LASAR: Towards Spatio-temporal Reasoning with Latent Cognitive Map

Supplementary Material

6. Model and Training Details

6.1. Train Data

To accomplish the dual tasks of visual navigation and spatial reasoning, our training dataset is constructed directly upon the **VLN-CE (Vision-and-Language Navigation in Continuous Environments) training set**.

Our core training curriculum is the **MindCraft-Train dataset**. As detailed in the main paper (Section 3.3), this dataset is generated by our procedural pipeline, which traverses the expert trajectories provided by VLN-CE within the Matterport3D environments. For each trajectory, our pipeline injects the three types of spatio-temporal cognitive queries (retrospective, introspective, and prospective) and logs the ground-truth answers.

This approach ensures that the agent is trained on a unified dataset where navigation actions and reasoning queries are concurrent. All models, including baselines adapted for our task, were trained exclusively on this dataset, ensuring a fair comparison. This single-source training setup is critical for validating our hypothesis that the reasoning task acts as an effective auxiliary supervision for improving navigation, and allows for a true **zero-shot** evaluation on downstream tasks like VSI-Bench, as neither the environments (for val-unseen splits) nor the general VQA task formats were seen during training.

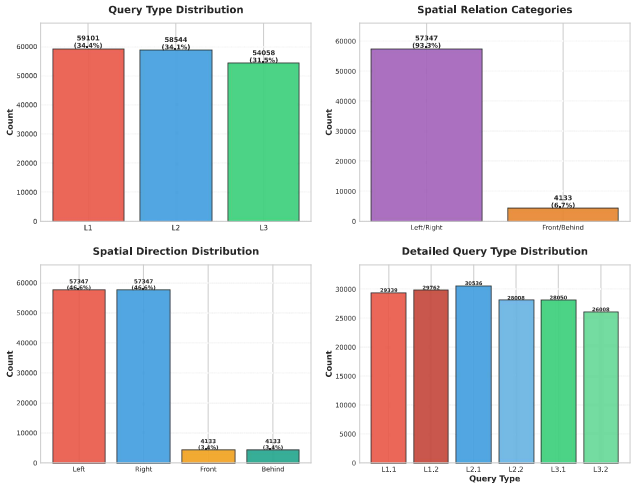

Figure 5. **MindCraft Data Statistics.**

6.2. MindCraft Dataset Generation

For each expert trajectory, our pipeline programmatically generates a corresponding MindCraft dataset through a structured, three-stage process designed to ensure that all generated queries are contextually relevant, unambiguous, and grounded in the agent's experiential history.

First, during Trajectory Traversal and Logging, a virtual agent precisely follows the expert path. At each step, it constructs a "Path Memory Log," a comprehensive record of its sensory experience. This log systematically captures the set of visible semantic objects, their properties, and the agent's spatiotemporal state, effectively chronicling what was perceived, where, and when.

Second, in the Cognitive Queries Generation stage, the pipeline synthesizes a query that targets a specific facet of spatial cognition. Our framework defines six query categories across three cognitive tiers: L1) Memory Recall (Retrospective Queries), L2) Situational Awareness (Introspective Queries), and L3) Predictive Reasoning (Prospective Queries). Crucially, the instantiation of a query is not arbitrary but is instead governed by a set of strict, task-specific preconditions. The pipeline first assesses the trajectory log to determine which types of cognitive queries are valid and meaningful to generate for that specific path. For instance, an Object Attribute Recall query is contingent upon the agent having previously observed an object that meets two criteria: (1) its bounding box occupied over 10% of the viewport area at the time of observation, and it is no longer in the current field of view. (2) To ensure referential unambiguity, the pipeline validates against the simulator's ground-truth scene graph that no other instance of the same semantic category (e.g., another "chair") is present within a 5-meter radius of the original observation's location.

Only the query types whose preconditions are satisfied by the agent's memory log become candidates for generation. From this valid set, one is selected and formulated.

Finally, during Ground-Truth Answer Generation, the pipeline programmatically derives the correct answer by leveraging the simulator's ground-truth information. The answer is computed based on the veridical state of the 3D environment and the agent's logged history. For example, the spatial relationship between the agent and a recalled object is determined by its position within the agent's egocentric visual field at the moment of initial observation.

6.3. Train Strategy

We train the LASAR model end-to-end using the composite objective function described in the main paper. The model is optimized to jointly predict the navigation action a_t (via imitation learning) and, when present, answer the cognitive query ans_t (via sequence generation). To enhance the model's performance on video spatial understanding tasks, we fine-tuned the Vicuna-7B model using LoRA (Low-Rank Adaptation). The core idea of LoRA is to achieve parameter-efficient optimization by inserting trainable low-rank matrices without making large-scale adjustments to the original weights of the pre-trained model. This approach allows the model to adapt to downstream task requirements while maintaining its performance.

Specifically, LoRA decomposes the weight matrix $W \in \mathbb{R}^{d\times k}$ into two low-rank matrices $A \in \mathbb{R}^{d\times r}$ and $B \in \mathbb{R}^{r\times k}$. Through low-rank decomposition, the weight update formula can be expressed as $W' = W + A \cdot B$. During this process, the original weights W remain frozen, and only the newly added low-rank matrices A and B are optimized. We inserted LoRA modules into key components of the model, including the query projection

(q_proj), value projection (v_proj), and language model head (lm_head), to enhance the model's ability to model dynamic relationships between video frames and perform cross-modal reasoning.

The full loss $\mathcal{L}_{\text{total}}$ combines the main task loss $\mathcal{L}_{\text{MindCraft}}$ with the three crucial auxiliary losses: $\mathcal{L}_{\text{crl}}$ (Spatio-temporal Contextual Representation Learning), $\mathcal{L}_{\text{sem}}$ (Semantic Atlas Learning), and $\mathcal{L}_{\text{retro}}$ (Episodic Discriminability).

For video spatial understanding tasks, the use of LoRA allows us to efficiently optimize the model, making it more adaptable to downstream tasks while maintaining reasonable computational resource usage. This fine-tuning approach provides strong technical support for our video spatial understanding tasks.

6.4. Hyperparameter Details

To ensure the reproducibility of our results, we provide a comprehensive list of the key hyperparameters used for training the LASAR model. We utilized the AdamW optimizer for stable and efficient training. The primary hyperparameters for the optimization and training process are summarized in Table 3. These settings were kept consistent across all main experiments unless otherwise specified in the ablation studies.

Table 3. Key hyperparameters for training the LASAR framework.

Hyperparameter	**Value**
Optimizer & Scheduler	
Optimizer	AdamW
Learning Rate (Peak)	1×10^{-4}
Betas (β_1, β_2)	(0.9, 0.999)
Weight Decay	0.01
Learning Rate Schedule	Cosine decay
Warmup Epochs	1
Training & Batching	
Total Training Epochs	10
Per-Device Batch Size	8
Gradient Accumulation Steps	8
Effective Batch Size	64 (8×8 GPUs)
Mixed Precision	bfloat16
LoRA Fine-tuning	
LoRA Rank (r)	16
LoRA Alpha (α)	32
LoRA Dropout	0.1
LoRA Target Modules	q_proj, v_proj, lm_head
Loss Configuration	
Query Answer Weight (λ_{qa})	1.0
ST-CRL Loss Weight (λ_c)	0.1
Semantic Atlas Loss Weight (λ_s)	0.2
Episodic Loss Weight (λ_r)	0.1
Contrastive Temperature (τ)	0.07
ST-CRL Negative Samples (N)	64

LoRA Configuration. As detailed in the 'Train Strategy' subsection, we employed Low-Rank Adaptation (LoRA) for parameter-efficient fine-tuning. The rank (r) was set to 16, providing a good balance between expressiveness and parameter efficiency. The scaling factor (α) was set to 32. To prevent overfitting on the adapter weights, a dropout rate of 0.1 was applied specifically to the LoRA modules. We targeted the query (q_proj) and value (v_proj) projections within the VLM's self-attention layers, as well as the final language model head (lm_head), as these are critical for adapting the model's reasoning and generation capabilities to our specific tasks.

7. Model Architecture Details

Our proposed framework, LASAR, is constructed around a central Vision-Language Model (VLM) and is augmented by a series of specialized encoders. Each encoder is designed to process a specific modality (vision, geometry, trajectory), transforming raw sensory input into rich feature representations. These features are then projected into a common embedding space, allowing the core VLM to perform holistic reasoning across all available information streams. Below, we detail the specifics of each key component.

Core VLM Backbone. The heart of our model is the **Vicuna-7B (v1.5)**, a powerful pretrained VLM that serves as our primary reasoning engine and Unified Decision Head. We leverage its advanced capabilities in understanding and integrating vision and language. The input to this model is a carefully structured sequence of embeddings from the episodic memory and the spatial semantic memory, as described in the main paper.

Visual Semantic Encoder. To extract high-level semantic information from each RGB frame I_t, we employ a frozen pretrained **Siglip** model. For each frame, we extract the final feature representation $F_{\text{vis},t}$, which captures the objects and their appearance. These features are then passed through a projection layer to match the VLM's hidden dimension, forming the input for the Geometric-Semantic Fusion module.

Image-based Geometric Encoder. Complementary to the semantic features, we extract immediate, view-dependent geometric cues directly from the 2D image I_t. For this, we use a specialized visual-geometry encoder, specifically a version of **VGGT** [40] pretrained on multiple geometry estimation tasks (e.g., point cloud prediction from 2D). This model processes the RGB frame and outputs a dense feature map that implicitly encodes geometric information $F_{\text{geo},t}$. These geometric features provide the spatial context necessary to ground the semantic features $F_{\text{vis},t}$ through our cross-attention mechanism, as defined in the main paper (Equation 2).

8. Evaluation Protocol Details

8.1. Overall Reasoning Accuracy (QA-Acc)

This metric measures the overall accuracy of the agent's answers to all cognitive queries in the test set. Let $\mathcal{Q}$ be the set of all queries in

the test set. For any query $q \in \mathcal{Q}$, let A_q be the agent's answer and G_q be the ground-truth answer. Let $\mathbb{I}(\cdot)$ be the indicator function.

The Overall Reasoning Accuracy (QA-Acc) is calculated as:

$$\text{QA-Acc} = \frac{1}{|\mathcal{Q}|} \sum_{q \in \mathcal{Q}} \mathbb{I}(A_q = G_q)$$

8.2. Goal-Conditioned Accuracy (GCA)

This metric measures the query accuracy *only* on trajectories where the agent completed its navigation task (i.e., Success Rate $SR = 1$). Let $\mathcal{T}_{succ}$ be the set of all trajectories where navigation was successful. Let $\mathcal{Q}_{succ} \subseteq \mathcal{Q}$ be the subset of all queries that were posed during the trajectories in $\mathcal{T}_{succ}$.

The Goal-Conditioned Accuracy (GCA) is calculated as:

$$\text{GCA} = \frac{1}{|\mathcal{Q}_{succ}|} \sum_{q \in \mathcal{Q}_{succ}} \mathbb{I}(A_q = G_q)$$

8.3. Cognitive Map Consistency (CMC)

This metric measures the stability of the agent's internal cognitive map. This metric calculates the consistency of the agent's answers to **Equivalent Probe Sets**. An "Equivalent Probe Set" S_i contains k_i queries $\{q_1, \ldots, q_{k_i}\}$ that all target the **same underlying spatial fact** and are designed to have the **exact same ground-truth answer**.

The sources for constructing these sets S_i include:

1. **Template & Relational Equivalence:** The query's phrasing or logical frame of reference is different, but the semantics and ground-truth answer are identical. *(e.g., "Is the sofa to the left of the lamp?" vs. "Is the lamp to the right of the sofa?" GT is "Yes" for both)*
2. **Temporal Misalignment:** Queries within the same trajectory that are based on temporally shifted clips (e.g., offset by 1-2 steps), but the core fact and GT answer remain unchanged.
3. **Cross-Query-Type Equivalence:** Targeting the same fact using queries of different cognitive types (e.g., introspective, retrospective). *(e.g., An **introspective** query at $t = 10$: "What color is the vase you see now?" (GT: "Red") vs. a **retrospective** query at $t = 25$: "What color was the vase you passed earlier?" (GT: "Red"))*
4. **Cross-Trajectory Metadata:** Queries from different trajectories that are determined (via metadata like environment/object ID) to be asking about the same fact with the same GT answer.

We partition all queries into M such Equivalent Probe Sets $\mathcal{S} = \{S_1, S_2, ..., S_M\}$. We only consider sets where $k_i \geq 2$.

1. **Calculate Total Consistent Pairs (C_{total}):** We sum the number of pairs of queries within each set S_i that received an identical answer ($A_{q_m} = A_{q_n}$).

$$C_{total} = \sum_{i=1}^{M} \left(\sum_{m=1}^{k_i - 1} \sum_{n=m+1}^{k_i} \mathbb{I}(A_{q_m} = A_{q_n}) \right)$$

2. **Calculate Total Pairwise Comparisons (P_{total}):** We sum the total number of unique pairwise comparisons possible within each set S_i. $\binom{k_i}{2}$ is the binomial coefficient "k choose 2".

$$P_{total} = \sum_{i=1}^{M} \binom{k_i}{2} = \sum_{i=1}^{M} \frac{k_i(k_i - 1)}{2}$$

3. **Calculate Final CMC:** The CMC is the ratio of total consistent pairs to the total pairwise comparisons.

$$\text{CMC} = \frac{C_{total}}{P_{total}}$$

8.4. Reasoning Failure Impact (SR@WA)

This metric measures the Navigation Success Rate *only* on trajectories where the agent answered **at least one query incorrectly**. Let $\mathcal{T}$ be the set of all trajectories. Let $\mathcal{T}_{WA} \subseteq \mathcal{T}$ (Wrong Answer) be the subset of trajectories containing at least one incorrect answer. For any trajectory $t \in \mathcal{T}$, let $S(t) \in \{0, 1\}$ be its navigation success status (1 = success, 0 = failure).

The SR@WA is calculated as:

$$\text{SR@WA} = \frac{1}{|\mathcal{T}_{WA}|} \sum_{t \in \mathcal{T}_{WA}} S(t)$$

8.5. Cross-dataset Evaluation

Metrics Calculation. All metrics were calculated using the official evaluation scripts provided by the respective benchmarks. For VLN-CE, we report Success Rate (SR), Success rate weighted by Path Length (SPL), **and for the R2R split, Oracle Success (OS)**. For VSI-Bench, we report Accuracy (ACC) for multiple-choice questions and Mean Relative Accuracy (MRA) for numerical questions.

Inference Setup. All our models were evaluated on a single NVIDIA A100 GPU. We used greedy decoding (i.e., beam size of 1) for generating all responses to ensure efficiency and deterministic outputs. All reported scores are the average of three evaluation runs with different random seeds to ensure statistical stability.

9. Additional Experimental Results

9.1. Baseline details

Specialized Navigation Models For all specialized navigation baselines on VLN-CE, including `Navid`, `NaviLLM`, and `NaVILA`, we adhered to the following protocol to ensure a fair and reproducible comparison:

- **Source:** We utilized the official codebases and pre-trained model weights released by the respective authors. No architectural modifications were made.
- **Evaluation Protocol:** We followed the standard evaluation scripts and environment setups provided with each baseline's repository. Results were generated on the `val-unseen` split as reported in the original papers.

General Vision-Language Models For all general-purpose VLMs evaluated on VSI-Bench, such as `GPT-4o`, `Gemini-1.5 Pro`, the `LLaVA` series, and the `Qwen` series, we used the following zero-shot evaluation setup:

- **Model Version:** We used the latest available official APIs for proprietary models (e.g., `gpt-4o-2024-05-13`) and the official Hugging Face implementations for open-source models.
- **Prompting Strategy:** A consistent, minimal prompt template was employed across all models to query their spatial reasoning capabilities without providing few-shot examples. The templates were structured as follows:

For Multiple-Choice Questions:

```
The following is a video of an
indoor scene.  Based on the video,
answer the following question
by choosing the best option.
<video_placeholder> Question:
[Question from VSI-Bench] Options:
(A) [Option A] (B) [Option B] (C)
[Option C] (D) [Option D] Answer
(Provide the letter only):
```

For Numerical Questions:

```
The following is a video of an
indoor scene.  Based on the
video, answer the following
question.  Provide only the
numerical value in your answer.
<video_placeholder> Question:
[Question from VSI-Bench] Answer:
```

9.2. Computational Cost and Efficiency

Training Efficiency Our training framework is designed to be resource-efficient despite the complexity of the multi-modal architecture. By leveraging the frozen pre-trained representations from Siglip and VGGT, and employing Low-Rank Adaptation (LoRA) for the Vicuna-7B backbone, we significantly reduce the computational burden. Specifically, with a LoRA rank of $r = 16$, the number of trainable parameters is approximately **160M**, constituting less than **2%** of the total model size ($\sim$**8.4B** parameters). Training was conducted on 8 NVIDIA A100 (80GB) GPUs. The entire training process on the MindCraft-Train dataset required approximately **6 days**. We utilized mixed-precision training (bfloat16) and a global batch size of 64, with a peak VRAM usage of approximately **60GB** per GPU during training.

Inference Latency and Memory Footprint During inference, we evaluate the model on a single NVIDIA A100 GPU. The inference process involves three main stages: visual and geometric feature extraction, latent cognitive map querying, and LLM token generation.

- **Latency:** The average inference latency per step is approximately **0.85s**. Compared to the navigation-only baseline (e.g., NaVILA), our model introduces a marginal overhead of only $\sim$**0.03s**. This slight increase is negligible in practice, especially considering the significant gains in reasoning accuracy (+4.7% QA-Acc) and navigation success.
- **Memory:** The peak VRAM usage during inference is approximately **22GB**. This memory footprint is well within the capacity of high-end consumer-grade GPUs (e.g., NVIDIA RTX 3090/4090), making our model feasible for deployment on workstations. While the Episodic Memory grows with trajectory length, our frame sampling strategy ($K = 32$) ensures that memory consumption remains bounded and predictable.

Summary In conclusion, LASAR achieves a superior balance between performance and efficiency. The inclusion of the Latent Cognitive Map and geometric encoders provides substantial improvements in spatial reasoning capabilities with minimal cost to inference speed and a manageable memory footprint.

9.3. Additional Qualitative Results

Figure 6. **Navigation Failure Case.** The model fails to execute the instruction "Walk into the living room and keep walking straight past the living room..." due to a misinterpretation of the global room layout.

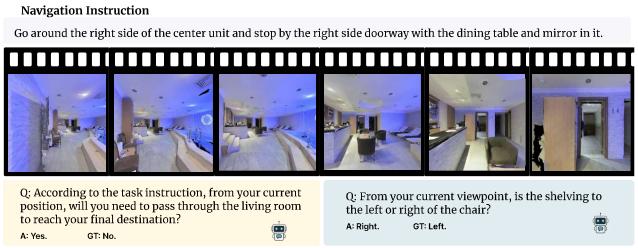

Figure 7. **Qualitative analysis of reasoning failures.** We illustrate two representative failure modes on the MindCraft-Test set. **Left (Prospective Failure):** The agent correctly identifies the scene semantics but fails to distinguish the subtle topological difference between "passing *through*" and "going *around*" an area. **Right (Introspective Failure):** The agent struggles with fine-grained geometric grounding, misclassifying the "Left/Right" relationship due to viewpoint sensitivity.

To provide a more comprehensive understanding of our model's capabilities and limitations, we present additional qualitative examples below.

Success Case. The main text has already highlighted our model's ability to follow long and complex instructions, successfully navigating through multiple rooms, while a strong baseline fails by becoming stuck in a local loop.

9.4. Additional Qualitative Results

To provide a more comprehensive understanding of our model's capabilities and limitations, we present additional qualitative examples below.

Success Case. The main text has already highlighted our model's ability to follow long and complex instructions, successfully navigating through multiple rooms, while a strong baseline fails by becoming stuck in a local loop.

Failure Case Visualization and Analysis. We further analyze the limitations of our model by categorizing failure modes into navigation execution failures and cognitive reasoning failures.

First, regarding navigation execution, our model occasionally struggles with complex multi-room layouts. As illustrated in Figure 6, we present a failure case with the instruction: "Walk into the living room and keep walking straight past the living room. Then walk into the entrance under the balcony and wait at the entrance to the other room." Despite the clear visual cues provided in the left image, the model struggles to accurately interpret the global spatial layout, leading to a failure in selecting the optimal path to the destination.

Second, regarding reasoning failures, even when navigation is successful, the model may produce incorrect answers to cognitive queries due to specific cognitive deficits. As visualized in Figure 7, we identify two representative reasoning failure modes. The first mode, Topological Ambiguity (Figure 7 Left), occurs when the model correctly identifies scene semantics but fails to parse fine-grained topological instructions. In the example shown, the instruction commands the agent to "Go around," but the model incorrectly predicts it must "pass through" the living room. This suggests a failure to distinguish the subtle linguistic distinction between intersecting a region and skirting its boundary. The second mode, Fine-grained Geometric Grounding (Figure 7 Right), reveals the model's sensitivity to egocentric viewpoint shifts. When asked about the spatial relationship between the "shelving" and the "chair," the model answers "Right" instead of "Left." This error highlights the difficulty of resolving precise depth and relative positioning in cluttered scenes where slight camera rotations can invert spatial relationships.

10. Feature Visualization

To qualitatively validate the structural impact of our ST-CRL objective, we conducted a comprehensive feature space analysis as presented in Figure 4 of the main paper. This visualization comprises two distinct components: a static manifold structure comparison (Top and Middle panels) and a dynamic trajectory evolution analysis (Bottom panel).

For the static manifold analysis, we aimed to demonstrate how ST-CRL shapes the latent space by comparing the learned representations of the baseline model (LASAR IL+QA) against our full model. We randomly sampled $N = 1,000$ distinct cognitive map states from the MindCraft-Test validation set for each model. We then performed t-SNE dimensionality reduction independently for each model using a perplexity of 30, a learning rate of 200, and PCA initialization. To interpret the resulting structures, we visualized the same set of projected points under two distinct coloring schemes. The "Query Type" coloring (Top Row) tests whether the model overfits to linguistic patterns, while the "Room Type" coloring (Middle Row) evaluates spatial grounding. As observed, the baseline exhibits an isotropic, unstructured distribution under both schemes, indicating a failure to organize memory spatially. In contrast, our model forms dense, distinct clusters when colored by Room Type while maintaining a uniform mix of Query Types within each cluster, confirming that the latent space is organized by spatial semantics rather than superficial query templates.

For the trajectory evolution dynamics shown in the bottom panel, we employed a joint embedding strategy to visualize a continuous navigation episode within the global semantic context. Instead of projecting the trajectory in isolation, we constructed a joint dataset consisting of a background reference set (S_{ref}, $N = 2,000$ global validation samples) and a foreground trajectory set (S_{traj}, the sequence of latent states from a single long-horizon episode). We concatenated these sets and performed a single t-SNE run on the combined data to ensure a unified coordinate system. In the final visualization, the background points are rendered with low opacity to delineate the semantic regions (e.g., Kitchen, Bedroom), while the foreground trajectory is connected sequentially using a temporal color gradient. This visualization confirms spatiotemporal consistency, as the agent's projected state smoothly transitions between the corresponding semantic clusters on the manifold as it physically navigates through the environment.

11. Limitations and Future Work

While our proposed LASAR framework demonstrates significant advancements in equipping embodied agents with spatio-temporal intelligence, we acknowledge several limitations that pave the way for compelling future research.

First, our model's multi-encoder architecture, while powerful, introduces a notable computational overhead in terms of both memory (VRAM) and floating-point operations (FLOPs) during inference. This currently may hinder its deployment on resource-constrained robotic platforms. Second, the performance of our **geometric inference module (VGGT)** is contingent on the quality of its **2D-to-3D pre-training** and may struggle with out-of-distribution visual cues in new environments. The model's robustness against noisy, incomplete, or drifting visual data is an important area for further investigation. Finally, our current framework operates under a static environment assumption. It is not designed to handle dynamic scenes with moving objects, interacting agents, or significant layout changes, which limits its applicability in more complex, human-centric settings.

Addressing these limitations points toward several exciting future directions. A primary focus will be on **improving model efficiency**. We plan to explore model compression techniques, such as knowledge distillation from our larger model to a more compact student network, and post-training quantization to create a "LASAR-Lite" version without substantially compromising performance. This would be a critical step towards real-world robotic deployment.

Another key avenue is to **enhance the model's robustness and generalization**. To bridge the sim-to-real gap highlighted by our data dependency, we aim to train LASAR on a much wider variety of 3D environments, incorporating simulated sensor noise and diverse visual styles. Furthermore, we will investigate advanced learning paradigms, such as meta-learning or online domain adaptation, to enable the agent to rapidly fine-tune its world model when introduced to a completely new and unseen environment.

Perhaps the most significant long-term vision is to **transcend the offline training paradigm towards interactive and lifelong learning**. This involves developing mechanisms for the agent to continuously update its semantic and episodic memories based on its own experiences and interactions within the world. By inte-

grating reinforcement learning principles, the agent could learn from trial-and-error, associate failed plans with specific environmental states, and implicitly refine its understanding of physical affordances. Such a system would move us closer to creating truly adaptive and autonomous embodied intelligences that learn and grow over their entire operational lifetime.