International Journal of Current Science Research and Review
ISSN: 2581-8341
Volume 09 Issue 10 October 2026
DOI: 10.47191/ijcsrr/V9-i10-27, Impact Factor: 8.048
IJCSRR @ 2026
www.ijcsrr.org

Mevronic Neural Networks for Explicit Reliability Regime Control. A Semantically Anchored Architecture with Analytical Properties and an Evaluation Protocol

Vasil Metodiev[1], Iliyan Vasilev[2]
[1,2] University of Chemical Technology and Metallurgy, Sofia, Bulgaria

ABSTRACT: Autonomous systems face a difficult problem when the information they rely on becomes unreliable: they must decide not only what to predict, but also how much confidence to place in those predictions. This paper introduces Mevronic Neural Networks (MNNs), a design approach that connects a neural representation network with a persistent state describing its operating reliability. Rather than proposing a new computational architecture, the focus is on making the relationship between incoming evidence, stored information, network behavior, and operational decisions more explicit. The proposed model uses three main state variables representing estimated prediction correctness, accumulated anomaly evidence, and sensor quality. These variables influence the system's alertness and determine how much adaptation is allowed. Their meanings are defined in advance to make the mechanism easier to examine, although the estimates used to update them may still contain errors. The paper explains how the system collects evidence, maintains its internal state, and modifies network features within specified limits. Then it switches to alternative responses when conditions require it. The mathematical results confirm that the state variables stay within their defined boundaries and show how earlier observations influence subsequent updates. They also provide an exact expression for the time needed to reach a given threshold when the incoming evidence remains constant. A numerical example helps illustrate a practical compromise: smoothing reduces rapid fluctuations in the system's response, but it can also delay reactions to changes in the environment. These results describe the behavior of the regulatory component and do not, by themselves, demonstrate that the complete system is safer or more accurate. To examine whether the proposed approach is useful in practice, the paper presents an experimental plan based on synthetic multisensor tracking. The plan includes comparisons with models given equivalent information, separate datasets for training and testing, and measures that assess both prediction quality and operational behavior. The discussion also considers reference with established techniques, such as recurrent gating, feature-wise modulation, hypernetworks, neuromodulated recurrence, differentiable plasticity, and runtime assurance. Whether MNNs provide sufficient practical benefits to justify their additional complexity remains an open question that requires experiments with trained models and validation in real-world settings.

KEYWORDS: autonomous systems, anomaly detection, neural modulation, reliability monitoring, regime control, selective prediction, sensor degradation.

INTRODUCTION

For an autonomous system to operate reliably, it must do more than produce accurate predictions. It also needs to determine how those predictions should be used when the quality of incoming information changes. A model may perform well when sensor data are reliable but continue to express high confidence when measurements are missing, biased, or affected by changes in system dynamics. Under these circumstances, it becomes important to consider not only the accuracy of the predictions, but also whether the system should act on them and whether adjustments to the model should be allowed. Anomaly detection in autonomous robotic missions involves spatial, temporal and spatiotemporal phenomena [1], while generalized out-of-distribution evaluation emphasizes the need to specify which kinds of novelty and shift a method is expected to handle [2], [3]. These considerations suggest that a clearer connection is needed between evidence about system reliability and the way a model operates. However, they do not necessarily mean that modifying the network internally is better than relying on an external supervisory mechanism.

Several established neural architectures already permit their behavior to change in response to incoming information. LSTM networks use gates to control how information is stored and retained [4], while attention mechanisms adjust interactions according to context [5]. FiLM modifies network features through input-dependent affine transformations [6], and hypernetworks produce

International Journal of Current Science Research and Review
ISSN: 2581-8341
Volume 09 Issue 10 October 2026
DOI: 10.47191/ijcsrr/V9-i10-27, Impact Factor: 8.048
IJCSRR @ 2026
www.ijcsrr.org

parameters that influence the operation of another network [7]. Neuromodulated recurrent networks are especially relevant because they use a secondary mechanism to influence the dynamics of the main network [8]. Since existing neural architectures already support changes in how information is processed, the aim of this work is not to introduce modulation as a new capability. Rather, it investigates whether a small set of regulatory variables, each with a clearly defined role, can help the system respond more appropriately when the quality and reliability of incoming information vary.

We use the term ***mevronic*** to designate this proposed interface and the associated design constraints. It is a technical label for an engineering construction, without a claim of biological fidelity or emotional modeling. An MNN combines evidence estimators, persistent bounded memory, explicitly defined readouts and a declared set of control effects. The proposed reference realization uses three independently anchored memory coordinates and two derived controls. Five coordinate names are not treated as five independently necessary latent dimensions.

This article makes four main contributions. First, it defines the proposed approach in operational terms and describes a reference design that specifies how the system functions. Second, it examines how the approach relates to existing architectures, including where their capabilities overlap and what the proposed semantic constraints actually add. Third, it presents mathematical bounds that help explain the behavior of the regulatory updates, supported by a numerical example that can be reproduced. Finally, it outlines an experimental evaluation designed to determine whether any observed improvements come from the defined meanings of the regulatory variables, the use of memory, access to additional information, or simply an increase in model capacity. No learned benchmark comparison or real-world autonomy experiment is reported. The main hypothesis is that regulatory memory with clearly defined roles can help the system maintain useful predictions while reducing the risk of relying on them when unreliable conditions persist. If the approach does not perform better than comparable general-purpose modulation methods or external supervisory mechanisms, there would be little practical reason to prefer its additional structure.

RELATIONSHIP WITH EXISTING METHODS

Conditional computation and modulation

Recurrent gating and attention mechanisms can both be used within an MNN, but neither determines whether the proposed regulatory interface is present. A recurrent network can use reliability information to maintain an internal state that guides its behavior over time. Similarly, an attention-based model can incorporate persistent memory and produce outputs that support supervisory decisions. This means that existing architectures may already be capable of reproducing the behavior described for MNNs. The present work therefore does not claim that such behavior is impossible to achieve through recurrent gating or attention alone [4], [5].

The feature transformation used below is an instance of conditional affine modulation, closely related to FiLM [6]. Hypernetworks demonstrate the broader principle of one network controlling the parameters of another [7]. Neuromodulated RNNs use a low-dimensional secondary signal to alter recurrent weights and explicitly analyze relationships to gating [8]. These precedents mean that a second network, a gain vector or a persistent modulator is not sufficient evidence of architectural novelty. The proposed distinction is the combination of fixed coordinate targets, a restricted state update, bounded control effects and evaluation against independent operational outcomes.

Differentiable plasticity studies trainable adaptation mechanisms [9], [12]. MNN does not introduce plasticity itself; it optionally supplies a bounded permission signal to a separately specified adaptation rule. The core inference experiment below disables deployment-time parameter updates, allowing regulatory inference to be evaluated without conflating it with online learning.

Reliability estimation and runtime assurance

Calibration describes how closely a model's predicted probabilities match its actual correctness. Simply assigning meaningful names to regulatory variables does not guarantee that their values are well calibrated [10]. Similarly, the usefulness of out-of-distribution (OOD) detection and anomaly scores depends on the task being evaluated and the information available to the model [2], [3]. In this study, the evidence estimators remain fixed and are shared across the main comparison models. The regulatory state stores and updates their outputs using predefined rules, while the accuracy and reliability of those estimators are evaluated separately.

The Neural Simplex architecture provides a relevant example of an existing approach to reliable autonomous control. It combines a neural controller with a backup controller and supervisory logic, allowing the system to switch between them, recover from unsafe conditions, and potentially improve through online learning [11]. This shows that fallback mechanisms and returning to normal

International Journal of Current Science Research and Review
ISSN: 2581-8341
Volume 09 Issue 10 October 2026
DOI: 10.47191/ijcsrr/V9-i10-27, Impact Factor: 8.048
IJCSRR @ 2026
www.ijcsrr.org

operation are already established concepts. The proposed MNN approach differs mainly in where reliability information is used: it introduces a regulatory mechanism directly into the inference process rather than relying entirely on external supervision. However, this does not eliminate the need for runtime safety assurance. Experiments must determine whether placing the regulatory mechanism inside the network offers advantages over an external supervisor that has access to the same evidence and memory. ***Table 1*** summarizes the intended comparisons. The categories overlap; they are not disjoint mathematical classes.

Table 1. Prior mechanisms and the additional question posed by MNN.

Mechanism	Established capability	Required MNN comparison
LSTM and attention [4], [5]	Context-dependent memory or interactions	Same evidence, comparable temporal state and capacity
FiLM [6]	Conditional affine feature modulation	Instantaneous conditioning versus persistent anchored conditioning
Hypernetworks [7]	Conditional parameter generation	Benefit of restrictions rather than arbitrary parameter modulation
Neuromodulated RNNs [8]	A secondary signal alters recurrent dynamics	Generic versus semantically anchored memory
Differentiable plasticity [9]	Trainable adaptation mechanisms	Added value and cost of permission-controlled adaptation
Neural Simplex [11]	Supervisory fallback and recovery	Internal modulation versus an evidence-matched external supervisor

OPERATIONAL DEFINITION AND SCOPE

Information and timing

At time step t, the available observations are denoted by x_t, and the primary network produces a representation $b_t \in \mathbb{R}^d$. Reliability is described using three quantities: an evidence vector $e_t \in [0,1]^3$, a regulatory memory $r_t \in [0,1]^3$, and an operational readout $m_t \in [0,1]^5$. The primary network may also retain an internal state h_t. In the reference implementation, a separate monitoring process estimates reliability from current and earlier observations. This process remains fixed and is not affected by feature modulation. During deployment, it must work without knowing the true state of the environment.

The order of computation matters because reliability estimates need to be available before they can affect a decision. Once an observation arrives, the monitor evaluates the evidence, which is then used to update regulatory memory. The updated values influence feature modulation before the network produces its task output and the final operational decision is made. In this way, information collected at step t can already affect the decision at that step. Model parameters may be adjusted when adaptation is permitted, but only after the current prediction has been generated. There is also a timing limitation. If the monitor cannot produce its estimate before the decision deadline, the system has to work with earlier evidence, and the delay must be reported. For the reference experiments, we assume that the full sequence can be completed within one sampling interval. Whether this is achievable in practice will need to be confirmed through runtime profiling..

Evidence is external to the regulatory recurrence, not necessarily external to the complete system. Frozen estimators may share the same observations and failure modes as the task model. Their errors can therefore be correlated. Replacing this monitoring path with a prediction-dependent feedback path changes the analytical assumptions and must be treated as a separate extension.

Definition of the MNN interface

Definition 1. An MNN interface combines together a neural network that processes information and a regulatory memory that keeps track of reliability over time. It also uses clearly defined reliability measures and mechanisms that determine how this information affects the network's predictions and its ability to adapt. Each regulatory variable must have a clearly identified quantity to measure or a specified method for calculating its value. Its update rule, permitted range, and function within the system must also be defined so that they can be examined independently. A model is considered to follow the MNN approach when these requirements are satisfied. This definition concerns how the model is organized and how its regulatory variables are interpreted, it does not claim that MNNs can represent functions beyond the capabilities of general recurrent neural networks. A generic learned vector whose

International Journal of Current Science Research and Review
ISSN: 2581-8341
Volume 09 Issue 10 October 2026
DOI: 10.47191/ijcsrr/V9-i10-27, Impact Factor: 8.048
IJCSRR @ 2026
www.ijcsrr.org

coordinates are named after training does not satisfy the anchoring requirement. An external supervisor can satisfy similar semantic requirements; internal feature modulation is the additional architectural choice studied here. For this article, a minimal MNN realization must influence at least one internal feature transformation as well as exposing its regulatory state. Operational fallback and online adaptation are optional extensions, with their own validation requirements.

For fixed inference-time parameters, a causal discrete implementation can be written with augmented state $s_t = (h_t, r_t, j_t)$, where j_t contains mode and counters:

$$s_t = G(s_{t-1}, x_t), \qquad \hat{y}_t = H(s_t, x_t) \quad (1)$$

Equation (1) places the implementation within the general class of recurrent state systems. It does not establish an exact realization by a particular finite LSTM with the same size. If online parameters change, those parameters and optimizer memory must also be included in the state. Architectural modularity, inductive bias, auditability and finite-resource performance remain meaningful even without a separation in computable functions.

Anchored coordinates

The evidence vector is $e_t = (\hat{c}_t, \hat{n}_t, \hat{s}_t)^\top$. Its components denote an estimated correctness probability for a fixed reference predictor, a normalized innovation score, and a sensor-quality score. The reference predictor is distinct from the modulated final predictor. Consequently, its correctness estimate must not be advertised as a calibrated confidence estimate for the final MNN output.

Table 2 gives the coordinate contract. Calibration and fidelity refer to properties that must be measured on held-out data; the definition alone does not provide them.

Table 2. Meaning and verification of the regulatory coordinates.

Coordinate	Construction	Interpretation and independent check
$\boldsymbol{c_t}$	Smoothed $\hat{c}_t$	Recent estimated correctness of the reference predictor; assess correctness estimation on held-out truth
$\boldsymbol{n_t}$	Smoothed $\hat{n}_t$	Accumulated normalized innovation evidence; evaluate against independently generated event labels
$\boldsymbol{s_t}$	Smoothed $\hat{s}_t$	Recent sensor-quality evidence; test response to dropout, inconsistency and common-mode faults
$\boldsymbol{a_t}$	$\max(1 - c_t, n_t, 1 - s_t)$	A declared caution index; test effects on modulation and selective reliance
$\boldsymbol{p_t}$	$p_{\max} c_t s_t (1 - a_t)$	Upper allowance for adaptation, not an instruction to update

The memory update and readout are

$$r_t = (I - D) r_{t-1} + D e_t, \quad D = \mathrm{diag}(\alpha_c, \alpha_n, \alpha_s), \quad 0 < \alpha_i \leq 1 \quad (2)$$

$$r_t = (c_t, n_t, s_t)^\top, \quad m_t = (a_t, c_t, n_t, s_t, p_t)^\top \quad (3)$$

If $p_{\max} \in [0,1]$ is set and initialize $r_0 = (0,1,0)^\top$, representing unavailable confidence, high unresolved novelty and no established sensor trust. A 20-step observation-only warm-up precedes scored decisions. The default is $\alpha_c = \alpha_n = \alpha_s = 0.2$. These values are design choices, not optimized findings.

Equation (2) defines the three memory variables and gives each one a specific meaning. The remaining two variables are calculated from these values rather than treated as independent memory components. This distinction is important because the model does not claim to have identified five separate underlying factors. Representing the reliability of individual sensors would require more state variables, along with experiments to determine whether they provide useful information. In this implementation, novelty refers to unexpected differences between predicted and observed measurements, not to unfamiliar concepts or content in images or language. Applying MNNs to such tasks would therefore require a different way of defining and estimating reliability evidence.

A CAUSAL REFERENCE ARCHITECTURE

Representation and evidence path

For the tracking experiment described in Section 7, each observation contains four inputs: measurements from two scalar sensors and two masks indicating whether those measurements are valid. Before entering the network, the sensor readings are normalized using statistics from the training set. If a sensor reading is missing, its value is set to zero. The corresponding validity mask is still

International Journal of Current Science Research and Review
ISSN: 2581-8341
Volume 09 Issue 10 October 2026
DOI: 10.47191/ijcsrr/V9-i10-27, Impact Factor: 8.048
IJCSRR @ 2026
www.ijcsrr.org

provided, allowing the network to recognize that the measurement is unavailable not interpreting it as a genuine zero reading. The resulting four inputs are passed to a single-layer GRU with 32 hidden units, which generates the representation $b_t \in \mathbb{R}^{32}$. The GRU is trained on the tracking task before the modulation experiments begin. Its parameters are then kept fixed so that the effects of the regulatory mechanism can be examined without changes to the underlying network. Keeping the backbone fixed allows the effect of the regulatory interface to be studied without changes in the underlying representation network. Fine-tuning the complete model is considered separately and its results must be reported as an additional experimental condition.

Reliability is evaluated by a separate monitoring component that operates independently of the network's feature modulation. It relies on two fixed elements a reference prediction head connected to b_t and a linear state estimator. Neither is modified during the comparison. In the tracking experiment, the reference head provides an estimate of the target position, denoted by z_t^{ref}. Its prediction is treated as correct if the difference from the true position is within $\varepsilon = 0.25$ simulation units. A logistic model is then used to estimate how likely the reference prediction is to be correct. It takes as inputs the standardized innovation values from the two sensors, their validity masks, the sensor-quality measure, and the normalized anomaly score. The logistic model is fitted on a separate calibration dataset, where the true correctness of each reference prediction is known. The test data and their labels are kept out of this fitting process.

For a valid sensor j, it is calculated the innovation as $\nu_{j,t} = o_{j,t} - H\xi_{t|t-1}$, with the corresponding variance $S_{j,t} = HP_{t|t-1}H^T + R_j$. Then take the largest absolute standardized innovation across the sensors that have provided measurements, giving v_t. To obtain the anomaly score n_t, Equation (4) compares this value with the distribution observed during nominal calibration. Two quantiles are used for this purpose the median κ_{50} and the 99th percentile κ_{99}. A lower bound of 10^{-6} is imposed on the denominator to avoid numerical problems when the quantiles are very close, while the resulting score is limited to [0, 1].

The anomaly score is

$$\hat{n}_t = \text{clip}\left(\frac{v_t - \kappa_{50}}{\max(\kappa_{99} - \kappa_{50}, 10^{-6})}, 0.1\right) \quad (4)$$

The treatment of missing measurements requires particular care. If neither sensor provides a reading, we assign $n_t = 1$. For the correctness estimator, unavailable innovation values are entered as zero, but the accompanying validity masks make their absence explicit. Valid standardized innovations, on the other hand, are clipped to [−10, 10] before being supplied to the logistic model. This limit applies only at that stage; it does not alter the innovation values used elsewhere. Keeping the masks alongside the innovation inputs matters because a missing reading carries no evidence of agreement between the predicted and observed values, even though its numerical placeholder is zero.

The sensor-quality score s_t shows the agreement between the two sensor measurements. Sensor quality is $\hat{s}_t = \exp[-1/2\left((o_{1,t} - o_{2,t})/(3\sqrt{R_1 + R_2})\right)^2]$ when both sensors are valid, 0.5 when only one is valid, and 0 when neither is valid. When both sensors provide valid readings, we calculate this score using an exponential function of their difference, taking the nominal sensor noise levels into account. The difference is measured in the original units, before the normalization used for the GRU inputs. The factor of 3 sets the tolerance for disagreement under nominal conditions and remains fixed throughout testing, including when sensor faults occur. If only one sensor is available, we assign $s_t = 0.5$. If neither sensor provides a valid reading, the score becomes zero.

There is an important limitation to using sensor agreement as a measure of quality. Two sensors may report nearly the same value even when both are affected by a common bias. Their agreement would produce a high quality score despite the measurements being incorrect. We therefore include common-mode faults in the evaluation to examine this weakness directly. When both measurements are missing, we also set the correctness estimate c_t to zero, without using the value returned by the fitted logistic model.

For the fixed state estimator, the transition and noise matrices are used given in Section 7. The initial state is $(0, 0)^T$, and the initial covariance is diag(1/3, 0.25^2/3). At each step, available measurements are incorporated through scalar Kalman updates, first using sensor 1 and then sensor 2. A measurement is simply omitted if its validity mask indicates that it is missing. The estimator retains the nominal noise settings throughout testing, even during periods of sensor corruption. Its position estimate, z_t^{fb}, is used as the fallback output when the main predictor is not selected. This gives us a consistent alternative for the experiments, although the fallback itself offers no guarantee of safety or accuracy when operating conditions deteriorate.

International Journal of Current Science Research and Review
ISSN: 2581-8341
Volume 09 Issue 10 October 2026
DOI: 10.47191/ijcsrr/V9-i10-27, Impact Factor: 8.048
IJCSRR @ 2026
www.ijcsrr.org

BOUNDED FEATURE MODULATION AND OUTPUT MIXING

The feature modulator is a 5-to-16 tanh layer followed by a 16-to-64 linear layer. Splitting its output into two 32-vectors g_t and d_t, define

$$\gamma_t = 1 + 0.25\tanh(g_t), \quad \beta_t = 0.25\tanh(d_t), \quad \tilde{b}_t = \gamma_t \odot b_t + \beta_t \quad (5)$$

Both the gain range [0.75,1.25] and the offset range $[-0.25,0.25]$ are explicit. This is a bounded FiLM-type operator [6], not a new primitive. A linear task head produces $z_t^{\text{nom}} = w^\top \tilde{b}_t + b$. To avoid assuming that latent interpolation preserves safety, the reference design mixes task estimates in the same physical units:

$$\tau_t = c_t s_t, \qquad \hat{z}_t = \tau_t z_t^{\text{nom}} + (1 - \tau_t) z_t^{\text{fb}} \quad (6)$$

Equation (6) defines estimator blending only. Even scalar convex mixing is not a general safety certificate. In protective mode the reported operational estimate is z_t^{fb}, and autonomous reliance on the estimate is withheld. All internal predictions are still logged so that abstention cannot conceal poor estimation.

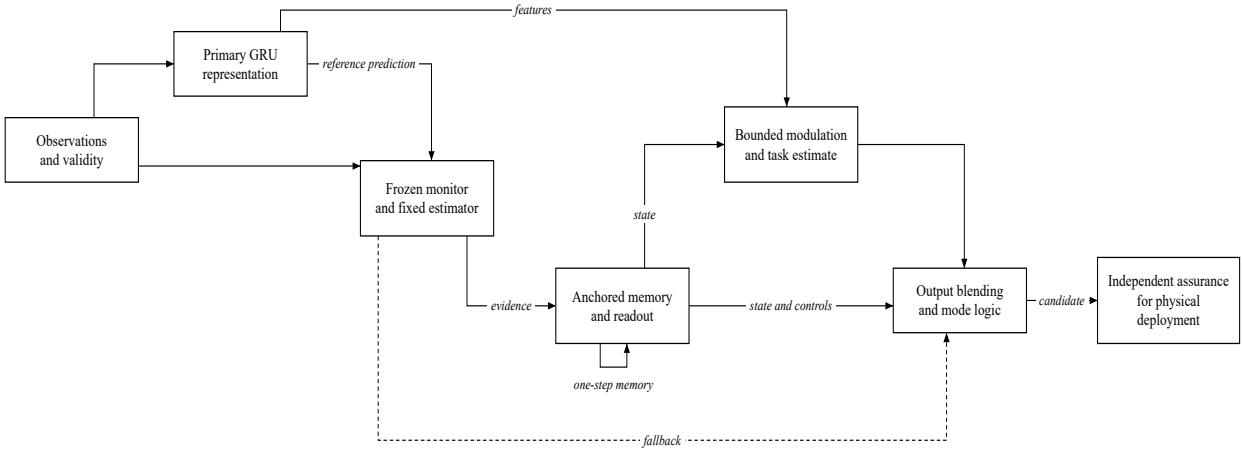


Figure 1. Evidence is computed from an unmodulated monitoring path before the current regulatory update.

Mode logic and latency

The system operates in four modes: nominal, suspicion, protective, and recovery. For the initial design, the thresholds are set to $Hsus = 0.4, Hprot = 0.8,$ and $Hexit = 0.2.$ These values are dimensionless and serve as fixed settings for the proposed implementation, rather than general safety limits. A hard input-validity flag is triggered if both sensors are unavailable or if any input contains a nonfinite value. Mode selection follows a set of priority rules.

1. If the hard validity flag is active or $a_t \geq 0.8$, the system immediately enters protective mode. The exit counter is cleared, and model adaptation is disabled.
2. Once in protective mode, the system checks whether $a_t \leq 0.2$ and both sensors are valid. If these conditions hold for five consecutive steps, it moves into recovery mode. Otherwise, it remains protective, and the counter returns to zero whenever either condition fails.
3. In recovery mode, the first rule still takes priority. The system can return to nominal operation after ten consecutive steps with $a_t < 0.4$ and both sensors valid. If this requirement is interrupted, the counter is reset, but the system stays in recovery unless the first rule forces it back into protective mode.
4. When the system is neither protective nor recovering, it determines which of the remaining modes is selected. Values of 0.4 or higher lead to suspicion mode, while lower values lead to nominal mode.

The system begins in protective mode, with regulatory states and counters updated throughout the warm-up period. In the selective-reliance benchmark, predictions are withheld during protective and recovery modes, while nominal and suspicion modes allow them to be used. Any change to this behavior is treated as a separate policy ablation.

Complete loss of sensing is handled directly through the validity flag, without waiting for the smoothed reliability estimate to respond. Other changes in reliability remain subject to the delays introduced by filtering and computation. An additional hazard

International Journal of Current Science Research and Review
ISSN: 2581-8341
Volume 09 Issue 10 October 2026
DOI: 10.47191/ijcsrr/V9-i10-27, Impact Factor: 8.048
IJCSRR @ 2026
www.ijcsrr.org

detector could also be used to trigger protective mode, whether learned or designed for a particular application. Such an extension would require separate testing to measure how often it raises false alarms or fails to detect relevant events.

Optional adaptation

The core benchmark sets the deployment adaptation flag $\chi_t = 0$. An optional extension allows updates only to the final task head, after the current prediction. With a specified self-supervised loss ℓ_t^{ss} and gradient cap G, the update is

$\theta_{t+1} = \Pi_C[\theta_t - \eta\chi_t p_t \mathrm{clip}_G(\nabla_\theta \ell_t^{ss})]$ (7)

The constraint C = {θ : ‖θ − θ₀‖₂ ≤ R} keeps the parameters from moving too far from the deployed head. An update is allowed only when χ_t confirms that the update window is approved, the teaching signals are valid, and the system is not in protective mode. One possible extension is to train a separate head for one-step prediction and compare its output with a trusted sensor reading obtained later. Gradients would not pass through this target. The new head would need its own offline training, rather than reusing the current-state task head. Sensor agreement alone cannot confirm that the target is reliable, especially when both sensors share the same bias. For this reason, the extension is not part of the main performance claim, and no improvement is assumed. Experiments using it must specify η, G, R, target validity, rollback conditions, update frequency, and computational cost. Equation (7) uses the correct sign for minimizing the loss and distinguishes how much adaptation is allowed from whether an update can actually take place.

Per-step algorithm

Algorithm 1. Reference inference with anchored regulatory memory. Inputs are observations, validity masks, previous GRU and estimator states, r_{t-1}, the previous mode and counters. Parameters and calibration constants are fixed.

1. Validate observations and compute the hard validity flag. Normalize values with training statistics and preserve masks.

2. Compute b_t and the unmodulated reference prediction. Run the fixed estimator prediction, record innovations, then assimilate valid observations.

3. Compute $\hat{c}_t, \hat{n}_t, \hat{s}_t$ using the frozen monitoring functions.

4. Apply (2), then construct a_t, p_t, m_t using Table 2 and (3).

5. Apply the priority rules in Section 4.3 to obtain the mode and counters.

6. Apply (5) and the task head, then (6); select the fallback estimate in protective mode.

7. Emit the estimate, continue/abstain decision, mode and complete diagnostic trace. A physical controller must additionally pass its own assurance check.

8. Only after emission, apply (7) if an adaptation experiment has explicitly enabled it; otherwise retain all parameters.

Implementation notes. The frozen monitoring functions and validity conventions are specified in Section 4.1. Innovations are recorded before measurement assimilation, which processes valid sensors in the prescribed order. The mode update follows the priority rules in Section 4.3. The diagnostic trace records observations and validity information, innovations, evidence, regulatory states, modulation coefficients, all intermediate estimates, mode counters and the reliance decision. Parameter adaptation is disabled in the primary benchmark. When explicitly enabled, it is performed only after output emission and

$(x_t, \bar{v}_t, f_t) \leftarrow \mathrm{ValidateAndNormalize}(o_t, v_t)$
$(b_t, h_t) \leftarrow \mathrm{GRUStep}(x_t, h_{t-1})$
$z_t^{\mathrm{ref}} \leftarrow \mathrm{ReferenceHead}(b_t)$
$K_t^- \leftarrow \mathrm{EstimatorPredict}(K_{t-1})$
$\nu_t \leftarrow \mathrm{ComputeInnovations}(o_t, \bar{v}_t, K_t^-)$
$K_t \leftarrow \mathrm{AssimilateValidObservations}(K_t^-, o_t, \bar{v}_t)$
$z_t^{\mathrm{fb}} \leftarrow \mathrm{PositionEstimate}(K_t)$
$e_t \leftarrow \mathrm{FrozenMonitor}(o_t, \bar{v}_t, \nu_t, K_t^-)$
$r_t \leftarrow (I - D)r_{t-1} + De_t$
$(c_t, n_t, s_t) \leftarrow r_t$
$a_t \leftarrow max(1 - c_t, n_t, 1 - s_t)$
$p_t \leftarrow p_{max} c_t s_t (1 - a_t)$
$m_t \leftarrow (a_t, c_t, n_t, s_t, p_t)^{\mathrm{T}}$
$(q_t, j_t) \leftarrow \mathrm{UpdateMode}(q_{t-1}, j_{t-1}, a_t, \bar{v}_t, f_t)$
$(g_t, d_t) \leftarrow \mathrm{FeatureModulator}(m_t)$
$\tilde{b}_t \leftarrow [1 + 0.25tanh(g_t)] \odot b_t + 0.25tanh(d_t)$
$z_t^{\mathrm{nom}} \leftarrow \mathrm{TaskHead}(\tilde{b}_t, \theta_t)$
$\tau_t \leftarrow c_t s_t$
$\hat{z}_t \leftarrow \tau_t z_t^{\mathrm{nom}} + (1 - \tau_t) z_t^{\mathrm{fb}}$
if $q_t = PROTECTIVE$ **then**
    $z_t^{\mathrm{op}} \leftarrow z_t^{\mathrm{fb}}$
else
    $z_t^{\mathrm{op}} \leftarrow \hat{z}_t$
end if
if $q_t \in PROTECTIVE, RECOVERY$ **then**
    $decision_t \leftarrow ABSTAIN$
else
    $decision_t \leftarrow CONTINUE$
end if
$S_t \leftarrow (h_t, K_t, r_t, q_t, j_t)$
$T_t \leftarrow \mathrm{BuildDiagnosticTrace}()$
$\mathrm{Emit}(z_t^{\mathrm{op}}, decision_t, q_t, T_t)$
$\theta_{t+1} \leftarrow \theta_t$
if $\mathrm{AdaptationAllowed}(t, q_t)$ **then**
    $u_t \leftarrow \mathrm{ClipGradient}(\nabla\theta \ell_t^{ss}, G)$
    $\theta_{t+1} \leftarrow \mathrm{ProjectC}(\theta_t - \eta p_t u_t)$
end if
return (S_t, θ_{t+1})

International Journal of Current Science Research and Review
ISSN: 2581-8341
Volume 09 Issue 10 October 2026
DOI: 10.47191/ijcsrr/V9-i10-27, Impact Factor: 8.048
IJCSRR @ 2026
www.ijcsrr.org

subject to all conditions in Section 4.4. A physical controller must additionally pass an independent assurance check before executing an action.

Analytical properties and their limits

The following results characterize the declared regulatory update. They are elementary consequences of bounded exponential filtering, presented as design guarantees for this realization. They do not establish novelty of filtering, superiority over other networks, calibration under shift or stability of an autonomous plant.

Invariance and fading memory

Proposition 1. Forward invariance. If $r_0 \in [0,1]^3$, $e_t \in [0,1]^3$, $0 < \alpha_i \leq 1$ and $0 \leq p_{\max} \leq 1$, then $r_t \in [0,1]^3$ and $m_t \in [0,1]^5$ for all $t \geq 0$.

Proof. Each coordinate of (2) is a convex combination of its previous value and a number in $[0,1]$. Induction gives the claim for r_t. The maximum of $1 - c_t, n_t, 1 - s_t$ lies in $[0,1]$, and the product defining p_t lies in $[0, p_{\max}]$. This proves the readout claim. Boundedness does not imply convergence for arbitrary time-varying evidence.

Proposition 2. Sensitivity to initial state and evidence. Let r_t and r'_t follow (2) with identical fixed D and possibly different evidence. Write $q = 1 - \alpha_{\min} < 1$ and $\delta_t = \| e_t - e'_t \|_\infty$. Then

$$\| r_t - r'_t \|_\infty \leq q^t \| r_0 - r'_0 \|_\infty + \alpha_{\max} \sum_{k=1}^{t} q^{t-k} \delta_k. \quad (8)$$

If $\delta_k \leq \delta$, the second term is at most $\alpha_{\max}(1 - q^t)\delta/\alpha_{\min}$. If all rates equal α, it is at most $(1 - (1 - \alpha)^t)\delta$.

Proof. Subtract the two updates, apply the induced infinity norms $\| I - D \|_\infty = q$ and $\| D \|_\infty = \alpha_{\max}$, and iterate the resulting scalar inequality. The uniform bound follows from the finite geometric sum.

With identical evidence, differences due to initialization decay geometrically. Persistent evidence bias is not removed by this result: for a common rate, a constant bias can be transmitted with unit steady-state gain. Smoothing therefore cannot repair a systematically incorrect monitor. The derived alertness satisfies $|a_t - a'_t| \leq \| r_t - r'_t \|_\infty$, while $|p_t - p'_t| \leq 3p_{\max} \| r_t - r'_t \|_\infty$ by expanding the product and using bounded factors.

If evidence depends on regulatory memory, (8) still describes two prescribed evidence streams but is not a closed-loop contraction claim. For a restricted extension $e_t = Q_t(r_{t-1}, x_t)$ that is uniformly L_Q-Lipschitz in r for fixed x_t, a sufficient per-step contraction condition is

$$1 - \alpha_{\min} + \alpha_{\max} L_Q < 1. \quad (9)$$

This follows by substitution into the one-step inequality. If Q_t contains additional evolving hidden states, the full joint recurrence must be analyzed; a bound on a partial map is insufficient. The reference design avoids this dependency by computing evidence in the frozen unmodulated path.

Response time under persistent evidence

Proposition 3. Threshold-crossing time. Consider $n_k = (1 - \alpha)n_{k-1} + \alpha N$ for constant N, $0 < \alpha < 1$, and $n_0 < H < N$. The first update index satisfying $n_k \geq H$ is

$$k_* = \left\lceil \frac{\log((N-H)/(N-n_0))}{\log(1-\alpha)} \right\rceil \quad (10)$$

Proof. Solving the recurrence gives $n_k = N - (N - n_0)(1 - \alpha)^k$. Rearranging $n_k \geq H$ and dividing by the negative number $\log(1 - \alpha)$ yields the lower bound on integer k. If $\alpha = 1$, crossing occurs at the first update; if $n_0 \geq H$, it has already occurred.

The formula refers to persistent evidence and a single coordinate. The full alertness can cross earlier because it is the maximum of three terms, and a raw override can trigger independently. A continuous-time delay estimate additionally requires sampling period and computation latency. Under the convention that an event occurs immediately after the previous sample, the upper sampling contribution is $k_* \Delta t$; actual phase can shorten this by up to one interval.

Semantic constraints and safety boundaries

Equation (2) fixes which evidence stream drives each coordinate. An arbitrary permutation would change the declared interface unless the coordinate contracts were also renamed. This is a design constraint, not statistical identification of a hidden physical variable. A named confidence coordinate remains only as valid as its correctness estimator. To test semantics, interventions must modify a coordinate while the representation and other coordinates are held fixed, then measure the declared downstream effect. Correlation alone does not establish that role.

International Journal of Current Science Research and Review
ISSN: 2581-8341
Volume 09 Issue 10 October 2026
DOI: 10.47191/ijcsrr/V9-i10-27, Impact Factor: 8.048
IJCSRR @ 2026
www.ijcsrr.org

Mode switching is discontinuous. Small state perturbations near a threshold can change a mode even when (8) is small. Hysteresis and consecutive-step exit rules reduce some switching sensitivity but do not guarantee a minimum dwell time under arbitrary input. Abrupt hazardous inputs can force immediate re-entry into protective mode.

For a physical system $\xi_{t+1} = f(\xi_t, u_t, w_t)$ with disturbance set $\mathcal{W}$, suppose an independent assurance mechanism knows a controlled invariant set $\mathcal{S}$ and selects only actions satisfying $f(\xi_t, u_t, w) \in \mathcal{S}$ for every $w \in \mathcal{W}$. If $\xi_0 \in \mathcal{S}$ and a valid action is always available, invariance follows by induction. These are substantial assumptions about state knowledge, dynamics and action availability. No such set or verified mechanism is constructed for MNN here. A learned penalty, fallback label or feature mixture cannot substitute for those assumptions. Runtime-assurance methods such as Neural Simplex provide a relevant comparison and integration route [11].

DETERMINISTIC ILLUSTRATION OF THE RESPONSE TRADE-OFF

This section reports direct evaluations of (10) and the corresponding recurrence, not results from training neural networks. The setting is a scalar novelty coordinate with $n_0 = 0$, evidence stepping to $N = 1$, threshold $H = 0.8$ and sample period $\Delta t = 0.1$ s. A separate decay calculation starts from $n_0 = 0.9$ with zero evidence and asks when $n_k \leq 0.2$. Table 3 contains computed values.

Table 3. Exact recurrence calculations with illustrative design parameters.

Update rate α	Updates to $n \geq 0.8$	Value at crossing	Upper sampling delay in seconds	Updates from 0.9 to $n \leq 0.2$
0.05	32	0.8063	3.2	30
0.10	16	0.8147	1.6	15
0.20	8	0.8322	0.8	7
0.50	3	0.8750	0.3	3

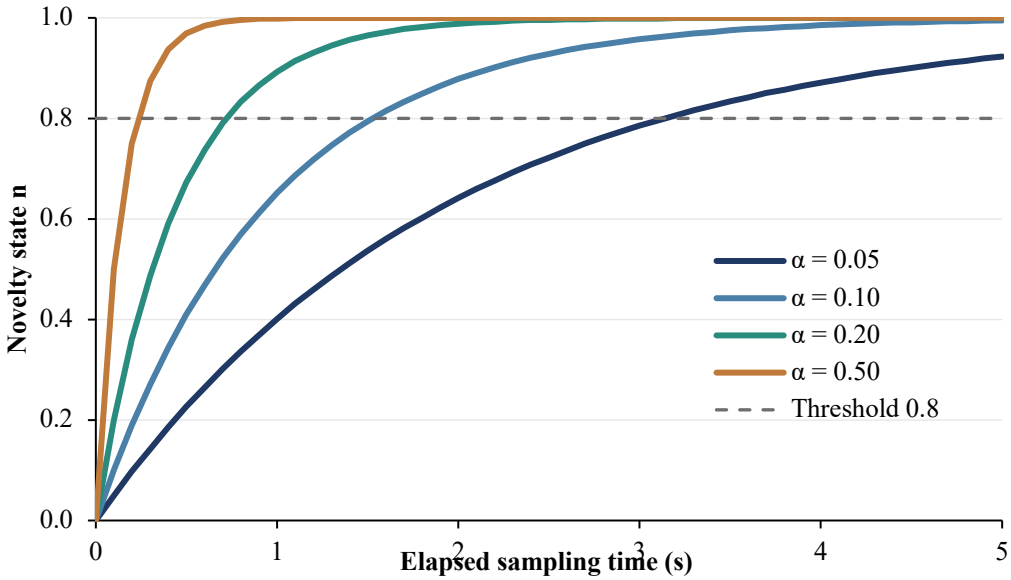


Figure 2. Increasing the update rate reduces threshold-crossing delay for this constant step input.

The default rate 0.2 requires eight updates for the novelty coordinate alone to cross 0.8. During recovery, seven updates bring a state of 0.9 below 0.2; the mode logic then additionally requires five consecutive qualifying observations before entering recovery. The first qualifying observation counts toward those five, and return to nominal requires ten further qualifying recovery steps. These delays must be included in an operational evaluation rather than hidden behind a smoothness score.

For equal rates and identical evidence, (8) predicts an initial-state discrepancy decaying as $(1 - \alpha)^t$. For a persistent evidence discrepancy of 0.1 and identical initialization, the worst-case bound tends to 0.1, independently of the common rate. The example

International Journal of Current Science Research and Review
ISSN: 2581-8341
Volume 09 Issue 10 October 2026
DOI: 10.47191/ijcsrr/V9-i10-27, Impact Factor: 8.048
IJCSRR @ 2026
www.ijcsrr.org

illustrates why smooth state trajectories and small initialization sensitivity do not establish monitor correctness. It also explains why temporal regularization cannot be interpreted as a safety guarantee.

PROSPECTIVE EVALUATION PROTOCOL

Purpose and benchmark construction

The following is a specified experiment to be performed, not a completed benchmark. It targets sensor degradation and dynamics shift in sequential estimation with selective reliance. It does not establish performance on visual semantic novelty, navigation or closed-loop vehicle control. A subsequent physical or high-fidelity closed-loop study is required before making autonomous-operation claims beyond this proxy task.

Generate independent trajectories of length 200 at 10 Hz. With state $\xi_t = (z_t, v_t)^\top$, define

$$\xi_{t+1} = A\xi_t + \omega_t, \quad A = \begin{bmatrix} 1 & 0.1 \\ 0 & 1 \end{bmatrix}, \quad \omega_t \sim \mathcal{N}(0, Q) \quad (11)$$

$$Q = \mathrm{diag}(10^{-4}, 10^{-3}), \quad o_{j,t} = H\xi_t + \varepsilon_{j,t}, \quad H = [1\ 0], \quad R_1 = 0.01,\ R_2 = 0.04 \quad (12)$$

Initial position is uniform on $[-1,1]$, velocity uniform on $[-0.25,0.25]$, and all nominal process and measurement noises are independent. Values are in declared simulation units, with position error tolerance $\epsilon = 0.25$. Normalize sensor channels using means and standard deviations estimated only from valid training observations; retain the same transformation at evaluation. Ground-truth trajectories and fault labels are available only for offline training and scoring. Use NumPy Generator with PCG64 and SeedSequence to create disjoint manifests for 4000 training, 1000 calibration, 1000 validation and 2000 test trajectories, with root seeds 1001, 2001, 3001 and 4001 respectively. Spawn one child seed per trajectory and persist the generated arrays, fault schedules and software versions. Train all models with five initialization seeds 11, 22, 33, 44 and 55. Dataset seeds are distinct from model seeds.

Each dataset partition contains equal numbers of sequences from five conditions: normal operation, single-sensor bias, increased sensor noise, sensor dropout, and changes in velocity dynamics. For sequences with faults, the event begins at a randomly selected step between 50 and 100 and lasts between 20 and 50 steps, with both limits included. The affected sensor is chosen randomly when relevant. Bias faults introduce a constant positive or negative offset with equal probability. Noise bursts increase the selected sensor's noise standard deviation, while dropout marks its measurements as invalid. During a velocity-drift event, a signed value b_v is added to the velocity at every transition. Table 4 gives the fault magnitudes used for development and held-out testing. Once the event ends, normal data generation resumes, but the system state continues from its current value rather than being reset.

Table 4. Perturbation definitions and held-out stress conditions.

Family	Training, calibration and validation magnitudes	Primary test magnitude	Additional stress test
Sensor bias	Absolute offset uniform on [0.3, 0.6]	Uniform on [0.8, 1.2]	Equal bias on both sensors
Noise burst	Standard-deviation multiplier uniform on [2, 4]	Uniform on [5, 8]	Simultaneous bursts on both sensors
Dropout	One sensor invalid in the event window	Same family with independent schedules	Both sensors unavailable
Dynamics shift	$\lvert b_v \rvert$ uniform on [0.01, 0.02]	Uniform on [0.03, 0.05]	Dynamics shift overlapping a sensor fault

The main test set contains 400 sequences for each condition. A separate stress set includes 1,000 sequences, with 250 per stress condition, generated using seed 5001. For combined faults, the test-level velocity drift is applied together with a bias affecting one randomly chosen sensor. The two events have independently selected time windows, so their overlap is not guaranteed. Results for overlapping and non-overlapping periods are reported separately. Fault types already used during training are not considered entirely unseen. The different test magnitudes represent changes in severity, while common-mode and combined faults provide new fault combinations. No parameters are adjusted using test or stress data.

International Journal of Current Science Research and Review
ISSN: 2581-8341
Volume 09 Issue 10 October 2026
DOI: 10.47191/ijcsrr/V9-i10-27, Impact Factor: 8.048
IJCSRR @ 2026
www.ijcsrr.org

Training stages and information control

The first stage trains a GRU and a linear position head without modulation. Training minimizes mean squared error using Adam, with a learning rate of 10^{-3}, batches of 64 sequences, and a gradient norm limit of 1. Training runs for at most 100 epochs, stopping after 10 epochs without validation improvement. The checkpoint with the lowest validation loss is retained. Only steps 21–200 contribute to training and evaluation; the first 20 steps initialize the recurrent state. A separate backbone is saved for each initialization seed and then frozen for the main comparison.

At this stage, the monitoring model remains frozen. Its outputs are used to train a logistic model that estimates whether the reference prediction is correct, without attempting to identify the type of fault. We fit this model using only the calibration data, with binary cross-entropy loss and an L2 penalty of 10^{-3}. The same partition provides the statistics for standardizing non-mask features, while innovation quantiles are calculated only from nominal calibration sequences. We use a deterministic convex optimizer, run it until numerical convergence, and report the stopping tolerance. Neither the calibration nor the validation sequences are reused for training the main prediction network. Second, compute calibration features with the frozen monitor. Fit the logistic correctness estimator by binary cross-entropy with L2 coefficient 10^{-3} on standardized non-mask features, using calibration data only. Fit feature standardization and innovation quantiles on that partition; derive quantiles from its nominal sequences only. Use a deterministic convex optimizer to numerical convergence and report tolerance. The logistic model predicts correctness of the frozen reference head, not fault identity. Calibration and validation samples never become task-training samples. Third, train comparator-specific feature modulators and final task heads with the same optimizer, epoch budget and stopping rule. The MNN rates, mode rules and output blend remain fixed in the primary experiment. The task objective is the all-step mean squared error of the pre-mode estimate in (6):

$$L_{\text{task}} = \frac{1}{N}\sum_{(i,t)\in\mathcal{J}} \left(\hat{z}_{i,t} - z_{i,t}\right)^2 \quad (13)$$

where $\mathcal{J}$ contains the scored training steps, including steps at which the operational policy would abstain. This prevents hiding training error through abstention. No separate regime-consistency loss is needed because the reference state is anchored by construction. No unnamed safety loss is claimed to enforce physical safety. The protocol logs final-output confidence only if a separate final-output correctness estimator has been trained on the calibration partition for each comparator.

All principal comparators receive identical observations, masks, frozen evidence, reference predictions and fallback estimates. The primary no-modulation control concatenates the three evidence values with b_t in its task head, so removing modulation does not remove information. A raw-observation-only model is a secondary diagnostic, explicitly labeled as information-restricted. Evidence acquisition, the fixed estimator and confidence estimation are included in total inference cost, even when shared across models.

Comparators and ablations

Table 5 separates the main causal questions. Each model uses the same trained backbone for a given seed. Share the fallback and mode machinery wherever the comparison is intended to isolate only feature modulation. For risk–coverage curves, apply the common additional gate $C(t; theta) = 1$ only when the mode is nominal or suspicion, the hard validity flag is absent, and $a(t) < theta$. Vary theta over the distinct validation alertness values and the endpoints 0 and 1.000001. Choose the threshold whose validation coverage is nearest 0.9, breaking ties toward the smaller threshold. Freeze that threshold for test evaluation. Main controlled comparisons use the same anchored supervisory state and therefore the same reliance decisions; they isolate prediction changes caused by internal modulation. A separately calibrated final-output policy may be assessed as an additional experiment.

Table 5. Required model comparisons.

Model	Specification	Question isolated
Information-matched task head	No internal modulation; evidence concatenated with representation	Does conditional feature modulation help?
Instantaneous FiLM	Same bounded modulator, using readout from current evidence without smoothing	Does persistent memory help?
Generic recurrent modulation	Three-dimensional tanh recurrent state; same evidence and bounded modulator, no coordinate anchoring	Do semantic restrictions help beyond memory?
Anchored MNN	Equations (2) - (6), fixed declared readouts	Proposed reference design

International Journal of Current Science Research and Review
ISSN: 2581-8341
Volume 09 Issue 10 October 2026
DOI: 10.47191/ijcsrr/V9-i10-27, Impact Factor: 8.048
IJCSRR @ 2026
www.ijcsrr.org

External supervisor	Same anchored memory, fallback and mode rules; unmodulated primary features	Does internal placement add value?
Fixed estimator	Nominal Kalman estimator and identical validity conventions	Is neural complexity justified on this task?

For generic recurrent modulation, use $\ell_t = \tanh(U\ell_{t-1} + Ve_t + b)$, $\ell_0 = 0$, and a learned affine map to five modulator inputs. Keep the same feature-modulator hidden width initially. For common output mixing and mode selection in the controlled comparison, use the shared anchored supervisory state; only the internal feature-conditioning state changes. This is a stringent test of anchoring inside modulation, not a claim that the generic model has interpretable coordinates. A separate unconstrained end-to-end supervisor may be evaluated, but it answers a broader question and requires a separate training specification.

Report both a common-backbone comparison and a parameter-budget-matched comparison. In the latter, search hidden widths from 4 through 64 to match the trainable controller-and-head parameter count within 5%; if no width satisfies that tolerance, report the nearest counts and do not label the comparison exactly matched. Give measured latency and memory as well as parameter count. Validation search budgets must be equal in number of tried configurations and completed training runs.

Mandatory ablations remove output blending, replace the three-coordinate memory with a scalar filtered maximum adverse-evidence score, vary the common rate over {0.05, 0.1, 0.2, 0.5, 1.0}, and permute evidence-to-coordinate assignments without changing the readout rules. The scalar model derives its five inputs as $(d, 1-d, d, 1-d, p_{\max}(1-d)^3)$; it tests whether distinct evidence histories are useful. Coordinate interventions are performed on the trained models with fixed recorded representations and evidence. Online adaptation is a separate experimental factor; an adaptation claim is prohibited when only the fixed-parameter protocol has been run.

Metrics and statistical analysis

Let $C_t = 1$ denote continued reliance and $E_t = 1\{|\hat{z}_t - z_t| > \epsilon\}$ denote unacceptable estimation error for a model's pre-mode output. Define the proxy unsafe-reliance rate, coverage and conditional error rate as

$$U = \frac{1}{N}\sum_t C_t E_t, \quad V = \frac{1}{N}\sum_t C_t, \quad R = \frac{\sum_t C_t E_t}{\sum_t C_t} \quad (14)$$

If $\sum_t C_t = 0$, report R as undefined, not zero. These are estimation-reliance proxies, not collision probabilities or certified physical safety metrics. Report root mean squared error on all steps and on continued steps, together with U, V and risk–coverage curves. A model that always abstains must receive zero coverage and cannot be declared superior on safety grounds alone.

To make missed events comparable across models, every model is judged against the same definition of a reference failure: any step where the frozen reference predictor's error goes above ε. During those failures, we check how often each model still goes ahead and uses its predictions, which is the case we'd want it to avoid. We also look at the opposite mistake. nder nominal conditions, when the reference prediction is actually correct, how often does the system abstain anyway? We report both rates for each model. Together they give us a common yardstick for comparing models, which we use alongside the model-specific error in (14). Anomaly detection gets its own evaluation, done separately for each corruption type. We score it with AUROC and average precision against the generated fault labels. It's worth keeping this apart from the failure analysis above, because spotting a fault and spotting an unacceptable prediction error are not the same thing. A detector can flag corruption correctly even when the prediction still turns out fine, and the reverse can happen too.We also give the false-positive rate at 95% true-positive rate, but only where the evaluation data are rich enough to actually estimate that operating point. When it is reported, we say which interpolation method was used. Any threshold applied at test time is chosen on validation data, never on the test set.

For correctness probabilities, use Brier score and binary negative log likelihood with probabilities clipped to [10-6,1-10-6] for numerical evaluation. Report ECE using 15 equal-width bins together with reliability diagrams and bin counts [10]. Evaluate instantaneous reference confidence against reference correctness. Evaluate a smoothed confidence coordinate against a correspondingly filtered correctness target, and separately disclose its calibration against instantaneous correctness. Do not transfer either result to the final MNN output without a final-output assessment.

We assess mode behavior in two ways. The first is simple: we count mode switches per 1,000 scored steps. The second is about responsiveness, i.e. how long after a fault starts the system actually does something protective. For each corruption event we measure the delay from onset to the first protective response and report the median and the 95th percentile. Some events get no response at

International Journal of Current Science Research and Review
ISSN: 2581-8341
Volume 09 Issue 10 October 2026
DOI: 10.47191/ijcsrr/V9-i10-27, Impact Factor: 8.048
IJCSRR @ 2026
www.ijcsrr.org

all, and we report what fraction that is too. One caveat about these delays: they only exist for events that were detected, so on their own they can look better than the system really is. That's why we always give them alongside the missed-event rate. A second subtlety is what to do when the system was already in protective mode before the fault began. We don't treat that as a detection (nothing was detected, really) and instead record it as pre-existing protection. It was recorded recovery time, meaning how long the system takes to return to nominal mode after the fault ends, and the share of time it spends in each mode overall. Changes in regulatory memory are measured as the mean infinity norm of $r_t - r_{t-1}$. Smaller isn't necessarily better here. If r_t is smoothed too much it will barely move, but the system may then react late to faults, or its predictions may become less useful.

All models are evaluated on the same trajectories, allowing paired comparisons. Results are reported for each of the five training seeds, together with their mean and standard deviation. We calculate 95% hierarchical bootstrap confidence intervals using 2,000 resamples and seed 6001. For resampling, we draw over training seeds and over whole trajectories within each condition, and we keep models paired throughout. We don't resample individual time steps on their own, since observations inside a sequence are correlated and treating them as independent would make the intervals look tighter than they should. That said, five training seeds isn't much, so these intervals should be read as fairly weak evidence. The main comparison is anchored MNN against generic recurrent modulation, with a target coverage of 0.9 set on validation data. We don't just report the target, though. We also give the coverage actually achieved on the test set, along with the full risk–coverage curves. If a model can't reach 0.9, we say so plainly rather than moving the target after seeing the test results.

Falsification and evidence-quality stress tests

It is set a fairly high bar for calling the framework practically useful. It either beat strong information-matched controls on a utility–risk trade-off that was fix in advance, or match them while offering something they can't, such as being easier to audit or cheaper to run. A better anomaly AUROC on its own doesn't count. Each of the controls also tells us something specific if it holds up. If the external supervisor does just as well, then this task gives no reason to build the modulation into the model. If the generic modulator matches performance and intervening on individual coordinates adds nothing, then restricting the state to semantic coordinates hasn't earned its keep. And if the scalar ablation is enough, the three-coordinate design should be simplified. To check how much the system leans on its monitor, every test replayed trajectory with the evidence channels delayed by 1, 3 and 5 steps. In a separate set of runs, we add a bias of +0.1 or −0.1 to one channel at a time, clipping the result to [0,1]. Every comparator receives exactly the same modified stream. On top of this we test frozen evidence, swapped sensor identities and common-mode corruption. Acquisition delay and computation delay are logged separately so the two can't be confused. The point of these tests is twofold- to see how operationally dependent the system is on the monitor, and to find out whether the regulatory state amplifies evidence that is unreliable rather than damping it.

Before reporting empirical results, release or archive the generator, arrays, split manifests, trained checkpoints, full configuration, environment versions, selected thresholds, per-seed results and scripts producing all tables. The specification in this article is sufficient to define the intended experiment, but implementation and execution are still required. It is not a validated benchmark standard.

DISCUSSION AND LIMITATIONS

The most that can defensibly claim for MNN is that it is a constrained regulatory interface, one member of the wider family of modulated recurrent systems. What it might offer is an explicit link between the history of evidence and the computation the model declares it is doing, together with state trajectories that can be audited and a small number of parameters. It is not stated that this beats a fully flexible network, which may well learn the same behavior or something better. It's also possible that the constraints help when data are scarce, or make failures easier to diagnose. But those are hypotheses. Nothing in the analytical bounds shows them.

Choosing clear coordinate meanings over flexibility comes at a price, and most of it shows up in the individual scores. Correctness, for instance, is only estimated with respect to one reference model at one error tolerance; change either and the estimate means something different. The innovation score is only as good as the nominal dynamics model behind it, while sensor quality, as we measure it, simply can't see certain correlated faults. Then there are the more mundane costs. Filtering adds delay, early behavior depends on how the state is initialized, and nobody can set the mode thresholds once and for all (they are tuned per task). As for the bounds in Section 5, they keep holding even when the evidence is bounded but wrong. That is precisely the reason they say nothing

International Journal of Current Science Research and Review
ISSN: 2581-8341
Volume 09 Issue 10 October 2026
DOI: 10.47191/ijcsrr/V9-i10-27, Impact Factor: 8.048
IJCSRR @ 2026
www.ijcsrr.org

about whether the inference chain as a whole is reliable. It should be clear how little the evidence in this article covers. Apart from checking that the stated recurrence calculations are correct, the numerical illustration does nothing, and at no point do we train or compare a learned model. Semantic OOD performance and physical-system safety are both out of reach for the synthetic task we propose. Since the main protocol freezes the backbone and fixes the deployment parameters, end-to-end learning and continual adaptation it should be studied in separate experiments. A fair account of computational cost has to include the monitor and the fallback estimator. Counting only the small regulatory vector would understate it.

Nothing about the interface entitles it to the final word in hazardous settings constraints justified on independent grounds have to take precedence over it. Anyone deploying it should keep a record of every override and decide in advance what happens if the monitor itself goes down. A further risk comes from the coordinate names. We chose them to be intuitive, yet that same intuitiveness could tempt a user into treating the regulatory state as a calibrated account of every way the system might fail, which it is not. Finally, the human oversight and certification a deployment needs will vary from one application to the next, and our guarantees do not extend to them.

CONCLUSION

In this article the Mevronic Neural Networks is set as a design pattern for controlling reliability regimes, one whose internal state is tied to declared meanings. The reference version has several working parts, three bounded memories of evidence, two controls derived from them, bounded modulation of features, explicit fallback logic, and an inference schedule that respects causality. For the regulatory module we prove three things, namely invariance, fading-memory sensitivity and how it behaves when it crosses a threshold. That is where the formal results stop. The open question is a practical one. Given the same information and the same resources, do these restrictions buy a better operational trade-off than generic modulation or an external supervisor? The evaluation we propose is designed to answer exactly that, and it also spells out the results that would tell us to simplify the MNN interpretation, or drop it altogether. How far the architecture reaches in the end should be settled by what the experiments show. The names we happened to give its state shouldn't decide it.

DECLARATIONS

Data and computational availability. Everything analytical can be reproduced directly from the equations and parameter values given in the manuscript. For the synthetic benchmark described, it is provided a standalone Python implementation that generates all of it: the data arrays and event schedules, plus manifests of the random seeds, the normalization statistics, and metadata recording the software versions used. We have run this generation procedure in full and checked the output, which comes to 9000 trajectories. The training, calibration and comparative evaluation laid out in Sections 7.2–7.5 have not been carried out yet. Accordingly, this article reports no trained model checkpoints and no comparative performance results.

REFERENCES

1. S. Chirayil Nandakumar, D. Mitchell, M. S. Erden, D. Flynn, and T. Lim, "Anomaly detection methods in autonomous robotic missions," *Sensors*, vol. 24, no. 4, Art. no. 1330, 2024, doi: 10.3390/s24041330.
2. J. Yang *et al.*, "OpenOOD: Benchmarking generalized out-of-distribution detection," in *Advances in Neural Information Processing Systems*, vol. 35, 2022, doi: 10.52202/068431-2362.
3. S. Lu, Y. Wang, L. Sheng, L. He, A. Zheng, and J. Liang, "Out-of-distribution detection: A task-oriented survey of recent advances," *ACM Computing Surveys*, vol. 58, no. 2, Art. no. 37, 2025, doi: 10.1145/3760390.
4. S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997, doi: 10.1162/neco.1997.9.8.1735.
5. A. Vaswani *et al.*, "Attention is all you need," in *Advances in Neural Information Processing Systems*, vol. 30, 2017, pp. 5998–6008.
6. E. Perez, F. Strub, H. de Vries, V. Dumoulin, and A. Courville, "FiLM: Visual reasoning with a general conditioning layer," in *Proc. AAAI Conf. Artificial Intelligence*, vol. 32, no. 1, 2018, pp. 3942–3951, doi: 10.1609/aaai.v32i1.11671.
7. D. Ha, A. Dai, and Q. V. Le, "HyperNetworks," arXiv:1609.09106, 2016, doi: 10.48550/arXiv.1609.09106.
8. J. C. Costacurta, S. Bhandarkar, D. Zoltowski, and S. W. Linderman, "Structured flexibility in recurrent neural networks via neuromodulation," in *Advances in Neural Information Processing Systems*, vol. 37, 2024, doi: 10.52202/079017-0062.

International Journal of Current Science Research and Review
ISSN: 2581-8341
Volume 09 Issue 10 October 2026
DOI: 10.47191/ijcsrr/V9-i10-27, Impact Factor: 8.048
IJCSRR @ 2026
www.ijcsrr.org

9. T. Miconi, K. Stanley, and J. Clune, "Differentiable plasticity: Training plastic neural networks with backpropagation," in *Proc. 35th Int. Conf. Machine Learning*, PMLR, vol. 80, 2018, pp. 3559–3568.
10. C. Guo, G. Pleiss, Y. Sun, and K. Q. Weinberger, "On calibration of modern neural networks," in *Proc. 34th Int. Conf. Machine Learning*, PMLR, vol. 70, 2017, pp. 1321–1330.
11. D. T. Phan, R. Grosu, N. Jansen, N. Paoletti, S. A. Smolka, and S. D. Stoller, "Neural Simplex Architecture," arXiv:1908.00528, 2019, revised Mar. 2020, doi: 10.48550/arXiv.1908.00528.
12. Stoyanov, K. (2024). An algorithm for the construction of improved training sets for neural networks. In 2024 IEEE 12th International Conference on Intelligent Systems (IS), pp. 1-10.

Cite this Article: Metodiev, V., Vasilev, I. (2026). Mevronic Neural Networks for Explicit Reliability Regime Control. A Semantically Anchored Architecture with Analytical Properties and an Evaluation Protocol. International Journal of Current Science Research and Review, 9(10), pp. 5485-5499. DOI: https://doi.org/10.47191/ijcsrr/V9-i10-27