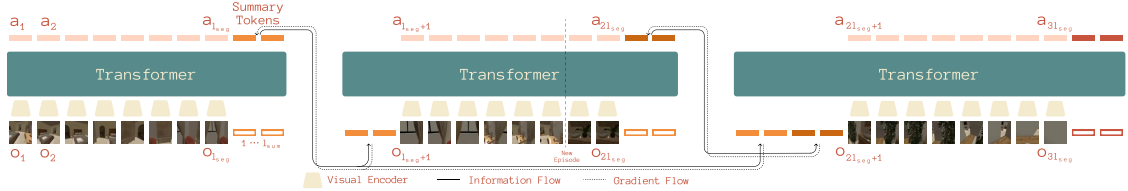


Figure 5.1: Architecture Diagram of Memo. The frames $O_{1-3l_{seg}}$ depict the input sensor observations to the agent at timesteps $1 - 3l_{seg}$. The figure depicts the information flow between three consecutive segments of a much longer context of inputs provided to a transformer during training. The summary tokens bottleneck the information passed on from one chunk of inputs to the next.

To stabilize transformer training in partially observable RL, Parisotto et al. [116] used KV caching while restricting gradient propagation to a small window of time steps, limiting the learning of long-term dependencies. More recently, ReLIC [43] improved convergence of on-policy RL algorithms by unfreezing the historical context within a rollout and applying frequent updates, while AMAGO [54] adapted off-policy RL to train with a shared transformer backbone. While these methods enhance long-horizon learning, ReLIC’s high computational cost and slow training hinder scalability, and AMAGO is limited to small state-space observations, reducing its applicability to more complex tasks. Our approach addresses these challenges by enabling efficient memory utilization and leveraging a more scalable transformer architecture for extreme long-horizon decision-making.

5.3 Methodology

This section outlines our methodology for memory-augmented in-context reinforcement learning (RL). We first formalize the problem and describe the in-context RL algorithms and the sequence model we consider, in Section 5.3.1. We then introduce our context summarization mechanism in section 5.3.2, and describe key implementation details including the attention masking, positional encoding, segment randomization and cache management scheme we use.

5.3.1 Problem Setting

In this work, we study tasks that require a model to perform sequential decision-making over long executions while leveraging its experience history to improve efficiency over time. Many diverse reasoning problems fall into this category, ranging from embodied navigation to language modeling.

We formalize these tasks as *partially observable Markov decision processes (POMDPs)*, which extend standard Markov decision processes (MDPs) to settings where the agent cannot directly observe the underlying environment state. A POMDP is defined by the tuple $\mathcal{M}_{\text{POMDP}} = (\mathcal{S}, \mathcal{A}, \mathcal{O}, \mathcal{P}, r, d_0, \gamma)$, where \mathcal{S} is the set of latent states, \mathcal{A} the action space, \mathcal{O} the set of possible observations, $\mathcal{P}(s' | s, a)$ the transition kernel, $r(s, a)$ the reward function, d_0 the initial state distribution, and γ the discount factor. At each timestep, the agent receives an observation $o_t \in \mathcal{O}$ generated according to the environment’s observation function, which may only partially reveal the underlying state s_t .

We adopt an *in-context reinforcement learning* perspective, in which models adapt and improve their behavior based on past inferences rather than relying on pre-collected supervised datasets. By integrating the history of observations and actions into the agent’s context, we can transform the POMDP into a fully observable MDP over historical sequences, enabling the use of standard on-policy and off-policy RL methods.

Concretely, we employ a sequence encoder to model the agent’s policy. At each timestep t , the encoder receives the sequence of past observations and actions $X_t = \{(o_1, a_1), \dots, (o_{t-1}, a_{t-1}), o_t\}$, and produces a hidden representation $h_t = \text{SeqEnc}(X_t)$, which integrates information from the entire history up to time t . The policy distribution and value estimate are then computed using learnable actor and critic heads: $\pi(a_t | h_t)$, $V(h_t)$, where $a_t \in \mathcal{A}$ is sampled according to the policy π . This design allows the agent to handle partial observability by conditioning decisions on the full sequence of past experiences.

In-Context RL Algorithms: In in-context RL, an agent adapts over time by conditioning on its entire past within a trial. A trial consists of multiple episodes, where each episode resets upon termination, but the agent retains memory across episodes. These episodes share a common underlying context—such as environment dynamics, task structure, or goal distribution—enabling the model to leverage past experiences for more efficient adaptation. Sequence models like Transformers encode history by attending to all previous timesteps within a trial, allowing the agent to infer temporal dependencies and adjust its behavior accordingly. However, as trial lengths increase, attending to all past timesteps becomes computationally infeasible. To address this, we introduce Memo, a method for constructing and accessing memory representations in Transformers to improve long-term context management in RL, as described in section 5.3.2. To demonstrate its efficacy and broad applicability to long-context RL, we integrate Memo with both on-policy and off-policy RL method:

- On-policy RL: We incorporate our memory mechanism into RELIC [43], an adaptation of DD-PPO [169], which applies frequent partial updates during trajectory rollouts to help transformer policies learn more effectively from in-context experiences.
- Off-policy RL: We extend Memo to AMAGO [54], which trains transformer-based policy using a shared sequence encoder and unified actor-critic loss to improve stability and learning efficiency.

5.3.2 Context Summarization in Memo

Following [30], we introduce context summarization as a learned sub-task that enables the transformer encoder to compress and retain task-relevant information from long observation histories. This mechanism allows the model to efficiently process past experiences while optimizing for task rewards in RL. Our approach uses learnable summary embeddings to prompt the transformer to generate summary tokens at predefined intervals, ensuring efficient memory utilization in long-horizon tasks.

Instead of attending to the full sequence history, the transformer periodically compresses past context into summary tokens, which are then fed back into the model in future timesteps. We partition long input sequences into segments of length l_{seg} , and generate l_{sum} summary tokens at the end of each segment. These tokens act as compact memory representations, allowing the model to condition on past experiences without requiring access to raw observations. The encoding and integration of summary tokens into future segments are illustrated in fig. 5.1, and the pseudocode is provided in appendix C.0.4. The summarization mechanism is trained end-to-end through the RL objective, allowing the model to attend over and refine all previously generated summary tokens. Gradients propagate through the attention mechanism, ensuring that memory updates remain task-driven.

Attention Masking: We use causal masking which includes all the previous summary tokens as well as the previous observations within the segment being processed at time step t . This excludes any previous observation that contributed to and that was processed before the most recent summarization. This creates a bottleneck for the flow of historical information to be solely through summary tokens.

Positional Encoding: To add positional awareness, we assign indices to an observation at time step t based on its relative position within a segment. We use the same positional encoding method (linear or ROPE) used by the baseline implementation which we will be comparing to. The position labeling scheme in a naive full-context transformer input has indices which run from 0 to the length of the context window. In Memo, the input context at time step t in the rollout consists of $n * l_{sum}$ summary tokens and the latest segment’s observations, where $n = \lfloor t/l_{seg} \rfloor$. The position indices thus range from 0 to $n * l_{sum} - 1$ for the summaries and then start from $n * l_{sum}$ and go to $t - n * l_{seg} + n * l_{sum}$ for the most recent observations in the latest segment. The summary embeddings at the end of a segment are assigned indices following those of the last segment observation.

Segment length randomization: Following Chevalier et al. [30], we randomize

segment lengths during training by sampling uniformly within $\pm 20\%$ of a fixed l_{seg} . This approach, which previously showed small improvements in perplexity for token prediction, will be ablated in 5.4.8. A fixed l_{seg} is used during data collection and evaluation, while training uses randomized segment lengths.

Maintaining the KV Cache: In on-policy RL, the KV cache stored during action selection can become stale after a model update, as the new weights would encode different representations for the same past context. This divergence is problematic since policy decisions depend on context, making consistency crucial. ReLIC addresses this by refreshing and re-encoding the KV cache with the latest model weights after every on-policy update. We extend this approach to memory tokens, ensuring consistency by recomputing the summary vectors alongside the KV cache refresh. This guarantees that both cached representations and learned memory remain aligned with the current policy.

5.4 Experiments

In this section, we present experimental results highlighting different aspects of our proposed method. We first describe the tasks and baselines used in our experiments, followed by different experimental insights in the following subsections.

5.4.1 Benchmarks

ExtObjNav: The Extended Object Navigation (EXTOBJNAV) task, first introduced in [43], builds on the OBJECTNAV task commonly used in embodied AI research [15, 100]. While OBJECTNAV requires an agent to navigate to a single object goal in an episode, EXTOBJNAV focuses on how performance evolves when the agent is repeatedly tasked with reaching different object goals within the same scene. The task evaluates the agent’s ability to leverage memory—utilizing information from prior exploration during navigation to reach subsequent goals.

The EXTOBJNAV task uses 37 training and 12 validation scenes from HSSD [75] and includes 20 object instances from the YCB dataset [26]. These objects can be

randomly placed on receptacles throughout the scene, with each *placement* featuring an average of 30 objects sampled from the 20 instances. We use 11,100 novel placements during training and 108 during evaluation. A sample scene is shown in fig. 5.2a.

In EXT OBJNAV, embodied agents are trained and evaluated over *trials*, each consisting of a sequence of episodes with the same object placement in a scene. Each episode of EXT OBJNAV mirrors OBJECTNAV in structure, with the agent being randomly spawned in the environment and tasked with locating an object from a sampled object category. Episodes are capped at 500 steps, while trials are limited to 4096 steps during training (unless specified otherwise) and 32k steps during evaluation.

The agent is a Fetch robot with a head mounted camera, which outputs 256x256 RGB images, and an odometry sensor that provides relative position and orientation data w.r.t. the episode’s starting position. It operates in a discrete action space: Forward (0.25m), TurnLeft (0.3°), TurnRight, and Stop. An episode is successful if the agent stops within 2m of the goal target with the object in view, occupying at least 10 pixels in the image.

In the following sections, we report results using the success rate (SR) and success weighted by path length (SPL) metrics, with SPL measuring the efficiency of the agent’s navigation relative to the shortest possible path. Results are computed over 32k interaction steps on the validation set and plotted at intervals of 500 steps by averaging the metrics for the episodes finished within each interval. All experiments are conducted over 10 random seeds. Further details are provided in appendix C.0.1

Dark-Key-To-Door: Dark-Key-To-Door [85] is a Meta-RL [16] benchmark where an agent needs to find an invisible key to open an invisible door in a 9x9 2D gridworld. The agent is given +1 reward for finding the key and door each. The agent only knows its (x, y) position at every timestep and needs to remember the key and door position based on the reward received during previous episodes.

Each episode can last for a maximum of 50 timesteps, while the total trial duration

is fixed at 500 steps. We evaluate the agent’s performance by reporting the average reward achieved across 960 trials and 3 seeds in validation environments. An agent that effectively utilizes contextual information to infer the key and door locations can complete multiple episodes efficiently within a trial, achieving a total reward greater than 50.

5.4.2 Baselines

Below, we describe the main transformer-based baselines that we compare Memo to in this work.

- The **full context transformer** baseline corresponding to ReLIC [43] and AMAGO [54]. We refer to it as FCT in the following sections for brevity.
- **Transformer without Inter-Episode Attention (no-IEA)**: We modify the attention masking such that the model can’t attend over previous trials in the rollout. This sets a lower bound for what the performance would be if the model didn’t do in-context learning and simply used its current trial’s observations to explore and find objects.
- **Recurrent Memory Transformers (RMT)**: We also include a recurrent-only version of Memo that does not accumulate summary tokens. This baseline represents an implementation of RMT [24] adapted for the RL setting. The purpose of this comparison is to evaluate the benefits of Memo’s summary accumulation against a fixed-size memory constraint. To ensure a fair comparison, we set the fixed memory size in RMT equal to the total number of summary tokens Memo would accumulate over multiple segments ($l_{sum} * \text{number-of-segments}$).
- **Autocompressors (AC)**: Memo with a pre-trained initialization and highly truncated backpropagation through time TBTT. This represents an implementation of AC [30] tailored to the RL setting.

We exclude RL^2 and Tr-XL due to their poor performance in our preliminary experiments and also in the results of [43, 54] - these baselines never managed to achieve more than a 10% success rate. In the case of Tr-XL, this is likely because

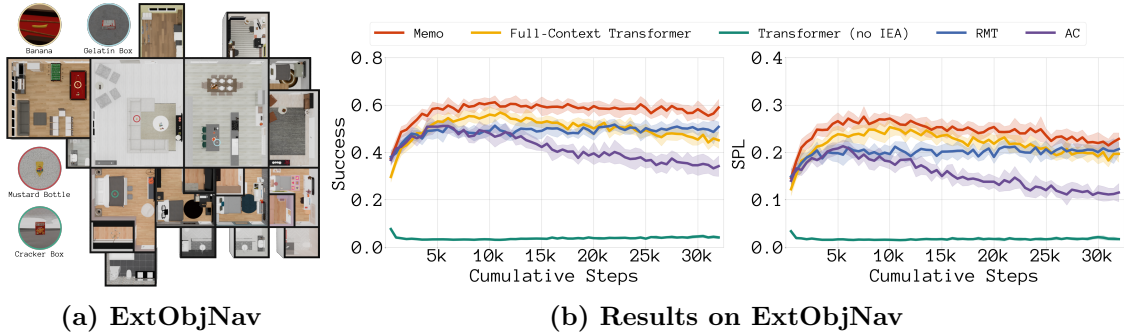


Figure 5.2: (Left) Overhead view of a training scene in Habitat simulator. The EXT OBJNAV task involves placing multiple objects around the house, and then sampling an object category as the goal each time the previous goal is reached. The figure shows some sample objects that may be placed around the house, such as Banana, Cracker Box, etc. (Right) Val success rate and SPL curves over 32k in-context learning steps in novel scenes, for different methods trained with ReLIC. We compare Memo to the full-context transformer (FCT), the FCT variant which does not attend over previous episodes (no IEA), the recurrent memory transformer (RMT), and the Autocompressors (AC) variant.

the cached history is frozen and there is no way for the model to learn how to encode inputs better such that these encodings are useful in the far future, but it is possible that given a much larger history window size and compute budget it could start to learn a better policy. In the following subsections, we present experimental insights examining how Memo’s memory creation mechanism and training recipe enhance long-horizon reasoning and in-context learning. Each subsection highlights and focuses on a specific finding and analyzes a subset of baselines drawn from figures that are shared across multiple sections, allowing us to do a more structured comparison.

5.4.3 Summarization Outperforms Full In-Context Access

We first show that SCR achieves higher performance and exhibits better ICL-ability compared to naive full context transformer (FCT) while using significantly less tokens in-context during evaluation.

In fig. 5.2b, we report the validation SR and SPL on the EXT OBJNAV task over 32k environment steps. All methods are trained using on-policy RL technique proposed by ReLIC. Memo is trained with a segment length of 256, 32 summarization

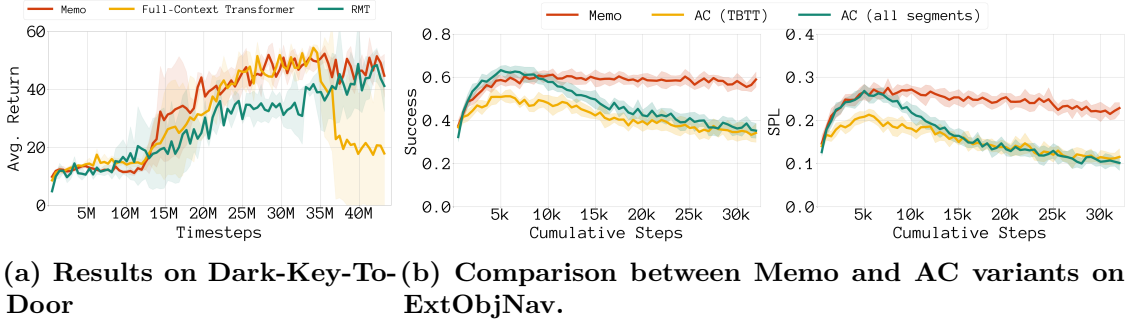
tokens, and a trainable rollout length of 4096. We observe that Memo outperforms FCT achieving 7.5% higher SR and 2.5% higher SPL on average while using 8x fewer tokens (details in Table C.1). We see that both Memo and FCT show in-context learning ability and continue to increase their performance up to 10k steps ($\sim 2.5x$ of the training context length). Beyond this point, both methods experience performance degradation due to limitations in context length extrapolation. Notably, FCT degrades more than Memo in SR, while both show similar degradation in SPL. The single-episode variant (Transformer - no IEA) performs significantly worse, highlighting the disadvantage of not leveraging historical context in this navigation task. We also compare the GPU memory and FLOPs for Memo and FCT in appendix C.0.5, and their sample-efficiency in appendix C.0.9.

In fig. 5.3a, we present a similar comparison on the Dark-Key-To-Door environment, reporting the average total return over trials as training progresses. All methods are trained using the off-policy RL algorithm AMAGO. We see that both Memo and RMT match the FCT baseline, achieving peak performance on validation episodes by 40M training steps. Interestingly, FCT shows a notable performance drop around the 35-40M step mark across all seeds, while Memo remains maintains stable convergence. This is likely due to training instability commonly observed in long-context RL [54, 116], highlighting further training challenges with full-context models.

The consistent performance gains observed across both on-policy (with ReLIC) and off-policy (with AMAGO) RL setups demonstrates that our proposal to augment transformer policies with summarization enhances long-context modeling capabilities while being independent of the specific RL algorithm used.

5.4.4 Advantage of Summary Accumulation Over Fixed Recurrent Memory

The summary accumulation mechanism in Memo addresses the limitations of both transformers and recurrent models. While transformers can model full context, they



(a) Results on Dark-Key-To-Door (b) Comparison between Memo and AC variants on ExtObjNav.

Figure 5.3: (Left) We plot the average return over evaluation trials of 500 steps, consisting of 10 or more episodes, plotted against the training progress. We compare Memo, RMT and FCT, when training with the AMAGO RL algorithm over 3 random seeds. (Right) On EXT OBJNAV, we find that restricting gradient propagation to a few consecutive windows (AC (TBTT)) performs much worse than allowing all past segments to remain trainable (AC (all segments)). While AC (all segments) initially performs slightly better than Memo, it suffers from severe performance degradation over time, converging to AC (TBTT) after 16k steps.

suffer from significant memory overhead with long sequences. In contrast, recurrent models are memory-efficient but struggle to propagate gradients effectively over long horizons due to vanishing or exploding gradients. Summary accumulation mitigates context explosion by periodically summarizing experiences and propagating these summaries forward, allowing earlier memories to directly influence outcomes at later timesteps. Unlike recurrent models, where gradients must pass through all intermediate steps, summary accumulation ensures that each summary vector contributes directly to the loss function at later timesteps without degradation. For example, at timestep 1000, all prior summaries still provide an undiminished signal to the optimization process.

We empirically validate this by comparing Memo’s summary accumulation to our implementation of Recurrent Memory Transformer (RMT) where only summaries generated at the end of the current segment are propagated to the next segment. To ensure fairness, we allow the recurrent model a larger summary size. Even with this adjustment, Memo achieves faster convergence, improving training speed by over 10M steps on the Dark-Key-To-Door task (fig. 5.3a). We also visualize the performance of the recurrent variant (RMT) on EXT OBJNAV in fig. 5.2b, using the same summary length as Memo. Here, RMT achieves a lower success rate, further

highlighting the benefits of Memo’s approach.

This analysis demonstrates that summary accumulation not only balances memory efficiency and expressivity but also enhances optimization by enabling direct gradient flow to earlier experiences, improving stability over long horizons.

5.4.5 Importance of Long-Horizon Gradient Propagation: Comparison to Autocompressors

Since Memo’s summarization approach is inspired by the Autocompressors (AC) method from the language modeling domain, we investigate how following the exact training setup from AC would perform in our setting. This involves pre-training a transformer policy on a shorter context length before fine-tuning it with summarization to extend its effective context length. We first train a baseline transformer with a context size of 1024 for 350M steps (half of the total training horizon for other baselines). This model serves as the weight initialization for our AC model, which is then fine-tuned for 350M steps with segment length $l_{seg} = 256$, summary length $l_{sum} = 32$, and a trainable rollout length of 4096. Following AC, we employ Truncated Backpropagation Through Time (TBTT) during training, restricting gradient propagation to summary tokens from only the two preceding segments; we refer to this variant as **AC (TBTT)**. For a fairer comparison to other methods, we also train a version of AC where gradients are propagated across all segments within the rollout, referred to as **AC (all segments)**. This setup is identical to Memo, except it’s bootstrapped from a pre-trained model with a shorter context length.

We discover that this limited propagation is insufficient for tasks requiring long-horizon memory, as AC (TBTT) fail to capture dependencies spanning extended horizons. As illustrated in fig. 5.3b, AC (TBTT) exhibits poorer in-context improvement in object-finding tasks compared to AC (all segments). While AC (all segments) slightly surpasses Memo in SR and matches it in SPL during the first 6-8k steps, its performance degrades significantly over time, eventually converging

to AC (TBTT) after 16k steps. This suggests that while a short context length pre-trained checkpoint can offer some initial advantage, it ultimately hinders effective generalization to longer contexts.

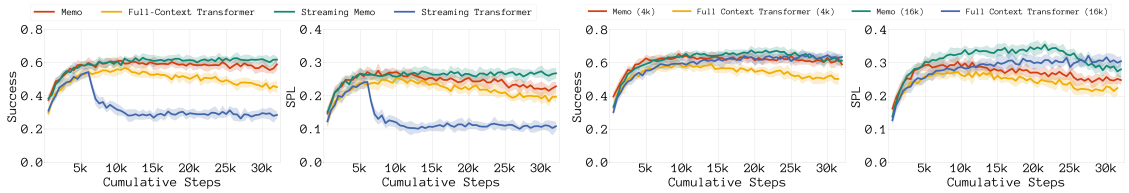
5.4.6 Extrapolation and Experience Subsampling During Evaluation

Avoiding memory explosion due to KV cache accumulation is a challenge for transformers at inference time. We evaluate Memo and the FCT baseline in a streaming setting, where only the most recent T KV cache elements are retained for inference, enforcing a fixed memory limit. To keep our setup simple, we do not update the KV cache to account for shifts in token position indices relative to the retained cache. We set T based on the context length at 6k env steps, as both Memo and FCT exhibit ICL generalization up to at least 6k steps (see fig. 5.2b). Thus, on EXT OBJNAV, we use $T = 6k$ for Streaming Transformer and $T = 1024 (= \text{round}(6000/256) * 32 + 256)$ for Streaming Memo, since 6k steps correspond to 24 sets of summary tokens when $l_{sum} = 32$ and $l_{seg} = 256$.

We present the results in fig. 5.4a. Streaming Memo not only maintains performance through the trial but even outperforms Memo with full memory access in terms of ICL trend. In contrast, the Streaming Transformer undergoes a significant performance drop once streaming begins at 6k steps. Although methods like StreamingLLM [171] address the issues faced by the Streaming Transformer, Memo achieves strong performance without requiring such modifications. Additionally, Streaming Memo provides a way to maintain peak ICL performance by identifying the highest-performing context length during evaluation and initiating streaming beyond that.

5.4.7 Finetuning

In figs. 5.2b and 5.4a, we see a saturation in Memo’s ICL performance at $\sim 60\%$ SR. To investigate whether this limitation stems from suboptimal context length



(a) Memo and Transformer streaming eval- (b) Finetuning for 16k steps on ExtObj-
uations. Nav.

Figure 5.4: (Left) We show a comparison between the accumulating context (default setting) and the streaming version of Memo and the full-context transformer, where streaming starts after 6k eval steps. This comparison highlights the robustness of Streaming Memo, which maintains and even slightly improves performance, whereas Streaming Transformer suffers a sharp decline. (Right) We see that finetuning over a longer number of steps during training serves to partially fix the degradation due to context length extrapolation.

extrapolation, we take Memo’s and FCT’s checkpoints trained until 1B steps and fine-tune them on a $4\times$ larger context size of 16,384 tokens for 500M steps. As shown in fig. 5.4b, the ICL performance of both models improves when evaluated over 32k environment steps. Notably, Memo (16k) outperforms Memo (4k) in SPL, indicating that training with longer contexts enables the policy to find more efficient paths. While the 16k full-context transformer (FCT) improves significantly over its 4k variant, it still achieves lower maximum SR and SPL compared to Memo (16k). Interestingly, Memo (16k) generalizes up to $1.5\times$ its training context length after which it sees a slight degradation, whereas FCT (16k) maintains performance up to at least $2\times$. This necessitates further research into the factors limiting Memo’s long-context generalization, which we leave for future work.

5.4.8 Ablations.

Summary Length Ablation: There is a trade-off between creating or retaining more memory and the compute saved when generating fewer tokens at training or inference time. Additionally, creating more tokens pushes the model further into the “extrapolation” regime, where its positional encodings stop generalizing effectively. We vary the summary length between values 16, 32, and 64 (thereby varying the compression ratio, l_{seg}/l_{sum} , between $16\times$, $8\times$, and $4\times$) on the EXT OBJ NAV task and plot the resulting performance in fig. 5.5a. We find that Memo is highly sensitive to the choice of summary length. The main performance gap emerges after 6k

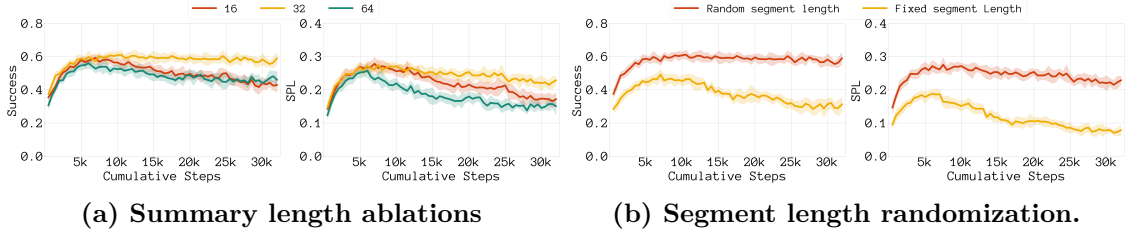


Figure 5.5: Memo ablations. (Left) Effect of varying the number of memory tokens (16/32/64), where 32 outperforms 16, and 16 outperforms 64. (Right) Performance of Memo on EXT OBJNAV with and without randomization of the segment lengths at the end of which summarization is done, showing the latter is significantly less data-efficient. environment steps, where a summary length of 32 achieves the best generalization across longer trajectories. In contrast, a summary length of 64 performs the worst, likely due to summary tokens overfitting to the training context length (see fig. C.4a), leading to redundant information storage and a lower signal-to-noise ratio.

Segment Randomization: In our main experiments, we adopted the strategy of randomizing segment lengths as in [30] for Memo, AC, and RMT to make token generation more robust to specific environment time steps. We randomized segment lengths between $[0.8, 1.2] \times 256$, resulting in a range of [205, 307]. In this ablation, we disable this randomization and evaluate the impact of using a fixed segment length of 256, as shown in fig. 5.5b. We observe that this variant performs significantly worse, not just during evaluation but in also training (see fig. C.4b). We hypothesize that segment length randomization not only improves generalization by preventing overfitting to fixed boundaries but also serves as a form of curriculum. By exposing the model to segments of varying lengths, it naturally encounters both easier (shorter) and harder (longer) compression tasks, fostering a progressive learning effect. We present further results and ablations in appendices C.0.6 and C.0.7.

5.5 Discussion

In this work, we proposed a simple yet effective approach for training transformers with memory in sequential decision-making tasks that require long-horizon reasoning. By introducing a context summarization mechanism, Memo enables transformers to retain and retrieve task-relevant information efficiently, significantly reducing

computational overhead while maintaining long-term dependencies. Our experiments demonstrated Memo’s key advantages over full-context transformer baselines. In terms of performance vs. efficiency, Memo achieves comparable or superior performance while attending to sequences over $8\times$ shorter than the full-context transformer. This improves scalability, enhances extrapolation to longer contexts, and ensures robust performance in streaming settings, where past context must be discarded due to inference constraints.

Furthermore, our analysis highlighted two key design choices: propagating gradients across multiple summarization steps and accumulating summaries rather than using a fixed memory size. We showed that allowing gradients to flow through sequential memory updates significantly enhances long-horizon reasoning, while memory accumulation improves retention of relevant past experiences, leading to more effective decision-making.

5.6 Limitations

Our experiments focus on object navigation in unseen scenes, where agents locate a fixed set of randomly placed objects. We do not evaluate semantic generalization—i.e., navigating to entirely new object categories—since this would conflate memorization with general object recognition capability. Future work could explore leveraging foundation models in a more open-ended setting.

Additionally, while Memo effectively summarizes experience, we do not explore more flexible memory mechanisms, such as memory consolidation, where past summaries are progressively compressed. Investigating these strategies could further improve long-term memory efficiency. We also do not explicitly study length extrapolation. Our results indicate that compressed memory tokens improve robustness to longer contexts and enable streaming inference, but extending generalization beyond training limits remains an open challenge. Lastly, our memory mechanism is trained end-to-end via the RL objective. Future work could explore self-supervised objectives, such as future prediction, to enhance data efficiency in training memory

representations.

While Memo demonstrates promising advances in memory-augmented policy learning, our evaluations have so far been limited to relatively constrained control tasks in simulation. Critically, there is a lack of established benchmarks that systematically test memory and long-horizon reasoning in open-ended, multimodal embodied settings. This gap makes it challenging to rigorously assess how well memory mechanisms—like those proposed in Chapter 5—generalize to more complex, realistic scenarios. To address this, the following chapter introduces a new benchmark specifically designed to evaluate and stress-test memory in embodied agents. This benchmark enables us to move beyond simplified settings and provide a principled framework for measuring progress on long-horizon memory, reasoning, and decision-making in richly interactive environments.

6 | Benchmarking Multimodal Agents on Memory-Intensive Tasks

Contents

6.1	Introduction	99
6.2	Related Work	101
6.2.1	Video QA benchmarks	102
6.2.2	Embodied AI benchmarks	102
6.3	FindingDory	104
6.3.1	Benchmark Overview	104
6.3.2	Tasks	106
6.3.3	Implementation details	107
6.4	Experiments	108
6.4.1	Evaluated Approaches	108
6.4.2	Evaluation Metrics	111
6.5	Results	112
6.6	Conclusion	116

6.1 Introduction

Memory is a fundamental capability for intelligent agents that allows them to recall past experiences, adapt to dynamic environments, and make informed decisions over extended timescales. In humans and animals, it plays a crucial role in navigation, reasoning, and goal-directed behavior. As we strive to develop more capable embodied systems, equipping them with robust memory mechanisms is essential — particularly in settings where agents must process high-dimensional multimodal inputs to interact with their surroundings. The ability to store and retrieve relevant information is the key to unlocking sophisticated behaviors, from household assistance to autonomous exploration.

Recent advances in Vision-Language Models (VLMs) have significantly improved high-level reasoning and planning for embodied tasks. These models leverage large-scale multimodal training to exhibit strong scene understanding and action-guided perception. However, most existing VLM applications focus on short-term or static tasks, such as image captioning or Visual Question Answering (VQA), which do not require sustained memory usage. Extending these models to long-horizon embodied control introduces new challenges: unlike VQA, which often involves processing a single image or short video clip, embodied agents must integrate observations over time, recall past experiences, and act accordingly within their environment.

While recent efforts have sought to extend long-context understanding in VLMs, primarily through video-based QA or long-document comprehension [145, 174, 98], these approaches fail to fully capture the complexities of long-term memory in decision-making. Many existing QA benchmarks either rely on multiple-choice formats, which enable guessing, or require extensive human annotation to curate question-answer pairs that require models to attend to long video sequences. In contrast, we propose embodied memory tasks—such as *“navigating to a soft toy that you did not rearrange yesterday”* or *“find the second object picked up earlier”*—that demand fine-grained recall and multi-hop reasoning over past observations. These tasks provide a more rigorous test of memory capabilities, requiring agents to track and respond to environmental changes rather than passively perceive static information.

In this work, we introduce FINDINGDORY, a benchmark designed to evaluate long-horizon memory in highly photorealistic simulated environments. Our benchmark consists of **60 diverse navigation tasks**, which we categorize along various axes of memory requirements, that require both long-range temporal and spatial reasoning while discouraging heuristic shortcuts or random guessing. To ensure that performance is directly tied to memory utilization rather than simple perception, we curate scenarios where agents must recall past interactions to succeed. Additionally, our benchmark features dynamic environments - where agent interactions modify the scene - making it essential to reason over changing contexts. Furthermore,

University of Oxford

FINDINGDORY is procedurally extensible, allowing tasks to scale in complexity as vision-language models improve, ensuring continued relevance as embodied AI progresses.

Unlike standard QA datasets that rely on subjective human annotations—such as manually selected keyframes or hand-crafted question-answer pairs—our benchmark leverages photorealistic simulation to enable automatic, objective evaluation of memory-based navigation. These tasks go beyond static recall since they require agents to perform complex spatio-temporal reasoning over past interactions. For instance, tasks such as “*navigate to any object that you interacted with yesterday*” demand not only accurate memory retrieval, but also strategic decision-making to identify and navigate to the *nearest* valid target from where the agent is currently positioned. In FINDINGDORY, we introduce metrics that quantify task completion efficiency and not just successful recall.

Our key contributions are:

1. A benchmark for evaluating long-horizon memory in embodied decision-making, featuring 60 diverse navigation tasks in highly realistic indoor environments.
2. A comprehensive evaluation of VLM based high-level policies combined with low-level navigation policies, analyzing their performance and limitations in memory-intensive tasks.
3. A systematic and extensible evaluation framework with metrics that enable future advancements in memory-efficient embodied agents.

6.2 Related Work

Table 6.1: Comparison of *FINDINGDORY* with popular memory-focused embodied AI benchmarks.

Benchmark	Photorealism	Semantic Reasoning	Task Specification	Template Categories	# Resulting Tasks	Memory Length (π^*)	Isolates Memory
MemoryMaze [118]	✗	✗	One-hot	✗	1	500-1000 steps	✗
MemoryGym [120]	✗	✗	One-hot	✗	3	128-512 steps (Extensible)	✗
MultiON [164]	✓	✗	One-hot	✗	1	500 steps	✗
SIF [105]	✓	✓	Language	3	240 (val/test)	✗ (Uses semantic map)	✗
OpenEQA (Active) [101]	✓	✓	Language	✗	1600	75 steps	✗
GoatBench [76]	✓	✓	Language + Image	✗	1800 – 3600 (val)	500 steps (Static Env)	✗
Excalibur [189]	✓	✓	Language	11	21	Requires exploration	✗
Ours	✓	✓	Language	11	≈ 6000 (val)	500-3500 steps (Extensible)	✓

In this section, we review related work on long-video question answering and embodied AI benchmarks that incorporate historical experience in their problem formulation. We defer the discussion of different approaches for augmenting policies with memory-like mechanisms to Appendix D.3.1.

6.2.1 Video QA benchmarks

Recent analyses of video QA benchmarks [103, 21] show that many tasks can be solved by attending to key frames, not full video sequences, suggesting that current benchmarks may favor keyframe selection over evaluating long-term memory. Long-context evaluations for foundation models also focus on “needle-in-a-haystack” tasks, evaluating retrieval of specific details rather than integrating information across many segments [185]. While recent video benchmarks feature more challenging questions [103, 90, 161, 47, 145, 144, 89, 124, 178, 187, 101], they adopt multiple-choice QA formats which are not suitable for evaluating fine-grained decision-making over long histories. They may also allow models to succeed through guessing and are not extensible since they require substantial human annotation effort.

6.2.2 Embodied AI benchmarks

Various benchmarks have been proposed to evaluate multi-modal language models in embodied settings where agents must leverage their experience in environments to improve future decision-making [164]. Many focus on embodied question answering (EQA), such as HM-EQA [125], which requires active exploration, and Open-EQA [101], which tests VLMs in a passive setting with pre-collected episode histories. Notably, Open-EQA found that a *blind* LLM baseline—one that ignores visual input—still performed well, suggesting that multiple-choice evaluation can introduce biases and reduce the need for genuine memory recall. Other benchmarks study the situated instruction following setting where agents must gather information to interpret ambiguous instructions [131, 96, 105]. However, many allow agents to sidestep memory challenges by precomputing relevant objects and locations based on recent history, effectively turning potentially memory-intensive tasks into structured

instruction-following problems. Memory-Maze and Memory-Gym [118, 120] focus on partially observable decision-making with history-based state modeling, but their simplistic visuals and small state spaces limit their relevance to complex real-world memory settings.

In Table 6.1, we compare our proposed benchmark to the most related benchmarks within embodied AI and RL research with respect to the following desired properties:

1. **Photorealism:** Measures the visual complexity of environments, affecting the meaningful evaluation of foundation models trained on real-world data.
2. **Scope of semantic reasoning needed over memory:** Distinguishes between benchmarks that require agents to recall diverse semantic information versus a narrow set of attributes such as object locations.
3. **Task Specification:** Evaluates whether tasks are defined using flexible language-based descriptions, enabling richer memory challenges.
4. **Memory horizon length:** Adapting the definition from [110], this measures the temporal scope over which memory must be maintained for optimal task performance. Unlike static benchmarks requiring localized frame recall, dynamic environments demand reasoning over events distributed across extended observation sequences. This metric becomes difficult to quantify when tasks require substantial exploration phases.
5. **Memory isolation:** Whether benchmark performance isolates memory evaluation from confounding factors, particularly exploration requirements, which represent orthogonal challenges to memory recall and reasoning.

Our benchmark isolates memory evaluation by avoiding confounding related to exploration and focuses on careful and extensible task curation with automated metrics, requiring agents to recall and reason over long-horizon environmental observations.

6.3 FindingDory

In this section, we describe the design and construction of our embodied memory benchmark, FINDINGDORY. We first outline the benchmark structure, then detail the constituent tasks and simulation setup.

6.3.1 Benchmark Overview

The goal of FINDINGDORY is to evaluate an agent’s ability to retain and utilize past experiences to efficiently complete future tasks. Unlike standard navigation or perception-based evaluations, our benchmark explicitly tests memory by structuring tasks around past interactions rather than only immediate observations. To achieve this, we design a two-phase setup consisting of an *experience collection phase*, where scripted oracle agents autonomously gather experiences in the environment, followed by an *interaction phase*, where they are evaluated on their ability to recall and act upon that history. This decomposition allows for a controlled and standardized evaluation of memory, ensuring that the agent’s ability to remember past interactions is isolated from its ability to explore effectively.

The experience collection phase introduces dynamism into the environment by having the agent interact with and rearrange different objects. This ensures that the history contains meaningful state changes, allowing us to construct tasks that require spatio-temporal reasoning. Using oracle information from the simulator, the agent picks up objects from one location and places them onto designated receptacles (e.g., a table, chair, or bed). Episodes vary in length, spanning 400 to 3500 frames, with 2 to 11 object pick-and-place interactions (further details in appendix D.2.2).

Once the experience collection phase ends, the interaction phase begins. Here, a task is sampled from a predefined set of templated instructions, requiring the agent to recall and act upon the state of the environment from the previous phase. The agent is provided with access to its full history, including recorded images, pose, and past actions, and is tested on its ability to integrate long-term memory into

decision-making.

Due to the photorealistic visuals and object diversity of the Habitat simulator, FINDINGDORY provides a practical and transferable setting for evaluating large-scale vision-language models. Since many VLMs are pretrained on real-world images, they can readily interpret frames from this environment—enabling realistic memory evaluations in simulation, where long-horizon, multimodal interaction data can be collected at scale. This makes FINDINGDORY especially suitable for studying how current foundation models integrate memory into decision-making.

Table 6.2: Task Categories with Example Prompts and their Challenges. See appendix D.6 for detailed examples.

	Task Category	Example Instructions	Memory Requirement
	Object Recall	Navigate to a <i>{category}</i> .	Object was previously seen or needs exploration.
	Interaction	Navigate to any object that you did not interact with yesterday.	Differentiate between interacted & uninteracted objects.
Spatial	Conditional Interaction	Navigate to a <i>{receptacle_category}</i> you picked an object from.	Object categories, object-receptacle relationships.
	Object Attribute	Navigate back to a <i>{target_color}</i> colored object that you interacted with yesterday.	Object attributes other than category.
	Spatial Relationship	Navigate to the object which you interacted with which is the farthest from your current location.	Spatial relationships to objects.
	Room Visitation	Navigate to a room that you did not visit yesterday.	Room boundaries and past interactions.
Temporal	Interaction Order	Navigate to the object you interacted with immediately after <i>{object_category}</i> .	Sequence of objects interactions.
	Time-Based	Navigate to the object that you interacted with at <i>{HH:MM}</i> yesterday.	Time of interaction & final position of objects.
	Duration Tracking	Navigate to the object which took the longest time to rearrange.	Duration of interactions.
Multi-Goal	Unordered Revisitation	Revisit all the receptacles you picked objects from yesterday.	Past locations of multiple objects and best visitation order.
	Ordered Revisitation	Revisit all the objects you interacted with yesterday in specific order.	Similar to unordered revisitation but with strict sequencing.

6.3.2 Tasks

Our benchmark is designed to evaluate a broad spectrum of memory challenges faced by embodied agents, moving beyond narrow tasks that could be solved using simple heuristics or non-generalizable strategies. We define several **templated tasks** that evaluate an agent’s ability to integrate and recall different forms of memory across **spatial**, **temporal**, and **semantic** dimensions. Some tasks require recalling specific objects or receptacles, tracking **when** (e.g., temporal ordering) and **how** (e.g., interaction type: picked vs. placed). Others demand awareness of room-level context (e.g., rooms visited, farthest location) or semantic attributes (e.g., shape, color, material, functionality). We also introduce **time-usage memory** tasks, such as identifying the room that took the longest to explore or the object with the longest rearrangement duration. Task instructions are generated by randomly sampling scene-specific entities, and the full list of 60 templates is shown in Table D.1.

A smaller subset of **attribute-based evaluations** is also included to assess whether agents can flexibly reason over visual and semantic memory. These tasks involve prompts that refer to objects indirectly via properties observed during prior interaction (e.g., “go to the striped object you interacted with previously”). This setup helps differentiate methods capable of recalling fine-grained visual details from those relying solely on static representations, such as semantic maps [27] or textual summaries [67], which often struggle to flexibly reconstruct such information.

Single vs. Multiple Goals. The majority of tasks in FINDINGDORY involve single-goal instructions, where the agent must reason over its experience to identify a target and navigate to it. A smaller subset contains multi-goal tasks that require reaching multiple destinations, sometimes in a specific order. Single-goal tasks can be equally challenging, often requiring agents to recall what was *not* interacted with or to trace specific object movements through rearrangement states—demanding more than simple recall from static frames.

Extensibility through Procedural Generation. Our task suite is procedurally extensible, allowing difficulty to scale with changes in the experience collection

phase. Increasing the number of pick-and-place interactions, for instance, expands the temporal reasoning space and creates more challenging memory dependencies.

Task Categorization. We group tasks into interpretable categories to analyze performance trends. Table 6.2 presents these categories along with representative instructions and the memory challenges they embody. To our knowledge, FINDINGDORY offers the most visually complex and extensible suite of embodied memory tasks to date, covering a wide range of interaction-driven and long-horizon reasoning scenarios.

6.3.3 Implementation details

Simulation Setup and Dataset Composition. We build FINDINGDORY using the Habitat simulator [132, 149] and scenes from the Habitat Synthetic Scenes Dataset (HSSD)[73]. Our benchmark uses 107 scenes from the training split and 30 from the validation split, resulting in 1,478 training and 100 validation episodes. We populate these scenes with the same object sets used in OVMM[180], combining assets from AI2Thor [78], Amazon-Berkeley Objects [35], Google Scanned Objects [41], and HSSD [73].

Across these episodes, we include 839/247 object instances spanning 84/72 distinct categories in the `train/val` splits, respectively. Objects are rearranged during the experience collection phase to induce temporally grounded memory challenges. Additionally, agents interact with 17/16 distinct receptacle categories in the `train/val` splits (e.g., tables, chairs, beds). To support attribute-based tasks (see Table 6.2), we generate semantic descriptions of object instances using GPT-4o. We prompt the model with multi-view images of each object to extract attributes such as shape, color, material, and functionality. These descriptions are used to populate the object attribute task templates (5 total), which are limited to the validation set and undergo manual quality review.

In total, our benchmark includes 82,174 training and 5,876 validation task instances. Details on the procedural generation pipeline for episodes and instructions are

provided in Appendix D.2.

Agent Setup. We adopt the Hello Robot Stretch embodiment [71] in our simulation. The agent is equipped with RGB and depth sensors (resolution 640×480) and a GPS+Compass sensor. The agent can execute one of four discrete navigation actions: `MOVE_FORWARD` (0.25m), `TURN_LEFT` / `TURN_RIGHT` (by 10°) and a `STOP` action. During the experience collection phase, the agent can also manipulate its arm joints (extension, lift, yaw, pitch, roll) and activate a `manipulation_mode` for pick-and-place actions, which rotates the base (90°) and directs the head camera toward the gripper.

PDDL-based Task Verification. Since we operate with a large and diverse set of instruction templates (see Table 6.2), we leverage the PDDL [2] specification system as instantiated in [150]. For each instruction template, we create a template goal condition that uses various predicate expressions such as `is_robot_at_object(x, "OBJECT"` or `is_robot_at_receptacle(x, "RECEPTACLE"` where `x` is some placeholder variable. These placeholder variables are sampled based on various constraints such as `target_color` or `interaction_order` that are defined through the template. Together, the placeholder constraints and goal conditions enable us to create scalable, automated verification routines for each task in `FINDINGDORY`.

6.4 Experiments

We evaluate several baseline approaches on `FINDINGDORY` and assess their ability to complete memory-intensive tasks using a combination of vision-language reasoning and navigation. Below, we describe the baseline architecture, implementation variants, and evaluation metrics.

6.4.1 Evaluated Approaches

To establish strong baselines, we adopt a hierarchical architecture (fig. D.7) comprising two modules:

- (1) a high-level goal selection module, which processes the interaction history to

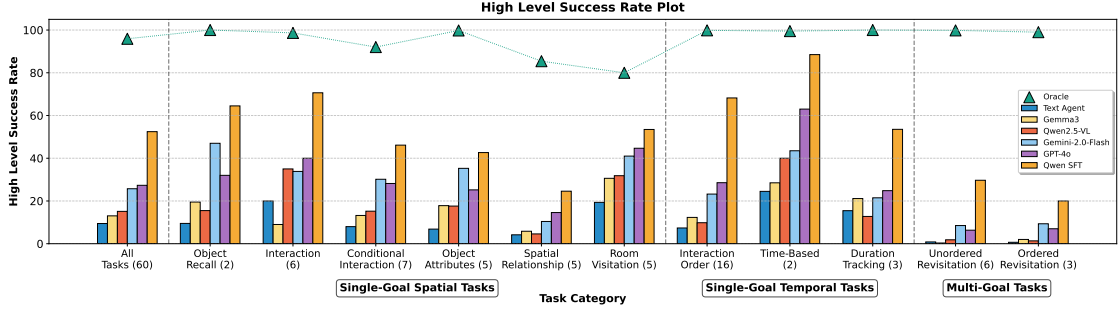


Figure 6.1: FINDINGDORY Task Performance: Proprietary VLMs such as Gemini-2.0-flash and GPT-4o fail to perform appreciably across most FINDINGDORY task categories. Agents struggle significantly on multi-goal tasks where multiple subgoals are to be achieved sequentially based on the interaction videos collected during Phase 1 of the task (sometimes in a specific order). The Qwen-SFT baseline (see appendix D.4.5) which is trained to predict ground-truth frame indices performs best but still reaches only $\approx 50\%$ on the high-level tasks. This highlights a considerable gap in the spatio-temporal reasoning capabilities of frontier models.

select goal frames, and

(2) a low-level navigation policy, which executes actions to reach the selected frame.

This architecture enables reasoning over long-horizon multimodal experience while delegating fine-grained control to a learned policy. Inspired by prior work [176], we avoid directly predicting actions using VLMs due to their poor performance in continuous control, and instead restrict them to high-level goal prediction.

High-Level Goal Selection

Video LM Agent. This agent receives the full interaction video along with a task instruction. Each frame in the video is annotated with its index and the time of day at which it was captured. The agent is tasked with predicting the index of a frame from the video that corresponds to a viable goal state. Specifically, the predicted frame should represent a viewpoint from which, if the agent were to navigate back to the same camera position, it could successfully complete the task. Although the model outputs a single frame index, selecting the correct one often requires reasoning over multiple temporally scattered observations, including frames where the agent may not be actively interacting. Since different VLMs are capable of handling different numbers of frames in context, we subsample the frames passed to each model based on what empirically performs best. We evaluate both

University of Oxford

proprietary (Gemini-2.0-Flash [152], GPT-4o) and open-source (Qwen2.5-VL [13], Gemma-3 [153]) vision-language models capable of processing long video contexts. See appendices D.4.1 to D.4.3 for more details.

Textual Memory Agent. Similar to [105, 10], we also consider an explicit, memory-building baseline which breaks down the video frames into chunks and uses a VLM to generate textual summaries for each video chunk. After “captioning” the entire video sequence, the VLM selects frame indices by reasoning over the generated textual descriptions. See appendix D.4.4 for more details.

Supervised Fine-Tuning (SFT). We additionally explore a supervised fine-tuning approach where the high-level VLM is trained to predict goal frame indices, given the task prompt and interaction history. Since multiple frames may validly represent the goal (e.g., several consecutive frames showing an object being placed), we supervise the model on the full list of acceptable target frames. At test time, a single frame is sampled from the list of predictions and is considered correct if it matches any of these valid frames. This setup enables the model to learn task-specific cues from examples and serves as a stronger baseline than zero-shot prompting, particularly for tasks requiring subtle temporal grounding. We provide additional details in appendix D.4.5.

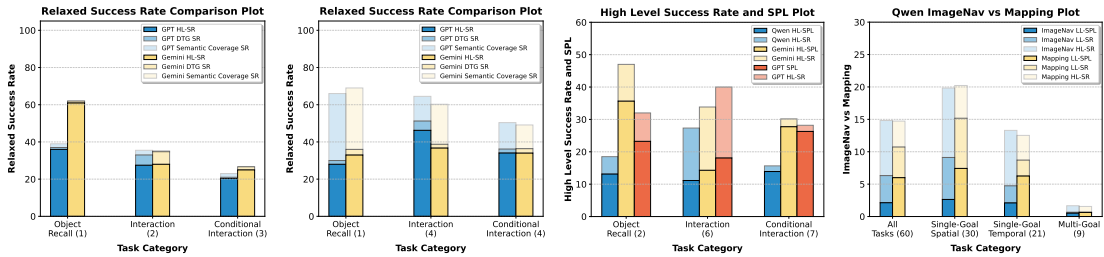
Low-Level Navigation

To execute the high-level plan, we use a low-level policy that takes the selected goal frame and outputs discrete actions: `MOVE-FORWARD`, `TURN-RIGHT`, `TURN-LEFT`, and `STOP`. Our primary policy is the LSTM-based controller from OVRLv2 [177], trained to reach image goals from egocentric RGB-D observations (see Appendix D.5.1).

We also evaluate a mapping-based policy that deterministically navigates to the location corresponding to the goal frame using a global map, without relying on learned visual features (Appendix D.5.2).

Solvability. Since the action space of our high-level baseline is constrained to frames observed during the experience collection phase, there are tasks for which

this is insufficient. For example, in the task “Navigate to a room that you did not visit yesterday,” all available waypoints correspond to previously visited locations, making it impossible to satisfy the instruction regardless of the selected frame. To establish an upper bound on task difficulty, we quantify the percentage of episodes that could theoretically be solved through optimal goal selection from interaction history, assuming perfect navigation (teleportation) to the selected waypoint. This upper bound is the dashed line (Oracle) in fig. 6.1.



(a) Relaxed metrics for Objects. (b) Relaxed metrics for Receptacles. (c) HL-SR vs HL-SPL Comparison. (d) Hierarchical policy results.

Figure 6.2: (a, b) Relaxed metrics for objects and receptacles tasks. We find that the models significantly improve in relaxed metrics for tasks involving receptacles while tasks targeting smaller objects do not exhibit similar improvements. **(c) Success Rate vs SPL performance** On tasks in which the VLM has the option to choose between multiple correct frames, we find that all models choose suboptimal frames sometimes resulting in the reduced HL-SPL metric. **(d) Hierarchical policy results on FindingDory** When Qwen is combined with an ImageNav or Mapping agent, the overall SR and SPL reduces significantly (seen in the LL-SR/LL-SPL metrics).

6.4.2 Evaluation Metrics

We benchmark agent performance on the FINDINGDORY tasks using **Low Level Success Rate** (LL-SR), assessing whether the agent finds the right target entities, and **Low Level Success-weighted-by-Path-Length** (LL-SPL), evaluating if the agent selects and navigates to the optimal entity using the shortest possible path. Since we use a hierarchical baseline, we additionally introduce a **High-Level Policy Success Rate** (HL-SR), and **High-Level Policy SPL** (HL-SPL) which measures the accuracy and efficiency of the high-level policy in selecting the correct and closest goal frames for navigation. To further analyze failure modes, we also report two relaxed variants: **Distance-to-Goal-Only SR** (DTG-SR), which relaxes the requirement that the target object must be visible in the selected frame, and

Semantic Coverage SR (SC-SR), which ignores spatial proximity. Additional metrics, success criteria, and threshold specifics are provided in appendix D.2.3.

6.5 Results

In this section, we analyze the performance of the outlined baselines on FINDING-DORY, presenting results grouped according to the categories described in Section 6.3.2.

State-of-the-Art VLMs struggle on the FindingDory benchmark. From fig. 6.1, we observe consistently low success rates on all task subsets across baselines, with GPT-4o achieving the highest performance at 27.3%, Gemini-2.0-Flash reaching 25.7%, while Qwen2.5-VL and Gemma-3 score in the 13–16% range. The text-based agent performs the worst overall, underscoring that textual memory representations alone are insufficient for disambiguating visual goals or reasoning over complex visual histories. Notably, the supervised fine-tuned (SFT) variant outperforms all frozen VLMs, achieving an average improvement of 25% across tasks, with considerable improvement on temporal and multi-goal tasks.

In single-goal tasks, all methods struggle particularly with the spatial relationship subset, with even GPT-4o achieving only 14.5% success. These tasks require reasoning about relative distances between entities and the agent—an ongoing challenge noted in prior work [178]. In contrast, room identification tasks yield the highest performance, likely due to the small number of distinct rooms and the lenient goal condition of reaching any location within the correct room. Performance on conditional interaction and object attribute tasks is comparable, suggesting that current VLMs handle both category-based queries (e.g., “apple”) and attribute-based descriptions (e.g., “red food”) with similar effectiveness.

Temporal tasks pose additional challenges to the existing VLMs, with models consistently failing to identify the correct order of interactions—requiring multi-hop reasoning. Duration tracking is similarly difficult as it requires inferring elapsed duration based on specific points in the video. In contrast, for the time-based tasks

we observe that baselines perform better as these tasks only require single frame retrieval without additional reasoning.

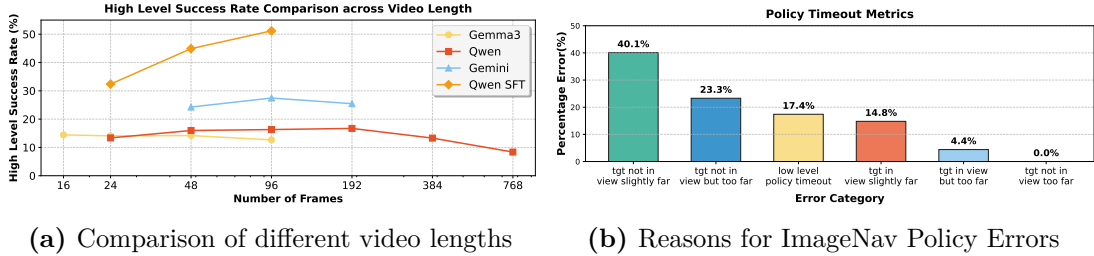


Figure 6.3: (a) Model performance at different number of subsampled video frames. (b) Reasons for ImageNav Policy Errors.

Poor performance on multi-goal tasks. From fig. 6.1, we observe that performance of each VLM baseline that we consider is close to zero on most multi-goal tasks. This suggests that VLMs find it hard to track and remember multiple object-receptacle interactions. For the multi-goal evaluations, we use a single-shot prompt and require the high level VLM agent to predict all target frame indices in one single response - primarily to avoid multiple inferences with long sequence lengths due to large number of frames. Qualitatively, we observe that the VLM generally fails to even predict the correct number of targets required to solve the multi-goal task. Our trained Qwen model performs slightly better, achieving close to 20% accuracy, but still falls significantly short of the achievable result of 99%.

VLMs can identify target receptacles but not precisely locate them spatially. We conduct an experiment to analyze the performance differences of VLMs between tasks involving objects versus receptacle entities. We extract a subset of FINDINGDORY tasks, categorizing them based on whether the valid entities are objects or receptacles. From fig. 6.2b, we observe that the relaxed **Semantic Coverage-SR (SC-SR)** metrics (see Appendix D.2.3 for details) significantly increases for tasks involving receptacle entities. The marked increase in high-level success rates with relaxed distance threshold (**DTG-SR**) suggests that VLMs are generally able to identify frames from which large entities (receptacles) can be viewed in the image but cannot accurately identify specific frames which are spatially closest

to the target entity. In contrast, tasks targeting smaller objects do not exhibit the same improvement (see fig. 6.2a), as fewer number of frames clearly show these small objects from farther distances. This highlights a significant limitation of current VLMs in constructing effective spatial representations to accurately localize target entities from the provided observation stream. We explicitly prompt high-level VLM agents to identify frames spatially closest to target entities, essential for task success (see figs. D.2 and D.4 for example prompts).

VLMs struggle with identifying the closest entity that solves the task. In Fig. 6.2c we report the HL-SPL metric (Section 6.4.2) which captures the efficiency with which the high level VLM-based modules select goal frames. During the experience collection phase, the agent can interact with multiple object instances of the same category due to which a task such as “*Navigate to a squeezer you interacted with*” could have multiple valid entities that the agent can navigate to. We observe that for such tasks there is a wide gap between HL-SR and HL-SPL (upto 50% in case of interaction-based tasks) for all model variants. This suggests that although the VLMs can correctly recognize a valid entity to solve the task, they still fail to perform a fine-grained spatial analysis of the multiple valid entities being rearranged dynamically in the video. Thus, they cannot spatially locate the valid entity nearest to its current position.

Policy decomposition causes performance degradation. To benchmark the hierarchical agent proposed in section 6.4.1, we experiment using Qwen2.5-VL-7B as the high-level VLM module and image-goal navigation policy and mapping policy as the low-level agents. The low-level policy activates only when the high-level VLM correctly selects the subgoal. As depicted in fig. 6.2d, the Low-Level-SR (LL-SR) metric significantly drops across all FINDINGDORY tasks for the image-goal navigation policy. We attribute this decline to a distribution shift in the visual goals provided to the navigation policy. During training, the policy encounters randomly sampled start and end goal poses, differing substantially from those selected by VLMs during evaluation. Successful VLM-selected frames typically correspond closely to pick place interaction routines, resulting in image goals predominantly

containing background walls and partial views of receptacle surfaces. Consequently, this limited visual context hinders effective navigation. In comparison, the mapping based navigation policy performs better, having only a 25% relative drop in SR. The SPL of both the methods remain in the single digits. This emphasizes the importance of the FINDINGDORY benchmark in revealing these gaps which motivates the development of more reliable memory and navigation strategies

Frame Subsampling. We analyze how the number of input frames affects high-level performance by evaluating agents on subsampled interaction videos of varying lengths (fig. 6.3a). This helps assess whether models benefit from longer histories or simply overfit to sparse visual cues. Interestingly, we observe that frozen VLMs—despite being capable of accepting long contexts—do not improve with more frames. In fact, their performance often degrades at higher frame counts (minimal subsampling), suggesting that they struggle to extract relevant information from densely packed, unfiltered input. This highlights a key limitation: naïvely increasing context length does not help unless the model can effectively attend to localized events.

In contrast, the fine-tuned model (Qwen SFT) demonstrates clear gains when trained with longer videos, leveraging additional context to improve reasoning. This indicates that with appropriate supervision, models can move beyond shallow matching and utilize richer temporal signals from the interaction history.

Sanity checks. We investigate how a VLM would perform at short-horizon visual comprehension on images from the same episodes used for long-horizon evaluation in our benchmark. This tests if poor performance of the evaluated models on our long-horizon reasoning tasks is due to a domain or sim2real gap. We assess this by sampling frames around object interactions, asking models to list out detected objects, and check accuracy compared to the ground truth category via string matching – a conservative metric that misses correct predictions using synonyms or alternative phrasings. This still results in image sequences of length 50-150, but Gemini achieves an accuracy of 67% which is significantly higher than it’s average high level success

rate across tasks in fig. 6.1. This shows that despite clear, close-up views, models struggle to reason over them when temporal sequence comprehension is required.

Low level navigation policy failures. From fig. 6.3b, we observe that the low-level image-goal navigation policy can fail due to invoking the termination (stop) action incorrectly or a navigation routine timeout (max budget of 1000 steps including the data collection phase). To further understand this behavior, we analyze if the navigation policy actually looks at the target entity when the stop action is invoked. Interestingly, we find that in several instances the target entity is visible from a slightly larger distance threshold (2.0m) from the chosen task success threshold (1.0 m, in case of receptacles). This often happens when VLMs select goal frames in which the target entity is visible in the 2D image but not necessarily closest in 3D space, and thus the goal frame coordinates are slightly further away than the threshold distance.

6.6 Conclusion

We introduced FINDINGDORY, a benchmark for evaluating long-horizon memory in embodied agents within a highly photorealistic indoor simulation. Our findings reveal limitations in VLM context scaling, particularly in handling long observation sequences required for memory-intensive tasks. FINDINGDORY provides informative metrics that disentangle different aspects of memory reasoning, allows for procedural task generation of increasing difficulty, and fine-grained, efficiency-focused failure analysis across high-level semantic retrieval and low-level spatial reasoning. Its extensibility allows for longer temporal dependencies, supporting research on context scaling and memory efficiency in multimodal models. In our results, we observe that the supervised fine-tuning baseline performance improves consistently when trained with longer videos which suggests future work should explore finetuning VLMs with effective memory compression techniques for enhanced spatio-temporal reasoning.

Our experiments also highlight challenges with hierarchical policies that integrate

long-context VLMs with low-level control. While high-level policies specify abstract goals—such as referencing a past frame in the interaction history—low-level navigation policies often struggle to reach the intended object precisely, exposing a gap between goal specification and reachability. Addressing this limitation requires better integration between high-level reasoning and low-level execution, either by improving hierarchical policies to leverage VLM guidance more effectively or by training end-to-end models that jointly learn memory retrieval and control.

Beyond evaluation, FINDINGDORY can serve as a training platform for memory-dependent tasks, enabling long-context-aware models and enabling dataset extraction for supervised learning tasks such as VQA over long videos. By bridging the gap between VLM advancements and real-world memory demands, FINDINGDORY facilitates systematic study of memory-augmented policy learning and long-term reasoning, driving progress in multimodal embodied AI.

7 | Discussion and Future Directions

This thesis explored three leverage points for building data-, compute-, and memory-efficient decision-making agents: *active sampling*, *representation reuse*, and *memory-guided control*. We close by situating these contributions within the rapidly evolving landscape of foundation models and outline open questions necessary for transitioning these proofs-of-concept into robust, real-world systems.

7.1 Active Sampling in the Scaling Era

From scaling to curation. Historically, uniformly sampling ever-larger datasets seemed like a straightforward method to enhance model performance. However, recent research has revealed diminishing returns from naive scaling and shown that intelligently curating training examples significantly improves efficiency [146, 46]. Our uncertainty-based active sampling approach (Chapter 3) embodies this perspective in control domains, replaying transitions based on ensemble uncertainty to pinpoint causally informative samples.

Looking forward. Large-scale pre-training methods are increasingly incorporating data curation—either by carefully selecting high-quality subsets [45] or by dynamically re-weighting training data to mitigate biases [46]. As foundation models are increasingly fine-tuned on specialized behavior datasets, which often exhibit long-tailed distributions and spurious correlations, adaptive sampling and data curation become critical. In reinforcement learning from human feedback (RLHF), models receive targeted feedback on their own outputs, allowing selective curation of experiences to address biases and enhance generalization. Similarly, adaptive data curation in fine-tuning leverages selective sampling to manage biases and improve model performance on specialized datasets. Key open questions include:

- Designing acquisition functions suitable for datasets with long-horizon interactions, emphasizing samples with meaningful long-range dependencies.

- Integrating active sampling with fine-tuning workflows for foundation models, particularly to handle spurious correlations effectively.

7.2 Representation Reuse in Embodied Control

Feature aggregation for embodied tasks. Our findings with Stable Control Representations (SCR, Chapter 4) highlight the critical role of how visual features from multi-layer encoders are combined. Proper feature aggregation significantly enhances spatial precision crucial for manipulation and navigation tasks. Similar insights have emerged from vision–language models (VLMs), where fusion modules that incorporate multiple encoder layers, such as cross-attention streams in Flamingo, have proven beneficial for tasks requiring spatial reasoning [56, 91].

Open problems. Future research must determine effective strategies for fine-tuning entire multimodal encoders, not just selecting subsets of frozen features. Critical open questions include:

- How can we effectively and efficiently fine-tune large encoders end-to-end within realistic GPU constraints?
- What approaches to multi-layer projection and fusion optimally balance performance gains with computational feasibility?

7.3 Memory and Long-Context Reasoning

Transformer context windows have dramatically expanded, but applying such extensive contexts at real-time inference speeds remains challenging for embodied agents. Memo (Chapter 5) demonstrated a practical approach using periodic summarization tokens to achieve sub-linear growth in memory usage, complementing efforts with state-space models, recurrent memory transformers, and retrieval-augmented networks [28, 63].

Future steps. Future research should address:

- **Long-horizon memory optimization.** Determining what information to retain or summarize when the usefulness of data only becomes apparent after extensive sequences of actions.
- **Retrieval integration.** Assessing the necessity and efficiency of retrieval-based components as interaction histories grow.
- **Realistic benchmarking.** Expanding datasets like FindingDory (Chapter 6) to include large-scale, open-world scenarios with realistic long-range dependencies.

7.4 The Road Ahead

This thesis advocates for integrated approaches that efficiently select relevant experiences, leverage pre-existing knowledge, and effectively retain critical contextual information. Extending these principles suggests several future research directions:

- **Closed-loop data curation.** Integrating active sampling into fine-tuning processes for foundation models, allowing continuous refinement based on task-specific data distributions.
- **Foundation model representations for embodied control.** Leveraging representations from foundation visual models, particularly those derived from self-supervised learning (SSL), to enhance spatial reasoning and task performance in embodied agents.
- **World modeling and representation learning.** Clarifying the role of world models in embodied control. Given the limitations of current world models like diffusion models in reliably simulating long-horizon agent interactions, future work could instead focus on exploiting these models primarily for their rich learned representations, as demonstrated by our proof-of-concept with diffusion-derived features.
- **Scalable memory architectures.** Developing memory management strategies that scale sub-linearly with the length of interaction history, utilizing

University of Oxford

consolidation, summarization, and selective retrieval to manage lifelong agent deployments.

Advancements along these lines will move us closer to practical, lifelong learning agents capable of robust, efficient, and reliable operation in diverse, real-world environments.

Bibliography

- [1] Jason Abrevaya, Yu-Chin Hsu, and Robert P. Lieli. Estimating conditional average treatment effects. *Journal of Business & Economic Statistics*, 33(4): 485–505, 2015. doi: 10.1080/07350015.2014.975555. URL <https://doi.org/10.1080/07350015.2014.975555>.
- [2] Constructions Aeronautiques, Adele Howe, Craig Knoblock, ISI Drew McDermott, Ashwin Ram, Manuela Veloso, Daniel Weld, David Wilkins Sri, Anthony Barrett, Dave Christianson, et al. Pddl| the planning domain definition language. *Technical Report, Tech. Rep.*, 1998.
- [3] Rishabh Agarwal, Dale Schuurmans, and Mohammad Norouzi. An optimistic perspective on offline reinforcement learning. In *International Conference on Machine Learning*, 2020.
- [4] Rishabh Agarwal, Dale Schuurmans, and Mohammad Norouzi. An optimistic perspective on offline reinforcement learning. In Hal Daumé III and Aarti Singh, editors, *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 104–114. PMLR, 13–18 Jul 2020. URL <https://proceedings.mlr.press/v119/agarwal20c.html>.
- [5] Rishabh Agarwal, Max Schwarzer, Pablo Samuel Castro, Aaron C Courville, and Marc Bellemare. Deep reinforcement learning at the edge of the statistical precipice. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, volume 34, pages 29304–29320. Curran Associates, Inc., 2021.
- [6] Anurag Ajay, Seungwook Han, Yilun Du, Shaung Li, Abhi Gupta, Tommi Jaakkola, Josh Tenenbaum, Leslie Kaelbling, Akash Srivastava, and Pulkit Agrawal. Compositional foundation models for hierarchical planning. *arXiv preprint arXiv:2309.08587*, 2023.

- [7] Jean-Baptiste Alayrac, Jeff Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, Arthur Mensch, Katherine Millican, Malcolm Reynolds, et al. Flamingo: a visual language model for few-shot learning. *Advances in Neural Information Processing Systems*, 35:23716–23736, 2022.
- [8] Gaon An, Seungyong Moon, Jang-Hyun Kim, and Hyun Oh Song. Uncertainty-based offline reinforcement learning with diversified q-ensemble. In A. Beygelzimer, Y. Dauphin, P. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, 2021. URL <https://openreview.net/forum?id=ZUvaSolQZh3>.
- [9] Abrar Anwar, John Welsh, Joydeep Biswas, Soha Pouya, and Yan Chang. ReMEmbR: Building and Reasoning Over Long-Horizon Spatio-Temporal Memory for Robot Navigation. *arXiv e-prints*, art. arXiv:2409.13682, September 2024. doi: 10.48550/arXiv.2409.13682.
- [10] Abrar Anwar, John Welsh, Joydeep Biswas, Soha Pouya, and Yan Chang. Remembr: Building and reasoning over long-horizon spatio-temporal memory for robot navigation. In *8th Annual Conference on Robot Learning*, 2024.
- [11] Alexei Baevski, Yuhao Zhou, Abdelrahman Mohamed, and Michael Auli. wav2vec 2.0: A framework for self-supervised learning of speech representations. In *Advances in Neural Information Processing Systems*, 2020. URL <https://arxiv.org/abs/2006.11477>.
- [12] Chenjia Bai, Lingxiao Wang, Zhuoran Yang, Zhi-Hong Deng, Animesh Garg, Peng Liu, and Zhaoran Wang. Pessimistic bootstrapping for uncertainty-driven offline reinforcement learning. In *International Conference on Learning Representations*, 2022. URL <https://openreview.net/forum?id=Y4cs1Z3HnqL>.
- [13] Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibao Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, Humen Zhong, Yuezhi Zhu, Mingkun Yang, Zhaohai Li, Jianqiang Wan, Pengfei Wang, Wei Ding, Zheren Fu, Yiheng Xu, Jiabo Ye, Xi Zhang, Tianbao Xie, Zesen Cheng, Hang Zhang,

- Zhibo Yang, Haiyang Xu, and Junyang Lin. Qwen2.5-vl technical report, 2025. URL <https://arxiv.org/abs/2502.13923>.
- [14] Dmitry Baranchuk, Andrey Voynov, Ivan Rubachev, Valentin Khruikov, and Artem Babenko. Label-efficient semantic segmentation with diffusion models. In *International Conference on Learning Representations*, 2022.
- [15] 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.
- [16] Jacob Beck, Risto Vuorio, Evan Zheran Liu, Zheng Xiong, Luisa Zintgraf, Chelsea Finn, and Shimon Whiteson. A survey of meta-reinforcement learning. *arXiv preprint arXiv:2301.08028*, 2023.
- [17] Sara Beery, Grant Van Horn, and Pietro Perona. Recognition in terra incognita. In Vittorio Ferrari, Martial Hebert, Cristian Sminchisescu, and Yair Weiss, editors, *Computer Vision – ECCV 2018*, pages 472–489, Cham, 2018. Springer International Publishing. ISBN 978-3-030-01270-0.
- [18] Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Joseph Dabis, Chelsea Finn, Keerthana Gopalakrishnan, Karol Hausman, Alex Herzog, Jasmine Hsu, Julian Ibarz, Brian Ichter, Alex Irpan, Tomas Jackson, Sally Jesmonth, Nikhil J Joshi, Ryan Julian, Dmitry Kalashnikov, Yuheng Kuang, Isabel Leal, Kuang-Huei Lee, Sergey Levine, Yao Lu, Utsav Malla, Deeksha Manjunath, Igor Mordatch, Ofir Nachum, Carolina Parada, Jodilyn Peralta, Emily Perez, Karl Pertsch, Jornell Quiambao, Kanishka Rao, Michael Ryoo, Grecia Salazar, Pannag Sanketi, Kevin Sayed, Jaspier Singh, Sumedh Sontakke, Austin Stone, Clayton Tan, Huong Tran, Vincent Vanhoucke, Steve Vega, Quan Vuong, Fei Xia, Ted Xiao, Peng Xu, Sichun Xu, Tianhe Yu, and Brianna Zitkovich. RT-1: Robotics Transformer for Real-World Control at Scale. *arXiv e-prints*, art. arXiv:2212.06817, December 2022. doi: 10.48550/arXiv.2212.06817.

- [19] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.
- [20] 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. Language models are few-shot learners, 2020.
- [21] Shyamal Buch, Cristóbal Eyzaguirre, Adrien Gaidon, Jiajun Wu, Li Fei-Fei, and Juan Carlos Niebles. Revisiting the “Video” in Video-Language Understanding. *arXiv e-prints*, art. arXiv:2206.01720, June 2022. doi: 10.48550/arXiv.2206.01720.
- [22] Jacob Buckman, Carles Gelada, and Marc G Bellemare. The importance of pessimism in fixed-dataset policy optimization. In *International Conference on Learning Representations*, 2021. URL <https://openreview.net/forum?id=E3Ys6a1NTGT>.
- [23] Aydar Bulatov, Yuri Kuratov, Yermek Kapushev, and Mikhail S Burtsev. Scaling transformer to 1m tokens and beyond with rmt. *arXiv preprint arXiv:2304.11062*, 2023.
- [24] Aydar Bulatov, Yuri Kuratov, and Mikhail S. Burtsev. Scaling transformer to 1m tokens and beyond with rmt. In *Association for the Advancement of Artificial Intelligence*, 2024.
- [25] Mark Craven Burr Settles and Soumya Ray. Multiple-instance active learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2008.

- [26] Berk Calli, Arjun Singh, Aaron Walsman, Siddhartha Srinivasa, Pieter Abbeel, and Aaron M Dollar. The ycb object and model set: Towards common benchmarks for manipulation research. In *2015 international conference on advanced robotics (ICAR)*, pages 510–517. IEEE, 2015.
- [27] Matthew Chang, Theophile Gervet, Mukul Khanna, Sriram Yenamandra, Dhruv Shah, So Yeon Min, Kavita Shah, Chris Paxton, Saurabh Gupta, Dhruv Batra, Roozbeh Mottaghi, Jitendra Malik, and Devendra Singh Chaplot. GOAT: GO to Any Thing. *arXiv e-prints*, art. arXiv:2311.06430, November 2023. doi: 10.48550/arXiv.2311.06430.
- [28] Eugene Cheah and Dan Fu. 2024 in post-transformers architectures. 2024. Latent Space Blog.
- [29] Xi Chen, Xiao Wang, Soravit Changpinyo, AJ Piergiovanni, Piotr Padlewski, Daniel Salz, Sebastian Goodman, Adam Grycner, Basil Mustafa, Lucas Beyer, et al. Pali: A jointly-scaled multilingual language-image model. *arXiv preprint arXiv:2209.06794*, 2022.
- [30] Alexis Chevalier, Alexander Wettig, Anirudh Ajith, and Danqi Chen. Adapting language models to compress contexts. In Houda Bouamor, Juan Pino, and Kalika Bali, editors, *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 3829–3846, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-main.232. URL <https://aclanthology.org/2023.emnlp-main.232>.
- [31] Cheng Chi, Siyuan Feng, Yilun Du, Zhenjia Xu, Eric Cousineau, Benjamin Burchfiel, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. *arXiv preprint arXiv:2303.04137*, 2023.
- [32] Kyunghyun Cho, Bart van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation, 2014. URL <https://arxiv.org/abs/1406.1078>.

- [33] Karl Cobbe, Christopher Hesse, Jacob Hilton, and John Schulman. Leveraging procedural generation to benchmark reinforcement learning. *arXiv preprint arXiv:1912.01588*, 2019.
- [34] Jasmine Collins, Shubham Goel, Kenan Deng, Achleshwar Luthra, Leon Xu, Erhan Gundogdu, Xi Zhang, Tomas F Yago Vicente, Thomas Dideriksen, Himanshu Arora, Matthieu Guillaumin, and Jitendra Malik. Abo: Dataset and benchmarks for real-world 3d object understanding. *CVPR*, 2022.
- [35] Jasmine Collins, Shubham Goel, Achleshwar Luthra, Leon Xu, Kenan Deng, Xi Zhang, Tomas F Yago Vicente, Himanshu Arora, Thomas Dideriksen, Matthieu Guillaumin, et al. Abo: Dataset and benchmarks for real-world 3d object understanding. In *CVPR*, 2022.
- [36] Zihang Dai, Zhilin Yang, Yiming Yang, Jaime Carbonell, Quoc Le, and Ruslan Salakhutdinov. Transformer-XL: Attentive language models beyond a fixed-length context. In Anna Korhonen, David Traum, and Lluís Màrquez, editors, *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2978–2988, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1285. URL <https://aclanthology.org/P19-1285>.
- [37] Dima Damen, Hazel Doughty, Giovanni Maria Farinella, Sanja Fidler, Antonino Furnari, Evangelos Kazakos, Davide Moltisanti, Jonathan Munro, Toby Perrett, Will Price, and Michael Wray. Scaling egocentric vision: The epic-kitchens dataset. In *European Conference on Computer Vision (ECCV)*, 2018.
- [38] Pim de Haan, Dinesh Jayaraman, and Sergey Levine. Causal confusion in imitation learning. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL <https://proceedings.neurips.cc/paper/2019/file/947018640bf36a2bb609d3557a285329-Paper.pdf>.

- [39] Jacob Devlin. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.
- [40] Alexey Dosovitskiy, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun. CARLA: An open urban driving simulator. In *Proceedings of the 1st Annual Conference on Robot Learning*, pages 1–16, 2017.
- [41] Laura Downs, Anthony Francis, Nate Koenig, Brandon Kinman, Ryan Hickman, Krista Reymann, Thomas B McHugh, and Vincent Vanhoucke. Google scanned objects: A high-quality dataset of 3d scanned household items. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 2553–2560. IEEE, 2022.
- [42] Yilun Du, Mengjiao Yang, Bo Dai, Hanjun Dai, Ofir Nachum, Joshua B Tenenbaum, Dale Schuurmans, and Pieter Abbeel. Learning universal policies via text-guided video generation. URL <https://arxiv.org/abs/2302.00111>, 2023.
- [43] Ahmad Elawady, Gunjan Chhablani, Ram Ramrakhya, Karmesh Yadav, Dhruv Batra, Zsolt Kira, and Andrew Szot. ReLIC: A Recipe for 64k Steps of In-Context Reinforcement Learning for Embodied AI. *arXiv e-prints*, art. arXiv:2410.02751, October 2024. doi: 10.48550/arXiv.2410.02751.
- [44] Zackory Erickson, Vamsee Gangaram, Ariel Kapusta, C. Karen Liu, and Charles C. Kemp. Assistive gym: A physics simulation framework for assistive robotics, 2019. URL <https://arxiv.org/abs/1910.04700>.
- [45] Talfan Evans, Nikhil Parthasarathy, Hamza Merzic, and Olivier J. Henaff. Data curation via joint example selection further accelerates multimodal learning, 2024. URL <https://arxiv.org/abs/2406.17711>.
- [46] Talfan Evans, Shreya Pathak, Hamza Merzic, Jonathan Schwarz, Ryutaro Tanno, and Olivier Hénaff. Bad students make great teachers: Active learning accelerates large-scale visual understanding. In *Proceedings of CVPR*, 2024.
- [47] Chaoyou Fu, Yuhan Dai, Yondong Luo, Lei Li, Shuhuai Ren, Renrui Zhang,

- Zihan Wang, Chenyu Zhou, Yunhang Shen, Mengdan Zhang, et al. Video-mme: The first-ever comprehensive evaluation benchmark of multi-modal llms in video analysis. *arXiv preprint arXiv:2405.21075*, 2024.
- [48] Justin Fu, Aviral Kumar, Ofir Nachum, George Tucker, and Sergey Levine. D4RL: Datasets for Deep Data-Driven Reinforcement Learning. *arXiv e-prints*, art. arXiv:2004.07219, April 2020.
- [49] Robert Geirhos, Jörn-Henrik Jacobsen, Claudio Michaelis, Richard Zemel, Wieland Brendel, Matthias Bethge, and Felix Wichmann. Shortcut learning in deep neural networks. *Nature Machine Intelligence*, 2:665–673, 11 2020. doi: 10.1038/s42256-020-00257-z.
- [50] Seyed Kamyar Seyed Ghasemipour, Shixiang Shane Gu, and Ofir Nachum. Why so pessimistic? estimating uncertainties for offline RL through ensembles, and why their independence matters., 2022. URL <https://openreview.net/forum?id=wQ7RCayXUS1>.
- [51] Google. Explore vision capabilities with the gemini api. <https://ai.google.dev/gemini-api/docs/vision?lang=python>, 2025.
- [52] Raghav Goyal, Samira Ebrahimi Kahou, Vincent Michalski, Joanna Materzynska, Susanne Westphal, Heuna Kim, Valentin Haenel, Ingo Freund, Peter Yianilos, Moritz Mueller-Freitag, et al. The "something something" video database for learning and evaluating visual common sense. In *Proceedings of the IEEE international conference on computer vision*, pages 5842–5850, 2017.
- [53] Robert M. Gray. *Vector Quantization and Signal Compression*. Springer, 1984.
- [54] Jake Grigsby, Linxi Fan, and Yuke Zhu. AMAGO: Scalable in-context reinforcement learning for adaptive agents. In *The Twelfth International Conference on Learning Representations*, 2024. URL <https://openreview.net/forum?id=M6XWoEdmwf>.

- [55] Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In Jennifer Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 1861–1870. PMLR, 10–15 Jul 2018. URL <https://proceedings.mlr.press/v80/haarnoja18b.html>.
- [56] Xiaofeng Han, Shunpeng Chen, Zenghuang Fu, Zhe Feng, Lue Fan, Dong An, Changwei Wang, Li Guo, Weiliang Meng, Xiaopeng Zhang, Rongtao Xu, and Shibiao Xu. Multimodal fusion and vision-language models: A survey for robot vision, 2025. URL <https://arxiv.org/abs/2504.02477>.
- [57] Philippe Hansen-Estruch, Ilya Kostrikov, Michael Janner, Jakub Grudzien Kuba, and Sergey Levine. Idql: Implicit q-learning as an actor-critic method with diffusion policies. *arXiv preprint arXiv:2304.10573*, 2023.
- [58] Zifan He, Zongyue Qin, Neha Prakriya, Yizhou Sun, and Jason Cong. Hmt: Hierarchical memory transformer for long context language processing. *arXiv preprint arXiv:2405.06067*, 2024.
- [59] Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. In *NeurIPS 2021 Workshop on Deep Generative Models and Downstream Applications*, 2021.
- [60] Jonathan Ho, William Chan, Chitwan Saharia, Jay Whang, Ruiqi Gao, Alexey Gritsenko, Diederik P. Kingma, Ben Poole, Mohammad Norouzi, David J. Fleet, and Tim Salimans. Imagen video: High definition video generation with diffusion models, 2022.
- [61] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Comput.*, 9(8):1735–1780, November 1997. ISSN 0899-7667. doi: 10.1162/neco.1997.9.8.1735. URL <https://doi.org/10.1162/neco.1997.9.8.1735>.
- [62] Gao Huang, Yu Sun, Zhuang Liu, Daniel Sedra, and Kilian Q Weinberger. Deep networks with stochastic depth. In *Computer Vision—ECCV 2016: 14th*

European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part IV 14, pages 646–661. Springer, 2016.

- [63] Yunpeng Huang, Jingwei Xu, Junyu Lai, Zixu Jiang, Taolue Chen, Zenan Li, Yuan Yao, Xiaoxing Ma, Lijuan Yang, Hao Chen, Shupeng Li, and Penghao Zhao. Advancing transformer architecture in long-context large language models: A comprehensive survey, 2024. URL <https://arxiv.org/abs/2311.12351>.
- [64] Aaron Hurst, Adam Lerer, Adam P Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, et al. Gpt-4o system card. *arXiv preprint arXiv:2410.21276*, 2024.
- [65] Michael Janner, Yilun Du, Joshua B. Tenenbaum, and Sergey Levine. Planning with diffusion for flexible behavior synthesis. In *International Conference on Machine Learning*, 2022.
- [66] Andrew Jesson, Panagiotis Tigas, Joost van Amersfoort, Andreas Kirsch, Uri Shalit, and Yarín Gal. Causal-bald: Deep bayesian active learning of outcomes to infer treatment-effects from observational data. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, volume 34, pages 30465–30478. Curran Associates, Inc., 2021. URL <https://proceedings.neurips.cc/paper/2021/file/ffa4eb0e32349ae57f7a0ee8c7cd7c11-Paper.pdf>.
- [67] Kumara Kahatapitiya, Kanchana Ranasinghe, Jongwoo Park, and Michael S. Ryoo. Language Repository for Long Video Understanding. *arXiv e-prints*, art. arXiv:2403.14622, March 2024. doi: 10.48550/arXiv.2403.14622.
- [68] Ivan Kapelyukh, Vitalis Vosylius, and Edward Johns. Dall-e-bot: Introducing web-scale diffusion models to robotics. *IEEE Robotics and Automation Letters*, 2023.
- [69] Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei.

- Scaling laws for neural language models. *arXiv preprint*, 2020. URL <https://arxiv.org/abs/2001.08361>.
- [70] Siddharth Karamcheti, Suraj Nair, Annie S. Chen, Thomas Kollar, Chelsea Finn, Dorsa Sadigh, and Percy Liang. Language-driven representation learning for robotics. In *Robotics: Science and Systems (RSS)*, 2023.
- [71] Charles C Kemp, Aaron Edsinger, Henry M Clever, and Blaine Matulevich. The design of stretch: A compact, lightweight mobile manipulator for indoor human environments. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 3150–3157. IEEE, 2022.
- [72] 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 (CVPR)*, June 2022.
- [73] Mukul Khanna*, Yongsen Mao*, Hanxiao Jiang, Sanjay Haresh, Brennan Shacklett, Dhruv Batra, Alexander Clegg, Eric Undersander, Angel X. Chang, and Manolis Savva. Habitat Synthetic Scenes Dataset (HSSD-200): An Analysis of 3D Scene Scale and Realism Tradeoffs for ObjectGoal Navigation. *arXiv preprint*, 2023.
- [74] Mukul Khanna, Yongsen Mao, Hanxiao Jiang, Sanjay Haresh, Brennan Shacklett, Dhruv Batra, Alexander Clegg, Eric Undersander, Angel X. Chang, and Manolis Savva. Habitat Synthetic Scenes Dataset (HSSD-200): An Analysis of 3D Scene Scale and Realism Tradeoffs for ObjectGoal Navigation. *arXiv preprint*, 2023.
- [75] Mukul Khanna, Yongsen Mao, Hanxiao Jiang, Sanjay Haresh, Brennan Shacklett, Dhruv Batra, Alexander Clegg, Eric Undersander, Angel X Chang, and Manolis Savva. Habitat synthetic scenes dataset (hssd-200): An analysis of 3d scene scale and realism tradeoffs for objectgoal navigation. In *Proceedings*

University of Oxford

of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 16384–16393, 2024.

- [76] Mukul Khanna*, Ram Ramrakhya*, Gunjan Chhablani, Sriram Yenamandra, Theophile Gervet, Matthew Chang, Zsolt Kira, Devendra Singh Chaplot, Dhruv Batra, and Roozbeh Mottaghi. Goat-bench: A benchmark for multi-modal lifelong navigation, 2024.
- [77] Diederik P Kingma. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.
- [78] Eric Kolve, Roozbeh Mottaghi, Winson Han, Eli VanderBilt, Luca Weihs, Alvaro Herrasti, Daniel Gordon, Yuke Zhu, Abhinav Gupta, and Ali Farhadi. AI2-THOR: an interactive 3d environment for visual AI. *arXiv*, 2017.
- [79] Aviral Kumar, Justin Fu, George Tucker, and Sergey Levine. Stabilizing Off-Policy Q-Learning via Bootstrapping Error Reduction. *arXiv e-prints*, art. arXiv:1906.00949, June 2019.
- [80] Aviral Kumar, Aurick Zhou, George Tucker, and Sergey Levine. Conservative q-learning for offline reinforcement learning. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 1179–1191. Curran Associates, Inc., 2020. URL <https://proceedings.neurips.cc/paper/2020/file/0d2b2061826a5df3221116a5085a6052-Paper.pdf>.
- [81] Aviral Kumar, Anikait Singh, Frederik Ebert, Yanlai Yang, Chelsea Finn, and Sergey Levine. Pre-Training for Robots: Offline RL Enables Learning New Tasks from a Handful of Trials. *arXiv e-prints*, art. arXiv:2210.05178, October 2022.
- [82] Vikash Kumar, Rutav Shah, Gaoyue Zhou, Vincent Moens, Vittorio Caggiano, Jay Vakil, Abhishek Gupta, and Aravind Rajeswaran. Robohive – a unified framework for robot learning. In *NeurIPS: Conference on Neural Information Processing Systems*, 2023.

- [83] Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.
- [84] Lauro Langosco Di Langosco, Jack Koch, Lee D Sharkey, Jacob Pfau, and David Krueger. Goal misgeneralization in deep reinforcement learning. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato, editors, *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 12004–12019. PMLR, 17–23 Jul 2022. URL <https://proceedings.mlr.press/v162/langosco22a.html>.
- [85] Michael Laskin, Luyu Wang, Junhyuk Oh, Emilio Parisotto, Stephen Spencer, Richie Steigerwald, DJ Strouse, Steven Hansen, Angelos Filos, Ethan Brooks, et al. In-context reinforcement learning with algorithm distillation. *arXiv preprint arXiv:2210.14215*, 2022.
- [86] Kimin Lee, Michael Laskin, Aravind Srinivas, and Pieter Abbeel. Sunrise: A simple unified framework for ensemble learning in deep reinforcement learning. In Marina Meila and Tong Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 6131–6141. PMLR, 18–24 Jul 2021. URL <https://proceedings.mlr.press/v139/lee21g.html>.
- [87] Yoonho Lee, Huaxiu Yao, and Chelsea Finn. Diversify and disambiguate: Learning from underspecified data, 2023. URL <https://arxiv.org/abs/2202.03418>.
- [88] David D. Lewis and William A. Gale. A sequential algorithm for training text classifiers. *Proceedings of SIGIR*, 1994.
- [89] Bohao Li, Rui Wang, Guangzhi Wang, Yuying Ge, Yixiao Ge, and Ying Shan. Seed-bench: Benchmarking multimodal llms with generative comprehension. *arXiv preprint arXiv:2307.16125*, 2023.

- [90] Kunchang Li, Yali Wang, Yinan He, Yizhuo Li, Yi Wang, Yi Liu, Zun Wang, Jilan Xu, Guo Chen, Ping Luo, Limin Wang, and Yu Qiao. MVBench: A Comprehensive Multi-modal Video Understanding Benchmark. *arXiv e-prints*, art. arXiv:2311.17005, November 2023. doi: 10.48550/arXiv.2311.17005.
- [91] Songtao Li and Hao Tang. Multimodal alignment and fusion: A survey, 2024. URL <https://arxiv.org/abs/2411.17040>.
- [92] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. *arXiv preprint arXiv:2304.08485*, 2023.
- [93] Ilya Loshchilov and Frank Hutter. Online Batch Selection for Faster Training of Neural Networks. *arXiv e-prints*, art. arXiv:1511.06343, November 2015.
- [94] Ilya Loshchilov and Frank Hutter. Sgdr: Stochastic gradient descent with warm restarts. *arXiv preprint arXiv:1608.03983*, 2016.
- [95] Clare Lyle. Resolving causal confusion in reinforcement learning via robust exploration. In *Self-Supervision for Reinforcement Learning Workshop - ICLR 2021*, 2021. URL <https://openreview.net/forum?id=DKCXncD4Xtq>.
- [96] Xiaojian Ma, Silong Yong, Zilong Zheng, Qing Li, Yitao Liang, Song-Chun Zhu, and Siyuan Huang. Sqa3d: Situated question answering in 3d scenes. In *International Conference on Learning Representations*, 2023. URL <https://openreview.net/forum?id=IDJx97BC38>.
- [97] Yecheng Jason Ma, William Liang, Vaidehi Som, Vikash Kumar, Amy Zhang, Osbert Bastani, and Dinesh Jayaraman. LIV: Language-Image Representations and Rewards for Robotic Control. *arXiv e-prints*, art. arXiv:2306.00958, June 2023. doi: 10.48550/arXiv.2306.00958.
- [98] Yubo Ma, Yuhang Zang, Liangyu Chen, Meiqi Chen, Yizhu Jiao, Xinze Li, Xinyuan Lu, Ziyu Liu, Yan Ma, Xiaoyi Dong, Pan Zhang, Liangming Pan, Yu-Gang Jiang, Jiaqi Wang, Yixin Cao, and Aixin Sun. Mmlongbench-doc: Benchmarking long-context document understanding with visualizations, 2024. URL <https://arxiv.org/abs/2407.01523>.

- [99] Arjun Majumdar, Gunjan Aggarwal, Bhavika Devnani, Judy Hoffman, and Dhruv Batra. Zson: Zero-shot object-goal navigation using multimodal goal embeddings. *Advances in Neural Information Processing Systems*, 35: 32340–32352, 2022.
- [100] Arjun Majumdar, Karmesh Yadav, Sergio Arnaud, Jason Ma, Claire Chen, Sneha Silwal, Aryan Jain, Vincent-Pierre Berges, Tingfan Wu, Jay Vakil, et al. Where are we in the search for an artificial visual cortex for embodied intelligence? *Advances in Neural Information Processing Systems*, 36:655–677, 2023.
- [101] Arjun Majumdar, Anurag Ajay, Xiaohan Zhang, Pranav Putta, Sriram Yenamandra, Mikael Henaff, Sneha Silwal, Paul Mccvay, Oleksandr Maksymets, Sergio Arnaud, Karmesh Yadav, Qiyang Li, Ben Newman, Mohit Sharma, Vincent Berges, Shiqi Zhang, Pulkit Agrawal, Yonatan Bisk, Dhruv Batra, Mrinal Kalakrishnan, Franziska Meier, Chris Paxton, Sasha Sax, and Aravind Rajeswaran. Openeqa: Embodied question answering in the era of foundation models. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024.
- [102] Zhao Mandi, Homanga Bharadhwaj, Vincent Moens, Shuran Song, Aravind Rajeswaran, and Vikash Kumar. Cacti: A framework for scalable multi-task multi-scene visual imitation learning. *arXiv preprint arXiv:2212.05711*, 2022.
- [103] Karttikeya Mangalam, Raiymbek Akshulakov, and Jitendra Malik. EgoSchema: A Diagnostic Benchmark for Very Long-form Video Language Understanding. *arXiv e-prints*, art. arXiv:2308.09126, August 2023. doi: 10.48550/arXiv.2308.09126.
- [104] Lina Mezghani, Sainbayar Sukhbaatar, Thibaut Lavril, Oleksandr Maksymets, Dhruv Batra, Piotr Bojanowski, and Karteek Alahari. Memory-augmented reinforcement learning for image-goal navigation. *arXiv preprint arXiv:2101.05181*, 2021.

- [105] So Yeon Min, Xavi Puig, Devendra Singh Chaplot, Tsung-Yen Yang, Akshara Rai, Priyam Parashar, Ruslan Salakhutdinov, Yonatan Bisk, and Roozbeh Mottaghi. Situated Instruction Following. *arXiv e-prints*, art. arXiv:2407.12061, July 2024. doi: 10.48550/arXiv.2407.12061.
- [106] Baharan Mirzasoleiman, Jeff Bilmes, and Jure Leskovec. Coresets for Data-efficient Training of Machine Learning Models. *arXiv e-prints*, art. arXiv:1906.01827, June 2019.
- [107] Tsendsuren Munkhdalai, Manaal Faruqui, and Siddharth Gopal. Leave no context behind: Efficient infinite context transformers with infini-attention. *arXiv preprint arXiv:2404.07143*, 2024.
- [108] Suraj Nair, Aravind Rajeswaran, Vikash Kumar, Chelsea Finn, and Abhinav Gupta. R3m: A universal visual representation for robot manipulation. In *6th Annual Conference on Robot Learning*, 2022.
- [109] Tianwei Ni, Michel Ma, Benjamin Eysenbach, and Pierre-Luc Bacon. When do transformers shine in rl? decoupling memory from credit assignment. *Advances in Neural Information Processing Systems*, 36:50429–50452, 2023.
- [110] Tianwei Ni, Michel Ma, Benjamin Eysenbach, and Pierre-Luc Bacon. When do transformers shine in RL? decoupling memory from credit assignment. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL <https://openreview.net/forum?id=APGXBNkt6h>.
- [111] OpenAI. GPT-4 technical report, 2023.
- [112] Ian Osband, Charles Blundell, Alexander Pritzel, and Benjamin Van Roy. Deep exploration via bootstrapped DQN. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2016.
- [113] Ian Osband, Charles Blundell, Alexander Pritzel, and Benjamin Van Roy. Deep exploration via bootstrapped dqn. In D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 29. Curran Associates,

- Inc., 2016. URL <https://proceedings.neurips.cc/paper/2016/file/8d8818c8e140c64c743113f563cf750f-Paper.pdf>.
- [114] Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. fairseq: A fast, extensible toolkit for sequence modeling, 2019. URL <https://arxiv.org/abs/1904.01038>.
- [115] Hsiao-Ru Pan, Nico Gürtler, Alexander Neitz, and Bernhard Schölkopf. Direct Advantage Estimation. *arXiv e-prints*, art. arXiv:2109.06093, September 2021.
- [116] Emilio Parisotto, H. Francis Song, Jack W. Rae, Razvan Pascanu, Caglar Gulcehre, Siddhant M. Jayakumar, Max Jaderberg, Raphaël Lopez Kaufman, Aidan Clark, Seb Noury, Matthew M. Botvinick, Nicolas Heess, and Raia Hadsell. Stabilizing transformers for reinforcement learning. In *Proceedings of the 37th International Conference on Machine Learning, ICML'20*. JMLR.org, 2020.
- [117] Jongjin Park, Younggyo Seo, Chang Liu, Li Zhao, Tao Qin, Jinwoo Shin, and Tie-Yan Liu. Object-aware regularization for addressing causal confusion in imitation learning. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, volume 34, pages 3029–3042. Curran Associates, Inc., 2021. URL <https://proceedings.neurips.cc/paper/2021/file/17a3120e4e5fbdc3cb5b5f946809b06a-Paper.pdf>.
- [118] Jurgis Pašukonis, Timothy P Lillicrap, and Danijar Hafner. Evaluating long-term memory in 3d mazes. In *The Eleventh International Conference on Learning Representations*, 2023. URL <https://openreview.net/forum?id=yHLvI1E9RGN>.
- [119] Tim Pearce, Tabish Rashid, Anssi Kanervisto, Dave Bignell, Mingfei Sun, Raluca Georgescu, Sergio Valcarcel Macua, Shan Zheng Tan, Ida Momennejad, Katja Hofmann, et al. Imitating human behaviour with diffusion models. *arXiv preprint arXiv:2301.10677*, 2023.

- [120] Marco Pleines, Matthias Pallasch, Frank Zimmer, and Mike Preuss. Memory gym: Partially observable challenges to memory-based agents. In *International Conference on Learning Representations*, 2023. URL <https://openreview.net/forum?id=jHc8dCx6DDr>.
- [121] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In Marina Meila and Tong Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 8748–8763. PMLR, 18–24 Jul 2021.
- [122] Jack W. Rae, Anna Potapenko, Siddhant M. Jayakumar, Chloe Hillier, and Timothy P. Lillicrap. Compressive transformers for long-range sequence modelling. In *International Conference on Learning Representations*, 2020. URL <https://openreview.net/forum?id=SylKikSYDH>.
- [123] Santhosh Kumar Ramakrishnan, Erik Wijmans, Philipp Kraehenbuehl, and Vladlen Koltun. Does spatial cognition emerge in frontier models? *arXiv preprint arXiv:2410.06468*, 2024.
- [124] Ruchit Rawal, Khalid Saifullah, Ronen Basri, David Jacobs, Gowthami Somepalli, and Tom Goldstein. Cinepile: A long video question answering dataset and benchmark. *arXiv preprint arXiv:2405.08813*, 2024.
- [125] Allen Z. Ren, Jaden Clark, Anushri Dixit, Masha Itkina, Anirudha Majumdar, and Dorsa Sadigh. Explore until confident: Efficient exploration for embodied question answering. In *arXiv preprint arXiv:2403.15941*, 2024.
- [126] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10684–10695, 2022.

University of Oxford

- [127] Egor Rotinov. Reverse Experience Replay. *arXiv e-prints*, art. arXiv:1910.08780, October 2019.
- [128] Donald B. Rubin. Estimating causal effects of treatments in randomized and nonrandomized studies. *Journal of Educational Psychology*, 66:688–701, 1974.
- [129] A. Rupam Mahmood, Huizhen Yu, Martha White, and Richard S. Sutton. Emphatic Temporal-Difference Learning. *arXiv e-prints*, art. arXiv:1507.01569, July 2015.
- [130] Daniel Russo, Benjamin Van Roy, Abbas Kazerouni, Ian Osband, and Zheng Wen. A tutorial on thompson sampling. In *Foundations and Trends in Machine Learning*, 2018.
- [131] Vishnu Sashank Dorbala, Prasoon Goyal, Robinson Piramuthu, Michael Johnston, Reza Ghanadhan, and Dinesh Manocha. S-EQA: Tackling Situational Queries in Embodied Question Answering. *arXiv e-prints*, art. arXiv:2405.04732, May 2024. doi: 10.48550/arXiv.2405.04732.
- [132] Manolis Savva, Abhishek Kadian, Oleksandr Maksymets, Yili Zhao, Erik Wijmans, Bhavana Jain, Julian Straub, Jia Liu, Vladlen Koltun, Jitendra Malik, Devi Parikh, and Dhruv Batra. Habitat: A Platform for Embodied AI Research. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019.
- [133] Tom Schaul, John Quan, Ioannis Antonoglou, and David Silver. Prioritized Experience Replay. *arXiv e-prints*, art. arXiv:1511.05952, November 2015.
- [134] Tom Schaul, John Quan, Ioannis Antonoglou, and David Silver. Prioritized experience replay. In *International Conference on Learning Representations (ICLR)*, 2016.
- [135] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. In *arXiv preprint*, 2017.
- [136] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg

University of Oxford

- Klimov. Proximal Policy Optimization Algorithms. *arXiv e-prints*, art. arXiv:1707.06347, July 2017. doi: 10.48550/arXiv.1707.06347.
- [137] James A Sethian. A fast marching level set method for monotonically advancing fronts. *proceedings of the National Academy of Sciences*, 93(4): 1591–1595, 1996.
- [138] Burr Settles. Active learning. In *Synthesis Lectures on Artificial Intelligence and Machine Learning*, 2012.
- [139] H. Sebastian Seung, Manfred Opper, and Haim Sompolinsky. Query by committee. In *Proceedings of the 5th Annual Workshop on Computational Learning Theory (COLT)*, 1992.
- [140] Dhruv Shah, Blazej Osiński, brian ichter, and Sergey Levine. Lm-nav: Robotic navigation with large pre-trained models of language, vision, and action. In Karen Liu, Dana Kulic, and Jeff Ichnowski, editors, *Proceedings of The 6th Conference on Robot Learning*, volume 205 of *Proceedings of Machine Learning Research*, pages 492–504. PMLR, 14–18 Dec 2023.
- [141] Noam Shazeer. Glu variants improve transformer. *arXiv preprint arXiv:2002.05202*, 2020.
- [142] Zhuoran Shen, Mingyuan Zhang, Haiyu Zhao, Shuai Yi, and Hongsheng Li. Efficient attention: Attention with linear complexities. In *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, pages 3531–3539, 2021.
- [143] Mohit Shridhar, Jesse Thomason, Daniel Gordon, Yonatan Bisk, Winson Han, Roozbeh Mottaghi, Luke Zettlemoyer, and Dieter Fox. Alfred: A benchmark for interpreting grounded instructions for everyday tasks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10740–10749, 2020.
- [144] Dingjie Song, Shunian Chen, Guiming Hardy Chen, Fei Yu, Xiang Wan,

University of Oxford

- and Benyou Wang. Milebench: Benchmarking mllms in long context. *arXiv preprint arXiv:2404.18532*, 2024.
- [145] Enxin Song, Wenhao Chai, Guanhong Wang, Yucheng Zhang, Haoyang Zhou, Feiyang Wu, Xun Guo, Tian Ye, Yan Lu, Jenq-Neng Hwang, et al. Moviechat: From dense token to sparse memory for long video understanding. *arXiv preprint arXiv:2307.16449*, 2023.
- [146] Ben Sorscher, Robert Geirhos, Shashank Shekhar, Surya Ganguli, and Ari S. Morcos. Beyond neural scaling laws: Beating power law scaling via data pruning. *arXiv preprint arXiv:2206.14486*, 2022.
- [147] Jianlin Su, Murtadha Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. Roformer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568:127063, 2024.
- [148] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018.
- [149] Andrew Szot, Alex Clegg, Eric Undersander, Erik Wijmans, Yili Zhao, John Turner, Noah Maestre, Mustafa Mukadam, Devendra Chaplot, Oleksandr Maksymets, Aaron Gokaslan, Vladimir Vondrus, Sameer Dharur, Franziska Meier, Wojciech Galuba, Angel Chang, Zsolt Kira, Vladlen Koltun, Jitendra Malik, Manolis Savva, and Dhruv Batra. Habitat 2.0: Training home assistants to rearrange their habitat. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.
- [150] Andrew Szot, Max Schwarzer, Harsh Agrawal, Bogdan Mazouze, Rin Metcalfe, Walter Talbott, Natalie Mackraz, R Devon Hjelm, and Alexander T Toshev. Large language models as generalizable policies for embodied tasks. In *The Twelfth International Conference on Learning Representations*, 2024. URL <https://openreview.net/forum?id=u6imHU4Ebu>.
- [151] Raphael Tang, Linqing Liu, Akshat Pandey, Zhiying Jiang, Gefei Yang, Karun Kumar, Pontus Stenetorp, Jimmy Lin, and Ferhan Ture. What the DAAM:

- Interpreting stable diffusion using cross attention. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2023.
- [152] Gemini Team, Petko Georgiev, Ving Ian Lei, Ryan Burnell, Libin Bai, Anmol Gulati, Garrett Tanzer, Damien Vincent, Zhufeng Pan, Shibo Wang, et al. Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context. *arXiv preprint arXiv:2403.05530*, 2024.
- [153] Gemma Team, Aishwarya Kamath, Johan Ferret, Shreya Pathak, Nino Vieillard, Ramona Merhej, Sarah Perrin, Tatiana Matejovicova, Alexandre Ramé, Morgane Rivièrè, et al. Gemma 3 technical report. *arXiv preprint arXiv:2503.19786*, 2025.
- [154] Junjiao Tian, Lavisha Aggarwal, Andrea Colaco, Zsolt Kira, and Mar Gonzalez-Franco. Diffuse, attend, and segment: Unsupervised zero-shot segmentation using stable diffusion. *arXiv preprint arXiv:2308.12469*, 2023.
- [155] Jeremy Tien, Jerry Zhi-Yang He, Zackory Erickson, Anca Dragan, and Daniel S. Brown. A study of causal confusion in preference-based reward learning. In *ICML 2022: Workshop on Spurious Correlations, Invariance and Stability*, 2022. URL <https://openreview.net/forum?id=WaZZOSw9fWf>.
- [156] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.
- [157] Jeremias Traub. Representation learning with diffusion models. *arXiv preprint arXiv:2210.11058*, 2022.
- [158] Aaron van den Oord, Oriol Vinyals, and Koray Kavukcuoglu. Neural discrete representation learning. In *Advances in Neural Information Processing Systems*, 2017. URL <https://arxiv.org/abs/1711.00937>.
- [159] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones,

- Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.
- [160] Homer Walke, Kevin Black, Abraham Lee, Moo Jin Kim, Max Du, Chongyi Zheng, Tony Zhao, Philippe Hansen-Estruch, Quan Vuong, Andre He, et al. Bridgedata v2: A dataset for robot learning at scale. *arXiv preprint arXiv:2308.12952*, 2023.
- [161] Cunxiang Wang, Ruoxi Ning, Boqi Pan, Tonghui Wu, Qipeng Guo, Cheng Deng, Guangsheng Bao, Qian Wang, and Yue Zhang. Novelqa: A benchmark for long-range novel question answering, 2024.
- [162] Jinglong Wang, Xiawei Li, Jing Zhang, Qingyuan Xu, Qin Zhou, Qian Yu, Lu Sheng, and Dong Xu. Diffusion Model is Secretly a Training-free Open Vocabulary Semantic Segmenter. *arXiv e-prints*, art. arXiv:2309.02773, September 2023. doi: 10.48550/arXiv.2309.02773.
- [163] Ke-Jyun Wang, Yun-Hsuan Liu, Hung-Ting Su, Jen-Wei Wang, Yu-Siang Wang, Winston Hsu, and Wen-Chin Chen. OCID-ref: A 3D robotic dataset with embodied language for clutter scene grounding. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5333–5338, Online, June 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.naacl-main.419.
- [164] Saim Wani, Shivansh Patel, Unnat Jain, Angel Chang, and Manolis Savva. Multion: Benchmarking semantic map memory using multi-object navigation. *NeurIPS*, 2020.
- [165] Chuan Wen, Jierui Lin, Trevor Darrell, Dinesh Jayaraman, and Yang Gao. Fighting copycat agents in behavioral cloning from observation histories. In *NeurIPS*, 2020. URL <https://proceedings.neurips.cc/paper/2020/hash/1b113258af3968aaf3969ca67e744ff8-Abstract.html>.
- [166] Chuan Wen, Jierui Lin, Jianing Qian, Yang Gao, and Dinesh Jayaraman.

- Keyframe-focused visual imitation learning. In Marina Meila and Tong Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 11123–11133. PMLR, 18–24 Jul 2021. URL <https://proceedings.mlr.press/v139/wen21d.html>.
- [167] Erik Wijmans, Samyak Datta, Oleksandr Maksymets, Abhishek Das, Georgia Gkioxari, Stefan Lee, Irfan Essa, Devi Parikh, and Dhruv Batra. Embodied Question Answering in Photorealistic Environments with Point Cloud Perception. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.
- [168] 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.
- [169] 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. In *ICLR*, 2019.
- [170] Fei Xia, Amir R Zamir, Zhiyang He, Alexander Sax, Jitendra Malik, and Silvio Savarese. Gibson env: Real-world perception for embodied agents. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9068–9079, 2018.
- [171] Guangxuan Xiao, Yuandong Tian, Beidi Chen, Song Han, and Mike Lewis. Efficient streaming language models with attention sinks. *arXiv preprint arXiv:2309.17453*, 2023.
- [172] Tete Xiao, Ilija Radosavovic, Trevor Darrell, and Jitendra Malik. Masked visual pre-training for motor control. *arXiv:2203.06173*, 2022.
- [173] Quanting Xie, So Yeon Min, Tianyi Zhang, Kedi Xu, Aarav Bajaj, Ruslan Salakhutdinov, Matthew Johnson-Roberson, and Yonatan Bisk. Embodied-

- RAG: General Non-parametric Embodied Memory for Retrieval and Generation. *arXiv e-prints*, art. arXiv:2409.18313, September 2024. doi: 10.48550/arXiv.2409.18313.
- [174] Haomiao Xiong, Zongxin Yang, Jiazuo Yu, Yunzhi Zhuge, Lu Zhang, Jiawen Zhu, and Huchuan Lu. Streaming video understanding and multi-round interaction with memory-enhanced knowledge, 2025. URL <https://arxiv.org/abs/2501.13468>.
- [175] Jiarui Xu, Sifei Liu, Arash Vahdat, Wonmin Byeon, Xiaolong Wang, and Shalini De Mello. ODISE: Open-Vocabulary Panoptic Segmentation with Text-to-Image Diffusion Models. *arXiv preprint arXiv: 2303.04803*, 2023.
- [176] Zhuo Xu, Hao-Tien Lewis Chiang, Zipeng Fu, Mithun George Jacob, Tingnan Zhang, Tsang-Wei Edward Lee, Wenhao Yu, Connor Schenck, David Rendleman, Dhruv Shah, Fei Xia, Jasmine Hsu, Jonathan Hoeh, Pete Florence, Sean Kirmani, Sumeet Singh, Vikas Sindhwani, Carolina Parada, Chelsea Finn, Peng Xu, Sergey Levine, and Jie Tan. Mobility VLA: Multimodal instruction navigation with long-context VLMs and topological graphs. In *8th Annual Conference on Robot Learning*, 2024. URL <https://openreview.net/forum?id=JScswMfEQ0>.
- [177] 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, 2023.
- [178] Jihan Yang, Shusheng Yang, Anjali W Gupta, Rilyn Han, Li Fei-Fei, and Saining Xie. Thinking in space: How multimodal large language models see, remember, and recall spaces. *arXiv preprint arXiv:2412.14171*, 2024.
- [179] Xingyi Yang and Xinchao Wang. Diffusion model as representation learner. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 18938–18949, 2023.
- [180] Sriram Yenamandra, Arun Ramachandran, Karmesh Yadav, Austin Wang,

- Mukul Khanna, Theophile Gervet, Tsung-Yen Yang, Vidhi Jain, Alexander William Clegg, John Turner, et al. Homerobot: Open-vocabulary mobile manipulation. *arXiv preprint arXiv:2306.11565*, 2023.
- [181] Tianhe Yu, Deirdre Quillen, Zhanpeng He, Ryan Julian, Karol Hausman, Chelsea Finn, and Sergey Levine. Meta-world: A benchmark and evaluation for multi-task and meta reinforcement learning. In *Conference on Robot Learning (CoRL)*, 2019.
- [182] Tianhe Yu, Ted Xiao, Austin Stone, Jonathan Tompson, Anthony Brohan, Su Wang, Jaspiar Singh, Clayton Tan, Jodilyn Peralta, Brian Ichter, et al. Scaling robot learning with semantically imagined experience. *arXiv preprint arXiv:2302.11550*, 2023.
- [183] Ce Zhang, Taixi Lu, Md Mohaiminul Islam, Ziyang Wang, Shoubin Yu, Mohit Bansal, and Gedas Bertasius. A simple llm framework for long-range video question-answering, 2023.
- [184] Peitian Zhang, Zheng Liu, Shitao Xiao, Ninglu Shao, Qiwei Ye, and Zhicheng Dou. Long Context Compression with Activation Beacon. *arXiv e-prints*, art. arXiv:2401.03462, January 2024. doi: 10.48550/arXiv.2401.03462.
- [185] Peiyuan Zhang, Kaichen Zhang, Bo Li, Guangtao Zeng, Jingkan Yang, Yuanhan Zhang, Ziyue Wang, Haoran Tan, Chunyuan Li, and Ziwei Liu. Long context transfer from language to vision. *arXiv preprint arXiv:2406.16852*, 2024. URL <https://arxiv.org/abs/2406.16852>.
- [186] Jing Zhao, Heliang Zheng, Chaoyue Wang, Long Lan, and Wenjing Yang. Magicfusion: Boosting text-to-image generation performance by fusing diffusion models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 22592–22602, October 2023.
- [187] Junjie Zhou, Yan Shu, Bo Zhao, Boya Wu, Shitao Xiao, Xi Yang, Yongping Xiong, Bo Zhang, Tiejun Huang, and Zheng Liu. Mlvu: A comprehen-

- sive benchmark for multi-task long video understanding. *arXiv preprint arXiv:2406.04264*, 2024.
- [188] Xingyi Zhou, Rohit Girdhar, Armand Joulin, Philipp Krähenbühl, and Ishan Misra. Detecting twenty-thousand classes using image-level supervision. In *ECCV*, 2022.
- [189] Hao Zhu, Raghav Kapoor, So Yeon Min, Winson Han, Jiatai Li, Kaiwen Geng, Graham Neubig, Yonatan Bisk, Aniruddha Kembhavi, and Luca Weihs. Excalibur: Encouraging and evaluating embodied exploration. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 14931–14942, June 2023.
- [190] Yuke Zhu, Roozbeh Mottaghi, Eric Kolve, Joseph J. Lim, Abhinav Kumar Gupta, Li Fei-Fei, and Ali Farhadi. Target-driven Visual Navigation in Indoor Scenes using Deep Reinforcement Learning. *ICRA*, 2017.
- [191] Brianna Zitkovich, Tianhe Yu, Sichun Xu, Peng Xu, Ted Xiao, Fei Xia, Jialin Wu, Paul Wohlhart, Stefan Welker, Ayzaan Wahid, Quan Vuong, Vincent Vanhoucke, Huong Tran, Radu Soricut, Anikait Singh, Jaspiar Singh, Pierre Sermanet, Pannag R Sanketi, Grecia Salazar, Michael S Ryoo, Krista Reymann, Kanishka Rao, Karl Pertsch, Igor Mordatch, Henryk Michalewski, Yao Lu, Sergey Levine, Lisa Lee, Tsang-Wei Edward Lee, Isabel Leal, Yuheng Kuang, Dmitry Kalashnikov, Ryan Julian, Nikhil J Joshi, Alex Irpan, brian ichter, Jasmine Hsu, Alexander Herzog, Karol Hausman, Keerthana Gopalakrishnan, Chuyuan Fu, Pete Florence, Chelsea Finn, Kumar Avinava Dubey, Danny Driess, Tianli Ding, Krzysztof Marcin Choromanski, Xi Chen, Yevgen Chebotar, Justice Carbajal, Noah Brown, Anthony Brohan, Montserrat Gonzalez Arenas, and Kehang Han. RT-2: Vision-language-action models transfer web knowledge to robotic control. In *7th Annual Conference on Robot Learning*, 2023.

A | Leveraging Uncertainty for Sample-Efficient Policy Learning

A.1 Implementation

All our environments use a discrete action space. Therefore we build our method on top of the double-DQN implementation similar to the original CQL paper. As stated in Equation (3.1), we use ensembles of Q -networks, and at evaluation time, we average the Q -value outputs of the ensemble, and select the action with the maximum Q -value. In other places where we need to do inference (for instance: to compute Q -values for the conservative loss) we take the mean across the ensemble.

A design choice we make is the network initialisation when we start to do active sampling: the uncertainty encoded by a random network at the start of training can be very inaccurate and biased towards some subset of the data. We instead train with uniform sampling for an epoch and then start active sampling second epoch onwards. As expected, we see that a partially trained ensemble encodes better uncertainty compared to a randomly initialised one and that bigger ensembles encode better uncertainty than smaller ones when the network is not trained enough with uniform sampling at the start, as seen in Figure A.3.

A.2 Further Analysis

Uncertainty Histogram: In Figure A.1, we plot a histogram for the computed *Variance-data* scores after one epoch of training with uniform sampling on the Traffic-World benchmark. We overlay the scores for a specific tail case over the scores for the entire dataset. We can verify that samples from the tail-case fall in the higher percentile of variance scores.

Ensemble Uncertainty over a longer period of training: In addition to the figure in the main paper analysing the quality of uncertainty estimation on the Atari

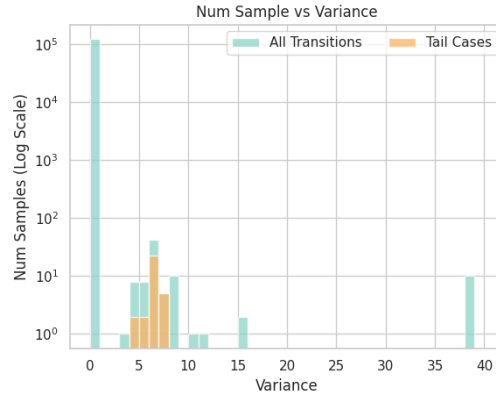


Figure A.1: (Traffic-World) Histogram of the advantage function variance estimates for the data action plotted for all transitions, overlaid by the scores for one tail case’s samples. The tail case we consider here comprises state-action pairs where the top-left tile flashes yellow but the data-collecting policy still moves one step ahead ignoring the tile, since there is space to move towards the goal without getting a negative penalty or triggering an episode termination. We can see that these samples fall in the 98th percentile of all points.

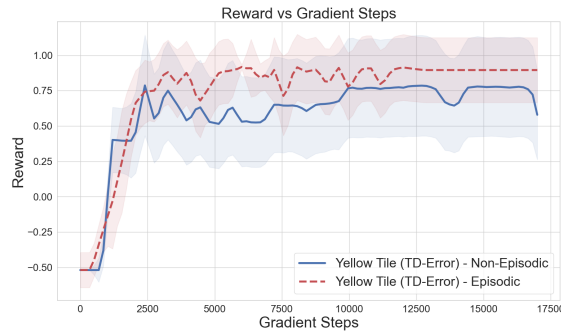


Figure A.2: (Traffic-World) Two agents trained with and without episodic scores for TD-Error-based active sampling of causally ambiguous data.

benchmark, we plot in Figure A.3 the training curves of two ensembles when they’ve been initialised with a longer period of uniform sampling at the start. We see that the gap between the informative-ness of samples fetched based on their uncertainties reduces (as compared to Figure 3.4). The larger ensemble’s uncertainty still leads to quicker convergence, however, than when both ensembles are trained using the smaller ensemble’s uncertainty for data-sampling.

***Q*-value divergence and Episodic Sampling to ensure better propagation of TD-Error:** Another observation we made was related to active sampling results in the case of benchmarks with short episodes ($n_{steps} \sim 20 - 30$) and sparse rewards,

in our case Traffic-World and Maze. Q-learning is trained through bootstrapping where we minimise the TD-error which involves estimating the Q -values at successive states across transitions in a trajectory. Additionally, the CQL objective has the gap-expanding property because the conservatism penalty tries to push Q -values of different actions at a state apart (pushing up the data-action value and pushing down others to some extent). Sometimes in the case of repeated sampling of a particular transition tuple, there is a potential divergence of Q -values of the nearby (preceding and following) state-action pairs. This can be seen as an explosion of Q -values in the training metrics and can be partially resolved by gradient clipping. We only observed this in the case of $TD - Error$ and $Variance - data$ variants.

One experiment we tried to address the above concern (with some success) was that of *episodic* sampling, where we use a heuristic to convert individual transition-wise acquisition scores to scores over entire episodes (for example taking the maximum acquisition score over transitions in an episode). This kind of episodic sampling turns out to give much more stable training curves but involves additional hyperparameters and heuristics. The reasoning for it’s success is likely related to the motivation of algorithms like emphatic-TD [129], Reverse Experience Replay (RER) [127]. In these works, they propose not just prioritising transitions with high TD-error, but also increasing the priority of transitions preceding these ones in the priority queue, since these transitions will have informative TD-updates in the subsequent time-steps. We show the reward curves with episodic and without episodic active sampling of data corresponding to the experiment in Traffic-World with the spurious correlate present, in Figure A.2.

A.3 Data Collection

Traffic-World. To collect data for Offline RL, we trained a PPO agent on a slightly modified version of the Traffic-world environment, with reward shaping on the environment [95], to incentivise the agent to reach the goal since this is a hard exploration environment (there is the potential to receive many negative rewards

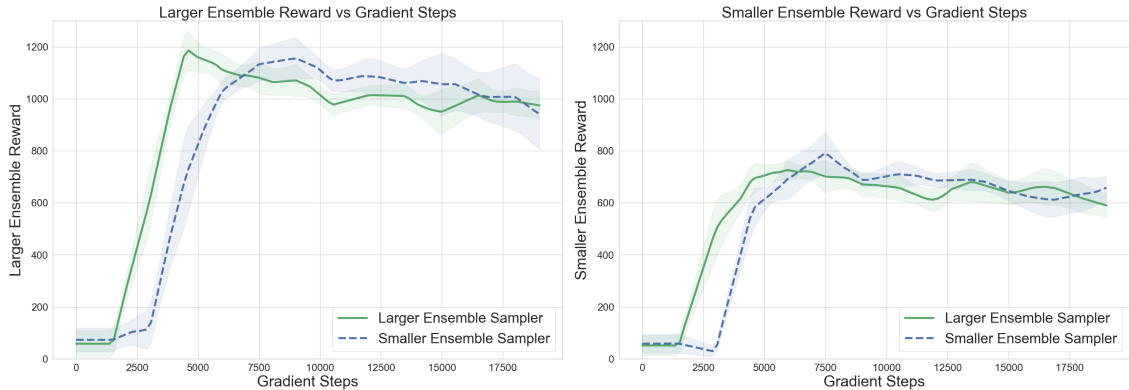


Figure A.3: Reward curves of the larger (left) and smaller (right) ensemble training when sampling on the basis of the uncertainty of either one: of the speed of convergence depending on the quality of uncertainty estimation. Here, the difference is that both ensembles start active sampling from an initialisation that is trained for longer with uniform sampling. We see a less drastic gap in convergence between the two samplers, as compared to the main paper’s Figure 3.4. This is expected since the uncertainties should get more informative (up to a point) as we train for longer with random sampling.

before receiving a positive reward, and without reward shaping the PPO agent just learns to toggle in-place till the episode ends to avoid negative penalties). **Maze.** We use the Impala-based PPO agent trained in [84] for 200M steps to collect the expert trajectories on 6000 episodes of episodes with randomised goals and 200 episodes of episodes with fixed goals.

A.4 Hyperparameters

For all benchmarks, we performed a grid-search around the hyperparameters from CQL [80]. We found the combination of parameters that matches previously reported scores for the Procgen and Atari benchmarks and achieves the highest score on the Traffic-World benchmark, when using uniform sampling with CQL. For active sampling, we used the same hyperparameters as for uniform sampling (batch size, learning rate, α). Additional parameters related to active sampling are: **(1)** n : the number of gradient steps we take before we recompute acquisition scores on the data. Thus, $n - 1$ is the number of gradient steps for which the scores remain stale. **(2)** the ensemble size which we keep constant across the active and active sampling variants for a fair comparison.

University of Oxford

For the *-batch* case of incrementally computed scores, the hyper-parameters involved are temperature and increment coefficients for importance weighting of gradients (β and *inc*), the temperature α for the acquisition scores and epsilon ϵ (the additive constant for the scores to ensure no data-point’s priority is equal to zero and thus never sampled). We used the same values as those used in the original PER implementation [133], and set β and *inc* to zero, as we found training to work better without importance weighting of gradients. Tables A.1 to A.3 list out the final values chosen for reporting results across the three benchmarks, along with the values used for the grid search.

Table A.1: Traffic-World experiments.

Hyperparameter	Value	Search
α (CQL)	1	1, 4
learning rate	5×10^{-3}	5×10^{-4} , 1×10^{-3} , 5×10^{-3}
batch-size	512	256, 512, 1024
n (steps before score recomp.)	4	2,4,8,16
gradient clipping norm	5	1,3,5,7
target update interval	4	1, 4, 16, 32
ensemble size	3	3,6
seeds	7	

Table A.2: Maze experiments.

Hyperparameter	Value	Search
α (CQL)	1	1, 4
learning rate	1×10^{-3}	5×10^{-4} , 1×10^{-3} , 5×10^{-3}
batch-size	2048	1024, 2048
n (steps before score recomp.)	8	4, 8, 16
gradient clipping norm	5	1,3,5,7
target update interval	50	20, 50, 100
ensemble size	5	3, 5
seeds	9	

Table A.3: Enduro experiments.

Hyperparameter	Value	Search
α (CQL)	1	1, 4
learning rate	5×10^{-3}	5×10^{-4} , 1×10^{-3} , 5×10^{-3}
batch-size	2048	1024, 2048
n (steps before score recomp.)	8	8, 16
gradient clipping norm	7	1,3,5,7
target update interval	100	10, 50, 100, 1000
ensemble size	5	3, 5, 10
seeds	9	

B | Leveraging Existing Knowledge for Sample-Efficient Policy Learning

B.1 Task Descriptions

B.1.1 Few-Shot Imitation Learning

For all baselines, we freeze the pre-trained vision model and train a policy using imitation learning on the provided set of 25 expert demonstrations. The results are then reported as the average of the best evaluation performance for 25 evaluation runs over 3 seeds. All experiments are conducted on a single A100 GPU with 24 CPUs and 188 GBs of RAM.

Meta-World. We follow Majumdar et al. [100] and use the hammer-v2, drawer-open-v2, bin-picking-v2, button-press-topdown-v2, assembly-v2 tasks from the Meta-World benchmark suite [181]. Each task provides the model with the last three 256×256 RGB images, alongside a 4-dimensional gripper pose. The model consists of a 3-layer MLP with a hidden dimension of 256, utilizing ReLU as the activation function. It undergoes training for 100 epochs, with evaluations conducted every 10 epochs, following the approach of Majumdar et al. (2023). The training uses a mini-batch size of 256 and a learning rate of 10^{-3} .

Franka-Kitchen. The tasks involved here include Knob On, Knob Off, Microwave Door Open, Sliding Door Open, and L Door Open, each observed from three distinct camera angles. For each task, the model receives a 256×256 RGB image and a 24-dimensional vector representing the manipulator’s proprioceptive state. For our experiments, we use the the RoboHive Kumar et al. [82] GitHub repository¹ and use a 2-layer MLP with a hidden dimension of 256 and train for 500 epochs. The mini-batch size is set at 128, with a learning rate of 10^{-4} . *We additionally correct a bug in the RoboHive implementation of the VC-1 baseline, specifically on input*

¹<https://github.com/vikashplus/robohive>

image normalization. Adjusting the image normalization to a 0-1 range resulted in a significant improvement in its performance.

B.1.2 Open Vocabulary Mobile Manipulation

Open-Vocabulary Mobile Manipulation [OVMM; 180] is a recently proposed embodied AI benchmark that evaluates an agent’s ability to find and manipulate objects of novel categories in unseen indoor environments. Specifically, the task requires an agent to “*Find and pick an object on the `start_receptacle` and place it on the `goal_receptacle`*”, where `object`, `start_receptacle` and `goal_receptacle` are the object category names. Given the long-horizon and sparse-reward nature of this task, current baselines [180] divide the problem into sub-tasks, which include navigation to the start receptacle, precise camera re-orientation to focus on the object (an abstracted form of grasping), navigating to the goal receptacle, and object placement.

Since our aim is to investigate the open-vocabulary capabilities of pre-trained representations, we choose to evaluate the models on only the precise camera re-orientation task (more commonly known as the **Gaze** task). In the original Gaze task, the agent is initialized within a distance of 1.5m and angle of 15° from the `object` which is lying on top of the `start_receptacle`. The episode is deemed successful when the agent calls the `Pick` action with the camera’s center pixel occupied by the target object and the robot’s gripper less than 0.8m from the object center. In our initial experiments, we found the current initialization scheme would lead the agent to learn a biased policy. This policy would call the `Pick` action after orienting towards the closest object in the field of view. Therefore, we chose to instantiate a harder version of the gaze task, where the episode starts with the agent spawned facing any random direction within 2.0m of the object.

We carry out our experiments using the HomeRobot GitHub repository². HomeRobot uses the Habitat simulator [149] with the episode dataset provided by Yena-

²<https://github.com/facebookresearch/home-robot>

mandra et al. [180]. This dataset uses 38 scenes for training and 12 scenes for validation, all originating from the Habitat Synthetic Scenes Dataset [HSSD; 74]. The episode dataset populates the scenes with extra objects from other datasets including Amazon Berkeley Objects [ABO; 34], Google Scanned Objects [ABO; 41] and Alfred [143]. The validation scenes are populated with objects not seen during training, spanning 106 seen and 22 unseen categories. The validation set consists of a total of 1199 episodes.

Our agent is designed to resemble the Stretch robot, characterized by a height of 1.41 meters and a radius of 0.3 meters. At a height of 1.31 meters from the base, a 640x480 resolution RGBD camera is mounted. This camera is equipped with motorized pan and tilt capabilities. The agent’s action space is continuous, allowing it to move forward distances ranging from 5 to 25 centimeters and to turn left or right within angles ranging from 5 to 30 degrees. Additionally, the agent can adjust the head’s pan and tilt by increments ranging from 0.02 to 1 radian in a single step.

In our experiments, we use a 2 layer LSTM policy and pass in the visual encoder representations after passing them through the compression layer. We initialize the LSTM weights with the LSTM weights of the Oracle model to get a slight boost in performance. We train our agents using the distributed version of PPO [168] with 152 environments spread across 4 80GB Nvidia A100 GPUs. Each run also has access to 96 CPUs and 754 GBs of RAM. We train for 100M environment steps while evaluating the agent every 5M steps and report the metrics based on the highest success rate observed on the validation set.

B.1.3 ImageNav

We conduct our ImageNav experiments in the Habitat simulator [132], using the episode dataset from Mezghani et al. [104]. These experiments are conducted using the VC-1 codebase³ [100]. The dataset uses 72 training and 14 validation scenes from the Gibson [170] scene dataset with evaluation conducted on a total of 4200

³<https://github.com/facebookresearch/eai-vc>

episodes. The agent is assumed to be in the shape of a cylinder of height 1.5m and radius 0.1m, with an RGB camera mounted at a height of 1.25m from the base. The RGB camera has a resolution of 128×128 and a 90° field-of-view.

At the start of each training episode, an agent is randomly initialized in a scene and is tasked to find the position from where the goal image was taken within 1000 simulation steps. At each step, the agent receives a new observation and is allowed to take one of the four discrete actions including MOVE_FORWARD (25cm), TURN_LEFT (30°), TURN_RIGHT (30°) and STOP. The episode is a success if the agent calls the STOP action within 1m of the goal viewpoint. Similar to [177, 100] we train our agents using a distributed version of DD-PPO [168] with 320 environments for 500M timesteps (25k updates). Our experiments are conducted using 2 nodes containing 8 A40 GPUs each with a total of 128 CPUs and 504 GBs of RAM. Each environment accumulates experience across up to 64 frames, succeeded by two PPO epochs using two mini-batches. While the pre-trained model is frozen, the policy is trained using the AdamW optimizer, with a learning rate of 2.5×10^{-4} and weight decay of 10^{-6} . We use the reward function proposed by [177]. Performance is assessed every 25M training steps, with reporting metrics based on the highest success rate observed on the validation set.

B.2 Implementation Details

B.2.1 Fine-tuning Stable Diffusion

We used the `runwayml/stable-diffusion-v1-5` model weights provided by Huggingface to initialize our models and fine-tuned them using the `diffusers` library⁴. As noted in Section 4.4.6, we used a subset of the frames from the EpicKitchens, Something-Something-v2, and Bridge-v2 datasets. We extracted the middle-third of the video clips and sampled four frames randomly from this chunk to increase the chances of sampling frames where the text prompt associated with the video clip is most relevant for describing the scene. Since the Something-Something-v2

⁴<https://huggingface.co/docs/diffusers/v0.13.0/en/training/text2image>

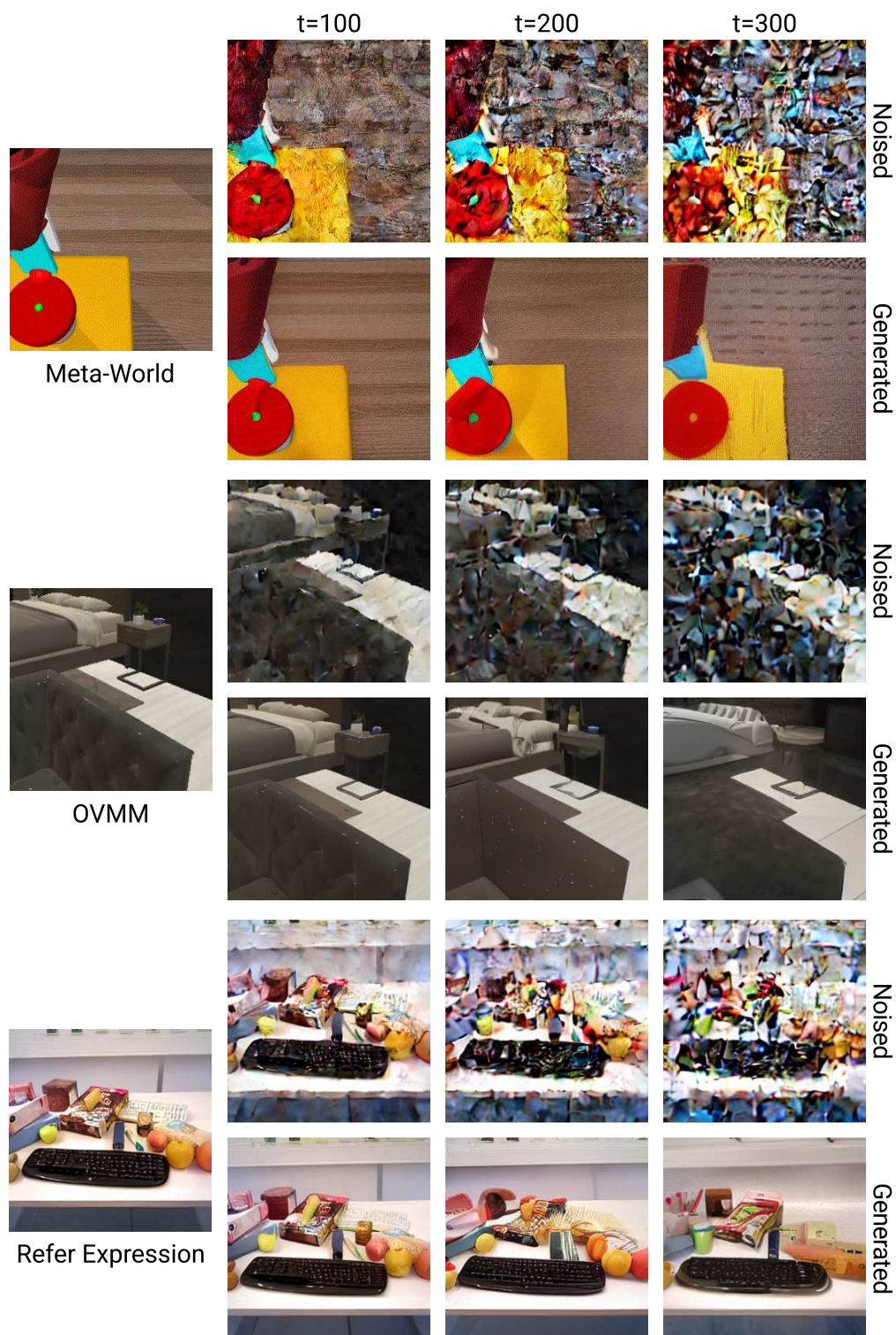


Figure B.1: Noised and denoised images from our tasks using the fine-tuned Stable Diffusion model. For each image, noise is added up to timestep $t \in 100, 200, 300$ and subsequently removed by denoising back to $t = 0$. The sensitivity to noising varies across tasks: Meta-World images remain accurately reconstructed up to $t = 200$, whereas referring expression images exhibit noticeable degradation as early as $t = 100$, particularly affecting small object details. In practice, using un-noised inputs ($t = 0$) yielded consistently reliable representations, rendering this hyperparameter unnecessary to tune.

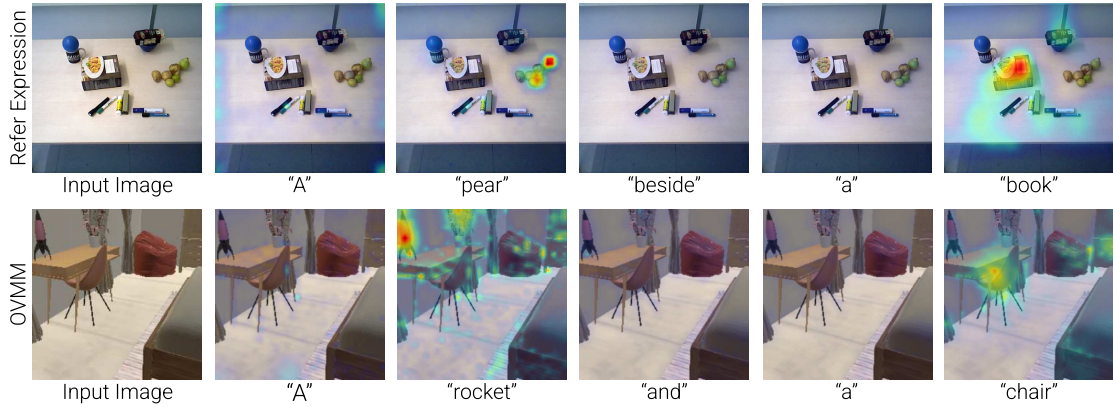


Figure B.2: The Stable Diffusion model allows us to extract word-level cross-attention maps for any given text prompt. We visualize these maps in a robotic manipulation environment and observe that they are accurate at localizing objects in a scene. Since these maps are category agnostic, downstream policies should become robust to unseen objects at test time.

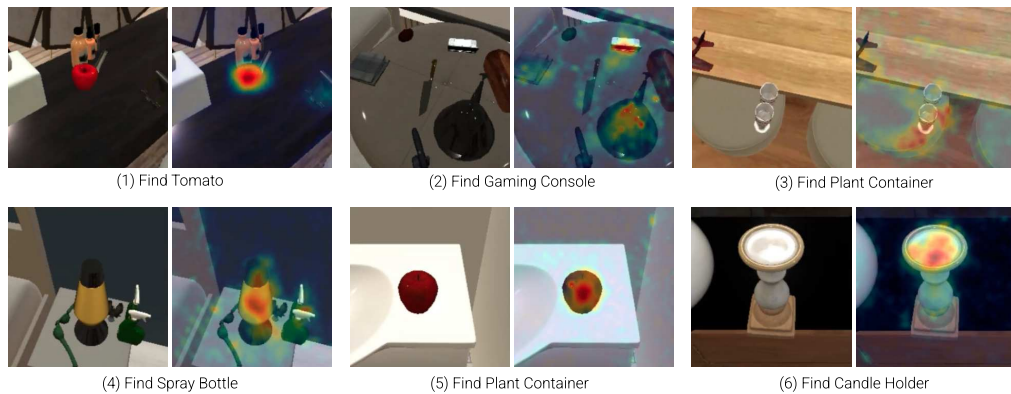


Figure B.3: Images from OVMM benchmark with their corresponding attention maps obtained from the fine-tuned Stable Diffusion (SD) model. The first 5 pairs of images correspond to failed episodes, with the bottom right pair corresponding to a successful episode. The attention maps help us interpret the cause of failure: (1) Tomato - SD wrongly attends strongly to an apple. (2) Gaming Console - visible at the top of the image; however, SD attends to multiple objects due to low visual quality. (3) Plant Container - SD instead focuses on the two glasses it sees in the image. (4) Spray Bottle - SD completely misses the spray bottles in the image and attends to the lava lamp. (5) Plant Container - SD wrongly attends to the apple. (6) Candle Holder - SD correctly attends.

(SS-v2) dataset includes a human hand manipulating table-top objects while the Bridge-V2 dataset includes a robot gripper, we append the text "human hand" and "robot hand" respectively to the text caption associated with the respective frames to reduce ambiguity in the image-text pairing. Using this procedure, we generated a paired image-language dataset with 1.3 million samples. Figure B.4 shows samples of the images from the fine-tuning datasets. Since different embodiments (human and robot) are visible in the training images, we prepended the corresponding embodiment name to the text prompt for the associated image during training.

We fine-tuned for only a single epoch ($\sim 5,000$ gradient steps) parallelly on 2 Nvidia A100 GPUs with a mini-batch size of 512 and a learning rate of $1e^{-4}$. Although the original Stable Diffusion model is trained on images of resolution 512×512 , we fine-tuned the model on images downsampled to 256×256 since it aligned with the resolution requirements of the downstream application. We show sample generations from the diffusion model after fine-tuning in Figure B.5. We found that the model learns to associate the prompt with not just the human or robot hand but also with the style of the background and objects of the training datasets.



Figure B.4: Snapshots from the datasets we use for fine-tuning the Stable Diffusion model.

B.2.2 Representation Extraction Details

Here, we describe the representation extraction details for all our baselines assuming a 224×224 input image.:

1. **Stable Control Representations:** The Stable Diffusion model downsamples the input images by a factor of 64. Therefore, we need to first resize the input image to a size of 256×256 . We pass the image to the VAE, which converts it into a latent vector of size $32 \times 32 \times 4$ and passes it to the U-Net. We use the

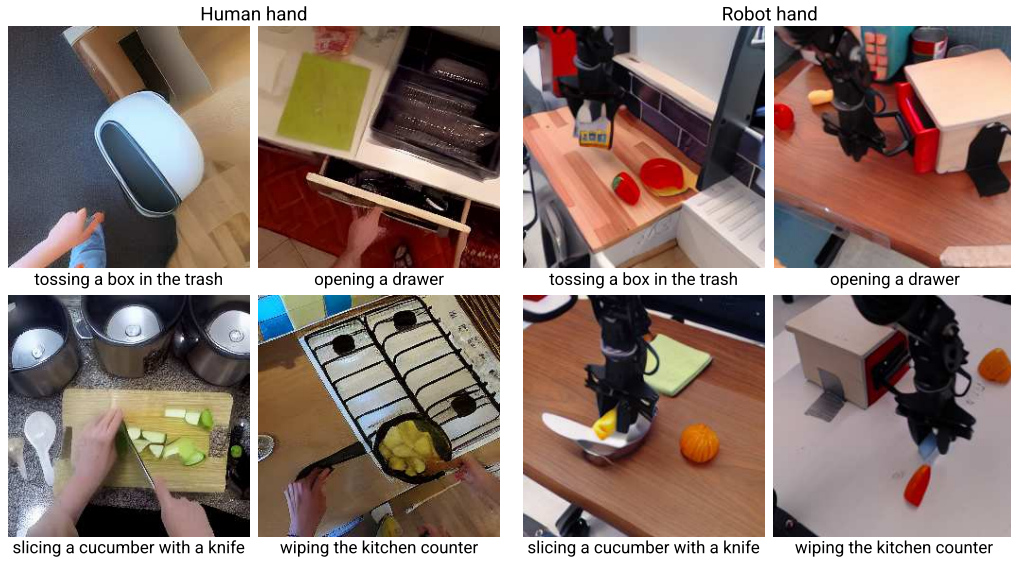


Figure B.5: Image generations from the fine-tuned Stable Diffusion model. We provided four different prompts, each prefixed with either “*Human hand*” or “*Robot hand*”.

last three downsampling blocks’ and the mid block’s output feature map of sizes $8 \times 8 \times 640$, $4 \times 4 \times 1,280$, $4 \times 4 \times 1,280$, and $4 \times 4 \times 1,280$, respectively. The total size is, therefore, 102,400, and we linearly interpolated them to the same spatial dimension (8×8) before concatenating them channel-wise.

2. **R3M** [108]: For most of our experiments we use the original ResNet50 model, which outputs a 2048 dimensional vector. For the referring expressions and grasp affordance prediction tasks from the Voltron evaluation suite [70], a ViT-S is used, which outputs an embedding of size $14 \times 14 \times 384 = 75,264$
3. **MVP** [172] and **VC-1** [100]: The last layer (24th) outputs an embedding of size $16 \times 16 \times 1,024 = 262,144$.
4. **CLIP** [121]: For ViT-B, the last layer (12th) outputs an embedding of size $14 \times 14 \times 768 = 150,528$. For ViT-L, the last layer (24th) outputs an embedding of size $16 \times 16 \times 1024 = 262,144$.
5. **Voltron** [70]: We use the VCond-Base model which outputs a representation of size $14 \times 14 \times 768 = 150,528$.
6. **SD-VAE** [126]: Outputs a latent vector of size $32 \times 32 \times 4 = 4,096$.

Table B.1: Runtimes for SCR and VC-1.

	SCR	VC-1
Time per forward pass (1,000 passes)	0.021 secs.	0.014 secs.

B.3 Runtime Analysis

We include a runtime analysis for SCR versus VC-1 in Table B.1 by reporting the time taken for a forward pass through both models over a single input image of size 256×256 . We use an A100 GPU for our experiment, and use both the models at half precision. We see that SCR takes 0.021 seconds per inference step, being 1.5x slower than the forward pass through VC-1 Large.

B.3.1 Hyperparameters

We provide the hyperparameters used in section 4.5 for Stable Control Representations in Table B.2.

Table B.2: Hyperparameters and configuration settings used across tasks and methods.

Benchmark	Timestep	Prompt	Attn	Layers	Post Compress. Dim
Meta-World	0/100/200	No	No	Mid + Down [1-3]	3072
Franka-Kitchen	0	No	No	Mid + Down [1-3]	2048
ImageNav	0	No	No	Mid + Down [1-3]	2048
OVMM	100	Yes	Yes	Mid + Down [1-3]	2048
Referring Expression	0	Yes	No	Mid + Down [1-3]	8192
Grasp Prediction	0	No	No	Mid + Down [1-3]	8192

C | Learning Policies to Leverage Experience

C.0.1 Task Setting and Model architecture: ExtObjNav

Our policy takes as input an RGB image, the agent’s relative position, the previous action, and the current goal object category. The RGB image is encoded using a ViT-B-sized VC-1 [100] visual encoder, finetuned by ReLIC to improve small object detection, a limitation of the original VC-1 model. The finetuned encoder remains frozen in all experiments, and we use VC-1’s CLS token downstream, applying an MLP for dimension reduction to size 256. The goal category one-hot vector and previous action are also embedded into 256 dimensions, while the agent’s position is converted into a 32-dimensional embedding. All these embeddings are concatenated together and passed to the sequence encoder.

All our sequence encoders (Memo, AC, etc) are based on a much smaller version of the causal auto-regressive transformer architecture used in LLaMA [156] (see details in table C.2). The weights of the model are trained from scratch unless specified otherwise. To accelerate data collection, the transformer’s KV cache is maintained between rollout steps. Following ReLIC, we shuffle older episodes within each sequence and update the KV cache after each policy update, ensuring efficient memory reuse.

We use the same reward function for the navigation task as defined in Elawady et al. [43]. The reward consists of three components: (1) Geodesic distance reward: The agent receives a reward

$$r_d = -\Delta d$$

where d is the geodesic distance to the closest object, which can change throughout the episode. (2) Slack penalty: A constant penalty of -0.003 per step encourages efficient navigation. (3) Success reward: The agent receives a reward of 2 upon successfully reaching the target.

The SPL (Success weighted by Path Length) metric is computed as:

University of Oxford

$$\text{SPL} = \frac{1}{N} \sum_{i=1}^N S_i \cdot \frac{L_i^*}{\max(L_i, L_i^*)}$$

where N is the number of episodes, S_i is a binary indicator of success in episode i , L_i is the length of the path taken by the agent, L_i^* is the length of the shortest path to the goal (computed using privileged simulator information). The SPL values reported in the main graphs are computed at intervals of 500 environment steps. Specifically, the SPL at step t is calculated by averaging the SPL of all episodes that completed between steps $t - 499$ and t . So the SPL at step 5000 reflects the performance of all episodes that finished between step 4501 and 5000.

C.0.2 Task Setting and Model architecture: Dark-Key-To-Door

We refer the reader to the AMAGO implementation [54] (<https://github.com/UT-Austin-RPL/amago>) for details on the policy architecture used on the Dark-Key-To-Door task. We keep all implementation and training details fixed.

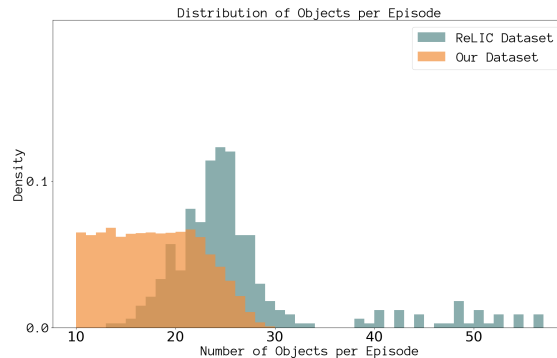


Figure C.1: Distribution of Objects in the training episodes: We visualize the changed distribution of objects sampled (and placed) randomly in the scenes corresponding to our dataset versus the original training dataset used in [43]. In our training scenes, avoid sampling more than 30 objects per scene to avoid cluttered rooms where multiple target objects could be placed together, to reduce the risk of models overfitting during training.

C.0.3 Training Scenes Dataset: ExtObjNav

Since the training dataset used in ReLIC only consisted of 333 episodes, we create our own training dataset while using the same validation dataset as them. We use

the same 37 train scenes as [43, 180]. However we generate 300 episodes per scene each having a different object placement. This leads to a total of 11100 training episode. In each episode we reduce the number of sampled objects to make the navigation task harder since the agent has to traverse longer, more targeted paths to reach objects instead of getting away by guessing (i.e. navigating to rooms with more clutter and many receptacles since the likelihood of finding an object would be higher there). We visualize the difference in sampled object distributions over episode, between the old [43] and new datasets in fig. C.1.

C.0.4 Pseudocode for Memo

We present the algorithmic pseudocode for Memo in algorithms 2 and 3. The modifications with respect to a distributed PPO implementation (corresponding to ReLIC) are highlighted in blue. We plan to fully open-source our implementation upon publication and include an early release version at <https://github.com/Memory-icrl/memo>.

C.0.5 Extended Result: FLOPs Comparison

Here, we include memory, FLOPs and latency comparison between Memo and Full-Context Transformer during evaluation on EXT OBJNAV at 32k env steps. Memo requires 10x lesser memory and 4.2x lesser floating point operation, being 2x faster than the Full-Context Transformer

Table C.1: GPU memory usage comparison: Memo versus the Full-Context Transformer (FCT) at the end of 32k steps of evaluation on the EXT OBJNAV task. We observe a 10× higher memory requirement for the KV cache of FCT, in line with the context compression ratio (~ 8) of Memo.

	Memo	Full-Context Transformer
GPU Memory	51.8 MB	546.5 MB
Model FLOPs	17.61 MFLOPs	74.49 MFLOPs
Latency	5.3 ms	10.1 ms

Algorithm 2 Memo

```

1: Initialize  $\theta, \phi$ , buffer  $\mathcal{B} \leftarrow \emptyset$ , summary tokens  $\mathcal{S} \leftarrow \emptyset$ 
2:  $o_1 \leftarrow \text{reset}()$ 
3:  $i \leftarrow 0$ 
4: for  $t = 1$  to  $N$  do
5:    $\mathcal{C}_{i:t-1} \leftarrow$  context from  $\mathcal{B}$ 
6:    $h_t \leftarrow \text{SeqEnc}_\phi(\mathcal{S}, \mathcal{C}_{i:t-1}, o_t)$ 
7:    $a \sim \pi_\theta(a \mid h_t)$ 
8:    $(r, o_{t+1}, \text{done}) \leftarrow \text{step}(a)$ 
9:   if  $\text{done}$  then
10:      $o_{t+1} \leftarrow \text{reset}()$ 
11:   end if
12:   if  $t \% l_{\text{seg}} = 0$  then  $\triangleright l_{\text{seg}}$  is segment length
13:      $\mathcal{S} \leftarrow \text{SeqEnc}_\phi(\mathcal{S}, \mathcal{C}_{i:t-1}, o_t)$ 
14:      $i \leftarrow t$ 
15:   end if
16:   Append  $(o_t, a, r, o_{t+1}, \text{done})$  to  $\mathcal{B}$ 
17:   if  $t \% \text{num\_steps} = 0$  then
18:      $\text{TRAIN}(\mathcal{B}, t)$ 
19:   end if
20:    $o_t \leftarrow o_{t+1}$ 
21: end for

```

Algorithm 3 Train Memo

```

1: procedure  $\text{TRAIN}(\mathcal{B}, t)$ 
2:   Initialize summary tokens  $\mathcal{S}_{\text{train}} \leftarrow \emptyset$ 
3:    $n \leftarrow \lfloor \frac{t}{l_{\text{seg}}} \rfloor$   $\triangleright n$  is num segments
4:   for  $j = 0$  to  $n$  do
5:      $\mathcal{C}_{j l_{\text{seg}}:(j+1) l_{\text{seg}}} \leftarrow$  context from  $\mathcal{B}$ 
6:      $\mathcal{S}_{\text{train}} \leftarrow \text{SeqEnc}_\phi(\mathcal{S}_{\text{train}}, \mathcal{C}_{j l_{\text{seg}}:(j+1) l_{\text{seg}}})$ 
7:   end for
8:    $\mathcal{C}_{n l_{\text{seg}}:t} \leftarrow$  context from  $\mathcal{B}$ 
9:    $\mathcal{L} \leftarrow \text{PPO}_{\text{loss}}(\mathcal{S}_{\text{train}}, \mathcal{C}_{n l_{\text{seg}}:t})$ 
10:   $\theta \leftarrow \theta - \eta \nabla_\theta \mathcal{L}$ 
11: end procedure

```

C.0.6 Extended Result: Memory-Maze

To further evaluate our method on long-horizon memory-intensive tasks, we include preliminary results on the Memory Maze benchmark. This environment poses a challenging multi-goal navigation task in a procedurally generated Mujoco-based maze. At the start of each episode, up to six colored goal objects are randomly placed in the maze, and the agent is given a randomly sampled sequence of goals to reach in order. Upon reaching the current goal, the next target is revealed, continuing until the episode ends. We evaluate Memo and the Full-Context Transformer (FCT) on the 9×9 version of the task, which includes two obstacles and has an episode length of 1000 steps. This setting demands both exploration and the ability to recall spatial information about previously encountered goals and maze structure. fig. C.2 show that Memo matches the performance of FCT on this task.

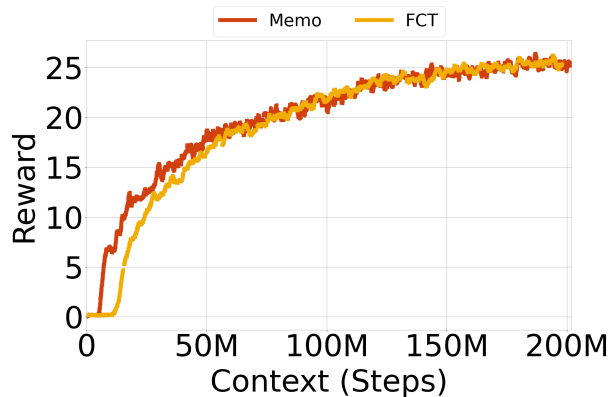


Figure C.2: Returns of FCT vs AC on Memory Maze 9x9.

C.0.7 Extended Ablation: Direct KV Cache Construction from Summary Embeddings

To assess whether the improved performance of Memo stems from the additional computation it uses (due to summary embeddings being processed through the transformer twice), we perform an ablation that removes the re-encoding step.

In this variant, instead of generating summary tokens and re-encoding them through the transformer, we directly construct the memory (KV cache) from the

internal embeddings produced at each layer while encoding the learnable summary embeddings. As in Memo, these summary embeddings are inserted at regular intervals between segments of tokens. However, unlike Memo, their layer-wise representations are extracted directly to serve as the KV cache, without further processing.

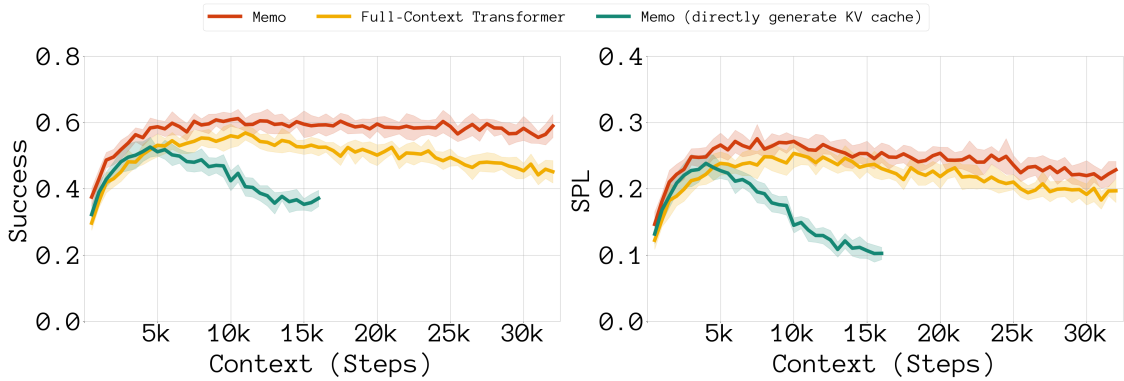


Figure C.3: Ablating the summary generation process in Memo.

This approach is more compute-efficient, as each token—including summaries—is passed through the transformer only once. However, as shown in fig. C.3, this variant performs poorly and quickly degrades beyond the training context length. We hypothesize that this is not only due to the lack of deeper summarization capacity, but also due to positional embedding generalization issues. Since summary embeddings are appended at the end of each segment, their positional indices lie in the range $[l_{seg}, l_{seg} + l_{sum}]$, with offsets accumulating across segments. Consequently, these summary positions - and those of all future tokens - progressively reach larger values, potentially leading to distribution shifts in the positional encodings that the model is unable to generalize over.

This result suggests that Memo’s re-encoding step not only adds depth for more expressive summarization but also resets the positional index range for the summary tokens, both of which appear beneficial for stability and generalization.

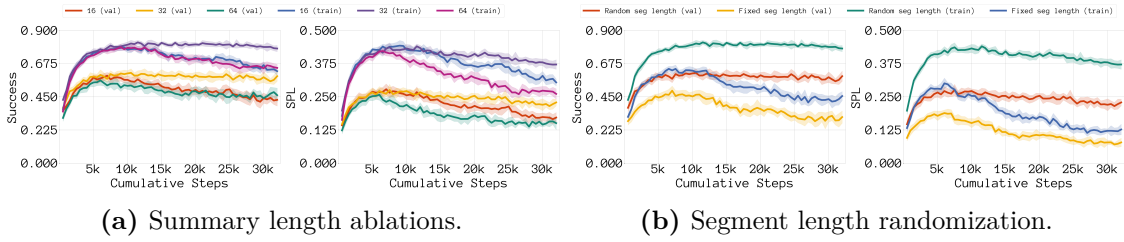


Figure C.4: Training-validation gap for Memo ablations on ExtObjNav. (Left) All summary tokens sizes (16/32/64) perform equally well on the training set for $\sim 7.5k$ environment steps after which 16 and 64 summary token policies start degrading faster. **(Right)** Fixed segment length’s training performance is worse than the val performance of random segment length.

C.0.8 Training-Validation Gap

In this section, we visualize the overlaid training and validation performance curves for Memo on the EXT OBJNAV task, which presents two key challenges. First, validation scenes are entirely novel, requiring models to generalize their in-context learning (ICL) solutions. Second, models are evaluated on $4\times$ larger trial sizes, testing their context extrapolation capabilities. To assess generalization, we compare overfitting levels across models, distinguishing whether low validation performance stems from limited encoding expressivity (indicated by low training performance) or poor generalization despite strong training performance. We present train versus validation curves for Memo while ablating the summary lengths in fig. C.4a and while ablating the segment length randomization in fig. C.4b.

C.0.9 Sample Efficiency of Memo

We observed higher sample-efficiency for Memo during training on the EXT OBJNAV and Dark-Key-To-Door tasks compared to the full context transformer. For Dark-Key-To-Door, this is directly observable in fig. 5.3a since the performance is plotted against training progress. We present this observation for the EXT OBJNAV task in this section by visualizing the performance of earlier and later checkpoints (700M and 1B training steps) of both FCT and Memo in fig. C.5. We see that even earlier into training, Memo exhibits higher-ICL success rates compared to FCT which starts to get closer to Memo’s SPL only after 300M more steps of training.

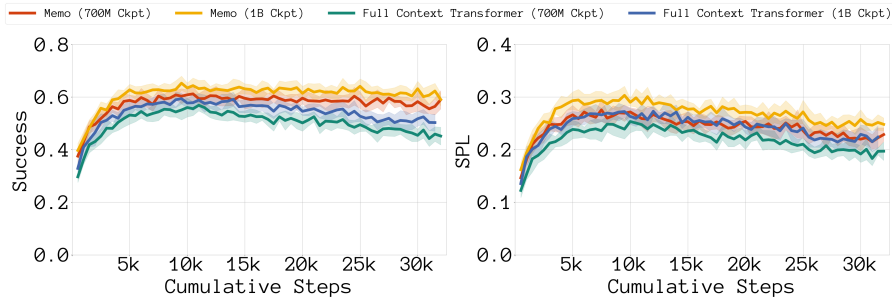


Figure C.5: Demonstrating Memo’s sample efficiency. We compare the validation reward curves of checkpoints saved at 700M and 1000M steps of training for Memo and the full context transformer (FCT) variants on the EXT OBJNAV task trained with ReLIC. Both checkpoints of Memo achieve higher success rate compared to the checkpoints of FCT, while its 1000M approaches the 700M Memo checkpoint’s performance in terms of the SPL metric. This highlights the sample efficiency of Memo, besides the compute efficiency which comes from using much smaller contexts in the transformer encoding and decoding process.

C.0.10 Hyperparameters

We list the hyperparameters for all experimental setups in this chapter in table C.2.

Table C.2: Comprehensive training setup and hyperparameters related to the EXT OBJ-NAV task.

Hyperparameter	Ours/AC/RMT	Transformer
Model Architecture		
# Layers		4
# Heads		8
Hidden dimensions		256
MLP Hidden dimensions		1024
Activation		GeLU [141]
# Sink-KV	0	1
Attention sink	N.A.	Sink KV_0
Episode index encoding	N.A.	RoPE [147]
Within-episode position encoding	RoPE [147]	Learnable
Training Setup		
Workers	320 (20 env workers per GPU \times 16 GPUs)	
Batch Size		160
RL Algorithm		DDPPO [169]
Discount Factor (γ)		0.99
GAE Parameter (τ)		0.95
Entropy Coefficient		0.1
Value Loss Coefficient		0.5
Optimization		
Optimizer		Adam [77]
Regularization	Depth dropout [62] with value 0.1 In-context episodes shuffled post partial updates	
Learning Rate Schedule	Warm-up for 100K env interactions	
Initial Learning Rate		4×10^{-7}
Learning Rate at Warm-up End		4×10^{-4}
Decay Schedule	Cosine decay [94] to 0 after 1B steps	
Computation		
Precision	Vis Encoder: FP16, Other model parts: FP32	
Rollout size		4096
Total # updates per rollout		16
# partial updates		15
# full updates		1
Hardware and Performance		
Environment Steps	700M Steps	
Hardware	16x NVIDIA A40 GPUs	
Training Time	2.5 days	

D | Benchmarking Multimodal Agents on Memory-Intensive Tasks

D.1 Limitations

While FINDINGDORY provides a rigorous evaluation framework for long-horizon memory in embodied AI, it has certain limitations that highlight areas for future improvement.

First, our most competitive baseline, which integrates a VLM with a navigation policy, is constrained by its low-level controller. Current memory-guided navigation policies struggle to take the shortest paths in large environments after reasoning over long interaction histories. As a result, models perform poorly on spatial awareness metrics, making it difficult to assess their true memory capabilities beyond basic path efficiency. Future work should explore improved memory-driven navigation policies to better leverage long-term spatial reasoning.

Second, in the experience collection phase, our scripted oracle policy for object interactions (Magic Grasp) executes pick-and-place actions in an unnatural way—objects are abruptly transferred from the receptacle surface to the agent’s gripper without smooth motion. This may confuse VLMs, as the sudden change is visually subtle and may be overlooked if the model does not attend to the specific frames where it occurs. Developing more naturalistic interaction models could improve the realism and interpretability of memory traces.

Third, while we introduce distractor objects to discourage guessing, we do not explicitly control how prominently these objects appear in the experience collection phase. Currently, objects the agent interacts with are often viewed more frequently and closely compared to distractors, which tend to be featured less prominently in the agent’s field of view. Although some tasks mitigate this by explicitly referencing non-interacted objects, future iterations of the benchmark may need stronger distractions by ensuring more balanced exposure between distractors and interacted

objects.

Finally, while the experience collection phase involves both navigation and manipulation, the current interaction phase tasks are all navigation-based, as memory reasoning is primarily required for locating objects rather than executing pick-and-place actions, which are Markovian once the goal is known. However, our benchmark is easily extensible to include manipulation-based memory tasks, which would further enrich the evaluation of long-term reasoning in embodied AI.

Despite these limitations, FINDINGDORY establishes a strong foundation for studying long-horizon memory, offering extensible challenges that will evolve alongside advances in vision-language models and embodied learning.

D.2 Benchmark Details

D.2.1 FindingDory Episode Creation

In this section, we give details about our episode creation pipeline that allows us to procedurally generate data for our benchmark.

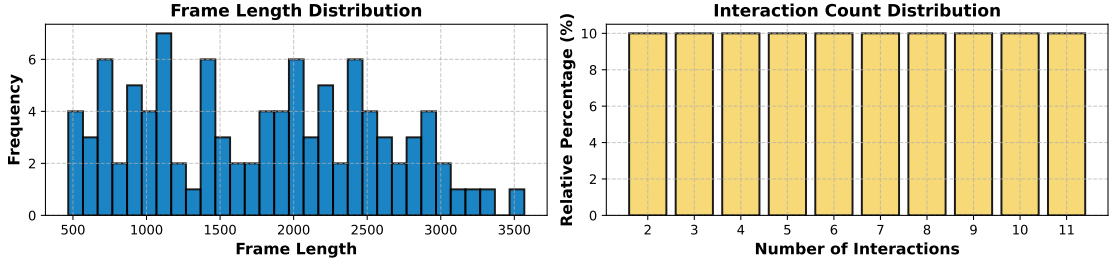
Object-Receptacle Pairing. The first step in the episode creation involves generating valid spawning positions for various object-receptacle pairs by sampling entities from the respective split in consideration (`train/val`). The objective is to identify *candidate objects* that can be picked up from a *start receptacle* and placed stably on a *goal receptacle*. We use the pipeline proposed in [180] to procedurally generate various object placements on receptacle surfaces by running physics checks for stable placements. To make the generation procedure easier, we assume that each object is picked and placed only once during an episode. Additionally, we constrain the *start receptacle* category to be distinct from the *goal receptacle* category. We also ensure that for each *candidate object* category, a corresponding *non-interacted object* of the same category is spawned to serve as distractor entities during the interaction phase. Figures D.1b and D.1c provide a distribution of distractor objects and

receptacles in the val episodes. The number of sampled object-receptacle pairings is equal to the number of rearrangements in the particular episode and can be configured through an argument. In essence, this controls the distribution of video frame lengths in the FINDINGDORY tasks(see Figure D.1a).

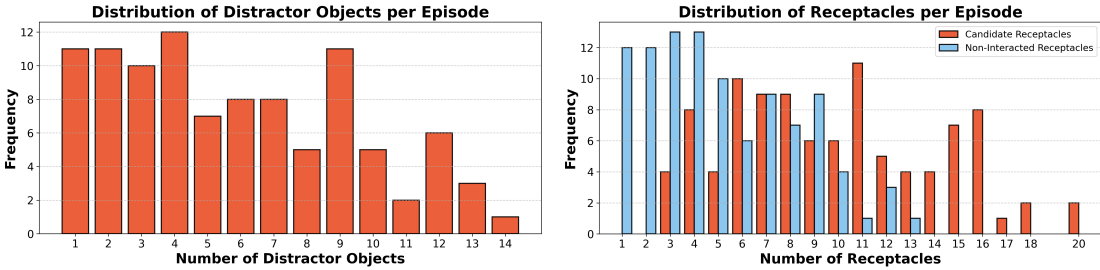
Object Placement Sequences. Once we have identified the set of *candidate objects* and the corresponding *goal receptacles*, we require an oracle policy through which we can “naturally” drop objects on the receptacle surfaces. In practice, one can achieve this by training a “place” skill through end-to-end reinforcement learning or use a heuristic agent that can place objects on receptacles but both frameworks do not guarantee perfect 100% success which is crucial for solvability in FINDINGDORY tasks. To address this, we construct an offline pipeline that produces the set of low-level “oracle” joint actions that lead to stably placing the object on the goal receptacle. We spawn the agent (with the object snapped in its end-effector) at the associated viewpoints of *goal receptacles* on the navigation mesh. From each viewpoint, we try to execute a heuristic, state-machine based placing policy which performs a set of predefined joint effector actions after identifying a “clutter-free” region in the receptacle pointcloud (using an onboard depth sensor). The output of this routine produces a set of low level joint actions and the corresponding receptacle viewpoint for each *goal receptacle* that was identified in the previous object-receptacle pairing stage. We reject episodes where the heuristic-based place policy cannot stably place the object from *any* of the associated goal receptacle viewpoints.

Task Creation. Once we have the final filtered set of episodes, we populate task instructions for each episode using `yaml`-based instruction templates as used in [150]. We employ the instruction templates to *procedurally* populate various placeholders such as object names, attributes and appearance orders that create specific task instructions grounded in the entities (objects and receptacles) present in the episode in an automated manner. We present the diverse instructions templates we use

for each task category in Table D.1. In total, we create 82174/5876 unique task instructions within the `train/val` split respectively.



(a) Distributions of frame counts and interaction counts across episodes.



(b) Number of objects per episode.

(c) Number of receptacles per episode.

Figure D.1: Summary statistics of episode data.

D.2.2 Experience Collection Phase

We now outline the details of the experience collection phase grounded in the episodes created by the pipeline outlined in the previous section. The experience collection phase is designed to collect a “clean” and “natural” video sequence of an agent interacting in an home environment while interacting with various entities. We focus on *object rearrangements* as the core interaction routine followed by the embodied agent. We construct an `OracleAgent` agent that uses privileged information from the simulator instance to efficiently generate such interaction routines. We now detail the `OracleAgent` policy to generate such video sequences for a particular episode.

Nav-to-Pick. The agent begins by being spawned at a random location in the scene associated with a particular episode. We then query a `ShortestPath` function

that leverages privileged simulator information to navigate to the object that has to be picked up. The `OracleAgent` follows the shortest path to the object and collects images as it navigates in the environment.

Pick. Once the `OracleAgent` reaches the object location, it executes a hardcoded pick policy sequence which results in the object being magically grasped by the `OracleAgent`. The pick policy uses the onboard depth sensor to orient itself with the object instance placed on the receptacle in front of the agent. Once the object is in clear view (using the ground truth instance mask from the simulator), the end effector extends outwards to perform a `magic_grasp` action. In cases when the object is not visible, we re-query the `ShortestPath` function to navigate to an alternate viewpoint associated with the receptacle that would lead to the object being clearly visible.

Nav-to-Place. After grasping the object, the `OracleAgent` navigates to the corresponding *goal receptacle* that was sampled during the episode creation phase. Specifically, the agent navigates to the specific viewpoint associated with the *goal receptacle* from which the pre-recorded placement action sequence is to be executed.

Place. Once the agent reaches the specific viewpoint, we re-play the end-effector action sequence that was found to lead to a stable object placement during episode creation. This marks the end of a single rearrangement interaction routine and the agent transitions to rearrange the next *candidate object* in the episode until no more objects are left.

Final Navigation. At the end of all interaction routines within an episode, the agent navigates to a predefined location to ensure that it does not remain in close proximity to the final interacted object or the final *goal_receptacle*. This ensures that tasks are not solvable by selecting only the last few frames in the video

sequence. We ensure that the agent navigates to a distance of 3 m away from the final interaction location.

The above `OracleAgent` routine enables us to procedurally create videos of the agent rearranging objects in the environment. Additionally, as the agent navigates the environment between the pick and place substeps, it collects additional frames detailing the scene environment which can be useful (or serve as distractor frames) when attempting to solve FINDINGDORY tasks. We present an example experience collection trajectory and associated “magic” pick-place sequence in Figures D.8 and D.9.

D.2.3 Evaluations

In this section, we define the primary evaluation metrics employed to benchmark the performance of various baselines on the FINDINGDORY tasks. Following [15], we focus on metrics that systematically quantify both the success and efficiency of the agent in solving specified instructions. Success focuses on accurately selecting frames from the interaction video that lead to task completion, while efficiency evaluates if the agent selects and navigates to the optimal frame that solves the task using the fewest low-level control actions. Notably, optimizing efficiency poses a significant challenge for current methods [123, 178].

High-Level Policy Success Rate (HL-SR). This metric measures the percentage of episodes in which the high-level agent correctly predicts frame indices required to solve the specified task. We define an episode as successful if the frame(s) predicted by the VLM satisfy the following criteria:

- *Distance-to-Goal:* The agent is within a specified distance of the target entity upon reaching the predicted frame. We use thresholds of 2.0 m for objects and 0.1 m for receptacles. The larger threshold for objects accounts for the possibility of objects being positioned on elongated receptacles (e.g., couches),

potentially affecting reachability.

- *Angle-to-Goal*: The agent is oriented within a specified angular distance toward the target entity’s center. We set angle thresholds of 45° for objects and 90° for receptacles due to their typically larger dimensions.
- *Semantic Coverage*: The semantic mask of the target entity covers at least 0.1% of the pixels in the frame.
- *Room Region Check*: For room visitation tasks, we verify if the agent is located within the designated target region.

These thresholds and criteria were qualitatively validated through manual inspection to ensure their appropriateness; During evaluation, these criteria are verified by teleporting the robot to the pose associated with the frame(s) selected by the high-level agent.

Low-Level Policy Success Rate (LL-SR). This metric measures the percentage of episodes where the low-level navigation agent successfully navigates to the target entity. The low-level agent is activated only if the high-level agent correctly predicts the frame indices, meaning LL-SR is evaluated conditionally upon HL-SR being true for each episode. The success criteria match those outlined for HL-SR.

Success-weighted-by-Path-Length. This metric evaluates task completion efficiency, computed per task episode as follows [15]:

$$\text{SPL}_i = S_i \cdot \frac{l_i}{\max(p_i, l_i)}$$

where S_i denotes HL-SR/LL-SR for computing the HL-SPL/LL-SPL respectively, l_i is the length of the shortest possible path to the closest successful goal frame (or shortest path to revisit all subgoals), and p_i is the actual path length traveled by the agent.

Additionally, we introduce two relaxed success metrics to analyze baseline failure modes:

- **Distance-to-Goal-Only Success Rate (DTG-SR)**. Computed identically to HL-SR but without the semantic coverage requirement. This metric helps quantify instances where the agent selects frames close to the goal without necessarily capturing it visually.
- **Semantic Coverage Success Rate (SC-SR)**. Computed identically to HL-SR but without the distance-to-goal requirement. This metric evaluates instances where the agent correctly selects frames showing the target entity but fails to meet proximity criteria.

D.3 Extended Related Work

D.3.1 An overview of approaches tackling memory in Long VQA and embodied tasks

Many recent works equip large language models with memory-like capabilities through task-specific knowledge bases or semantic maps for navigation planning [105, 27]. MobilityVLA [176] leverages long-context VLMs to process past frames, enabling goal-frame selection via Gemini, while topological mapping guides navigation without explicit pathfinding. ReMEmbR [9] and Embodied RAG [173] introduce nonparametric memory trees to store and retrieve past experiences for planning. For video question answering, Zhang et al. [183], Kahatapitiya et al. [67] propose a streaming-based approach that accumulates frame-wise captions for later querying. However, this method constrains memory to the expressivity of vision-language or captioning models, potentially omitting critical details. We include the hierarchical approach involving long-context VLMs and the approach utilizing textual memory traces as baselines in our evaluations on the benchmark tasks.

gemini-2.0-flash Task Prompt

You are an intelligent question answering agent. In the prompt video preceding the text, you will be shown a long stream of images that have been collected by a robot while navigating around a home and asked to answer a question about the space to assist a user in completing tasks in this home. In the video stream, the robot is picking up objects and rearranging them to different arbitrary locations throughout the house.

The questions asked by user will require being able to look at the full set of images collected by robot to be able to provide the answer. Your task is to identify the exact point in the video (by timestamp) where the user should move to best accomplish their goal. You can do this by outputting the exact timestep in the video where you are most confident that the object or place that they should move to, was viewed closely.

The images in the video also have the time of day information in top left. You need to use the time of day information for tasks that require to revisit objects or receptacles at a specific time of day.

For some tasks that require revisiting multiple receptacles or objects, you should output multiple frame indices corresponding to the correct order of revisitation (only if specified). For all user goals, identify the minimum number of objects or receptacles (targets) that you need to revisit to complete the user's goal. For each revisitation, you should select one corresponding timestep in the video that you will move to. You need to ensure that you are not revisiting targets that are not relevant to the user's goal.

Generate timestamps in the MM:SS format where the first two digits represent minutes and the last two digits represent seconds. To solve the tasks effectively, you should try to summarize the video by listing out the objects that were picked and placed, the receptacles from which each object was picked up and the receptacle where it was dropped.

The user's goal is to: {goal}. Now look at the full video containing all the images. Give your final response with NUM_TARGETS_TO_REVISIT: ||<num_target>||. TIMESTAMP_INDEX: ||<timestamp_1>, ..., <timestamp_num_target>||. You can include multiple timestamps if the task requires visiting more than one target (one timestamp for each target). Do not add any other text.

Figure D.2: System Prompt for gemini-2.0-flash model

D.4 Baseline Details

D.4.1 Gemini Agent

We experiment with the `gemini-2.0-flash` model which can process up to 1 million tokens in the context window. We strictly follow the outlined Google Gemini API developer guidelines [51] to structure the prompt and perform API calls over the long videos in the FINDINGDORY tasks. We do not use a “structured output” format such as json in the output responses as we empirically observed it failed to produce coherent outputs when supplied with long videos in the context. Since the `gemini-2.0-flash` model expects the input frames to be sampled at 1 FPS, we prompt the model to directly generate specific timestamps in the recommended `MM:SS` format to localize the frame corresponding to the target entity. We provide the detailed prompt that we use for evaluations in Fig. D.2.

D.4.2 Qwen and Gemma Agent

We run the Qwen2.5-VL-7B [13] and Gemma3-12B [153] locally for evaluations on the FINDINGDORY tasks. The Qwen model uses a maximum of 768 frames sampled uniformly across the input sequence while Gemma does not impose a maximum frame limit. For evaluation on 768 frames, we use two A40 GPUs with 48GB of memory. For experiments with shorter context lengths and higher subsampling (24–384 frames), a single A40 GPU suffices. We provide the full task prompt we use for inference with Qwen and Gemma in Fig. D.3 and utilize structured JSON outputs.

D.4.3 GPT Agent

We use the official OpenAI GPT-4o [64] API to run evaluations on the FINDINGDORY tasks with video frames subsampled at 96 frames to optimize API costs. For running complete evaluation on the entire validation task suite, we incurred a total cost of ≈ 400 USD. We provide the full system prompt used for GPT-4o evaluations in

Qwen/Qwen2.5-VL-7B-Instruct and google/gemma-3-12b-it Task Prompt

You are an expert and intelligent question answering agent. You will be shown a video that was collected by a robot yesterday while navigating around a house and picking and placing objects. Your job is to help the robot complete a task today by looking at the video and finding the frame indices that the robot should move to. Make sure your response is all within the JSON.

Note:

- The robot uses a magic grasp action to pick up an object, where a gripper goes close to the object and the object gets magically picked up.

Output Format (should be in JSON format):

```
'''json
{
  "chain_of_thought": str.
  "frame_indices": list[int].
}
'''
```

Where:

- Chain of Thought: A detailed explanation of the robot's thought process in determining the frame indices.
- Frame Indices: The frame indices in the video stream showing the object or place that the robot should move to.
- When deciding which frame indices to choose, make sure you choose the frame indices that are closest to the object/place.
- If task requires the agent to go to multiple places, output one frame index per object/place. (Do not use ellipsis)

The robot's goal is: {goal}

Figure D.3: System Prompt for Qwen2.5-VL and Gemma3 model

Fig. D.4.

D.4.4 Text Agent

We implement the the text-based VLM agent using the Qwen2.5-VL [13] model. For every `chunk_size` number of frames in the video, we generate a description using the VLM. We then attach the frame indices and *time of day* information to the generated summary for the frame window in consideration. We use the prompt shown in Fig. D.5 to generate the structured JSON outputs for each chunk in the

gpt-4o-2024-07-18 Task Prompt

**** OBJECTIVE ****

You are an expert and intelligent question answering agent. You will be shown a video that was collected by a robot yesterday while navigating around a house and picking and placing objects.

Your job is to help the robot complete a task today by looking at the video and finding the frame indices that the robot should move to, to complete the task.

**** OUTPUT FORMAT ****

```
```json
 "chain_of_thought": str.
 "frame_indices": list[int].
```
```

Where:

- Chain of Thought: A detailed explanation of the robot's thought process in determining the frame indices.
- Frame Indices: The frame indices in the video stream showing the object or place that the robot should move to.
- When deciding which frame indices to choose, make sure you choose the frame indices that are closest to the object/place.
- If task requires the agent to go to multiple places, output one frame index per object/place.

**** NOTES ****

- The robot uses a magic grasp action to pick up an object, where a gripper goes close to the object and the object gets magically picked up.

**** TASK ****

The robot's goal is: {goal}

Figure D.4: System Prompt for gpt-4o model

video. We then concatenate all the generated JSON outputs and pass it to the Qwen model to perform text-only reasoning over the entire text-based history of the video. The text-based reasoning prompt is provided in Fig. D.6. We note that using the same Qwen model for generating the individual text summaries and performing the final text-based reasoning helps in optimizing inference load as we maintain only a single instance of the VLM through the entire evaluation.

Text Agent (VLM) Task Prompt

You are an expert and intelligent question answering agent. You will be shown a stream of `{chunk_size}` images taken by a robot while navigating around a simulated house and picking and placing objects. Your job is to describe the image based on the output format. Make sure your response is all within the JSON format.

Output Format (should be in JSON format):

```
'''json
{
  "room_name": str.
  "picking_placing_or_navigating": str("picking", "placing" or "navigating").
  "object_being_manipulated": str.
  "receptacle_being_manipulated": str.
  "other_objects_in_scene": list[str].
}
'''
```

Where:

- Room name: The name of the room the robot is in.
- Picking, placing, or navigating: Whether the robot is picking or placing an object, or navigating around the house.
- Object being manipulated: The object that the robot is picking or placing (if relevant).
- Receptacle being manipulated: The receptacle that the robot is picking up from or placing an object into (if relevant).
- Other objects in scene: Objects present in the scene besides the main object and the receptacle.

Note:

- The robot uses a magic grasp action to pick up an object, where a gripper goes close to the object and the object gets magically picked up.

Figure D.5: System Prompt for Qwen2.5-VL model to generate text summaries based on `chunk_size` frames

D.4.5 Supervised Finetuning Baseline

We leverage the FINDINGDORY training split which consists of 82174 unique task instantiations across the categories mentioned in Table 6.2. Each video-instruction pair is coupled with a list of ground-truth frame indices that solve the task. For multi-goal tasks, each subgoal has a corresponding ground-truth frame indices sublist. Since the FINDINGDORY video sequences can be extremely long, we

Text Agent (LLM) Task Prompt

You are an expert and intelligent question answering agent. You will be provided a list of textual descriptions of the environment created by an agent while navigating around a house and picking and placing objects. The descriptions correspond to logs made after collecting and viewing a small chunk of frames, and contain information about which frame numbers they correspond to. You will also be provided a goal that now needs to be accomplished, for which you will need to use the history to decide where to go. Your job is to identify the desired frame index to navigate to, based on the provided task. Make sure your response is all within the JSON.

Agent's History information:

- Room name: The name of the room the agent was in.
- Picking, placing, or navigating: Whether the robot was picking or placing an object, or navigating around the house.
- Object being manipulated: The object that the robot was picking or placing (if relevant).
- Receptacle being manipulated: The receptacle that the robot was picking up from or placing an object into (if relevant).
- Other objects in scene: Objects present in the scene besides the main object and the receptacle.

Output Format (should be in JSON format):

```
'''json
{
  "chain_of_thought": str.
  "frame_indices": list[int].
}
'''
```

Where:

- Chain of Thought: A detailed explanation of your thought process in determining the frame indices.
- Frame Indices: A list of one or more frame indices relevant to accomplishing the goal.
- If its a single object/place task, output only one frame index.
- If task requires the agent to go to multiple places, output one frame index per object/place. (Do not use ellipsis)

Agent's history: history

The goal is {goal}

Figure D.6: System Prompt for Qwen2.5-VL model to perform textual reasoning over the concatenated textual summaries

uniformly subsample all videos to 96 frames. The ground truth frame indices are also subsampled to match the shorter video lengths to ensure the model chooses from within the subsampled frames.

Training Details. We conduct full-finetuning of the Qwen/Qwen2.5-VL-3B model using the open-source `huggingface/trl` library. For hyperparameter optimization, we utilized a smaller representative split of the training dataset with 36274 samples and conducted a grid search over the learning rate, `num_epochs` and `weight_decay` parameters. The best checkpoints from all runs were selected by evaluating performance on all video-instruction pairs in the full validation split. This “offline” evaluation is conducted without instantiating the simulator by directly comparing the model’s predicted keyframe indices with the ground-truth keyframe indices using exact string matching (thus, we don’t compute metrics proposed in Section 6.4.2). We use the cosine learning rate decay scheduler for all experiments. For the final training on the larger dataset, we found that a learning rate of 5×10^{-6} over 5 epochs with 0 weight decay worked best. For all experiments, we train in `bfloat16` format using 8 A40 GPUs on a single node with a batch size of 1 and `gradient_accumulation` of 4 - providing an effective `batch_size` of 32 samples. To further optimize training, we use `gradient_checkpointing` and highly optimized `flash_attention` implementation. In this setup, the full training run takes ≈ 120 hrs to complete.

Goal Sampling for Online Simulator Eval. The best trained checkpoint is used in online simulator evaluations on the FINDINGDORY task suite. Since the model is trained to predict the complete list of ground-truth frame indices, we select a single representative index from each generated subgoal solution list to serve as the predicted frame index. Empirically, we found that choosing either the first or a random index (of a sublist corresponding to a subgoal) yields comparable performance, while using the last index results in slightly worse performance. Therefore, we use the first index of each list as a simple and effective heuristic.

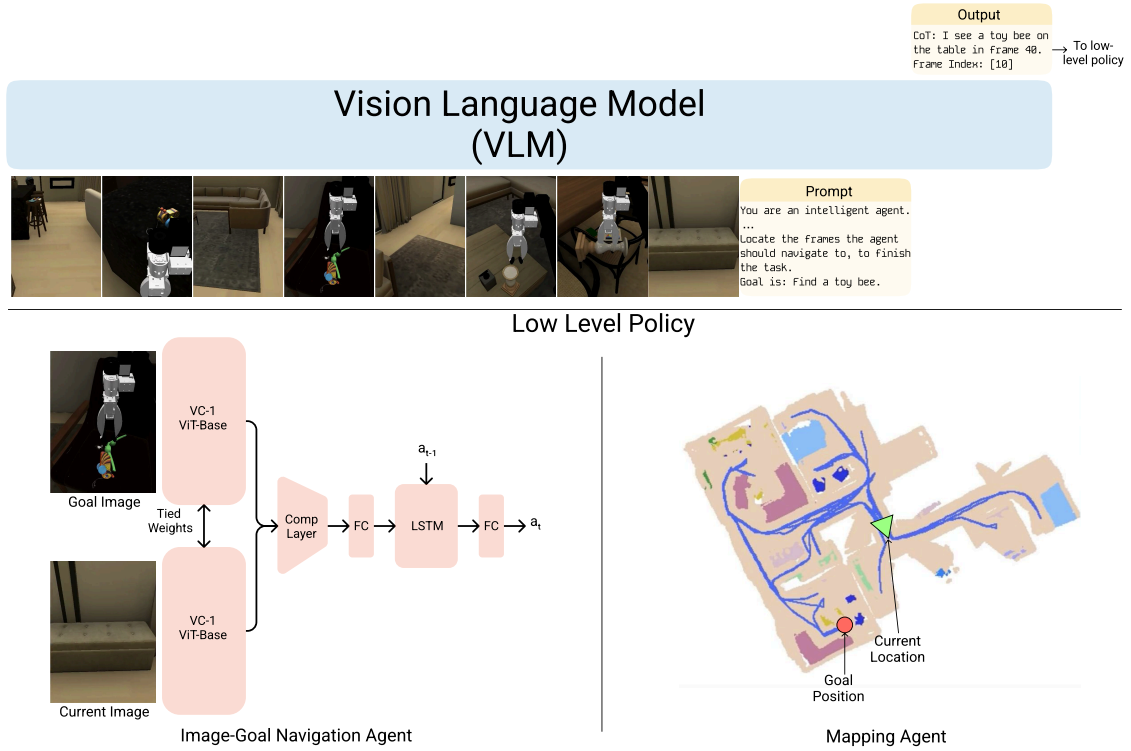


Figure D.7: Hierarchical agent architecture when using a VLM for goal frame selection and low level policies for action generation.

D.5 Low-level Policy Details

D.5.1 Image-Goal Navigation Agent

Policy Training. We train the image-goal navigation policy using the architecture and training recipe described in [177]. Specifically, we employ an end-to-end reinforcement learning policy trained using DDPPO, which predicts discrete navigation actions conditioned on visual RGB inputs and a goal image. The agent utilizes a ViT based visual encoder along with a 2-layer LSTM backbone. Similar to OVRL-v2 [177], the output patch representations from the ViT are reshaped into a 3D grid and downsampled to a lower dimension using a convolutional layer called the compression layer. However, we slightly deviate from OVRL-v2 by first concatenating the patch representations from the current and goal images before passing them through the compression layer.

The policy training occurs in the training scene split of the HSSD dataset [73],

consisting of 1166 episodes distributed across 111 scenes (a subset of the original 125 training scenes). In the Habitat simulator, the agent is modeled as a Hello Robot Stretch [71] with a height of 1.41 m and a cylindrical base radius of 0.25 m. The agent’s RGB sensor is positioned at a height of 1.31 m, with a resolution of 160×120 pixels and a horizontal field of view of 43° . During training, each episode has a maximum step budget of 1000 steps, with success defined by invoking the `STOP` action within 1.0 m of the goal image position.

As our visual encoder, we use the VC-1-Base [100] model which was previously finetuned with ImageNav on a smaller training scene split from the HSSD dataset. Empirically, we found that freezing the visual encoder after this targeted finetuning significantly accelerates the overall training performance. During the VC-1 finetuning, we use a smaller learning rate of 1.5×10^{-6} for the encoder.

We train the agents for a total of 500M timesteps on 32 A40 GPUs running 32 parallel environments each (1024 envs in total). Following the approach from [100], we collect 64 steps of experience and subsequently perform 2 PPO epochs with a mini-batch size of 2. The reward function employed is the improved formulation proposed by [177], using parameters: success weighting $c_s = 5.0$, angle success weighting $c_a = 5.0$, goal radius $r_g = 1.0$, angle threshold $\theta_g = 25^\circ$, and slack penalty $\gamma = -0.002$. We optimize using AdamW with a learning rate of 2.5×10^{-4} and weight decay 10^{-6} .

We select the best-performing policy checkpoint based on a validation success rate obtained by evaluating on 1000 image-goal navigation episodes from the FINDINGDORY validation set, achieving a success rate of 78% and SPL of 48.21%.

D.5.2 Explicit Mapping Agent

We augment the `OracleAgent` (see Appendix D.2.2) with a top-down occupancy map as it navigates the environment to generate the video sequence. In addition, we also store the 2D poses corresponding to each frame that is collected in the video sequence. The constructed map is employed by a path planning module

to generate a deterministic navigation sequence to the 2D map coordinates that correspond to the image-goal location selected by the high-level goal selection agent (see Section 6.4.1).

Map Construction. We leverage the onboard depth sensor and project the depth image at each timestep into an egocentric pointcloud using camera intrinsics. The pointcloud is then binned along the z-axis to compute occupancy values on a local 2D grid map centered around the current position of the agent. The pose sensor provides the instantaneous pose of the agent which is employed to register the computed local map in the global map of the environment. In this way, we continuously update the global map as the `OracleAgent` navigates the environment and visits multiple pick-place targets. We freeze the map updates during the pick-place subroutines as no additional occupancy information is encountered during their execution.

Planner. When the high-level goal agent selects a particular goal frame, we lookup the corresponding target 2D pose in the global map. Similar to the local navigation policy employed in [27], we use the Fast Marching Method [137] to generate the shortest path to the goal pose. We use one of four discrete navigation actions (see Section 6.3.3) to reach each waypoint on the generated shortest path.

D.6 Qualitative Examples

In this section, we include representative qualitative examples from the FINDINGDORY task suite. In Figure D.8, we provide a birds-eye view depiction of the trajectory executed during experience collection (see Appendix D.2.2) in a validation episode involving 5 pick-place interaction routines. For this episode, the collected video consists of 1308 frames in total. We provide detailed responses (and meta-analysis) from various baselines across the various task categories in Figures D.10 to D.15. In the following, we briefly discuss the different failure modes encountered in the generated responses across the 3 high-level task categories defined in Table 6.2.

Single-Goal Spatial Tasks. We observe that for the *Object Recall* task (fig. D.10), the models are able to identify a frame with the target *chest of drawers* but do not select the closest frames to ensure success. In the *Interaction* based tasks (figs. D.10 and D.11), we observe models fail to accurately list the sequence of interaction events which can be partially attributed to supplying only 96 subsampled frames from the original video. In one case, **GPT-4o** instead focuses on a “distractor” target object (fig. D.11). All models perform poorly on the *Spatial Relationship* tasks (fig. D.11) as they are unable to coherently reason about the spatial layout of the house in the provided video.

Single-Goal Temporal Tasks. For the *Interaction Order* task, we observe that the closed-source models are able to accurately identify the target object but the open-source variants fail to do so (fig. D.13). Models are generally able to select the correct frame in the *Time-Based* task as the model only needs to retrieve a single frame corresponding to the correct overlaid timestamp. For the *Duration Tracking* task, only **GPT-4o** generates the correct reasoning and frame prediction. Interestingly, **Gemma-3** tries to track the duration spent in each room by matching the floor color observed in the images but eventually fails at the task (fig. D.14).

Multi-Goal Tasks. All models fail to perform successfully on tasks under this category. Only **GPT-4o** gets a correct response for all subgoals. Surprisingly, the reasoning trace has a hallucination but this mistake is omitted in the final frame sequence prediction (fig. D.15). The models struggle to coherently list the event sequence and also hallucinate interactions. Furthermore, **Gemini-2.0-flash** also struggles to adapt the sequence of events it lists into the query revisitation sequence specified in the task (fig. D.15).

Table D.1: Task Categories with all associated prompts

| Memory Type | Example Instructions |
|----------------------|---|
| Object Recall | Navigate to a {target_object_name}. |
| | Navigate to a {target_receptacle_name}. |

Continued on the next page...

| Memory Type | Example Instructions |
|---|--|
| Interaction | Navigate to any receptacle you interacted with. |
| | Navigate to any receptacle you did not interact with. |
| | Navigate to any object that you interacted with yesterday. |
| | Navigate to any object that you did not interact with yesterday. |
| | Navigate to any receptacle you picked an object from. |
| | Navigate to any receptacle you placed an object from. |
| Conditional Interaction | Navigate to the {target_object_name} that you interacted with yesterday. |
| | Navigate to a {target_object_name} that you did not interact with yesterday. |
| | Navigate to a {target_receptacle_name} you did not interact with yesterday. |
| | Navigate to a {target_receptacle_name} you picked an object from. |
| | Navigate to a {target_receptacle_name} you placed an object on. |
| | Navigate to the receptacle that you picked the {target_object_name} from. |
| | Navigate to the object that you picked from the {target_receptacle_name}. |
| Object Attributes | Navigate back to a {target_shape} shaped object that you interacted with yesterday. |
| | Navigate back to a {target_color} colored object that you interacted with yesterday. |
| | Navigate to an interacted object with {target_print_or_design} on it. |
| | Find an already interacted object that is made of {target_material}. |
| | Go back to an interacted object that is used for {target_functionality}. |
| Navigate to the receptacle that you interacted with which is the farthest from your current location. | |

Continued on the next page...

| Memory Type | Example Instructions |
|-----------------------------|---|
| | <p>Navigate to the receptacle that you did not interact with which is the farthest from your current location.</p> <p>Navigate to the receptacle that you picked an object from which is the farthest from your current location.</p> |
| Spatial Relationship | <p>Navigate to the receptacle that you placed an object on which is the farthest from your current location.</p> <p>Navigate to the object which you interacted with which is the farthest from your current location.</p> |
| Room Visitation | <p>Navigate to the room where you picked the {target_appearance_order} object from.</p> <p>Navigate to the room where you placed the {target_appearance_order} object in.</p> <p>Navigate to the room where you picked the {target_object_name} from.</p> <p>Navigate to the room where you placed the {target_object_name} in.</p> <p>Navigate to a room that you did not visit yesterday.</p> |
| | <p>Navigate to the {target_appearance_order} object that you interacted with yesterday.</p> <p>Navigate to the {target_appearance_order} receptacle that you picked an object from.</p> <p>Navigate to the {target_appearance_order} receptacle that you placed an object on.</p> <p>Navigate to the receptacle that you picked the {target_appearance_order} object from.</p> <p>Navigate to the object that you picked from the {target_appearance_order} receptacle.</p> <p>Navigate to the object you interacted with immediately after ending the interaction with {target_object_name}.</p> |

Continued on the next page...

| Memory Type | Example Instructions |
|--------------------------|---|
| | <p>Navigate to the object you interacted with immediately before interacting with {target_object_name}.</p> <p>Navigate to the object you interacted with {target_num_counts} interactions after {target_object_name}.</p> <p>Navigate to the object you interacted with {target_num_counts} interactions before {target_object_name}.</p> <p>Navigate to the object that you interacted with between the interactions with {target_object_name_1} and {target_object_name_2}.</p> |
| Interaction Order | <p>Navigate to the receptacle that you placed an object on right before you started interacting with {target_object_name}.</p> <p>Navigate to the receptacle that you picked an object from right after you finished interacting with {target_object_name}.</p> <p>Navigate to the receptacle that you placed an object on {target_num_counts} interactions before you started interacting with {target_object_name}.</p> <p>Navigate to the receptacle that you picked an object from {target_num_counts} interactions after you finished interacting with {target_object_name}.</p> <p>Navigate to the receptacle that you placed an object on between the interactions with {target_object_name_1} and {target_object_name_2}.</p> <p>Navigate to the receptacle that you picked an object from between the interactions with {target_object_name_1} and {target_object_name_2}.</p> |
| Time-Based | <p>Navigate to the receptacle that you interacted with at {XX:XX} yesterday.</p> <p>Navigate to the object that you interacted with at {XX:XX} yesterday.</p> |

Continued on the next page...

| Memory Type | Example Instructions |
|-------------------------------|--|
| Duration Tracking | Navigate to the object which took the longest time to rearrange. |
| | Navigate to the room that you spent the most time in. |
| | Navigate to the object which took the shortest time to rearrange. |
| Unordered Revisitation | Revisit all the receptacles you picked objects from yesterday. |
| | Revisit all the receptacles you placed objects on yesterday. |
| | Revisit all the {target_receptacle_name} you placed objects on yesterday. |
| | Revisit all the {target_receptacle_name} you picked objects from yesterday. |
| | Revisit all the objects you interacted with yesterday. |
| Ordered Revisitation | Revisit all the receptacles you picked objects from yesterday in specific order. |
| | Revisit all the receptacles you placed objects on yesterday in specific order. |
| | Revisit all the objects you interacted with yesterday in specific order. |

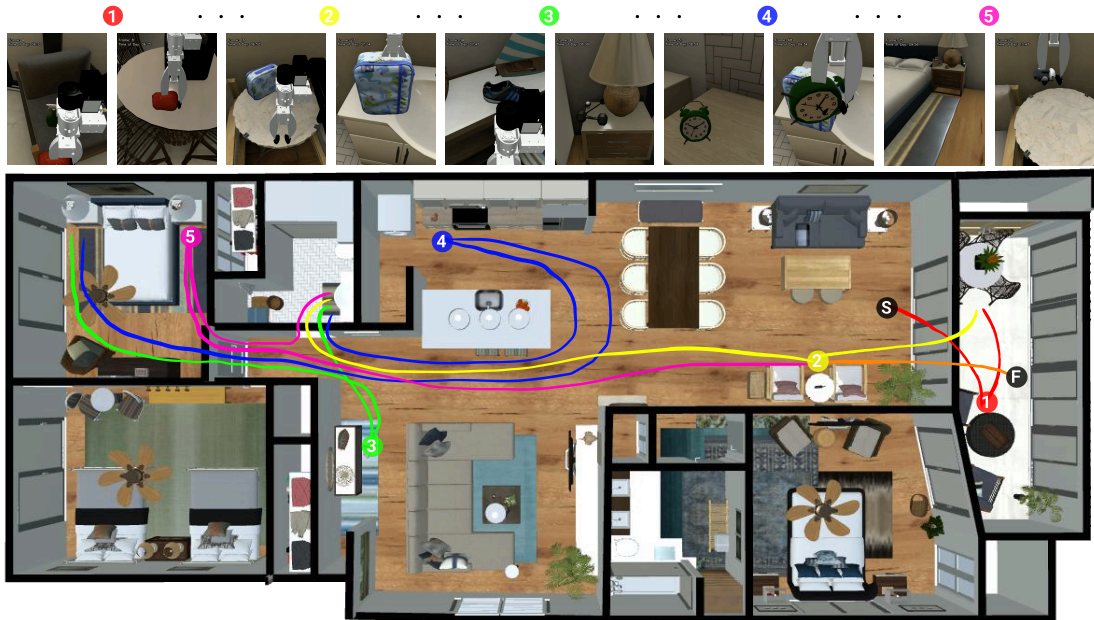


Figure D.8: An example experience collection trajectory executed by OracleAgent.

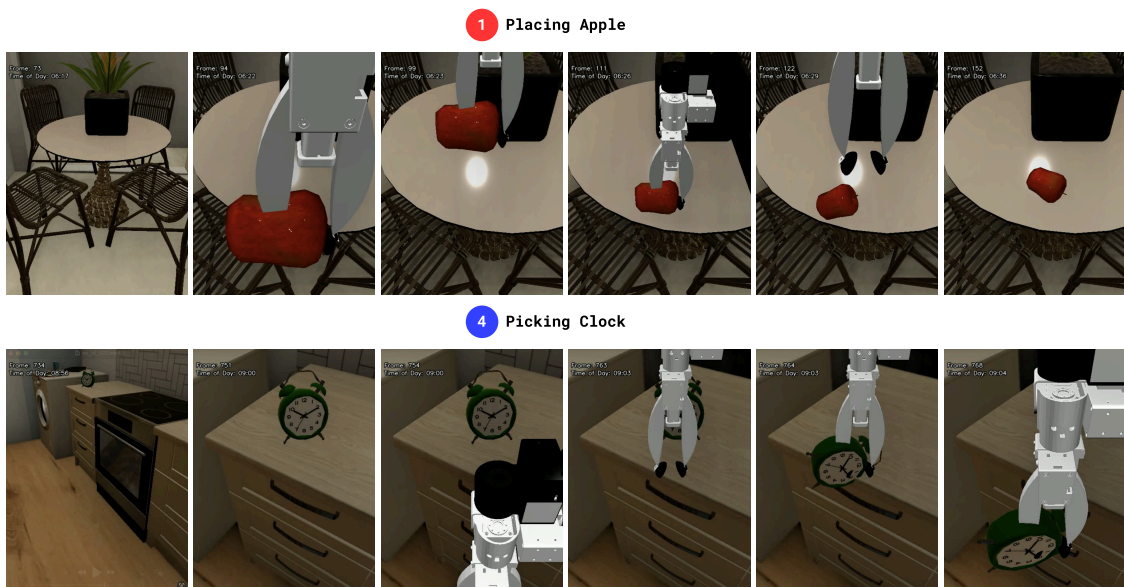


Figure D.9: Example “magic” pick-place sequences executed by the OracleAgent for sample episode in Fig. D.8.

Single-Goal Spatial Tasks

Object Recall

Task: Navigate to a chest of drawers.

✗ **GPT-4o:** To accomplish the task of navigating to a chest of drawers, I need to identify the frames where the chest of drawers is visible. The focus will be on frames showing the chest of drawers most prominently and closely. Starting from frame 0, I will scan through the images focusing on furniture that matches a typical chest of drawers. I observed a chest of drawers in frames 53 and 54 located in the kitchen area, which confirms the presence in those frames.

frame_indices: [53]

Meta Analysis: Misidentifies kitchen counter as target object

✓ **Gemini-2.0-flash:** The robot performs the following actions: - Picks up a green spray bottle and blue lunchbox from a chair and places it on a table. - Picks up an apple from a table and places it on another chair. - Picks up the lunchbox from the table and puts it in the bathroom sink. - Picks up a shoe from a shelf and places it next to a lamp on a bedside table. - Picks up a green alarm clock from a dresser and places it near the washing machine. - Picks up a panda figurine from a shelf and places it next to the lamp. - Picks up the spray bottle from the bedside table. - Picks up the panda figurine from bedside table and places it near a chair. The user wants to navigate to a chest of drawers. The video shows chest of drawers at [00:37], [00:47], [00:53], [00:71], [00:78]. NUM_TARGETS_TO_REVISIT: ||1||. TIMESTAMP_INDEX: ||00:37||

frame_indices: [37]

✗ **Qwen-2.5-VL:** The robot needs to navigate to a chest of drawers. In the video, the robot moves through various rooms and eventually reaches a room with a chest of drawers visible on the right side of the screen.

frame_indices: [49]

Meta Analysis: Selects a frame too distant from the target despite a closer one being available

✗ **Gemma-3:** The robot needs to navigate to a chest of drawers. I will examine the video frames to identify the chest of drawers and select the frame index that shows the chest of drawers clearly. The chest of drawers appears in frames 71, 72, and 78. Frame 71 shows the chest of drawers clearly, so I will select that frame index.

frame_indices: [71]

Meta Analysis: Selects a frame too distant from the target despite a closer one being available

✗ **Text Agent:** The goal is to navigate to a chest of drawers. From the history, we see that the agent navigated through various rooms including the bedroom and living room. The dresser mentioned in the placing action at frame index 672 could be a chest of drawers. Therefore, the most relevant frame index to achieve the goal is 672.

frame_indices: [672]

Meta Analysis: Selects a frame where target objects is not visible but was visible in nearby frames

Interaction

Task: Navigate to any receptacle you placed an object on.

✓ **GPT-4o:** To find a frame where the robot placed an object on a receptacle, I analyzed the video frames. In Frame 15, the robot places a lunchbox on a table. This frame clearly shows the object being placed, which matches the goal of navigating to a receptacle where an object was placed. Thus, Frame 15 is the most relevant frame for the task.

frame_indices: [15]

✗ **Gemini-2.0-flash:** Here's a breakdown of the objects picked and placed in the video, to help answer the user's question: * **Spray bottle:** Picked from a chair (00:03), placed on a nightstand (01:16). * **Lunchbox:** Picked from a chair (00:03), placed on a bathroom sink (00:25). * **Apple:** Picked from a table (00:07), kept on the ottoman (01:34). * **Shoe:** Picked from shelf (00:33), placed under lamp (00:41). * **Alarm clock:** Picked from a side table (00:55), kept on bathroom sink (01:05). * **Toy:** Picked from a table (01:27), placed on a chair (01:33). The user wants to navigate to any receptacle where an object was placed. So, we can choose any of the destination receptacles. Let's pick the ottoman (where the apple was placed). NUM_TARGETS_TO_REVISIT: ||1||. TIMESTAMP_INDEX: ||01:34||

frame_indices: [94]

Meta Analysis: Misidentifies the location of placing the apple which is the table

✗ **Qwen-2.5-VL:** The robot needs to navigate to a receptacle where it previously placed an object. From the video, we can see that the robot has placed various objects on different surfaces such as tables, chairs, and dressers. To achieve the goal, the robot should identify the last object it placed and then move to the receptacle where it was placed.

frame_indices: [12, 25, 47, 63, 78, 93]

Meta Analysis: Provides incorrect reasoning and identifies multiple frames (some incorrect) instead of single frame

✗ **Gemma-3:** The robot needs to navigate to the receptacles where it placed objects. Let's analyze the video to identify those frames. 1. **Frame 11:** The robot places a suitcase on a table. 2. **Frame 26:** The robot places a suitcase on a sink. 3. **Frame 65:** The robot places a clock on a sink. 4. **Frame 93:** The robot places a suitcase on a chair. Therefore, the frame indices to navigate to are 11, 26, 65, and 93.

frame_indices: [11, 26, 65, 93]

Meta Analysis: Incorrect event tracking and reasoning leads to misidentified frames

✓ **Text Agent:** The goal is to navigate to any receptacle where an object has been placed. From the history, the last action was placing an alarm clock on the dresser in the bedroom. Therefore, I should navigate to the bedroom.

frame_indices: [576]

Meta Analysis: Incorrect event tracking but still selects a valid frame

Figure D.10: FINDINGDORY Spatial Tasks for the sample episode in Figure D.8. Each VLM model uses 96 subsampled frames. *Text Agent* processes full video as defined in appendix D.4.4.

Single-Goal Spatial Tasks

Conditional Interaction

Task: Navigate to the receptacle that you picked the backpack from.

X GPT-4o: The task requires the robot to return to the receptacle where it previously picked up the backpack. By reviewing the sequence of frames in the video, frames 3 through 17 show the robot's interaction with the backpack. The backpack is first clearly seen on a chair in frame 3, and the robot interacts with it until frame 17. After frame 17, the robot seems to have moved away from the chair with the backpack. Thus, the frame closest to the object (the chair with the backpack) before it was picked is frame 3.

frame_indices: [3]

Meta Analysis: Ignores backpack that was picked from the table and focuses on "distractor" backpack that agent never interacted with

X Gemini-2.0-flash: Here's a breakdown of the video and the solution: ****Video Summary:**** * The robot picks up a green spray bottle and a dinosaur lunchbox from a chair (00:03). * The robot picks up a red apple from a table (00:07). * The robot picks up the dinosaur lunchbox from a table (00:15). * The robot places the dinosaur lunchbox on a bathroom sink (00:26). * The robot picks up a black-and-blue sneakers from a shelf (00:33). * The robot places a black-and-blue sneakers on the cabinet next to a table lamp (00:41). * The robot picks up the green alarm clock (00:56). * The robot placed the alarm clock next to the lunch box on the bathroom counter (01:05). * The robot picks up the green spray bottle and panda toy from the cabinet next to the lamp (01:16). * The robot places the green spray bottle and the lunchbox back on the chair (01:33). ****Task:**** Navigate to the receptacle that you picked the backpack from. ****Solution:**** The robot picked the backpack from the chair at 00:03. Therefore, the response is: NUM_TARGETS_TO_REVISIT: ||1||. TIMESTAMP_INDEX: ||00:03||

frame_indices: [3]

Meta Analysis: Correctly tracks events but makes error in reasoning over the events

✓ Qwen-2.5-VL: The robot needs to navigate back to the location where it picked up the backpack. From the video, the backpack was picked up on frame 16 when the robot was in the bathroom. Therefore, the robot should move to the frame where the backpack was last seen.

frame_indices: [16]

Meta Analysis: Backpack was dropped in the bathroom

✓ Gemma-3: The robot picked up the backpack from a receptacle. I need to find the frame index where the receptacle is visible. The backpack was picked up in frames 15-16. The receptacle is the small table with a dinosaur-themed bag on it. The frame index where the receptacle is visible is frame 15.

frame_indices: [15]

X Text Agent: The goal is to navigate to the receptacle where the backpack was picked. From the history, we see that the backpack was picked in the bedroom at frame index 384. Therefore, we should navigate to the bedroom.

frame_indices: [384]

Meta Analysis: Incorrect event tracking

Spatial Relationship

Task: Navigate to the receptacle that you picked an object from which is the farthest from your current location.

X GPT-4o: The robot picked up objects from various receptacles during the navigation. To determine the farthest receptacle from the current location, it's important to identify the locations where the robot previously picked up items and then calculate which of these locations is the farthest from the end point of the video. Key frames to consider include: - Frame 4: Picked up an item from the chair near the glass door. - Frame 17: Picked up an item from a table in what seems to be the living area. - Frame 41: Picked up a shoe from a different location. The current location (final frame) seems to indicate the robot is in a bedroom. Relative to this, the farthest picking location would likely be Frame 4, where the robot picked an item up from a chair near the glass door. This probable distance is because the living room and shoe area seem to be closer to the bedroom than the chair near the glass door. Thus, the robot should navigate to the receptacle at Frame 4.

frame_indices: [4]

Meta Analysis: Incorrect description of final agent location as bedroom. In addition, chair from which item was picked is visible clearly visible in final frame

X Gemini-2.0-flash: Here's how we can solve this problem: 1. ****Summarize the Robot's Actions:**** * ****Chair:**** * Picked up a spray bottle and a lunchbox from the chair near the sliding doors (00:03). * Placed the spray bottle on the nightstand next to the lamp in the bedroom (01:16). * Picked up an apple from the table in the balcony (00:07). * Dropped the lunchbox on the bathroom sink (00:26). * ****Table:**** * Picked up a shoe from the shelf behind the sofa (00:33). * Dropped the shoe to the nightstand (00:41). * ****Washing Machine:**** * Picked up an alarm clock from the top of washing machine(00:56). * Dropped the alarm clock near the lunchbox in the bathroom sink (01:08). * ****Chair near sofa:**** * Picked up a toy from the chair near the sofa(01:27). * Dropped the lunchbox near the chair in the balcony (01:35). 2. ****Determine Distances:**** * The chair near the sliding doors and washing machine appears to be the farthest receptacles from current position (frame 00:00). 3. ****Identify the First Receptacle:**** The first receptacle is the chair near the sliding doors. 4. ****Final Answer:**** Based on the analysis, the final answer is: NUM_TARGETS_TO_REVISIT: ||1||. TIMESTAMP_INDEX: ||00:03||

frame_indices: [3]

Meta Analysis: Incorrect event tracking and no clear spatial reasoning exhibited

X Qwen-2.5-VL: The robot needs to navigate to the farthest receptacle from its current location. It has previously picked up a red object from a table and a lunchbox from a sink. The red object is on a table, and the lunchbox is on a sink. The table is located in the dining area, and the sink is in the bathroom. The dining area is further away from the current location than the bathroom. Therefore, the robot should move towards the dining area.

frame_indices: [12]

Meta Analysis: Agent final location is close to the chair from which the red object was picked

X Gemma-3: The robot's goal is to navigate to the receptacle that it picked an object from which is the farthest from its current location. The robot picked up a blue object from a table in the first few frames. The receptacle is the table. The table is located in the outdoor area of the house. The robot is currently in the bathroom. Therefore, the robot should navigate to frame 84, which shows the table in the outdoor area.

frame_indices: [84]

Meta Analysis: Incorrect event tracking and spatial reasoning

X Text Agent: To determine the farthest receptacle from the current location, we need to calculate the distance between the agent and each receptacle it has interacted with. However, since we don't have explicit distance data, we'll assume the order of rooms visited reflects the order of interaction. The farthest room from the starting point would likely be the last one visited. In this case, the last room visited is the 'living room' at frame index 1308.

frame_indices: [1308]

Meta Analysis: No spatial reasoning and reliance on unfounded assumptions

Figure D.11: FINDINGDORY Spatial Tasks for the sample episode in Figure D.8. Each VLM model uses 96 subsampled frames. *Text Agent* processes full video as defined in appendix D.4.4.

Single-Goal Spatial Tasks

Object Attributes

Task: Navigate to a red colored object again that you interacted with yesterday.

✓ **GPT-4o:** The video shows consecutive frames captured by a robot inside a house. The task is to identify the frame where the robot had an interaction with a red-colored object. Upon reviewing the frames, the robot interacted closely with a red apple. This interaction can be observed between frames 6 and 11, with the robot's gripper appearing close to the apple. Therefore, the frame where the robot seems closest to the apple is frame 7.
frame_indices: [7]

✓ **Gemini-2.0-flash:** The robot picks up the following objects and places them at different locations: - A green spray bottle and a lunch box from a chair and places them on a table in the bathroom (frame 26). - A red apple from a table and does not drop it, implies it is still holding it (frame 11). - A shoe from a shelf and drops it next to a lamp in a bedroom (frame 41). - An alarm clock from a counter and drops it near the lunch box in the bathroom (frame 65). - A panda toy from a shelf in a bedroom and drops it near the spray bottle (frame 77). The user wants to revisit the red colored object (red apple). Since the robot never dropped it, we should revisit the table where it picked up the apple to see if it placed it back there. NUM_TARGETS_TO_REVISIT: ||1||. TIMESTAMP_INDEX: ||00:07||
frame_indices: [7]

Meta Analysis: Does not identify the dropping of apple on table but selects correct frame based on where the apple was viewed

✓ **Owen-2.5-VL:** The robot needs to navigate back to the red object it interacted with yesterday. From the video frames, the red object appears on a table in the living room at frame 10.
frame_indices: [10]

✗ **Gemma-3:** The robot's goal is to navigate to a red colored object that it interacted with yesterday. Looking through the video, I can see the robot interacting with a red suitcase multiple times. The first time the robot interacts with the red suitcase is around frame 15. The robot then interacts with the suitcase again around frame 25, 66, 68, 86, 92, and 95. I will choose frame 15 as the frame index to navigate to the red suitcase.
frame_indices: [15]

Meta Analysis: Fails to identify the red apple object that was dropped on the table

✗ **Text Agent:** To accomplish the goal of navigating to a red colored object, I need to identify the red object from the history. From the given history, there is no explicit mention of a red object. However, if we assume the red object was picked or placed at some point, we should look for the most recent navigation action involving a red object. Since the history does not specify the color, we'll choose the last navigation action as it might be the most recent interaction.
frame_indices: [1308]

Meta Analysis: Fails to identify the red apple object that was dropped on the table

Room Visitation

Task: Navigate to the room where you placed the backpack in.

✗ **GPT-4o:** The robot needs to find the room where it placed the backpack. While reviewing the frames, the backpack appears prominently in frames 3, 25, and 93, among others. The most recent placement seems to be in frame 93, which occurs at time 11:06, showing the backpack is located on a chair initially visible in the earlier part of the video. This suggests that the robot placed the backpack there later in the sequence since it was previously interacting with it near the bathroom sink (frame 25). Thus, the correct location the robot should navigate to is found in frame 93.
frame_indices: [93]

Meta Analysis: Focuses on "distractor" backpack that agent never interacted with

✓ **Gemini-2.0-flash:** Here's a breakdown of the objects picked and placed, to help answer the question: * **Spray Bottle:** Picked from the chair at 00:03, placed on the nightstand at 01:16 * **Red Apple:** Picked from the table at 00:07, placed on the chair at 00:03 * **Shoe:** Picked from shelf at 00:33, placed near the lamp at 00:41 * **Lunchbox:** Picked from table at 00:15, placed on bathroom counter at 00:26 * **Clock:** Picked from the drawer at 00:56, placed on the lunchbox on bathroom counter at 01:05 * **Panda Toy:** Placed on the nightstand at 01:16, picked from the nightstand lamp table at 01:16, placed on table at 00:15 The lunchbox (referred to as "backpack" in the prompt) was placed on the bathroom counter. So, to navigate to where the backpack was placed, we should go to the bathroom. NUM_TARGETS_TO_REVISIT: ||1||. TIMESTAMP_INDEX: ||00:26||
frame_indices: [26]

✓ **Owen-2.5-VL:** The robot needs to navigate to the room where it placed the backpack. From the video, we can see the backpack being placed on a sink in a bathroom. The robot should move to this location to find the backpack.
frame_indices: [25]

✗ **Gemma-3:** The robot needs to navigate to the room where the backpack was placed. From the video, the backpack was placed on a chair in a room with a table and a plant. The frames showing the backpack on the chair are frames 93 and 95. I will choose frame 93 as it is a clearer view of the backpack and the surrounding environment.
frame_indices: [93]

Meta Analysis: Focuses on "distractor" backpack that agent never interacted with

✗ **Text Agent:** To determine the room where the backpack was placed, I need to find the frame where the agent was placing an object. From the history, the last frame where the agent was placing an object was at frame 672, where the alarm clock was placed on the dresser in the bedroom. Since the goal is to navigate to the room where the backpack was placed, we should look for a frame where the agent was navigating to a specific room. The next navigation frame is at 768, which is in the bathroom. However, there is no indication that the backpack was placed in the bathroom. The next navigation frame is at 864, which is in the bathroom as well. The next navigation frame is at 960, which is in the bedroom. Therefore, the backpack was likely placed in the bedroom.
frame_indices: [960]

Meta Analysis: Misses tracking frames where backpack is picked from table in living room and placed in bathroom sink

Figure D.12: FINDINGDORY Spatial Tasks for the sample episode in Figure D.8. Each VLM model uses 96 subsampled frames. *Text Agent* processes full video as defined in appendix D.4.4.

Single-Goal Temporal Tasks

Interaction Order

Task: Navigate to the object you that you interacted with between the interactions with apple and shoe.

✓ **GPT-4o:** First, identify the interactions involving the apple and shoe. The apple interaction occurs at frames 7 to 11. The shoe interaction occurs at frames 33 to 43. **Between these interactions, the robot interacted with a lunchbox. The relevant frames for this interaction are from 25 to 28 and 64 to 68.** The robot should navigate to the lunchbox in this task.
frame_indices: [26]

✗ **Gemini-2.0-flash:** Here's a breakdown of the robot's actions in the video: 1. ****00:03**:** Robot picks up a spray bottle and lunchbox from a chair. 2. ****00:07**:** Robot picks up an apple from a table. 3. ****00:15**:** Robot places the lunchbox on another table. 4. ****00:25**:** Robot places the lunchbox on the bathroom sink. 5. ****00:33**:** Robot picks up a shoe from a shelf. 6. ****00:41**:** Robot places the shoe under a lamp on a nightstand. 7. ****00:55**:** Robot picks up an alarm clock from a table. 8. ****01:05**:** Robot places the alarm clock near the lunchbox on the bathroom sink. 9. ****01:16**:** Robot places the spray bottle on a nightstand. **The object interacted with between the apple and shoe is the lunchbox. It was picked up at 00:03 and placed on the sink at 00:25. NUM_TARGETS_TO_REVISIT: ||1||. TIMESTAMP_INDEX: ||00:03||**
frame_indices: [3]

Meta Analysis: Correct event tracking and target identification but incorrectly navigates to the object picking location instead of final placement location

✗ **Qwen-2.5-VL:** The robot needs to navigate from the apple interaction to the shoe interaction. The apple interaction occurred at frame 10, and the shoe interaction occurred at frame 35. **Therefore, the robot should move to the frame right after the apple interaction.**
frame_indices: [11]

Meta Analysis: Fails to identify the backpack target object for navigation

✗ **Gemma-3:** The robot interacted with an apple in frame 7 and a shoe in frame 43. **The object between these interactions is the green alarm clock.** The robot interacted with the alarm clock in frame 55. Therefore, the frame index to move to is 55.
frame_indices: [55]

Meta Analysis: Incorrect target object identification due to temporal reasoning error

✗ **Text Agent:** The goal is to navigate to the object you interacted with between the interactions with apple and shoe. **From the history, the interaction with the apple occurred at frame 672 and the interaction with the shoe occurred at frame 384.** Therefore, we need to find the frame index between these two events.
frame_indices: [576]

Meta Analysis: Incorrect event tracking and ignores the initial interaction with the apple

Time-Based

Task: Navigate to the object that you interacted with at 8:26 yesterday.

**For this task, the query timestamp can be slightly different but target solution object remains constant

✓ **GPT-4o:** To find the frame index where the robot should move, we need to first identify the object that was interacted with at 8:06. Reviewing the timestamps on the images, we can see that the robot interacted with an object around that time. At 8:08 (frame 39), the robot is near the nightstand with a lamp and small weights. The robot's interaction here suggests it is the object of interest. **Moving a bit further into frames, the shoe is being interacted with from frame 39 to frame 41.** Therefore, the frame closest to this moment, and thus relevant for the task, is frame 39.
frame_indices: [39]

✓ **Gemini-2.0-flash:** The robot performs the following actions: - Picks up a spray bottle from the chair (00:03), - Picks up an apple from the table (00:07), - Picks up a lunchbox from the table (00:16), - Places lunchbox on bathroom sink (00:26), - **Picks up a shoe from the shelf (00:33), - Places shoe on the nightstand (00:41),** - Picks up an alarm clock from the table next to the washing machine (00:56), - Places alarm clock on the bathroom sink next to lunchbox (01:05), - Picks up a panda figurine next to the lamp (01:16), - Places spray bottle next to the lamp (01:17). The user is asking to revisit the object that was interacted with at 08:21, which is the shoe. **NUM_TARGETS_TO_REVISIT: ||1||. TIMESTAMP_INDEX: ||00:41||**
frame_indices: [41]

✓ **Qwen-2.5-VL:** The robot needs to navigate back to the object it interacted with at 8:26 yesterday. **From the video frames, the object it interacted with appears to be a shoe on the floor near the bed at frame 43.**
frame_indices: [43]

✗ **Gemma-3:** **The robot interacted with a suitcase at 08:27. I need to find the frame index where the robot is closest to the suitcase.** Looking at the images, the suitcase is visible in frames 64, 65, 66, 67, 68, 69, 88, 90, 92, 93, 94, 95. **Frame 93 seems to be the closest to the suitcase.**
frame_indices: [93]

Meta Analysis: Identifies the incorrect target object. In addition, focuses on the "distractor" backpack which was not interacted with

✗ **Text Agent:** To determine the frame index corresponding to the interaction at 8:27 yesterday, I need to find the timestamp closest to 8:27. **The timestamps provided are in a 24-hour format, so 8:27 would be 20:27 in military time. The closest timestamp in the history is 09:27, which corresponds to the frame indices 864 and 960.** Since the goal is to navigate to the object interacted with at 8:27, we should choose the frame index that aligns with this timestamp.
frame_indices: [864]

Meta Analysis: Erroneous conversion of query timestamp to a new value (09:27). But retrieves the correct object corresponding to interaction at 09:27

Figure D.13: FINDINGDORY Temporal Tasks for the sample episode in Figure D.8. Each VLM model uses 96 subsampled frames. *Text Agent* processes full video as defined in appendix D.4.4.

Single-Goal Temporal Tasks

Duration Tracking

Task: Navigate to to the room that you spent most time in.

✓ **GPT-4o:** I need to identify which room the robot spent the most time in during the navigation. Analyzing the frames, the setting shifts around a few distinct locations, primarily different rooms in the house. The living room appears frequently, with repetitive views of chairs, tables, and decorative setups. The frames in the living room span from the start, go through frames with consistent furniture and decor like in frames 1-12, and reappear frequently towards the latter part, such as around frames 50-60ish. Other parts like the bathroom or kitchen have fewer frames indicating less time spent. Thus, identifying the living room as the room where the robot spent the most time, the earliest close frame in that setting is a good reference for moving to the dominant room setting.
frame_indices: [1]

✗ **Gemini-2.0-flash:** The robot spends most of its time in the living room and the bedroom. Since both rooms are spent similar amount of time, I will default to choosing the living room. NUM_TARGETS_TO_REVISIT: ||1||. TIMESTAMP_INDEX: ||01:34||
frame_indices: [94]

Meta Analysis: Decides to choose the living room but selects a frame where agent is in the smaller patio area

✗ **Qwen-2.5-VL:** The robot needs to navigate to the room it spent the most time in. Based on the frames, the robot spends more time in the bedroom, as indicated by the higher frame numbers and the presence of personal items like a bed, nightstand, and alarm clock. Therefore, the robot should move to the bedroom.
frame_indices: [40]

Meta Analysis: Ignores the large proportion of frames that are situated in the living room

✗ **Gemma-3:** The robot spent most of its time in the room with the blue floor. The video starts in a room with a blue wall, then moves to a room with a white floor, then to a room with a blue floor. The robot spends a significant amount of time in the room with the blue floor, moving around and interacting with various objects. The robot then moves to a room with a white floor, then to a room with a blue wall, then to a bathroom, and then back to the room with the blue floor. Therefore, the robot should navigate to the room with the blue floor.
frame_indices: [1, 51, 52, 58, 60, 81, 82, 83, 92, 93, 94, 95]

Meta Analysis: Does not reason about the general semantic structure of the rooms. Selects multiple correct frames based on matching the floor color but selects incorrect frames from the patio area due to similar floor color

✓ **Text Agent:** To determine the room where the agent spent the most time, I need to calculate the duration of time spent in each room. The timestamps provide the start and end times for each navigation event. By subtracting the start timestamp from the end timestamp, we can calculate the duration of time spent in each room. The room with the longest duration is the one where the agent spent the most time.
frame_indices: [1248]

Meta Analysis: Selects correct frame but does not provide concrete duration spent in each room

Figure D.14: FINDINGDORY Temporal Tasks for the sample episode in Figure D.8. Each VLM model uses 96 subsampled frames. *Text Agent* processes full video as defined in appendix D.4.4.

Multi-Goal Tasks

Unordered Revisitation

Task: Revisit all the receptacles you picked objects from yesterday.

✓ **GPT-4o:** The video shows the robot navigating through a house while picking up objects from various receptacles. To complete the task, the robot needs to revisit the same receptacles where objects were picked up. 1. Frame 3 shows the robot picking an object from the chair. Revisit this chair. 2. Frame 7 shows the robot picking an object (an apple) from the dining table. Revisit this table. 3. Frame 15 shows the robot picking an object from a round table beside a chair. Revisit this table. 4. Frame 33 shows the robot picking a shoe from a low shelf. Revisit this shelf. 5. Frame 55 shows the robot picking a clock from a countertop. Revisit this countertop. 6. Frame 76 shows the robot picking a spray bottle from a side table near a lamp. Revisit this table. These are the main locations the robot picked objects from, so these frames need to be revisited.

frame_indices: [3, 15, 33, 55, 76]

Meta Analysis: Incorrectly identifies apple being picked from table whereas the apple was dropped on table. But omits index 7 in final solution without any clear reason

✗ **Gemini-2.0-flash:** Here's a breakdown of the objects picked and placed, to determine the receptacles that need revisiting: * **Spray bottle & Lunchbox:** * Picked from: Chair in the living room (00:03) * Placed on: Chair in the living room (00:04), then plant table (00:15), then sink (00:26). * **Apple:** * Picked from: Plant table (00:07) * Placed on: Plant table (00:11). * **Shoe:** * Picked from: Shelf in living room (00:33) * Placed on: Nightstand in bedroom (00:41). * **Lunchbox:** * Picked from: Sink in bathroom (00:27) * Placed on: Ottoman (01:24) * **Clock:** * Picked from: Washing Machine (00:56) * Placed on: Sink in bathroom (01:08) * **Panda Toy:** * Picked from: Nightstand in bedroom (01:16) * Placed on: Chair in living room (01:27) Now, let's identify the timestamps for revisiting the original receptacles: 1. **Chair in living room:** Where the spray bottle and lunchbox were initially placed. (00:03) 2. **Plant table:** Where the apple was initially placed. (00:07) 3. **Shelf in living room:** Where the shoe was initially placed. (00:33) 4. **Washing machine:** Where the clock was initially placed. (00:56) 5. **Nightstand in bedroom:** Where the panda toy was initially placed. (01:16) 6. **Sink in bathroom:** Where the lunchbox was initially placed. (00:27) NUM_TARGETS_TO_REVISIT: ||6||. TIMESTAMP_INDEX: ||00:03,00:07,00:33,00:56,01:16,00:27||

frame_indices: [3, 7, 33, 56, 76, 27]

Meta Analysis: Incorrect event tracking and misidentifies backpack placement frames as pick frames

✗ **Qwen-2.5-VL:** The robot needs to revisit all the receptacles it picked objects from yesterday. From the video, we can see that the robot picked up objects from various receptacles such as a table, a suitcase, a nightstand, and a dresser. To achieve the goal, the robot should revisit these locations in the order they were visited.

frame_indices: [12, 16, 34, 50]

Meta Analysis: Inadequate reasoning and hallucinates suitcase as a receptacle

✗ **Gemma-3:** Error parsing output

frame_indices: [-1]

Meta Analysis: Model fails to produce any coherent output

✗ **Text Agent:** To revisit all the receptacles from which objects were picked yesterday, I need to identify the rooms where objects were picked and then navigate back to those rooms. From the history, we see that objects were picked from the living room, kitchen, bedroom, and bathroom. Therefore, I should navigate to these rooms.

frame_indices: [0, 96, 192, 288, 384, 480, 576, 672, 768, 864, 960, 1056, 1152, 1248]

Meta Analysis: Incorrectly focuses only on room revisitation instead of revisiting specific receptacles

Ordered Revisitation

Task: Revisit all the receptacles you placed objects on yesterday in specific order. The order to revisit them is: fifth, second, first, third.

✗ **GPT-4o:** The robot needs to revisit specific receptacles in order. To do this, it needs to identify the frames where objects were placed: 1. **First Receptacle (Fifth in order of placement):** - Frame 64: Lunchbox on the sink in the bathroom. 2. **Second Receptacle (Second in order of placement):** - Frame 7: Apple on the table in the dining area. 3. **Third Receptacle (First in order of placement):** - Frame 16: Lunchbox on the small table in the living room. 4. **Fourth Receptacle (Third in order of placement):** - Frame 33: Shoe on the shelf in the entryway. The robot should start with the lunchbox on the sink (fifth placement), then the apple on the table (second placement), revisit the lunchbox in the living room (first placement), and finally go to the shoe on the shelf (third placement).

frame_indices: [64, 7, 16, 33]

Meta Analysis: Unable to list out the original object placement events and then adapt to the query revisitation sequence

✗ **Gemini-2.0-flash:** Here's a summary of the objects picked and placed, and their locations: * **Apple:** Picked from the table near the plant (00:07), placed on the table on the balcony (00:03). * **Spray bottle:** Picked from the chair on the balcony (00:03), placed under the lamp on the nightstand (01:16). * **Shoe:** Picked from the shelf (00:33), placed under the lamp on the nightstand (01:41). * **Lunchbox:** Picked from the table (00:15), placed on the bathroom counter (00:25). * **Clock:** Picked from the table next to the washing machine (00:55), placed on the bathroom counter next to lunchbox (01:05). * **Panda:** Picked from under the lamp on the nightstand, placed on the side table in living room (01:27). Given the objects and placements, let's identify the receptacles for revisitation in the required order: 1. The fifth item that was placed was the **clock** on the bathroom counter. 2. The second item that was placed was the **spray bottle** under the lamp on the nightstand. 3. The first item that was placed was the **apple** on the chair on the balcony. 4. The third item that was placed was the **shoe** under the lamp on the nightstand. Here are the timestamps: * **Lunchbox and clock on the bathroom counter:** 00:25 * **Spray bottle and shoe on the nightstand:** 01:16 * **Apple on the chair on the balcony:** 00:03 text NUM_TARGETS_TO_REVISIT: ||4||. TIMESTAMP_INDEX: ||00:25,01:16,00:03,01:16||

frame_indices: [25, 76, 3, 76]

Meta Analysis: Correctly lists out original placement sequence although hallucinates interaction with spray bottle. Confuses picking and placement timestamps while generating final solution

✗ **Qwen-2.5-VL:** The robot needs to revisit the receptacles it placed objects on yesterday in the specified order: fifth, second, first, third. To achieve this, we need to identify the frames where these receptacles were visited. The frames provided show the robot interacting with various objects and moving through different rooms. We need to find the frames corresponding to the receptacles in the given order.

frame_indices: [59, 13, 26, 48]

Meta Analysis: No clear reasoning on how the original placement sequence was adapted to the query revisitation sequence

✗ **Gemma-3:** Error parsing output

frame_indices: [-1]

Meta Analysis: Model fails to produce any coherent output

✗ **Text Agent:** To accomplish the goal of revisiting the receptacles in the specified order, I need to identify the receptacles that were placed with objects. From the history, the receptacles mentioned are 'dresser' and 'sink'. The order to revisit them is fifth, second, first, third. Based on the timestamps and room names, the 'dresser' was visited at frame indices 672 and 768, and the 'sink' was visited at frame indices 288 and 384. The order given is fifth, second, first, third, so we should start with the 'dresser' (frame indices 672 and 768), then the 'sink' (frame indices 288 and 384).

frame_indices: [672, 768, 288, 384]

Meta Analysis: Only lists out two receptacles on which objects were placed but task requires revisiting all of them

Figure D.15: FINDINGDORY Multi-Goal Tasks for the sample episode in Figure D.8. Each VLM model uses 96 subsampled frames. *Text Agent* processes full video as defined in appendix D.4.4.