

KeyVID: Keyframe-Aware Video Diffusion for Audio-Synchronized Visual Animation

Supplementary Material

A. Details of Keyframe Localization Network from Audio

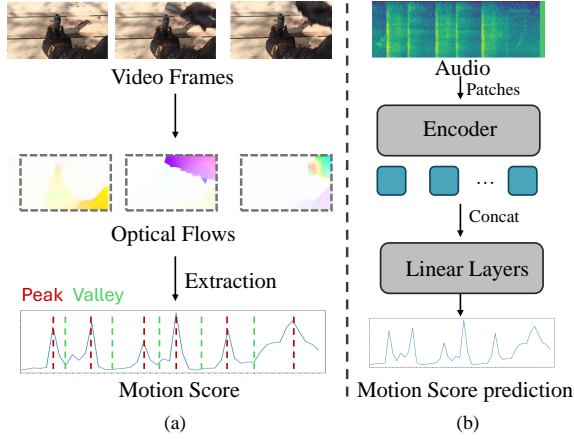


Figure 5. (a) We first calculate the optical flow and then take the average across all pixels for each frame to form a curve of motion score. The peaks (red) and valleys (green) indicate key frames. (b) key frame prediction network from audio, described in Sec. 3.1.

In the Sec. 3.1 of main paper, we introduce that we need to know the position of key frame at the begin of inference by predicting optical motion scores. Here is the detailed structure of this network. The network processes raw audio by converting it into a spectrogram $\mathbf{A} \in \mathbb{R}^{C_A \times T_A}$, where C_A denotes the number of frequency channels and T_A represents the temporal length. The original ImageBind preprocessing pipeline applies a CNN with a kernel stride of (10, 10) to patchify the input spectrogram, producing feature embeddings that are then processed by a transformer-based encoder $f_{\text{audio}} \in \mathbb{R}^{B \times T \times C}$. However, this results in T (e.g. $T=19$) being misaligned with the temporal resolution of the dense motion curve sequence (e.g. 48).

To address this, we modify the CNN stride to (10, 4), increasing the temporal resolution of extracted features (e.g. increase to 46). The transformer encoder then processes the updated feature sequence:

$$\mathbf{F}_{\text{audio}} = f_{\text{audio}}(\mathbf{A}), \quad \mathbf{F}_{\text{audio}} \in \mathbb{R}^{B \times T' \times C}, \quad (1)$$

where $T' > T$ reflects the increased temporal resolution. Since the transformer relies on positional embeddings, we interpolate the pretrained positional embeddings to match the new sequence length T'_A and keep them frozen during training.

The extracted features are passed through fully connected layers to predict a sequence of confidence scores $\mathbf{s} \in \mathbb{R}^{B \times T'}$, where each s_t represents the likelihood of a keyframe occurring at time step t :

$$\mathbf{s} = \sigma(\mathbf{W}\mathbf{F}_{\text{audio}} + \mathbf{b}), \quad (2)$$

where $\mathbf{W} \in \mathbb{R}^{C \times 1}$ and $\mathbf{b} \in \mathbb{R}^{T'_A}$ are learnable parameters, and $\sigma(\cdot)$ is the sigmoid activation function. The model is trained using an L1 loss:

$$\mathcal{L} = \|\mathbf{s} - \hat{\mathbf{s}}\|_1, \quad (3)$$

where $\hat{\mathbf{s}}$ represents the ground-truth keyframe labels derived from optical flow analysis.

B. Details of Keyframe Selection

B.1. Detect Peak and Valley

To identify the local maxima (*peaks*) and minima (*valleys*) from a one-dimensional motion score $\{M(t)\}_{t=1}^T$, we perform the following steps:

- Smoothing:** Convolve the raw score $M(t)$ with a short averaging filter with a window size 5, producing a smoothed label $\tilde{M}(t)$. This helps reduce noise and minor fluctuations.
- Peak Detection:** Finds all local maxima by simple comparison of neighboring values for $\tilde{M}(t)$. We force a minimum distance of 5 frames between any two detected peaks and requiring a prominence (height relative to its surroundings) of at least 0.1. This returns the indices of the local maxima.
- Valley Detection:** Repeat the same peak-finding procedure on the negative of the smoothed signal.

B.2. Sample keyframes

In the main text, we discuss the process of selecting $T_K \ll T$ keyframes based on the motion score $M(t)$ for each frame. Specifically, we first pick the initial frame, then select up to $\frac{T_K}{2} - 1$ peaks among all detected ones (or all peaks if fewer are found). Next, we include a valley between each consecutive pair of selected peaks. Finally, we sample any remaining frames by an evenly distributed (proportional) strategy, which approximates uniform downsampling if few peaks and valleys are present. This approach ensures that smooth motion or weak audio signals, producing limited peaks and valleys, do not degrade the consistency of training for video diffusion models.

Algorithm 1 is the detailed pseudo-code for the full procedure, including both peak and valley selection and the final proportional allocation of remaining key frames.

C. Structure of Motion Interpolation

As shown in Fig. 6, we present the pipeline of motion interpolation network as introduced in Sec. 3.3. After generating T_K keyframes, we use a *motion interpolator* to generate the missing frames back to the full video sequence of length T . Interpolation has been widely used in uniform frame generation [1, 24], where a model predicts a fixed number of intermediate frames given the

Algorithm 1: Keyframe Selection Algorithm

Input: Motion scores $\{M(t)\}_{t=1}^T$, desired keyframe count $T_K \ll T$.

Output: A set of T_K keyframes.

- 1 **Step 1: Detect peaks and valleys** based on $M(t)$.
 - 2 **Step 2: Initialize keyframe list:**
Keyframes $\leftarrow \{\text{first_frame}\}$.
 - 3 **Step 3: Randomly select peaks**
Choose up to $\lfloor \frac{T_K}{2} - 1 \rfloor$
from the detected peaks and add to Keyframes.
 - 4 **Step 4: Insert valleys**
for each pair of consecutive peaks in Keyframes do
 Select one valley in between and add it to Keyframes.
 - 5 **Step 5: Compute how many more keyframes are needed:**
 $R \leftarrow T_K - |\text{Keyframes}|$.
 - 6 **if** $R > 0$ **then**
 - 7 Define a list of N remaining frames (unselected) with some weights $\{w_1, \dots, w_N\}$.
 - 8 $W \leftarrow \sum_{i=1}^N w_i$
 - 9 **for** $i \leftarrow 1$ **to** N **do**
 - ideal_share $_i \leftarrow R \cdot \frac{w_i}{W}$;
 - allocated $_i \leftarrow \lfloor \text{ideal_share}_i \rfloor$;
 - 10 $r \leftarrow R - \sum_{i=1}^N \text{allocated}_i$; // Remainder after flooring
 - 11 **if** $r > 0$ **then**
 - for** $i \leftarrow 1$ **to** N **do**
 - frac $_i \leftarrow \text{ideal_share}_i - \text{allocated}_i$;
 - Sort frames by frac $_i$ in descending order.
 - for** $j \leftarrow 1$ **to** r **do**
 - $i^* \leftarrow$ index of the j -th largest frac $_i$;
 - allocated $_{i^*} \leftarrow \text{allocated}_{i^*} + 1$;
 - 12 **for** $i \leftarrow 1$ **to** N **do**
 - if** allocated $_i > 0$ **then**
 - Keyframes $\leftarrow \text{Keyframes} \cup \{\text{frame}_i\}$;
 - 13 **return** Keyframes
-

first and last frame. However, for keyframe-based generation, the positions of missing and available frames vary, introducing additional challenges.

To address this, we adapt our *keyframe generator* diffusion model into a *motion interpolator* model that generates T_K frames at once using masked frame conditioning. The overall architecture remains nearly unchanged, with the primary difference lying in how image conditions are incorporated. Rather than conditioning solely on the first frame, the model utilizes the features of generated keyframes as conditions, thereby learning to synthesize the missing frames in between. This approach facilitates interpolation

between non-uniformly distributed keyframes while maintaining temporal consistency. A pipeline can be found in Appendix C.

To generate a full video with T frames in a single pass, we incorporate FreeNoise [15] to increase the number of output frames during inference. This allows the interpolation model to take all generated keyframes as conditioning inputs and predict all missing frames in one single step.

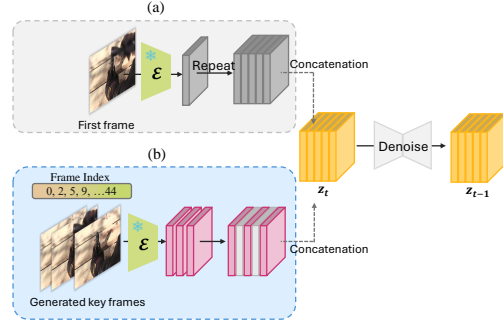


Figure 6. The frame interpolation models has the same structure as the original keyframe generation model, but has different image features for concatenation. (a) For the keyframe generation in Sec 3.2.2, the first frame features are repeated to match the frame length of the latent vector; (b) In frame interpolation, the condition feature from keyframes are padded with zero tensor between keyframe locations to match the frame length.

D. Motion score prediction evaluation

Quantitative result. We evaluate the keypoint detected from the predicted motion score with the ground truth score. We calculate the average precision with a distance threshold t . In this way, for each keypoint in ground truth motion score curve, if it can match with a predicted keypoint with distance lower than t , it will be consider as a successful match. The average precision means the the average of $N_{match}/N(total)$ across all instance, denoted as $AP@t$. We achieve the $AP@3 = 60.57\%$ and $AP@5 = 77.92\%$.

Visualization

E. More Qualitative Results of Video Generation

As the generation result need to be watch with audio for the best experience, we have put more visualization result into the supplementary as mp4 files.

F. Experimental Details

For the experiments of KeyVID on three dataset AVSyncD, Landscape, and TheGreatestHit, we all train on resolution 320×512 as Dynamicrafter [24]. During the inference time, we use ddim sampling with step 90. The temporal length of both key frame generation and interpolation model are all 12. As our interpolation module use freenoise[15] techniquial to obtain the final 48 frames in one run. we change the windows size 12 and the stride 6 to fit our temporal length.

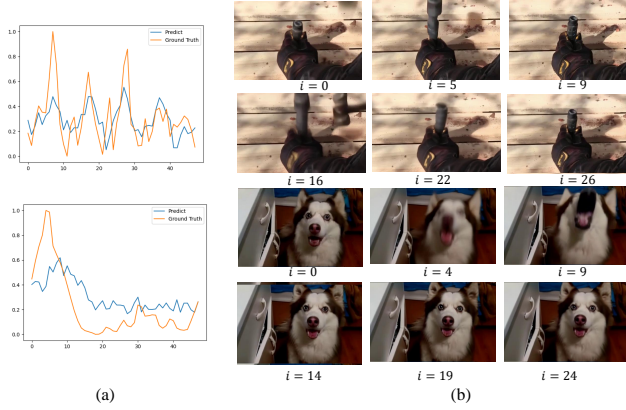


Figure 7. Visualization of (a) Predicted motion score from audio with the ground truth caluate from video data; and (b) the generated video keyframe by diffusion network described in Sec. 3.2.2 before interpolations.

G. Multimodal Classifier Free Guidance

Similar to Xing et al. [24], we introduce three guidance scales s_{img} , s_{txt} , and s_{aud} to extend video generation with additional audio control. These scales allow balancing the influence of different conditioning modalities in video generation. The modified noise estimation function is defined as:

$$\begin{aligned} \hat{\epsilon}_{\theta}(\mathbf{z}_t, \mathbf{c}_{\text{img}}, \mathbf{c}_{\text{txt}}, \mathbf{c}_{\text{aud}}) &= \epsilon_{\theta}(\mathbf{z}_t, \emptyset, \emptyset, \emptyset) \\ &+ s_{\text{img}} (\epsilon_{\theta}(\mathbf{z}_t, \mathbf{c}_{\text{img}}, \emptyset, \emptyset) - \epsilon_{\theta}(\mathbf{z}_t, \emptyset, \emptyset, \emptyset)) \\ &+ s_{\text{txt}} (\epsilon_{\theta}(\mathbf{z}_t, \mathbf{c}_{\text{img}}, \mathbf{c}_{\text{txt}}, \emptyset) - \epsilon_{\theta}(\mathbf{z}_t, \mathbf{c}_{\text{img}}, \emptyset, \emptyset)) \\ &+ s_{\text{aud}} (\epsilon_{\theta}(\mathbf{z}_t, \mathbf{c}_{\text{img}}, \mathbf{c}_{\text{txt}}, \mathbf{c}_{\text{aud}}) - \epsilon_{\theta}(\mathbf{z}_t, \mathbf{c}_{\text{img}}, \mathbf{c}_{\text{txt}}, \emptyset)). \end{aligned} \quad (4)$$

Here, \mathbf{c}_{img} , \mathbf{c}_{txt} , and \mathbf{c}_{aud} represent image, text, and audio conditioning, respectively. The newly introduced audio guidance scale s_{aud} enables the model to integrate temporal audio cues, ensuring synchronized motion generation in audio-reactive video synthesis. By adjusting these guidance parameters, we can control the relative impact of each modality in the final video output.

In experiment, we choose the audio guidance scale to 7.5 and image guidance scale to 2, for both keyframe generation network and frame interpolation network. As we add the audio guidance as a new feature, we compare the result from different audio guidance from 4.0 to 11.0 as list in Tab. 3. Although the higher audio guidance obtains a better audio synchronization score (RelSync and AlignSync) we finally choose the one with the best visual quality (FVD and FID) but still ahiveve compatible audio synchronization score.

s_{aud}	FID↓	FVD↓	AlignSync↑	RelSync↑
4.0	11.4	270.5	48.18	24.14
7.5	11.0	262.3	48.33	24.08
9.0	11.1	277.2	48.55	24.16
11.0	11.1	278.6	48.66	24.22

Table 3. Performance metrics for different guidance values.