

Online Twins for Continuous Agents: An Architectural Analysis of TwinTrace and the Surrounding State-of-the-Art

The development of embodied artificial intelligence, robotics, and autonomous control systems has historically operated within a bifurcated paradigm: extensive offline training over episodic datasets followed by static, online deployment. However, the operational reality of nonstationary environments, degrading hardware, and drifting physical plants necessitates a fundamental shift toward continuous, always-on learning architectures. Edge devices require the ability to adapt in real-time without the computational luxury of massive replay buffers or backpropagation through time.

The "TwinTrace" framework, specifically the dtg/v3 iteration, embodies this architectural transition. Rather than optimizing for incremental benchmark superiority on offline reinforcement learning tasks, TwinTrace introduces a highly specific product shape: a unified, replay-free application programming interface (API) that functions simultaneously as a dynamics world model, an online system identifier, a short-horizon forecaster, a discrete policy imitator, and a residual anomaly detector¹.

This report presents an exhaustive deconstruction of the TwinTrace architecture, isolating its five semantic bindings and analyzing them against the broader landscape of machine learning and control theory. By mapping these bindings to state-of-the-art frameworks—spanning test-time adaptation, recursive least squares (RLS), dynamic mode approximation, reservoir computing, and Bayesian changepoint detection—this analysis defines the current boundaries of continuous online twins.

The TwinTrace Architecture: Reconciling Plasticity and Consolidation

At the core of the TwinTrace framework is a dual-phase online learning lifecycle designed to solve the plasticity-stability dilemma without relying on episodic replay banks. The architecture establishes a continuous "Acquire" phase that transitions into a frozen "Deploy" phase.

Phase A and Phase B Dynamics

The cognitive load of the agent is divided between two distinct neural pathways that co-adapt to a live observation stream:

- **Phase A (The Fast Path):** A highly plastic, synchronous neural network (termed a fast Velvet) that tracks the immediate environmental dynamics or teacher signal. It processes the full computational chain during every tick, serving as an aggressive, short-term adapter.
- **Phase B (The Consolidator):** A persistent, volumetric neural network placed on a spatial

3D grid. Unlike traditional recurrent neural networks that unroll over time via backpropagation through time (BPTT), Phase B utilizes a "Step*" credit assignment where each tick advances activity through a single layer hop (stem, mid, cams) within the network pipe¹.

The architecture orchestrates knowledge transfer from the volatile Phase A to the stable Phase B through a mechanism called TeacherMix, which blends the ground-truth teacher signal with Phase A's predictions based on real-time error metrics.

Mathematical Formulation of the Acquire Tick

During continuous operation, the system receives an observation x_t , a target y_t^* , and a quality gate $q \in \{0, 1\}$. The framework calculates the pre-update mean squared error (MSE) for both phases simultaneously:

$$\text{aMse} = \text{mse}(A_\theta(x_t), y_t^*), \quad \text{bMse} = \text{mse}(B_\phi(x_t), y_t^*)$$

If the quality gate $q > 0$, Phase A undergoes a single-step weight update. To transfer this immediate learning to the consolidator without retaining a historical data buffer, Phase B is presented with a blended target y_B . This target incorporates Phase A's post-update prediction, $\hat{a}_1$, scaled by a dynamically calculated trust factor μ :

$$\mu = \text{TeacherMix} \cdot \exp\left(-\frac{\text{aMse}}{\text{TrustScale}}\right)$$

$$y_B = (1 - \mu)y_t^* + \mu\hat{a}_1$$

Phase B then executes its localized Step* observation on y_B . Upon reaching the "Freeze" state, Phase A is systematically destroyed, and the system relies exclusively on Phase B for forward inference (Model.Predict)¹.

Component / Parameter	Architectural Function
TwinTrace-R	The overarching lifecycle managing the Acquire-to-Freeze transition, focusing on recursive/revision-oriented design.
Phase A (A_θ)	Fast Welvet path; tracks the teacher signal with high plasticity. Destroyed upon Freeze.

Phase B (B_ϕ)	Volumetric consolidator operating on a 3x3x3 spatial grid. Survives Freeze to become the deployable model.
Step* Credit	Velvet training mode advancing activity one layer hop per tick, eliminating the need for BPTT unrolling.
TeacherMix	A dynamic blending scalar $\mu \in [0, 1]$ that dictates how much of Phase A's prediction is distilled into Phase B's target.
Feedback Loop	Recirculation configuration where the previous Phase B output u_{t-1} is concatenated into the next input $[x_t; u_{t-1}]$.
Quality Gate (q)	A binary switch determining if the current observation should trigger a weight update ($q = 1$) or merely register a residual metric ($q = 0$).

The operational validation of this architecture rests entirely on its ability to support five distinct semantic bindings through this exact mathematical pathway. The following sections isolate each binding and compare it to the nearest equivalent methodologies in contemporary research.

The w1 Binding: Next-State Dynamics and Test-Time Adaptation

The w1 binding instantiates a continuous next-state dynamics twin. Given a state vector and a control action $x_t = [p_t, v_t, u_t]$, Phase B is trained to predict the subsequent state $y_t^* = [p_{t+1}, v_{t+1}]$. Crucially, the system does not utilize a neural policy to generate actions. Instead, a classical Proportional-Derivative (PD) controller acts upon the *forecasted* state. The controller iteratively queries the twin ("What happens if I apply this control input?") and utilizes

the predicted state to settle the physical plant in a closed-loop fashion¹.

Sandwich-Residuals and Parameter-Efficient Adaptation

The necessity for world models to adapt to shifting test-time distributions is well documented. The most prominent contemporary equivalent to the w1 adaptation strategy is the Sandwich-Residuals framework, designed to perform parameter-efficient test-time adaptation on pretrained latent world models like AdaJEPA and DINO-WM².

Where TwinTrace achieves adaptation by distilling a plastic Phase A into a permanent Phase B, Sandwich-Residuals takes a diametrically opposed architectural stance. It operates under the assumption that the core pretrained world model should remain completely frozen to prevent catastrophic forgetting. Instead, it introduces highly lightweight, learnable residual multilayer perceptrons (MLPs) before and after the frozen predictor module².

In the Sandwich-Residuals architecture, an action embedding is corrected based on proprioceptive representations before being fed to the frozen predictor:

$$\tilde{z}_t^a = z_t^a + r_{\phi}^a([z_t^a; z_t^p])$$

Subsequently, the output from the predictor is corrected by a second residual module to compensate for any remaining distribution shift:

$$\tilde{z}_{t+1}^v = \hat{z}_{t+1}^v + r_{\phi, out}^v(\hat{z}_{t+1}^v)$$

Both visual and action residuals are optimized online during the plan-execute cycle using a simple Adam update based on self-supervised prediction error, modifying merely 1% to 3.1% of the total network parameters². This allows a system operating in complex, high-dimensional visual navigation tasks to recover performance under severe dynamics shifts (e.g., 50x damping or visual blur) without corrupting the underlying pretrained foundation².

Reflexive and Self-Correcting World Models

Beyond simple prediction, integrating world models directly into the control loop is explored in Reflexive World Models (RWM) and self-correcting Vision-and-Language Navigation (VLN-CE) frameworks⁵. RWM explicitly uses the predictions of the world model as implicit reference trajectories, bridging learned predictive models with classical control theory to facilitate rapid adaptation⁵.

Similarly, self-correcting world models extract internal feedback signals from the agent's foresight state. By mapping a provisional action to an anticipated latent state, and subsequently decoding that state into a visual representation, the model generates an internal error signal⁶. This allows the agent to iteratively refine its action distribution before executing a command in the physical environment. TwinTrace's w1 binding achieves this exact mechanism by feeding the twin's forecast into a PD controller, demonstrating that the boundary between classical deterministic control and learned neural foresight is rapidly dissolving.

Feature /	TwinTrace-R (w1	Sandwich-Residua	Self-Correcting
-----------	-----------------	------------------	-----------------

Architecture	Binding)	Is	VLN-CE
Adaptation Mechanism	TeacherMix distillation from plastic Phase A to consolidating Phase B.	Lightweight residual MLPs wrapping a completely frozen core model.	Internal feedback signals modifying the foresight state.
Parameter Efficiency	Low compute per tick (Step* hops), modifies Phase B volumetric grid.	Extremely high; modifies ~1-3% of total network parameters.	Modifies latent space dynamics via transition models.
Control Integration	Classical PD controller acting on neural forecast.	Model Predictive Control (MPC) operating on corrected latent states.	Iterative plan refinement based on decoded visual foresight.
Target Domain	Low-dimensional continuous control streams.	High-dimensional visual navigation and 3D robotic manipulation.	Vision-and-Language Navigation in complex environments.

The i1 Binding: Online System Identification and Adaptive Filtering

The i1 binding transitions the API into a pure adaptive filter designed for online system identification (SysID). Operating without feedback recirculation, it observes a set of features $\phi(x)$ to track a drifting, nonlinear plant gain $g(t)$ embedded in heavy noise. Phase B perpetually refits this nonlinear map online, adopting the exact operational role traditionally occupied by Recursive Least Squares (RLS) or Extended Kalman Filters¹.

Dynamic Mode Adaptive Control (DMAC)

In contemporary classical control literature, the integration of online system identification with controller synthesis is formalized in Dynamic Mode Adaptive Control (DMAC)⁹. DMAC recursively estimates a local linear approximation of nonlinear system dynamics and synthesizes an optimal control input (such as an LQR formulation) to achieve reference tracking

without requiring a prior global model⁹.

DMAC relies heavily on a matrix-valued extension of the classical RLS cost function to estimate the system matrices $A_\star$ and $B_\star$ over a local operational window. The RLS update in DMAC is mathematically equivalent to a recursive implementation of dynamic mode decomposition (DMD)⁹. While TwinTrace uses its volumetric neural grid to map the drifting plant, DMAC relies on rigorous linear algebraic updates. The application of RLS and its variant, Recursive Weighted Least Squares (RWLS), is prevalent in physical systems lacking accurate white-box models, such as bidirectional DC motors exhibiting complex Wiener model structures⁸. RWLS processes streaming data recursively, reducing estimation errors and eliminating the need for extensive pretraining, much like TwinTrace's continuous "Acquire" loop⁸.

Weak-SINDy and the Preservation of Interpretability

For identifying the governing equations of nonlinear dynamics from streaming data, Sparse Identification of Nonlinear Dynamics (SINDy) and its streaming variant, Weak-SINDy, represent the pinnacle of current research¹¹.

Traditional SINDy is highly susceptible to measurement noise because it relies on estimating derivatives directly from data. Weak-SINDy circumvents this by rewriting the dynamics in an integral, or weak, formulation, acting as a natural low-pass filter over the data¹². When deployed in adaptive control, Streaming Weak-SINDy addresses the critical issue of parameter convergence. Classical adaptive controllers usually require a stringent Persistent Excitation (PE) condition. Weak-SINDy utilizes a Memory Regressor Extension (MRE) coupled with directional forgetting to ensure parameter convergence under much weaker excitation conditions¹⁴.

The system dynamics are modeled over a known library of nonlinear basis functions $Y(x)$:

$$\dot{x} = Y(x)\theta + g(x)u + \epsilon(x)$$

The adaptive update law utilizes a Clarke sub-differential to apply a sparsity-promoting ℓ_1 penalty, forcing the system to isolate the exact, sparse set of physical equations governing the drift¹⁴.

TwinTrace's i1 binding and Streaming Weak-SINDy share the exact same objective: online tracking of nonstationary plants. However, their philosophical approaches represent a fundamental divergence in modern engineering. TwinTrace embraces the black-box nature of neural consolidators, prioritizing integration with policy and forecasting bindings. Conversely, Weak-SINDy prioritizes symbolic, physical interpretability and rigorous Lyapunov stability guarantees, making it the superior choice for safety-critical aerospace and fluid dynamics applications where black-box behavior is unacceptable¹².

Methodology	Core Mechanism	Strengths	Limitations
TwinTrace (i1)	Volumetric neural	Shares API with	Lacks mathematical

	consolidation (Phase B) replacing RLS.	world models/policies; highly versatile.	interpretability and formal stability bounds.
DMAC	Recursive implementation of Dynamic Mode Decomposition.	Integrates seamlessly with LQR control; robust linear approximations.	Inherently limited by local linear assumptions in highly nonlinear regimes.
Weak-SINDy	Integral formulation of sparse regression with ℓ_1 penalties.	Yields human-readable physical equations; highly robust to noise.	Computationally intensive library evaluations; requires partial knowledge of basis functions.

The f1 Binding: Short-Horizon Feedback Forecasting

The f1 binding executes short-horizon, multi-channel forecasting on telemetry streams. By enabling the "Feedback" configuration, the system feeds its previous prediction u_{t-1} back into the input pipeline alongside the current state x_t . This allows the volumetric Phase B to perform a recursive k -step rollout (e.g., $k = 4$) without utilizing transformer-based sequence-to-sequence architectures or Backpropagation Through Time (BPTT)¹.

Reservoir Computing, ESNs, and FORCE Learning

The pursuit of continuous, online temporal processing without the explosive computational overhead of BPTT is heavily anchored in the field of Reservoir Computing, specifically via Echo State Networks (ESNs) trained with FORCE (First-Order Reduced and Controlled Error) learning¹⁷.

In a standard ESN, a high-dimensional recurrent neural network (the reservoir) is randomly initialized and its internal weights remain entirely fixed. The input signal excites the reservoir, creating a rich, non-linear projection of the temporal data. Only the final linear readout weights are trained²¹. This architecture naturally bypasses the vanishing gradient problems that plague standard recurrent networks. However, to produce complex, self-sustaining temporal patterns, the network's output must be fed back into the reservoir—a structural choice identical to TwinTrace's Feedback loop²¹.

When the reservoir operates in a chaotic, spontaneous regime, standard gradient descent is

completely ineffective. To harness this chaos, Sussillo and Abbott introduced FORCE learning¹⁸. FORCE applies an online Recursive Least Squares (RLS) algorithm to rapidly modify the readout weights, effectively forcing the chaotic activity into coherent, target-driven temporal signals¹⁷.

The FORCE update rule for the output weight matrix $W(t)$ is executed continuously at each time step:

$$W(t) = W(t - \Delta t) - e(t)P(t)x(t)$$

Where $e(t)$ represents the error vector, $x(t)$ is the network activity, and $P(t)$ is the inverse correlation matrix updated online¹⁸. This mechanism has proven highly effective in modeling severe non-linear dynamics, such as the chaotic time series generated by the Mackey-Glass system and the Lorenz attractor¹⁷.

Expanding to Full-FORCE and RFLO Learning

Advancements in this domain include full-FORCE, which improves synchronization by imposing a target-based method on recurrent networks, and Random Feedback Local Online (RFLO) learning¹⁷. RFLO specifically addresses the biological implausibility of traditional BPTT by providing a localized, continuous weight update mechanism that does not require non-local information about all recurrent weights in the network²³.

TwinTrace's f1 binding and the broader reservoir computing ecosystem share a fundamental design philosophy: the rejection of deep temporal unrolling in favor of localized, continuous updates on a streaming sequence. TwinTrace's Phase B volumetric grid operates conceptually as a structured reservoir, consolidating the highly plastic adaptations of Phase A. The elimination of BPTT in favor of "Step*" pipe hops in TwinTrace is a modern, computationally constrained realization of the principles established by ESNs and FORCE learning decades prior¹.

The p1 Binding: Discrete Policy and Action-Score Imitation

The p1 binding transitions the framework from dynamics and forecasting into direct action selection. Given a continuous spatial observation vector, Phase B outputs scores for a small discrete set of actions (e.g., North, East, South, West, Stay). The teacher signal is a greedy expert score vector, and Phase B optimizes an MSE loss on these scores, utilizing an argmax function for policy execution¹.

Behavioral Cloning in Low-Dimensional Spaces

The approach taken by TwinTrace operates as a continuous, online variant of behavioral cloning. In traditional offline reinforcement learning, behavioral cloning attempts to map observations directly to expert actions. Recent literature highlights advanced methodologies like Diffusion Policies and Flow Matching Policies, which model complex, multimodal behaviors

by learning continuous action score matching over time²⁶.

However, TwinTrace explicitly acknowledges a critical boundary condition for its architecture: MSE-based score matching on a volumetric twin is highly efficient for micro-action spaces (4 to 32 discrete choices) but fails catastrophically when scaled to massive categorical distributions¹. The source data explicitly notes that a 1,200-way text vocabulary stress test resulted in a plateauing failure mode¹. This aligns precisely with the broader consensus in machine learning: modeling immense discrete spaces requires softmax cross-entropy distributions, massive transformer-based context windows, and attention mechanisms. Such heavyweight architectures are inherently incompatible with the low-latency, recursive pipe-hop design that allows TwinTrace to execute continuously on edge hardware without replay buffers. By constraining the policy to a small discrete set over a continuous latent space, the p1 binding successfully maps decision-making onto the same API used for physical dynamics.

The a1 Binding: Gated Residual Anomaly Detection

The a1 binding demonstrates how world models can dual-function as health monitors. The twin learns the nominal dynamics of a physical plant. When a mechanical fault occurs, the residual error between the twin's forecast and the physical reality spikes, triggering an alarm based on an exponential moving average (EMA) threshold. Crucially, the system employs a quality gate:

when $q = 0$ (fault detected), weight updates are entirely suspended¹. This prevents the twin from adapting to the broken physics, ensuring it remains a model of a *healthy* world.

Changepoint-Aware Dynamics Learning: ARCADE

Distinguishing between transient environmental noise and permanent structural faults is a foundational challenge in robotics. If an adaptive controller adjusts to a transient disturbance, it degrades its own performance; if it fails to adjust to a permanent fault, catastrophic failure ensues.

The most mathematically rigorous analogue to TwinTrace's q -gate is the ARCADE framework (Adaptive Robot Control with Online Changepoint-Aware Bayesian Dynamics Learning)²⁷.

ARCADE decouples offline representation learning from online Bayesian adaptation, parameterizing the shift in system dynamics via a latent-variable model:

$$f_{\Delta}(x_k, u_k) = \theta_k \phi(x_k, u_k)$$

Here, ϕ is a nonlinear encoder and θ_k is a linear decoder updated online²⁷.

To manage abrupt faults, ARCADE maintains a posterior probability over latent changepoint variables $c_k \in \{0, 1\}$. The joint score for any hypothesis captures both data fit and model prior. The cumulative log-likelihood of a hypothesis h is formulated as:

$$\mathcal{L}_{k,h} = \mathcal{L}_{k-1,h} + \log p(\delta_k \mid z_k, c_k) + \log p(c_k \mid c_{1:k-1}, x_{1:k}, u_{1:k})$$

This equation balances how well the hypothesis explains the observed residual against the plausibility of the changepoint event itself²⁷. When a changepoint is detected, ARCADE rapidly resets its Bayesian priors, preventing the model from overcommitting to stale posteriors and capping the growth of predictive variance²⁷.

TwinTrace's binary q -gate is effectively a heuristic, hard-coded manifestation of ARCADE's probabilistic inference. While TwinTrace relies on a simple standard deviation threshold to halt learning, ARCADE formalizes this into a Bayesian framework that gracefully manages adaptation rates across shifting regimes. Both systems arrive at the same operational conclusion: continuous learning models must possess active self-defense mechanisms to prevent catastrophic adaptation to faulty hardware.

Gated Residual Networks in Industrial Systems

In the broader context of industrial digital twins, the utilization of Gated Residual Networks (GRNs) for anomaly detection is highly prevalent. Systems monitoring power grids, wind turbines, and diesel generators frequently employ deep learning convolutional models interwoven with GRNs to process time and frequency-domain vibration features²⁹. These networks dynamically suppress irrelevant noise and amplify anomalous feature variations, enabling sub-millisecond fault detection³². TwinTrace's a1 binding effectively distills the core concept of industrial GRNs into an embodied, onboard architecture, tightly integrating the anomaly detector directly into the world model's predictive loop.

Framework	Anomaly Detection Mechanism	Response to Fault	Theoretical Foundation
TwinTrace (a1)	Residual error vs. $\text{EMA} + 3\sigma$ threshold.	Hard binary quality gate ($q = 0$) suspends all learning.	Heuristic thresholding on world model divergence.
ARCADE	Cumulative log-likelihood of latent changepoint variables.	Resets Bayesian priors to prevent stale posterior overcommitment.	Bayesian changepoint inference and bounded predictive variance.
Industrial GRNs	Time/frequency feature extraction via Gated Residual Networks.	Generates structured outputs for digital twin supervisory	Deep convolutional feature fusion and attention mechanisms.

		platforms.	
--	--	------------	--

Macro-Integration: The Unified Digital Twin Ecosystem

TwinTrace claims significant novelty in its ability to condense five operational roles into a single API¹. However, when viewed through the lens of macro-engineering, the concept of a digital twin performing simultaneous, multi-faceted roles is well established. Digital twins have become the foundational architecture for managing complex infrastructure, supported heavily by institutions like Monash University and CSIRO Data61³⁴. These frameworks span multiple domains:

- Railway and Fleet Dynamics:** Cloud-based digital twins integrate physics-based models with machine learning to predict curve negotiation dynamics in real-time, executing both operational control and structural health anomaly detection simultaneously³⁹.
- Smart Energy Grids:** Digital twins in IoT-driven energy systems utilize hybrid models—combining data-driven AI with discrete-time transient state estimators—to balance supply loads and classify physical faults within microseconds³².
- Industrial Manufacturing:** Complex casting and pouring operations utilize multimodal data streams (video, images, sensors) processed through Mixture-of-Experts (MoE) frameworks. These systems execute local event detection, global environment evaluation, and dynamic flow characterization concurrently⁴¹.

The true distinction of TwinTrace does not lie in the *convergence* of these tasks, but rather in the *scale and location* of the computation. Traditional industrial digital twins rely on massive cloud-edge orchestration, federated learning protocols, and heavy back-end server clusters³¹. TwinTrace compresses this multi-role paradigm into a micro-scale, on-device runtime for continuous agents. By executing the entire Acquire-Freeze-Deploy lifecycle locally, without episodic storage, TwinTrace provides an elegant, resource-constrained alternative to facility-scale infrastructure, tailored specifically for autonomous robotics.

Synthesis and Emergent Trajectories

Analyzing the TwinTrace architecture alongside Sandwich-Residuals, DMAC, FORCE learning, and ARCADE reveals three profound, second-order trajectories redefining the landscape of autonomous control.

1. The Dissolution of the Classical vs. Deep Learning Boundary

For decades, classical control (System ID, LQR, PD) and deep learning operated as mutually exclusive domains. Classical methods offered rigorous stability guarantees but failed in highly dimensional, non-linear environments; deep learning mastered complex environments but remained notoriously brittle and mathematically opaque. The current state-of-the-art demonstrates a violent, necessary fusion of these disciplines. TwinTrace’s w1 binding places a classical PD controller directly atop a neural forecast. DMAC

utilizes recursive linear algebra to approximate spaces for LQR synthesis⁹. Weak-SINDy uses neural gradient descent mechanisms to extract classical differential equations¹⁴. The industry is universally converging on a hybrid paradigm: utilizing deep learning exclusively for high-dimensional perception and forecasting, while relying on classical deterministic controllers to synthesize the final actuation command.

2. The Obsolescence of the Replay Buffer at the Edge

Traditional offline reinforcement learning (e.g., PPO, SAC) fundamentally relies on massive episodic replay buffers to randomize data sampling, breaking temporal correlations to stabilize gradient descent. For continuous, always-on edge agents, storing and continually retraining on infinite data streams is impossible due to memory, power, and compute constraints.

The architectural choices across the surveyed frameworks indicate that the replay buffer is becoming obsolete for edge deployment. TwinTrace uses continuous TeacherMix distillation; FORCE and RFLO learning utilize recursive least squares on sequential data; ARCADE utilizes recursive Bayesian updates¹. The future of edge AI requires systems that act as selective, real-time filters—consolidating vital environmental dynamics into permanent network weights

while instantly discarding the raw sequential data, maintaining a strict $O(1)$ memory footprint regardless of deployment lifespan.

3. Anomaly Detection as an Emergent Property of World Modeling

Historically, anomaly detection required deploying dedicated, parallel architectures—such as autoencoders or isolation forests—alongside the primary control system¹. The data indicates a structural shift where anomaly detection is no longer a separate task, but an emergent property of rigorous world modeling.

When an architecture like TwinTrace or ARCADE trains its predictive twin strictly on nominal data, any divergence in the physical plant (hardware degradation, environmental shifts) instantly manifests as a spike in predictive residual error¹. The world model naturally acts as the health monitor. By gating the learning process during these spikes, the system prevents the twin from corrupting its understanding of a healthy world. This self-reinforcing loop drastically reduces the total parameter count required on autonomous agents, collapsing the roles of controller and diagnostic monitor into a singular, unified neural volume.

The TwinTrace framework, while perhaps overstating its novelty against the backdrop of macro-scale digital twins, successfully captures and operationalizes these three emergent trajectories, offering a highly robust blueprint for the next generation of continuous embodied agents.

Works cited

1. ASSESSMENT.md
2. Parameter-Efficient Test-time Adaptation of World Models - arXiv, <https://arxiv.org/html/2609.21740v1>
3. Sandwich-Residuals: Parameter-Efficient Test-time Adaptation of,

- <https://arxiv.org/pdf/2609.21740>
4. [2606.32026] AdaJEPa: An Adaptive Latent World Model - arXiv, <https://arxiv.org/abs/2606.32026>
 5. World Models as Reference Trajectories for Rapid Motor ... - arXiv, <https://arxiv.org/html/2505.15589v1>
 6. SC²-WM: A Self-Correcting World Model with Closed-Loop ... - arXiv, <https://arxiv.org/pdf/2608.07548>
 7. World Models as Reference Trajectories for Rapid Motor Adaptation, <https://arxiv.org/abs/2505.15589>
 8. Wiener Model Structure Estimation of DC Motor Through Online, https://www.researchgate.net/publication/384559418_Wiener_Model_Structure_Estimation_of_DC_Motor_Through_Online_Neural_Network_System_Identification
 9. Model-Free Dynamic Mode Adaptive Control for Data-Driven ... - arXiv, <https://arxiv.org/pdf/2604.26682>
 10. Model-Free Dynamic Mode Adaptive Control for Data-Driven ... - arXiv, <https://arxiv.org/html/2604.26682>
 11. On sparsity and directional forgetting in adaptive control - arXiv, <https://arxiv.org/html/2609.23932v1>
 12. Sparse Identification of Nonlinear Dynamics with Control (SINDYc, https://www.researchgate.net/publication/312243811_Sparse_Identification_of_Nonlinear_Dynamics_with_Control_SINDYcSLB_acknowledges_support_from_the_US_Air_Force_Center_of_Excellence_on_Nature_Inspired_Flight_Technologies_and_Ideas_FA9550-14-1-0398_JLP
 13. Online Dynamic Mode Decomposition for Time-Varying Systems, <https://epubs.siam.org/doi/abs/10.1137/18M1192329>
 14. Adaptive Control with Sparse Identification of Nonlinear Dynamics, <https://arxiv.org/html/2604.06338v3>
 15. Adaptive Control with Sparse Identification of Nonlinear Dynamics, <https://arxiv.org/pdf/2604.06338>
 16. On sparsity and directional forgetting in adaptive control - arXiv, <https://arxiv.org/pdf/2609.23932>
 17. Composite FORCE learning of chaotic echo state networks for time, https://www.researchgate.net/publication/361807441_Composite_FORCE_learning_of_chaotic_echo_state_networks_for_time-series_prediction
 18. Online supervised learning of temporal patterns in biological neural, <https://www.pnas.org/doi/10.1073/pnas.2521560123>
 19. Brain-Inspired Recurrent Neural Networks for Biomedical Applications, <https://escholarship.org/content/qt0xc25962/qt0xc25962.pdf?v=lg>
 20. Learning long temporal sequences in spiking networks by, https://ruor.uottawa.ca/bitstream/10393/39960/1/Vincent-Lamarre_Philippe_2019_thesis.pdf
 21. Toward an Integration of Deep Learning and Neuroscience - PMC, <https://pmc.ncbi.nlm.nih.gov/articles/PMC5021692/>
 22. Machine Learning for Wireless Networks with Artificial Intelligence, <https://infinite-vigilant.github.io/AI.pdf>

23. Local online learning in recurrent networks with random feedback,
<https://pmc.ncbi.nlm.nih.gov/articles/PMC6561704/>
24. Local online learning in recurrent networks with random feedback,
<https://elifesciences.org/articles/43299>
25. Time-Series Forecasting and Sequence Learning Using Memristor,
<https://arxiv.org/pdf/2405.13347?>
26. Adversarial Dual On-Policy Distillation from Expressive Flow-based,
<https://arxiv.org/html/2605.27095v1>
27. ARCADE: Adaptive Robot Control with Online Changepoint-Aware,
<https://arxiv.org/html/2512.14331v1>
28. Avirup Das - Research Explorer - The University of Manchester,
<https://research.manchester.ac.uk/en/persons/avirup-das/>
29. Data Processing Pipelines, Predictive Models, and Intelligent Control,
<https://www.mdpi.com/1996-1073/19/18/4244>
30. A Dual-Stream Gated Attention Network for Fault Diagnosis of Drill,
https://www.researchgate.net/publication/397957711_A_Dual-Stream_Gated_Attention_Network_for_Fault_Diagnosis_of_Drill_Bit_Using_time_and_Frequency_Domain-based_Vibration_Features
31. Design and Implementation of Digital Twin Diesel Generator Systems,
https://www.researchgate.net/publication/373704802_Design_and_Implementation_of_Digital_Twin_Diesel_Generator_Systems
32. Digital Twins for IoT-Driven Energy Systems: A Survey - ResearchGate,
https://www.researchgate.net/publication/386151227_Digital_Twins_for_IoT-Driven_Energy_Systems_A_Survey
33. Enhancing Intelligent Fault Detection and Classification in Power,
<https://journals.riverpublishers.com/index.php/DGA EJ/article/download/30665/24085>
34. Mixed Reality Outperforms Virtual Reality for Remote Error,
https://www.researchgate.net/publication/391681632_Mixed_Reality_Outperforms_Virtual_Reality_for_Remote_Error_Resolution_in_Pick-and-Place_Tasks
35. Digital Twins in Electric Vehicles Review | PDF - Scribd,
<https://www.scribd.com/document/896990155/1-s2-O-S209575642400093X-main>
36. Manufacturers' Monthly November 2022 by Prime Group - Issuu,
https://issuu.com/primecreativemedia-2016/docs/mm1122_lr
37. Events - Robotics Australia Group, <https://www.roboausnet.com.au/events>
38. A Robotics Roadmap for Australia 2022 - ANU Open research,
<https://openresearch-repository.anu.edu.au/server/api/core/bitstreams/995f7101-b742-449c-88c3-afd3abb1ff50/content>
39. A digital twin-driven machine learning framework for real-time,
https://www.researchgate.net/publication/415381237_A_digital_twin-driven_machine_learning_framework_for_real-time_prediction_of_railway_vehicle_curve_negotiation_dynamics
40. A systematic literature review on the use of digital twin technology in,
<https://www.frontiersin.org/journals/smart-grids/articles/10.3389/frsgr.2026.18050>

[35/pdf](#)

41. Multimodal Industrial Scene Characterisation for Pouring Process,
<https://www.mdpi.com/2076-3417/16/7/3430>