ADAM v0.9

Hierarchical Autonomous Agent

Technical Benchmark Report

Zero-shot navigation, adaptive combat, open-world survival

No pre-training | No neural networks in decision loop

608 unit tests | 12+ modules | 4 environment families

Report generated: February 10, 2026

1. Executive Summary

ADAM (Adaptive Decision-Action Memory) is a hierarchical autonomous agent that navigates, solves puzzles, avoids obstacles, engages in combat, and survives in open-world environments - all zero-shot, without any pre-training or gradient-based learning.

Key results across four distinct environment families:

- MiniGrid Navigation: 100% goal rate on Empty, MultiRoom (N2-N6), Unlock - zero-shot
- LavaCrossing: 100% on S9N1, 90% on S9N2, 70% on S11N5 - zero-shot (vs ~95% for trained PPO)
- Prince of Persia: 100% goal, 10/10 guard kills, 0 deaths - trial-and-error combat discovery
- Crafter (experimental): 15 unique achievements in 2h self-learning run (4553 episodes), no gradients

The agent uses no neural networks in its decision loop. All behavior emerges from the interaction of cognitive modules: spatial memory, diffusion-based wavefront planning, drive-based excitation, trial-and-error combat learning, cross-episode methodology, and macro-skill ladders.

1.1 Counter-Intuitive Finding

The central finding challenges the prevailing assumption that more data and larger models always produce better agents. On LavaCrossing-S9N1, standard RL (PPO) achieves ~95% after 1M+ training steps. ADAM achieves 100% with zero training steps. The difference is architectural: correct cognitive decomposition eliminates the need for statistical learning on problems that have structural solutions.

A single cognitive rule - 'static danger is a wall, not a monster' - improved goal rates by +60 to +85 percentage points across three LavaCrossing variants. No retraining, no hyperparameter sweep. This suggests that for a class of navigation problems, the right cognitive prior is worth more than millions of training steps.

2. Architecture Overview

ADAM implements a three-level cognitive hierarchy inspired by subsumption architecture, with higher levels modulating lower ones:

Level 1 - Atomic Perception

Pool of atomic feature detectors (color, shape, texture, motion). Each detector outputs a binary prediction with confidence. No CNNs - uses hand-crafted features with learned thresholds. Serves as the sensory foundation.

Level 2 - Object Recognition

Pattern matcher combines atomic features into object hypotheses. Maintains a running codebook of discovered object types. Classifies objects into categories: food, danger, tool, gate, shelter. Category assignments are learned from reward feedback, not hardcoded.

Level 3 - Planning & Navigation

The highest level integrates multiple cognitive modules:

- Excitation Policy: internal drives (hunger, fear, curiosity) compete to set the current behavioral mode
- Priority Arbiter: resolves conflicts between flee, fight, eat, explore with strict ordering
- Spatial Memory: maintains an internal grid map of explored cells with category labels
- Diffusion Planner: wavefront expansion from target through explored topology - routes around walls and danger
- Geographer: fog-of-war tracking, compass orientation, room graph for multi-room layouts
- Mathematician: distance estimation, reachability scoring, cost-benefit analysis for targets
- Methodist: cross-episode learning - remembers which actions worked in which contexts (4 levels)
- Methodist Explorer: context-bucketed action-value memory + macro-skill ladder that learns priorities via bandit-style exploration across episodes
- Goal Stack: hierarchical subgoal decomposition (find key -> unlock door -> reach goal)
- Combat Memory: trial-and-error action discovery with EMA scoring and epsilon-greedy exploration

3. MiniGrid Benchmark Results

MiniGrid is a standard benchmark suite for grid-world agents. All results are zero-shot (no training episodes), 20 episodes per environment.

3.1 Navigation Tasks

Environment	Goal %	Died %	Avg Steps	Status
Empty-16x16	100%	0%	26	PASS
MultiRoom-N2-S4	100%	0%	7	PASS
MultiRoom-N4-S5	100%	0%	40	PASS
MultiRoom-N6-S6	100%	0%	115	PASS
DoorKey-5x5	75%	0%	39	PASS
DoorKey-6x6	95%	0%	19	PASS
Unlock	100%	0%	19	PASS
UnlockPickup	70%	0%	44	PASS

Navigation performance is near-optimal. Agent handles multi-room exploration (up to 6 rooms), key-door puzzles, and large grids efficiently. DoorKey-5x5 (75%) is lower due to asymmetric room layouts where key placement is adversarial.

3.2 LavaCrossing (Obstacle Avoidance)

LavaCrossing tests navigation through corridors blocked by lava streams. Touching lava = instant death. This tests the agent's ability to plan safe paths around static hazards.

Environment	Goal %	Died %	Steps	Before	Delta
S9N1 (1 lava)	100%	0%	17	30%	+70pp
S9N2 (2 lava)	90%	5%	26	5%	+85pp
S11N5 (5 lava)	70%	20%	35	10%	+60pp

Key insight: "Lava is a wall, not a monster." Previous version triggered panic-flee toward lava because it treated all danger identically. The fix introduces cognitive separation: static dangers (lava, spikes) are routed around by the planner, dynamic dangers (enemies) trigger flee/fight response. One architectural change, three environments fixed.

3.3 Comparison: ADAM vs Random Baseline

Environment	Random Goal%	ADAM Goal%	ADAM Steps
Empty-8x8	25%	100%	10
DoorKey-8x8	5%	100%	30
FourRooms	0%	70%	32

For reference: standard RL agents (PPO/DQN) on LavaCrossing-S9N1 achieve ~95% after 1M+ training steps. ADAM achieves 100% with zero training steps. On harder variants (S11N5), RL typically reaches 80-90% after extensive training; ADAM reaches 70% zero-shot.

4. Prince of Persia Benchmark

Prince of Persia (via SDLPoP) provides a fundamentally different challenge: platformer physics, sword combat, moving enemies, traps. This tests whether the same architecture generalizes beyond grid worlds.

4.1 Combat System - Trial-and-Error Discovery

The combat system uses zero hardcoded strategies. The agent discovers effective actions through experimentation:

- Agent encounters guard, takes damage, enters combat mode
- Tries random actions from action space (eat, stay, up, down, left, right)
- Discovers "eat" deals damage to guard (EMA reward tracking)
- Reinforces successful actions via epsilon-greedy with decay
- Learns combo sequences for multi-hit efficiency

4.2 Results (Mock Environment)

Level	Goal %	Died %	Steps	Kills
Level 1 (no combat)	100%	0%	25	0
Level 2 (guard)	100%	0%	33	10/10

Level 2 introduces an armed guard blocking the path. The agent consistently discovers the effective combat action, defeats the guard (HP 3->0 in 3 hits), and proceeds to the goal. 10/10 kills across 10 episodes, 0 agent deaths.

4.3 Key Technical Details

- CombatMemory: EMA scoring (alpha=0.3) tracks action effectiveness, epsilon-greedy exploration (eps=0.4, decay to 0.05)
- Priority 1.5: combat priority sits between flee (P1) and goal navigation (P2) in the arbiter
- Stuck detection bypass: combat actions (stay/eat) don't move the agent but are intentional - stuck detector is disabled during combat

5. Crafter Survival Benchmark (Experimental)

Crafter is a 2D open-world survival environment inspired by Minecraft. Unlike MiniGrid, Crafter requires long-horizon competence: continuous resource acquisition (food/drink), tool progression (wood -> stone -> iron), and reactive threat handling (zombies/skeletons). This tests whether the cognitive architecture extends to open-ended survival tasks.

5.1 Protocol

- Runtime budget: 2 hours wall-clock
- Episodes: 4553 (all ended by death/terminal condition)
- Max steps per episode: 800
- Exploration: epsilon-greedy direct action probes (base_epsilon=0.25)
- Threat trigger radius: defense_dist=6 (Manhattan distance)
- Learning: no backpropagation, no replay buffer; purely EMA-based cross-episode memory

5.2 Results Summary

Metric	Value	Note
Unique achievements	15	discovered
Mean survival steps	204.9	per episode
Median survival	194	steps
Total episodes	4553	2h runtime

5.3 Death Cause Analysis

Death Cause	Count	% of Total	Source
Combat	2331	51.2%	zombie/skeleton
Thirst	1610	35.4%	dehydration
Hunger	330	7.3%	starvation
Unknown	278	6.1%	misc
Lava	3	0.07%	terrain
Exhaustion	1	0.02%	energy

5.4 Behavioral Telemetry

Defense actions under close danger:

- Flee: 184,249 actions
- Attack (eat): 9,792 actions
- Barrier (place_stone): 77 actions

Direct crafting/survival actions discovered:

- sleep: 3,547 | place_table: 2,926
- make_wood_sword: 1,483 | make_wood_pickaxe: 598
- place_furnace: 21

5.5 Learning Dynamics and Plateau

Unique achievements increased rapidly early on: by episode 1050 (~30 min) the agent reached 15 unique achievements. For the remaining ~90 minutes (episodes 1050-4553), no new achievements were discovered.

Interpretation: the plateau exposes a fundamental limitation of step-wise trial-and-error. Crafter achievements typically require multi-step causal chains (e.g. 'find water -> secure shelter -> upgrade tools -> mine iron'). When experiments are single-step probes, credit cannot propagate over the full chain. This motivated the macro-skill ladder extension.

6. Methodist Explorer: Self-Learning Macro-Skills

To address the credit-horizon plateau observed in Crafter, we extended the Methodist module with an Explorer layer that learns which multi-step skill families to prioritize, rather than selecting atomic actions.

6.1 Core Idea: Skills as Learned Priorities

Instead of a flat policy over actions, the agent selects among interpretable skills:

- defense: react to threats (flee/attack/barrier/turn)
- hydrate: seek water and drink
- food: seek food sources and eat
- rest: sleep when safe and energy-limited
- progress: execute tool-progression pipeline (table -> sword/pickaxe -> stone -> furnace -> iron)

Each skill is multi-step: it can remain active for up to K steps (ladder_max_steps=80), generating one atomic action per environment step, but preserving temporal coherence.

6.2 Skill Selection: UCB + Risk Penalty

For each skill s, maintain EMA statistics (expected return, death risk). Select skills using a risk-aware UCB score:

score(s) = prior(s) + ema_reward(s) - lambda * ema_death(s) + c * sqrt(log(1+N)/(1+n_s))

- prior(s): urgency from internal state (low drink increases hydrate prior)
- lambda: death_penalty=6.0 (risk sensitivity)
- c: ladder_ucb_c=0.7 (exploration pressure)
- Sticky continuation: keep current skill if still applicable (reduce thrashing)
- Survival interrupts: critically low drink/food immediately preempts progress

6.3 Context-Bucketed Action-Value Memory

Below the skill layer, Explorer learns action preferences in coarse context buckets (energy/safety bins), with diffusion over neighboring bins to generalize across similar states. Terminal events propagate credit/blame to the last 8 decisions ('what happened before death'), enabling causal learning without backpropagation.

Three granularity levels: object-specific -> context-bucketed -> global. Combined via weighted average proportional to evidence count. This allows the agent to both specialize (zombie at low health -> flee) and generalize (unknown danger -> cautious default).

6.4 Design Properties

- No neural networks: all memory is symbolic EMA tables
- Interpretable: every decision has a readable 'reason' string
- Portable: skills are defined by internal state, not environment-specific rewards
- Credit horizon extended from ~1 step to ~80 steps per skill

7. Architectural Properties

7.1 Module Reuse Across Environments

The same cognitive modules work across MiniGrid (grid navigation), LavaCrossing (obstacle avoidance), Prince of Persia (platformer combat), and Crafter (open-world survival). Only the environment adapter changes.

Module	MiniGrid	PoP	Crafter
Spatial Memory	Grid map	Level layout	World map
Diffusion Plan.	Around lava	Around traps	Around danger
Goal Stack	Key -> door	Sword -> fight	N/A
Combat Memory	N/A	Sword attacks	N/A
Methodist	Cross-ep nav	Cross-ep acts	Skill ladder
Explorer	N/A	N/A	Macro-skills

7.2 Zero-Shot Generalization

ADAM requires zero training episodes. All behavior emerges from the interaction of its cognitive modules at runtime. The agent:

- Explores via internal curiosity drive + fog-of-war tracking
- Builds spatial maps from observations in real-time
- Plans paths using wavefront diffusion through explored topology
- Discovers combat strategies through trial-and-error within a single episode
- Learns macro-skill priorities across episodes via bandit-style exploration

7.3 Cognitive Separation of Concerns

A key architectural principle: different types of danger require different cognitive responses:

- Static danger (lava, spikes, pits): treated as impassable terrain. Planner routes around. No panic response.
- Dynamic danger (enemies, guards): triggers flee (if unarmed) or fight (if armed). Real-time action selection.
- Survival pressure (thirst, hunger): activates corresponding macro-skill with urgency prior. No flee behavior.

This cognitive taxonomy - one if-statement per category - improved LavaCrossing goal rates from 5-30% to 70-100% across all variants.

7.4 Current Limitations

- Discrete action space only - no continuous control
- Grid-world tested - not validated on pixel-based or 3D environments
- Real PoP via SDLPoP: 0% success - screen parsing and timing not yet solved
- LavaCrossing S11N5: 20% death rate - planner limited by fog-of-war on dense lava
- No learned visual features - atomic detectors are hand-crafted
- Crafter plateau at 15 achievements - step-wise credit assignment insufficient for long causal chains
- Open-world resource localization needs persistent landmark memory; current anchors are episode-local

8. Engineering Quality

Project statistics:

- Test coverage: 608 passing tests, 0 failures
- Module count: 12+ cognitive modules + 4 environment adapters
- Code size: ~20,000 lines (agent) + ~12,000 lines (tests)
- Dependencies: numpy only (no PyTorch/TensorFlow in agent)
- Environments: MiniGrid (11 variants), Prince of Persia, Crafter
- Benchmarking: automated suite with JSON result logging, 58 benchmark files

9. Relevance to Robotics

While ADAM operates in grid worlds, the architectural principles address problems common to robotics autonomy stacks:

- Hierarchical decision-making: perception -> object recognition -> planning mirrors robot cognitive architectures
- Spatial memory + wavefront planning: analogous to SLAM + path planning in mobile robotics
- Trial-and-error action discovery: similar to RL-based manipulation, but without neural network overhead or sim2real gap
- Morphology-agnostic design: same modules work across different environments via adapter layer - like a brain that works across body types
- Zero-shot adaptation: no retraining when environment changes - critical for real-world deployment where retraining is expensive
- Static vs dynamic danger: cognitive separation maps directly to robot safety (avoid walls vs react to moving obstacles)
- Macro-skill learning: multi-step skill selection via bandit-style UCB parallels options/skills frameworks in robot learning

10. Open Research Question

The central question this work raises: if correct cognitive decomposition gives 100% zero-shot performance where trained RL gives 95%, does this principle scale to continuous 3D environments with real physics?

Two hypotheses follow from ADAM's results:

- Hypothesis A (strong): A transformer-based autonomy stack would benefit from explicit cognitive priors (static/dynamic danger separation, hierarchical macro-skills, spatial memory as a distinct module) rather than learning them implicitly from data.
- Hypothesis B (weak): Even if end-to-end learning eventually matches hand-structured cognition, the data/compute cost to learn what one if-statement provides is a significant practical inefficiency.

ADAM does not claim to solve continuous robotics. It demonstrates that cognitive structure can substitute for statistical learning on a meaningful class of problems, and proposes this as a complementary direction to pure end-to-end approaches.

Report date: 2026-02-10

ADAM v0.9 | *608 tests* | *MiniGrid + PoP + Crafter*