IMAGE DIFFUSION MODELS ARE GOOD 3D HUMAN MOTION GENERATORS

Anonymous authors
Paper under double-blind review

ABSTRACT

Text-to-motion generation typically relies on dataset-specific skeletal representations, with models trained separately for benchmarks with different joint layouts. We investigate whether image diffusion architectures can model 3D human motion through a common image-compatible representation that accommodates heterogeneous skeletal layouts while preserving motion-specific kinematic structure. To this end, we organize root-centered joint trajectories into topology-informed joint–time images and model global root dynamics through a complementary latent stream. A motion autoencoder conditions local-motion decoding on the root latent, while a text-conditioned Scalable Interpolant Transformer (SiT) jointly generates both latent streams, with supervision on decoded 3D joint positions further constraining generation in motion space. In addition to dataset-specific models, we train a single motion autoencoder on HumanML3D and KIT-ML, followed by a single text-conditioned SiT shared across both datasets, preserving their respective skeletal definitions without skeleton retargeting to a common joint layout. Experiments on HumanML3D and KIT-ML show that our dataset-specific models achieve state-of-the-art performance, while shared training yields further gains on both benchmarks, demonstrating the benefits of shared modeling across heterogeneous skeletal layouts.

1 INTRODUCTION

Text-to-motion generation aims to synthesize realistic 3D human motion from natural-language descriptions (Guo et al., 2022; Petrovich et al., 2022). Recent methods have made substantial progress in motion quality and text–motion alignment (Tevet et al., 2023; Zhang et al., 2023a; Guo et al., 2024), but commonly operate on representations tied to particular skeletal definitions. Widely used benchmarks such as HumanML3D (Guo et al., 2022) and KIT-ML (Plappert et al., 2016) use different skeletal definitions, including different joint counts and indexing conventions. In vector-based representations, these differences affect both the vector dimensionality and the correspondence between vector entries and skeletal joints, requiring explicit handling when sharing model inputs and outputs across datasets. Recent work has explored motion generation across heterogeneous skeletal structures under different conditioning settings, including skeleton-conditioned synthesis (Gat et al., 2025), style- and trajectory-controlled generation (Khani et al., 2025), and text- and skeleton-conditioned generation (Liu et al., 2025). Within this broader goal, we investigate whether an image-compatible representation can support shared motion autoencoding and text-conditioned generation while preserving each dataset's native skeletal definition.

Root-centered joint trajectories naturally form a joint–time array, with the three spatial coordinates serving as separate channels. Mapping this array to a common image resolution makes the spatial dimensions presented to the model independent of the native joint count, providing a basis for shared modeling across datasets (Fig. 1). This image-compatible form also allows motion to be processed by image autoencoding and latent generative architectures (Rombach et al., 2022; Peebles & Xie, 2023; Ma et al., 2024). However, a common image shape alone does not encode anatomical relationships or capture dependencies between global movement and local articulation. These considerations motivate a representation and modeling framework that explicitly incorporates both.

To address these challenges, we adopt the compact motion representation used in MARDM (Meng et al., 2025), which demonstrated that root motion and root-centered local joint positions can sup-

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

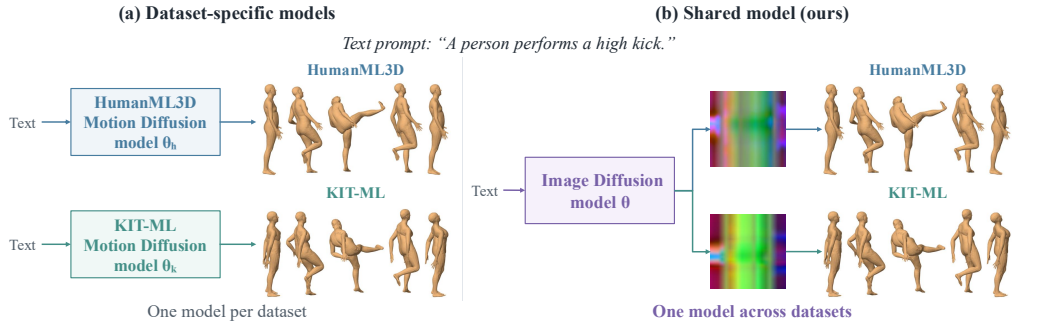


Figure 1: **Dataset-specific versus shared motion modeling.** (a) Separate models for HumanML3D and KIT-ML. (b) Our method uses a single model across both datasets despite their different skeletal definitions. The motion-to-image mapping is detailed in Figure 2.

port strong motion generation. We investigate how these two motion components can be organized for image-based generative modeling. Specifically, we reorder local joints according to body-part organization and kinematic topology and arrange their trajectories into joint–time images, giving the joint axis a meaningful anatomical organization. Global root dynamics do not share this joint-wise organization and are therefore modeled through a complementary latent stream. To capture dependencies between global movement and local articulation, our motion autoencoder encodes skeleton images and root motion into separate latent streams and conditions image decoding on the root latent. A text-conditioned Scalable Interpolant Transformer (SiT) then jointly generates both latent streams from noise, modeling their interaction during generation. To further constrain generation in motion space, we supervise decoded 3D joint positions.

Beyond dataset-specific training, we use the proposed representation to train a single motion autoencoder across HumanML3D and KIT-ML, followed by a single text-conditioned SiT shared across both datasets. A dataset condition specifies the target dataset during latent generation. Decoded motions are recovered and supervised according to each dataset's native skeletal definition, enabling parameter sharing without skeleton retargeting or joint padding.

Experiments on HumanML3D and KIT-ML show that our dataset-specific models achieve state-of-the-art performance, while shared training further improves all six reported metrics on both datasets. These results demonstrate that the proposed representation supports both strong dataset-specific generation and shared modeling across heterogeneous skeletal layouts.

Our main contributions are summarized as follows:

- We introduce an image-compatible motion representation that organizes root-centered local joint trajectories into topology-informed joint-time images while representing global root dynamics through a complementary stream.
- We develop a coupled motion generation framework combining root-conditioned image decoding, joint text-conditioned generation of both latent streams with SiT, and supervision on decoded 3D joint positions.
- We enable shared motion autoencoding and text-conditioned generation across HumanML3D and KIT-ML using a single motion autoencoder and a single SiT, while preserving each dataset's native skeletal definition without skeleton retargeting or joint padding.

2 Related Work

Text-to-Motion Generation. Diffusion-based methods generate text-conditioned motion directly in motion space, as in MDM (Tevet et al., 2023) and MotionDiffuse (Zhang et al., 2024), or in a learned latent space, as in MLD (Chen et al., 2023). MARDM (Meng et al., 2025) combines diffusion with masked autoregressive modeling using a compact motion representation. Another line of work represents motion using discrete tokens. T2M-GPT (Zhang et al., 2023a) uses autoregressive generation, MoMask (Guo et al., 2024) combines hierarchical motion quantization with masked prediction, and BAMM (Pinyoanuntapong et al., 2024a) integrates masked and autoregressive modeling for

flexible-length generation and editing. Building on discrete motion tokens, MotionLLM (Wu et al., 2025) uses a pretrained language model for motion generation, editing, and understanding, while IRG-MotionLLM (Li et al., 2026) interleaves generation with assessment and refinement. Complementary to these advances in generative modeling, we investigate an image-compatible motion representation that supports image generative architectures while incorporating skeletal organization and enabling shared modeling across heterogeneous skeletal layouts.

Motion Representations and Image-Compatible Generative Modeling. The standard text-to-motion representation (Guo et al., 2022) combines root motion, joint positions, rotations, velocities, and foot-contact information. Following MARDM (Meng et al., 2025), which demonstrates strong generation performance using only root motion and root-centered joint positions, we adopt this compact representation for image-based generative modeling. Image-based skeleton representations have also been explored for action recognition. Ke et al. (2017) transform skeleton trajectories into image-based clips encoding spatial relationships for convolutional networks, while SkeleMotion (Caetano et al., 2019) encodes joint-motion magnitudes and orientations as skeleton images. These approaches demonstrate that skeletal sequences can be organized into structured images for action recognition. Our work explores this general idea for text-conditioned generation: we encode root-centered 3D joint trajectories in topology-informed joint–time images, model global root dynamics through a complementary stream, and recover generated outputs as 3D motion sequences.

In image synthesis, latent diffusion models (Rombach et al., 2022) generate in an autoencoder's latent space, while DiT (Peebles & Xie, 2023) and SiT (Ma et al., 2024) develop Transformer architectures for diffusion and interpolant-based generation. We build on these advances to jointly generate image and root-motion latents, using root-conditioned decoding and decoded 3D joint supervision to connect latent generation to motion geometry.

Motion Modeling Across Skeletal Structures. Motion retargeting transfers motion from one skeleton to another. Aberman et al. (2020) introduce skeleton-aware convolution, pooling, and unpooling operators to embed motions from different skeletal structures into a shared latent space for cross-structural retargeting. More recent work directly generates motion across heterogeneous skeletal structures. AnyTop (Gat et al., 2025) incorporates skeletal topology and textual joint descriptions into a diffusion model for skeleton-conditioned synthesis. UniMoGen (Khani et al., 2025) dynamically processes character joints within a skeleton-agnostic diffusion model for style- and trajectory-controlled generation. OmniSkel (Liu et al., 2025) generates motion conditioned on text and a target skeleton, using a skeleton-aware motion autoencoder. Our image-compatible approach maps local joint trajectories to a common image resolution, enabling shared motion autoencoding and generation across datasets with different native joint counts. We investigate this formulation on HumanML3D and KIT-ML with a single motion autoencoder and a single text-conditioned SiT, using dataset conditioning and dataset-specific recovery to preserve each dataset's native skeletal definition without retargeting or joint padding.

3 METHOD

Our framework makes human motion compatible with image generative architectures while modeling dependencies between local articulation and global movement. We organize root-centered joint trajectories into topology-informed joint–time images (Fig. 2) and represent root dynamics through a complementary stream (Sec. 3.1). Our motion autoencoder encodes these components into separate latent streams and conditions image decoding on the root latent (Sec. 3.2). Building on this latent representation, a text-conditioned SiT jointly generates both streams from noise, modeling their interaction during generation (Sec. 3.3). We extend this framework to shared training across HumanML3D and KIT-ML, using a single motion autoencoder and a single SiT without skeleton retargeting (Sec. 3.4). During generator training, supervision on decoded 3D joint positions further constrains generation in motion space (Sec. 3.5). Figure 3 illustrates the overall framework.

3.1 IMAGE-COMPATIBLE MOTION REPRESENTATION

Following MARDM (Meng et al., 2025), we represent a motion sequence of length T from dataset d using root features $\mathbf{R}^{(d)} \in \mathbb{R}^{T\times 4}$ and root-centered local joint coordinates $\mathbf{P}^{(d)} \in \mathbb{R}^{T\times J_d\times 3}$. Here, J_d is the number of non-root joints: 21 for HumanML3D and 20 for KIT-ML. We arrange local joint trajectories into skeleton images and retain root features as a complementary stream.

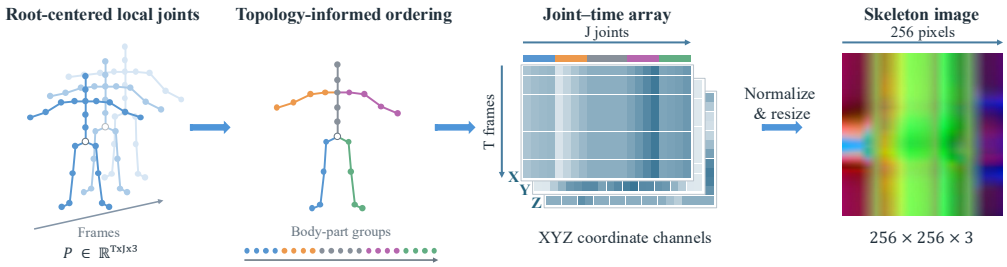


Figure 2: **Topology-informed skeleton image construction.** Local joint coordinates are reordered according to body-part grouping and kinematic-chain topology, arranged into a joint–time array with x, y, and z coordinate channels, and resized to 256×256.

Topology-informed skeleton images. Directly using native joint indices as the image axis would make its spatial organization depend on dataset-specific indexing conventions. As shown in Fig. 2, we reorder local joints into a topology-informed layout and arrange their trajectories into a joint–time array. The ordering follows the left leg, left arm, torso and head, right arm, and right leg. Within each limb, joints follow the kinematic chain from distal to proximal on the left and from proximal to distal on the right, producing an approximately mirrored layout around the central body. This arrangement keeps joints within each body part contiguous and gives the joint axis a meaningful anatomical organization. The complete dataset-specific joint orderings are provided in Appendix A. For dataset d, let $\mathcal{N}_d$ denote the normalization of local joint coordinates and π_d the corresponding joint permutation. We construct

$$\widetilde{\mathbf{P}}^{(d)}_{j,t,:} = \left[\mathcal{N}_d\left(\mathbf{P}^{(d)}\right)\right]_{t,\pi_d(j),:}, \qquad \widetilde{\mathbf{P}}^{(d)} \in \mathbb{R}^{J_d\times T\times 3}. \tag{1}$$

The joint and temporal dimensions form the two image axes, while the x, y, and z coordinates form three channels. We resize this array to a common resolution of $H = W = 256$:

$$\mathbf{I}^{(d)}_{\text{mot}} = \text{Resize}_{H\times W}\left(\widetilde{\mathbf{P}}^{(d)}\right) \in \mathbb{R}^{H\times W\times 3}, \tag{2}$$

Across datasets, the same grouping and ordering principles provide a similar coarse anatomical organization along the joint axis. Together with the common image resolution, this allows the same image encoder and decoder to process motions with different native joint counts, without skeleton retargeting or joint padding. Decoded images are mapped back to each dataset's native joint-time resolution, joint order, and coordinate scale through the recovery procedure in Sec. 3.2.

Complementary root representation. We keep the root features outside the skeleton image. They encode heading changes, two planar root displacements in the heading-aligned coordinate system, and root height above the ground. These quantities describe global root dynamics and do not share the joint-wise spatial organization of local articulation. We therefore model $\mathbf{R}^{(d)}$ through a complementary root-motion stream, coupled with local motion during autoencoding and generation.

For brevity, we omit the dataset superscript below when the formulation applies to both datasets.

3.2 Root-Conditioned Motion Autoencoding

Given the two motion components in Sec. 3.1, our motion autoencoder uses separate variational encoders to map the skeleton image $\mathbf{I}_{\text{mot}}$ and root sequence $\mathbf{R}$ into an image latent and a root latent:

$$\mathbf{z}_{\text{img}} \sim q_{\phi_{\text{img}}}\left(\mathbf{z}_{\text{img}} \mid \mathbf{I}_{\text{mot}}\right), \qquad \mathbf{z}_{\text{root}} \sim q_{\phi_{\text{root}}}\left(\mathbf{z}_{\text{root}} \mid \mathbf{R}\right). \tag{3}$$

We omit root-feature normalization and denormalization from the equations for clarity. The root latent is decoded independently to reconstruct the global root sequence as $\widehat{\mathbf{R}} = D_{\text{root}}(\mathbf{z}_{\text{root}})$. Although encoded separately, local articulation and global movement remain coupled in natural motion. We therefore condition image decoding on the root latent, as shown in Fig. 3 (a). The image decoder first maps the image latent to decoder features. A learned conditioner transforms the root latent to match their channel dimension and temporal resolution, then broadcasts the resulting features along the joint axis. These conditioning features are added to the image decoding features before the first upsampling stage. We denote the complete root-conditioned image decoder by D_{img}:

$$\hat{\mathbf{I}}_{\text{mot}} = D_{\text{img}}(\mathbf{z}_{\text{img}}; \mathbf{z}_{\text{root}}). \tag{4}$$

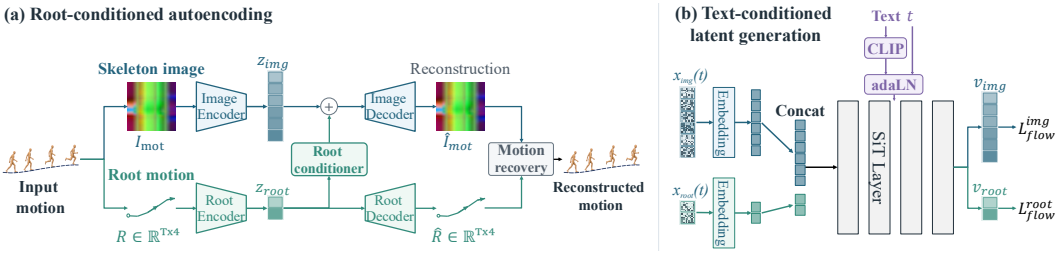


Figure 3: **Overview of the proposed framework.** (a) Separate encoders map skeleton images and root motion into two latent spaces. Root-conditioned decoding and motion recovery reconstruct the input motion. (b) With the autoencoder frozen, noisy image and root latents are tokenized and jointly processed by a text-conditioned SiT to predict their flow-matching velocities.

The root latent thus supports both root-motion reconstruction and local-motion decoding, coupling the two components while retaining separate latent representations.

Motion recovery. The decoded skeleton image is resized to the original joint–time resolution, restored to the native joint order, and denormalized to recover the root-centered 3D joint coordinates $\widehat{\mathbf{P}}$. The reconstructed root features are likewise mapped back to their original scale, yielding the recovered motion representation $(\widehat{\mathbf{R}}, \widehat{\mathbf{P}})$.

3.3 Coupled Root–Pose Latent Generation

With the motion autoencoder frozen, we use a text-conditioned Scalable Interpolant Transformer (SiT) to jointly generate the image and root latents, as illustrated in Fig. 3 (b).

Latent interpolation. For each stream $s \in \{\text{img}, \text{root}\}$, we use the encoder's posterior mean as the clean latent $\mathbf{z}_s$ and independently sample Gaussian noise $\boldsymbol{\epsilon}_s \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$. Both streams share the same generative time $t \in [0, 1]$, which is distinct from the motion-frame index. We use a linear interpolant with its corresponding velocity target:

$$\mathbf{x}_s(t) = t\mathbf{z}_s + (1-t)\boldsymbol{\epsilon}_s, \qquad \mathbf{u}_s = \mathbf{z}_s - \boldsymbol{\epsilon}_s. \tag{5}$$

Text-conditioned joint latent modeling. Given a text description, a frozen text encoder provides both a global text condition $\mathbf{c}_{\text{text}}$ and token-level text features. The global condition is combined with the timestep embedding to modulate the SiT blocks through adaptive layer normalization, while the token-level features are projected into text prefix tokens $\mathbf{h}_{\text{text}}$ for joint self-attention with the motion tokens.

To construct the joint input sequence, we tokenize the current image and root states separately and project the resulting motion tokens to the Transformer's hidden dimension. The text prefix tokens use the same hidden dimension, allowing all three token sequences to be concatenated:

$$\mathbf{H}(t) = \left[\mathbf{h}_{\text{text}}; \text{Tok}_{\text{root}}\left(\mathbf{x}_{\text{root}}(t)\right); \text{Tok}_{\text{img}}\left(\mathbf{x}_{\text{img}}(t)\right)\right], \tag{6}$$

Each token sequence includes its own learnable positional embeddings before concatenation; implementation details are provided in Appendix B. Shared Transformer blocks process this sequence through joint self-attention, allowing the two motion streams to exchange information with each other and the text tokens. Separate output heads map the corresponding Transformer features back to their respective latent spaces and predict a velocity field for each stream:

$$\left(\mathbf{v}_\theta^{\text{img}}, \mathbf{v}_\theta^{\text{root}}\right) = F_\theta\left(\mathbf{H}(t), t; \mathbf{c}_{\text{text}}\right), \tag{7}$$

Each prediction therefore depends on both current latent states as well as the text description, allowing local articulation and global root dynamics to inform one another throughout generation.

Simultaneous sampling. At inference, we initialize both streams with independent Gaussian noise and evolve them jointly from $t = 0$ to $t = 1$ using the predicted velocities. The generated root latent reconstructs the root features and conditions local-motion reconstruction from the generated image latent. This couples global movement and local articulation during both latent generation and motion decoding.

3.4 SHARED MODELING ACROSS SKELETAL LAYOUTS

The common image representation enables shared modeling across HumanML3D and KIT-ML despite their different native joint counts and skeletal definitions. We first train a single motion autoencoder on both datasets, sharing the image and root encoders, decoders, and root conditioner. Each motion is normalized using dataset-specific training statistics and mapped to a fixed-resolution skeleton image with a complementary root sequence. Decoded outputs are restored to the corresponding dataset's native joint–time resolution, joint order, and coordinate scale. This enables parameter sharing without skeleton retargeting or joint padding.

We then freeze the shared autoencoder and train a single SiT on paired latents from both datasets. To specify the target dataset during generation, we augment the global condition with a learnable dataset embedding $\mathbf{e}_d$:

$$\mathbf{c}_d = \tau(t) + \psi(\mathbf{c}_{\text{text}}) + \mathbf{e}_d, \tag{8}$$

where $\tau(t)$ is the timestep embedding, ψ projects the global text condition $\mathbf{c}_{\text{text}}$ to the SiT hidden dimension, and $\mathbf{e}_d$ is the learnable embedding associated with target dataset d. The combined condition modulates the shared Transformer blocks through adaptive layer normalization, while text prefix conditioning remains unchanged.

For classifier-free guidance, we jointly drop the text condition and replace the dataset ID with a null ID during training. At inference, the conditional branch uses the target dataset embedding, while the unconditional branch uses a learned null embedding $\mathbf{e}_0$. Generated latents are decoded by the shared autoencoder and recovered in the target dataset's native skeletal representation.

3.5 TRAINING OBJECTIVES

We first train the motion autoencoder with reconstruction losses and variational regularization, then freeze it and train the SiT with latent flow matching and decoded joint-position supervision.

Autoencoder training. Each latent stream is trained with a reconstruction loss and KL regularization. For the image stream, reconstruction combines mean absolute and mean squared errors between the input and reconstructed skeleton images; for the root stream, it uses an L1 loss on root features. We additionally recover root-centered local 3D joint positions from the reconstructed skeleton images and apply an L1 reconstruction loss in the native joint order and coordinate scale. The autoencoder objective is

$$\mathcal{L}_{\text{VAE}} = \mathcal{L}_{\text{img-rec}} + \mathcal{L}_{\text{root-rec}} + \mathcal{L}_{\text{local-rec}} + \beta_{\text{img}}\mathcal{L}_{\text{KL}}^{\text{img}} + \beta_{\text{root}}\mathcal{L}_{\text{KL}}^{\text{root}}. \tag{9}$$

Here, $\mathcal{L}_{\text{KL}}^{s}$ regularizes each variational posterior toward a standard Gaussian prior, with weights β_{img} and β_{root} for the image and root streams, respectively. Through root-conditioned decoding, the image and local-joint reconstruction losses also propagate gradients to the root encoder. For shared training, we compute the objective for each dataset and aggregate it as $\mathcal{L}_{\text{VAE}}^{\text{shared}} = \sum_d w_d \mathcal{L}_{\text{VAE}}^{(d)}$, where w_d denotes the dataset weight and $\sum_d w_d = 1$.

Latent generator training. With the autoencoder frozen, we train the SiT to predict the velocity targets. For each stream $s \in \{\text{img}, \text{root}\}$, we minimize $\mathcal{L}_{\text{flow}}^{s} = \mathbb{E}[\text{MSE}(\mathbf{v}_\theta^s, \mathbf{u}_s)]$. The expectation is taken over paired motion–text samples, interpolation times, and Gaussian noise.

To constrain generation in motion space, we additionally supervise local 3D joint positions decoded from predicted clean latents. Under the linear interpolant, these estimates are

$$\widehat{\mathbf{z}}_s = \mathbf{x}_s(t) + (1-t)\mathbf{v}_\theta^s. \tag{10}$$

We decode the estimated image latent conditioned on the estimated root latent and recover local joint coordinates:

$$\hat{\mathbf{P}}_\theta = \mathcal{G}(D_{\text{img}}(\hat{\mathbf{z}}_{\text{img}}; \hat{\mathbf{z}}_{\text{root}})), \tag{11}$$

where D_{img} is the root-conditioned image decoder, and $\mathcal{G}$ is the image-to-motion recovery operation. For shared modeling, we use the corresponding dataset-specific operator $\mathcal{G}_d$. The position loss is $\mathcal{L}_{\text{pos}} = \mathbb{E}[\text{MAE}(\widehat{\mathbf{P}}_\theta, \mathbf{P})]$. Although the autoencoder parameters remain frozen, gradients propagate through the decoder and root conditioner to both predicted latent streams.

Combining latent flow matching with decoded geometry supervision, the final SiT objective is

$$\mathcal{L}_{\text{SiT}} = \mathcal{L}_{\text{flow}}^{\text{img}} + \lambda_{\text{root}}\mathcal{L}_{\text{flow}}^{\text{root}} + \lambda_{\text{pos}}\mathcal{L}_{\text{pos}}. \tag{12}$$

Table 1: Quantitative comparison on HumanML3D using the MARDM evaluation protocol. **Bold** and underlined values indicate the best and second-best results, respectively. Shared models are jointly trained on HumanML3D and KIT-ML. Confidence intervals are included where reported.

Method	R-Precision ↑			FID ↓	Matching ↓	CLIP-score ↑
	Top-1	Top-2	Top-3			
Motion Discrete / Token-based						
T2M-GPT (Zhang et al., 2023a)	$0.470^{\pm.003}$	$0.659^{\pm.002}$	$0.758^{\pm.002}$	$0.335^{\pm.003}$	$3.505^{\pm.017}$	$0.607^{\pm.005}$
MMM (Pinyoanuntapong et al., 2024b)	$0.487^{\pm.003}$	$0.683^{\pm.002}$	$0.782^{\pm.001}$	$0.132^{\pm.004}$	$3.359^{\pm.009}$	$0.635^{\pm.003}$
MoMask (Guo et al., 2024)	$0.490^{\pm.004}$	$0.687^{\pm.003}$	$0.786^{\pm.003}$	$0.116^{\pm.006}$	$3.353^{\pm.010}$	$0.637^{\pm.003}$
BAMM (Pinyoanuntapong et al., 2024a)	$0.480^{\pm.003}$	$0.671^{\pm.003}$	$0.772^{\pm.003}$	$0.147^{\pm.005}$	–	$0.630^{\pm.001}$
MoMask++ (Guo et al., 2025)	$0.492^{\pm.003}$	$0.683^{\pm.003}$	$0.782^{\pm.002}$	$0.108^{\pm.007}$	–	$0.634^{\pm.001}$
Motion Diffusion / Flow-based						
MDM-50Step (Tevet et al., 2023)	$0.440^{\pm.007}$	$0.636^{\pm.006}$	$0.742^{\pm.004}$	$0.518^{\pm.032}$	$3.640^{\pm.028}$	$0.578^{\pm.003}$
MotionDiffuse (Zhang et al., 2024)	$0.450^{\pm.006}$	$0.641^{\pm.005}$	$0.753^{\pm.005}$	$0.778^{\pm.005}$	$3.490^{\pm.023}$	$0.606^{\pm.004}$
MLD (Chen et al., 2023)	$0.461^{\pm.004}$	$0.651^{\pm.004}$	$0.750^{\pm.003}$	$0.431^{\pm.014}$	$3.445^{\pm.019}$	$0.610^{\pm.003}$
ReMoDiffuse (Zhang et al., 2023b)	$0.468^{\pm.003}$	$0.653^{\pm.003}$	$0.754^{\pm.005}$	$0.883^{\pm.021}$	$3.414^{\pm.020}$	$0.621^{\pm.003}$
MARDM-DDPM-XL (Meng et al., 2025)	$0.492^{\pm.006}$	$0.690^{\pm.005}$	$0.790^{\pm.005}$	$0.116^{\pm.004}$	$3.349^{\pm.010}$	$0.637^{\pm.005}$
MARDM-SiT-XL (Meng et al., 2025)	$0.500^{\pm.004}$	$0.695^{\pm.003}$	$0.795^{\pm.003}$	$0.114^{\pm.007}$	$3.270^{\pm.009}$	$0.642^{\pm.002}$
MLD++ (Dai et al., 2025)	$0.500^{\pm.003}$	$0.691^{\pm.002}$	$0.789^{\pm.001}$	$2.027^{\pm.021}$	$3.220^{\pm.008}$	$0.639^{\pm.002}$
MotionLCM-V2 (Dai et al., 2025)	$0.501^{\pm.002}$	$0.693^{\pm.002}$	$0.790^{\pm.002}$	$2.267^{\pm.023}$	$3.192^{\pm.009}$	$0.640^{\pm.003}$
Motion LLM-based						
MotionLLM (Wu et al., 2025)	0.470	0.664	0.771	0.240	3.377	–
IRG-MotionLLM (S1 + IRMoGen) (Li et al., 2026)	0.507	0.696	0.791	0.161	3.234	–
IRG-MotionLLM (S3 + IRMoGen) (Li et al., 2026)	0.509	0.701	0.802	0.259	<u>3.166</u>	–
Image Diffusion / Flow-based						
Ours (SiT-L)	<u>$0.540^{\pm.004}$</u>	<u>$0.727^{\pm.002}$</u>	<u>$0.818^{\pm.002}$</u>	<u>$0.089^{\pm.005}$</u>	$3.226^{\pm.010}$	<u>$0.654^{\pm.001}$</u>
Ours (SiT-L, Shared)	$\mathbf{0.546^{\pm.002}}$	$\mathbf{0.733^{\pm.002}}$	$\mathbf{0.822^{\pm.002}}$	$\mathbf{0.082^{\pm.004}}$	$\mathbf{3.154^{\pm.006}}$	$\mathbf{0.659^{\pm.001}}$

Here, λ_{root} and λ_{pos} weight the root-stream flow-matching loss and decoded joint-position loss, respectively. For shared training, samples from both datasets use their corresponding dataset conditions. We compute the flow-matching and decoded position losses for each dataset and aggregate the generator objectives as $\mathcal{L}_{\text{SiT}}^{\text{shared}} = \sum_d w_d \mathcal{L}_{\text{SiT}}^{(d)}$.

4 Experiments

4.1 Experimental Setup

Datasets. We evaluate on HumanML3D (Guo et al., 2022) and KIT-ML (Plappert et al., 2016), following the standard T2M preprocessing protocol (Guo et al., 2022). HumanML3D contains 14,616 motion sequences paired with 44,970 textual descriptions, while KIT-ML contains 3,911 motion sequences with 6,278 textual annotations. Motions are processed at 20 FPS for HumanML3D and 12.5 FPS for KIT-ML. We follow the standard training, validation, and test splits (approximately 80%, 5%, and 15%, respectively) and use mirroring augmentation with corresponding text adjustments. Following MARDM (Meng et al., 2025), we use four-dimensional root features and root-centered local joint positions, yielding 67-dimensional and 64-dimensional per-frame representations for HumanML3D and KIT-ML, respectively (Sec. 3.1).

Evaluation Metrics. We follow the MARDM evaluation protocol (Meng et al., 2025), using its pretrained evaluators for the 67-dimensional HumanML3D and 64-dimensional KIT-ML representations. The T2M-style evaluator (Guo et al., 2022) measures text–motion alignment through R-Precision (Top-1, Top-2, and Top-3; higher is better) and Matching Score (lower is better). We also report Fréchet Inception Distance (FID; lower is better) between generated and real motion embeddings to assess distributional similarity. Finally, we use MARDM's CLIP-based motion–text evaluator to report CLIP-score (higher is better) as an additional measure of text–motion alignment. We repeat evaluation 20 times and report mean results with 95% confidence intervals.

Implementation Details. We train both dataset-specific models and a shared model on HumanML3D and KIT-ML using the two-stage procedure in Sec. 3.5. The motion autoencoder uses an ImageNet-pretrained ResNet-18 image encoder (He et al., 2016; Deng et al., 2009) and a 1D convolutional root encoder. With the autoencoder frozen, we train SiT-L (Ma et al., 2024) to jointly generate both latent streams, using a frozen CLIP ViT-B/32 text encoder (Radford et al., 2021) for conditioning. For shared training, we mix HumanML3D and KIT-ML samples at a 3:1 ratio, approximately reflecting their relative dataset sizes, and use corresponding dataset weights of 0.75 and

Table 2: Quantitative comparison on KIT-ML using the MARDM evaluation protocol.

Method	R-Precision ↑			FID ↓	Matching ↓	CLIP-score ↑
	Top-1	Top-2	Top-3			
Motion Discrete / Token-based						
T2M-GPT (Zhang et al., 2023a)	$0.359^{\pm.007}$	$0.553^{\pm.007}$	$0.690^{\pm.013}$	$0.593^{\pm.053}$	$3.765^{\pm.046}$	$0.651^{\pm.005}$
MMM (Pinyoanuntapong et al., 2024b)	$0.363^{\pm.005}$	$0.569^{\pm.006}$	$0.724^{\pm.006}$	$0.478^{\pm.034}$	$3.629^{\pm.028}$	$0.660^{\pm.003}$
MoMask (Guo et al., 2024)	$0.369^{\pm.005}$	$0.588^{\pm.005}$	$0.731^{\pm.005}$	$0.411^{\pm.026}$	$3.577^{\pm.021}$	$0.669^{\pm.002}$
Motion Diffusion / Flow-based						
MDM (Tevet et al., 2023)	$0.333^{\pm.012}$	$0.561^{\pm.009}$	$0.689^{\pm.009}$	$0.585^{\pm.043}$	$4.002^{\pm.033}$	$0.605^{\pm.007}$
MotionDiffuse (Zhang et al., 2024)	$0.344^{\pm.009}$	$0.536^{\pm.007}$	$0.658^{\pm.007}$	$3.845^{\pm.087}$	$4.167^{\pm.054}$	$0.626^{\pm.006}$
MLD (Chen et al., 2023)	$0.351^{\pm.007}$	$0.536^{\pm.007}$	$0.658^{\pm.007}$	$0.492^{\pm.047}$	$3.746^{\pm.044}$	$0.646^{\pm.006}$
ReMoDiffuse (Zhang et al., 2023b)	$0.356^{\pm.004}$	$0.572^{\pm.007}$	$0.706^{\pm.009}$	$1.725^{\pm.053}$	$3.735^{\pm.036}$	$0.665^{\pm.005}$
MARDM-DDPM-XL (Meng et al., 2025)	$0.375^{\pm.006}$	$0.597^{\pm.008}$	$0.739^{\pm.006}$	$0.340^{\pm.020}$	$3.489^{\pm.018}$	$0.681^{\pm.003}$
MARDM-SiT-XL (Meng et al., 2025)	$0.387^{\pm.006}$	$0.610^{\pm.006}$	$0.749^{\pm.006}$	$0.242^{\pm.014}$	$3.374^{\pm.019}$	$0.692^{\pm.002}$
Image Diffusion / Flow-based						
Ours (SiT-L)	$\underline{0.395}^{\pm.004}$	$\underline{0.620}^{\pm.007}$	$\underline{0.758}^{\pm.004}$	$\underline{0.171}^{\pm.010}$	$\underline{3.331}^{\pm.020}$	$\underline{0.696}^{\pm.002}$
Ours (SiT-L, Shared)	$\mathbf{0.405}^{\pm.006}$	$\mathbf{0.634}^{\pm.006}$	$\mathbf{0.770}^{\pm.007}$	$\mathbf{0.145}^{\pm.010}$	$\mathbf{3.292}^{\pm.011}$	$\mathbf{0.708}^{\pm.002}$

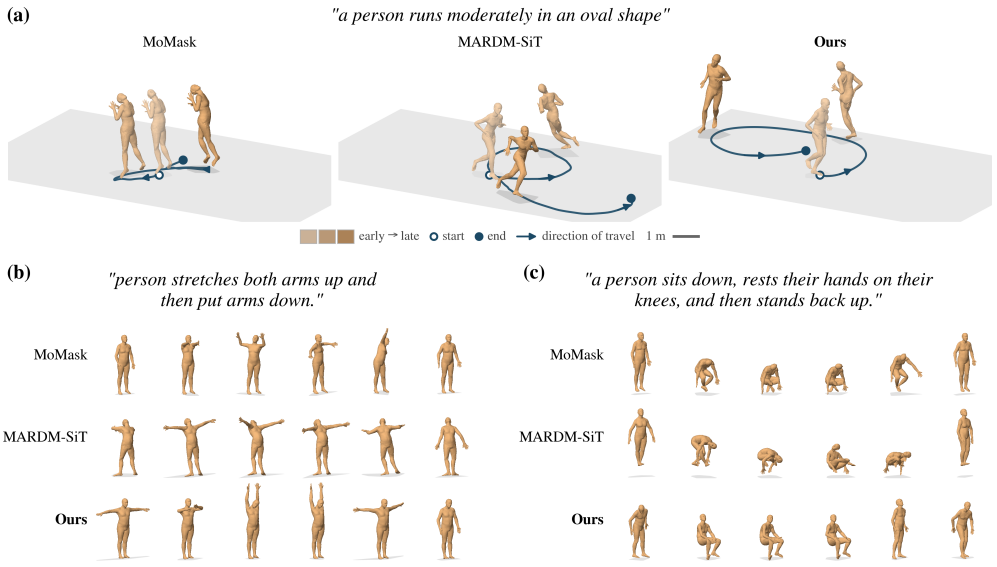


Figure 4: **Qualitative comparisons on HumanML3D.** We compare our method with MoMask and MARDM-SiT. (a) Motion snapshots with root trajectories, where lighter-to-darker meshes indicate temporal progression. (b)–(c) Motion sequences shown chronologically from left to right.

0.25. At inference, we jointly sample both latent streams with a 50-step Euler solver and classifier-free guidance (Ho & Salimans, 2022), using guidance scales of 5.5 for HumanML3D and 4.5 for KIT-ML. All experiments are implemented in PyTorch and run on NVIDIA A100 GPUs. Further architecture and training details are provided in Appendix B.

4.2 Comparison with Existing Methods

We compare with representative text-to-motion approaches on HumanML3D and KIT-ML, including discrete/token-based, diffusion/flow-based, and LLM-based methods (Tables 1 and 2).

Dataset-specific generation. On HumanML3D, our dataset-specific SiT-L model achieves the highest R-Precision at all three retrieval levels, the lowest FID, and the highest CLIP-score among the dataset-specific methods reported in Table 1. It also obtains a competitive Matching Score of 3.226. Compared with MARDM-SiT-XL, which uses the same compact root-plus-position motion representation, our model improves Top-1 R-Precision from 0.500 to 0.540 and reduces FID from 0.114 to 0.089. On KIT-ML, our model achieves the best results across all six reported metrics among the dataset-specific methods in Table 2. In particular, FID decreases from 0.242 for MARDM-SiT-XL to 0.171, while Top-1 R-Precision increases from 0.387 to 0.395. These results demonstrate the effectiveness of our framework for text-to-motion generation on both benchmarks.

Shared modeling across datasets. We train a single motion autoencoder on HumanML3D and KIT-ML, then freeze it to train a shared SiT-L generator. Compared with our dataset-specific models, shared training improves all six reported metrics on both benchmarks. On HumanML3D, FID decreases from 0.089 to 0.082 and Top-1 R-Precision increases from 0.540 to 0.546. On KIT-ML, FID decreases from 0.171 to 0.145 and Top-1 R-Precision increases from 0.395 to 0.405. The shared model achieves the best reported results across all six metrics in both tables. These results support the effectiveness of shared modeling across the two skeletal layouts using our common image representation, without skeleton retargeting or joint padding.

4.3 Qualitative Results

Fig. 4 compares our method with MoMask and MARDM-SiT on three text prompts. Our method captures running along an oval path in (a), overhead arm extension followed by lowering in (b), and sitting with hands resting on the knees followed by standing in (c). These examples illustrate the model's ability to capture both global movement and local articulation specified by text.

4.4 Ablation Studies

We conduct ablations on HumanML3D using SiT-B with $\lambda_{\text{pos}} = \lambda_{\text{root}} = 0.25$ unless otherwise specified. The dataset-conditioning ablation uses shared models trained on HumanML3D and KIT-ML; all other ablations use dataset-specific training. More ablations are provided in Appendix C.

Table 3: Model ablations on HumanML3D.

Setting	FID ↓	R-Precision ↑		
		Top-1	Top-2	Top-3
Model scaling				
SiT-S	$0.139^{\pm.004}$	$0.527^{\pm.004}$	$0.713^{\pm.002}$	$0.804^{\pm.002}$
SiT-B	$0.106^{\pm.006}$	$0.533^{\pm.003}$	$0.717^{\pm.002}$	$0.810^{\pm.002}$
SiT-L	$\mathbf{0.089^{\pm.005}}$	$\mathbf{0.540^{\pm.004}}$	$\mathbf{0.727^{\pm.002}}$	$\mathbf{0.818^{\pm.002}}$
ImageNet initialization				
w/o ImageNet	$0.145^{\pm.006}$	$0.517^{\pm.003}$	$0.705^{\pm.003}$	$0.800^{\pm.002}$
w/ ImageNet	$\mathbf{0.106^{\pm.006}}$	$\mathbf{0.533^{\pm.003}}$	$\mathbf{0.717^{\pm.002}}$	$\mathbf{0.810^{\pm.002}}$
Root-conditioned decoding				
w/o condition	$0.116^{\pm.005}$	$0.525^{\pm.004}$	$0.714^{\pm.003}$	$0.807^{\pm.003}$
w/ condition	$\mathbf{0.106^{\pm.006}}$	$\mathbf{0.533^{\pm.003}}$	$\mathbf{0.717^{\pm.002}}$	$\mathbf{0.810^{\pm.002}}$
Dataset conditioning (Shared)				
w/o condition	$0.278^{\pm.015}$	$0.515^{\pm.005}$	$0.694^{\pm.008}$	$0.781^{\pm.004}$
w/ condition	$\mathbf{0.093^{\pm.008}}$	$\mathbf{0.536^{\pm.004}}$	$\mathbf{0.726^{\pm.007}}$	$\mathbf{0.818^{\pm.004}}$

Model scaling. We compare SiT-S, SiT-B, and SiT-L to examine the effect of generator capacity. As shown in Table 3, FID decreases from 0.139 to 0.106 and 0.089, while all three R-Precision metrics improve. These results show that our framework benefits from increased generator capacity.

ImageNet initialization. We examine the effect of ImageNet initialization for the image encoder. Table 3 shows that using pretrained weights reduces FID from 0.145 to 0.106 and improves Top-1 R-Precision from 0.517 to 0.533. These results show that ImageNet initialization benefits downstream motion generation with our image-compatible representation.

Root-conditioned decoding. We evaluate the effect of conditioning image decoding on the root latent. As reported in Table 3, removing this conditioning increases FID from 0.106 to 0.116 and decreases Top-1 R-Precision from 0.533 to 0.525. This supports the benefit of incorporating root-motion context into local-motion decoding.

Dataset conditioning. We compare shared SiT-B models with and without the dataset condition. The results in Table 3 show that adding this condition reduces FID from 0.278 to 0.093 and improves Top-3 R-Precision from 0.781 to 0.818. Without it, shared training performs worse than the dataset-specific baseline with FID 0.106. These results suggest that explicit dataset conditioning helps distinguish the motion distributions of the two datasets within the shared representation.

5 Conclusion

In this work, we show that image generative architectures can effectively model text-conditioned 3D human motion through a structured, image-compatible representation. Our framework combines topology-informed joint–time images with a complementary root-motion stream, coupling the two through root-conditioned autoencoding and joint SiT generation. Supervision on decoded 3D joint positions further constrains generation in motion space. This representation also enables a single motion autoencoder and a single SiT to model HumanML3D and KIT-ML jointly, preserving their native skeletal definitions without retargeting. Experiments demonstrate state-of-the-art dataset-specific results and further improvements through shared training on both benchmarks. These findings highlight the potential of image-compatible representations for motion generation across heterogeneous skeletal layouts. Future work will explore this formulation across a broader range of skeletal structures and motion datasets.

AI USE STATEMENT

In this work, we used generative AI tools to polish the writing and check experimental code for errors and consistency. We have not used generative AI tools to generate or modify experimental results. We have reviewed all AI-assisted work, including writing revisions and code-related suggestions, and independently verified the experimental results, quantitative comparisons, and scientific claims. We take responsibility for the final content of this work, including text, claims, or artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

We provide detailed descriptions of the motion representation, model architecture, training objectives, dataset preprocessing, optimization settings, and inference procedure in the main paper and appendix. The experiments use the public HumanML3D and KIT-ML datasets and follow their standard train/validation/test splits and evaluation protocols. Additional implementation details, including the shared-training configuration, dataset-specific normalization and recovery procedures, and hyperparameter settings, are provided to facilitate reproduction. We plan to release the training and evaluation code upon publication.

REFERENCES

Kfir Aberman, Peizhuo Li, Dani Lischinski, Olga Sorkine-Hornung, Daniel Cohen-Or, and Baoquan Chen. Skeleton-aware networks for deep motion retargeting. *ACM Transactions on Graphics (ToG)*, 39(4):62–1, 2020.

Carlos Caetano, Jessica Sena, François Brémond, Jefersson A Dos Santos, and William Robson Schwartz. Skelemotion: A new representation of skeleton joint sequences based on motion information for 3d action recognition. In *2019 16th IEEE international conference on advanced video and signal based surveillance (AVSS)*, pp. 1–8. IEEE, 2019.

Xin Chen, Biao Jiang, Wen Liu, Zilong Huang, Bin Fu, Tao Chen, and Gang Yu. Executing your commands via motion diffusion in latent space. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 18000–18010. IEEE, 2023.

Wenxun Dai, Ling-Hao Chen, Jingbo Wang, Jinpeng Liu, Bo Dai, and Yansong Tang. Motionlcm: Real-time controllable motion generation via latent consistency model. In Aleš Leonardis, Elisa Ricci, Stefan Roth, Olga Russakovsky, Torsten Sattler, and Gül Varol (eds.), *Computer Vision – ECCV 2024*, pp. 390–408, Cham, 2025. Springer Nature Switzerland. ISBN 978-3-031-72640-8.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 248–255. IEEE, 2009.

Inbar Gat, Sigal Raab, Guy Tevet, Yuval Reshef, Amit Haim Bermano, and Daniel Cohen-Or. Anytop: Character animation diffusion with any topology. In *Proceedings of the Special Interest Group on Computer Graphics and Interactive Techniques Conference Conference Papers*, pp. 1–10, 2025.

Chuan Guo, Shihao Zou, Xinxin Zuo, Sen Wang, Wei Ji, Xingyu Li, and Li Cheng. Generating diverse and natural 3d human motions from text. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5142–5151. IEEE, 2022.

Chuan Guo, Yuxuan Mu, Muhammad Gohar Javed, Sen Wang, and Li Cheng. Momask: Generative masked modeling of 3d human motions. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1900–1910. IEEE, 2024.

Chuan Guo, Inwoo Hwang, Jian Wang, and Bing Zhou. Snapmogen: Human motion generation from expressive texts. In D. Belgrave, C. Zhang, H. Lin, R. Pascanu, P. Koniusz, M. Ghassemi, and N. Chen (eds.), *Advances in Neural Information Processing Systems*, volume 38, Main Conference, pp. 99939–99955. Curran Associates, Inc., 2025. doi: 10.52202/085713-3341. URL https://proceedings.neurips.cc/paper_files/paper/2025/file/9028b8a3ca98f58e373f0c1497a17448-Paper-Conference.pdf.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598*, 2022. URL https://arxiv.org.

Qiuhong Ke, Mohammed Bennamoun, Senjian An, Ferdous Sohel, and Farid Boussaid. A new representation of skeleton sequences for 3d action recognition. In *2017 IEEE conference on computer vision and pattern recognition (CVPR)*, pp. 4570–4579. IEEE, 2017.

Aliasghar Khani, Arianna Rampini, Evan Atherton, and Bruno Roy. Unimogen: Universal motion generation. *arXiv preprint arXiv:2505.21837*, 2025.

Yuan-Ming Li, Qize Yang, Nan Lei, Shenghao Fu, Ling-An Zeng, Jian-Fang Hu, Xihan Wei, and Wei-Shi Zheng. Irg-motionllm: Interleaving motion generation, assessment and refinement for text-to-motion generation. In *European Conference on Computer Vision*, pp. 37–57. Springer, 2026.

Qingyuan Liu, Ke Lv, Kun Dong, Jian Xue, Zehai Niu, and Jinbao Wang. Text-to-any-skeleton motion generation without retargeting. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 12926–12936, 2025.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations (ICLR)*, 2019. URL https://openreview.net/forum?id=Bkg6RiCqY7.

Nanye Ma, Mark Goldstein, Michael S Albergo, Nicholas M Boffi, Eric Vanden-Eijnden, and Saining Xie. Sit: Exploring flow and diffusion-based generative models with scalable interpolant transformers. In *European Conference on Computer Vision*, pp. 23–40. Springer, 2024.

Zichong Meng, Yiming Xie, Xiaogang Peng, Zeyu Han, and Huaizu Jiang. Rethinking diffusion for text-driven human motion generation: Redundant representations, evaluation, and masked autoregression. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 27859–27871. IEEE, 2025.

William Peebles and Saining Xie. Scalable diffusion models with transformers. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 4172–4182. IEEE, 2023.

Mathis Petrovich, Michael J. Black, and Gül Varol. TEMOS: Generating diverse human motions from textual descriptions. In *European Conference on Computer Vision (ECCV)*, 2022.

Ekkasit Pinyoanuntapong, Muhammad Usama Saleem, Pu Wang, Minwoo Lee, Srijan Das, and Chen Chen. Bamm: Bidirectional autoregressive motion model. In *European Conference on Computer Vision*, pp. 172–190. Springer, 2024a.

Ekkasit Pinyoanuntapong, Pu Wang, Minwoo Lee, and Chen Chen. Mmm: Generative masked motion model. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1546–1555. IEEE, 2024b.

Matthias Plappert, Christian Mandery, and Tamim Asfour. The kit motion-language dataset. *Big data*, 4(4):236–252, 2016.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Pamela Askell, Amanda Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning (ICML)*, pp. 8748–8763. PMLR, 2021. URL https://proceedings.mlr.press/v139/radford21a.html.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *2022 IEEE/CVF conference on computer vision and pattern recognition (CVPR)*, pp. 10674–10685. ieee, 2022.

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

Qi Wu, Yubo Zhao, Yifan Wang, Xinhang Liu, Yu-Wing Tai, and Chi-Keung Tang. Motion-agent: A conversational framework for human motion generation with llms. In *International Conference on Learning Representations*, volume 2025, pp. 48023–48043, 2025.

Jianrong Zhang, Yangsong Zhang, Xiaodong Cun, Yong Zhang, Hongwei Zhao, Hongtao Lu, Xi Shen, and Ying Shan. Generating human motion from textual descriptions with discrete representations. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 14730–14740, 2023a.

Mingyuan Zhang, Xinying Guo, Liang Pan, Zhongang Cai, Fangzhou Hong, Huirong Li, Lei Yang, and Ziwei Liu. Remodiffuse: Retrieval-augmented motion diffusion model. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 364–373. IEEE, 2023b.

Mingyuan Zhang, Zhongang Cai, Liang Pan, Fangzhou Hong, Xinying Guo, Lei Yang, and Ziwei Liu. Motiondiffuse: Text-driven human motion generation with diffusion model. *IEEE transactions on pattern analysis and machine intelligence*, 46(6):4115–4128, 2024.

A Topology-informed Joint Ordering

We organize local joints into five consecutive body-part groups: left leg, left arm, torso and head, right arm, and right leg. Within the left limbs, joints are ordered from distal to proximal; within the right limbs, they are ordered from proximal to distal. The central group follows the torso toward the head. This arrangement keeps each body part contiguous along the joint axis and provides a similar coarse anatomical organization across datasets.

Table 4 lists the joint names and native indices in the order used to construct the skeleton images. The root joint has native index 0 in both datasets and is excluded from the local joint array, leaving 21 local joints for HumanML3D and 20 for KIT-ML. We retain each dataset's native root definition: the pelvis in HumanML3D and the root joint in KIT-ML.

Table 4: Body-part grouping and joint ordering for skeleton image construction. Rows follow the group order along the joint axis; joints within each cell are listed in traversal order. Parentheses indicate zero-based native joint indices, before root removal.

Body part	HumanML3D	KIT-ML
Left leg	Foot (10), ankle (7), knee (4), hip (1)	Toe (20), mid-foot (19), ankle (18), knee (17), hip (16)
Left arm	Wrist (20), elbow (18), shoulder (16), collar (13)	Wrist (10), elbow (9), shoulder (8)
Torso and head	Spine1 (3), spine2 (6), spine3 (9), neck (12), head (15)	Pelvis spine (1), thorax (2), lower neck (3), upper neck (4)
Right arm	Collar (14), shoulder (17), elbow (19), wrist (21)	Shoulder (5), elbow (6), wrist (7)
Right leg	Hip (2), knee (5), ankle (8), foot (11)	Hip (11), knee (12), ankle (13), mid-foot (14), toe (15)

The datasets differ in both joint counts and anatomical partitioning. Each leg contains four local joints in HumanML3D and five in KIT-ML, whereas each arm contains four and three, respectively; the torso-and-head groups contain five and four.

Concatenating the ordered groups in Table 4 specifies the joint-axis ordering. The table uses native skeleton indices, whereas the permutation π_d in Sec. 3.1 acts on the local joint array after root removal. During recovery, the inverse permutation restores the dataset-specific local joint order.

The grouping does not establish a one-to-one joint correspondence between datasets and requires neither skeleton retargeting nor joint padding.

B Implementation Details

B.1 Motion Autoencoder Architecture

Image encoder. Skeleton images have a resolution of 256×256. The image encoder uses the stem and the first three residual stages of an ImageNet-pretrained ResNet-18 backbone. The resulting image latent has 16 channels and a spatial resolution of 16×16.

Image decoder. The image decoder first maps the 16-channel image latent to 256-channel features using a 3×3 convolution with padding 1, followed by group normalization with 32 groups, a SiLU activation, and a residual block. Root conditioning is added to these features before the first upsampling stage. Four successive stages then increase the spatial resolution from 16×16 to 256×256, with channel dimensions $256 \to 128 \to 64 \to 32 \to 16$. Each stage consists of $2\times$ nearest-neighbor upsampling, a 3×3 convolution, group normalization, a SiLU activation, and a residual block. Group normalization uses 32 groups in the first three stages and 16 groups in the final stage. The output head applies a 3×3 convolution with 16 output channels, group normalization with 16 groups, a SiLU activation, and a final 3×3 convolution to reconstruct the three-channel skeleton image. Together with the root conditioner, this complete decoding process is denoted by $D_{\mathrm{img}}(\mathbf{z}_{\mathrm{img}}; \mathbf{z}_{\mathrm{root}})$ in Sec. 3.2.

Root encoder. The four-channel root sequence is first linearly interpolated to 16 temporal positions, then processed by two length-preserving 1D convolutions with kernel size 3 and channel dimensions $4 \to 128 \to 128$. Each convolution is followed by group normalization with 32 groups and a SiLU activation. Separate kernel-size-one convolutional heads predict the posterior mean and log variance, yielding a root latent of size 8×16 (channels $\times$ temporal positions). Temporal compression is performed by the initial interpolation, while the convolutional layers preserve the sequence length.

Root decoder. The root decoder applies two length-preserving 1D convolutions with kernel size 3 and channel dimensions $8 \to 128 \to 128$. Each convolution is followed by group normalization with 32 groups and a SiLU activation. A final kernel-size-one convolution maps the features to four root channels, which are then linearly interpolated to the original sequence length T.

Root conditioner. The root conditioner uses two kernel-size-one 1D convolutions with channel dimensions $8 \to 256 \to 256$. Group normalization with 32 groups and a SiLU activation are applied between the two convolutions. The resulting features are temporally aligned with the image decoding features by linear interpolation when needed, then broadcast along the joint axis. In our configuration, this produces a conditioning tensor with 256 channels and a spatial resolution of 16×16. It is addedafter the image decoder's initial feature-mapping block, before any upsampling.

B.2 SiT Architecture and Text Conditioning

Model configurations. We use SiT-L for the main experiments and examine SiT-S and SiT-B in the model-size ablation. Table 5 summarizes their configurations. All variants use an attention head dimension of 64, an MLP ratio of 4, and an image-latent patch size of 1×1. Parameter counts refer to the trainable parameters of our instantiated generators.

Table 5: SiT configurations and trainable parameter counts.

Model	Depth	Hidden dimension	Heads	Parameters
SiT-S	12	384	6	33.4M
SiT-B	12	768	12	132.3M
SiT-L	24	1024	16	461.3M

Latent tokenization and prediction. The image and root latents are tokenized into 256 image tokens and 16 root tokens, respectively. Each image token corresponds to one spatial position in the

Table 6: Motion autoencoder training settings.

Setting	HumanML3D	KIT-ML	Shared
Training window (frames)	128	64	128 (HumanML3D) 64 (KIT-ML)
Effective batch size	256	512	256
Epochs	100	50	100
Optimizer	AdamW		
Learning rate	10^{-3}		
Weight decay	0.05		
Learning-rate schedule	Cosine		
LR warmup (epochs)	1	5	1
KL warmup (epochs)	2	5	2
Image KL weight β_{img}	7×10^{-4}		
Root KL weight β_{root}	1×10^{-4}		
Training precision	FP32		

image latent, while each root token corresponds to one temporal position in the root latent. Separate learned projections map both token sequences to the Transformer hidden dimension d.

Independent learnable positional embeddings are added to the image, root, and text prefix tokens before concatenation. The corresponding embedding tables have shapes $256 \times d$, $16 \times d$, and $8 \times d$, and are initialized from a zero-mean Gaussian distribution with a standard deviation of 0.02. The sequences are concatenated in the order of text, root, and image, yielding a joint sequence of 280 tokens. No additional positional embedding is applied after concatenation. Shared Transformer blocks process this sequence through joint self-attention. Separate output heads map the image and root token features back to their respective latent spaces to predict the two velocity fields.

Text and dataset conditioning. Text descriptions are encoded using a frozen CLIP ViT-B/32 text encoder. Its pooled embedding provides the global text condition, which is combined with the interpolation-time embedding to condition the Transformer blocks through adaptive layer normalization. To also retain token-level text information, we uniformly select eight valid text tokens and project their embeddings to the Transformer hidden dimension. After adding their learnable positional embeddings, these tokens are prepended to the motion tokens as described above. For shared modeling, a learned dataset embedding is additionally included in the global condition (Sec. 3.4).

B.3 Training and Sampling

We follow the two-stage procedure in Sec. 3.5, first training the motion autoencoder and then freezing it to train SiT. We report the settings for HumanML3D, KIT-ML, and shared training separately in Tables 6 and 7. All batch sizes are effective batch sizes across GPUs.

Autoencoder training. We train the motion autoencoder on sliding windows of 128 frames for HumanML3D and 64 frames for KIT-ML. Shared training retains these dataset-specific window lengths. We use AdamW (Loshchilov & Hutter, 2019) with a learning-rate warmup followed by cosine decay. The KL regularization weights are also warmed up at the beginning of training, with the warmup durations and target weights listed in Table 6.

SiT training. With the motion autoencoder frozen, we use its posterior means as clean latent targets and train SiT on motions of up to 196 frames. We optimize SiT using AdamW with a constant learning rate, without learning-rate warmup or weight decay. Training uses automatic mixed precision (FP16) without gradient clipping. We maintain an exponential moving average (EMA) of the generator parameters and use the EMA weights for evaluation. Conditioning dropout enables classifier-free guidance, while the root-stream and decoded joint-position loss weights control their contributions to the training objective. The corresponding settings are listed in Table 7.

Table 7: SiT training settings.

Setting	HumanML3D	KIT-ML	Shared
Maximum motion length (frames)	196		
Effective batch size	256	16	256
Epochs	2000		
Optimizer	AdamW		
Adam betas (β_1, β_2)	$(0.9, 0.999)$		
Learning rate	10^{-4}		
Learning-rate schedule	Constant		
EMA decay	0.9999		
Training precision	AMP (FP16)		
Conditioning dropout probability	0.1		
Root loss weight λ_{root}	0.25		
Position loss weight λ_{pos}	0.25		

Table 8: Decoded geometry weight ablation on HumanML3D, with $\lambda_{\text{root}} = 0.25$.

λ_{pos}	FID ↓	R-Precision ↑		
		Top-1	Top-2	Top-3
0 (w/o loss)	$0.133^{\pm.006}$	$0.520^{\pm.003}$	$0.709^{\pm.003}$	$0.802^{\pm.002}$
0.25 (default)	$\mathbf{0.106}^{\pm.006}$	$\mathbf{0.533}^{\pm.003}$	$\mathbf{0.717}^{\pm.002}$	$\mathbf{0.810}^{\pm.002}$
0.50	$0.112^{\pm.005}$	$0.529^{\pm.003}$	$0.715^{\pm.002}$	$0.809^{\pm.003}$
0.75	$0.114^{\pm.004}$	$0.529^{\pm.002}$	$0.716^{\pm.002}$	$0.808^{\pm.003}$
1.00	$0.118^{\pm.005}$	$0.528^{\pm.002}$	$0.713^{\pm.003}$	$0.805^{\pm.002}$

Table 9: Root flow-matching weight ablation on HumanML3D, with $\lambda_{\text{pos}} = 0.25$.

λ_{root}	FID ↓	R-Precision ↑		
		Top-1	Top-2	Top-3
0.25 (default)	$\mathbf{0.106}^{\pm.006}$	$\mathbf{0.533}^{\pm.003}$	$\mathbf{0.717}^{\pm.002}$	$\mathbf{0.810}^{\pm.002}$
0.50	$0.108^{\pm.005}$	$0.532^{\pm.003}$	$\mathbf{0.717}^{\pm.003}$	$\mathbf{0.810}^{\pm.002}$
0.75	$0.109^{\pm.004}$	$0.531^{\pm.003}$	$0.716^{\pm.002}$	$0.808^{\pm.002}$
1.00	$0.111^{\pm.006}$	$0.532^{\pm.003}$	$0.716^{\pm.002}$	$0.809^{\pm.002}$

Shared training. In both stages, each mixed batch contains HumanML3D and KIT-ML samples at a 3:1 ratio, approximately reflecting their relative dataset sizes. We compute the objectives separately for each dataset and aggregate them using weights $w_{\text{HumanML3D}} = 0.75$ and $w_{\text{KIT-ML}} = 0.25$, as defined in Sec. 3.5. During SiT training, each sample uses its corresponding dataset condition. We jointly drop the text and dataset conditions with probability 0.1, following Sec. 3.4.

Sampling. At inference, we initialize both latent streams with independent Gaussian noise and jointly integrate them using a 50-step Euler solver. We apply classifier-free guidance (Ho & Salimans, 2022) with scales of 5.5 for HumanML3D and 4.5 for KIT-ML. For the shared model, the conditional branch uses the target dataset embedding, while the unconditional branch uses the learned null embedding. The generated latents are decoded and recovered to the target dataset's native motion representation, following Sec. 3.2.

C More Ablations

We evaluate the decoded geometry weight λ_{pos} and root flow-matching weight λ_{root} on HumanML3D using dataset-specific SiT-B models. We vary one weight at a time while fixing the other at its default value of 0.25.

Decoded geometry weight. We examine the effect of supervision on decoded 3D joint positions by varying λ_{pos}. As shown in Table 8, removing this loss increases FID from 0.106 to 0.133 and decreases Top-1 R-Precision from 0.533 to 0.520. Among the tested values, $\lambda_{\text{pos}} = 0.25$ achieves the best mean FID and all three mean R-Precision scores. Larger weights still outperform the setting without this loss, but do not improve upon the default setting. These results support the benefit of decoded geometry supervision for downstream motion generation.

Root flow-matching weight. We vary λ_{root} to examine the balance between the image and root flow-matching objectives. Table 9 shows that performance varies only slightly over the tested range.

The default value $\lambda_{\text{root}} = 0.25$ achieves the lowest mean FID and highest mean Top-1 R-Precision, while matching the best reported Top-2 and Top-3 R-Precision. These results support our default choice of $\lambda_{\text{root}} = 0.25$.