ROBUST SPATIO-TEMPORAL ADAPTATION TOWARDS 4D POINT CLOUD UNDERSTANDING

Anonymous authors
Paper under double-blind review

ABSTRACT

4D point cloud understanding jointly models fine-grained 3D geometry and temporal dynamics, playing a crucial role in robotics, autonomous driving and AR/VR. However, point cloud videos are often degraded by hardware noise, object occlusion and stream disruptions in real-world deployment. We reveal that existing state-of-the-art 4D point cloud understanding methods are vulnerable to such disturbances and formulate a unified evaluation protocol for spatial and temporal corruptions. To address these intrinsic weaknesses, we further propose a novel training-free method, Spatio-TEmporal Adaptation with Decoding dYnamics (STEADY), which adapts corrupted geometric evidence and recovers disrupted temporal dynamics at test time. Specifically, our proposed STEADY method comprises two stages: (1) The Hierarchical Spatial Cache retrieves complementary global and local geometric cues while Soft Temporal Consistency Gate modulates cache corrections based on inter-frame consistency. (2) The Dynamics-Aware Decoder incorporates dynamics of transition and duration priors to recover temporally coherent predictions over the complete sequence. We conduct extensive experiments on three widely used 4D point cloud understanding benchmarks, including MSR-Action3D, NTU-RGBD60 and HOI4D, under eight corruption conditions across both spatial and temporal dimensions. Experimental results demonstrate that STEADY consistently improves the overall robustness of various representative backbones across spatial and temporal corruptions. Our anonymous code is available at `https://anonymous.4open.science/r/STEADY/`.

1 INTRODUCTION

4D point cloud video understanding requires jointly modeling spatial structure and temporal evolution, supporting applications in robotics, autonomous driving, and AR/VR. However, point cloud streams in real-world environments are often affected by sensor noise, transmission jitter, occlusion, and calibration errors. These disturbances can weaken geometric cues within individual frames, obscure motion patterns and destabilize predicted segments, impairing geometric perception and temporal reasoning. Recent advances in architectures and representation learning have achieved strong performance on clean benchmarks (Jing et al., 2024; Liu et al., 2025; Wang et al., 2025; Huang et al., 2026), yet the robustness of these models under out-of-distribution scenarios remains unclear. Although corruption robustness has been studied for static point clouds (Ren et al., 2022; Li et al., 2023; Dong et al., 2023) and conventional videos (Schiappa et al., 2023; Zeng et al., 2026), the robustness of point cloud video models to spatio-temporal corruptions remains underexplored.

To systematically investigate this problem, we establish a unified robustness evaluation protocol by applying 5 spatial corruptions and 3 temporal perturbations to three widely used datasets: MSR-Action3D (Li et al., 2010), NTU-RGBD60 (Shahroudy et al., 2016) and HOI4D (Liu et al., 2022). As illustrated in Figure 1, the spatial corruptions simulate clutter, occlusion, measurement noise and pose drift, while the temporal perturbations simulate frame stalls, losses and reordering. The resulting evaluation covers two downstream tasks including action recognition and segmentation. Our evaluation reveals that representative models suffer substantial performance degradation under spatial and temporal corruptions, with robustness varying across architectures and corruption types.

To address these issues, we explore a training-free test-time adaptation approach for robust 4D point cloud understanding. Unlike spatial patches that describe local regions within a single point cloud,

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

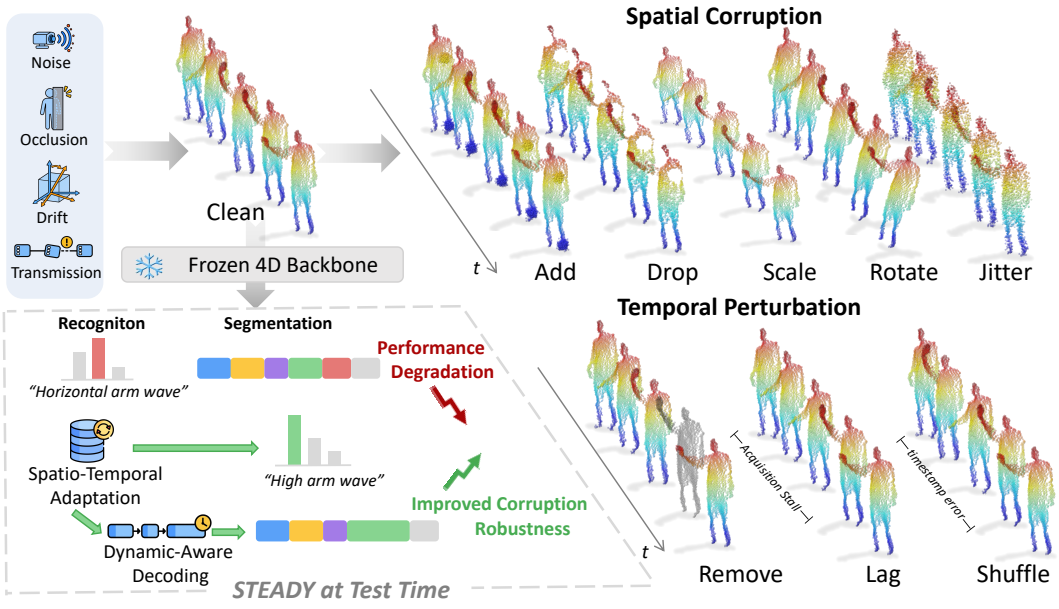


Figure 1: Overview of the robustness evaluation protocol under spatial and temporal corruptions. Our proposed STEADY method improves 4D point cloud model robustness at test time.

point cloud video models typically construct spatio-temporal point tubes that link geometric structures across frames (Fan et al., 2021a; Lv et al., 2025; Liu et al., 2025). Corruptions can therefore degrade not only the geometric cues within individual tubes but also their cross-frame continuity and the underlying sequence dynamics, posing additional challenges for test-time adaptation. We consequently propose ***S**patio-**TE**mporal **A**daptation with **D**ecoding d**Y**namics (**STEADY**)*, which combines spatio-temporal test-time adaptation with dynamics-aware decoding to refine corrupted geometric representations and recover disrupted temporal structures. The dynamics priors are estimated once from training annotations and STEADY operates without additional neural parameters.

Specifically, STEADY first employs a *Hierarchical Spatial Cache (HSC)* that updates online as point cloud videos are processed. It maintains positive and negative entries for global geometric features and local tube features, capturing complementary spatial cues from test observations. For each new observation, HSC retrieves similar entries and aggregates their class-wise contributions to correct model scores under corruption. The *Soft Temporal Consistency Gate (STCG)* then modulates this correction according to agreement between neighboring frames, allowing the test-time adaptation to respond to degraded spatial observations and disrupted temporal continuity. Beyond neighboring-frame consistency, the *Dynamics-Aware Decoder (DAD)* models temporal structure over the full video using an explicit-duration hidden semi-Markov model (Yu, 2010). It jointly evaluates the adapted frame scores and dynamic priors, favoring sequences with plausible state changes and persistence. This structured decoding strengthens temporal robustness under corruption and promotes coherent sequence-level predictions. Extensive experiments across multiple tasks and backbones demonstrate that STEADY consistently improves robustness to spatial and temporal corruptions.

Our contributions are summarized as follows:

- We establish a unified evaluation protocol for robust 4D point cloud perception, covering 5 spatial corruptions and 3 temporal perturbations across 3 datasets and 2 downstream tasks. Our evaluation reveals substantial performance degradation across state-of-the-art methods under these corruptions, exposing their limited robustness in real-world scenarios.
- We develop a Hierarchical Spatial Cache coupled with a Soft Temporal Consistency Gate for test-time adaptation. These components leverage global and local geometric cues with inter-frame consistency to improve robustness against spatial and temporal corruptions.
- We propose a Dynamics-Aware Decoder built on an explicit-duration hidden semi-Markov model, jointly modeling action transitions and duration dynamics to recover coherent temporal structures and improve temporal stability under corruption.

2 Related Work

2.1 Point Cloud Video Understanding

Early approaches (Wang et al., 2020; Choy et al., 2019) typically simplify this spatio-temporal structure of point cloud video through voxelization, while recent point-based methods (Liu et al., 2019; Fan et al., 2021b) operate directly on point cloud sequences to preserve fine-grained geometric information. P4Transformer (Fan et al., 2021a) and PST-Transformer (Fan et al., 2022) use attention to capture spatial and temporal dependencies, while PPTr (Wen et al., 2022) uses Transformers operating on points to model global temporal dependencies. MotionPDE (Huang et al., 2026) explores a PDE-based formulation to regularize motion representation learning and Mamba4D (Liu et al., 2025) investigates efficient temporal modeling with state space models. Existing architectures assume clean observations and lack mechanisms for handling corrupted geometry and disrupted temporal dynamics. We systematically investigate these limitations through robustness evaluation.

2.2 Corruption Robustness

ImageNet-C (Hendrycks & Dietterich, 2019) established common-corruption evaluation for image recognition. This paradigm has been progressively extended to 3D point clouds, spanning point cloud classification (Ren et al., 2022), detection (Li et al., 2023; Dong et al., 2023), and broader 3D perception systems (Kong et al., 2023). Recent efforts have also considered corruption robustness in multimodal large language models (Qiu et al., 2025) and temporal video understanding (Schiappa et al., 2023; Zeng et al., 2026). Despite substantial progress in 2D and 3D robust perception, 4D point cloud understanding is more challenging due to its complex structure combining 3D geometry with evolving temporal dynamics. These challenges motivate us to evaluate existing 4D point cloud models and further develop a training-free method for robust spatio-temporal understanding.

2.3 Test-Time Adaptation

Test-time adaptation (TTA) improves robustness to distribution shifts using unlabeled target observations (Liang et al., 2025; Dong et al., 2026). Early TTA methods, designed for vision models, adapt parameters through regularization during inference (Zhang et al., 2022; Wang et al., 2021). Recent studies, such as TDA (Karmanov et al., 2024) and COSMIC (Huang et al., 2025), refine predictions by matching test features against cached high-confidence prototypes. For point clouds, Point-Cache (Sun et al., 2025) uses a hierarchical cache of point patch cues, while other methods explore prototype-based and Bayesian online adaptation (Tamjidi et al., 2026; Zhu et al., 2026). However, these methods target static point clouds and neglect the temporal dependencies and cross-frame consistency of point cloud videos. Our method instead extends robustness adaptation to the spatio-temporal setting by exploiting intra-frame cues and inter-frame consistency.

3 Evaluation Protocol

3.1 Corruption Taxonomy

Our evaluation protocol comprises five spatial corruptions and three temporal perturbations, each applied independently to clean point cloud videos. Spatial corruptions degrade 3D observations while preserving frame order, whereas temporal perturbations disrupt frame continuity or ordering without modifying per-frame geometry. We present more detailed definitions and implementation procedures of our corruption simulation algorithms in Appendix A.

Spatial corruptions. Spatial degradation can arise from changes in point density, noisy measurements, or geometric misalignment (Ren et al., 2022). *Add* inserts clusters of noisy points around locations sampled from the first frame and held fixed throughout the video, approximating clutter or spurious sensor returns. *Drop* removes points near fixed locations across frames, creating local missing regions that approximate occlusion or incomplete scanning. *Scale* expands or contracts one randomly selected coordinate axis using a factor shared across the video, approximating errors in geometric calibration or coordinate normalization. *Jitter* independently perturbs point coordinates in each frame with Gaussian noise to approximate measurement uncertainty. *Rotate* progressively rotates each frame around its centroid, producing changes in orientation that approximate pose drift.

Temporal perturbations. Beyond geometric degradation, irregular acquisition and transmission can disrupt frame continuity and order (Yi et al., 2021; Zeng et al., 2026). *Lag* repeats a randomly selected frame and skips subsequent frames to preserve video length, approximating an acquisition stall or frozen stream. *Remove* randomly discards frames while retaining the order of those that remain, approximating dropped captures or transmission loss. *Shuffle* permutes frames within a local temporal interval, approximating out-of-order delivery or timestamp errors. For HOI4D-P, shuffling is restricted to intervals within the same annotated segment. When perturbations change frame indices, the corresponding frame annotations are transformed with the point cloud sequence.

3.2 Benchmark Datasets

For recognition, we consider MSR-Action3D (Li et al., 2010) and NTU-RGBD60 (Shahroudy et al., 2016). MSR-Action3D contains 567 depth videos spanning 20 action classes and approximately 23K frames in total. NTU-RGBD60 comprises 56K videos from 60 action classes and 4M frames, captured from 40 subjects and three camera views. We follow the standard cross-subject evaluation protocol of prior work (Wang et al., 2012; Liu et al., 2019) for both datasets. For segmentation, HOI4D (Liu et al., 2022) provides 2971 training and 892 evaluation sequences, each containing 150 frames with 2048 points per frame. Each frame is annotated with one of 19 fine-grained action classes. Applying the aforementioned corruptions independently to each dataset's evaluation split yields eight corrupted subsets for each of MSRAction3D-C, NTU60-C, and HOI4D-P.

3.3 Evaluation Metrics

To assess corruption robustness more comprehensively, we use overall accuracy (OA) as the mean accuracy across the eight corrupted subsets. For action segmentation, overall edit score (OE) and overall F1 score (OF) are additionally used, each averaging the corresponding segmental edit score or F1 score at temporal IoU threshold τ across the same subsets.

Following Hendrycks & Dietterich (2019); Ren et al. (2022), we measure robustness using mean corruption error (mCE). We choose P4Transformer (Fan et al., 2021a), a representive 4D model, as the reference baseline for all tasks. For recognition:

$$\mathrm{CE}_i = \frac{1 - \mathrm{OA}_i}{1 - \mathrm{OA}_i^{\mathrm{baseline}}}, \qquad \mathrm{mCE} = \frac{1}{N}\sum_{i=1}^{N} \mathrm{CE}_i. \tag{1}$$

We also report mean relative corruption error (mRCE), which quantifies the relative performance degradation under each corruption with respect to the clean test set:

$$\mathrm{RCE}_i = \frac{\mathrm{OA}_{\mathrm{clean}} - \mathrm{OA}_i}{\mathrm{OA}_{\mathrm{clean}}^{\mathrm{baseline}} - \mathrm{OA}_i^{\mathrm{baseline}}}, \qquad \mathrm{mRCE} = \frac{1}{N}\sum_{i=1}^{N} \mathrm{RCE}_i. \tag{2}$$

Here, N is the total number of evaluated corruption types, OA_i denotes recognition accuracy under corruption i, and $\mathrm{OA}_{\mathrm{clean}}$ denotes the corresponding accuracy on the clean test set. For segmentation, we compute CE and RCE from the segmental edit score:

$$\mathrm{CE}_i = \frac{1 - \mathrm{OE}_i}{1 - \mathrm{OE}_i^{\mathrm{baseline}}}, \qquad \mathrm{RCE}_i = \frac{\mathrm{OE}_{\mathrm{clean}} - \mathrm{OE}_i}{\mathrm{OE}_{\mathrm{clean}}^{\mathrm{baseline}} - \mathrm{OE}_i^{\mathrm{baseline}}}. \tag{3}$$

4 Method

4.1 Overview of STEADY

Spatial and temporal corruptions affect point cloud videos differently: degraded observations weaken geometric cues, whereas irregular frame sequences disrupt temporal continuity. As shown in Figure 2, STEADY addresses these distribution shifts at test time using a frozen 4D backbone. Given a corrupted video $\widetilde{\mathcal{P}}$, the backbone outputs initial prediction scores $\mathbf{z}$, global features $\mathbf{h}$, and point-tube features $\mathbf{L}$, denoted by $f_\theta(\widetilde{\mathcal{P}}) = (\mathbf{z}, \mathbf{h}, \mathbf{L})$. STEADY maintains an online cache that retrieves evidence from global features and clustered tube descriptors to refine these scores. Neighboring feature and prediction consistency controls the cache correction according to its temporal reliability.

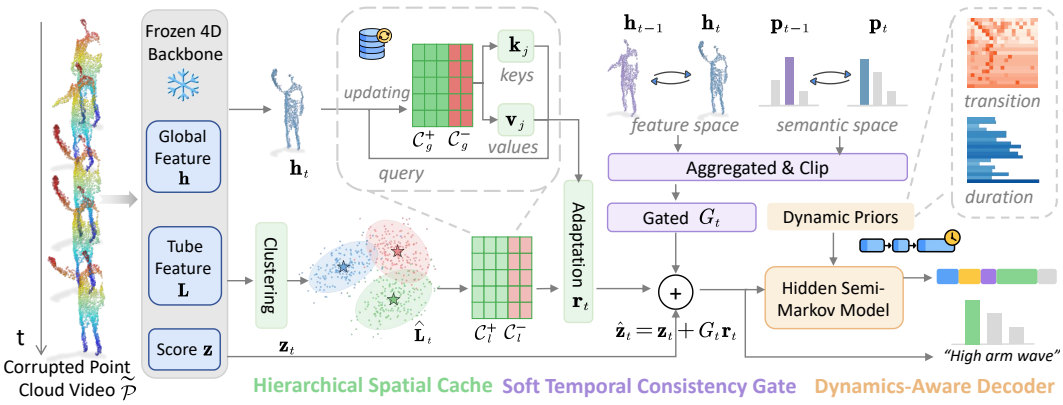


Figure 2: Framework of STEADY. HSC and STCG adapt predictions using spatial cues and temporal consistency, while DAD incorporates transition and duration priors for sequence decoding.

For framewise prediction, the adapted scores are decoded using dynamic priors estimated from training annotations. STEADY thus couples test-time geometric adaptation with sequence-level temporal modeling, improving spatio-temporal robustness without adding trainable neural parameters.

4.2 Spatio-Temporal Test-Time Adaptation

Hierarchical Spatial Cache. Adaptation methods for static point clouds represent local geometry using patches (Sun et al., 2025; Tamjidi et al., 2026; Zhu et al., 2026). In point cloud videos, local features instead form point tubes spanning neighboring frames. Caching every tube is redundant and sensitive to local corruption and HSC represents each test observation at global and local scales. Given the global feature $\mathbf{h}_t$ and M point-tube features $\mathbf{L}_t = \{\mathbf{l}_{t,n}\}_{n=1}^{M}$ of the t-th observation, HSC retains $\mathbf{h}_t$ globally and applies K-means clustering to $\mathbf{L}_t$ in feature space. The resulting R cluster centroids, denoted by $\hat{\mathbf{L}}_t = \{\hat{\mathbf{l}}_{t,r}\}_{r=1}^{R}$, provide a compact representation of the local tube structures.

HSC maintains positive and negative caches at the global and local representation levels, denoted by $\mathcal{C}_g^+$, $\mathcal{C}_l^+$, $\mathcal{C}_g^-$, and $\mathcal{C}_l^-$. Global cache entries use global features as keys, whereas local entries use clustered tube descriptors. Positive entries associate their keys with one-hot pseudo labels and provide support for likely classes. Negative entries provide suppression scores for ambiguous classes. Constructed online with bounded capacity for each predicted class, the caches use prediction uncertainty to prioritize reliable observations and replace less reliable entries during inference. The complete cache construction and replacement procedure is provided in Appendix B.

For a query representation $\mathbf{x}$ and a cache $\mathcal{C} = \{(\mathbf{k}_j, \mathbf{v}_j)\}_{j=1}^{K}$, the cache score is computed by aggregating its class-wise values according to feature similarity:

$$\Phi(\mathbf{x}; \mathcal{C}) = \alpha \sum_{j=1}^{K} \exp[-\beta\,(1 - s(\mathbf{x}, \mathbf{k}_j))]\,\mathbf{v}_j, \tag{4}$$

where K is the cache size, $\mathbf{k}_j$ and $\mathbf{v}_j$ denote the cached key and its corresponding class-wise value, respectively. The hyperparameters α and β control the magnitude and concentration of the retrieved scores. For the global branch, $s(\mathbf{h}_i, \mathbf{k}_j)$ is cosine similarity. For the local branch, similarity is averaged over the clustered tube descriptors:

$$s(\widehat{\mathbf{L}}_t, \mathbf{k}_j) = \frac{1}{R} \sum_{r=1}^{R} \cos(\hat{\mathbf{l}}_{t,r}, \mathbf{k}_j). \tag{5}$$

This hierarchical retrieval combines the overall geometric configuration captured by the global feature with the finer spatial structures encoded by point tubes. Finally, HSC combines positive support and negative suppression from both levels to produce a cache correction:

$$\mathbf{r}_t = \Phi(\mathbf{h}_t; \mathcal{C}_g^+) + \Phi(\widehat{\mathbf{L}}_t; \mathcal{C}_l^+) - \Phi(\mathbf{h}_t; \mathcal{C}_g^-) - \Phi(\widehat{\mathbf{L}}_t; \mathcal{C}_l^-). \tag{6}$$

The residual $\mathbf{r}_t$ complements the initial prediction with class information retrieved from similar cached observations. Since the usefulness of this correction can vary with the surrounding perturbations, we further regulate its contribution using the consistency mechanism introduced next.

Soft Temporal Consistency Gate. Although HSC retrieves complementary information, its correction varies in reliability with temporal context. Corrupted observations or irregular transitions may reduce the agreement between adjacent frames, making an unmodulated cache correction less reliable. We therefore introduce a Soft Temporal Consistency Gate (STCG) that regulates the contribution of HSC using temporal consistency. For framewise observation t, let $\mathbf{r}_t$ denote the cache correction produced by HSC and $\mathbf{p}_t = \mathrm{softmax}(\mathbf{z}_t)$ the initial prediction probability. STCG combines geometric feature consistency with semantic prediction consistency across adjacent frames:

$$s_t^h = \frac{1 + \cos(\mathbf{h}_t, \mathbf{h}_{t-1})}{2}, \qquad s_t^p = \cos(\mathbf{p}_t, \mathbf{p}_{t-1}). \tag{7}$$

where s_t^h and s_t^p measure inter-frame consistency in the feature and semantic prediction spaces, respectively. We rescale s_t^h to $[0, 1]$ to match the range of s_t^p. Together, they assess whether the cache correction agrees with both geometric continuity and semantic stability across adjacent frames.

STCG averages two consistency scores and uses the resulting gate to modulate the cache correction:

$$G_t = \mathrm{clip}\left(\frac{s_t^h + s_t^p}{2}, G_{\min}, 1\right), \qquad \hat{\mathbf{z}}_t = \mathbf{z}_t + G_t \mathbf{r}_t. \tag{8}$$

Here, $G_{\min}$ retains a minimum contribution from the retrieved cache information and we set $G_1 = 1$ for the first frame. High agreement across the two spaces preserves more of the cache correction, whereas low agreement attenuates its contribution. In this way, HSC refines degraded geometric evidence using global and local spatial representations, while STCG aligns the refinement with temporal continuity, jointly improving robustness to spatial and temporal corruptions.

4.3 Dynamics-Aware Decoding

Framewise adaptation improves the local prediction evidence but does not explicitly model the structure of the complete sequence. To incorporate longer-range temporal dynamics, we introduce a Dynamics-Aware Decoder (DAD) based on a hidden semi-Markov model (Yu, 2010). DAD jointly evaluates adapted frame scores, state transitions, and durations for each class to decode the video.

Dynamics Prior. We derive a compact description of sequence dynamics from the training annotations. Let n_c^{start} count sequences starting with class c, and let $n_{a,c}^{\mathrm{trans}}$ count segment-level transitions from class a to class c. The initial and transition priors are

$$\pi_c = \frac{n_c^{\mathrm{start}} + \alpha_0}{\sum_{j=1}^{C} n_j^{\mathrm{start}} + C\alpha_0}, \qquad A_{a,c} = \frac{n_{a,c}^{\mathrm{trans}} + \alpha_0}{\sum_{j=1}^{C} n_{a,j}^{\mathrm{trans}} + C\alpha_0}, \tag{9}$$

where C is the number of classes and α_0 is the smoothing coefficient. Consecutive frames with the same annotation are first merged into segments before transitions are counted.

To model class persistence, let $\mathcal{D}_c$ denote the durations of all training segments labeled as class c. We fit a log normal duration model for each class, with $\mu_c = \mathrm{mean}_{d \in \mathcal{D}_c}(\log d)$ and $\sigma_c = \mathrm{std}_{d \in \mathcal{D}_c}(\log d)$. These priors are estimated once and fixed during inference; no labels from corrupted test videos are used. They provide soft decoder scores rather than imposing a predetermined sequence, while smoothing assigns nonzero probability to transitions unseen during training.

Dynamics-Aware Decoder. Building on the adapted scores from STCG, DAD uses the estimated priors to assess predictions within the temporal structure of the full video. Given $\hat{\mathbf{z}}_t$, the frame evidence for class c is $\ell_{t,c} = [\log \mathrm{softmax}(\hat{\mathbf{z}}_t)]_c$. For a segment of class c lasting d frames, the duration score is given by the log-density of the fitted log-normal distribution:

$$\rho_c(d) = -\log\left(d\sigma_c\sqrt{2\pi}\right) - \frac{(\log d - \mu_c)^2}{2\sigma_c^2}. \tag{10}$$

To incorporate transitions between successive classes, DAD decodes the full video into a sequence of segments. Let $Y = (c_s, d_s, I_s)_{s=1}^{S}$ denote a sequence of S segments, where c_s, d_s and I_s specify the class, duration and frame interval of segment s, respectively. The intervals partition the video of

Table 1: Recognition performance on MSRAction3D-C and NTU60-C measured by Accuracy.

Method	Clean	Corruption Type								OA↑	mCE↓	mRCE↓
		Add	Drop	Scale	Jitter	Rotate	Lag	Remove	Shuffle			
MSRAction3D-C												
P4Transformer (Fan et al., 2021a)	90.9	90.1	79.8	69.7	90.2	83.8	86.2	82.2	87.5	83.7	1.00	1.00
+ STEADY (Ours)	**91.6**	89.9	**82.5**	**71.7**	**92.3**	**86.5**	**87.9**	**83.5**	**87.9**	**85.3**	**0.90**	**0.80**
PPTr (Wen et al., 2022)	91.5	84.5	75.8	64.3	88.2	70.4	85.2	85.2	84.9	79.8	1.26	2.89
+ STEADY (Ours)	**91.6**	**84.9**	**77.4**	62.6	**89.6**	**72.7**	**86.2**	84.9	**85.2**	**80.4**	**1.21**	**2.54**
PST-Transformer (Fan et al., 2022)	93.7	90.2	74.4	70.4	93.6	84.2	90.2	89.9	84.9	84.7	0.92	1.56
+ STEADY (Ours)	**94.0**	90.2	**78.5**	**71.0**	**93.9**	**84.5**	**90.6**	87.5	**86.2**	**85.3**	**0.88**	**1.56**
Mamba4D (Liu et al., 2025)	91.8	87.9	84.9	62.6	87.9	78.1	85.9	85.5	86.5	82.4	1.09	2.24
+ STEADY (Ours)	**92.1**	**90.6**	**86.5**	58.3	**89.6**	**79.5**	**86.5**	**86.5**	**87.2**	**83.1**	**1.01**	**1.57**
MotionPDE (Huang et al., 2026)	96.4	94.3	86.9	77.4	94.6	88.6	91.3	86.2	92.6	89.0	0.65	1.43
+ STEADY (Ours)	96.3	**94.6**	86.2	**79.5**	94.3	**88.8**	**91.9**	**87.2**	**93.3**	**89.5**	**0.63**	**1.33**
NTU60-C												
P4Transformer (Fan et al., 2021a)	89.0	79.7	52.5	77.0	86.9	81.1	86.6	85.1	77.6	78.3	1.00	1.00
+ STEADY (Ours)	**89.7**	**80.4**	**53.9**	77.0	**87.2**	**81.2**	86.4	84.6	**81.7**	**79.1**	**0.97**	1.09
PST-Transformer (Fan et al., 2022)	91.0	85.0	56.2	83.3	87.6	82.4	88.3	86.3	79.2	81.0	0.87	1.04
+ STEADY (Ours)	90.9	**85.7**	**57.2**	**83.4**	**88.1**	**83.2**	88.0	86.0	**81.5**	**81.6**	**0.85**	**0.96**
MotionPDE (Huang et al., 2026)	92.1	85.1	56.0	80.0	86.5	83.5	88.0	84.1	76.7	80.0	0.93	1.45
+ STEADY (Ours)	**92.2**	**85.5**	**57.4**	**80.1**	**87.8**	**83.9**	**88.1**	82.9	**79.0**	**80.6**	**0.90**	**1.38**

Table 2: Segmentation performance on HOI4D-P measured by Edit Accuracy.

Method	Clean	Corruption Type								OE↑	mCE↓	mRCE↓
		Add	Drop	Scale	Jitter	Rotate	Lag	Remove	Shuffle			
P4Transformer (Fan et al., 2021a)	73.1	62.7	66.9	55.4	64.3	63.7	59.3	46.5	62.4	60.1	1.00	1.00
+ STEADY (Ours)	**82.1**	**79.2**	**76.4**	**73.3**	**81.2**	**78.2**	**71.9**	**48.1**	**81.3**	**73.7**	**0.64**	**0.54**
PPTr (Wen et al., 2022)	80.1	75.8	75.9	65.4	75.8	71.7	67.1	42.8	69.5	68.0	0.79	0.83
+ STEADY (Ours)	**87.0**	**85.7**	**83.8**	**80.7**	**86.5**	**83.6**	**76.4**	42.3	**86.4**	**78.2**	**0.52**	**0.49**
X4D-SceneFormer (Jing et al., 2024)	91.1	90.2	88.3	84.5	76.4	88.3	72.3	51.2	77.8	78.6	0.52	0.87
+ STEADY (Ours)	**92.7**	**92.0**	**90.2**	**89.6**	**84.6**	**90.8**	**79.5**	**56.9**	**88.6**	**84.0**	**0.38**	**0.56**
Mamba4D (Liu et al., 2025)	91.3	91.4	89.0	86.9	80.6	89.3	72.1	61.4	76.2	80.9	0.47	0.75
+ STEADY (Ours)	**93.5**	**93.1**	**91.7**	**91.1**	**87.2**	**92.0**	**78.9**	**67.0**	**88.2**	**86.2**	**0.33**	**0.49**

T frames, with $\sum_{s=1}^{S} d_s = T$ and different classes for adjacent segments. DAD jointly evaluates frame evidence, segment durations and state transitions to obtain the final sequence:

$$Y^\star = \arg\max_{Y} \sum_{s=1}^{S} \left[\sum_{t \in I_s} \ell_{t,c_s} + \lambda_D \rho_{c_s}(d_s) + \lambda_T \begin{cases} \log \pi_{c_s}, & s = 1, \\ \log A_{c_{s-1},c_s}, & s > 1 \end{cases} \right]. \tag{11}$$

where λ_T and λ_D control the contributions of transition and duration dynamics. Because the objective decomposes over segments, its exact optimum can be obtained using hidden semi-Markov dynamic programming. The inference procedure is detailed in Appendix C. By combining the adapted frame evidence with class transitions and temporal persistence, DAD extends the refinement of HSC and STCG to full-sequence structure, strengthening temporal robustness under corruption.

5 Experiments

5.1 Experimental Setup

We conduct experiments on two downstream tasks, action recognition and segmentation, following dataset splits and evaluation metrics described in Sections. 3.2 and 3.3. We evaluate multiple representative point cloud video backbones, including P4Transformer (Fan et al., 2021a), PPTr (Wen et al., 2022), PST-Transformer (Fan et al., 2022), X4D-SceneFormer (Jing et al., 2024), Mamba4D (Liu et al., 2025), and MotionPDE (Huang et al., 2026). During evaluation, all backbone weights remain frozen. HSC uses $R = 3$ tube clusters and per-class cache capacities of 3 and 2 for the positive and negative caches, respectively. STCG sets $G_{\min} = 0.05$, while DAD uses $\lambda_T = 5$ and $\lambda_D = 1$. Additional implementation and runtime details are provided in Appendix D. For recognition, all clean-data results in the main text are obtained using the official codebases. Additional comparisons with the original paper results are provided in Appendix E.

Table 3: Component ablation under spatial and temporal corruptions.

HSC	STCG	DAD	Spatial		Temporal	
			OE↑	mRCE↓	OE↑	mRCE↓
–	–	–	87.45	0.407	69.88	1.310
✓	–	–	87.66	0.479	70.69	1.312
✓	✓	–	87.86	0.476	71.00	1.303
✓	✓	✓	**91.00**	**0.274**	**78.02**	**0.853**

Table 4: Comparison of memory usage (MB) and inference speed. The batch size is 1 and the device is an A6000. The number under each dataset denotes the number of classes.

Method	MSRAction3D-C (20)	NTU60-C (60)	HOI4D-P (19)	Throughput (iter/s)
P4Transformer	1022	2967	1508	3.9
+ STEADY (Ours)	1024	2972	1544	1.0

5.2 Overall Comparison Results

Robust Recognition. Table 1 reports recognition results on MSRAction3D-C and NTU60-C. Across both benchmarks, STEADY consistently improves OA for all evaluated backbones, with the corresponding reductions in mCE confirming its effectiveness after normalization against the reference model. This trend holds across models with different spatio-temporal designs, showing that STEADY is not tied to a particular architecture. Clean accuracy remains comparable after adaptation, demonstrating that STEADY preserves the original recognition capability while improving corruption robustness. Although mRCE can fluctuate because it normalizes corruption degradation by the performance gap between clean and corrupted inputs, the consistent OA and mCE improvements demonstrate reliable gains under diverse spatial and temporal corruptions.

Robust Segmentation. As shown in Table 2 , STEADY consistently improves OE and reduces both mCE and mRCE across all evaluated backbones on HOI4D-P. The improvement is observed under both spatial and temporal corruptions, demonstrating that STEADY enhances the overall robustness of sequence prediction. For X4D-SceneFormer and Mamba4D, the average improvement under temporal corruptions is greater than that under spatial corruptions, highlighting its ability to handle disrupted temporal dependencies. In particular, the strong improvements under *Lag* and *Shuffle* indicate that STEADY better preserves action ordering when frame continuity is disrupted. Additional framewise accuracy and F1 score results are provided in Appendix E, further supporting the effectiveness of STEADY from complementary evaluation perspectives.

5.3 Further Analysis

Overall Component Ablation. As shown in Table 3, each component progressively improves robustness under both corruptions. HSC strengthens framewise evidence by retrieving complementary global and local geometric information. STCG further regulates cache corrections using inter-frame agreement, bringing a larger benefit under temporal corruptions. DAD provides the largest improvement by incorporating transition and duration dynamics over the full sequence. Overall, STEADY improves OE by 3.55% and 8.14% under spatial and temporal corruptions. The larger gain on the temporal subset confirms that STCG and DAD effectively complement the spatial refinement of HSC when frame continuity and sequence dynamics are disrupted.

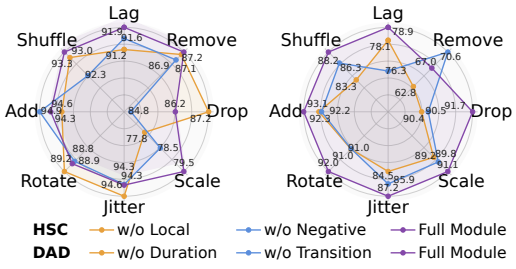
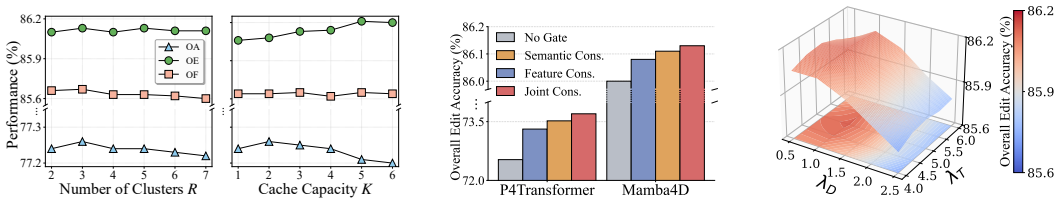


Figure 3: Component ablations of HSC (left) and DAD (right) across eight corruption types.

Analysis on HSC module. Figure 3 evaluates the local representations and negative cache entries in HSC. The global-only variant improves the backbone but remains inferior to the complete HSC on average, demonstrating the complementary value of clustered tube descriptors. Removing negative entries also reduces overall performance, confirming their role in suppressing ambiguous predictions. Although partial variants may perform better under individual corruptions, the complete HSC provides the most balanced robustness. We further study the number of tube clusters R and local cache capacity K in Figure 4a. The three evaluation metrics remain stable across the tested settings, indicating that HSC is not sensitive to moderate changes in feature granularity or cache capacity.

Analysis on STCG module. Figure 4b evaluates consistency modeling in the representation and semantic prediction spaces. Both spaces improve over the ungated correction, and their combination achieves the best performance on both backbones. Specifically, joint consistency improves OE by



(a) Effect of the number of clusters R and local cache capacity K.

(b) Ablation of STCG consistency signals.

(c) Sensitivity to λ_D and λ_T.

Figure 4: Comprehensive ablation and sensitivity analyses of STEADY components.

1.17% on P4Transformer and 0.13% on Mamba4D. The larger gain on P4Transformer suggests that STCG is particularly beneficial when the backbone is less robust to corruptions.

Analysis on DAD module. We examine the two forms of sequence dynamics used by DAD in Figure 3. Removing the transition term weakens the modeling of plausible action ordering, whereas removing the duration term limits the decoder's ability to capture class-specific temporal persistence. Combining both terms allows DAD to evaluate when an action should change and how long it should continue, producing the strongest overall performance across the evaluated corruption settings. The sensitivity analysis in Figure 4c shows relatively stable performance across different values of λ_T and λ_D, suggesting that DAD is reasonably robust to these hyperparameters.

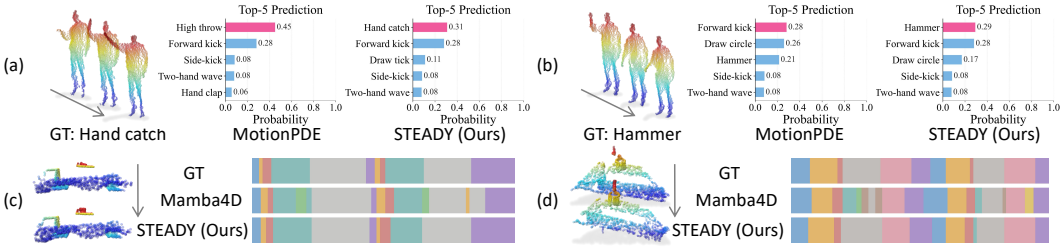


Figure 5: Qualitative results of STEADY on recognition and segmentation.

Qualitative Analysis. The qualitative comparisons in Figure 5 further illustrate the test-time robustness of STEADY. Under corrupted inputs, MotionPDE confuses *Hand Catch* and *Hammer* with similar actions, whereas STEADY correctly recognizes both actions. Likewise, Mamba4D produces fragmented segments and inaccurate boundaries, while STEADY yields predictions that are more coherent and better aligned with the ground truth. These results highlight improved semantic reliability and temporal consistency during test-time adaptation.

Efficiency Analysis. Table 4 reports the inference overhead of STEADY. The bounded caches and fixed dynamics priors introduce less than 2.5% memory overhead across all three datasets. Compared with static 3D point clouds (Sun et al., 2025), point cloud videos require adaptation over successive frames and sequence-level decoding, resulting in lower throughput. Improving the efficiency of cache retrieval and temporal decoding is a promising direction for future work.

6 Conclusion

In this work, we systematically investigated the corruption robustness of 4D point cloud understanding. We established a unified protocol covering five spatial corruptions and three temporal perturbations across three datasets and two downstream tasks, revealing substantial and architecture-dependent degradation in representative methods. To address these limitations, we proposed Spatio-TEmporal Adaptation with Decoding dYnamics (STEADY), a training-free test-time adaptation method that combines hierarchical spatial refinement guided by inter-frame consistency with dynamics-aware sequence decoding. We hope that our evaluation protocol and method will facilitate future research on robust 4D point cloud understanding.

AI USE STATEMENT

In this work, generative AI tools are used to support research ideation and execution, draft portions of the manuscript and polish the paper writing. They are not involved in research ideation, experimental design, data analysis or the formulation of conclusions. All substantive intellectual contributions are made by the authors.

REFERENCES

Christopher Choy, JunYoung Gwak, and Silvio Savarese. 4d spatio-temporal convnets: Minkowski convolutional neural networks. In *CVPR*, pp. 3075–3084, 2019.

Hao Dong, Moru Liu, Kaiyang Zhou, Eleni Chatzi, Juho Kannala, Cyrill Stachniss, and Olga Fink. Advances in multimodal adaptation and generalization: From traditional approaches to foundation models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2026.

Yinpeng Dong, Caixin Kang, Jinlai Zhang, Zijian Zhu, Yikai Wang, Xiao Yang, Hang Su, Xingxing Wei, and Jun Zhu. Benchmarking robustness of 3d object detection to common corruptions in autonomous driving. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1022–1032. IEEE, 2023.

Hehe Fan, Yi Yang, and Mohan Kankanhalli. Point 4d transformer networks for spatio-temporal modeling in point cloud videos. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 14199–14208. IEEE, 2021a.

Hehe Fan, Xin Yu, Yuhang Ding, Yi Yang, and Mohan Kankanhalli. Pstnet: Point spatio-temporal convolution on point cloud sequences. In *International Conference on Learning Representations*, 2021b.

Hehe Fan, Yi Yang, and Mohan Kankanhalli. Point spatio-temporal transformer networks for point cloud video modeling. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(2): 2181–2192, 2022.

Dan Hendrycks and Thomas G. Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*, 2019.

Fanding Huang, Jingyan Jiang, Qinting Jiang, Hebei Li, Faisal Nadeem Khan, and Zhi Wang. Cosmic: Clique-oriented semantic multi-space integration for robust clip test-time adaptation. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 9772–9781. IEEE, 2025.

Zhuoxu Huang, Zhenkun Fan, Jungong Han, and Josef Kittler. Paving the way for point cloud video representation learning using a pde model. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2026.

Linglin Jing, Ying Xue, Xu Yan, Chaoda Zheng, Dong Wang, Ruimao Zhang, Zhigang Wang, Hui Fang, Bin Zhao, and Zhen Li. X4d-sceneformer: Enhanced scene understanding on 4d point cloud videos through cross-modal knowledge transfer. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 2670–2678, 2024.

Adilbek Karmanov, Dayan Guan, Shijian Lu, Abdulmotaleb El Saddik, and Eric Xing. Efficient test-time adaptation of vision-language models. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 14162–14171. IEEE, 2024.

Lingdong Kong, Youquan Liu, Xin Li, Runnan Chen, Wenwei Zhang, Jiawei Ren, Liang Pan, Kai Chen, and Ziwei Liu. Robo3d: Towards robust and reliable 3d perception against corruptions. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 19937–19949. IEEE, 2023.

Shuangzhi Li, Zhijie Wang, Felix Juefei-Xu, Qing Guo, Xingyu Li, and Lei Ma. Common corruption robustness of point cloud detectors: Benchmark and enhancement. *IEEE Transactions on Multimedia*, 27:848–859, 2023.

Wanqing Li, Zhengyou Zhang, and Zicheng Liu. Action recognition based on a bag of 3d points. In *2010 IEEE computer society conference on computer vision and pattern recognition-workshops*, pp. 9–14. IEEE, 2010.

Jian Liang, Ran He, and Tieniu Tan. A comprehensive survey on test-time adaptation under distribution shifts. *International Journal of Computer Vision*, 133(1):31–64, 2025.

Jiuming Liu, Jinru Han, Lihao Liu, Angelica I Aviles-Rivero, Chaokang Jiang, Zhe Liu, and Hesheng Wang. Mamba4d: Efficient 4d point cloud video understanding with disentangled spatial-temporal state space models. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 17626–17636. IEEE, 2025.

Xingyu Liu, Mengyuan Yan, and Jeannette Bohg. Meteornet: Deep learning on dynamic 3d point cloud sequences. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 9245–9254. IEEE, 2019.

Yunze Liu, Yun Liu, Che Jiang, Kangbo Lyu, Weikang Wan, Hao Shen, Boqiang Liang, Zhoujie Fu, He Wang, and Li Yi. Hoi4d: A 4d egocentric dataset for category-level human-object interaction. In *CVPR*, pp. 21013–21022, 2022.

Baixuan Lv, Yaohua Zha, Tao Dai, Xue Yuerong, Ke Chen, and Shu-Tao Xia. Adapting pre-trained 3d models for point cloud video understanding via cross-frame spatio-temporal perception. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 12413–12422. IEEE, 2025.

Xinkuan Qiu, Meina Kan, Yongbin Zhou, and Shiguang Shan. Benchmarking multimodal large language models against image corruptions. In *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 9014–9023. IEEE, 2025.

Jiawei Ren, Liang Pan, and Ziwei Liu. Benchmarking and analyzing point cloud classification under corruptions. In *International conference on machine learning*, pp. 18559–18575. PMLR, 2022.

Madeline Chantry Schiappa, Naman Biyani, Prudvi Kamtam, Shruti Vyas, Hamid Palangi, Vibhav Vineet, and Yogesh S Rawat. A large-scale robustness analysis of video action recognition models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 14698–14708, 2023.

Amir Shahroudy, Jun Liu, Tian-Tsong Ng, and Gang Wang. Ntu rgb+ d: A large scale dataset for 3d human activity analysis. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1010–1019, 2016.

Hongyu Sun, Qiuhong Ke, Ming Cheng, Yongcai Wang, Deying Li, Chenhui Gou, and Jianfei Cai. Point-cache: Test-time dynamic and hierarchical cache for robust and generalizable point cloud analysis. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1263–1275. IEEE, 2025.

Mehran Tamjidi, Hamidreza Dastmalchi, Mohammadreza Alimoradijazi, Ali Cheraghian, Aijun An, and Morteza Saberi. Adapt-as-you-walk through the clouds: training-free online test-time adaptation of 3d vision-language foundation models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 9296–9304, 2026.

Dequan Wang, Evan Shelhamer, Shaoteng Liu, Bruno Olshausen, and Trevor Darrell. Tent: Fully test-time adaptation by entropy minimization. In *International Conference on Learning Representations*, 2021.

Jiang Wang, Zicheng Liu, Ying Wu, and Junsong Yuan. Mining actionlet ensemble for action recognition with depth cameras. In *2012 IEEE conference on computer vision and pattern recognition*, pp. 1290–1297. IEEE, 2012.

Jie Wang, Tingfa Xu, Lihe Ding, Xinjie Zhang, Long Bai, and Jianan Li. Pvnext: Rethinking network design and temporal motion for point cloud video recognition. In *International Conference on Learning Representations*, volume 2025, pp. 54260–54276, 2025.

Yancheng Wang, Yang Xiao, Fu Xiong, Wenxiang Jiang, Zhiguo Cao, Joey Tianyi Zhou, and Junsong Yuan. 3dv: 3d dynamic voxel for action recognition in depth video. In *CVPR*, pp. 511–520, 2020.

Hao Wen, Yunze Liu, Jingwei Huang, Bo Duan, and Li Yi. Point primitive transformer for long-term 4d point cloud video understanding. In *European Conference on Computer Vision*, pp. 19–35. Springer, 2022.

Chenyu Yi, SiYuan Yang, Haoliang Li, Yap-peng Tan, and Alex Kot. Benchmarking the robustness of spatial-temporal models against corruptions. In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks*, volume 1, 2021.

Shun-Zheng Yu. Hidden semi-markov models. *Artificial intelligence*, 174(2):215–243, 2010.

Runhao Zeng, Jiaming Liang, Jiaqi Mao, Xiaoyong Chen, Wei Wang, Yong Guo, Limin Wang, Victor Leung, and Xiping Hu. Towards robust temporal action detection: Benchmark and a strong baseline. *International Journal of Computer Vision*, 134(8):380, 2026.

Marvin Zhang, Sergey Levine, and Chelsea Finn. Memo: Test time robustness via adaptation and augmentation. *Advances in neural information processing systems*, 35:38629–38642, 2022.

Xingyu Zhu, Liang Yi, Shuo Wang, Wenbo Zhu, Yongliang Wu, Beier Zhu, and Hanwang Zhang. Adapting point cloud analysis via multimodal bayesian distribution learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9976–9985, 2026.

APPENDIX

A CORRUPTION GENERATION DETAILS

Let $\mathcal{P} = (\mathcal{P}_1, \ldots, \mathcal{P}_T)$ denote a point cloud video, where $\mathcal{P}_t = \{\mathbf{x}_{t,i}\}_{i=1}^{N_t}$ and $\mathbf{x}_{t,i} \in \mathbb{R}^3$ is a row vector. Each corruption is applied separately to the original video. For a spatial corruption, the output has the same number and order of frames as $\mathcal{P}$. For a temporal perturbation, we construct an index sequence $\boldsymbol{\iota} = (\iota_1, \ldots, \iota_{T'})$ and set $\widetilde{\mathcal{P}}_t = \mathcal{P}_{\iota_t}$. For HOI4D-P, frame labels and stored centers follow the same indices, i.e., $\widetilde{y}_t = y_{\iota_t}$ and $\widetilde{\mathbf{c}}_t = \mathbf{c}_{\iota_t}$.

A.1 SPATIAL CORRUPTIONS

Add and Drop. For each video, we sample a number of clusters K uniformly from $\{1, \ldots, 7\}$ and select their centers $\{\mathbf{a}_j\}_{j=1}^{K}$ from distinct points in the first frame. The centers remain fixed in the video's coordinate system. We randomly allocate these B points among the clusters, giving counts $n_1, \ldots, n_K$ with $\sum_j n_j = B$. For *Add*, each frame receives newly sampled points around the centers:

$$\widetilde{\mathcal{P}}_t = \mathcal{P}_t \cup \bigcup_{j=1}^{K} \{\mathbf{a}_j + \sigma_{\mathrm{add}} \boldsymbol{\epsilon}_{t,j,k} : k = 1, \ldots, n_j\}, \qquad \boldsymbol{\epsilon}_{t,j,k} \sim \mathcal{N}(\mathbf{0}, I_3). \tag{12}$$

The cluster centers and sizes are shared across frames, whereas the Gaussian offsets are sampled independently for each frame. For *Drop*, we sequentially remove the n_j points nearest to $\mathbf{a}_j$ from each frame's current point set. Let $\mathrm{NN}_n(\mathbf{a}; S)$ denote the n points in S nearest to $\mathbf{a}$. Then

$$S_t^{(0)} = \mathcal{P}_t, \qquad S_t^{(j)} = S_t^{(j-1)} \setminus \mathrm{NN}_{n_j}(\mathbf{a}_j; S_t^{(j-1)}), \qquad \widetilde{\mathcal{P}}_t = S_t^{(K)}. \tag{13}$$

The removed points may differ across frames even though the centers are fixed. Both operations change local point density without changing the temporal order.

Scale. We sample one axis $a \in \{x, y, z\}$ and one factor s for the whole video. Writing $D_a(s)$ for a diagonal matrix whose a-th entry is s and whose other entries are one, each point is scaled about its frame centroid $\boldsymbol{\mu}_t$:

$$\widetilde{\mathbf{x}}_{t,i} = \boldsymbol{\mu}_t + (\mathbf{x}_{t,i} - \boldsymbol{\mu}_t) D_a(s), \qquad \boldsymbol{\mu}_t = \frac{1}{N_t} \sum_{i=1}^{N_t} \mathbf{x}_{t,i}. \tag{14}$$

The axis and factor remain the same across frames; the centroid is computed separately for each frame.

Jitter. We add independently sampled Gaussian noise to each coordinate of every point:

$$\widetilde{\mathbf{x}}_{t,i} = \mathbf{x}_{t,i} + \sigma_{\mathrm{jit}} \boldsymbol{\epsilon}_{t,i}, \qquad \boldsymbol{\epsilon}_{t,i} \sim \mathcal{N}(\mathbf{0}, I_3). \tag{15}$$

The noise is resampled across points and frames.

Rotate. We sample a target Euler angle $\boldsymbol{\alpha} = (\alpha_x, \alpha_y, \alpha_z)$, with each component drawn uniformly from $[-\pi/6, \pi/6]$. At frame t, the angles are multiplied by $q_t = (t-1)/\max(1, T-1)$. The resulting rotation is applied about the frame centroid:

$$\widetilde{\mathbf{x}}_{t,i} = \boldsymbol{\mu}_t + (\mathbf{x}_{t,i} - \boldsymbol{\mu}_t) Q_t, \qquad Q_t = R_z(q_t \alpha_z) R_y(q_t \alpha_y) R_x(q_t \alpha_x). \tag{16}$$

Consequently, the rotation progresses from zero in the first frame to the sampled target angles in the last frame.

A.2 TEMPORAL PERTURBATIONS

Lag. We choose a freeze index f uniformly from $\{1, \ldots, T-1\}$ and let F denote the lag duration in frames. The selected frame replaces the subsequent frames within a window of length F:

$$\iota_t = \begin{cases} f, & f \le t < \min(f + F, T + 1), \\ t, & \text{otherwise}, \end{cases} \qquad t = 1, \ldots, T. \tag{17}$$

The output therefore retains T frames; the repeated frame takes the place of frames that would otherwise have appeared in the frozen interval.

Remove. We uniformly sample n_{rem} distinct frame indices to discard. The remaining indices are sorted before constructing the output, so $T' = T - n_{\mathrm{rem}}$ and $\iota_1 < \iota_2 < \cdots < \iota_{T'}$. This shortens the sequence while preserving the order of retained frames.

Shuffle. For the recognition datasets, we choose a contiguous window of length W and randomly permute the indices within that window; indices outside it remain unchanged. If the window starts at u and ω is a random permutation of its indices, the mapping is

$$\iota_t = \begin{cases} \omega(t), & u \le t < u + W, \\ t, & \text{otherwise.} \end{cases} \tag{18}$$

For HOI4D-P, we instead select non-overlapping windows within contiguous runs of identical frame labels and shuffle indices only inside each selected window. This process continues until a target number n_{shuf} of frames has been shuffled or no eligible interval remains. This restriction preserves segment labels at their original time positions while changing the ordering of observations within segments. For all temporal perturbations, the point clouds, labels, and stored centers are indexed by the same $\boldsymbol{\iota}$.

B Online Cache Construction and Update

Algorithm. 1 details the online construction and update of HSC. For each incoming observation, the frozen backbone extracts prediction scores together with global and point-tube features. The tube features are clustered into compact local descriptors, while normalized prediction entropy measures the reliability of the observation. HSC uses the predicted class to organize global and local representations into paired class-specific caches. Positive entries are associated with one-hot pseudo-labels and provide supportive class information. For observations with intermediate uncertainty, negative entries store a binary mask identifying ambiguous classes whose probabilities fall within a prescribed interval.

The shared UpdatePair procedure enforces a per-class capacity of K for every cache. Before reaching this capacity, incoming entries are appended directly. Once a cache is full, an observation replaces the entry with the highest entropy only when its own entropy is lower. The global feature and its associated tube descriptors are updated at the same cache position, preserving their correspondence. For a local entry, the one-hot value or ambiguity mask is shared by its R tube centroids. After the update, the current observation queries all four caches using the similarity-weighted function Φ from Eq. 4. Positive global and local scores provide class support, whereas their negative counterparts suppress ambiguous alternatives. Their combination produces the cache correction $\mathbf{r}_t$ without gradient updates to the backbone.

C Dynamic Programming for the Hidden Semi-Markov Model

Algorithm 2 summarizes exact inference for the explicit-duration hidden semi-Markov model in DAD. Given the adapted scores $\hat{\mathbf{z}}_t$, we compute $\ell_{t,c} = [\log \mathrm{softmax}(\hat{\mathbf{z}}_t)]_c$ and evaluate $\rho_c(d)$ from Eq. 10 for all integer durations $d = 1, \ldots, T$. All priors remain fixed during decoding.

Segment scoring. We form class-wise prefix sums $P_c(t) = \sum_{u=1}^{t} \ell_{u,c}$ with $P_c(0) = 0$. A segment of class c ending at frame t with duration d covers frames $t-d+1, \ldots, t$. Its combined frame and duration score is

$$E(t, c, d) = P_c(t) - P_c(t-d) + \lambda_D \rho_c(d). \tag{19}$$

Thus, each interval score is evaluated in constant time after the prefix sums and duration scores have been computed.

Dynamic programming. Let $V(t, c)$ be the maximum score of a segmentation covering frames $1, \ldots, t$ and ending with a segment of class c. To separate the predecessor search from the duration search, define $B(q, c)$ as the best prefix score, including the initial or transition contribution for a new segment of class c:

$$B(q, c) = \begin{cases} \lambda_T \log \pi_c, & q = 0, \\ \max\limits_{a \neq c} \left[V(q, a) + \lambda_T \log A_{a,c} \right], & q \ge 1. \end{cases} \tag{20}$$

Algorithm 1 Online Construction and Update of HSC

Require: Test stream $\{\widetilde{\mathcal{P}}_t\}$; frozen backbone f_θ; number of classes C; number of tube clusters R; per-class cache capacity K; uncertainty thresholds $\tau_{\text{low}}, \tau_{\text{high}}$; probability thresholds $\delta_{\text{low}}, \delta_{\text{high}}$.
Ensure: Cache Adaptations $\mathbf{r}_t$

Initialize $\mathcal{C}_g^+, \mathcal{C}_l^+, \mathcal{C}_g^-, \mathcal{C}_l^- \leftarrow \varnothing$
for each observation $\widetilde{\mathcal{P}}_t$ **do**

  Encode and summarize the incoming observation.
  Encode the observation: $(\mathbf{z}_t, \mathbf{h}_t, \mathbf{L}_t) \leftarrow f_\theta(\widetilde{\mathcal{P}}_t)$.
  Cluster tube features: $\widehat{\mathbf{L}}_t \leftarrow \mathrm{KMeans}(\mathbf{L}_t, R)$.
  Compute probabilities and pseudo-label: $\mathbf{p}_t \leftarrow \mathrm{softmax}(\mathbf{z}_t)$, $\hat{y}_t \leftarrow \arg\max_c p_{t,c}$.
  Compute normalized entropy: $u_t \leftarrow -(\log C)^{-1} \sum_{c=1}^{C} p_{t,c} \log p_{t,c}$.

  Update the positive cache pair.
  $\mathbf{v}_t^+ \leftarrow \mathrm{onehot}(\hat{y}_t, C)$
  UPDATEPAIR($\mathcal{C}_g^+, \mathcal{C}_l^+, \hat{y}_t, \mathbf{h}_t, \widehat{\mathbf{L}}_t, \mathbf{v}_t^+, u_t, K$)

  Update the negative cache pair.
  if $\tau_{\text{low}} < u_t < \tau_{\text{high}}$ **then**
    $\mathbf{v}_t^- \leftarrow (\mathbb{I}[\delta_{\text{low}} < p_{t,c} < \delta_{\text{high}}])_{c=1}^{C}$.
    UPDATEPAIR($\mathcal{C}_g^-, \mathcal{C}_l^-, \hat{y}_t, \mathbf{h}_t, \widehat{\mathbf{L}}_t, \mathbf{v}_t^-, u_t, K$)
  end if

  Compute the cache correction.
  $\mathbf{r}_t = \Phi(\mathbf{h}_t; \mathcal{C}_g^+) + \Phi(\widehat{\mathbf{L}}_t; \mathcal{C}_l^+) - \Phi(\mathbf{h}_t; \mathcal{C}_g^-) - \Phi(\widehat{\mathbf{L}}_t; \mathcal{C}_l^-)$
end for
return $\mathbf{r}_t$

procedure UPDATEPAIR($\mathcal{C}_g, \mathcal{C}_l, c, \mathbf{h}, \widehat{\mathbf{L}}, \mathbf{v}, u, K$)
  if $|\mathcal{C}_{g,c}| < K$ **then**
    Insert $(\mathbf{h}, \mathbf{v}, u)$ into $\mathcal{C}_{g,c}$
    Insert $(\widehat{\mathbf{L}}, \mathbf{v}, u)$ into $\mathcal{C}_{l,c}$
  else
    $j^\star \leftarrow \arg\max_{j \in \mathcal{C}_{g,c}} u_j$
    if $u < u_{j^\star}$ **then**
      Replace entry $j^\star$ in $\mathcal{C}_{g,c}$ with $(\mathbf{h}, \mathbf{v}, u)$
      Replace entry $j^\star$ in $\mathcal{C}_{l,c}$ with $(\widehat{\mathbf{L}}, \mathbf{v}, u)$
    end if
  end if
end procedure

The first case applies when the new segment starts at the first frame. Otherwise, the maximization considers all preceding classes except c, matching the constraint that adjacent segments have different labels. Persistence within a class is represented by its segment duration.

The optimal prefix score is obtained by considering every possible duration of its final segment:

$$V(t,c) = \max_{1 \le d \le t} \left[E(t,c,d) + B(t-d,c) \right], \qquad t = 1, \ldots, T. \tag{21}$$

At each frame, the algorithm computes $V(t,c)$ for all classes before updating $B(t,c)$. It stores the maximizing durations $d^\star(t,c)$ and predecessor classes $a^\star(t,c)$ to recover the optimal segmentation by backtracking.

Every valid segmentation consists of an optimal prefix followed by a final segment. Since the recurrence evaluates all admissible durations and preceding classes, it returns the exact maximum of Eq. 11. Precomputing the predecessor scores requires $O(TC^2)$ operations, while evaluating all

Algorithm 2 Dynamic Programming for the Hidden Semi-Markov Model

Require: Adapted scores $\{\hat{\mathbf{z}}_t\}_{t=1}^{T}$; priors π, A, and $\{(\mu_c, \sigma_c)\}_{c=1}^{C}$; weights λ_T, λ_D
Ensure: Optimal segmentation $Y^\star$ and frame labels $\{\hat{y}_t\}_{t=1}^{T}$
Compute frame scores $\ell_{t,c}$, prefix sums $P_c(t)$, and duration scores $\rho_c(d)$.
Initialize $B(0, c) \leftarrow \lambda_T \log \pi_c$ for every class c.

Compute optimal prefix scores.
for $t = 1, \ldots, T$ **do**
    for $c = 1, \ldots, C$ **do**
        Select duration $d^\star(t, c) \leftarrow \arg\max_{1 \le d \le t} [E(t, c, d) + B(t - d, c)]$.
        Set $d \leftarrow d^\star(t, c)$ and $V(t, c) \leftarrow E(t, c, d) + B(t - d, c)$.
    end for
    for $c = 1, \ldots, C$ **do**
        Select predecessor $a^\star(t, c) \leftarrow \arg\max_{a \neq c} [V(t, a) + \lambda_T \log A_{a,c}]$.
        Set $a \leftarrow a^\star(t, c)$ and $B(t, c) \leftarrow V(t, a) + \lambda_T \log A_{a,c}$.
    end for
end for

Recover segments and frame labels.
Initialize $t \leftarrow T$, $c \leftarrow \arg\max_a V(T, a)$, and $Y^\star$ as an empty list.
while $t > 0$ **do**
    Recover $d \leftarrow d^\star(t, c)$ and set $q \leftarrow t - d$.
    Assign $\hat{y}_u \leftarrow c$ for $u = q + 1, \ldots, t$.
    Prepend $(c, d, \{q + 1, \ldots, t\})$ to $Y^\star$.
    if $q > 0$ **then**
        Recover the preceding class $c \leftarrow a^\star(q, c)$.
    end if
    Move to the preceding prefix: $t \leftarrow q$.
end while
return $Y^\star, \{\hat{y}_t\}_{t=1}^{T}$

durations requires $O(T^2C)$, giving a total time complexity of $O(T^2C + TC^2)$. The score and backpointer tables require $O(TC)$ memory, in addition to the $O(C^2)$ transition matrix.

D Additional Implementation Details

Evaluation protocol. All methods are evaluated using their pretrained checkpoints, with the backbone set to evaluation mode and its parameters kept frozen. STEADY operates entirely at inference time under `no_grad` and does not perform gradient-based optimization. Each corrupted subset is evaluated independently, and its cache is initialized as empty before evaluation. For recognition, the cache is updated sequentially over the test stream. For framewise segmentation, the cache is reset at the beginning of each video and updated online as its frames are processed. The corrupted test labels are used only for evaluation and are never accessed by STEADY.

Hierarchical Spatial Cache. The point-tube features of each observation are summarized into $R = 3$ clusters using K-means. The positive and negative caches retain at most $K^+ = 3$ and $K^- = 2$ entries per predicted class, respectively. Cache entries are ranked by normalized prediction entropy, and a full cache replaces its most uncertain entry only when the incoming observation has lower entropy. The global feature and clustered tube descriptors derived from the same observation are always inserted or replaced together. For recognition, the positive-cache retrieval parameters in Eq. 4 are set to $\alpha^+ = 2$ and $\beta^+ = 3$. For segmentation, they are set to $\alpha^+ = 4$ and $\beta^+ = 5$. The negative-cache parameters are fixed to $\alpha^- = 0.117$ and $\beta^- = 1$ for both tasks. Negative entries are constructed when the normalized entropy lies in $(0.2, 0.5)$, and their class-wise ambiguity masks retain probabilities within $(0.03, 1.0)$. These settings are fixed across all backbones evaluated for the same task.

Table 5: Recognition performance on MSRAction3D-C and NTU60-C. Unstarred clean results are reproduced using official implementations, while * denotes clean accuracy reported in the original papers. For starred rows, corruption results, OA, and mCE remain identical to the corresponding unstarred rows, while mRCE is recomputed using the reported clean accuracy. STEADY results that outperform both backbone rows are shown in **bold**.

Method	Clean	Corruption Type								OA↑	mCE↓	mRCE↓
		Add	Drop	Scale	Jitter	Rotate	Lag	Remove	Shuffle			
MSRAction3D-C												
P4Transformer (Fan et al., 2021a)	90.9	90.1	79.8	69.7	90.2	83.8	86.2	82.2	87.5	83.7	1.00	1.00
P4Transformer*	90.9	90.1	79.8	69.7	90.2	83.8	86.2	82.2	87.5	83.7	1.00	1.00
+ STEADY (Ours)	**91.6**	89.9	**82.5**	**71.7**	**92.3**	**86.5**	**87.9**	**83.5**	**87.9**	**85.3**	**0.90**	**0.80**
PPTr (Wen et al., 2022)	91.5	84.5	75.8	64.3	88.2	70.4	85.2	85.2	84.9	79.8	1.26	2.89
PPTr*	92.3	84.5	75.8	64.3	88.2	70.4	85.2	85.2	84.9	79.8	1.26	3.25
+ STEADY (Ours)	91.6	**84.9**	**77.4**	62.6	**89.6**	**72.7**	**86.2**	84.9	**85.2**	**80.4**	**1.21**	**2.54**
PST-Transformer (Fan et al., 2022)	93.7	90.2	74.4	70.4	93.6	84.2	90.2	89.9	84.9	84.7	0.92	1.56
PST-Transformer*	93.0	90.2	74.4	70.4	93.6	84.2	90.2	89.9	84.9	84.7	0.92	1.24
+ STEADY (Ours)	**94.0**	90.2	**78.5**	**71.0**	**93.9**	**84.5**	**90.6**	87.5	**86.2**	**85.3**	**0.88**	1.56
Mamba4D (Liu et al., 2025)	91.8	87.9	84.9	62.6	87.9	78.1	85.9	85.5	86.5	82.4	1.09	2.24
Mamba4D*	93.4	87.9	84.9	62.6	87.9	78.1	85.9	85.5	86.5	82.4	1.09	2.95
+ STEADY (Ours)	92.1	**90.6**	**86.5**	58.3	**89.6**	**79.5**	**86.5**	**86.5**	**87.2**	**83.1**	**1.01**	**1.57**
MotionPDE (Huang et al., 2026)	96.4	94.3	86.9	77.4	94.6	88.6	91.3	86.2	92.6	89.0	0.65	1.43
MotionPDE*	96.6	94.3	86.9	77.4	94.6	88.6	91.3	86.2	92.6	89.0	0.65	1.52
+ STEADY (Ours)	96.3	**94.6**	86.2	**79.5**	94.3	**88.8**	**91.9**	**87.2**	**93.3**	**89.5**	**0.63**	**1.33**
NTU60-C												
P4Transformer (Fan et al., 2021a)	89.0	79.7	52.5	77.0	86.9	81.1	86.6	85.1	77.6	78.3	1.00	1.00
P4Transformer*	90.2	79.7	52.5	77.0	86.9	81.1	86.6	85.1	77.6	78.3	1.00	1.24
+ STEADY (Ours)	89.7	**80.4**	**53.9**	77.0	**87.2**	**81.2**	86.4	84.6	**81.7**	**79.1**	**0.97**	1.09
PST-Transformer (Fan et al., 2022)	91.0	85.0	56.2	83.3	87.6	82.4	88.3	86.3	79.2	81.0	0.87	1.04
PST-Transformer*	91.0	85.0	56.2	83.3	87.6	82.4	88.3	86.3	79.2	81.0	0.87	1.04
+ STEADY (Ours)	90.9	**85.7**	**57.2**	**83.4**	**88.1**	**83.2**	88.0	86.0	**81.5**	**81.6**	**0.85**	**0.96**
MotionPDE (Huang et al., 2026)	92.1	85.1	56.0	80.0	86.5	83.5	88.0	84.1	76.7	80.0	0.93	1.45
MotionPDE*	92.4	85.1	56.0	80.0	86.5	83.5	88.0	84.1	76.7	80.0	0.93	1.51
+ STEADY (Ours)	92.2	**85.5**	**57.4**	**80.1**	**87.8**	**83.9**	**88.1**	82.9	**79.0**	**80.6**	**0.90**	**1.38**

Soft Temporal Consistency Gate. STCG assigns equal contributions to the feature-space and prediction-space consistency scores. Their arithmetic mean is clipped to $[G_{\min}, 1]$, where $G_{\min} = 0.05$. For the first frame of each video, the gate is set to $G_1 = 1$ because no preceding observation is available. All subsequent gates are computed online using only the current and previous frames.

Dynamics-Aware Decoder. DAD is applied to framewise segmentation predictions with $\lambda_T = 5$ and $\lambda_D = 1$ in Eq. 11. The initial-state, transition, and duration priors are estimated once from the training annotations and remain fixed during evaluation. No annotations from corrupted test videos are involved in their estimation. The adapted logits are converted into log probabilities before decoding, and the optimal segment sequence is obtained using the hidden semi-Markov dynamic programming procedure described in Appendix C. Candidate durations are positive integers bounded by the video length, and consecutive decoded segments are constrained to have different class labels.

E Additional Experimental Results

E.1 Complementary Recognition Results

Table 5 complements the main recognition results by comparing our reproduced clean-data performance with the values reported in the original papers. The unstarred rows are obtained using the official implementations, whereas the starred rows replace only the clean accuracy with the originally reported value while retaining the same corruption results. Consequently, OA and mCE remain unchanged, while mRCE is recomputed according to the substituted clean accuracy.

The comparison shows that mRCE is sensitive to the choice of clean reference. A higher clean accuracy generally enlarges the relative performance degradation under corruption and therefore increases mRCE, whereas a lower clean accuracy can reduce it. Nevertheless, STEADY achieves

Table 6: Segmentation performance on HOI4D-P measured by the segmental F1 score at $\tau = 0.25$. STEADY results that outperform the corresponding backbone are shown in **bold**.

Method	Clean	Corruption Type								OF↑	mCE↓	mRCE↓
		Add	Drop	Scale	Jitter	Rotate	Lag	Remove	Shuffle			
P4Transformer (Fan et al., 2021a)	69.2	59.2	62.9	47.4	62.0	59.3	56.3	22.3	60.5	53.7	1.00	1.00
+ STEADY (Ours)	**73.7**	**70.8**	**66.5**	**60.1**	**73.2**	**68.7**	**63.8**	20.8	**73.2**	**62.1**	**0.80**	**0.57**
PPTr (Wen et al., 2022)	78.5	73.8	73.0	59.7	74.4	67.9	65.6	28.6	68.4	63.9	0.76	0.88
+ STEADY (Ours)	**82.5**	**81.1**	**77.5**	**69.7**	**82.4**	**76.2**	**72.4**	25.6	**81.4**	**70.8**	**0.59**	**0.54**
X4D-SceneFormer (Jing et al., 2024)	90.8	90.3	87.8	83.7	74.8	88.2	74.5	48.2	77.8	78.2	0.46	0.87
+ STEADY (Ours)	**92.8**	**92.1**	**89.2**	**87.9**	**81.7**	**90.9**	**80.1**	**51.5**	**85.5**	**82.4**	**0.36**	**0.66**
Mamba4D (Liu et al., 2025)	91.2	91.9	89.1	87.4	80.9	89.8	74.8	59.7	77.7	81.4	0.39	0.69
+ STEADY (Ours)	**93.8**	**93.4**	**91.4**	**91.0**	**86.6**	**92.2**	**80.2**	**64.5**	**86.2**	**85.7**	**0.30**	**0.53**

Table 7: Segmentation performance on HOI4D-P measured by frame accuracy. STEADY results that outperform the corresponding backbone are shown in **bold**.

Method	Clean	Corruption Type								OA↑	mCE↓	mRCE↓
		Add	Drop	Scale	Jitter	Rotate	Lag	Remove	Shuffle			
P4Transformer (Fan et al., 2021a)	71.2	65.7	65.5	55.3	69.0	65.0	58.3	27.3	69.4	59.4	1.00	1.00
+ STEADY (Ours)	64.4	61.2	57.5	51.9	63.8	59.6	54.1	24.5	64.3	54.6	1.13	**0.67**
PPTr (Wen et al., 2022)	77.4	75.8	74.8	64.1	77.2	70.9	65.2	33.1	76.1	67.2	0.79	0.67
+ STEADY (Ours)	73.1	71.2	67.8	60.3	72.7	66.3	62.0	27.4	72.4	62.5	0.92	0.70
X4D-SceneFormer (Jing et al., 2024)	84.1	84.5	81.9	79.3	71.0	83.0	68.8	46.4	75.8	73.8	0.64	1.67
+ STEADY (Ours)	**85.3**	84.2	80.5	79.2	**71.3**	82.9	68.8	45.6	**76.4**	73.6	0.65	1.91
Mamba4D (Liu et al., 2025)	85.5	85.5	83.2	82.4	77.0	84.2	70.0	55.7	76.3	76.8	0.58	1.46
+ STEADY (Ours)	**86.2**	**85.6**	**83.4**	**82.8**	**77.7**	**84.4**	**70.2**	**56.6**	**77.4**	**77.3**	**0.56**	1.46

favorable mRCE results for most backbones under both clean references. Together with the unchanged improvements in OA and mCE, these results show that the overall robustness conclusions are not determined by differences in clean-data reproduction.

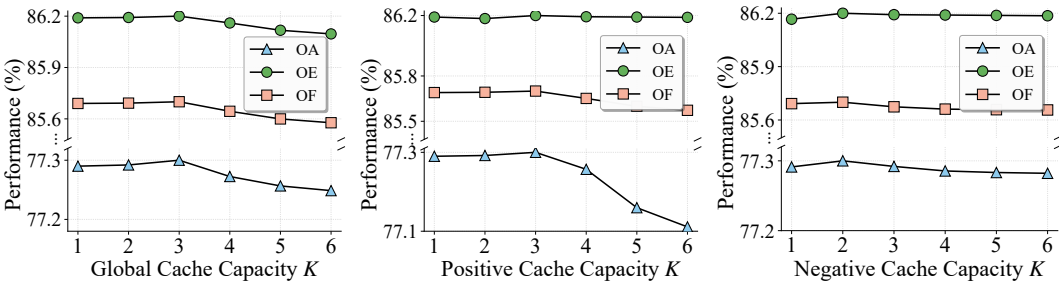
E.2 Complementary Segmentation Results

Table 6 reports the segmental F1 score at an overlap threshold of $\tau = 0.25$. STEADY consistently improves segment-level robustness across all four backbones. These results indicate that the adapted predictions and dynamics-aware decoding reduce prediction fragmentation and recover more coherent action boundaries.

Table 7 further presents frame-level accuracy. Unlike the consistent improvements in segmental metrics, the effects on frame accuracy vary across backbones. STEADY remains comparable to X4D-SceneFormer and slightly improves Mamba4D, while decreasing the frame accuracy of P4Transformer and PPTr. This difference suggests that STEADY primarily improves sequence-level temporal structure rather than uniformly optimizing individual frame predictions. Overall, the complementary metrics show that its main benefit lies in producing coherent action segments under corruption.

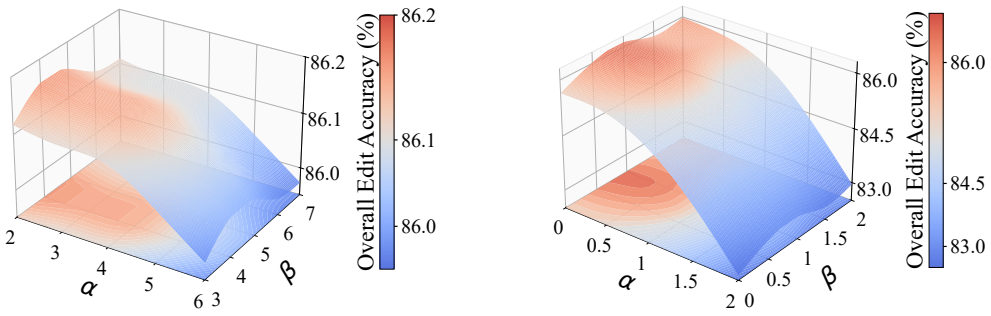
E.3 Additional Sensitivity Analysis of HSC

Cache capacity. Figure 6 examines the effects of the global, positive, and negative cache capacities on OA, OE, and OF. Performance remains stable across moderate capacity changes, demonstrating that HSC is not sensitive to the exact cache size. The global and positive caches achieve their best or near-best performance with a capacity of 3, whereas the negative cache performs best with a capacity of 2. Increasing the capacities further provides no additional benefit and can slightly reduce performance, potentially because larger caches retain outdated or less reliable pseudo-labeled observations.



(a) Global cache capacity. (b) Positive cache capacity. (c) Negative cache capacity.

Figure 6: Effect of cache capacities across different dimensions.



(a) Sensitivity to the positive-cache parameters α and β.

(b) Sensitivity to the negative-cache parameters α and β.

Figure 7: Sensitivity analysis of the cache-retrieval parameters for positive and negative caches.

Cache retrieval parameters. Figure 7 further studies the kernel parameters α and β, which control the magnitude and concentration of cache retrieval scores, respectively. The positive cache maintains stable performance across a broad parameter range, indicating limited sensitivity to moderate variations in retrieval strength. The negative cache is relatively more sensitive, particularly when α becomes large, as excessive negative corrections may suppress plausible classes. Nevertheless, the adopted settings lie within stable and high-performing regions for both caches, confirming that the effectiveness of HSC does not depend on narrowly tuned parameters.

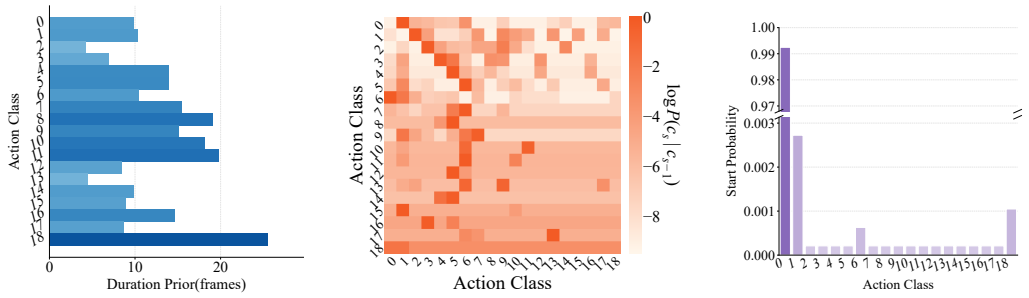
E.4 Visualization of Dynamics Priors

Figure 8 visualizes the duration, transition, and start priors used by the Dynamics-Aware Decoder. These priors are estimated once from the training annotations and remain fixed throughout test-time decoding. Figure 8a shows that action classes exhibit substantially different duration patterns, supporting the use of class-specific duration modeling. Figure 8b presents a non-uniform transition structure, indicating that only certain action transitions frequently occur in the training sequences.

Together, these priors provide complementary information about how actions begin, transition, and persist over time. The decoder combines them as soft structural guidance with the adapted frame scores, rather than imposing fixed action sequences. This allows STEADY to suppress implausible state changes and fragmented predictions while preserving observation-dependent decisions under corruption.

E.5 Additional Qualitative Results

Figure 9 presents additional qualitative results for action recognition under corrupted inputs. MotionPDE produces ambiguous class distributions and assigns the highest probabilities to incorrect actions. In particular, it confuses *Draw circle* with *Forward kick*, *Draw cross* with *High arm wave*,



(a) Class specific duration priors. (b) Transition log prior matrix. (c) Class specific start priors.

Figure 8: Visualization of the dynamics priors used by DAD: class specific duration priors, transition log priors between action classes, and start probabilities. These priors are estimated from training annotations and remain fixed during test time decoding.

and *High throw* with *Forward kick*. These examples illustrate the difficulty of distinguishing actions with degraded geometric observations and similar motion patterns.

In contrast, STEADY increases the probabilities of the corresponding ground-truth classes and promotes them to the top-ranked predictions in all three examples. The refined distributions indicate that STEADY can better distinguish ambiguous actions by exploiting complementary geometric cues and temporal consistency at test time.

Figure 10 further presents qualitative results for temporal action segmentation. The baseline predictions exhibit fragmented segments, incorrect action states, and temporally misaligned boundaries. In contrast, STEADY produces predictions that more closely follow the ground truth action progression, with fewer spurious transitions and more coherent temporal segments. These results qualitatively demonstrate the benefit of combining test-time geometric adaptation with dynamics-aware decoding for recovering temporal structures under corruption.

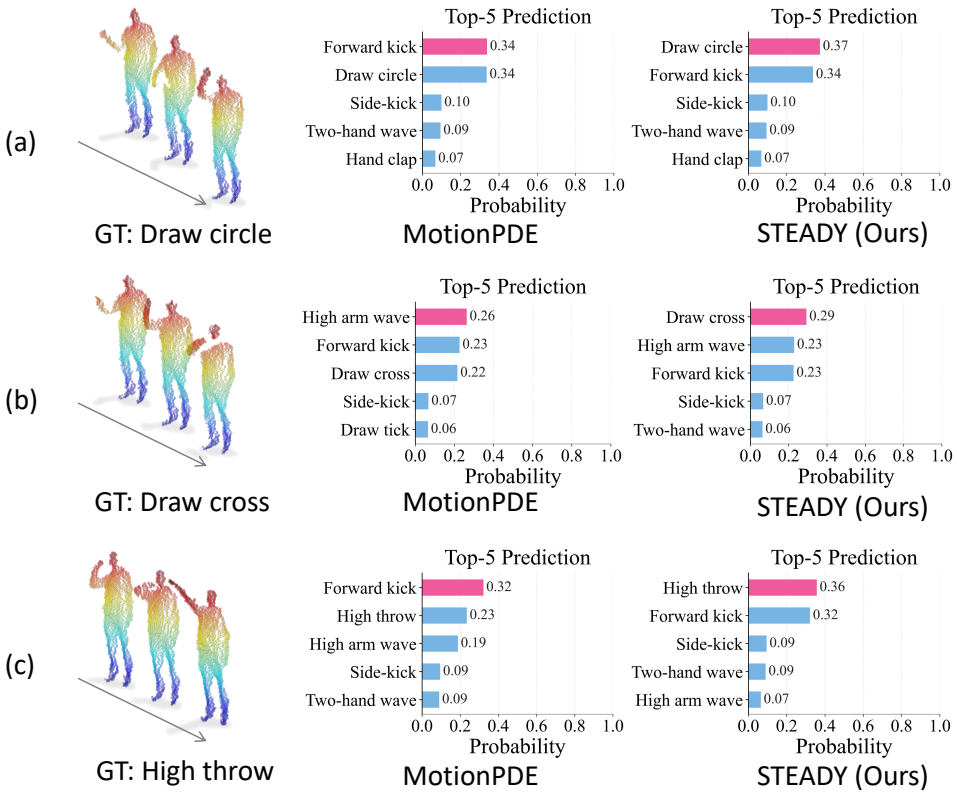


Figure 9: Additional qualitative results for action recognition. STEADY corrects ambiguous predictions under corrupted inputs and assigns higher probabilities to the ground-truth actions.

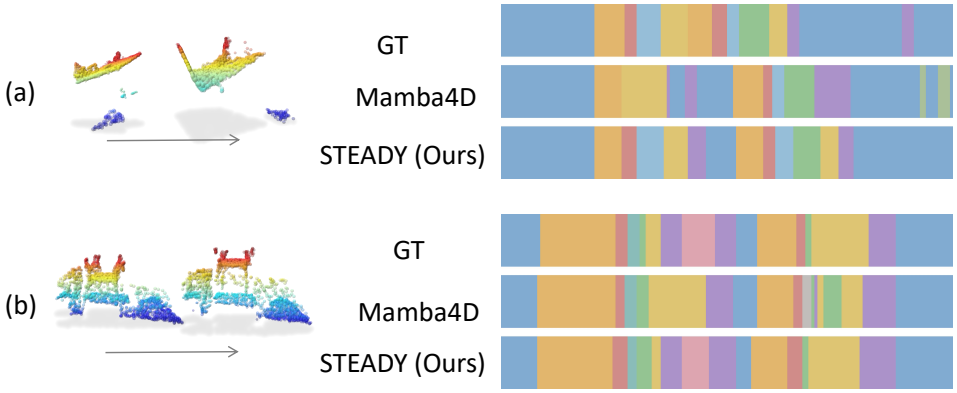


Figure 10: Additional qualitative results for temporal action segmentation. Compared with the corresponding backbone, STEADY reduces fragmented predictions and recovers more coherent action sequences under corrupted inputs.