Kimodo: Scaling Controllable Human Motion Generation

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, Sanja Fidler**

NVIDIA

* Co-First Authors

https://research.nvidia.com/labs/sil/projects/kimodo

Abstract

High-quality human motion data is becoming increasingly important for applications in robotics, simulation, and entertainment. Recent generative models offer a potential data source, enabling human motion synthesis through intuitive inputs like text prompts or kinematic constraints on poses. However, the small scale of public mocap datasets has limited the motion quality, control accuracy, and generalization of these models. In this work, we introduce Kimodo, an expressive and controllable kinematic motion diffusion model trained on 700 hours of optical motion capture data. Our model generates high-quality motions while being easily controlled through text and a comprehensive suite of kinematic constraints including full-body keyframes, sparse joint positions/rotations, 2D waypoints, and dense 2D paths. This is enabled through a carefully designed motion representation and two-stage denoiser architecture that decomposes root and body prediction to minimize motion artifacts while allowing for flexible constraint conditioning. Experiments on the large-scale mocap dataset justify key design decisions and analyze how the scaling of dataset size and model size affect performance.

1. Introduction

While human motion data has always been central to games and other media, recent advances in robotics and physical AI have increased the demand for such data. In robotics, human demonstrations allow humanoids to move realistically and complete complex tasks [39, 40, 20]. Digital twins and industrial simulations need dynamic humans to populate and interact with environments. And in established domains such as game development, the growing scope of interactive experiences necessitates plausible digital humans at scale.

Obtaining high-quality 3D human motion data is challenging, though. Traditional hand animation is often tedious and requires substantial domain expertise, while studio motion capture (mocap) is expensive and requires heavy instrumentation of both the actors and environments. Teleoperation has become a popular method to collect demonstrations directly with robots [49], but this process is slow and results in awkward, unnatural behaviors. While videos offer a rich source of human motions [15], recovering high-quality 3D motions from monocular videos remains a challenging research problem.

In this work, we contend that generative models offer an alternative motion data acquisition paradigm, which can easily synthesize high-quality human motion data with precise control. For such a model to be useful in practice, it should maintain the advantages of traditional motion acquisition approaches while increasing accessibility for novice animators (*e.g.*, roboticists). In particular, an ideal model should: (1) produce high-quality motions on par with optical mocap, (2) provide a versatile and directable interface akin to hand animation, but with more intuitive controls, and (3) be able to generate a diverse corpus of motions to support a large variety of applications.

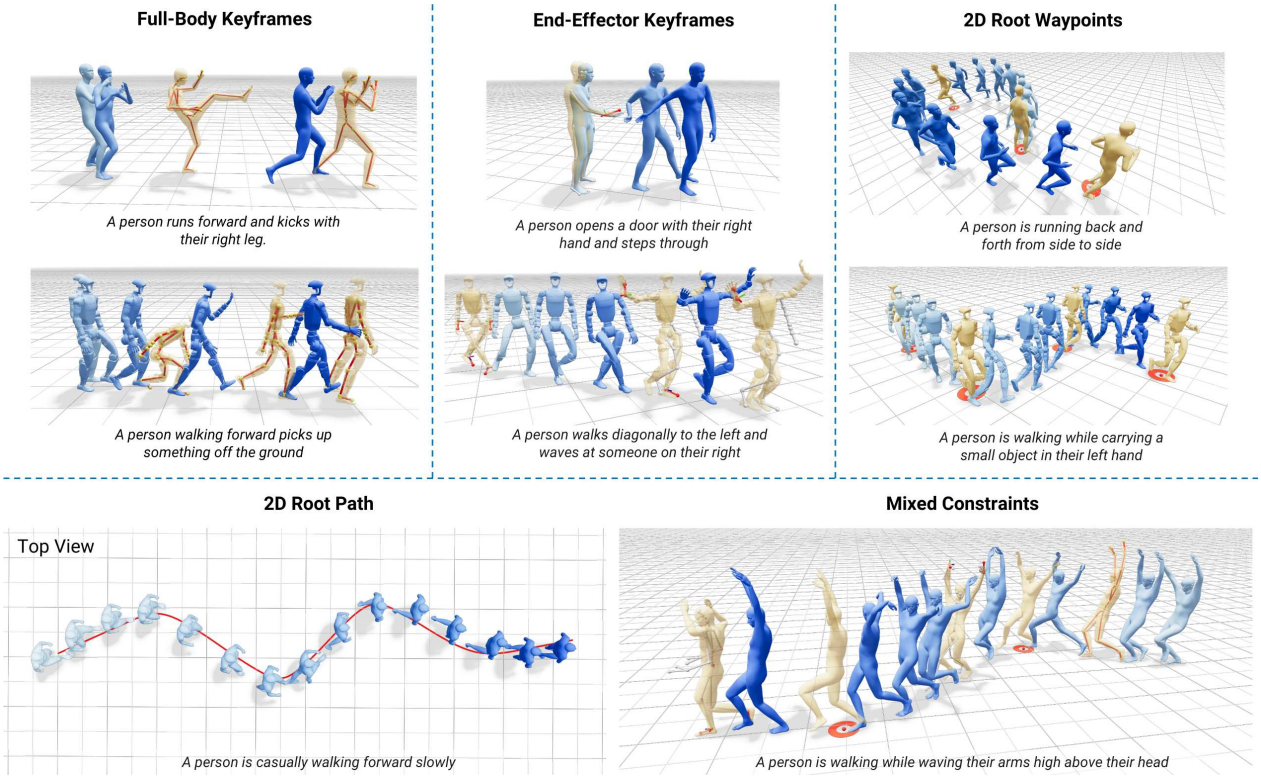

Figure 1: **Controllable Motion Generation.** Kimodo supports flexible and intuitive control for motion generation through text prompting combined with an extensive suite of kinematic constraints. By training on 700 hours of optical mocap data, the model achieves precise control accuracy for a large variety of behaviors. In each example, constrained joints are indicated with a red color, and generated poses at constrained frames are highlighted in yellow. Time progression is indicated by lighter to darker blue coloring.

Recently, rapid progress has been made in human motion generation. Modern generative models such as diffusion [42, 52], masked models [9], and tokenized transformers [50] have enabled intuitive motion generation by conditioning on text prompts. Some approaches also support *control* over generation through various kinematic constraints such as keyframe poses, 2D waypoints, and joint trajectories [9, 14, 47, 33]. These works show promising results and have inspired a wave of new research, but their motion quality, control accuracy, and generalization ability are limited by relatively small publicly available datasets [8, 28]. Furthermore, saturated benchmarks on these datasets make it difficult to differentiate which design decisions are critical to achieve effective modeling. Several works attempt to scale up human motion generation models and improve generalization by training on large volumes of motion data recovered from videos [38, 46, 19, 16], but these approaches tend to compromise motion quality due to the inherent inaccuracies in video reconstruction.

In this report, we introduce **Kimodo**, a **ki**nematic **mo**tion **d**iffusi**o**n model that trains at scale to enable intuitive authoring of high-quality motions through text prompting and an extensive set of kinematic constraints. Our model uses a carefully designed motion representation and two-stage diffusion architecture that decomposes root and body motion to accurately follow user specifications while minimizing common artifacts, such as floating and foot skating. In addition to text inputs, the model natively supports a suite of constraints on generated poses, including full-body keyframes, sparse joint positions/rotations, 2D waypoints, 2D path following, and foot contact patterns. A key component to effectively train Kimodo is the Bones Rigplay [2] dataset, a large studio mocap dataset containing 700 hours of production-quality human motion with corresponding text descriptions. This large dataset also enables a comprehensive evaluation of various design decisions on a wide range of behaviors and scenarios.

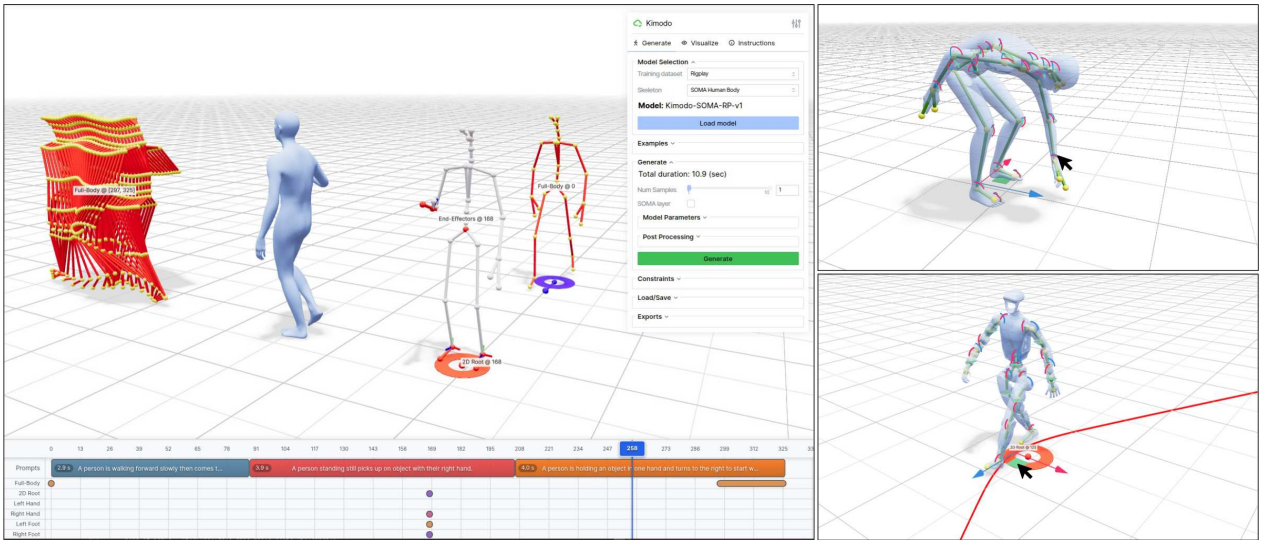

Figure 2: **Motion Authoring Demo.** (Left) Our authoring interface built with Viser [48] allows intuitive control over Kimodo for motion generation. The timeline panel allows users to specify text prompts and constraints at specific frames or intervals, which are displayed in the 3D viewer. The options panel on the right side of the interface controls various generation parameters. (Right) In editing mode, users have fine-grained control to pose and translate the character at constrained frames. Editing and generation can be done on either the SOMA body skeleton [32] or Unitree G1 robot.

To facilitate motion generation for robotics, simulation, games, and other applications, we have released model checkpoints for Kimodo trained on the SOMA body model [32] and the Unitree G1 robot [43]. Along with these models, we have included generation code and an interactive interface for motion authoring (see Fig. 2) to demonstrate the model's full capabilities. In the remainder of this report, Sec. 2 shows the key capabilities of our model, which enable practical authoring of human motions. Next, Sec. 3 describes the Bones Rigplay dataset and Sec. 4 details the core design of our two-stage transformer denoiser along with the training recipe. Sec. 5 positions our work in the context of prior systems, and lastly, Sec. 6 provides a comprehensive evaluation of key design decisions in our model and explores how scaling dataset size, model size, and batch size (GPUs) impacts model performance.

2. Key Results

Kimodo is designed for intuitive authoring of high-quality human motions. To showcase its capabilities, we developed an interactive demo with Viser [48], which allows a user to generate motions from a combination of text prompts and kinematic constraints (see Fig. 2). In this section, we detail the key features of the model within this demo interface, explore how scaling affects the model performance, and demonstrate downstream use of the model for humanoid robotics. The motions produced by our model are best viewed in the supplementary videos on the project webpage (https://research.nvidia.com/labs/sil/projects/kimodo/) or by running the demo in the released codebase (https://github.com/nv-tlabs/kimodo).

2.1. Motion Generation with Text Control

As shown in Fig. 3, motion generation with Kimodo can be easily controlled through intuitive text prompting. Our model, trained on the SOMA body skeleton [32] at 30 fps, enables generating realistic human motions for a variety of behaviors contained in the training data, including locomotion, everyday activities, dancing, actions and combat, and different motion styles. The model can handle potentially complex text descriptions that contain multiple actions performed in sequence ("A person walks forward *then* waves their arms.") or simultaneously ("A person walks forward *while* waving their arms."). Additionally, we retargeted the training

Figure 3: **Text-to-Motion Results.** (Top) Kimodo enables generating high-quality human motions for a variety of behaviors on the SOMA body skeleton. Time progression is indicated by lighter to darker blue coloring. (Middle) Motions can also be generated directly on the G1 robot to easily collect plausible demonstrations. (Bottom) The same frame is visualized from ten different generated motion samples for the same prompt, demonstrating the diversity of Kimodo outputs.

dataset to the Unitree G1 robot using the SOMA retargeter [23], allowing direct generation of kinematic robot demonstrations for important skills like recovering from stumbles and falls, and object interactions. The bottom row of Fig. 3 demonstrates the diversity of generated motions. The same frame from ten samples from the model is visualized, each with a different color. Though all samples use the same text prompt describing a put-down motion, the model generates plausible variations including one and two-handed, high and low, and variable timing.

Kimodo is trained to take a single prompt as input and expects the prompt to start with the subject, such as "A person...", "An old person...", or "A zombie...". However, our interactive authoring demo enables chaining together multiple prompts in sequence with plausible transitions. As shown in Fig. 4, using multiple prompts can be effective for performing a sequence of actions that the model may struggle with when specified as a single prompt. Please see Sec. 4.4 for technical details of multi-prompt generation.

Kimodo is designed as an offline motion authoring model, and is best suited for generating one or more motions in a batched fashion. On an NVIDIA RTX 3090, generating a motion from a single prompt takes anywhere from 2 to 5 sec, depending on the specified duration. The model is trained on a maximum of 10 sec sequences.

2.2. Kinematic Control with Pose Constraints

Precise control can be achieved through kinematic constraints on generated poses. As shown in Fig. 1, Kimodo supports a wide range of constraint types. *Full-body keyframes* constrain all the joint positions of the character at specific frames. *End-effector keyframes* specify the position and rotation of one or more hand or foot joints. For 2D root constraints, *waypoints* specify sparse targets on the ground for the character to hit, or a full dense *path* can be used to constrain an interval of motion. Constraints can be mixed arbitrarily to achieve desired motions.

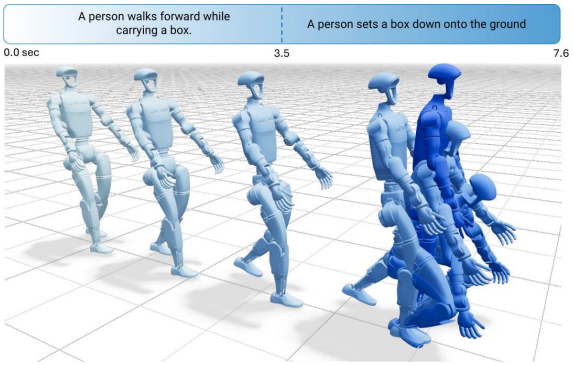

Figure 4: **Multi-Prompt Generation.** Longer motion sequences can be generated from multiple prompts with the demo's timeline interface. Motions are generated sequentially with constraints between them for continuity.

Kinematic control enables several useful motion authoring applications. Animators can use Kimodo to generate plausible transitions between existing mocap clips through in-betweening, with full-body constraints placed at the start and end of the generated sequence. For characters to realistically move about an environment, 2D paths or waypoints can be specified automatically using traditional animation tools like navigation meshes. For object interactions, users or automated planners can place constraints, such as hand positions, on objects to encourage Kimodo to generate pick and place motions that are plausible for a specific object size. To generate large scale motion datasets, constraints can be randomized to ensure diversity. For example, for creating a locomotion move set, a 2D root waypoint can be placed at varying angles from the starting location to synthesize locomotion in all different directions.

We evaluated Kimodo trained for the SOMA skeleton on a diverse test suite of constraint-conditioned motion generation cases (see Sec. 6.1 for details). On average, the model achieves 3.21 cm joint errors for full-body keyframes, 3.63 cm joint position errors for end-effector constraints, 6.88 deg joint rotation errors for end-effectors, and 3.63 cm root errors across waypoints and paths. Given such precise model outputs, light post-processing can be applied on generated motions to ensure they *exactly* hit user constraints without severely degrading motion quality. Note that, while we use the same evaluation protocol, these numbers are not comparable to those in Sec. 6, where models are trained on a different character skeleton at 20 fps.

2.3. Scaling Behavior

A key component of Kimodo's success is scaling along several axes. To demonstrate this, we evaluate the model while varying data size, model size, and batch size (number of GPUs). We summarize key results in this section, while full details are provided in Sec. 6.3.

Fig. 5 plots key metrics affected by each scaling axis. Increasing dataset size is particularly helpful for the model to see a larger variety of motions during training, which improves its ability to follow constraints and decreases errors across all constraint types. Increasing model size greatly improves text-following capabilities, as indicated by R-precision, along with motion quality, as indicated by FID. Finally, increasing the number of GPUs available for training, thereby increasing batch size, allows for further improvements particularly in text-following.

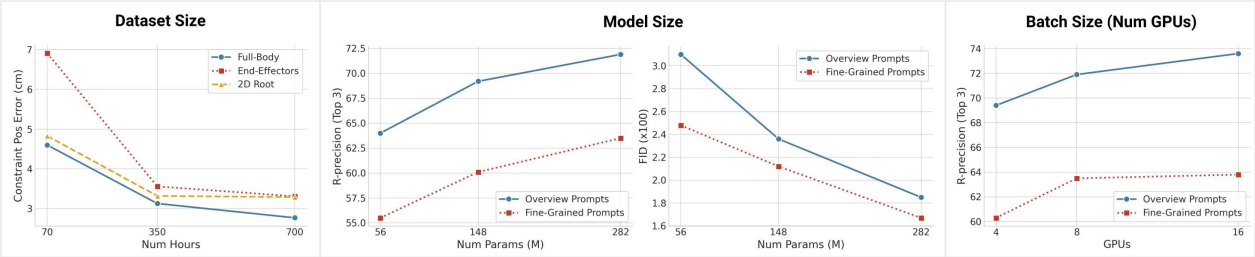

Figure 5: **Scaling Results.** Scaling dataset size, model size, and batch size improves controllability and motion quality. Increased dataset size results in greatly improved constraint following, while model size and batch size are particularly helpful for text following (R-precision) and motion quality (FID). See Tab. 2 for full results.

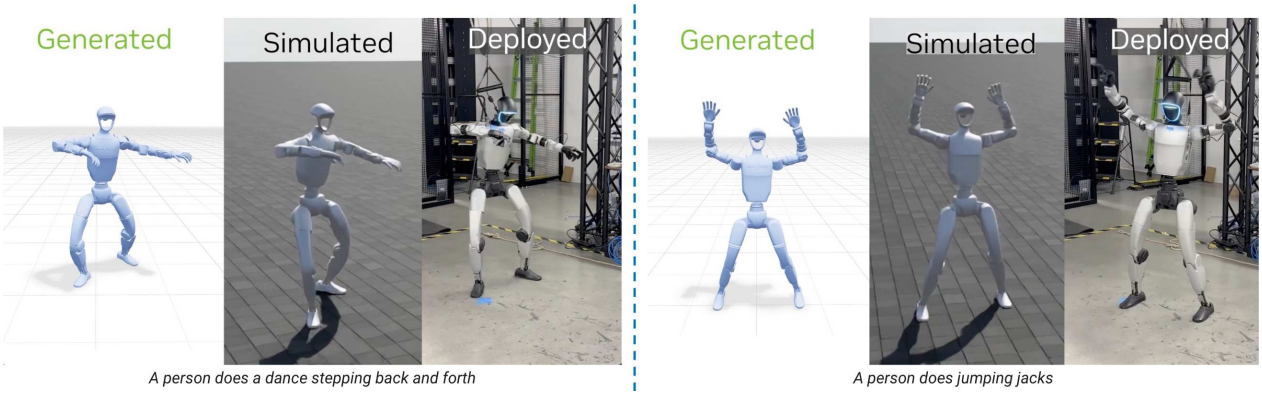

Figure 6: **Generating Robot Demonstrations.** In these results, Kimodo is used to generate demonstration data directly on the G1 robot, which is then tracked by a physics-based humanoid policy trained with ProtoMotions [41] and deployed to a real-world robot.

2.4. Application: Demonstration Data for Humanoid Robots

As shown in Fig. 6, Kimodo trained on G1 can directly generate demonstrations for robotics applications. Unlike traditional mocap or teleoperation, acquiring these motions takes a few seconds with a simple text prompt as input. Larger demonstration datasets can be created with batched generation and by employing constraints to ensure diversity of motion variations.

3. Large-Scale Motion Capture Dataset

For training and evaluation, we leverage Bones Rigplay [2], a large-scale optical motion capture dataset containing 700 hours of motions from 170 human subjects with a roughly equal number of male and female participants. The dataset contains thousands of unique actions covering a range of locomotion, gestures, everyday activities, common object interactions, videogame combat, dancing, athletics, and more. Many actions are also performed in different styles including tired, angry, happy, sad, scared, drunk, injured, stealthy, old, and childlike. Actions are performed by multiple subjects across multiple takes, providing a rich diversity of performers, semantics, and motion variability. As shown in Fig. 7, each clip in the dataset is labeled with an *overview* text description of the entire motion at a high level. Additionally, the clip is broken down into more *fine-grained* atomic action sub-clips, each containing a text description of the contained action.

The dataset contains body-only motions (i.e., no finger motions) that have been retargeted to a uniform-proportion 27-joint skeleton to be standardized across different performers. We use this "native" 27-joint skeleton for quantitative experiments presented in Sec. 6, however, we also retarget the dataset to other skeletons to train variations of our model, including SOMA [32], the Unitree G1 Robot [43], and SMPL-X [18, 24].

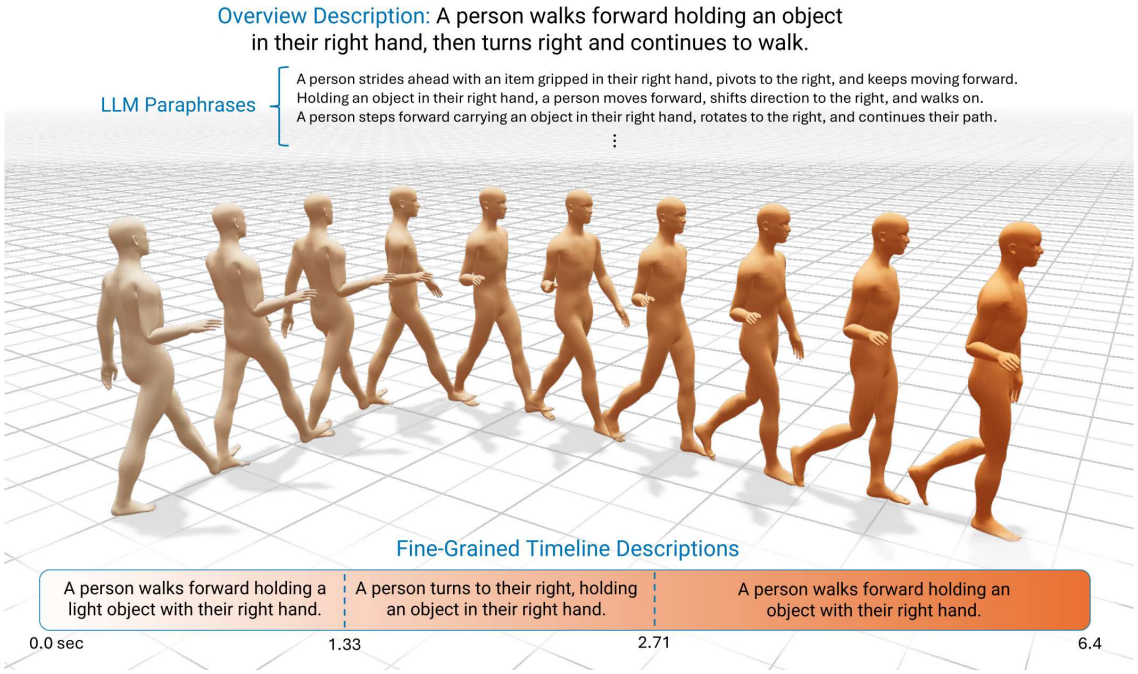

Figure 7: **Example Motion Data.** Each mocap sequence in our training dataset includes a high-level overview description, fine-grained descriptions of sub-clips containing atomic actions, and augmented LLM paraphrases.

Augmentations. For training the denoising model introduced in Sec. 4.2, we apply augmentations to both the text and motions in the dataset. For text, we use an LLM (Qwen3-32B [37]) to paraphrase the motion descriptions into a consistent prompt structure (always starting with "A [subject]...") with various levels of detail. To improve handling complex prompts that contain a composition of multiple actions, we also augment the motion in the dataset by stitching together random pairs of motion clips. To ensure natural transitions between the stitched clips, we use our diffusion model trained on the non-augmented dataset to generate short transition motions between the clips.

During training, we randomly sample from the available data and augmentation variations according to a pre-specified distribution. In particular, we train on a combination of full motion clips, single or combined action sub-clips, augmented stitched motion clips, original text descriptions, and augmented LLM paraphrases.

4. Method: Kimodo

Our model is designed to provide users with an intuitive and versatile interface for authoring high-quality motions. The core component of our framework is an explicit motion diffusion model [42, 52], which has been shown to effectively capture the complex distribution of text and motion. It also enables intuitive kinematic controls through simple conditioning mechanisms, such as direct imputation of pose features [33, 34].

Background. Explicit motion diffusion [42, 52] applies ideas from earlier image diffusion models [11], but operates directly on body pose features. Given a clean motion $\mathbf{x}_0$, the forward diffusion process is defined as a Gaussian process that adds noise to the poses until they are approximately $\mathcal{N}(\mathbf{0}, \mathbf{I})$. To generate a motion, the reverse process is used to iteratively denoise a sequence of noisy human poses $\mathbf{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$. To enable this reverse process, a denoiser $\mathcal{D}_\theta(\mathbf{x}_t, C, t)$, which takes a noisy motion $\mathbf{x}_t$, conditioning signals C, and the current denoising step $t \in \{1, \ldots, T\}$ as input, is trained to output a prediction of the clean motion $\hat{\mathbf{x}}_0$. The predicted clean motion is then re-noised to obtain an estimate of $\mathbf{x}_{t-1}$, which serves as input to the next denoising step in the reverse diffusion process. The conditioning signals may include text or other constraints on the motion.

The key design decisions to enable high-quality motion and control are the motion representation (Sec. 4.1)

and the denoiser architecture (Sec. 4.2). In the following sections, we describe these components of our system, along with the training recipe (Sec. 4.3) and generation process (Sec. 4.4).

4.1. Motion Representation

A motion sequence is represented as $\mathbf{x} = [\mathbf{x}^1, \mathbf{x}^2, \ldots, \mathbf{x}^N]$, where N is the number of frames. We assume the motion is standardized such that the root at the first frame is above the origin. For a skeleton with J joints, each pose in the motion is represented by a vector of features $[\mathbf{r}^p, \mathbf{r}^a, \mathbf{j}^p, \mathbf{j}^v, \mathbf{j}^a, \mathbf{f}]$ consisting of:

- $\mathbf{r}^p \in \mathbb{R}^3$: **smoothed global root position**. The root motion is computed by taking the pelvis position and heavily smoothing its 2D horizontal components (x, z), while keeping the y height unchanged.
- $\mathbf{r}^a \in \mathbb{R}^2$: **global root heading direction**. This is represented as $[\cos(\psi), \sin(\psi)]$ with the heading angle ψ. The heading angle is computed based on the projection of the cross product $\mathbf{e}_y \times \mathbf{v}_{\text{hips}}$ onto the xz (ground) plane, where $\mathbf{e}_y$ is the unit up vector and $\mathbf{v}_{\text{hips}}$ is the vector from left to right hip joints.
- $\mathbf{j}^p \in \mathbb{R}^{3J}$: **joint positions**. The 2D horizontal components (x, z) of each joint are represented relative to the smoothed root position, while the y height is global. Note that these joint positions are *not* rotated to be relative to the root heading direction.
- $\mathbf{j}^v \in \mathbb{R}^{3J}$: **global joint velocities**. The joint velocities are computed from the *global* joint positions, rather than the (partially) root-relative positions $\mathbf{j}^p$.
- $\mathbf{j}^a \in \mathbb{R}^{6J}$: **global joint angles** encoded using the 6D representation [55].
- $\mathbf{f} \in \{0, 1\}^4$: **foot contacts** boolean flags for the [left heel, left toe, right heel, right toe].

Considering the decomposed architecture introduced later, it is helpful to think of each pose as containing a global root component $\mathbf{r}^{\text{glob}} = [\mathbf{r}^p, \mathbf{r}^a]$ and a body component $\mathbf{b} = [\mathbf{j}^p, \mathbf{j}^v, \mathbf{j}^a, \mathbf{f}]$.

Discussion. This motion representation is carefully designed for motion quality and to be amenable for conditioning through direct imputation (*i.e.*, overwriting) of input pose features during denoising (see Sec. 4.2). First, our mostly global representation allows for sparse constraints on the root, joint positions, and joint rotations throughout a pose sequence. This is challenging for other common representations [8] that are purely local/velocity-based, since they require integration to determine global positions/rotations at a specific frame and are therefore better suited for temporally dense constraints [22]. Second, our joint position representation is not canonicalized with respect to the root heading in each frame. In prior work, joint positions are commonly canonicalized with respect to the root heading, but this design decision can lead to sudden changes in the local joint position representation due to abrupt flips in heading direction, such as during somersaults and cartwheels. These discontinuities can lead to training instabilities and compromise motion quality. Third, using a global joint rotation representation enables users to directly specify sparse joint rotation constraints in world space (through imputation), which is rarely supported in prior systems. While this is a common requirement for animation applications, it is difficult to achieve if joint rotations are represented relative to their parent in the kinematic chain (*e.g.*, in the SMPL body model [18]), since forward kinematics through the entire chain is required to recover a joint's global orientation. Lastly, compared to directly using the pelvis position projected to the ground as the root, our smoothed root position better emulates the smooth curves and straight lines that users are likely to specify as

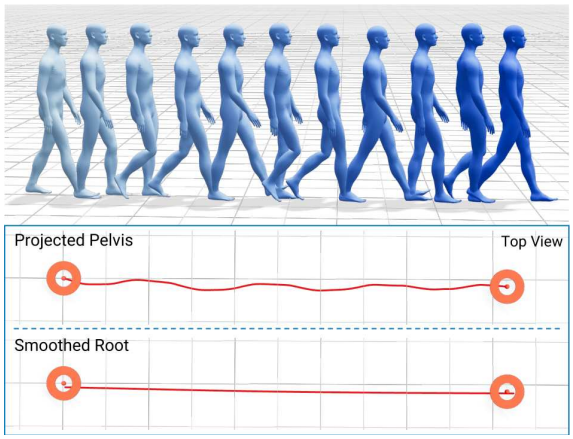

Figure 8: **Smoothed Root Representation.** For a simple walking motion, the projected pelvis path captures the sway of the hips, while the smoothed root is nearly a straight line. This smoothed trajectory offers a stable frame of reference for representing joint positions, and emulates paths drawn in practical animation tools.

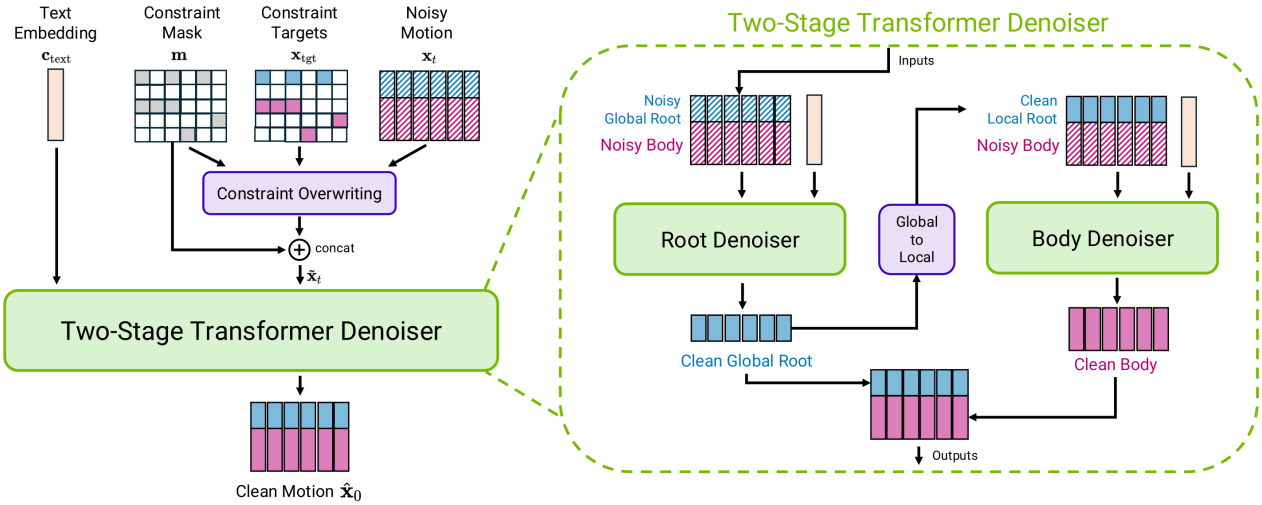

Figure 9: **Denoiser Architecture.** (Left) Kimodo predicts clean motion given a noisy motion, pose constraints, and a text embedding. Specified pose constraints directly overwrite the noisy motion before it is given to the denoiser. (Right) The two-stage denoiser decomposes root and body motion prediction. The root denoiser first predicts the global root motion, which is transformed into a local representation as input to the body denoiser. The final output of the denoising step is the concatenation of the outputs from the two stages.

2D waypoint and path constraints (Fig. 8). This smoothed root representation enables our model to generate motions that follow smooth target paths, while also providing flexibility for the pelvis joint to move naturally around the smoothed path.

4.2. Two-Stage Transformer Denoiser

Given a noisy motion $\mathbf{x}_t$ at denoising step t, the denoiser predicts the clean motion $\hat{\mathbf{x}}_0$ using a two-stage transformer architecture shown in Fig. 9. Denoising is optionally conditioned on a text prompt and/or constraints.

Inputs. The input noisy motion $\mathbf{x}_t$ to the transformer is treated as a sequence of pose tokens. Kinematic constraints are specified as partial pose features in the same representation as the input motion. In particular, the target pose features are specified by a (usually sparse) target motion $\mathbf{x}_{\text{tgt}}$ along with a binary control mask $\mathbf{m}$, which indicates which features are constrained. We impute (*i.e.*, overwrite) the noisy motion with the target pose features according to the control mask as $\tilde{\mathbf{x}}_t = \mathbf{m} \odot \mathbf{x}_{\text{tgt}} + (1 - \mathbf{m}) \odot \mathbf{x}_t$, where $\odot$ is the element-wise product [33]. Finally, we concatenate the control mask with the imputed motion along the feature dimension to produce the final input tokens $\mathbf{x}_{\text{in}} = [\tilde{\mathbf{x}}_t; \mathbf{m}]$. When no constraints are specified, the mask is simply all zeros and $\tilde{\mathbf{x}}_t = \mathbf{x}_t$.

The denoiser prediction is $\hat{\mathbf{x}}_0 = \mathcal{D}_\theta(\mathbf{x}_{\text{in}}, C, t)$ where $C = \{\mathbf{c}_{\text{text}}, \mathbf{c}_{\text{dir}}, \mathbf{c}_{\text{extra}}\}$ is the set of additional conditioning tokens. $\mathbf{c}_{\text{text}} \in \mathbb{R}^{4096}$ is the LLM2Vec embedding of the input text prompt [1], which we found to outperform common alternatives like CLIP [29] and T5 [30] in early experiments. $\mathbf{c}_{\text{dir}} \in \mathbb{R}^2$ is the desired heading direction of the first frame. Lastly, $\mathbf{c}_{\text{extra}} \in \mathbb{R}^{P \times 4096}$ contains P extra all-zero tokens. While these extra tokens cannot exactly be considered "register" tokens [4], since they are not learned, they achieve a similar effect of enhancing the representational capacity of the model as shown in our experiments (Sec. 6). In practice, we use $P = 49$. All pose and conditioning tokens are embedded to the same dimensionality and added to a sinusoidal positional encoding before going into the transformer.

Architecture. As shown in Fig. 9, the denoiser is decomposed into two transformer encoders: one for predicting root motion and the other for body motion. The root denoiser first predicts the global root motion $\hat{\mathbf{r}}_0^{\text{glob}}$. Despite only predicting the root, this denoiser is conditioned on the full noisy motion $\mathbf{x}_{\text{in}}$, such that the output can be

coordinated with the body motion and any relevant constraints. After the first stage, the global root prediction is transformed into a local representation $\hat{\mathbf{r}}_0^{\text{local}}$, concatenated with the body features from $\mathbf{x}_{\text{in}}$, and inputted to the second transformer to predict the body motion $\hat{\mathbf{b}}_0$. The final output of the denoiser is the concatenated global root and body predictions $\hat{\mathbf{x}}_0 = [\hat{\mathbf{r}}_0^{\text{glob}}; \hat{\mathbf{b}}_0]$. In practice, both transformers use the same architecture with 16 layers, 8 heads, and a latent size of 1024, totaling 282 M learnable parameters in our full model.

While the global root representation is advantageous when modeling root motion, we found that using a *local* root representation is more effective when conditioning the second stage of the model to denoise body motion. Inspired by Guo *et al*. [8], we define the local root pose as $\mathbf{r}^{\text{local}} = [\dot{\mathbf{r}}^a, \dot{\mathbf{r}}_{xz}^p, \mathbf{r}_y^p]$ where $\dot{\mathbf{r}}^a \in \mathbb{R}$ is the angular velocity of the heading, $\dot{\mathbf{r}}_{xz}^p \in \mathbb{R}^2$ is the planar translation velocity of the root, and $\mathbf{r}_y^p \in \mathbb{R}$ is the absolute height. Converting from the global to local root representation is straightforward using finite differences to estimate velocities.

Discussion. The two-stage denoiser design is key to maximizing motion quality and control accuracy together. The global root representation used in the first stage enables conditioning on sparse global root constraints through direct imputation, while the second stage benefits from being conditioned on a more invariant local root representation. Moreover, we hypothesize that body motion prediction is an easier problem when conditioned on the root motion. This is supported experimentally in Sec. 6, where ablations of the two-stage model and the local root representation result in worse performance. Note that unlike prior two-stage approaches, our two-stage model is *interleaved* since the root and body predictions are made at every denoising step. Prior work performs the root denoising process in full before generating the body motion [14, 12], while our model enables root and body corrections throughout denoising to attain better alignment in the end.

The need for the initial heading token $\mathbf{c}_{\text{dir}}$ as input stems from our global representation of joint rotations. To generalize more effectively with this global representation, we speculate that it is helpful for the model to see a wide distribution of rotations during training. Therefore, we choose not to canonicalize motions relative to the heading at the first frame, as is common practice, and instead apply a random first-frame heading augmentation to motions during training (see Sec. 4.3). This means the model can generate motion that starts at any arbitrary heading, which we aim to control at test time with $\mathbf{c}_{\text{dir}}$. The choice not to canonicalize the motion and use $\mathbf{c}_{\text{dir}}$ is also practically convenient at test time. When the model is conditioned on global constraints in the scene or the user wants to generate motion from a sequence of multiple prompts (*e.g.*, from an editing timeline), there is no need to apply a canonicalizing transform before generation and subsequently an inverse transform afterwards, since we can simply specify the desired initial heading with $\mathbf{c}_{\text{dir}}$.

4.3. Training

The denoiser is trained according to the DDPM [11] framework using a modified version of the simplified loss function. At each training iteration, a ground truth motion $\mathbf{x}_0$ is sampled from the dataset and a diffusion timestep $t \sim \mathcal{U}\{1, \dots, T\}$ and Gaussian noise $\epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ are used to noise the motion to $\mathbf{x}_t = q(\mathbf{x}_t \mid \mathbf{x}_0, t)$. After obtaining the denoiser prediction $\hat{\mathbf{x}}_0$, the loss is computed on each component of the motion representation:

$$\begin{aligned}\mathcal{L} = \gamma_1 ||\hat{\mathbf{r}}_0^p - \mathbf{r}_0^p||_1 + \gamma_2 ||\hat{\mathbf{r}}_0^a - \mathbf{r}_0^a||_1 + \gamma_3 ||\hat{\mathbf{j}}_0^p - \mathbf{j}_0^p||_1 + \gamma_4 ||\hat{\mathbf{j}}_0^v - \mathbf{j}_0^v||_1 \\ + \gamma_5 ||\hat{\mathbf{j}}_0^a - \mathbf{j}_0^a||_1 + \gamma_6 ||\hat{\mathbf{f}}_0 - \mathbf{f}_0||_1 + \gamma_7 ||\text{FK}(\hat{\mathbf{j}}_0^a) - \mathbf{j}_0^p||_1. \end{aligned} \tag{1}$$

where $||\cdot||_1$ is a smooth L1 loss that uses an L2 term when the loss is small and an L1 term otherwise [7]. $\text{FK}(\cdot)$ is the forward kinematics function, which computes joint positions from joint rotations. In practice, the losses are weighted as $\gamma_1 = \gamma_3 = \gamma_5 = 10.0$, $\gamma_2 = 2.0$, $\gamma_4 = 3.0$, $\gamma_6 = 4.0$, and $\gamma_7 = 5.0$. Training uses variable length sequences within each batch, so loss functions are masked accordingly. We use $T = 1000$ diffusion steps for training.

To improve training stability, we use the Adam-atan2 optimizer [5] with a learning rate of $2e{-}5$. Ground truth motions are cropped to a maximum length of 10 sec and translated such that the root position is above the

origin at the first frame. The heading direction at the first frame is randomized to ensure the model is robust to variations in global rotations. Our best model configuration is trained with a batch size of 2048 across 16 NVIDIA A100 (SXM4-80GB) GPUs and generates motions at 30 fps.

Training Curriculum. The denoiser is trained in two phases. For the first 500k steps (phase 1), the model is trained purely on the text-to-motion task with no constraints given as input. For the second 500k steps (phase 2), the model is trained on a mix of text and kinematic constraints. During phase 2, kinematic constraints are randomly sampled from a set of pre-defined constraint patterns designed to enable specific functionality at test time. These include: full-body joint positions at sparse keyframes, random subsets of hands and feet positions/rotations at sparse keyframes, 2D root position/heading at sparse keyframes, 2D root position/heading on dense paths, and foot contact configuration at sparse keyframes. Two constraint patterns are mixed together 25% of the time, and 10% of the time no constraints are used (leaving only text input). During phase 2, the maximum number of keyframes sampled for sparse constraints increases linearly from 1 to 20, and sampling is biased towards fewer keyframes to reflect common real-world use cases. Dropout with a rate of 0.1 is used during phase 1, but is removed for phase 2 to avoid dropping out conditioning constraints that are directly overwritten to the noisy motion input. During both phases, the text input is dropped 10% of the time to enable classifier-free guidance at test time [10]. Exponential Moving Average (EMA) is applied every 10 steps throughout training with a decay of 0.995 to maintain an average of the denoiser parameters, which is then used at test time.

4.4. Motion Generation

After training, motions are generated using the DDIM [36] inference process with, by default, 100 denoising steps. We leverage a classifier-free guidance approach that decomposes text and constraint conditioning to allow control over each one individually. In particular, the model output at each denoising step is computed as $\hat{\mathbf{x}}_0 = \mathcal{D}_\varnothing + w_{\text{text}}(\mathcal{D}_{\text{text}} - \mathcal{D}_\varnothing) + w_{\text{constr}}(\mathcal{D}_{\text{constr}} - \mathcal{D}_\varnothing)$ where $\mathcal{D}_\varnothing$ is the model output using no text or constraint conditioning, $\mathcal{D}_{\text{text}}$ uses only text conditioning (no constraints), and $\mathcal{D}_{\text{constr}}$ uses only constraint conditioning (no text). By default, we use $w_{\text{text}} = 2$ and $w_{\text{constr}} = 2$, but a user can adjust each to vary the influence of text and constraint conditioning on the model output. Several prior works make use of gradient-based guidance to further improve constraint following at test-time [31, 14], but we found that since our model is already directly conditioned on the constraints, adding gradient-based guidance gave minimal improvement, substantially increased generation time, and was generally unstable and difficult to tune.

Multi-Prompt Sequencing. While Kimodo is trained to take one text prompt as input, it is practically desirable to generate motions for a sequence of prompts (see Sec. 2.1). We achieve this by generating the motion for each prompt in sequence and adding constraints to maintain plausible transitions. After generating motion for the first prompt, the subsequent prompt is generated with an overlap to the first prompt where several full-body keyframe constraints are added to encourage consistent joint positions and accelerations with the previously generated motion. To ensure a smooth transition, we blend the constrained frames that are shared by the first and second prompt after generation.

Motion Post-Processing. In practice, post-processing can be performed on the model outputs to improve the generated motion. Simple foot locking and IK can clean up any undesirable foot skate using the foot contact classification directly from the model output. It is also helpful to perform a short optimization on the output motion to ensure it *exactly* hits the kinematic constraints, which is challenging for the model to achieve. We use these post-processing approaches in our released demo and codebase, but not for the experiments in Sec. 6 to ensure a fair comparison between methods.

5. Related Work

Our approach is closely related to previous works in generative human motion modeling, while improving upon them in key ways. Besides explicit motion diffusion [42, 52], several alternative approaches have proven successful at text-conditioned human motion generation. Latent motion diffusion [3] denoises motion in a learned latent space rather than the explicit pose space, thereby improving efficiency. In a similar spirit, MMM [26] and MoMask [9] learn a discretized latent space via a VQ-VAE [44], then train a model to generate a sequence of latents through a progressive masked prediction procedure. A different class of methods treat discretized latents as tokens and autoregressively predict them in sequence, similar to large language models [50, 13]. Such latent approaches focus primarily on text control. Those that do handle kinematic constraints require latent test-time optimization to achieve high accuracy [45, 27].

Our choice of explicit motion diffusion is motivated by the ease of controllability on the pose features. OmniControl [47] demonstrated dense and sparse positional control with a ControlNet [51] fine-tuned on top of a base motion diffusion model. GMD [14] uses a combination of imputation and test-time guidance to handle potentially sparse positional constraints. Learning an RL policy on top of a diffusion model has also been explored for kinematic controls [35, 54]. Our method is most related to methods that use imputation to condition generation by directly overwriting constraints in the input motion [34, 12]. CondMDI uses the same imputation approach that we do with a concatenated mask as input to the denoiser [33].

Our method, Kimodo, supports a suite of kinematic controls that is more extensive than prior works. Through direct imputation during training, our denoiser can handle both sparse and dense constraints on positions and joint rotations. This is achieved without additional ControlNet fine-tuning, test-time guidance/optimization, or RL. Moreover, our smoothed root representation lends itself to root constraints commonly found in motion editing applications. While prior work has leveraged a global joint position representation to handle sparse constraints [14, 33, 12, 22], we also adopt the global representation for joint rotations, enabling sparse rotation control. Additionally, previous approaches that use a two-stage model [14, 12] train each stage independently, while our interleaved two-stage denoiser trains end-to-end.

A major challenge in learning effective motion generation models is the relative lack of large-scale datasets, with the most common benchmark HumanML3D [8] containing only 30 hours of motion. Several works try to address this through increasingly larger datasets. Motion-X was an early work to collect a dataset with a majority of motions recovered from online video sources, totaling 144 hours of motion [16]. Datasets have continued to grow through a combination of mocap and videos from several sources [19, 17, 46, 53, 38], with the recent MotionMillion dataset containing 2000 hours of motions [6]. While such diverse data is exciting to study the scaling properties of motion generation, these datasets rely heavily on motions reconstructed from monocular videos and/or mocap from a variety of sources with different framerates and skeletons. Additionally, text descriptions are often labeled through automatic LLM-driven approaches. As a result, the average motion and text quality is considerably lower than the 700 hours of optical mocap data and human-labeled text annotations that Kimodo is trained on. An exciting future direction is how to leverage large-scale video data without compromising the motion quality that can be learned from mocap.

6. Quantitative Evaluation

In this section, we perform a detailed experimental analysis of key design decisions of our approach (Sec. 6.2) and scaling behavior in terms of data size, model size, and batch size (Sec. 6.3).

Most prior works evaluate on the public HumanML3D benchmark [8], which is relatively small scale and is becoming increasingly saturated, as indicated by better-than-ground-truth metrics reported for recent methods. For our experiments, we choose to exclusively use the large-scale, high-quality Bones Rigplay [2] mocap dataset described in Sec. 3, as it provides a robust benchmark for highlighting differences between ablations and variations of our method.

6.1. Experiment Setting

Dataset. We hold out 10% of the motions in Rigplay from training for evaluation. Splits are determined based on unique behaviors (*i.e.*, action types) in the dataset, such that the test set contains only novel behaviors that were not seen during training. For results reported here, we evaluate on a subset containing about 5k motions that cover all distinct behaviors in the test set. For these experiments, models are trained on the native 27-joint skeleton version of the dataset.

Test Cases. Models are evaluated in three different settings. (1) To evaluate the text-to-motion task with no kinematic constraints, we prompt the model with the high-level *Overview Prompt* for each motion (see Sec. 3). (2) We also evaluate unconstrained text-to-motion with a *Fine-Grained Prompt* that tends to describe a shorter motion containing an atomic action. (3) To evaluate combined text+constraint conditioning, we aggregate results from a test suite containing diverse variations of kinematic constraints sampled from the ground truth motion. Constraints in this suite reflect real-world use cases including sparse full-body joint position keyframes, in-betweening with blocks of keyframes at the start and/or end of a motion, sparse keyframes on position/rotation of end-effectors (hands and feet), sparse 2D waypoint keyframes for the root, dense 2D root paths, and mixing multiple constraints together. Constrained generation is evaluated at motion lengths from 3 to 9 sec, both with and without text-conditioning. For a subset of test cases, we apply random perturbations on the global translation and heading of constraints to evaluate generalization.

Evaluation Metrics. We adapt common metrics from prior work [8]. To evaluate text-following, we report *Top-3 R-precision (R@3)* using a TMR embedding model [25] trained on the full Rigplay dataset, including both train and test split. Notably, retrieval with TMR is performed over the entire test set rather than within small batches of 32 as is common in prior work, significantly increasing the challenge. For motion quality, we measure *FID* using the same TMR model. We also report a *Foot Skate* metric that computes the mean velocity of feet joints in the output motion for frames where the model predicts the joint should be in static contact with the ground. This foot skate metric depends on the *Foot Contact Classification Accuracy*, which is generally very good across all methods, but is also reported for completeness.

For constraint-conditioned generation, the *average distance error* between the input constraint and the generated motion at constrained frames is reported. Errors are split into full-body positions, end-effector position/rotation, and 2D root position and averaged over all test cases. Because our model uses a smoothed root representation, the model can generate motion where the smoothed root matches the constraint, but the projected pelvis of the character still deviates from the smooth path/waypoint constraint. As discussed in Sec. 4.1, this flexibility is useful for maintaining natural motion when following root constraints, however, significant deviation of the pelvis from the constraint is undesirable as the character can appear to drift from the constraint path. To evaluate the extent of this deviation, we report the *95th percentile of the 2D position error* between the smoothed root constraint and projected pelvis. This value indicates the maximum that the pelvis will stray from smoothed root constraints for the vast majority of generated motions; closest to ground truth is optimal.

Model Setting. Unless otherwise noted, the models presented in these experiments are trained at 20 fps using 8 GPUs (resulting in a batch size of 1024 motions) and on data with all the augmentations described in Sec. 3. This differs slightly from our best models demonstrated in Sec. 2, which are 30 fps and trained with 16 GPUs, but the trends are still informative.

6.2. Ablation Study

In Tab. 1, we compare the full model architecture and training curriculum to several strong baselines to justify key design decisions. Metrics computed on the ground truth data are shown for reference.

Two-Stage Denoiser. We first compare our decomposed two-stage denoiser design to a *One-Stage* baseline that uses a single transformer to simultaneously denoise root and body motion. To ensure a fair comparison,

	Text-Following Evaluation								Constrained Evaluation				
	Overview Prompt Test Set				Fine-Grained Prompt Test Set				Full-Body	End-Effector Joints		2D Root	2D Pelvis
Method	**R@3**↑	**FID** ↓	**Skate** (cm/s)↓	**Contact**↑	**R@3**↑	**FID**↓	**Skate** (cm/s)↓	**Contact**↑	**Pos** (cm)↓	**Pos** (cm)↓	**Rot** (deg)↓	**Pos** (cm)↓	**Pos@95%** (cm)
Ground Truth	75.6	0.0	2.21	1.00	79.4	0.00	2.23	1.00	-	-	-	-	6.3
Full Model (Ours)	**71.9**	1.85	**3.87**	**0.98**	63.5	1.67	**3.88**	**0.98**	2.67	3.09	4.18	2.90	9.7
One-Stage Arch	71.5	**1.65**	7.59	0.94	63.5	**1.51**	6.80	0.95	8.37	10.19	5.19	7.74	21.1
Second Stage Global	70.3	1.87	4.17	**0.98**	63.2	1.66	4.07	**0.98**	2.97	3.39	5.67	3.25	10.2
No Smoothed Root	71.6	1.75	4.39	0.97	**64.0**	1.55	4.27	**0.98**	2.68	3.19	**3.93**	3.21	7.9
No Extra Tokens	70.9	1.95	4.28	0.97	61.6	1.77	4.17	**0.98**	**2.40**	**2.59**	5.55	**2.85**	9.18
No Train Curriculum	71.3	1.84	3.92	**0.98**	63.2	1.66	3.91	**0.98**	5.80	6.59	4.34	5.71	15.5

Table 1: **Ablation Study.** Evaluation of text and constraint-conditioned motion generation on the Rigplay test set. The full model is compared to various baselines to justify key design decisions, including the two-stage denoiser, smoothed root representation, and dual-phase training curriculum. All models are trained using a medium batch size (8 GPU) at 20 fps. FID is multiplied $\times 100$ for readability.

we increase the number of layers and latent size of the baseline such that the number of learnable parameters is similar to the two-stage model. While the text-following capabilities of this baseline are about the same as the full model, we note a substantial increase in foot skating, indicating that generating body motion conditioned on root motion is indeed easier than generating both simultaneously. The one-stage baseline also causes a substantial increase in constraint errors.

The *Second Stage Global* baseline uses the global root representation in the second (body) stage of the denoiser instead of converting the input root motion to a local representation. This tends to cause an increase in foot skating, likely due to the lack of invariance of the global root representation.

Smoothed Root Representation. The next baseline directly uses the pelvis joint projected to the ground as the root instead of the smoothed representation used in the full model, which causes notably increased foot skate. Without the smoothed root, body joint positions are represented with respect to the pelvis, which may be more difficult to learn due to the high-frequency motions of the pelvis. Despite this foot skate, constraint accuracy in on par with the smoothed root representation. As expected, the 95th percentile of pelvis error is lower than the full model, since the baseline is directly trained to match the pelvis to the root constraint. However, in practice this can cause qualitatively unnatural motions when constraining motions with straight lines or smoothed curves as is common in animation applications. For example, asking the baseline to generate a walking motion along a straight line results in a motion with a stealthy/sneaking style or old person style, since these minimize lateral pelvis movements and therefore deviate less from the straight line constraint.

Extra "Register" Tokens. The *No Extra Tokens* baseline removes the extra register tokens, leaving only the text embedding and heading token as additional conditioning for the model. This tends to reduce performance particularly for text-following and motion quality, indicating that the added tokens increase the representational capacity of the model.

Dual-Phase Training Curriculum. The *No Train Curriculum* baseline directly trains the model for text+constraint conditioning from scratch, rather than training in two phases, as described in Sec. 4.3. For a fair comparison, the baseline is trained for 1 million steps, the same as the total steps of phased training. During training, the baseline receives text-only or text+constraint inputs with equal probability and does not use dropout. This baseline reaches comparable text-following accuracy and motion quality as using phased training, but sees an increase in constraint errors. In two-phase training, phase 1 is dedicated to pre-training on text-to-motion so that phase 2 can be dedicated to constraint-following, but non-phased training must balance text and constraint learning for the entire duration of training. While it is possible there is a perfect balance between text-only and text+constraint input sampling that will result in competitive performance across the board, we find the phased training works well without additional hyperparameter tuning.

	Text-Following Evaluation								Constrained Evaluation				
	Overview Prompt Test Set				Fine-Grained Prompt Test Set				Full-Body	End-Effector Joints		2D Root	2D Pelvis
Method	**R@3**↑	**FID** ↓	**Skate** (cm/s)↓	**Contact**↑	**R@3**↑	**FID**↓	**Skate** (cm/s)↓	**Contact**↑	**Pos** (cm)↓	**Pos** (cm)↓	**Rot** (deg)↓	**Pos** (cm)↓	**Pos@95%** (cm)
Ground Truth	75.6	0.00	2.21	1.00	79.4	0.00	2.23	1.00	-	-	-	-	6.3
Full Dataset	**71.5**	1.84	**4.23**	**0.97**	63.0	1.07	**4.15**	**0.98**	**2.77**	**3.31**	**5.36**	**3.29**	10.7
50% Dataset	70.8	**1.81**	4.43	**0.97**	**63.4**	**1.06**	4.27	0.97	3.13	3.56	6.29	3.32	10.7
10% Dataset	71.0	2.07	5.28	0.96	62.4	1.41	5.12	0.97	4.60	6.91	10.03	4.83	15.5
L Model (282 M)	**71.9**	**1.85**	**3.87**	**0.98**	**63.5**	**1.67**	**3.88**	**0.98**	**2.67**	**3.09**	**4.18**	**2.90**	9.7
M Model (148 M)	69.2	2.36	4.45	0.97	60.1	2.12	4.31	0.97	3.26	3.72	4.70	3.34	10.0
S Model (56 M)	64.0	3.10	4.53	0.97	55.5	2.48	4.47	0.97	3.56	3.98	11.27	3.49	10.8
L Batches (16 GPU)	**73.6**	**1.61**	3.97	**0.98**	**63.8**	**1.52**	**3.87**	**0.98**	**2.33**	**2.71**	**4.09**	**2.35**	8.5
M Batches (8 GPU)	71.9	1.85	**3.87**	**0.98**	63.5	1.67	3.88	**0.98**	2.67	3.09	4.18	2.90	9.7
S Batches (4 GPU)	69.4	2.01	4.45	0.97	60.3	1.98	4.27	**0.98**	2.97	3.68	5.61	3.42	10.1

Table 2: **Scaling Analysis.** Evaluation of text and constraint-conditioned motion generation on the Rigplay test set. (Top) Increasing the amount of training data improves motion quality and constraint accuracy due to increased diversity. (Middle) Increased model size improves performance on all metrics. (Bottom) Increasing batch size by using more GPUs generally improves performance across the board.

6.3. Scaling Analysis

Tab. 2 evaluates how scaling affects model performance across three different axes.

Data Size. The top part of the table compares using the full training dataset to training on a subset of 50% and 10% of training motions. Subsets are strategically sampled to include all unique behaviors (action types) from the full dataset, but with the number of performances for each behavior reduced to the desired fraction. As a result, this experiment primarily evaluates how much repeated performances of the same behavior across many actors influences motion generation ability. For this experiment only, we do not use the augmented stitched motions described in Sec. 3.

Comparing the three dataset sizes, we see that foot skate and constraint accuracy monotonically improve with more available training data, with significantly decreased performance when using 10% of the data (*i.e.*, the same order of magnitude as popular AMASS [21] and HumanML3D datasets). Curiously, R-precision and FID are not significantly affected by dataset size, which we believe is an artifact of how we subsample the training data. Since the baselines are trained on the same unique behaviors as the full training set, they should still generate reasonable results for test prompts. Therefore, retrieval with TMR is still successful and R-precision is not greatly affected. Similarly, the generated motion distributions for the test set will be similar even if using only 10% of the data, especially after embedding with TMR and being fit with a unimodal Gaussian, as is done in the FID metric. What R-precision and FID do not capture are the fine-grained differences in motion distribution and subtle but important variations in motion, which manifests as reduced constraint following accuracy due to training on less diverse data.

Model Size. The middle part of Tab. 2 compares large (L), medium (M), and small (S) variants of our model in terms of learnable parameters. Our full best model uses the large size. The medium variant uses 8 layers in the transformer encoders instead of 16, while the small variant additionally decreases the latent size to 512 from 1024. We see that increasing model size improves performance across all metrics. While continuing to increase model size to the order of 500M or 1B parameters could potentially further improve performance, we found training stability becomes a bigger challenge and we speculate that without more data there will be diminishing returns.

Batch Size. The bottom part of the table evaluates how increasing batch size affects performance. In practice, a larger batch size means using more GPUs so we compare small (S), medium (M), and large (L) batch sizes using 4, 8, and 16 GPUs, respectively. This corresponds to 512, 1024, and 2048 batch sizes. As shown in the table, using more GPUs generally improves performance across the board as the gradient estimate during optimization becomes more accurate. We use the large batch size with 16 GPUs to train our best model.

7. Conclusion

We have introduced Kimodo, a kinematic motion diffusion model trained on a large-scale motion capture dataset that generates high-quality human motions and can be controlled through text and a variety of kinematic constraints. Our model enables easy authoring of motions using intuitive interfaces as shown in Sec. 2, and can be applied directly to generating robot motion after retargeting the training dataset. After training entirely on optical mocap data, our model gives a strong foundation for further scaling up of human motion generation.

Future Challenges. Looking forward, a promising direction is to further scale up the model with motions reconstructed from internet videos or generated videos. An important challenge here will be how to combine clean and noisy data sources without compromising output motion quality from the model. While Kimodo is designed specifically for "offline" motion authoring and can take several seconds to generate a motion, applications such as robotics and digital twin simulation require a runtime model that dynamically controls humanoids and reacts to changing environments. To this end, an interesting avenue is moving diffusion to a learned latent space and reformulating motion generation to be an autoregressive problem. Finally, scene and object interactions are crucial to making motion generation models truly practical for most applications. Gathering data for this problem becomes even more challenging, and will require creative solutions.

References

[1] Parishad BehnamGhader, Vaibhav Adlakha, Marius Mosbach, Dzmitry Bahdanau, Nicolas Chapados, and Siva Reddy. LLM2Vec: Large language models are secretly powerful text encoders. In *First Conference on Language Modeling*, 2024. URL https://openreview.net/forum?id=IW1PR7vEBf. 9

[2] Bones Studio. Ai datasets for machine learning and motion capture. https://bones.studio/ai-datasets/, 2026. 2, 6, 12

[3] 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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18000–18010, 2023. 12

[4] Timothée Darcet, Maxime Oquab, Julien Mairal, and Piotr Bojanowski. Vision transformers need registers. *International Conference on Learning Representations*, 2024. 9

[5] Katie Everett, Lechao Xiao, Mitchell Wortsman, Alexander A Alemi, Roman Novak, Peter J Liu, Izzeddin Gur, Jascha Sohl-Dickstein, Leslie Pack Kaelbling, Jaehoon Lee, et al. Scaling exponents across parameterizations and optimizers. *International Conference on Machine Learning*, 2024. 10

[6] Ke Fan, Shunlin Lu, Minyue Dai, Runyi Yu, Lixing Xiao, Zhiyang Dou, Junting Dong, Lizhuang Ma, and Jingbo Wang. Go to zero: Towards zero-shot motion generation with million-scale data. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2025. URL https://arxiv.org/abs/2507.07095. 12

[7] Ross Girshick. Fast r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 1440–1448, 2015. 10

[8] 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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5152–5161, June 2022. 2, 8, 10, 12, 13

[9] Chuan Guo, Yuxuan Mu, Muhammad Gohar Javed, Sen Wang, and Li Cheng. Momask: Generative masked modeling of 3d human motions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1900–1910, 2024. 2, 12

[10] Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598*, 2022. 11

[11] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020. 7, 10

[12] Inwoo Hwang, Jinseok Bae, Donggeun Lim, and Young Min Kim. Motion synthesis with sparse and flexible keyjoint control. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2025. 10, 12

[13] Biao Jiang, Xin Chen, Wen Liu, Jingyi Yu, Gang Yu, and Tao Chen. Motiongpt: Human motion as a foreign language. *Advances in Neural Information Processing Systems*, 36, 2024. 12

[14] Korrawe Karunratanakul, Konpat Preechakul, Supasorn Suwajanakorn, and Siyu Tang. Guided motion diffusion for controllable human motion synthesis. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2151–2162, 2023. 2, 10, 11, 12

[15] Jiefeng Li, Jinkun Cao, Haotian Zhang, Davis Rempe, Jan Kautz, Umar Iqbal, and Ye Yuan. Genmo: A generalist model for human motion. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2025. URL https://research.nvidia.com/labs/dair/gem/. 1

[16] Jing Lin, Ailing Zeng, Shunlin Lu, Yuanhao Cai, Ruimao Zhang, Haoqian Wang, and Lei Zhang. Motion-x: A large-scale 3d expressive whole-body human motion dataset. *Advances in Neural Information Processing Systems*, 36:25268–25280, 2023. 2, 12

[17] Jing Lin, Ruisi Wang, Junzhe Lu, Ziqi Huang, Guorui Song, Ailing Zeng, Xian Liu, Chen Wei, Wanqi Yin, Qingping Sun, et al. The quest for generalizable motion generation: Data, model, and evaluation. *arXiv preprint arXiv:2510.26794*, 2025. 12

[18] Matthew Loper, Naureen Mahmood, Javier Romero, Gerard Pons-Moll, and Michael J. Black. SMPL: A skinned multi-person linear model. *ACM Trans. Graphics (Proc. SIGGRAPH Asia)*, 34(6):248:1–248:16, October 2015. 6, 8

[19] Shunlin Lu, Jingbo Wang, Zeyu Lu, Ling-Hao Chen, Wenxun Dai, Junting Dong, Zhiyang Dou, Bo Dai, and Ruimao Zhang. Scamo: Exploring the scaling law in autoregressive motion generation model. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 27872–27882, 2025. 2, 12

[20] Zhengyi Luo, Ye Yuan, Tingwu Wang, Chenran Li, Sirui Chen, Fernando Castañeda, Zi-Ang Cao, Jiefeng Li, David Minor, Qingwei Ben, Xingye Da, Runyu Ding, Cyrus Hogg, Lina Song, Edy Lim, Eugene Jeong, Tairan He, Haoru Xue, Wenli Xiao, Zi Wang, Simon Yuen, Jan Kautz, Yan Chang, Umar Iqbal, Linxi Fan, and Yuke Zhu. Sonic: Supersizing motion tracking for natural humanoid whole-body control. *arXiv preprint arXiv:2511.07820*, 2025. 1

[21] Naureen Mahmood, Nima Ghorbani, Nikolaus F Troje, Gerard Pons-Moll, and Michael J Black. Amass: Archive of motion capture as surface shapes. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 5442–5451, 2019. 15

[22] Zichong Meng, Zeyu Han, Xiaogang Peng, Yiming Xie, and Huaizu Jiang. Absolute coordinates make motion generation easy. *arXiv preprint arXiv:2505.19377*, 2025. 8, 12

[23] NVIDIA. Soma retargeter. https://github.com/NVIDIA/soma-retargeter, 2026. 4

[24] Georgios Pavlakos, Vasileios Choutas, Nima Ghorbani, Timo Bolkart, Ahmed A. A. Osman, Dimitrios Tzionas, and Michael J. Black. Expressive body capture: 3D hands, face, and body from a single image. In *Proceedings IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 10975–10985, 2019. 6

[25] Mathis Petrovich, Michael J. Black, and Gül Varol. TMR: Text-to-motion retrieval using contrastive 3D human motion synthesis. In *International Conference on Computer Vision (ICCV)*, 2023. 13

[26] Ekkasit Pinyoanuntapong, Pu Wang, Minwoo Lee, and Chen Chen. Mmm: Generative masked motion model. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1546–1555, 2024. 12

[27] Ekkasit Pinyoanuntapong, Muhammad Saleem, Korrawe Karunratanakul, Pu Wang, Hongfei Xue, Chen Chen, Chuan Guo, Junli Cao, Jian Ren, and Sergey Tulyakov. Maskcontrol: Spatio-temporal control for masked motion synthesis. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 9955–9965, 2025. 12

[28] Matthias Plappert, Christian Mandery, and Tamim Asfour. The KIT motion-language dataset. *Big Data*, 2016. 2

[29] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PmLR, 2021. 9

[30] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67, 2020. URL http://jmlr.org/papers/v21/20-074.html. 9

[31] Davis Rempe, Zhengyi Luo, Xue Bin Peng, Ye Yuan, Kris Kitani, Karsten Kreis, Sanja Fidler, and Or Litany. Trace and pace: Controllable pedestrian animation via guided trajectory diffusion. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023. 11

[32] Jun Saito, Jiefeng Li, Michael de Ruyter, Miguel Guerrero, Edy Lim, Ehsan Hassani, Roger Blanco Ribera, Hyejin Moon, Magdalena Dadela, Marco Di Lucca, Qiao Wang, Jan Kautz, Simon Yuen, and Umar Iqbal. Soma: Unifying parametric human body models. *arXiv*, 2026. 3, 6

[33] Cohan Setareh, Guy Tevet, Daniele Reda, Xue Bin Peng, and Michiel van de Panne. Flexible motion in-betweening with diffusion models. *ACM SIGGRAPH 2024 Conference Proceedings*, 2024. 2, 7, 9, 12

[34] Yoni Shafir, Guy Tevet, Roy Kapon, and Amit Haim Bermano. Human motion diffusion as a generative prior. In *The Twelfth International Conference on Learning Representations*, 2024. 7, 12

[35] Yi Shi, Jingbo Wang, Xuekun Jiang, Bingkun Lin, Bo Dai, and Xue Bin Peng. Interactive character control with auto-regressive motion diffusion models. *ACM Trans. Graph.*, 43, jul 2024. 12

[36] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502*, 2020. 11

[37] Qwen Team. Qwen3 technical report, 2025. URL https://arxiv.org/abs/2505.09388. 7

[38] Tencent Hunyuan 3D Digital Human Team. Hy-motion 1.0: Scaling flow matching models for text-to-motion generation. *arXiv preprint arXiv:2512.23464*, 2025. 2, 12

[39] Chen Tessler, Yunrong Guo, Ofir Nabati, Gal Chechik, and Xue Bin Peng. Maskedmimic: Unified physics-based character control through masked motion inpainting. *ACM Transactions on Graphics (TOG)*, 2024. 1

[40] Chen Tessler, Yifeng Jiang, Erwin Coumans, Zhengyi Luo, Xue Bin Peng, and Gal Chechik. Masked-manipulator: Versatile whole-body control for loco-manipulation. In *Proceedings of the SIGGRAPH Asia 2025 Conference Papers*, SA Conference Papers '25, New York, NY, USA, 2025. Association for Computing Machinery. ISBN 9798400721373. doi: 10.1145/3757377.3763934. URL https://doi.org/10.1145/3757377.3763934. 1

[41] Chen Tessler*, Yifeng Jiang*, Xue Bin Peng, Erwin Coumans, Yi Shi, Haotian Zhang, Davis Rempe, Gal Chechik†, and Sanja Fidler†. Protomotions3: An open-source framework for humanoid simulation and control. https://github.com/NVLabs/ProtoMotions/, 2025. 6

[42] 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. 2, 7, 12

[43] Unitree Robotics. Unitree g1 humanoid robot. https://www.unitree.com/g1, 2024. 3, 6

[44] Aaron Van Den Oord, Oriol Vinyals, et al. Neural discrete representation learning. *Advances in neural information processing systems*, 30, 2017. 12

[45] Weilin Wan, Zhiyang Dou, Taku Komura, Wenping Wang, Dinesh Jayaraman, and Lingjie Liu. Tlcontrol: Trajectory and language control for human motion synthesis. In *European Conference on Computer Vision*, pages 37–54. Springer, 2024. 12

[46] Ye Wang, Sipeng Zheng, Bin Cao, Qianshan Wei, Weishuai Zeng, Qin Jin, and Zongqing Lu. Scaling motion generation model with million-level human motions. In *International Conference on Machine Learning (ICML)*, 2025. 2, 12

[47] Yiming Xie, Varun Jampani, Lei Zhong, Deqing Sun, and Huaizu Jiang. Omnicontrol: Control any joint at any time for human motion generation. In *The Twelfth International Conference on Learning Representations*, 2024. 2, 12

[48] Brent Yi, Chung Min Kim, Justin Kerr, Gina Wu, Rebecca Feng, Anthony Zhang, Jonas Kulhanek, Hongsuk Choi, Yi Ma, Matthew Tancik, and Angjoo Kanazawa. Viser: Imperative, web-based 3d visualization in python. *arXiv preprint arXiv:2507.22885*, 2025. 3

[49] Yanjie Ze, Zixuan Chen, João Pedro Araújo, Zi ang Cao, Xue Bin Peng, Jiajun Wu, and C. Karen Liu. Twist: Teleoperated whole-body imitation system. *arXiv preprint arXiv:2505.02833*, 2025. 1

[50] Jianrong Zhang, Yangsong Zhang, Xiaodong Cun, Shaoli Huang, Yong Zhang, Hongwei Zhao, Hongtao Lu, and Xi Shen. T2m-gpt: Generating human motion from textual descriptions with discrete representations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023. 2, 12

[51] Lvmin Zhang, Anyi Rao, and Maneesh Agrawala. Adding conditional control to text-to-image diffusion models, 2023. 12

[52] 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. 2, 7, 12

[53] Mingyuan Zhang, Daisheng Jin, Chenyang Gu, Fangzhou Hong, Zhongang Cai, Jingfang Huang, Chongzhi Zhang, Xinying Guo, Lei Yang, Ying He, and Ziwei Liu. Large motion model for unified multi-modal motion generation. *arXiv preprint arXiv:2404.01284*, 2024. 12

[54] Kaifeng Zhao, Gen Li, and Siyu Tang. DartControl: A diffusion-based autoregressive motion model for real-time text-driven motion control. In *The Thirteenth International Conference on Learning Representations (ICLR)*, 2025. 12

[55] Yi Zhou, Connelly Barnes, Jingwan Lu, Jimei Yang, and Hao Li. On the continuity of rotation representations in neural networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 5745–5753, 2019. 8

A. Acknowledgments

We would like to thank John Malaska, Will Telford, Jon Shepard, and Anna Minx for helpful guidance throughout development. Thanks to Or Litany, Zhengyi Luo, Yeongho Seol, Jun Saito, and Michael Buttner for their insightful discussions on human motion. Thanks to Cyrus Hogg, Lindsey Pavao, Jenna Diamond, Rizwan Khan, Samantha Shinagawa, and Akanksha Shukla for their efforts on data acquisition and labeling. Thanks to Kaifeng Zhao and Sunmin Lee for research discussions and testing and feedback of the model and code.

B. Contributors

- **Project Lead**: Davis Rempe
- **Model**: Mathis Petrovich, Davis Rempe, Ye Yuan, Haotian Zhang, Xue Bin (Jason) Peng, Yifeng Jiang, Tingwu Wang, Umar Iqbal, David Minor, Michael de Ruyter, Jiefeng Li, Chen Tessler
- **Data**: Edy Lim, Eugene Jeong, Sam Wu, Ehsan Hassani, Michael Huang, Jin-Bey Yu, Chaeyeon Chung, Lina Song, Olivier Dionne
- **Advising**: Sanja Fidler, Simon Yuen, Jan Kautz