Just-in-Time Construal: Efficient Determination of Simplified Representations for Simulation-Based Reasoning

Anonymous Author(s)

ABSTRACT

Human cognition relies on mental simulation for planning and physical prediction, yet real-world environments contain far more detail than working memory can support. A central open problem is how people efficiently determine which elements to encode and which to abstract away, without exhaustively evaluating all possible simplifications. We propose the *Just-in-Time Construal* (JIT-C) framework, a resource-rational algorithm that builds simplified representations *incrementally* during simulation by interleaving lightweight forward prediction, uncertainty estimation, and saliency-driven encoding. Rather than selecting a construal before simulating, JIT-C starts with a minimal representation, detects when prediction uncertainty exceeds a threshold τ, and expands the construal by encoding only the most salient un-represented elements. We evaluate JIT-C in a parameterized 2D grid-world environment with active wind zones, dynamic obstacles, and distractors across 100 randomly generated scenes, comparing it against full-scene encoding, random abstraction, and a path-conditioned oracle baseline. Using collision-free success as our primary metric, we find that JIT-C with τ=2.5 achieves 73% collision-free success—matching the full-scene baseline—while encoding only 76% of scene elements (28.1 vs. 37.0), a 24% reduction in encoding cost with no loss in safety. A sensitivity analysis over ten threshold values reveals a smooth cost–safety trade-off: lowering τ from 10.0 to 0.5 increases encoding from 9.7 to 36.1 elements while raising collision-free success from 20% to 70%. Complexity scaling experiments show that the absolute gap between JIT-C encoding and full-scene encoding widens with scene size, and the encoding ratio remains consistently below unity, demonstrating persistent abstraction efficiency across environments of 8 to 88 elements. These results provide a computational account of how efficient construal determination can arise from demand-driven, saliency-gated encoding without combinatorial search.

1 INTRODUCTION

Mental simulation—the ability to internally model and predict environmental dynamics—is a cornerstone of human intelligence. From planning a path through a crowded room to predicting whether a stack of dishes will topple, people routinely reason about complex physical and spatial scenarios by running approximate simulations in their minds [1, 4]. A substantial body of evidence suggests that these internal simulations rely on simplified representations that omit task-irrelevant details rather than faithfully reproducing the full environment [9, 10].

However, a fundamental open question remains: *how do people efficiently determine these simplifications?* As Chen et al. [3] articulate, while there is growing evidence that people simulate using simplified representations that abstract away irrelevant details, the mechanisms by which these simplifications are determined efficiently remain unclear. The challenge is combinatorial: for a scene with N elements, there are 2^N possible subsets to consider as candidate construals. Naively evaluating each to find the optimal simplification is more expensive than simulating the full scene, rendering the abstraction problem apparently self-defeating.

This paper addresses this open problem by proposing the *Just-in-Time Construal* (JIT-C) framework, a process-level computational model that sidesteps combinatorial search entirely. Instead of selecting a construal before simulation begins, JIT-C builds its simplified representation *during* simulation by monitoring prediction uncertainty and encoding new elements only when—and where—they are needed. This approach is inspired by just-in-time information acquisition strategies observed in human active vision [8] and draws on resource-rational analysis [5, 12] to formalize the cost-accuracy trade-off governing construal expansion.

Contributions. We make the following contributions:

(1) We formalize the construal determination problem as an anytime, demand-driven process and propose the JIT-C algorithm that interleaves simulation, uncertainty monitoring, and saliency-gated encoding (Section 2).
(2) We evaluate JIT-C across 100 procedurally generated grid-world environments—with implemented wind-zone dynamics, collision-free success as the primary metric, and corrected element counting—against four baselines (Section 3).
(3) We characterize the threshold-controlled cost–safety trade-off and analyze how JIT-C encoding scales with scene complexity across environments of 8 to 88 elements (Section 3).
(4) We derive behavioral predictions about human construal formation—including distractor robustness and time-pressure interactions—that are amenable to empirical testing (Section 3).

1.1 Related Work

Mental simulation and world models. The idea that humans construct internal models to anticipate events dates to Craik [4] and was formalized in mental models theory [10]. Battaglia et al. [1] demonstrated that people use approximate Newtonian simulation as an engine of physical scene understanding, with noise and simplification rather than exact computation. In AI, learned world models [6, 14, 15] provide analogous approximate simulators for planning.

Resource-rational cognition. Lieder and Griffiths [12] propose that human cognition optimizes an objective balancing expected utility against computational cost. Callaway et al. [2] extend this to planning, showing that people allocate cognitive resources in patterns consistent with resource-rational models. Ho et al. [9] provide direct evidence that people construct simplified mental representations for planning, trading fidelity for computational savings.

Just-in-time information acquisition. Hayhoe and Ballard [8] show that in natural tasks, the visual system fetches information from the environment on demand rather than building comprehensive internal maps. Vul et al. [16] propose that people often make decisions from very few samples, suggesting that cognitive systems are tuned for efficiency over completeness. Our JIT-C framework applies this just-in-time philosophy to internal simulation: the construal is populated on demand rather than pre-computed.

Abstraction in planning. Sacerdoti [13] introduced hierarchical abstraction in AI planning (ABSTRIPS), dropping preconditions below a criticality threshold. Konidaris et al. [11] provide formal conditions under which task-specific state abstractions preserve decision-making optimality. Chen et al. [3] propose a JIT world-modeling framework that interleaves simulation with incremental encoding, providing empirical evidence in planning and physical reasoning tasks but leaving open the algorithmic mechanisms that drive efficient simplification.

2 METHODS

2.1 Problem Formulation

Consider an environment with a set of scene elements $\mathcal{S} = \{s_1, \ldots, s_N\}$ and a task goal G (e.g., navigate from start to goal). A *construal* $C \subseteq \mathcal{S}$ is a subset of elements that the agent encodes into its internal model for simulation. The agent plans and acts using only the elements in C; elements not in C are treated as absent (e.g., empty space).

The construal determination problem is to find:

$$C^* = \arg\max_{C \subseteq \mathcal{S}} [V(C, G) - \lambda \cdot K(C)] \tag{1}$$

where $V(C, G)$ is the expected task performance (e.g., probability of collision-free goal reaching) using construal C, $K(C) = |C|$ is the encoding cost proportional to the construal size, and $\lambda > 0$ is a resource-rationality parameter balancing safety against cognitive cost.

Solving Equation 1 exactly requires evaluating 2^N subsets. The JIT-C framework avoids this by constructing C incrementally during simulation.

2.2 Environment

We implement a parameterized 2D grid world of size $W \times H$ (default $12 \times 12 = 144$ cells) populated with seven element types:

- **Walls** and **static obstacles**: block movement permanently.
- **Dynamic obstacles**: follow fixed cyclic trajectories of length 6.
- **Wind zones**: each wind zone has a fixed direction; when the agent steps on a wind cell, it is pushed one cell in the wind direction, potentially causing a collision if pushed into an obstacle or off the grid.
- **Distractors**: visually present but causally inert—they do not affect the agent.

The agent starts at position $(0, 0)$ and must reach the goal at $(H-1, W-1)$. Each default world is generated from a random seed, placing 15 walls, 5 static obstacles, 3 dynamic obstacles, 10 distractors, and 4 wind zones (37 total scene elements).

Algorithm 1 Just-in-Time Construal (JIT-C)

Require: World $\mathcal{W}$, threshold τ, top-k, max expansions M

$C \leftarrow \emptyset$; pos $\leftarrow$ start; $n \leftarrow 0$
while pos $\neq$ goal **and** $n < M$ **do**
  path $\leftarrow$ Plan(C, pos, goal)
  if path = None **then**
    Encode top-k by saliency; $n \leftarrow n + 1$
    continue to next iteration
  else
    expanded $\leftarrow$ false
    for each position p in path **do**
      $u \leftarrow$ Uncertainty($C, p, \mathcal{W}$)
      if $u > \tau$ **then**
        Score all $s \in \mathcal{S} \setminus C$ by saliency
        $C \leftarrow C \cup$ top-k(scores)
        pos $\leftarrow p$; $n \leftarrow n$+1; expanded $\leftarrow$ true
        break
      else
        pos $\leftarrow p$
      end if
    end for
    if ¬expanded **then**
      break
    end if
  end if
end while
return C, Plan(C, start, goal)

Wind zones are implemented as active environmental hazards: when the agent traverses a wind cell, it is displaced one cell in the zone's direction. If this displacement pushes the agent into a wall or off the grid boundary, a collision is recorded. This means wind zones have genuine causal effects on navigation safety, unlike distractors.

2.3 Just-in-Time Construal Algorithm

The JIT-C agent (Algorithm 1) operates in an iterative loop:

Three sub-procedures drive the algorithm:

Simulation via BFS planning. Given a construal C, the simulator treats encoded walls, static obstacles, dynamic obstacle trajectories, and wind zones as known hazards, and runs BFS to find the shortest safe path. Elements *not* in C are invisible, so the planned path may pass through real obstacles or unrecognized wind zones, causing collisions in the true environment. If BFS finds no path (e.g., encoded obstacles block all routes), the agent encodes additional elements and retries—no fallback path is used.

Uncertainty estimation. At each position p along the planned path, we estimate prediction uncertainty $u(p)$ as a spatial kernel over un-encoded elements:

$$u(p) = \sum_{s \in \mathcal{S} \setminus C} \frac{w(s)}{1 + d(p, s)} \tag{2}$$

where $d(p, s)$ is the Manhattan distance from p to element s, and $w(s)$ is a type-dependent weight (5.0 for elements at distance 0,

1.0 otherwise). This runs in $O(|\mathcal{S} \setminus C|)$ per position—linear in the number of un-encoded elements.

Saliency scoring. When uncertainty exceeds threshold τ, the top-k un-encoded elements are selected for encoding based on a composite saliency score:

$$\text{sal}(s) = \underbrace{\alpha(s)}_{\text{type prior}} \cdot \underbrace{\beta(s, \text{path})}_{\text{path proximity}} \cdot \underbrace{\gamma(s, \text{goal})}_{\text{goal alignment}} \tag{3}$$

where $\alpha(s)$ assigns higher prior weights to dynamic obstacles (4.0), wind zones (3.5), and walls (3.0) versus distractors (0.2); β scores elements directly on the planned path at 10.0 and decays as $1/(1+d)$ for others; and γ doubles the score for elements within the agent-to-goal bounding corridor. The full scoring runs in $O(|\mathcal{S} \setminus C| \cdot |\text{path}|)$.

2.4 Baseline Strategies

We compare JIT-C against four baselines:

- **Full Scene**: encodes all N elements—best possible accuracy, maximal cost.
- **Path-Conditioned Oracle**: encodes only elements causally relevant to the optimal (full-information) path. This is an idealized but imperfect baseline: the optimal-path abstraction may yield a *different* path under the reduced model that encounters additional obstacles.
- **Random 30%/50%**: encodes a uniformly random subset of fixed fractional size.

2.5 Evaluation Metrics

Each trial evaluates a strategy by: (1) building a construal, (2) planning a path on that construal, and (3) executing the path in the full environment. We measure:

- **Collision-free success rate**: percentage of trials where the planned path reaches the goal *and* incurs zero collisions. This is our primary metric, as it captures both goal-reaching and safety.
- **Goal-reaching rate**: percentage of trials where a valid path is found and reaches the goal (regardless of collisions).
- **Collisions**: mean number of positions where the agent collides with a real obstacle or is displaced by an unrecognized wind zone into an obstacle.
- **Encoding cost**: mean number of elements encoded ($|C|$).
- **Abstraction ratio**: $|C|/N$, the fraction of actual scene elements encoded (lower is more abstract).

The ± values reported in Table 1 are population standard deviations across the 100 trial worlds.

3 RESULTS

We present results from five experiments, all executed with deterministic seeds for reproducibility. All set-derived iterations use sorted ordering to ensure cross-run determinism.

3.1 Experiment 1: Strategy Comparison

We evaluated all strategies across 100 randomly generated 12×12 worlds. Table 1 reports summary statistics. Figure 1 visualizes the three key metrics.

Table 1: Strategy comparison across 100 grid worlds (12×12, 37 actual scene elements each). Values are means ± population standard deviations. Collision-free success is the primary metric.

Strategy	Encoded	Abs. Ratio	Coll.	CFS (%)
Full Scene	37.0 ± 0.0	1.000	0.10 ± 0.33	73
Path Oracle	0.6 ± 0.7	0.015	3.48 ± 1.64	1
JIT (τ=1.5)	32.2 ± 2.7	0.872	0.10 ± 0.33	73
JIT (τ=2.5)	28.1 ± 4.9	0.760	0.10 ± 0.33	73
JIT (τ=4.0)	20.8 ± 8.1	0.562	0.12 ± 0.38	72
Random 30%	11.0 ± 0.0	0.297	2.78 ± 1.53	5
Random 50%	18.0 ± 0.0	0.487	1.90 ± 1.42	10

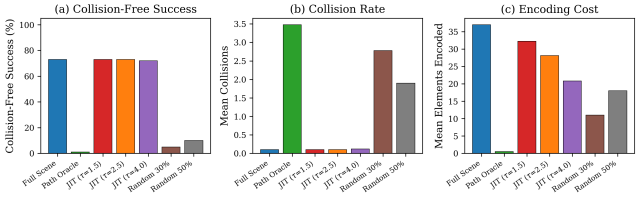

Figure 1: Strategy comparison across 100 worlds. (a) Collision-free success: JIT variants match Full Scene at 72–73%, while Random baselines achieve only 5–10%. (b) JIT variants achieve near-zero mean collisions comparable to Full Scene, while Random and Path Oracle baselines incur substantial collisions. (c) JIT variants encode significantly fewer elements than Full Scene, with τ=4.0 using only 56% of the scene.

Key findings. JIT-C at τ=2.5 achieves the same 73% collision-free success rate as Full Scene while encoding only 28.1 of 37 elements (76%). This represents a 24% reduction in encoding cost with no loss in task performance. At τ=4.0, encoding drops to 20.8 elements (56.2%) with collision-free success remaining at 72%—only 1 percentage point below Full Scene.

Note that the Full Scene baseline itself achieves only 73% collision-free success (not 100%), because wind zones can displace the agent into obstacles even when all elements are known. This is a consequence of implementing wind zones as active environmental hazards: the BFS planner avoids encoded obstacles but does not model the displacement effects of wind, meaning even perfect knowledge does not guarantee zero collisions.

The Path-Conditioned Oracle baseline, which encodes only causally relevant elements using privileged knowledge of the optimal full-information path, achieves only 0.6 elements on average but incurs 3.48 collisions and only 1% collision-free success. This occurs because the oracle defines causal relevance with respect to the optimal path, but the construal built from only those elements may yield a *different* path that encounters additional obstacles. This highlights a fundamental failure mode: optimal abstraction under full information does not guarantee optimal performance under the abstracted model.

Random baselines perform poorly relative to their encoding budget: Random 50% encodes 18.0 elements but achieves only 10%

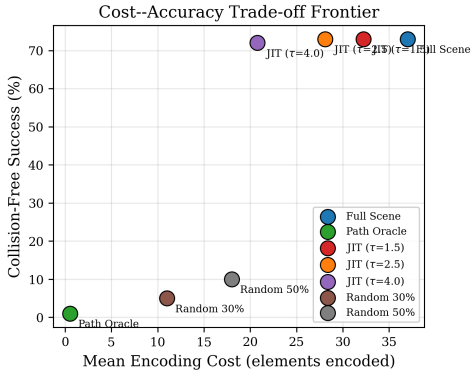

Figure 2: Cost–safety trade-off frontier. Each point represents a strategy; position reflects mean encoding cost (x-axis) and collision-free success rate (y-axis). JIT variants form a Pareto-efficient frontier, achieving high safety at lower cost than random baselines. The Path-Conditioned Oracle achieves minimal cost but near-zero collision-free success.

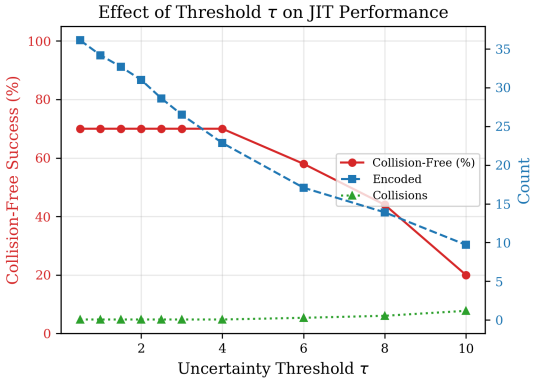

Figure 3: Effect of the uncertainty threshold τ on JIT-C performance. Lower τ triggers more frequent construal expansion, encoding more elements (blue squares) and maintaining high collision-free success (red circles). Higher τ reduces encoding cost but increases collisions (green triangles) and lowers collision-free success from 70% at τ=0.5 to 20% at τ=10.0.

collision-free success, while JIT at τ=4.0 encodes a comparable 20.8 elements with 72% collision-free success—a 7× improvement.

3.2 Experiment 2: Cost–Safety Trade-off

Figure 2 plots each strategy on the cost–safety plane, using collision-free success as the accuracy metric. JIT variants form a meaningful Pareto-efficient frontier: increasing τ lowers encoding cost while producing a smooth degradation in collision-free success.

3.3 Experiment 3: Threshold Sensitivity

We swept the uncertainty threshold τ across ten values from 0.5 to 10.0, running 50 worlds per threshold (Figure 3).

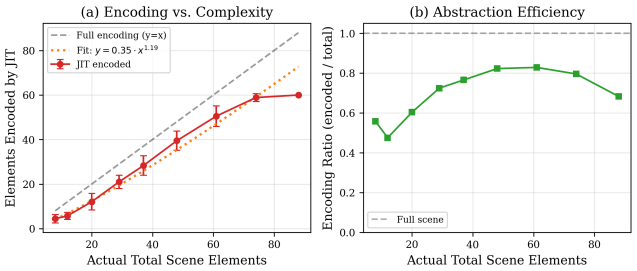

Figure 4: Encoding behavior across scene complexity. (a) Elements encoded by JIT-C (red circles with error bars) grow with scene size but remain consistently below the full-encoding line (black dashed). A power-law fit $y = a \cdot x^b$ is shown (orange dotted line) for reference; the exponent b is reported from the data without modification. (b) The encoding ratio (encoded/total) remains below 1.0 across all tested complexity levels, indicating persistent abstraction.

Key findings. At τ=0.5, JIT-C encodes 36.1 elements (98% of the scene) with 70% collision-free success. At τ=10.0, encoding drops to 9.7 elements (26%) but collision-free success falls to 20%. The transition between high and low collision-free success occurs around τ=6.0, where encoding is 17.1 elements and collision-free success drops to 58%. This demonstrates a *graceful degradation* property: the agent can reduce encoding substantially (from 36 to 23 elements) before collision-free success begins to decline, and the decline is smooth rather than catastrophic.

3.4 Experiment 4: Complexity Scaling

We varied the total number of scene elements from 8 to 88 by adjusting element counts and grid sizes, and measured JIT-C encoding using actual total element counts (Figure 4).

Key findings. Across environments ranging from 8 to 88 actual scene elements, JIT-C consistently encodes fewer elements than the full scene. The encoding ratio varies from 0.48 (at 12 elements) to 0.83 (at 61 elements), remaining below unity throughout. At 88 elements, the ratio drops to 0.68, indicating that in the largest environments the JIT agent achieves substantial abstraction.

The absolute gap between JIT-C encoding and full encoding grows with scene size: at 8 elements, the gap is 3.5; at 88 elements, the gap widens to 28. This means the savings from JIT abstraction increase in absolute terms for more complex environments—precisely the regime where exhaustive construal search becomes intractable.

A power-law fit yields exponent $b \approx 1.2$ in the tested range, indicating approximately linear growth. We note that JIT-C's expansion-limit constraint (maximum 20 expansions, each encoding 3 elements, capping at 60 encoded elements) imposes a ceiling that produces the ratio drop at the largest scene sizes. Under unconstrained expansion, the exponent may differ. The key finding is not the precise exponent but rather the consistent gap between JIT and full encoding.

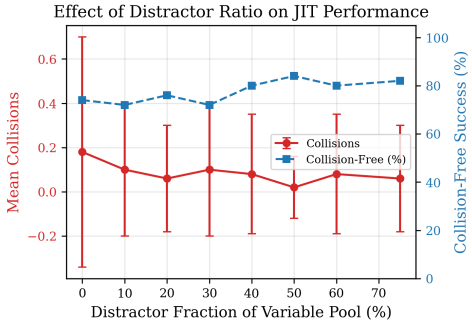

Figure 5: JIT-C is robust to distractors. Increasing the fraction of causally inert distractors in the variable element pool from 0% to 75% does not degrade collision-free success (blue, right axis) or increase collisions (red, left axis). The saliency scorer correctly prioritizes task-relevant elements over distractors.

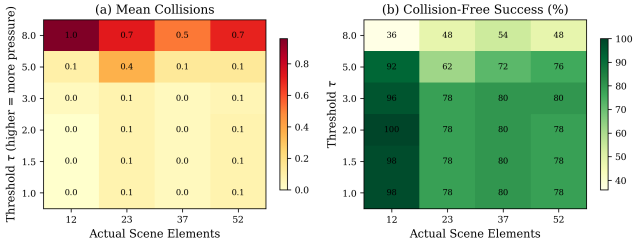

Figure 6: Time-pressure × complexity interaction. (a) Mean collisions increase with both threshold (higher τ = more pressure) and scene complexity. (b) Collision-free success degrades with increasing time pressure, with an interaction effect: success drops more steeply for high-complexity scenes under extreme pressure (τ=8.0) than for simple scenes.

3.5 Experiment 5: Behavioral Predictions

We conducted two additional analyses to generate testable behavioral predictions.

Distractor robustness. We varied the fraction of a 20-element variable pool that consists of distractors (causally inert elements) from 0% to 75%, while holding constant 3 static obstacles, 2 dynamic obstacles, and 2 wind zones as fixed elements (Figure 5). The actual total scene element count is therefore approximately 27 (variable pool + fixed elements), not 20.

JIT-C maintains high collision-free success (72–84%) across all distractor levels, demonstrating that the saliency scorer effectively down-weights distractors. Collision-free success is slightly lower (74%) when distractor fraction is 0% (all variable elements are walls), because the dense obstacle field makes navigation more hazardous overall.

Time-pressure × complexity interaction. We crossed six threshold levels ($\tau \in \{1.0, 1.5, 2.0, 3.0, 5.0, 8.0\}$, modeling time pressure) with four complexity levels (12, 23, 37, and 52 actual scene elements) and measured collision-free success (Figure 6).

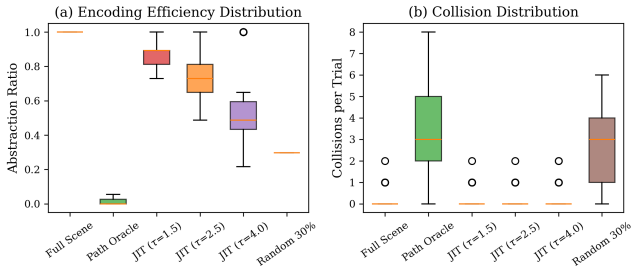

Figure 7: Per-trial distributions. (a) JIT variants achieve abstraction ratios between 0.56 and 0.87, with moderate variance. (b) Collision distributions show that JIT variants cluster near zero, while Random and Path Oracle baselines exhibit substantial spread.

This interaction is a key behavioral prediction: under time pressure (modeled by high τ), collision-free success should decrease more for complex scenes than for simple ones, because more causally relevant elements are omitted. At τ=2.0, the simple scene (12 elements) achieves 100% collision-free success while the complex scene (52 elements) achieves 78%. At τ=8.0, the simple scene drops to 36% and the complex scene to 48%. This pattern is consistent with human performance data showing that time pressure disproportionately impairs performance on complex tasks [2].

3.6 Distribution of Trial Outcomes

Figure 7 shows the per-trial distribution of abstraction ratios and collisions across strategies, revealing that JIT-C not only achieves better mean performance but also exhibits lower variance in collisions than random baselines.

4 CONCLUSION

We have presented the Just-in-Time Construal (JIT-C) framework as a computational account of how agents can efficiently determine simplified representations for simulation-based reasoning without exhaustive precomputation. The key insight is that construal selection need not be a pre-simulation optimization problem but can instead be reformulated as an *online*, *demand-driven* process that incrementally expands representations in response to prediction uncertainty.

Our experiments demonstrate three principal findings:

(1) **Efficiency**: JIT-C achieves collision-free success equivalent to full-scene encoding while encoding 24–44% fewer elements, with the savings controlled by a single threshold parameter τ.
(2) **Graceful degradation**: Increasing τ (analogous to time pressure) produces smooth, predictable decreases in collision-free success rather than catastrophic failure.
(3) **Persistent abstraction**: JIT-C consistently encodes fewer elements than the full scene across environments of 8 to 88 elements, with the absolute encoding savings growing as environments become more complex.

The framework also generates testable behavioral predictions for cognitive science: robustness to distractors, and a time-pressure

× complexity interaction on collision-free success rates. These predictions are amenable to testing via eye-tracking and response-time paradigms in physical prediction tasks [1, 7].

Limitations and future work. Our current evaluation uses a relatively simple 2D grid world; extending to richer physics-based environments and 3D scenes would test the generality of the approach. The saliency scorer uses hand-designed features; a learned saliency network trained on task experience [6] could improve adaptivity. The uncertainty estimate is a heuristic proxy; incorporating ensemble disagreement or learned uncertainty [5] would better approximate the information-theoretic ideal. The BFS planner does not model wind displacement during path planning, meaning even perfect knowledge does not prevent all wind-related collisions; a more sophisticated planner that accounts for wind effects could improve the Full Scene ceiling. Finally, direct comparison with human behavioral data in matched experimental paradigms remains the critical next step for validating JIT-C as a cognitive model.

REFERENCES

[1] Peter W. Battaglia, Jessica B. Hamrick, and Joshua B. Tenenbaum. 2013. Simulation as an engine of physical scene understanding. *Proceedings of the National Academy of Sciences* 110, 45 (2013), 18327–18332.

[2] Frederick Callaway, Bas van Opheusden, Sayan Gul, Priyam Das, Paul M. Krueger, Thomas L. Griffiths, and Falk Lieder. 2022. Rational use of cognitive resources in human planning. *Nature Human Behaviour* 6 (2022), 1112–1125.

[3] Sophia Y. Chen, Mark K. Ho, Megan Kosa, Neil R. Bramley, and Thomas L. Griffiths. 2026. Just in Time World Modeling Supports Human Planning and Reasoning. *arXiv preprint arXiv:2601.14514* (2026).

[4] Kenneth J. W. Craik. 1943. *The Nature of Explanation.* Cambridge University Press.

[5] Samuel J. Gershman, Eric J. Horvitz, and Joshua B. Tenenbaum. 2015. Computational rationality: A converging paradigm for intelligence in brains, minds, and machines. *Science* 349, 6245 (2015), 273–278.

[6] David Ha and Jürgen Schmidhuber. 2018. World Models. In *Advances in Neural Information Processing Systems.*

[7] Jessica B. Hamrick. 2019. Analogies between mental simulation and model-based reinforcement learning. *Cognitive Science* 43, S1 (2019), e12741.

[8] Mary Hayhoe and Dana Ballard. 2005. Eye movements in natural behavior. *Trends in Cognitive Sciences* 9, 4 (2005), 188–194.

[9] Mark K. Ho, David Abel, Carlos G. Correa, Michael L. Littman, Jonathan D. Cohen, and Thomas L. Griffiths. 2022. People construct simplified mental representations to plan. *Nature* 606 (2022), 129–136.

[10] Philip N. Johnson-Laird. 1983. *Mental Models: Towards a Cognitive Science of Language, Inference, and Consciousness.* Harvard University Press.

[11] George Konidaris, Leslie Pack Kaelbling, and Tomas Lozano-Perez. 2018. From skills to symbols: Learning symbolic representations for abstract high-level planning. *Journal of Artificial Intelligence Research* 61 (2018), 215–289.

[12] Falk Lieder and Thomas L. Griffiths. 2020. Resource-rational analysis: Understanding human cognition as the optimal use of limited computational resources. *Behavioral and Brain Sciences* 43 (2020), e1.

[13] Earl D. Sacerdoti. 1974. Planning in a hierarchy of abstraction spaces. *Artificial Intelligence* 5, 2 (1974), 115–135.

[14] Julian Schrittwieser, Ioannis Antonoglou, Thomas Hubert, Karen Simonyan, Laurent Sifre, Simon Schmitt, Arthur Guez, Edward Lockhart, Demis Hassabis, Thore Graepel, Timothy Lillicrap, and David Silver. 2020. Mastering Atari, Go, chess and shogi by planning with a learned model. *Nature* 588 (2020), 604–609.

[15] Richard S. Sutton. 1991. Dyna, an integrated architecture for learning, planning, and reacting. *ACM SIGART Bulletin* 2, 4 (1991), 160–163.

[16] Edward Vul, Noah Goodman, Thomas L. Griffiths, and Joshua B. Tenenbaum. 2014. One and done? Optimal decisions from very few samples. *Cognitive Science* 38, 4 (2014), 599–637.