Video Motion Graphs

Supplementary Material

This supplemental document contains five sections:

- Video and Codes (Section A).
- Importance of Non-linear Blending (Section B).
- Implementation Details (Section C).
- Rule-based Searching (Section D).
- Baseline Settings (Section E).
- Evaluation Metrics (Section F).
- Ablation study of CLIP and Reference Net (Section G).

A. Video and Codes

We provide a comprehensive video and separate videos to demonstrate the performance of our system, including:

- General results for dance, gesture, and action-to-motion generation.
- Additional applications, including real-time multiple motion generation, keyframe editing and replacement.
- Comparison of different Video Frame Interpolation (VFI) methods.
- Evaluation of the effectiveness of the Motion Diffusion Model (MDM).
- Evaluation of the effectiveness of the Reference Decoder.

The anonymous scripts including the Video Motion Graph system, the system first uses PoseInterp for motion interpolation and then employs HMInterp for video frame interpolation.

B. Importance of Non-linear Blending

Linear blending in prior works such as GVR [69] and TANGO [33], offers a straightforward approach to guiding motion interpolation. While effective for simple motion scenarios like talk shows, it struggles with complex and dynamic motions, such as dance. This limitation underscores the need for non-linear blending techniques to achieve realistic motion generation. To illustrate this, we analyze the differences between linearly blended motion and ground-truth motion using sequences from *Show-Oliver* [63] and *Champ-dance* [70]. By setting a fixed threshold (e.g., 0.001) to measure deviations, we observe a clear discrepancy between linear blending and the ground truth, particularly for complex motions. Specifically, 78% of the samples from *Show-Olive* falls below the threshold, indicating that linear blending suffices for most cases in this dataset. However, only 17% of the samples from *Champ-dance* is in the same threshold, highlighting that linear blending is unsuitable for high-dynamics motions.

C. Implementation Details

We initialize our VFI model using the pretrained weights from Stable Diffusion 1.5 [46] for spatial layers and AnimateDiff [14] for temporal layers. All the training begins with a learning rate of 1×10^{-5}. The VFI model (both spatial and temporal layers) is initially trained without motion guidance at a resolution of 256×256 on Nvidia H100 GPUs. This training is conducted for 100K iterations. Next, we train the VFI model with pose guidance, keeping the weights of the image guider fixed, at a resolution of 512×512 for 8K iterations. The reference decoder is trained separately for 100K iterations. The Motion Diffusion Model (MDM) is trained on Nvidia A100 GPU for 120K iterations. We retrain all baseline models with our mixed dataset including MotionX, TED and Champ dataset. Our HMInterp has similar inference speed and memory cost with AnimateAnyone (50.7s vs. 48.1s, 39.2G vs. 37.5G for 768×768 14-frame videos on A100 40G).

D. Rule-based Searching

Our system could support motion retrieval for diverse scenarios by introducing task-specific rules. We implement three types of conditions in this paper: Music2Dance, Action2Motion, and Speech2Gesture. These rules-based searches demonstrate the system's ability to align generated motions with the different conditions. We define the path cost functions as task-specific CLIP-like feature distance, *e.g.*, CLIP-like joint embedding in MotionClip for text[53], ChoreoMaster [8] for music, TANGO for speech audio [33]. Once the graph and path costs are defined, path searching can be performed using dynamic programming (DP) for offline processing or with efficient Beam Search [52] for real-time applications. More details on task definitions and task-specific cost designs are provided in the following paragraphs. In addition, we introduce node-level path searching using shortest path algorithms on weighted graphs. Given the target sequence with K keyframes, we separately search the discontinuous region with the Dijkstra algorithm [54], using a length scale factor D, which allows the target clip to be $(1 - D)$ times the length of the target, and then re-interpolate the searched path to the target length.

Music2Dance For Music2Dance retrieval, we focus on beat matching to synchronize dance motions with the rhythm of the music. Inspired by AIST++ [27], we detect beat points in dance motion by computing local minima in motion velocity and extracting beat information from music. Besides, in the reference video, both motion and music BEAT points are evenly distributed. In addition to beat score, similar to Choreomaster [8], we introduce a structural penalty term. This term penalizes repeated motion patterns excessively. We also adopt the CLIP-like feature distance trained from joint embedding in Choreomaster for music-motion content matching.

Action2Motion For Action2Motion retrieval, we employ a combination of keyword matching and action segmentation to retrieve motions effectively for specific actions. Videos containing multiple actions are first segmented into smaller segments based on action segmentation. Each segment is then tagged with either unsupervised labels (e.g., Action A, B, C) or manually assigned labels (e.g., sitting, walking). During retrieval, each motion segment is assigned two labels: a global action tag and a local ordering tag. The system first matches the global action type and then selects frames from the closest matching local order, ensuring temporal coherence across action segments. For Text2motion, we adopt the CLIP-like feature distance from MotionCLIP.

Speech2Gesture For Speech2Gesture retrieval, we adopt a latent-space-based approach inspired by TANGO [33]. We use the pretrained weights from TANGO to calculate the audio-motion difference in latent-space features to determine the optimal path. We minimize the global audio-motion distance using Dynamic Programming (DP). After retrieval and sampling, we adopt a lip-sync model [42] to post-processing the output. This alignment not only improves visual coherence but also enhances the emotional expressiveness of the generated gestures.

E. Baseline Settings

For comparison with previous fully generative human motion video systems, we select state-of-the-art generation methods for various sub-tasks. Specifically, we compare against DanceAnyBEAT [55] for dance generation, S2G-Diffusion [15] for gesture generation, and Text2Performer [23] for action-to-motion generation. DanceAnyBEAT is a video diffusion model that incorporates audio features via cross-attention and integrates text features into the UNet architecture. S2G-Diffusion is an end-to-end diffusion model designed to generate co-speech gesture videos directly from speech input. Text2Performer generates human motion videos based on action descriptions. For the evaluation, we compare the demo videos available in their repositories with our results through a user study.

F. Evaluation Metrics

We use both pixel-level and feature-level evaluations to evaluate the quality of the generated videos. The metrics are Peak Signal-to-Noise Ratio (PSNR), MOtion-based Video Integrity Evaluation (MOVIE) [47], Learned Perceptual Image Patch Similarity (LPIPS) [67], and Fréchet Video Distance (FVD) [51].

Peak Signal-to-Noise Ratio (PSNR) PSNR measures how similar the generated frames are to the ground truth frames at the pixel level. It is based on the mean squared error (MSE) and is expressed in decibels. Higher PSNR values mean less reconstruction error. The formula is:

$$\text{PSNR} = 10 \cdot \log_{10}\left(\frac{\text{MAX}_I^2}{\text{MSE}}\right) \tag{1}$$

Here, MAX_I is the maximum possible pixel value (e.g., 255 for 8-bit images), and MSE is the average squared difference between the original and generated frames.

MOtion-based Video Integrity Evaluation (MOVIE) MOVIE evaluates both spatial and temporal differences in video frames to evaluate the video quality. It assesses how well frames are interpolated and how smooth the transitions are. The formula for MOVIE is:

$$\begin{aligned}\text{MOVIE} = &\frac{1}{N}\sum\left[(v_1(t+1) - v_1(t)) - (v_2(t+1) - v_2(t))\right]^2 \\ &+ \frac{1}{N}\sum(v_1 - v_2)^2\end{aligned} \tag{2}$$

Here, v_1 and v_2 represent the ground truth and generated video frames, respectively, and N is the total number of frames. Lower MOVIE values indicate better video quality.

Learned Perceptual Image Patch Similarity (LPIPS) LPIPS measures how similar two images look in terms of features learned by a neural network. It focuses on perceptual quality rather than just pixel accuracy. The formula is:

$$\text{LPIPS}(x, y) = \sum_l w_l \left\|\phi_l(x) - \phi_l(y)\right\|_2 \tag{3}$$

Here, x and y are the generated and reference images, ϕ_l is the feature map from the l-th layer of a pre-trained network, and w_l is a weight for that layer. Smaller LPIPS scores mean better perceptual similarity. We adopt the pre-trained VGG [50] in Pytorch LPIPS as the feature extractor.

Fréchet Video Distance (FVD) FVD measures how similar the real and generated videos are in a feature space. It considers both the average features and their variability over time. The formula is:

$$\text{FVD} = ||\mu_r - \mu_g||^2 + \text{Tr}(\Sigma_r + \Sigma_g - 2(\Sigma_r\Sigma_g)^{1/2}) \tag{4}$$

Here, μ_r and μ_g are the means of the features for real and generated videos, and Σ_r and Σ_g are their covariances. Lower FVD values indicate better realism and smoother motion in the generated videos. We adopt the pretrained I3D [3] network as the feature extractor.

G. Ablation study of CLIP and Reference Net

Table 7. **Objective Comparison of CLIP and Reference Net.** Without ReferenceNet and CLIP image encoder, the results perform worse on all objective scores

Method	**PSNR** $\uparrow$	**LPIPS** $\downarrow$	**MOVIE** $\downarrow$	**FVD** $\downarrow$
HMInterp ($s = 1$)	39.53	0.034	39.18	1.210
w/o CLIP	34.07	0.069	65.47	1.588
w/o Reference Net	32.68	0.110	78.19	2.098

We show that CLIP and Reference Net is necessary to minimize low-level artifacts, as shown in the Table and Figure, removing CLIP weakens denoising, causing noisier videos, while removing Reference Net results in blurriness.

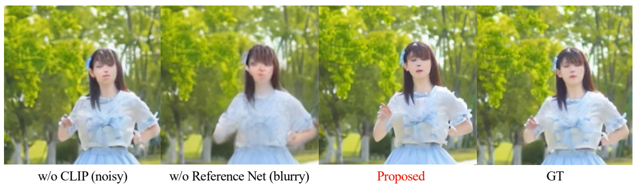

Figure 10. **Subjective Comparison of CLIP and Reference Net.** We remove the CLIP and Refernece Net on our (denoted as proposed) method. Results show that removing CLIP weakens denosing, and removing Reference Net results in clearly blurriness.