# DEVELOPMENTAL WORLD MODEL: A CHILD-MACHINE FRAMEWORK FOR HETEROGENEOUS WORLD DYNAMICS

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

World models are commonly learned for one environment or through a shared parameterization across related tasks. We study heterogeneous worlds whose mature dynamics can differ substantially while following a common organization of how state evolves, behavior is generated, events change state, observations are produced, and autonomous rollout proceeds. Inspired by Turing's child-machine distinction, we introduce Developmental World Model (DWM), which supplies this event-time organization before fitting and learns a separate numerical realization for each world. Development moves from teacher-supported fitting toward autonomous generation, with candidate stages selected by autonomous prediction and a fixed-event-path contraction certificate. A final post-hoc affine-logit calibration adjusts predictive probabilities after autonomous trajectory generation. The mathematical analysis connects finite-dimensional behavioral prediction to the DWM event-time form, establishes contraction along realized event paths, and derives a Wasserstein contraction condition for autonomous transition kernels. Experiments across sequential, temporal, multi-agent, and embodied environments show that the same dynamical organization can develop distinct world-specific dynamics and state semantics. Code: `https://anonymous.4open.science/r/DWM-29D4/`.

## 1 INTRODUCTION

World models learn how environments evolve from experience and support prediction, planning, and control across domains from visual games to continuous control (Ha & Schmidhuber, 2018; Hafner et al., 2019; 2020; Schrittwieser et al., 2020; Hansen et al., 2024; Hafner et al., 2025). Most work learns one environment or reuses a parameterized model across related tasks. We study heterogeneous-world modeling under a common structural organization: the organization is fixed before fitting, while each environment develops its own mature numerical dynamics and state semantics.

Turing's child-machine proposal separates an initial organization from the mature machine shaped by education and experience (Turing, 1950). This provides the developmental idea behind DWM. The mathematical construction is a hybrid state-space realization whose predictive structure and stability are analyzed through realization and contraction theory. DWM instantiates the supplied organization as an event-time process that couples latent evolution, behavior generation, event-driven state change, observation, and autonomous rollout. Connections to world models, adaptive dynamics, and structured dynamical learning are discussed in Appendix A.

Let $\mathcal{G}$ denote the supplied dynamical organization and let $\theta_w$ denote the parameters learned in world $w$. A mature world is written as

$$\mathcal{M}_w = \mathcal{M}(\mathcal{G}, \theta_w). \tag{1}$$

The state evolves continuously between events, generates the current behavior, changes when that behavior is realized, and then produces the observation. During autonomous rollout, generated behavior drives the next state transition, so the model evolves on the trajectory created by its own predictions. Development shapes $\theta_w$ within the supplied organization as training moves from teacher-supported acquisition toward autonomous generation.



Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

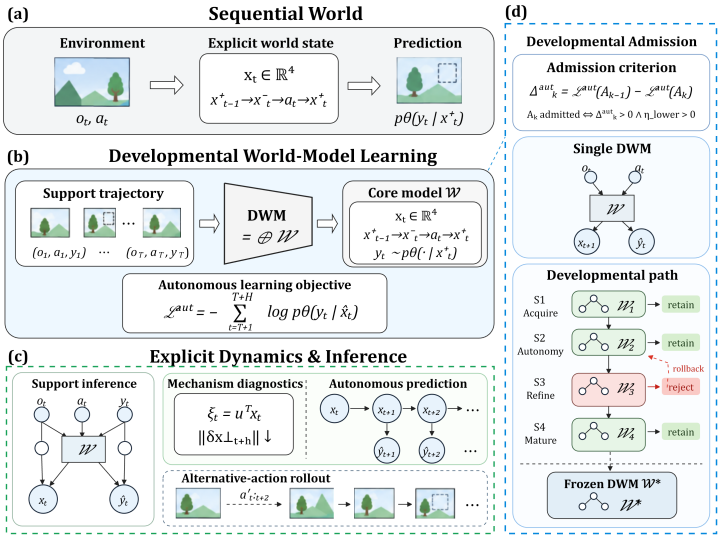



Figure 1: **Developmental World Model (DWM).** (a) A sequential world is represented by an explicit predictive state. (b) DWM learns world-specific dynamics from observed trajectories. (c) The mature model supports explicit event-time inference and autonomous rollout. (d) Development retains stages according to predictive improvement and the dynamical certificate.

Cross-environment methods differ in what they share. Meta-learning and adaptive dynamics reuse parameterizations across tasks (Finn et al., 2017; 2019; Nagabandi et al., 2019; Lee et al., 2020); TrajWorld pretrains a predictive model across heterogeneous control environments (Yin et al., 2025); and M3W shares dynamics modules across multi-agent tasks (Zhao et al., 2025). DWM places the commonality in the dynamical organization and developmental rule, leaving the mature numerical realization to each world. Its explicit transition and stability structure connects to work on structured dynamics, stable evolution, and invariance (Brunton et al., 2016; Chen et al., 2018; Manek & Kolter, 2019; Lagemann et al., 2024).

## 2 Developmental World Model

The child-machine idea motivates a separation between a supplied initial organization and a mature world-specific realization (Turing, 1950). DWM gives this separation a dynamical form. Let $\mathcal{G}$ denote the organization shared across environments, including the latent state, flow and event-update families, causal order, rollout rule, stability criterion, and developmental procedure. For world $w$, $\theta_w \in \Theta_{\mathcal{G}}$ contains the numerical parameters learned from that world's data. The mature world model is

$$\mathcal{M}_w = \mathcal{M}(\mathcal{G}, \theta_w). \tag{2}$$

Learning determines the world-specific numerical realization $\theta_w$ within this shared form. Every environment uses the same initialization rule and developmental procedure, while its mature parameters are determined from the corresponding data. Given $\mathcal{D}_w$, let $\theta_w^{(0)} = \mathrm{Init}(\mathcal{D}_w; \mathcal{G})$. The mature parameters are obtained as

$$\theta_w^\star = \mathrm{Develop}\left(\mathcal{G}, \theta_w^{(0)}, \mathcal{D}_w\right). \tag{3}$$

Different environments can therefore produce $\theta_{w_1}^\star \neq \theta_{w_2}^\star$ while retaining the same latent dimension, event-time order, dynamical families, inference rule, developmental procedure, and stability criterion. The common structural organization is $\mathcal{G}$; the mature parameters and task-facing state semantics are world specific.

DWM represents the latent state of world $w$ by

$$X_t^{(w)} \in \mathbb{R}^d. \tag{4}$$

The general formulation allows any finite $d$. In all experiments we use the fixed setting

$$d = 4. \tag{5}$$

This is the common experimental state dimension across environments with different behavior spaces, interaction structures, and observation interfaces.

### 2.1 Hybrid World Process

Each DWM event cycle follows the same causal order: the latent state evolves between events, generates the current behavior, is updated by that realized behavior, and then produces an observation. For two consecutive events separated by $\Delta_t$, the pre-event state is

$$X_t^- = e^{A\Delta_t} X_{t-1}^+ + \left(I - e^{A\Delta_t}\right)\mu, \tag{6}$$

where $X_{t-1}^+$ is the state immediately after the previous event and $X_t^-$ is the state immediately before the current one. The matrix $A$ governs continuous evolution, while $\mu$ defines its reference state.

The pre-event state generates behavior. For categorical actions, DWM uses a reference-category affine-logit channel,

$$z_{t,0} = 0, \qquad z_{t,a} = \beta_a + S_a X_t^- + C_a c_t, \tag{7}$$

with

$$P_\theta(a_t = a \mid X_t^-, c_t) = \mathrm{softmax}(z_t)_a. \tag{8}$$

Here $c_t$ contains contextual information available before event $t$. The latent state carries the evolving history of the world, while $c_t$ supplies event-specific information that need not be stored dynamically.

Once behavior is realized, it changes the latent state through

$$X_t^+ = \left[M_0 + \sum_\ell \phi_{t,\ell} M_\ell\right] X_t^- + b_0 + \sum_\ell \phi_{t,\ell} b_\ell. \tag{9}$$

The features $\phi_{t,\ell}$ encode the realized event. In the categorical cross-world instantiations used in our experiments, the action-dependent matrix terms are set to zero, yielding

$$X_t^+ = M_0 X_t^- + b_0 + b_{a_t}. \tag{10}$$

Partial observations are incorporated through the task-facing measurement model. The event-time order is therefore

$$X_{t-1}^+ \to X_t^- \to a_t \to X_t^+ \to Y_t. \tag{11}$$

The state first generates behavior. That behavior then changes the state from which subsequent evolution begins. This ordering is preserved in every environment.

**Theorem 2.1 (Event-Time Realization with Stable Flow).** Within the fixed-step feature-affine process class, an admissible stable affine flow can be combined with a transformed feature-affine jump and affine-logit behavior channel while preserving the induced finite-horizon behavior–observation law. The formal assumptions, construction, and proof are given in Appendix C.4.

The theorem shows that a stable continuous flow can represent the same event-time law when the jump and behavior coordinates are transformed consistently. At a single event interval, the split between flow and jump is not unique. Appendix C.5 characterizes this equivalence class and gives temporal conditions for identification; Appendix C.6 characterizes the accompanying balance between continuous contraction and event amplification.

A complementary result is given in Appendix C.2. Under finite closure of continued behavior log-odds, a deterministic feature-affine, affine-logit predictive realization exists with a well-defined minimal dimension. Appendix C.3 distinguishes this realization dimension from probability-Hankel rank.

### 2.2 Inference and Autonomous Generation

The latent state is only partially observed. DWM maintains a particle approximation to the filtering distribution and propagates each particle through the same flow, behavior, jump, and measurement sequence. The observation interface can vary across environments, including its dimensionality; the experiments keep the latent dynamical state at $d = 4$. Formal observation-dimension and causal non-anticipation properties are developed in Appendix C.7.

During an observed support prefix, realized actions and available observations update the posterior over latent states. At the beginning of the query period, the process becomes autonomous. At query step $t$,

$$\widehat{a}_t \sim P_\theta\left(\cdot \mid \widehat{\mathcal{H}}_{t-1}\right), \tag{12}$$

and the generated action drives the next state jump. The model therefore constructs its own future history,

$$X_t \to \widehat{a}_t \to X_{t+1} \to \widehat{a}_{t+1} \to \cdots. \tag{13}$$

Recorded query actions are retained as scoring targets and are not fed back into the query trajectory; query observations are excluded from filtering.

The primary autonomous objective is

$$\mathcal{L}_{\text{auto}}(\theta) = -\frac{1}{|\mathcal{Q}|}\sum_{t\in\mathcal{Q}} \log P_\theta\left(a_t \mid \widehat{\mathcal{H}}_{t-1}\right), \tag{14}$$

This objective is the mean autonomous NLL per query event, in nats. Here $\widehat{\mathcal{H}}_{t-1}$ contains the model-generated query history. The sampled behavior $\widehat{a}_t$ determines the state transition and subsequent conditioning history, while $a_t$ enters the log-loss term. Early rollout errors can therefore alter later latent states and predictions, so horizon-dependent performance probes the learned dynamics.

The hybrid process is constrained by

$$\mu_w(A) \le -\lambda, \qquad e^{-\lambda(t-s)} \prod_{j\in\mathcal{J}(s,t)} \rho_j \le M e^{-\eta(t-s)}, \tag{15}$$

where $\mu_w(A)$ measures contraction of the continuous flow and $\rho_j$ bounds the sensitivity introduced by event $j$.

**Theorem 2.2 (Fixed-Event-Path Incremental Stability).** If Eq. (15) holds, then two trajectories driven by the same realized event path and disturbance realization satisfy

$$|X(t) - \widetilde{X}(t)|_w \le M e^{-\eta(t-s)} |X(s) - \widetilde{X}(s)|_w. \tag{16}$$

The proof and an average-dwell-time sufficient condition are given in Appendix C.11.

Theorem 2.2 supplies the contraction certificate used during development. Autonomous generation induces a transition kernel in which the latent state also changes future behavior probabilities and action-conditioned jumps. Appendix C.11.2 gives the one-step coefficient $\kappa_{\text{auto}} = L_h(L_J + L_\pi D_Q)$; $\kappa_{\text{auto}} < 1$ yields Wasserstein contraction of the autonomous kernel. Appendix C.11.1 gives the corresponding fixed-path and autonomous formulations.

### 2.3 Development from an Initial Organization

Equation (3) defines development within one world. Starting from $\theta_w^{(0)}$, training moves from teacher-supported acquisition toward autonomous generation. After each stage, cross-fitting evaluates autonomous likelihood together with the fixed-event-path contraction certificate in Eq. (15). An accepted stage is replayed on the full training set for its selected duration and becomes the parent of the next stage; otherwise development continues from the current parent.

The data determine the mature parameters within the same latent dimension, flow–jump organization, behavior-before-jump ordering, rollout semantics, and stability criterion.

**Theorem 2.3 (Developmental Closure).** Suppose each initialized model belongs to the certificate-eligible DWM class and every accepted developmental update satisfies the same certificate. Then every accepted developmental state, including the mature realization $\theta^{\star}_{w}$, remains inside that fixed-event-path-certified class. The developmental operator, acceptance rule, and proof are given in Appendix C.12.2.

## 3 Experiments

All experiments use the same support–query semantics: an observed prefix initializes the latent state, after which the query rollout is autonomous. We evaluate predictive accuracy across sequential and temporal tasks, analyze learned state organization in multi-agent and embodied environments, and examine the developmental progression from teacher-supported fitting to autonomous generation. Unless otherwise stated, autonomous NLL is the mean negative log-probability per query event, computed with natural logarithms and therefore reported in nats per query event. Protocols and additional results are provided in Appendix D.

### 3.1 Standard Sequential Prediction

We evaluate DWM on Public Goods (Cox & Stoddard, 2021), DeliData (Karadzhov et al., 2023), and PersuasionForGood (P4G) (Wang et al., 2019), with two, three, and 37 behavior classes, respectively. Baselines include GRU (Cho et al., 2014), LSTM (Hochreiter & Schmidhuber, 1997), Transformer (Vaswani et al., 2017), Mamba-2 (Dao & Gu, 2024), TD-MPC2 (Hansen et al., 2024), and DreamerV3 (Hafner et al., 2025). Frequency and first-order Markov predictors are included for DeliData and P4G. Dataset and evaluation settings are given in Appendix D.2, with model configurations in Appendix D.3.

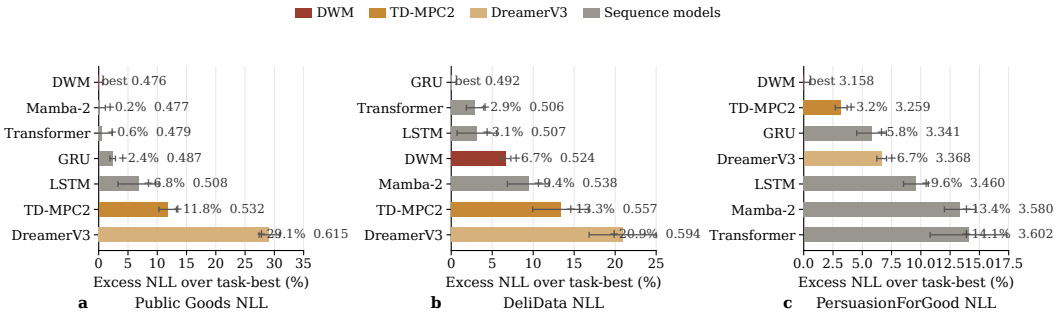



Figure 2: **Autonomous prediction on the standard sequential tasks.** Test autonomous NLL is reported for Public Goods, DeliData, and PersuasionForGood; lower is better. Stochastic-model values are means across five independent seeds; sample standard deviations are reported in Table 1.

Table 1: **Test autonomous NLL on the standard sequential prediction tasks.** Stochastic methods are mean ± sample standard deviation across five independent seeds; lower is better.

| Model | Public Goods | DeliData | PersuasionForGood |
|---|---|---|---|
| DWM | $\mathbf{0.4761 \pm 0.0032}$ | $0.5244 \pm 0.0030$ | $\mathbf{3.1579 \pm 0.0162}$ |
| Frequency | – | 0.5516 | 3.3508 |
| Markov | – | 0.5485 | 3.2422 |
| GRU | $0.4875 \pm 0.0023$ | $\mathbf{0.4916 \pm 0.0030}$ | $3.3407 \pm 0.0400$ |
| LSTM | $0.5085 \pm 0.0168$ | $0.5070 \pm 0.0119$ | $3.4604 \pm 0.0334$ |
| Transformer | $0.4787 \pm 0.0085$ | $0.5059 \pm 0.0052$ | $3.6022 \pm 0.1035$ |
| Mamba-2 | $0.4771 \pm 0.0045$ | $0.5378 \pm 0.0125$ | $3.5796 \pm 0.0425$ |
| TD-MPC2 | $0.5322 \pm 0.0071$ | $0.5572 \pm 0.0167$ | $3.2589 \pm 0.0160$ |
| DreamerV3 | $0.6148 \pm 0.0083$ | $0.5944 \pm 0.0201$ | $3.3681 \pm 0.0131$ |

Figure 2 and Table 1 show different model orderings across the three tasks. DWM gives the lowest NLL on Public Goods and P4G, while GRU gives the lowest NLL on DeliData.

**Additional temporal environments.** We additionally evaluate DWM on compatible bsuite environments (Osband et al., 2020). K2, K3, and K5 contain multi-event trajectories that can be evaluated using the same support–query protocol. Data conversion and task details are given in Appendix D.4.

Table 2: **Test autonomous NLL across additional temporal environments.** Lower is better.

| Task | Obs. dim. | Test eps. | Length | Test NLL ↓ |
|---|---|---|---|---|
| K2 | 2506 | 72 | 3–100 | $0.69242 \pm 0.00014$ |
| K3 | 60 | 120 | 9–128 | $1.08365 \pm 0.00029$ |
| K5 | 3 | 12 | 100 | $1.44052 \pm 0.00012$ |

Table 2 reports test autonomous NLL using the same fixed dynamical-state setting in all three environments.

### 3.2 Interpreting the Learned State

We analyze frozen out-of-fold development models for Melting Pot and strict scene-LOSO development models for PARTNR. Principal directions of the posterior-mean state trajectories are related to the fitted behavior and measurement maps, action-conditioned state changes, and continuous flow. Fold-level analyses are reported in Appendix D.6.1.

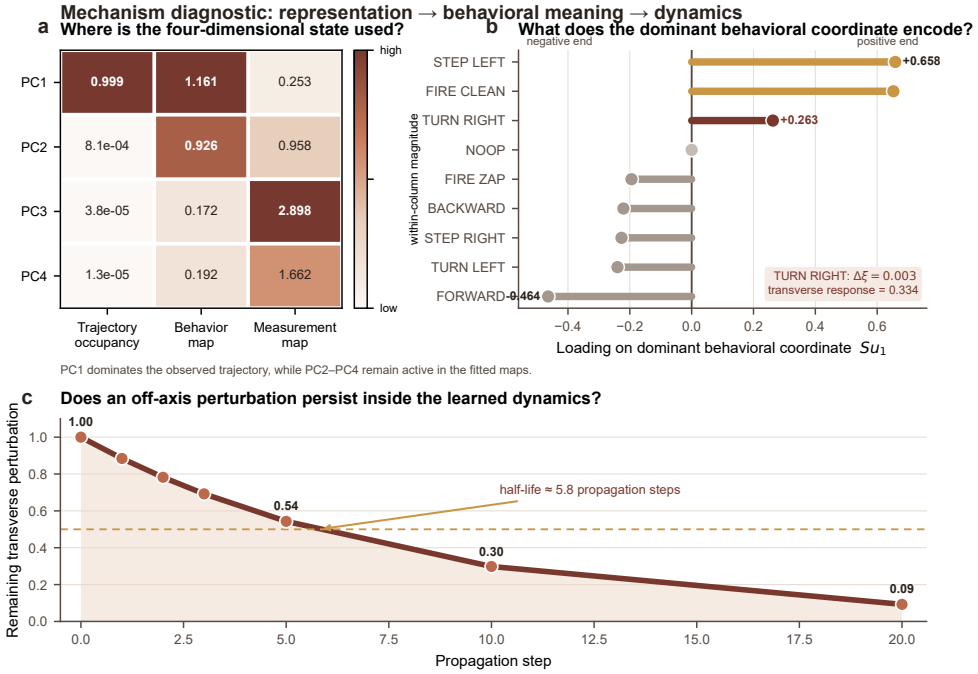



Figure 3: **Internal structure of the learned DWM state in Melting Pot.** (a) State geometry and functional responses. (b) Behavioral-logit loadings along the dominant coordinate, with selected state-response annotations. (c) Propagation of transverse perturbations.

The state acquires a different behavioral meaning in each environment. In Melting Pot, the dominant direction organizes cleaning, movement, and interaction responses; in PARTNR, the corresponding direction separates manipulation from navigation. The remaining coordinates enter the fitted behav-

ior and measurement maps and participate in transverse event responses. Figure 3 shows these state, jump, and propagation patterns, with action-level and fold-level results reported in Appendix D.6.1.

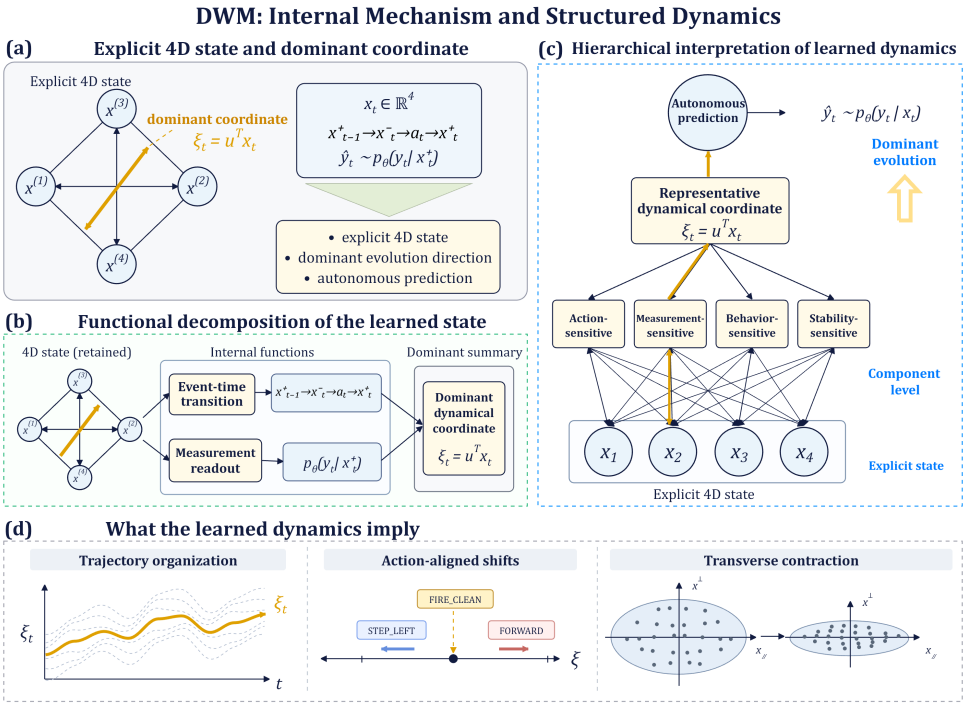



Figure 4: **Mechanistic view of the DWM state.** Within each mature world, the learned coordinates organize behavior, measurement, event-driven change, and autonomous prediction.

### 3.3 High-Dimensional Observation Interfaces

DWM is also evaluated with higher-dimensional observation interfaces in Melting Pot (Leibo et al., 2021) and PARTNR (Chang et al., 2025). Melting Pot uses 64 low-frequency DCT coefficients derived from local $88 \times 88 \times 3$ RGB observations, while PARTNR uses a 120-dimensional multimodal representation. Both environments use the same fixed dynamical-state setting. Full observation interfaces and evaluation protocols are given in Appendix D.5.

Table 3: **Test autonomous NLL with high-dimensional observation interfaces.** Values are mean $\pm$ sample standard deviation across three independent evaluation seeds; lower is better.

| Environment | Model | Observation | Test queries | Test NLL ↓ |
|---|---|---|---|---|
| Melting Pot | DWM | local RGB → DCT-64 | 72,000 | $\mathbf{1.30894 \pm 0.0152}$ |
| Melting Pot | DreamerV3 | DCT-64 | 72,000 | $1.37106 \pm 0.02448$ |
| PARTNR | DWM | 120-D multimodal | 633 | $\mathbf{1.35116 \pm 0.00008}$ |
| PARTNR | DreamerV3 | 120-D multimodal | 633 | $2.15450 \pm 0.00832$ |

Table 3 reports final autonomous test NLL, with DWM lower than DreamerV3 in both environments. Observation ablations and horizon-specific diagnostics are reported in Appendix D.5.

### 3.4 Developmental Learning

For each environment, DWM develops its dynamics through stages S1–S4. Candidate stages are selected on development folds using autonomous prediction and the fixed-event-path contraction certificate. Held-out development diagnostics trace the retained path, and final test evaluation is performed separately. The full procedure and stage-wise results are given in Appendix D.6.

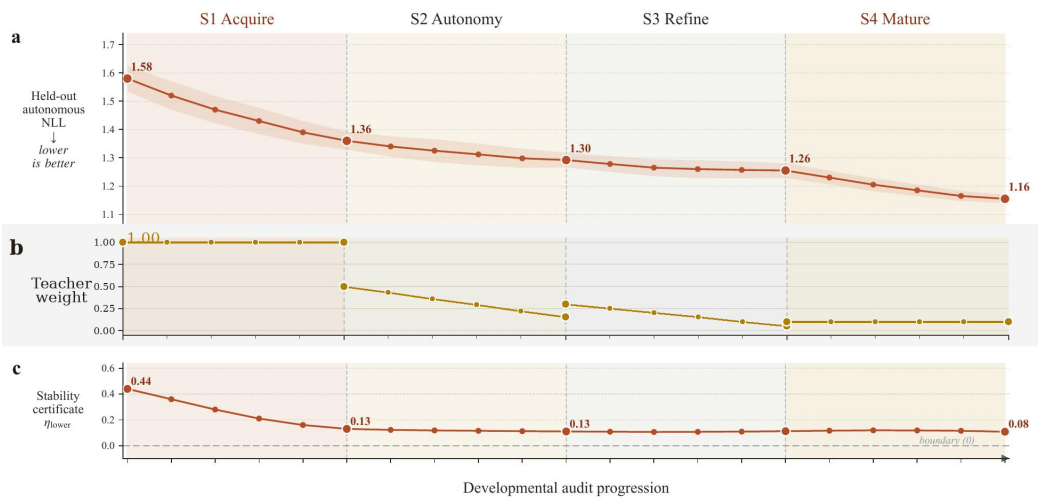



Figure 5: **Developmental audit trajectory for Melting Pot.** Held-out five-fold development autonomous NLL, stage-local teacher weight, and the lower fixed-event-path contraction rate $\eta_{\text{lower}}$ across stages S1–S4. Here $\eta_{\text{lower}}$ is the largest rate $\eta$ for which the fixed-event-path certificate in Eq. (15) holds over all audited held-out event paths. NLL is pooled over held-out development query events.

Figure 5 traces held-out development prediction, teacher support, and the fixed-event-path contraction rate through S1–S4. The retained path is world specific: Melting Pot and PARTNR retain all four stages, while P4G retains S1, S2, and S4.

Table 4: **Ablation on Public Goods.** Test autonomous NLL is reported as mean ± sample standard deviation across five independent seeds; lower is better.

| Variant | Development | Calibration | Test NLL ↓ |
|---|---|---|---|
| Full DWM | S1–S4 | affine | $\mathbf{0.476087 \pm 0.003192}$ |
| w/o affine calibration | S1–S4 | identity | $0.487607 \pm 0.000684$ |
| w/o development | direct fitting | identity | $0.490294 \pm 0.000667$ |

The last two rows isolate S1–S4 development under identity calibration. The first two rows isolate affine calibration after development.

## 4 Scientific Interpretation of the Mature Dynamics

DWM exposes a mature world through an explicit event-time state whose coordinates enter behavior generation, event-driven state change, observation, and continuous evolution. The functional meaning of a learned direction is determined jointly by trajectory geometry, the fitted behavior and measurement maps, and its propagation through the learned dynamics.

**State geometry and behavioral organization.** Posterior-state principal directions identify where mature trajectories concentrate, and the fitted behavior map quantifies how movement along those directions changes action logits. In Melting Pot, the dominant direction organizes cleaning, movement, and interaction responses; in PARTNR, the corresponding direction separates manipulation-oriented behavior from navigation. The remaining directions continue to enter the behavior and measurement maps, giving the four-dimensional state functional structure beyond its dominant trajectory axis. Figure 3 reports these relationships.

**Event-driven state change.** Each realized behavior induces a structured displacement through the event-update map. Projection onto the learned state directions separates longitudinal changes along the dominant behavioral coordinate from transverse responses that also enter measurement

and subsequent prediction. The behavior, event-update, and measurement maps therefore attach distinct functional roles to the learned coordinates.

**Dynamical propagation.** Propagation under the fitted continuous flow shows how off-axis state information persists between events. In Melting Pot, transverse perturbations decay over successive propagation steps while remaining present over a finite horizon. This links the geometry of the learned representation to its temporal role inside autonomous rollout.

Figure 4 summarizes the resulting mechanism: state geometry organizes the learned representation, fitted maps connect its directions to behavior and observation, and the flow controls how those directions influence later evolution. Melting Pot and PARTNR develop different state semantics and response patterns within the same event-time organization.

## 5 Discussion

The current formulation fixes the initial dynamical organization across worlds and learns a separate mature numerical realization from each environment. A direct extension is to learn selected components of the event-time structure across environments while retaining a common stability-constrained core. Such a model would separate persistent cross-world dynamical relations from world-specific mechanisms within the same structural class.

**Development under autonomous dynamics.** The developmental procedure combines autonomous predictive performance with a fixed-event-path contraction certificate. Section 2 also derives an autonomous-kernel condition that includes state-dependent behavior probabilities and action-conditioned transitions. Incorporating this condition directly into stage promotion would align developmental selection with stability of the trajectories generated by the model.

**Functional organization of the learned state.** The learned state is characterized jointly by trajectory geometry and its role in behavior, measurement, and event-driven change. The experiments show that a direction with limited trajectory variance can remain active through these fitted maps and through subsequent dynamical propagation. Extending this analysis across independently developed worlds could reveal whether recurring functional directions emerge within the shared dynamical organization.

**Scaling explicit world dynamics.** The experiments use an explicit low-dimensional dynamical state with visual and multimodal observation interfaces. Learned encoders, nonlinear event maps, and richer continuous dynamics provide a path to higher-capacity models while retaining the event-time decomposition of behavior, state change, observation, and temporal propagation. The central scaling problem is to preserve inspectable dynamical structure as representation learning and world dynamics become more expressive.

## 6 Conclusion

We introduced Developmental World Model (DWM), an event-time world-model class in which a supplied dynamical organization develops into a world-specific mature realization. The model makes behavior part of the evolving world process: state generates behavior, realized behavior changes the state, and autonomous rollout continues from the resulting trajectory. Its theory connects predictive realization with stable hybrid dynamics, while the experiments show competitive autonomous prediction across heterogeneous environments and expose distinct mature state semantics through the same explicit dynamical form. The resulting picture is a world model whose commonality lies in how dynamics are organized and developed, while the learned mechanisms remain specific to the world that produced them.

### AI use statement

Generative AI tools were used to assist with language polishing and with checking and refining parts of the mathematical proofs. All mathematical claims, derivations, proofs, and final manuscript text

were reviewed and verified by the authors. The authors take full responsibility for the correctness and content of the paper.

### Reproducibility statement

The assumptions and complete proofs underlying the theoretical results are provided in Appendix C. Model specifications, developmental procedures, dataset construction and preprocessing, experimental protocols, baseline configurations, optimization settings, ablations, and additional numerical results are documented in Appendix D. Anonymous source code and experiment configurations are provided as supplementary material to support reproduction of the reported results.

## References

Eloi Alonso, Adam Jelley, Vincent Micheli, Anssi Kanervisto, Amos Storkey, Tim Pearce, and Francois Fleuret. Diffusion for world modeling: Visual details matter in atari. In *Advances in Neural Information Processing Systems*, volume 37, pp. 58757–58791, 2024.

M. Sanjeev Arulampalam, Simon Maskell, Neil Gordon, and Tim Clapp. A tutorial on particle filters for online nonlinear/non-gaussian bayesian tracking. *IEEE Transactions on Signal Processing*, 50 (2):174–188, 2002.

Peter W. Battaglia, Razvan Pascanu, Matthew Lai, Danilo Jimenez Rezende, and Koray Kavukcuoglu. Interaction networks for learning about objects, relations and physics. In *Advances in Neural Information Processing Systems*, volume 29, 2016.

Samy Bengio, Oriol Vinyals, Navdeep Jaitly, and Noam Shazeer. Scheduled sampling for sequence prediction with recurrent neural networks. In *Advances in Neural Information Processing Systems*, volume 28, 2015.

Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. Curriculum learning. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pp. 41–48, 2009. doi: 10.1145/1553374.1553380.

Jake Bruce, Michael D. Dennis, Ashley Edwards, Jack Parker-Holder, Yuge Shi, Edward Hughes, Matthew Lai, Aditi Mavalankar, Richie Steigerwald, Chris Apps, Yusuf Aytar, Sarah Maria Elisabeth Bechtle, Feryal Behbahani, Stephanie C. Y. Chan, Nicolas Heess, Lucy Gonzalez, Simon Osindero, Sherjil Ozair, Scott Reed, Jingwei Zhang, Konrad Zolna, Jeff Clune, Nando de Freitas, Satinder Singh, and Tim Rocktäschel. Genie: Generative interactive environments. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235, pp. 4603–4623, 2024.

Steven L. Brunton, Joshua L. Proctor, and J. Nathan Kutz. Discovering governing equations from data by sparse identification of nonlinear dynamical systems. *Proceedings of the National Academy of Sciences*, 113(15):3932–3937, 2016.

Matthew Chang, Gunjan Chhablani, Alexander Clegg, Mikael Dallaire Cote, Ruta Desai, Michal Hlavac, Vladimir Karashchuk, Jacob Krantz, Roozbeh Mottaghi, Priyam Parashar, Siddharth Patki, Ishita Prasad, Xavier Puig, Akshara Rai, Ram Ramrakhya, Daniel Tran, Joanne Truong, John Turner, Eric Undersander, and Tsung-Yen Yang. PARTNR: A benchmark for planning and reasoning in embodied multi-agent tasks. In *International Conference on Learning Representations*, 2025.

Ricky T. Q. Chen, Yulia Rubanova, Jesse Bettencourt, and David Duvenaud. Neural ordinary differential equations. In *Advances in Neural Information Processing Systems*, volume 31, 2018.

Kyunghyun Cho, Bart van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using RNN encoder–decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 1724–1734, 2014.

Caleb A. Cox and Brock Stoddard. Common-value public goods and informational social dilemmas. *American Economic Journal: Microeconomics*, 13(2):343–369, 2021. doi: 10.1257/mic.20180275.

Tri Dao and Albert Gu. Transformers are SSMs: Generalized models and efficient algorithms through structured state space duality. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235, pp. 10041–10071, 2024.

Sergey Dashkovskiy and Vitalii Slynko. Dwell-time stability conditions for infinite dimensional impulsive systems. *Automatica*, 147:110695, 2023.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70, pp. 1126–1135, 2017.

Chelsea Finn, Aravind Rajeswaran, Sham Kakade, and Sergey Levine. Online meta-learning. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97, pp. 1920–1930, 2019.

Carles Gelada, Saurabh Kumar, Jacob Buckman, Ofir Nachum, and Marc G. Bellemare. Deepmdp: Learning continuous latent space models for representation learning. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97, pp. 2170–2179, 2019.

Sam Greydanus, Misko Dzamba, and Jason Yosinski. Hamiltonian neural networks. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

David Ha and Jurgen Schmidhuber. Recurrent world models facilitate policy evolution. In *Advances in Neural Information Processing Systems*, volume 31, 2018.

Danijar Hafner, Timothy Lillicrap, Ian Fischer, Ruben Villegas, David Ha, Honglak Lee, and James Davidson. Learning latent dynamics for planning from pixels. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97, pp. 2555–2565, 2019.

Danijar Hafner, Timothy Lillicrap, Jimmy Ba, and Mohammad Norouzi. Dream to control: Learning behaviors by latent imagination. In *International Conference on Learning Representations*, 2020.

Danijar Hafner, Jurgis Pasukonis, Jimmy Ba, and Timothy Lillicrap. Mastering diverse control tasks through world models. *Nature*, 640:647–653, 2025.

Nicklas Hansen, Hao Su, and Xiaolong Wang. Td-mpc2: Scalable, robust world models for continuous control. In *International Conference on Learning Representations*, 2024.

Joao P. Hespanha, Daniel Liberzon, and Andrew R. Teel. Lyapunov conditions for input-to-state stability of impulsive systems. *Automatica*, 44(11):2735–2744, 2008.

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Computation*, 9(8): 1735–1780, 1997.

Junteng Jia and Austin R. Benson. Neural jump stochastic differential equations. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

Leslie Pack Kaelbling, Michael L. Littman, and Anthony R. Cassandra. Planning and acting in partially observable stochastic domains. *Artificial Intelligence*, 101(1–2):99–134, 1998.

Georgi Karadzhov, Tom Stafford, and Andreas Vlachos. DeliData: A dataset for deliberation in multi-party problem solving. *Proceedings of the ACM on Human-Computer Interaction*, 7 (CSCW2):1–25, 2023.

Maximilian Karl, Maximilian Soelch, Justin Bayer, and Patrick van der Smagt. Deep variational bayes filters: Unsupervised learning of state space models from raw data. In *International Conference on Learning Representations*, 2017.

Thomas Kipf, Elise van der Pol, and Max Welling. Contrastive learning of structured world models. In *International Conference on Learning Representations*, 2020.

Thomas N. Kipf, Ethan Fetaya, Kuan-Chieh Wang, Max Welling, and Richard Zemel. Neural relational inference for interacting systems. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80, pp. 2688–2697, 2018.

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A. Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, Demis Hassabis, Claudia Clopath, Dharshan Kumaran, and Raia Hadsell. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences*, 114(13):3521–3526, 2017.

Kai Lagemann, Christian Lagemann, and Sach Mukherjee. Invariance-based learning of latent dynamics. In *International Conference on Learning Representations*, 2024.

Alex M. Lamb, Anirudh Goyal, Ying Zhang, Saizheng Zhang, Aaron C. Courville, and Yoshua Bengio. Professor forcing: A new algorithm for training recurrent networks. In *Advances in Neural Information Processing Systems*, volume 29, 2016.

Kimin Lee, Younggyo Seo, Seunghyun Lee, Honglak Lee, and Jinwoo Shin. Context-aware dynamics model for generalization in model-based reinforcement learning. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119, pp. 5757–5766, 2020.

Joel Z. Leibo, Edgar A. Duenez-Guzman, Alexander Vezhnevets, John P. Agapiou, Peter Sunehag, Raphael Koster, Jayd Matyas, Charlie Beattie, Igor Mordatch, and Thore Graepel. Scalable evaluation of multi-agent reinforcement learning with melting pot. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139, pp. 6187–6199, 2021.

Michael L. Littman, Richard S. Sutton, and Satinder Singh. Predictive representations of state. In *Advances in Neural Information Processing Systems*, volume 14, 2001.

Yuren Liu, Biwei Huang, Zhengmao Zhu, Honglong Tian, Mingming Gong, Yang Yu, and Kun Zhang. Learning world models with identifiable factorization. In *Advances in Neural Information Processing Systems*, volume 36, pp. 31831–31864, 2023.

David Lopez-Paz and Marc'Aurelio Ranzato. Gradient episodic memory for continual learning. In *Advances in Neural Information Processing Systems*, volume 30, 2017.

Bethany Lusch, J. Nathan Kutz, and Steven L. Brunton. Deep learning for universal linear embeddings of nonlinear dynamics. *Nature Communications*, 9:4950, 2018.

Gaurav Manek and J. Zico Kolter. Learning stable deep dynamics models. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

John Miller and Moritz Hardt. Stable recurrent models. In *International Conference on Learning Representations*, 2019.

Anusha Nagabandi, Ignasi Clavera, Simin Liu, Ronald S. Fearing, Pieter Abbeel, Sergey Levine, and Chelsea Finn. Learning to adapt in dynamic, real-world environments through meta-reinforcement learning. In *International Conference on Learning Representations*, 2019.

Ian Osband, Yotam Doron, Matteo Hessel, John Aslanides, Eren Sezener, Andre Saraiva, Katrina McKinney, Tor Lattimore, Csaba Szepesvari, Satinder Singh, Benjamin Van Roy, Richard S. Sutton, David Silver, and Hado van Hasselt. Behaviour suite for reinforcement learning. In *International Conference on Learning Representations*, 2020.

Kate Rakelly, Aurick Zhou, Chelsea Finn, Sergey Levine, and Deirdre Quillen. Efficient off-policy meta-reinforcement learning via probabilistic context variables. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97, pp. 5331–5340, 2019.

Yulia Rubanova, Ricky T. Q. Chen, and David Duvenaud. Latent ordinary differential equations for irregularly-sampled time series. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

Alvaro Sanchez-Gonzalez, Jonathan Godwin, Tobias Pfaff, Rex Ying, Jure Leskovec, and Peter W. Battaglia. Learning to simulate complex physics with graph networks. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119, pp. 8459–8468, 2020.

Julian Schrittwieser, Ioannis Antonoglou, Thomas Hubert, Karen Simonyan, Laurent Sifre, Simon Schmitt, Arthur Guez, Edward Lockhart, Demis Hassabis, Thore Graepel, Timothy Lillicrap, and David Silver. Mastering atari, go, chess and shogi by planning with a learned model. *Nature*, 588: 604–609, 2020.

Alan M. Turing. Computing machinery and intelligence. *Mind*, 59(236):433–460, 1950.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30, 2017.

Tongzhou Wang, Simon S. Du, Antonio Torralba, Phillip Isola, Amy Zhang, and Yuandong Tian. Denoised MDPs: Learning world models better than the world itself. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162, pp. 22591–22612, 2022.

Xuewei Wang, Weiyan Shi, Richard Kim, Yoojung Oh, Sijia Yang, Jingwen Zhang, and Zhou Yu. Persuasion for good: Towards a personalized persuasive dialogue system for social good. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 5635–5649, 2019.

Alistair White, Niki Kilbertus, Maximilian Gelbrecht, and Niklas Boers. Stabilized neural differential equations for learning dynamics with explicit constraints. In *Advances in Neural Information Processing Systems*, volume 36, 2023.

Maciej Wolczyk, Michal Zajac, Razvan Pascanu, Lukasz Kucinski, and Piotr Milos. Continual world: A robotic benchmark for continual reinforcement learning. In *Advances in Neural Information Processing Systems*, volume 34, 2021.

Zhongwen Xu, Hado van Hasselt, and David Silver. Meta-gradient reinforcement learning. In *Advances in Neural Information Processing Systems*, volume 31, 2018.

Shaofeng Yin, Jialong Wu, Siqiao Huang, Xingjian Su, Xu He, Jianye Hao, and Mingsheng Long. Trajectory world models for heterogeneous environments. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 72462–72484. PMLR, 2025.

Friedemann Zenke, Ben Poole, and Surya Ganguli. Continual learning through synaptic intelligence. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70, pp. 3987–3995, 2017.

Xiaoyuan Zhang, Chengdong Ma, Yizhe Huang, Weidong Huang, Siyuan Qi, Song-Chun Zhu, Xue Feng, and Yaodong Yang. World models should prioritize the unification of physical and social dynamics. In *Advances in Neural Information Processing Systems*, volume 38, 2025.

Zijie Zhao, Zhongyue Zhao, Kaixuan Xu, Yuqian Fu, Jiajun Chai, Yuanheng Zhu, and Dongbin Zhao. Learning and planning multi-agent tasks via an moe-based world model. In *Advances in Neural Information Processing Systems*, volume 38, 2025.

Luisa M. Zintgraf, Kyriacos Shiarlis, Maximilian Igl, Sebastian Schulze, Yarin Gal, Katja Hofmann, and Shimon Whiteson. Varibad: A very good method for bayes-adaptive deep rl via meta-learning. In *International Conference on Learning Representations*, 2020.

# A Extended Related Work

## A.1 World Models

World models learn an internal representation of how an environment evolves and use that representation to predict, imagine, plan, or act. Early latent-state approaches already separated the state needed for prediction from the raw observation itself. Deep Variational Bayes Filters learn nonlinear state-space models directly from high-dimensional observations (Karl et al., 2017), while DeepMDP studies latent representations that preserve the transition and reward structure relevant to decision

making (Gelada et al., 2019). This distinction between observation and predictive state became central to modern world models. Recurrent World Models use compact latent dynamics as an internal environment for policy learning (Ha & Schmidhuber, 2018); PlaNet learns stochastic latent dynamics from pixels and plans inside the learned state space (Hafner et al., 2019); Dreamer learns behavior from trajectories imagined by the latent model rather than repeatedly interacting with the external environment (Hafner et al., 2020). MuZero goes further in separating predictive state from reconstruction: its learned model is required to preserve quantities useful for planning, but not to reproduce every observable detail (Schrittwieser et al., 2020). These works established a view of world modeling in which the latent state is defined less by resemblance to the observation than by the future computations it supports.

A second line of work concerns the information retained by a predictive state. A latent state that reconstructs everything may preserve nuisance variation, whereas a representation tailored too narrowly to one objective may discard structure needed elsewhere. Denoised MDPs formalize this tension by separating controllable and reward-relevant information from nuisance factors (Wang et al., 2022). IFactor develops the idea further by factorizing latent state according to its relations with actions and rewards and establishing block-wise identifiability (Liu et al., 2023). Structured World Models take another route, representing the world as objects and relations rather than as a single undifferentiated latent vector (Kipf et al., 2020). These approaches show that compactness depends on both latent dimension and the organization imposed on the state, including which aspects of the environment survive compression.

The scope of world modeling has expanded at the same time. DreamerV3 demonstrates that one algorithm and configuration can solve more than 150 tasks spanning markedly different observations, actions, rewards, and domains (Hafner et al., 2025). TD-MPC2 similarly seeks a scalable world-model architecture across large collections of continuous-control tasks and also studies multi-task learning within a single large agent (Hansen et al., 2024). TrajWorld pretrains a single predictive model across heterogeneous control environments with varying sensor and actuator spaces (Yin et al., 2025). These results make generality itself part of the world-modeling problem. Yet generality can be realized in several ways. One learning algorithm may work across many independently trained tasks; one parameterized model may serve several tasks simultaneously; or a model may share only selected components across related dynamics. M3W makes the last possibility explicit. It finds that multi-agent tasks exhibit uneven similarity in their underlying dynamics and uses a mixture-of-experts world model to share dynamics modules where such similarity is useful (Zhao et al., 2025). Thus, moving from a single environment to many environments does not lead to one unique notion of what should be shared.

Large generative world models broaden the problem along another axis. Genie learns interactive, action-controllable environments from large-scale video (Bruce et al., 2024), while DIAMOND uses diffusion-based prediction to retain visually important details during imagined rollouts (Alonso et al., 2024). Here the challenge is not primarily to discover a minimal state, but to model increasingly rich observable worlds with enough temporal coherence to support interaction. Recent work has also argued that world models should eventually encompass social as well as physical dynamics (Zhang et al., 2025). This extension is consequential because an interacting agent changes the future not only through externally supplied control inputs. Its behavior may itself be generated by the current world state and, once realized, become part of the mechanism that changes that state.

DWM follows this view of a world as an evolving interaction process. In most control world models, the learned transition is organized around an action-conditioned prediction, schematically $p(z_{t+1} \mid z_t, a_t)$. In DWM, the pre-event state first induces a distribution over behavior; the realized behavior then changes the state from which subsequent evolution proceeds. Behavior is thus both a prediction target and an endogenous input to future dynamics.

### A.2 Structured Dynamical Learning

The internal form of a dynamical model has long been treated as scientifically meaningful rather than merely computational. Sparse identification of nonlinear dynamics searches for a small set of active terms in a candidate library and thereby recovers explicit governing equations from observations (Brunton et al., 2016). Neural ODEs preserve the differential-equation form while replacing the vector field with a learned function (Chen et al., 2018); Latent ODEs extend continuous-time latent

dynamics to irregularly observed sequences (Rubanova et al., 2019). Koopman-based methods seek coordinates in which nonlinear dynamics admit simpler evolution laws (Lusch et al., 2018), while Hamiltonian Neural Networks incorporate conservation structure directly into the learned dynamics (Greydanus et al., 2019). These approaches differ in how strongly they constrain the model, but they share the premise that the organization of temporal change can be part of what is learned and interpreted.

Structure also enters through the decomposition of a world into interacting components. Interaction Networks model objects and pairwise interactions explicitly (Battaglia et al., 2016); Neural Relational Inference recovers latent interaction graphs from trajectories (Kipf et al., 2018); graph-network simulators use local relational computation to predict large physical systems (Sanchez-Gonzalez et al., 2020). Contrastively-trained Structured World Models bring this object-relational organization directly into latent world modeling (Kipf et al., 2020). In these models, structure is not an auxiliary explanation added after training. It specifies what the latent state is composed of and how the components are allowed to influence one another.

A different structural distinction concerns how change unfolds in time. Continuous-time models describe evolution through a flow, but many real processes also contain abrupt events. Neural Jump Stochastic Differential Equations combine continuous latent evolution with state-dependent stochastic events and discrete jumps (Jia & Benson, 2019). This establishes flow–jump decomposition as a general language for learned hybrid systems. It also exposes an important modeling choice: the mathematical existence of a jump does not determine its semantic role. A jump may represent an external shock, an observed event, a failure, a transaction, or the consequence of an agent's behavior. What matters for a world model is how the event is generated and how it becomes coupled to subsequent evolution.

For interactive systems, that coupling creates a closed temporal mechanism. The state before an event carries information about what an agent is likely to do. The realized behavior then changes the state from which the next period begins. Consequently, the transition cannot always be understood as a passive response to an externally specified action sequence. The distribution over actions and the consequences of those actions belong to the same evolving system. This distinction becomes especially important in repeated strategic, social, and multi-agent environments, where the action sequence itself is one of the quantities the model must generate during rollout.

Once flow and events are composed, long-horizon behavior depends on more than one-step predictive accuracy. Stability-constrained learning addresses related concerns from several directions. Stable Recurrent Models study recurrent architectures whose state evolution remains contractive (Miller & Hardt, 2019); Lyapunov-based neural dynamics impose stability through properties of the learned evolution law (Manek & Kolter, 2019); stabilized neural differential equations introduce explicit stability constraints into continuous-time learning (White et al., 2023). Invariance provides another structural principle: latent dynamics can be organized so that physically or scientifically meaningful transformations leave the relevant evolution law unchanged (Lagemann et al., 2024). These results show that constraints on dynamics can concern not only prediction at the next step, but also the geometry and asymptotic behavior of repeated application of the learned model.

For hybrid systems, however, stability belongs to the composition of the continuous and discrete parts. A contracting flow may still be followed by an expansive jump, and the sequence of events may itself depend on the current state. The relevant dynamical object is therefore an event-time process rather than an isolated vector field or transition matrix. This distinction is particularly important under autonomous generation, because discrepancies in the state can alter the distribution of future behaviors, which in turn alters the sequence of subsequent jumps.

DWM adopts this event-time view by coupling between-event flow, state-dependent behavior, behavior-induced jumps, and partial observation in a fixed causal order. Its stability analysis acts on the resulting hybrid dynamics, including the additional sensitivity introduced when autonomous behavior changes the distribution of subsequent events.

### A.3 Adaptation and Development Across Worlds

Learning across heterogeneous environments has most often been formulated as a problem of adaptation. Meta-learning assumes that tasks share enough statistical structure that experience on a task

distribution can improve learning on a new task. MAML places that commonality in an initialization from which task-specific parameters can be reached with a small number of gradient updates (Finn et al., 2017); online meta-learning extends the same principle to tasks encountered sequentially (Finn et al., 2019). Meta-gradient methods move one level higher by learning parameters of the learning process itself (Xu et al., 2018). In meta-reinforcement learning, PEARL infers a probabilistic context describing the current task (Rakelly et al., 2019), while VariBAD maintains a latent belief over task uncertainty within a Bayes-adaptive formulation (Zintgraf et al., 2020). Across these approaches, previous experience shapes either the starting point, the update rule, or the representation through which the current task is inferred.

When what changes is specifically the environment dynamics, the distinction becomes sharper. Nagabandi et al. meta-learn a dynamics prior that can be rapidly adapted when physical properties, terrain, payload, or embodiment change (Nagabandi et al., 2019). CaDM separates adaptation into two components: a context encoder summarizes recent experience into a latent variable describing local dynamics, and a shared forward model predicts transitions conditioned on that context (Lee et al., 2020). This decomposition is conceptually important. Variation across environments is not treated as unstructured model error; it becomes a latent quantity to be inferred and supplied to the predictor. M3W arrives at a related conclusion from the multi-task side: dynamics may be similar within some task groups and different across others, so modular sharing can be more natural than a single undifferentiated transition model (Zhao et al., 2025). These methods show several ways in which heterogeneous dynamics can coexist with a common learning system.

Continual learning asks a different question. Once a learner has acquired one task, how should new experience modify the model without erasing what came before? Elastic Weight Consolidation restricts changes to parameters important for previous tasks (Kirkpatrick et al., 2017); Synaptic Intelligence accumulates an online estimate of parameter importance (Zenke et al., 2017); Gradient Episodic Memory constrains new updates using stored experience (Lopez-Paz & Ranzato, 2017). Continual World brings these questions into sequential robotic reinforcement learning, where transfer to later tasks and retention of earlier capabilities can be studied together (Wolczyk et al., 2021). Here the central object is a learner whose history extends across tasks.

In DWM, development occurs during the acquisition of a world: a common initial organization is progressively shaped into a mature world-specific dynamics. The training path also moves from teacher-supported sequence learning toward autonomous generation. Teacher forcing trains a sequential model under histories containing observed previous outputs, whereas deployment exposes the model to histories partly generated by its own predictions. Scheduled Sampling addresses this discrepancy by gradually replacing observed previous outputs with model-generated ones during training (Bengio et al., 2015). Professor Forcing instead encourages the hidden dynamics induced by teacher-forced and free-running operation to become similar (Lamb et al., 2016). Both works recognize that a sequence model can behave differently when it must inhabit the trajectory generated by its own predictions rather than trajectories anchored at every step by observed outcomes.

This issue is especially consequential for a world model. During autonomous rollout, an erroneous prediction does not merely affect the next output. It changes the history from which the model generates the next behavior, and that behavior can change the latent state from which later predictions are made. The training-to-generation transition is therefore also a transition between two dynamical regimes. A developmental schedule that moves from externally supported acquisition toward autonomous operation can be understood as shaping the model under progressively more of the distribution that its own dynamics induce.

Turing's child-machine proposal provides the developmental separation used in DWM: an initial organization is supplied before learning, while education and experience shape the mature machine (Turing, 1950). Curriculum learning later formalized a related intuition in optimization: the sequence through which training examples are encountered can alter the learning process and the solution reached (Bengio et al., 2009). DWM instantiates this separation between the structure supplied before a particular world is learned and the dynamical realization produced from that world's experience.

# B Architecture Details

This appendix gives the architectural specification of the Developmental World Model introduced in Section 2. We first describe how an observed event sequence is converted into a predictive learning problem, then give the explicit state space formulation of the model, and finally describe the concrete realizations used for inference, autonomous rollout, development, and the structural variants evaluated in the experiments.

## B.1 Learning Scheme Details

### B.1.1 Episode Formulation

We represent each environment as a collection of event sequences. An episode is written as

$$\mathcal{E} = (e_1, \ldots, e_T), \tag{17}$$

where event $e_t$ contains the elapsed time $\Delta_t$, the context $c_t$, the behavior $a_t$, and an observation $Y_t$ when available.

DWM associates the observed history with a latent state

$$X_t \in \mathbb{R}^d. \tag{18}$$

The same latent dimension is used throughout the experiments,

$$d = 4. \tag{19}$$

At event $t$, the model first constructs the state immediately before behavior is generated. The resulting behavior distribution has the general form

$$P_\theta\left(a_t \mid X_t^-, c_t\right). \tag{20}$$

The realized behavior then enters the state update and determines the state from which the next event begins.

The episode is divided into an observed support segment and a query segment. Let

$$\mathcal{S} = \{1, \ldots, T_s\}, \qquad \mathcal{Q} = \{T_s + 1, \ldots, T\}. \tag{21}$$

The support segment is used to infer the latent state distribution. The query segment is generated recursively from the model.

### B.1.2 Support Inference and Autonomous Prediction

**Support inference.** During support, observed behaviors and available observations are incorporated sequentially. Let $\mathcal{H}_t$ denote the observed history through event $t$. The corresponding filtering distribution is

$$q_t(X) = p_\theta\left(X_t^+ \mid \mathcal{H}_t\right). \tag{22}$$

At each support event, the latent state first undergoes continuous evolution. The behavior likelihood is then evaluated from the resulting pre event state. After the observed behavior has been incorporated, the same behavior enters the event map. Process disturbance and the observation update follow in that order.

**Autonomous prediction.** At the first query event, the filtering distribution obtained from support becomes the initial latent distribution for generation. Let $\widehat{\mathcal{H}}_{t-1}$ denote the support history together with the behaviors generated during previous query events. The model produces

$$\widehat{p}_t(a) = P_\theta\left(a \mid \widehat{\mathcal{H}}_{t-1}, c_t, \Delta_t\right) \tag{23}$$

and samples

$$\widehat{a}_t \sim \widehat{p}_t. \tag{24}$$

The generated behavior enters the event map and produces the latent state used by the next query event. Recursive application gives

$$X_t^- \to \widehat{a}_t \to X_t^+ \to X_{t+1}^- \to \widehat{a}_{t+1} \to \cdots . \tag{25}$$

The recorded query behavior is used to evaluate the predictive probability,

$$\ell_t^{\text{auto}} = -\log \widehat{p}_t(a_t). \tag{26}$$

The recorded query behavior enters only this score. The state sequence is generated from the sampled behaviors in Equation (24).

### B.1.3 Learning Objectives

**Observed objective.** For observed training events, the behavioral loss is

$$\mathcal{L}_{\text{beh}} = -\frac{1}{N_{\text{beh}}} \sum_{t \in \mathcal{I}_{\text{beh}}} \log P_\theta \left( a_t \mid X_t^-, c_t \right), \tag{27}$$

where $\mathcal{I}_{\text{beh}}$ denotes the corresponding set of behavior events.

When an observation channel is present, its likelihood contributes

$$\mathcal{L}_{\text{meas}} = -\frac{1}{N_Y D_Y} \sum_{t \in \mathcal{I}_Y} \log p_\theta \left( Y_t \mid X_t^+ \right), \tag{28}$$

where $D_Y$ is the observation dimension. We write

$$\mathcal{L}_{\text{obs}} = \mathcal{L}_{\text{beh}} + \mathcal{L}_{\text{meas}}, \tag{29}$$

with the measurement term omitted when no separate observation channel is used.

**Autonomous objective.** The autonomous objective evaluates predictions along the generated query trajectory:

$$\mathcal{L}_{\text{auto}} = -\frac{1}{|\mathcal{Q}|} \sum_{t \in \mathcal{Q}} \log P_\theta \left( a_t \mid \widehat{\mathcal{H}}_{t-1}, c_t, \Delta_t \right). \tag{30}$$

This is the mean autonomous NLL per query event, in nats.

At developmental stage $s$, the fitting objective is

$$\mathcal{L}_s = \omega_s \mathcal{L}_{\text{obs}} + (1 - \omega_s)\, \mathcal{L}_{\text{auto}} + \lambda_{\text{cert}} \mathcal{R}_{\text{cert}}. \tag{31}$$

The coefficient $\omega_s$ controls the relative contribution of observed and autonomous learning, while $\mathcal{R}_{\text{cert}}$ is associated with the stability condition developed in Appendix C.11.

### B.1.4 Developmental Learning

The complete procedure consists of initialization, dynamical development, and post-hoc scoring calibration:

$$S0 \to S1 \to S2 \to S3 \to S4 \to S5. \tag{32}$$

Here $S0$ initializes the model, $S1$–$S4$ form the dynamical developmental sequence, and $S5$ applies post-hoc scoring calibration. Their roles are:

- $S0$**: Initialization.** The initial parameterization is constructed for the current environment using the common DWM initialization rule.
- $S1$**: Observed acquisition.** Learning is driven by observed behavior and measurement likelihood.
- $S2$**: Initial autonomous development.** Autonomous prediction enters the learning objective and generated event effects are represented through the soft transition.
- $S3$**: Autonomous refinement.** The contribution of autonomous prediction is increased while the same model equations are retained.

- $S4$**: Categorical maturation.** Generated categorical events are used directly in the forward transition.
- $S5$**: Post-hoc scoring calibration.** Predictive probabilities are calibrated after autonomous trajectory generation through an affine transformation of their natural parameters; the calibrated probabilities are used for scoring and are not fed back into the rollout.

For environment $r$, initialization is written as

$$\theta_r^{(0)} = \mathrm{Init}\left(\mathcal{D}_r^{\mathrm{train}}; \mathfrak{W}\right), \tag{33}$$

where $\mathfrak{W}$ denotes the common DWM specification. The mature model is

$$\theta_r^{\star} = \mathrm{Develop}\left(\mathcal{D}_r^{\mathrm{train}}; \theta_r^{(0)}, \mathfrak{W}\right). \tag{34}$$

### B.2 Model Structure Details

#### B.2.1 Continuous Flow

The latent state evolves continuously between interaction events. For elapsed time $\Delta_t$, define

$$\Phi_t = e^{A\Delta_t}, \qquad d_t = \left(I - e^{A\Delta_t}\right)\mu. \tag{35}$$

The pre event state is then

$$X_t^- = \Phi_t X_{t-1}^+ + d_t. \tag{36}$$

Here $A \in \mathbb{R}^{d\times d}$ determines the continuous dynamics and $\mu \in \mathbb{R}^d$ is the corresponding reference state. In the implementation, $A$ is diagonal,

$$A = \mathrm{diag}\left(a_1, \ldots, a_d\right), \tag{37}$$

so the flow of coordinate $j$ can be written as

$$X_{t,j}^- = e^{a_j\Delta_t} X_{t-1,j}^+ + \left(1 - e^{a_j\Delta_t}\right)\mu_j. \tag{38}$$

This form is used directly in the state propagation step.

#### B.2.2 Behavior Models

**Categorical behavior.** For a categorical variable with $K$ possible behaviors, category $0$ is used as the reference. Set

$$z_{t,0} = 0. \tag{39}$$

For $a = 1, \ldots, K-1$,

$$z_{t,a} = \beta_a + S_a X_t^- + C_a c_t. \tag{40}$$

The corresponding probability is

$$P_\theta\left(a_t = a \mid X_t^-, c_t\right) = \frac{\exp(z_{t,a})}{\sum_{j=0}^{K-1}\exp(z_{t,j})}. \tag{41}$$

The nonreference logits therefore satisfy

$$\log\frac{P_\theta(a_t = a \mid X_t^-, c_t)}{P_\theta(a_t = 0 \mid X_t^-, c_t)} = \beta_a + S_a X_t^- + C_a c_t. \tag{42}$$

**Bernoulli behavior.** For a binary component $m$, define

$$z_{t,m} = \beta_m + S_m X_t^- + C_m c_t. \tag{43}$$

The conditional probability is

$$P_\theta\left(a_{t,m} = 1 \mid X_t^-, c_t\right) = \sigma(z_{t,m}). \tag{44}$$

The categorical and Bernoulli forms use the same pre event state and differ only in the output probability map.

### B.2.3 Event Maps

Let

$$\phi_t = (\phi_{t,1}, \ldots, \phi_{t,p}) \tag{45}$$

denote the feature vector associated with the realized event. The general event map is

$$\bar{X}_t^+ = M(\phi_t) X_t^- + b(\phi_t), \tag{46}$$

where

$$M(\phi_t) = M_0 + \sum_{\ell=1}^{p} \phi_{t,\ell} M_\ell \tag{47}$$

and

$$b(\phi_t) = b_0 + \sum_{\ell=1}^{p} \phi_{t,\ell} b_\ell. \tag{48}$$

For the categorical instantiation used in the experiments, the action dependent matrix terms are zero. The resulting event map is

$$\bar{X}_t^+ = M_0 X_t^- + b_0 + b_{a_t}. \tag{49}$$

The matrix $M_0$ acts on the pre event state, while $b_{a_t}$ gives the additive contribution associated with the realized category.

### B.2.4 Process and Observation Models

After the deterministic event map, process disturbance is added:

$$X_t^+ = \bar{X}_t^+ + \xi_t, \qquad \xi_t \sim \mathcal{N}(0, Q). \tag{50}$$

When a separate observation is available, the observation model is

$$Y_t = H X_t^+ + \nu_t, \qquad \nu_t \sim \mathcal{N}(0, R). \tag{51}$$

The corresponding observation density is

$$p_\theta\left(Y_t \mid X_t^+\right) = \mathcal{N}\left(Y_t; H X_t^+, R\right). \tag{52}$$

Combining the components above gives the complete event calculation

$$X_t^- = e^{A\Delta_t} X_{t-1}^+ + \left(I - e^{A\Delta_t}\right)\mu, \tag{53}$$

$$a_t \sim P_\theta\left(\cdot \mid X_t^-, c_t\right), \tag{54}$$

$$\bar{X}_t^+ = J_{e_t}\left(X_t^-\right), \tag{55}$$

$$X_t^+ = \bar{X}_t^+ + \xi_t, \tag{56}$$

$$Y_t \sim \mathcal{N}\left(H X_t^+, R\right). \tag{57}$$

## B.3 Implementation Details

### B.3.1 Function Instantiation

**Default instantiation.** We use the diagonal flow in Equation (37), the affine behavior models in Equations (40) and (43), and the categorical event map in Equation (49). The resulting categorical DWM block can be written as

$$X_t^- = e^{A\Delta_t} X_{t-1}^+ + \left(I - e^{A\Delta_t}\right)\mu, \tag{58}$$

$$z_{t,a} = \beta_a + S_a X_t^- + C_a c_t, \tag{59}$$

$$a_t \sim \mathrm{softmax}(z_t), \tag{60}$$

$$\bar{X}_t^+ = M_0 X_t^- + b_0 + b_{a_t}, \tag{61}$$

$$X_t^+ = \bar{X}_t^+ + \xi_t. \tag{62}$$

This is the default categorical realization used for DeliData, PersuasionForGood, Melting Pot, and PARTNR. Public Goods uses the corresponding Bernoulli behavior map while retaining the same latent flow and event organization.

**Observation variants.** The model uses two observation forms according to the environment:

- **Behavior only.** The observed behavior likelihood supplies the information used by the latent filter.
- **Behavior with measurement.** The behavior likelihood is followed by the observation density in Equation (52).

The observation map may therefore change in dimension across environments, while its mathematical role in the event calculation remains the same.

### B.3.2 Particle Filtering and Autonomous Generation

**Particle representation.** We approximate the filtering distribution by

$$q_t(X) \approx \sum_{n=1}^{P} w_t^{(n)} \delta_{X_t^{(n)}}(X), \qquad \sum_{n=1}^{P} w_t^{(n)} = 1. \tag{63}$$

This is the standard sequential Monte Carlo representation of a latent filtering distribution (Arulampalam et al., 2002).

At a support event, each particle is propagated through the continuous flow. Its behavior likelihood gives

$$\widetilde{w}_{t,a}^{(n)} = w_{t-1}^{(n)} P_\theta \left( a_t \mid X_t^{-,(n)}, c_t \right), \tag{64}$$

which is normalized across particles. The observed behavior is then applied through the event map and process disturbance is added. When a measurement is present, the weights are further multiplied by

$$p_\theta \left( Y_t \mid X_t^{+,(n)} \right). \tag{65}$$

**Autonomous generation.** For query event $t$, particle $n$ produces

$$p_t^{(n)}(a) = P_\theta \left( a \mid X_t^{-,(n)}, c_t \right). \tag{66}$$

The predictive probability is the weighted mixture

$$\widehat{p}_t(a) = \sum_{n=1}^{P} w_t^{(n)} p_t^{(n)}(a). \tag{67}$$

One categorical behavior is then generated for each particle,

$$\widehat{a}_t^{(n)} \sim \text{Categorical} \left( p_t^{(n)} \right), \tag{68}$$

and the particle follows

$$\bar{X}_t^{+,(n)} = M_0 X_t^{-,(n)} + b_0 + b_{\widehat{a}_t^{(n)}}. \tag{69}$$

After process disturbance, the new particle population is propagated to the next query event.

### B.3.3 Soft and Categorical Transitions

**Soft transition.** For developmental stages that use a soft categorical transition, the action contribution is replaced by its conditional expectation,

$$\bar{b}_t^{(n)} = \sum_{a=0}^{K-1} p_t^{(n)}(a) b_a. \tag{70}$$

The event map becomes

$$\bar{X}_t^{+,(n)} = M_0 X_t^{-,(n)} + b_0 + \bar{b}_t^{(n)}. \tag{71}$$

**Categorical transition.** The later developmental stage uses a sampled categorical behavior in the forward calculation,

$$\widehat{a}_t^{(n)} \sim \text{Categorical}\left(p_t^{(n)}\right), \tag{72}$$

and applies the corresponding action vector $b_{\widehat{a}_t^{(n)}}$. Optimization uses the associated straight through categorical gradient.

Thus the soft and categorical versions share the same event map. They differ only in how the action dependent vector is selected during developmental optimization.

### B.3.4 Developmental Procedure

**Stagewise development.** Stages $S1$ through $S4$ use the same model parameters and the same event equations. Their distinction lies in the learning objective and the autonomous transition used during fitting:

- $S1$**:** observed likelihood with the soft transition;
- $S2$**:** observed and autonomous objectives with the soft transition;
- $S3$**:** increased autonomous contribution with the soft transition;
- $S4$**:** observed and autonomous objectives with categorical sampled transitions.

Candidate stages are evaluated only on held-out development partitions. The standard fold-based protocol uses five-fold cross-fitting within the TRAIN/development data; PARTNR instead uses six strict scene-level leave-one-scene-out development folds so that scenes remain separated. These development partitions are used for stage selection and are distinct from the final test split. For the five-fold protocol, if $D_k$ denotes fold $k$,

$$D_k^{\text{fit}} = D^{\text{train}} \setminus D_k, \qquad D_k^{\text{meta}} = D_k. \tag{73}$$

A candidate is fitted on $D_k^{\text{fit}}$ and evaluated on $D_k^{\text{meta}}$ using autonomous likelihood. Promotion requires the held-out autonomous NLL improvement to meet the stage-specific minimum gain in Table 7 and the fixed-event-path certificate to be satisfied. The selected stage is subsequently fitted on the complete development/training data under the dataset-specific protocol and becomes the parent of the next stage. Final-test data are not used for this decision.

**Post-hoc scoring calibration.** After completion of the $S1$–$S4$ dynamical stages, $S5$ fits an affine calibration of the predictive natural parameters using out-of-fold TRAIN predictions. For categorical behavior,

$$z_a^{\text{cal}} = \alpha z_a + \gamma_a, \qquad a = 1, \ldots, K-1, \qquad \alpha > 0, \tag{74}$$

with

$$z_0^{\text{cal}} = 0. \tag{75}$$

The calibrated probabilities are

$$p_a^{\text{cal}} = \frac{\exp\left(z_a^{\text{cal}}\right)}{\sum_{j=0}^{K-1} \exp\left(z_j^{\text{cal}}\right)}. \tag{76}$$

For Bernoulli behavior, the corresponding transformation is

$$z^{\text{cal}} = \alpha z + \gamma. \tag{77}$$

Calibration is applied to predictive probabilities after an autonomous rollout has been generated. It does not alter the actions sampled during that rollout, the latent-state trajectory, or the event transitions. The calibration parameters are estimated from out-of-fold TRAIN predictions after completion of the dynamical stages.

# C PROOFS OF THEOREMS

This appendix develops the theoretical results used in Section 2. We begin by fixing the notation and assumptions shared by the subsequent results. We then study three questions in sequence. First, we characterize when observable behavior admits a finite dimensional affine logit predictive realization. Second, we relate the event time representation to the continuous flow and event map used by DWM and examine the identification of that decomposition. Third, we establish the stability properties of the resulting dynamics, including the autonomous transition induced when behavior is generated by the model itself. The final subsection gives the closure result for the developmental procedure.

## C.1 SETUP AND ASSUMPTIONS

Let

$$X_t \in \mathbb{R}^d \tag{78}$$

denote the latent state. The theoretical statements below are written for finite $d$; the experimental realization uses $d = 4$.

For an elapsed interval $\Delta > 0$, define the affine flow

$$h_\Delta(x) = \Phi_\Delta x + d_\Delta, \qquad \Phi_\Delta = e^{A\Delta}, \qquad d_\Delta = (I - \Phi_\Delta)\mu. \tag{79}$$

The matrix exponential is nonsingular, hence

$$\Phi_\Delta^{-1} = e^{-A\Delta}. \tag{80}$$

An event $e$ is represented by

$$\psi(e) = (1, \psi_1(e), \ldots, \psi_p(e)). \tag{81}$$

Its deterministic state update is feature affine,

$$J_e(z) = M(e)z + b(e), \tag{82}$$

where

$$M(e) = M_0 + \sum_{\ell=1}^{p} \psi_\ell(e) M_\ell, \qquad b(e) = b_0 + \sum_{\ell=1}^{p} \psi_\ell(e) b_\ell. \tag{83}$$

This form is closely related to hybrid and jump dynamical representations in which continuous evolution is interleaved with event dependent state changes (Hespanha et al., 2008; Jia & Benson, 2019).

For the categorical realization used in the principal experiments, the action dependent matrix terms are zero. If $a$ denotes the realized category,

$$M(e) = M_0, \qquad b(e) = b_0 + b_a. \tag{84}$$

Categorical behavior is written using category $0$ as the reference. Its natural parameters are

$$\eta_0(z, c) = 0, \qquad \eta_a(z, c) = \beta_a + s_a^\top z + C_a c, \qquad a = 1, \ldots, K-1, \tag{85}$$

and the corresponding probabilities are

$$\pi_a(z, c) = \frac{\exp\{\eta_a(z, c)\}}{\sum_{j=0}^{K-1} \exp\{\eta_j(z, c)\}}. \tag{86}$$

The complete event calculation follows one order. Let $\bar{X}_t^+$ denote the deterministic state after the event map and let $\xi_t$ denote process disturbance:

$$\begin{aligned} X_t^- &= h_{\Delta_t}(X_{t-1}^+), \\ a_t &\sim \pi(\cdot \mid X_t^-, c_t), \\ \bar{X}_t^+ &= J_{e_t}(X_t^-), \\ X_t^+ &= \bar{X}_t^+ + \xi_t. \end{aligned} \tag{87}$$

When a separate observation channel is present,

$$Y_t \sim \mathcal{O}\left(\cdot \mid X_t^+, c_t\right). \tag{88}$$

For the affine Gaussian measurement model,

$$Y_t = HX_t^+ + \nu_t. \tag{89}$$

Thus

$$X_{t-1}^+ \longrightarrow X_t^- \longrightarrow a_t \longrightarrow \bar{X}_t^+ \longrightarrow X_t^+ \longrightarrow Y_t. \tag{90}$$

For the stability results, $\|\cdot\|_w$ denotes the norm associated with the certificate, $\|\cdot\|_{w,*}$ its dual norm, and $\mu_w(\cdot)$ the induced matrix measure.

### C.2 Observable Predictive Realization

The latent state used by DWM is estimated from data. A separate question is whether the observable behavioral process itself admits a finite dimensional deterministic predictive realization. Predictive state representations address this general problem by representing histories through quantities that determine future observable behavior (Littman et al., 2001). Here we consider the particular realization class that matches the affine event structure and affine logit behavior model used by DWM.

Let $\mathcal{H}$ be the set of finite histories and let $\mathcal{E}$ be a finite event alphabet. Assume every legal action has strictly positive conditional probability. Choose a reference action $a_0$ and define the observable log odds

$$\ell_a(h) = \log \frac{P(a \mid h)}{P(a_0 \mid h)}, \qquad a \neq a_0. \tag{91}$$

For $e \in \mathcal{E}$, define the continuation operator

$$(U_e f)(h) = f(he). \tag{92}$$

For a finite string $v = e_1 \cdots e_m$, write

$$U_v = U_{e_1} \cdots U_{e_m}, \qquad U_\epsilon = I. \tag{93}$$

The continuation space generated by the observable log odds is

$$\mathcal{V}_\ell = \operatorname{span}\left\{1,\, U_v \ell_a : v \in \mathcal{E}^*,\, a \neq a_0\right\}. \tag{94}$$

The key requirement is finite closure of this space together with a feature affine action of event continuation.

**Assumption C.1** (Finite Predictive Closure)**.** *Suppose*

$$\dim \mathcal{V}_\ell = d + 1 < \infty, \tag{95}$$

*and, on $\mathcal{V}_\ell$, every legal event satisfies*

$$U_e = T_0 + \sum_{j=1}^{p} \psi_j(e) T_j. \tag{96}$$

The assumption states that the family of all continued log odds closes in a finite linear space and that event continuation acts through the same declared feature family. Under this condition, the predictive state can be constructed directly from a basis of observable functions.

**Theorem C.2** (Observable Logit Realization)**.** *Under Assumption C.1, there exists a deterministic predictive state*

$$x(h) \in \mathbb{R}^d \tag{97}$$

*satisfying*

$$x(he) = F(e)x(h) + g(e), \tag{98}$$

*where*

$$[F(e) \;\; g(e)] = [F_0 \;\; g_0] + \sum_{j=1}^{p} \psi_j(e)[F_j \;\; g_j]. \tag{99}$$

*For every nonreference action,*

$$\log \frac{P(a \mid h)}{P(a_0 \mid h)} = \beta_a + s_a^\top x(h). \tag{100}$$

*The associated reference category softmax reproduces the original conditional action distribution. Among deterministic affine state, affine logit realizations of the same process, the dimension $d$ is minimal.*

*Proof.* Choose a basis

$$q_0 = 1, q_1, \ldots, q_d \tag{101}$$

of $\mathcal{V}_\ell$ and define

$$x(h) = \begin{bmatrix} q_1(h) & \cdots & q_d(h) \end{bmatrix}^\top. \tag{102}$$

Since $\mathcal{V}_\ell$ is closed under every continuation operator, each $U_e q_i$ lies in the same space. Hence there are coefficients $c_{ij}(e)$ such that

$$(U_e q_i)(h) = c_{i0}(e) + \sum_{j=1}^{d} c_{ij}(e) q_j(h). \tag{103}$$

Using $(U_e q_i)(h) = q_i(he)$ and stacking the $d$ equations gives Equation (98). The operator relation in Equation (96) makes the coefficients affine in $\psi(e)$, which gives Equation (99).

Every $\ell_a$ belongs to $\mathcal{V}_\ell$. It therefore has the representation

$$\ell_a(h) = \beta_a + s_a^\top x(h). \tag{104}$$

Strict positivity of the action probabilities gives

$$P(a_0 \mid h) = \frac{1}{1 + \sum_{a \neq a_0} \exp\{\ell_a(h)\}}, \tag{105}$$

and

$$P(a \mid h) = \exp\{\ell_a(h)\} P(a_0 \mid h), \tag{106}$$

which is exactly the reference category softmax.

For minimality, consider another deterministic affine state realization $\widetilde{x}(h) \in \mathbb{R}^n$. Repeated composition of its affine transitions gives

$$\widetilde{x}(hv) = \widetilde{F}_v \widetilde{x}(h) + \widetilde{g}_v. \tag{107}$$

Every continued log odds function must consequently lie in

$$\operatorname{span}\{1, \widetilde{x}_1, \ldots, \widetilde{x}_n\}. \tag{108}$$

It follows that

$$d + 1 = \dim \mathcal{V}_\ell \leq n + 1. \tag{109}$$

Therefore $n \geq d$. □

**Corollary C.3** (Predictive Dimension)**.** *Under Assumption C.1,*

$$d_{\min} = \dim \mathcal{V}_\ell - 1. \tag{110}$$

The quantity in Corollary C.3 is the minimal dimension of the deterministic affine logit predictive realization defined above. It is an observable property of that realization class.

### C.3 Predictive Dimension and Probability Rank

Finite dimensional predictive state representations are sometimes characterized through probability Hankel rank. The realization in the previous subsection is built from continued log odds instead of probabilities themselves, and the nonlinear logit map changes the algebraic structure. The two dimensions therefore need not coincide.

**Proposition C.4** (Predictive Dimension and Probability Rank)**.** *Finite dimensional deterministic affine logit realization and finite probability Hankel rank do not imply one another in general.*

*Proof.* We establish the two directions separately.

For the first direction, consider the scalar affine system

$$X_{k+1} = \rho X_k, \qquad 0 < \rho < 1, \tag{111}$$

with Bernoulli behavior

$$P(a_k = 1 \mid X_k) = \sigma(X_k). \tag{112}$$

For deterministic $X_0 = c \neq 0$,

$$P(a_k = 1) = \sigma(c\rho^k). \tag{113}$$

Thus the behavioral process has a one dimensional affine logit state.

A finite probability linear realization would imply that the scalar sequence in Equation (113) satisfies a finite order linear recurrence. For sufficiently large $k$, analyticity of the logistic function around zero gives

$$\sigma(c\rho^k) = \frac{1}{2} + \sum_{m \in \mathcal{M}} \alpha_m c^m \rho^{mk}, \tag{114}$$

where $\mathcal{M}$ contains infinitely many positive odd integers and infinitely many $\alpha_m$ are nonzero. The generating function of the centered sequence contains terms

$$\frac{\alpha_m c^m \rho^{mK}}{1 - \rho^m z} \tag{115}$$

for infinitely many distinct values of $\rho^m$. It is therefore not rational, so the sequence cannot satisfy a finite order linear recurrence. The probability rank is consequently not forced to be finite by the one dimensional affine logit state.

For the reverse direction, let

$$\Theta \in \left\{ \frac{1}{4}, \frac{3}{4} \right\}, \qquad P\left(\Theta = \frac{1}{4}\right) = P\left(\Theta = \frac{3}{4}\right) = \frac{1}{2}, \tag{116}$$

and let the actions be conditionally independent Bernoulli$(\Theta)$. Every finite string probability is a linear combination of two product measures, so the probability Hankel rank is finite.

After observing $k$ consecutive ones,

$$P\left(\Theta = \frac{3}{4} \mid 1^k\right) = \frac{3^k}{1 + 3^k}. \tag{117}$$

The next behavior probability is

$$p_k = P(a_{k+1} = 1 \mid 1^k) = \frac{1 + 3^{k+1}}{4(1 + 3^k)}, \tag{118}$$

with log odds

$$\ell_k = \log \frac{1 + 3^{k+1}}{3 + 3^k}. \tag{119}$$

For $k \geq 2$,

$$\ell_k = \log 3 + \sum_{m=1}^{\infty} \frac{(-1)^{m+1}}{m} \left(3^{-m} - 3^m\right) 3^{-mk}. \tag{120}$$

A finite dimensional deterministic affine state would make $\ell_k$ satisfy a finite order linear recurrence by the Cayley Hamilton theorem. The expansion above contains infinitely many distinct exponential modes. Hence no such finite affine logit realization follows from finite probability rank. □

The proposition separates two notions of predictive complexity. The continued log odds dimension in Section C.2 characterizes the affine logit realization used in our theory, whereas probability Hankel rank characterizes a different linear representation of string probabilities.

### C.4 Event-Time Realization with Stable Flow

We next connect a discrete event time representation to the continuous flow and event map used by DWM. Consider the event process

$$a_k \sim \pi(\cdot \mid x_{k-1}, c_k), \qquad e_k = e(a_k, c_k), \tag{121}$$

followed by

$$x_k = F(e_k)x_{k-1} + g(e_k) + \varepsilon_k, \tag{122}$$

where

$$[F(e) \;\; g(e)] = \sum_{\ell=0}^{p} \psi_\ell(e)[F_\ell \;\; g_\ell], \qquad \psi_0 \equiv 1. \tag{123}$$

Assume a fixed event interval $\Delta > 0$. The initial state law, process disturbance law, and observation kernel are held fixed throughout the construction.

The following result gives the representation used by Theorem 2.1 in the main text.

**Theorem C.5** (Event-Time Realization with Stable Flow)**.** *Let $A$ be an admissible Hurwitz matrix and let $\mu \in \mathbb{R}^d$. Define*

$$\Phi = e^{A\Delta}, \qquad d = (I - \Phi)\mu, \tag{124}$$

*and, for each event feature component,*

$$M_\ell = F_\ell \Phi^{-1}, \qquad b_\ell = g_\ell - F_\ell \Phi^{-1} d. \tag{125}$$

*Then*

$$M(e)(\Phi x + d) + b(e) = F(e)x + g(e) \tag{126}$$

*for every admissible state $x$ and event $e$.*

*If the behavior channel is transformed according to*

$$\pi_h(a \mid z, c) = \pi\left(a \mid \Phi^{-1}(z - d), c\right), \tag{127}$$

*then a Bernoulli affine logit channel or a reference category categorical affine logit channel remains in the same family. With common initial, disturbance, and observation laws, the event process and the resulting DWM representation induce the same finite horizon behavior observation law.*

*Proof.* Since $\Phi^{-1} = e^{-A\Delta}$,

$$\begin{aligned} M(e)(\Phi x + d) + b(e) &= F(e)\Phi^{-1}(\Phi x + d) + g(e) - F(e)\Phi^{-1} d \\ &= F(e)x + g(e), \end{aligned} \tag{128}$$

which proves the state transition identity.

Consider a nonreference affine logit

$$\eta_a(x, c) = \beta_a + s_a^\top x + C_a c. \tag{129}$$

Using

$$x = \Phi^{-1}(z - d) \tag{130}$$

gives

$$\eta_a^h(z, c) = \left(\beta_a - s_a^\top \Phi^{-1} d\right) + (\Phi^{-\top} s_a)^\top z + C_a c. \tag{131}$$

The transformed natural parameter is again affine in the state and context.

To compare the finite horizon laws, use the same initial state and the same auxiliary random variables for the two constructions. Suppose their post event states agree at event $k-1$. The DWM flow gives $z_k = \Phi x_{k-1} + d$, and Equation (127) gives the same conditional behavior law. Using the same sampling variable therefore produces the same behavior and event. Equation (126) then gives the same deterministic post event state. Applying the same process disturbance and observation noise preserves equality through the remainder of the event. Induction over $k$ establishes equality of the finite horizon joint law. □

The transformation also preserves the affine organization of the event operators. Let

$$\mathcal{H}_\Delta = \begin{bmatrix} \Phi & d \\ 0 & 1 \end{bmatrix}. \tag{132}$$

Then

$$[M(e) \;\; b(e)] = [F(e) \;\; g(e)]\mathcal{H}_\Delta^{-1}. \tag{133}$$

Since $\mathcal{H}_\Delta$ is invertible, affine relations among the original event operators are preserved under the transformation.

**Corollary C.6** (Event Time Correspondence)**.** *Conversely, a fixed interval DWM with flow* $(\Phi, d)$ *and event map* $(M(e), b(e))$ *induces the event time operators*

$$F(e) = M(e)\Phi, \qquad g(e) = M(e)d + b(e). \tag{134}$$

### C.5 Flow and Jump Identification

Theorem C.5 establishes representational equivalence at a fixed interval. Identification is a different question. If only the complete event time operator is observed at one interval, its decomposition into continuous flow and event jump is generally not unique. Variation in elapsed time provides the additional information needed to identify the common generator.

**Proposition C.7** (Flow and Jump Equivalence)**.** *Fix* $\Delta > 0$ *and event time operators* $(F(e), g(e))$ *in a common latent coordinate system. For any admissible Hurwitz matrix* $A$ *and any* $\mu \in \mathbb{R}^d$*, define*

$$M(e) = F(e)e^{-A\Delta}, \tag{135}$$

*and*

$$b(e) = g(e) - F(e)e^{-A\Delta}(I - e^{A\Delta})\mu. \tag{136}$$

*Then the resulting flow and event map produce the same complete event time transition. With the behavior transformation in Equation (127), they also produce the same finite horizon behavior observation law.*

*Proof.* Substituting Equations (135) and (136) gives

$$\begin{aligned} &M(e)\left[e^{A\Delta}x + (I - e^{A\Delta})\mu\right] + b(e) \\ &\quad = F(e)x + g(e). \end{aligned} \tag{137}$$

The behavior and observation statements follow from Theorem C.5. □

The proposition identifies the complete event operator as the quantity determined at a single sampling interval. We now consider a family of intervals.

**Theorem C.8** (Timing Identification)**.** *Let* $I \subset (0, \infty)$ *be an interval. Suppose, for every event* $e$*, that* $F_e(\delta)$ *and* $g_e(\delta)$ *are continuously differentiable in* $\delta \in I$ *within a common latent coordinate system.*

*There exist a common Hurwitz matrix* $A$*, a common vector* $\mu$*, and event dependent quantities* $M_e, b_e$ *that do not depend on* $\delta$*, satisfying*

$$F_e(\delta) = M_e e^{A\delta}, \tag{138}$$

*and*

$$g_e(\delta) = M_e(I - e^{A\delta})\mu + b_e, \tag{139}$$

*if and only if there exist a common Hurwitz matrix* $A$ *and a common vector* $v$ *such that*

$$F_e'(\delta) = F_e(\delta)A, \qquad g_e'(\delta) = F_e(\delta)v \tag{140}$$

*for every* $e$ *and* $\delta \in I$*. In that case,*

$$\mu = -A^{-1}v. \tag{141}$$

*Proof.* Suppose first that Equations (138) and (139) hold. Differentiating the first equation yields

$$F_e'(\delta) = M_e e^{A\delta} A = F_e(\delta) A. \tag{142}$$

A Hurwitz matrix is nonsingular. Setting $v = -A\mu$ and differentiating the second equation gives

$$g_e'(\delta) = -M_e e^{A\delta} A\mu = F_e(\delta) v. \tag{143}$$

Conversely, assume Equation (140). Then

$$\begin{aligned} \frac{d}{d\delta}\left(F_e(\delta)e^{-A\delta}\right) &= F_e'(\delta)e^{-A\delta} - F_e(\delta)Ae^{-A\delta} \\ &= 0. \end{aligned} \tag{144}$$

Thus

$$M_e = F_e(\delta)e^{-A\delta} \tag{145}$$

is constant in $\delta$, proving Equation (138).

Define $\mu = -A^{-1}v$. Choose any $\delta_0 \in I$ and set

$$b_e = g_e(\delta_0) - M_e(I - e^{A\delta_0})\mu. \tag{146}$$

The derivative of

$$M_e(I - e^{A\delta})\mu + b_e \tag{147}$$

is $F_e(\delta)v$, which equals $g_e'(\delta)$ by assumption. The two expressions agree at $\delta_0$, hence they agree throughout $I$. □

**Corollary C.9** (Temporal Excitation)**.** *Suppose $(A, v)$ and $(\widetilde{A}, \widetilde{v})$ both satisfy Equation (140). If event interval pairs $(e_i, \delta_i)$ can be chosen so that*

$$\mathcal{F} = \begin{bmatrix} F_{e_1}(\delta_1) \\ \vdots \\ F_{e_m}(\delta_m) \end{bmatrix} \tag{148}$$

*has full column rank, then*

$$A = \widetilde{A}, \qquad v = \widetilde{v}. \tag{149}$$

*Proof.* Subtracting the two derivative identities gives

$$F_e(\delta)(A - \widetilde{A}) = 0, \qquad F_e(\delta)(v - \widetilde{v}) = 0. \tag{150}$$

Stacking the selected equations and using the full column rank of $\mathcal{F}$ gives the result. □

For two informative intervals, the same conclusion can be expressed through a matrix logarithm. If

$$F_i = Me^{A\delta_i}, \qquad i = 1, 2, \qquad \delta_1 \neq \delta_2, \tag{151}$$

and $F_1$ is nonsingular, then

$$F_1^{-1}F_2 = e^{A(\delta_2 - \delta_1)}. \tag{152}$$

Whenever the matrix logarithm is unique within the admissible generator class,

$$A = \frac{1}{\delta_2 - \delta_1} \log\left(F_1^{-1}F_2\right). \tag{153}$$

The corresponding $M$, $\mu$, and $b$ then follow from the preceding identities.

### C.6 Flow and Event Contraction Balance

The previous subsection concerns identification. We now consider the relation between a continuous flow and the complete event transition from the perspective of stability. The central object is the composed event map: continuous contraction and event amplification enter through their product.

Suppose an exact representation satisfies

$$F(e) = M(e)\Phi. \tag{154}$$

**Proposition C.10** (Flow and Event Factorization)**.** *For a realized event sequence $e_1, \ldots, e_N$, the Jacobian of the deterministic event to event composition is*

$$F(e_N) \cdots F(e_1) = M(e_N)\Phi \cdots M(e_1)\Phi. \tag{155}$$

*Proof.* The Jacobian of one complete deterministic event step is $M(e)\Phi = F(e)$. Composition over the realized sequence gives Equation (155). □

The expression makes the balance between the two components explicit. For the isotropic flow

$$A = -\alpha I, \qquad \Phi = e^{-\alpha\Delta} I, \tag{156}$$

the exact factorization gives

$$M(e) = e^{\alpha\Delta} F(e). \tag{157}$$

Consequently,

$$e^{-\alpha N\Delta} \prod_{k=1}^{N} \|M(e_k)\| = \prod_{k=1}^{N} \|F(e_k)\|. \tag{158}$$

Increasing continuous contraction therefore changes the corresponding event map in the exact representation so that the complete event operator remains the same. Stability of DWM is accordingly evaluated on the combined flow and event sequence, as in the certificate analyzed in Section C.11. This is the same general principle that appears in dwell time analyses of impulsive systems (Hespanha et al., 2008; Dashkovskiy & Slynko, 2023).

### C.7 Architectural Properties

This subsection records three properties used by the architecture and inference discussion in Section 2.2: closure of affine calibration, separation between observation dimension and latent dimension, and the causal measurability of autonomous rollout.

#### C.7.1 Affine Calibration

Suppose the nonreference natural parameters are

$$\eta_a = \beta_a + s_a^\top X + C_a c. \tag{159}$$

**Proposition C.11** (Affine Calibration)**.** *For*

$$\eta_a^{\mathrm{cal}} = \alpha\eta_a + \gamma_a, \qquad \alpha > 0, \qquad \eta_0^{\mathrm{cal}} = 0, \tag{160}$$

*the calibrated behavior channel remains in the reference category affine logit family.*

*Proof.* Substitution gives

$$\eta_a^{\mathrm{cal}} = (\alpha\beta_a + \gamma_a) + (\alpha s_a)^\top X + (\alpha C_a)c. \tag{161}$$

The calibrated natural parameter therefore has the same affine form. □

### C.7.2 Observation Dimension

Let

$$X_t^+ \in \mathbb{R}^d, \qquad Y_t \in \mathbb{R}^D, \tag{162}$$

with

$$Y_t = HX_t^+ + \nu_t, \qquad H \in \mathbb{R}^{D\times d}. \tag{163}$$

**Proposition C.12** (Observation Dimension Decoupling)**.** *Changing the observation dimension $D$, the measurement matrix $H$, the measurement noise law, or a deterministic task facing observation representation does not change the dimension or algebraic form of the latent flow and event map.*

*Proof.* The continuous parameters satisfy

$$A \in \mathbb{R}^{d\times d}, \qquad \mu \in \mathbb{R}^d, \tag{164}$$

while the event parameters satisfy

$$M_\ell \in \mathbb{R}^{d\times d}, \qquad b_\ell \in \mathbb{R}^d. \tag{165}$$

The observation dimension enters through the codomain of $H$ and through the measurement law. The latent transition therefore remains defined on $\mathbb{R}^d$. □

### C.7.3 Nonanticipative Autonomous Rollout

Let $\widehat{\mathcal{F}}_{t-1}$ be the sigma field generated by the observed support history, legal exogenous context, randomness already used by the model, and the generated query history strictly before query event $t$.

**Proposition C.13** (Nonanticipative Rollout)**.** *The predictive distribution*

$$\widehat{p}_t = P_\theta\left(\cdot \mid \widehat{\mathcal{F}}_{t-1}\right) \tag{166}$$

*is measurable with respect to $\widehat{\mathcal{F}}_{t-1}$. A behavior generated from $\widehat{p}_t$ determines the current event transition and hence the state used at the next query event.*

*Proof.* At the first query event, the latent belief is determined by the support prefix and the model randomness already used in inference. The behavior distribution is evaluated from the pre event state, so $\widehat{p}_t$ is determined before the current event transition. A behavior is then sampled from this distribution and used by $J_{e_t}$, followed by process disturbance. Repeating the same construction enlarges the generated history by one event. Induction over the query sequence establishes the claim. □

The proposition corresponds directly to the autonomous evaluation procedures in Appendices B.1.2 and B.3.2: recorded query behavior supplies the score, while generated behavior supplies the transition input.

## C.8 DWM Structure

### C.8.1 Latent Dynamics and Behavior

DWM belongs to the broader family of state-space models in which a compact latent state summarizes the information required for future prediction. The use of a latent predictive state is related to predictive-state representations (Littman et al., 2001), while the explicit separation between continuous evolution and event updates connects the model to hybrid and impulsive dynamical systems (Hespanha et al., 2008; Jia & Benson, 2019). The particular structure used here is fixed across all experimental environments.

Let

$$X_t \in \mathbb{R}^d \tag{167}$$

denote the latent state. The theoretical statements below are written for finite $d$. The experimental implementation fixes

$$d = 4. \tag{168}$$

Consider two consecutive events separated by an interval $\Delta > 0$. Between these events, the latent state follows the affine flow

$$h_\Delta(x) = \Phi_\Delta x + d_\Delta, \qquad \Phi_\Delta = e^{A\Delta}, \qquad d_\Delta = (I - \Phi_\Delta)\mu. \tag{169}$$

The matrix $A$ determines the continuous dynamics and $\mu$ is the corresponding reference state. Since the matrix exponential is invertible,

$$\Phi_\Delta^{-1} = e^{-A\Delta}. \tag{170}$$

Events modify the state through a feature-affine map. Let

$$\psi(e) = (1, \psi_1(e), \ldots, \psi_p(e)) \tag{171}$$

denote the feature vector associated with event $e$. The event map is

$$J_e(z) = M(e)z + b(e), \tag{172}$$

where

$$M(e) = M_0 + \sum_{\ell=1}^{p} \psi_\ell(e) M_\ell, \qquad b(e) = b_0 + \sum_{\ell=1}^{p} \psi_\ell(e) b_\ell. \tag{173}$$

The categorical instantiations used in the experiments employ the corresponding restricted form. The action-dependent matrix terms are zero, while the additive event term depends on the categorical action:

$$M(e) = M_0, \qquad b(e) = b_0 + b_a. \tag{174}$$

This is the FeatureAffine categorical specialization used by the frozen model.

Behavior is generated before the current event modifies the latent state. For a categorical behavior space with reference category $0$, define

$$\eta_0(z, c) = 0, \tag{175}$$

and

$$\eta_a(z, c) = \beta_a + s_a^\top z + C_a c, \qquad a = 1, \ldots, K-1. \tag{176}$$

The conditional action probabilities are therefore

$$\pi_a(z, c) = \frac{\exp\{\eta_a(z, c)\}}{\sum_{j=0}^{K-1} \exp\{\eta_j(z, c)\}}. \tag{177}$$

The reference-category parameterization fixes a unique log-odds coordinate once category $0$ has been chosen. In particular,

$$\log \frac{\pi_a(z, c)}{\pi_0(z, c)} = \eta_a(z, c), \tag{178}$$

which will be useful when we construct observable predictive realizations in Section C.9.

### C.8.2 Causal Order and Observation

The latent state is only partially observed. This places DWM in the standard partially observed state-space setting (Kaelbling et al., 1998); in the implementation, inference over the latent state is represented through particles, following the usual sequential Monte Carlo interpretation (Arulampalam et al., 2002). The observation mechanism is separated from the latent transition itself.

To make the event order precise, let $\bar{X}_t^+$ denote the deterministic state immediately after the event map and let $\xi_t$ denote process noise. Starting from the previous post-event state $X_{t-1}^+$, one event cycle is

$$X_t^- = h_{\Delta_t}(X_{t-1}^+), \tag{179}$$

followed by behavior generation

$$a_t \sim \pi(\cdot \mid X_t^-, c_t). \tag{180}$$

The realized or generated event then changes the state,

$$\bar{X}_t^+ = J_{e_t}(X_t^-), \tag{181}$$

after which process noise is applied:

$$X_t^+ = \bar{X}_t^+ + \xi_t. \tag{182}$$

The observation is generated from this post-disturbance state,

$$Y_t \sim \mathcal{O}(\cdot \mid X_t^+, c_t). \tag{183}$$

For the affine measurement model used in the implementation,

$$Y_t = HX_t^+ + \nu_t. \tag{184}$$

The complete event order is consequently

$$X_{t-1}^+ \longrightarrow X_t^- \longrightarrow a_t \longrightarrow \bar{X}_t^+ \longrightarrow X_t^+ \longrightarrow Y_t. \tag{185}$$

This ordering is central to both inference and autonomous generation. During support, the recorded action is scored and then enters the event map. During query rollout, the sampled action enters the event map and the recorded query action is used only for scoring.

During the observed support prefix, actions and available observations are used to update the latent belief. Once the query period begins, generated actions determine the transition inputs and subsequent query history. Recorded query actions remain scoring targets, while query observations are not assimilated into the state estimate.

**Proposition C.14** (Autonomous Rollout)**.** *Let $\widehat{\mathcal{F}}_{t-1}$ denote the sigma-field generated by the observed support prefix, legal exogenous context, model randomness already used, and the model-generated query history before query step $t$. Then the predictive distribution*

$$\widehat{p}_t = P_\theta\left(\cdot \mid \widehat{\mathcal{F}}_{t-1}\right) \tag{186}$$

*is measurable with respect to $\widehat{\mathcal{F}}_{t-1}$. The generated action sampled from $\widehat{p}_t$ determines the current event update and hence the state used at the next query step.*

*Proof.* At the beginning of the query interval, the latent belief is a function of the support prefix and the randomness used by the inference procedure. The behavior channel in Equation (177) is evaluated from the pre-event state, so $\widehat{p}_t$ is already determined before any current query action is introduced into the transition.

A model action $\widehat{a}_t$ is then sampled from $\widehat{p}_t$. By the event order in Equation (185), $\widehat{a}_t$ determines the event $e_t$ and therefore the subsequent event map. Process noise is applied afterwards. The resulting state is consequently determined by $\widehat{\mathcal{F}}_{t-1}$ together with the newly generated action and fresh model randomness.

Repeating the same construction enlarges the generated history from $\widehat{\mathcal{F}}_{t-1}$ to $\widehat{\mathcal{F}}_t$. Induction over the query interval establishes the result. □

## C.9 Predictive Realization

### C.9.1 Setup and Continuation Space

The latent state of DWM is learned rather than defined directly from observable conditional probabilities. Nevertheless, it is useful to ask when a behavioral process itself admits a finite-dimensional deterministic predictive representation. This question is closely related to predictive representations of state, where state is characterized by its implications for future observable quantities (Littman et al., 2001). Here we study a more specific class in which state transitions are affine in a prescribed event feature family and behavior is represented through affine logits.

Let $\mathcal{H}$ denote the set of finite histories and let $\mathcal{E}$ denote a finite event alphabet. Assume that every legal action has strictly positive conditional probability. Choose a reference action $a_0$. For each $a \neq a_0$, define the observable log-odds function

$$\ell_a(h) = \log \frac{P(a \mid h)}{P(a_0 \mid h)}. \tag{187}$$

For an event $e \in \mathcal{E}$, define the continuation operator

$$(U_e f)(h) = f(he). \tag{188}$$

If $v = e_1 \cdots e_m$ is a finite event string, let

$$U_v = U_{e_1} \cdots U_{e_m}, \tag{189}$$

with $U_\epsilon = I$ for the empty string.

The collection of all continued log-odds functions spans

$$\mathcal{V}_\ell = \operatorname{span}\{1,\, U_v \ell_a : v \in \mathcal{E}^*,\, a \neq a_0\}. \tag{190}$$

The dimension of this space determines the amount of linear predictive structure required to represent all future log-odds obtained by extending a history with admissible event strings.

### C.9.2 Finite-Dimensional Realization

**Theorem C.15** (Observable Logit Realization)**.** *Assume*

$$\dim \mathcal{V}_\ell = d + 1 < \infty, \tag{191}$$

*and suppose that continuation by each legal event acts on $\mathcal{V}_\ell$ according to the feature relation*

$$U_e = T_0 + \sum_{j=1}^{p} \psi_j(e) T_j. \tag{192}$$

*Then there exists a deterministic predictive state $x(h) \in \mathbb{R}^d$ satisfying*

$$x(he) = F(e)x(h) + g(e), \tag{193}$$

*where*

$$[F(e) \;\; g(e)] = [F_0 \;\; g_0] + \sum_{j=1}^{p} \psi_j(e)[F_j \;\; g_j]. \tag{194}$$

*For every nonreference action,*

$$\log \frac{P(a \mid h)}{P(a_0 \mid h)} = \beta_a + s_a^\top x(h). \tag{195}$$

*The corresponding reference-category softmax reproduces the original conditional action distribution. Within the deterministic affine-state, affine-logit realization class, the dimension $d$ is minimal.*

*Proof.* The proof follows directly from the finite closure of the observable continuation space.

Choose a basis

$$q_0 = 1, q_1, \ldots, q_d \tag{196}$$

of $\mathcal{V}_\ell$ and define

$$x(h) = [q_1(h) \quad \cdots \quad q_d(h)]^\top. \tag{197}$$

Because $\mathcal{V}_\ell$ contains every continued log-odds function and is invariant under each $U_e$, continuation of any basis function remains inside the same space. Hence, for $i = 1, \ldots, d$,

$$(U_e q_i)(h) = c_{i0}(e) + \sum_{j=1}^{d} c_{ij}(e) q_j(h). \tag{198}$$

But $(U_e q_i)(h) = q_i(he)$. Stacking these equations over $i$ gives

$$x(he) = F(e)x(h) + g(e). \tag{199}$$

The feature relation in Equation (192) implies that the coefficients $c_{ij}(e)$ depend affinely on $\psi(e)$. The state transition therefore has the form in Equation (194).

The output relation follows from the definition of $\mathcal{V}_\ell$. Each $\ell_a$ belongs to this space, so there exist coefficients $\beta_a$ and $s_a$ such that

$$\ell_a(h) = \beta_a + s_a^\top x(h). \tag{200}$$

Because all legal actions have positive probability,

$$P(a_0 \mid h) = \frac{1}{1 + \sum_{a \neq a_0} \exp\{\ell_a(h)\}}, \tag{201}$$

and

$$P(a \mid h) = \exp\{\ell_a(h)\} P(a_0 \mid h). \tag{202}$$

These are exactly the probabilities generated by the reference-category softmax.

It remains to establish minimality. Suppose another deterministic affine-state, affine-logit realization has state $\widetilde{x}(h) \in \mathbb{R}^n$. Repeated application of its affine transition gives, for every finite string $v$,

$$\widetilde{x}(hv) = \widetilde{F}_v \widetilde{x}(h) + \widetilde{g}_v. \tag{203}$$

Every continued log-odds function is therefore contained in

$$\operatorname{span}\{1, \widetilde{x}_1, \ldots, \widetilde{x}_n\}. \tag{204}$$

Consequently,

$$d + 1 = \dim \mathcal{V}_\ell \leq n + 1. \tag{205}$$

Thus $n \geq d$, which proves minimality within the stated realization class. □

The theorem provides an observable construction of a predictive state. Its dimension is determined by closure of the continued log-odds functions rather than by the dimensionality of any particular stochastic latent parameterization. In the experiments, the DWM latent state is fixed at four dimensions by model design; Theorem C.15 instead characterizes the deterministic predictive dimension implied by the observable process.

**Corollary C.16** (Predictive Dimension)**.** *Under the assumptions of Theorem C.15,*

$$d_{\min} = \dim \mathcal{V}_\ell - 1. \tag{206}$$

### C.9.3 Predictive Dimension and Probability Rank

The construction above uses log-odds functions, whereas classical finite-rank stochastic representations are often formulated directly in terms of string probabilities. These notions are related but not interchangeable. The next result makes the distinction explicit.

**Proposition C.17** (Predictive Dimension and Probability Rank)**.** *Finite-dimensional deterministic affine-logit realization and finite probability-Hankel rank do not imply one another in general.*

*Proof.* We give one construction for each direction.

**Finite affine-logit state with infinite probability rank.** Consider the scalar system

$$X_{k+1} = \rho X_k, \qquad 0 < \rho < 1, \tag{207}$$

with Bernoulli behavior

$$P(a_k = 1 \mid X_k) = \sigma(X_k). \tag{208}$$

If $X_0 = c \neq 0$, then

$$P(a_k = 1) = \sigma(c\rho^k). \tag{209}$$

A finite probability-linear realization would imply that this scalar sequence satisfies a finite-order linear recurrence. For sufficiently large $k$, analyticity of the logistic function gives an expansion of the form

$$\sigma(c\rho^k) = \frac{1}{2} + \sum_{m \in \mathcal{M}} \alpha_m c^m \rho^{mk}, \tag{210}$$

where $\mathcal{M}$ contains infinitely many positive odd integers and infinitely many coefficients $\alpha_m$ are nonzero. The generating function of the centered sequence consequently contains terms

$$\frac{\alpha_m c^m \rho^{mK}}{1 - \rho^m z} \tag{211}$$

for infinitely many distinct values of $\rho^m$. It therefore has infinitely many poles and cannot be rational. The sequence does not satisfy a finite-order linear recurrence, even though its predictive state is one-dimensional.

**Finite probability rank without finite affine-logit state.** Now let

$$\Theta \in \left\{\frac{1}{4}, \frac{3}{4}\right\}, \qquad P\left(\Theta = \frac{1}{4}\right) = P\left(\Theta = \frac{3}{4}\right) = \frac{1}{2}, \tag{212}$$

and let actions be conditionally i.i.d. Bernoulli($\Theta$). Every finite string probability is a sum of two product measures, so the probability-Hankel rank is two.

After observing $k$ consecutive actions equal to one,

$$P\left(\Theta = \frac{3}{4} \mid 1^k\right) = \frac{3^k}{1+3^k}. \tag{213}$$

The next-step predictive probability is

$$p_k = P(a_{k+1} = 1 \mid 1^k) = \frac{1+3^{k+1}}{4(1+3^k)}, \tag{214}$$

and its log-odds satisfy

$$\ell_k = \log \frac{1+3^{k+1}}{3+3^k}. \tag{215}$$

For $k \geq 2$, this admits the convergent expansion

$$\ell_k = \log 3 + \sum_{m=1}^{\infty} \frac{(-1)^{m+1}}{m} \left(3^{-m} - 3^m\right) 3^{-mk}. \tag{216}$$

If a finite-dimensional deterministic affine realization existed along this repeated event sequence, the Cayley–Hamilton theorem would imply a finite-order linear recurrence for $\ell_k$. Equation (216) contains infinitely many distinct exponential modes, contradicting such a recurrence. Hence finite probability rank does not by itself yield a finite affine-logit predictive state. □

### C.10 Continuous Dynamics and Event Updates

#### C.10.1 Stable Flow Representation

The event-time process in Section C.9 describes how the state changes from one event to the next. DWM resolves this complete event transition into two components: continuous evolution before the event and an explicit event update afterwards. This decomposition follows the general logic of hybrid and impulsive systems (Hespanha et al., 2008; Jia & Benson, 2019), but here the event maps and behavior channel are coupled through the same predictive state.

Consider an event-time process

$$a_k \sim \pi(\cdot \mid x_{k-1}, c_k), \tag{217}$$

$$e_k = e(a_k, c_k), \tag{218}$$

and

$$x_k = F(e_k)x_{k-1} + g(e_k) + \varepsilon_k, \tag{219}$$

with the feature-affine operator family

$$[F(e) \;\; g(e)] = \sum_{\ell=0}^{p} \psi_\ell(e)[F_\ell \;\; g_\ell], \qquad \psi_0 \equiv 1. \tag{220}$$

For this subsection, assume a fixed event interval $\Delta > 0$. The initial-state law, post-event process-noise law, and observation kernel are kept fixed.

**Theorem C.18** (Stable Flow Representation)**.** *Let $A$ be any admissible Hurwitz matrix and let $\mu \in \mathbb{R}^d$. Define*

$$\Phi = e^{A\Delta}, \qquad d = (I - \Phi)\mu, \tag{221}$$

*and, for every event feature component,*

$$M_\ell = F_\ell \Phi^{-1}, \qquad b_\ell = g_\ell - F_\ell \Phi^{-1} d. \tag{222}$$

*Then the DWM flow followed by its event map reproduces the original event-time transition:*

$$M(e)(\Phi x + d) + b(e) = F(e)x + g(e). \tag{223}$$

*If the behavior channel is transformed according to*

$$\pi_h(a \mid z, c) = \pi\left(a \mid \Phi^{-1}(z - d), c\right), \tag{224}$$

*then an affine-logit Bernoulli or reference-category categorical channel remains affine-logit. Under the same initial-state, process-noise, and observation laws, the two representations induce the same finite-horizon behavior-observation distribution.*

*Proof.* The state-transition identity is algebraic. Since $\Phi^{-1} = e^{-A\Delta}$,

$$M(e)(\Phi x + d) + b(e) = F(e)\Phi^{-1}(\Phi x + d) + g(e) - F(e)\Phi^{-1}d \tag{225}$$

$$= F(e)x + g(e). \tag{226}$$

It remains to verify that the behavior family is preserved. Let the nonreference logit of the event-time model be

$$\eta_a(x, c) = \beta_a + s_a^\top x + C_a c. \tag{227}$$

The inverse flow coordinate is

$$x = \Phi^{-1}(z - d). \tag{228}$$

Substitution gives

$$\eta_a^h(z, c) = \left(\beta_a - s_a^\top \Phi^{-1} d\right) + (\Phi^{-\top} s_a)^\top z + C_a c. \tag{229}$$

Thus the transformed logit remains affine in the latent state and context.

We finally compare the finite-horizon laws. Couple the original event-time process and the DWM representation using the same initial state and the same auxiliary random variables for action sampling. Suppose their post-event states agree after event $k - 1$. The DWM first produces

$$z_k = \Phi x_{k-1} + d. \tag{230}$$

Equation (224) gives the same conditional action distribution in both representations, so the common sampling variable produces the same action and therefore the same event. Equation (223) then gives the same deterministic post-event state. Applying the same process-noise draw and the same observation kernel preserves equality through the remainder of the event cycle.

Induction over $k$ establishes equality of the finite-horizon latent, behavior, and observation laws. Marginalizing the latent states gives equality of the observable behavior-observation law. □

The theorem shows that stable continuous evolution can be introduced without changing the event-time process, provided the event map and behavior coordinates are transformed consistently. This is a representation result. Stability of the complete event-to-event dynamics is studied separately in Section C.11.

**Corollary C.19** (Feature Family Preservation)**.** *Define*

$$\mathcal{H}_\Delta = \begin{bmatrix} \Phi & d \\ 0 & 1 \end{bmatrix}. \tag{231}$$

*Then*

$$[M(e) \;\; b(e)] = [F(e) \;\; g(e)]\mathcal{H}_\Delta^{-1}. \tag{232}$$

*Because $\mathcal{H}_\Delta$ is invertible, the transformation preserves affine relations among the event operators and therefore preserves the affine dimension of their operator family.*

### C.10.2 Identification at a Fixed Event Interval

The construction above is not unique when observations are available only at one event interval. Indeed, the complete event operator identifies the product of the continuous flow and the event map. Separating these two components requires additional temporal information.

**Proposition C.20** (Flow and Event Equivalence). *Fix $\Delta > 0$ and event-time operators $(F(e), g(e))$. For any admissible Hurwitz matrix $A$ and any $\mu \in \mathbb{R}^d$, define*

$$M(e) = F(e)e^{-A\Delta}, \tag{233}$$

*and*

$$b(e) = g(e) - F(e)e^{-A\Delta}(I - e^{A\Delta})\mu. \tag{234}$$

*Then the resulting DWM has the same event-time transition. Combined with Equation (224), it also induces the same finite-horizon behavior-observation law.*

*Proof.* Substitution yields

$$M(e)\left[e^{A\Delta}x + (I - e^{A\Delta})\mu\right] + b(e) \tag{235}$$

$$= F(e)x + g(e). \tag{236}$$

The behavior transformation and equality of observable laws follow from Theorem C.18. □

This equivalence identifies the object determined at a fixed sampling interval: the complete event-time operator. The continuous generator becomes recoverable once the same event dynamics are observed across informative time intervals.

### C.10.3 Identification from Temporal Variation

**Theorem C.21** (Timing Identification). *Let $I \subset (0, \infty)$ be an interval. Suppose that, for every event $e$, the event-time operators $F_e(\delta)$ and $g_e(\delta)$ are continuously differentiable in $\delta \in I$.*

*There exist a common Hurwitz matrix $A$, a common vector $\mu$, and event-dependent quantities $M_e, b_e$ that do not depend on $\delta$, satisfying*

$$F_e(\delta) = M_e e^{A\delta}, \tag{237}$$

*and*

$$g_e(\delta) = M_e(I - e^{A\delta})\mu + b_e, \tag{238}$$

*if and only if there exist a common Hurwitz matrix $A$ and a common vector $v$ such that*

$$F_e'(\delta) = F_e(\delta)A, \tag{239}$$

*and*

$$g_e'(\delta) = F_e(\delta)v \tag{240}$$

*for every $e$ and every $\delta \in I$. In that case,*

$$\mu = -A^{-1}v. \tag{241}$$

*Proof.* We first prove necessity. Differentiating Equation (237) gives

$$F_e'(\delta) = M_e e^{A\delta}A = F_e(\delta)A. \tag{242}$$

Since $A$ is Hurwitz, zero is not an eigenvalue and $A$ is invertible. Set

$$v = -A\mu. \tag{243}$$

Differentiating Equation (238) then gives

$$g_e'(\delta) = -M_e e^{A\delta}A\mu \tag{244}$$

$$= F_e(\delta)v. \tag{245}$$

For sufficiency, assume Equations (239) and (240). Consider

$$F_e(\delta)e^{-A\delta}. \tag{246}$$

Its derivative is

$$\frac{d}{d\delta}\left(F_e(\delta)e^{-A\delta}\right) = F_e'(\delta)e^{-A\delta} - F_e(\delta)Ae^{-A\delta} \tag{247}$$

$$= 0. \tag{248}$$

Hence

$$M_e = F_e(\delta)e^{-A\delta} \tag{249}$$

is constant in $\delta$, proving Equation (237).

Now define $\mu = -A^{-1}v$. Choose any $\delta_0 \in I$ and set

$$b_e = g_e(\delta_0) - M_e(I - e^{A\delta_0})\mu. \tag{250}$$

The derivative of

$$M_e(I - e^{A\delta})\mu + b_e \tag{251}$$

equals $F_e(\delta)v$, which is exactly $g_e'(\delta)$ by assumption. The two expressions agree at $\delta_0$, so they agree throughout $I$. This proves Equation (238). □

The derivative conditions give a direct characterization of when event-time operators observed at multiple intervals arise from one common continuous generator. They also lead to a simple uniqueness condition.

**Corollary C.22** (Temporal Excitation)**.** *Suppose $(A, v)$ and $(\widetilde{A}, \widetilde{v})$ both satisfy Equations (239) and (240). If event-interval pairs $(e_i, \delta_i)$ can be chosen such that*

$$\mathcal{F} = \begin{bmatrix} F_{e_1}(\delta_1) \\ \vdots \\ F_{e_m}(\delta_m) \end{bmatrix} \tag{252}$$

*has full column rank, then*

$$A = \widetilde{A}, \qquad v = \widetilde{v}. \tag{253}$$

*Proof.* Subtracting the two versions of Equation (239) gives

$$F_e(\delta)(A - \widetilde{A}) = 0. \tag{254}$$

Similarly,

$$F_e(\delta)(v - \widetilde{v}) = 0. \tag{255}$$

Stacking the selected equations yields

$$\mathcal{F}(A - \widetilde{A}) = 0, \qquad \mathcal{F}(v - \widetilde{v}) = 0. \tag{256}$$

Full column rank of $\mathcal{F}$ gives the result. □

### C.11 Stability and Autonomous Generation

#### C.11.1 Setup and Fixed-Event-Path Certificate

Stable learned dynamics are commonly enforced by constraining the evolution law itself (Manek & Kolter, 2019). For hybrid or impulsive systems, continuous contraction and event amplification must instead be considered together; dwell-time conditions are a standard way to express this balance (Hespanha et al., 2008; Dashkovskiy & Slynko, 2023). DWM uses the same principle at the latent-state level.

Let $\|\cdot\|_w$ denote the norm used by the stability certificate, let $\|\cdot\|_{w,*}$ denote its dual norm, and let $\mu_w(\cdot)$ denote the induced matrix measure.

Suppose the continuous vector field satisfies

$$\mu_w\left(D_X f(X)\right) \le -\lambda, \qquad \lambda > 0. \tag{257}$$

For the DWM affine flow

$$f(X) = A(X - \mu), \tag{258}$$

this condition reduces to

$$\mu_w(A) \le -\lambda. \tag{259}$$

**Lemma C.23** (Continuous Flow Contraction)**.** *Under Equation (257), two continuous trajectories satisfy*

$$\|X(t) - \widetilde{X}(t)\|_w \le e^{-\lambda(t-s)} \|X(s) - \widetilde{X}(s)\|_w \tag{260}$$

*on every event-free interval* $[s, t]$.

*Proof.* Let

$$\delta(t) = X(t) - \widetilde{X}(t). \tag{261}$$

Using the mean-value representation of the vector field,

$$\dot{\delta}(t) = \left( \int_0^1 D_X f \left( \widetilde{X}(t) + \tau \delta(t) \right) d\tau \right) \delta(t). \tag{262}$$

The defining property of the matrix measure gives

$$D^+ \|\delta(t)\|_w \le -\lambda \|\delta(t)\|_w. \tag{263}$$

Applying Gronwall's inequality yields Equation (260). □

At an event, let $J_j$ denote the corresponding state update. Assume

$$\|J_j(x) - J_j(y)\|_w \le \rho_j \|x - y\|_w. \tag{264}$$

For the affine event map $J_j(x) = M_j x + b_j$, one may take

$$\rho_j = \|M_j\|_w. \tag{265}$$

The continuous and event contributions can now be combined.

**Theorem C.24** (Fixed-Event-Path Incremental Stability)**.** *Consider two DWM trajectories driven by the same realized event sequence and the same disturbance realization. If Equation (257) holds between events and Equation (264) holds at every event, then*

$$\|X(t) - \widetilde{X}(t)\|_w \le e^{-\lambda(t-s)} \prod_{j \in \mathcal{J}(s,t)} \rho_j \, \|X(s) - \widetilde{X}(s)\|_w. \tag{266}$$

*If the product satisfies*

$$e^{-\lambda(t-s)} \prod_{j \in \mathcal{J}(s,t)} \rho_j \le M e^{-\eta(t-s)}, \tag{267}$$

*then*

$$\|X(t) - \widetilde{X}(t)\|_w \le M e^{-\eta(t-s)} \|X(s) - \widetilde{X}(s)\|_w. \tag{268}$$

*Proof.* Partition the interval $(s, t)$ at the event times. On each continuous segment of length $\Delta_i$, Lemma C.23 contributes a factor no larger than $e^{-\lambda \Delta_i}$. At the event terminating that segment, Equation (264) contributes the factor $\rho_i$.

Multiplying these inequalities over the complete sequence gives

$$\|X(t) - \widetilde{X}(t)\|_w \le \left( \prod_i e^{-\lambda \Delta_i} \right) \left( \prod_{j \in \mathcal{J}(s,t)} \rho_j \right) \|X(s) - \widetilde{X}(s)\|_w. \tag{269}$$

Since the continuous intervals sum to $t - s$,

$$\prod_i e^{-\lambda \Delta_i} = e^{-\lambda(t-s)}, \tag{270}$$

which proves Equation (266). Substituting the certificate condition in Equation (267) gives Equation (268). □

A useful sufficient form follows from an average dwell-time condition, as in the classical analysis of impulsive systems (Hespanha et al., 2008; Dashkovskiy & Slynko, 2023).

**Corollary C.25** (Average Dwell Time)**.** *Suppose*

$$N(s,t) \le N_0 + \frac{t-s}{\tau_D}, \tag{271}$$

*and every event satisfies*

$$\rho_j \le \bar{\rho}. \tag{272}$$

*Let*

$$\bar{\rho}_+ = \max\{1, \bar{\rho}\}. \tag{273}$$

*Then*

$$e^{-\lambda(t-s)} \prod_j \rho_j \le \bar{\rho}_+^{N_0} \exp\left[-\left(\lambda - \frac{\log \bar{\rho}_+}{\tau_D}\right)(t-s)\right]. \tag{274}$$

*Hence the hybrid dynamics are exponentially contractive whenever*

$$\lambda > \frac{\log \bar{\rho}_+}{\tau_D}. \tag{275}$$

The certificate is expressed through a matrix measure rather than eigenvalues alone. This distinction matters for nonnormal dynamics. For example,

$$A = \begin{bmatrix} -1 & K \\ 0 & -1 \end{bmatrix} \tag{276}$$

is Hurwitz for every finite $K$, but under the Euclidean norm,

$$\mu_2(A) = -1 + \frac{|K|}{2}. \tag{277}$$

A stable spectrum therefore need not imply contraction in the norm used by the certificate.

### C.11.2 Autonomous Transition Kernel

The previous theorem conditions on the same realized event sequence. During autonomous rollout, however, actions are generated from the latent state itself. A perturbation can therefore influence the next state in two ways: it changes the state entering a fixed action branch, and it changes the probabilities assigned to the available branches. The autonomous analysis must include both effects.

For one event interval, write

$$h(x) = \Phi x + d. \tag{278}$$

Let

$$Q_a(z) = \mathcal{L}\left(J_a(z) + \varepsilon\right) \tag{279}$$

denote the post-event state distribution conditional on action $a$. The complete autonomous kernel is

$$K(x) = \sum_a \pi_a(h(x)) Q_a(h(x)). \tag{280}$$

Assume that the flow satisfies

$$\|h(x) - h(y)\| \le L_h \|x - y\|, \tag{281}$$

each action-conditioned transition satisfies

$$W_1\left(Q_a(z), Q_a(w)\right) \le L_J \|z - w\|, \tag{282}$$

and the action distribution satisfies

$$\mathrm{TV}\left(\pi(z), \pi(w)\right) \le L_\pi \|z - w\|. \tag{283}$$

Finally, define the largest cross-action consequence difference by

$$D_Q = \sup_z \max_{a,b} W_1\left(Q_a(z), Q_b(z)\right). \tag{284}$$

**Theorem C.26** (Autonomous Wasserstein Contraction)**.** *Under Equations (281)–(284),*

$$W_1\left(K(x), K(y)\right) \leq L_h\left(L_J + L_\pi D_Q\right)\|x - y\|. \tag{285}$$

*Thus a valid one-step contraction coefficient is*

$$\kappa_{\text{auto}} = L_h\left(L_J + L_\pi D_Q\right). \tag{286}$$

*Proof.* Let

$$z = h(x), \qquad w = h(y), \tag{287}$$

and introduce the intermediate distribution

$$R = \sum_a \pi_a(z) Q_a(w). \tag{288}$$

The triangle inequality gives

$$W_1(K(x), K(y)) \leq W_1(K(x), R) + W_1(R, K(y)). \tag{289}$$

The first term compares the same action mixture evaluated at two latent states. Couple each branch $Q_a(z)$ with $Q_a(w)$ while retaining the action index. Then

$$W_1(K(x), R) \leq \sum_a \pi_a(z) W_1\left(Q_a(z), Q_a(w)\right) \tag{290}$$

$$\leq L_J\|z - w\|. \tag{291}$$

The second term keeps the latent state $w$ fixed and changes only the action mixture. The common probability mass assigned to each action can remain on the same branch. The unmatched mass is exactly the total variation distance between the two action distributions, and transporting this mass between any two branches costs at most $D_Q$. Hence

$$W_1(R, K(y)) \leq D_Q\,\mathrm{TV}\left(\pi(z), \pi(w)\right) \tag{292}$$

$$\leq D_Q L_\pi\|z - w\|. \tag{293}$$

Combining Equations (291) and (293) gives

$$W_1(K(x), K(y)) \leq \left(L_J + L_\pi D_Q\right)\|z - w\|. \tag{294}$$

Applying the flow bound in Equation (281) completes the proof. □

For the affine-logit behavior channel, the sensitivity term can be written directly in terms of the learned logit vectors.

**Corollary C.27** (Softmax Sensitivity)**.** *For*

$$\pi_a(z) = \mathrm{softmax}_a\left(\beta_a + s_a^\top z\right), \tag{295}$$

*with* $s_0 = 0$*,*

$$L_\pi \leq \frac{1}{4}\max_{a,b}\|s_a - s_b\|_*. \tag{296}$$

*Proof.* Fix a unit direction $v$ and define

$$t_a = s_a^\top v. \tag{297}$$

Differentiating the softmax gives

$$D_v\pi_a = \pi_a\left(t_a - \mathbb{E}_\pi t\right). \tag{298}$$

Therefore

$$\frac{1}{2}\sum_a |D_v\pi_a| = \frac{1}{2}\mathbb{E}_\pi\left|t - \mathbb{E}_\pi t\right|. \tag{299}$$

If a scalar random variable is supported on $[t_{\min}, t_{\max}]$, its mean absolute deviation from its mean is at most $(t_{\max} - t_{\min})/2$. Hence

$$\frac{1}{2}\sum_a |D_v \pi_a| \le \frac{1}{4}(t_{\max} - t_{\min}). \tag{300}$$

Moreover,

$$t_{\max} - t_{\min} \le \max_{a,b} \|s_a - s_b\|_*. \tag{301}$$

Taking the supremum over unit directions and integrating along the segment connecting two latent states gives Equation (296). □

For the categorical FeatureAffine implementation, the branch matrices are common across actions. If

$$J_a(z) = M_0 z + b_0 + b_a, \tag{302}$$

then

$$L_J \le \|M_0\|, \tag{303}$$

while the cross-action distance depends on the action-specific offsets:

$$W_1\left(Q_a(z), Q_b(z)\right) = \|b_a - b_b\| \tag{304}$$

when all branches use the same additive process-noise law. Therefore

$$D_Q \le \max_{a,b} \|b_a - b_b\|. \tag{305}$$

This specialization matches the categorical event map used in the experiments.

### C.11.3 Long-Horizon Consequence

If the autonomous one-step kernel contracts, the same property propagates over multiple generated steps.

**Corollary C.28** (Long-Horizon Contraction)**.** *Assume the latent state space is complete and separable, the autonomous kernel maps probability measures with finite first moment into the same space, and*

$$W_1\left(K(x), K(y)\right) \le \kappa \|x - y\|, \qquad 0 \le \kappa < 1. \tag{306}$$

*Then, for any $\mu, \nu \in \mathcal{P}_1$,*

$$W_1\left(\mu K^H, \nu K^H\right) \le \kappa^H W_1(\mu, \nu). \tag{307}$$

*If $K$ is time homogeneous, the corresponding Markov operator has a unique invariant probability measure in $\mathcal{P}_1$.*

*Proof.* Let $\Gamma$ be a coupling of $\mu$ and $\nu$. For each pair $(x, y)$, couple $K(x, \cdot)$ and $K(y, \cdot)$ so that their expected transport cost is arbitrarily close to $W_1(K(x), K(y))$. Integrating these conditional couplings with respect to $\Gamma$ gives

$$W_1(\mu K, \nu K) \le \kappa \int \|x - y\| \, d\Gamma(x, y). \tag{308}$$

Taking the infimum over $\Gamma$ yields

$$W_1(\mu K, \nu K) \le \kappa W_1(\mu, \nu). \tag{309}$$

Applying the same inequality recursively proves Equation (307). The invariant-measure statement follows because the Markov operator is a contraction on $(\mathcal{P}_1, W_1)$. □

Theorem C.24 and Theorem C.26 address two levels of the dynamics. The first is the certificate used for the learned flow and event maps along a common event path. The second incorporates the additional sensitivity introduced when future event probabilities are generated by the model itself.

## C.12 Developmental Learning

### C.12.1 Stagewise Development

DWM formalizes Turing's child-machine separation between an initial organization and a mature system shaped by experience (Turing, 1950). This developmental construction is implemented as a fixed sequence of developmental stages. The model family and causal structure stay fixed; the parameters mature within that family.

Let $\Theta$ denote the parameter space and define the certificate-eligible set

$$\mathcal{C} = \{\theta \in \Theta : \mathrm{Cert}(\theta) = 1\}. \tag{310}$$

The dynamical stages are indexed by

$$s = 1, \ldots, 4. \tag{311}$$

Let $\theta^{(s-1)}$ denote the parent entering stage $s$. The candidate stage is evaluated on held-out development partitions: five-fold cross-fitting is used for the standard fold-based protocol, while PARTNR uses six strict scene-level leave-one-scene-out development folds. Each candidate is evaluated by autonomous predictive likelihood on its held-out development partition together with the same fixed-event-path certificate used for the deployed model. The final test split is not used for stage promotion or calibration.

If the held-out autonomous NLL gain meets the stage-specific minimum and the certificate is satisfied, the selected developmental duration is replayed from the current parent on the full TRAIN/development set, producing

$$\widetilde{\theta}^{(s)}. \tag{312}$$

The resulting certified model becomes the parent of the next stage:

$$\theta^{(s)} = \widetilde{\theta}^{(s)} \in \mathcal{C}. \tag{313}$$

If the stage is not promoted, development continues from the current parent,

$$\theta^{(s)} = \theta^{(s-1)}. \tag{314}$$

$S5$ is a post-hoc scoring-calibration stage applied after $S1$–$S4$ dynamical development. It applies an affine calibration to predictive natural parameters using out-of-fold development predictions after the autonomous trajectory has been generated. The calibrated probabilities are used for scoring and do not determine sampled rollout actions or state transitions. For every nonreference category,

$$z_a^{\mathrm{cal}} = \alpha z_a + \gamma_a, \qquad \alpha > 0, \tag{315}$$

with

$$z_0^{\mathrm{cal}} = 0. \tag{316}$$

The identity transformation is included as a member of this calibration family.

**Proposition C.29** (Affine Calibration)**.** *If*

$$z_a = \beta_a + s_a^\top X + C_a c, \tag{317}$$

*then the calibrated natural parameter in Equation (315) can be written as*

$$z_a^{\mathrm{cal}} = \beta_a^{\mathrm{cal}} + (s_a^{\mathrm{cal}})^\top X + C_a^{\mathrm{cal}} c, \tag{318}$$

*where*

$$\beta_a^{\mathrm{cal}} = \alpha\beta_a + \gamma_a, \qquad s_a^{\mathrm{cal}} = \alpha s_a, \qquad C_a^{\mathrm{cal}} = \alpha C_a. \tag{319}$$

*Thus $S5$ preserves the affine-logit form of the scored predictive distribution.*

*Proof.* Substituting the original logit into Equation (315) gives

$$z_a^{\mathrm{cal}} = \alpha\left(\beta_a + s_a^\top X + C_a c\right) + \gamma_a \tag{320}$$

$$= (\alpha\beta_a + \gamma_a) + (\alpha s_a)^\top X + (\alpha C_a) c. \tag{321}$$

The result follows by identifying the transformed coefficients. □

### C.12.2 DEVELOPMENTAL CLOSURE

We can now state the closure property of the complete developmental sequence.

**Theorem C.30** (Developmental Closure)**.** *Assume the initial model satisfies*

$$\theta^{(0)} \in \mathcal{C}. \tag{322}$$

*Suppose that every promoted full-TRAIN realization satisfies*

$$\widetilde{\theta}^{(s)} \in \mathcal{C}. \tag{323}$$

*Then*

$$\theta^{(s)} \in \mathcal{C}, \qquad s = 0, \dots, 4. \tag{324}$$

*$S5$ preserves the affine-logit form of the scored predictive probabilities by Proposition C.29. Because the calibration is applied after autonomous trajectory generation, it does not alter transition-driving probabilities, sampled rollout actions, flow or event-update parameters, and therefore leaves the dynamical certificate unchanged. The mature dynamical model remains in the same certified DWM structural class.*

*Proof.* The proof is by induction over the first four developmental stages.

The base case follows from Equation (322). Assume that the parent entering stage $s$ satisfies

$$\theta^{(s-1)} \in \mathcal{C}. \tag{325}$$

There are two possible stage outcomes. If the stage is promoted, Equation (323) gives

$$\theta^{(s)} = \widetilde{\theta}^{(s)} \in \mathcal{C}. \tag{326}$$

If the current parent is retained, Equation (314) gives

$$\theta^{(s)} = \theta^{(s-1)} \in \mathcal{C}. \tag{327}$$

Thus $\theta^{(s)} \in \mathcal{C}$ in either case. Induction proves Equation (324) for $s = 0, \dots, 4$.

$S5$ changes only the post-hoc natural parameters used to score predictive probabilities. Proposition C.29 shows that the calibrated categorical distribution remains in the same affine-logit family. The autonomous rollout has already been generated from the uncalibrated behavior probabilities, so the sampled actions, latent trajectory, flow matrix, reference state, event maps, and process-noise mechanism are unchanged. The dynamical certificate inherited from stage 4 is therefore unchanged as well. □

### C.12.3 INITIAL ORGANIZATION AND SHARED STRUCTURAL CLASS

The final result formalizes the initial organization shared across environments. This structural specification governs initialization, development, inference, and autonomous rollout for each world.

Let

$$\mathfrak{W} = (d, \mathcal{F}, \mathcal{J}, \mathcal{B}, \mathcal{O}, \mathcal{R}, \mathcal{C}, \mathrm{Init}, \mathrm{Dev}) \tag{328}$$

denote the DWM structural specification. Here $\mathcal{F}$ denotes the flow family, $\mathcal{J}$ the event-update family, $\mathcal{B}$ the behavior family, $\mathcal{O}$ the observation interface, $\mathcal{R}$ the autonomous rollout rule, and $\mathcal{C}$ the certificate class. The operators $\mathrm{Init}$ and $\mathrm{Dev}$ specify the common initialization rule and developmental procedure.

For environment $r$, initialization gives

$$\theta_r^{(0)} = \mathrm{Init}\left(\mathcal{D}_r^{\mathrm{train}}; \mathfrak{W}\right), \tag{329}$$

and development produces

$$\theta_r^{\star} = \mathrm{Dev}\left(\mathcal{D}_r^{\mathrm{train}}; \theta_r^{(0)}, \mathfrak{W}\right). \tag{330}$$

The initialization rule is shared, while its numerical instantiation depends on the TRAIN data and the task-facing categorical and observation interfaces. In the experiments, environments use the same structural specification with $d = 4$ as the common latent-state setting; numerical initial parameters depend on their action vocabularies and measurement dimensions.

**Proposition C.31** (Shared Structural Class)**.** *Suppose task adaptation changes only quantities permitted by the common behavior and observation interfaces, including the categorical vocabulary, observation dimension, and TRAIN-fitted deterministic observation representation. Suppose further that every developmental stage follows the common procedure in $\mathfrak{W}$. Then the mature models*

$$\{\theta_r^\star\}_r \tag{331}$$

*are realizations of the same DWM structural class, although their task-facing dimensions and learned numerical parameters may differ.*

*Proof.* Every environment uses the same latent dimension, flow family, event-update family, causal order, behavior family, autonomous rollout semantics, certificate definition, initialization rule, and developmental rule encoded by $\mathfrak{W}$. The task adapter determines only quantities admitted by the corresponding task-facing interfaces.

Equation (329) therefore produces an admissible initial realization of the common structural model for each environment. Theorem C.30 shows that stagewise development preserves the certified dynamical class, while Proposition C.29 shows that the final calibration remains inside the same behavior family. Hence Equation (330) changes the mature realization without changing the structural specification $\mathfrak{W}$.

It follows that the models $\theta_r^\star$ may represent different learned worlds while sharing the same DWM organization. □

# D Experimental Details

## D.1 Hardware

Experiments are conducted on NVIDIA RTX PRO 6000 GPUs. The same implementation is used across the standard sequential tasks, bsuite benchmarks, and high-dimensional environments unless otherwise specified.

**Data partitions and evaluation protocol.** All dataset-level partitions are fixed before model fitting. For the standard sequential tasks, the TRAIN/VAL/TEST splits contain 51/16/15 Public Goods sequences, 353/77/69 DeliData dialogues, and 210/45/45 PersuasionForGood dialogues, respectively. Developmental stage promotion and affine calibration use five-fold cross-fitting within TRAIN: each candidate is fitted on four folds and evaluated on the held-out TRAIN fold, and an accepted stage is subsequently replayed on the full TRAIN partition. VAL remains separate from this TRAIN cross-fitting and is used for development-side evaluation, while TEST is reserved for final reporting.

Melting Pot uses a frozen source-episode split of 120 TRAIN, 40 VAL, and 40 TEST episodes. Its five-fold developmental cross-fitting is performed within TRAIN, with VAL retained as a separate development-side diagnostic partition after the TRAIN-only development procedure is frozen. The 40 TEST episodes yield 240 focal-agent trajectories and 72,000 query events for final evaluation. PARTNR uses six development scenes with strict scene-level leave-one-scene-out development folds for selection and analysis. After development is frozen, the final model uses all six development scenes and is evaluated on the separate blind-test scene containing 76 episodes and 633 query events.

Unless otherwise stated, quantities reported as mean $\pm$ sample standard deviation are computed across the independent runs used for the corresponding experiment.

## D.2 Standard Sequential Prediction Tasks

**Datasets.** We use three sequential interaction datasets: Public Goods, DeliData (Karadzhov et al., 2023), and PersuasionForGood (Wang et al., 2019). Public Goods is derived from the experimental data of Cox & Stoddard (2021), which records repeated binary contribution decisions in fixed three-player groups over ten rounds. We convert the original participant-round records into 82 sequential trajectories under our trajectory-construction protocol. DeliData contains multi-party deliberation

during collaborative problem solving, and PersuasionForGood contains dyadic persuasive conversations. For PersuasionForGood, we use the 300 dialogues with strategy annotations. Dataset statistics and the support–query construction are summarized in Table 5.

Table 5: Datasets used in the standard sequential prediction tasks.

| Dataset | Sequences | Interaction | Support | Query | Target |
|---|---|---|---|---|---|
| Public Goods | 82 | Strategic | 4 rounds | 6 rounds | Binary |
| DeliData | 499 | Multi-party | 40% prefix | Remaining events | Categorical |
| PersuasionForGood | 300 | Dyadic | 40% prefix | Remaining events | Categorical |

The observed segment of each sequence is used to initialize the model state, after which predictions are evaluated over the remaining events. Public Goods provides an 18-dimensional causal context at each round. For DeliData and PersuasionForGood, events retain their original chronological order.

**Baseline Models.** We compare DWM with empirical predictors, recurrent models, attention-based sequence models, a structured state-space model, and latent world models. The complete comparison set is summarized in Table 6.

Table 6: Baseline models used in the standard sequential prediction tasks.

| Methodological family | Models |
|---|---|
| Empirical predictors | Frequency, Markov |
| Recurrent models | GRU, LSTM |
| Attention model | Transformer |
| State-space model | Mamba-2 |
| Latent world models | TD-MPC2, DreamerV3 |

**Data preprocessing.** Continuous variables are standardized using statistics computed from the training data. Categorical variables are encoded before model fitting. Sequence order is preserved throughout preprocessing.

**Evaluation metric.** Negative log-likelihood is computed as the mean negative log-probability per query event, consistent with Equations (14) and (30). For Figure 2 and Table 1, stochastic methods are reported as mean $\pm$ sample standard deviation across five independent seeds; the deterministic Frequency and Markov predictors are reported as single values.

### D.3 DWM and Baseline Configurations

**DWM configuration.** All experiments use a four-dimensional dynamical state. Model parameters are optimized with AdamW and gradient norm is clipped at 5. The developmental schedule is given in Table 7.

Table 7: Developmental training schedule used by DWM.

| Stage | Epochs | Particles | LR | Teacher weight | Min. gain |
|---|---|---|---|---|---|
| S1: Teacher Acquisition | 300 | 256 | $3 \times 10^{-3}$ | $1.00 \rightarrow 1.00$ | 0.005 |
| S2: Soft Autonomy | 300 | 256 | $1 \times 10^{-3}$ | $0.50 \rightarrow 0.15$ | 0.001 |
| S3: Soft Refinement | 400 | 512 | $6 \times 10^{-4}$ | $0.30 \rightarrow 0.05$ | 0.0002 |
| S4: Categorical Maturation | 250 | 512 | $3 \times 10^{-4}$ | $0.10 \rightarrow 0.10$ | 0.001 |

The teacher-weight schedule is stage-local: each stage starts from the value listed in Table 7 and follows the indicated within-stage schedule. The contribution is fixed at 1.00 in S1, decreases from 0.50 to 0.15 in S2 and from 0.30 to 0.05 in S3, and is fixed at 0.10 in S4. Figure 5 uses this same schedule. Affine categorical calibration is fitted after development and applied post hoc to predictive probabilities.

**Neural baseline configurations.** GRU, LSTM, and Transformer use weight decay $10^{-5}$ and dropout $0.2$. Their configurations are reported in Table 8.

Table 8: Configurations of recurrent and attention baselines.

| Model | Task | Width | Layers | Heads | LR | Batch | Epochs | Pat. |
|---|---|---|---|---|---|---|---|---|
| GRU | Public Goods | 200 | 2 | – | $3 \times 10^{-4}$ | 16 | 100 | 20 |
| LSTM | Public Goods | 200 | 2 | – | $1 \times 10^{-3}$ | 16 | 100 | 20 |
| Transformer | Public Goods | 200 | 2 | 2 | $3 \times 10^{-3}$ | 16 | 100 | 20 |
| GRU | DeliData/P4G | 128 | 2 | – | $1 \times 10^{-3}$ | 32 | 120 | 18 |
| LSTM | DeliData/P4G | 128 | 2 | – | $1 \times 10^{-3}$ | 32 | 120 | 18 |
| Transformer | DeliData/P4G | 128 | 2 | 4 | $1 \times 10^{-3}$ | 32 | 120 | 18 |

Mamba-2 (Dao & Gu, 2024) uses $d_{\text{model}} = 128$, $d_{\text{state}} = 64$, $d_{\text{conv}} = 4$, expansion factor 2, and head dimension 64. TD-MPC2 (Hansen et al., 2024) uses a 128-dimensional latent state, a 384-unit dynamics MLP, a two-layer 256-unit encoder, planning horizon 3, learning rate $3 \times 10^{-4}$, and batch size 256. DreamerV3 (Hafner et al., 2025) uses deterministic state size 512, hidden size 64, 32 categorical stochastic variables with four classes each, and encoder/decoder width 64 for the Public Goods experiment.

### D.4 Generic Temporal Benchmarks

**Datasets.** We additionally evaluate DWM on trajectories generated from bsuite (Osband et al., 2020). Environments are grouped according to their native action cardinality $K = |\mathcal{A}|$. We report three shards covering binary-action, three-action, and five-action environments. Their composition is shown in Table 9.

Table 9: Test autonomous NLL on the bsuite temporal benchmark shards.

| Shard | Families | Obs. dim. | Test eps. | Test NLL ↓ |
|---|---|---|---|---|
| K2 | `deep_sea`, `deep_sea_stochastic`, `memory_len`, `memory_size`, `umbrella_distract`, `umbrella_length` | 2506 | 72 | $0.69242 \pm 0.00014$ |
| K3 | `cartpole`, `cartpole_noise`, `cartpole_scale`, `cartpole_swingup`, `catch`, `catch_noise`, `catch_scale`, `mountain_car`, `mountain_car_noise`, `mountain_car_scale` | 60 | 120 | $1.08365 \pm 0.00029$ |
| K5 | `discounting_chain` | 3 | 12 | $1.44052 \pm 0.00012$ |

The three shards span memory tasks, controlled dynamical systems, tracking environments, exploration problems, and delayed consequences. The same four-dimensional DWM state and developmental procedure are used throughout.

### D.5 High-Dimensional Observation Interfaces

**Datasets and preprocessing.** We use Melting Pot (Leibo et al., 2021) and PARTNR (Chang et al., 2025). Melting Pot provides visual multi-agent interactions. RGB observations of size $88 \times 88 \times 3$ are represented using 64 DCT coefficients. PARTNR provides embodied human–robot collaboration trajectories and is represented by a 120-dimensional multimodal vector.

Table 10 summarizes the experimental data.

**Development and test protocol.** The high-dimensional experiments follow the data-partition convention defined at the beginning of Appendix D. The corresponding development and final-test populations are summarized in Table 10.

**Evaluation aggregation.** For the high-dimensional experiments, autonomous NLL is the pooled mean negative log-probability over query events within each evaluation run. Table 3 reports mean $\pm$ sample standard deviation across three independent evaluation seeds for each stochastic model. In particular, Melting Pot DWM is evaluated per seed before aggregation, using the same seed-level reporting convention as the DreamerV3 comparison.

Table 10: High-dimensional environments used in the experiments.

| Property | Melting Pot | PARTNR |
|---|---|---|
| Development data | 120 train + 40 validation episodes | 6 scenes |
| Test data | 40 source episodes | 76 episodes |
| Test trajectories | 240 focal-agent trajectories | 76 trajectories |
| Observation | RGB $88 \times 88 \times 3$ | Multimodal |
| Representation | DCT-64 | 120-D |
| Dynamical state | 4-D | 4-D |
| Query events | 72,000 | 633 |

For the Melting Pot DreamerV3 comparison, the recurrent state uses deterministic size 2048, hidden size 256, 32 categorical stochastic variables with 16 classes, eight blocks, unimix ratio 0.01, and free-nats value 1.

**Additional diagnostics.** We examine two properties of the Melting Pot model: dependence on visual information and prediction over increasing rollout horizons. Results are reported together in Table 11. For the rollout-horizon diagnostic, $h$ is the query position counted in autonomous propagation steps after the support prefix; the NLL at horizon $h$ pools the recorded targets at that position over held-out trajectories that reach $h$.

Table 11: Out-of-fold development diagnostics on Melting Pot. These values use held-out development data and are distinct from the final-test results in Table 3.

| Analysis | Condition | Held-out dev. NLL ↓ |
|---|---|---|
| Observation | Full | 1.15489 |
| | Action only | 1.16664 |
| | Shuffled visual | 1.15837 |
| Rollout horizon | $h = 1$ | 1.18349 |
| | $h = 5$ | 1.16584 |
| | $h = 10$ | 1.08277 |
| | $h = 20$ | 1.13601 |
| | $h = 50$ | 1.12387 |

These diagnostics are computed from frozen out-of-fold development models and use a different evaluation population from the final test results in Table 3. Diagnostic NLL is pooled over held-out development query events; no cross-seed uncertainty is reported. Removing the visual input produces the largest change in the observation ablation, while shuffled visual features remain closer to the full condition. The rollout results show a non-monotonic profile across prediction horizons, with likelihood quality remaining comparable across the evaluated positions.

### D.6 Developmental Analysis

Developmental decisions are made after each stage using autonomous prediction on held-out development partitions. Table 12 reports the corresponding stage-wise autonomous prediction gains on those development partitions for Melting Pot and PARTNR; these values are not final-test results.

Table 12: Held-out development autonomous prediction gains across developmental stages.

| Environment | S1 | S2 | S3 | S4 |
|---|---|---|---|---|
| Melting Pot | 0.21484 | 0.05871 | 0.04162 | 0.10447 |
| PARTNR | 0.63078 | 0.08940 | 0.05376 | 0.02102 |

The two environments display different developmental profiles. PARTNR concentrates most of its improvement in the first stage and then shows progressively smaller gains. Melting Pot also improves most strongly at S1, followed by smaller intermediate gains and a larger change again at S4. In PersuasionForGood, S1, S2, and S4 are retained, while S3 is not retained.

### D.6.1 LATENT-STATE GEOMETRY AND DECODER SENSITIVITY

How is the four-dimensional latent state organized after development? We analyze posterior-mean state trajectories from the frozen out-of-fold Melting Pot models and the strict scene-LOSO PARTNR models. For each fold, we compute principal components from the autonomous posterior-mean trajectories. We then evaluate how these directions enter the learned behavior and measurement maps.

**Latent-State Geometry.** Table 13 reports the variance explained by the four principal components. In Melting Pot, the first principal component accounts for more than 99.8% of posterior-mean state variance in every fold. PARTNR shows the same concentration at a lower level, with the first principal component explaining between 95.8% and 99.5% of the variance across the six scene-LOSO development folds.

Table 13: Posterior-mean latent-state geometry in Melting Pot and PARTNR. Principal components are computed separately within each frozen fold.

| Environment | Fold | PC1 | PC2 | PC3 | PC4 | Effective rank |
|---|---|---|---|---|---|---|
| Melting Pot | 0 | 0.998662 | 0.001303 | 0.000024 | 0.000011 | 1.0104 |
| Melting Pot | 1 | 0.999190 | 0.000752 | 0.000044 | 0.000013 | 1.0068 |
| Melting Pot | 2 | 0.999089 | 0.000860 | 0.000041 | 0.000010 | 1.0075 |
| Melting Pot | 3 | 0.999336 | 0.000614 | 0.000037 | 0.000013 | 1.0057 |
| Melting Pot | 4 | 0.999418 | 0.000520 | 0.000046 | 0.000017 | 1.0052 |
| PARTNR | 0 | 0.979333 | 0.010911 | 0.007428 | 0.002328 | 1.1278 |
| PARTNR | 1 | 0.958304 | 0.034688 | 0.006826 | 0.000181 | 1.2129 |
| PARTNR | 2 | 0.971675 | 0.021832 | 0.005883 | 0.000610 | 1.1574 |
| PARTNR | 3 | 0.994685 | 0.004664 | 0.000618 | 0.000033 | 1.0359 |
| PARTNR | 4 | 0.975789 | 0.021774 | 0.002392 | 0.000045 | 1.1299 |
| PARTNR | 5 | 0.961504 | 0.029424 | 0.005762 | 0.003309 | 1.2094 |

The concentration is also present in state changes. In PARTNR, the first principal component of consecutive posterior-mean state increments explains between 98.6% and 99.8% of increment variance. The action-dependent jump vectors show a similar structure: their leading singular direction accounts for 86.2% to 94.2% of jump energy, and its cosine with the empirical state PC1 ranges from 0.962 to 1.000.

**Decoder Sensitivity.** We next measure how movement along each latent principal direction affects the fitted behavior and measurement maps. Let $u_k$ denote a unit principal direction, $S$ the linear map from latent state to behavior logits, and $H$ the measurement map. We report

$$\|Su_k\|_2 \qquad \text{and} \qquad \|Hu_k\|_2$$

as behavior and measurement sensitivities, respectively.

Table 14 reports the Melting Pot results, averaged across the five folds. PC1 has the largest behavior sensitivity, PC2 contributes a behavior sensitivity of 0.9259, and PC3 has the largest measurement sensitivity, followed by PC4.

Table 14: Sensitivity of Melting Pot latent principal directions to the behavior and measurement maps. Values are means across five frozen folds.

| Direction | State variance | $\|Su_k\|_2$ | $\|Hu_k\|_2$ |
|---|---|---|---|
| PC1 | 0.999139 | 1.1611 | 0.2529 |
| PC2 | 0.000810 | 0.9259 | 0.9583 |
| PC3 | 0.000038 | 0.1725 | 2.8981 |
| PC4 | 0.000013 | 0.1921 | 1.6618 |

Both learned maps have rank four across all five Melting Pot folds,

$$\operatorname{rank}(S) = 4, \qquad \operatorname{rank}(H) = 4.$$

After normalizing $S$ and $H$ by their Frobenius norms and stacking the two maps, the combined matrix also has rank four in every fold. Its fourth singular value remains between 0.457 and 0.542 of its largest singular value.

In PARTNR, the effective rank of the action-dependent jump matrix ranges from 1.28 to 1.67 across the six LOSO folds. The effective rank of the behavior map ranges from 1.57 to 1.87, and that of the measurement map ranges from 1.33 to 2.41. These values describe how behavior, observation, and event responses are distributed across the learned PARTNR state coordinates.

**Cross-Fold Alignment.** The orientation of a latent principal component can vary across independently fitted models. We therefore compare both the latent direction $u_f$ and its behavior-space image $S_f u_f$ across folds. Principal-component signs are aligned through their behavior-space responses before the latter comparison.

Table 15: Cross-fold alignment of the dominant latent direction. Raw latent alignment is the mean absolute cosine between fold-local PC1 directions. Behavior alignment is computed after sign alignment of $S_f u_f$.

| Environment | Raw latent PC1 | Behavior-space $S_f u_f$ |
|---|---|---|
| Melting Pot | 0.990 | 0.992 |
| PARTNR | 0.694 | 0.972 |

Behavior-space alignment is 0.992 in Melting Pot and 0.972 in PARTNR. The corresponding raw latent alignment is 0.990 and 0.694, respectively. For PARTNR, the mean absolute cosine of the measurement-space vectors $H_f u_f$ is 0.439.

**Action Responses Along the Dominant Direction.** To examine the meaning of the dominant direction, we compare two quantities for each action. The behavior loading $(Su)_a$ measures how movement along PC1 changes the relative logit of action $a$. The state response $\Delta\xi_a$ measures the action-dependent displacement along the same principal direction.

Table 16 reports the main responses. In Melting Pot, `STEP_LEFT` and `FIRE_CLEAN` have the largest positive behavior loadings and positive state displacements. `FORWARD`, `FIRE_ZAP`, `STEP_RIGHT`, and `TURN_LEFT` lie on the negative side of the same direction. `TURN_RIGHT` differs from these actions: its displacement along PC1 is close to zero, while its transverse displacement is the largest among the reported actions.

In PARTNR, positive loadings are concentrated on `PICK`, `PLACE`, and `OPEN` actions, whereas joint actions containing `NAV` have negative loadings. The same sign pattern appears in the action-dependent state displacements.

The zero behavior loading for `WAIT|WAIT` follows from its role as the reference category in the categorical behavior model.

The transverse response of `TURN_RIGHT` provides an additional view of the Melting Pot dynamics. Starting from the post-jump transverse component and propagating the continuous flow without further jumps, its normalized magnitude decreases from 1.000 at horizon 0 to 0.884 at horizon 1, 0.543 at horizon 5, 0.298 at horizon 10, and 0.093 at horizon 20. This gives the learned flow a short-lived transverse response around the dominant trajectory direction.

Table 16: Behavior loadings and action-dependent state responses along the dominant latent direction. PARTNR rows are restricted to actions observed in all six development folds with a minimum fit count of at least 10.

| **Melting Pot** | | | |
|---|---|---|---|
| Action | $(Su)_a$ | $\Delta\xi_a$ | $\|\Delta z_{\perp,a}\|_2$ |
| STEP_LEFT | 0.658 | 0.700 | 0.171 |
| FIRE_CLEAN | 0.652 | 0.658 | 0.152 |
| TURN_RIGHT | 0.263 | 0.003 | 0.334 |
| NOOP | 0.000 | -0.392 | 0.162 |
| FIRE_ZAP | -0.195 | -0.427 | 0.156 |
| BACKWARD | -0.221 | -0.369 | 0.145 |
| STEP_RIGHT | -0.227 | -0.394 | 0.170 |
| TURN_LEFT | -0.240 | -0.343 | 0.176 |
| FORWARD | -0.464 | -0.555 | 0.116 |

| **PARTNR** | | | | |
|---|---|---|---|---|
| Joint action | $(Su)_a$ | SD | $\Delta\xi_a$ | Min. fit count |
| WAIT\|PLACE | 0.302 | 0.025 | 1.155 | 268 |
| WAIT\|OTHER | 0.296 | 0.026 | 1.624 | 30 |
| PLACE\|WAIT | 0.295 | 0.025 | 1.497 | 361 |
| PICK\|WAIT | 0.268 | 0.019 | 1.758 | 363 |
| WAIT\|PICK | 0.255 | 0.016 | 1.530 | 263 |
| OPEN\|WAIT | 0.187 | 0.030 | 1.762 | 170 |
| WAIT\|OPEN | 0.184 | 0.029 | 1.515 | 116 |
| WAIT\|WAIT | 0.000 | 0.000 | -3.434 | 244 |
| WAIT\|NAV | -0.090 | 0.012 | -3.765 | 470 |
| NAV\|WAIT | -0.103 | 0.021 | -3.727 | 750 |