

Interactive Generative Motion Editing via Scheduled Inpainting

Dhruv Agrawal^{1,2} , Dominik Borer² , Luca Voğeli² , Robert W. Sumner^{1,2} , Martin Guay² , Jakob Buhmann² 

¹ETH Zürich, Switzerland ²DisneyResearch|Studios, Switzerland

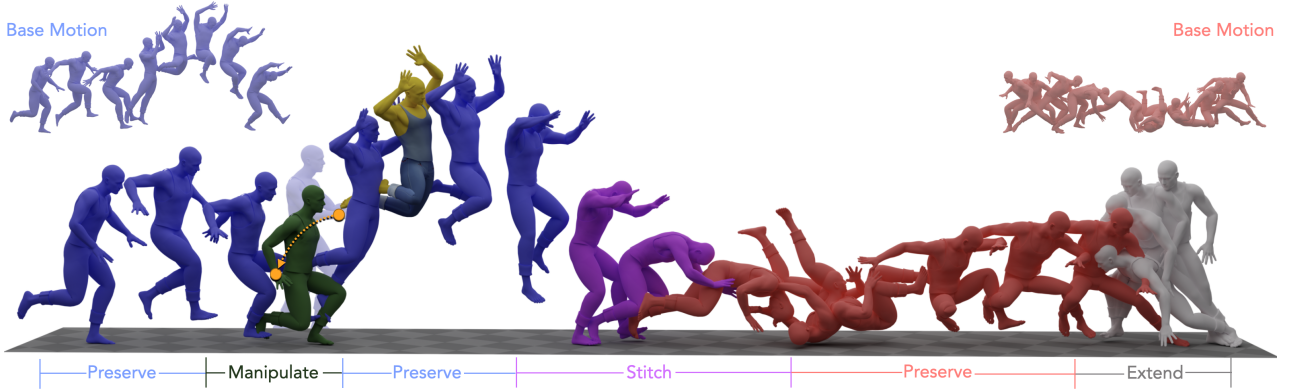


Figure 1: Scheduled inpainting allows for multiple generative motion editing tasks within the same real-time framework, while resulting in natural motion. We see here two clips in blue and red being stitched together (purple), extended (grey), as well as directly modified by dragging points (yellow dots).

Abstract

Motion editing is central to VFX and game development, where it is used extensively to modify and augment existing movements to conform to new environments or changes in artistic direction. While traditional motion editing can do small modifications, it cannot accommodate larger structural edits, resulting in visual warping artifacts that require authoring new motion. Conversely, recent advances in large-scale generative modeling have unlocked newfound capabilities for authoring entire movements by directly manipulating sparse spatial constraints. While impressive at creating new movements, these methods lack the capability to preserve and edit existing motion interactively. In this work, we introduce scheduled inpainting, a method that enables interactive generative motion editing, a novel paradigm unifying motion synthesis and editing by leveraging generative models. Scheduled inpainting is a simple yet powerful inference-based technique that enables fine-grained spatiotemporal control over the balance between preserving the original motion and generating new content. By building atop generative models that support direct manipulation, our system allows artists to interactively refine existing animations while ensuring results remain natural and consistent with the learned motion distribution. Scheduled inpainting is versatile and supports many editing applications, such as extending, stitching, and compositing different clips. Finally, we extensively validate our approach by comparing with four baselines, conducting ablations of our design, and reporting user feedback.

CCS Concepts

• **Computing methodologies** → **Artificial intelligence**; **Animation**; • **Human-centered computing** → **User centered design**;

1. Introduction

High-quality motion is both time-consuming and challenging to author and acquire. Motion editing is often used in game and movie productions to fix, alter, or augment movements, but is limited to small changes and often results in unnatural motion due to warping when larger structural changes are required. For example, sub-

stantial manual effort is required to maintain physical plausibility when changes in the height of a platform cause the character to jump down instead of stepping down, as shown in our accompanying video.

Recent advances in generative motion authoring allow users to create entire motions either via textual [TRG*23; TCL23;

STKB24] or spatial [SAB*24; HBLK25; VAG*25] conditioning. Among these models, only spatially conditioned models can interactively modify their own output via direct manipulation of sparse spatial constraints. Crucially, no prior model has demonstrated interactively editing existing motion clips; that is, motion sequences from another source and not generated by the model itself. The closest existing works for generative motion editing include MotionFix [ACD*24] and MotionLab [GHSZ25], which edit using text commands. However, text is a coarse condition modality and does not allow interactive spatial control, which is essential for artists who require precision when manipulating movements.

In this paper, we introduce a novel workflow that we call *interactive generative motion editing*. In contrast to the limited and coarse control offered by prior motion editing systems, interactive generative motion editing enables large-scale structural changes to existing animations—such as extension, stitching, and composition—while simultaneously allowing users to manipulate these motion sequences both spatially and temporally. Moreover, it combines synthesis and editing into a common framework, allowing artists to seamlessly switch between creating and editing animations.

To solve this task, we introduce *scheduled inpainting*, a novel inference-based technique that extends pretrained generative motion models with the ability to edit existing movements—without requiring any additional training and at virtually no additional computational cost. By dynamically blending source motion at inference time, we leverage the generative prior of these models in a controllable way, enabling both large and small edits, while consistently resulting in natural motion.

While simple inpainting [TRG*23; TCL23] and blending [STKB24] have been explored for motion generation, these methods render the inpainted parts of motion completely uneditable and uncontrollable, making them problematic for generative motion editing. Beyond extending naive inpainting, our primary contribution lies in building the first comprehensive framework that solves this novel task. More specifically, our scheduled inpainting framework comprises three components: a user-controlled schedule that modulates the amount of preservation, a spatiotemporal strength mask controlling the location of the preservation, and a robust representation space optimized through extensive ablations to enable compositing movements from very different global orientations and scales. Although simple, our framework is versatile and supports many applications. By configuring the spatiotemporal mask, artists can alter movements directly and procedurally, as well as stitch, extend, and compose clips, as shown in Figure 1 and our accompanying video.

In summary, our main contributions include:

- **Interactive Generative Motion Editing:** A novel generative workflow that simultaneously allows structurally editing and interactively manipulating existing animations.
- **Scheduled Inpainting:** An inference technique that extends binary inpainting with a user-controllable schedule and a user-defined spatiotemporal mask to enable the first framework capable of interactive generative motion editing.
- **Unified Motion Framework:** A versatile system with animation software integration that solves a wide range of animation tasks,

including stitching, extension, and composition, within a single interactive framework.

To our knowledge, scheduled inpainting is the first interactive system for generative motion editing. To evaluate its performance, we compare with several adapted baselines, including noise inversion [SME22] and noise optimization [KPA*24], and ablate key design choices. We further demonstrate our system’s capabilities through a series of results in our supporting video and report qualitative feedback from professional artists.

2. Related Work

Seminal work in motion editing introduced layered cubic spline interpolation [WP95] and remains the standard in many authoring and game engine software [Aut25; Hou25]. While both optimization [Gle97; LS99] and statistical models [CH07; MCC09] have been explored to go beyond warping movements, they could not generalize to large datasets, and could not make big changes due to local minima issues.

Recently, advances in deep learning, in particular diffusion models, have enabled training on large datasets that incorporate a vast range of motion into a single model. The first wave of motion diffusion focused on text conditioning [TRG*23], while subsequent efforts led to a vast expansion into different capabilities such as contact-aware generation [MCZT24], multi-character interaction [XFHH25; WMZ*25; RBP*25], keyframe-centric generation [BHL*25; GZLF25], sketch-based conditioning [ZGX*25], and multi-modal conditioning [ZWM*25; LCZ*25; LHS*25]. Binary spatiotemporal masks over frames and joints have been used as training-time objectives or conditioning for generating new motion [PSK*25; Che24; TGN*24; CTR*24]. While these models are powerful generators, they are not designed for preserving and editing existing motion.

2.1. Direct Manipulation Models

Most related to our work are models focused on interaction such as CondMDI [CTR*24], which enable interactive authoring by conditioning on full-body keyframes or joint trajectories. Studer et al. [SAB*24] and Vögeli et al. [VAG*25] showed that compressing the motion representation spatially and temporally improves constraint satisfaction, allowing for both efficient and precise manipulation. Similarly, Hwang et al. [HBLK25] use a two-stage approach but adopt a slower generative model instead of the efficient neural IK model [OBH*22] used in previous methods. While these models can interactively modify their generated motion, they cannot edit an exemplar motion clip in a structure-preserving manner. A related line of work targets real-time interactive control of motion diffusion models [SWJ*24; GvdPH25], but likewise steers online generation rather than editing an existing clip.

2.2. Motion Editing

To our knowledge, the only model explicitly designed for motion editing in a direct manipulation setting is the work of Agrawal et al. [ABB*24], which trains a deterministic motion editing model (which is limited to smaller datasets). Their technique also requires

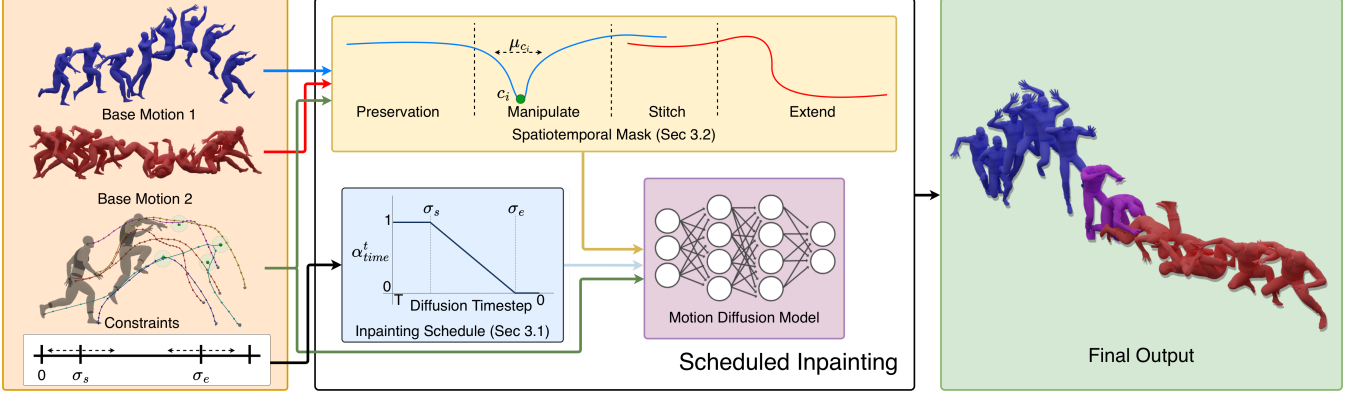


Figure 2: Overview of our scheduled inpainting for interactive generative motion editing. A user can specify which motion should be used (red and blue), as well as control where and how much the motion should be preserved via controls impacting the spatiotemporal mask, which in turn dictates the schedule.

training a specific model, in contrast to our approach that works at inference with different pretrained diffusion models.

Text-based motion editing has been explored using LLM-extracted keywords to guide motion generation [ZLC*23; HWY*24]. Chen et al. [CLD*24] then showed that modifying the attention between motion and word tokens could replace, erase, and re-sequence parts of movements. Recently, MotionFix [ACD*24], STANCE [JLY*25], and FineMotion [WXD*25] released motion editing datasets with triplets of original motion, text, and final motion. To improve quality, Jiang et al. [JLY*25] explored leveraging unannotated data, Guo et al. [GHSZ25] jointly training for editing and synthesis, while Li et al. [LCG*25] explored predicting the similarity between edited and original motion. Although text-based motion editing is useful to make broad changes, it does not provide the fine-grained and direct spatial control artists need in order to achieve their full vision.

2.3. Inference-based techniques

These were first developed in the image domain working alongside StableDiffusion [RBL*22]. Shi et al. [SXL*24] use DDIM noise inversion [SME22] to recover the latent code for a given image, to then optimize in latent space to match displaced points on the image. Similarly, Liu et al. [LXY*24] optimized the added noise to perform edits. In the motion domain, optimizing the initial noise for better control was explored by Karunratanakul et al. [KPA*24]. While it is possible to edit movements once inverted, the optimization is too costly to enable iterative workflows. Each time an artist needs to edit a different motion, a new offline optimization is required, breaking real-time workflows and preventing fluid interactions. In contrast, our inpainting approach operates at a constant cost and allows for larger changes in real-time.

3. Scheduled Inpainting

To achieve the full potential of what we call interactive generative motion editing, we need spatial control over the amount of

preservation and generation. While inpainting an example clip during inference is an obvious choice for its efficiency, we observed it completely replaced the generated motion, resulting in unnatural transitions and leaving no flexibility to edit.

We introduce a user-controllable schedule that modulates the inpainting strength and then augment it with a spatiotemporal mask that provides control over regions of generation and preservation. Note that in order to support direct manipulation, we focus our exposition and results on models such as IBMM [VAG*25] and SF-control [HBLK25]. To achieve consistent movement alignment, we carefully construct a normalization state as a naive approach of inpainting clips from disparate spaces, like a front roll with a back roll, would introduce spurious motion artifacts.

Figure 2 shows an overview of our method. Formally, **scheduled inpainting** works with an example motion, which we call base motion $\mathcal{M}_{base}$, with T frames and J joints, together with a set of user-defined constraints $\mathcal{C}$. Our inference-based method operates with a diffusion model ϕ , and we refer to the generated motion as $\mathcal{M}_{gen}^0 = \phi(\mathcal{C}, \mathcal{M}_{gen}^t)$ where $\mathcal{M}_{gen}^t$ is the noisy motion according to the diffusion timestep t . The inpainted motion at timestep t is then given by:

$$\widehat{\mathcal{M}}_{gen}^0 = \alpha^t \times \mathcal{M}_{base} + (1 - \alpha^t) \times \mathcal{M}_{gen}^0 \quad (1)$$

$$\text{with } \alpha^t = \alpha_{time}^t \times \alpha_{mask}, \quad (2)$$

where a timestep-regulated scalar α_{time}^t and application mask α_{mask} modulate the inpainting strength. The input to ϕ at timestep t is then $\widehat{\mathcal{M}}_{gen}^t$ instead of $\mathcal{M}_{gen}^t$.

3.1. Inpainting Schedule

The temporal scalar α_{time}^t controls the strength of inpainting through the diffusion process. It is parameterized by two user-controlled variables σ_s and σ_e that define the start and the end of

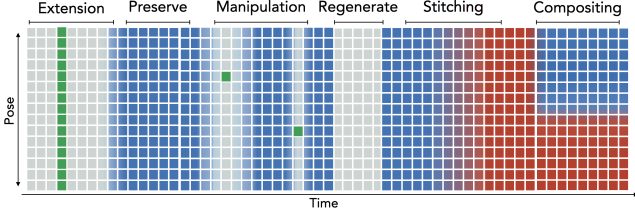


Figure 3: Conceptual overview of our spatiotemporal mask α_{mask} for the different motion editing applications using two different motions (red and blue): direct manipulation with full-body and sparse constraints (green), extending (grey), as well as stitching, and compositing clips.

the schedule respectively. It is defined as:

$$\alpha_{time}^t = \begin{cases} 1 & \sigma_s < t \\ \frac{t - \sigma_e}{\sigma_s - \sigma_e} & \sigma_e \leq t \leq \sigma_s \\ 0 & t < \sigma_e \end{cases} \quad (3)$$

Hence, for noise levels larger than σ_s , the generated motion is completely overwritten by the base motion $\mathcal{M}_{base}$ and conversely, for noise levels lower than σ_e , no inpainting is applied. Between the two noise levels, we linearly interpolate between the base and generated motions. By manipulating $\sigma_{s/e}$, a user can set the strength of preservation versus generation during inference. Note that we show the effect of different schedules $\sigma_{s/e}$ in Section 5.3.1.

3.2. Spatiotemporal Mask

The spatiotemporal mask $\alpha_{mask} \in \mathbb{R}^{T \times J}$ in Eq. (2) weighs how much the base motion should be preserved, both per joint j and per frame t . It can be constructed differently to support various applications as shown in Figure 3. For simplicity of exposition, we focus here on constructing a direct manipulation mask and defer to Section 4 for remaining applications.

By default, scheduled inpainting completely preserves the base motion and our spatiotemporal mask is set to 1. To manipulate the motion at a user-specified joint location called constraint $c_i \in \mathcal{C}$, we modify the mask to reduce the inpainting weight in the neighborhood of the constraint, with user-controllable kernel functions, as shown in Figure 4. Given constraints $\mathcal{C}$, we construct the mask with the following equation:

$$G(j, t; c_i) = \exp\left(-\frac{(t - \text{FrameIdx}(c_i))^2}{\mu_{c_i}}\right) \quad (4)$$

$$\alpha_{mask}(j, t) = \max\left(1 - \sum_{c_i \in \mathcal{C}} G(t; c_i), 0\right), \quad (5)$$

where the user can control the influence width via the scalar μ_{c_i} , and the max function ensures the mask is never negative.

3.3. Inpainting Space

Naively mixing two unrelated motions can lead to unwanted artifacts. For example, mixing two clips that start at the same position

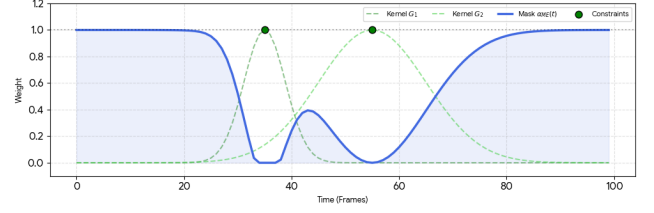


Figure 4: An example of the spatiotemporal mask α_{mask} for a single joint when a user manipulates two constraints at frames 35 and 55. The motion is allowed to be generated in these areas, and is steered towards preservation in other parts of the timeline.

but walk away in opposite directions results in an average motion that is mostly static near the starting position. Or mixing two clips that are of very different speeds introduces discontinuities in accelerations.

Therefore, when inpainting $\mathcal{M}_{base}$ onto $\mathcal{M}_{gen}^0$, it is important to align and normalize both sequences to avoid such artifacts. First, we align each motion such that the first frame is at the origin, and the direction from the first frame to the last is along the positive x-axis. Additionally, we individually normalize both sequences to 0 mean and 1 variance to ensure that the two motions are on the same scale during blending. To avoid discontinuities when extending or mixing animations, we represent the root using differential coordinates, together with root-relative positions and orientations for the remaining joints.

Note that our representation space is independent of the pretrained model’s representation space and normalization. As ϕ generates clean motion at each diffusion timestep, we transform the generation into our representation space for blending before inverting to the model’s native representation. Hence, the pretrained model is completely ignorant of our alignment and normalization, which are applied in addition to any of its own data processing.

We show ablations of the different components of our inpainting space in Section 5.3.2 and Appendix B.

4. Applications

By constructing α_{mask} in Eq. (2) differently, our scheduled inpainting technique can enable a number of generative motion editing tasks, as shown in Figure 3—each time enabling sampling variations due to the underlying generative motion. In Section 3.2, we described forming a direct manipulation mask and now turn to describing generative extension, stitching, compositing, and retiming.

4.1. Motion Extension

A given motion sequence $\mathcal{M}_{base}$ of length T can be expanded by T_e frames at any position by setting the strength mask $\alpha_{extend} \in \mathbb{R}^{(T+T_e) \times J}$ to 0 for the interval $[t_s, t_e)$, which defines the region where the new motion is generated:

$$\alpha_{extend}[t, :] = \begin{cases} 0 & t_s \leq t < t_e \\ 1 & \text{otherwise} \end{cases} \quad (6)$$

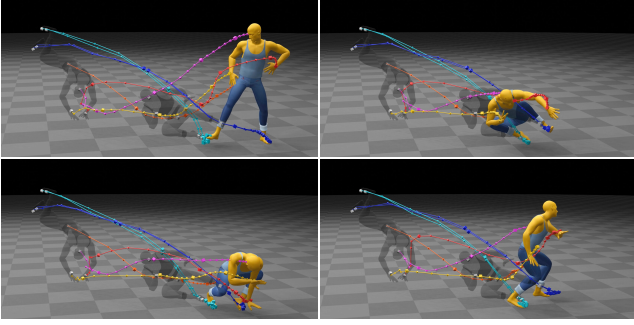


Figure 5: Generative motion editing allows sampling different motions, such as different extensions of a rolling motion. Each curve represents the motion path for an end-effector joint and the two black silhouettes represent the start and end of the base motion.

Note that $\mathcal{M}_{base}$ must be padded with additional frames to align with α_{extend} in Eq. (1).

In Figure 5, we extend a roll beyond the second silhouette, which is where the original motion ends ($t_s = T$ and $t_e = T + T_e$). Because we use a generative model, we can sample various extensions for the same base clip, as demonstrated by the different end poses. Furthermore, since the preserved $\mathcal{M}_{base}$ is inpainted as opposed to input as constraints, the entire motion remains fully editable via direct manipulation described in Eq. (4).

In the case where the generated motion is in the middle of the base motion, our inpainting space of root velocities and root-relative transforms for the remaining joints allows the second half of the motion to continue from the end of the generated sequence without any discontinuity in root trajectory. Please refer to our video for these results.

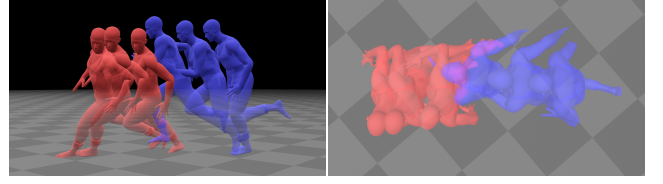
4.2. Motion Stitching

Another common animation task involves creating transitions between independent motion clips. Unfortunately, naively stitching clips creates discontinuity or sliding artifacts, requiring manual clean-up. Given two motion clips $\mathcal{M}_{base}^0$ and $\mathcal{M}_{base}^1$ of lengths T_0 and T_1 , respectively, we can set the spatiotemporal mask to $\alpha_{mask} \in \mathbb{R}^{(T_0+T_{gen}+T_1) \times J}$, such that we inpaint the first frames T_0 in the sequence with the first base motion $\mathcal{M}_{base}^0$, smoothly transitioning to generating frames T_{gen} in the middle, and finally smoothly transitioning to inpainting the last frames T_1 with the second base motion $\mathcal{M}_{base}^1$.

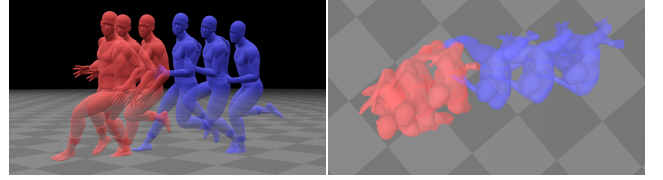
Note that stitching clips is a good example of where our normalization is essential for quality results. We show in Figure 6a that even for manually aligned clips there is a sharp change in poses and root orientations at the boundary. In contrast, scheduled inpainting modifies the boundary between the clips to generate a realistic transition, as shown in Figure 6b and our accompanying video.

4.3. Motion Compositing

We define motion composition as applying different animation clips to different body parts within the same sequence. Motion is



(a) Naive stitch



(b) Scheduled Inpainting

Figure 6: Comparison of stitching clips naively to our scheduled inpainting method. Top down view shows that scheduled inpainting resolves discontinuity artifacts seen in naive stitching.

highly correlated, and combining different movements easily results in unrealistic movements. In contrast, composing clips with our method results in natural motion due to the underlying generative prior.

Note that this task is similar to motion stitching, but instead we blend different motions along the joint dimension instead of time. Consequently, the compositing mask can be constructed similarly to a stitching clip, but by varying the mask across joints rather than frames, where a set of joints such as the upper body get inpainted with a first clip, while the second set of joints are inpainted from a second clip.

4.4. Adapting animation cycles

Much production motion data consists of animation cycles that are short sequences of repetitive actions such as walking and running that can be looped seamlessly to create arbitrarily long animations. Such cycles are often used for characters in games or for crowds in movies. However, motion controllers typically override the root motion to adapt in real-time to different speeds and environments, introducing foot sliding.

Using scheduled inpainting, we can adapt an animation cycle for arbitrary trajectories by either fully generating the root joint or constraining it to a user-drawn curve while inpainting the remaining joints with the animation cycle. The resulting motion naturally follows the desired path, but without foot sliding artifacts. As shown in our accompanying video, the artist can interactively update the path with resulting motion updated in real-time. The generated motion follows the specified path but can maintain highly stylized animation cycles such as *Zombie*, *Chicken*, and *Monkey*.

4.5. Retiming Motion

In practice, motion is often adapted to new pacing, dialogue, or music. A simple approach is to fit a parametric curve to the original animation and resample it with the new timing. However, this

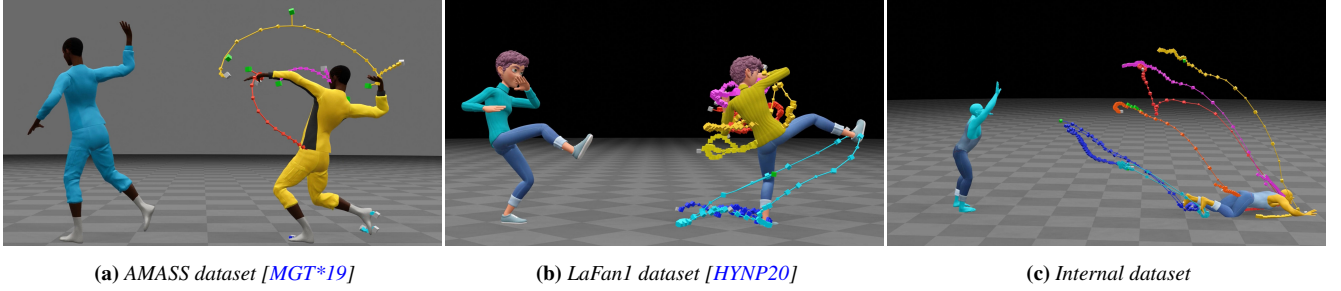


Figure 7: Motion editing examples using point-based constraints (green) for AMASS (left), LaFan1 (middle) and internal (right) datasets. Original motion is shown in cyan and edited motion in yellow.

only warps the motion and cannot accommodate structural changes when timings change significantly. By setting $\alpha_{mask} = 0.8 \cdot \mathbb{1}_{J \times T}$, our scheduled inpainting method uses the warped motion as the base motion to recover natural movement through generation. We refer to our supporting video for retiming results.

5. Evaluation & Results

In this section, we validate the benefits of scheduled inpainting, as well as each of its components. After introducing alternative generative motion editing baselines—which did not demonstrate direct manipulation before—we show multiple comparisons, as well as showcase the universality of our method by applying it to different models and datasets.

5.1. Baselines

Our first baseline is MotionLab [GHSZ25], trained on the MotionFix [ACD*24] dataset. Note that although it supports spatial constraints, it requires dense trajectory conditioning and is not interactive, requiring 0.88 secs for a single sample.

As interactive generative motion editing has not been demonstrated before, we developed three additional baselines for holistic evaluation. First, we introduce *CondEditor*, which adapts CondMDI [CTR*24] using the motion editing training scheme proposed by [ABB*24] and trains on synthetic triplets of original, edited, and final motion.

Our last two baselines, DNO [KPA*24] and noise-inversion are inference techniques that recover noise for an example motion via optimization and DDIM-inversion [SME22], respectively. Note that in our experiments, noise-inversion could not run in real-time, as it requires at least 400 denoising steps for high-fidelity motion reconstruction in comparison to only 25 steps with our method. See Appendix D for further details.

5.2. Results & Comparisons

We now report using scheduled inpainting with different models and datasets and compare with the previously described baselines. Note that our general workflow for these results consisted in setting full-body constraints to make large edits, followed by individual position and orientation constraints to refine the motion curves.

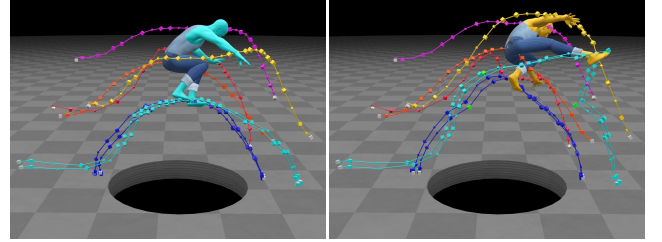


Figure 8: With scheduled inpainting, we can directly manipulate a few joints to modify a jumping motion (left) to include a natural kick in midair (right). The manipulated constraints are highlighted in green and also shown in our video.

5.2.1. Results

First, we use the IBMM [VAG*25] motion model to edit a jumping motion to include a kick in midair, shown in Figure 8. Notice in our accompanying video how without our scheduled inpainting, the generated motion simply runs between the keyframes. As IBMM is a synthesis model, it is unaware of the jump without introducing additional constraints. In contrast, with our method the user only needs to manipulate a single kicking pose to obtain a coherent and natural midair kicking motion.

Additionally, we show results of using scheduled inpainting with two interactive models and three different datasets. In Figure 7a, we use the SF-control model [HBLK25] to exaggerate a throwing motion clip from AMASS [MGT*19], by moving the hand constraint backwards. Similarly, Figure 7b illustrates a kick from the LaFan1 dataset [HYNP20], edited using IBMM to add body twist and height. Finally, we use IBMM to edit a 'falling when stretching' motion from our internal dataset, in order to change the timing of the fall and the final pose.

5.2.2. Trained baselines

In Figure 10, we show a comparison with the MotionLab [GHSZ25] model. While MotionLab can match hip trajectories when inside the learned distribution, it does not generalize well when stretching outside. For example, taking a walking forward motion from the MotionFix validation set and moving the hip trajectory further back results in walking backward only halfway, as shown in Figure 10a. This problem is even more prominent when

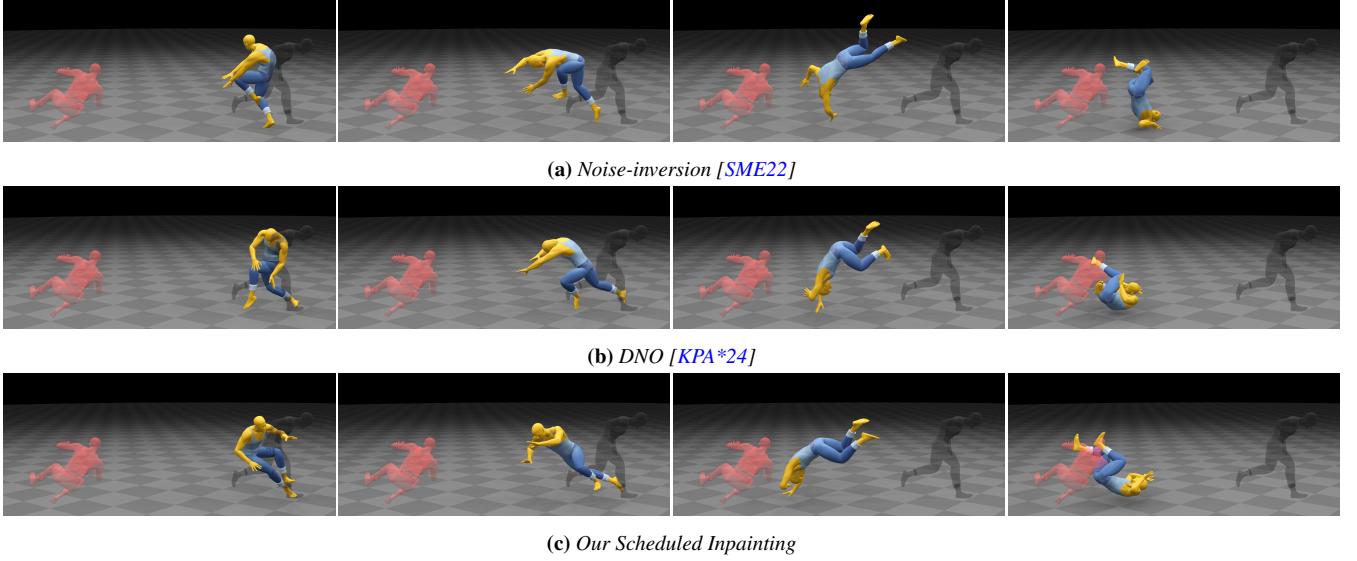


Figure 9: We edit the end frame (shown in red) of a rolling forward clip to go behind the starting frame (shown in black). Each row shows the output from noise-inversion [SME22] (top), DNO [KPA*24] (middle), and our scheduled inpainting method (bottom) at three different moments or frames of the motion. While noise-inversion and DNO give an unnatural turn that is too fast in the first two frames, our method generates a realistic turn before the roll.

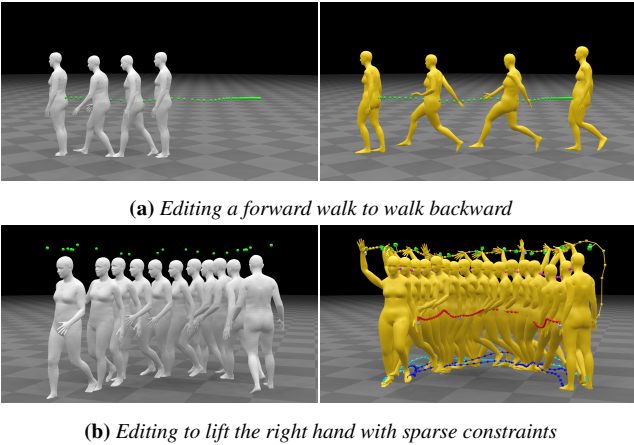


Figure 10: Comparison of scheduled inpainting (right) to MotionLab [GHSZ25] (left). MotionLab cannot extend the walking distance (a) and ignores sparse constraints (green) (b), while our approach follows them accurately.

using sparse constraints, instead of dense trajectories, as shown in green in Figure 10b, where we edit the arm position. MotionLab completely ignores the constraints while the SF-control model using our method successfully generates the required edits.

Lastly, in our video we show an example of direct manipulation with CondEditor. We can observe that it struggles to follow constraints and cannot perform structural changes as the deformation becomes larger. In contrast, our method maintains both constraint precision and natural results throughout the editing process.

5.2.3. Inference-based baselines

Scheduled inpainting, thanks to our alignment and normalization (see Section 3.3), allows users to make large edits in real-time, in cases where even offline noise inversion baselines fail. For example, consider changing a rolling forward motion to a rolling backward motion, as shown in Figure 9 and our video. We can see that both DNO and noise-inversion result in many artifacts and unnatural motion. Since the target motion is quite different from the original, the recovered noise of the inversion is no longer compatible with the target motion, causing artifacts such as unnatural poses and interpenetration during takeoff and landing. In particular, we can see that the backward turn happens too quickly with the character hovering, resulting in self-penetration artifacts. In contrast, our method results in natural motion, including taking more realistic timing to turn.

5.2.4. Quantitative Analysis

Table 1 reports the impact of randomly displacing the final frame of a sequence by up to one meter horizontally when using IBMM with and without our scheduled inpainting. To evaluate the results, we measure foot-sliding and root-relative reconstruction error of non-root joints, reported as L2 position error (L2P) and L2 rotation error (L2R), to quantify the motion naturalness and similarity to base motion. Without displacement, our method matches the foot-sliding performance of IBMM while significantly improving base motion preservation. When the final frame is displaced, our method's foot-sliding increases marginally (by ~ 1 mm/frame) because the constraint of preserving the original trajectory conflicts with the edit. Despite this, our approach consistently outperforms IBMM in maintaining the integrity of the original motion.

These results demonstrate that our method effectively executes

Table 1: Foot sliding (m/frame) and root-relative reconstruction metrics when randomly displacing the final frame of the animation sequence on our internal dataset.

Task	Edited Final Frame?	Foot-sliding ↓	L2P ↓	L2R ↓
GT	-	0.0116	-	-
IBMM	No	0.0101	0.0122	0.1484
Ours	No	0.0105	0.0034	0.0886
IBMM	Yes	0.0114	0.0131	0.1566
Ours	Yes	0.0125	0.0063	0.1051

edits while simultaneously preserving the original motion’s characteristics and naturalness.

5.3. Ablations

In this section, we ablate our design choices. We quantitatively evaluate the impact of the inpainting schedule on the reconstruction error. Additionally, we qualitatively assess how our motion alignment and influence radius affect the generated motion.

5.3.1. Inpainting Schedule

Table 2 compares different inpainting schedules on reconstructing $\mathcal{M}_{base}$ on our internal and LaFan1 [HYNP20] datasets. Compared to no inpainting ($\sigma_s = \sigma_e = 1000$), applying light inpainting with $\sigma_s = 1000$ and $\sigma_e = 700$ significantly reduces both reconstruction errors. This stems from the fact that inpainting at high noise levels dictates the general structure of the motion and accounts for the large differences between the synthesized and base motion. For stronger inpainting schedules, the reconstruction error improves linearly with the strongest schedule of $\sigma_s = 300$ and $\sigma_e = 50$ achieving the best metrics. Note that we always include some inference steps without inpainting to ensure that the motion model resolves any interpolation artifacts. We find that the schedule with $\sigma_s = 500$ and $\sigma_e = 50$ best balances maintaining $\mathcal{M}_{base}$ and performing natural-looking edits.

Additionally, we see that noise-inversion with 25 evaluation steps performs similarly to an inpainting schedule with $\sigma_s = 700$ and $\sigma_e = 500$, which we find is insufficient to maintain the details from $\mathcal{M}_{base}$. While increasing the number of evaluation steps improves reconstruction error, we list results with 25 evaluation steps to be consistent with our scheduled inpainting results. We refer to Appendix D for analysis of evaluation steps versus reconstruction quality for noise-inversion.

Similarly, 20 optimization steps, which need 10.3 sec to compute, with DNO only match an inpainting schedule of $\sigma_s = 1000$ and $\sigma_e = 500$ (0.192 sec) and are insufficient to reconstruct details in $\mathcal{M}_{base}$. While more optimization steps help improve reconstruction, note that each step requires evaluating the entire diffusion process. Although this optimization is only required once for each $\mathcal{M}_{base}$, it hinders more experimental tasks such as searching for sub-clips within long motion capture sequences. We provide an analysis of increasing gradient steps in Appendix E.

Table 2: Reconstruction error using different inpainting schedule with IBMM model for inbetweening motion clips on our internal and LaFan1 [HYNP20] datasets. CondEditor, noise-inversion [SME22] and DNO [KPA*24] all result in too high errors for interactive motion editing while our method allows user control over the level of preservation and editing.

σ_s	σ_e	Internal		LaFan1		Time (s)
		L2P ↓	L2R ↓	L2P ↓	L2R ↓	
300	50	0.003	0.047	0.004	0.044	0.192
500	50	0.004	0.051	0.005	0.049	
700	500	0.014	0.075	0.017	0.071	
1000	500	0.028	0.091	0.033	0.089	
1000	700	0.049	0.109	0.058	0.108	
1000	1000	0.173	0.157	0.178	0.141	
CondEditor		0.035	0.148	-	-	0.192
Noise-inversion		0.017	0.069	0.018	0.068	4.5
DNO		0.030	0.156	0.029	0.128	10.3

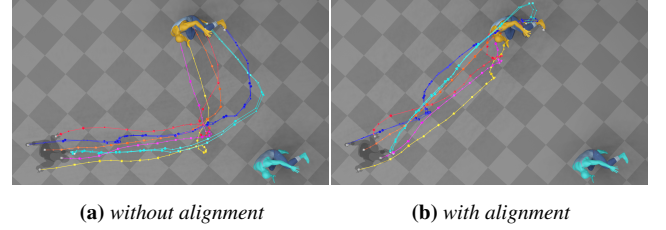


Figure 11: Effect of aligning motions during inpainting. Without alignment, the character unnaturally moves towards the original keyframe (in cyan) before warping to the target keyframe. In contrast, with alignment the motion remains natural and consistent.

Lastly, CondEditor similarly exhibits large reconstruction errors while struggling to perform realistic edits, as described earlier.

5.3.2. Motion Alignment

In Figure 11, we edit a forward roll’s end frame to demonstrate the importance of aligning the base motion and generated motions before inpainting. When inpainting without aligning (a), the resulting motion drifts towards the original end frame before warping unnaturally. Aligning forward directions (b), the motion moves naturally and consistently towards the new constraint.

5.3.3. Influence Radius μ_{c_i}

Figure 12 edits a 90-frame ‘walk forward’ motion to avoid the obstacle using a fixed schedule ($\sigma_s = 500$ and $\sigma_e = 50$). Increasing the radius of influence μ_{c_i} from 5 to 40 smoothly expands the number of adjusted frames, as seen by the motion curves. Even for $\mu_{c_i} = 40$, the Gaussian kernel allows sufficient inpainting to preserve the base motion’s style and cadence.

5.3.4. Kernel Shape

We ablate the shape of the kernel $G(j, t; c_i)$ defined in Eq. (4), used to construct the spatiotemporal mask for direct manipulation. We

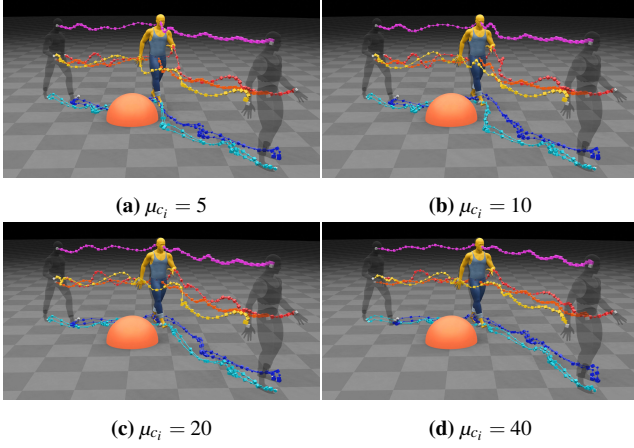


Figure 12: We can see the effect of the influence radius μ_{c_i} when editing to avoid an obstacle. Notice how more frames are adapted to avoid the obstacle, as the radius increases.

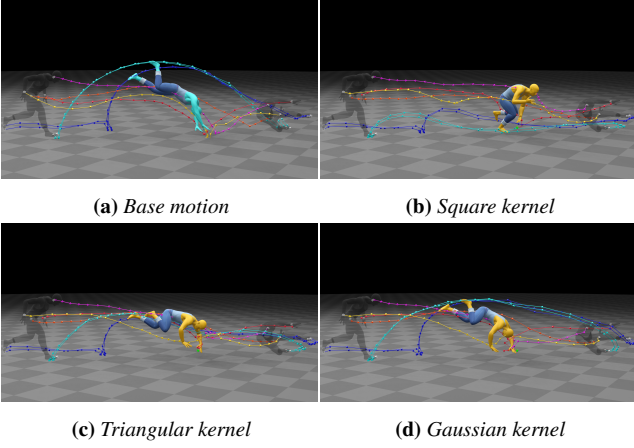


Figure 13: Ablating the weight decay kernel function for direct manipulation. While the piecewise linear and gaussian kernels can recreate the roll, the square wave results in a ducking motion.

require that the partial inpainting at the kernel’s edges sufficiently anchors the generated motion to the base motion, so the edit follows a trajectory close to the original rather than diverging arbitrarily. Additionally, the resulting motion should not have abrupt changes in velocity or cadence at the kernel boundaries.

In Figure 13, we edit the forward roll replacing the Gaussian with a width-modulated square wave and a triangular kernel. With each kernel’s minimum value set to zero, the square wave inpaints no fraction of the roll pose, and the edit collapses into a ducking motion, as seen in Figure 13b. Raising the square wave’s minimum weight can recover the roll, but at the cost of manipulability, as the motion is then interpolated away from the user’s constraints. In contrast, both the Gaussian and triangular kernels retain the roll but using the gaussian kernel more closely matches the base motion.

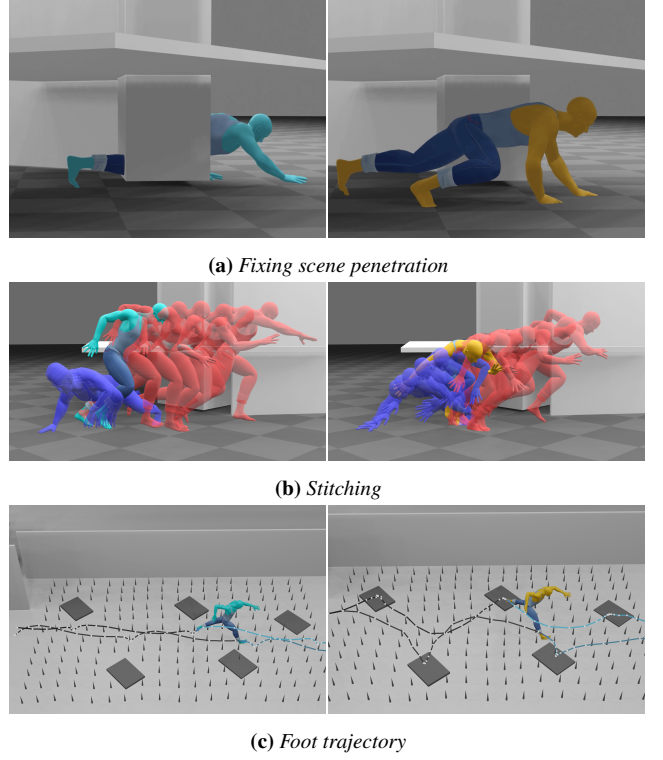


Figure 14: Before (left) and after (right) fixing scene penetration and stitching artifacts in the artist test scene.

5.4. Usability Test

To evaluate the usability of our method, we conducted a test with two professional digital artists tasked with animating a character through a parkour course. This task required running, crawling, and jumping, using a library of two running clips and one crawling motion clip. As shown in Figure 14, the provided motion capture contained artifacts such as environmental penetration (e.g., the character crawling through a pillar) and discontinuities at the boundary between naively stitched motions.

The artists were given one hour to fix these artifacts. Both participants reported that this time limit would be infeasible with traditional motion editing tools, but they successfully completed the task using our framework, as shown in our supporting video. During the process, both artists leveraged the capability to preserve specific motion segments while sampling variations for the rest. One of the artists noted that the transition from crawling to running would traditionally require significant manual effort, while it was automatic using our method. Additionally, they reported that the editing process being non-destructive was an advantage.

The test also identified areas for future refinement. While neural motion curves [ABB*24] were intuitive, a better interface was needed for interacting with foot contact points. Furthermore, due to the computational overhead of the full diffusion process, the artists had to adapt to a lower frame rate compared to traditional interpolation. However, it is worth noting that slower frame rates are standard in professional animation workflows utilizing complex non-

linear rigs. Finally, we observed that when generating sequences significantly longer than those in the training set, the method’s ability to closely preserve the base motion begins to degrade.

6. Discussion & Limitations

In the context of image editing, interactive generative editing capabilities, such as generative fill, image extension, object removal, image blending, and stylization, have found great success in professional software. While interactive manipulation and motion editing have each been explored individually within generative motion research, combining these applications holds vast potential for creating similar transformative workflows in professional animation, increasing reusability of existing data and reducing artist effort, but remains underexplored. To this end, our intuitive method addresses this gap by implementing important applications in animation, including direct manipulation, stitching, and compositing, within a single framework. This allows users to seamlessly switch between applications or even combine multiple together, as shown in our supporting video, where we combine direct manipulation with extension and stitching.

Inference-time techniques for pretrained models have been explored in previous works, such as inpainting in MDM [TRG*23] and blending in PriorMDM [STKB24]. However, the binary inpainting mask used in MDM renders the inpainted section completely unmodifiable. Similarly, PriorMDM leverages the blending of boundaries between different samples to generate longer sequences, but crucially without any manipulation control. Consequently, both methods restrict an artist’s ability to fine-tune the generated motion. In contrast, our method extends these techniques to expose granular control over both the inpainting schedule and the spatial regions to the user. This maximizes user agency over the preservation, blending, and manipulation of the motion within a single, unified framework.

As previously discussed, DDIM inversion requires many evaluation steps to generate high quality samples. To accelerate this process, prior work [SXL*24] has explored partially inverting to timestep $t \ll T$. Similarly to our method, this results in a mixture of base motion and noise at timestep t . However, unlike these methods, which lack fine-grained spatial control over preservation and generation, our method provides intuitive spacetime control, allowing more creative freedom.

To enable direct manipulation, we focused on models such as IBMM [VAG*25] and SF-control [HBLK25], which provide more granular interaction than models relying solely on full poses and trajectory constraints such as CondMDI [CTR*24]. Nonetheless, as we build upon pretrained diffusion models, we inherit their limitations as well. Most notably, reconstructing movements with a generative model results in a small loss of high-frequency details, which by extension affects the edited movements we generate with our method. Furthermore, preserving a base motion and satisfying new constraints are inherently adversarial tasks, which can cause some generative models to begin ignoring constraints, as the level of preservation increases.

Generative models are inherently limited by the motion distribution on which they were trained. Therefore, when a user edit

goes out of the training distribution, generative motion authoring can produce unexpected results, causing users to spend time trying to create a motion infeasible for the model. In the future, it would be valuable to explore visualizing feasibility for a given model.

Finally, we have shown that simply overwriting different clips for stitching and compositing already results in natural motion with our method. However, by integrating traditional authoring tools one can leverage layer-based editing to create base motion with fewer starting artifacts, further improving the generative results and potentially unlocking new workflows.

7. Conclusion

In this work, we introduced a new motion editing task that expands motion editing with generative capabilities while simultaneously allowing for interactive manipulation. To achieve this, we introduce scheduled inpainting, an inference framework that supports many real-time editing applications, from stitching and extending to compositing diverse movements and generating stylized motion, while consistently resulting in natural motion. Our inference-based approach is training-free and compatible with any pretrained interactive motion diffusion model and dataset.

Animating and capturing movements is famously challenging and time-consuming. Generative motion authoring significantly lowered the entry barrier. However, it lacked the ability to preserve movements. By unifying preservation and generation into a shared workflow, we bring generative motion authoring closer to solving real-world production tasks. We hope that our work will inspire and serve as a foundation for new workflows leveraging such generative models for real world tasks in VFX such as motion editing, enabling creatives with different backgrounds and skill sets to achieve their full creative visions more directly and intuitively.

References

- [ABB*24] AGRAWAL, DHURV, BUHMANN, JAKOB, BORER, DOMINIK, et al. “SKEL-Between: A Neural Motion Rig for Interactive Motion Authoring”. *ACM Transactions on Graphics (TOG)* 43.6 (2024), 1–11 [2](#), [6](#), [9](#).
- [ACD*24] ATHANASIOU, NIKOS, CSEKE, ALPÁR, DIOMATARIS, MARKOS, et al. “MotionFix: Text-driven 3D Human Motion Editing”. *SIGGRAPH Asia 2024 Conference Papers*. 2024, 1–11 [2](#), [3](#), [6](#).
- [Aut25] AUTODESK INC. *Autodesk MotionBuilder*. Version 2025, Autodesk, Inc., San Rafael, CA. 2025. URL: <https://www.autodesk.com/products/motionbuilder/> [2](#).
- [BHL*25] BAE, JINSEOK, HWANG, INWOO, LEE, YOUNG-YOON, et al. “Less is More: Improving Motion Diffusion Models with Sparse Keyframes”. *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2025, 11069–11078 [2](#).
- [CH07] CHAI, JINXIANG and HODGINS, JESSICA K. “Constraint-based motion optimization using a statistical dynamic model”. *ACM SIGGRAPH 2007 papers*. 2007 [2](#).
- [Che24] CHEN, XINGYU. “Text-driven Human Motion Generation with Motion Masked Diffusion Model”. *arXiv preprint arXiv:2409.19686* (2024) [2](#).
- [CLD*24] CHEN, LING-HAO, LU, SHUNLIN, DAI, WENXUN, et al. “Pay attention and move better: Harnessing attention for interactive motion generation and training-free editing”. *arXiv preprint arXiv:2410.18977* (2024) [3](#).

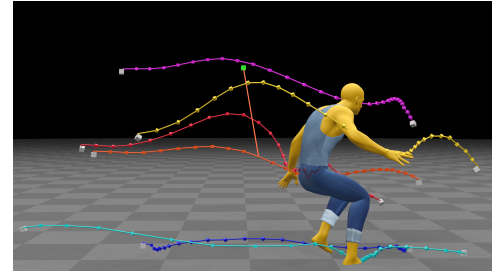
- [CTR*24] COHAN, SETAREH, TEVET, GUY, REDA, DANIELE, et al. “Flexible motion in-betweening with diffusion models”. *ACM SIGGRAPH 2024 Conference Papers*. 2024, 1–9 **2, 6, 10, 12**.
- [GHSZ25] GUO, ZIYAN, HU, ZEYU, SOH, DE WEN, and ZHAO, NA. “MotionLab: Unified Human Motion Generation and Rediting via the Motion-Condition-Motion Paradigm”. *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2025, 13869–13879 **2, 3, 6, 7**.
- [Gle97] GLEICHER, MICHAEL. “Motion editing with spacetime constraints”. *Proceedings of the 1997 symposium on Interactive 3D graphics*. 1997, 139–ff **2**.
- [GvdPH25] GOU, RUIYU, van de PANNE, MICHIEL, and HOLDEN, DANIEL. “Control Operators for Interactive Character Animation”. *ACM Transactions on Graphics (TOG)* 44.6 (2025), 1–20 **2**.
- [GZLF25] GOEL, PURVI, ZHANG, HAOTIAN, LIU, C KAREN, and FATAHALIAN, KAYVON. “Generative Motion Infilling from Imprecisely Timed Keyframes”. *Computer Graphics Forum*. Wiley Online Library. 2025, e70060 **2**.
- [HBLK25] HWANG, INWOO, BAE, JINSEOK, LIM, DONGGEUN, and KIM, YOUNG MIN. “Motion Synthesis with Sparse and Flexible Keyjoint Control”. *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. 2025 **2, 3, 6, 10**.
- [Hou25] HOUDINI. *KineFX*. 2025. URL: <https://www.sidefx.com/docs/houdini/character/kinefx/index.html> **2**.
- [HS22] HO, JONATHAN and SALIMANS, TIM. “Classifier-free diffusion guidance”. *arXiv preprint arXiv:2207.12598* (2022) **14**.
- [HWY*24] HUANG, YIMING, WAN, WEILIN, YANG, YUE, et al. “COMO: Controllable motion generation through language guided pose code editing”. *European Conference on Computer Vision*. Springer. 2024, 180–196 **3**.
- [HYNP20] HARVEY, FÉLIX G, YURICK, MIKE, NOWROUZEZAHRAI, DEREK, and PAL, CHRISTOPHER. “Robust motion in-betweening”. *ACM Transactions on Graphics (TOG)* 39.4 (2020), 60–1 **6, 8**.
- [JLY*25] JIANG, NAN, LI, HONGJIE, YUAN, ZIYE, et al. “Dynamic motion blending for versatile motion editing”. *Proceedings of the Computer Vision and Pattern Recognition Conference*. 2025, 22735–22745 **3**.
- [KPA*24] KARUNRATANAKUL, KORRAWE, PREECHAKUL, KONPAT, AKSAN, EMRE, et al. “Optimizing diffusion noise can serve as universal motion priors”. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2024, 1334–1345 **2, 3, 6–8, 13, 14**.
- [LCG*25] LI, ZHENGYUAN, CHENG, KAI, GHOSH, ANINDITA, et al. “SimMotionEdit: Text-Based Human Motion Editing with Motion Similarity Prediction”. *Proceedings of the Computer Vision and Pattern Recognition Conference*. 2025, 27827–27837 **3**.
- [LCZ*25] LI, JIEFENG, CAO, JINKUN, ZHANG, HAOTIAN, et al. “GENMO: A GENeralist Model for Human MOTion”. *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2025, 11766–11776 **2**.
- [LHS*25] LI, RONGHUI, HAN, LINGXIAO, SHU, SHI, et al. “A Motion is Worth a Hybrid Sentence: Taming Language Model for Unified Motion Generation by Fine-grained Planning”. *Proceedings of the 33rd ACM International Conference on Multimedia*. 2025, 1404–1413 **2**.
- [LS99] LEE, JEHEE and SHIN, SUNG YONG. “A hierarchical approach to interactive motion editing for human-like figures”. *Proceedings of the 26th annual conference on Computer graphics and interactive techniques*. 1999, 39–48 **2**.
- [LXY*24] LIU, HAOFENG, XU, CHENSHU, YANG, YIFEI, et al. “Drag your noise: Interactive point-based editing via diffusion semantic propagation”. *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2024, 6743–6752 **3**.
- [MCC09] MIN, JIANYUAN, CHEN, YEN-LIN, and CHAI, JINXIANG. “Interactive generation of human animation with deformable motion models”. *ACM Transactions on Graphics (TOG)* 29.1 (2009), 1–12 **2**.
- [MCZT24] MA, SIHAN, CAO, QIONG, ZHANG, JING, and TAO, DACHENG. “Contact-aware Human Motion Generation from Textual Descriptions”. *arXiv preprint arXiv:2403.15709* (2024) **2**.
- [MGT*19] MAHMOOD, NAUREEN, GHORBANI, NIMA, TROJE, NIKOLAUS F., et al. “AMASS: Archive of Motion Capture as Surface Shapes”. *International Conference on Computer Vision*. Oct. 2019, 5442–5451 **6**.
- [OBH*22] ORESHKIN, BORIS N., BOCQUELET, FLORENT, HARVEY, FELIX G., et al. “ProtoRes: Proto-Residual Network for Pose Authoring via Learned Inverse Kinematics”. *International Conference on Learning Representations*. 2022 **2**.
- [PSK*25] PINYOANUNTAPONG, EKKASIT, SALEEM, MUHAMMAD USAMA, KARUNRATANAKUL, KORRAWE, et al. “MaskControl: Spatio-Temporal Control for Masked Motion Synthesis”. *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. 2025 **2**.
- [RBL*22] ROMBACH, ROBIN, BLATTMANN, ANDREAS, LORENZ, DOMINIK, et al. “High-resolution image synthesis with latent diffusion models”. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2022, 10684–10695 **3**.
- [RBP*25] RUIZ-PONCE, PABLO, BARQUERO, GERMAN, PALMERO, CRISTINA, et al. “MixerMDM: Learnable composition of human motion diffusion models”. *Proceedings of the Computer Vision and Pattern Recognition Conference*. 2025, 12380–12390 **2**.
- [SAB*24] STUDER, JUSTIN, AGRAWAL, DHARUV, BORER, DOMINIK, et al. “Factorized Motion Diffusion for Precise and Character-Agnostic Motion Inbetweening”. *Proceedings of the 17th ACM SIGGRAPH Conference on Motion, Interaction, and Games*. 2024, 1–10 **2, 12**.
- [SHW24] SADAT, SEYEDMORTEZA, HILLIGES, OTMAR, and WEBER, ROMANN M. “Eliminating oversaturation and artifacts of high guidance scales in diffusion models”. *The Thirteenth International Conference on Learning Representations*. 2024 **14**.
- [SME22] SONG, JIAMING, MENG, CHENLIN, and ERMON, STEFANO. “Denoising Diffusion Implicit Models”. *International Conference on Learning Representations*. 2022 **2, 3, 6–8, 13**.
- [STKB24] SHAFIR, YONATAN, TEVET, GUY, KAPON, ROY, and BERMANO, AMIT. “Human motion diffusion as a generative prior”. *International Conference on Learning Representations*. Vol. 2024. 2024, 8717–8733 **2, 10**.
- [SWJ*24] SHI, YI, WANG, JINGBO, JIANG, XUEKUN, et al. “Interactive Character Control with Auto-Regressive Motion Diffusion Models”. *ACM Transactions on Graphics (TOG)* 43.4 (2024), 1–14 **2**.
- [SXL*24] SHI, YUJUN, XUE, CHUHUI, LIEW, JUN HAO, et al. “DragDiffusion: Harnessing diffusion models for interactive point-based image editing”. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2024, 8839–8849 **3, 10**.
- [TCL23] TSENG, JONATHAN, CASTELLON, RODRIGO, and LIU, KAREN. “EDGE: Editable dance generation from music”. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2023, 448–458 **1, 2**.
- [TGN*24] TESSLER, CHEN, GUO, YUNRONG, NABATI, OFIR, et al. “MaskedMimic: Unified Physics-Based Character Control Through Masked Motion Inpainting”. *ACM Transactions on Graphics (TOG)* 43.6 (2024), 1–21 **2**.
- [TRG*23] TEVET, GUY, RAAB, SIGAL, GORDON, BRIAN, et al. “Human Motion Diffusion Model”. *The Eleventh International Conference on Learning Representations*. 2023 **1, 2, 10**.
- [VAG*25] VÖGELI, LUCA, AGRAWAL, DHARUV, GUAY, MARTIN, et al. “Implicit Bézier Motion Model for Precise Spatial and Temporal Control”. *Proceedings of the 2025 18th ACM SIGGRAPH Conference on Motion, Interaction, and Games*. 2025, 1–10 **2, 3, 6, 10**.
- [WDA*24] WANG, XI, DUFOUR, NICOLAS, ANDREOU, NEFELI, et al. “Analysis of Classifier-Free Guidance Weight Schedulers”. *Transactions on Machine Learning Research* (2024). ISSN: 2835-8856. URL: <https://openreview.net/forum?id=SUMtDJqicd> **14**.

- [WMZ*25] WANG, RUNQI, MA, CAOYUAN, ZHAO, JIAN, et al. "Leader is Guided: Interactive Motion Generation via Lead-Follow Paradigm and Trajectory Guidance". *Proceedings of the 33rd ACM International Conference on Multimedia*. 2025, 1802–1811 2.
- [WP95] WITKIN, ANDREW and POPOVIC, ZORAN. "Motion warping". *Proceedings of the 22nd annual conference on Computer graphics and interactive techniques*. 1995, 105–108 2.
- [WXD*25] WU, BIZHU, XIE, JINHENG, DING, MEIDAN, et al. "FineMotion: A Dataset and Benchmark with both Spatial and Temporal Annotation for Fine-grained Motion Generation and Editing". *arXiv preprint arXiv:2507.19850* (2025) 3.
- [XFHH25] XU, WENNING, FAN, SHIYU, HENDERSON, PAUL, and HO, EDMOND SL. "Multi-Person Interaction Generation from Two-Person Motion Priors". *Proceedings of the Special Interest Group on Computer Graphics and Interactive Techniques Conference Conference Papers*. 2025, 1–11 2.
- [ZGX*25] ZHONG, LEI, GUO, CHUAN, XIE, YIMING, et al. "Sketch2Anim: Towards Transferring Sketch Storyboards into 3D Animation". *ACM Transactions on Graphics (TOG)* 44.4 (2025), 1–15 2.
- [ZLC*23] ZHANG, MINGYUAN, LI, HUIRONG, CAI, ZHONGANG, et al. "FineMoGen: Fine-Grained Spatio-Temporal Motion Generation and Editing". *Advances in Neural Information Processing Systems* 36 (2023), 13981–13992 3.
- [ZWM*25] ZHANG, ZEYU, WANG, YIRAN, MAO, WEI, et al. "Motion Anything: Any to Motion Generation". *arXiv preprint arXiv:2503.06955* (2025) 2.

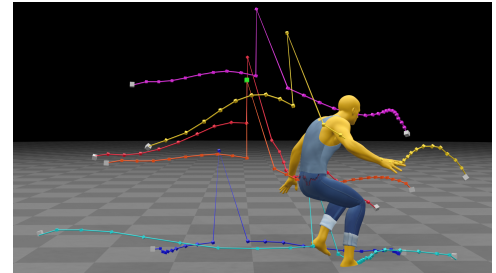
Appendix A: Scheduled Inpainting for Constraints

Dense motion representation models such as CondMDI [CTR*24] struggle at click-and-drag interactions due to the sparsity of point constraints in the input [SAB*24]. As shown in Figure 15a, a single hip constraint fails to trigger a jump, resulting in large errors and a running sequence. Using simple inpainting results in harsh peaks in the generated motion (see Figure 15b) although the constrained joint now satisfies the condition, it does not address the sparsity of information that causes the conditions to be ignored. Consequently, the neighboring frames are not updated correctly.

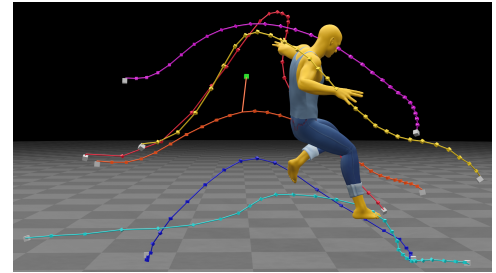
In contrast, we propagate constraints to neighboring frames using a Gaussian kernel. We schedule the kernel width to shrink with the diffusion noise such that broad kernels guide global trajectory at high noise, while narrow kernels enforce local constraints at low noise. This results in the neighboring frames being updated according to the point constraints and producing a natural jumping motion, as seen in Figure 15c.



(a) Base CondMDI

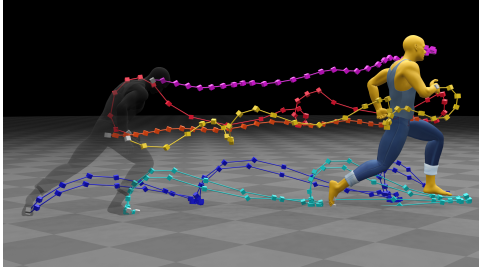


(b) Standard Inpainting

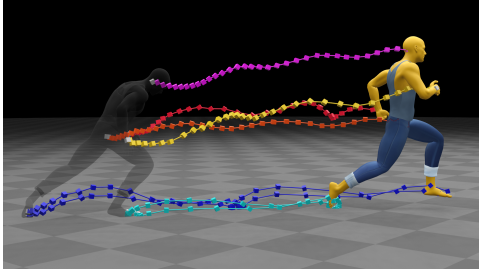


(c) Scheduled Inpainting (Ours)

Figure 15: Comparison of using sparse constraints with CondMDI [CTR*24]. By default, CondMDI completely ignores the sparse hip constraint (in green), while using standard inpainting results in large spikes. Using scheduled inpainting better follows the constraint while creating smooth motion.



(a) Without normalization



(b) With normalization

Figure 16: We edit a running clip to run half the distance of the original sequence. Without normalization, the base motion results in an unnatural warp to the final frame with foot sliding. In contrast, with normalization, the resulting motion does not have such artifacts.

Appendix B: Ablating Normalization

When moving the boundary frames to stretch or squash an original animation, the overall scale of the animation changes for inpainting. If this is ignored, it can cause artifacts in the generated motion. In Figure 16, we squash a running forward sequence to run only half of the original distance. Without normalization, this edit creates artifacts where the motion goes past the final frame and then slides backwards to the target frame. In contrast, with normalization the motion correctly adjusts the scale and generates a slower running motion.

Appendix C: Editing Outside the Convex Hull

We find that current generative motion models struggle to sample highly exaggerated or periodic motions that are far away from the input conditions, such as the ‘smashing an object’ action shown by the cyan character in Figure 17. We refer to this loss of variability as the *convex hull problem for motion generation*, as the generated motion is mostly bound within the convex hull of the input conditions. Consequently, when reconstructing such motion using our inpainting method, the regenerated motion loses the periodic dynamics, as seen in the middle of Figure 17.

To better reconstruct such motions, we automatically detect and add the extrema points on the base motion as additional constraints for the motion model. Given these constraints, the base motion is now within the convex hull of the conditions, and hence the motion model can reconstruct it more closely. In Figure 17, this results in the motion curves of the right character matching more

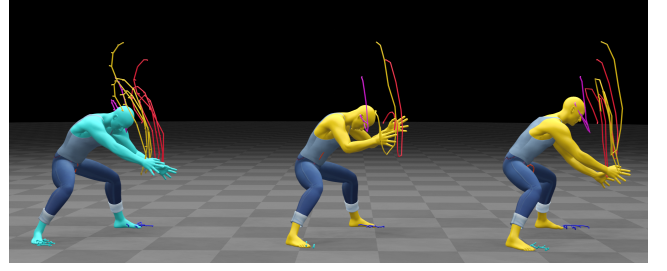


Figure 17: From an exaggerated base motion (left), we can see how automatically adding some sparse constraints with our method (shown on the right), better re-constructs the motion than without any constraint (shown in the middle).

closely those of the cyan character than those of the middle character. While those additional constraints help to better match the original base motion, they also come at the cost of managing them when one wants to edit on top. We hope that future research can better mitigate this convex hull problem to avoid this additional use of added constraints.

Appendix D: Number of Diffusion Steps for Noise-inversion

In Figure 18, we use DDIM noise inversion to invert a jumping clip using different number of evaluation steps. We use the same number of steps for the inversion and the denoising processes, since we observed that using different numbers of steps resulted in artifacts in our experiments. As seen in the figure, using fewer than 400 steps results in large offsets between the ground truth in cyan and the generated motion in white. Interestingly, for this example, 600 evaluation steps results in the best performance with more steps also hurting reconstruction.

DDIM-Inversion [SME22] assumes that the noise added in time steps t and $t + 1$ is almost identical. Therefore, it uses the noise removed by the motion model ϕ at t as the noise to be added to create the noised motion at timestep $t + 1$. However, this assumption does not hold when using much fewer diffusion steps. Consequently, the inverted motion does not closely match the base motion.

Appendix E: Optimization Steps for DNO

In Table 3, we measure the number of optimization steps required for DNO [KPA*24] to match the reconstruction performance of our recommended inpainting schedule ($\sigma_s = 500$ & $\sigma_e = 50$) on our internal dataset. We see that even after 100 optimization steps, which need ~ 50 seconds to compute a batch of 256 samples, DNO performs marginally worse than our method. However, the computation time makes DNO too slow for more experimental workflows, where artists try out different edits on different clips to see which best fits their vision. In contrast, we can better reconstruct base motion in just one inference process, which requires 0.192 seconds for the same batch size, and additionally allow the user to control the level of preservation desired.

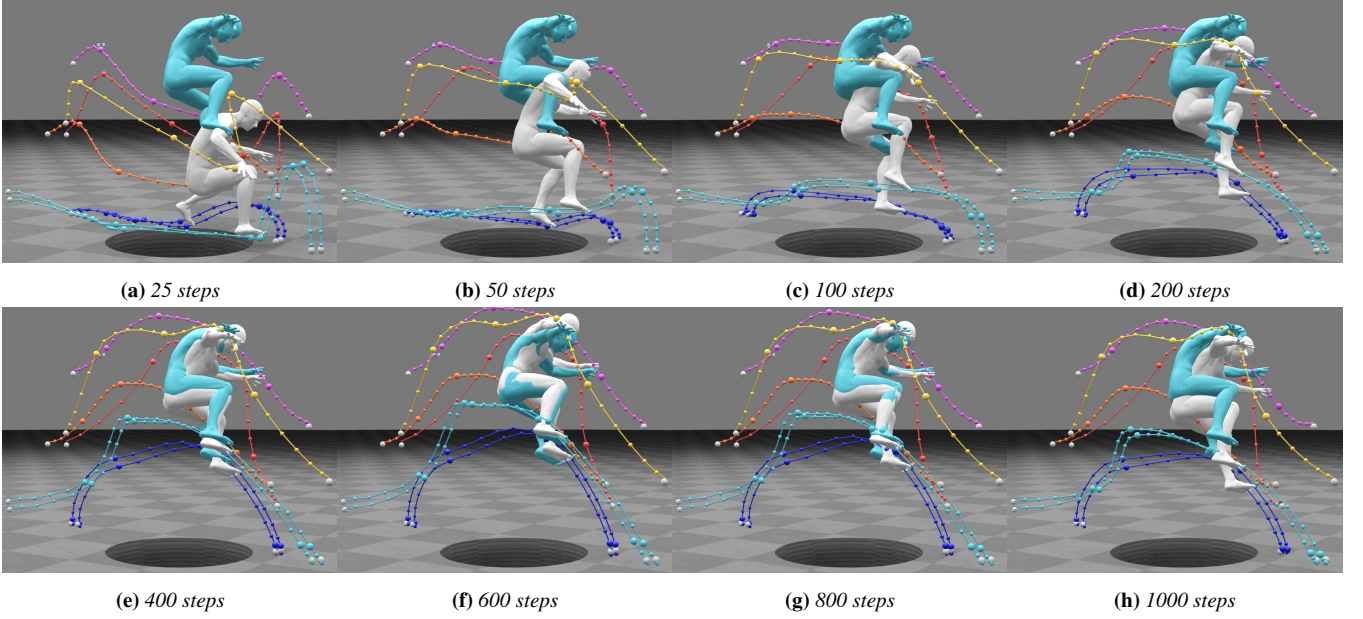


Figure 18: Number of steps required for noise-inversion. Evaluating too few steps results in the inverted motion (white) not matching the ground truth (cyan) in height. At higher number of evaluation steps, the method is not real-time as shown in our video.

Table 3: Measuring the reconstruction error on our internal dataset for different amounts of optimization steps using DNO [KPA*24]. Even with 100 optimization steps, our scheduled inpainting gives better reconstruction with much faster compute time.

#of optimization steps	L2P ↓	L2R ↓	Time (s) ↓
20	0.0303	0.1424	10.306
40	0.0162	0.1234	20.032
60	0.0104	0.1064	29.553
80	0.0077	0.0921	39.179
100	0.0060	0.0812	49.129
Ours ($\sigma_s = 500$ & $\sigma_e = 50$)	0.0044	0.0516	0.192

task as inpainting, we allow the user to manipulate the schedule for closer control.

When applying CFG, a common exploit is to set the interpolation coefficient to greater than 1. However, we find that for scheduled inpainting, setting $\alpha^t > 1$ in Eq. (1) results in artifacts where the generated motion goes beyond the convex hull of the base motion, similar to over-saturation in images. Sadat et al. [SHW24] show that adding only the orthogonal component for CFG, which they call APG, eliminates over-saturation in images. We leave exploring parallels of APG for scheduled inpainting for future work.

Appendix F: Connections to Classifier Free Guidance

Classifier-Free Guidance (CFG) [HS22], moves the conditioned generated motion away from an unconditionally generated motion to better satisfy the given conditions. Applying CFG with a static weight results in over-saturation in the generated samples. Consequently, previous works have explored applying CFG on different interpolation schedules. Wang et al. [WDA*24] provide an analysis of these different schedules. However, these schedules are quite simple and fixed, i.e. not interactive for users. For motion generation, in place of evaluating the model unconditionally, one could consider $-\mathcal{M}_{base}$ as a fixed unconditional sample that the motion generation should move away from. This results in a formulation similar to our scheduled inpainting. But the models need to be specifically trained for CFG while our method can be applied to any motion generation model. Additionally, by formulating the