ADAPTIVE LONG-HORIZON LATENT ACTION MODEL

Anonymous authors
Paper under double-blind review

ABSTRACT

Latent action models aim to learn controllable actions from unlabeled trajectories. However, the action-induced changes between consecutive frames are hard for models to distinguish from noise. Without action labels, reconstructing observed transitions alone does not explicitly require the model to separate action effects from noise. To overcome this problem, Adaptive Long-Horizon LAM is introduced. In the real world, few controllable actions can be completed to reach a target in one frame. Also, they will exhibit strong temporal persistence and dictate a consistent state transition throughout their lifespan, while exogenous noise is stochastic and lacks dynamical consistency over time. The key idea of ALH-LAM is to force the model to learn the consistent parts of changes. This constraint can be viewed as an unsupervised consistency bottleneck. In the meantime, fixed long-horizon cannot fit into all situations. These temporal supports also guide action chunk discovery. The experiments on world models show that ALHLAM is better than fixed long-horizon.

1 INTRODUCTION

Learning behaviors from observations makes action-unlabeled demonstrations possible to become learning resources. Self-supervised pretraining has enabled effective visual representation learning from images and video He et al. (2022); Assran et al. (2025). Latent Action Models (LAMs) further try to learn action representations from observation changes and use them as subsequent policy learning Schmidt & Jiang (2023); Ye et al. (2024). The value of these representations is determined by how much actions related information can retain.

However, predicting one transition does not mean the model can distinguish changes caused by actions from other changes. Recent theories and experimental studies show that latent representations may encode exogenous noise or other distractors which help reconstruct observations Zhang et al. (2025a); Nikulin et al. (2025). Hence, which observations should be used to infer latent actions needs to be examined. In particular, an action which continues over several time slots can be observed through multiple transitions, which can provide more evidence than an isolated transition.

Using longer observation intervals brings another trade-off. If only requires the latent action to encode a single transition, the encoder may not really use the other observations in the interval. While if the latent action is asked to also infer other transitions in the window, another problem will be introduced. If the action effect is constant and noise is independent and zero-mean, repeated observations can help better estimate the common effect. However, the interval across over multiple actions requires a shared representation to explain different effects. The useful interval therefore depends on both noise and action persistence. This paper asks how such intervals can be determined without real labels.

Existing methods use temporal information through historical context or structure relations between latent transitions Liang et al. (2025); Tang et al. (2026b). This paper investigates another design choice: using one shared latent action to explain every transition over a temporal interval and determining its effective temporal support. For each candidate window, this method learns a shared latent action, with treating every transition in the window as reconstruction target. Then, balance the reconstruction consistency and length supporting, allowing the various window choices for different centers. Training and support selection of ALHLAM use neither real actions nor boundaries.

Among experiments, the combined shared-window objective and adaptive support selection achieve higher action-probe accuracy than a fixed-context baseline which only predicts the center transi-

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.

tion. Higher proportion of selected window remain with single action. These experiments provide evidence for the design.

The selected temporal supports can also help with predicting actions changing. These cues are used for offline action chunk discovery while keeping the trained model frozen. As an exploratory extension, concurrent action factors with separate temporal supports are also examined, but the factor collapse reveals the limitation of learning effect basis jointly.

Main contributions are as follows:

1. Propose latent action learning through shared window reconstruction with per-center support selection, balancing multiple transitions clues and risk of mixing different actions.
2. Evaluate this design under controlled noise environment and further use the results for action chunk discovery without additional model training.

2 Related Work

2.1 Learning Latent Action From Action-Free observation

Action representations have been learned from transitions with observed actions (Chandak et al., 2019) or through multi-task policy training (Hua et al., 2022). Other work infers latent actions from trajectories without action labels. Early observation-only imitation learning has explored inferring non-observable action variables from change of status and mapping by using a small environment interaction data set Edwards et al. (2018). On this basis, LAPO Schmidt & Jiang (2023) used the Inverse Dynamic Model and the Forward Dynamic Model to create a prediction bottleneck, which allows the model to learn latent actions from action-free observation transitions and pre-train policies by utilizing the latent actions. After that, latent action is used for interactive world models and training world models using unlabeled trajectories Bruce et al. (2024); Gao et al. (2025); Tharwat et al. (2025); Alles et al. (2025); Garrido et al. (2026). While another kind of work is pre-training pseudo-labeling for robot policy Cui et al. (2024); Ye et al. (2024); Liang et al. (2025); Bu et al. (2025). A recent systematic compare different ways of using latent action supervision to train vision-language-action policies (Lin et al., 2026). Instead of downstream applications, this article focuses on determining the time supporting scope when estimating a single latent action.

2.2 Temporal Structure in latent action inference

Current latent action model usually pre-specifying time units corresponding to the latent actions. Basically, observe o_t and o_{t+1} to infer a latent action and use it to predict state change in a single time slot Schmidt & Jiang (2023). CLAM Liang et al. (2025) additionally provides its IDM and FDM with a fixed-length observation history. However, z_t is still inferred for the adjacent transition from o_t to o_{t+1}, and the reconstruction loss only supervises o_{t+1}, which means the history only serves as context. LAPA Ye et al. (2024) extends the structure to x_t and x_{t+H}, but the H is hand-crafted before training, and IDM only uses two endpoints. UniVLA Bu et al. (2025) also get latent action from two observations of time difference of about one second. LAOM Nikulin et al. (2025) sampled k uniformly and getting o_t and o_{t+k}. However, this still only uses two endpoints, and the model did not infer an efficient supporting length. Farsighted-LAM Cai et al. (2026) processing current frame and multiple future frames simultaneously, but each produces a single z, and time scale is still hand-crafted.

Hence, these methods extend input history, endpoints distance, and number of future frames, but none used what this article is studying, which is forcing a shared latent action to explain all of the transitions, and deciding effective supporting scope for each frame by comparing internal consistency.

2.3 Noise, exogenous factor, and multi-step inverse dynamics

Analysis Zhang et al. (2025a); Lee et al. (2026a) have already shown standard reconstruction objectives do not naturally distinguish controllable change and exogenous noise, so latent action may encode information that can help predict future observation but irrelevant to real action. LAOM

Nikulin et al. (2025) further illustrates this issue within environments of dynamic backgrounds, camera shake, or color variations. AD3 (Wang et al., 2024) uses known agent actions and inferred distractor actions to separate task-related dynamics and distractor dynamics. MaskLAM (Fechner et al., 2026) uses spatial agent masks to reduce the influence of noise, while MVP-LAM (Lee et al., 2026b) uses learning action-centric representations through cross-view reconstruction.

Related works in state representation learning Efroni et al. (2022); Lamb et al. (2023); Levine et al. (2024) used multi-step inverse dynamics to filter out exogenous distractors or to learn control-endogenous state that retain necessary information for control. However, these methods usually used real actions as supervision of inverse dynamics or only studied representation learning between distant endpoints. They did not study how to choose shared temporal support for unsupervised latent actions. This paper does not claim solving all correlated distractors in all real videos but isolates zero-mean transition noise in a controlled environment to study if sharing latent representation across multiple transitions with identical actions improve the separability of action and noise.

2.4 Temporal abstraction and action segment discovery

Prior works have explored the discovery of reusable, variable-length behavior segments from unsegmented trajectories, or the learning of hierarchical temporal representations with latent boundaries Kipf et al. (2019); Kim et al. (2019). HiLAM first used inverse dynamics of fixed time intervals to extract stepwise low-level latent actions, then used dynamic chunking mechanism of H-Net to compress into variable-length high-level latent skills Kim et al. (2026); Hwang et al. (2025). Another work Zhang et al. (2025b) utilized a motion tokenizer to obtain latent action sequences and detected boundaries of action primitives via Latent Action Energy. Meanwhile, SUNTA Iiyama et al. (2026) interpreted prediction errors as surprise signals within a hierarchical state space model, using them to determine the temporal chunking boundaries of video sequences. Du et al. (2022) detect action boundaries from temporal changes in video features without training a segmentation model.

The segmentation extension in this paper also utilizes unsupervised dynamics signals to discover temporal boundaries, but the evidence is directly from estimated effective temporal support in the proposed latent-action model.

2.5 Structured and Factorized latent actions

Besides temporal structure, other works restricted latent actions through space or object structures. Object-Centric Latent Action Learning and PlaySlot used object or slot representations, separating action-related variations from background changing Klepach et al. (2025); Villar-Corrales & Behnke (2025). FLAM inferred latent actions for different factors in the scenario, while OTF-LAM first decomposed observations transitions into sparse, reusable transition effects, and subsequently abstracted these effects into action-like latent variables Wang et al. (2026); Nam et al. (2026). AC-LAM and ALAM Wei et al. (2026); Tang et al. (2026a) regularize the algebraic structure of latent transitions through addition composition, inverse transformation, and cycle consistency. These works mainly study space, objects or algebraic compositions of latent actions, while this paper focuses on the selecting of temporal extent of the observation window.

3 Method

3.1 Shared Latent Action Learning under Adaptive temporal support

Learning latent actions from unlabeled observation trajectories is considered. With trajectories containing T transitions, observations sequences are marked as $o_0, \ldots, o_T$, where $o_t \in \mathbb{R}^d$, and d is the observation's dimension. Real actions, actions boundaries, actions effects without noise, and noise information do not join training of latent action model or temporal support selection, only for subsequent evaluations.

To study the relationship between temporal support and transition noise, this study adopts a controllable additive dynamics setting:

$$\Delta o_t \equiv o_{t+1} - o_t = q_t + \varepsilon_t, \tag{1}$$

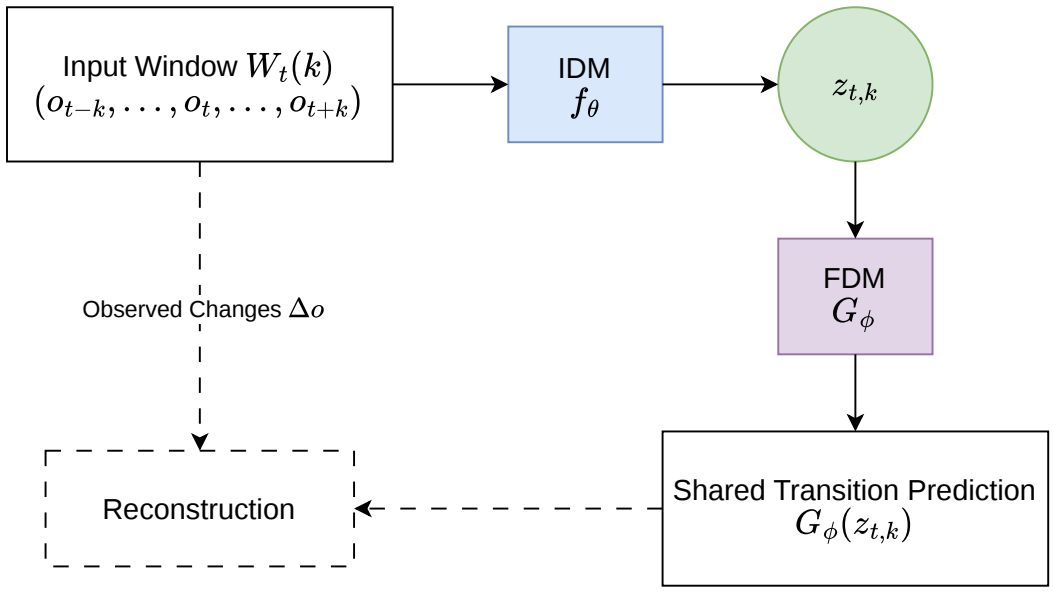


Figure 1: Shared latent action learning with inverse and forward dynamics models.

among with q_t represents noise-free transition effect produced by actions, ε_t represents noise. Identical actions will produce fixed transition effect under this setting and will remain unchanged during a consecutive time slot. Segments' length and boundaries are unknown to the model. Exogenous noise is independent from each other between transitions, independent from actions' procedures, and with zero mean and finite second moments.

For central moment t and candidate radius k, define observation window:

$$W_t(k) = (o_{t-k}, \ldots, o_t, \ldots, o_{t+k}), \tag{2}$$

among with $1 \leq k \leq K_{\max}$, where $K_{\max}$ is the default maximum candidate radius, and every effective window must be in one same trajectory. Each window contains $2k+1$ observations and $2k$ transitions in between. Inverse Dynamic Model (IDM) infers one shared latent action from the entire window:

$$z_{t,k} = f_\theta(W_t(k)) \in \mathbb{R}^{d_z}, \tag{3}$$

where, f_θ is the IDM model which θ is the parameter, and d_z is the dimension of latent action. IDM read raw observations and window validity mask, a binary mask indicating valid positions in the window, not taking default observation difference Δo as input.

The Forward Dynamic Model (FDM) G_ϕ takes $z_{t,k}$ as input, outputting a predicted transition effect $G_\phi(z_{t,k}) \in \mathbb{R}^d$. Here, ϕ is the parameter of this model. The identical predicted effect should be used to explain all of the transitions in the window. The average reconstruction error is defined as:

$$\mathcal{R}_{t,k} = \frac{1}{2k} \sum_{u=t-k}^{t+k-1} \|\Delta o_u - G_\phi(z_{t,k})\|_2^2, \tag{4}$$

where u is the starting moment of transitions in window. Here, observation difference is treated as the reconstruction target of FDM. All of $2k$ transitions in the window jointly restrict the common $z_{t,k}$. Figure 1 illustrates the adaptive long-horizon latent action model.

During training, sample t from training centers which possess complete maximum window $W_t(K_{\max})$, and uniformly sample radius k from $1, \ldots, K_{\max}$. This makes the same group of models learn from different lengths of temporal support. Through minimizing total loss $\mathcal{L}(\theta, \phi)$, train IDM and FDM:

$$\mathcal{L}(\theta, \phi) = \mathbb{E}_{t,k}\left[\frac{\mathcal{R}_{t,k}}{d} + \beta \frac{\|z_{t,k}\|_2^2}{d_z}\right], \tag{5}$$

where, $\mathbb{E}_{t,k}$ represents the expectation over window centers t and radius k sampled as described above, β is the weight of latent representation regulation. Reconstruction loss first gets the average from transitions in each window, then gets the average from windows in each batch. Therefore, all sampled windows have the same weight. Divided by d makes reconstruction loss normalized as mean squared error per observation dimension.

After completing model training, parameters θ and ϕ are fixed, and effective candidate windows are compared for every center. Let $\mathcal{K}_t$ represent the available candidate radiuses set of center t, among with each radius satisfies $1 \leq k \leq K_{\max}$, and corresponding window is completely located in an identical trajectory. Selected radius is marked as k_t^*, and corresponding latent action is marked as z_t, defined as:

$$\arg\min_{k \in \mathcal{K}t} \left[\mathcal{R}_{t,k} - \lambda \log k\right], \qquad z_t = z_{t,k_t^*}, \tag{6}$$

where, $\lambda \geq 0$ controls the balance between reconstruction consistency and support length. In selection stage, reconstruction loss uses squared L_2 error of each transition, which is $\mathcal{R}_{t,k}$ as described above.

This design uses action consistency in local temporal scope: when actions are the same in the window, multiple transitions provide repeated action information, helping to decrease the effect of single noise to latent representation; when the window crosses multiple actions, one shared transition effect is usually much harder to explain changes in the window. Reconstruction loss is used as measuring consistency inside candidate windows, and reward encourages longer support. Hence, this study chooses effective temporal support for each center and get shared latent actions from chosen window. This support is not predefined as complete scope of real actions segment.

3.2 Action Chunk Discovery based on Adaptive Temporal Support

Section 3.1 yields a chosen time radius k_t^* and a corresponding latent action z_t. This study moves on using these temporal support and model reconstruction loss to discover offline action chunk and choosing a representative latent action for every prediction chunk. Since reconstruction target did not explicitly restrict relation between distance of latent representation and action effects, this method uses temporal support and prediction consistency as chunking cues. The whole procedures reuse trained IDM and FDM model, fixing the parameters, not additionally train chunking model.

First, find local minimums in chosen radius sequence. Windows under action changing may mix up different transition effects, which limit scope that a shared latent action can explain. Hence, shorter local support can provide candidates boundaries cues. Let r be the valley detecting neighborhood radius, in the usable positions of both sides' neighborhoods, define valley depth as

$$v_t = \min\left\{\text{median}_{1 \leq j \leq r}\, k_{t-j}^*,\, \text{median}_{1 \leq j \leq r}\, k_{t+j}^*\right\} - k_t^*, \tag{7}$$

where, j represents time offset relative to center. Combining consecutive positions which satisfies $v_t \geq \delta_k$ to a candidate valley V, where $\delta_k > 0$ is the minimum valley depth. Every area produces a candidate boundary; position is the minima k_t^* center in the area. In case of ties, choose the position closest to the area's midpoint, then if still draw, choose earlier position.

Then, use reconstruction loss under different support scopes confirming candidate boundaries. Continuously using average reconstruction error $\mathcal{R}_{t,k}$ excluding extent reward in section 3.1, define

$$D_t = \mathcal{R}_{t,2} - \mathcal{R}_{t,1} \tag{8}$$

This difference measures change of average reconstruction error when candidate radius extends from 1 to 2. For every candidate area V, finding maximum difference within a confirmation neighborhood by extending the candidate valley area on both sides, marked as $D_V^{\max}$. Taking difference values near the peak value, out of confirmation neighborhood as background, calculating background median μ_V and scale s_v. Default scaling takes 1.4826 times the median absolute deviation from the median. If background values are not enough, instead use the available values outside the confirmation neighborhood in the same trajectory. If the scale is too small, the standard deviation is used and, if necessary, a positive numerical floor. Only when

$$D_V^{\max} > 0, \qquad \frac{D_V^{\max} - \mu_V}{s_V} \geq \gamma \tag{9}$$

accepting the candidate, where $\gamma > 0$ is confirming threshold. This step only determines whether the candidate is retained. The boundary location remains the candidate location identified from the valley in the selected-radius sequence.

Combine the accepted boundaries and trajectory endpoints, getting

$$0 = \tau_0 < \tau_1 < \cdots < \tau_M = T, \tag{10}$$

where, M is the number of predicted chunks and τ_m represents a chunk boundary. The m-th chunk contains transition start moments.

$$\mathcal{I}_m = \{\tau_{m-1}, \ldots, \tau_m - 1\}, \qquad m = 1, \ldots, M. \tag{11}$$

These chunks cover all transitions in the trajectory. No additional minimum chunk length is given.

Finally, select a representative latent action z_m for each chunk. Within the centers whose adaptive outputs are available, retain those satisfying

$$[t - k_t^*,\ t + k_t^* - 1] \subseteq [\tau_{m-1},\ \tau_m - 1] \tag{12}$$

which makes sure that the full transition support is inside the predicted chunk. Then choose the remaining center with the largest k_t^*, marked as t_m, and set $z_m = z_{t_m}$. In case of ties, choose the candidate center closest to the midpoint of the chunk's transition start times, $(\tau_{m-1} + \tau_m - 1)/2$, then if still draw, choose the earlier center.

If no candidate satisfies the conditions, select the available center t_m which is closest to the midpoint in the chunk and use the latent action produced by the frozen model at $k = 1$, and set $z_m = z_{t_m,1}$. If the chunk contains no available center, use the center which is closest to the chunk midpoint in the same trajectory, still prefer earlier one in case of ties. The fallback support is not guaranteed to entirely be inside the predicted chunk. Chunks using this fallback are reported separately in the experiments.

Therefore, this obtains a representative latent action for each predicted chunk. The temporal support used to estimate this latent action may be shorter than the full chunk. Evaluate how well z_m matches action information of entire chunk and how well $G_\phi(z_m)$ can represent the transition effects. Neither boundary detection nor representative selection uses real actions or boundaries.

4 Experiments

4.1 Latent Action Learning under Transition Noise

In the additive environment described in section 3.1, evaluate action information in the learned latent actions and boundary crossing situation of selected temporal supports. Observations have 64 dimensions. There are four action classes, each associated with a fixed 64-dimensional transition effect of Euclidean norm 2. Each action chunk contains 4-24 transitions, and consecutive chunks use different actions. Each noise condition contains 300 trajectories, and each trajectory starts with a zero observation and contains 1000 transitions.

Every transition is joined by a zero-mean Gaussian noise, which is independent of the action and independent of each other. Let ρ represents the root-mean-square of noise vector, and I_{64} means the 64-dimensional identity matrix. Both conditions settings use the same data seed, keeping the action effects, action chunks, and underlying Gaussian samples, while only changing the noise amplitude.

$$\varepsilon_t \sim \mathcal{N}\left(0, \frac{\rho^2}{64} I_{64}\right), \qquad \rho = \sqrt{\mathbb{E}\left[\|\varepsilon_t\|_2^2\right]} \in \{2, 4\}. \tag{13}$$

The IDM is a single-layer bidirectional GRU, where hidden dimension is 256 per direction. Its outputs are averaged over valid window positions to get a four-dimensional latent action. The FDM is an MLP with two 512 dimensions hidden layers. It only reads latent actions to predict the transition effects. Training uses Adam with learning rate 10^{-3}, batch size of 128 for 20 epochs. Set $K_{\max} = 8$, $\beta = 10^{-4}$, and $\lambda = 0.1$, using the same settings for both noise levels.

Table 1: Action information decoded by linear probes on validation trajectories. Results are means ± sample standard deviations across three training seeds.

Noise RMS	Method	Accuracy (%) ↑	Center-effect R^2 ↑
2	Adaptive	84.13 ± 0.72	0.486 ± 0.007
2	Fixed-4-OneStep	74.71 ± 0.21	0.410 ± 0.010
4	Adaptive	79.49 ± 0.85	0.432 ± 0.005
4	Fixed-4-OneStep	72.75 ± 1.04	0.399 ± 0.005

Table 2: Temporal-support selection on validation trajectories. Clean windows contain transitions from one true action class. Results are means ± sample standard deviations across three training seeds.

Noise RMS	Method	Mean radius	Clean windows (%) ↑	Avoidable crossings (%) ↓
2	Adaptive	4.419 ± 0.045	72.93 ± 0.74	21.53 ± 0.80
2	Fixed-4-OneStep	4 (fixed)	52.13 ± 0.00	43.87 ± 0.00
4	Adaptive	4.020 ± 0.026	68.79 ± 0.32	25.99 ± 0.35
4	Fixed-4-OneStep	4 (fixed)	52.13 ± 0.00	43.87 ± 0.00

Baseline, Fixed-4-OneStep, uses a fixed long context and single-step supervision. It also uses the identical network structure and latent dimension with ALHLAM, inferring a latent action from the 9 observations of fixed window $W_t(4)$, but the reconstruction loss only supervises the central transition $o_t \to o_{t+1}$. ALHLAM samples radius during training and lets a shared latent action to explain all transitions in the window, then choose temporal support for each center. Two methods both use the identical training centers and optimization settings.

Trajectories 0-239 are used for model training, while trajectories 240-299 are used for testing and evaluation, which are not for model parameter optimization. Both methods sample from training centers which can be available with maximum radius of 8. For each method, separately train with seeds 0, 1, 2. Tables report the mean and sample standard deviation across these three runs.

Model training and temporal support selection do not use true actions, noise-free action effects, or noise labels. During evaluation, freeze the model and fit linear ridge probes on latent actions from the training trajectories. Latent features are standardized using training-set statistics with an unregularized intercept. The action probe fits scores to the four one-hot actions and gets the class with the highest score. The action-effect probe predicts noise-free action effect q_t of the center transition, with report on variance-weighted R^2 across target dimensions. See Table 1.

ALHLAM has higher action accuracy and center-effect R^2 under two noise levels. Strong noise decreases both method's accuracy, but ALHLAM retains an advantage.

Also, whether temporal supports across real action boundaries is evaluated. If all transitions of a window are the same real action, it is called a clean window. The avoidable-crossing rate is the fraction of selected windows which cross boundaries but with at least one clean candidate window. Centers which even radius 1 still cross boundary are not included in this metric. See Table 2.

Since boundary crossing by the baseline is determined by the fixed radius, values do not vary due to different training seeds or noise levels. These results show that adaptive temporal support reduces the rate of boundary crossings.

Finally, test whether the FDM uses the latent action. After fixing model parameters and evaluation targets, separately using the original latent action, shuffled latent actions, mean training latent action, or zero vector. See Table 3.

At both noise levels, both methods have lower errors with the correct z than with the three replacements. This shows that FDM's predictions depend on sample-corresponding latent actions. With the action probe and boundary-crossing result, these experiments support shared window training and adaptive temporal support selection. This comparison does not separate the contribution of either individual component.

Table 3: Latent-action replacement diagnostics. Errors are squared L_2 errors per transition, summed over 64 coordinates (lower is better). Adaptive averages over its selected window; Fixed-4-OneStep evaluates only the center transition. Compare replacements within each row. Results are means $\pm$ sample standard deviations across three training seeds.

Noise RMS	Method	Correct z	Shuffled z	Mean z	Zero z
2	Adaptive	4.759 ± 0.016	8.423 ± 0.051	6.854 ± 0.010	6.947 ± 0.035
2	Fixed-4-OneStep	5.231 ± 0.017	8.685 ± 0.025	6.983 ± 0.012	7.180 ± 0.260
4	Adaptive	16.316 ± 0.023	19.520 ± 0.041	18.148 ± 0.008	18.263 ± 0.082
4	Fixed-4-OneStep	17.344 ± 0.033	20.640 ± 0.084	19.025 ± 0.102	19.285 ± 0.260

Table 4: Action chunk boundary detection on validation trajectories. F1 is evaluated at matching tolerances of 0, 1, and 2 time steps (higher is better). Results are means $\pm$ sample standard deviations across three training seeds.

Noise RMS	Method	F1 (± 0)	F1 (± 1)	F1 (± 2)
2	Full method	0.065 ± 0.004	0.501 ± 0.006	0.606 ± 0.003
2	Valleys only	0.057 ± 0.002	0.419 ± 0.004	0.545 ± 0.002
4	Full method	0.095 ± 0.005	0.367 ± 0.002	0.496 ± 0.004
4	Valleys only	0.090 ± 0.005	0.343 ± 0.004	0.481 ± 0.006

4.2 Action Chunk discovery from Temporal Support

Examine whether the temporal supports learned in Section 3.1 can also support action chunk discovery. Continue using data, validation split, and three training seeds from Section 4.1, reusing corresponding frozen models, and keeping chunking parameters unchanged within both noise levels. Boundary detection and representation selection neither use real actions nor real boundaries and not additionally train chunking model. Results are means and standard deviations across the three seeds.

Evaluate predicted boundary recovery through one-to-one matching with true boundaries, calculating F1 scores at tolerances of 0, 1, 2 time steps. To test the function of reconstruction error, the full method is compared with a version that retains the same valley candidates but omits confirmation, marked as Valleys Only. This is an ablation of the confirmation mechanism. See Table 4.

The confirmation procedure increases precision and F1 scores in every run, at the cost of lower recall. It filters false detections but also discards some candidates near true boundaries. With stronger noise, both F1 at tolerances of 1-2 steps and the benefits of confirmation decrease. Exact-match F1 is low, inferring locating ability accuracy is limited. Confirmation procedures filters candidate, not changing their positions.

To evaluate chunk representatives, a linear action classifier is fitted on training trajectories and freezes it. Each z_m predicts an action and assigns this action to the whole chunk. Accuracy is measured with transitions. As the upper bound of diagnosis, predicted chunks are fixed and assign each chunk its real majority action. Subsets using fully contained supports and the $k = 1$ fallback group are also reported. Effect MSE compares $G_\phi(z_m)$ with q_t at each transition in the chunk, getting the average over transitions and observation dimensions. See Table 5.

The gap between representatives' accuracy and upper bound is increasing with strong noise, which indicates limitations in representative selection. Higher upper bound does not necessarily imply better boundary recovery, because stronger noise leads to shorter predicted chunks, which are more likely to be action-consistent. Fallback group accuracy is lower, because the classifier is fitted on adaptive latent representations. Under stronger noise, whole chunk effect error also increases.

These results show that using temporal support as boundary cue requires no additional chunking model, and reconstruction error confirmation also helps select candidates. Precise boundaries recovery and the ability of one representative latent action of whole chunk are limited.

Table 5: Evaluation of action chunk representatives on validation trajectories. Accuracies are transition-weighted; effect MSE averages over transitions and observation dimensions. The majority-action oracle uses true labels for evaluation. Results are means ± sample standard deviations across three training seeds.

Metric	RMS $= 2$	RMS $= 4$
Whole-chunk action accuracy (%) ↑	70.36 ± 0.48	63.63 ± 0.73
Majority-action oracle (%)	81.07 ± 0.41	82.34 ± 0.22
Contained-support subset accuracy (%) ↑	76.20 ± 0.70	69.97 ± 0.77
Fallback subset accuracy (%) ↑	38.37 ± 0.77	38.89 ± 0.60
Whole-chunk effect MSE ↓	0.02536 ± 0.00048	0.03003 ± 0.00007

4.3 Exploring Concurrent Latent Action Factorization

Further explore whether ALHLAM can recover concurrent action factors and their temporal supports jointly. A noise-free four-dimensional environment is constructed with binary actions $a_t \in 0, 1^4$ and observations governed by $o_{t+1} = o_t + a_t$. Actions remain during consecutive intervals that may overlap. At least one action is active at any time. The identity basis used to generate the data is not provided to the model.

The IDM predicts $z_t \in [0, 1]^4$ and separate left and right extents for each factor from the observation window. Temporal masks $m_{t,j} \in [0, 1]^4$ are defined for these extents, where j is the offset from the center. The forward model uses an action-effect basis $W \in \mathbb{R}^{4\times4}$ to predict transitions through

$$\widehat{\Delta o}_{t+j} = W\left(m_{t,j} \odot z_t\right). \tag{14}$$

where, $\odot$ represents element-wise multiplication, and each column of W has unit norm. Besides reconstruction loss, training also use a loss for predicting changes between adjacent transition effects, without sparsity penalty loss. A diagnostic is also examined to filter training windows which include actions that are absent at the center. Real actions are only used for training the window filter, not for network inputs or prediction targets. Inference does not use this filter.

Both methods are trained once. Exact accuracy over 60,000 validation centers is 7.26% and 7.15%. This metric requires, after a single global factor alignment, all four activation states correct. Both models activate the same single latent factor over 99% of centers. Hence, this is a failure to recover the concurrent actions decomposition.

The factor decoder column approaches $\frac{1}{2}(1, 1, 1, 1)^\top$, satisfying the unit norm constraint, and producing transition predictions in center near 0.5 in every coordinate. Since each true coordinate is either 0 or 1, this prediction has a per-coordinate squared error of 0.25, which is consistent with the selected-window reconstruction MSE of about 0.25. The results show the limit of current model and training method: adaptive temporal support did not factorize correctly when action-effect basis was learned jointly.

5 Conclusion

This paper proposes ALHLAM, a method for learning shared latent action with adaptive temporal support from action label free observed trajectories. By requiring one latent action to explain all transitions within a window, this method balanced the reconstruction consistency and support length. In a controlled additive environment with independent transition noise, compared with a fixed-context model trained by single-step supervision, ALHLAM improves action decoding and reduces boundary crossing. In the meantime, the selected supports also enable action chunk discovery using the same frozen model, although the precise boundary localization and representation of entire chunks are still limited. The exploratory extension of concurrent actions did not recover the underlying action factors reliably. Future work will continue to explore combining adaptive temporal support with factored latent actions learning to recover concurrent actions.

AI USE STATEMENT

In this work, I used generative AI tools for code implementation, debugging, and experimental design. I have reviewed all AI-assisted work. LLM-generated code was verified and tested for correctness. I take responsibility for the final content of this work.

ETHICS STATEMENT

This study uses synthetic trajectories generated in controlled environments and does not involve human participants or personal data.

REPRODUCIBILITY STATEMENT

Code for reproducing the experiments is provided in the supplementary material, including synthetic trajectory generation, model training, and evaluation. Section 3 describes the shared latent action model, adaptive temporal support selection, and action segment discovery procedure. Section 4 presents the experimental setup, baselines, and evaluation protocols. For the main experiments, means and standard deviations are reported across three training seeds.

REFERENCES

Marvin Alles, Xingyuan Zhang, Patrick van der Smagt, and Philip Becker-Ehmck. Latent action world models for control with unlabeled trajectories. *ArXiv*, abs/2512.10016, 2025. URL https://api.semanticscholar.org/CorpusID:283737080.

Mahmoud Assran, Adrien Bardes, David Fan, Quentin Garrido, Russell Howes, Mojtaba Komeili, Matthew Muckley, Ammar Rizvi, Claire Roberts, Koustuv Sinha, Artem Zholus, Sergio Arnaud, Abha Gejji, Ada Martin, Francois Robert Hogan, Daniel Dugas, Piotr Bojanowski, Vasil Khalidov, Patrick Labatut, Francisco Massa, Marc Szafraniec, Kapil Krishnakumar, Yong Li, Xiaodong Ma, Sarath Chandar, Franziska Meier, Yann LeCun, Michael Rabbat, and Nicolas Ballas. V-jepa 2: Self-supervised video models enable understanding, prediction and planning. *arXiv preprint arXiv:2506.09985*, 2025.

Jake Bruce, Michael Dennis, Ashley Edwards, Jack Parker-Holder, Yuge Shi, Edward Hughes, Matthew Lai, Aditi Mavalankar, Richie Steigerwald, Chris Apps, Yusuf Aytar, Sarah Bechtle, Feryal M. P. Behbahani, Stephanie Chan, Nicolas Manfred Otto Heess, Lucy Gonzalez, Simon Osindero, Sherjil Ozair, Scott E. Reed, Jingwei Zhang, Konrad Zolna, Jeff Clune, Nando de Freitas, Satinder Singh, and Tim Rocktaschel. Genie: Generative interactive environments. *ArXiv*, abs/2402.15391, 2024. URL https://api.semanticscholar.org/CorpusID:267897982.

Qingwen Bu, Yanting Yang, Jisong Cai, Shenyuan Gao, Guanghui Ren, Maoqing Yao, Ping Luo, and Hongyang Li. Univla: Learning to act anywhere with task-centric latent actions. *ArXiv*, abs/2505.06111, 2025. URL https://api.semanticscholar.org/CorpusID:278481174.

Zhejia Cai, Yandan Yang, Xinyuan Chang, Shiyi Liang, Ronghan Chen, Feng Xiong, Mu Xu, and Ruqi Huang. Seeing space and motion: Enhancing latent actions with geometric and dynamic awareness for vision-language-action models, 2026. URL https://arxiv.org/abs/2509.26251.

Yash Chandak, Georgios Theocharous, James Kostas, Scott Jordan, and Philip Thomas. Learning action representations for reinforcement learning. In Kamalika Chaudhuri and Ruslan Salakhutdinov (eds.), *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 941–950. PMLR, 09–15 Jun 2019. URL https://proceedings.mlr.press/v97/chandak19a.html.

Zichen Jeff Cui, Hengkai Pan, Aadhithya Iyer, Siddhant Haldar, and Lerrel Pinto. Dynamo: In-domain dynamics pretraining for visuo-motor control. *ArXiv*, abs/2409.12192, 2024. URL https://api.semanticscholar.org/CorpusID:272709175.

Zexing Du, Xue Wang, Guoqing Zhou, and Qing Wang. Fast and unsupervised action boundary detection for action segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3323–3332, June 2022.

Ashley D. Edwards, Himanshu Sahni, Yannick Schroecker, and Charles Lee Isbell. Imitating latent policies from observation. *ArXiv*, abs/1805.07914, 2018. URL https://api.semanticscholar.org/CorpusID:29156793.

Yonathan Efroni, Dipendra Kumar Misra, Akshay Krishnamurthy, Alekh Agarwal, and John Langford. Provably filtering exogenous distractors using multistep inverse dynamics. In *International Conference on Learning Representations*, 2022. URL https://api.semanticscholar.org/CorpusID:251649239.

Marcus Fechner, Hamza Adnan, Constantin C. Lüth, Matthew T. Jackson, Alexey Zakharov, and J. Marius Zöllner. Segment to focus: Guiding latent action models in the presence of distractors, 2026. URL https://arxiv.org/abs/2602.02259.

Shenyuan Gao, Siyuan Zhou, Yilun Du, Jun Zhang, and Chuang Gan. Adaworld: Learning adaptable world models with latent actions. *ArXiv*, abs/2503.18938, 2025. URL https://api.semanticscholar.org/CorpusID:277940714.

Quentin Garrido, Tushar Nagarajan, Basile Terver, Nicolas Ballas, Yann LeCun, and Michael Rabbat. Learning latent action world models in the wild. *ArXiv*, abs/2601.05230, 2026. URL https://api.semanticscholar.org/CorpusID:284543722.

Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 16000–16009, June 2022.

Pu Hua, Yubei Chen, and Huazhe Xu. Simple emergent action representations from multi-task policy training. *ArXiv*, abs/2210.09566, 2022. URL https://api.semanticscholar.org/CorpusID:252967887.

Sukjun Hwang, Brandon Wang, and Albert Gu. Dynamic chunking for end-to-end hierarchical sequence modeling. *ArXiv*, abs/2507.07955, 2025. URL https://api.semanticscholar.org/CorpusID:280250049.

Tomoshi Iiyama, Masahiro Suzuki, and Yutaka Matsuo. Sunta: Hierarchical video prediction with surprise-based chunking, 2026. URL https://arxiv.org/abs/2607.02087.

Hanjung Kim, Lerrel Pinto, and Seon Joo Kim. Hierarchical latent action model. *ArXiv*, abs/2603.05815, 2026. URL https://api.semanticscholar.org/CorpusID:286365580.

Taesup Kim, Sungjin Ahn, and Yoshua Bengio. Variational temporal abstraction. *ArXiv*, abs/1910.00775, 2019. URL https://api.semanticscholar.org/CorpusID:202780888.

Thomas Kipf, Yujia Li, Hanjun Dai, Vinicius Zambaldi, Alvaro Sanchez-Gonzalez, Edward Grefenstette, Pushmeet Kohli, and Peter Battaglia. CompILE: Compositional imitation learning and execution. In Kamalika Chaudhuri and Ruslan Salakhutdinov (eds.), *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 3418–3428. PMLR, 09–15 Jun 2019. URL https://proceedings.mlr.press/v97/kipf19a.html.

Albina Klepach, Alexander Nikulin, Ilya Zisman, Denis Tarasov, Alexander Derevyagin, Andrei Polubarov, Nikita Lyubaykin, and Vladislav Kurenkov. Object-centric latent action learning. In *AAAI Conference on Artificial Intelligence*, 2025. URL https://api.semanticscholar.org/CorpusID:276394815.

Alex Lamb, Riashat Islam, Yonathan Efroni, Aniket Didolkar, Dipendra Misra, Dylan Foster, Lekan Molu, Rajan Chari, Akshay Krishnamurthy, and John Langford. Guaranteed discovery of control-endogenous latent states with multi-step inverse models. *Transactions on Machine Learning Research*, 2023-February, February 2023. ISSN 2835-8856. Publisher Copyright: © 2023, Transactions on Machine Learning Research. All rights reserved.

Jung Min Lee, Taehyun Cho, Li Zhao, and Jungwoo Lee. Why latent actions fail, and how to prevent it. *ArXiv*, abs/2605.20223, 2026a. URL https://api.semanticscholar.org/CorpusID:288654525.

Jung-Min Lee, Dohyeok Lee, Seokhun Ju, Taehyun Cho, Jin Woo Koo, Li Zhao, Sangwoo Hong, and Jungwoo Lee. Mvp-lam: Learning action-centric latent action via cross-viewpoint reconstruction. *ArXiv*, abs/2602.03668, 2026b. URL https://api.semanticscholar.org/CorpusID:285276215.

Alexander Levine, Peter Stone, and Amy Zhang. Multistep inverse is not all you need. *RLJ*, 2:884–925, 2024. URL https://api.semanticscholar.org/CorpusID:268531208.

Anthony Liang, Pavel Czempin, Matthew Hong, Yutai Zhou, Erdem Biyik, and Stephen Tu. Clam: Continuous latent action models for robot learning from unlabeled demonstrations. *ArXiv*, abs/2505.04999, 2025. URL https://api.semanticscholar.org/CorpusID:278394547.

Yihan Lin, Haoyang Li, Yang Li, Haitao Shen, Yihan Zhao, Chao Shao, and Jing Zhang. From pixels to tokens: A systematic study of latent action supervision for vision-language-action models, 2026. URL https://arxiv.org/abs/2605.04678.

Heejeong Nam, Chandradithya S Jonnalagadda, Harshit Aggarwal, Eric Xu, and Randall Balestriero. Latent actions from factorized transition effects under agent ambiguity, 2026. URL https://arxiv.org/abs/2606.30544.

Alexander Nikulin, Ilya Zisman, Denis Tarasov, Nikita Lyubaykin, Andrei Polubarov, Igor Kiselev, and Vladislav Kurenkov. Latent action learning requires supervision in the presence of distractors. *ArXiv*, abs/2502.00379, 2025. URL https://api.semanticscholar.org/CorpusID:276095040.

Dominik Schmidt and Minqi Jiang. Learning to act without actions. *ArXiv*, abs/2312.10812, 2023. URL https://api.semanticscholar.org/CorpusID:266359570.

Zuojin Tang, Haoyun Liu, Xinyuan Chang, Changjie Wu, Dongjie Huo, Yandan Yang, Bin Liu, Zhejia Cai, Feng Xiong, Mu Xu, jiachen Luo, De Ma, Zhiheng Ma, and Gang Pan. Alam: Algebraically consistent latent action model for vision-language-action models, 2026a. URL https://arxiv.org/abs/2605.10819.

Zuojin Tang, Feifan Luo, Haoyun Liu, Botai Yuan, Dekang Qi, Ronghan Chen, Yandan Yang, Tong Lin, Xinyuan Chang, Mu Xu, Bin Liu, De Ma, and Zhiheng Ma. Dlam: Distributional latent actions with temporal constraints, 2026b. URL https://arxiv.org/abs/2607.27138.

Bahey Tharwat, Yara Nasser, Ali Abouzeid, and Ian Reid. Latent action pretraining through world modeling. *ArXiv*, abs/2509.18428, 2025. URL https://api.semanticscholar.org/CorpusID:281496561.

Angel Villar-Corrales and Sven Behnke. Playslot: Learning inverse latent dynamics for controllable object-centric video prediction and planning. *ArXiv*, abs/2502.07600, 2025. URL https://api.semanticscholar.org/CorpusID:276259192.

Yucen Wang, Shenghua Wan, Le Gan, Shuai Feng, and De-Chuan Zhan. Ad3: Implicit action is the key for world models to distinguish the diverse visual distractors. *ArXiv*, abs/2403.09976, 2024. URL https://api.semanticscholar.org/CorpusID:268510062.

Zizhao Wang, Chang Shi, Jiaheng Hu, Kevin Rohling, Roberto Mart'in-Mart'in, Amy Zhang, and Peter Stone. Factored latent action world models. *ArXiv*, abs/2602.16229, 2026. URL https://api.semanticscholar.org/CorpusID:285725804.

Hangxing Wei, Xiaoyu Chen, Chuheng Zhang, Tim Pearce, Jianyu Chen, Alex Lamb, Li Zhao, and Jiang Bian. Learning additively compositional latent actions for embodied ai. *ArXiv*, abs/2604.03340, 2026. URL https://api.semanticscholar.org/CorpusID:287200353.

Seonghyeon Ye, Joel Jang, Byeongguk Jeon, Se June Joo, Jianwei Yang, Baolin Peng, Ajay Mandlekar, Reuben Tan, Yu-Wei Chao, Bill Yuchen Lin, Lars Lidén, Kimin Lee, Jianfeng Gao, Luke S. Zettlemoyer, Dieter Fox, and Minjoon Seo. Latent action pretraining from videos. *ArXiv*, abs/2410.11758, 2024. URL https://api.semanticscholar.org/CorpusID:273351190.

Chuheng Zhang, Tim Pearce, Pushi Zhang, Kaixin Wang, Xiaoyu Chen, Wei Shen, Li Zhao, and Jiang Bian. What do latent action models actually learn? *ArXiv*, abs/2506.15691, 2025a. URL https://api.semanticscholar.org/CorpusID:279465001.

Jiajie Zhang, Sören Schwertfeger, and Alexander Kleiner. From observation to action: Latent action-based primitive segmentation for vla pre-training in industrial settings. *ArXiv*, abs/2511.21428, 2025b. URL https://api.semanticscholar.org/CorpusID:283262346.