HAWKES-AUGMENTED WORLD MODEL FOR EVENT-DRIVEN REINFORCEMENT LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

World models enable reinforcement learning agents to improve policies through imagined latent trajectories. In environments with temporally clustered events, recent event history can provide useful context for predicting subsequent dynamics and rewards. We introduce the *Hawkes-Augmented World Model* (HAWM), which equips a recurrent stochastic state-space model with learned event inference and a Hawkes-style occurrence prior. An observation-conditioned posterior infers latent event occurrences and types from observed transitions, while a latent-conditioned prior predicts them during imagination. Both pathways share an exponentially decaying memory of binary event occurrences. The prior combines this memory with latent context to predict subsequent occurrence probabilities, while event types are predicted separately to condition the world model's prediction heads. The event-memory embedding also conditions recurrent dynamics and actor–critic learning, allowing accumulated event history to inform prediction and control. An auxiliary future-reward objective encourages the event representation to capture information relevant to subsequent rewards. HAWM thus introduces an explicit event-history state whose evolution can be inferred from observations and predicted during observation-free imagination.

1 INTRODUCTION

World models enable reinforcement learning (RL) agents to improve policies through imagined experience, reducing their reliance on environment interactions (Hafner et al., 2023). Their effectiveness depends on retaining information that helps predict the consequences of future actions. In interactive tasks, this information can include events whose relevance extends beyond a single transition. In robotic manipulation, for example, establishing or losing contact with an object can alter which subsequent motions are feasible, even when the visual difference between adjacent observations is small. Such transitions motivate learning event representations that preserve information about consequential changes and their influence on future dynamics and rewards.

Recurrent world models can encode these changes in latent summaries of observation and action histories. However, their standard predictive objectives do not prescribe a separate representation of event occurrences. Explicit event variables provide an inductive bias for retaining such changes, which may be useful when their contribution to general prediction objectives is limited. Related approaches (Peng et al., 2026; Kim et al., 2019; Gumbsch et al., 2024) introduce such structure through event-aware representation objectives and learned temporal abstractions (see Appendix A for detailed related works). Building on these directions, we investigate a complementary role for events: an accumulated occurrence history that contributes directly to subsequent event probabilities and remains part of the predictive state during imagination. The resulting question is whether explicitly modeling this history can improve the predictions used for control.

Constructing this memory requires learning both which transitions to record and how to propagate their history. Event annotations are unavailable in our setting, and the magnitude of an observation change (Gumbsch et al., 2024; Peng et al., 2026) alone does not establish its relevance to future prediction (see Figure 1). Event inference must therefore be trained through objectives that connect detected changes to predictive outcomes. However, during imagination, the observations used for inference are unavailable. The model must instead generate events from latent information and carry their accumulated history through the rollout.

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.

The temporal structure of this memory is motivated by tasks in which interactions occur in clusters. Recent occurrences may then signal an elevated likelihood of further events, with this influence diminishing over time. We express this hypothesis through the exponential-history structure of a Hawkes process (Hawkes, 1971). When an event occurs, it can temporarily increase the probability of related future events. This effect is strongest soon after the event and gradually fades as time passes. Combining this memory with latent context makes the history contribution explicit in occurrence prediction while preserving dependence on the current situation. This provides a specific inductive bias for event-history modeling within the broader latent dynamics.

We integrate this memory into the *Hawkes-Augmented World Model* (HAWM), which augments the recurrent state-space model (RSSM). An observation-conditioned posterior learns event occurrence and type representations from observed transitions through predictive objectives. A latent-conditioned prior learns to match these distributions and generates event updates during imagination. Both pathways share the memory update and embedding, so inferred and imagined occurrences contribute to history according to the same temporal rule. The resulting memory embedding conditions recurrent dynamics, prediction heads, and the actor and critic, allowing event history to participate in future prediction and action selection throughout primitive-step rollouts. Our contributions are threefold:

- We introduce a Dreamer-style world model with an explicit binary event-occurrence state and a positive exponential memory that is shared between observation-conditioned inference and latent imagination.
- We develop a training scheme that learns event occurrence and type without event annotations, using world-model prediction, future-reward supervision, event-rate control, and posterior-to-prior matching. We explicitly evaluate free-running prior calibration and distinguish sparse event memory from dense event-feature side channels.
- We evaluate the method on seven robotic-control tasks and controlled scene/illumination shifts, with matched same-backbone memory baselines, quantitative event diagnostics, per-task ablations, and compute reporting.

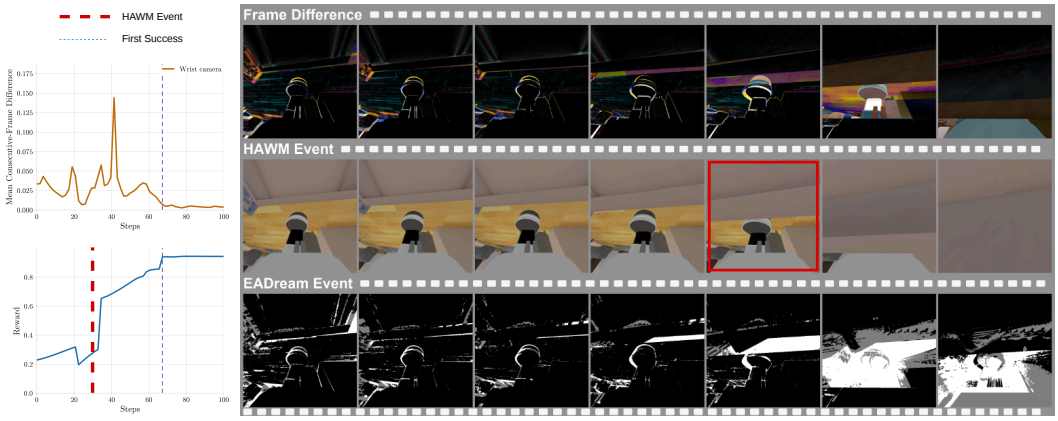

Figure 1: **Reward relevance versus pixel-level visual surprise.** On a `CloseKitchenCounter` trajectory collected by a trained checkpoint, HAWM identifies an interaction transition around the end effector and counter region before the later reward increase. The red dashed line marks the detected HAWM event, while the blue dashed line marks the first-success timestep. In contrast, EADream (Peng et al., 2026) constructs visual events from deviations under a per-pixel Adaptive Gaussian Mixture Model (AGMM). In this egocentric manipulation setting, camera motion and parallax cause widespread pixel-level surprise over background regions, producing event responses that are not well localized to the gripper-counter interaction. This qualitative example illustrates that pixel-level surprise can differ from reward-relevant task progress.

2 PRELIMINARIES & PROBLEM FORMULATION

2.1 PROBLEM SETTING

We consider a partially observed control problem with trajectories $\tau = \{o_t, a_t, r_t, d_t\}_{t=1}^{T}$, where o_t, a_t, r_t, and $d_t \in \{0, 1\}$ denote the observation, action, reward, and continuation indicator, respectively. Since the environment state is not directly available, the policy selects actions according to the interaction history, $\pi(a_t \mid x_{\leq t}, a_{<t})$. The objective is to maximize the expected discounted return:

$$J(\pi) = \mathbb{E}_{\tau\sim\pi}\left[\sum_{t=1}^{T} \gamma^{t-1} r_t\right], \tag{1}$$

where γ is the discount factor. The agent stores interaction trajectories in a replay buffer $\mathcal{D}$ and learns a world model from the recorded observations, actions, rewards, and continuation indicators. The world model generates imagined trajectories for policy and value learning. No event-level annotations are provided; event representations must be learned from the available trajectories and support prediction without future observations during imagination.

2.2 HAWKES PROCESSES

A univariate Hawkes process models an event rate that depends on previous event times $\mathcal{H}_t = \{t_i \mid t_i < t\}$ Hawkes (1971). With an exponential kernel, its conditional intensity is

$$\lambda(t \mid \mathcal{H}_t) = \mu + \sum_{t_i<t} \alpha \exp[-\beta(t - t_i)], \tag{2}$$

where $\mu > 0$ is the baseline intensity, $\alpha \geq 0$ is the excitation, and $\beta > 0$ is the decay rate. The accumulated excitation is

$$M(t) = \sum_{t_i<t} \alpha \exp[-\beta(t - t_i)], \qquad \lambda(t \mid \mathcal{H}_t) = \mu + M(t). \tag{3}$$

We refer to $M(t)$ as the Hawkes memory. For binary events $y_t \in \{0, 1\}$ on unit-spaced steps, an exponential event trace obeys

$$M_t = \bar{M}_t + \alpha y_t, \qquad \bar{M}_t = \exp(-\beta) M_{t-1}, \tag{4}$$

where $\bar{M}_t$ is the memory before incorporating the current event. Under a piecewise-constant intensity approximation, the probability of at least one event in an interval of unit length is

$$p_t^{\text{evt}} = 1 - \exp(-\lambda_t), \qquad \lambda_t = \mu + \bar{M}_t. \tag{5}$$

HAWM uses this exponential-memory structure in a discrete-time, state-conditioned event prior whose event sequence is learned jointly with the world model.

3 METHOD

We propose the *Hawkes-Augmented World Model* (HAWM). HAWM learns sparse events from observed transitions, summarizes their history with a decaying memory, and uses this information to support prediction and recurrent dynamics. A Hawkes posterior infers events when observations are available, while a Hawkes prior predicts events from latent transitions during imagination.

3.1 HAWM FRAMEWORK

Components. HAWM builds on the recurrent state-space model of DreamerV3 Hafner et al. (2023). Let h_t and z_t denote its deterministic and stochastic states, respectively, with $s_t = [h_t; z_t]$. The event posterior $\mathcal{H}^+$ and prior $\mathcal{H}^-$ each provide an event-memory embedding, an occurrence probability, and a distribution over K event types. The embedding conditions recurrent dynamics and prediction, while the prediction heads additionally receive occurrence and type probabilities. The components

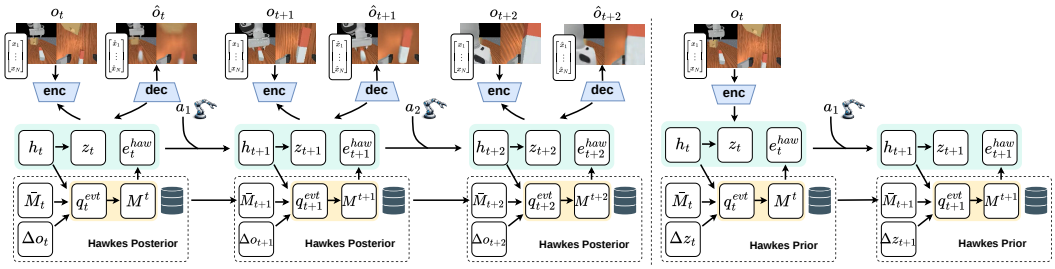

Figure 2: **HAWM carries event history from observed experience into imagined rollouts.** An interaction can help predict later transitions even when it is no longer visible in the current observation. HAWM retains this information in a decaying event memory alongside the RSSM state. Observed transitions teach the model which events to record (left); predicted events extend that history during imagination (right), using the same learned memory dynamics. action a_1 need to be fix

are summarized as follows:

$$\text{RSSM} \begin{cases} \text{Recurrent model:} & h_t = f_\theta\big(h_{t-1}, z_{t-1}, a_{t-1}, (e^{\text{haw}}_{t-1})^+\big) \\ \text{Rep. model:} & z_t \sim q_\phi(z_t \mid h_t, o_t) \\ \text{Dynamics prior:} & \hat{z}_t \sim p_\theta(z_t \mid h_t) \\ \text{Predictors:} & \hat{u}_t \sim p_\theta\big(u_t \mid s_t, (e^{\text{haw}}_t)^+, \mathbf{c}_t, q^{\text{evt}}_t\big), \quad u \in \{r, d, o\}, \\ \text{Horizon Predictors:} & R^{(H)}_t \sim p_\omega(R^{(H)}_t \mid \text{sg}(z_t), (e^{\text{haw}}_t)^+, \mathbf{c}_t, q^{\text{evt}}_t) \end{cases} \tag{6}$$

$$\text{Event} \begin{cases} \text{Hawkes posterior:} & \big(e^{\text{haw}}_t, M_t, \lambda_t\big)^+, \mathbf{c}_t, q^{\text{evt}}_t = \mathcal{H}^+\big(\text{sg}(h_t), \Delta o_t, M_{t-1}\big) \\ \text{Hawkes prior:} & \big(e^{\text{haw}}_t, M_t, \lambda_t\big)^-, \hat{\mathbf{c}}_t, p^{\text{evt}}_t = \mathcal{H}^-\big(\text{sg}(h_t), \Delta z_t, M_{t-1}\big) \end{cases}$$

Here, $u \in \{r, d, o\}$ indexes reward, continuation, and observation prediction, and $R^{(H)}_t$ is an auxiliary H-step discounted reward target. The prediction heads are shown with posterior inputs for observed training. During imagination, they use $e^{\text{haw},-}_t$, $\hat{\mathbf{c}}_t$, and p^{evt}_t instead. The symbol z_t denotes the active latent sample in either mode, drawn from the representation posterior on observed trajectories and from the dynamics prior during imagination.

The inputs Δo_t and Δz_t summarize observation changes and latent-distribution changes, respectively; observation encoding is absorbed into the module definitions. The operators $\mathcal{H}^\pm$ include internal typed-event sampling and the corresponding memory update. They share the memory-update and embedding operations defined below. Their updated embeddings support current-step prediction and enter the next recurrent update. Thus, the current event first influences recurrent dynamics through h_{t+1}. Throughout, $\text{sg}(\cdot)$ denotes stop-gradient.

3.2 Hawkes Event Inference and Prediction

HAWM represents event history through a memory learned from posterior-inferred events on observed trajectories and reused with prior-predicted events during imagination.

Shared event memory. We introduce an explicit memory of event occurrences, following the exponential-history structure of Hawkes processes (Kim & Iwata, 2026; Jin & Huang, 2026).

Definition 1 (Event memory) *For binary event occurrences $y_t \in \{0, 1\}$ and learned parameters $\alpha > 0$ and $\beta > 0$, the event memory is initialized with $M_0 = 0$ at each episode boundary and updated at unit-spaced transitions as*

$$\bar{M}_t = \exp(-\beta) M_{t-1}, \qquad M_t = \bar{M}_t + \alpha y_t, \tag{7}$$

where $\bar{M}_t$ and M_t denote the memory before and after incorporating the current event, respectively. The parameters α and β control the contribution of each occurrence and its decay rate.

Remark 1 (Interpretation as decaying event history) *Unrolling Eq. (7) gives*

$$\bar{M}_t = \alpha \sum_{j=1}^{t-1} \exp[-\beta(t-j)] y_j. \tag{8}$$

Thus, the recursive update summarizes previous event occurrences through exponentially decaying weights. Recent occurrences contribute more strongly, while the current event is excluded from $\bar{M}_t$. When this memory enters the prior intensity in Eq. (13), a larger $\bar{M}_t$ increases the occurrence probability at a fixed latent context.

To condition world-model predictions and the next recurrent update, we embed the updated memory together with the current intensity parameter:

$$e_t^{\mathrm{haw}} = \Omega([M_t; \log(1+\lambda_t)]) \,. \tag{9}$$

During observed training, posterior-inferred events update the memory, and predictive objectives train its parameters and readout. During imagination, prior-predicted events update the memory using the same α, β, and Ω. Inferred and imagined events therefore contribute to history according to the same learned temporal rule. We use M_t for the active trajectory's memory; superscripts $+$ and $-$ distinguish posterior- and prior-conditioned intensities and embeddings. This scalar memory tracks event occurrence; event types condition the prediction heads separately.

Observation-conditioned posterior. When observations are available, their changes provide evidence for event inference. We summarize consecutive encoded observations by a change feature Δo_t containing their difference and its magnitude. Combining this feature with recurrent context and pre-event memory gives

$$\lambda_t^{+} = \mathrm{softplus}\big(b^{+} + \bar{M}_t + g_\psi(\mathrm{sg}(h_t), \Delta o_t)\big) \,, \tag{10}$$

$$q_t^{\mathrm{evt}} = 1 - \exp(-\lambda_t^{+}), \tag{11}$$

$$\mathbf{c}_t = C_\phi\big(\mathrm{sg}(h_t), \mathrm{sg}(\Delta o_t), \mathrm{sg}(q_t^{\mathrm{evt}})\big) \in \Delta^{K-1}. \tag{12}$$

Here, b^{+} is a learned baseline, g_ψ adjusts the intensity using observation changes, and C_ϕ applies a softmax over K types. Detaching the type-network inputs blocks gradients from the type branch into the occurrence detector and its input features. We use *posterior* to denote this learned, observation-conditioned inference pathway.

We sample $y_t \sim \mathrm{Bernoulli}(q_t^{\mathrm{evt}})$ and use a straight-through gradient estimator during training. Applying Eqs. 7 and 9 updates M_t and produces $e_t^{\mathrm{haw},+}$, completing $\mathcal{H}^{+}$. The sample updates the memory, while the prediction heads also receive the soft occurrence and type probabilities directly.

Latent-conditioned prior. During imagination, observations are unavailable, so the prior predicts events from latent context and the accumulated memory. Its change feature Δz_t is the detached difference between consecutive RSSM categorical log-probability vectors after uniform mixing. These vectors come from the latent posterior during observed training and from the dynamics prior during imagination. Combining this distributional change with recurrent context and decayed memory gives

$$\begin{aligned} \lambda_t^{-} &= \mathrm{softplus}\big(b^{-} + \bar{M}_t + g_\eta(\mathrm{sg}(h_t), \Delta z_t)\big) \,, \\ p_t^{\mathrm{evt}} &= 1 - \exp(-\lambda_t^{-}), \\ \hat{\mathbf{c}}_t &= P_\vartheta\big(\mathrm{sg}(h_t), \Delta z_t, \mathrm{sg}(p_t^{\mathrm{evt}})\big) \in \Delta^{K-1}. \end{aligned} \tag{13}$$

Here, b^{-} is a learned baseline, g_η adjusts the intensity using latent context, and P_ϑ predicts the type distribution with a softmax. These parameters are separate from their posterior counterparts, and latent inputs are detached during prior fitting. The history term $\bar{M}_t$ contributes the exponentially decaying excitation in Eq. (8). At a fixed latent context, this term increases the occurrence probability. We adopt this positive dependence as an inductive bias for interaction phases in which consequential transitions may occur close together, such as tool engagement followed by cube displacement or successive alignment corrections during insertion. The context term can lower the overall intensity as the task state changes, although the explicit history contribution remains excitatory.

Sampling $y_t \sim \mathrm{Bernoulli}(p_t^{\mathrm{evt}})$ during imagination supplies the same memory-update and embedding operations, producing the updated M_t and $e_t^{\mathrm{haw},-}$. These operations define $\mathcal{H}^{-}$: its event features support current predictions, and its updated memory is carried into the next transition. This is a scalar event-occurrence model with a Hawkes-style history term; event types do not introduce separate excitation channels. The following subsection describes how prediction objectives train the posterior and posterior matching trains the prior.

3.3 World-Model and Behavior Learning

Prediction objectives. Let $\mathcal{V}$ denote the valid transitions in sampled replay trajectories. Observed training uses posterior event features to fit the observation, reward, and continuation heads:

$$\mathcal{L}_{\text{pred}} = -\frac{1}{|\mathcal{V}|} \sum_{t \in \mathcal{V}} \sum_{u \in \{r,d,o\}} \log p_\theta \Big(u_t \mid s_t, e_t^{\text{haw},+}, \mathbf{c}_t, q_t^{\text{evt}} \Big) . \tag{14}$$

We retain the standard DreamerV3 latent dynamics and representation losses, denoted together by $\mathcal{L}_{\text{RSSM}}$, to align the stochastic latent posterior and prior. All three prediction heads can shape the event features through their differentiable input paths. An auxiliary head supplies an additional learning signal from the H-step discounted future reward:

$$\mathcal{L}_{\text{future}} = -\frac{1}{|\mathcal{U}|} \sum_{t \in \mathcal{U}} \log p_\omega \Big(R_t^{(H)} \mid \text{sg}(z_t), e_t^{\text{haw},+}, \mathbf{c}_t, q_t^{\text{evt}} \Big) , \quad R_t^{(H)} = \sum_{j=0}^{H-1} \gamma^j r_{t+j}. \tag{15}$$

The set $\mathcal{U} \subseteq \mathcal{V}$ contains transitions whose complete prediction horizon remains within the same episode. Future rewards are prediction targets and are not supplied to the event posterior. Detaching z_t blocks the direct gradient through this latent input. The auxiliary loss can still update event inference, the memory embedding, and other parameters reached through the event-feature paths.

Event regularization and prior fitting. A small target event rate $\rho \in (0, 1)$ controls the average frequency of binary memory updates:

$$\mathcal{L}_{\text{rate}} = D_{\text{KL}}(\text{Bernoulli}(\rho) \,\|\, \text{Bernoulli}(\bar{q})) , \quad \bar{q} = \frac{1}{|\mathcal{V}|} \sum_{t \in \mathcal{V}} q_t^{\text{evt}}. \tag{16}$$

This regularizer controls binary event frequency; the prediction heads continue to receive dense occurrence and type probabilities. The prior matches detached posterior predictions through

$$\mathcal{L}_{\text{haw}} = \frac{1}{|\mathcal{V}|} \sum_{t \in \mathcal{V}} D_{\text{KL}} \big(\text{Bernoulli}(\text{sg}(q_t^{\text{evt}})) \,\|\, \text{Bernoulli}(p_t^{\text{evt}}) \big) , \tag{17}$$

$$\mathcal{L}_{\text{type}} = \frac{1}{|\mathcal{V}|} \sum_{t \in \mathcal{V}} D_{\text{KL}}(\text{sg}(\mathbf{c}_t) \,\|\, \hat{\mathbf{c}}_t) . \tag{18}$$

Type alignment is applied to all valid transitions because the prediction heads use type probabilities even when no binary event is realized. For occurrence-prior fitting, the memory history is constructed from detached posterior event samples. Gradients can update the prior and shared excitation parameters without updating the detector through this history. The world-model prediction path retains a differentiable memory update so that event inference can receive learning signals from subsequent recurrent states.

Joint objective. The complete world-model objective is

$$\mathcal{L}_{\text{HAWM}} = \mathcal{L}_{\text{pred}} + \mathcal{L}_{\text{RSSM}} + w_{\text{future}} \mathcal{L}_{\text{future}} + w_{\text{rate}} \mathcal{L}_{\text{rate}} + w_{\text{haw}} \mathcal{L}_{\text{haw}} + w_{\text{type}} \mathcal{L}_{\text{type}}, \tag{19}$$

where the w coefficients are nonnegative loss weights. Prediction and event-rate objectives train the observed event pathway, while detached posterior matching trains the occurrence and type priors. Shared memory and embedding parameters are learned through the paths described above.

Behavior learning. The policy $\pi(\cdot \mid s_t, e_t^{\text{haw}})$ and value function $V(s_t, e_t^{\text{haw}})$ are optimized on imagined trajectories generated by the latent and event priors. The reward and continuation distributions in Eq. (6) are evaluated using the prior features $(e_t^{\text{haw}})^-$, $\hat{\mathbf{c}}_t$, and p_t^{evt}. Their decoded reward predictions and continuation probabilities are used to construct imagined returns for the DreamerV3 actor-critic objectives.

During environment interaction, the same policy receives observation-conditioned latent states and posterior event-memory embeddings, preserving its input structure across both phases. The auxiliary horizon-reward head is used only for world-model training and does not contribute to imagined returns or actor–critic targets.

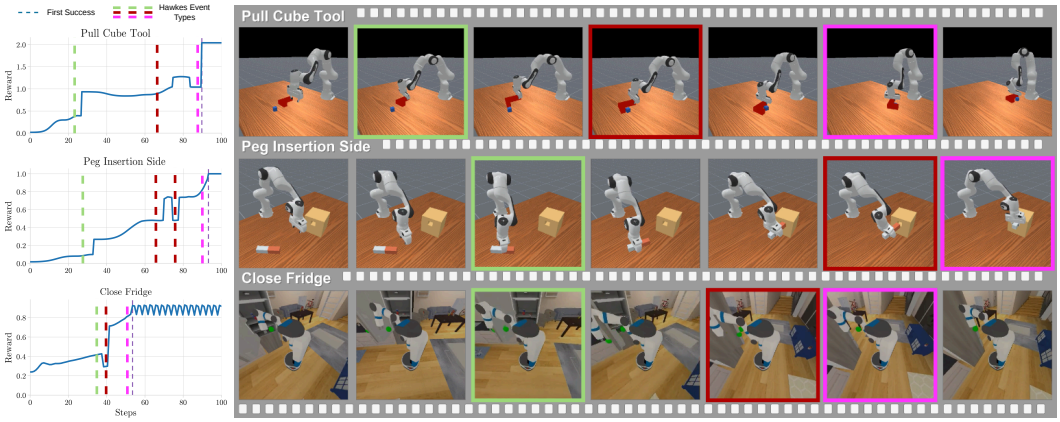

Figure 3: **Predicted events during successful manipulation trajectories.** Each row presents a representative rollout alongside its environment reward. Green, red, and magenta dashed lines indicate predicted event times of different types, with matching borders identifying the corresponding observations. The blue dashed line marks the first task success. Predicted events align with distinct interaction stages and subsequent changes in reward.

4 Experimental Evaluations

We evaluate whether explicitly modeling sparse event-time structure improves both the interpretability and control performance of a latent world model. Our experiments address three questions:

- **Does HAWM discover meaningful events?** We evaluate whether the events learned by HAWM align with semantically salient transitions and changes in task progression, despite the lack of event-level supervision.
- **Can event prediction in HAWM enhance model-based reinforcement learning (MBRL)?** We show that HAWM achieves higher returns than other baselines in multi-stage table-top manipulation and mobile robot manipulation benchmarks.
- **Which components of HAWM contribute to policy performance?** We systematically examine the roles of Hawkes-style event memory, future-reward supervision, and learned event types, and assess how the size of the event vocabulary affects performance.

Benchmarks. We evaluate seven tasks across two robotic-control suites. ManiSkill3 (Tao et al., 2025) provides four tabletop manipulation tasks: `PickCube`, `PlaceSphere`, `PullCubeTool`, and `PegInsertionSide`. MS-HAB (Shukla et al., 2025) provides three mobile-manipulation tasks in ReplicaCAD scenes (Szot et al., 2021): `NavigateAndPick`, `CloseFridge`, and `CloseKitchenCounter`.

Baselines. We compare against DreamerV3 (Hafner et al., 2023), the backbone baseline; EAD-ream (Peng et al., 2026), an event-aware Dreamer variant; and TD-MPC2 (Hansen et al., 2024), an alternative approach to control through latent dynamics.

4.1 Interpretable Event Discovery

Figure 3 illustrates the events predicted by HAWM on three representative manipulation trajectories. In `PullCubeTool`, the first event coincides with establishing contact with the tool and the first substantial reward increase. The subsequent event occurs as the tool engages and displaces the cube, while the final event captures the transition toward the target configuration immediately before task success. In `PegInsertionSide`, the first event occurs when the peg is acquired, followed by events during the approach and alignment with the insertion target. The repeated event type during this stage reflects the corrective interactions required for precise alignment, and the final event coincides with successful insertion. In `CloseFridge`, the events are more closely spaced because effective closing requires sustained interaction with the articulated door. They mark initial contact, the onset of meaningful door motion, and the transition to the closed configuration, after

which the reward remains high despite small variations in the door state. Across the three tasks, the predictions capture both intermediate interaction outcomes and terminal completion. Their temporal distribution also reflects the task structure: events are separated across distinct manipulation stages in `PullCubeTool` and `PegInsertionSide`, but cluster around a short period of continuous contact in `CloseFridge`.

4.2 Model-based Reinforcement Learning

We evaluate whether Hawkes-aware event modeling improves policy learning on table-top and mobile-robot manipulation tasks.

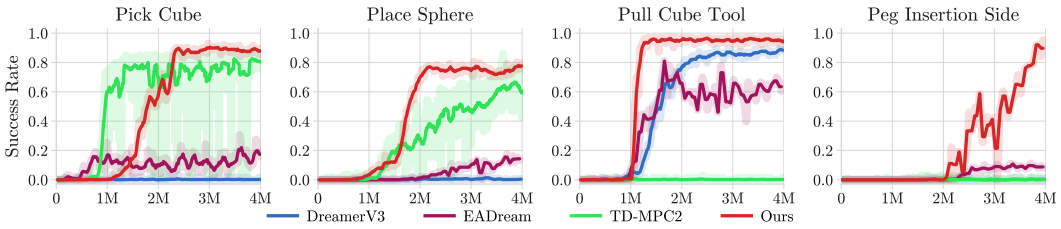

Figure 4: Success rates comparisons on four multi-stage table-top manipulation tasks.

Multiple-stage Table-Top Manipulation. As shown in Figure 4, our method achieves the strongest final performance across all four tasks. The gains are most pronounced on `PullCubeTool` and `PegInsertionSide`, where success depends on a sequence of coordinated interactions. In `PegInsertionSide`, successful insertion requires preserving the consequences of earlier grasping and alignment decisions until the final contact, while the reward provides limited guidance during the intermediate motion. Our method identifies these consequential transitions as events and carries their decaying influence through the latent dynamics. This event-level context supports a coherent progression from grasping to alignment and insertion.

`PullCubeTool` presents a complementary challenge: the agent must first establish effective contact with the tool and then translate that interaction into the desired motion of the cube. The relevant consequence of tool contact is therefore delayed and mediated by another object. By retaining the influence of contact and displacement events, our method better connects the initial tool interaction to the subsequent cube motion. This enables more reliable reasoning about tool-mediated dynamics and explains the clear advantage over the baselines on this task.

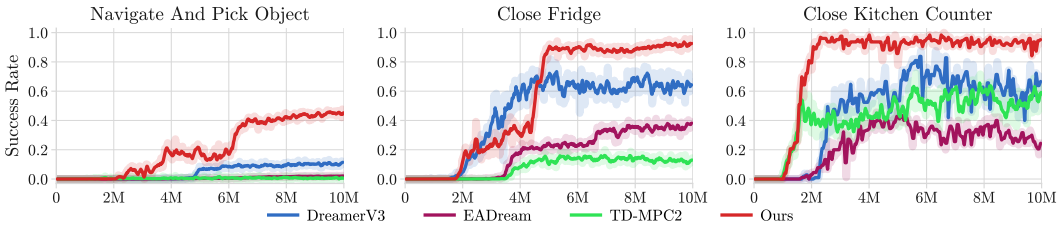

Figure 5: Success rates comparisons on three mobile-robot manipulation tasks. For `NavigateAndPick`, we use the most challenging setting, training a single policy jointly across nine YCB objects. In each episode, the robot must pick a target selected from these objects, with its position provided as input.

Mobile-Robot Manipulation. Figure 5 shows that the improvements extend to room-scale mobile manipulation. `NavigateAndPickObject` highlights a different challenge: long navigation segments contain many weakly informative transitions, whereas success depends on a small number of events surrounding target acquisition and grasping. Conditioning the world model on event history reduces the dilution of these interactions by the preceding navigation sequence. Consequently, imagined trajectories better connect reaching the target with the subsequent manipulation outcome.

In `CloseFridge`, the agent must establish contact with the handle and maintain an appropriate force direction as the door rotates. The visual state changes continuously, but only a subset of transitions indicates effective contact or meaningful door motion. Retaining these events helps the world model distinguish productive closing behavior from motion that does not change the articulation, leading to more reliable policy optimization.

`CloseKitchenCounter` requires repeated coordination between the mobile base and arm while closing a sliding drawer. Contact can be intermittent, and progress is accumulated over several local corrections. The event-conditioned latent state preserves information about these partial advances, allowing imagined rollouts to represent the cumulative effect of successive pushes. The strong performance on both closing tasks therefore indicates that our method captures not only navigation-to-manipulation transitions, but also the fine-grained interaction dynamics of articulated objects.

4.3 Ablations

Table 1 evaluates the contributions of event-history modeling, future-reward supervision, and learned event types across four ManiSkill3 tasks. Among the component ablations, we show in "w/o Hawkes memory" variant that omitting Hawkes memory and using MLP as event predictor results in the largest performance drop. This suggests that event prediction becomes more useful when recent occurrences are accumulated and carried forward. The Hawkes-style memory provides this temporal context for subsequent prediction and policy learning.

Variant	Success Rate (%)
Full HAWM ($K = 2$)	86.6 ± 6.3
Full HAWM ($K = 3$)	$\mathbf{93.4 \pm 5.8}$
Full HAWM ($K = 4$)	83.1 ± 8.0
Full HAWM ($K = 5$)	72.8 ± 8.7
w/o future-reward	56.4 ± 6.1
w/o learned event types	64.9 ± 8.2
w/o Hawkes memory	34.9 ± 7.4
w/o Hawkes module (DreamerV3)	24.3 ± 7.5

Table 1: HAWM ablations averaged over four ManiSkill tasks after 4M environment steps per task. Yellow rows vary the number of event types K; green rows report component ablations. Bold indicates the best-performing configuration ($K = 3$), adopted as the default setting.

The usefulness of this memory also depends on which events the model learns to represent. Removing future-reward supervision leads to a larger degradation than removing learned event types, supporting the importance of connecting event discovery to subsequent task progress. The future-reward objective encourages event features to capture the consequences of intermediate interactions, providing a learning signal beyond observation reconstruction and immediate reward prediction. These results suggest complementary roles for event supervision and memory: the former guides the information captured by events, while the latter preserves their occurrence history for later use.

Within this reward-guided representation, learned event types further improve performance by providing distinctions useful for predicting observations and rewards. However, increasing the number of types beyond $K = 3$ does not improve performance, suggesting that these distinctions can be represented with a compact vocabulary at the evaluated training budget. The ablations therefore support combining a small set of reward-relevant event types with explicit occurrence memory, with the strongest contribution coming from how event history is modeled.

5 Conclusion

In this work, we presented HAWM, a DreamerV3-based world model that augments latent dynamics with an inferred binary event process and a positive, exponentially decaying occurrence memory. An observation-conditioned posterior constructs event histories from real trajectories, while a latent-conditioned prior extends those histories during imagination. Our experimental results suggest that this event state can improve control on the evaluated manipulation tasks, particularly when task progress depends on a sequence of interactions. The analysis also identifies important boundaries where the model uses a discrete-time Hawkes-style trace rather than a full point-process likelihood. Also, prior errors can compound during free-running imagination. Quantitative event diagnostics, same-backbone memory controls, and free-running calibration are therefore necessary to attribute gains specifically to self-exciting event history.

AI USE STATEMENT

AI tools were used only to assist with language editing and improve the clarity of the manuscript. All scientific content, analyses, results, and conclusions were produced and verified by the authors.

ETHICS STATEMENT

This work does not involve new human-subject data collection or deployment decisions. We use existing research datasets and follow their established research-use protocols.

REFERENCES

Berk Çalli, Aaron Walsman, Arjun Singh, Siddhartha S. Srinivasa, Pieter Abbeel, and Aaron M. Dollar. Benchmarking in manipulation research: The YCB object and model set and benchmarking protocols. *arXiv preprint arXiv:1502.03143*, 2015. doi: 10.48550/arXiv.1502.03143. URL https://arxiv.org/abs/1502.03143.

Tal Daniel, Carl Qi, Dan Haramati, Amir Zadeh, Chuan Li, Aviv Tamar, Deepak Pathak, and David Held. Latent particle world models: Self-supervised object-centric stochastic dynamics modeling. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=lTaPtGiUUc.

Christian Gumbsch, Noor Sajid, Georg Martius, and Martin V. Butz. Learning hierarchical world models with adaptive temporal abstractions from discrete latent dynamics. In *International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=TjCDNssXKU.

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 of *Proceedings of Machine Learning Research*, pp. 2555–2565. PMLR, 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. URL https://openreview.net/forum?id=S1lOTC4tDS.

Danijar Hafner, Timothy Lillicrap, Mohammad Norouzi, and Jimmy Ba. Mastering atari with discrete world models. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=0oabwyZbOu.

Danijar Hafner, Jurgis Pasukonis, Jimmy Ba, and Timothy Lillicrap. Mastering diverse domains through world models. *arXiv preprint arXiv:2301.04104*, 2023.

Nicklas Hansen, Hao Su, and Xiaolong Wang. TD-MPC2: Scalable, robust world models for continuous control. In *International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=Oxh5CstDJU.

Nicklas A. Hansen, Hao Su, and Xiaolong Wang. Temporal difference learning for model predictive control. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 8387–8406. PMLR, 2022.

Alan G Hawkes. Spectra of some self-exciting and mutually exciting point processes. *Biometrika*, 58 (1):83–90, 1971.

Songyao Jin and Biwei Huang. Causal structure learning in hawkes processes with complex latent confounder networks. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=mA78uXqcnl.

Hideaki Kim and Tomoharu Iwata. A representer theorem for hawkes processes via penalized least squares minimization. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=gJjRdLG5MY.

Taesup Kim, Sungjin Ahn, and Yoshua Bengio. Variational temporal abstraction. *Advances in neural information processing systems*, 32, 2019.

Paul Mattes, Rainer Schlosser, and Ralf Herbrich. Hieros: Hierarchical imagination on structured state space sequence world models. In *Proceedings of the 41st International Conference on Machine Learning*, 21–27 Jul 2024.

Hongyuan Mei and Jason M Eisner. The neural hawkes process: A neurally self-modulating multivariate point process. *Advances in neural information processing systems*, 30, 2017.

Vincent Micheli, Eloi Alonso, and Francois Fleuret. Transformers are sample-efficient world models. In *International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=vhFu1Acb0xb.

Zhao-Han Peng, Shaohui Li, Zhi Li, Shulan Ruan, Yu Liu, and You He. From observations to events: Event-aware world models for reinforcement learning. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=OWkkFaq1IZ.

Chao Qu, Xiaoyu Tan, Siqiao Xue, Xiaoming Shi, James Zhang, and Hongyuan Mei. Bellman meets hawkes: Model-based reinforcement learning via temporal point processes. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pp. 9543–9551, 2023.

Julian Schrittwieser, Ioannis Antonoglou, Thomas Hubert, Karen Simonyan, Laurent Sifre, Simon Schmitt, Arthur Guez, Edward Lockhart, Demis Hassabis, Thore Graepel, et al. Mastering atari, go, chess and shogi by planning with a learned model. *Nature*, 588(7839):604–609, 2020.

Arth Shukla, Stone Tao, and Hao Su. ManiSkill-HAB: A benchmark for low-level manipulation in home rearrangement tasks. In *International Conference on Learning Representations*, 2025.

Andrew Szot, Alexander Clegg, Eric Undersander, Erik Wijmans, Yili Zhao, John Turner, Noah Maestre, Mustafa Mukadam, Devendra Singh Chaplot, Oleksandr Maksymets, et al. Habitat 2.0: Training home assistants to rearrange their habitat. In *Advances in Neural Information Processing Systems*, 2021.

Stone Tao, Fanbo Xiang, Arth Shukla, Yuzhe Qin, Xander Hinrichsen, Xiaodi Yuan, Chen Bao, Xinsong Lin, Yulin Liu, Tse-Kai Chan, Yuan Gao, Xuanlin Li, Tongzhou Mu, Nan Xiao, Arnav Gurha, Viswesh Nagaswamy Rajesh, Yong Woo Choi, Yen-Ru Chen, Zhiao Huang, Roberto Calandra, Rui Chen, Shan Luo, and Hao Su. Demonstrating GPU parallelized robot simulation and rendering for generalizable embodied AI with ManiSkill3. In *Proceedings of Robotics: Science and Systems (RSS)*, Los Angeles, CA, USA, 2025. doi: 10.15607/RSS.2025.XXI.021.

Shengjie Wang, Shaohuai Liu, Weirui Ye, Jiacheng You, and Yang Gao. EfficientZero v2: Mastering discrete and continuous control with limited data. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 51041–51062. PMLR, 2024.

Wenlong Wang, Ivana Dusparic, Yucheng Shi, Ke Zhang, and Vinny Cahill. Drama: Mamba-enabled model-based reinforcement learning is sample and parameter efficient. In *International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=7XIkRgYjK3.

Weirui Ye, Shaohuai Liu, Thanard Kurutach, Pieter Abbeel, and Yang Gao. Mastering atari games with limited data. In *Advances in Neural Information Processing Systems*, volume 34, 2021.

Boxuan Zhang, Weipu Zhang, Zhaohan Feng, Wei Xiao, Jian Sun, Jie Chen, and Gang Wang. Mixture-of-world models: Scaling multi-task reinforcement learning with modular latent dynamics. In *The Fourteenth International Conference on Learning Representations*, 2026a. URL https://openreview.net/forum?id=qUQARlAx5y.

Shuzhe Zhang, Xin Zhu, Yinling Qian, and Qiong Wang. S2-hwm: Sparse event-structured hierarchical world model for long-horizon surgical robot manipulation. *arXiv preprint arXiv:2608.13103*, 2026b.

Weipu Zhang, Gang Wang, Jian Sun, Yetian Yuan, and Gao Huang. STORM: Efficient stochastic transformer based world models for reinforcement learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 36, pp. 27147–27166, 2023.

Gaoyue Zhou, Hengkai Pan, Yann LeCun, and Lerrel Pinto. DINO-WM: World models on pre-trained visual features enable zero-shot planning. In *Proceedings of the 42nd International Conference on Machine Learning (ICML)*, volume 267 of *Proceedings of Machine Learning Research*. PMLR, 2025.

Simiao Zuo, Haoming Jiang, Zichong Li, Tuo Zhao, and Hongyuan Zha. Transformer hawkes process. In *International conference on machine learning*, pp. 11692–11702. PMLR, 2020.

Supplementary: Hawkes-Aware Transformer World Models for Event-Driven Reinforcement Learning

A Related Work

Latent world models for control. Latent world models predict action consequences in compact representations to support planning and policy learning. PlaNet combines recurrent stochastic dynamics with online planning (Hafner et al., 2019), while Dreamer and its successors optimize actor-critic policies through imagined latent trajectories (Hafner et al., 2020; 2021; 2023). Complementary approaches use learned dynamics for online search: MuZero and the EfficientZero family employ tree search (Schrittwieser et al., 2020; Ye et al., 2021; Wang et al., 2024), and TD-MPC and TD-MPC2 combine task-oriented dynamics with terminal-value estimates for model predictive control (Hansen et al., 2022; 2024). Beyond control formulations, research has expanded the architectures and representations used for prediction. IRIS and STORM use Transformer dynamics (Micheli et al., 2023; Zhang et al., 2023), while Drama adopts Mamba-based sequence modeling (Wang et al., 2025). DINO-WM predicts pretrained visual features for goal-directed planning (Zhou et al., 2025), LPWM learns object-centric stochastic dynamics (Daniel et al., 2026), and Mixture-of-World Models combines shared dynamics with task-conditioned experts for multitask RL (Zhang et al., 2026a).

Event-aware and temporally structured world models. Event-aware world models use observation changes to guide representation learning. EAWM Peng et al. (2026) derives event targets from observations and combines event prediction with event-aware reconstruction objectives to shape the learned representations. A complementary line of work learns temporal structure within latent dynamics. VTA (Kim et al., 2019) infers temporal boundaries in a hierarchical recurrent state-space model, enabling abstract transitions that skip intermediate time steps. THICK (Gumbsch et al., 2024) constructs adaptive temporal abstractions by identifying sparse changes in latent context and modeling their evolution at a higher level. Temporal abstraction also supports behavior learning: HIEROS (Mattes et al., 2024) learns hierarchical policies through imagined latent trajectories at multiple time scales . Connecting event discovery with hierarchical control, S^2-HWM (Zhang et al., 2026b) uses learned event evidence to schedule high-level goals and construct variable-duration segments, over which it predicts subsequent boundary states, durations, and accumulated rewards.

Hawkes processes and event-driven prediction. Hawkes processes model temporal clustering through additive excitation, allowing past events to increase subsequent event intensities (Hawkes, 1971). Neural point processes extend the flexibility of this history dependence. Neural Hawkes uses a continuous-time LSTM to model excitation and inhibition across event types (Mei & Eisner, 2017), while Transformer Hawkes uses self-attention to capture dependencies within event histories (Zuo et al., 2020). These approaches learn predictive models from timestamped event sequences. Connecting event prediction to control, Bellman Meets Hawkes models asynchronous observations and actions through Hawkes dynamics with external interventions and incorporates these dynamics into a continuous-time RL formulation (Qu et al., 2023). Recent work further advances the estimation and structural analysis of Hawkes processes. Authors in (Kim & Iwata, 2026) develop an RKHS-based representer theorem for nonparametric triggering-kernel estimation, while authors in (Jin & Huang, 2026) study causal structure discovery in multivariate Hawkes processes with latent confounding subprocesses.

B Implementation Details

B.1 Environments and Agent Interfaces

We evaluate four tabletop tasks from ManiSkill3 (Tao et al., 2025): `PickCube`, `PlaceSphere`, `PullCubeTool`, and `PegInsertionSide`. These tasks cover grasping, placement, tool use, and insertion. We additionally evaluate `NavigateAndPick`, `CloseFridge`, and `CloseKitchenCounter` from MS-HAB (Shukla et al., 2025), using ReplicaCAD apartment scenes (Szot et al., 2021) and YCB objects (Çalli et al., 2015).

ManiSkill3. ManiSkill3 (Tao et al., 2025) is a GPU-parallelized benchmark for robot manipulation. Here we consider four tabletop tasks performed by a fixed-base 7-DoF Franka Emika Panda arm with a parallel-jaw gripper:

- `PickCube`: grasp a cube and move it to a randomly sampled 3D goal position, whose marker is not shown in the camera images. Success requires the cube to be within 2.5 cm of the goal while the robot is static.
- `PlaceSphere`: grasp a small sphere and place it in a shallow bin. Success requires the sphere to be released and at rest in the bin.
- `PullCubeTool`: grasp an L-shaped tool and use it to pull a cube lying beyond the arm's reach. Success requires the cube to be within 0.6 m of the robot base in the horizontal plane.
- `PegInsertionSide`: grasp a peg and insert it sideways into a hole in a box. Success requires the peg to be inserted at least halfway, with a 1.5 cm tolerance.

At every reset, object and goal poses are sampled uniformly within task-specific regions of the table, and Gaussian noise with a standard deviation of 0.02 rad perturbs the arm's home joint configuration. In `PegInsertionSide`, the peg's length and radius, together with a matching hole, are also sampled for each parallel environment when the scenes are built; training scenes are built once. Each task is trained for 4M environment steps.

MS-HAB. MS-HAB (Shukla et al., 2025) is a benchmark for low-level mobile manipulation in home rearrangement, built on ManiSkill3. A Fetch mobile manipulator operates in ReplicaCAD apartments (Szot et al., 2021) containing YCB objects (Çalli et al., 2015), and each household activity is decomposed into Pick, Place, Open, Close, and Navigate subtasks. ReplicaCAD provides 105 apartment layouts: five macro-variations, which change the arrangement of large furniture, each with 21 micro-variations, which change the placement of smaller furnishings. Here we consider three tasks:

- `NavigateAndPick` is the Pick subtask of the TidyHouse activity. We use this name because MS-HAB spawns the robot facing the target object up to 2 m away. The object is therefore often out of the arm's reach when an episode begins, and the robot must first drive its base toward the object before grasping it. A single policy is trained on all nine TidyHouse target objects; the target is resampled every episode, and its pose is given in the state. Success requires the target to be grasped, the end effector to be within 5 cm of its rest position, the joints to be near their rest configuration, the robot to be static, and the cumulative contact force on the robot to remain below 5,000 N.
- `CloseFridge` and `CloseKitchenCounter` are the Close subtask of the SetTable activity, applied to the fridge door and the kitchen-counter drawer, respectively. The door or drawer starts open, and the robot starts in front of it. Success requires the door or drawer to be within 1% of its closed limit, the end effector to be within 5 cm of its rest position, the joints to be at their rest configuration, the robot to be static, and the cumulative contact force to remain below 10,000 N.

Outside the scene-transfer experiment (Appendix B.3), we train on all 63 layouts in MS-HAB's training split. Each layout is assigned to three of the 189 parallel training environments, and the scenes are not rebuilt during training. At every reset, one of the layout's episode configurations is sampled uniformly, which fixes the placement of movable objects and, for `NavigateAndPick`, the target. The robot's base pose and joint configuration are then drawn uniformly from MS-HAB's precomputed, collision-free spawns, which perturb the base position, heading, and arm joints with clipped Gaussian noise. Each task is trained for 10M environment steps.

Observations and actions. At every step, the agent receives 128×128 RGB images and a low-dimensional state vector. The tabletop tasks use a fixed third-person camera, together with the Panda wrist camera for `PegInsertionSide`; the MS-HAB tasks use the Fetch head camera and the gripper-mounted hand camera. Images from multiple cameras are concatenated along the channel axis. We use single frames without depth or image augmentation; the RSSM supplies temporal context. The state vector contains the robot's joint positions and velocities, the pose of the tool center point (TCP), and the task quantities each benchmark provides: a grasp flag and the goal position in `PickCube`; a grasp flag and the bin position in `PlaceSphere`; none in `PullCubeTool` and `PegInsertionSide`; and, in MS-HAB, a grasp flag together with the target object's pose

(`NavigateAndPick`) or the handle position (Close tasks), expressed in the robot base frame. The scene-transfer experiment removes the grasp, object, and goal quantities. The Panda action is 8-dimensional: delta joint-position targets for the seven arm joints, at most 0.1 rad per step, and an absolute gripper target. The Fetch action is 13-dimensional: delta joint-position targets for the seven arm joints, the torso lift, and the head pan and tilt joints; a gripper target; and base forward and yaw velocity commands of up to 1 m/s and π rad/s. All actions are normalized to $[-1, 1]$ and executed by ManiSkill3's PD controllers. Physics is simulated at 100 Hz and actions are applied at 20 Hz, so each environment step spans five physics substeps; we use no action repeat.

Rewards and episodes. We use each benchmark's normalized dense reward without modification or event-specific shaping. It is a staged reward with terms for reaching, grasping, task progress, and success, divided by a task-specific constant. In MS-HAB, the reward also encourages the robot to return to its rest pose after the interaction; the Pick reward additionally penalizes contact forces on the robot, and the Close reward favors closing slowly. ManiSkill3 tasks keep their registered episode limits: 50 steps for `PickCube` and `PlaceSphere`, and 100 steps for `PullCubeTool` and `PegInsertionSide`. MS-HAB episodes are limited to 100 steps during training, following the MS-HAB SAC baseline, and to the benchmark's 200-step subtask horizon during evaluation. Episodes do not terminate early: neither success nor an MS-HAB failure condition, such as exceeding the contact-force limit, ends an episode, and success is evaluated at every step. The time limit is treated as a truncation rather than a terminal state, so values are bootstrapped through it.

B.2 Model Implementation and Training

HAWM implementation. HAWM augments the DreamerV3 RSSM (Hafner et al., 2023) with the event modules described in the main method section. The event-memory embedding conditions recurrent transitions and the actor and critic. Observation, reward, and continuation heads additionally receive the occurrence and type probabilities. Observed trajectories use posterior event features; imagined trajectories use prior event features with the same dimensions. The auxiliary future-reward head is used only during world-model training.

Each trajectory maintains a scalar memory of binary event occurrences. The excitation and decay parameters and memory-embedding network are shared between observed and imagined rollouts; posterior-inferred events update the memory during observation, and prior-predicted events update it during imagination. Memory resets at episode boundaries, not at replay-chunk boundaries. A chunk beginning within an episode therefore requires preceding context or carried state sufficient to reconstruct its memory and change features. Context is processed before scored transitions, and transitions without a valid predecessor are excluded from $\mathcal{V}$.

The observation-change feature applies T in Eq. (20) to consecutive observation encodings, as defined in Eq. (21). It remains differentiable for occurrence inference, whereas the type posterior receives a detached copy. The latent-change feature uses detached differences of uniformly mixed categorical log-probabilities, with posterior logits on observed trajectories and dynamics-prior logits during imagination. We use $\varepsilon = 10^{-8}$ for the magnitude calculation and the RSSM mixing coefficient $\upsilon = 0.01$. The first imagined change compares the new prior log-probabilities with those of the carried observed posterior; later changes compare consecutive prior log-probabilities.

Interaction budgets and optimization. One environment step is one action and resulting transition in one environment; steps are summed across parallel environments. Initial data collection counts toward the budget, and actions are not repeated. We train for 4M environment steps per ManiSkill3 task and 10M per MS-HAB setting.

Training configuration. The RSSM uses 32 categorical variables. Its deterministic state has width 6144 with 48 classes per variable for ManiSkill3, and width 8192 with 64 classes per variable for MS-HAB. The image encoder and decoder use CNNs alongside state features, with SiLU activations and RMS normalization. Reward and value predictions use 255-bin two-hot distributions, while continuation uses a Bernoulli distribution. The actor and critic are three-layer MLPs with 768 units for ManiSkill3 or 1024 for MS-HAB; the actor uses a bounded-normal action distribution. We use a learning rate of 4×10^{-5} with 1000 updates of warm-up followed by a constant schedule. Uniform replay stores up to 300K ManiSkill3 or 700K MS-HAB transitions. Each update samples 32

sequences of length 64, with 64 replayed transitions per collected environment step. Imagination lasts 15 steps, with discount $\gamma = 1 - 1/333$, return mixing coefficient 0.95, and actor entropy coefficient 3×10^{-4}. The RSSM uses dynamics and representation loss weights of 1 and 0.1, a free-nats threshold of 1 nat, and categorical uniform mixing of 0.01. Computation uses bfloat16, adaptive gradient clipping at 0.3, and a slow-value update rate of 0.02.

HAWM-specific configuration. HAWM uses a scalar occurrence memory and a 32-dimensional memory embedding. Full HAWM uses $K = 3$ event types; the type-count ablation also evaluates $K = 2$, $K = 4$ and $K = 5$. Posterior and prior occurrence and type networks have separate parameters and 256-unit hidden layers, while the memory-update parameters α, β and embedding network Ω are shared. The target event rate is $\rho = 0.03$, and the auxiliary future-reward horizon is $H = 15$. The learned positive parameters α and β are initialized at 0.1 and 1.0, respectively; the posterior and prior have separate learned baselines b^+ and b^-. The objective weights are $w_{\text{future}} = 0.1$ and $w_{\text{rate}} = w_{\text{haw}} = w_{\text{type}} = 1$. Event occurrences are sampled from Bernoulli distributions, with a straight-through estimator for posterior samples during training.

Baseline implementations. We compare with DreamerV3 (Hafner et al., 2023), EADream (Peng et al., 2026), and TD-MPC2 (Hansen et al., 2024). DreamerV3 provides the backbone comparison without HAWM's event modules; EADream supplies an event-aware world-model comparison; and TD-MPC2 supplies a planning-based latent-model comparison. We evaluate the methods on the same tasks at the interaction budgets above, using the observation, action, reward, and episode protocols described in this appendix.

B.3 Scene Generalization

We evaluate zero-shot scene transfer on `NavigateAndPick`, keeping the target object `004_sugar_box` fixed. Training uses five ReplicaCAD scene configurations, and evaluation uses 50 unseen scene configurations. The split is fixed across methods and training seeds. No policy adaptation is performed on the evaluation configurations.

B.4 Illumination Robustness

We evaluate the same `NavigateAndPick` task and target object `004_sugar_box` on the five training scene configurations. Training uses nominal illumination without lighting randomization. At evaluation, we scale the simulator's ambient, point, and directional light intensities by 0.1, 1.0, and 4.0 for dim, nominal, and bright conditions. The change is made during rendering, without image post-processing. For each scene, we evaluate ten matched initializations under all three conditions using the same frozen checkpoint. The scene and initial physical state are identical within each comparison; subsequent trajectories may differ as the policy responds to the changed observations.

B.5 Hardware.

Each run was on 1 NVIDIA H100 GPUs with 80GB of memory.

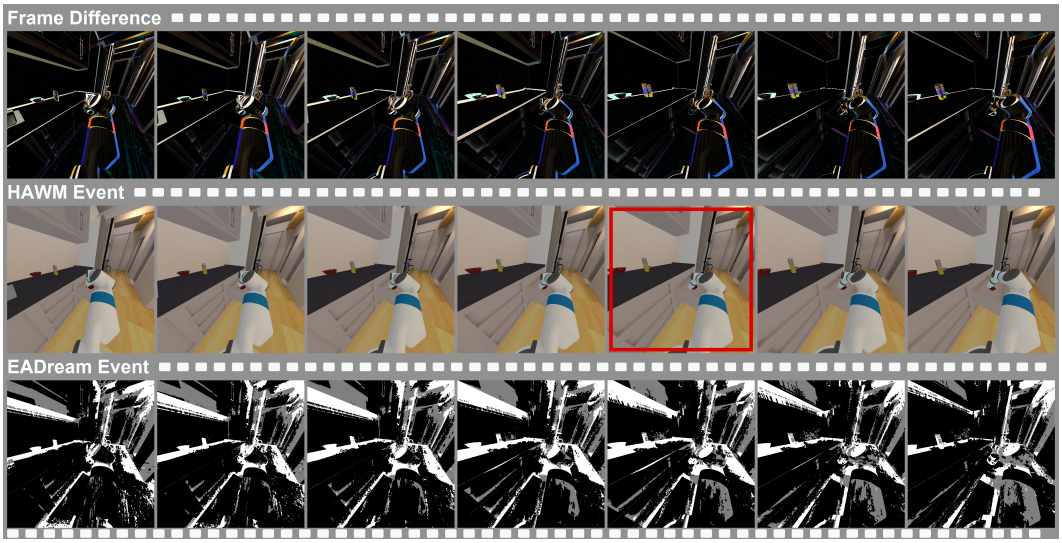

Figure 6: **Event detection from the head camera, sequence 1.** The red panel marks a HAWM event shortly before the object interaction. Frame differences and EADream's AGMM events respond to visual changes throughout the approach, while HAWM identifies a transition informative about subsequent task progress.

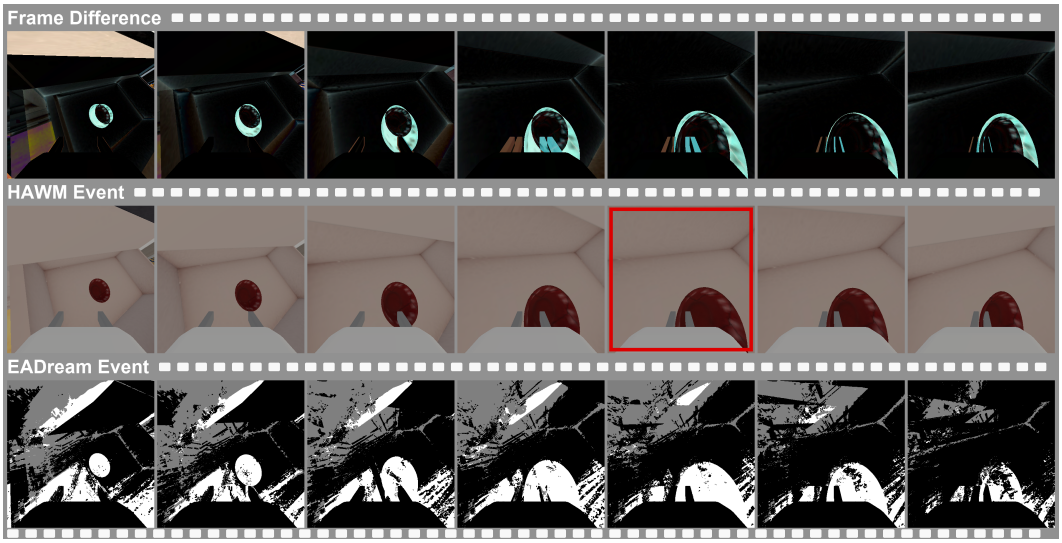

Figure 7: **Event detection from the wrist camera, sequence 1.** HAWM identifies the pre-interaction event despite substantial changes in the object's image position and scale. The egocentric motion produces widespread frame differences and AGMM events before contact.

C Additional Experimental Results

C.1 Trained Policy Event Detection

Figures 6–9 show a consistent distinction between visual change and predictive relevance. EADream generates visual event targets when pixels deviate from their learned AGMM appearance distributions. In moving head and wrist cameras, viewpoint changes and parallax can trigger these targets across the scene even when the underlying manipulation state changes little. HAWM instead learns event features partly through their ability to predict future reward. Its events occur before the consequential interactions in both sequences, suggesting that the model can retain information about emerging task progress before the resulting outcome becomes visually obvious.

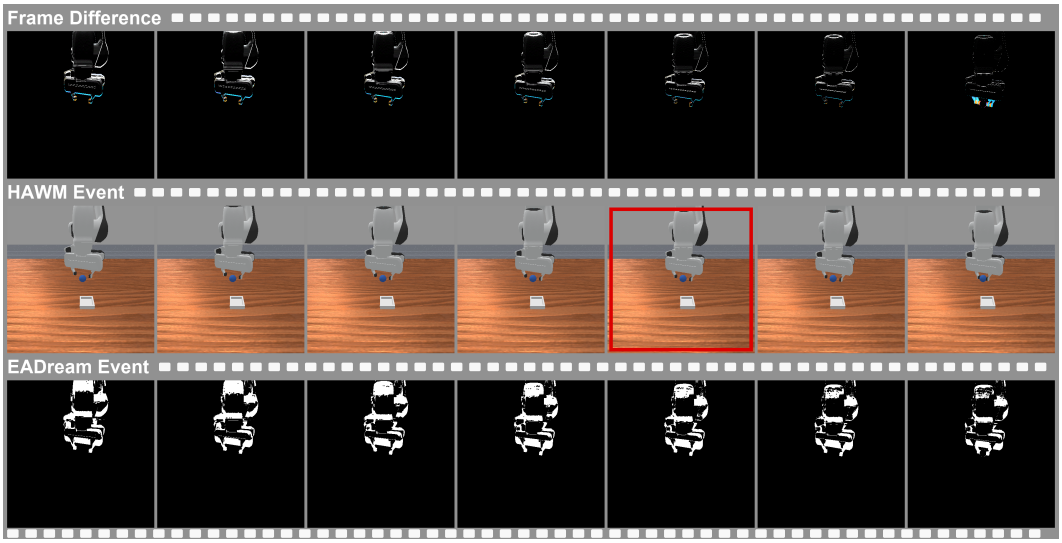

Figure 8: **Event detection from the head camera, sequence 2.** HAWM marks a transition preceding the sphere interaction. The corresponding visual change is localized, illustrating why the magnitude of frame differences alone need not reflect an event's importance for control.

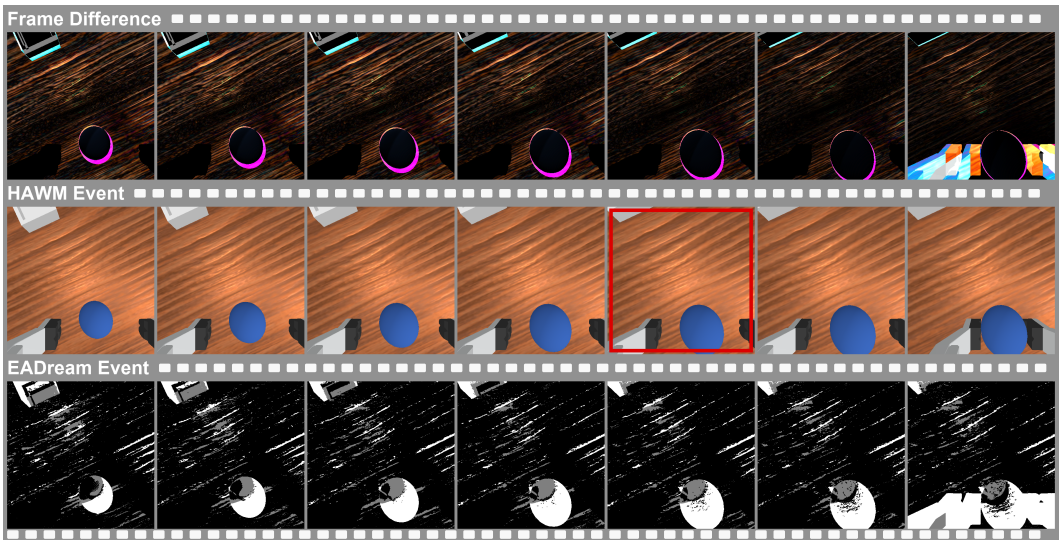

Figure 9: **Event detection from the wrist camera, sequence 2.** HAWM identifies the pre-interaction transition as the gripper approaches the sphere. Camera motion shifts both the object and textured background across pixels, eliciting visual-change responses that are less specific to the interaction.

Figures 10–13 illustrate the distinction between visual change and task-relevant transitions. In `PegInsertionSide`, reward remains flat across the displayed frames before increasing at the next step; the highlighted event therefore carries information about an imminent outcome that immediate reward alone does not reveal. In `CloseFridge`, the event similarly precedes a sharp reward increase despite substantial egocentric visual change throughout the sequence. The head and wrist views show why pixel-based event targets can vary with camera motion and viewpoint. Together, these examples are consistent with HAWM's future-reward objective encouraging events that anticipate changes in task progress.

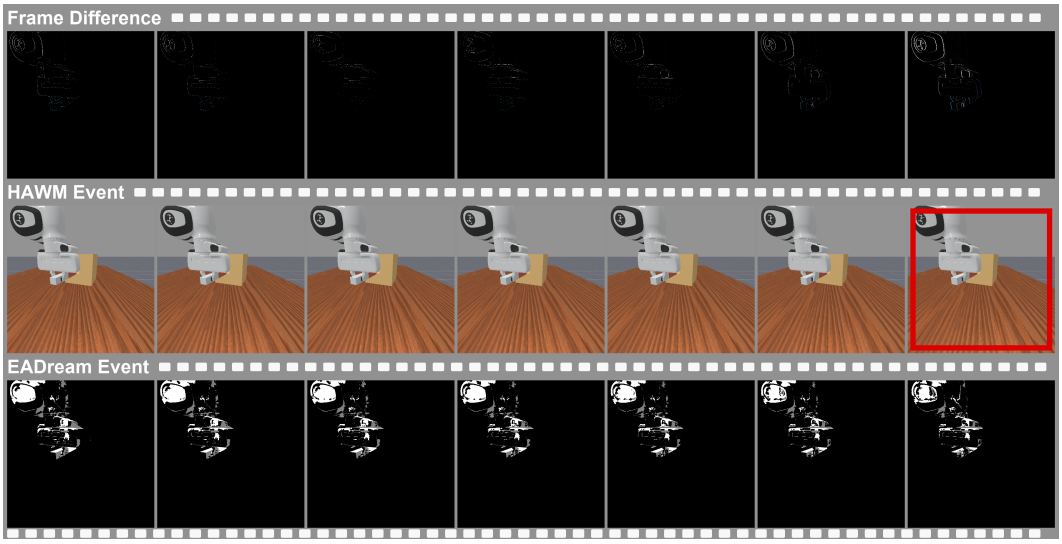

Figure 10: **PegInsertionSide, head camera.** Frames 118–124 precede a reward increase at step 125. The red border highlights the HAWM event at step 124. The rows show frame differences, observations, and EADream visual events, respectively.

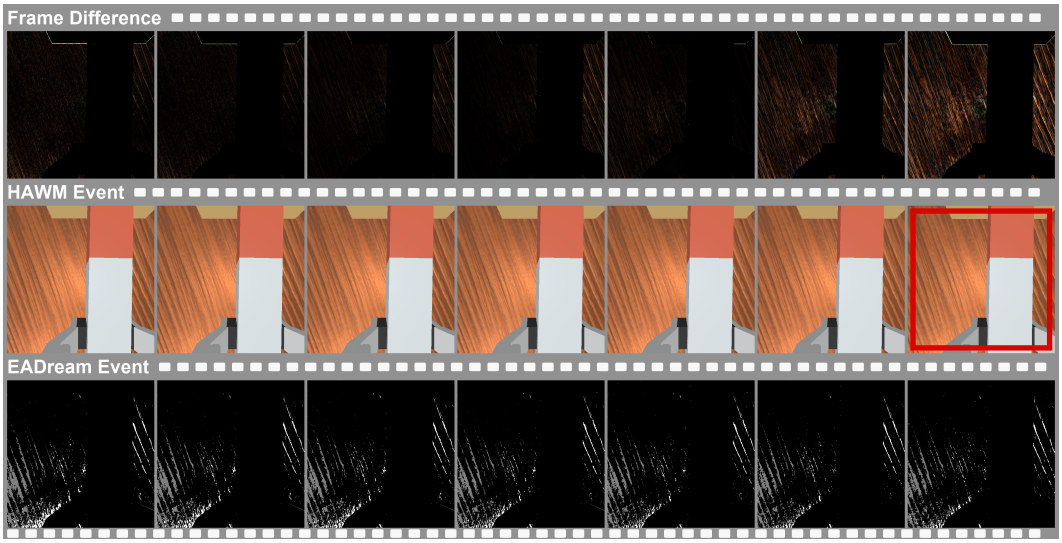

Figure 11: **PegInsertionSide, wrist camera.** The same seven steps from the wrist viewpoint. HAWM's highlighted event precedes the reward increase, while the frame differences and EADream events reflect changes in the egocentric image.

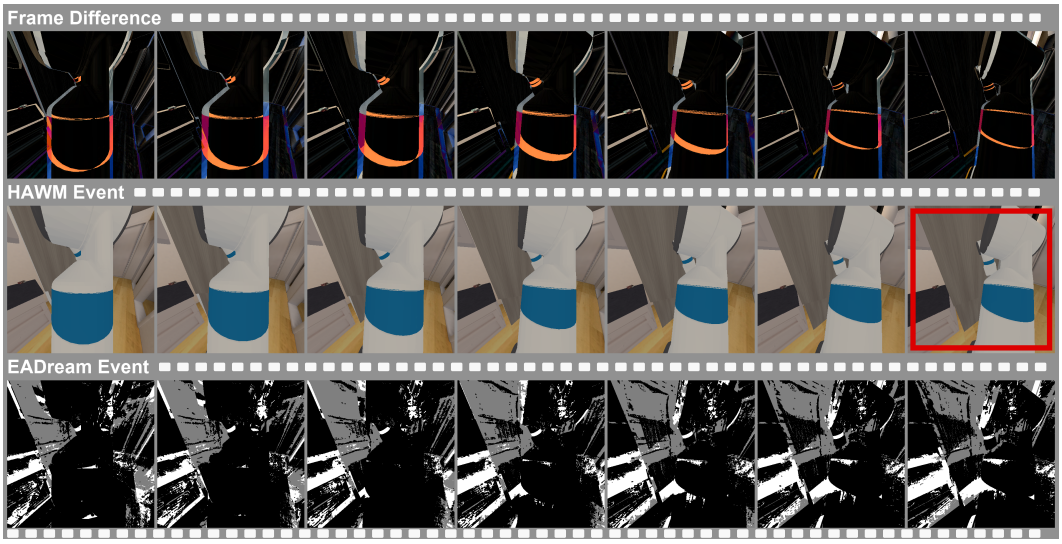

Figure 12: **CloseFridge, head camera.** Frames 35–41 precede the reward increase at step 42. The red border highlights the HAWM event at step 41, immediately before the increase.

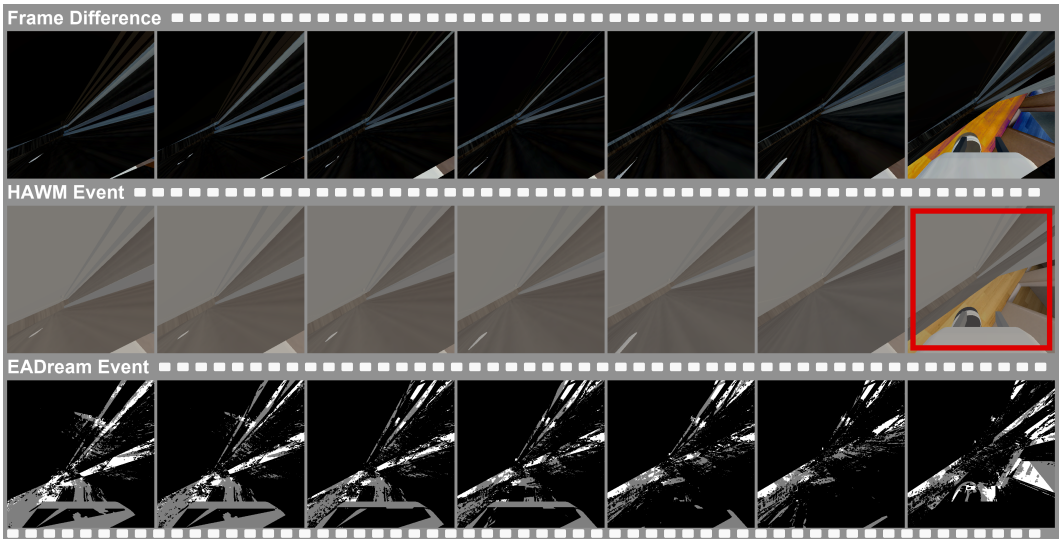

Figure 13: **CloseFridge, wrist camera.** The same transition from the wrist viewpoint. Camera motion produces extensive visual changes, whereas the highlighted HAWM event aligns with the subsequent reward increase.

C.2 On-Learning Frame Difference

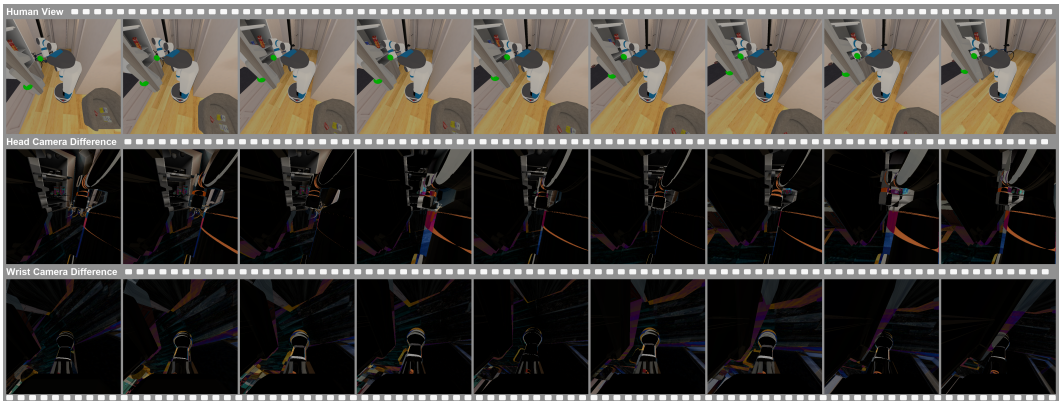

Figure 14: **Frame difference are mostly unrelated to key frame.** On a `CloseFridge` trajectory collected by random checkpoint, we show that model cannot learn useful signal from frame difference introduced by random action.

Figure 14 illustrates a difficulty with defining events from frame differences during exploration. A random policy moves the robot and, with it, the head and wrist cameras. Changes in camera pose, parallax, and the visible background can therefore produce substantial differences between consecutive frames even when little has changed in the task state of the fridge. An event detector driven primarily by the magnitude of visual change may record these camera-induced changes frequently, causing its event sequence to reflect exploratory motion rather than transitions that matter for control. Conversely, an interaction such as establishing contact with the handle may be spatially localized, yet alter the actions available to the robot and the likelihood of a later reward. Pixel change is thus an imperfect proxy for the consequences of a transition.

HAWM connects event learning to those consequences through the world-model prediction objectives and an auxiliary objective for predicting future reward. The latter encourages event features to retain information about transitions that precede task progress, without requiring the reward to change at the event timestep. Future rewards are used as training targets, not supplied to the event posterior as inputs. This distinction matters during both interaction and imagination, when event occurrences must be inferred or predicted from information available at the current step. The figure provides a qualitative motivation for this reward-guided criterion; it does not, on its own, measure event-detection accuracy.

C.3 Scene and Appearance Generalization

We study zero-shot generalization to unseen scene configurations and illumination shifts. Results are reported in Table 2.

HAWM transfers to unseen scene configurations. HAWM retains its advantage when evaluated in layouts that were not seen during training. A new layout changes the robot's route, viewpoints, and surrounding objects, but it does not change the interactions required to complete the task: the robot must still reach, grasp, and maintain control of the target. This creates a useful distinction between visual details specific to a scene and transitions whose consequences are shared across scenes. By training event features to help predict future reward, HAWM encourages the model to represent the latter. Its event memory then carries information about recent interactions forward as the robot navigates and the target moves in and out of view. The results are consistent with this interaction history providing useful context for prediction and control in unfamiliar layouts.

HAWM remains effective under illumination shifts. HAWM also maintains the strongest performance when the scene is rendered under substantially different lighting. Illumination changes the appearance of objects, surfaces, and shadows without changing the underlying task dynamics. A representation that relies heavily on particular visual appearances may therefore become less reliable, even though the actions needed for success are similar. HAWM's event representation is trained through predictive objectives, including future reward prediction, which encourages attention to the consequences of interactions rather than visual change alone. Its decaying event memory provides temporal context beyond the current observation and may be especially useful when a dim image gives ambiguous evidence about the robot's progress. This does not make event inference independent of vision, but it gives the world model another source of information to use when appearance changes.

Together, these results suggest that the event history learned by HAWM captures aspects of task progress that remain useful across changes in layout and rendering. The generalization experiments support this interpretation, while the contribution of event learning and memory should be assessed through the corresponding component ablations.

Table 2: **Scene and appearance generalization.** Zero-shot success rates (%, mean $\pm$ standard deviation).

Setting	Evaluation	DreamerV3	TD-MPC2	EADream	Ours
Scene Generalization	Training scene	45.7 ± 8.2	27.7 ± 5.8	31.3 ± 6.1	**57.6** ± 5.9
	Unseen scenes	34.8 ± 11.8	21.8 ± 9.2	20.8 ± 9.2	**49.8** ± 7.0
Illumination Generalization	Dark ($0.1\times$)	23.4 ± 7.3	13.0 ± 4.1	17.1 ± 7.2	**40.4** ± 5.9
	Bright ($4.0\times$)	35.1 ± 7.8	20.4 ± 6.1	26.2 ± 8.8	**47.1** ± 7.5
	Average	29.3 ± 7.6	16.7 ± 5.1	21.7 ± 8.0	**43.8** ± 6.7

C.4 LEARNING-RATE SENSITIVITY

Table 3 varies the learning rate by factors of two and four around the default while holding the model and interaction budget fixed. This probes whether the latent dynamics and event posterior–prior can be learned reliably at different update scales. A smaller rate may delay learning of sparse, reward-relevant events within the fixed budget; a larger rate may disrupt the coupling between event inference, memory, and imagined dynamics. The completed results will determine whether the default setting is a robust operating point or a narrow optimum.

Learning rate	Success rate (%)
1×10^{-5}	82.2 ± 6.3
2×10^{-5}	89.2 ± 5.9
$\mathbf{4 \times 10^{-5}}$ (default)	$\mathbf{93.4 \pm 5.8}$
8×10^{-5}	90.4 ± 6.4
1.6×10^{-4}	86.4 ± 7.8

Table 3: Learning-rate sensitivity of HAWM, averaged over four ManiSkill3 tasks after 4M environment steps per task. All settings use $K = 3$ event types and the same training protocol.

D Detailed Method

This appendix specifies feature construction, event sampling, gradient routing, and rollout bookkeeping for the model in Section 3. The memory, predictors, and learning objectives are defined in the main text; the details below explain how those components are evaluated.

D.1 Observation and Latent Change Features

Let $v_t = E_\phi(o_t) \in \mathbb{R}^{d_v}$ be the observation encoding. For a vector $w \in \mathbb{R}^d$, define

$$\begin{aligned} \mathrm{symlog}(w) &= \mathrm{sign}(w)\log(1+|w|), \\ \mathrm{mag}(w) &= \log\left(1+\sqrt{\|w\|_2^2/d+\varepsilon}\right), \\ T(w) &= [\mathrm{symlog}(w); \mathrm{mag}(w)], \end{aligned} \tag{20}$$

where the first operation is elementwise and $\varepsilon > 0$ provides numerical stabilization. The observation-change feature in Eq. (11) is

$$\Delta v_t = v_t - v_{t-1}, \qquad \Delta o_t = T(\Delta v_t). \tag{21}$$

Transition context enters through h_t, which already depends on the preceding action. Following the main detector equation, Δo_t remains differentiable for occurrence inference. Event losses can therefore update the observation encoder through this input. The type posterior instead receives detached h_t, Δo_t, and q_t^{evt}, as specified in Eq. (12). These detachments block the type branch's gradients into the detector and its input features.

The prior uses changes in latent distributions. For RSSM categorical logits ℓ_t, define

$$\begin{aligned} \zeta_t &= \log\left[(1-\upsilon)\,\mathrm{softmax}(\ell_t) + \frac{\upsilon}{C}\right], \\ \Delta z_t &= \mathrm{sg}\left(\mathrm{vec}(\zeta_t - \zeta_{t-1})\right). \end{aligned} \tag{22}$$

Here, υ is the uniform mixing coefficient and C is the number of classes per categorical factor. Softmax and mixing are applied separately to each factor, and vec flattens the resulting differences. The occurrence-context and type-prior networks apply $T(\Delta z_t)$ internally, with separate learned parameters. Observed trajectories supply posterior logits; imagined transitions supply dynamics-prior logits. This feature describes distributional change without depending on a particular latent sample.

D.2 Sampling and Gradient Routing

During observed training, the occurrence sample uses the straight-through estimator

$$y_t^+ = \mathrm{sg}(\tilde{y}_t^+ - q_t^{\mathrm{evt}}) + q_t^{\mathrm{evt}}, \quad \tilde{y}_t^+ \sim \mathrm{Bernoulli}(q_t^{\mathrm{evt}}), \tag{23}$$

The forward value is binary, while its surrogate derivative with respect to q_t^{evt} is one. This sample updates the shared memory in Eq. (7). The prediction heads also receive q_t^{evt} and $\mathbf{c}_t$ directly as soft features, including when the sampled occurrence is zero.

For the embedding in Eq. (9), evaluate λ_t^+ as the softplus of the detector logit. This implements Eq. (10) without subtracting a near-unit probability from one. The prior occurrence probability can be evaluated as $-\mathrm{expm1}(-\lambda_t^-)$. Both pathways use the same α, β, and Ω.

World-model training retains the differentiable posterior memory path, allowing prediction losses at later steps to reach earlier occurrence decisions. Occurrence-prior fitting uses a second computation of the same history with detached event samples:

$$\bar{M}_t^{\mathrm{fit}} = \exp(-\beta) M_{t-1}^{\mathrm{fit}}, \quad M_t^{\mathrm{fit}} = \bar{M}_t^{\mathrm{fit}} + \alpha\,\mathrm{sg}(y_t^+). \tag{24}$$

At episode starts, $M_0^{\mathrm{fit}} = 0$. A carried fitting-memory value is treated as detached boundary context at the start of a replay chunk. The prior intensity for step t uses $\bar{M}_t^{\mathrm{fit}}$, before adding that step's event. With the same initialization and samples, this trace equals the posterior trace in the forward pass, but blocks gradients into the detector through the event history. Recomputing the trace retains gradients into α and β; detaching the entire accumulated memory would also remove those history gradients.

Together with detached latent inputs and posterior targets, this construction makes $\mathcal{L}_{\mathrm{haw}}$ update b, g_η, α, and β without updating event inference or the RSSM through its inputs. The type-matching loss updates P_ϑ. As in Eq. (18), type alignment is averaged over all valid transitions because type probabilities are supplied to the prediction heads at every such transition.

The observation, reward, continuation, and auxiliary future-reward objectives train the posterior event features through their differentiable paths. In the auxiliary head, detaching z_t blocks its direct latent-input gradient; it does not block the paths through the occurrence features, observation encoder, and memory embedding. The event-rate regularizer controls the average occurrence probability.

D.3 Rollout Order and Valid Transitions

On an observed transition, first advance the recurrent state using the previous action and memory embedding, and infer the current latent posterior. Next compute Δo_t, infer q_t^{evt} and $\mathbf{c}_t$, sample y_t^+, and construct M_t^+ and $e_t^{\mathrm{haw},+}$. These features condition the observation, reward, continuation, and horizon predictors in Eq. (6). Prior fitting uses the corresponding latent-distribution change and detached posterior targets.

An imagined rollout starts from an observed state after its memory update, carrying s_t, M_t^+, $e_t^{\mathrm{haw},+}$, and ζ_t. The policy selects the next action from the latent state and memory embedding. Each subsequent step advances the RSSM, computes the latent-distribution change and event-prior outputs, and draws one Bernoulli occurrence sample. That same sample determines both the memory used in the current embedding and the memory carried to the next step. The first imagined change compares the new prior log-probabilities with the carried observed posterior log-probabilities; later changes compare consecutive prior log-probabilities.

Imagined prediction uses $e_t^{\mathrm{haw},-}$, p_t^{evt}, and $\hat{\mathbf{c}}_t$ in the same heads used during observed training. The actor and critic receive the latent state and active memory embedding in both modes. Decoded rewards and continuation probabilities determine imagined returns. The auxiliary horizon predictor is used only for world-model training.

Memory is reset to zero at episode boundaries, and feature differences must never cross those boundaries. The set $\mathcal{V}$ excludes padding and transitions lacking a valid predecessor. A replay chunk beginning inside an episode needs preceding context or a carried state to reconstruct its memory and change features; a chunk boundary alone does not define an episode reset. Such context is processed before the transitions whose losses are scored. The future-reward loss uses the subset $\mathcal{U}$ for which all H target rewards are available within the same episode, as specified in Eq. (15). If a masked loss has no valid entries, its contribution is omitted.

E Limitations

HAWM uses future-reward supervision to guide event discovery. With sparse or delayed rewards, consequential interactions may fall outside the prediction horizon and receive weak task-specific supervision. Extending temporal credit assignment or incorporating learned progress signals could help identify these interactions. Furthermore, HAWM must represent their differences within a fixed event vocabulary of size K. Although our ablations favor a compact vocabulary, the required capacity may vary across tasks. Adapting K according to predictive utility is a promising direction, provided that type representations remain consistent between observed and imagined trajectories.