MUSICMORPHER: MUSIC MORPHING VIA STYLE TOKENS

Anonymous authors
Paper under double-blind review

ABSTRACT

Continuous-strength instruction editing lets users specify what to change and by how much. Such methods provide this control for images, but remain underdeveloped for music because they require source-target examples or morph paths for supervision, which are hard to obtain. For example, image-based Kontinuous Kontext derives such trajectories from FreeMorph outputs, but requires post-filtering since interpolating zero-noise latents is non-smooth. We find image-based methods transfer poorly to music: they can act akin to a volume slider and ignore musical traits like tempo and beat alignment. We therefore identify music morphing, rather than editor fine-tuning, as the upstream bottleneck. We introduce MusicMorpher, a music morphing method that generates instruction-conditioned morph paths between a source and target. We do so by learning a per-song style token, applied through modulation so that it shifts global style without touching temporal structure, and interpolating it together with each song's inverted structural latent, each in its own space. At inference, we tempo- and beat-align songs before style-token computation to prevent rhythmic discontinuities. We compare MusicMorpher against SDEdit, RF-Inversion, FreeMorph, and FlowMorph on morph quality and smoothness across varying rhythmic alignment, introducing trajectory-level measures absent from prior single-sample metrics. Results show our method produces higher-quality, smoother morphs, and, without supervised instruct-edit training, extends to continuous-strength semantic edits reaching quality-alignment trade-offs beyond baseline sweeps; our analyses explain prior methods' inconsistent results. Examples at `https://musicmorpher.github.io/`.

1 INTRODUCTION

Recent advances in generative models have established instruction-based editing, which directly modifies the semantic attributes of content via natural language, as a new editing interface across images, audio, and music. Music is an especially demanding setting for this interface: it is experienced as a continuous temporal signal, so an edit must remain coherent throughout the piece rather than merely plausible at any single moment, a constraint that motivates our focus on music in this work. However, existing work has largely focused on specifying *what* should change, without treating *how strongly* an instruction should be applied as an explicit control axis. Recent studies have introduced continuous-strength editing (Parihar et al., 2026; Wolf et al., 2026; Zarei et al., 2026), which conditions on strength to enable continuous control over the transition from source to fully edited target. This axis, however, exposes an upstream problem: a strength value is only meaningful if it maps to a real point along a source-to-target trajectory, which first requires constructing a meaningful intermediate target between the two endpoints.

In the image domain, this intermediate morph path supervision is typically obtained via a three-stage pipeline: an instruction-driven editor produces a full-strength target from the source, an off-the-shelf morphing model (Cao et al., 2025; Zheng et al., 2026; Zhang et al., 2024a; Yang et al., 2024) synthesizes intermediate-strength samples toward it, and a continuous-strength editor is fine-tuned on the result. The first stage rests on a mature instruction-editing literature and the third is downstream of the first two, leaving the middle stage, the morphing trajectory itself, as the least explored and, as we argue below, the bottleneck of the pipeline. This bottleneck is far more severe for music: an intermediate image need only appear visually plausible between its two endpoints, whereas 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.

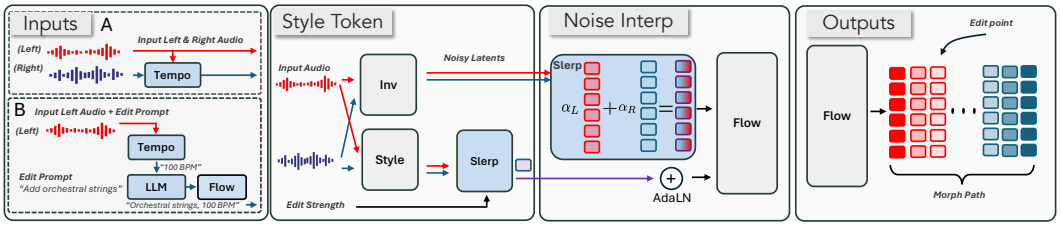

Figure 1: **Inference.** (Inputs) We input a pair of two audio examples (top) or a single input audio example and edit direction (bottom). (Style Token) We blend the style of two songs via spherical interpolation of their respective style tokens. (Noise Interp) We apply our style tokens via smooth AdaLN modulation and mitigate structure change between two songs via interpolating their (inverted) noisy latents. (Outputs) We output a trajectory (morph path) or single outputs.

is perceived sequentially, so any discontinuity in beat, tempo, or timbre is immediately audible as a structural breakdown rather than a subtle interpolation artifact.

Existing continuous-control methods for music do not resolve this problem either, as we detail in Section 2. We therefore treat music morphing as a self-contained upstream problem, deliberately separated from training a downstream continuous-strength instruct-editor (out of scope).

To this end, we introduce MusicMorpher, a new music morphing method that generates continuous-strength instruction-conditioned morphing without supervised instruct-edit training, as shown in Fig. 1. We learn a style token that carries a reference's global style; because it enters through modulation and is position-invariant, it can shift overall character but cannot encode temporal structure, which we leave to the latent. To morph between two songs at inference time, we (1) tempo- and beat-align each song to prevent rhythmic discontinuities, (2) compute and interpolate per-song style tokens to embody a new song *style*, (3) invert each song to obtain initial noise latents embodying *structure*, and (4) run flow sampling under the interpolated style-token conditioning while interpolating the noise latents at a high-noise level t_{inv}, all without inference-time optimization or attention-level intervention.

We evaluate against SDEdit (Meng et al., 2022), RF-Inversion (Rout et al., 2025), FreeMorph (Cao et al., 2025), and FlowMorph (Zheng et al., 2026) on morph quality, morph smoothness, and semantic editing. We also propose Morph Path Smoothness (MPS), since prior first-order smoothness metrics correlate poorly with human preference on trajectories. Results show our method achieves both better quality (lower FAD_{sum}) and better structural smoothness (lower MPS) than baselines on every condition and embedding backbone. This indicates that morphing is itself an important upstream problem for continuous-strength music editing, and that MusicMorpher can serve as a training-free primitive for it.

To summarize, our contributions are as follows:

- We introduce MusicMorpher, which generates semantically and structurally coherent music morphs by independently interpolating a semantic style token and structural latents.
- We propose Morph Path Smoothness (MPS), a metric quantifying how monotonically and linearly each intermediate sample moves away from the left endpoint in embedding space.
- We show MusicMorpher outperforms adapted image-morphing baselines in quality, semantic alignment, and smoothness across all evaluated conditions.

2 Related Works

2.1 Continuous-Strength Editing and Morphing-Based Supervision

A related line of work learns continuous attribute directions or sliders in GANs and diffusion models (Härkönen et al., 2020; Zhuang et al., 2021; Gandikota et al., 2024; 2025; Baumann et al., 2025; Ye et al., 2026; Zarei et al., 2026; Chiu et al., 2026; Liu et al., 2026). We build most directly on Kontinuous Kontext (Parihar et al., 2026), which instantiates the three-stage pipeline described above with Flux Kontext as the instruction-driven editor, drawn from a mature instruction-editing literature (Brooks et al., 2023; Zhang et al., 2023; Fu et al., 2024; Huang et al., 2024; Geng et al., 2024;

Kumari et al., 2026; Wu et al., 2025; Black Forest Labs et al., 2025; Liu et al., 2025; Wu et al., 2026), and FreeMorph as the morphing stage. Kontinuous Kontext itself notes that FreeMorph's (Cao et al., 2025) latent space is not semantically smooth, producing artifacts and abrupt transitions as an unrobust inference-time heuristic.

2.2 Image Morphing

Image morphing, the most direct ancestor of our work, synthesizes intermediate states between a source and a target atop a training-free line of inversion and attention-control editing (Meng et al., 2022; Hertz et al., 2023; Kulikov et al., 2025; Zhang et al., 2024a; Yang et al., 2024). We adopt two recent methods as baselines. FreeMorph (Cao et al., 2025) encodes the inputs into VAE space, spherically interpolates the clean latents at $t = 0$, and only then inverts and injects guidance by aggregating the endpoints' key and value features into self-attention under a step-dependent schedule, preserving identity and transition directionality; applied to audio, however, the approximate linearity of audio VAE latents (Bralios et al., 2025; Koops et al., 2026) means this interpolation step simply mixes the two unrelated songs. FlowMorph (Zheng et al., 2026) instead reparameterizes each latent neighborhood into a shape offset and a one-step semantic vector, optimizes them separately per input, and interpolates the offsets linearly and the vectors spherically, assuming a shared spatial coordinate system between inputs (layout alignment and a depth-conditioned backbone). Both methods thus assume content perceived in parallel over shared spatial coordinates; music, perceived sequentially, exposes discontinuities in beat, tempo, or timbre as audible structural breakdowns, and no shared coordinate system exists between temporally unaligned tracks, so primitives such as linear offset mixing or position-indexed attention blending lose meaning on the temporal axis and yield audio unreliable as training supervision.

2.3 Continuous Control and Morphing for Music

Unlike images, semantic morphing in music remains largely unexplored; existing continuous-control methods for music instead traverse a pre-defined or learned axis via activation steering, sparse-autoencoder concept manipulation, inference-time intervention, or training-free sliders extended to audio (Singh et al., 2026; Zhao et al., 2026; Koo et al., 2025; Ezra et al., 2025). Our problem differs not in performance but in definition: rather than traversing a given axis, we construct the axis itself from an open-vocabulary instruction. Because morph paths span source to target, a point on it yields a single-strength semantic edit as a by-product, so we examine this secondary capability below.

2.4 Text-Guided Music Editing

As background for our secondary evaluation of semantic editing, training-based approaches require paired data or retraining (Wang et al., 2023; Han et al.; Zhang et al., 2025). Closer to our setting, zero-shot methods edit genre, mood, and instrumentation by word swapping (MusicMagus (Zhang et al., 2024b)) or self-attention probing (Melodia (Yang et al., 2026)); these and a broader set of zero-shot approaches (Manor & Michaeli, 2024; Xu et al., 2024; Sioros et al., 2025; Ungersböck et al., 2025) produce only a single, cross-attention-conditioned edit per instruction rather than a morph trajectory.

3 Preliminaries

Our method builds on latent diffusion (Rombach et al., 2022), where data $\mathbf{x} \sim p_{\mathrm{data}}(\mathbf{x})$ is encoded to a latent $\mathbf{z}$ by a pre-trained variational autoencoder (Kingma & Welling, 2014) and corrupted as $\mathbf{z}_t = (1-t)\mathbf{z} + t\boldsymbol{\epsilon}$ with $\boldsymbol{\epsilon} \sim \mathcal{N}(0, I)$ under flow matching (Lipman et al., 2023). A diffusion transformer (Peebles & Xie, 2023) learns the velocity field $v_{\boldsymbol{\theta}}(\mathbf{z}_t, t, \mathbf{c})$, conditioned on an arbitrary signal $\mathbf{c}$, by minimizing $\mathbb{E}[\|v_{\boldsymbol{\theta}}(\mathbf{z}_t, t, \mathbf{c}) - (\boldsymbol{\epsilon} - \mathbf{z})\|_2^2]$, and generation integrates the ODE $\dot{\mathbf{z}}_t = v_{\boldsymbol{\theta}}(\mathbf{z}_t, t, \mathbf{c})$ from $t = 1$ (a Gaussian $\mathbf{z}_1$) to $t = 0$. Because this ODE is deterministic, it can also be run forward from a clean sample to yield a latent $\mathbf{z}_\tau$ from which reverse integration exactly recovers it, a pass referred to as *inversion* (Mokady et al., 2023; Rout et al., 2025).

Conditioning enters through a few pathways: with cross-attention, $\mathbf{c}$ retains its sequence structure and each latent token attends to it individually; with self-attention, $\mathbf{c}$ is concatenated to the tar-

get input tokens and masked from the output. With modulation, all global, sequence-independent conditions are instead collapsed into a single per-timestep vector $\mathbf{c}_t = \mathbf{e}_t + \mathbf{c}$ where $\mathbf{e}_t$ is a time embedding and $\mathbf{c}$ is the sum of the remaining global conditions. A per-block MLP g_ℓ then maps $\mathbf{c}_t$ to a channel-wise scale and shift applied after LayerNorm: $(\mathbf{a}_{\ell,t}, \mathbf{b}_{\ell,t}) = g_\ell(\mathbf{c}_t)$, $\text{AdaLN}_\ell(\mathbf{h}; \mathbf{c}_t) = \mathbf{a}_{\ell,t} \odot \text{LayerNorm}(\mathbf{h}) + \mathbf{b}_{\ell,t}$ (Perez et al., 2018; Peebles & Xie, 2023). Since $(\mathbf{a}_{\ell,t}, \mathbf{b}_{\ell,t})$ broadcasts identically across positions, modulation is best suited for style token.

4 METHOD

Our goal is to generate a sequence of music audio signals $\mathbf{A} = \{A_j\}_{j=0}^{J}$, with $A_0 = A_L$ and $A_J = A_R$, that morphs between two given input audio endpoints while remaining semantically and structurally aligned to both. The central difficulty is that style and structure are not cleanly separable: any single representation carries a mixture of the two. Rather than trying to disentangle them, we exploit an asymmetry in how conditioning enters the model. A global summary routed through modulation is position-invariant and can only shift a reference's overall character, whereas temporal structure, what distinguishes one moment from the next, can be represented only by the latent token sequence. We therefore carry style through a modulation token and leave structure to the latent, and interpolate along each independently.

We first tempo- and beat-align the two endpoints so that they describe a common musical time, a preprocessing step that lets every later interpolation operate on directly comparable inputs (Section 4.1). We then extract a single per-song style token that summarizes global style and interpolate it spherically to obtain a blended style (Section 4.2), and, independently, invert each endpoint into structural noise latents and interpolate those in their own space (Section 4.3). Finally, we run flow sampling under the interpolated style-token conditioning while interpolating the latents, producing the full morph path without inference-time optimization or attention-level intervention (Section 4.4).

4.1 TEMPO AND BEAT ALIGNMENT

Two tracks cannot be blended meaningfully until they share a common musical time, and this is exactly the problem a DJ solves before mixing two songs: they first match the tempo of the two tracks and then align their beats and downbeats, so that the mix lands on a single coherent pulse rather than two competing ones. We adopt the same principle as a preprocessing stage that precedes the morph itself. If the endpoints disagree on tempo or bar position, that disagreement is baked into whatever we later extract from them and no downstream interpolation can remove it: drum hits smear, harmonic changes conflict, and intermediate samples inherit conflicting bar positions.

We therefore rhythmically align the endpoints before extracting the style token and structural latent, using an off-the-shelf tempo and beat tracker (Böck et al., 2016; Davies & Böck, 2019) that estimates each track's tempo (BPM) along with its beat and downbeat markers. Beats are the moments when you tap your foot to a song, and downbeats are the first beat of each musical measure.

Obtaining aligned endpoints. Our morph procedure always operates on a pair of tempo- and beat-aligned endpoints A_L, A_R. We are indifferent to where the endpoint comes from, however, and support two input forms.

A pair of tracks. When both endpoints are given, we estimate the tempo and beat pattern of the left track A_L and right track A_R, and time-stretch A_R to match A_L's tempo (or equivalent matching). This removes the largest source of rhythmic disagreement: without it, the two endpoints may differ not only in style or instrumentation but in their underlying pulse. Tempo matching alone, however, is not sufficient, as two clips can share the same BPM yet still be misaligned if they begin at different positions in the bar. We therefore also time-shift the right (or left) track so that the downbeat and beat patterns coincide, and crop both tracks to a common length.

A track and an edit prompt. When only A_L and an edit direction are given, we first synthesize the right endpoint and then morph A_L toward it. We analyze A_L's tempo and beat pattern, convert the *edit prompt* into a *generation prompt* using the right source caption (provided directly or generated) and the estimated tempo via a standard text LLM, generate a synthetic target song slightly longer than the source, and beat-align and crop it. The synthesized A_R serves as an edit-direction exemplar,

Algorithm 1: MusicMorpher Algorithm

Input : Frozen DiT $v_{\boldsymbol{\theta}}$, endpoint left and optional right A_L, A_R, style encoder E_s, latent encoder E_v, interpolation steps J

Tempo and beat align inputs. If left only, generate right endpoint following Section 4.1.

$\mathbf{z}^L, \mathbf{z}^R = E_v(A_L), E_v(A_R)$

Solve $\dot{\mathbf{z}}_t^L = v_{\boldsymbol{\theta}}(\mathbf{z}_t^L, t, \emptyset)$ and $\dot{\mathbf{z}}_t^R = v_{\boldsymbol{\theta}}(\mathbf{z}_t^R, t, \emptyset)$ from $t = 0$ to $t = t_{\text{inv}}$ `// Invert endpoints`

$\mathbf{s}^L, \mathbf{s}^R \leftarrow E_s(\mathbf{z}^L), E_s(\mathbf{z}^R)$ `// Style tokens`

$\mathbf{s}^j \leftarrow \text{Slerp}(\mathbf{s}^L, \mathbf{s}^R, j/J)$ `// Interpolate style tokens`

$\mathbf{z}^j_{t_{\text{inv}}} \leftarrow \text{Slerp}(\mathbf{z}^L_{t_{\text{inv}}}, \mathbf{z}^R_{t_{\text{inv}}}, j/J)$ `// Interpolate latents`

Solve $\dot{\mathbf{z}}_t^j = v_{\boldsymbol{\theta}}(\mathbf{z}_t^j, t, \mathbf{c}_t^j)$ backward from $t = t_{\text{inv}}$ to $t = 0$

return $[\mathbf{z}^L, \mathbf{z}^1, \ldots, \mathbf{z}^{J-1}, \mathbf{z}^R]$

which lets a point on the resulting morph path act as a continuous-strength instruction edit without any instruct-edit training.

4.2 Semantic Style Token

A song's global style, meaning its performance, instrumentation, and the character of its mix, is largely a property of the whole clip rather than of any particular moment, so we summarize it into a single vector that emphasizes global style over temporal structure. Because this vector is global and position-invariant, we inject it through AdaLN modulation rather than attention: routing it through cross- or self-attention would entangle style with the per-block query, key, and value projections, whereas modulation applies it uniformly across the sequence and leaves the attention stream to carry the bulk of temporal and structural details. This mirrors findings in text-to-image transformers, where the modulation space is semantically editable and differences between modulation vectors act as edit directions (Garibi et al., 2025). Because the inverted latent is not itself style-free, this separation is one of emphasis, not a clean split; what makes it work is that during denoising the modulation signal steers the trajectory toward the interpolated style, so the token governs the resulting style while the latent anchors structure.

A single style token also makes interpolation natural. Blending two styles is just a matter of combining their tokens, and the normalization inside AdaLN dictates how that combination must be done. AdaLN normalizes each token by its own mean and standard deviation, so the normalized token $\hat{\mathbf{u}}$ has exactly zero mean and norm $\sqrt{d}$, not merely on average. Every token therefore lies on the same surface, a sphere of radius $\sqrt{d}$ sliced by a hyperplane through the origin. Linear interpolation leaves that surface, since the straight segment between two tokens cuts through the sphere's interior, and the subsequent affine step $\gamma \odot \hat{\mathbf{u}} + \beta$ then carries such a point off the surface real tokens occupy, giving a conditioning vector the model was never trained to read. We therefore blend the two tokens with spherical linear interpolation (slerp) (Shoemake, 1985; Karras et al., 2019), which traces the arc along the sphere rather than a straight chord through its interior:

$$\hat{\mathbf{u}}_\lambda = \frac{\sin((1-\lambda)\omega)}{\sin\omega}\hat{\mathbf{u}}_1 + \frac{\sin(\lambda\omega)}{\sin\omega}\hat{\mathbf{u}}_2, \qquad \mathbf{s}_\lambda = \gamma \odot \hat{\mathbf{u}}_\lambda + \beta,$$

which keeps every $\mathbf{s}_\lambda$ on the surface the endpoint tokens occupy, for any $0 \le \lambda \le 1$. Slerp is thus not a stylistic choice borrowed from prior morphing work but the path that keeps intermediate tokens inside the distribution the frozen model conditions on, a principle we apply again to the structural latent in Section 4.3 for a different underlying reason.

4.3 Structurally Aligned Latents

Structure is what distinguishes one moment of a piece from the next, and only the latent token sequence can represent it. The style token enters through modulation, whose scale and shift broadcast identically across every position, so it can carry only a global, position-invariant summary of a reference. Whatever structure it reflects is an average over the whole clip, never a temporal deviation: it cannot place a kick here and a snare there. The moment-to-moment structure that does place them lives in the latent, which is therefore where we act on structure. In diffusion models the high-noise

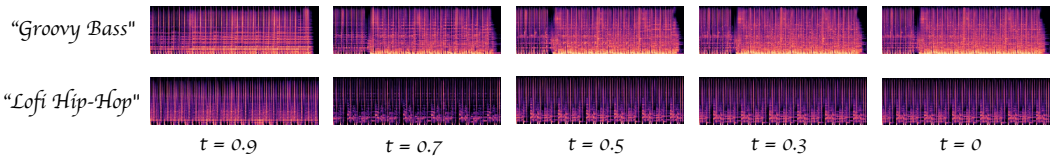

Figure 2: Generated music across denoising timesteps. Each panel shows the model's predicted clean latent decoded to a mel-spectrogram, where t is the denoising progress (t=1: pure noise, t=0: fully denoised). Left: the text prompt condition for each sample. The coarse spectrogram structure emerges early — by t=0.7 the estimate already closely resembles the final output at t=0.

steps fix global structure while later steps fill in detail, so inverting an endpoint only up to a noise level t_{inv} yields a latent that still carries its structural skeleton (Novack et al., 2024a;b; Mahajan et al., 2026; Luo et al., 2024), see Fig. 2. The level matters: our FreeMorph baseline instead interpolates in clean latent space at $t = 0$, which we later show forfeits structure and underlies its near-mixing behavior in Section 5.3. We therefore invert each endpoint to t_{inv}, obtaining latents $\mathbf{z}^L_{t_{\text{inv}}}, \mathbf{z}^R_{t_{\text{inv}}}$ where structure is exposed to interpolation.

What is specific to our method is that we interpolate this latent independently of the style token, each in its own space, so blending one never perturbs the other. We do so with slerp, as for the style token described in Section 4.2, but now for a distributional reason: a latent at high noise sits near an isotropic Gaussian, whose mass concentrates on a thin shell, and a straight segment between two such latents slips off that shell to a noise level the model never saw, whereas the arc stays on it. The same rule holds for both, for different reasons, the LayerNorm sphere for the token and the Gaussian shell for the latent. We keep t_{inv} high enough that structure has formed yet low enough that this geometry holds.

4.4 Complete Inference Algorithm

Given two tempo- and beat-aligned endpoint audios A_L and A_R, we encode each into a VAE latent (E_v). That latent then feeds two separate routes. For structure, we invert Rout et al. (2025) it to t_{inv}, yielding $\mathbf{z}^L_{t_{\text{inv}}}$ and $\mathbf{z}^R_{t_{\text{inv}}}$; for style, we pool it into a style token $\mathbf{s}^L$ and $\mathbf{s}^R$ using a style encoder (E_s) without any inversion. For each intermediate $j = 1, ..., J-1$, we interpolate within each route with slerp, $\mathbf{s}^j := \text{slerp}(\mathbf{s}^L, \mathbf{s}^R, \lambda = j/J)$ and $\mathbf{z}^j_{t_{\text{inv}}} = \text{slerp}(\mathbf{z}^L_{t_{\text{inv}}}, \mathbf{z}^R_{t_{\text{inv}}}, \lambda = j/J)$, then denoise from t_{inv} to 0 starting from $\mathbf{z}^j_{t_{\text{inv}}}$ with the style token $\mathbf{s}^j$, supplied through modulation. Together with the endpoints at $j = 0$ and $j = J$, this gives the full morph path $\mathbf{A}$ under the frozen model, see Algorithm 1.

4.5 Training Style Tokens

We visualize our training process in Fig. 3. We use a standard flow transformer text-to-music generation architecture and add our per-song style token. The style token processing consists of feeding variable-length VAE latents into a learned attention pooling layer, trained on-the-fly during flow model training, that summarizes the style of a song and is applied via AdaLN modulation. Its input content differs from the target flow inputs, which are noised, passed to the transformer, and used in the flow loss.

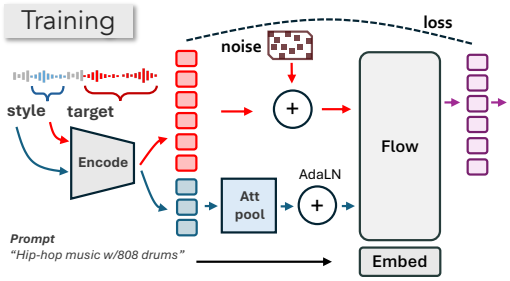

Figure 3: **Training.** From one song we take a style region and a target region. The style latents are attention-pooled into a single token, applied to the flow transformer via AdaLN, while the noised target latents drive a standard flow loss.

For the learned attention pooling, a single learned query $\mathbf{q}$ attends over the reference latent $\mathbf{z}^{\text{ref}} \in \{\mathbf{z}^L, \mathbf{z}^R\}$ through one multi-head cross-attention layer (Yu et al., 2022), $\mathbf{u} = \text{MHA}(\mathbf{q}, \mathbf{z}^{\text{ref}}, \mathbf{z}^{\text{ref}})$, pulling in information that reflects persistent stylistic factors, followed by an affine LayerNorm $\mathbf{s} = \text{LayerNorm}_{\gamma,\beta}(\mathbf{u})$ that places the pooled vector in the regime the model reads. This adapter is trained via that same target/reference pairing, reconstructing the target conditioned on the reference; the setup follows

Table 1: Comparison of the three morphing methods under the default Tempo & Beat setting, across three embedding backbones. Our method attains the best score on both morphing quality ($FAD_{sum}\downarrow$) and path smoothness (MPS, reported via its representative statistic MAR↓) in every backbone. Full results across all backbones and alignment conditions (including MSR) are given in Appendix F.

	$FAD_{sum}\downarrow$			MAR↓		
Method	LAION-CLAP	MuQ	MusicFM	LAION-CLAP	MuQ	MusicFM
FreeMorph	<u>0.9638</u>	<u>131.3430</u>	<u>122.3049</u>	<u>0.1226</u>	0.1265	<u>0.0907</u>
FlowMorph	2.8445	212.3344	429.9906	0.1549	<u>0.1238</u>	0.1166
MusicMorpher (Ours)	**0.4914**	**41.2353**	**46.9421**	**0.0968**	**0.0934**	**0.0664**

DiffRhythm (Ning et al., 2025), but replaces its LSTM final-state pooling with query-based attention pooling. Compressing the reference into a single order-independent vector makes it harder to copy fine-grained temporal content verbatim, although the token may still carry over some musical content. See Appendix A for further details.

5 Experiments

5.1 Experimental Details

Overview. We implement our method on top of a standard latent-flow transformer (Peebles & Xie, 2023), fixing the number of interpolation steps to $J = 10$ and the noise-caching timestep to $t_{\text{inv}} = 0.7$; Tempo & Beat aligned is the default condition unless stated otherwise. For evaluation, an LLM generates 4,500 text-prompt pairs spanning diverse genres, styles, and moods; the first prompt of each pair yields the left endpoint, and the right endpoint is generated under each of three alignment conditions between the endpoints, Tempo & Beat (both matched), Tempo (tempo matched only), and None (no alignment), for $4{,}500 \times 3 = 13{,}500$ morph paths total. We compare against two image-morphing methods, FreeMorph (Cao et al., 2025) and FlowMorph (Zheng et al., 2026), and two semantic-editing methods SDEdit (Meng et al., 2022) and RF-Inversion (Rout et al., 2025), all reimplemented under the same backbone, scheduler, and decoder for fairness. For simplicity, we measure morphing quality with FAD_{sum}: for each intermediate A_j $(j = 1, ..., J-1)$ we compute its FAD against the two endpoints $\{A_L, A_R\}$ as the reference set, and sum these $J-1$ FAD values. Near $\lambda = 0$ or 1 an intermediate almost coincides with an endpoint, so its FAD is tiny; summing across the path is what captures the overall trend rather than these near-trivial extremes. Because it sums $J - 1$ per-point FADs, FAD_{sum} is large in absolute magnitude, so only relative comparisons across methods, computed identically, are meaningful. Crucially, the reference endpoints are themselves generated by our base model rather than real audio. This is a deliberate choice: against real references, a large FAD_{sum} has two indistinguishable causes, a base model that generates poorly and a morph that interpolates poorly, and the metric cannot separate them. Generating the endpoints from the same model holds generation quality fixed on both sides, so FAD_{sum} varies only with the morph. We report it across the LAION-CLAP, MuQ, and MusicFM backbones (full details in Appendix F).

Morph Path Smoothness (MPS). We further propose Morph Path Smoothness (MPS), a new metric that quantifies the smoothness of a morph path. The first-order smoothness used in the image domain (Parihar et al., 2026) measures only the largest change between two adjacent points; in preliminary experiments, we found that rankings based on this metric correlate poorly with human preference. We therefore start from two properties a smooth path should satisfy: the distance from the left endpoint should increase monotonically, and the distance between the two endpoints should serve as its upper bound; a path satisfying both traces a straight line in normalized distance-to-left. Concretely, we average each path's normalized distance-to-left curve across the evaluation set, smooth it, and fit a least-squares line to this empirical mean curve. We then measure, as a residual, how far each individual morph point's normalized distance departs from this fitted line, and call the mean of its absolute value and the mean of its square **MAR** and **MSR**, respectively, see Appendix E. Since the two statistics reproduce the same ranking under every condition in Appendix F, we report MAR as the representative value.

5.2 Quantitative Results

Morphing quality and smoothness. Table 1 shows that our method attains the lowest FAD_{sum} and MAR across all three backbones. Because a summed score could in principle come from a few favorable operating points, Fig. 4 breaks FAD down over the morph strength λ: our method stays below both baselines at every λ, not only in aggregate. FreeMorph and ours peak near $\lambda = 0.5$, where the two clips are mixed most evenly and the morph is hardest to keep realistic, and the ordering holds throughout: our summed FAD is roughly a third of FreeMorph's and nearly an order of magnitude below FlowMorph's.

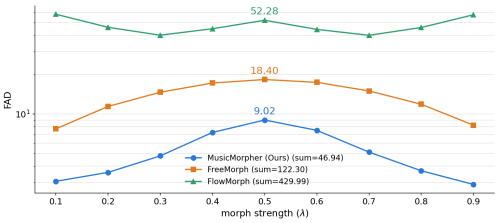

Figure 4: FAD across morph strength λ. Our method stays below both baselines at every λ; FreeMorph and ours peak at $\lambda = 0.5$. Legend gives the summed FAD.

Semantic Editing. Figure 5 evaluates semantic editing obtained for free from a midpoint of the morph path, as a trade-off between audio quality (FAD) and edit-direction alignment (CLAP directional similarity). We sweep each baseline along its natural strength axis: noise strength for SDEdit and RF-Inversion, edit-target strength λ for FreeMorph. In every case higher CLAP alignment comes at higher FAD; FlowMorph is excluded as its FAD exceeds the plot range. In contrast, our method reaches quality-alignment trade-offs beyond any baseline's sweep, all without instruct-edit training.

5.3 Qualitative Results

FreeMorph reduces to naive mixing. The metrics above do not explain *why* the baselines fail; we now show it directly. A real morph should produce intermediates that go beyond a trivial blend of the two endpoints. We test this with a mixture-divergence measure. At each strength λ, we ask how well the generated intermediate can be reproduced by simply blending the two endpoints' log-mel spectrograms: we take the best-matching blend over all ratios β and measure the residual ℓ_2 distance, $\text{minMel}(\lambda)$. A smaller value means the output is closer to a plain linear mix of the two songs. Figure 6 shows that FreeMorph sits far closer to this mixing path than ours across the entire range of λ, and the gap is widest in the middle, where a genuine morph should depart most from a blend. Notably, this holds even under Tempo & Beat alignment, the setting most favorable to morphing: even when handed two rhythmically matched endpoints, it largely just mixes them rather than synthesizing a new intermediate.

This mixture-like behavior is audible as rhythmic breakdown. Mixture divergence shows FreeMorph stays close to a linear mix, but not what that sounds like. We examine this with the *onset-strength* curve, which traces how strongly percussive events occur over time. Fig. 7 makes the equivalence visual on an unaligned pair, where the two songs have distinct rhythms. FreeMorph's onset curve is nearly indistinguishable from the naive interpolation, third versus fourth column: the source's onsets and the target's onsets both survive, each in its own place, so FreeMorph simply

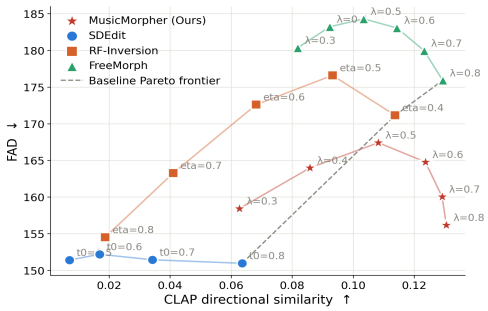

Figure 5: Semantic editing as a by-product of morphing, as a quality (FAD) versus edit-alignment (CLAP directional similarity) trade-off. Our method reaches points beyond every baseline's Pareto frontier.

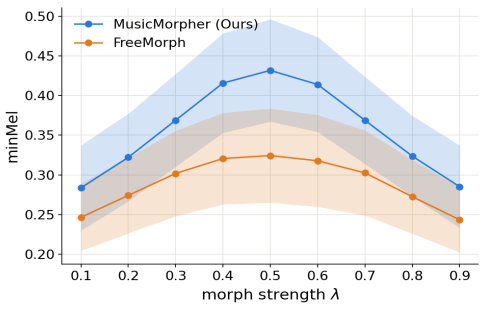

Figure 6: Mixture divergence $\text{minMel}(\lambda)$ vs. morph strength, averaged over the eval subset (shaded: ± 1 std). FreeMorph stays closer to the linear-mixing path across λ, while ours departs from it in the middle.

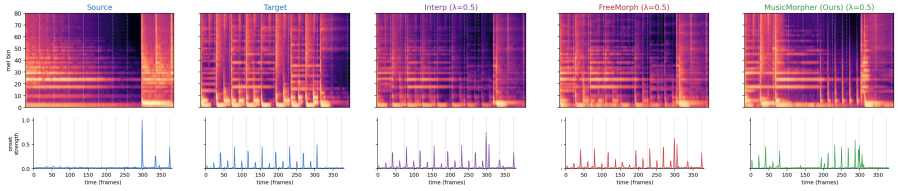

Figure 7: Source, target, and three $\alpha = 0.5$ midpoints of naive interpolation, FreeMorph, and ours, on an unaligned pair. Onset-strength curves below each spectrogram; dashed lines mark source beats. FreeMorph's onset curve nearly matches the naive interpolation, overlaying both rhythms without combining them, while ours forms a new rhythm matching neither.

lays one rhythm over the other without combining them. Ours looks nothing like the interpolation: its onset curve is neither endpoint's, nor their sum, but a new pattern, showing that our midpoint synthesizes a single rhythm rather than superposing two.

5.4 Ablation Studies

Both components matter. We now remove each of our own two design components, the semantic style token and the structural-latent (noise) interpolation, and measure how MAR degrades under the default Tempo & Beat Aligned setting. Table 2 shows that both components matter, but by markedly different amounts. Removing the style token has the larger overall impact: MAR degrades by 16% (MuQ) to 35% (MusicFM) across every backbone even though the structural-latent interpolation is left intact. Removing the noise interpolation instead degrades MAR by 10% (LAION-CLAP) to 19% (MuQ)—less than half the effect of the style token, but consistent in direction across all three backbones. Our smoothness gains thus come from two independent components contributing at different magnitudes.

Table 2: Component ablation, measured by MAR↓ (Tempo & Beat). Removing either component degrades smoothness; the style token matters more, consistently across backbones.

Configuration	LAION-CLAP	MuQ	MusicFM
Ours (full)	**0.0968**	**0.0934**	**0.0664**
w/o style token	0.1181	0.1079	0.0896
w/o noise interp.	0.1063	0.1116	0.0783

Robustness to input alignment. Tempo & Beat alignment has been our default setting so far, so here we ask how much performance varies it across our alignment configurations (Tempo & Beat, Tempo, and None). Since two rhythmically mismatched songs cannot be blended without collision, we track it with FAD_{sum} in Table 3. Our method attains the lowest FAD_{sum} under every condition. Absolute scores rise as alignment weakens, as expected, but ours degrades far more gently than the baselines: from Tempo & Beat to None, FAD_{sum} grows by 15% for us against 27% for FreeMorph. Our tempo- and beat-alignment stage removes rhythmic disagreement before interpolation, so even unaligned inputs are brought onto a common grid rather than blended in conflict.

Table 3: Robustness across endpoint alignment, FAD_{sum} on MusicFM embedding. Ours wins under every condition and degrades gently as alignment weakens.

	Alignment		
Method	T&B	Tempo	None
FreeMorph	122.30	129.77	168.61
FlowMorph	429.99	433.16	443.72
MusicMorpher (Ours)	**46.94**	**47.71**	**55.11**

6 Conclusion

We present MusicMorpher, a method that utilizes style tokens and structurally aligned latents to produce a coherent intermediate trajectory or morph path for music. We find that our semantic style tokens that are conditioned in modulation space along with inverted latents can be both independently interpolated to accomplish this goal. Additionally, we provide a metric to measure the overall smoothness of a morph sequence, called Morph Path Smoothness (MPS) and perform within and across method comparison for both morphing and semantic editing. Results show our work outperforms previous known image based morphing methods adapted for music morphing both in terms of overall quality and morph smoothness. We believe our method can serve as a data pipeline to train a general music editing model as well as provide a baseline continuous-strength instruction edit method without the need for supervised instruct-edit training.

AI USE STATEMENT

We used generative AI tools for agentic coding, paper writing, paper editing (to reduce length), create synthetic captions for our evaluation dataset across broad styles, assist with translation, search for related work, implement methods, support qualitative and thematic data analysis, and interpret results. We have not used generative AI tools to help develop theoretical models or conceptual frameworks, formulate mathematical claims, provide critical ingredients for proving mathematical claims, assist in the writing of proofs, propose or refine hypotheses, design or provide feedback on research methodology or experiments, clean and reformat dataset and the rest of the required disclosure tasks are not applicable to this work. We have reviewed all AI-assisted work including both code correctness, dataset caption, and all paper editing.

ETHICS STATEMENT

We are focused on advancing human creative control via editing and controllable generation tools with the goal of building tools useful for musicians. For training, we use licensed, internal data.

REPRODUCIBILITY STATEMENT

We cannot release our code, model checkpoints, or training data as our base model is trained on licensed data. To keep our results as reproducible as possible, however, we specify our method as an explicit, deterministic algorithm (Algorithm 1) built from off-the-shelf, publicly documented components (a pretrained VAE, an existing tempo and beat tracker (Böck et al., 2016; Davies & Böck, 2019), and standard flow-matching inversion), and give full base-model architecture and training details in Appendix A. Because our morphing baselines, FreeMorph and FlowMorph, were designed for image latents with no existing audio or music implementation, we reimplemented both ourselves for a text-to-music backbone; Appendix B documents this reimplementation in full so others adapting image-based morphing methods to audio can verify or reproduce our porting choices. We also fully specify our proposed metric (Morph Path Smoothness; Appendix E) and our hyperparameter selection procedure (Appendix C), so the evaluation protocol itself is reproducible on any comparable model or dataset, independent of our specific trained model. We intend to release the MPS scoring implementation itself, independent of our model and data release restrictions, so it can be applied directly to other morphing methods and datasets.

REFERENCES

Stefan Andreas Baumann, Felix Krause, Michael Neumayr, Nick Stracke, Melvin Sevi, Vincent Tao Hu, and Björn Ommer. Continuous, subject-specific attribute control in T2I models by identifying semantic directions. In *Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2025.

Black Forest Labs, Stephen Batifol, Andreas Blattmann, Frederic Boesel, Saksham Consul, Cyril Diagne, Tim Dockhorn, Jack English, Zion English, Patrick Esser, et al. FLUX.1 Kontext: Flow matching for in-context image generation and editing in latent space. *arXiv preprint arXiv:2506.15742*, 2025.

Sebastian Böck, Filip Korzeniowski, Jan Schlüter, Florian Krebs, and Gerhard Widmer. madmom: A new Python audio and music signal processing library. In *Proceedings of the 24th ACM international conference on Multimedia*, 2016.

Dimitrios Bralios, Paris Smaragdis, and Jonah Casebeer. Learning to upsample and upmix audio in the latent domain. In *Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*. IEEE, 2025.

Tim Brooks, Aleksander Holynski, and Alexei A Efros. InstructPix2Pix: Learning to follow image editing instructions. In *Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2023.

Yukang Cao, Chenyang Si, Jinghao Wang, and Ziwei Liu. FreeMorph: Tuning-free generalized image morphing with diffusion model. In *International Conference on Computer Vision (ICCV)*. IEEE, 2025.

Jonah Casebeer, Ge Zhu, Zhepei Wang, and Nicholas J Bryan. A generative-first neural audio autoencoder. In *International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2026.

Pin-Yen Chiu, I-Sheng Fang, and Jun-Cheng Chen. Text Slider: Efficient and plug-and-play continuous concept control for image/video synthesis via LoRA adapters. In *Winter Conference on Applications of Computer Vision (WACV)*, 2026.

Tri Dao, Dan Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. FlashAttention: Fast and memory-efficient exact attention with IO-awareness. *Neural Information Processing Systems (NeurIPS)*, 2022.

Matthew E. P. Davies and Sebastian Böck. Temporal convolutional networks for musical audio beat tracking. In *2019 27th European Signal Processing Conference (EUSIPCO)*. IEEE, 2019.

Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorenz, Axel Sauer, Frederic Boesel, et al. Scaling rectified flow transformers for high-resolution image synthesis. In *International Conference on Machine Learning (ICML)*, 2024.

Rotem Ezra, Hedi Zisling, Nimrod Berman, Ilan Naiman, Alexey Gorkor, Liran Nochumsohn, Eliya Nachmani, and Omri Azencot. FreeSliders: Training-free, modality-agnostic concept sliders for fine-grained diffusion control in images, audio, and video. *arXiv preprint arXiv:2511.00103*, 2025.

Tsu-Jui Fu, Wenze Hu, Xianzhi Du, William Wang, Yinfei Yang, and Zhe Gan. Guiding instruction-based image editing via multimodal large language models. In *International Conference on Learning Representations (ICLR)*, 2024.

Rohit Gandikota, Joanna Materzyńska, Tingrui Zhou, Antonio Torralba, and David Bau. Concept sliders: LoRA adaptors for precise control in diffusion models. In *European Conference on Computer Vision (ECCV)*. Springer, 2024.

Rohit Gandikota, Zongze Wu, Richard Zhang, David Bau, Eli Shechtman, and Nick Kolkin. SliderSpace: Decomposing the visual capabilities of diffusion models. In *International Conference on Computer Vision (ICCV)*. IEEE, 2025.

Daniel Garibi, Shahar Yadin, Roni Paiss, Omer Tov, Shiran Zada, Ariel Ephrat, Tomer Michaeli, Inbar Mosseri, and Tali Dekel. TokenVerse: Versatile multi-concept personalization in token modulation space. *ACM Transactions On Graphics (TOG)*, 2025. doi: 10.1145/3730843.

Zigang Geng, Binxin Yang, Tiankai Hang, Chen Li, Shuyang Gu, Ting Zhang, Jianmin Bao, Zheng Zhang, Houqiang Li, Han Hu, et al. InstructDiffusion: A generalist modeling interface for vision tasks. In *Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2024.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. The Llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

Bing Han, Junyu Dai, Weituo Hao, Xinyan He, Dong Guo, Jitong Chen, Yuxuan Wang, Yanmin Qian, and Xuchen Song. InstructME: An instruction guided music edit framework with latent diffusion models. In *Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence (IJCAI)*.

Erik Härkönen, Aaron Hertzmann, Jaakko Lehtinen, and Sylvain Paris. GANSpace: Discovering interpretable GAN controls. In *Neural Information Processing Systems (NeurIPS)*, 2020.

Amir Hertz, Ron Mokady, Jay Tenenbaum, Kfir Aberman, Yael Pritch, and Daniel Cohen-Or. Prompt-to-prompt image editing with cross-attention control. In *International Conference on Learning Representations (ICLR)*, 2023.

Yuzhou Huang, Liangbin Xie, Xintao Wang, Ziyang Yuan, Xiaodong Cun, Yixiao Ge, Jiantao Zhou, Chao Dong, Rui Huang, Ruimao Zhang, et al. SmartEdit: Exploring complex instruction-based image editing with multimodal large language models. In *Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2024.

Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2019.

Diederik P. Kingma and Max Welling. Auto-encoding variational bayes. In *International Conference on Learning Representations (ICLR)*, 2014.

Junghyun Koo, Gordon Wichern, François G Germain, Sameer Khurana, and Jonathan Le Roux. SMITIN: Self-monitored inference-time intervention for generative music transformers. *IEEE Open Journal of Signal Processing*, 2025. doi: 10.1109/OJSP.2025.3534686.

Hendrik Vincent Koops, Hao Hao Tan, and Elio Quinton. On the geometry of music bandwidth extension in latent spaces of audio codecs. In *International Society for Music Information Retrieval Conference (ISMIR)*, 2026.

Vladimir Kulikov, Matan Kleiner, Inbar Huberman-Spiegelglas, and Tomer Michaeli. FlowEdit: Inversion-free text-based editing using pre-trained flow models. In *International Conference on Computer Vision (ICCV)*. IEEE, 2025.

Nupur Kumari, Sheng-Yu Wang, Nanxuan Zhao, Yotam Nitzan, Yuheng Li, Krishna Kumar Singh, Richard Zhang, Eli Shechtman, Jun-Yan Zhu, and Xun Huang. Learning an image editing model without image editing pairs. In *International Conference on Learning Representations (ICLR)*, 2026.

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matthew Le. Flow matching for generative modeling. In *International Conference on Learning Representations (ICLR)*, 2023.

Shiyu Liu, Yucheng Han, Peng Xing, Fukun Yin, Rui Wang, Wei Cheng, Jiaqi Liao, Yingming Wang, Honghao Fu, Chunrui Han, et al. Step1X-Edit: A practical framework for general image editing. *arXiv preprint arXiv:2504.17761*, 2025.

Xingchao Liu, Chengyue Gong, and Qiang Liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. In *International Conference on Learning Representations (ICLR)*, 2023.

Zhixuan Liu, Peter Schaldenbrand, Yijun Li, Long Mai, Aniruddha Mahapatra, Cusuh Ham, Jean Oh, and Jui-Hsien Wang. TokenDial: Continuous attribute control for text-to-video generation in visual dial space. *arXiv preprint arXiv:2603.27520*, 2026.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations (ICLR)*, 2019.

Grace Luo, Trevor Darrell, Oliver Wang, Dan B Goldman, and Aleksander Holynski. Readout guidance: Learning control from diffusion features. In *Computer Vision and Pattern Recognition (CVPR)*, 2024.

Jay Mahajan, Chang Liu, Rauf Makharov, Viraj Shah, Alexander Schwing, and Svetlana Lazebnik. Readit guidance: Control for image and video generation using diffusion transformer features. *arXiv preprint arXiv:2609.04649*, 2026.

Hila Manor and Tomer Michaeli. Zero-shot unsupervised and text-based audio editing using DDPM inversion. In *International Conference on Machine Learning (ICML)*, PMLR, 2024.

Chenlin Meng, Yutong He, Yang Song, Jiaming Song, Jiajun Wu, Jun-Yan Zhu, and Stefano Ermon. SDEdit: Guided image synthesis and editing with stochastic differential equations. In *International Conference on Learning Representations (ICLR)*, 2022.

Ron Mokady, Amir Hertz, Kfir Aberman, Yael Pritch, and Daniel Cohen-Or. Null-text inversion for editing real images using guided diffusion models. In *Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2023.

Ziqian Ning, Huakang Chen, Yuepeng Jiang, Chunbo Hao, Guobin Ma, Shuai Wang, Jixun Yao, and Lei Xie. DiffRhythm: Blazingly fast and embarrassingly simple end-to-end full-length song generation with latent diffusion. *arXiv preprint arXiv:2503.01183*, 2025.

Zachary Novack, Julian McAuley, Taylor Berg-Kirkpatrick, and Nicholas J. Bryan. DITTO: Diffusion inference-time t-optimization for music generation. In *International Conference on Machine Learning (ICML)*, PMLR, 2024a.

Zachary Novack, Julian McAuley, Taylor Berg-Kirkpatrick, and Nicholas J. Bryan. DITTO-2: Distilled diffusion inference-time t-optimization for music generation. In *International Society for Music Information Retrieval Conference (ISMIR)*, 2024b.

Rishubh Parihar, Or Patashnik, Daniil Ostashev, Venkatesh Babu Radhakrishnan, Daniel Cohen-Or, and Kuan-Chieh Jackson Wang. Kontinuous Kontext: Continuous strength control for instruction-based image editing. In *Computer Vision and Pattern Recognition (CVPR)*, 2026.

William Peebles and Saining Xie. Scalable diffusion models with transformers. In *International Conference on Computer Vision (ICCV)*, 2023.

Ethan Perez, Florian Strub, Harm De Vries, Vincent Dumoulin, and Aaron Courville. FiLM: Visual reasoning with a general conditioning layer. In *Association for the Advancement of Artificial Intelligence (AAAI)*, 2018.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Computer Vision and Pattern Recognition (CVPR)*, 2022.

Litu Rout, Yujia Chen, Nataniel Ruiz, Constantine Caramanis, Sanjay Shakkottai, and Wen-Sheng Chu. Semantic image inversion and editing using rectified stochastic differential equations. In *International Conference on Learning Representations (ICLR)*, 2025.

Noam Shazeer. GLU variants improve transformer. *arXiv preprint arXiv:2002.05202*, 2020.

Ken Shoemake. Animating rotation with quaternion curves. In *Proceedings of the 12th annual conference on Computer graphics and interactive techniques*, 1985.

Nikhil Singh, Manuel Cherep, and Pattie Maes. Discovering and steering interpretable concepts in large generative music models. In *International Conference on Learning Representations (ICLR)*, 2026.

Vassilis Sioros, Alexandros Potamianos, and Giorgos Paraskevopoulos. EditGen: Harnessing cross-attention control for instruction-based auto-regressive audio editing. *arXiv preprint arXiv:2507.11096*, 2025.

Jianlin Su, Murtadha Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. RoFormer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 2024.

Michael Ungersböck, Florian Grötschla, Luca Lanzendörfer, June Young Yi, Changho Choi, and Roger Wattenhofer. SAO-Instruct: Free-form audio editing using natural language instructions. In *Neural Information Processing Systems (NeurIPS)*, 2025.

Yuancheng Wang, Zeqian Ju, Xu Tan, Lei He, Zhizheng Wu, Jiang Bian, et al. AUDIT: Audio editing by following instructions with latent diffusion models. In *Neural Information Processing Systems (NeurIPS)*, 2023.

Alon Wolf, Chen Katzir, Kfir Aberman, and Or Patashnik. Continuous control of editing models via adaptive-origin guidance. In *Special Interest Group on Computer Graphics Interactive Techniques (SIGGRAPH)*, 2026. doi: 10.1145/3799902.3811077.

Minz Won, Yun-Ning Hung, and Duc Le. A foundation model for music informatics. In *International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2024.

Chenfei Wu, Jiahao Li, Jingren Zhou, Junyang Lin, Kaiyuan Gao, Kun Yan, Sheng-ming Yin, Shuai Bai, Xiao Xu, Yilei Chen, et al. Qwen-Image technical report. *arXiv preprint arXiv:2508.02324*, 2025. The Qwen-Image-Edit model is released as part of the Qwen-Image family.

Chenyuan Wu, Jiahao Wang, Pengfei Zheng, Ruiran Yan, Shitao Xiao, Xin Luo, Yueze Wang, Wanli Li, Xiyan Jiang, Yexin Liu, Junjie Zhou, Ziyi Xia, Ze Liu, Chaofan Li, Haoge Deng, Kun Luo, Bo Zhang, Jiajun Zhang, Dong Liu, Defu Lian, Xinlong Wang, Zhongyuan Wang, Tiejun Huang, and Zheng Liu. OmniGen2: Towards instruction-aligned multimodal generation. In *Computer Vision and Pattern Recognition (CVPR)*, 2026.

Yusong Wu, Ke Chen, Tianyu Zhang, Yuchen Hui, Taylor Berg-Kirkpatrick, and Shlomo Dubnov. Large-scale contrastive language-audio pretraining with feature fusion and keyword-to-caption augmentation. In *International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2023.

Manjie Xu, Chenxing Li, Duzhen Zhang, Dan Su, Wei Liang, and Dong Yu. Prompt-guided precise audio editing with diffusion models. In *International Conference on Machine Learning (ICML)*, PMLR, 2024.

Yi Yang, Haowen Li, Tianxiang Li, Boyu Cao, Xiaohan Zhang, Liqun Chen, and Qi Liu. Melodia: Training-free music editing guided by attention probing in diffusion models. In *Association for the Advancement of Artificial Intelligence (AAAI)*, 2026.

Zhaoyuan Yang, Zhengyang Yu, Zhiwei Xu, Jaskirat Singh, Jing Zhang, Dylan Campbell, Peter Tu, and Richard Hartley. IMPUS: Image morphing with perceptually-uniform sampling using diffusion models. In *International Conference on Learning Representations (ICLR)*, 2024.

Weixin Ye, Hongguang Zhu, Wei Wang, Yahui Liu, Mengyu Wang, and Xuecheng Nie. All-in-one slider for attribute manipulation in diffusion models. In *Computer Vision and Pattern Recognition (CVPR)*, 2026.

Jiahui Yu, Zirui Wang, Vijay Vasudevan, Legg Yeung, Mojtaba Seyedhosseini, and Yonghui Wu. CoCa: Contrastive captioners are image-text foundation models. *Transactions on Machine Learning Research (TMLR)*, 2022.

Arman Zarei, Samyadeep Basu, Mobina Pournemat, Sayan Nag, Ryan A Rossi, and Soheil Feizi. SliderEdit: Continuous image editing with fine-grained instruction control. In *Computer Vision and Pattern Recognition (CVPR)*, 2026.

Biao Zhang and Rico Sennrich. Root mean square layer normalization. *Neural Information Processing Systems (NeurIPS)*, 32, 2019.

Kai Zhang, Lingbo Mo, Wenhu Chen, Huan Sun, and Yu Su. MagicBrush: A manually annotated dataset for instruction-guided image editing. 2023.

Kaiwen Zhang, Yifan Zhou, Xudong Xu, Bo Dai, and Xingang Pan. DiffMorpher: Unleashing the capability of diffusion models for image morphing. In *Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2024a.

Yixiao Zhang, Yukara Ikemiya, Gus Xia, Naoki Murata, Marco A Martínez-Ramírez, Wei-Hsiang Liao, Yuki Mitsufuji, and Simon Dixon. MusicMagus: Zero-shot text-to-music editing via diffusion models. In *International Joint Conference on Artificial Intelligence (IJCAI)*, 2024b.

Yixiao Zhang, Yukara Ikemiya, Woosung Choi, Naoki Murata, Marco A. Martínez-Ramírez, Liwei Lin, Gus Xia, Wei-Hsiang Liao, Yuki Mitsufuji, and Simon Dixon. Instruct-MusicGen: Unlocking text-to-music editing for music language models via instruction tuning. In *International Society for Music Information Retrieval Conference (ISMIR)*, 2025.

Daniel Zhao, Daniel Beaglehole, Taylor Berg-Kirkpatrick, Julian McAuley, and Zachary Novack. Steering autoregressive music generation with recursive feature machines. In *International Conference on Learning Representations (ICLR)*, 2026.

Yan Zheng, Yi Yang, Lanqing Guo, and Zhangyang Wang. FlowMorph: Revealing an optimizable flow latent space for controlled image morphing. In *Winter Conference on Applications of Computer Vision (WACV)*, 2026.

Haina Zhu, Yizhi Zhou, Hangting Chen, Jianwei Yu, Ziyang Ma, Rongzhi Gu, Yi Luo, Wei Tan, and Xie Chen. MuQ: Self-supervised music representation learning with mel residual vector quantization. *Transactions on Audio, Speech and Language Processing (TASLP)*, 2025.

Peiye Zhuang, Sanmi Koyejo, and Alex Schwing. Enjoy Your Editing: Controllable GANs for Image Editing via Latent Space Navigation. In *International Conference on Learning Representations (ICLR)*, 2021.

APPENDIX

This appendix is structured as follows: in Sec. A we discuss our base model implementation, in Sec. B we discuss our implementation for our morphing baselines, in Sec. C we discuss how we selected our hyperparameters for our method and the baselines, in Sec. D we conduct an ablation for our our prior baseline, FreeMorph, in Sec. E we provide an algorithm for our morphing metric, morph path smoothness (MPS), and in Sec. F we provide our full quantitative results.

A BASE MODEL PRETRAINING DETAILS

This appendix summarizes the pretraining setup of the base model. The model is a latent generative model whose generator is a diffusion transformer, v_θ, trained via conditional flow matching in the latent space of a frozen stereo audio autoencoder.

Audio autoencoder We represent audio in the continuous latent space of a frozen music autoencoder (Casebeer et al., 2026). The autoencoder encodes $44.1\,\mathrm{kHz}$ stereo waveforms as sequences of $d_z = 128$-dimensional latent frames at a rate of approximately $12.3\,\mathrm{Hz}$ and decodes these sequences back to waveforms. Consequently, each $32\,\mathrm{s}$ training crop is represented by 394 latent frames.

Generative objective We train a velocity field v_θ in this latent space using conditional flow matching along the straight probability paths of rectified flow (Liu et al., 2023; Lipman et al., 2023). Let $\mathbf{x}_0$ denote a clean latent sequence drawn from the data distribution, and let $\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$. The interpolation path is

$$\mathbf{x}_t = (1-t)\mathbf{x}_0 + t\mathbf{z}, \qquad t \in [0,1], \tag{1}$$

such that $t = 0$ corresponds to clean data and $t = 1$ to Gaussian noise. Given conditioning information $\mathbf{c}$, the model is trained to predict the constant velocity $\mathbf{z} - \mathbf{x}_0$:

$$\mathcal{L}_{\mathrm{CFM}}(\theta) = \mathbb{E}_{\mathbf{x}_0,\mathbf{z},t}\left[\left\|v_\theta(\mathbf{x}_t, t, \mathbf{c}) - (\mathbf{z} - \mathbf{x}_0)\right\|_2^2\right]. \tag{2}$$

We sample t from a logit-normal distribution following (Esser et al., 2024). At inference time, we generate samples by integrating the resulting flow ODE backward from $t = 1$ to $t = 0$ with Euler method.

Backbone The velocity field is parameterized by a 1.6-billion-parameter diffusion transformer (Peebles & Xie, 2023) operating over latent audio frames. The transformer comprises 16 blocks with model width $d = 2048$ and 32 attention heads. Each block follows a LLaMA-style design (Grattafiori et al., 2024), using RMSNorm (Zhang & Sennrich, 2019), SwiGLU activations (Shazeer, 2020), and rotary position embeddings (Su et al., 2024). Attention is implemented using FlashAttention (Dao et al., 2022). The model is conditioned on the flow time t, a text caption, and the style token; text conditioning is incorporated through cross-attention in every transformer block.

Training data The model is pretrained on an internally curated, fully licensed corpus containing approximately 24,000 hours of music recordings. The corpus is predominantly instrumental.

Optimization Pretraining uses FSDP2 sharding, BF16 arithmetic, and activation checkpointing within the attention modules. We train with AdamW (Loshchilov & Hutter, 2019) at a batch size of 2048 across 8 NVIDIA A100 nodes, maintaining an exponential moving average (EMA) of the model parameters.

B BASELINE IMPLEMENTATIONS

Our two morphing baselines, FreeMorph and FlowMorph, are originally designed for image-based morphing and have no official audio or music implementation or reported numbers to compare against; we therefore reimplement both ourselves for our text-to-music model, porting each paper's operations as faithfully as possible and documenting every deviation below.

B.1 FREEMORPH

Overview. For FreeMorph, we port the method from its image backbone to our internal flow-transformer, so the two input images become two audio endpoints. We retain the paper's two-stage recipe: a forward diffusion pass followed by a reverse denoising pass—together with the guidance-aware spherical interpolation of the clean latents ($z_{0,j} = \mathrm{slerp}(z_{0,\text{left}}, z_{0,\text{right}}, \alpha_j)$) and the phase-gated self-attention schedule controlled by $\lambda_1, \ldots, \lambda_4$. We strictly follow the same attention operations (Eq. 5, 6, and 7) in the paper, implemented via the "Outer" interpolation in which the morph query attends into each endpoint's keys/values separately and the resulting outputs are blended. Since FreeMorph performs its latent inversion via DDIM inversion and our backbone is a rectified-flow model on which DDIM inversion is not applicable, we substitute RF-Inversion Rout et al. (2025) with $\eta = 0$—i.e. a pure unguided ODE inversion, matching the paper's unguided forward pass—to invert the latent variables. Following the released code, we invert only the first $\lfloor \text{edit_strength} \cdot T \rfloor$ steps to partial noise (edit_strength $= 0.8$) and denoise back, rather than looping the full T of the paper's Algorithm 1 pseudocode. Additionally, we avoid injecting high-frequency noise at the end of the forward diffusion process as it produces artifacts in our final output. Finally, whereas the paper conditions on LLaVA-generated captions with classifier-free guidance (CFG $= 7.5$), each endpoint's prompt conditions only that side's forward-diffusion pass and the morph itself runs unguided (CFG $= 1.0$) under null text, so the morph is driven by the interpolated latent and blended attention alone.

Hyperparameters. For $(\lambda_1, \lambda_2, \lambda_3, \lambda_4)$ we use the same values suggested in the paper, $(0.3, 0.6, 0.2, 0.6)$, applied as fractions of the T steps so they transfer directly to our sampler. We keep $T = 80$ and set $\eta = 0$ and edit_strength $= 0.8$. For the number of intermediate samples we set $J = 10$, finer than the paper's $J = 5$, to obtain a smoother morph trajectory and for a fairer comparison to our method.

B.2 FLOWMORPH

Overview. For FlowMorph, we port the method from its FLUX.1-schnell image backbone to our internal flow-transformer; because the method is training-free—the flow network stays frozen and only latent-space variables are optimized—it transfers directly to any rectified-flow model. We keep the paper's two-variable parameterization of a local neighborhood around a clean latent z at a single noise level σ_i, $s(\Delta, u) = (z + \Delta) - (\sigma_{\text{last}} - \sigma_i)\, u$ (Eq. 5), where the offset Δ carries geometry and the one-step vector u carries semantics, and we fit (Δ, u) with the frozen backbone so a single Tweedie revert reconstructs z (Eq. 6). We adopt the paper's Flow-Interpolation mode (rather than its Flow-Optimizer mode): each endpoint is optimized independently, and the two solutions are then mixed in a decoupled way—the offsets Δ linearly (geometry) and the one-step vectors u spherically (semantics: slerp direction plus lerp magnitude, Eq. 9–12)—before the mixed state is advanced deterministically down the schedule to a clean latent. The main adaptation is to our sampler's noise schedule, which descends from $\sigma{=}1$ (pure noise) to $\sigma{=}0$ (clean): we anchor the parameterization at the VAE-encoded clean latent z and set the revert target to $\sigma_{\text{last}} = 0$, so a single step reverts fully to the clean level and drives the strongest possible semantic signal into u. The optimization noise level is taken from the same schedule a fixed fraction of the way through denoising, $\sigma_i = \sigma[\lfloor 0.35\, T \rfloor]$, matching the paper's $t_i = 35/100$. Because our backbone is not guidance-distilled—unlike FLUX.1-schnell, whose decode uses no CFG and a sparse 4-step schedule $[35, 55, 75, 95]$—we advance the mixed state with the full deterministic Euler schedule from σ_i down to 0, running the optimization at CFG $= 1.0$ (plain conditional velocity, matching the reference's guidance-free optimization) and decoding under the model's own classifier-free guidance. Finally, we make two smaller deviations from the released code, each of which improved quality in our setting: we replace the code's unsquared L_2 reconstruction norm $\|\hat{z} - z\|$ with the mean-squared error, and we initialize $\Delta = 0$ with u drawn from a shared Gaussian—keeping the two endpoints' independently-optimized u vectors in a common frame for the spherical mix—rather than the paper's zero-init warm-started from the source's own averaged one-step predictions.

Hyperparameters. We keep the paper's optimization settings: AdamW for 100 steps with $\text{lr}_\Delta = 0.04$ (offset) and $\text{lr}_u = 0.01$ (one-step vector), ti_frac $= 0.35$, and $\sigma_{\text{last}} = 0$. The L_2 regularizer weights $(\lambda_\Delta, \lambda_u)$ are left unspecified in the paper, so we default them to 0 (disabled) and expose them as knobs. We use $T = 80$ sampler steps and obtain $J = 10$ intermediate samples.

Table 4: Morph FAD (sum over weights) by alignment and method. Lower is better. **Bold**: best; underline: second best.

Alignment	Method	LAION-CLAP	MuQ	MuQ-MuLan	MusicFM
Tempo & Beat	Ours ($\delta = 0.8$)	0.5054	**34.5601**	**0.5746**	**46.3453**
	Ours ($\delta = 0.7$)	0.4914	41.2353	0.6351	46.9421
	Ours ($\delta = 0.6$)	**0.4893**	48.7937	0.6881	48.5295
Tempo	Ours ($\delta = 0.8$)	0.5024	**35.0949**	**0.5811**	**46.2386**
	Ours ($\delta = 0.7$)	**0.4904**	42.8571	0.6451	47.7108
	Ours ($\delta = 0.6$)	0.5266	57.1140	0.7283	56.6565
None	Ours ($\delta = 0.8$)	0.5287	**44.2323**	**0.6345**	**51.9771**
	Ours ($\delta = 0.7$)	0.5117	56.9850	0.6977	55.1062
	Ours ($\delta = 0.6$)	**0.5115**	70.1255	0.7713	59.7302

Table 5: Morph attribute (MAR) and smoothness (MSR) results by alignment and method. Lower is better. **Bold**: best; underline: second best.

		MusicFM		MuQ		LAION-CLAP	
Alignment	Method	MAR↓	MSR↓	MAR↓	MSR↓	MAR↓	MSR↓
Tempo & Beat	Ours ($\delta = 0.8$)	0.0693	0.0073	0.0958	0.0142	0.0995	0.0143
	Ours ($\delta = 0.7$)	**0.0664**	**0.0066**	**0.0934**	**0.0133**	**0.0968**	**0.0134**
	Ours ($\delta = 0.6$)	0.0678	0.0068	0.0977	0.0146	0.0984	0.0138
Beat	Ours ($\delta = 0.8$)	0.0697	0.0074	0.0967	0.0144	0.1003	0.0146
	Ours ($\delta = 0.7$)	**0.0673**	**0.0067**	**0.0954**	**0.0139**	**0.0976**	**0.0136**
	Ours ($\delta = 0.6$)	0.0683	0.0069	0.0995	0.0152	0.0993	0.0142
None	Ours ($\delta = 0.8$)	0.0711	0.0077	**0.1001**	**0.0157**	0.1004	0.0146
	Ours ($\delta = 0.7$)	**0.0701**	**0.0074**	0.1038	0.0172	**0.0978**	**0.0138**
	Ours ($\delta = 0.6$)	0.0735	0.0081	0.1144	0.0210	0.0992	0.0141

C Configuration Selection

C.1 MusicMorpher Parameters

We choose $\delta = 0.7$ and $J = 10$ for our method. We choose $J = 10$ to get enough intermediate samples. For δ, we decide this based on rank-based maximin. We run our method on $\delta = 0.8, 0.7, 0.6$ and collect FAD_{sum} and Morph Path Smoothness (MPS) scores across the configurations. The results of all the configurations can be seen in Tables 4 and 5, Since we want a configuration that performs well across all quality, attribute, and smoothness measurements, we rank the three settings separately on each individual metric: the four FAD_{sum} measures (LAION-CLAP, MuQ, MuQ-MuLan, and MusicFM), the three MAR measures, and the three MSR measures. Rank 1 denotes the best value. For each setting, we then take the worst (maximum) rank across all individual metrics, and rank-based maximin selects the setting with the best worst-case rank. This procedure yields $\delta = 0.7$ consistently across the Beat Aligned, Tempo Aligned, and No Alignment conditions, with a worst rank of 2 in each case. We therefore use $\delta = 0.7$ for subsequent experiments.

C.2 Semantic Editing Baselines

Similarly for both RF-Inversion and SDEdit, we use rank-based maximin to decide which parameters to use in our evaluation. For SDEdit, we test values $t_0 = 0.5, 0.6, 0.7, 0.8$ and for RF-Inversion, we set $\gamma = 0.5$ and test values $\eta = 0.4, 0.5, 0.6, 0.7, 0.8$. We measure FAD, Chroma L2, and CLAP directional scores. The results are seen in tables 6 and 7. Computing the rank-maximin for each quantitative score, yield us $t_0 = 0.8$ for SDEdit and $\eta = 0.4$ for RF-Inversion.

Table 6: FAD, $\text{Chroma}_{\text{L2}}$, and CLAP directional similarity for SDEdit at different noising levels t_0. FAD and Chroma are distances (lower is better); CLAP cosine is similarity to the target prompt (higher is better). **Bold**: best; underline: second best.

Config	$\text{FAD}_{\text{musicfm}}\downarrow$	$\text{Chroma}_{\text{L2}}\downarrow$	$\text{CLAP}_{\text{laion}}\uparrow$
$t_0 = 0.5$	151.4017	1.4506	0.0070
$t_0 = 0.6$	152.1933	1.4503	0.0169
$t_0 = 0.7$	151.4414	1.4510	0.0342
$t_0 = 0.8$	**150.9620**	**1.4461**	**0.0636**

Table 7: FAD, $\text{Chroma}_{\text{L2}}$, and CLAP directional similarity for RF-Inversion at different inversion strengths η. FAD and Chroma are distances (lower is better); CLAP cosine is similarity to the target prompt (higher is better). **Bold**: best; underline: second best.

Config	$\text{FAD}_{\text{musicfm}}\downarrow$	$\text{Chroma}_{\text{L2}}\downarrow$	$\text{CLAP}_{\text{laion}}\uparrow$
$\eta = 0.4$	171.1623	**1.3993**	**0.1137**
$\eta = 0.5$	176.6164	1.4060	0.0932
$\eta = 0.6$	172.6232	1.4191	0.0682
$\eta = 0.7$	163.2779	1.4327	0.0410
$\eta = 0.8$	**154.5221**	1.4436	0.0187

D Ablations

D.1 FreeMorph Ablation

Additionally, we conduct an ablation on FreeMorph. For FreeMorph, the forward process starts with clean latents, interpolates them, and applies inversion along with attention operations. We argue interpolating the latents in the clean space is a critical flaw and believe that clean latents cannot hold proper structure. We perform an ablation where we invert the latents to $t_{\text{inv}} = 0.7$, skip the forward process and then continue the denoising process. The result of this ablation can be seen in Table 8 for Tempo and Beat Aligned audio. We observe that this in fact improves the smoothness significantly, however, the quality drops significantly. This indicates that interpolating in a noise space where structure is preserved can greatly improve the performance, however, semantic alignment using attention operations can hinder performance and therefore suggests that operating the modulation space for semantic editing is more ideal.

E Morph Path Smoothness Algorithm

We provide a general algorithm for our morph metric, Morph Path Smoothness (MPS), in Algorithm 2.

F Full Quantitative Results

This appendix reports the complete quantitative results summarized in Section 5.2: every embedding backbone, every alignment condition, and both smoothness statistics (MAR and MSR).

Embedding backbones. For the FAD embeddings we use MusicFM (Won et al., 2024), MuQ and MuQ-MuLan (Zhu et al., 2025), and LAION-CLAP (Wu et al., 2023). MusicFM and MuQ are per-frame encoders: we extract their hidden states (MusicFM from an intermediate conformer layer, MuQ from the final layer) and mean-pool along the token dimension into a single clip-level vector. LAION-CLAP yields a single projected, L2-normalized clip embedding. MuQ-MuLan's audio embedding is taken from the final average-pooled layer of its audio encoder, projected into the joint audio–text latent space and L2-normalized.

Morphing quality and smoothness. Across all endpoint alignment settings and embedding configurations, our method achieves the lowest FAD_{sum}, indicating that intermediate samples maintain

Algorithm 2: Morph Path Smoothness Score (MPS)

Input : Morph paths $P = \{A^i\}_{i=0}^{n-1}$ (each a sweep of $J{+}1$ samples), encoder M, steps J, residual power $p \in \{1, 2\}$
Output: Linearity score s (lower is better)

$\mathbf{D} \leftarrow [\,0\,]^{J+1}$; // accumulated mean curve
$\mathcal{C} \leftarrow []$; // per-path normalized curves, kept for pass 2
$\mathbf{g} \leftarrow [\,0, \frac{1}{J}, \ldots, \frac{J-1}{J}, 1\,]$; // uniform weight grid α_j

// Pass 1: per-path normalized distance-to-left, averaged
for $i = 0$ **to** $n-1$ **do**
  $\mathbf{E} \leftarrow M(A^i)$; // embed all $J{+}1$ steps
  $c \leftarrow \mathrm{MMD}(\mathbf{E}_0, \mathbf{E}_J)$; // $\mathrm{MMD}(L, R)$, the normalizer
  for $j = 0$ **to** J **do**
    $\mathbf{d}_j^i \leftarrow \mathrm{MMD}(\mathbf{E}_j, \mathbf{E}_0) \,/\, c$; // normalized dist. to left
  $\mathcal{C}[i] \leftarrow \mathbf{d}^i$;
  $\mathbf{D} \leftarrow \mathbf{D} + \mathbf{d}^i$;
$\mathbf{D} \leftarrow \mathbf{D} \,/\, n$; // mean curve over paths

$\mathbf{X} \leftarrow \mathrm{GaussKernelSmooth}(\mathbf{g}, \mathbf{D})$; // 1-D Gaussian kernel-inverse warp
$f \leftarrow \mathrm{BestFit}(\mathbf{X}, \mathbf{D})$; // least-squares line $f(x) = ax + b$

// Pass 2: residual of every sample about the fitted line
$\mathbf{S} \leftarrow []$;
for $i = 0$ **to** $n-1$ **do**
  for $j = 0$ **to** J **do**
    $r \leftarrow \left|\, \mathcal{C}[i]_j - f(\mathbf{X}_j) \,\right|^p$;
    $\mathbf{S}$.append(r);
$s \leftarrow \mathrm{Mean}(\mathbf{S})$; // p=1: mean abs residual; p=2: MSE
return s

Table 8: FreeMorph Ablation Results. Best value per embedding backbone is in **bold**.

	$FAD_{sum}\downarrow$			MAR↓		
Method	LAION-CLAP	MuQ	MusicFM	LAION-CLAP	MuQ	MusicFM
FreeMorph	**0.9638**	**131.3430**	**122.3049**	0.1226	0.1265	0.0907
+ Inversion	1.1375	168.3498	147.6169	**0.0690**	**0.0665**	**0.0496**

Table 9: FAD_{sum} (summed over the intermediate morph sequence; lower is better) per embedding backbone, grouped by beat-alignment condition with the three morphing methods compared within each. Best method per column (within a condition) in **bold**, second best underlined.

Alignment	Method	LAION-CLAP	MuQ	MuQ-MuLan	MusicFM
Tempo & Beat	FreeMorph	0.9638	131.3430	1.6156	122.3049
	FlowMorph	2.8445	212.3344	2.5587	429.9906
	Ours	**0.4914**	**41.2353**	**0.6351**	**46.9421**
Tempo	FreeMorph	1.0047	139.7602	1.7368	129.7738
	FlowMorph	2.8774	214.9338	2.5991	433.1636
	Ours	**0.4904**	**42.8571**	**0.6451**	**47.7108**
None	FreeMorph	1.0955	201.4324	2.1062	168.6095
	FlowMorph	2.8609	230.5778	2.6549	443.7206
	Ours	**0.5117**	**56.9850**	**0.6977**	**55.1062**

quality as the morph progresses, as well as the lowest mean absolute and mean squared residual distances, showing that our morphs are substantially smoother than both baselines.

Semantic editing. We report FAD with MusicFM embeddings for quality, the L2-normalized Chroma distance for structural alignment, and directional CLAP score for edit alignment.

Additional Morphing FAD results. We also compute the FAD_{mean}, the FAD mean between all morph points and an internal human music dataset for each method using the MusicFM embedding as a backbone.

Table 10: Morph Path Smoothness (MPS) on the linear-fit residual metric (Algorithm 2), computed on the kernel-warped x-axis. For each embedding we report the mean absolute residual (MAR) and mean squared residual (MSR) of the normalized MMD trajectory about its line of best fit; lower is better. Per column within each alignment, the best is in **bold** and the second best is underlined. In all cases, our method outperform the baselines showcasing its superior performance.

		MusicFM		MuQ		LAION-CLAP	
Alignment	Method	MAR↓	MSR↓	MAR↓	MSR↓	MAR↓	MSR↓
Tempo & Beat	FreeMorph	0.0907	0.0137	0.1265	0.0263	0.1226	0.0238
	FlowMorph	0.1166	0.0204	0.1238	0.0217	0.1549	0.0367
	Ours	**0.0664**	**0.0066**	**0.0934**	**0.0133**	**0.0968**	**0.0134**
Tempo	FreeMorph	0.0935	0.0144	0.1281	0.0267	0.1242	0.0244
	FlowMorph	0.1172	0.0206	0.1240	0.0217	0.1545	0.0365
	Ours	**0.0673**	**0.0067**	**0.0954**	**0.0139**	**0.0976**	**0.0136**
None	FreeMorph	0.1063	0.0181	0.1541	0.0372	0.1255	0.0248
	FlowMorph	0.1143	0.0197	0.1261	0.0225	0.1527	0.0354
	Ours	**0.0701**	**0.0074**	**0.1038**	**0.0172**	**0.0978**	**0.0138**

Table 11: FAD, $\text{Chroma}_{\text{L2}}$, and CLAP directional similarity across methods. FAD (lower is better); CLAP cosine is similarity to the target prompt (higher is better). **Bold**: best; underline: second best. For Chroma distances: **Bold** has the highest change; underline has the lowest change.

Configuration	$\text{FAD}_{\text{musicfm}}$↓	$\text{Chroma}_{\text{L2}}$↑↓	$\text{CLAP}_{\text{laion}}$↑
SDEdit (t_0=0.8)	**150.9620**	**1.4461**	0.0636
RF-Inversion (η=0.4)	171.1623	1.3993	**0.1137**
FreeMorph (Tempo & Beat)	184.3211	0.8317	0.1034
FreeMorph (Tempo)	186.0725	0.8336	0.1037
FreeMorph (None)	193.2654	0.8357	0.1061
FlowMorph (Tempo & Beat)	227.4422	0.9751	0.0699
FlowMorph (Tempo)	229.4218	0.9757	0.0710
FlowMorph (None)	234.5856	0.9796	0.0727
Ours (Tempo & Beat)	167.4406	1.0968	0.1082
Ours (Tempo)	168.2825	1.0998	0.1076
Ours (None)	171.1934	1.1095	0.1099

Table 12: FAD_{mean} (mean over the intermediate morph sequence; lower is better) for the MusicFM embedding backbone, grouped by alignment condition with the three morphing methods compared within each. Best method within each condition in **bold**, second best underlined.

Alignment	Method	MusicFM
Tempo & Beat	FreeMorph	178.3410
	FlowMorph	216.9040
	Ours	**159.0383**
Tempo	FreeMorph	179.6549
	FlowMorph	217.4598
	Ours	**159.3640**
None	FreeMorph	185.1685
	FlowMorph	218.8247
	Ours	**160.3493**