DANCEMAR: LOCAL GENERATION OVER RETRIEVED MOTION PRIORS FOR FULL-BODY DANCE SYNTHESIS

Anonymous authors
Paper under double-blind review

ABSTRACT

Music-driven full-body dance synthesis requires realistic body and hand motion that follows the music and remains temporally coherent. Current methods primarily generate complete sequences or reuse retrieved examples, with less emphasis on local adaptation that preserves retrieved motion context. In this paper, we introduce **DanceMAR**, a framework that adapts retrieved motion priors through music-conditioned local residual generation. Specifically, we represent retrieved motion with a continuous hierarchy, retaining the lower-resolution context while restricting updates to local regions of the finest body-motion stream. To improve temporal coherence, we train a bidirectional masked model with local perturbations and motion-space supervision to learn residual corrections from the music and surrounding motion. In addition, we combine the adapted body with hand articulation from the same retrieved example to obtain full-body motion. Extensive experiments on FineDance demonstrate that DanceMAR achieves state-of-the-art body and hand motion fidelity while maintaining competitive beat alignment and foot stability. Further analyses show that motion changes remain localized, with smooth transitions between the adapted regions and the surrounding motion.

1 INTRODUCTION

With the development of digital humans and virtual performance, music-driven 3D dance generation has become an important task in multimedia content creation (Huang et al., 2021; Li et al., 2021). Given a music track, the goal is to synthesize body motion and hand articulation that follow its rhythm, dynamics, and style. A realistic full-body dance also requires coherent changes in pose and smooth transitions between movements.

Earlier studies model music-to-dance generation as sequence prediction, learning temporal dependencies from paired music and motion (Huang et al., 2021; Li et al., 2021). Bailando uses learned choreographic tokens for dance generation (Siyao et al., 2022). More recently, EDGE supports music-conditioned generation and motion editing through diffusion (Tseng et al., 2023), while LODGE adopts a coarse-to-fine framework for long dance sequences (Li et al., 2024a). FineDance introduces a benchmark and generation method that include detailed body and hand motion (Li et al., 2023). Recorded motion also provides reusable examples: example-based systems select and compose motion segments (Fan et al., 2012), while retrieval-guided motion models use relevant examples to condition generation (Zhang et al., 2023b).

Although these methods have advanced dance generation, local adaptation that preserves the retrieved motion context remains less explored. A recorded dance already provides a sequence of body movements and hand gestures. Direct reuse retains this sequence but offers limited adaptation to the target music, while using it only as a condition allows the generator to change both local movement and the surrounding context. As illustrated in Fig. 1, the challenge is to adapt the retrieved motion to the target music while retaining its surrounding context. Each local update also needs to remain consistent with the poses and dynamics on both sides, so that adaptation does not introduce abrupt transitions.

To address this problem, we introduce **DanceMAR**, a framework for local residual generation over retrieved motion priors. Each retrieved body–hand example is represented by a continuous motion hierarchy. We retain the lower-resolution body streams and allow updates only within a contiguous region of the finest stream. A bidirectional masked model predicts residuals from the target music

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.

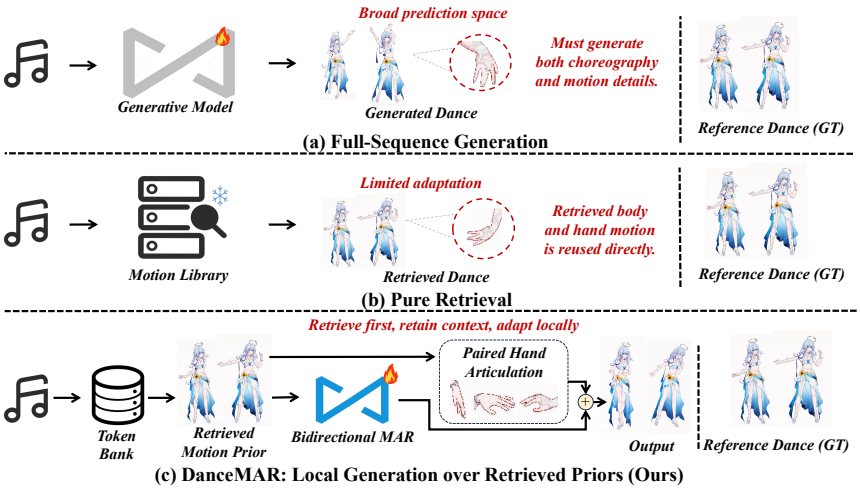

Figure 1: **Comparison of three paradigms for music-driven dance generation.** (a) Full-space generation synthesizes the entire dance from a broad motion space, requiring the model to determine both plausible choreography and detailed motion. (b) Pure retrieval directly reuses a realistic motion exemplar but provides limited adaptation to the target music. (c) DanceMAR retrieves a motion prior, retains its motion context, and predicts local residual updates in the finest stream. Boundary weighting connects the adapted region to the surrounding motion.

and the surrounding motion. The selected interval limits where updates are permitted, while the predicted residual determines the change at each position. To support local restoration and temporal coherence, we train the model with masking and local perturbations, together with joint position, velocity, and acceleration supervision. These objectives guide the decoded motion toward the target poses and dynamics, including the transitions between adapted and retained regions. The residual is added to the retrieved fine stream and gradually reduced near the interval boundaries to connect the updated tokens to the retained context. In addition, we obtain hand articulation from the same retrieved example and combine it with the adapted body. The resulting motion windows are assembled into a full-song sequence. Extensive experiments on FineDance (Li et al., 2023) demonstrate that DanceMAR achieves state-of-the-art body and hand motion fidelity while maintaining competitive beat alignment and foot stability. Further analyses examine the effects of music conditioning, adaptation scope, and boundary weighting on motion quality and temporal continuity. Our main contributions can be summarized as follows:

- We formulate music-driven full-body dance synthesis as local adaptation over retrieved motion priors, retaining the retrieved context in the output and restricting generation to music-conditioned local updates.
- We propose **DanceMAR**, which predicts fine-level residuals from music and bidirectional motion context. Training with masking, local perturbations, and motion-space supervision supports local restoration and temporally coherent adaptation.
- Extensive quantitative and qualitative evaluations on FineDance show improved body and hand motion fidelity with competitive beat alignment and foot stability. Additional analyses examine how adaptation scope and boundary weighting affect locality and continuity.

2 Related Work

2.1 Music-Driven 3D Dance Generation

Early neural methods formulate music-to-dance synthesis as sequence prediction, using curriculum learning, large paired datasets, and autoregressive motion modeling to improve realism and long-term coherence (Huang et al., 2021; Li et al., 2021; Siyao et al., 2022). More recent diffusion-based methods provide stronger sequence modeling and flexible conditioning. EDGE supports music-

conditioned generation and motion editing (Tseng et al., 2023); FineDance introduces a fine-grained full-body benchmark and models body and hand motion separately (Li et al., 2023); and LODGE adopts a coarse-to-fine framework for long-sequence synthesis (Li et al., 2024a). FODGE further improves generated motion through full-body optimization (Huang et al., 2026). DanceMosaic leverages a pretrained generative masked motion prior and introduces music- and pose-guidance branches to support high-fidelity dance synthesis and semantic motion editing (Shah et al., 2026). DanceMAR starts from a retrieved motion prior and learns music-conditioned local residual updates, while retaining the retrieved context in the output representation.

2.2 Motion Representations and Retrieval-Guided Generation

Compact motion representations reduce the difficulty of modeling high-dimensional human motion. Vector-quantized autoencoders (van den Oord et al., 2017) provide discrete motion vocabularies used by T2M-GPT, MotionGPT, and Bailando (Zhang et al., 2023a; Jiang et al., 2023; Siyao et al., 2022), while MoMask combines hierarchical quantization with bidirectional masked modeling (Guo et al., 2024). Continuous alternatives include direct motion diffusion and latent diffusion (Tevet et al., 2023; Chen et al., 2023). DanceMAR uses a continuous hierarchy in which the lower-resolution streams provide retained motion context, while the finest stream supports local residual prediction.

Recorded motion has also long been used as a source of examples and constraints. Motion graphs and example-based dance synthesis search and compose motion segments from a database (Kovar et al., 2002; Fan et al., 2012). You Never Stop Dancing constrains generation with a motion bank (Sun et al., 2022), while ReMoDiffuse retrieves relevant motions to guide text-to-motion diffusion (Zhang et al., 2023b). In these approaches, retrieved motion commonly serves as a candidate, constraint, initialization, or auxiliary reference. DanceMAR retains the retrieved context in the output and learns residual updates within a selected fine-level interval.

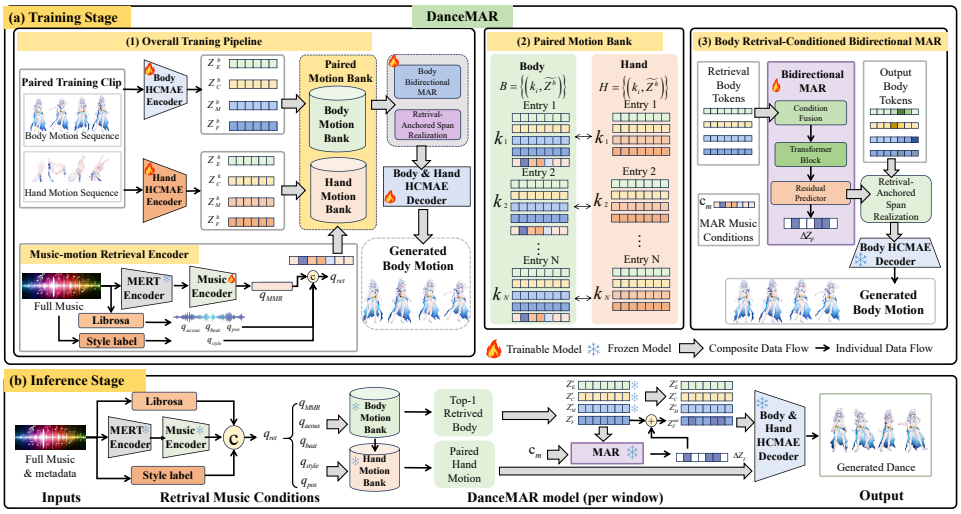

Figure 2: **DanceMAR framework.** We train the body and hand autoencoders, music–motion retrieval encoder, and masked residual model. At inference, music retrieves a paired body–hand prior. The masked model predicts local fine-token residuals for the body, followed by boundary weighting, decoding, and window assembly. Hand articulation from the same retrieved example is attached to the wrists of the final body trajectory.

3 Method

Figure 2 presents DanceMAR. Music features guide paired body–hand retrieval from a fixed training bank and condition local residual prediction. The model retains the lower-resolution body streams and updates a selected region of the finest stream. The adapted body and paired hand motion are assembled across windows into a full-song dance. The motion autoencoders and music–motion retrieval encoder are pretrained and frozen before residual-model training.

3.1 Motion Representation and Music–Motion Alignment

Hierarchical motion representation. We use separate Hierarchical Continuous Motion Autoencoders (HCMAEs) for body and hand motion, with the same design and independent parameters. For branch $s \in \{b, h\}$,

$$\mathcal{Z}^s = E^s(\mathbf{x}^s) = \{\mathbf{Z}_E^s, \mathbf{Z}_C^s, \mathbf{Z}_M^s, \mathbf{Z}_F^s\}, \qquad \mathbf{x}_{\text{rec}}^s = D^s(\mathcal{Z}^s). \tag{1}$$

The streams have strides 32, 32, 8, and 4, giving 10, 10, 38, and 75 continuous tokens for a 300-frame window. They are constructed progressively: the middle stream receives the lower-resolution features, and the fine stream also receives the middle-level features. The decoder resamples and combines all four streams. The autoencoders are pretrained with full-motion and multi-resolution reconstruction objectives. During body adaptation, $\mathbf{Z}_E^b$, $\mathbf{Z}_C^b$, and $\mathbf{Z}_M^b$ remain fixed, while $\mathbf{Z}_F^b$ supports local updates. The hierarchy provides retained context and a local adaptation interface within a shared decoder. Body and hand hierarchies from the same window retain a common source index. Architecture details are in Appendix A.

Music feature extraction. We extract two time-aligned sequences from each music window. We average the final four hidden-layer outputs of frozen MERT-v1-330M (Li et al., 2024b) to obtain 1024-dimensional features. After alignment to 30 FPS, we standardize the features, retain 256 whitened PCA components, and standardize again using training-fitted transforms. The resulting sequence is denoted by $\mathbf{H}_m$ (MERT256). In parallel, librosa extracts a 35-dimensional acoustic sequence (Music35), containing onset, MFCC, chroma, onset-peak, and beat features. We use this sequence to construct the acoustic descriptor $\mathbf{q}_{\text{acous}}$ and beat descriptor $\mathbf{q}_{\text{beat}}$. Relative window position and the provided song-level style labels supply $\mathbf{q}_{\text{pos}}$ and $\mathbf{q}_{\text{style}}$, respectively. Extraction details are given in Appendix B.1.

Music–motion alignment. The music–motion retrieval encoder (MMR) learns a shared space from paired MERT256 and 268-dimensional body-motion features. Both branches produce 256-dimensional variational representations, and a shared motion decoder reconstructs paired body motion from either modality. Training combines reconstruction, latent-mean alignment, KL regularization, and symmetric contrastive learning. The frozen music branch supplies the window descriptor $\mathbf{q}_{\text{MMR}} = E_{\text{MMR}}^m(\mathbf{H}_m)$, using its latent mean rather than a stochastic sample. Unlike the frame-level MERT256 input, this descriptor summarizes the complete window. MMR architecture and objectives are given in Appendix B.2.

3.2 Music-Conditioned Paired Motion Bank

Query construction. For music $\mathbf{m}$, style labels $\mathbf{y}$, and relative position τ, we construct

$$\mathbf{q}_{\text{ret}} = \mathcal{Q}(\mathbf{m}, \mathbf{y}, \tau) = \text{Std}([\mathbf{q}_{\text{MMR}} \| \mathbf{q}_{\text{acous}} \| \mathbf{q}_{\text{beat}} \| \mathbf{q}_{\text{style}} \| \mathbf{q}_{\text{pos}}]). \tag{2}$$

Here, $\|$ denotes feature concatenation. The acoustic descriptor $\mathbf{q}_{\text{acous}}$ concatenates the temporal mean, standard deviation, and mean absolute first difference of Music35. The beat descriptor $\mathbf{q}_{\text{beat}}$ contains beat density and music energy. The style descriptor $\mathbf{q}_{\text{style}}$ encodes the provided coarse and fine style labels, while $\mathbf{q}_{\text{pos}}$ contains relative window position and its sine and cosine encodings. The five groups have dimensions 256, 105, 2, 22, and 3, respectively, giving a 388-dimensional query. Std uses per-dimension training-bank statistics, and all five groups have unit weight. An overline denotes ℓ_2 normalization of the complete vector for cosine search.

Bank construction and retrieval. Once MMR and the HCMAEs are frozen, each training window provides a key $\mathbf{k}_i = \mathcal{Q}(\mathbf{m}_i, \mathbf{y}_i, \tau_i)$ and paired hierarchies $\mathcal{Z}_i^s = E^s(\mathbf{x}_i^s)$. The key is derived from the window's music and metadata, whereas the stored motion hierarchies are encoded from its paired recording. We construct the non-parametric bank

$$\mathcal{B} = \{(\mathbf{k}_i, \mathcal{Z}_i^b, \mathcal{Z}_i^h)\}_{i=1}^N. \tag{3}$$

The bank contains 4,558 entries and remains fixed during MAR training and inference. At inference, the query and stored keys use the same five feature groups, ordering, and normalization. We perform exact top-1 cosine retrieval:

$$r = \arg\max_i \overline{\mathbf{q}}_{\text{ret}}^\top \overline{\mathbf{k}}_i, \qquad (\mathcal{Z}^{r,b}, \mathcal{Z}^{r,h}) = (\mathcal{Z}_r^b, \mathcal{Z}_r^h). \tag{4}$$

The returned body hierarchy becomes the source for local adaptation, and the complete hand hierarchy retains the same source index and temporal correspondence.

Music condition for MAR. The music condition $\mathbf{c}_m$ is constructed from the same music and metadata used for $\mathbf{q}_{\text{ret}}$. Music35, its first differences, and their energy cues are resampled to the fine-token grid, while $\mathbf{q}_{\text{MMR}}$, $\mathbf{q}_{\text{style}}$, and $\mathbf{q}_{\text{pos}}$ are repeated along it. These features are concatenated and projected with the motion inputs inside MAR. Thus, $\mathbf{q}_{\text{ret}}$ summarizes the window for retrieval, whereas $\mathbf{c}_m$ retains time-aligned information for local adaptation. Details are given in Appendix B.5.

3.3 Masked Residual Generation

We adapt the retrieved body through residual prediction in its finest stream. In this subsection, the branch superscript is omitted for body tokens. We normalize the fine stream using training-bank statistics and write

$$\mathbf{U}^r = \mathcal{N}_F(\mathbf{Z}_F^r), \qquad \mathcal{Z}_{\text{ctx}}^r = \{\mathbf{Z}_E^r, \mathbf{Z}_C^r, \mathbf{Z}_M^r\}. \tag{5}$$

Here, $\mathcal{Z}_{\text{ctx}}^r$ denotes the retained lower-resolution context. Let $\mathbf{U}$ denote the fine-token input to be adapted. At inference, $\mathbf{U} = \mathbf{U}^r$; during training, local perturbations are applied to construct restoration examples, as described below.

A binary mask $\mathbf{M} \in \{0,1\}^{L_F}$ selects a contiguous region in which updates are permitted. We form a masked view of the input and predict a residual:

$$\widetilde{\mathbf{U}} = (\mathbf{1} - \mathbf{M}) \odot \mathbf{U} + \mathbf{M} \odot \mathbf{e}_{\text{mask}}, \qquad \Delta\mathbf{U} = G_\theta\Big(\widetilde{\mathbf{U}}, \mathbf{U}, \mathcal{Z}_{\text{ctx}}^r, \mathbf{c}_m, \mathbf{M}\Big), \tag{6}$$

where $\mathbf{e}_{\text{mask}}$ is a learned embedding and $\odot$ denotes element-wise multiplication with feature broadcasting. The masked view presents the interval in its surrounding context, while the input stream $\mathbf{U}$ provides the motion values from which the residual is predicted. Both views are constructed from the same input, including its perturbations during training.

The model projects motion and music features to a shared hidden width and processes them with a bidirectional Transformer. Each position can use motion on both sides of the interval together with the target music. The final projection predicts all fine-token residuals jointly in one forward pass. The selected region specifies the temporal extent of adaptation, while the residual determines the actual change at each position. A zero residual leaves the input unchanged.

Residual update and decoding. We apply the residual within the selected region and gradually reduce its contribution near the two boundaries:

$$\mathbf{U}^{\text{out}} = \mathbf{U} + \mathbf{M} \odot \boldsymbol{\omega} \odot \Delta\mathbf{U}, \qquad \hat{\mathbf{x}}^b = D^b\big(\mathbf{Z}_E^r, \mathbf{Z}_C^r, \mathbf{Z}_M^r, \mathcal{N}_F^{-1}\left(\mathbf{U}^{\text{out}}\right)\big). \tag{7}$$

The cosine envelope $\boldsymbol{\omega}$ increases from zero near the entrance of the interval, reaches one in its interior, and decreases toward the exit. This gradual weighting reduces abrupt latent changes where the adapted region meets the retained tokens. The lower-resolution streams and the fine tokens outside the selected region are copied from the source. The frozen decoder then combines the retained context and the updated fine stream to produce body motion.

3.4 Training with Masking and Local Perturbations

Each paired music–motion example provides target fine tokens $\mathbf{U}^{\text{gt}} = \mathcal{N}_F(\mathbf{Z}_F^{\text{gt}})$ through the frozen body encoder. Retrieval supplies the source, and random additive noise corrupts the selected region:

$$\mathbf{U} = \mathcal{C}(\mathbf{U}^r; \mathbf{M}) = \mathbf{U}^r + \mathbf{M} \odot \boldsymbol{\xi}. \tag{8}$$

During training, we vary mask locations and lengths while retaining the surrounding tokens and lower-resolution context. When the source matches the target, perturbation yields a restoration task; without perturbation, an identity update is valid. The model is supervised against the target rather than a prescribed amount of change. Masking details are in Appendix C.2.

Fine-token supervision. We supervise the updated tokens at valid masked positions Ω:

$$\mathcal{L}_{\text{token}} = \frac{1}{|\Omega|} \sum_{t \in \Omega} \left[\text{SmoothL1}(\mathbf{U}_t^{\text{out}}, \mathbf{U}_t^{\text{gt}}) + \lambda_{\cos}\ell_{\cos}(\mathbf{U}_t^{\text{out}}, \mathbf{U}_t^{\text{gt}})\right], \tag{9}$$

where $\ell_{\cos}$ is one minus cosine similarity.

Motion-space supervision. Following LODGE (Li et al., 2024a), we supervise joint positions from forward kinematics and their first- and second-order temporal differences using squared errors. The overall objective is

$$\mathcal{L}_{\mathrm{MAR}} = \mathcal{L}_{\mathrm{token}} + \mathcal{L}_{\mathrm{kin}}, \quad \mathcal{L}_{\mathrm{kin}} = \lambda_{\mathrm{pos}}\mathcal{L}_{\mathrm{pos}} + \lambda_{\mathrm{vel}}\mathcal{L}_{\mathrm{vel}} + \lambda_{\mathrm{acc}}\mathcal{L}_{\mathrm{acc}}. \tag{10}$$

Loss definitions are given in Appendix C.5. Only MAR is optimized, with gradients passing through the frozen decoder and forward kinematics. Losses exclude padding; temporal differences use consecutive valid frames across adapted-region boundaries.

3.5 Full-Body Sequence Synthesis

Given a music track $\mathbf{m}$ and its style labels $\mathbf{y}$, DanceMAR generates a full-body dance by processing and assembling overlapping windows. On the 30 FPS timeline, we construct

$$\{(\mathbf{m}_k, \tau_k)\}_{k=1}^{K} = \mathcal{W}_{L,S}(\mathbf{m}), \qquad L = 300, \quad S = 150, \tag{11}$$

where τ_k is the relative position of window k. The windowing operator $\mathcal{W}_{L,S}$ appends an end-aligned window when the regular stride leaves uncovered frames.

For each window, the music features provide a retrieval query $\mathbf{q}_{\mathrm{ret},k}$ and an aligned MAR condition $\mathbf{c}_{m,k}$, as defined in Section 3.2. The window-level inference mapping is

$$\begin{aligned} \mathbf{q}_{\mathrm{ret},k} &= \mathcal{Q}(\mathbf{m}_k, \mathbf{y}, \tau_k), \qquad r_k = \arg\max_i \overline{\mathbf{q}}_{\mathrm{ret},k}^{\top}\overline{\mathbf{k}}_i, \\ (\widehat{\mathbf{x}}_k^b, \widehat{\mathbf{x}}_k^h) &= \left(D^b\big(\mathcal{A}_\theta(\mathcal{Z}_{r_k}^b, \mathbf{c}_{m,k})\big), D^h(\mathcal{Z}_{r_k}^h)\right). \end{aligned} \tag{12}$$

Here, $\mathcal{A}_\theta$ denotes the masked residual prediction and boundary-weighted hierarchy update in Eqs. (6)–(7). It retains the lower-resolution body streams and updates a middle interval covering 20% of valid fine tokens in one MAR pass, without artificial noise. The hand decoder uses the same retrieved index r_k. The complete dance is obtained by

$$\widehat{\mathbf{x}}^{1:T} = \mathrm{DanceMAR}(\mathbf{m}, \mathbf{y}; \mathcal{B}) = \mathcal{S}\big(\{(\widehat{\mathbf{x}}_k^b, \widehat{\mathbf{x}}_k^h)\}_{k=1}^{K}; \mathbf{m}\big). \tag{13}$$

The operator $\mathcal{S}$ aligns roots across overlapping windows, linearly blends non-rotational channels, and interpolates rotations with quaternion SLERP. Both branches share the same temporal segmentation. It then refines the body's music phase, local dynamics, and foot contacts before attaching wrist-relative hand articulation to the final body wrists. Details are given in Appendix D.

4 Experiments

4.1 Experimental Setup

Dataset and representation. We evaluate on FineDance (Li et al., 2023) using the LODGE (Li et al., 2024a) split of 183 training and 19 test songs at 30 FPS. The paired motion bank and all fitted normalization statistics use only training data. Each frame contains 268 body and 360 hand features. Body features comprise foot contacts, root translation, 6D rotations, local joint positions, and velocities; hand features describe bilateral articulation. Music35 and MERT256 are extracted as in Section 3.1; the five query groups use equal weights. Retrieval also uses the provided style labels and relative window position.

Evaluation metrics. Following LODGE (Li et al., 2024a), we report body FID-K/FID-G, DIV-K/DIV-G, Beat Alignment Score (BAS), and Foot Skating Ratio (FSR, %), together with hand FID and diversity metrics. Diversity is ranked by proximity to the ground-truth references.

Implementation details. We pretrain the body and hand HCMAEs independently on 300-frame windows, and MMR on paired music and body motion. Both pretraining stages use AdamW with a batch size of 128. HCMAE uses a 600-epoch training schedule with an initial learning rate of 2×10^{-4}, while MMR is trained for 500 epochs at 10^{-4}. The pretrained models are frozen to construct a bank of 4,558 training entries, each pairing a music-derived key with body and hand hierarchies. We train the bidirectional MAR model with masking and local perturbations for 20 epochs using AdamW, a learning rate of 2×10^{-4}, and a batch size of 256. All experiments run on a single NVIDIA RTX PRO 5000 GPU (48 GB). Further training details are provided in the appendix.

Table 1: Quantitative comparison on FineDance. Best and second-best results are shown in bold and underlined, respectively. Diversity is ranked by proximity to the ground-truth references: $9.30/7.30$ for body DIV-K/DIV-G and $6.12/2.21$ for hand DIV-K/DIV-G. Ground-truth BAS and FSR are 0.2120 and 6.22%. An asterisk denotes our extension with a hand model following FineNet.

Method	Body Motion						Hand Motion			
	FID-K↓	FID-G↓	DIV-K	DIV-G	BAS↑	FSR(%)↓	FID-K↓	FID-G↓	DIV-K	DIV-G
FACT (Li et al., 2021)	113.38	97.05	3.36	6.37	0.1831	28.44		–		
MNET (Kim et al., 2022)	104.71	90.31	3.12	6.14	0.1864	39.36		–		
Bailando (Siyao et al., 2022)	82.81	28.17	6.66	5.98	0.2029	18.76		–		
EDGE* (Tseng et al., 2023)	94.34	50.38	6.51	5.79	0.2116	20.04	29.02	8.91	2.88	1.01
FineNet (Li et al., 2023)	66.97	49.12	6.67	5.93	0.2121	18.39	22.17	8.76	2.75	0.97
LODGE* (Li et al., 2024a)	45.56	34.29	4.39	5.10	0.2397	2.76	17.17	5.51	2.98	1.12
FODGE* (Huang et al., 2026)	44.51	27.46	6.42	6.03	**0.2509**	**2.38**	17.22	5.39	2.93	1.07
LODGE++ (Li et al., 2025)	40.77	30.79	5.53	5.01	0.2423	4.27		–		
DanceMAR (Ours)	**25.33**	**14.37**	**6.81**	**6.77**	0.2366	4.10	**14.08**	**3.57**	**3.16**	**1.19**

4.2 Comparison with State-of-the-Art Methods

Table 1 compares DanceMAR with FACT, MNET, Bailando, EDGE*, FineNet, LODGE*, FODGE*, and LODGE++ on FineDance.

In **body motion,** DanceMAR achieves the lowest body FID-K and FID-G among the compared methods. Its FID-K of 25.33 improves upon LODGE++ by 37.9%, while its FID-G of 14.37 improves upon FODGE* by 47.7%. Body DIV-K and DIV-G reach 6.81 and 6.77, respectively, both closest to their ground-truth references in this comparison. These results indicate improved distributional fidelity and diversity closer to the ground-truth references. In **music alignment and foot stability,** DanceMAR obtains a BAS of 0.2366, compared with 0.2397 for LODGE* and 0.2423 for LODGE++. Its FSR is 4.10%, below the 4.27% of LODGE++. FODGE* achieves the highest BAS and the lowest FSR. DanceMAR therefore achieves higher body-motion fidelity while maintaining competitive music alignment and foot stability.

In **hand motion,** DanceMAR achieves the lowest hand FID-K and FID-G, at 14.08 and 3.57. The best baseline scores are 17.17 from LODGE* for FID-K and 5.39 from FODGE* for FID-G. Its hand DIV-K and DIV-G of 3.16 and 1.19 are also closest to the ground-truth references among the compared methods. These results support using paired hand articulation from retrieved examples.

Additional AIST++ results and a user study are provided in Appendices E.1 and G, respectively.

4.3 Analysis of DanceMAR

Effect of local adaptation. Table 2 compares the full model with retrieval-only synthesis. With MAR disabled, we set the residual to zero and retain the full model's retrieved entries, decoding, window assembly, phase adjustment, local-dynamics processing, and foot-contact refinement.

With this matched output processing, local adaptation reduces FID-K from 34.82 to 25.33 and FID-G from 19.46 to 14.37, corresponding to reductions of 27.3% and 26.2%. BAS increases from 0.2138 to 0.2366, and FSR decreases from 6.60% to 4.10%. Both diversity scores also move closer to their ground-truth references. This comparison shows that the learned local update improves the retrieved motion beyond the shared deterministic processing.

Boundary weighting. We compare the cosine transition with a uniform residual weight within the same interval. Both variants use the same trained model, predicted residuals, and music conditions. We match the Frobenius norm of the full model's latent update in each window:

$$\gamma = \frac{\|\mathbf{M} \odot \boldsymbol{\omega} \odot \Delta\mathbf{U}\|_F}{\|\mathbf{M} \odot \Delta\mathbf{U}\|_F}, \qquad \mathbf{U}^{\text{uniform}} = \mathbf{U}^r + \gamma\, \mathbf{M} \odot \Delta\mathbf{U}. \tag{14}$$

We set $\gamma = 0$ when the denominator is zero. Both variants share decoding and post-processing. The control redistributes the update temporally while preserving its latent norm.

Table 2: Local adaptation and boundary-weighting controls. All variants use the same retrieval results and output-processing procedures. Uniform weighting matches the full model's latent update norm in each window before decoding. B-Jump measures boundary discontinuity, and Edit measures local change relative to retrieval; both use the same fixed direct-insertion normalization reference across variants. Best results are shown in bold. Edit has no preferred direction and is not ranked.

Variant	FID-K↓	FID-G↓	DIV-K	DIV-G	BAS↑	FSR(%)↓	B-Jump↓	Edit
w/o MAR (retrieval only)	34.82	19.46	5.92	6.31	0.2138	6.60	–	0.00
Uniform weighting (matched update norm)	28.18	18.01	6.21	6.39	0.2287	4.16	0.29	0.45
Full DanceMAR	**25.33**	**14.37**	**6.81**	**6.77**	**0.2366**	**4.10**	**0.14**	0.43

Table 3: Music conditioning and adaptation scope. Only the indicated condition or set of updated streams is changed. Best results within each block are shown in bold. Edit uses the same fixed normalization reference as Table 2 and is not ranked.

Study	Variant	FID-K↓	FID-G↓	BAS↑	FSR(%)↓	Edit
Music	Full conditioning	**25.33**	**14.37**	**0.2366**	**4.10**	0.43
	Shuffled retrieval query $\mathbf{q}_{\mathrm{ret}}$	31.92	18.05	0.2189	5.88	0.45
	Shuffled generation condition $\mathbf{c}_m$	28.76	16.21	0.2052	5.17	0.42
Scope	w/o MAR (retrieval only)	34.82	19.46	0.2138	6.60	0.00
	Adapt $\mathbf{Z}_F$	**25.33**	**14.37**	**0.2366**	**4.10**	0.43
	Adapt $\mathbf{Z}_M + \mathbf{Z}_F$	27.46	15.61	0.2306	5.32	0.62
	Adapt $\mathbf{Z}_C + \mathbf{Z}_M + \mathbf{Z}_F$	31.42	18.36	0.2207	6.21	0.84
	Adapt the full hierarchy	35.98	21.16	0.2113	6.77	1.08

Compared with uniform weighting, the cosine transition reduces B-Jump from 0.29 to 0.14, a reduction of 51.7%. FID-K and FID-G also decrease from 28.18 and 18.01 to 25.33 and 14.37, while Edit remains similar at 0.45 and 0.43. These results show that boundary weighting improves temporal continuity and motion fidelity at a matched latent update magnitude.

Music conditioning and adaptation scope. Table 3 examines the two music pathways and adaptation scope. Shuffling $\mathbf{q}_{\mathrm{ret}}$ changes the selected prior while keeping $\mathbf{c}_m$ fixed, whereas shuffling $\mathbf{c}_m$ leaves the prior unchanged.

Shuffling the retrieval query increases FID-K from 25.33 to 31.92 and reduces BAS from 0.2366 to 0.2189. Shuffling the generation condition increases FID-K to 28.76 and reduces BAS to 0.2052. Edit remains similar across these settings. The selected prior and generator-side music condition therefore both contribute to the quality and music alignment of the final motion.

Adapting only $\mathbf{Z}_F$ gives the lowest FID and FSR and the highest BAS. Updating coarser streams increases Edit and degrades these metrics. Full-hierarchy adaptation raises FID-K to 35.98 versus 25.33 for $\mathbf{Z}_F$, supporting fine-stream updates with retained lower-resolution context.

Hierarchical token roles and adaptation locality. Figure 3(a) reports token-swap distances averaged over both swap directions. Each hybrid combines the lower-resolution streams of one source with the fine stream of another. Root motion is closer to the lower-resolution source, whereas low-pass motion, velocity, and acceleration are closer to the fine-stream source. These results indicate different influences of the two stream groups on the decoded motion.

Figure 3(b) examines the temporal extent of the resulting change. Pose, velocity, and acceleration differences are concentrated inside the adapted interval and decrease near its boundaries. Outside the interval, the normalized change is 0.01 for all three measurements. Together with the nonzero interior change, this result shows that adaptation remains local in the decoded motion. Additional mask-policy comparisons and detailed boundary measurements are provided in the appendix.

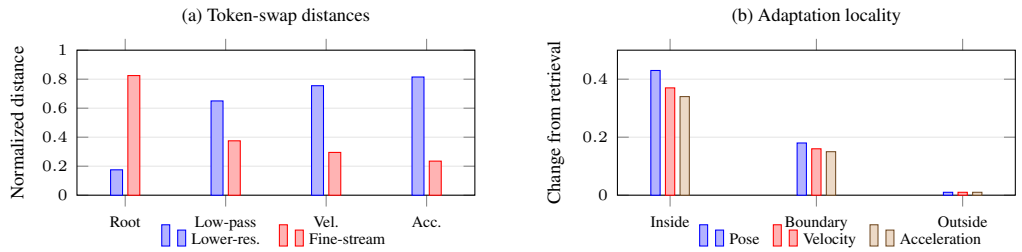

Figure 3: Hierarchical token roles and locality. (a) Normalized distances from hybrids to their token sources. (b) Changes from retrieval inside, near, and outside the adapted interval.

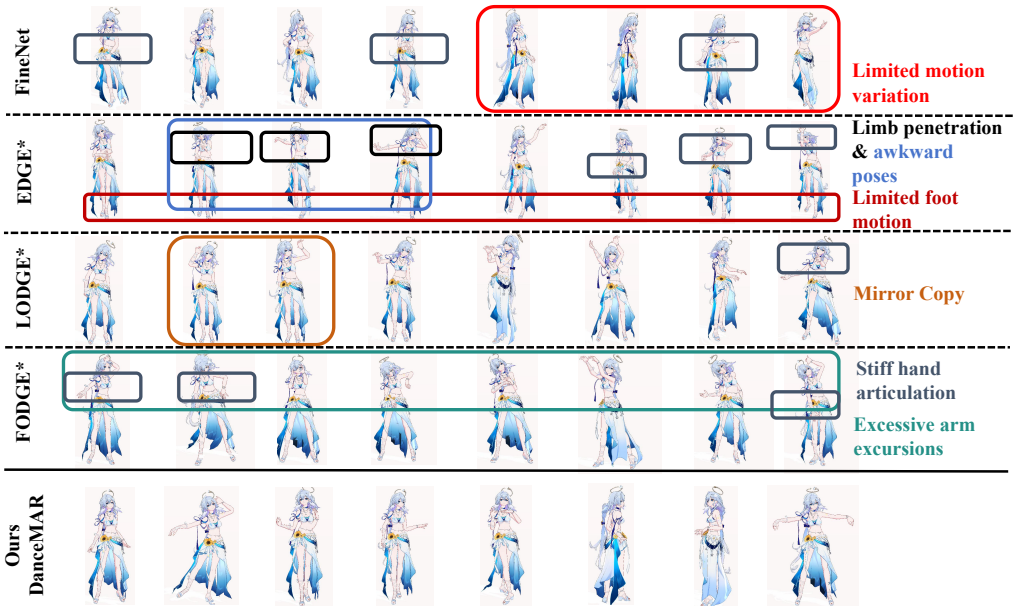

Figure 4: Rendered comparison on FineDance using the same avatar, camera, music segment, and sampled timestamps. Rows show FineNet, EDGE*, LODGE*, FODGE*, and DanceMAR. Starred methods include the added hand model described in Table 1. The comparisons show body configurations, foot placement, and hand articulation.

Qualitative comparison. Figure 4 compares FineNet, EDGE*, LODGE*, FODGE*, and DanceMAR under matched rendering and sampling conditions. The selected frames show body configurations, foot placement, and hand articulation at the same timestamps. The starred methods use the hand extensions described above. Full-song sequences in the supplementary video show the motion progression and transitions between sampled poses.

5 Conclusion

We presented DanceMAR, a framework for local generation over retrieved motion priors in music-driven full-body dance synthesis. Rather than regenerating a complete sequence or directly reusing a retrieved example, DanceMAR retains the retrieved motion context and predicts local residuals in the finest body-motion stream. Boundary weighting connects the adapted regions to the surrounding motion. Hand articulation from the same retrieved example is attached to the wrists of the generated body. Experiments on FineDance demonstrate state-of-the-art body and hand distributional quality together with competitive beat alignment and foot stability. Retrieval, adaptation-scope, token-swap, and boundary analyses further show that the retrieved prior, fine-level residual adaptation, and boundary weighting serve complementary roles. These results show that local residual adaptation improves retrieved motion while preserving its surrounding context.

AI USE STATEMENT

Generative AI tools assisted with manuscript drafting, language polishing, and method implementation, and provided feedback on experimental design and interpretation. All AI-assisted work was reviewed by the authors, and the experimental results, analyses, and scientific claims were verified by the authors. The authors take full responsibility for the final content of this work.

REPRODUCIBILITY STATEMENT

The appendix provides the data processing, model architecture, training, retrieval, and inference details needed to reproduce DanceMAR. The experimental setup specifies the dataset split, evaluation metrics, and implementation settings used for the main results, while the additional experiments document the corresponding ablation and cross-dataset protocols. We also provide the implementation and evaluation code with the supplementary submission.

ETHICS STATEMENT

This work uses existing motion and music datasets and includes a user study based on pairwise comparisons of rendered dance videos. Only aggregate preference results are reported, and the study is used solely to evaluate the perceived quality of generated motion.

REFERENCES

Xin Chen, Biao Jiang, Wen Liu, Zilong Huang, Bin Fu, Tao Chen, and Gang Yu. Executing your commands via motion diffusion in latent space. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 18000–18010, 2023.

Rukun Fan, Songhua Xu, and Weidong Geng. Example-based automatic music-driven conventional dance motion synthesis. *IEEE Transactions on Visualization and Computer Graphics*, 18(3): 501–515, 2012. doi: 10.1109/TVCG.2011.73.

Chuan Guo, Yuxuan Mu, Muhammad Gohar Javed, Sen Wang, and Li Cheng. MoMask: Generative masked modeling of 3D human motions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 1900–1910, 2024.

Ruozi Huang, Huang Hu, Wei Wu, Kei Sawada, Mi Zhang, and Daxin Jiang. Dance revolution: Long-term dance generation with music via curriculum learning. In *International Conference on Learning Representations*, 2021.

Xiaoying Huang, Sanyi Zhang, Qin Zhang, Xiaoxuan Guo, and Long Ye. FODGE: High-fidelity dance generation via full-body optimization. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, 2026. doi: 10.1109/ICASSP55912.2026.11463867.

Biao Jiang, Xin Chen, Wen Liu, Jingyi Yu, Gang Yu, and Tao Chen. MotionGPT: Human motion as a foreign language. In *Advances in Neural Information Processing Systems*, volume 36, pp. 20067–20079, 2023.

Jinwoo Kim, Heeseok Oh, Seongjean Kim, Hoseok Tong, and Sanghoon Lee. A brand new dance partner: Music-conditioned pluralistic dancing controlled by multiple dance genres. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3490–3500, June 2022.

Lucas Kovar, Michael Gleicher, and Frédéric Pighin. Motion graphs. *ACM Transactions on Graphics*, 21(3):473–482, 2002.

Ronghui Li, Junfan Zhao, Yachao Zhang, Mingyang Su, Zeping Ren, Han Zhang, Yansong Tang, and Xiu Li. FineDance: A fine-grained choreography dataset for 3D full body dance generation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 10234–10243, 2023.

Ronghui Li, Yuxiang Zhang, Yachao Zhang, Hongwen Zhang, Jie Guo, Yan Zhang, Yebin Liu, and Xiu Li. Lodge: A coarse to fine diffusion network for long dance generation guided by the characteristic dance primitives. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 1524–1534, 2024a.

Ronghui Li, Hongwen Zhang, Yachao Zhang, Yuxiang Zhang, Youliang Zhang, Jie Guo, Yan Zhang, Xiu Li, and Yebin Liu. Lodge++: High-quality and long dance generation with robust choreography patterns. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 1–15, 2025. doi: 10.1109/TPAMI.2025.3618016.

Ruilong Li, Shan Yang, David A. Ross, and Angjoo Kanazawa. AI choreographer: Music conditioned 3D dance generation with AIST++. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 13401–13412, 2021.

Yizhi Li, Ruibin Yuan, Ge Zhang, Yinghao Ma, Xingran Chen, Hanzhi Yin, Chenghao Xiao, Chenghua Lin, Anton Ragni, Emmanouil Benetos, Norbert Gyenge, Roger Dannenberg, Ruibo Liu, Wenhu Chen, Gus Xia, Yemin Shi, Wenhao Huang, Zili Wang, Yike Guo, and Jie Fu. MERT: Acoustic music understanding model with large-scale self-supervised training. In *The Twelfth International Conference on Learning Representations*, 2024b. URL https://arxiv.org/abs/2306.00107.

Foram N Shah, Parshwa N Shah, Muhammad Usama Saleem, Ekkasit Pinyoanuntapong, Pu Wang, Hongfei Xue, and Ahmed Helmy. Walk before you dance: High-fidelity and editable dance synthesis via generative masked motion prior. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 8796–8804, 2026.

Li Siyao, Weijiang Yu, Tianpei Gu, Chunze Lin, Quan Wang, Chen Qian, Chen Change Loy, and Ziwei Liu. Bailando: 3D dance generation by actor-critic GPT with choreographic memory. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11050–11059, 2022.

Jiangxin Sun, Chunyu Wang, Huang Hu, Hanjiang Lai, Zhi Jin, and Jian-Fang Hu. You never stop dancing: Non-freezing dance generation via bank-constrained manifold projection. In *Advances in Neural Information Processing Systems*, volume 35, pp. 9995–10007, 2022. doi: 10.52202/068431-0726.

Guy Tevet, Sigal Raab, Brian Gordon, Yonatan Shafir, Daniel Cohen-Or, and Amit H. Bermano. Human motion diffusion model. In *International Conference on Learning Representations*, 2023.

Jonathan Tseng, Rodrigo Castellon, and C. Karen Liu. EDGE: Editable dance generation from music. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 448–458, 2023.

Aäron van den Oord, Oriol Vinyals, and Koray Kavukcuoglu. Neural discrete representation learning. In *Advances in Neural Information Processing Systems*, volume 30, 2017.

Jianrong Zhang, Yangsong Zhang, Xiaodong Cun, Shaoli Huang, Yong Zhang, Hongwei Zhao, Hongtao Lu, and Xi Shen. T2M-GPT: Generating human motion from textual descriptions with discrete representations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023a.

Mingyuan Zhang, Xinying Guo, Liang Pan, Zhongang Cai, Fangzhou Hong, Huirong Li, Lei Yang, and Ziwei Liu. ReMoDiffuse: Retrieval-augmented motion diffusion model. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 364–373, 2023b.

A Data and Hierarchical Motion Representation

This appendix describes the motion representation, paired retrieval, and local residual adaptation, followed by additional experimental results. The notation follows the main paper: retrieval supplies paired body and hand hierarchies, and the masked model predicts a residual in the finest body-motion stream.

A.1 Data Processing

We use the FineDance song-level split described in the main paper, with 183 training songs and 19 test songs. Music and motion are aligned at 30 FPS. A full-body frame has 628 features, comprising 268 body features and 360 hand features. The body representation includes foot contacts, root translation, 6D joint rotations, local joint positions, and local joint velocities. The hand representation contains bilateral hand articulation.

Music35 and MERT256 provide the acoustic and high-level music sequences described in Section B.1. The frozen MMR supplies a window-level music descriptor. Retrieval combines it with acoustic statistics, beat cues, provided style labels, and relative position using equal group weights. All transforms and bank entries are derived from training data only.

A.2 HCMAE Architecture

Body and hand motion use separate Hierarchical Continuous Motion Autoencoders (HCMAEs) with the same design and independent parameters. Figure A.1 shows the architecture. For branch $s \in \{b, h\}$, the encoder and decoder are

$$\begin{aligned} \mathcal{Z}^s &= E^s(\mathbf{x}^s) = \{\mathbf{Z}^s_E, \mathbf{Z}^s_C, \mathbf{Z}^s_M, \mathbf{Z}^s_F\}, \\ \mathbf{x}^s_{\text{rec}} &= D^s(\mathcal{Z}^s). \end{aligned} \tag{A.1}$$

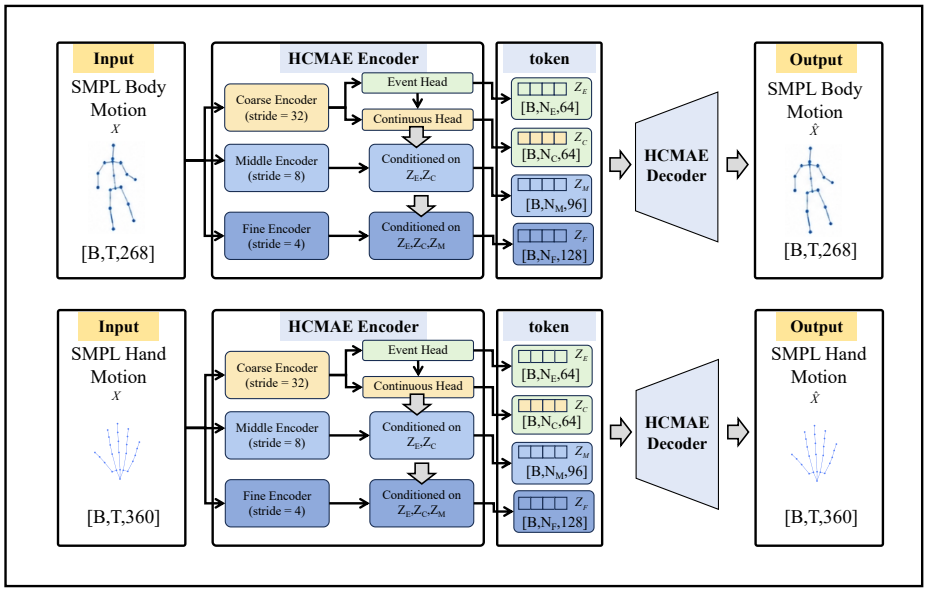

Figure A.1: Architecture of the Hierarchical Continuous Motion Autoencoder. The encoder progressively constructs four continuous streams. The middle stream receives the lower-resolution context, and the fine stream additionally receives the middle-level features. The decoder resamples and combines all four streams to reconstruct motion. Body and hand branches follow the same design with independent parameters.

The hierarchy is constructed progressively. Omitting the branch superscript, we write

$$\begin{aligned}(\mathbf{Z}_E, \mathbf{Z}_C) &= E_c(\mathbf{x}),\\ \mathbf{Z}_M &= E_m(\mathbf{x}, \mathrm{Up}(\mathbf{Z}_E, \mathbf{Z}_C)),\\ \mathbf{Z}_F &= E_f(\mathbf{x}, \mathrm{Up}(\mathbf{Z}_E, \mathbf{Z}_C, \mathbf{Z}_M)).\end{aligned} \tag{A.2}$$

Here, Up resamples the preceding streams to the resolution used by the next stage. The decoder resamples all streams to the frame grid, concatenates their features, and reconstructs the motion jointly. The stream dimensions are listed in Table A.1.

Table A.1: HCMAE streams for a 300-frame motion window.

Stream	Temporal stride	Tokens	Channels
$\mathbf{Z}_E$	32	10	64
$\mathbf{Z}_C$	32	10	64
$\mathbf{Z}_M$	8	38	96
$\mathbf{Z}_F$	4	75	128

Autoencoder learning. Both autoencoders use full-motion and multi-resolution reconstruction with Smooth-L1 objectives. The body branch additionally uses activity and direction supervision. We train the body and hand HCMAEs independently on 300-frame windows using the same optimization settings, as summarized in Table A.2. Both autoencoders are frozen before bank construction and residual-model training.

Table A.2: Architecture and optimization settings of the body and hand HCMAEs. The two branches have independent parameters and share the same optimization settings.

Setting	Body	Hand
Input/output dimension	268	360
Transformer width	384	384
Attention heads	8	8
Feed-forward width	1536	1536
Encoder layers per stage	2	2
Decoder layers	3	3
Training-window length	300 frames	
Optimizer	AdamW	
Adam coefficients (β_1, β_2)	$(0.9, 0.99)$	
Initial learning rate	2×10^{-4}	
Weight decay	10^{-4}	
Batch size	128	
Training schedule	600 epochs	
Learning-rate decay	$\times 0.1$ at epoch 420	
Gradient clipping	1.0	

B Music Features and Paired Retrieval

This section follows the construction order used by the system: frame-level music features, music–motion alignment, window-level query construction, offline paired-bank construction, and query-time retrieval. The same feature sources also condition local residual prediction.

B.1 Music Feature Extraction

Music35. We extract the low-level sequence $\mathbf{A}_m \in \mathbb{R}^{T\times 35}$ with librosa. Audio is resampled to 15,360 Hz and processed with a hop length of 512, matching the 30 FPS motion grid. The feature vector contains one onset feature, 20 MFCC coefficients, 12 chroma features, one onset-peak indicator, and one beat indicator. The sequence is indexed by time so that a music window covers the same interval as its paired body and hand motion.

MERT256. We run frozen MERT-v1-330M (Li et al., 2024b) on audio resampled to 24 kHz. The final four hidden layers are averaged to obtain a 1024-dimensional feature sequence, which is linearly interpolated to 30 FPS. We first standardize the features, fit whitening PCA on training frames, and retain 256 components. The reduced features are standardized again to obtain $\mathbf{H}_m \in \mathbb{R}^{T\times 256}$. All fitted transforms are kept fixed for bank construction and test queries. MERT256 is the frame-level input to MMR; it is distinct from the 256-dimensional window-level descriptor produced by MMR.

B.2 Music–Motion Alignment

MMR is trained on paired MERT256 and 268-dimensional body-motion windows. The music and motion encoders produce a latent mean and log variance, each with 256 channels. A shared motion decoder reconstructs the paired body sequence from either modality. Table B.1 gives the network structure. MMR body inputs use training-fitted min–max normalization and clipping to $[-1, 1]$, separately from the HCMAE motion normalization.

Table B.1: Architecture of the music–motion retrieval encoder.

Component	Input and layers	Output
Music encoder	Linear $256 \to 256$; three residual temporal convolution blocks; two variational tokens; four Transformer encoder layers	256-D mean and log variance
Motion encoder	Linear $268 \to 256$; two variational tokens; six Transformer encoder layers	256-D mean and log variance
Motion decoder	Positional time queries; six Transformer decoder layers; linear output projection	Body268 sequence

The Transformer layers use four attention heads, feed-forward width 1024, and dropout 0.2. The training objective is

$$\mathcal{L}_{\mathrm{MMR}} = 0.5\mathcal{L}_{\mathrm{rec}} + 10^{-4}\mathcal{L}_{\mathrm{latent}} + 10^{-5}\mathcal{L}_{\mathrm{KL}} + 0.5\mathcal{L}_{\mathrm{NCE}}. \tag{B.1}$$

The reconstruction term compares motion decoded from either modality with the paired body-motion target. The latent term aligns the music and motion means with a Smooth-L1 loss. KL regularization includes cross-modal and unit-Gaussian terms. The symmetric InfoNCE objective uses cosine similarity with a temperature of 0.15. After training, MMR is frozen and we use the music-branch mean rather than a stochastic sample:

$$\mathbf{q}_{\mathrm{MMR}} = E^{m}_{\mathrm{MMR}}(\mathbf{H}_m) \in \mathbb{R}^{256}. \tag{B.2}$$

Only the music branch is used to form retrieval keys and new music queries. The motion branch and shared decoder are used to learn the alignment; HCMAE separately provides the motion hierarchies stored in the bank.

B.3 Query Construction

The window-level query combines the MMR descriptor with acoustic, beat, style, and position information:

$$\begin{aligned}\mathbf{q}_{\mathrm{ret}} &= \mathcal{Q}(\mathbf{m}, \mathbf{y}, \tau) \\ &= \mathrm{Std}([\mathbf{q}_{\mathrm{MMR}} \| \mathbf{q}_{\mathrm{acous}} \| \mathbf{q}_{\mathrm{beat}} \| \mathbf{q}_{\mathrm{style}} \| \mathbf{q}_{\mathrm{pos}}])\,.\end{aligned} \tag{B.3}$$

For Music35, we pool each channel by its temporal mean, standard deviation, and mean absolute first difference, yielding 105 acoustic features. The beat group contains beat density and music energy. Style uses the provided song-level coarse and fine labels. Position uses relative window position and its sine and cosine encodings. Table B.2 lists the five groups.

Std standardizes each feature dimension using statistics fitted to the training-bank descriptors. All five group weights are one; no additional group weighting is applied. For cosine search, we normalize the complete descriptor as

$$\overline{\mathbf{q}}_{\mathrm{ret}} = \frac{\mathbf{q}_{\mathrm{ret}}}{\|\mathbf{q}_{\mathrm{ret}}\|_2}, \qquad \overline{\mathbf{k}}_i = \frac{\mathbf{k}_i}{\|\mathbf{k}_i\|_2}. \tag{B.4}$$

The same standardization and normalization are applied to stored keys and incoming queries. MMR learns from music–motion pairs, but the resulting query contains music and metadata features rather than features extracted from the target motion.

Table B.2: Composition of the 388-dimensional retrieval query. All groups use the same unit weight after per-dimension standardization; the complete vector is then ℓ_2 normalized.

Group	Construction	Dimension
$\mathbf{q}_{\mathrm{MMR}}$	Music-branch latent mean	256
$\mathbf{q}_{\mathrm{acous}}$	Music35 mean, standard deviation, mean absolute first difference	105
$\mathbf{q}_{\mathrm{beat}}$	Beat density and music energy	2
$\mathbf{q}_{\mathrm{style}}$	5-D coarse and 17-D fine labels	22
$\mathbf{q}_{\mathrm{pos}}$	Relative position, sine, cosine	3
Total		388

B.4 Offline Paired-Bank Construction

After freezing MMR and both HCMAEs, we extract the five raw query groups for all training windows. We fit a single query standardizer to these descriptors and store each standardized key $\mathbf{k}_i = \mathcal{Q}(\mathbf{m}_i, \mathbf{y}_i, \tau_i)$. The paired motion window is encoded independently by the body and hand HCMAEs:

$$\mathcal{Z}_i^b = E^b(\mathbf{x}_i^b), \qquad \mathcal{Z}_i^h = E^h(\mathbf{x}_i^h). \tag{B.5}$$

The bank stores

$$\mathcal{B} = \left\{ \left(\mathbf{k}_i, \mathcal{Z}_i^b, \mathcal{Z}_i^h, \mathbf{V}_i, \boldsymbol{\eta}_i \right) \right\}_{i=1}^{N}, \tag{B.6}$$

where $\mathbf{V}_i$ records valid tokens and $\boldsymbol{\eta}_i$ records the source recording and window interval. The FineDance bank contains 4,558 training windows. Each key indexes the body and hand hierarchies from the same interval, so the two branches retain their pairing. The bank has no trainable parameters and is not rebuilt from test music or generated motion.

B.5 Retrieval and Generator-Side Music Conditioning

Query-time retrieval. An incoming music window follows the same Music35, MERT256, MMR, and query-normalization steps. Exact top-1 cosine search gives

$$\begin{aligned} r &= \arg\max_i \, \overline{\mathbf{q}}_{\mathrm{ret}}^{\top} \overline{\mathbf{k}}_i, \\ (\mathcal{Z}^{r,b}, \mathcal{Z}^{r,h}) &= (\mathcal{Z}_r^b, \mathcal{Z}_r^h). \end{aligned} \tag{B.7}$$

The selected body hierarchy supplies the input for local residual adaptation. Its paired hand hierarchy retains the same source index and temporal interval.

Music condition for MAR. The generator uses the same music sources as retrieval, while retaining time-varying acoustic information. We resample Music35, its first temporal differences, and the corresponding music and difference energy cues to the fine-token grid. The window-level MMR descriptor, style labels, and position encoding are repeated along this grid. We denote the resulting music-side feature collection by $\mathbf{c}_m$; its components are listed in Table B.3.

Table B.3: Music-side components of $\mathbf{c}_m$ before projection into MAR. Temporal sequences are resampled to the fine-token grid; window-level descriptors are repeated along it.

Component	Temporal alignment	Channels
Music35	Resampled sequence	35
Music35 first difference	Resampled sequence	35
Music and difference energies	Resampled sequences	2
MMR music descriptor	Repeated window descriptor	256
Style	Repeated labels	22
Position	Repeated encoding	3

These features are projected with the retrieved motion inputs to MAR's hidden width of 384. The query $\mathbf{q}_{\mathrm{ret}}$ uses pooled statistics for bank lookup, whereas $\mathbf{c}_m$ preserves acoustic variation across

the fine-token positions. They share the frozen feature pipeline, rather than requiring separate music encoders. The music-conditioning controls change only one use at a time: shuffling the lookup query leaves MAR's music condition unchanged, and shuffling the MAR condition leaves the selected bank entry unchanged.

C Masked Residual Adaptation

C.1 Fine-Token Normalization

The residual model operates in the normalized fine-token space of the body HCMAE:

$$\begin{aligned} \mathbf{U}^r &= \mathcal{N}_F(\mathbf{Z}_F^r) = \frac{\mathbf{Z}_F^r - \boldsymbol{\mu}_F}{\boldsymbol{\sigma}_F + \epsilon}, \\ \mathcal{N}_F^{-1}(\mathbf{U}) &= \mathbf{U} \odot (\boldsymbol{\sigma}_F + \epsilon) + \boldsymbol{\mu}_F. \end{aligned} \tag{C.1}$$

The statistics are fitted to valid training-bank tokens. Padding is excluded. We use $\mathbf{U}$ for the input being adapted and $\widetilde{\mathbf{U}}$ for its masked view.

C.2 Masking and Local Perturbations

A binary mask $\mathbf{M}$ selects the fine-token positions at which updates are permitted. The validity mask $\mathbf{V}$ excludes padding, and $\mathbf{M}_t$ is zero at invalid positions. Training uses middle, random, and two-span contiguous masks. Their locations and lengths vary across training examples.

For each paired training music–motion window, retrieval provides a source hierarchy and the frozen body encoder provides the target fine tokens $\mathbf{U}^{\mathrm{gt}}$. Training queries may retrieve their own source window. We construct the input through a local perturbation operator:

$$\mathbf{U} = \mathcal{C}(\mathbf{U}^r; \mathbf{M}) = \mathbf{U}^r + \mathbf{M} \odot \boldsymbol{\xi}. \tag{C.2}$$

The tensor $\boldsymbol{\xi}$ contains random additive noise that corrupts fine-token values within the selected region. Tokens outside that region remain unchanged. At inference, no artificial noise is added, and $\mathbf{U} = \mathbf{U}^r$.

The masked view is

$$\widetilde{\mathbf{U}} = (\mathbf{1} - \mathbf{M}) \odot \mathbf{U} + \mathbf{M} \odot \mathbf{e}_{\mathrm{mask}}, \tag{C.3}$$

where $\mathbf{e}_{\mathrm{mask}}$ is a learned embedding. Both the masked and reference views use the same input $\mathbf{U}$, including its perturbations during training. Masking supplies temporal context for local prediction, and perturbation supplies examples that require correction. For a matching, unperturbed source, an identity update is also a valid solution.

C.3 Residual Prediction

The model receives the masked input, its reference stream, the three lower-resolution streams, and the target music condition $\mathbf{c}_m$ from Section B.5:

$$\Delta\mathbf{U} = G_\theta\Big(\widetilde{\mathbf{U}}, \mathbf{U}, \mathbf{Z}_E^r, \mathbf{Z}_C^r, \mathbf{Z}_M^r, \mathbf{c}_m, \mathbf{M}, \mathbf{V}\Big). \tag{C.4}$$

Motion and music conditions are projected to the model's hidden width and processed by a bidirectional Transformer. Each position can use motion on both sides of the selected region. The output projection predicts a 128-dimensional residual at each fine-token position in a single forward pass.

The residual model has six bidirectional Transformer blocks, hidden width 384, and eight attention heads. Its output dimension matches the 128 channels of the fine stream.

C.4 Residual Update and Boundary Weighting

For a contiguous interval $[a, b]$, define $d_t = \min(t - a, b - t)$ inside the interval. With transition width $w > 0$, the cosine envelope is

$$\omega_t = \begin{cases} \dfrac{1}{2}\left[1 - \cos\left(\pi \min\left(1, \dfrac{d_t}{w}\right)\right)\right], & a \le t \le b, \\ 0, & \text{otherwise.} \end{cases} \tag{C.5}$$

This envelope describes the contiguous interval used by the default model. For the zero-width interval setting, the weight is one inside the selected region.

The predicted residual is added in normalized space with unit scale in all experiments:

$$\begin{aligned} \mathbf{U}^{\text{out}} &= \mathbf{U} + \mathbf{M} \odot \boldsymbol{\omega} \odot \Delta\mathbf{U}, \\ \mathbf{Z}_F^{\text{out}} &= \mathcal{N}_F^{-1}(\mathbf{U}^{\text{out}}), \\ \widehat{\mathbf{x}}^b &= D^b\big(\mathbf{Z}_E^r, \mathbf{Z}_C^r, \mathbf{Z}_M^r, \mathbf{Z}_F^{\text{out}}\big)\,. \end{aligned} \tag{C.6}$$

The lower-resolution streams remain fixed. Fine tokens outside the selected region also remain unchanged. Within the region, the residual determines the correction and the envelope reduces its contribution near the boundaries. A zero residual preserves the input, regardless of the selected region.

The default model adapts one middle interval covering 20% of valid fine tokens, with a four-token cosine transition. All residuals are predicted jointly in one forward pass. The interval size and transition width are examined in Section E.5.

C.5 Training Objectives

Fine-token supervision. Let $\Omega = \{t \mid M_t V_t = 1\}$ denote valid positions in the selected region. The updated fine tokens are supervised by

$$\begin{aligned} \mathcal{L}_{\text{token}} = \frac{1}{|\Omega|} \sum_{t \in \Omega} \Big[&\text{SmoothL1}(\mathbf{U}_t^{\text{out}}, \mathbf{U}_t^{\text{gt}}) \\ &+ 0.05\, \ell_{\cos}(\mathbf{U}_t^{\text{out}}, \mathbf{U}_t^{\text{gt}}) \Big], \end{aligned} \tag{C.7}$$

where $\ell_{\cos}$ is one minus cosine similarity. The loss supervises the updated representation rather than requiring a particular residual magnitude.

Motion-space supervision. Following the motion-space supervision used by LODGE (Li et al., 2024a), we supervise the decoded body through joint positions and their temporal differences. Let $\mathbf{p}_t$ and $\widehat{\mathbf{p}}_t$ denote stacked target and predicted joint positions obtained by forward kinematics in the same coordinate system. At the fixed frame rate, define

$$\mathbf{v}_t = \mathbf{p}_{t+1} - \mathbf{p}_t, \qquad \mathbf{a}_t = \mathbf{v}_{t+1} - \mathbf{v}_t, \tag{C.8}$$

with predicted quantities defined in the same way. For T valid frames, the losses are

$$\begin{aligned} \mathcal{L}_{\text{pos}} &= \frac{1}{T} \sum_{t=1}^{T} \|\widehat{\mathbf{p}}_t - \mathbf{p}_t\|_2^2, \\ \mathcal{L}_{\text{vel}} &= \frac{1}{T-1} \sum_{t=1}^{T-1} \|\widehat{\mathbf{v}}_t - \mathbf{v}_t\|_2^2, \\ \mathcal{L}_{\text{acc}} &= \frac{1}{T-2} \sum_{t=1}^{T-2} \|\widehat{\mathbf{a}}_t - \mathbf{a}_t\|_2^2. \end{aligned} \tag{C.9}$$

Position supervision constrains body configurations, while velocity and acceleration supervision constrain temporal dynamics. Temporal differences are evaluated across the boundaries between adapted and retained regions. Padded frames are excluded, and each temporal difference uses only valid consecutive frames.

The weighted kinematic term and complete objective are

$$\begin{aligned} \mathcal{L}_{\text{kin}} &= \lambda_{\text{pos}}\mathcal{L}_{\text{pos}} + \lambda_{\text{vel}}\mathcal{L}_{\text{vel}} + \lambda_{\text{acc}}\mathcal{L}_{\text{acc}}, \\ \mathcal{L}_{\text{MAR}} &= \mathcal{L}_{\text{token}} + \mathcal{L}_{\text{kin}}. \end{aligned} \tag{C.10}$$

The joint-position, velocity, and acceleration terms and their weights follow LODGE (Li et al., 2024a). Only G_θ is optimized at this stage. Motion-space gradients pass through the frozen decoder and forward kinematics to the residual model. The autoencoder and retrieval-encoder parameters remain fixed.

Optimization. We train the bidirectional MAR model for 20 epochs using AdamW with a learning rate of 2×10^{-4} and a batch size of 256. Training uses the masking and local perturbations described in Section C.2. The HCMAEs, MMR, and motion bank remain fixed throughout this stage. All experiments run on a single NVIDIA RTX PRO 5000 GPU with 48 GB of memory.

D Full-Body Inference

D.1 Window Processing and Assembly

We process each song with 300-frame windows and a stride of 150 frames. An additional end-aligned window is used when the regular stride leaves uncovered frames. For each window, retrieval returns a paired body and hand hierarchy. The residual model updates the middle interval covering 20% of valid fine tokens, and the body decoder produces the adapted body window.

Let $\mathcal{O}$ denote the overlap between the assembled sequence and an incoming window. We translate the incoming root by

$$\boldsymbol{\delta} = \frac{1}{|\mathcal{O}|} \sum_{t \in \mathcal{O}} \left(\mathbf{r}_t^{\text{old}} - \mathbf{r}_t^{\text{new}} \right). \tag{D.1}$$

After root alignment, non-rotational channels are linearly blended and rotations are interpolated using quaternion SLERP:

$$\begin{aligned} \mathbf{y}_t &= (1 - u_t)\mathbf{y}_t^{\text{old}} + u_t \mathbf{y}_t^{\text{new}}, \\ \mathbf{R}_t &= \text{SLERP}\left(\mathbf{R}_t^{\text{old}}, \mathbf{R}_t^{\text{new}}; u_t \right), \end{aligned} \tag{D.2}$$

where $u_t \in [0, 1]$ increases across the overlap.

D.2 Paired Hand Decoding and Attachment

Hand articulation uses the same source index as the retrieved body prior. For window k with retrieved index r_k, we decode

$$\widehat{\mathbf{x}}_k^h = D^h(\mathcal{Z}_{r_k}^h). \tag{D.3}$$

The hand and body windows therefore share the same recording identity and source time interval. Hand windows follow the same windowing and overlap schedule as the body windows.

After body refinement, hand positions are attached using their wrist-relative offsets. For hand joint j associated with wrist $w(j)$, the attachment is

$$\begin{aligned} \mathbf{h}_{t,j}^{\text{local}} &= \mathbf{h}_{t,j}^{\text{src}} - \mathbf{h}_{t,w(j)}^{\text{src}}, \\ \widehat{\mathbf{h}}_{t,j} &= \widehat{\mathbf{b}}_{t,w(j)}^{\text{final}} + \mathbf{h}_{t,j}^{\text{local}}. \end{aligned} \tag{D.4}$$

All terms use the same coordinate frame. This operation preserves the decoded wrist-relative articulation and leaves the final body trajectory unchanged.

Hand models for the compared methods. For full-body evaluation, we add a hand model following FineNet (Li et al., 2023) to EDGE, LODGE, and FODGE. We denote the extensions by EDGE*, LODGE*, and FODGE*. We evaluate their hand outputs using the same FID and diversity metrics as for DanceMAR. The asterisk distinguishes the extensions from the original methods.

D.3 Deterministic Body Refinement

The assembled body undergoes phase adjustment, local-dynamics and energy processing, and foot-contact refinement. The phase delay is 11 frames, residual smoothing uses a 13-frame window, and the adapted support has a 16-frame margin. The energy threshold and minimum scale are both 0.50. Foot-contact refinement uses a blend of 0.60 and a minimum contact duration of two frames. The same operations and settings are used for the full model and the retrieval-only control. Hand articulation is attached afterward. The refinement stages are evaluated separately in Section E.7.

Inference order. For each music window, we extract Music35 and MERT256, compute $\mathbf{q}_{\mathrm{MMR}}$ with the frozen music branch, and form $\mathbf{q}_{\mathrm{ret}}$ using acoustic statistics, beat cues, style labels, and relative position. The normalized query retrieves one paired entry from the fixed training bank. The same feature sources form the aligned condition $\mathbf{c}_m$ for one MAR pass. We decode the updated body and paired hand hierarchies, assemble the windows, refine the body, and attach wrist-relative hand articulation.

E Additional Experiments

The following experiments extend the main-paper evaluation. Unless otherwise stated, the analyses use FineDance and its ground-truth diversity references: $9.30/7.30$ for body DIV-K/DIV-G and $6.1169/2.2133$ for hand DIV-K/DIV-G. The AIST++ comparison uses the metric notation and reference values in Table E.1. FSR is reported as a percentage.

E.1 AIST++ Transfer and Bank Expansion

We evaluate DanceMAR on AIST++ (Li et al., 2021) under the LODGE/Bailando protocol. This experiment evaluates body motion only. Table E.1 reports the published baseline values from LODGE (Li et al., 2024a), retaining the FID-M and DIV-M notation used in that comparison.

We compare two bank configurations using the same frozen body HCMAE, music–motion retrieval encoder, and MAR model. DanceMAR (FineDance Bank) directly uses the original FineDance motion bank. DanceMAR (Expanded Bank) retains these entries and appends body-motion exemplars from the AIST++ training split. For each appended window, its music and metadata produce a retrieval key through the same query pipeline, while the frozen body HCMAE encodes the paired motion into the stored hierarchy. No learned module is retrained or fine-tuned; the two variants differ only in the available bank entries.

Table E.1: Body-motion comparison on AIST++ under the LODGE/Bailando protocol. The two DanceMAR variants use identical frozen learned modules and differ only in their motion-bank contents. Published baseline values are taken from LODGE (Li et al., 2024a). Best and second-best results among the compared methods are shown in bold and underlined. Diversity is ranked by proximity to ground truth.

Method	FID-K↓	FID-M↓	DIV-K→GT	DIV-M→GT	BAS↑
Ground Truth	–	–	8.19	7.45	0.2374
Li et al.	86.43	43.46	6.85	3.32	0.1607
DanceNet	69.18	25.49	2.86	2.85	0.1430
DanceRevolution	73.42	25.92	3.52	4.87	0.1950
FACT	35.35	22.11	5.94	6.18	0.2209
Bailando	28.16	<u>9.62</u>	<u>7.83</u>	6.34	0.2332
EDGE	42.16	22.12	3.96	4.61	0.2334
LODGE	37.09	18.79	5.58	4.85	**0.2423**
DanceMAR (FineDance Bank)	<u>27.31</u>	16.32	7.61	<u>6.49</u>	0.2344
DanceMAR (Expanded Bank)	**19.05**	**9.41**	**8.04**	**7.02**	<u>0.2401</u>

With the FineDance bank, DanceMAR obtains FID-K of 27.31 and BAS of 0.2344 without target-domain bank expansion. Appending AIST++ training exemplars reduces FID-K to 19.05 and FID-M from 16.32 to 9.41. Both diversity scores move closer to their ground-truth references, and BAS increases to 0.2401. The expanded-bank variant achieves the lowest FID-K/FID-M and DIV-K/DIV-M closest to ground truth among the compared methods, while LODGE retains the highest BAS. These results show that the frozen model can use additional motion exemplars to improve AIST++ synthesis without further model training.

E.2 Local Adaptation and Matched Boundary Control

Table E.2 compares the full model with retrieval-only synthesis and uniform boundary weighting. The retrieval-only variant sets the residual to zero. All three variants use the same retrieved entries, decoding, window assembly, phase adjustment, local-dynamics processing, and foot-contact refinement. The retrieval-only row is therefore a matched output-processing baseline rather than an intermediate stage before refinement.

Table E.2: Local adaptation with matched output processing. The uniform control matches the full model's latent update norm in each window. Edit is descriptive and has no preferred direction.

Variant	FID-K↓	FID-G↓	DIV-K	DIV-G	BAS↑	FSR(%)↓	B-Jump↓	Edit
w/o MAR (retrieval only)	34.82	19.46	5.92	6.31	0.2138	6.60	–	0.00
Uniform weighting (matched norm)	28.18	18.01	6.21	6.39	0.2287	4.16	0.29	0.45
Full DanceMAR	25.33	14.37	6.81	6.77	0.2366	4.10	0.14	0.43

For the uniform control, we reuse the same predicted residual and replace the cosine envelope with a scalar computed separately for each window:

$$\gamma = \frac{\|\mathbf{M} \odot \boldsymbol{\omega} \odot \Delta\mathbf{U}\|_F}{\|\mathbf{M} \odot \Delta\mathbf{U}\|_F}, \qquad \mathbf{U}^{\text{uniform}} = \mathbf{U}^r + \gamma\, \mathbf{M} \odot \Delta\mathbf{U}. \tag{E.1}$$

We set $\gamma = 0$ when the denominator is zero. This operation matches the update norm before decoding, not the motion-space Edit after decoding and refinement. With the same latent update norm, the cosine transition reduces B-Jump from 0.29 to 0.14, while FID-K and FID-G decrease from 28.18 and 18.01 to 25.33 and 14.37. Motion-space Edit remains similar at 0.45 and 0.43.

Hand articulation in the final model uses the paired source index and wrist attachment described in Section D.2. The final hand FID-K/FID-G are $14.08/3.57$, and DIV-K/DIV-G are $3.16/1.19$, as reported in the main comparison.

E.3 Music and Prior Controls

Table E.3 examines two uses of the shared music features: query-based retrieval and MAR conditioning. Shuffling the retrieval query changes the selected prior while leaving the generator-side music condition unchanged. Shuffling $\mathbf{c}_m$ changes the generation condition while retaining the same prior. Both controls increase FID and reduce BAS in the reported results. Random priors and joint shuffling degrade these metrics further.

Table E.3: Expanded music and prior controls. Perturbed conditions report mean ± sample standard deviation over three runs. Index change is measured relative to full conditioning. Foot stability for the single-path controls is reported in the main paper.

$\mathbf{q}_{\text{ret}}$	$\mathbf{c}_m$	FID-K↓	FID-G↓	BAS↑	Index change (%)	Edit
Correct	Correct	25.33	14.37	0.2366	0.00	0.43
Shuffled	Correct	31.92 ± 0.74	18.05 ± 0.39	0.2189 ± 0.0021	95.84 ± 0.84	0.45 ± 0.03
Random prior	Correct	39.72 ± 1.18	22.64 ± 0.73	0.1978 ± 0.0032	99.46 ± 0.21	0.62 ± 0.06
Correct	Shuffled	28.76 ± 0.52	16.21 ± 0.28	0.2052 ± 0.0026	0.00	0.42 ± 0.02
Shuffled	Shuffled	40.86 ± 1.35	23.28 ± 0.81	0.1907 ± 0.0036	95.84 ± 0.84	0.67 ± 0.08

E.4 Hierarchy and Adaptation Scope

Table E.4 compares the streams opened to adaptation. Updating only $\mathbf{Z}_F$ gives the lowest FID and FSR in this comparison. Updating additional streams increases the reported edit magnitude and the change outside the designated interval. Updating $\mathbf{Z}_M + \mathbf{Z}_F$ gives the closest DIV-K to the ground-truth reference, whereas updating only $\mathbf{Z}_F$ gives the closest DIV-G. The lowest-FID setting is therefore not optimal for both diversity metrics.

Table E.4: Adaptation scope over the retrieved hierarchy. Outside-span Δ measures normalized pose change beyond the selected region and transition support.

Updated streams	FID-K↓	FID-G↓	DIV-K	DIV-G	BAS↑	FSR(%)↓	Edit	Outside-span Δ ↓
Retrieval only	34.82	19.46	5.92	6.31	0.2138	6.60	0.00	0.00
$\mathbf{Z}_F$	25.33	14.37	6.81	6.77	0.2366	4.10	0.43	0.01
$\mathbf{Z}_M + \mathbf{Z}_F$	27.46	15.61	6.98	6.64	0.2306	5.32	0.62	0.09
$\mathbf{Z}_C + \mathbf{Z}_M + \mathbf{Z}_F$	31.42	18.36	6.44	6.21	0.2207	6.21	0.84	0.21
Full hierarchy	35.98	21.16	6.06	6.02	0.2113	6.77	1.08	0.35

The 64-pair token-swap experiment examines how the streams affect decoding. Each hybrid combines the three lower-resolution streams from one source with the fine stream from another. The normalized L1 distances in Table E.5 show that root motion is closer to the lower-resolution source, while low-pass motion, velocity, and acceleration are closer to the fine-stream source.

Table E.5: Normalized distances in the 64-pair token-swap analysis. Lower values indicate greater similarity to the comparison source.

Hybrid	Compared with	Root L1	Low-pass L1	Velocity L1	Acceleration L1
A lower + B fine	A	0.17	0.64	0.76	0.82
A lower + B fine	B	0.83	0.38	0.29	0.23
B lower + A fine	B	0.18	0.66	0.75	0.81
B lower + A fine	A	0.82	0.37	0.30	0.24

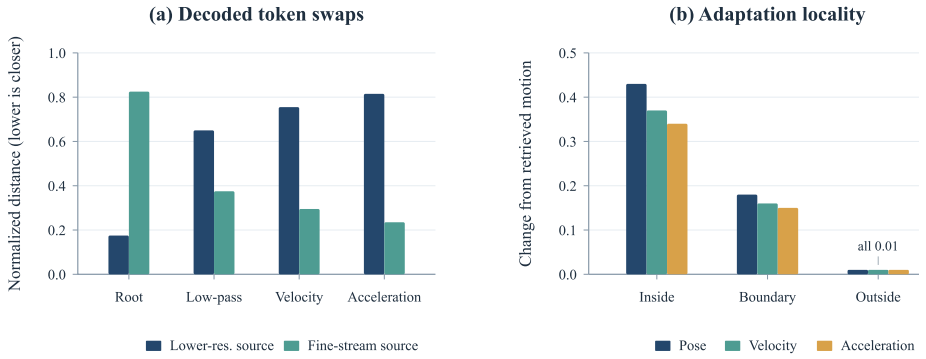

Figure E.1: Hierarchical representation and adaptation locality. The left panel compares decoded hybrids with their two source motions. The right panel shows changes inside the adapted interval, near its boundaries, and outside it.

E.5 Mask Design and Boundary Weighting

Table E.6 compares masking patterns under the same editable-token budget. The contiguous patterns have lower FID and boundary jump than the fragmented patterns in this comparison. The single middle interval gives the lowest FID and the highest edit locality among the tested settings. The random interval has a slightly lower FSR, at 4.06% versus 4.10%.

Table E.7 reports sensitivity to the editable ratio and transition width. The 20% setting gives lower FID than 10% or 30% in this comparison. Increasing transition width reduces the boundary jump and also reduces the retained edit. Residuals are added at unit scale throughout these comparisons.

Table E.8 compares linear and cosine transition weights with the same predicted residual and subsequent body refinement. These schedules are not additionally norm-matched; Eq. (E.1) provides the

Table E.6: Fine-token mask policies. Edit locality is the percentage of pose-change energy within the mask and transition support.

Policy	FID-K↓	FID-G↓	BAS↑	FSR(%)↓	B-Jump↓	Edit locality (%)
Single middle interval	25.33	14.37	0.2366	4.10	0.14	97.64
Single random interval	25.72	14.61	0.2351	4.06	0.17	96.18
Two intervals	26.83	15.14	0.2312	4.42	0.26	92.46
Scattered tokens	30.47	17.69	0.2186	5.47	0.61	76.43
Alternating tokens	33.58	19.34	0.2094	7.16	0.79	68.35

Table E.7: Sensitivity to editable ratio and transition width. Each block varies one setting, with unit residual scale.

Parameter	Setting	FID-K↓	FID-G↓	B-Jump↓	Edit
Editable ratio ρ	0.10	28.49	16.22	0.12	0.24
	0.20	25.33	14.37	0.14	0.43
	0.30	27.43	15.59	0.23	0.64
Transition width	0	27.82	15.74	0.44	0.51
	4	25.33	14.37	0.14	0.43
	8	25.68	14.58	0.12	0.34

separate norm-matched control. The cosine schedule gives lower FID and B-Jump with a similar retained edit.

Table E.8: Linear and cosine boundary transitions. Both use unit residual scale and the same output processing. Edit is not ranked.

Boundary weight	FID-K↓	FID-G↓	B-Jump↓	Edit
Linear transition	25.71	14.58	0.18	0.44
Cosine transition	25.33	14.37	0.14	0.43

E.6 Temporal Locality

Table E.9 measures changes relative to retrieval inside the selected interval, in the transition region, and outside the interval. The reported pose and temporal-difference changes are largest inside the interval and smaller near its boundaries. The token-change column measures changes after re-encoding the output, rather than direct edits to the latent tokens.

B-Jump measures discontinuity at the adapted-interval boundaries, while Edit measures local change relative to retrieval. Both use the same fixed direct-insertion normalization reference as the main paper. Edit locality reports the percentage of pose-change energy within the selected region and transition support.

E.7 Deterministic Output Refinement

Table E.10 separates the deterministic output stages. Phase adjustment primarily improves BAS. Local dynamics and energy control reduce FID and the derivative peak. The final foot-contact correction further reduces FSR.

E.8 Additional Qualitative Results

The supplementary video provides complete songs and synchronized comparisons. Local adaptation is limited by the motion available in the bank: a fine-level update does not replace the entire choreography when the retrieved example is a poor match.

Table E.9: Temporal locality relative to the retrieved prior.

Region	Pose Δ	Velocity Δ	Acceleration Δ	Token change (%)
Inside adapted interval	0.43	0.37	0.34	41.86
Transition boundary	0.18	0.16	0.15	16.42
Outside adapted interval	0.01	0.01	0.01	0.32

Table E.10: Cumulative effects of deterministic body-refinement stages. Derivative peak is normalized by its value before refinement.

Stage	FID-K↓	FID-G↓	BAS↑	FSR(%)↓	Derivative peak↓
Before refinement	27.16	15.32	0.2251	7.87	1.00
+ Phase adjustment	26.68	15.04	0.2351	6.72	0.98
+ Local dynamics and energy control	25.49	14.45	0.2368	5.43	0.57
+ Foot-contact correction	25.33	14.37	0.2366	4.10	0.56

F Model Size and Runtime

Table F.1 reports parameter counts and runtime on an NVIDIA RTX PRO 5000 GPU with 48 GB of memory. Window latency is measured for 300 frames, and full-song latency is measured for one 99-second music track. The parameter subtotal excludes the frozen MERT feature extractor; the motion bank has no learned parameters. Timings cover the components listed in the table.

Table F.1: Model size and runtime. Window latency is for 300 frames, and full-song latency is for a 99-second music track. The parameter subtotal excludes MERT.

Component	Parameters (M)	Window latency (ms)	99-second song latency (s)
Body HCMAE	23.72	22.40	0.36
Hand HCMAE	23.96	11.60	0.19
Music–motion encoder	16.39	14.80	0.22
Masked residual model	11.21	96.50	1.65
Retrieval, update, and assembly	0.00	9.70	0.16
Listed-component subtotal	75.28	155.00	2.58

The reported motion-synthesis stages take 2.58 seconds for the 99-second track. The masked residual model accounts for most of this time. Retrieval, residual weighting, and window assembly add no trainable parameters.

G User Study

Protocol. We conduct a pairwise user study with 20 participants, comparing DanceMAR with seven baseline methods and ground truth. For each of these eight comparison targets, three clips are sampled from the 19-song FineDance test set. Each participant evaluates all 24 pairs, giving 480 judgments in total. Each pair compares DanceMAR with the target method or ground-truth motion for the same music segment.

All motions are retargeted to the same avatar and rendered in Blender using the same camera, lighting, background, frame rate, and video settings. The paired videos include the corresponding music. The DanceMAR output is randomly placed on either side. The clip schedule and side-assignment policy are shared across participants. Participants answer one forced-choice question:

> Which dance has better overall dance quality?

Ties are not allowed.

Results. The DanceMAR win rate is the proportion of judgments selecting DanceMAR. Each comparison target receives $20 \times 3 = 60$ binary judgments, so one vote changes the reported rate by approximately 1.67 percentage points. Table G.1 reports the integer counts and percentages rounded to two decimal places.

Table G.1: Pairwise perceptual results on FineDance. Each target receives 60 forced-choice judgments from 20 participants evaluating three clips. Win rate denotes the percentage of judgments selecting DanceMAR. Starred methods use the hand-model extensions described in the main paper.

Comparison target	DanceMAR wins	Win rate (%)
Ground Truth	28/60	46.67
FACT	60/60	100.00
MNET	59/60	98.33
Bailando	55/60	91.67
EDGE*	52/60	86.67
FineNet	47/60	78.33
LODGE*	45/60	75.00
FODGE*	45/60	75.00

DanceMAR is selected more often than each of the seven baseline methods, with win rates ranging from 75.00% to 100.00%. Against ground truth, DanceMAR is selected in 28 of 60 judgments (46.67%). Figure G.1 shows the interface used for the pairwise comparisons.

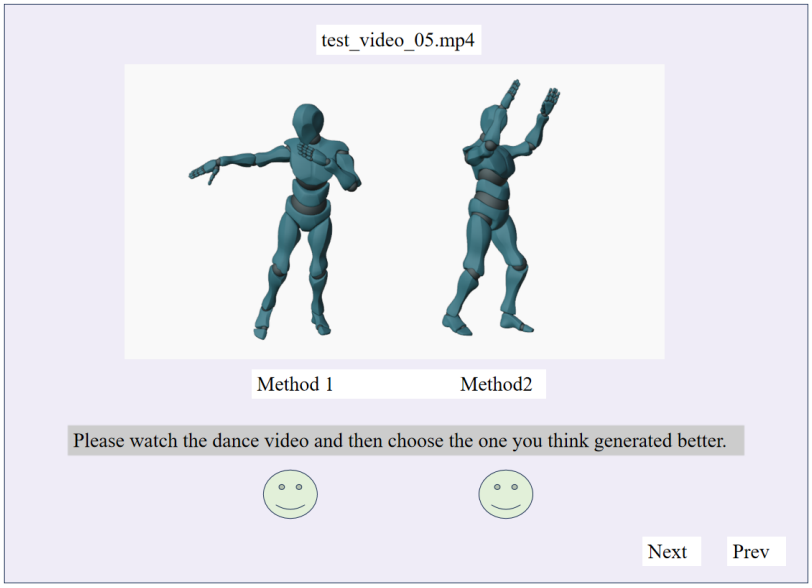

Figure G.1: Interface used for the pairwise user study.