Cartoon Motion Model

GUILHERME G. HAETINGER, ETH Zürich & DisneyResearch|Studios, Switzerland
DHRUV AGRAWAL, ETH Zürich & DisneyResearch|Studios, Switzerland
AMIT H. BERMANO, ETH Zürich & DisneyResearch|Studios, Switzerland
ROBERT SUMNER, DisneyResearch|Studios & ETH Zürich, Switzerland
MARTIN GUAY, DisneyResearch|Studios, Switzerland
JAKOB BUHMANN, DisneyResearch|Studios, Switzerland

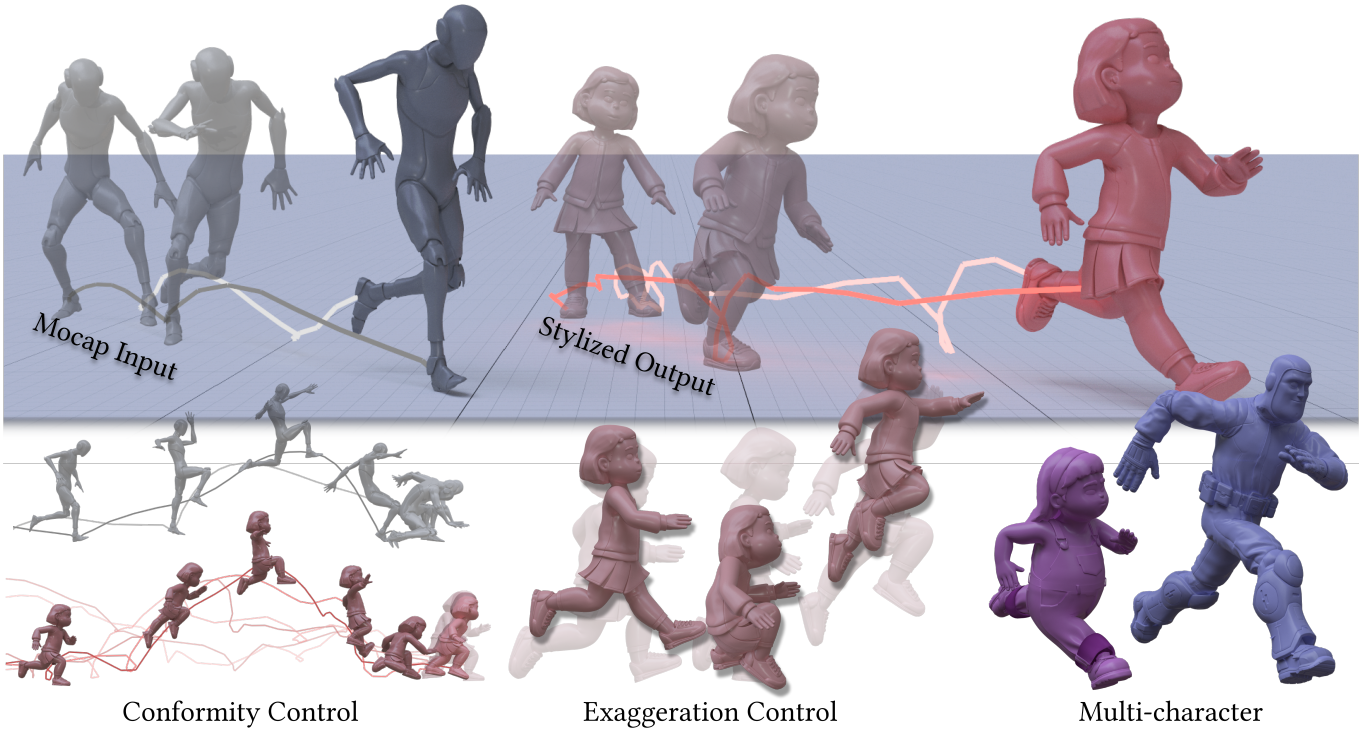


Fig. 1. Our generative motion system stylizes neutral mocap clips (left) into movement of hand-animated characters (right). The system offers control over the amount of conformity to the mocap clip, exposes a way to exaggerate the stylization, and works for multiple characters from the training dataset. ©Disney/Pixar.

Generative 3D motion has shown impressive capabilities for authoring movements, but the integration into animation production is currently hindered by the discrepancy between realistic human motion capture and the exaggerated, non-physical movements of stylized animated feature characters. Nonetheless, hand-animated data is sparse, heterogeneous and incomplete, making it a challenging domain for data-hungry generative models. In this work, we introduce a framework that enables generation and control of stylized 3D motion by leveraging a pre-trained human motion prior. Our approach introduces robust data filtering techniques to handle inconsistent and noisy production movements. We train an unconditional ControlNet that eases adaptation by optimizing for maximum alignment between the motion capture and hand-animated domains. The generations of such model are also controllable through a spectral inpainting scheme that exposes parameters for exaggeration, source consistency and multi-character identities. We bridge the gap between physical conformity and artistic stylization and enable a diverse set of applications ranging from retargeting human performances to character-to-character motion transfer.

Authors' Contact Information: Guilherme G. Haetinger, haetinger@disneyresearch.com, ETH Zürich & DisneyResearch|Studios, Zurich, Switzerland; Dhruv Agrawal, dhruv.agrawal@disneyresearch.com, ETH Zürich & DisneyResearch|Studios, Zurich, Switzerland; Amit H. Bermano, amit.bermano@gmail.com, ETH Zürich & DisneyResearch|Studios, Zurich, Switzerland; Robert Sumner, sumner@disneyresearch.com, DisneyResearch|Studios & ETH Zürich, Zurich, Switzerland; Martin Guay, martin.guay@disneyresearch.com, DisneyResearch|Studios, Zurich, Switzerland; Jakob Buhmann, jakob.buhmann@disneyresearch.com, DisneyResearch|Studios, Zurich, Switzerland.

This work is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 International License.
Conference acronym 'XX, Woodstock, NY
© 2018 Copyright held by the owner/author(s).
ACM ISBN 978-1-4503-XXXX-X/2018/06
https://doi.org/XXXXXXX.XXXXXXX

CCS Concepts: • **Computing methodologies → Machine learning**; **Animation**; **Artificial intelligence**; • **Human-centered computing → Interaction design**.

Additional Key Words and Phrases: Motion, Diffusion, Style Adapters

ACM Reference Format:
Guilherme G. Haetinger, Dhruv Agrawal, Amit H. Bermano, Robert Sumner, Martin Guay, and Jakob Buhmann. 2018. Cartoon Motion Model. In *Proceedings of Make sure to enter the correct conference title from your rights confirmation email (Conference acronym 'XX).* ACM, New York, NY, USA, 13 pages. https://doi.org/XXXXXXX.XXXXXXX

1 Introduction

Motion capture is widely used in live-action visual effects—not only for capturing performers, but also by animators as reference when animating. In contrast, in feature animation, motion capture is rarely employed due to the fundamental disparity between physically realistic motion and the highly stylized movements that define the medium. Animation as a genre is more flexible, creative, and expressive, allowing characters to exaggerate actions and poses by stretching, breaking anatomy, and violating physics—all in service of visual clarity and expressivity. Generative motion models have demonstrated strong capabilities for text-driven synthesis [Tevet et al. 2023], motion completion, and control from sparse constraints [Cohan et al. 2024; Vögeli et al. 2025]. These advances have sparked interest in integrating generative tools into animation workflows to enable rapid exploration during pre-production. However, current methods are trained exclusively on motion capture datasets and produce motions that, while physically plausible, lack the exaggeration of hand-animated characters. In this work, we bridge motion generation with stylistic timing, exaggeration, and personality, key features of hand-animated motion style.

Nevertheless, motion style has already been the focus of research in academia, although with a different definition resembling simple labels and crude characteristic of movements (*e.g.* "swing arms" and "airplane" [Aberman et al. 2020; Mason et al. 2022; Xia et al. 2015]). Besides being hard to attain at scale, hand animated data is complicated to work with, as it is sparse (often under tens of minutes per character), heterogeneous (with varying skeletal topologies and motifs) and often inconsistent. Animators often rely on production shortcuts that create broken shots outside of camera view. Thus, to make training feasible, we develop a data curation pipeline that addresses the unique challenges of production data: filtering static or explosively fast segments, detecting off-screen joints via ray casting, and modulating training losses based on joint visibility.

We present a generative model for cartoon motion trained on production hand-animated data. To compensate for data scarcity, we adapt a mocap-pretrained diffusion model to benefit from its learned motion manifold. However, due to the large domain gap, we find that standard fine-tuning or low-rank adaptation [Sawdayee et al. 2025] cannot be trained without overfitting or being restricted by its capabilities, respectively. Thus, we introduce a minimally coupled ControlNet adapter [Zhang et al. 2023] that transfers style residuals only at the final layer of the base network. The model does not require paired mocap and hand-animated data. Unlike standard ControlNet architectures that inject residuals at every layer, this minimal coupling prevents the adapter from dominating and collapsing the generation when fitting our small dataset. We condition the model with a character identity, adapting the output to each character's proportions and motifs. To enable spatial control during inference, we propose a spectral-domain inpainting scheme that decomposes motion into low-frequency content (trajectory, broad actions) and higher-frequency stylistic detail. The user controls the energy ratio between the input clip's low-frequency intent and the high-frequency stylization output. We also expose the user to an exaggeration control scalar by leveraging the ControlNet residual's magnitude. Fitting all the pieces together, we implement a controllable pipeline for adaptation of arbitrarily long motion capture sequences into a character's style through puppeteering. We validate our approach through quantitative evaluation, qualitative comparisons, and a user study. Overall, our contributions are:

- A generative model for hand-animated motion trained on a sparse and heterogeneous dataset and a given motion prior
- We discuss the inherent challenges of working with production data with view inconsistencies and non-physically grounded movements.
- A spectral inpainting control scheme that transforms mocap into a character's motion with user controlled exaggeration and conformity to the input.

2 Related Work

Given the constant need for expressive motion in games and feature films, significant efforts have gone into stylizing neutral motion.

Early methods used heuristics such as Laplacian of Gaussian filters [Wang et al. 2006] and physical simulation alterations [Chenney et al. 2002] with the aim of mimicking hand-animated character motion by injecting artistic principles such as anticipation and follow-through [Lasseter 1998; Thomas 1995]. Other methods even talked about retargeting animation captured from animated features into different media (3D structures, 2D drawings, photographs, etc.) [Bregler et al. 2002]. These rule-based and hand-constrained approaches produce convincing exaggeration for isolated effects but do not generalize across actions.

Subsequent efforts explored motion style transfer with machine learning using adaptive instance normalization [Huang and Belongie 2017] to remove an input clip's style before injecting the style of another clip [Aberman et al. 2020; Guo et al. 2025b; Park et al. 2021]. Tang et al. combined AdaIN with graph convolution networks and disentangled contact pattern from style to reduce foot sliding [Tang et al. 2024]. In the absence of paired neutral and stylized motion data, another line of work [Guo et al. 2024; Huang et al. 2022; Park et al. 2021] employs cycle-consistency loss [Zhu et al. 2017]. RSMT [Tang et al. 2023] and [Dai et al. 2025] leverage the cyclic nature of locomotion data using a periodic phase autoencoder [Starke et al. 2022]. Stylizing individual body parts has also been explored [Dai et al. 2025; Jang et al. 2022]. More recently, contrastive learning [Hadsell et al. 2006] has been used to separate style embeddings [Yang et al. 2025; Zargarbashi et al. [n. d.]], with Zargarbashi et al. applying contrastive learning to a RVQ-VAE [Lee et al. 2022]



Fig. 2. Visualization of the on screen visibility (green). The regions in red are either occluded or off-screen for this frame. ©Disney/Pixar.

to further disentangle style from content within the residual quantizations of an input motion. However, all these methods operate on locomotion data and do not extend to arbitrary actions.

As diffusion models demonstrated strong general-purpose generative capabilities for motion synthesis [Cohan et al. 2024; Hwang et al. 2025; Studer et al. 2024; Vögeli et al. 2025], they were also applied to stylizing motion. SMooDi [Zhong et al. 2024] adapts a pretrained text-conditioned motion generator using a ControlNet [Zhang et al. 2023] to inject a style encoding, steering the generation towards a target style. They additionally use classifier [Dhariwal and Nichol 2021] and classifier-free guidance [Ho and Salimans 2021] to enhance the style at inference time. Wu et al. presents a method for unsupervised semantically-aligned motion style learning that does not require labeled data or classifier-guidance. Instead, they use auxiliary losses to avoid content forgetting and keep the generated motion aligned with the content text while still training for the style condition, gating it into the model through a cross-attention mechanism. Since then, StyleMotif [Guo et al. 2025a] and MulSMo [Li et al. 2024] extend SMooDi for multi-modal motion stylization such as audio and images. Sawdayee et al. instead use a LoRA [Hu et al. 2022] to adapt an MDM model [Tevet et al. 2023] to a particular style. SMooGPT [Zhong et al. 2025] uses an LLM to generate text descriptions for a given style label which is then used to condition the generation.

While diffusion-based methods handle non-locomotion actions more robustly, they are trained on clean mocap datasets such as [Aberman et al. 2020], [Xia et al. 2015], and 100Styles [Mason et al. 2022], which consist primarily of locomotion and lack the noise, heterogeneity, and incompleteness of hand-animated data. Furthermore, their style categories do not reflect the production definition where style is a nuanced, character-specific quality orthogonal to the action. As we show in our experiments, without careful design, existing methods fail to learn from such data. To our knowledge, our method is the first to generate motion under a production definition of style, trained directly on hand-animated sequences.

3 Dataset Preparation for Animated Characters

Our system leverages a human motion prior to learn a style motion generator. Thus, it is trained on two data sources: a mocap dataset, containing ~40 hours of motion recorded for live action movies, and an ~8 hour highly stylistic dataset. The stylized clips are drawn from a background crowds dataset and hand-animated sequences from two distinct movies, each with their own unique animation style. The hand-animated clips span 8 characters, each unique in action space, style, proportions, and data volume. While a hero character could have up to an hour of data, most characters average 10 to 25 minutes (see Table 1).

Table 1. Number of filtered in frames and visibility percentage by character.

Name	Total Frames	Filtered In			Visible %
		Frames	Time (Min.)	%	
A.1	67,912	35,630	24.74	52	53.11
A.2	33,595	15,492	10.76	46	50.97
B.1	100,463	51,458	35.73	51	39.88
B.2	50,445	23,501	16.32	47	39.02
Crowds (C)	357,591	273,571	189.98	76	—

Data Alignment and Challenges. Hand-animated data poses unique challenges. Unlike motion capture, hand-animators freely edit bone scales to achieve "squash-and-stretch" effects [Lasseter 1998; Thomas 1995]. Moreover, animation is authored relative to a particular camera angle. Thus, the artist's stylistic and emotional intent conveyed from one angle may show artifacts from another. Lastly, since hand animation is extremely laborious, off-camera parts are often not polished to final production standard, producing artifacts such as overly fast transitions and unnatural stretching from locked IK handles. See Fig. 2 for a visualization of pose artifacts.

In this work, we simplify the motion space by retargeting all stylized and mocap data to a common skeleton and removing stretching by using fixed proportions per character. Although this adds some foot sliding on certain clips, it significantly simplifies the learning task, while preserving the rhythm and intent of the motion.

Data Filtering. Due to how characters are animated, hand-animated data is far more fragmented than mocap. Hand-animated character motion is split into movie shots and can range from being meticulously animated for heros to—more commonly—little to no meaningful movement for a background actor. Moreover, animated features are highly stylized, exhibiting characteristically unnatural and physically exaggerated motions such as running at incredible pace or cadence, jumping enormous heights, and bending joints without temporal interpolation. Such contrasting dataset of explosive movements and frozen characters challenges deep learning methods that rely on uniformly distributed manifolds.

Training purely on such data introduces irregular gradients, causing poor convergence in the model. Thus, we need to filter the data, not only to improve quality, but also to ensure training stability. To prevent catastrophic gradients, we filter data that moves too fast, or has too strong accelerations. Additionally, clips that show little to no animations are filtered out to prevent a bias for the system to predict too static motions. Details on the exact filtering parameters are explained in Appendix A.

Besides filtering subset of clips out, we also detect which joints are off-screen. To do so, we cast rays from the camera and detect

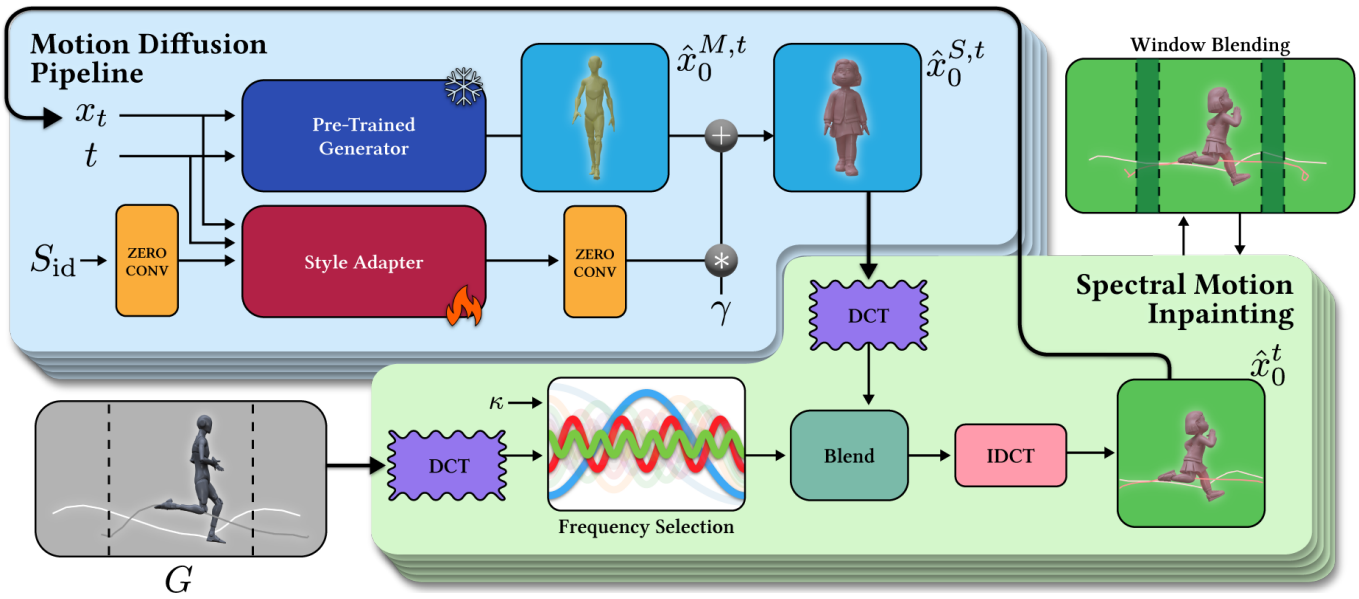


Fig. 3. Overview of our Style Adapter, consisting of a generative ControlNet (blue) and a spectral inpainting scheme (green) to stylized a given mocap clip G. Note that the meshes are only used for visualizing the domain of the underlying motion data. Window blending smoothly connects the batch of generated clips for longer sequences. ©Disney/Pixar

which region of the mesh is intersected. This is then mapped back to the joints through lookup of the skinning weights (see Fig. 2). This visibility flag can be directly exploited to filter out (or downscale) gradients in our loss formulation as described in Section 4.2.

After filtering the data, we end up with 5.3 hours of overall motion data, of which 46% is actually on screen. Thus, the amount of data is only at a fraction of the size of a typical motion data set used for training generative motion models such as HumanML 3D [Guo et al. 2022], which has ~29h of visible data. Lastly, after filtering, the median clip length is in the range of 1-2 seconds, while HumanML averages ~7.1 seconds long.

4 Method

In this section, we detail our Cartoon Adapter and inference-time puppeteering scheme. An overview of our method is shown in Fig. 3.

4.1 Motion Representation

As our stylized dataset consists of characters with vastly varying proportions, we opt to represent the motion using global root position and parent space orientations for the non-root joints. Compared to root-local or global representations, using parent-space naturally aligns the different proportions. We ablate this choice in Section 6.4 and show that we can stylize more consistently in orientation space.

Furthermore, similar to IBMM [Vögeli et al. 2025], we align each sample such that the planar mean of the sequence is at the origin and the direction from the first to the last frame is aligned to the Y-axis. Given the high heterogeneity of hand-animated data, we see that the ground plane may change within a single shot in our data. When sampling individual samples from a longer shot, this results in some shots floating in midair. Therefore, we additionally fix each sample to a common ground plane by removing the smallest height offset in the sample.

4.2 Cartoon Adapter

Our main architecture implements a simplified ControlNet [Zhang et al. 2023]. As our main contribution lies in the adapter design and the puppeteering scheme, we do not build our own base model architecture and, similar to MDM [Tevet et al. 2023], use a transformer encoder with additional hyperparameter details provided in Appendix B. This base model is trained on a large corpus of mocap data to build a strong prior on human motion. As we puppeteer the generation via a custom inpainting strategy, described in Section 4.3, we train the base model to generate motion unconditionally to best learn the manifold of the mocap dataset.

Next, we add a second copy of this trained network to the base model along with a new condition S_{id} that encodes character identity via one-hot encoding embedded using zero-convolution layers. However, we empirically find that following the conventional formulation of ControlNet, as in SMooDi [Zhong et al. 2024], where the residual from the adapter branch is added to the base branch at each layer, overfits to our small dataset. Therefore, we limit the modeling capacity of the Style Adapter by only transferring the residual at the very last layer, defining the entire transformer as one large network block.

As our dataset does not have paired stylized and unstylized motions with identical content, we train the Style Adapter using reconstruction on the stylized data. Therefore, we adopt the similar loss formulation as [Agrawal et al. 2024], but without the constraints-based losses. This entails an L2 loss on the global root trajectory and parent joint orientations between the generated and ground truth

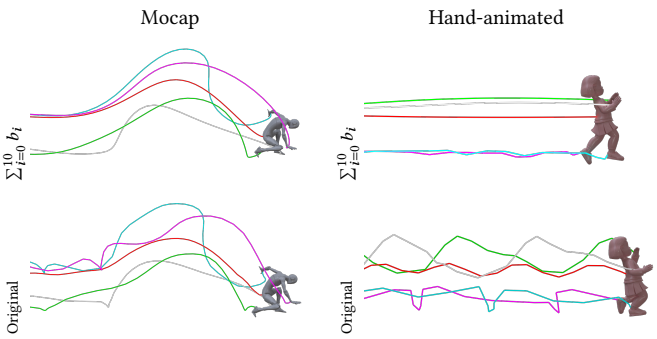


Fig. 4. Reconstructions from subset of DCT frequencies for ~100 frame sequences. Each column shows one sample. Rows show the reconstruction using bands up to b_1 and up to b_{10}, compared against the original clip. ©Disney/Pixar.

sequences, and an L2 loss on the global transforms for all joints, recovered using forward kinematics with the respective character's bone proportions. To ensure that the Style Adapter correctly adapts the motion to the specified character proportions, we also add a foot-sliding loss on the global feet positions[Tevet et al. 2023]. Lastly, to maximize the number of data points that can be used for training, we opted for a short window length of 30 frames.

Visibility mask. As described in Section 3, we computed the visibility flag for each joint at each frame. While the artist often do not animate a hidden joint, we empirically find that it is likely to be animated if its parent joint is on screen. Therefore, we use the joint's visibility flag and this observation to modulate the above mentioned losses based on its proximity to a visible joint. More specifically, we reduce its weight by a factor of 0.7 for every joint not visible along its kinematic chain.

Given the small volume of data available and only 46% of it visible (see Table 1), we find that leveraging more noisy hidden data helps to improve the final performance of our model.

4.3 Stylized Puppeteering

While the ControlNet adapter allows generating motion conditioned on the character identity S_{ID}, it does not enable any spatial control. Hence, we introduce a control mechanism based on inpainting but in a spectral domain, see green box in Fig. 3.

Spectral style inpainting. Given a source clip G, such as a mocap sequence, we aim to generate stylized motion following its high level semantics, such as direction, accelerations, and actions like jumping or sitting. Inpainting has proven effective for inbetweening [Tevet et al. 2023], clip stitching [Shafir et al. 2024], and matching spatial constraints [Rempe et al. 2026]. However, in these cases, specific frames must be maintained as accurately as possible, while other frames are fully generated. Yet in our case, we must modify the full motion clip while maintaining its high level features. Thus, we convert our representation from frame-based to spectral via a discrete cosine transform (DCT). As seen in Fig. 4, the main structure lives in the lower frequencies, while the higher frequencies capture the style.

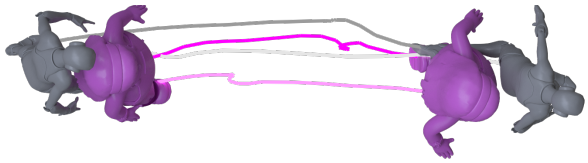


Fig. 5. Shortened path when inpainting a movement on a smaller character. ©Disney/Pixar.

To detect the transition from content to style frequencies, we measure the percentage of the energy of the motion explained by the first n frequencies in the driving motion, defined as the sum of their amplitudes. Given an energy threshold $0 < \kappa \leq 1$, we identify the smallest set of low-frequency components whose cumulative energy reaches a fraction κ of the total, and inpaint these components with the driving motion—preserving them as content—while letting the model freely generate the remaining high-frequency components as style. The corresponding mask M_κ^τ is constructed as

$$K_i = \frac{\sum_{j \leq i} |\mathrm{DCT}(G)|_j}{\sum_j |\mathrm{DCT}(G)|_j}, \quad M_\kappa^\tau = \mathbb{I}[K < \kappa], \tag{1}$$

where the cumulative sum and comparison are applied along the temporal axis.

To further smoothen this blending, we stop the inpainting early at the last inference steps, i.e. at low noise values $t < \tau$. This early stopping is particularly important when the character sizes are vastly different, since the first bands also maintain the overall coverage of motion, which needs to be adapted accordingly. For instance, as shown in Fig. 5, a small character covers less space in the same time and thus the motion needs to be scaled down by the ControlNet.

After blending the different frequency bands together, we convert the motion back to the frame-based space via inverse discrete cosine transform (IDCT) and feed it back into the inference diffusion loop. Thus, the full spectral style blending is formulated as follows:

$$\hat{x}_0^t = \mathrm{IDCT}\left(M_\kappa^\tau \otimes \mathrm{DCT}(G) + \left(1 - M_\kappa^\tau\right) \otimes \mathrm{DCT}\left(\hat{x}_0^{S,t}\right)\right), \tag{2}$$

where M_κ^τ is the mask for the lower and higher frequency bands constructed using Eq. (1).

Stylization signal. While the spectral blending allows to maintain higher frequency bands, the stylization of the final motion is, by default, weaker than for the unconditional samples from the ControlNet. To counter this loss in stylization, we modulate the weight of the ControlNet residual by increasing $\gamma > 1$ [Bhat et al. 2024]. Note that by going to $\gamma > 1$ we are extrapolating to motion outside the training data. While this can cause artifacts in full space, such as the root overshooting too much, since we maintain the low frequencies from G, the motion remains well behaved.

Sequence Blending. To generate longer sequences with our model, trained on 30 frames motion clips, we generate independent batches and blend them at each diffusion step, following Tseng et al. [2023]. In contrast to their method, our model adapts the motion to the scale of the specified character, as seen in Fig. 5. Thus, shortening or expanding each individual segment of the motion capture being stylized.

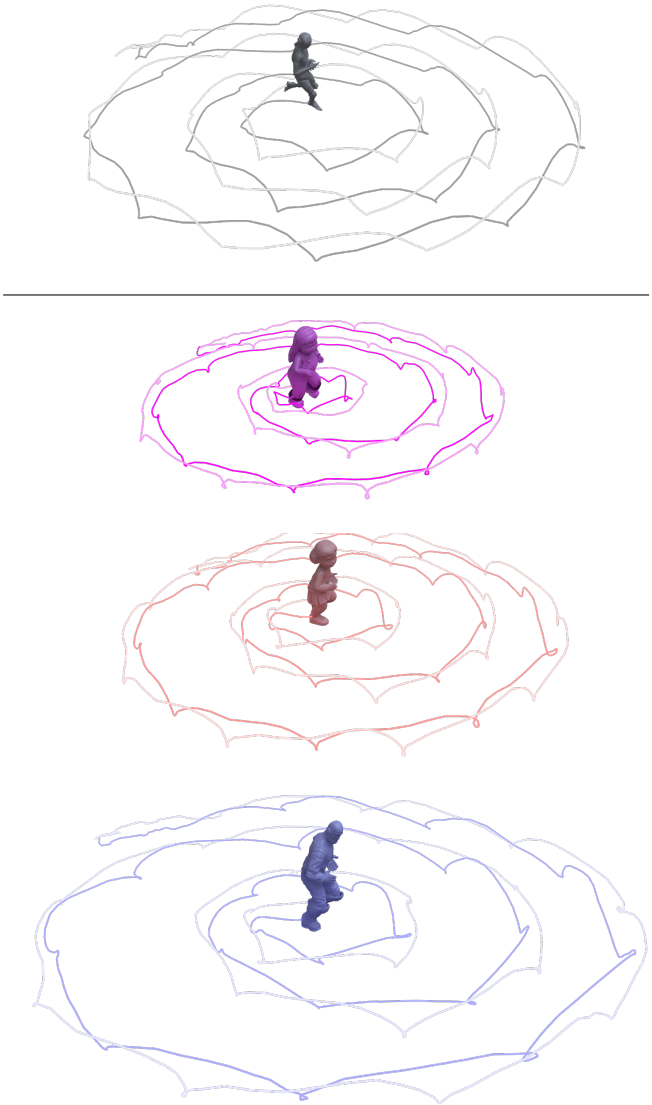


Fig. 6. Generation for multiple characters using a circular run base motion (on top) by applying different S_{id}. ©Disney/Pixar.

To mitigate this, when blending neighboring sequences, we first invert the alignment described in Section 4.1 to recover world coordinates. As after blending the original alignment statistics are invalid, we update the alignment for sequence i with respect to sequence $i-1$ at every step. In other words, we enforce that the difference between the first frame in the blended window is in the same position after reverting the alignment.

5 Results

In this section, we showcase results with multi-character generation and user control via γ and κ parameters.

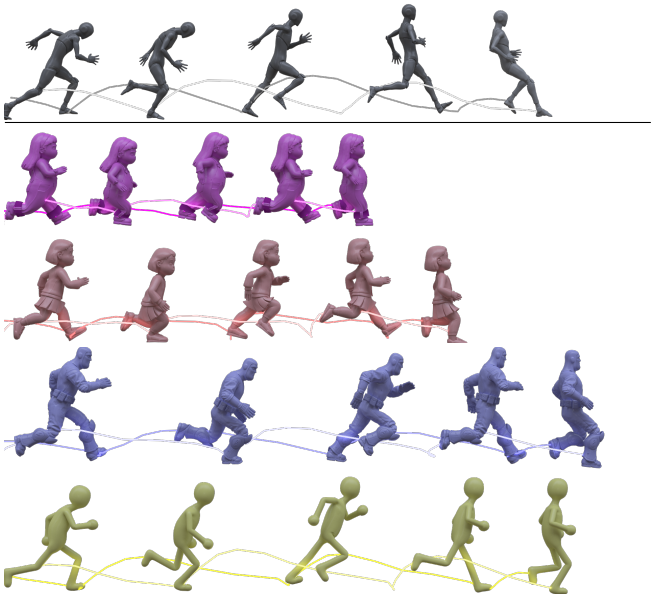


Fig. 7. Generation for multiple characters using a forward run motion (on top) by applying different S_{id}. ©Disney/Pixar.

5.1 Multi-Character Generation

We demonstrate multi character generation in Fig. 6 and Fig. 7 where the top row shows input motion capture and the remaining rows are the generated motion for characters A.2, A.1, B.1, and C, respectively. In Fig. 6, note that the generated motion is adjusted to the scale of the character, without introducing foot sliding, so that shorter characters, such as A.2, cover a smaller radius while stylizing the same input motion. Furthermore, in Fig. 7, the style of each generation is distinct where characters A.1 and A.2 appear more childlike and run while standing upright but the adult characters B.1 and C lean forward. Furthermore, as the crowds dataset has the most neutral style among the different characters, it resembles the motion capture the closest.

In addition to stylizing neutral motion capture, we can also use one character as the source motion and transfer to another, as in Fig. 8, where we transfer motion from character B.1 to A.1. We include the full resulting motion and other examples of motion transfer in our supporting video.



Fig. 8. Transferring motion from character B.1 (in grey) to character A.1 (red). ©Disney/Pixar.



Fig. 9. How different κ allow for original mocap frequencies to blend in. Following the sequence above, the only parameter that is able to maintain the original frequency is $\kappa = 0.8$, which identifies the mocap's smashing action as part of the 80% energy buildup. ©Disney/Pixar.

5.2 User Control

We propose two different control variables: γ and κ. The first can be used to exaggerate the motion even further into the stylized distribution. As seen in Fig. 10, increasing the γ shifts the output sequence in space and time. The product of this is a sequence with a more intensified ease-in and follow-through factors, as well compresses or expands extreme poses in the cycle even further.

Secondly, κ controls which frequencies of the content motion are transferred to the stylized generation. We empirically find that $\kappa = 0.6$ is sufficient to transfer the primary motifs of most content motions to the generator. However, to stylize arbitrary motions, we allow the user to tune κ to generate the best results. For example, in Fig. 9, we stylize a motion capture sequence in which an actor smashes an object with successive fast strikes. At smaller κs, we see that while the generated motions in green and teal match the overall poses from the motion capture, the smashing frequencies are not transferred. By increasing κ to 0.8, sufficient mocap energy is transferred such that the generated motion performs the smash.

Note that increasing κ increases the number of frequencies inpainted from the base motion, leaving fewer bands that are stylized. However, γ complements this reduced capacity by exaggerating the remaining frequencies for a final highly stylized output.

6 Validation

Next, we quantitatively evaluate our method against several baselines and qualitatively in a user study.

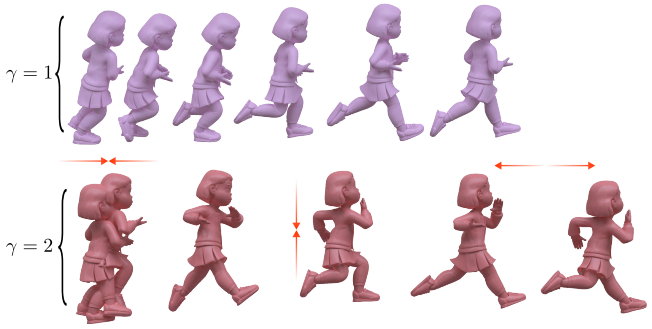


Fig. 10. Setting γ to 1.0 (top) and 2.0 (bottom) for a running clip ($\kappa = 0.4$). ©Disney/Pixar.

6.1 Baselines & Metrics

As we adapt a pre-trained generator, we compare our architecture to LoRA-MDM [Raab et al. 2024] and SMooDi [Zhong et al. 2024], which similarly adapt a pre-trained generator for motion stylization. Note that for uniformity all three approaches use the same base MDM network but differ in the type of adapter used. SMooDi also uses a ControlNet, but as our data is too small to train a robust style classifier, we replace their style encoder with a one-hot character identity encoding. Lastly, as motion matching [Clavet and Büttner 2015] is predominantly used for building locomotion from a small animation set, we qualitatively compare to it in our user study.

We evaluate the baselines on three aspects: bone proportion adjustment, base motion alignment, and stylization quality. As a motion sequence poorly adjusted to new bone proportions introduces large foot sliding (FS), we use it to measure fidelity to character-specific proportions. Our foot sliding metric is defined by the sum of the foot joint velocities in the XY-plane when they are closer than 0.1 meters to the ground. For base motion alignment between the motion capture and stylized generation, we compute, RV-MSE, a mean squared error on the global velocities of the two root joints to measure the discrepancy in the overall path of the character.

Lastly, we measure the stylization quality through the kernel inception distance (KID) [Bińkowski et al. 2018] between generated and ground truth motions using a pretrained feature extractor. KID computes the squared maximum mean discrepancy between the two feature distributions using a polynomial kernel, and compared to the more commonly used FID [Heusel et al. 2017], is more reliable for small sample sets such as ours. More information on our feature extractor can be found in Appendix C.

6.2 Quantitative Analysis

In Table 2, we compare our method to other baselines when puppeteering the generation using our inpainting scheme described in Section 4.3. Note that these baselines have not been shown to generate puppeteered motion, which is only made possible using our method. We perform all the experiments for the same inpainting parameters.

We first compare to finetuning the base motion generator. We see that while Fine-tuned MDM achieves the best FS and RV-MSE, it does not stylize the motion sufficiently, as demonstrated by a high KID. This suggests that finetuning the pretrained generator did not learn to adapt the input motion beyond fixing the bone proportions. In contrast, our method reports similar RV-MSE but has significantly

Table 2. Quantitative comparisons between different architectural baselines for stylizing motion capture to **Character A.1** style and proportions. ***All models use our inference guidance pipeline.**

Model	KID ↓	FS ↓	RV-MSE (1e−2) ↓
SMooDi*	**0.054**	1.199	1.257
LoRA-MDM*	0.077	1.246	1.254
Fine-tuned MDM*	0.084	1.388	1.261
Ours ($\gamma = 1.0$)	0.072	1.164	**1.253**
Ours ($\gamma = 1.25$)	0.064	1.166	1.256
Ours ($\gamma = 1.5$)	0.058	**1.099**	1.258

lower KID, indicating that we can follow the input mocap while stylizing it.

Comparing to LoRA-MDM, we see that it perform worse at generating motion in the style of Character A.1, as shown by the higher KID. SMooDi in particular registers the best KID, but the additional modeling capacity causes the ControlNet to overfit to the small training data. We note this overfitting in our supporting video, where SMooDi fights the condition window and tries to fit a straight running sequence in a circular run clip, taking very sudden and discrete turns. That represents how the full-layered ControlNet holds too strong an influence on the generation process and can bend it as it sees fit. In contrast, LoRA-MDM has comparable KID to our model at $\gamma = 1.0$ with only slightly worse FS. However, LoRA-MDM is trained for a single specific character and our model generalizes across different ones.

6.3 User Study

As style is ambiguous to quantitatively evaluate, we measure preference via a user study. We tasked 32 users with indicating their preferred motion between two randomized puppeteering results for 50 different samples. The users were asked to judge the motions in aesthetics, level of stylization, and conformity to the corresponding motion capture data, which was also visualized.

As demonstrated by Fig. 11, our method was heavily favored over all three baselines while the two γ levels were equally preferred. We include more details about the user study and analysis of comparisons between baselines in Appendix D.

6.4 Representation Space Ablation

As we described in Section 4.1, we generate motion in parent-local space compared to other methods that opt for global [Hwang et al. 2025; Vögeli et al. 2025] or root-centric [Tevet et al. 2023] space. As previous methods have shown that generating in global space generates higher fidelity motion compared to root-centric space, we trained an ablation of our ControlNet model generating global position and orientations. We find that for both $\gamma = 1.0$ and $\gamma = 1.5$, our unconditional generation reports a lower KID compared to this global-space variant, with 0.019 and 0.014 for ours, while 0.024 and 0.026 for global variant, respectively. Furthermore, since adapting bone proportions in global space equates to scaling joint positions, exaggerating γ also exaggerates this scaling, causing the generated motion to have inconsistent bone proportions.

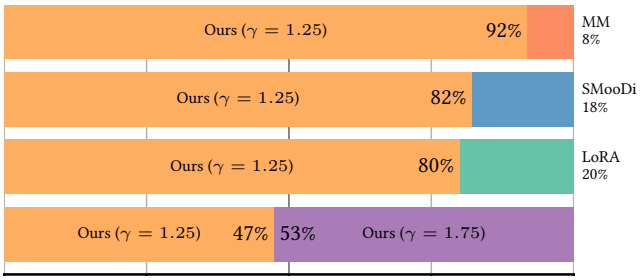


Fig. 11. User preference between our method and baselines for puppeteering

7 Discussion & Future Work

In this section, we discuss intuition behind key design choices and avenues for future work.

We hypothesized that the style of a character is primarily encoded in the relative rotations of each joint to their parent. Furthermore, parent-space representations disentangle a joint's intrinsic motion from the accumulated motion of its ancestors. Thus, generating motion in parent-space eases the task of adapting the motion capture to the different character proportions and styles, particularly for smaller data volumes such as ours.

Heterogeneous datasets like ours pose new challenges that remain underexplored. First, fragmentation and noise create ambiguity between stylistic intent and artifacts in a motion sequence. We train on off-camera data with a lower loss weight to maximize the information seen during training, but this causes our final model to exhibits some artifacts such as foot sliding. Secondly, jointly training on characters with vastly different character proportions (see Fig. 7) causes the model to scale motion accordingly, creating ambiguity when stylizing a mocap sequence as to whether the output should preserve the mocap's scale or the character's. While we chose the latter to maintain in-character velocities, the former might also be desired in certain settings. Therefore, improving motion quality from heterogeneous datasets remains an open direction.

Although our method generalizes to arbitrary actions, it is more suited to locomotion, as capturing stylistic nuances in more static actions is substantially harder than in dynamic sequences like running. Non-locomotion sequences are very sensitive to misplaced feet and foot-planting, which makes some of our foot-sliding artifacts more perceivable. Stylizing non-locomotion movements therefore requires manually tuning γ and κ. However, as feature films are dominated by non-locomotion actions, automatically stylizing arbitrary motion remains important future work. Identifying better contacts, introducing ground-level constraints and applying inverse kinematics post-processing steps are directions to further improve our generated animations.

8 Conclusion

In this work, we built the first generative motion style transfer method that aligns with a production definition of style. Given a small and challenging hand-animated dataset—with many hidden joints, large style and scale diversity, and broken motion sequences—we successfully learn a motion generator by leveraging a larger mocap prior. We show that our method can transfer motion from

a mocap sequence or from another character to a specified character, simultaneously adjusting to the new bone proportions. Our spectral-domain inpainting scheme leverages DCT decomposition of motion to expose intuitive user controls over the level of stylistic exaggeration and the amount of content transferred from a source motion. By aligning style transfer to production data, we build a system that serves as a foundation for production-ready motion authoring applications.

References

Kfir Aberman, Yijia Weng, Dani Lischinski, Daniel Cohen-Or, and Baoquan Chen. 2020. Unpaired motion style transfer from video to animation. *ACM Transactions on Graphics (TOG)* 39, 4 (2020), 64–1.

Dhruv Agrawal, Jakob Buhmann, Dominik Borer, Robert W. Sumner, and Martin Guay. 2024. SKEL-Betweener: A Neural Motion Rig for Interactive Motion Authoring. *ACM Transactions on Graphics* 43, 6 (Dec. 2024), 1–11. doi:10.1145/3687941

Anonymous. 2025. Absolute Coordinates Make Motion Generation Easy. https://openreview.net/forum?id=TaURJrZcjJ

Shariq Farooq Bhat, Niloy Mitra, and Peter Wonka. 2024. LOOSECONTROL: Lifting ControlNet for Generalized Depth Conditioning. In *Special Interest Group on Computer Graphics and Interactive Techniques Conference Conference Papers*. ACM, Denver CO USA, 1–11. doi:10.1145/3641519.3657525

Mikołaj Bińkowski, Dougal J. Sutherland, Michael Arbel, and Arthur Gretton. 2018. Demystifying MMD GANs. In *International Conference on Learning Representations*. https://openreview.net/forum?id=r1lUOzWCW

Christoph Bregler, Lorie Loeb, Erika Chuang, and Hrishi Deshpande. 2002. Turning to the masters: motion capturing cartoons. *ACM Trans. Graph.* 21, 3 (July 2002), 399–407. doi:10.1145/566654.566595

Prashanth Chandran, Agon Serifi, Markus Gross, and Moritz Bächer. 2024. Spline-Based Transformers. In *Computer Vision – ECCV 2024: 18th European Conference, Milan, Italy, September 29–October 4, 2024, Proceedings, Part LXXXVI* (Milan, Italy). Springer-Verlag, Berlin, Heidelberg, 1–17. doi:10.1007/978-3-031-73016-0_1

Stephen Chenney, Mark Pingel, Rob Iverson, and Marcin Szymanski. 2002. Simulating cartoon style animation. In *Proceedings of the 2nd international symposium on Non-photorealistic animation and rendering*. ACM, Annecy France, 133–138. doi:10.1145/508530.508553

Simon Clavet and Michael Büttner. 2015. *Motion Matching - The Road to Next Gen Animation.*

Setareh Cohan, Guy Tevet, Daniele Reda, Xue Bin Peng, and Michiel van de Panne. 2024. Flexible Motion In-betweening with Diffusion Models. In *ACM SIGGRAPH 2024 Conference Papers* (Denver, CO, USA) *(SIGGRAPH '24)*. Association for Computing Machinery, New York, NY, USA, Article 69, 9 pages. doi:10.1145/3641519.3657414

Minyue Dai, Jingbo Wang, Ke Fan, Bin Ji, Haoyu Zhao, Junting Dong, and Bo Dai. 2025. Towards Synthesized and Editable Motion In-Betweening Through Part-Wise Phase Representation. *arXiv preprint arXiv:2503.08180* (2025).

Prafulla Dhariwal and Alexander Nichol. 2021. Diffusion models beat gans on image synthesis. *Advances in neural information processing systems* 34 (2021), 8780–8794.

Chuan Guo, Yuxuan Mu, Xinxin Zuo, Peng Dai, Youliang Yan, Juwei Lu, and Li Cheng. 2024. Generative Human Motion Stylization in Latent Space. *CoRR* (2024).

Chuan Guo, Shihao Zou, Xinxin Zuo, Sen Wang, Wei Ji, Xingyu Li, and Li Cheng. 2022. Generating Diverse and Natural 3D Human Motions From Text. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 5152–5161.

Tong Guo, Zhiqian Xia, Feng Yu, Haifeng Xia, and Siyu Xia. 2025b. A Generalized Two-stage Approach to Motion Style Transfer. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 6567–6576.

Ziyu Guo, Yizhak Ben-Shabat, Young Yoon Lee, Joseph Liu, Victor Zordan, and Mubbasir Kapadia. 2025a. StyleMotif: Multi-Modal Motion Stylization using Style-Content Cross Fusion. In *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*. 13349–13359. doi:10.1109/ICCV51701.2025.01240

Raia Hadsell, Sumit Chopra, and Yann LeCun. 2006. Dimensionality reduction by learning an invariant mapping. In *2006 IEEE computer society conference on computer vision and pattern recognition (CVPR'06)*, Vol. 2. IEEE, 1735–1742.

Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. 2017. GANs trained by a two time-scale update rule converge to a local nash equilibrium. In *Proceedings of the 31st International Conference on Neural Information Processing Systems* (Red Hook, NY, USA, 2017-12-04) *(NIPS'17)*. Curran Associates Inc., 6629–6640. https://dl.acm.org/doi/10.5555/3295222.3295408

Jonathan Ho and Tim Salimans. 2021. Classifier-Free Diffusion Guidance. In *NeurIPS 2021 Workshop on Deep Generative Models and Downstream Applications*. https://openreview.net/forum?id=qw8AKxfYbI

Edward J Hu, yelong shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. LoRA: Low-Rank Adaptation of Large Language Models. In *International Conference on Learning Representations*. https://openreview.net/forum?id=nZeVKeeFYf9

Xun Huang and Serge Belongie. 2017. Arbitrary style transfer in real-time with adaptive instance normalization. In *Proceedings of the IEEE international conference on computer vision*. 1501–1510.

Yue Huang, Haoran Mo, Xiao Liang, and Chengying Gao. 2022. Unpaired motion style transfer with motion-oriented projection flow network. In *2022 IEEE International Conference on Multimedia and Expo (ICME)*. IEEE, 1–6.

Inwoo Hwang, Jinseok Bae, Donggeun Lim, and Young Min Kim. 2025. Motion Synthesis with Sparse and Flexible Keyjoint Control. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. 13203–13213.

Deok-Kyeong Jang, Soomin Park, and Sung-Hee Lee. 2022. Motion puzzle: Arbitrary motion style transfer by body part. *ACM Transactions on Graphics (TOG)* 41, 3 (2022), 1–16.

John Lasseter. 1998. Principles of Traditional Animation Applied to 3D Computer Animation. In *Seminal Graphics*. ACM, New York, NY, USA, 263–272. doi:10.1145/280811.281005

Doyup Lee, Chiheon Kim, Saehoon Kim, Minsu Cho, and Wook-Shin Han. 2022. Autoregressive image generation using residual quantization. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 11523–11532.

Zhe Li, Yisheng He, Lei Zhong, Weichao Shen, Qi Zuo, Lingteng Qiu, Zilong Dong, Laurence Tianruo Yang, and Weihao Yuan. 2024. Mulsmo: Multimodal stylized motion generation by bidirectional control flow. *arXiv preprint arXiv:2412.09901* (2024).

Ian Mason, Sebastian Starke, and Taku Komura. 2022. Real-time style modelling of human locomotion via feature-wise transformations and local motion phases. *Proceedings of the ACM on Computer Graphics and Interactive Techniques* 5, 1 (2022), 1–18.

Soomin Park, Deok-Kyeong Jang, and Sung-Hee Lee. 2021. Diverse motion stylization for multiple style domains via spatial-temporal graph-based generative model. *Proceedings of the ACM on Computer Graphics and Interactive Techniques* 4, 3 (2021), 1–17.

Sigal Raab, Inbar Gat, Nathan Sala, Guy Tevet, Rotem Shalev-Arkushin, Ohad Fried, Amit Haim Bermano, and Daniel Cohen-Or. 2024. Monkey See, Monkey Do: Harnessing Self-attention in Motion Diffusion for Zero-shot Motion Transfer. In *SIGGRAPH Asia 2024 Conference Papers* (Tokyo, Japan) *(SA '24)*. Association for Computing Machinery, New York, NY, USA, Article 87, 13 pages. doi:10.1145/3680528.3687579

Davis Rempe, Mathis Petrovich, Ye Yuan, Haotian Zhang, Xue Bin Peng, Yifeng Jiang, Tingwu Wang, Umar Iqbal, David Minor, Michael de Ruyter, Jiefeng Li, Chen Tessler, Edy Lim, Eugene Jeong, Sam Wu, Ehsan Hassani, Michael Huang, Jin-Bey Yu, Chaeyeon Chung, Lina Song, Olivier Dionne, Jan Kautz, Simon Yuen, and Sanja Fidler. 2026. Kimodo: Scaling Controllable Human Motion Generation. *arXiv:2603.15546* (2026).

Haim Sawdayee, Chuan Guo, Guy Tevet, Bing Zhou, Jian Wang, and Amit H. Bermano. 2025. Dance Like a Chicken: Low-Rank Stylization for Human Motion Diffusion. *Computer Graphics Forum* 44, 8 (2025), e70365. doi:10.1111/cgf.70365

Yoni Shafir, Guy Tevet, Roy Kapon, and Amit Haim Bermano. 2024. Human Motion Diffusion as a Generative Prior. In *The Twelfth International Conference on Learning Representations*.

Sebastian Starke, Ian Mason, and Taku Komura. 2022. DeepPhase: Periodic Autoencoders for Learning Motion Phase Manifolds. *ACM Trans. Graph.* 41, 4 (July 2022), 136:1–136:13. doi:10.1145/3528223.3530178

Justin Studer, Dhruv Agrawal, Dominik Borer, Seyedmorteza Sadat, Robert W. Sumner, Martin Guay, and Jakob Buhmann. 2024. Factorized Motion Diffusion for Precise and Character-Agnostic Motion Inbetweening. In *The 17th ACM SIGGRAPH Conference on Motion, Interaction, and Games*. ACM, Arlington VA USA, 1–10. doi:10.1145/3677388.3696338

Xiangjun Tang, Linjun Wu, He Wang, Bo Hu, Xu Gong, Yuchen Liao, Songnan Li, Qilong Kou, and Xiaogang Jin. 2023. Rsmt: Real-time stylized motion transition for characters. In *ACM SIGGRAPH 2023 Conference Proceedings*. 1–10.

Xiangjun Tang, Linjun Wu, He Wang, Yiqian Wu, Bo Hu, Songnan Li, Xu Gong, Yuchen Liao, Qilong Kou, and Xiaogang Jin. 2024. Decoupling Contact for Fine-Grained Motion Style Transfer. In *SIGGRAPH Asia 2024 Conference Papers*. 1–11.

Guy Tevet, Sigal Raab, Brian Gordon, Yoni Shafir, Daniel Cohen-Or, and Amit Haim Bermano. 2023. Human Motion Diffusion Model. In *The Eleventh International Conference on Learning Representations*. https://openreview.net/forum?id=SJ1kSyO2jwu

Frank Thomas, 1912-2004. [1995]. *The Illusion of Life : Disney Animation.* 1st Hyperion ed. New York : Hyperion, [1995].

Jonathan Tseng, Rodrigo Castellon, and C. Karen Liu. 2023. EDGE: Editable Dance Generation From Music. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, Vancouver, BC, Canada, 448–458. doi:10.1109/CVPR52729.2023.00051

Luca Vögeli, Dhruv Agrawal, Martin Guay, Dominik Borer, Robert W. Sumner, and Jakob Buhmann. 2025. Implicit Bézier Motion Model for Precise Spatial and Temporal Control. In *Proceedings of the 2025 18th ACM SIGGRAPH Conference on Motion, Interaction, and Games*. ACM, Zurich Switzerland, 1–10. doi:10.1145/3769047.3769052

Jue Wang, Steven M. Drucker, Maneesh Agrawala, and Michael F. Cohen. 2006. The cartoon animation filter. *ACM Transactions on Graphics* 25, 3 (July 2006), 1169–1173. doi:10.1145/1141911.1142010

Linjun Wu, Xiangjun Tang, Jingyuan Cong, He Wang, Bo Hu, Xu Gong, Songnan Li, Yuchen Liao, Yiqian Wu, Chen Liu, and Xiaogang Jin. 2025. Semantically Consistent Text-to-Motion with Unsupervised Styles. In *Proceedings of the Special Interest Group on Computer Graphics and Interactive Techniques Conference Conference Papers*. ACM, Vancouver BC Canada, 1–10. doi:10.1145/3721238.3730641

Shihong Xia, Congyi Wang, Jinxiang Chai, and Jessica Hodgins. 2015. Realtime style transfer for unlabeled heterogeneous human motion. *ACM Transactions on Graphics (TOG)* 34, 4 (2015), 1–10.

Zirui Yang, Zhongwen Luo, Zhengrui Wang, and Yuanyuan Liu. 2025. Arbitrary Motion Style Transfer via Contrastive Learning. *Applied Sciences* 15, 4 (2025), 1817.

Fatemeh Zargarbashi, Dhruv Agrawal, Jakob Buhmann, Martin Guay, Stelian Coros, and Robert W. Sumner. [n. d.]. VQ-Style: Disentangling Style and Content in Motion with Residual Quantized Representations. *Computer Graphics Forum* n/a, n/a ([n. d.]), e70377. arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1111/cgf.70377 doi:10.1111/cgf.70377

Lvmin Zhang, Anyi Rao, and Maneesh Agrawala. 2023. Adding Conditional Control to Text-to-Image Diffusion Models. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*. 3813–3824. doi:10.1109/ICCV51070.2023.00355

Lei Zhong, Yiming Xie, Varun Jampani, Deqing Sun, and Huaizu Jiang. 2024. SMooDi: Stylized Motion Diffusion Model. In *Computer Vision – ECCV 2024: 18th European Conference, Milan, Italy, September 29–October 4, 2024, Proceedings, Part I* (Milan, Italy). Springer-Verlag, Berlin, Heidelberg, 405–421. doi:10.1007/978-3-031-73232-4_23

Lei Zhong, Yi Yang, and Changjian Li. 2025. SMooGPT: Stylized Motion Generation Using Large Language Models. arXiv:2509.04058 [cs] doi:10.48550/arXiv.2509.04058

Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. 2017. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE international conference on computer vision*. 2223–2232.

A Dataset Analysis

We first provide more details on the amount and nature of our data stylized data. Table 3 shows the per character number of frames, the amount of frames filtered in and the total visibility percentages. Notice that while the hero characters A.1 and B.1 have ~25 and ~36 minutes of data respectively, other characters such as A.4 and A.6 have less than 10 minutes of data after filtering. In Fig. 12, we also plot per-joint visibility percentages for each character. We see that across all characters, the lower body is hidden significantly often than the upper body. Consequently, the feet are least likely to be animated and thus contain artifacts such as foot sliding, that our trained models inherits. We use two simple filters to define whether or not a window of 30 frames is valid:

Extreme root velocity. We eliminate windows that contains a maximum root velocity over **0.5 meters** per frame. This removes any artifacts caused by teleportation and keep the window displacement in a reasonable magnitude for the model to learn from.

Static root-space velocity. We use root-space velocities to define whether a window is static and does not contribute to learning character locomotion or actions. Going over root-space keeps us from being deceived by constant root movement that could be performed by vehicular motion of very simple camera-panned sequences. We define a window as static if there are less than **3 joints** with an average velocity of **0.3 centimeters** per frame.

In Fig. 13, we plot the pairwise KID for the different characters in our stylized dataset. Characters A.1 (blue) and A.2 (green) belong to movie, B.1 to movie B, C (red) is the *crowds* dataset. Lastly, we also measure the average style of each movie in plots A (orange) and B (brown). As we pairwise compare ground samples, each plot achieves a 0 KID for their respective character. We see that the characters from movie A (A.1, A.2, A) all lean towards the right of the plot while the movie B characters (B.1 and B) lean towards the left, showing that the two movies have an overall style distinct from each other. The crowds plot, C, is more centered, suggesting a more neutral style.

Furthermore, we see that the average movie plots, A and B, are more similar to each other and crowds dataset, than to any other character from the opposite movie. Demonstrating that combining styles of different characters, even within the movie, results in a reduced level of style.

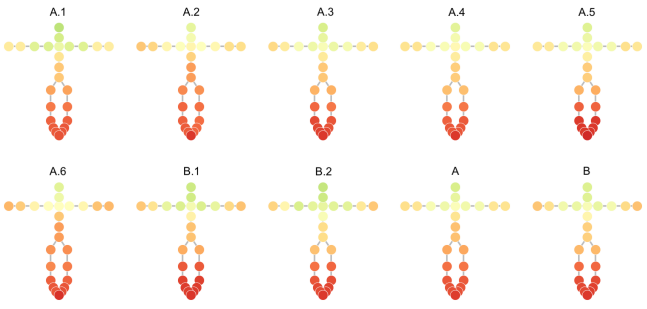


Fig. 12. Per-joint visibility percentage across different characters.

Table 3. Number of filtered in frames and visibility percentage by character.

Name	Total Frames	Filtered In			Visible %
		Frames	Time (Min.)	%	
A.1	67,912	35,630	24.74	52	53.11
A.2	33,595	15,492	10.76	46	50.97
A.3	39,637	18,293	12.70	46	53.59
A.4	31,710	13,079	9.08	41	47.96
A.5	41,515	21,972	15.26	53	52.35
A.6	21,006	8,820	6.12	42	43.49
B.1	100,463	51,458	35.73	51	39.88
B.2	50,445	23,501	16.32	47	39.02
C	357,591	273,571	189.98	76	—
Total	**743,874**	**461,816**	**320.69**	**62**	**46.54**

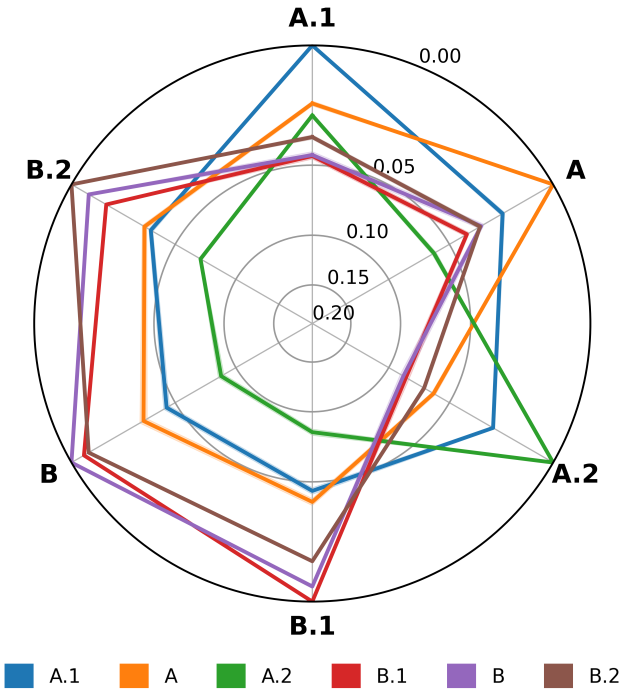


Fig. 13. KID of different characters in the stylized dataset.

B Implementation Details

Our pipeline predicts sequences of 30 frames. The dataset shows numerous short clips that would have to be discarded if we were to predict longer sequences. On top of that, since we filter sequences that contain faulty frames, it is better to have a smaller context window. That is so we do not sacrifice meaningful sequences because one frame is not up to standards.

Our Transformer network is heavily inspired by MDM [Tevet et al. 2023], with 8 layers, 4 attention heads, a latent dimension of 1024, feed-forward layers of dimension 2048 and a 0.1 dropout rate.

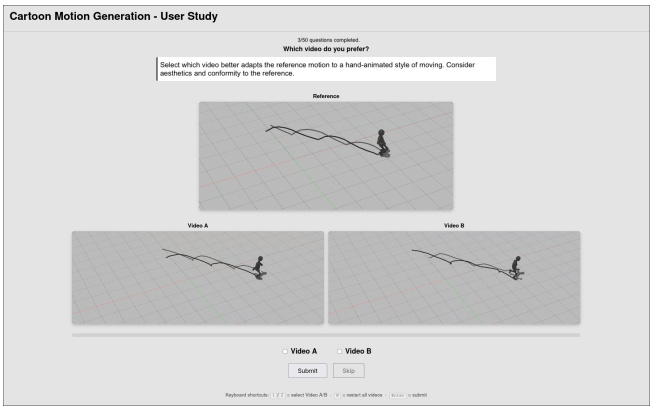


Fig. 14. UI used for user study.

Our base model was trained for a total of 5 days while our adapters were trained for 2 days. We used the NVIDIA RTX 4090 and RTX 5090 series GPUs to train these models.

C KID Feature Extractor

In order to compare results quantitatively, we used the Kernel Inception Difference (KID) metric [Bińkowski et al. 2018]. Statistical methods like KID or FID [Heusel et al. 2017] work over a compressed meaningful latent space to determine the distance between two sampled manifolds. Hence, we were tasked with building such latent space. We opted to train a VAE, more specifically, a Spline-based VAE [Chandran et al. 2024], which has been shown to work nicely to compress temporal domains. We trained on all our characters applying the same filters we applied for the Diffusion Transformer. As expected from the scarcity of our data, the Variational Auto-encoder did not generalize too well, but we were more interested in the latent space it gave us rather than the reconstruction error for the validation set. In order to build comparisons between the generated samples and the characters they were intended to represent, we first compress the ground-truth character samples in the latent space, compress the generated samples in the latent space, and, finally, apply the KID method to compute the distance.

D User Study Details

An example screenshot from the user study is shown in Fig. 14. The participants were shown three videos, a reference mocap motion at the top and puppeteering using two randomized methods, and asked to indicate their preference based on aesthetics and conformity to base motion. All videos were rendered using the same template mesh to avoid biasing style perception. In the event the participant could not differentiate between the two motions, they could skip to another example. However, skipping did not count towards the number of questions completed, hence each participant indicated their preference for 50 different combinations.

While we discussed the results of comparing our method against baselines in the main text, as the two methods shown was completely randomized, we also collected results between the baselines. The expanded results are presented in Fig. 15. First, we notice that

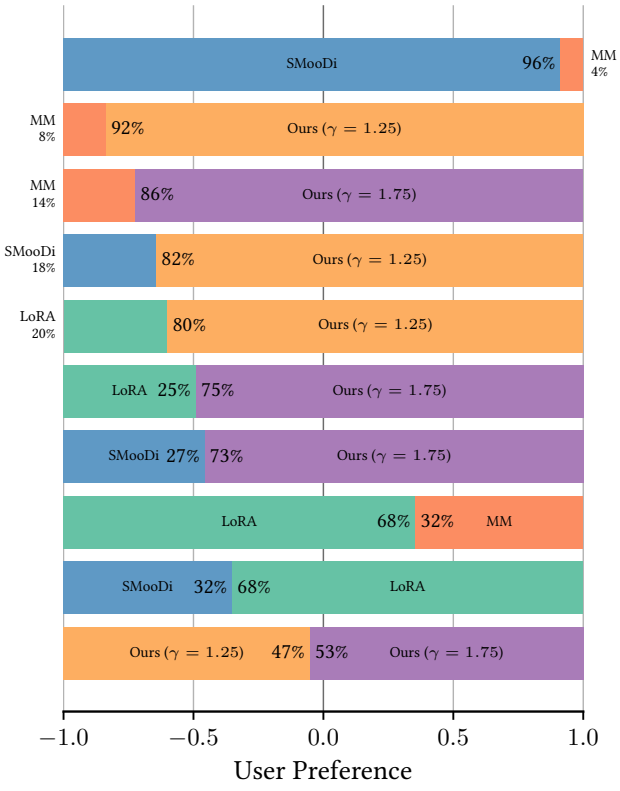


Fig. 15. Pair wise comparisons between all baselines on user preference.

both variants of our method ($\gamma = 1.25$ and $\gamma = 1.75$) significantly outperformed all other baselines. While both variants were equally preferred in their head to head examples, $\gamma = 1.25$ was slightly stronger preferred at comparisons to other samples. We expect this to be because increasing γ can introduce artifacts that, while looking more stylized, can hurt the overall perception. Between LoRA-MDM [Anonymous 2025] and SMooDi [Zhong et al. 2024], we see that LoRA-MDM was preferred. As LoRA-MDM was trained only on character A.1 data, which was also used to generate the samples for user study, we expect that it better adapts to one character's particular style better than SMooDi.

E Additional Evaluations

In this section, we include additional evaluation of our three user controls, S_{id}, γ, and κ.

E.1 Multi-character generation

In Fig. 16, we plot KID for unconditional generation with four different characters where A.1 and A.2 belong to movie A while B.1 and B.2 to movie B. We also plot the KID to the average style of each film represented by all its characters together. We see that A.1 (blue) and A.2 (orange) both peak at their respective characters and have lower KID to movie A compared to movie B. As noted in Table 1,

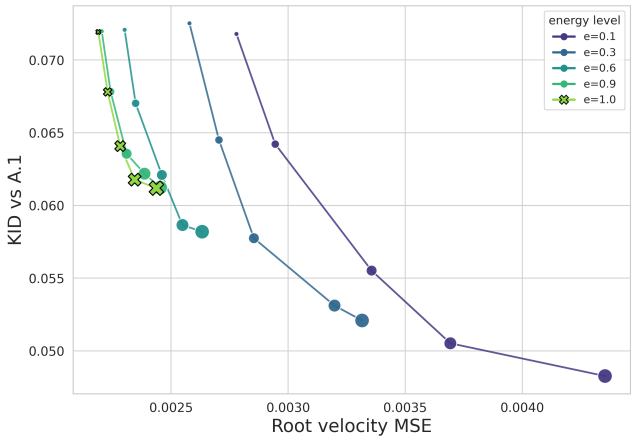


Fig. 17. Comparing KID and RV-MSE for inpainting in spectral domain vs Cartesian domain for different residual strengths $1.0 \leq \gamma \leq 2.0$ and energy threshold $0.9 \geq \kappa \geq 0.5$.

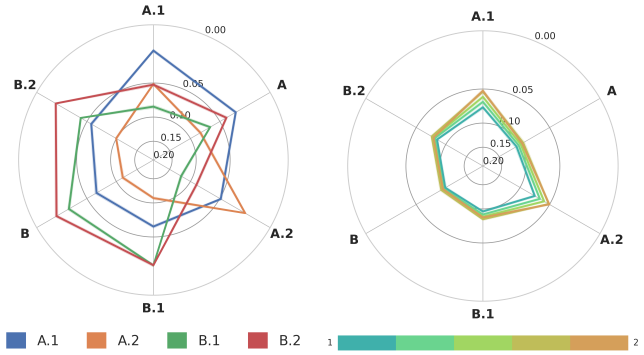


Fig. 16. (a) KID for unconditionally generating motion for different input character identities (S_{id}). (b) KID for conditionally generating motion for **Character A.1** with $1 < \gamma \leq 2$.

character A.1 has about ~24 minutes of data compared to ~10 minutes of character A.2. Therefore, the average style of movie A is heavily towards A.1 resulting in the generated character A.1 motion having lower movie A KID than character A.2 motion. Similarly, generated motion for characters B.1 and B.2 results in strongest peaks at each respective character and the second strongest peak at movie B. Combining the results across all the different characters with the qualitative results in our supporting video, this shows that our model generates motion in the specified character style.

E.2 Spectral Inpainting Ablation

We compare inpainting in the spectral domain scheme to in the Cartesian domain for varying energy level thresholds κ and residual strength γ in Fig. 17. We use identical inpainting cutoffs for both settings at $t = 300$, and measure the KID for character A.1 and RV-MSE. Firstly, we see that increasing γ improves KID at the cost of RV-MSE for both inpainting domains. This is expected as preserving the base motion and stylizing are inherently adversarial tasks.

However, our method enables additional fine-grained user control over this tradeoff by allowing the user to set the energy level κ threshold for any particular residual strength γ.This allows one to individually control the level of stylization for each animation sequence. Hence, for clips closer to training distribution, a lower energy threshold would more strongly stylize the animation without degrading the content of the clip. Conversely, for out of distribution animations, a high energy threshold ensures that the content is still preserved but at the cost of lower level of stylization.