Beyond Behavioral Sequences: Leveraging Short-Term Memory to Optimize Decision Policies in Anticipatory Learning Classifier System

Mateusz Łabędzki and Olgierd Unold
Wroclaw University of Science and Technology, Wrocław, Poland
{mateusz.labedzki, olgierd.unold}@pwr.edu.pl

***Abstract*—The Behavioral Enhanced Anticipatory Classifier System (BEACS) has demonstrated robust capabilities in developing compact and interpretable representations of non-deterministic environments. However, recent evaluations have highlighted a critical limitation: the system often fails to achieve optimal decision policies, resulting in suboptimal average path lengths to the goal in complex aliased environments. This work introduces an augmented BEACS framework designed to address this deficiency by incorporating a Short-Term Memory (STM) mechanism. Unlike standard Anticipatory Learning Classifier System (ALCS) models that rely solely on immediate sensory input, the proposed memory component enables the agent to maintain an internal representation of recent state-action history, effectively resolving ambiguities in states where optimal behavior depends on temporal dependencies. Furthermore, we extend the classifier enhancement process to evolve memory-dependent rules, allowing the system to differentiate between perceptually identical states based on their historical context. Experimental results across a comprehensive suite of benchmark mazes demonstrate that the integration of Short-Term Memory significantly reduces the average path length to the goal while preserving the interpretability and generalization performance characteristic of the original BEACS framework. These findings suggest that incorporating temporal context is a vital strategy for improving path efficiency and achieving optimal performance in partially observable environments within the ALCS paradigm.**

***Index Terms*—Anticipatory Learning Classifier Systems, Machine Learning, Perceptual Aliasing Issue**

I. Introduction

Learning Classifier Systems (LCSs) [1] offer a compelling framework for evolving rule-based representations of environments. Within this family, the Anticipatory Learning Classifier System (ALCS) [2] occupies a unique position due to its fundamental capacity to build internal cognitive maps through sensory anticipation. Based on the principle of Anticipatory Behavioral Control (ABC) [3], ALCS models do not merely map state-action pairs to scalar rewards; they learn to predict the specific sensory consequences of their actions. While recent advancements like PEPACS [4], BACS [5], and the Behavioral Enhanced Anticipatory Classifier System (BEACS) [6] have demonstrated a sophisticated ability to handle non-determinism and develop compact, interpretable models, a critical performance gap remains. In complex aliased environments, BEACS often fails to achieve optimal decision policies because it lacks the temporal context needed to break ties between perceptually identical states. We address this limitation by moving "beyond behavioral sequences" to introduce an augmented framework integrated with a Short-Term Memory (STM) mechanism. By transcending the standard Markovian reliance on immediate sensory input and evolving memory-dependent rules, our system differentiates identical perceptions based on historical context, leveraging the unique representative strengths of the ALCS lineage to restore policy efficiency.

Our primary contribution is the formal integration of an ordered-list Short-Term Memory (STM) within the BEACS framework. Specifically, we: (1) introduce a context-augmentation mechanism that resolves perceptual aliasing by expanding the classifier condition from a single perception p_t to a temporal sequence $(p_t, p_{t-1}, \ldots, p_{t-n})$, (2) implement a sequence-length penalty in the reward function to prevent the "over-generalization" typical of memory-based LCS, and (3) provide a comprehensive evaluation across a suite of benchmark aliased mazes, demonstrating that this integration significantly reduces path length and restores policy optimality without sacrificing the model's characteristic interpretability.

The remainder of this paper is structured as follows: Section II describes related works, whereas Section III reviews the mechanisms of aliasing and the evolution of the ALCS lineage; Section IV details the architecture of the memory-augmented BEACS; Section V presents our experimental results and a comparison of path efficiencies; and Section VI concludes with a discussion on the implications for partially observable environments.

II. Related Work

Learning Classifier Systems have a long history of addressing perceptual aliasing, also known as the non-Markovian property, which occurs when an agent's sensory input is insufficient to distinguish between unique environmental states. Research in this domain can be categorized into two primary clusters: those extending the

accuracy-based XCS framework and those building upon the Anticipatory (ACS) lineage.

A. XCS-based Approaches: Internal Memory and Code Paths

The most established solution within XCS-based systems is the integration of internal memory registers, allowing the agent to utilize past experiences to disambiguate the current perceptual state.

Lanzi introduced XCS with internal memory (XCSM), which successfully solved simple non-Markovian tasks [7], [8]. Zang et al. further improved this approach by introducing a numbered memory list to enhance XCS performance in non-Markovian settings [9]. However, these methods often face scalability issues, as the search space expands exponentially with memory depth, highlighting the need for more generalized pattern extraction.

To avoid explicit memory dependency, Siddique et al. introduced *FoRsXCS* [10]. This system utilizes *Code Paths* (CPs)—chains of state-action-state links—to form policies. CPs serve as primitive features that capture environmental structure, enabling the agent to create unique "versioned states." This approach was further refined in *Hierarchical FoRsXCS* (*Hi-FoRsXCS*), which manages sequentially aliased states by organizing these structures into a hierarchy [11]. Recently, Uwano and Browne proposed FoRsXCS based on Aliasing-Group (AGFX) [12] and AGFX2 [13], which unify aliased states into "aliasing groups" based on regular environmental patterns. These approaches enable the identification of sequential aliasing patterns and the disambiguation of specific states through temporal observation. By clustering these perceptions, AGFX and AGFX2 effectively disentangle complex state transitions within non-Markovian environments.

A notable parallel to our Short-Term Memory (STM) approach is the Corporate XCS proposed by Tomlinson and Bull [14], which utilizes rule linkage to form chains of classifiers. While rule linkage creates implicit temporal dependencies through the population, our proposal uses explicit temporal windowing. This allows for a direct "look-back" mechanism that resolves ambiguities without the overhead of maintaining complex internal registers.

B. ACS-based Approaches. Anticipatory Maps and Behavioral Sequences

Anticipatory Learning Classifier Systems address aliasing by focusing on the predicted sensory consequences of actions, effectively building an internal cognitive map.

To address nondeterminism, ACS-PEP introduced a classifier structure in which the effect component forecasts multiple potential outcomes paired with probabilities (the so-called probability-enhanced effect) [15]. This approach was later integrated into the ACS2 framework as PEPACS [4], which uses attribute specialization to achieve complete environmental knowledge while mitigating the risks of overgeneralization.

Orhand et al. introduced two complementary extensions to ACS2: BACS, which utilizes behavioral sequences to bypass aliased states and prioritize effective decision policies [5], and BEACS, which synthesizes the strengths of its predecessors [6]. BEACS replaces traditional probabilistic effects with "Enhanced Predictions through Experience" (EPE) and integrates behavioral sequences with Double Q-learning. This hybrid approach enables the model to achieve near-complete environmental knowledge while maintaining superior decision-making performance, even in high-noise scenarios.

While BEACS significantly mitigates perceptual aliasing, it remains limited by suboptimal decision policies and model aliasing due to over-generalization, highlighting the necessity for further refinement [16].

C. Pittsburgh-based Approach

Early work by Enée and Peroumalnaïk extended Pittsburgh-based LCS, which utilized a novel covering mechanism to outperform memory-based systems like XCSM and XCSMH [17]. This lineage focuses on making predictions reliable even in the presence of nondeterminism.

Despite the success of these methods, a critical gap remains: current ALCS models often fail to achieve *optimal* path lengths in highly complex environments. This necessitates the incorporation of Short-Term Memory mechanisms to resolve temporal dependencies that behavioral sequences alone cannot address.

III. Preliminaries

A. Perceptual Aliasing Issue

In Reinforcement Learning (RL), agents acquire decision-making capabilities through dynamic interactions with their environments, aiming to maximize cumulative rewards. However, the inherent nondeterminism of these environments—arising from sensor noise, irrelevant attributes, or insufficient information—presents significant challenges. A primary example of such nondeterminism is *perceptual aliasing*. This phenomenon occurs when an agent is unable to distinguish between distinct environmental states based solely on current sensory inputs [18]. These indistinguishable states, termed *aliased states*, lead to uncertain decision-making and suboptimal performance, especially when different actions are required at each location.

The maze environment is a widely utilized benchmark for testing Learning Classifier Systems due to its discrete, scalable, and easily visualized structure. In these environments, an agent's vision is typically restricted to the cells immediately adjacent to its current position. This perception is represented as an 8-tuple, corresponding to the eight neighboring cells listed clockwise starting from the North. This sensory limitation gives rise to the Perceptual Aliasing Issue (PAI). As shown in Figure 1,

the blue-bordered states are perceptually identical despite being in different locations.

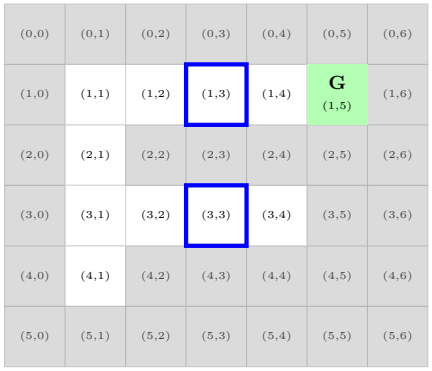

Fig. 1: Perceptual aliasing in the MazeF4 environment. Coordinate labels (x, y) map to (row, column), G denotes goal. The blue-bordered cells at $(1, 3)$ and $(3, 3)$ indicate the states of interest where perceptual aliasing occurs because their immediate 8-neighbor sensors return identical values (Wall, Wall, Space, Wall, Wall, Wall, Space, Wall), making them indistinguishable to a reactive agent.

The complexity of a maze can be quantified by the count and type of aliased states. Following the taxonomy provided by [19], we identify four types of aliasing based on the target (food) position: (1) Pseudo-Aliasing: Distance and direction to the target are identical. (2) Type I: The direction to the target is consistent, but distances vary. (3) Type II: Distances are equivalent, but directions differ. (4) Type III: Both distance and direction to the target differ.

Beyond environmental factors, *model aliasing* can emerge from the learning process itself. This occurs when generalization mechanisms conflate distinct states that require divergent actions. Although generalization is often beneficial for compact representations, a lack of effective specialization can result in an agent being correct only half the time in these conflated states.

B. Anticipatory Learning Classifier Systems

Learning Classifier Systems, introduced by Holland [1], are a class of rule-based evolutionary algorithms that evolve a population of classifiers. Each classifier typically follows an "if-then" structure: a condition (when to act), an action (what to do), and a set of parameters quantifying its quality and expected reward. A primary advantage of the LCS paradigm is its interpretability; the evolved rules are human-readable, allowing for transparent decision-making in complex environments. Classifiers are generated and refined via two main mechanisms:

- Covering: Creates new generalized classifiers when the system encounters a state not represented in the current population.
- Genetic Algorithms (GA): Evolves the population by selecting high-fitness classifiers for crossover and mutation, promoting the discovery of superior rules.

The Anticipatory Learning Classifier System extends this framework by incorporating the theory of Anticipatory Behavioral Control (ABC) [3]. Unlike standard RL agents that map states to rewards, ALCS agents learn to anticipate the sensory consequences of their actions. The ACS2 algorithm [2] is the standard implementation of this theory. It augments classifiers with an effect component, representing the predicted state change. A classifier in ACS2 follows the triplet: *Condition-Action-Effect* $(C - A - E)$. During the Anticipatory Learning Process (ALP), the system evaluates these predictions against the actual environmental outcome, leading to three scenarios: (1) Expected Case: The prediction is correct; classifier quality is increased. (2) Useless Case: No sensory change occurs; quality is decreased. (3) Unexpected Case: The prediction is incorrect; quality is decreased, and a new, specialized classifier is generated to match the actual outcome. In ACS2, both the ALP and reinforcement learning are applied to the action set—the group of classifiers matching the current state and advocating for the same action. A genetic algorithm runs concurrently to encourage generalization, counteracting the specialization pressure of the ALP. While other variants exist, such as YACS [20] and MACS [21], this study focuses on the ACS2 lineage and its behavioral enhancements in aliased environments.

C. ALCSs Addressing Perceptual Aliasing

Although the standard ACS2 framework excels in deterministic environments, it struggles with the nondeterminism inherent in aliased mazes. During the last two decades, several architectural enhancements have been proposed to resolve these ambiguities.

1) ACS-PEP and PEPACS: The ACS-PEP model [15] introduced probability-enhanced effects to handle sensory noise and action nondeterminism. Instead of a discrete prediction, the classifier's effect part maps a condition-action pair to a distribution of potential sensory outcomes. PEPACS [4] successfully ported this mechanism to the ACS2 architecture. While PEPACS achieves "full knowledge"—an accurate internal model of every environmental transition—it does not inherently optimize the decision policy. Consequently, the agent may understand the environment without knowing the most efficient route to a goal.

2) BACS: Behavioral Sequences: In contrast to the predictive focus of PEPACS, the Behavioral Anticipatory Classifier System (BACS) [5] targets policy optimization by modifying the action component. BACS allows classifiers to evolve *behavioral sequences*—macro-actions consisting of multiple steps. This allows the agent to treat a series of aliased states as a single traversable corridor, effectively "jumping" from one unique state to another. While this improves navigation, BACS often fails to build a complete model of the environment because it intentionally bypasses the details of aliased regions.

3) BEACS: The Hybrid Benchmark: The Behavioral Enhanced Anticipatory Classifier System (BEACS) [6] was designed to bridge the gap between PEPACS and BACS. It introduces several key innovations:

- Enhanced Predictions through Experience (EPE): Instead of static probabilities, BEACS records the frequency of encountered perceptions, allowing for dynamic probability estimation without explicit storage.
- Targeted Macro-actions: Behavioral sequences are generated specifically when the system detects a Perceptual Aliasing Issue (PAI), linking representation to action.
- Double Q-learning: To mitigate the overestimation bias common in Reinforcement Learning, BEACS utilizes a Double Q-learning update rule for its reward values.

Recently, an updated version of BEACS was introduced to solve complex control tasks [22]. Debiasing BEACS allows the system to value actions that maintain a state, which is essential for stability-based tasks. By utilizing discretization and non-determinism handling, d-BEACS can now successfully solve classic reinforcement learning problems, specifically the Cart-Pole problem, for the first time in ALCS history. The system is now coupled with an ontology that acts as a "semantic safeguard". This allows expert-encoded knowledge to supervise inductive learning.

Despite these advancements, recent evaluations reveal a critical ceiling for BEACS: in environments with complex temporal dependencies, the system's reliance on immediate sensory input leads to *suboptimal decision policies*. While BEACS can represent the environment, it cannot always distinguish the shortest path when the optimal choice depends on the agent's previous locations. This limitation provides the primary motivation for the Short-Term Memory augmentation proposed in this work.

IV. Memory-Augmented BEACS: Integrating Short-Term Memory for Temporal Disambiguation

STM-BEACS extends BEACS by introducing an explicit memory mechanism into the classifier representation. In standard BEACS, classifier conditions match only the current environmental perception, resulting in purely reactive decision-making that cannot directly exploit temporal dependencies.

To overcome this limitation, classifier conditions are augmented to include contextual information from previous perceptions. Each classifier maintains an ordered list of conditions, where the first corresponds to the current state and subsequent elements represent progressively earlier perceptions. A classifier therefore matches a short history of interaction rather than a single state. This enables the system to distinguish between perceptually identical states that differ in recent history, improving performance in partially observable environments or in domains with delayed effects. The resulting architecture preserves the anticipatory and learning principles of ACS2 while supporting temporally extended state representations.

Context extension is triggered by the detection of aliasing states. A classifier is identified as matching an aliased state when it has previously anticipated incorrectly for a given situation but subsequently anticipates correctly under the same current perception. This inconsistency indicates that the present perception is insufficient to uniquely characterize the underlying state.

When aliasing is detected, two classifiers are generated: a classifier with extended context and a behavioral classifier with an extended action component (see Algorithm 1). Context extension is performed by appending the immediately preceding environmental perception to the classifier's existing context list, increasing its temporal depth by one. The added perception is selected from the current interaction history, ensuring consistency with the experienced perception sequence.

Behavioral classifier construction follows the principles of BACS and BEACS but is generalized. Instead of deriving the behavioral classifier solely from the immediately preceding classifier in the action chain, all classifiers in the current action set with unique contextual representations are considered as candidates. This allows behavioral generalization to be informed by multiple temporally distinct experiences that lead to the same action outcome, improving robustness in environments with richer temporal structure.

Extended effects are created using contextual information and are triggered in two cases. First, in an aliased state, if reliable and experienced contextual classifiers are present in the action set, they are grouped by contextual representation. The best classifier from each group is selected, and if all selected classifiers propose the same action, a new classifier with an extended effect is created by combining their anticipated effects (see Algorithm 2). This process is applied separately for each behavioral sequence length to preserve temporal alignment.

Algorithm 1 Creation of behavioral classifiers and classifiers with extended context

```
C ← list of created classifiers
A ← previous action set
if classifier's mark matches the current situation then
    if classifier's effect is not enhanced then
        C ← create classifier with extended context
    end if
    candidates ← unmarked contextual classifiers
    from population that expect change sorted by
    quality
    contexts ← unique contexts from candidates
    for context in contexts do
        C ← create behavioral classifier using context
    end for
end if
```

Second, when experienced non-contextual classifiers are present but unreliable, and contextual classifiers anticipate multiple competing effects, the system selects the classifier with the highest obtained reward and creates a new classifier from it. The effect of this classifier is then extended by combining the anticipated effects observed among the contextual classifiers, yielding a unified representation of context-dependent outcomes (Algorithm 3).

Algorithm 2 Creation of extended classifiers in the first case

$P \leftarrow$ reliable, experienced and unmarked classifiers matching current state
$C \leftarrow$ dictionary with unique contexts as keys and lists of classifiers with the given context as values based on P
if size(C) < 2 **then**
  return
end if
for *sequenceLength* **in** $1..\omega$ **do**
  $SC \leftarrow$ contexts from C with length of *sequenceLength*
  bestAction $\leftarrow$ action of the classifier from SC with highest reward
  for $classifier$ **in** SC **do**
    if $classifier.action == bestAction$ **then**
      $UQ \leftarrow$ all unique effects from SC specialized with previous state
      $BQ \leftarrow$ classifier with highest reward from SC
      $generatedClassifiers \leftarrow$ enhanced classifier based on BQ with effect composed of non-overlapping effects from UQ
    end if
  end for
end for
return $generatedClassifiers$

In both cases, extended effects are introduced only when a single simple effect model is insufficient, ensuring that effect complexity increases only when supported by evidence.

The reinforcement learning mechanism is modified by introducing a penalty proportional to behavioral sequence length, replacing the double Q-learning update used in BEACS. The discounted reward decreases as sequence length increases, biasing learning toward shorter and more efficient action–perception sequences. This discourages looping behavior and mitigates overfitting when contextual conditions are used.

Action selection is additionally biased toward classifiers whose anticipations predict a change in the environmental state. Classifiers anticipating state transitions are prioritized over those predicting no change, encouraging exploratory and behaviorally meaningful actions that generate informative perception dynamics and support more effective anticipatory learning.

Algorithm 3 Creation of extended classifiers in the second case

$C \leftarrow$ experienced classifiers from the population that match previous situation
if empty(C) **then**
  return
end if
$BQ \leftarrow$ classifier with highest reward from C
for *sequenceLength* **in** $1..\omega$ **do**
  $SC \leftarrow$ contexts with length of *sequenceLength*
  $CCE \leftarrow$ whether a reliable classifier without context exists in C
  $UE \leftarrow$ unique specialized effects from C
  if $size(UE) > 1$ & & $CCE == true$ **then**
    $generatedClassifiers \leftarrow$ enhanced classifier based on BQ with effect composed of non-overlapping effects from UE
  end if
end for
return $generatedClassifiers$

V. Experiments

A. Protocol

The evaluation is conducted in two primary stages. First, a comparative analysis is performed on the original BEACS framework and its Short-Term-Memory-augmented version. Second, the STM-BEACS algorithm is isolated for a more granular investigation of its memory-augmented capabilities.

All experiments are averaged over 10 independent runs to ensure statistical robustness. The training schedule for each iteration consists of a 10,000-trial exploration phase, followed by a 1500-trial exploitation phase. The experimental parameters are standardized as follows. The exploration probability ϵ is set to 0.8 during the initial phase, reducing to 0.2 in the first exploitation phase, and 0.0 in the final two phases to measure pure greedy performance. A learning rate of $\beta = 0.05$ and a discount factor of $\gamma = 0.95$ are applied. The probability of crossover is 0.8 with a mutation rate of 0.3. For BEACS and STM-BEACS, the maximum length of behavioral sequences is capped at 5. For STM-BEACS, the maximum size of the memory component is set to 5.

The performance of the augmented BEACS along with the vanilla version was assessed using the average number of steps required to reach the goal during the exploitation phases.

B. Environments: Maze Benchmarks

The models were evaluated across 23 maze environments, ranging from simple deterministic grids to complex "Woods" environments. Following the methodology of [19], we use Optimal Average Steps (OptAvgSteps) and Maze

Complexity to quantify difficulty. Complexity represents the ratio of time required by a Q-learning agent to reach the goal relative to the average distance to that goal [16].

TABLE I: Summary of the 23 maze environments, sorted by ascending Complexity.

Maze	Aliasing Type	OptAvgSteps	Complexity
MazeF3 [23]	Not aliased	3.38	0.99
Woods1 [15]	Not aliased	1.62	1.00
MazeF1 [23]	Not aliased	1.80	1.00
Maze4 [15]	Not aliased	3.50	1.00
MiyazakiA [24]	I	3.03	1.01
MazeA [25]	Not aliased	4.22	1.01
MazeF2 [23]	Not aliased	2.50	1.02
MazeD [25]	I	2.70	1.02
Cassandra4x4 [19]	I	2.26	1.03
MazeB [25]	I	3.64	1.04
Littman57 [19]	I	3.71	1.07
Littman89 [26]	I	3.50	1.36
MiyazakiB [24]	II	3.29	1.58
Woods102 [9]	III	2.76	138.47
Maze10 [9]	III	4.55	156.21
MazeF4 [23]	II	4.30	176.82
Woods101 [19]	III	2.70	181.92
MazeE1 [27]	III	2.45	184.22
Maze7 [9]	II	4.10	187.80
Woods101.5 [9]	III	2.80	250.26
Woods100 [27]	III	2.00	265.32
MazeE2 [19]	III	2.25	271.48
MazeE3 [6]	III	2.95	278.13

TABLE II: Mean path length and Nemenyi ranks (lower is better) for the STM-BEACS (STM) and the baseline BEACS across all mazes. The STM version significantly outperforms BEACS (Nemenyi test, $\alpha = 0.05$, CD=0.61). Mazes are sorted by ascending Complexity.

Maze	STM	BEACS	Rank STM	Rank BEACS
MazeF3	3.38	5.27	1	2
Woods1	1.80	1.84	1	2
MazeF1	1.80	2.16	1	2
Maze4	3.50	4.30	1	2
MiyazakiA	3.44	5.29	1	2
MazeA	4.22	5.69	1	2
MazeF2	2.50	3.39	1	2
MazeD	2.70	3.51	1	2
Cassandra4x4	3.85	4.38	1	2
MazeB	3.52	5.48	1	2
Littman57	4.76	6.63	1	2
Littman89	3.77	4.71	1	2
MiyazakiB	4.13	5.92	1	2
Woods102	3.30	5.45	1	2
Maze10	6.84	11.33	1	2
MazeF4	4.50	6.21	1	2
Woods101	2.90	3.48	1	2
MazeE1	3.44	5.62	1	2
Maze7	4.33	5.59	1	2
Woods101.5	3.10	3.79	1	2
Woods100	2.33	2.71	1	2
MazeE2	4.64	7.68	1	2
MazeE3	45.44	38.77	2	1
Avg. rank	**-**	**-**	**1.04**	**1.96**

C. Experiments

1) Average steps to goal: We evaluate the performance of STM-BEACS against the baseline BEACS using the average steps to goal metric across a set of maze environments with varying degrees of perceptual aliasing.

To assess whether the observed performance differences between the proposed STM-BEACS and the baseline BEACS are statistically significant across the maze benchmark suite, we applied a non-parametric Friedman test followed by the Nemenyi post-hoc rank test, using the average path length as the performance measure (lower is better). The Friedman test rejected the null hypothesis of equal performance, and the subsequent Nemenyi test revealed that the STM-BEACS model obtains a significantly better average rank than the baseline BEACS at the $\alpha = 0.05$ significance level (Critical Difference CD = 0.61; observed difference = 0.91).

Table II reports the mean path length per maze as well as the corresponding per-maze ranks (1 for the better, 2 for the worse method). The STM-BEACS achieves an average rank of 1.04 compared to 1.96 for BEACS, confirming a consistent advantage over the entire collection of aliased maze environments.

In environments devoid of perceptual aliasing, STM-BEACS converges to the optimal path, matching the theoretical minimum number of steps required to reach the goal. In aliased scenarios, it reaches the minimum steps possible given the agent's specific perceptual constraints. In these instances, suboptimality is not a failure of the learning mechanism itself, but rather a consequence of the inherent limitations of incomplete state observability. Under such constraints, STM-BEACS exhibits behavior that is optimal relative to the information available to the agent. A clear example is the F4 maze, where states $(1, 3)$ and $(3, 3)$ are aliased (see Fig. 1). If the agent begins at either position, it cannot precisely disambiguate its location without first executing an action. Given the proximity of state $(1, 3)$ to the goal, moving East is the most effective strategy under uncertainty. If the agent is indeed at $(1, 3)$, this move leads to a non-aliased position adjacent to the goal, allowing for an immediate transition to the objective. If the agent was actually at $(3, 3)$, the Eastward move leads to position $(3, 4)$; here, the agent recovers state certainty and can proceed directly to the goal. Although moving East from $(3, 3)$ is suboptimal under full environmental knowledge, it represents the best possible decision under sensory constraints. Consequently, the average path length is slightly higher than the theoretical minimum, yet it remains the absolute lower bound achievable given the lack of information.

In contrast, in aliased environments, where multiple environmental states produce identical perceptions, STM-BEACS does not always reach the globally optimal solution. However, it still shows a significant improvement over BEACS, consistently reducing the average number of steps required to reach the goal. This indicates that the additional Short-Term Memory mechanism allows the agent to partially resolve aliasing effects and make more informed decisions than the baseline algorithm. The STM-BEACS algorithm is also visibly capable of generating a consistent policy where BEACS failed to do so, such as most non-aliased mazes and, notably, aliased mazes such as Woods100 and Woods101, which show that the added memory mechanism allows for better operation in many aliased mazes.

TABLE III: Average steps to exit and standard deviation for three mazes and three algorithms used in the ablation study. The notation STMnoL refers to the STM-BEACS variant without the sequence-length penalty. Bold indicates the lowest mean in the row. Superscripts summarize Holm-corrected one-sided Welch's t-tests: † significantly better than both alternatives, * significantly better than BEACS only. Mazes are sorted by ascending Complexity.

Maze	BEACS	STMnoL	STM
Littman57	7.45 ± 0.85	7.36 ± 1.07	$\mathbf{4.75 \pm 0.48}^{\dagger}$
Maze10	12.74 ± 2.19	$\mathbf{7.87 \pm 2.82}^{*}$	$\mathbf{6.83 \pm 1.18}^{*}$
MazeF4	6.98 ± 0.65	$\mathbf{4.50 \pm 0.00}^{*}$	$\mathbf{4.50 \pm 0.00}^{*}$

2) Ablation study: To assess the contribution of the two main additions in STM-BEACS, we compared the baseline BEACS with the full STM-BEACS model and an ablated variant without the sequence-length penalty (STMnoL). The results for MazeF4, Littman57, and Maze10 are summarized in Table III.

To complement the descriptive comparison, we performed Holm-corrected one-sided Welch's t-tests on the eight per-run results available for each maze. In MazeF4 and Maze10, both STM-based variants significantly outperform BEACS, while no significant difference is observed between STM and STMnoL. In Littman57, only the full STM-BEACS model is significantly better than both BEACS and STMnoL.

These results suggest that the STM mechanism is the main source of improvement in MazeF4 and Maze10, where it substantially reduces path length even without the penalty term. In contrast, Littman57 benefits primarily from the sequence-length penalty, indicating that discouraging unnecessarily long trajectories is particularly important in that environment. Overall, the memory component appears to be most beneficial in mazes with stronger state ambiguity, whereas the penalty term mainly refines behavior by promoting shorter paths and reducing loops.

3) Generated policy: Figure 2 illustrates two example trajectories generated by the learned policy. In the left image, the agent starts from the state $(4, 1)$, while in the right image it starts from the state $(4, 5)$. These two states are perceptually aliased; in the absence of contextual information, the agent would select the same action in both cases, despite occupying different true states.

After executing the initial action sequence, the agent reaches state $(3, 1)$ or $(3, 5)$, which are also aliased and produce identical perceptions. At this stage, however, the agent has already traversed multiple states and accumulated contextual information stored in its Short-Term Memory, as shown by the blue floating panels in the right panel of Fig. 2, which depict the state of the agent's memory during the first and second visit to the $(3, 5)$ state. This additional information enables the policy to disambiguate the otherwise identical perceptions and select different actions for each trajectory.

As a result, the subsequent action differs between the two paths, directing the agent closer to the goal in each case. This example demonstrates that the Short-Term Memory mechanism in STM-BEACS effectively mitigates perceptual aliasing by incorporating recent interaction history, allowing the agent to reorient itself and make more accurate, goal-directed decisions compared to a setting in which no contextual information is available.

VI. Conclusion

The experimental results demonstrate a clear improvement of STM-BEACS over the baseline BEACS algorithm in the average steps-to-goal metric. The reduction in the number of steps required to reach the goal indicates that the proposed memory mechanism enables the system to construct more accurate and effective policies. By incorporating short-term contextual information into the classifier representation, STM-BEACS can disambiguate perceptually identical states and condition action selection on recent interaction history. This leads to more consistent decision-making and more efficient behavioral trajectories. Overall, the observed performance gains provide empirical evidence that explicit temporal context supports the emergence of more diverse and better-adapted policies compared to the purely reactive BEACS algorithm. Although in some mazes the best possible policy in terms of the average step count was obtained with STM-BEACS, there is still room for improvement. Generating context-aware classifiers is computationally expensive and requires many iterations to train the model sufficiently. Action selection mechanisms that cover the action space in a more efficient way could prove to be very beneficial, allowing for faster training as well as exposing the model to more diverse situations which could further improve the model. Even though the memory mechanism resulted in more directed paths towards the goal, the action sequences can be further optimized. The model often produces actions that revisit the same states, which increased the path to the goal and could be improved.

Acknowledgment

The authors used Codex GPT 5.4 to edit and improve the clarity of a few selected sections of the manuscript. All AI-generated text was reviewed, edited, and verified by the authors.

References

[1] J. H. Holland and J. S. Reitman, "Cognitive systems based on adaptive algorithms," in *Pattern-directed inference systems*. Elsevier, 1978, pp. 313–329.

[2] M. V. Butz, *Anticipatory learning classifier systems*. Springer Science & Business Media, 2002, vol. 4.

[3] J. Hoffmann, "Vorhersage und erkenntnis [prediction and reason]," *Göttingen, Germany: Hogrefe*, 1993.

[4] R. Orhand, A. Jeannin-Girardon, P. Parrend, and P. Collet, "Pepacs: Integrating probability-enhanced predictions to acs2," in *Proceedings of the 2020 Genetic and Evolutionary Computation Conference Companion*, 2020, pp. 1774–1781.

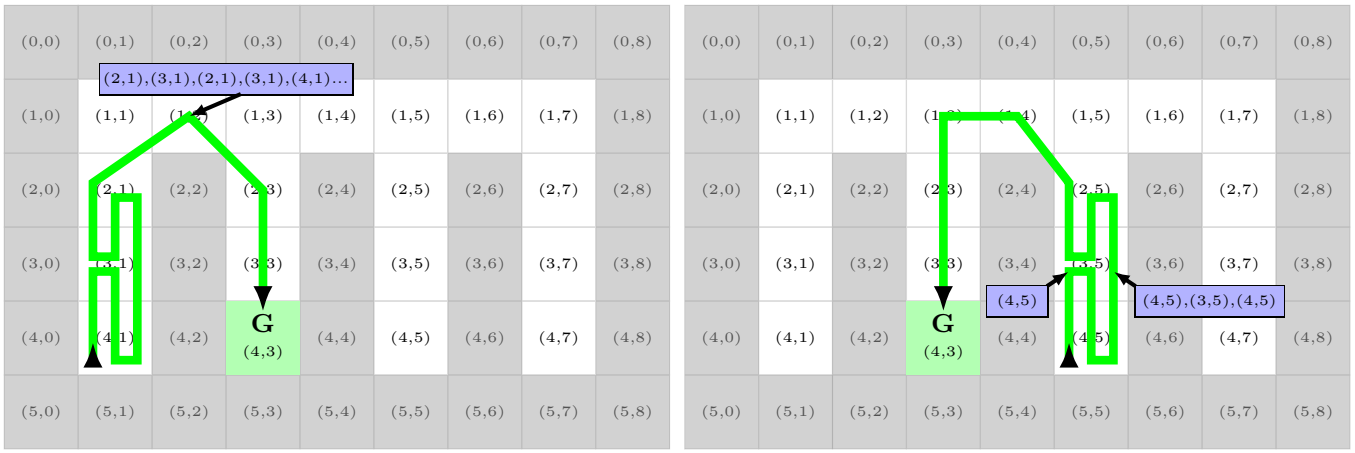

Fig. 2: Comparison of two example trajectories generated by the learned STM-BEACS policy in Maze10. In the left panel, the agent starts from state $(4, 1)$; in the right panel, it starts from state $(4, 5)$. Blue floating panels depict the agent's short-term memory state at specific points along its trajectory. This memory augments the current perceptual state and influences action selection. This effect is visible in the right panel: when the agent revisits state $(3, 5)$, it selects a different action than in the earlier visit, due to the additional contextual information encoded in the memory of previously visited states accumulated during the trajectory.

[5] ——, "Bacs: A thorough study of using behavioral sequences in acs2," in *Parallel Problem Solving from Nature–PPSN XVI: 16th International Conference, PPSN 2020, Leiden, The Netherlands, September 5-9, 2020, Proceedings, Part I 16*. Springer, 2020, pp. 524–538.

[6] ——, "Accurate and interpretable representations of environments with anticipatory learning classifier systems," in *European Conference on Genetic Programming (Part of EvoStar)*. Springer, 2022, pp. 245–261.

[7] P. L. Lanzi, "Adding memory to xcs," in *1998 IEEE International Conference on Evolutionary Computation Proceedings. IEEE World Congress on Computational Intelligence (Cat. No. 98TH8360)*. IEEE, 1998, pp. 609–614.

[8] P. L. Lanzi *et al.*, "An analysis of the memory mechanism of xcsm," *Genetic Programming*, vol. 98, pp. 643–651, 1998.

[9] Z. Zang, D. Li, and J. Wang, "Learning classifier systems with memory condition to solve non-markov problems," *Soft Computing*, vol. 19, pp. 1679–1699, 2015.

[10] A. Siddique, W. N. Browne, and G. M. Grimshaw, "Frames-of-reference-based learning: Overcoming perceptual aliasing in multistep decision-making tasks," *IEEE Transactions on Evolutionary Computation*, vol. 26, no. 1, pp. 174–187, 2021.

[11] F. Uwano and W. Browne, "Hierarchical frames-of-references in learning classifier systems," in *Proceedings of the Companion Conference on Genetic and Evolutionary Computation*, 2023, pp. 335–338.

[12] F. Uwano and W. N. Browne, "Cognitive learning system for sequential aliasing patterns of states in multistep decision-making," in *Proceedings of the Genetic and Evolutionary Computation Conference Companion*, 2024, pp. 315–318.

[13] ——, "Enhancing xcs with dual-stream identification for perceptual aliasing in multi-step decision-making," in *Proceedings of the Genetic and Evolutionary Computation Conference*, 2025, pp. 499–507.

[14] A. Tomlinson and L. Bull, "A corporate xcs," in *International Workshop on Learning Classifier Systems*. Springer, 1999, pp. 195–208.

[15] M. V. Butz, D. E. Goldberg, and W. Stolzmann, "Probability-enhanced predictions in the anticipatory classifier system," in *International Workshop on Learning Classifier Systems*. Springer, 2000, pp. 37–51.

[16] M. Łabędzki and O. Unold, "Why state differentiation in acs2 is not enough in aliased environments," in *Proceedings of the Genetic and Evolutionary Computation Conference Companion*, 2025, pp. 2249–2257.

[17] G. Énée and M. Peroumalnaïk, "Adapted pittsburgh classifier system: building accurate strategies in non markovian environments," in *Proceedings of the 10th annual conference companion on Genetic and evolutionary computation*, 2008, pp. 2001–2008.

[18] L. Chrisman, "Reinforcement learning with perceptual aliasing: The perceptual distinctions approach," in *AAAI*, vol. 1992. Citeseer, 1992, pp. 183–188.

[19] A. J Bagnall and Z. V Zatuchna, "On the classification of maze problems," *Foundations of Learning Classifier Systems*, pp. 305–316, 2005.

[20] P. Gerard, W. Stolzmann, and O. Sigaud, "Yacs: a new learning classifier system using anticipation," *Soft Computing*, vol. 6, no. 3-4, pp. 216–228, 2002.

[21] P. Gérard, J.-A. Meyer, and O. Sigaud, "Combining latent learning with dynamic programming in the modular anticipatory classifier system," *European Journal of Operational Research*, vol. 160, no. 3, pp. 614–637, 2005.

[22] R. Orhand and A. Ayadi, "Towards a neurosymbolic approach based on anticipatory learning classifier systems and ontologies," *Procedia Computer Science*, vol. 270, pp. 1232–1241, 2025.

[23] W. Stolzmann, "An introduction to anticipatory classifier systems," in *International Workshop on Learning Classifier Systems*. Springer, 1999, pp. 175–194.

[24] K. Miyazaki and S. Kobayashi, "Proposal for an algorithm to improve a rational policy in pomdps," in *IEEE SMC'99 Conference Proceedings. 1999 IEEE International Conference on Systems, Man, and Cybernetics (Cat. No. 99CH37028)*, vol. 5. IEEE, 1999, pp. 492–497.

[25] S. Arai and K. Sycara, "Credit assignment method for learning effective stochastic policies in uncertain domains," in *Proceedings of the 3rd Annual Conference on Genetic and Evolutionary Computation*, 2001, pp. 815–822.

[26] J. Loch and S. Singh, "Using eligibility traces to find the best memoryless policy in partially observable markov decision processes." in *ICML*, vol. 98. Citeseer, 1998, pp. 323–331.

[27] M. Métivier and C. Lattaud, "Anticipatory classifier system using behavioral sequences in non-markov environments," in *International Workshop on Learning Classifier Systems*. Springer, 2002, pp. 143–162.