Interactive Character Control with Auto-Regressive Motion Diffusion Models

YI SHI, Simon Fraser University, Canada and Shanghai AI Lab, China
JINGBO WANG, Shanghai AI Lab, China
XUEKUN JIANG, Shanghai AI Lab, China
BINGKUN LIN, Xmov, China
BO DAI, Shanghai AI Lab, China
XUE BIN PENG, Simon Fraser University, Canada and NVIDIA, Canada

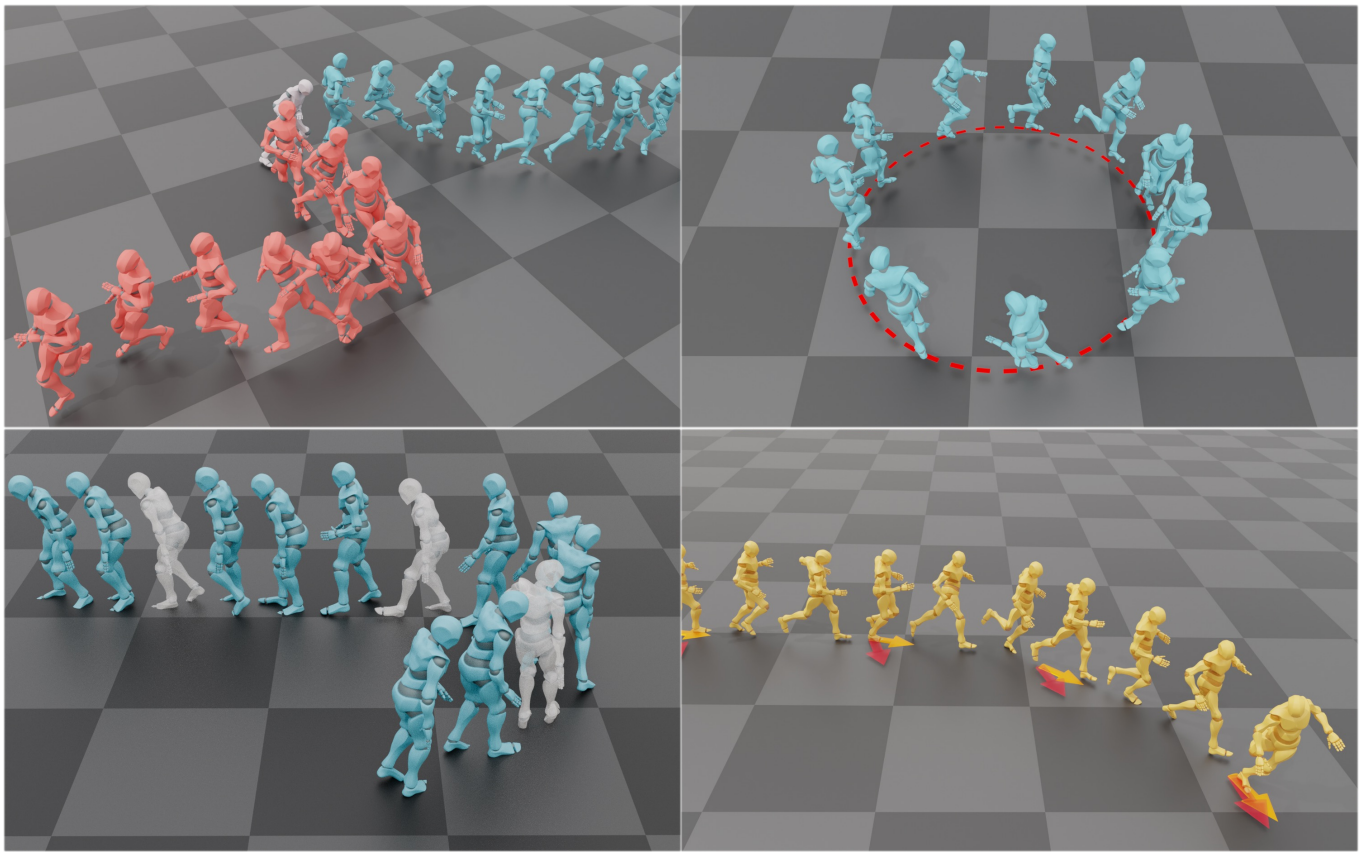


Fig. 1. We present Auto-Regressive Motion Diffusion Model (A-MDM), a framework for generating high-fidelity kinematic motion sequences. Once trained, A-MDM can be reused to perform new tasks through different control strategies, such as inpainting (upper right, and lower left), and hierarchical control via reinforcement learning (lower right).

Authors' addresses: Yi Shi, Simon Fraser University, Vancouver, Canada and Shanghai AI Lab, Shanghai, China, ysa273@sfu.com; Jingbo Wang, Shanghai AI Lab, Shanghai, China, wangjingbo1219@gmail.com; Xuekun Jiang, Shanghai AI Lab, Shanghai, China, jiangxuekun@pjlab.org.cn; Bingkun Lin, Xmov, Shanghai, China, linbingkun014@gmail.com; Bo Dai, Shanghai AI Lab, Shanghai, China, daibo@pjlab.org.cn; Xue Bin Peng, Simon Fraser University, Vancouver, Canada and NVIDIA, Vancouver, Canada, xbpeng@sfu.ca.

0730-0301/2024/7-ART143 $15.00
https://doi.org/10.1145/3658140

Real-time character control is an essential component for interactive experiences, with a broad range of applications, including physics simulations, video games, and virtual reality. The success of diffusion models for image synthesis has led to the use of these models for motion synthesis. However, the majority of these motion diffusion models are primarily designed for offline applications, where space-time models are used to synthesize an entire sequence of frames simultaneously with a pre-specified length. To

enable real-time motion synthesis with diffusion model that allows time-varying controls, we propose A-MDM (Auto-regressive Motion Diffusion Model). Our conditional diffusion model takes an initial pose as input, and auto-regressively generates successive motion frames conditioned on the previous frame. Despite its streamlined network architecture, which uses simple MLPs, our framework is capable of generating diverse, long-horizon, and high-fidelity motion sequences. Furthermore, we introduce a suite of techniques for incorporating interactive controls into A-MDM, such as task-oriented sampling, in-painting, and hierarchical reinforcement learning (See Figure 1). These techniques enable a pre-trained A-MDM to be efficiently adapted for a variety of new downstream tasks. We conduct a comprehensive suite of experiments to demonstrate the effectiveness of A-MDM, and compare its performance against state-of-the-art auto-regressive methods.

CCS Concepts: • **Computing methodologies** → **Computer graphics**; **Animation**; *Motion processing*.

Additional Key Words and Phrases: Motion Synthesis, Diffusion Model, Reinforcement Learning

ACM Reference Format:
Yi Shi, Jingbo Wang, Xuekun Jiang, Bingkun Lin, Bo Dai, and Xue Bin Peng. 2024. Interactive Character Control with Auto-Regressive Motion Diffusion Models. *ACM Trans. Graph.* 43, 4, Article 143 (July 2024), 14 pages. https://doi.org/10.1145/3658140

1 INTRODUCTION

Synthesizing high-fidelity and controllable motions for virtual characters is one of the central challenges in computer animation, with broad applications for visual effects, games, simulations, and virtual reality. The intricate nature of human motion, which encompasses a vast array of styles and tasks, presents a formidable challenge for high-quality motion generation. In recent years, researchers have drawn on the success of data-driven methods utilized in text and image synthesis to address these challenges [Ho et al. 2020; Rombach et al. 2022; Ruiz et al. 2022; Saharia et al. 2022; Song et al. 2021]. These methods have achieved promising results in terms of motion quality and diversity. Among them, diffusion models have shown great potential in synthesizing diverse and high-quality motion sequences. However, the majority of these methods employ space-time models to generate sequences of frames simultaneously via a single diffusion process [Dabral et al. 2023; Huang et al. 2023; Ma et al. 2022; Raab et al. 2023; Shafir et al. 2023; Tevet et al. 2023; Xin et al. 2023; Zhang et al. 2022, 2023a]. This approach inherently restricts the applicability of diffusion models for tasks that require real-time interactivity.

In this work, we present Auto-regressive Motion Diffusion Models (A-MDM), a redesign of the motion diffusion model in an auto-regressive form to generate motions frame-by-frame. To improve run-time performance, our approach predicts the next frame using under 50 denoising steps, which is significantly less than the 1000 steps commonly used in prior methods [Shafir et al. 2023; Tevet et al. 2023; Xin et al. 2023], thus enabling our model to run in real-time with modest computational resources. This design choice, combined with our incremental frame-by-frame generation approach, allows our A-MDM model to be used for motion control applications with real-time user inputs. Through both qualitative and quantitative assessments, we show that this lightweight design still retains competitive performance in terms of motion quality and diversity when compared to state-of-the-art motion synthesis models. Our model can be effectively trained using large-scale motion datasets, such as AMASS [Mahmood et al. 2019]. Furthermore, we also explore techniques for combining A-MDM with high-level policies to create character controllers for a variety of downstream tasks, such as joystick control, target reaching, and trajectory tracking.

Once trained, A-MDM can act as a flexible base model that can be used in a variety of downstream applications. We present a suite of techniques for controlling A-MDM for downstream tasks, including task-oriented sampling, motion in-painting, keyframe in-betweening, and hierarchical control. The key contributions of this work include:

- We propose A-MDM, an auto-regressive motion diffusion model, which is more amenable for real-time interactive character control. A-MDM can generate more diverse motion sequences than previous VAE-based auto-regressive models [Ling et al. 2020; Rempe et al. 2021].
- We present a suite of methods for controlling A-MDM to synthesize motions for new tasks without any additional training or fine-tuning of the model.
- We present a hierarchical reinforcement learning approach for controlling A-MDM, which is able to train hierarchical policies for new downstream tasks.

2 RELATED WORKS

Developing procedural methods that can automatically synthesize life-like animations for virtual characters has been one of the fundamental problems in computer graphics. A large suite of techniques has been proposed to tackle this problem, which can be broadly categorized into kinematic and physics-based techniques [Li et al. 2022; Peng et al. 2018, 2022, 2021; Rempe et al. 2021; Tevet et al. 2023].

Since our work falls into the class of kinematic methods, our discussion will focus on the most closely related work on kinematic motion generation. The core of our system is an auto-regressive motion diffusion model trained on a dataset of motion clips, which can then reused by a variety of control techniques to synthesize motions for new downstream tasks.

2.1 Kinematic Motion Generation

Kinematic techniques typically follow one of two main paradigms: space-time models and auto-regressive models.Space-time models produce an entire motion sequence simultaneously [Li et al. 2022; Tevet et al. 2023; Xin et al. 2023; Yan et al. 2019]. Alternatively, auto-regressive methods generate a motion sequentially frame-by-frame [Fragkiadaki et al. 2015; Ling et al. 2020; Martinez et al. 2017; Rempe et al. 2021], allowing greater responsiveness to time-varying objectives, and enabling these models to be more easily used for real-time applications. In this paper, we focus on auto-regressive generation, which can produce motions for real-time interactive applications. In this section, we will provide a review of both space-time and auto-regressive methods.

Auto-Regressive Models. Machine learning techniques for auto-regressive motion generation have often used CNNs, LSTMs, and

MLPs to sequentially predict the next frame conditioned on previously generated frames [Aksan et al. 2019; Fragkiadaki et al. 2015; Gopalakrishnan et al. 2019; Li et al. 2017; Martinez et al. 2017; Pavllo et al. 2018]. However, applying these models directly to generate long-horizon motions often leads to drift or convergence to a mean pose, resulting in unnatural motions. Researchers have introduced custom architectures to address these issues, which can improve motion quality and diversity. For instance, phase-functioned neural networks dynamically blend between different model parameters based on the motion phase to generate coherent long-horizon motions [Holden et al. 2017]. However, this dependency on phase undermines its ability to model acyclic behaviors, and behaviors that do not progress according to a well defined phase variable.

Variational auto-encoder models (VAE) have been a popular class of generative model for motion synthesis. These methods leverage a VAE framework to learn a motion manifold from motion data, which can enhance diversity and generalization in the generated motions [Kingma and Welling 2013; Ling et al. 2020; Rempe et al. 2021]. Motion VAE (MVAE) employs a Mixture-of-Expert (MoE) network alongside a VAE model to generate diverse motions [Ling et al. 2020].

Humor models the manifold of transitions between two adjacent frames via a VAE model [Rempe et al. 2021]. Hassan et al. [2021] utilizes a VAE model to synthesize long-horizon motions of human-scene interactions. In our work, we introduces an auto-regressive diffusion model, which leverages the expressive power of diffusion model to better represent the complex multi-modal nature of human motions. This expressiveness enables our model to produce more diverse and higher fidelity motions compared to previous auto-regressive models.

Space-Time Models. Unlike auto-regressive models, space-time models generate a sequence of motion frames simultaneously. As result, these models are typically more suitable for offline applications. Space-time models based on CNNs [Wang et al. 2021; Yan et al. 2019], and transformer are popular choices in prior studies [Kim et al. 2022; Petrovich et al. 2021; Wang et al. 2022], as these architectures are able to capture temporal correlation across long motion sequences. VAE [Kingma and Welling 2013] and GAN [Goodfellow et al. 2020] frameworks have been used to trained space-time models.

Recent works have applied diffusion model to space-time motion synthesis models [Tevet et al. 2023; Zhang et al. 2022]. However, a common limitation of diffusion models is the significant computational cost and time needed for sampling, which can preclude effective use in real-time applications. Xin et al. [2023] attempted to address this challenge by building a diffusion model in a compact VAE-encoded latent space. While their method demonstrated some real-time generation capabilities, the time required to generate a single frame is still too slow for real-time applications. In this work, we propose a lightweight auto-regressive diffusion model with specialized design decisions to speedup generation of motions in an auto-regressive fashion, which then enables our A-MDM model to synthesize high-fidelity motions in real-time and respond to interactive control inputs.

2.2 Latent-Space Models

Latent-space models typically leverage a learned latent motion representation, which can then be used to generate new motions for downstream tasks. Commonly used approaches for learning such latent-space models utilize generative modeling techniques, such as VAEs and GANs. MVAE generates motions using latent motion representations learned through a conditional variational auto-encoder [Ling et al. 2020]. Won et al. [2022] and Yao et al. [2022] leverage model-based RL methods to train physics-based motion VAEs. Once a latent space of motions has been constructed, these frameworks then train task-specific high-level controllers to sample the appropriate latent codes for solving new tasks. In practice, VAE models tend to generate lower-quality results compared to diffusion models in the image synthesis domain [Bredell et al. 2021; Ho et al. 2020], and similar issues have also been observed for motion generation with space-time model [Guo et al. 2022; Petrovich et al. 2022; Tevet et al. 2023]. In this work, we show that with the appropriate design decisions, diffusion models can effectively generate high-quality motions for real-time responsive character control.

GAN-based models learn a latent motion representation by optimizing a variational approximation of the divergence between the dataset and samples from the model [Barsoum et al. 2017; Li et al. 2022; Men et al. 2022; Peng et al. 2022, 2021; Tessler et al. 2023].

This is done by training an adversarial discriminator to distinguish samples from the dataset and samples from the model, thereby encouraging the model to produce samples that resemble the data. However, GAN training tends to be unstable and susceptible to mode collapse, requiring a myriad of heuristics to stabilize training.

In this paper, our method leverages diffusion models for motion synthesis, which has been shown to be much more stable and easier to train compared to GANs. It is also known to achieve better generation quality and data distribution coverage than adversarial methods in other domains, such as image synthesis [Prafulla Dhariwal 2021].

To apply VAE and GAN models to new tasks, hierarchical controllers are often trained to select the appropriate behaviors from the learned latent space that enable a character to fulfill the desired task objectives [Ling et al. 2020; Peng et al. 2022; Tessler et al. 2023].

We show that A-MDM is also amenable to hierarchical control, where a task-specific high-level controller can be trained to steer the denoising process in order to generate motions that satisfies the desired task objectives.

3 METHOD

Our framework consists of two key components: a base auto-regressive motion diffusion model (A-MDM) and an RL-based task controller. The base A-MDM predicts the next motion frame f based on the previous frame $f-1$. This auto-regressive procedure enables A-MDM to synthesize long motions of arbitrary lengths in real time. To achieve task-based character control, we utilize a task-specific high-level controller, which is trained using reinforcement learning to direct the base A-MDM model to produce motions that fulfill a given task objective. This framework enables our system to generate high-quality, task-oriented motions for a wide range of applications. Furthermore, we demonstrate that inpainting techniques can also be

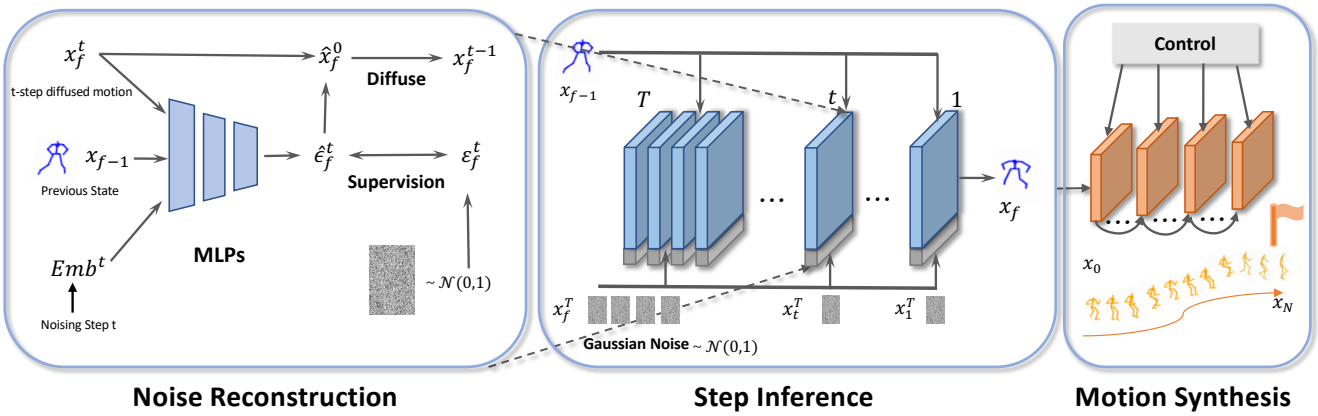


Fig. 2. **Framework of our A-MDM.** Our A-MDM is trained following DDPM [Ho et al. 2020]. During training, the goal of our A-MDM is to reconstruct the sampled noise vector ϵ_f^t at each step. After training, our A-MDM is capable of generating long-horizon motion with arbitrary lengths under different controlling strategies in an autoregressive manner.

incorporated into our auto-regressive model, which enables users to directly specify desired characteristics of the generated motions. We show that the model is able to generate natural and responsive motions that adhere to the target specifications provided by the user.

3.1 Auto-regressive Motion Diffusion Model

In this section, we will present the details of our auto-regressive motion diffusion model. Given the state of the character at frame $f-1$, A-MDM models the distribution of possible character states at frame f. The training procedure of A-MDM follows DDPM, where the model is trained to model the distribution of the motion data by denoising Gaussian noise introduced by a forward diffusion process. Once trained this model can then produce diverse, high-fidelity, and long-horizon motion sequences by sampling from the reverse diffusion process [Ho et al. 2020].

Motion Representation. Given the initial state x_0, A-MDM can generate a motion sequence $X = \{x_1, x_2, .., x_F\}$ with arbitrary length F in an auto-regressive manner. The state x_f at frame f is represented by features consisting of: planar linear velocity of the root ($d_x \in \mathbb{R}$, $d_y \in \mathbb{R}$), angular velocity ($d_r \in \mathbb{R}$) around the up-axis, joint positions ($j_p \in \mathbb{R}^{j\times 3}$), joint velocities ($j_v \in \mathbb{R}^{j\times 3}$), and joint orientations in the 6D representation ($j_o \in \mathbb{R}^{j\times 6}$) [Zhou et al. 2019]. The root features d_x, d_y, and d_r are recorded in the character's local coordinate frame.

Pipeline Overview. Given an input character state, x_{f-1}, A-MDM predicts a potential next state x_f in an auto-regressive manner. The model is trained using a procedure based on DDPM [Ho et al. 2020], which includes a forward diffusion process and a backward reconstruction process. During the forward diffusion process, Gaussian noise is introduced at each diffusion step t into the next state x_f^t. This process is applied for T steps until x_f^T converges to an isotropic Gaussian distribution. The backward reconstruction process again consists of T steps, representing a reversed procedure where a conditional denoising model is trained to gradually remove the noise in x_f^t conditioned on the previous state x_{f-1}, yielding the final prediction x_f.

During training, Gaussian noise is added progressively through a Markov process to the ground truth character state x_f. For a single forward step t, the noising process is given by:

$$q(x_f^t|x_f^{t-1}) = \mathcal{N}(x_f^t; \sqrt{1-\beta^t}x_f^{t-1}, \beta^t I), \tag{1}$$

where $\beta^t \in (0,1)$ is a hyper-parameter that determines the noise schedule. The whole forward diffusion process can be described according to:

$$q(x_f^{1:T}|x_f^0) = \prod_{t=1}^{T} q(x_f^t|x_f^{t-1}), \tag{2}$$

We denote the noise applied to state x_f after $t \in [1,T]$ steps as ϵ_f^t. The denoising process is performed using a neural network p_θ. The network takes as input the perturbed target state x_f^t, which is produced by applying noise ϵ_f^t from the standard Gaussian distribution to the state $\hat{x}_f^0$ at denoising step $t+1$. Additional input to the network includes the previous state x_{f-1}, the time embedding e^t at time step t. The denonising network then predicts the noise vector $\hat{\epsilon}_f^t$ that was applied to the original input. An MSE loss is applied between the predicted noise and the ground truth noise ϵ_f^t:

$$L_t^{simple}(x) = \mathbb{E}_{t\sim[1:T],x_f^0,\epsilon_f^t}||\epsilon_f^t - p_\theta(x_{f-1}, x_f^t, e^t)||_2. \tag{3}$$

During inference, the reverse denoising process starts from Gaussian noise ϵ_f^T. At each diffusion step t, the denoising network predicts the noise $\hat{\epsilon}_f^t$, which is used to reconstruct the original state $\hat{x}_f^t$. Then, $\hat{x}_f^t$ serves as the input to the next denoising step. The final predicted state $\hat{x}_f$ is used as the output $\hat{x}_f^0$ after T denoising steps.

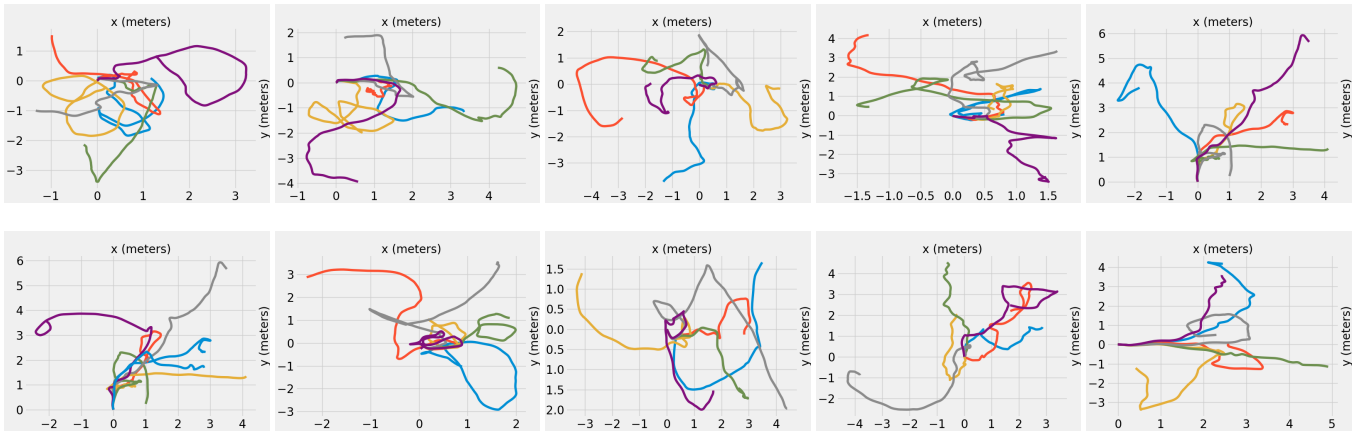


Fig. 3. **Trajectories for Target Reaching.** We show that our framework is capable of generating diverse motion trajectories, with the same initial state and target goals.

3.2 Scheduled Sampling with Diffusion Model

In Section 3.1, we discussed the basic training and inference procedure for A-MDM. However, generating motion with an auto-regressive model is susceptible to drift due to error accumulation. This phenomenon can lead to catastrophic failures when generating long-horizon motions. To mitigate drift, prior methods leverage student forcing within the framework of schedule sampling [Ling et al. 2020; Rempe et al. 2021]. In comparison to teacher forcing, where both the input and the target output are sampled from the training data, student forcing uses the model's own predicted state from a past frame $\hat{x}_{f-1}$ as the input to predict the subsequent state $\hat{x}_{f+1}$. This process can be repeated over multiple steps N_r. Scheduled sampling enables an auto-regressive model to compensate for drift due to errors in its previous predictions. This can then enable auto-regressive models to generate long-horizon motions with arbitrary length, which is essential for online interactive applications. In this work, we observed that scheduled sampling is also crucial for mitigate drift with A-MDM. To implement student forcing, for every frame x_f in a batch of samples, we compute the loss described in Equation 3, and then use the reconstructed $\hat{x}_f^t$ as the next frame for reconstructing subsequent frames. Following the same procedure, we can sample N_r subsequent frames auto-regressively from the model $\{\hat{x}_{f+1}, \hat{x}_{f+2}, ..., \hat{x}_{f+N_r}\}$. The generated frames are then used as the input conditioning for computing the training loss for subsequent frames instead of the ground-truth frames from the dataset.

3.2.1 Network Architecture. To enable real-time motion control, our framework leverages a lightweight network architecture for A-MDM. Unlike prior work, which often uses computationally expensive architectures, such as Transformers [Tevet et al. 2023; Xin et al. 2023], our A-MDM model features a streamlined architecture centered around a lightweight Multi-Layer Perceptron (MLP), as illustrated in Figure 2. The use of a diffusion model significantly enhances our model's capacity, enabling the generation of diverse and high-quality motions using a simple model architecture. We demonstrate that using MLPs in diffusion models with fewer denoising steps is sufficient for synthesizing high-quality motions, outperforming prior auto-regressive models, such as MVAE, in both motion fidelity and diversity. Our model is composed of 10 fully-connected layers, with each layer containing 1024 hidden units with SiLU activations followed by a layer norm. The input to this model includes the state at the previous step x_{f-1}, the perturbed target state x_f^t, and the time embedding e^t of the diffusion step t.

Real-time Design Decisions. In diffusion models, a common practice is to use a large number of diffusion steps (*e.g.*, 1000 steps as used in DDPM). However, for auto-regressive motion generative models, like A-MDM, this can lead to slow generation times, rendering them impractical for real-time applications. Our experiments demonstrate that achieving high-quality motion generation with A-MDM is possible using a smaller number of diffusion steps T. This is in part attributed to the smaller dimensional output for a single frame compared to an entire motion sequence. Our empirical findings suggest that very few steps (*e.g.*., 10-50) are sufficient for motion generation tasks.

4 MOTION SYNTHESIS

Once trained, A-MDM is able to generate long-horizon motions with unbounded lengths by auto-regressively sampling from the diffusion model at each time step. We present four methods for long-term motion generation: random sampling, task-oriented sampling, conditional inpainting, and hierarchical control.

Random Sampling: The first method, random sampling involves sampling from A-MDM without imposing any constraints. We show that our model is able to generate diverse multi-modal motions starting from a fixed initial state.

Task-Oriented Sampling: Task-oriented sampling involves generating future states through beam search with a user-defined target score function. Several randomly sampled candidates trajectories are generated and evaluated by the score function, and the first frame

from the best trajectory is selected as the next frame. This task-oriented sampling approach enables A-MDM to perform tasks, such as goal-reaching, without requiring further training. However, it is essential to note that task-oriented sampling often introduces unpredictable behaviors, as also observed in prior work [Ling et al. 2020], where sub-optimal behaviors such as running in circles around a goal may arise. It also lacks support for the user to specify precise fine-grained constraints, beyond defining how the target score is calculated.

Conditional Inpainting: The third method employs inpainting techniques to generate motions that adhere to user specifications. This technique has been highly effective for image synthesis [Lugmayr et al. 2022], and space-time motion models [Tevet et al. 2023]. With inpainting, users can specify trajectories and positions for various joints in the character's body, and A-MDM is then used to generate a plausible full-body motion that adheres to those constraints.

Hierarchical Control: Finally, we show that A-MDM can also be incorporated into a hierarchical reinforcement learning framework to train task-specific high-level controllers on top of the base A-MDM model. This enables our framework to create controllers for accomplishing new tasks, while producing higher motion quality and robustness compared to task-oriented sampling and inpainting.

4.1 Synthesis via Random Sampling

Given the initial character state, our A-MDM can generate plausible motions for future frames by sampling auto-regressively from the diffusion model, enabling the generation of diverse long-horizon motion sequences in real time. The diverse trajectories of synthesized motion from different initial states are shown in Figure 3. Our supplementary video also demonstrates the capability of our A-MDM to transition from a stationary pose to diverse behaviors, such as walking, running, and jumping. When presented with different initial states, A-MDM is capable of producing motions that are consistent with the initial state. For instance, it can generate hopping motions when initialized with a hopping pose. These instances showcase the model's ability to learn the distribution of plausible motions associated with specific states. Our supplementary video offers additional examples, providing a comprehensive showcase of the diverse motions that can be generated by A-MDM.

4.2 Synthesis via Task Oriented Sampling

Random sampling alone cannot generate motions that conform to user commands. Therefore, following the sampling-based control method presented by Ling et al. [2020], we implement a task-oriented sampling technique to generate motions that follow a user-defined tasks. First, a pool of P random candidate trajectories are generated by sampling from A-MDM. The candidates are ranked according to a user-defined score function. For example, in the goal-reaching task, the score function is determined by the 2D Euclidean distance on the ground plane between the character and goal. Then the candidate trajectory with the best score is selected and its first frame is used as the character's next state. This process is repeated

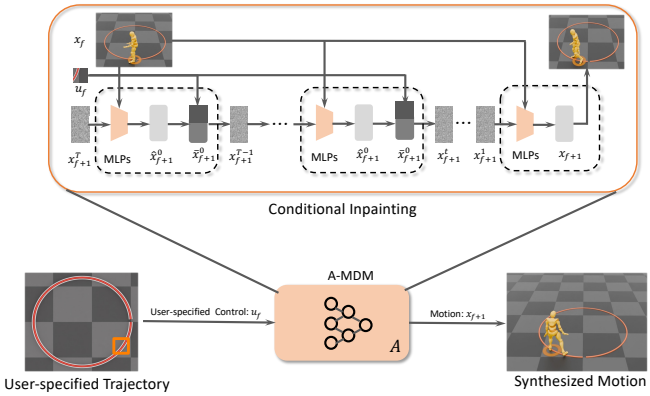


Fig. 4. Inpainting can generate seamless motion transitions between user-specified motions and arbitrary character states. We introduce a series of buffer frames where inpainting stops at an early diffusion step. While playing out the user-imposed target motion, inpainting is done until the last denoising step.

Fig. 5. **Transition In-betweening through inpainting.** To generate smoother transitions, we initialize the denoising process using the target frame at different denoising steps. As the frames approach the time of the target frame, the denoising process is initialized at a later and later denoising step with the target frame, which leads the generated frames to more closely conform to the target frame.

for every frame until the task is completed or terminated by other constraints.

We find that the task-oriented sampling works well for simple locomotion tasks, such as target reaching (as shown in Section 6.1). Figure 10 illustrates trajectories of sampled motion for this task. Compared to Motion VAE [Ling et al. 2020], our method can often successfully navigate to the target motion directly, rather than producing less efficient circling motions. However, for more precision-oriented tasks such as joystick and path following, task-oriented sampling can still struggle to closely follow the specified commands.

4.3 Synthesis via Conditional Inpainting

For tasks that require more precise control of the generated motions, task-oriented sampling may not be able to closely follow a user's specifications. To enable more fine-grain control, we show that A-MDM is also amenable to diffusion model-based inpainting, as shown in Figure 4. This approach enables A-MDM to be used for a wide range of tasks without requiring further fine-tuning of the

model. When applying inpainting to A-MDM, the user first specifies desired features $\tilde{x}_f$ for the motion at each frame f, along with a binary mask m_f that records which features are explicitly specified by the user. Each entry in m_f is assigned a value of 1 if the feature is specified by the user, and 0 if the feature was not assigned specific values by the user. To incorporate the user's control into the denoising process, the values of features x_f^t after each denoising step is directly replaced with the values specified by the user,

$$\hat{x}_f^t = (1 - m_f) \odot x_f^t + m_f \odot \tilde{x}_f^t, \quad x_f^t \sim q\left(x_f^t \middle| x_f^{t-1}\right), \tag{4}$$

where $\odot$ denotes a component-wise multiplication. The resulting frame $\hat{x}_f^t$ is then used as the input to the next denoising step. This simple inpainting method then enables A-MDM to produce coherent full body motions that precisely follow controls from the user as shown in Figure 7. For example, the user can specify trajectories for various joints, such as the root, and A-MDM can then generate a plausible full body motion that follows the desired trajectory. We refer to this form of inpainting as *spatial inpainting*.

In addition to spatial inpainting, where a full-body motion is generated from a user-specified trajectory for a subset of the joints, A-MDM can also be used for *temporal inpainting*, more commonly referred to as *keyframe inbetweening*. For keyframe inbetweening, the user specifies target key frames $\tilde{x}_f$ at a sparse number of frames, where each keyframe specifies values for the full state of the character. The goal then is to generate a natural motion for the intermediate frames between two adjacent keyframes. We propose a simple method to generate natural transitions between two keyframes using A-MDM by simply initializing the denoising process with the target keyframe starting at different denoising steps. Given an initial frame x_0 and a target keyframe $\tilde{x}_N$ at frame N, the target keyframe is gradually introduced into the denoising process by first initializing the denoising process at frame f with the target keyframe $\tilde{x}_N$ an then applying the denoising process starting at diffusion step t_0,

$$t_0 = \left(1 - \frac{f}{N}\right) t_{\max}, \tag{5}$$

where $t_{\max}$ is the maximum number of diffusion steps for each denoising process. For early frames $f \approx 0$ in the inbetweening process, $\tilde{x}_N$ is used to initialize the early denoising steps $t \approx t_{\max}$, and many denoising steps are then applied, which allows the model flexibility to deviate from the keyframe as needed to generate a more natural motion at the early frames. As the frames get closer to the keyframe at $f = N$, fewer and fewer denoising steps are applied to $\tilde{x}_N$, which enforces that the later frames closely match the desired keyframe, as shown in Figure 5. In our experiments, we show that this simple inbetweening method allows A-MDM to produce realistic motions between keyframes, as well as generate natural transitions between disparate motion clips. An example of the motion inbetweening is shown in Figure 6, and more examples are available in the supplementary video.

5 HIERARCHICAL CONTROL

To address the limitation of task-oriented sampling and inpainting, we propose using hierarchical reinforcement learning to train high-level controllers that steer the base A-MDM model towards producing motions that follow the user's commands. Once the high-level controller is trained, the proposed approach demonstrates significantly more effective performance on new tasks compared to the sampling-based techniques. Furthermore, we show that this approach offers greater flexibility when completing the same tasks than conditional inpainting, as shown in Figure 13.

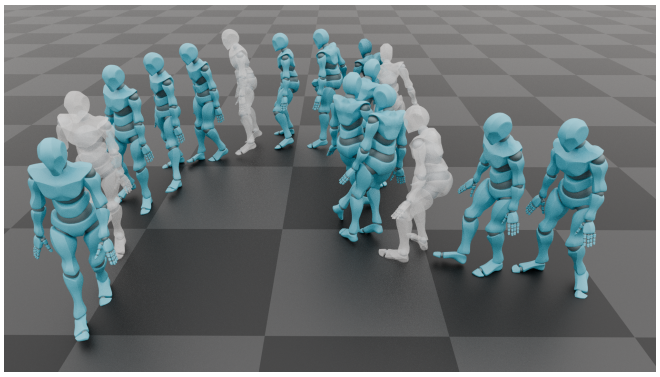


Fig. 6. A-MDM can be used for key-frame in-betweening to generate plausible motions (blue) between user specified key-frames (white).

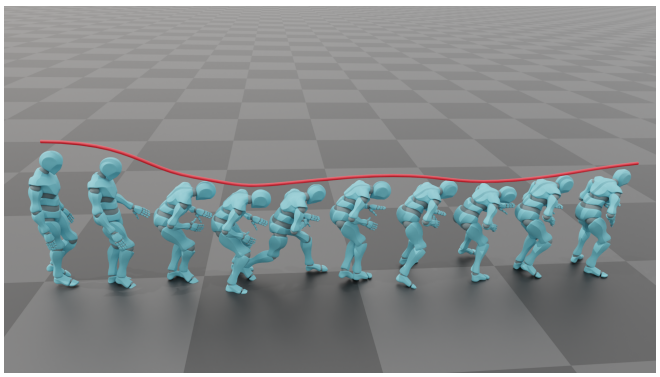


Fig. 7. **Joint position control through inpainting.** Given desired trajectories for the head and root, A-MDM is able to synthesize high-quality motions via inpainting techniques without additional finetuning.

5.1 Reinforcement Learning

Reinforcement Learning (DRL) is used to train task-specific policies for directing the low-level A-MDM model towards completing new tasks. With reinforcement learning, policies are trained by optimizing a policy's expected return:

$$J_{RL}(\pi) = \mathbb{E}_{\tau \sim p(\tau|\pi)}\left[\sum_{f=0}^{\infty} \gamma^f r(s_f, a_f)\right], \tag{6}$$

where $p(\tau|\pi)$ is the distribution of trajectories τ induced by a policy π, $r(s_f, a_f)$ is the reward that defines a desired task, and $\gamma \in [0, 1]$ is a discount factor. The agent receives a reward at each timestep f by executing an action a_f at state s_f. We optimize this objective using Proximal Policy Optimization (PPO) [Schulman et al. 2017] to train task-specific high-level controllers for each task. Noted

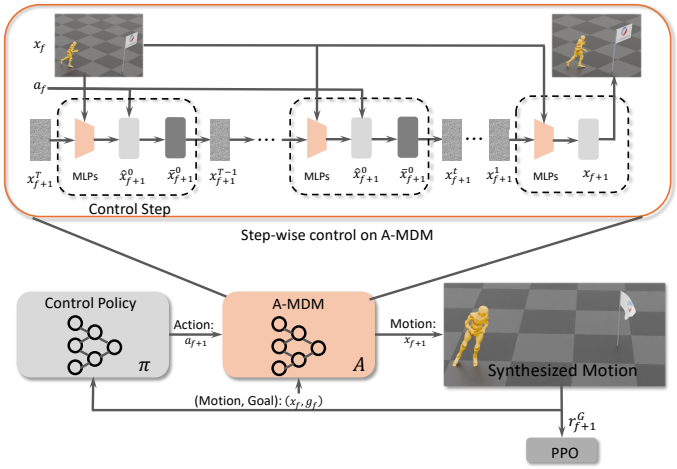


Fig. 8. Hierarchical Control for A-MDM. The high-level controller predicts the residual vectors a_f^t of $\hat{x}_f^t$ to steer the denoising process of the base A-MDM.

we use f to represent the MDP timestep to distinguish it from the diffusion timestep t. At each frame f, the agent takes as input the observation of the current state s_f of the environment, and then selects an action that is used to steer the sampling procedure of the pretrained A-MDM.

5.2 Hierarchical Control for A-MDM

In previous VAE-based models [Ling et al. 2020; Won et al. 2022; Yao et al. 2022], a base VAE model is trained to produce a latent space that can be directly used to sample motions for the next frame. Hierarchical controllers are then trained to sample latent codes from the latent manifold of the base VAE model. In contrast, it's not straightforward to train a hierarchical controller to control diffusion model, where Gaussian noises is introduced iteratively during inference. To apply a pre-trained A-MDM model to new tasks, we train task-specific high-level controllers to steer the denoising process to produce motions that satisfy desired task objectives. Figure 8 provides a schematic illustration of our hierarchical controller. Given the character's current state x_f, the high-level controller predicts an action $a_f = \{a_f^T, .., a_f^1\}$, which consists of multiple residual vectors for different denoising steps. Each residual vector a_f^t is applied as a perturbation to the output of a denoising step t,

$$\bar{x}_{f+1}^t = \hat{x}_{f+1}^t + w^t a_f^t. \tag{7}$$

The perturbed output $\bar{x}_{f+1}^t$ is then used as the input for the next denoising step $t-1$. w^t is a weight that anneals the perturbation according to the denoising step. The annealing schedule is designed to gradually attenuate towards the end of the denoising steps to prevent sudden changes. This annealing is crucial since large perturbations from the high-level controller at the later denoising steps can lead to motion artifacts.

6 TASKS

To evaluate the effectiveness of our hierarchical model, we evaluate our framework through three tasks: Target Reaching (Section 6.1), Joystick Control (Section 6.2), and Path Following (Section 6.3).

6.1 Target Reaching

Target reaching is a locomotion task in which the character is required to move towards a user-designated target location. The location of the target can be altered by the user at any time within an episode.

The task is completed if the character's hip joint is within 15 cm of the target. Once the character has reached a target, a new target is randomly placed within a 20m×20m region around the character's current location. The task observation of the high-level controller consists of the target position G^* relative to the character.

The reward is computed using a combination of a distance reward between the character's position and G^*, as well as the progress made towards G^*. When the agent reaches the goal, the agent will also receive a one-time bonus reward before a new target is set.

6.2 Joystick Control

The Joystick Control task allows the user to specify the speed and orientation of the character. The character is trained to generate natural motions that follow the user specified velocity and heading. To train the high-level controller for this task, random joystick commands are generated by changing the desired direction and speed every 120 and 240 frames respectively. The task observation consists of the facing direction d_r^g, which is uniformly sampled between 0 and 2π, as well as the desired velocity d_v^g, which is uniformly sampled from 0.6m/s to 7.2m/s. The reward function is given by

$$r = e^{cos(d_r - d_r^g) - 1} \times e^{-|d_v - d_v^g|}, \tag{8}$$

where d_r is the character's angular velocity, and d_v is the character's planar linear velocity. To ensure that the policy satisfies both of these objectives simultaneously, the two rewards are combined multiplicatively.

6.3 Path Following

In the Path Following task, the character is required to navigate along a user-defined path composed of several waypoints. The character must reach each waypoint in the specified order. The task observation consists of the locations of the future waypoints along the pre-defined path. During training, paths are procedurally generated by randomly sampling linear and angular accelerations for each waypoint along the path. Once trained, the controller is able to follow arbitrary paths at runtime. Figure 13 shows examples of our hierarchical controller following user-specified trajectories. The base A-MDM model enables the character to follow the target trajectories with natural behaviors.

7 EXPERIMENTS

In this section, we demonstrate the effectiveness of our A-MDM framework. First, we evaluate the base A-MDM model and compare it with prior VAE-based models (*e.g.*, MVAE and HuMoR [Ling et al. 2020; Rempe et al. 2021]) to showcase the effectiveness of our design choices for motion modeling. We then show that A-MDM can be

Table 1. Comparisons on AMASS, 100STYLE and LaFAN1. 50 motion sequences are generated starting at fixed initial states. Each motion is 60 frames long when evaluating ADE and 150 frames long for calculating APD. (unit: cm).

	APD ↑	ADE ↓	FDE ↓	FS ↓	Bone.Err ↓	Jnt.Accel→	Pen.Freq ↓	Pen.Dist ↓
AMASS	-	-	-	-	-	7.28	-	-
MVAE	40.97±3.97	24.42±1.04	42.22±3.64	1.44±0.12	0.98±0.12	7.54 ±0.51	1.94±0.56%	1.26±0.82
HuMoR	44.95±4.49	17.96±1.17	41.10±3.98	1.35±0.11	1.04±0.07	7.69±0.49	1.78±0.62%	1.17±0.93
AMDM (Ours)	**61.08**±1.35	**10.40**±0.66	**21.12**±1.16	**1.06**±0.11	**0.82**±0.04	**7.26**±0.19	**0.4**±0.03%	**1.07**±0.32
100STYLE	-	-	-	-	-	9.87	-	-
MVAE	58.44±1.36	23.47±0.42	48.24±1.52	1.62±0.05	0.25±0.02	9.48±0.52	1.87±0.84%	0.06±0.02
HuMoR	68.25±1.45	17.41±0.47	43.34±1.74	1.57±0.06	0.23±0.02	**9.59**±0.48	1.80±0.92%	0.07±0.04
AMDM (Ours)	**102.52**±1.17	**10.36**±0.22	**24.58**±0.06	**1.53**±0.02	**0.19**±0.01	9.37±0.28	**1.56**±0.24	**0.04**±0.01
LaFAN1	-	-	-	-	-	12.56	-	-
MVAE	110.48 ±5.67	35.50±3.43	82.93±4.8	2.42±0.49	0.66±0.02	12.81±0.37	0.83±0.09%	0.65±0.14
HuMoR	132.76±4.25	24.35±2.19	41.61±2.97	2.20±0.26	**0.53**±0.03	13.03±1.19	**0.71**±0.03%	0.50±0.06
AMDM (Ours)	**134.92**±6.01	**14.22**±2.20	**34.53**±4.16	**2.10**±0.56	0.54±0.02	**12.74**±0.35	0.76±0.07%	**0.49**±0.12

combined with hierarchical RL to solve downstream motion control tasks.

7.1 Experimental Setup

Our framework is implemented with Pytorch. The experiments with hierarchical controllers are conducted using OpenAI Gym [Brockman et al. 2016]. All experiments are performed on a PC with an NVIDIA GeForce 4090 GPU and Intel Core i9-13900K.

7.2 Dataset

Our framework is evaluated on three datasets that vary in terms of size and motion diversity:

(1) **100STYLE** contains more than 4,000,000 frames of motion capture data of 100 diverse styles of locomotion [Mason et al. 2022]. For each style, the dataset consists of motions with different velocities (idle, running, and walking), and different headings (sidewalk, forward, and backward). For evaluation, we partition the data according to the official training and testing splits.
(2) **AMASS** is a large-scale motion capture database, represented with a unified SMPL body model [Mahmood et al. 2019]. We evaluate our model on this dataset to show that A-MDM can be effectively applied to large dataset and produce better performance than prior auto-regressive models.
(3) **LaFAN1** is a high-quality motion capture dataset containing highly dynamic motions, including dancing, crawling, and fast locomotions, with a size of more than 400,000 frames [Harvey et al. 2020]. We train our model with a subset of the original LaFAN1 dataset, excluding environment interaction motions.

All motion clips are standardized to a frame rate of 30 Hz.

7.3 Evaluation Metrics

To evaluate the quality of the generated motion, we generate 50 motion sequences from each model, where each motion is initialized in fixed initial states. Each motion has a length of 150 frames. To quantify the diversity among these generated motions, we measure the Average Pairwise Distance (APD):

$$\text{APD}(x_i ... x_K) = \frac{1}{K(K-1)} \sum_{i=1}^{K} \sum_{j \neq i}^{K} ||x_i - x_j||, \tag{9}$$

where K is the total number of sequences, and x_i denotes one of the generated motion. A large APD indicates the model can generate diverse motions. However, a limitation of this metric is its tendency to favour models that generate motions with faster velocities, which are more likely to lead to larger joint position differences.

To evaluate the similarity of the generated motion with respect to the original motion data, we calculated the Average Displacement Error (ADE) on the first 60 frames of the same 50 generated motions:

$$\text{ADE}(x_i ... x_K, x_{ref}) = \min_{i \in K} ||x_i - x_{ref}||, \tag{10}$$

where K is the total number of clips, and x_i is the i-th clip of the generated motions, x_{ref} represents the ground truth motion that shared the initial state with the generated motions. ADE quantifies the distance between the distribution of generated motions and the motion data.

To evaluate how close the endpoint of the predicted motion is to endpoint of the ground-truth motion, we calculate the Final Displacement Error (FDE). FDE measures the distance between the joint positions of the generated motion to the ground-truth motion with a prediction horizon of 60 frames. 50 different motions are generated using A-MDM, and the FDE is recorded using the closest sample from the ground-truth trajectory. FDE provides an additional evaluation metric to assess the similiarity between the distribution of generated motions and the ground truth data.

However, APD and ADE alone are not sufficient for a comprehensive evaluation of motion quality. A model can produce a motion sequence with notable artifacts, such as deformation, and still achieve high APD and acceptable ADE scores. To address this issue, we leverage the assumption that the character's bones are rigid bodies. This allows us to gauge the extent of deformation by comparing the bone lengths in generated motions against those in the ground truth mocap data. Specifically, we calculate the average deviation of bone lengths in the generated joint positions from those of a

Table 2. To evaluate the models' generalization capabilities when generating new motions not in the dataset, we use the models to generate continuation motions starting at the last frame of motion clips in the dataset. We compare the models on the AMASS, 100STYLE, and LaFAN1 datasets. (unit: cm).

	APD ↑	FS ↓	Bone.Err ↓	Pen.Freq↓	Pen.Dist↓
AMASS					
MVAE	41.27±4.06	1.52±0.14	2.96±0.42	2.07±0.89%	1.14±0.73
HuMoR	43.26±5.77	1.47±0.16	3.05±0.35	1.89±1.07%	1.01± 0.91
AMDM (Ours)	**59.91**±1.03	**1.10**±0.14	**2.01**±0.06	**0.38**±0.05%	**0.89**± 0.22
100STYLE					
MVAE	59.44±0.94	1.66±0.05	0.26±0.02	1.78±0.85%	0.06± 0.04
HuMoR	67.83±0.82	1.64±0.05	0.23±0.02	1.86±0.95%	0.04± 0.03
AMDM (Ours)	**109.11**±0.07	**1.56**±0.02	**0.20**±0.02	**1.70**±0.38%	**0.03**±0.01
LaFAN1					
MVAE	96.26±16.40	2.15±0.47	0.57±0.20	1.01±0.46%	0.33±0.17
HuMoR	123.78±8.64	2.07±0.05	**0.43**±0.05	1.07±0.24%	0.46±0.21
AMDM (Ours)	**124.46**±12.69	**1.94**±0.01	0.45±0.16	**0.86**±0.08%	**0.30**±0.03

standard skeleton. We refer to this measure as the "Bone Length Error". Through empirical analysis, we have found that this metric is well correlated with motion quality and robustness of the motion synthesis model. Bone Length Error is computed according to:

$$\text{Bone\_Err} = \frac{1}{B}\sum_{b\in B} ||\hat{L_b} - L_b||, \tag{11}$$

where B represents the number of bones in the character's body. $\hat{L_b}$ denotes the length of a bone estimated from the joint positions of the generated motions, and L_b is the ground truth bone length from the dataset.

We additionally record the occurrence of common artifacts in motion synthesis, such as foot penetration frequency (Pen.Freq), average foot penetration distance (Pen.Dist), foot sliding (FS), and joint acceleration (Jnt.Accel). We follow the implementation of Ling et al. [2020] to evaluate foot sliding. For evaluating foot penetration, we adopt the methodology outlined from Rempe et al. [2021].

7.3.1 Random Sampling. In this experiment, we benchmark A-MDM against previous auto-regressive motion models in the context of random synthesis, as outlined in Section 7.3. This comparison is conducted using AMASS, 100STYLE, and our subset of the LaFAN1 datasets. For the evaluation presented in Table 1, initial states are uniformly sampled from each dataset. To evaluate the models' ability to generalize and generate motions not in the original dataset, we introduce an evaluation scheme, termed "Motion Continuation", which assesses a model's ability to extend a ground truth motion sequence. In this setting, initial states are extracted from the last 30 frames of the mocap clips, and the model then generates a motion that extends beyond the end of the original motion clip. The results of this assessment are documented in Table 2. In general, A-MDM is capable of generating motions with greater diversity and higher fidelity compared to other auto-regressive generative models.

7.3.2 Denoising Steps and Inference Time. Next, we investigate the influence of different diffusion steps during inference. The performance of A-MDM trained with different number of denoising steps are summarized in Tables 3 and 4. We observe that models trained on larger datasets tend to require a greater number of steps to reach optimal performance. Given A-MDM's primary objective is for real-time motion synthesis, it is crucial to strike a balance between computational efficiency and motion quality. Through our experiments on 100STYLE, we find that 40 denoising steps offers an effective trade-off between reducing motion artifacts and preserving motion diversity. As result, this 40-step configuration is used in subsequent experiments.

Table 3. Comparison of A-MDM with different number of diffusion steps on 100STYLE. (unit:cm)

Step	APD ↑	ADE ↓	Time(s) ↓	FS ↓
10	96.70	10.44	**0.009**	1.55
20	101.17	10.16	0.012	1.57
30	101.19	**10.31**	0.015	1.55
40	**102.52**	10.36	0.021	**1.53**
50	100.97	10.36	0.026	**1.53**

Table 4. Comparison of A-MDM with different numbers of diffusion steps on the **full** LaFAN1, excluding environment interaction motions. Distance error units are measured in cm.

Step	APD ↑	ADE ↓	Time(ms) ↓	FS ↓
1	53.12	25.05	**0.55**	**1.60**
2	100.43	19.00	1.08	1.72
5	111.78	17.96	2.80	2.06
10	118.93	16.91	5.31	2.02
25	130.34	18.66	12.95	2.11
40	128.91	18.01	20.96	2.22
50	129.32	18.11	26.28	2.15
100	**130.28**	**16.18**	52.78	2.06

Our findings show that the models with the fewest denoising steps exhibits the lowest motion diversity, yet they tend to exhibit

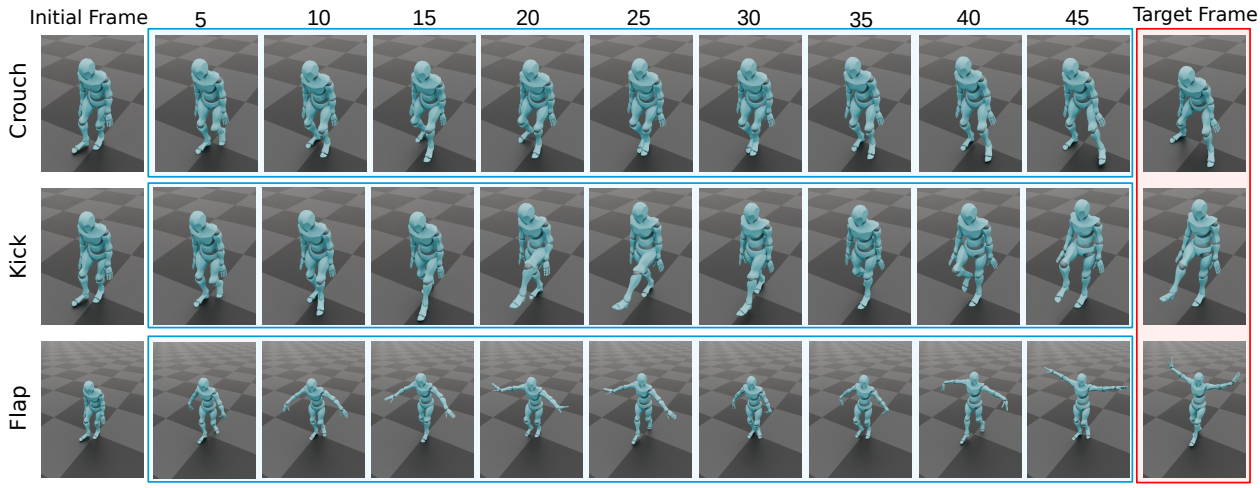
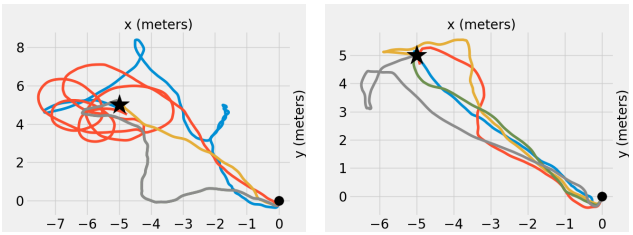


Fig. 9. **Qualitative results for Motion In-betweening via inpainting**. Red represents the target frame, and blue indicates the inbtweening frames. We use a 40-step A-MDM to generate the transition between the same initial frame and different target frames. A-MDM is able to generate natural transitions between keyframes in real-time.



(a) HuMoR (b) A-MDM

Fig. 10. Task-oriented sampling using HuMoR (**Left**) vs. A-MDM (**Right**). The trajectories of A-MDM are more direct and take fewer steps. Results are generated using models trained on 100STYLE.

less foot sliding. Fewer denoising steps are more susceptible to mode-collapse, leading to models that predominantly generates simpler motions like standing and walking, which are less prone to producing artifacts such as bone length distortion or foot sliding. Please refer to our supplementary video for a qualitative comparison of models with different numbers of denoising steps.

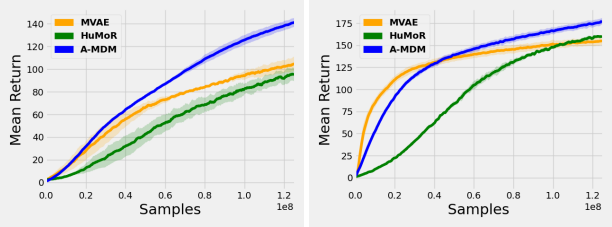
With a larger number of denoising steps, the model exhibits a higher Average Pairwise Distance (APD), indicating an increase in the model's capability to synthesize diverse and complex motions. However, the improvements from increasing the number of denoising steps begin to attenuate, and inference speed takes precedence over further increases in denoising steps.

7.3.3 Task Oriented Sampling. In this section, we present a comparison of the performance of the VAE-based models and A-MDM on the target-reaching task using task-oriented sampling. We conducted five trials for each base model, as depicted in different colors in Figure 10. At each step, the base model generates 100 candidate successor states through random sampling, and the candidate state with the shortest distance to the target is selected as the next frame.

For this experiment, we initialize the character in a fixed pose at the origin (black dots) and set the target at position (-5m, 5m) (black stars) as show in Figure 10. We compare A-MDM with a HuMor model trained on the 100STYLE dataset [Mason et al. 2022]. In Figure 10, the trajectories generated by HuMor are consistent with the findings from Ling et al. [2020], where VAE-based methods tend to wander instead of taking the most direct path to the target, especially when they are close to the target. In contrast, A-MDM is able to follow more direct paths toward the target. This improvement may be attributed to more diverse motions when sampling from A-MDM, thereby providing the character with more flexible options, leading to significantly faster task completion. However, due to its inherent short-sightedness, Task Oriented Sampling often fails to identify the optimal behavior for completing a task. While task-oriented sampling is a useful tool for evaluating auto-regressive motion synthesis models, it may not always yield the best results in practice. In the upcoming experiments, we will explore strategies to leverage the base A-MDM model more effectively to further enhance performance on downstream tasks.

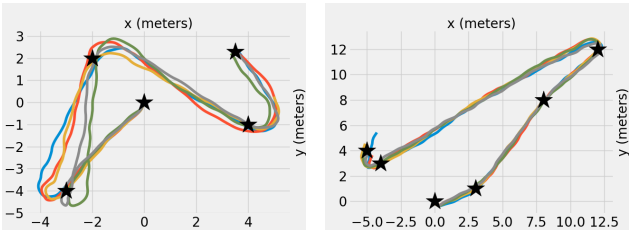
7.4 Hierarchical Control

To evaluate the effectiveness of hierarchical control using different auto-regressive models, we compare the learning curve when training high-level controllers for new tasks. The base model for each method are trained on the same dataset. For the MVAE, the hierarchical controller's training adheres to Ling et al. [2020]. In the case of HuMoR, the hierarchical controller is trained to predict the residual latent of the output of the prior network. For A-MDM, the training of its hierarchical controller follows the procedures detailed in the Section 5. Figure 11 compares the learning curves of



(a) LaFAN1 (b) 100STYLE

Fig. 11. Learning Curve of Target Reaching Hierarchical Controllers. A-MDM achieves higher returns on new tasks compared to the other hierarchical models.



(a) Target set 1 (short distance) (b) Target set 2 (longer distance)

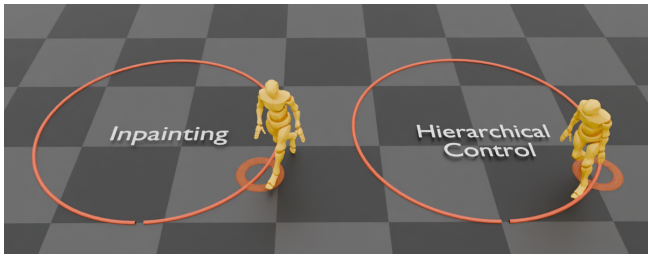
Fig. 12. **Trajectories for Target Reaching with Hierarchical Control.** The stars stand for a fixed set of targets and lines of various colors represent different runs starting from the same initial character state. We show that our hierarchical model is capable of generating diverse motions, with the same initial state and target goals.

the various models. A-MDM is able to achieve consistently higher returns compared to the other methods.

One of the features of the design of our hierarchical controller is that it can preserve the diversity of A-MDM, which allows the hierarchical model to produce a variety of different motions for performing the same task, as illustrated in Figure 12. This is incontrast to the hierarchical models proposed by Ling et al. [2020], which selects deterministic latents for steering the base model and therefore results in deterministic motions that sacrifice much of the diversty of the original base model. This experiment shows that our method can produce diverse motions while being controlled by a hierarchical controller for new tasks.

7.5 Hierarchical Control vs Inpainting

Hierarchical Control and Inpainting are able to perform identical tasks. As shown in Figure 13, with inpainting, the generated motion is restricted to exactly follow the user-specified trajectory. Therefore, if the target trajectory is unnatural, this can lead to unnatural full-body motions. With hierarchical control, the generated motion has more flexibility to deviate from the specific target trajectory as needed to produce more natural behaviors at the cost of not following the target trajectory exactly. More examples are available in the supplementary video.



(a) Comparison between Inpainting and Hierarchical Control in trajectory following

(b) Inpainting (c) Hierarchical Control

Fig. 13. Character trajectory (in white) from inpainting (**b**) and hierarchical control(**c**) when following a user-specified circular trajectory (in red). The trajectory of inpainting matches user's target trajectory exactly, while the hierarchical controller can deviate from the target trajectory as needed in order to produce more natural motions.

7.6 A-MDM vs Space-time Diffusion Models

Recent systems that apply diffusion models to motion synthesis have primarily focused on training space-time models for the conditional synthesis of motion sequences [Karunratanakul et al. 2023; Tevet et al. 2023; Zhang et al. 2022]. This section highlights the key differences between A-MDM and space-time models. We start by comparing the performance of selected auto-regressive models and diffusion-based space-time models. In this experiment, we train our model with features provided by the data preprocessing pipeline of HumanML3D[Guo et al. 2022]. We generally follow the evaluation pipeline of the main experiment of MDM[Tevet et al. 2023] on HumanML3D. We measure Frechet Inception Distance (FID), diversity, and additionally foot skating ratio as in Guided Motion Diffusion (GMD) [Karunratanakul et al. 2023].

FID measures the difference between the distribution of the generated motion and that of the ground truth in learned latent space from a pre-trained motion model. Diversity computes the average distance of generated motions in latent space.

During the evaluation, motions are generated with a fixed length of 196 frames for both auto-regressive and space-time models. Since auto-regressive models require an initial frame, we randomly sample this frame from the motion dataset. For space-time models, we sample motions unconditionally. The key difference between our evaluation and those conducted in the research of space-time models is that we emphasize the quality of unconditional generation, as opposed to text-conditioned generation. As demonstrated in Table5, A-MDM demonstrates the best performance among auto-regressive models. However, space-time models still demonstrate stronger performance across the metrics. This difference may be due to A-MDM

using a smaller number of diffusion steps than space-time models, and simpler network structures (MLP) for real-time synthesis). These factors result in significantly faster inference when evaluated under the standard DDPM sampling procedure without additional speedup techniques.

Table 5. Comparison between generative auto-regressive models and space-time models on HumanML3D [Guo et al. 2022]

Model	FID↓	Diversity→	Foot Skat. Ratio↓
Real	0.002±0.00	9.5002 ± 0.002	-
MDM	0.9157 ± 0.0533	9.0123 ± 0.0602	0.0930 ± 0.0021
GMD	**0.5727** ± 0.0681	**9.1714** ± 0.0789	**0.0657**±0.0016
MVAE	11.2393± 0.1607	6.1503 ± 0.0601	0.4153 ± 0.0025
HuMoR	8.2444±0.2437	7.7396 ± 0.0643	0.1210 ± 0.0011
Ours	1.7435 ± 0.0813	7.8998 ± 0.0638	0.1010 ± 0.0012

A-MDM and other auto-regressive models are trained to perform next-frame prediction. It can therefore generate long-horizon motions even when trained with a dataset that predominantly consists of short clips. In contrast, space-time models are generally trained with a maximum motion length, therefore using these models to generate longer motions is prone to drifting out of distribution from the training data. Please refer to the supplementary video for a comparison of this property.

7.7 A-MDM vs Non-Generative Models

To analyze A-MDM's capability to model diverse and highly dynamic motions, we compare our model with a non-generative auto-regressive model, Neural State Machine (NSM)[Starke et al. 2019]. We first evaluate the architecture and the general design between A-MDM and NSM by comparing the performance in an unconditional synthesis setting. For the sake of a fair comparison, we directly use the same network as NSM and use phase information as the input for the gate network, which predicts the blending weights to the MoE framework. Additionally, we predict the phase value for the next time step in this model, as in NSM. We then train it on the same LaFAN1 dataset (without files concerning obstacles) as A-MDM. We compare the performance between A-MDM with NSM in Table 6. We found that our A-MDM effectively generates diverse motions with fewer artifacts than NSM. Results are available in the supplementary video. The difference in motion quality is particularly evident during a transition from a walk to a quick turn.

Table 6. Comparison between NSM and A-MDM. (unit: cm).

Model	FS↓	Pen.Freq↓	Pen.Dist↓
GT	1.58	0.79 %	0.05
NSM	2.25	0.87 %	0.07
Ours	**1.99**	**0.82** %	**0.04**

8 DISCUSSION AND FUTURE WORK

In this work, we presented an auto-regressive diffusion model for kinematic motion synthesis. Unlike recent diffusion models for motion synthesis, which use space-models to generate motion sequences, we demonstrate that an auto-regressive diffusion model can synthesize high-quality and diverse motions, and can be effectively trained on large motion datasets. To achieve real-time inference, we propose a lightweight architecture and use significantly fewer steps than conventional diffusion models. Once trained, A-MDM can be combined with a variety of different control methods to generate motions for new downstream tasks.

While our designs decisions improve overall stability and motion quality of A-MDM, training auto-regressive diffusion models for generate long-horizon motion is still challenging. A-MDM can occasionally exhibit unstable behaviours and failures under extreme circumstances. For instance, A-MDM is prone to generating motions that exhibit foot sliding and jittering when the user-specified controls are unnatural. We are interested in exploring techniques that can further mitigate these artifacts from A-MDM and improve the robustness of the model across different applications. Our work has focused primarily on single-frame autoregressive models, but for some applications it may be beneficial to use multi-frame models, which predicts a sliding window of frames at a time. This may improve temporal coherence of the generated motions and allow the model tackle more complex tasks. Recent works have also proposed a variety of acceleration techniques for motion diffusion models [Song et al. 2023; Zhang et al. 2023b; Zhou et al. 2023]. Integrating these methods into A-MDM may further improve the runtime performance of these models for real-time applications.

REFERENCES

Emre Aksan, Manuel Kaufmann, and Otmar Hilliges. 2019. Structured prediction helps 3d human motion modelling. In *Proceedings of the IEEE International Conference on Computer Vision*. 7144–7153.

Emad Barsoum, John Kender, and Zicheng Liu. 2017. HP-GAN: Probabilistic 3D human motion prediction via GAN. *CoRR* (2017).

Gustav Bredell, Kyriakos Flouris, Krishna Chaitanya, Ertunc Erdil, and Ender Konukoglu. 2021. Minimizing the Blur Error of Variational Autoencoders. In *Advances in Neural Information Processing Systems 34 (NeurIPS 2021)*.

Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. 2016. OpenAI Gym. arXiv:arXiv:1606.01540

Rishabh Dabral, Muhammad Hamza Mughal, Vladislav Golyanik, and Christian Theobalt. 2023. MoFusion: A Framework for Denoising-Diffusion-based Motion Synthesis. In *CVPR*.

Katerina Fragkiadaki, Sergey Levine, Panna Felsen, and Jitendra Malik. 2015. Recurrent network models for human dynamics. In *Proceedings of the IEEE International Conference on Computer Vision*. 4346–4354.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2020. Generative adversarial networks. *Commun. ACM* 63, 11 (2020), 139–144.

Anand Gopalakrishnan, Ankur Mali, Dan Kifer, Lee Giles, and Alexander G Ororbia. 2019. A neural temporal model for human motion prediction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 12116–12125.

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

Félix G. Harvey, Mike Yurick, Derek Nowrouzezahrai, and Christopher Pal. 2020. Robust Motion In-Betweening. 39, 4 (2020).

Mohamed Hassan, Duygu Ceylan, Ruben Villegas, Jun Saito, Jimei Yang, Yi Zhou, and Michael J Black. 2021. Stochastic scene-aware motion prediction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 11374–11384.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. 2020. Denoising diffusion probabilistic models. *NIPS* (2020).

Daniel Holden, Taku Komura, and Jun Saito. 2017. Phase-functioned neural networks for character control. *ACM Transactions on Graphics (TOG)* 36, 4 (2017), 1–13.

Siyuan Huang, Zan Wang, Puhao Li, Baoxiong Jia, Tengyu Liu, Yixin Zhu, Wei Liang, and Song-Chun Zhu. 2023. Diffusion-based Generation, Optimization, and Planning in 3D Scenes. *CVPR* (2023).

Korrawe Karunratanakul, Konpat Preechakul, Supasorn Suwajanakorn, and Siyu Tang. 2023. Guided Motion Diffusion for Controllable Human Motion Synthesis. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2151–2162.

Jihoon Kim, Taehyun Byun, Seungyoun Shin, Jungdam Won, and Sungjoon Choi. 2022. Conditional motion in-betweening. *Pattern Recognition* 132 (2022), 108894.

Diederik P Kingma and Max Welling. 2013. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114* (2013).

Peizhuo Li, Kfir Aberman, Zihan Zhang, Rana Hanocka, and Olga Sorkine-Hornung. 2022. GANimator: Neural Motion Synthesis from a Single Sequence. *ACM Transactions on Graphics (TOG)* 41, 4 (2022), 138.

Zimo Li, Yi Zhou, Shuangjiu Xiao, Chong He, Zeng Huang, and Hao Li. 2017. Auto-conditioned recurrent networks for extended complex human motion synthesis. *arXiv preprint arXiv:1707.05363* (2017).

Hung Yu Ling, Fabio Zinno, George Cheng, and Michiel van de Panne. 2020. Character Controllers Using Motion VAEs. *ACM Trans. Graph.* 39, 4 (2020).

Andreas Lugmayr, Martin Danelljan, Andres Romero, Fisher Yu, Radu Timofte, and Luc Gool. 2022. RePaint: Inpainting using Denoising Diffusion Probabilistic Models. (2022).

Jianxin Ma, Shuai Bai, and Chang Zhou. 2022. Pretrained Diffusion Models for Unified Human Motion Synthesis. *arXiv* (2022).

Naureen Mahmood, Nima Ghorbani, Nikolaus F Troje, Gerard Pons-Moll, and Michael J Black. 2019. AMASS: Archive of motion capture as surface shapes. In *ICCV*.

Julieta Martinez, Michael J Black, and Javier Romero. 2017. On human motion prediction using recurrent neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2891–2900.

Ian Mason, Sebastian Starke, and Taku Komura. 2022. Real-Time Style Modelling of Human Locomotion via Feature-Wise Transformations and Local Motion Phases. *Proceedings of the ACM on Computer Graphics and Interactive Techniques* 5, 1, Article 6 (may 2022). https://doi.org/10.1145/3522618

Qianhui Men, Hubert P.H. Shum, Edmond S.L. Ho, and Howard Leung. 2022. GAN-Based Reactive Motion Synthesis with Class-Aware Discriminators for Human–Human Interaction. *Comput. Graph.* 102, C (feb 2022), 634–645. https://doi.org/10.1016/j.cag.2021.09.014

Dario Pavllo, David Grangier, and Michael Auli. 2018. Quaternet: A quaternion-based recurrent model for human motion. *arXiv preprint arXiv:1805.06485* (2018).

Xue Bin Peng, Pieter Abbeel, Sergey Levine, and Michiel van de Panne. 2018. Deepmimic: Example-guided deep reinforcement learning of physics-based character skills. *ACM Transactions on Graphics (TOG)* 37, 4 (2018), 1–14.

Xue Bin Peng, Yunrong Guo, Lina Halper, Sergey Levine, and Sanja Fidler. 2022. ASE: Large-scale Reusable Adversarial Skill Embeddings for Physically Simulated Characters. *ACM Trans. Graph.* 41, 4, Article 94 (July 2022).

Xue Bin Peng, Ze Ma, Pieter Abbeel, Sergey Levine, and Angjoo Kanazawa. 2021. AMP: Adversarial Motion Priors for Stylized Physics-Based Character Control. *ACM Trans. Graph.* 40, 4, Article 1 (July 2021), 15 pages. https://doi.org/10.1145/3450626.3459670

Mathis Petrovich, Michael J Black, and Gül Varol. 2021. Action-conditioned 3D human motion synthesis with transformer VAE. In *ICCV*. 10985–10995.

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

Alexander Nichol Prafulla Dhariwal. 2021. Diffusion Models Beat GANs on Image Synthesis. In *Advances in Neural Information Processing Systems 34 (NeurIPS 2021)*.

Sigal Raab, Inbal Leibovitch, Guy Tevet, Moab Arar, Amit H Bermano, and Daniel Cohen-Or. 2023. Single Motion Diffusion. *arXiv* (2023).

Davis Rempe, Tolga Birdal, Aaron Hertzmann, Jimei Yang, Srinath Sridhar, and Leonidas J Guibas. 2021. Humor: 3d human motion model for robust pose estimation. In *ICCV*.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. 2022. High-Resolution Image Synthesis with Latent Diffusion Models.

Nataniel Ruiz, Yuanzhen Li, Varun Jampani, Yael Pritch, Michael Rubinstein, and Kfir Aberman. 2022. DreamBooth: Fine Tuning Text-to-image Diffusion Models for Subject-Driven Generation. (2022).

Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily L Denton, Kamyar Ghasemipour, Raphael Gontijo Lopes, Burcu Karagol Ayan, Tim Salimans, et al. 2022. Photorealistic text-to-image diffusion models with deep language understanding. *NIPS* (2022).

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347* (2017).

Yonatan Shafir, Guy Tevet, Roy Kapon, and Amit H Bermano. 2023. Human Motion Diffusion as a Generative Prior. *arXiv* (2023).

Jiaming Song, Chenlin Meng, and Stefano Ermon. 2021. Denoising diffusion implicit models. *ICLR* (2021).

Yang Song, Prafulla Dhariwal, Mark Chen, and Ilya Sutskever. 2023. Consistency Models. *arXiv preprint arXiv:2303.01469* (2023).

Sebastian Starke, He Zhang, Taku Komura, and Jun Saito. 2019. Neural state machine for character-scene interactions. *ACM Trans. Graph.* 38, 6 (2019), 209–1.

Chen Tessler, Yoni Kasten, Yunrong Guo, Shie Mannor, Gal Chechik, and Xue Bin Peng. 2023. CALM: Conditional Adversarial Latent Models for Directable Virtual Characters. *arXiv preprint arXiv:2305.02195* (2023).

Guy Tevet, Sigal Raab, Brian Gordon, Yonatan Shafir, Daniel Cohen-Or, and Amit H Bermano. 2023. Human motion diffusion model. *ICLR* (2023).

Jingbo Wang, Yu Rong, Jingyuan Liu, Sijie Yan, Dahua Lin, and Bo Dai. 2022. Towards diverse and natural scene-aware 3d human motion synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 20460–20469.

Jingbo Wang, Sijie Yan, Bo Dai, and Dahua Lin. 2021. Scene-aware generative network for human motion synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 12206–12215.

Jungdam Won, Deepak Gopinath, and Jessica Hodgins. 2022. Physics-based character controllers using conditional VAEs. *ACM Transactions on Graphics (TOG)* 41, 4 (2022), 1–12.

Chen Xin, Biao Jiang, Wen Liu, Zilong Huang, Bin Fu, Tao Chen, Jingyi Yu, and Gang Yu. 2023. Executing your Commands via Motion Diffusion in Latent Space. In *CVPR*.

Sijie Yan, Zhizhong Li, Yuanjun Xiong, Huahan Yan, and Dahua Lin. 2019. Convolutional sequence generation for skeleton-based action synthesis. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 4394–4402.

Heyuan Yao, Zhenhua Song, Baoquan Chen, and Libin Liu. 2022. ControlVAE: Model-Based Learning of Generative Controllers for Physics-Based Characters. *ACM Trans. Graph.* 41, 6, Article 183 (nov 2022), 16 pages. https://doi.org/10.1145/3550454.3555434

Mingyuan Zhang, Zhongang Cai, Liang Pan, Fangzhou Hong, Xinying Guo, Lei Yang, and Ziwei Liu. 2022. MotionDiffuse: Text-Driven Human Motion Generation with Diffusion Model. *arXiv* (2022).

Mingyuan Zhang, Xinying Guo, Liang Pan, Zhongang Cai, Fangzhou Hong, Huirong Li, Lei Yang, and Ziwei Liu. 2023a. ReMoDiffuse: Retrieval-Augmented Motion Diffusion Model. *arXiv* (2023).

Zihan Zhang, Richard Liu, Kfir Aberman, and Rana Hanocka. 2023b. TEDi: Temporally-Entangled Diffusion for Long-Term Motion Synthesis. arXiv:2307.15042 [cs.CV]

Wenyang Zhou, Zhiyang Dou, Zeyu Cao, Zhouyingcheng Liao, Jingbo Wang, Wenjia Wang, Yuan Liu, Taku Komura, Wenping Wang, and Lingjie Liu. 2023. EMDM: Efficient Motion Diffusion Model for Fast, High-Quality Motion Generation. *arXiv preprint arXiv:2312.02256* (2023).

Yi Zhou, Connelly Barnes, Lu Jingwan, Yang Jimei, and Li Hao. 2019. On the Continuity of Rotation Representations in Neural Networks. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.