# Brain-Inspired Resilient Systems for AI

## The PURER Loop: Perceive, Update, Reason, Reflect, Execute

**Presenter:** Ravi K. Iyer[2]

**Collaborators:** Haoran Qiu[1], Phuong Cao[2,3], Shengkun Cui[2], Archit Patke[2]

[1]Microsoft Azure Research [2]University of Illinois Urbana-Champaign [3]NCSA

ILLINOIS
Electrical & Computer Engineering
GRAINGER COLLEGE OF ENGINEERING

Microsoft Research

ILLINOIS CSL NCSA

# State of the Practice in Dependable Computing (incomplete List)

## Hardware

- ECC and parity
- TMR and N-modular redundancy
- JPL STAR computer
- Lockstep executions
- Time redundancy
- IBM 360 and beyond (Z series)
- Tandem Computers
- Measurement and modeling, analytics

## Networks and Distributed Systems

- Checksums and CRCs
- Heartbeats and failure detectors
- Primary-backup failover, Mirroring, RAID
- Consensus protocols and Byzantine fault tolerance

## Software

- N-version programming
- Recovery blocks
- IBM recovery routines
- Checkpointing and rollback
- Assertions
- Formal methods

## Operating Systems

- Virtual memory and address-space protection
- Process isolation
- Filesystem journaling
- Privilege separation and sandboxing

Classic dependability methods continue to evolve, but have begun to plateau.

**Machine learning—an old idea in a new system-level role—introduces new sources of uncertainty that create substantial concern among both established and emerging constituents, especially in high-stakes domains (e.g., Healthcare, Aviation, Autonomy Finance).**

# Massive infusion of Accelerators (GPUs) in Cloud and HPC- Spectacular Failures

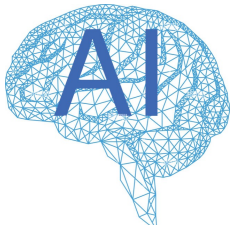
- **Large failures in critical AI infrastructures and agentic methods are increasingly impacting the world, often creating disruption and chaos.**
  - Cloudflare 2025 Outage
  - AWS 2025 Outage
  - Sullivan & Cromwell court-filing errors due to AI “hallucinations,” 2026

- **We can fight the current battle:**
  - by combining our classic successes with new methods to handle the new level of uncertainty, introduced by variety and ubiquitous adaption of machine learning methods.

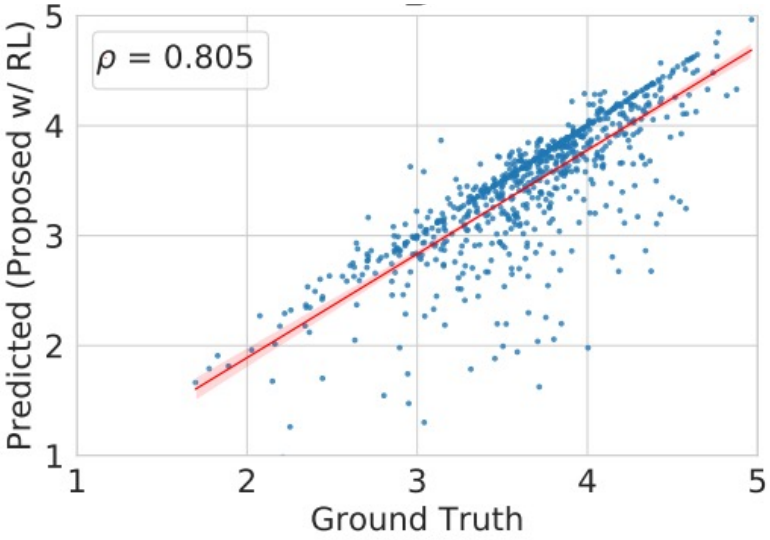
- **Or we can take a leap forward to brain-inspired resilience.**
  - not an unfamiliar idea, but one not yet fully achieved.
  - High Resilience, low energy consumption, “slow” learning through reward and reinforcement, fully generative



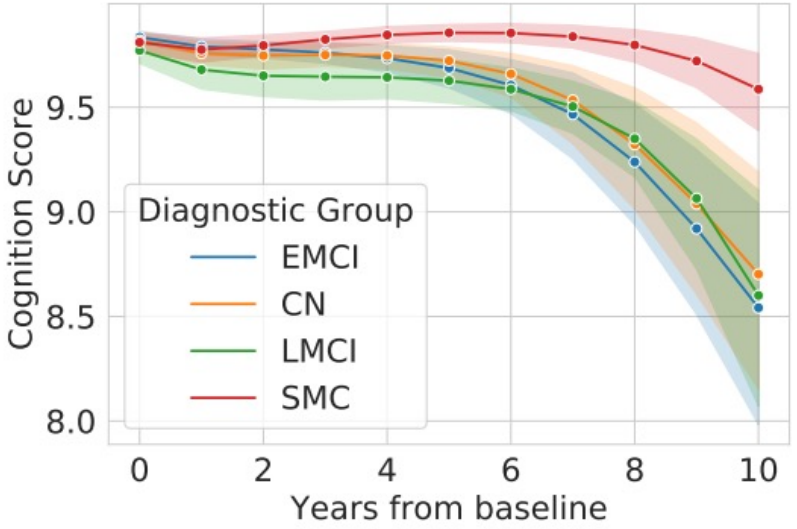


# Why Brain-Inspired Resilience: Group-level Trends from an Alzheimer's model




Predicted vs ground truth size of hippocampus (ADNI).

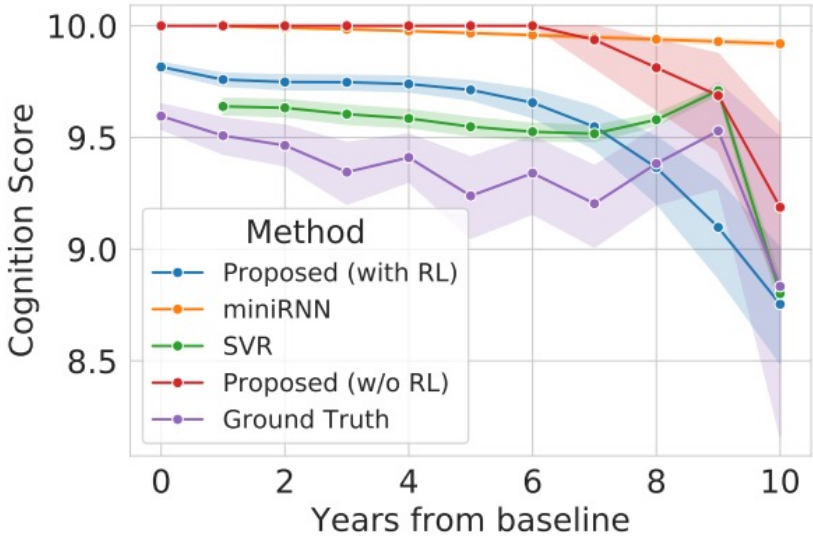



Predicted cognitive trajectories averaged across people within different diagnosis groups (ADNI). CN, cognitively normal; EMCI, early mild cognitive impairment; LMCI, late mild cognitive impairment; SMC, significant memory concern.

- High correlation between predicted and true size of brain regions
- SMC showed better cognition after 10 years than MCI groups
  - Diagnosis information was not used in the model

# More on Why Brain-Inspired Resilience: Individualized 10-year cognition prediction

Proposed model generates better/more realistic individualized cognitive trajectories that other methods



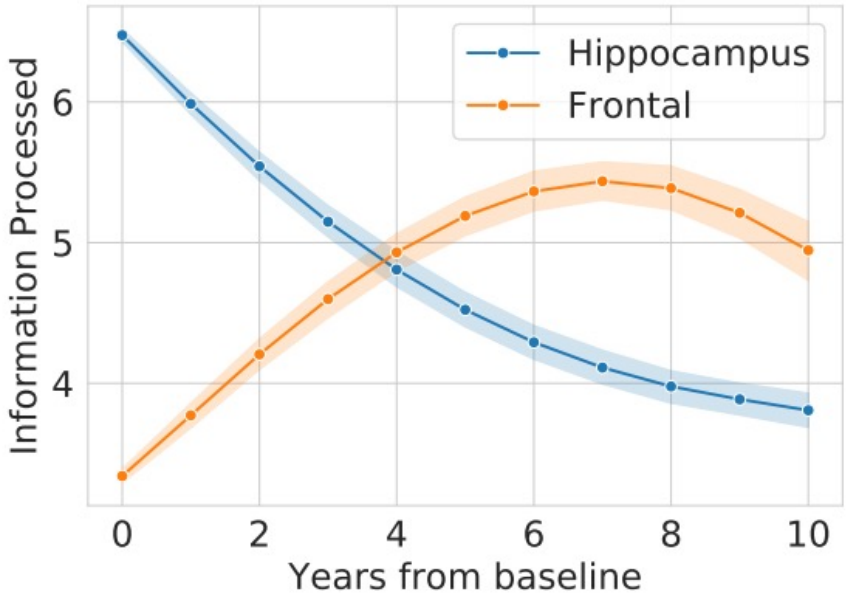

Predicted cognitive trajectories averaged across the population (ADNI).

| Method | Synthetic Data | | ADNI | |
|---|---|---|---|---|
| | MAE | MSE | MAE | MSE |
| **Proposed (with RL)** | **0.641 (0.090)** | **0.910 (0.229)** | 0.537 (0.127) | 0.761 (0.370) |
| Proposed (w/o RL) | 1.009 (1.670) | 3.806 (8.073) | 0.595 (0.137) | 1.112 (0.406) |
| minimalRNN | 1.395 (0.149) | 6.971 (0.753) | 0.599 (0.137) | 0.984 (0.659) |
| SVR | 0.658 (0.050) | 0.997 (0.112) | **0.495 (0.067)** | **0.574 (0.230)** |

Mean squared error (MSE) and mean absolute error (MAE) for 10-year cognition trajectory prediction.

# How the Brain Achieves Resilience Model demonstrates compensation

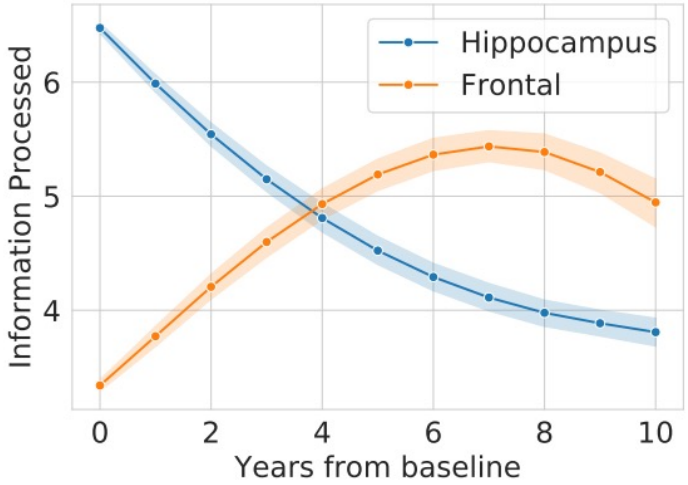
- Model shows biologically plausible behaviors that were not explicitly encoded
- Shift in contribution to cognition could be related to compensatory processes




Shift in contribution to cognition from hippocampus to PFC. Average $I_v(t)$ over the population (ADNI).

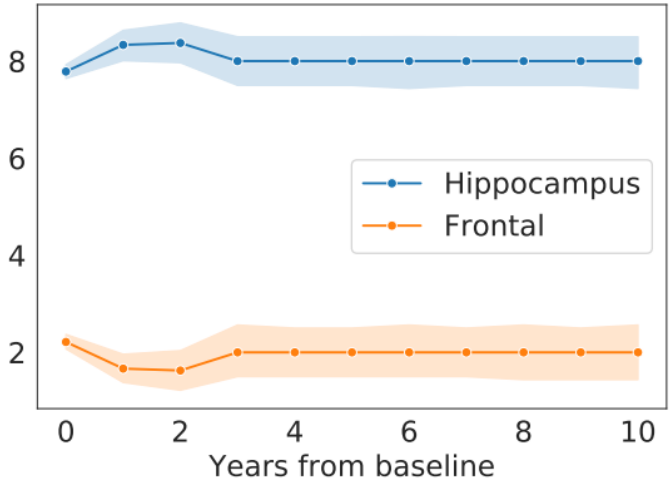
# Insight into Compensation and Resilience

**A.** Balance cognition and cost



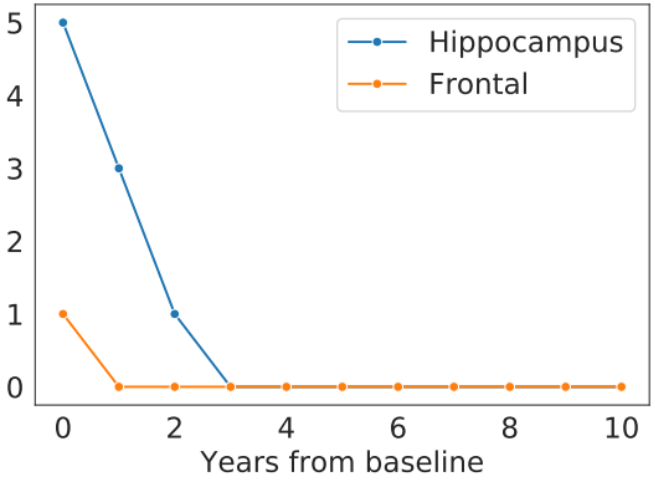

$$R(t) = -[\lambda(C_{task} - C(t)) + M(t)]$$

**B.** Maximize cognition




$$R(t) = -(C_{task} - C(t))$$

**C.** Minimize energetic cost



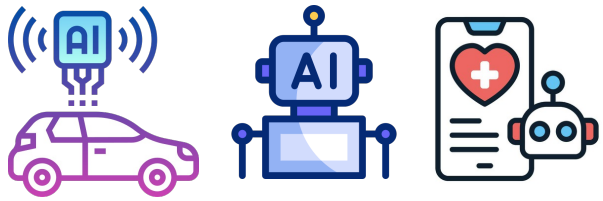

$$R(t) = -M(t)$$

Average information processed by hippocampus and PFC. Models were trained with different reward functions (ADNI).

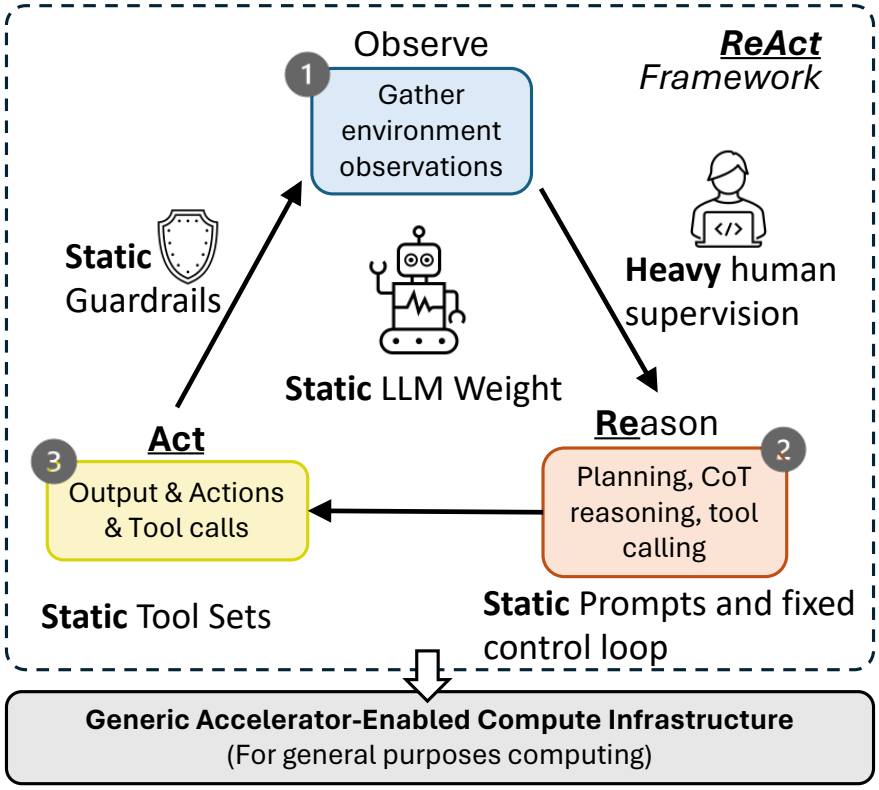
In the model, compensation was a result of both terms in the reward function.

# State of the Practice in Dependable Computing

- The “NEW” Emergent dependable/trustworthy computing community sometimes reinvents and adapts methods developed by this community, often with impunity.
- Yet, important new problems introduced by machine learning and its associated uncertainty have not been addressed, classic uncertainty combined with AI uncertainty – managing and controlling superlarge-scale systems, resilience/trustworthiness – a major challenge.
- Especially, classic methods could form a basis for addressing these emerging problems.
- At the core of ML in practice are large language models, which have become ubiquitous.
- **Agents leverage LLMs to address a variety of problems**, including managing large-scale clouds, autonomous vehicles, major infrastructure diagnosis and recovery, finance, law, and healthcare, but with uncertain success.



# State of the Practice of Agentic Framework



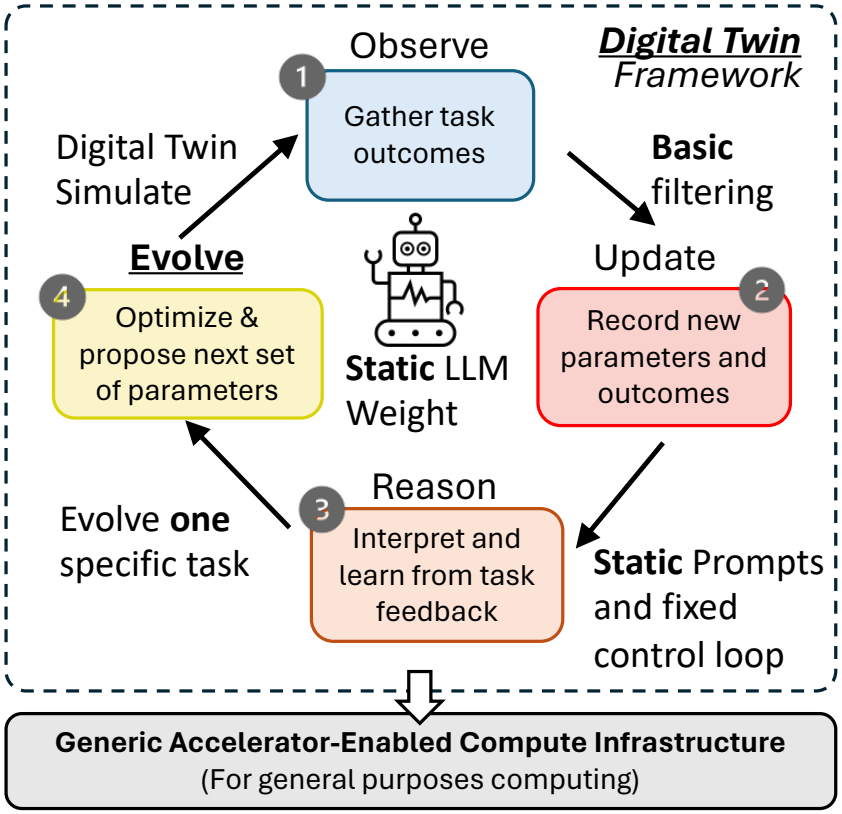

## Characteristics

- **Substantially static** prompts, model weight, tool sets, and guardrails. Counterfactual reasoning
- **Minimally dynamic** control logic.
- **Emergent task-specific CoT systems.**
- Infrastructure and Major application failure diagnosis feedback is incorporated **offline** and **in hindsight.**

## Limitations

- Prompts, tools, and guardrails limit to the specific task, **often fail short of OOD tasks and scenarios.**
- **Reliability and safety** heavily rely on quality of the tool and guardrails, **intensive human efforts.**
- **Little to no online active learning** from errors, failures, and mistakes, or prior experience.

# Emerging: “Self) Evolving System; Digital Twins



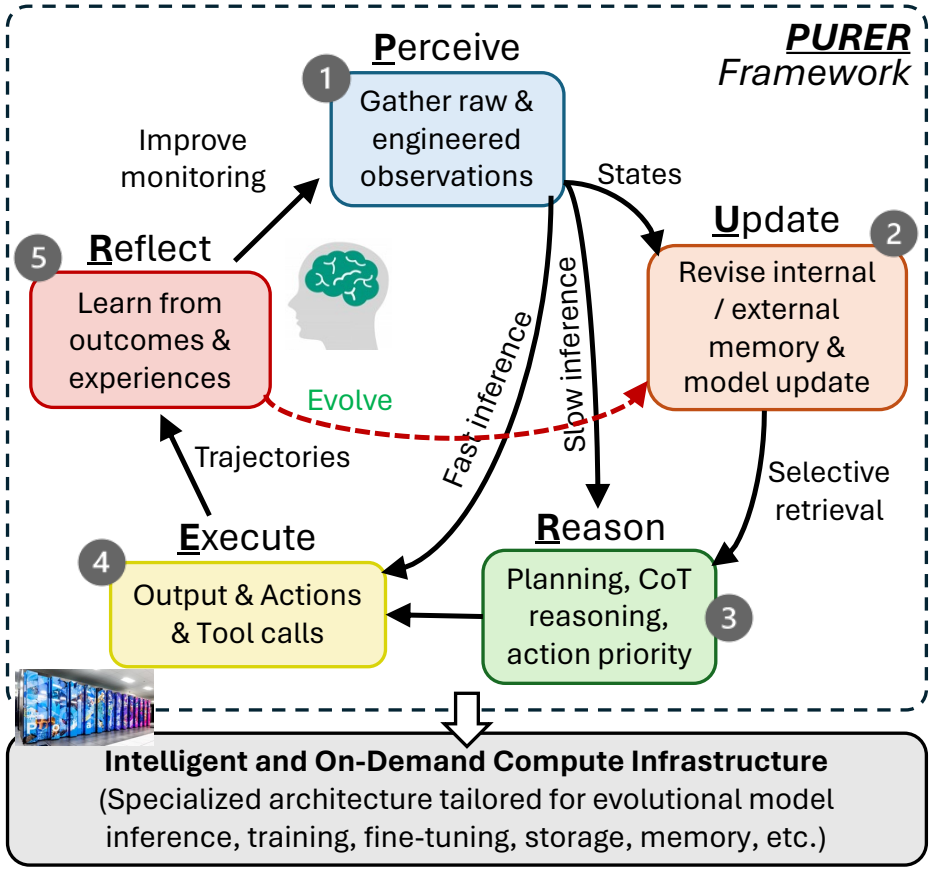

## An attempt to improve:

- **Task-specific** optimization via parameter search with evolutionary algorithms.
- **Improves safety and robustness**: agent as a parameter searcher using digital twins, but not as an action-taker.
- **Trial-and-error:** learn and optimize from previous suboptimal.

## Limitations

- **Static and manually crafted** prompts, model weight, tool sets, and guardrails still apply.
- **Requires** high-fidelity and high-performance simulation for quality parameter search.
- **Often stuck** in local optima.
- **No meta-level reasoning**: reasoning local to specific tasks or steps.

# PURER: Perceive Update Reason Reflect Execute




## PURER:

- **Self-evolving:** Meta loop learns how/what/when to evolve: tool sets, contexts, structures, and guardrails.
- Massive and rapid **counterfactual reasoning** under duress.
- **Reflect, revitalized, and renew**: scalable and collaborative decision-making.
- **Continued online learning** and adaptation from failures and experience.
- **Minimal energy consumption**.

## Advantages

- **Adaptive control and decision-making:** long-horizon robustness achieved by evolving context, memory, and control structure.
- **Experience transfer:** automatically adapt prior errors, failures, and experiences to overcome local optima.
- **New abstractions:** jointly manage and update memory, context, model updates, and resources while ensuring stability, resilience, observability, and correctness.

## None of these methods currently exists.

- Online model update and evolution
- Uncertainty management
- Verification & validation
- Self-evolving abstractions
- Energy-efficient system designs

**Thank You!**

# Backup Slides Starts Here

# PURER: The Five-Phase Self-Evolving Loop

**Perceive** — Ingests raw observations: sensor inputs, logs, tool outputs, and metadata. Operates under partial observability.

**Update** — Revises internal state: updates memory, refines world models, and compresses long-term experience.

**Reason** — Plans and prioritizes conditioned on perceptions, goals, and reward models. Weighs uncertainty and long-term utility.

**Execute** — Enacts decisions through tool invocations, control commands, and environment interactions. Closes the cognition-action loop.

**Reflect** — Evaluates outcomes vs. expectations. Transforms successes and failures into learning signals that inform what should evolve next.

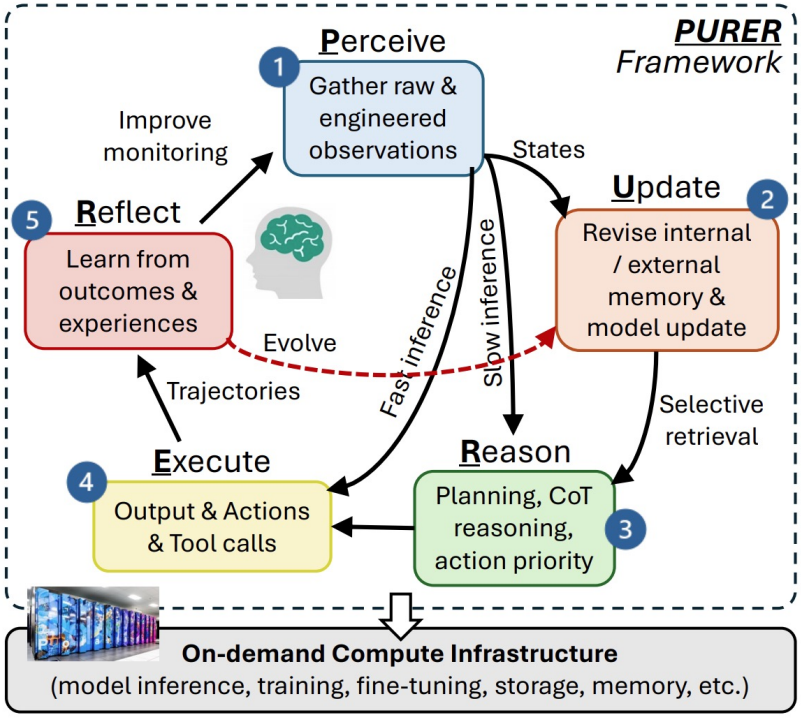



**Figure 1.** The PURER self-evolving framework: a closed loop where operational experience continuously drives system adaptation.

# What is an agent?

- An agent is a system that can observe a situation, decide what to do, take actions, and use feedback to continue working toward a goal.

- An agent follows the **observe → think/plan → act → observe result → adjust → act again** loop, until either exhaustion of allowable steps or until the goal is reached.

# From Static to Self-Evolving

***Current State of the Art*** ***Emerging Approaches*** ***Our Proposal (PURER)***

### Existing Agentic Systems

*Static, reactive, human-maintained*

- Static structure and fixed tools
- Hardcoded control logic (e.g., ReAct)
- Predefined recovery mechanisms
- Manual guardrails and rule-based
- No online learning from operational experience

### Emerging Approaches

*Partial autonomy*

- **AlphaEvolve**: evolutionary policy improvements via parameter search
- **PRAXIS (DSN 2026)**: graph constrained reasoning for specific domain.
- Skill/memory-based learning
- No meta-level reasoning
- Guardrails: static and external

### PURER: Meta-Guided Evolution

***Our proposal – human-inspired autonomy***

- Meta loop learns how/what/when to evolve
- RL-guided tool selection and creation
- Dynamic guardrail generation
- Counterfactual reasoning for decisions
- Reflection-driven adaptation signals
- Evolution itself becomes a learnable policy

# The Resilience Gap in Modern AI Systems

### Static Guardrails & Rules

- Failures must be enumerated **a priori**
- Brittle policies, frequent false positives/negatives
- Adversaries iteratively exploit weaknesses
- Prompt injection remains pervasive

### Offline Post-Training Recovery

- Slow, costly, reactive — long turnaround
- Fine-tuning causes **catastrophic forgetting**
- May worsen robustness under distribution shift
- Treats failures as isolated events

### Context & Memory Mitigations

- Limited continuity, not true adaptation
- Context windows bounded by attention decay
- External memory can be incomplete or stale
- Poor generalizability across domains

### Core Insight

Today's dependability pipelines are **fundamentally reactive** — fixed detection and recovery that neither anticipates novel failures nor learns persistently from experience. We need paradigms that treat **adaptation and learning as first-class runtime concerns**.

# Why We Need Reflection in the Loop

## Human Decision-Making Under Stress

**1. Slow but robust under uncertainty**

Humans prioritize adaptability over short-term optimality when operating under incomplete information.

**2. Continuous experience integration**

New observations are integrated with prior experience, transferring lessons across tasks and contexts.

**3. Failure as a learning signal**

Failure is not an exception to be masked, but a valuable signal that informs future behavior.

## The Missing Piece in Modern AI

Current AI lacks a **self-evolving loop** that treats operational experience as a first-class signal for improvement.

- Failures are **detected and masked**, not learned from
- Degradations and near-misses are **discarded**
- Agents and agentic systems remain **static** until manual intervention
- No mechanism for **functional transfer** across agents

Result: similar errors recur, adversaries adapt faster than defenses, and long-horizon robustness remains elusive.

## Research Question

Can we design AI systems that **continuously evolve** their cognition, memory, and operational substrate by treating every experience — including failures — as a fundamental learning trigger?

Our proposal: the **PURER loop** (Perceive, Update, Reason, Execute, Reflection) — a cognitive-inspired framework for resilient AI systems.

# The PURER Framework

A Self-Evolving Decision Framework Inspired by Human Resilience

# PURER: The Five-Phase Self-Evolving Loop

**Perceive** — Ingests raw observations: sensor inputs, logs, tool outputs, and metadata. Operates under partial observability.

**Update** — Revises internal state: updates memory, refines world models, and compresses long-term experience.

**Reason** — Plans and prioritizes conditioned on perceptions, goals, and reward models. Weighs uncertainty and long-term utility.

**Execute** — Enacts decisions through tool invocations, control commands, and environment interactions. Closes the cognition-action loop.

**Reflect** — Evaluates outcomes vs. expectations. Transforms successes and failures into learning signals that inform what should evolve next.

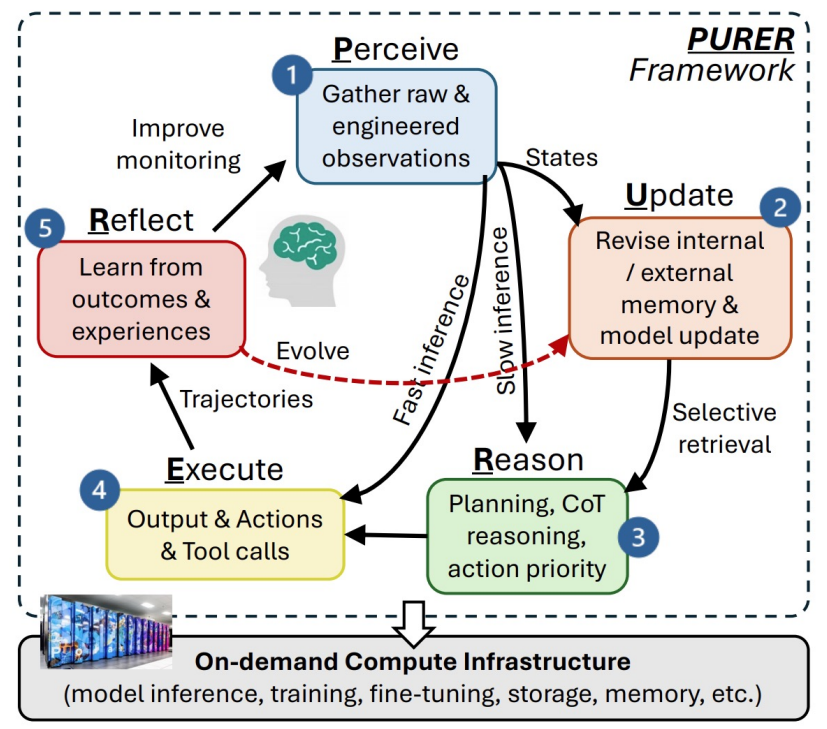



**Figure 1.** The PURER self-evolving framework: a closed loop where operational experience continuously drives system adaptation.

# Core Components & Deep Dive

**Model** — Core decision-making substrate: dense LLMs or MoEs. Provides abstraction, reasoning, and synthesis capabilities.

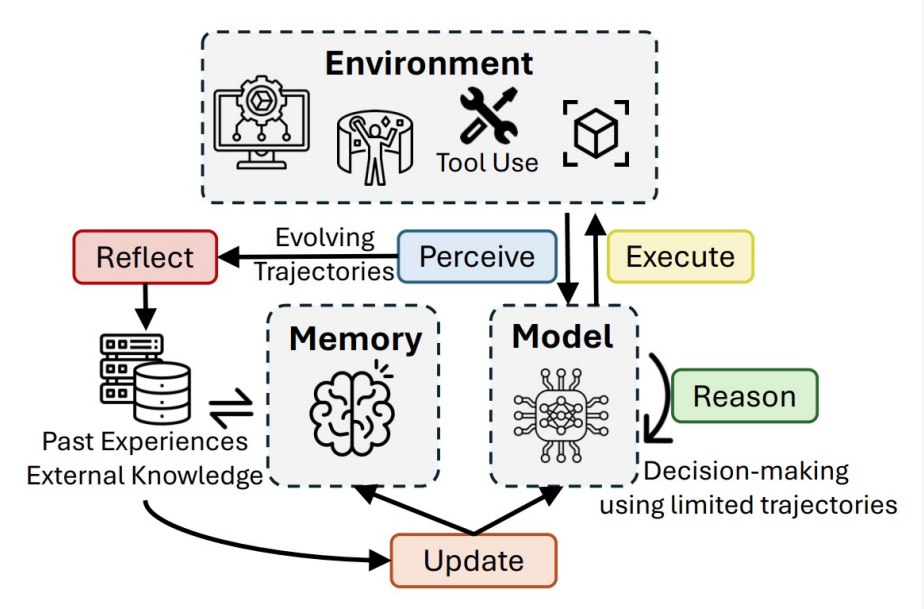
**Memory** — Internal (latent states, short-term context) and external (RAG, knowledge bases). Selectively retrieved and updated by the model.

**Environment** — External world: tools, APIs, simulators, actuators. Execution closes the loop between cognition and action.

---

**Trajectory / Experience Database** — Structured repository of past interactions, outcomes, and rewards. Unlike ephemeral context windows, this persists learning signals across sessions.

**Design Principle:** PURER co-evolves models, memory, context, and tools through closed-loop environment interaction. Failure, environmental change, and partial observability are first-class adaptation signals.



# Self-Evolution via Reinforcement Learning

**Key Innovation:** RL decides **(1) which component should evolve** and **(2) how to evolve** — enabling meta-level learning across cognition, memory, and operational substrate.

### 1. External Memory

Update or reorganize memory stores based on observed failure patterns and retrieval inefficiencies.

### 2. Perception Strategy

Modify context construction and perception strategies to better capture relevant signals under partial observability.

### 3. Model Parameters

Fine-tune or replace partial model weights when durable learning is needed, despite higher computational cost.

### 4. Environment & Tools

Evolve the environment by introducing new tools, abstractions, or execution pathways to expand capability.

## Insight: Adaptation as a Control Decision

PURER uses **reward-driven control** to decide when to reason, reflect, or update, and **where learning should reside** — context, memory, or model weights — to achieve long-horizon robustness.

This reframes adaptation as a **runtime decision problem** across timescales, requiring new abstractions to reason about learning cost, benefit, and risk.

Unlike static systems, PURER supports sustained robustness in long-running, safety-critical, and rapidly changing environments.

# Validation & Challenges

Live Testbed Validation and Open Research Problems

# Validation & System-Level Challenges

## Live Traffic Testbed (Top 500 Supercomputer)

Validation on a Top 500 supercomputer with **NVIDIA H200 and Blackwell GPUs**, encompassing realistic AI training, inference, and HPC traffic at national scale over a 1,200 Gbps network.

### Perceive & Update

Optical tap (Arista) on /16 CIDR subnet mirrors operational and disrupted inference traffic to Zeek observability cluster in real time.

### Reason & Execute

Replay of AI-system disruptions in live traffic. Targets (faulty GPU, backdoored model) integrated to collect GPU driver logs.

### Reflect & Optimize

Objective function balances information entropy $H(U|M_x)$ and risk $R(M_x)$. Near-miss signals inferred from internal traces.

## System-Level Challenges & Research Agenda

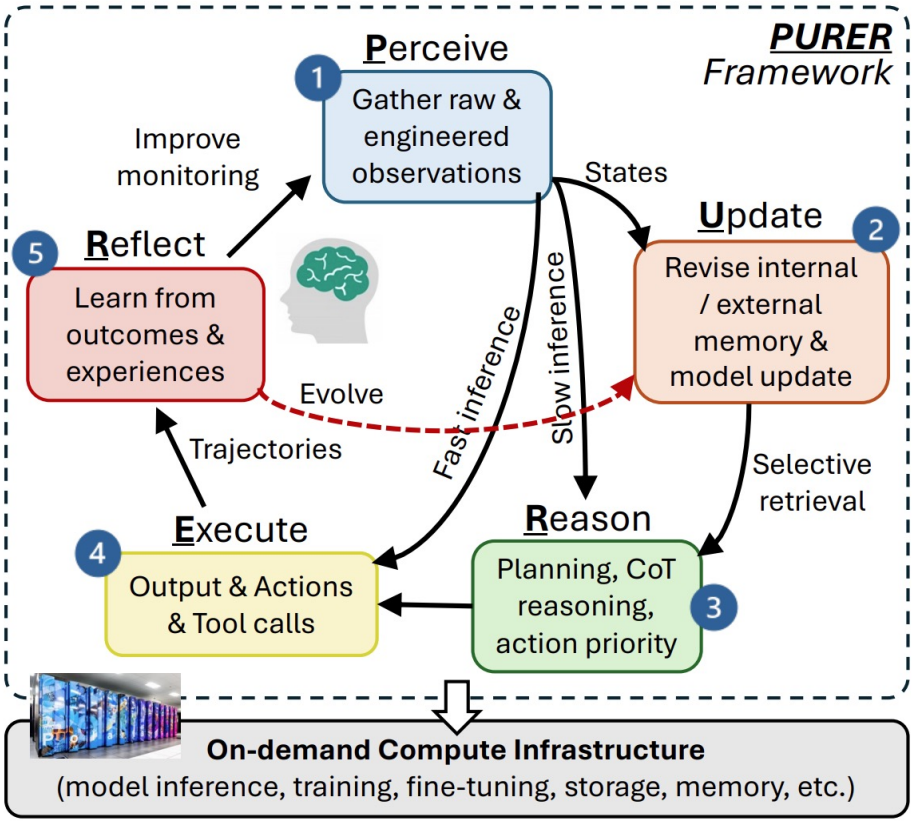
**1. Adaptation as a Control Decision** — PURER uses reward-driven control to decide when to reason, reflect, or update, and where learning should reside (context, memory, or model weights) for long-horizon robustness.

**2. Experience Transfer Across Agents** — Enable new agents to recover lost capabilities by transferring knowledge and failure signals from prior agents, breaking local optima.

**3. Novel System-Level Abstractions** — Design runtime mechanisms that jointly manage memory, context, model updates, and resources while ensuring stability, observability, and correctness under continuous evolution.

**4. Specific risks:** destabilizing feedback loops, reward hacking, unintended behavioral drift, and maintaining correctness under non-stationarity.

# Thank You!


PURER
Framework
Perceive
1
Gather raw & engineered observations
Improve monitoring
States
Update
2
Revise internal / external memory & model update
Reflect
5
Learn from outcomes & experiences
Evolve
Trajectories
Fast inference
Slow inference
Selective retrieval
Execute
4
Output & Actions & Tool calls
Reason
Planning, CoT reasoning, action priority
3
On-demand Compute Infrastructure
(model inference, training, fine-tuning, storage, memory, etc.)