# Controllable 3D Character Motion Generation

by

Yuxuan Mu

A thesis submitted in partial fulfillment of the requirements for the degree of

Master of Science
in
Signal and Image Processing

Department of Electrical & Computer Engineering

University of Alberta

# Abstract

3D character animation plays a pivotal role in video games and film productions. One crucial aspect that captivates audiences is the visual appeal and fluidity of character motion within these digital mediums. However, creating realistic character motion requires significant efforts and resources, including access to Motion Capture (MoCap) studios and skilled animation artists. The capability to automatically generate 3D character motion as desired would greatly empower the production of 3D digital content. Moreover, by learning to generate lifelike character motions, we can gain deeper insights into the physical motor behavior of real humans. Ultimately, this could lead to the creation of digital dynamic clones of ourselves. In this thesis, we make a step towards solving this problem, to generate lively character motion as wishes. We present controllable 3D character motion generation techniques that could create human-like motion following versatile instructions, such as natural language commands, motion examples, game-pad signals, and terrain environments.

We begin by presenting a novel generative framework that enables general text-to-motion generation. A neural network learns to generate corresponding 3D skeletal animation based on the given descriptive natural language sentence. The network not only learns the text-conditioned dataset distribution but also models the unconditional motion prior. By utilizing this learned generative prior, this method can produce robust and life-like motion and extend any given motion clips. We then integrate the text-to-motion framework with

the off-the-shelf stylization method to generate even more expressive motions. Furthermore, we develop a hierarchy reinforcement learning system that can enable the digital character to perform specific motions with spontaneous interaction in a physically simulated environment. Two policies are trained in an end-to-end manner to explicitly leverage the motion expert databases through retrieval and drive the simulated character with torques. The agents trained in the physical simulator inherently learn to interact and be resilient against perturbations from the environment.

# Preface

This thesis is an original work by Yuxuan Mu.

Chapter 4 of this thesis has been published under the title [28] Chuan Guo*, **Yuxuan Mu***, Muhammad Gohar Javed*, Sen Wang, Li Cheng. "MoMask: Generative Masked Modeling of 3D Human Motions." In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 1900-1910. 2024. "*" indicates equal contributions. The research problem was first presented by Chuan Guo and extended by me. I was the primary contributor to the technical design, implementation and paper writing. Muhammad Gohar Javed conducted the parameter sweeping experiments on the KIT dataset. Chuan Guo supported me with paper writing, ablation experiments, and visualization.

Chapter 5 of this thesis has been accepted for The 2024 IEEE International Conference on Multimedia and Expo (ICME) and will be published under the title [72] **Yuxuan Mu**, Shihao Zou, Kangning Yin, Zheng Tian, Li Cheng, Weinan Zhang, Jun Wang. "RACon: Retrieval-Augmented Simulated Character Locomotion Control." In the 2024 IEEE International Conference on Multimedia and Expo (ICME), 2024. This research problem was identified through the collaboration of Zheng Tian, Weinan Zhang, Li Cheng and me. I was the principal contributor to the simulation setup, technical design, experiments, and paper writing. Kangning Yin helped me run experiments on baseline methods. Shihao Zou helped me with the paper revision.

*To my family.*

*We choose to go to the Moon in this decade and do the other things, not because they are easy, but because they are hard.*

– John F. Kennedy, 1962.

# Acknowledgements

I would like to express my deepest gratitude to my supervisor, Dr. Li Cheng, for his exceptional mentoring and unwavering support during my time at the University of Alberta. Your passion and dedication to rigorous research in computer vision have been a constant source of inspiration, driving me to strive for excellence in my work. Your guidance has shaped my academic journey and will continue to influence my future endeavors.

I extend my heartfelt thanks to my committee members, Dr. Di Niu and Dr. Xingyu Li for their invaluable feedback and insightful advice. Your constructive comments have played a crucial role in refining and strengthening my thesis, making it a more solid and insightful piece of work.

I am also grateful to my colleagues in the Vision and Learning Lab, particularly Shihao Zou, Chuan Guo, Muhammad Gohar Javed, and all others who have made my master's journey joyful and inspiring. Your camaraderie, support, and collaborative spirit have enriched my learning experience and motivated me to push the boundaries of my research.

Finally, I want to acknowledge and thank my family for their unconditional love, encouragement, and unwavering support throughout my life. Your constant belief in me has been an enduring source of inspiration and courage, propelling me forward in both my academic and personal endeavors.

# Contents

# List of Tables

# List of Figures

# Chapter 1

# Introduction

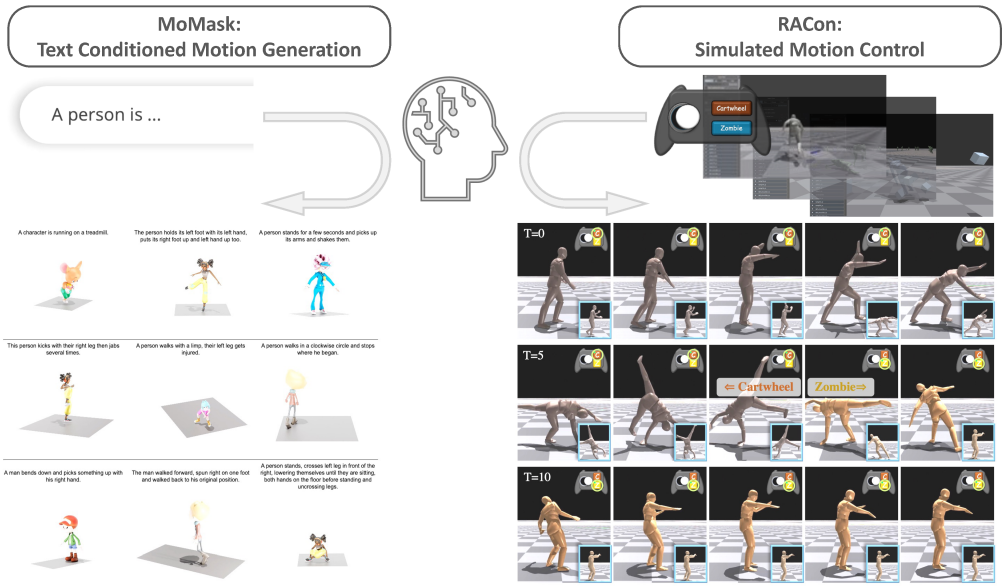



Figure 1.1: Overview of controllable motion generation in this thesis. Our works generate motion conditioned on text descriptions, game-pad inputs, motion examples and simulated environments.

Human motions can convey abundant information in daily life through subtle variance of actions, even more effectively than language. Beyond real-world scenario, human motions play a crucial role in creating lifelike characters and animations in digital media , significantly enhancing our everyday entertainment. However, people are highly sensitive to artifacts in human motions such as unreasonable joint rotations, trembling, and sliding, which can be perceived as unrealistic and distracting. Early approaches to human motion focused on developing powerful regression methods in a data-driven manner but often

overlooked the stochastic nature of human motion. Humans can perform diverse motions in response to the same instruction. A modern perspective views motion modeling as a process of learning data distribution. By modeling the pure dataset prior, the model learns the natural and reasonable appearance of human motion. When learning from conditional data distributions with additional control prompts, the model can generate motion in a controllable manner.

In this thesis, we adopt generative modeling approaches to develop intelligent 3D skeletal motion performers by leveraging prior knowledge from human motion data. At the core of our work is the generative learning paradigm, which emphasizes the stochastic property of motions. Our works learn the nature of motion distribution and its variations conditioned on text descriptions, game-pad inputs, and motion examples within kinematics or physically constrained dynamics environment, as illustrated in Fig. 1.1. Fig. 1.4 demonstrated some applications from the community built with our works. We believe that controllable 3D character motion generation in both contexts could be of interest to other domains, particularly in applications such as augmented reality, virtual reality and robotics.

## 1.1 Motivation

Human motions are inherently complex and captivating, involving a wide array of skills and tasks. Even minor variations in these movements can convey a wealth of information. In the realm of 3D digital characters, replicating such lifelike motions typically requires the expertise of skilled actors or artists to meticulously capture every detail. This leads to intriguing questions: Can we create 3D characters that can follow simple language instructions just like humans? Furthermore, can we broaden this capability to control them through various inputs, such as motion examples or game-pads?

Addressing these challenges is crucial, as achieving versatile and controllable 3D character motion generation has profound implications for augmented reality, virtual reality, and robotics. Our approach leverages generative model-

ing techniques, which are pivotal for capturing the stochastic nature of human motions and learning from extensive motion data. By using state-of-the-art generative learning paradigms, we aim to develop methods that can generate lifelike movements under different control signals. Specifically, our work focuses on generative modeling, which emphasize the non-determinism of human motion. We utilize advanced methods such as masked transformers for generative motion modeling and residual vector quantization for compact yet almost lossless latent representation. Additionally, we propose a hierarchical reinforcement learning framework with adversarial training in physics simulators, ensuring legal dynamics.

Through these innovative methodologies, we seek to push the boundaries of digital character animation, making it more accessible, realistic, and adaptable to a wide range of applications. Our goal is to develop 3D characters that not only mimic human movements but also respond dynamically to diverse control signals, thereby advancing the field of digital character animation.

## 1.2 Thesis Overview

The work presented in this thesis explores 3D human character motion generation conditioned on text and game-pad-controlled simulated environment. We begin with a brief background review of the fundamental concepts in generative models, reinforcement learning, imitation learning, and 3D character motion (Chapter 2). Then we give an overview of related techniques on latent representation, generative mask modeling, and retrieval augmented generation. We also review the most closely related work in motion matching, generative human motion modeling, and simulated character control (Chapter 3). This is followed by the two main works of this thesis: MoMask (Chapter 4) and RACon (Chapter 5). The kinematic-based motion generation method MoMask (Chapter 4) develops the state-of-the-art text2motion framework to generate life-like character motion from text descriptions, shown in Fig. 1.2. Our MoMask, when provided with a text input, generates high-quality 3D character motion with diversity and precise control over subtleties such as "*two strides*

*forward*", "*pivot on left foot*", and "*pivot swiftly*". Our proposed masked transformer for generative motion modeling is much swifter and born with frame editing capacity, which reveals the unique characteristics of motion compared with natural language and image. Moreover, on the value continuity of motion representation, a residual vector quantization variational autoencoder is adopted to obtain compact yet almost lossless latent representation. It enables easier generative modeling and ensures good motion quality. Next, in simulated character control with game-pad (Chapter 5), we propose a hierarchical reinforcement learning framework to combine a novel task-oriented motion retrieval and a model-based controller in an end-to-end manner. Thanks to the end-to-end learnable retriever, we can control the locomotion of characters seamlessly using a game-pad and even switch moving styles in real-time by switching the retrieval databases, shown in Fig. 1.1. Additionally, the proposed retrieval-augmented generative adversarial imitation learning pipeline further improves and stabilizes the training. Finally, in Chapter 6, we conclude with a discussion of the limitations of our methods and suggest possible directions for future work .

## 1.3 MoMask: Generative Masked Modeling for Text-to-Motion Generation


Figure 1.2: Overview motion generation results from MoMask.

MoMask is a novel masked modeling framework for text-driven 3D human motion generation. MoMask employs a hierarchical quantization scheme

to represent human motion as multi-layer discrete motion tokens with high-fidelity details. Starting at the base layer with a sequence of motion tokens obtained by vector quantization, residual tokens of increasing orders are derived and stored at subsequent layers of the hierarchy. For the base-layer motion tokens, a Masked Transformer predicts randomly masked motion tokens conditioned on text input during training. During inference, the Masked Transformer iteratively fills in missing tokens, starting from an empty sequence. Extensive experiments demonstrate that MoMask outperforms state-of-the-art methods in text-to-motion generation, and gains superior performance against the strongest GPTs-based and diffusion-based counter parts. Diffusion Models are not well-suited for sequence generation, while GPTs tend to be overly cumbersome. MoMask can be understood as a nonlinear language model, potentially representing the best modeling paradigm for motion in recent times.

## 1.4 RACon: Retrieval-Augmented Simulated Character Locomotion Control

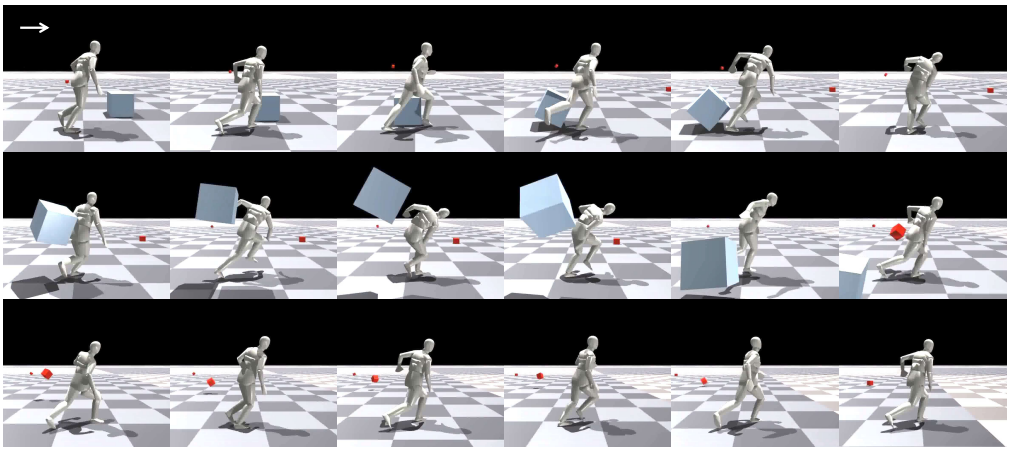


Figure 1.3: Example of a simulated character running against physical perturbation from the environment, controlled by RACon.

RACon is an advanced hierarchical reinforcement learning system designed to enhance the realism and responsiveness of simulated character motion. By integrating a retriever and a motion controller, RACon utilizes task-oriented retrieval to search for motion experts from a user-specified database, which

are then used alongside manipulation signals to drive the character. The system features a retrieval-augmented discriminator to stabilize training and maintain the naturalness of the motions. Key contributions of RACon include an end-to-end integrated approach that incorporates two RL frameworks as an HRL system, a retrieval-augmented (RA) discriminator as a training-time motion prior, and a set of rewards designed to jointly optimize the policies. Empirical studies show RACon outperforms existing methods in both quality and adaptability of locomotion control, enabling seamless transitions between different motion types without extensive tuning.

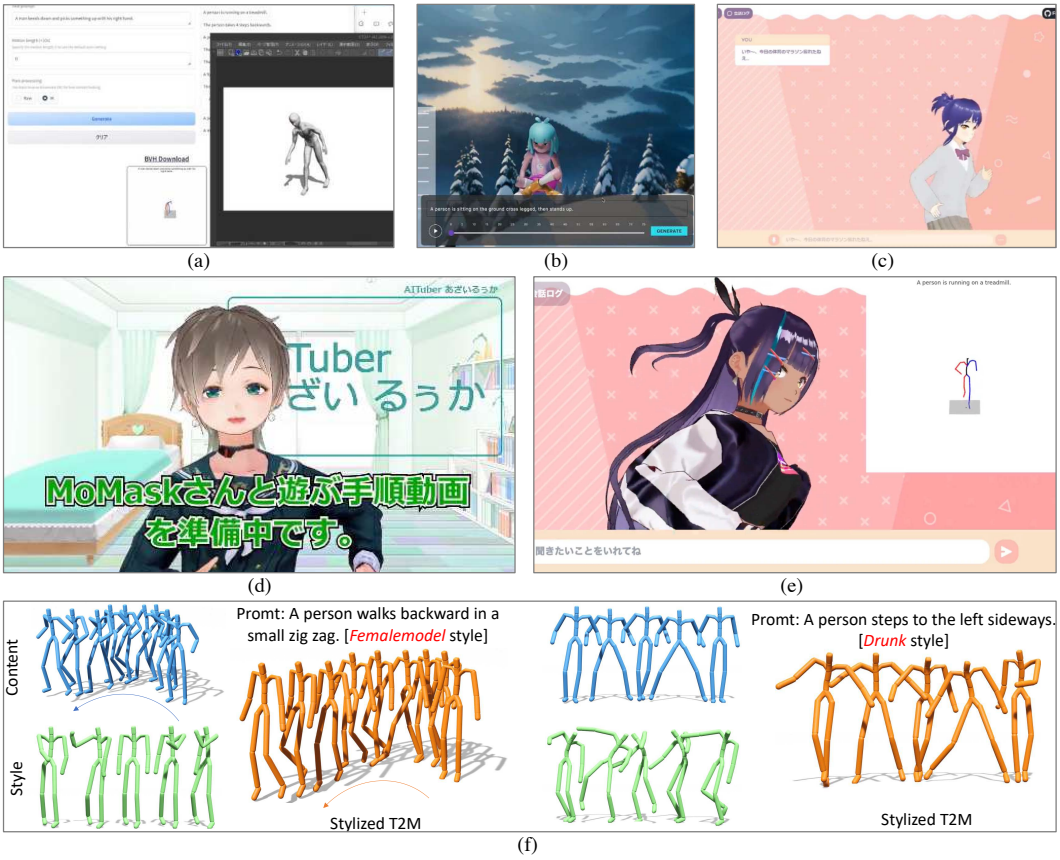



Figure 1.4: Applications built on the work in this thesis by the community[1] and our lab [29]. (a) The motion generated by our work being used as a plugin in ClipStudio, a popular software for digital creation; (b)(c)(d)(e) web games developed by individual creators using our API; (f) an application to generate stylized motion from text, built by our lab team.

[1]https://x.com/tnksoft; NickFisherAU; tegnike; azailuhca; npaka123.

# Chapter 2

# Background

This thesis leverages supervised learning, reinforcement learning, and imitation learning to train generative models capable of understanding and replicating the distribution of motion, thereby generating 3D character motion as desired. In this chapter, we provide a brief overview of the fundamental concepts in generative models, reinforcement learning, imitation learning, and 3D character motion.

## 2.1 Generative Models

A generative model is a statistical model of the joint probability distribution $P(X, Y)$ over a given observable variable $X$ and target variable $Y$ [73]. Such a model can be employed to 'generate' random instances of an observation $x$ [70].

In recent years, the field of deep learning has given rise to a popular trend of methods known as deep generative models (DGMs). These methods blend generative models with deep neural networks. Achieving high performance with DGMs often requires scaling up both the neural networks and the associated training datasets. Some of the most well-known DGMs are variational autoencoders (VAEs) [43], generative adversarial networks (GANs) [26], autoregressive models [84], and diffusion models [37]. A notable recent development in this field is the push towards creating larger and more efficient deep generative models.

## 2.2 Reinforcement Learning

Reinforcement Learning (RL) is a type of machine learning where an agent learns to make decisions by interacting with an environment, which is an external system. The objective is to learn a policy that maximizes cumulative rewards over time [91]. It is typically modeled as a Markov Decision Process (MDP), characterized by:

**States ($\mathcal{S}$)**: The set of all possible states, which is a representation of the current situation in the environment.

**Actions ($\mathcal{A}$)**: The set of all possible actions, which are decisions made by the agent that affect the environment.

**Transition Function**: The probability of transitioning from one state to another, given an action:

$$\mathcal{T}(s', s, a) = \Pr(S_{t+1} = s' \mid S_t = s, A_t = a), \tag{2.1}$$

where $S_t$ is the state at time $t$, $A_t$ is the action taken at time $t$, and $S_{t+1}$ is the next state.

**Reward Function**: The immediate reward received after transitioning from state $s$ to state $s'$ under action $a$:

$$r(s', s, a). \tag{2.2}$$

A basic reinforcement learning agent interacts with its environment in discrete time steps. At each time $t$, the agent receives the current state $S_t$ and reward $r_t$. It then chooses an action $A_t$ from the set of available actions, which is subsequently sent to the environment. The environment moves to a new state $S_{t+1}$ and the reward $r_{t+1}$ associated with the transition $(S_t, A_t, S_{t+1})$ is determined [91]. The goal of a reinforcement learning agent is to learn a policy, which is a strategy used by the agent to decide actions based on the current state:

$$\pi : \mathcal{S} \times \mathcal{A} \to [0, 1], \tag{2.3}$$

where

$$\pi(s, a) = \Pr(A_t = a \mid S_t = s), \tag{2.4}$$

that maximizes the expected cumulative return. The objective is to find the optimal policy $\pi^*$ can be written as:

$$\pi^* = \arg\max_{\pi} J(\pi), \tag{2.5}$$

where the expected discounted return $J(\pi)$ could be formulated as:

$$J(\pi) = \mathbb{E}_{\tau \sim p(\tau|\pi)} \left[ \sum_{t=0}^{T-1} \gamma^t r_t \right], \tag{2.6}$$

$p(\tau|\pi) = p(s_0) \prod_{t=0}^{T-1} p(s_{t+1}|s_t, a_t)\pi(a_t|s_t)$ is the distribution of trajectories $\tau$ induced by a policy $\pi$, and $\gamma \in [0, 1]$ is a discount factor.

### 2.2.1 Goal-Conditioned Reinforcement Learning

In Goal-Conditioned Reinforcement Learning [58], an agent interacts with the *environment* via the *policy* $\pi(a_t|s_t, g)$, aiming to accomplish a specific *goal* $g \sim p_g$, with $p_g$ representing the desired goal distribution. The reward is then typically determined by the function $r(s_t, a_t, g)$, which assesses the action based on the current state and the goal. The learning objective entails maximizing the cumulative return $J(\pi)$, which can be expressed as:

$$J(\pi) = \mathbb{E}_{\substack{a_t \sim \pi(\cdot|s_t,g), g \sim p_g \\ s_{t+1} \sim \mathcal{T}(\cdot|s_t,a_t)}} \left[ \sum_{t} \gamma^t r(s_t, a_t, g) \right], \tag{2.7}$$

where $\mathcal{T}(\cdot|s_t, a_t)$ represents the environment's forward dynamic distribution and $\gamma$ stands for the discount factor.

### 2.2.2 Deep Reinforcement Learning

Deep Reinforcement Learning (DRL) combines RL with deep learning, allowing the agent to handle high-dimensional state spaces, such as images. Neural networks are used to approximate the value functions or policies [71]. The two typical algorithm families for deep reinforcement learning are Deep Q-Networks (DQNs) and Policy Gradient Methods. DQN extends Q-learning using deep neural networks to approximate the Q-value function. This approach has been successful in various domains, including playing Atari games

at superhuman levels [71]. Policy gradient methods learn the policy directly by optimizing the expected cumulative reward using gradients. These methods are effective in continuous action spaces and complex environments. A typical policy gradient method, Proximal Policy Optimization, is used in the work of this thesis thanks to its robustness and efficacy.

### Proximal Policy Optimization (PPO)

Proximal Policy Optimization (PPO) is a type of policy gradient method that aims to improve training stability and efficiency. PPO strikes a balance between policy optimization and preventing too-large updates, which can destabilize training [87]. The three-key design of PPO leads to its stable and efficient performance. Firstly, PPO modifies the policy gradient objective to ensure the new policy does not deviate too much from the old policy. The objective function includes a clipping term that limits the policy update size:

$$L^{\text{CLIP}}(\theta) = \mathbb{E}_t \left[ \min \left( r_t(\theta)\hat{A}_t, \text{clip}(r_t(\theta), 1-\epsilon, 1+\epsilon)\hat{A}_t \right) \right], \tag{2.8}$$

where $r_t(\theta) = \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{old}}(a_t|s_t)}$ is the probability ratio of the new and old policies, $\hat{A}_t$ is the advantage estimate, and $\epsilon$ is a small hyperparameter that controls the clipping range. Secondly, the clipped objective is used which is a surrogate for the true objective, providing a pessimistic estimate of the expected reward to ensure safe updates. Lastly, PPO uses generalized advantage estimation (GAE) to reduce variance and improve the efficiency of the advantage estimates:

$$\hat{A}_t = \sum_{l=0}^{\infty} (\gamma\lambda)^l \delta_{t+l}, \tag{2.9}$$

where $\delta_t$ is the temporal difference error, $\gamma$ is the discount factor, and $\lambda$ is a parameter that controls the bias-variance trade-off.

PPO makes multiple updates using the same batch of data, improving sample efficiency. Moreover, the clipping mechanism ensures stable and reliable policy updates. In summary, it is a robust and efficient DRL algorithm that balances exploration and exploitation while ensuring stable policy updates, making it a popular choice for various RL tasks.

## 2.3 Imitation Learning

Imitation learning involves training an agent to replicate the behavior of an expert by observing examples of the expert's actions. This process can be categorized into two main approaches: behavioral cloning and inverse reinforcement learning (IRL).

**Behavioral Cloning.** This approach treats imitation as a supervised learning problem, where the goal is to learn a policy that maps states to actions based on expert demonstrations [97]. While straightforward, behavioral cloning often requires large amounts of data to avoid compounding errors due to covariate shift, where the states encountered during training differ from those encountered during deployment.

**Inverse Reinforcement Learning (IRL).** IRL involves inferring a cost function that explains the expert's behavior and then using reinforcement learning to derive a policy that optimizes this cost function [117]. Although IRL can handle more complex scenarios and reduce compounding errors, it is computationally intensive due to the need for repeated reinforcement learning in an inner loop.

### 2.3.1 Generative Adversarial Imitation Learning

Generative Adversarial Imitation Learning (GAIL) [36] emerges as a powerful approach to teaching agents to perform tasks by imitating expert behaviors without requiring direct reinforcement signals or continuous interaction with the expert. GAIL combines the strengths of generative adversarial networks (GANs) from deep learning with the principles of imitation learning to directly learn policies from expert demonstrations without explicitly modeling the cost function. The key idea is to draw an analogy between imitation learning and GANs, where the learning process involves a game between two entities: a generator (policy) and a discriminator.

**Generator (Policy).** The generator attempts to produce behaviors that are indistinguishable from those of the expert. It generates actions based on the current state to maximize the likelihood of being perceived as an expert

by the discriminator.

**Discriminator.** The discriminator distinguishes between the expert's behavior and the generator's behavior. It provides feedback to the generator, helping it improve by penalizing actions that deviate from the expert's behavior.

The interaction between the generator and the discriminator forms an adversarial process, where the generator improves its policy by learning from the discriminator's feedback, and the discriminator becomes better at distinguishing between expert and generated behaviors.

In GAIL, the objective is to find a policy $\pi$ that minimizes the discrepancy between the occupancy measures of the expert and the generated trajectories. Formally, this involves optimizing the following objective:

$$\min_{\pi} \max_{D} \mathbb{E}_{\pi_E}[\log D(s,a)] + \mathbb{E}_{\pi}[\log(1 - D(s,a))] \tag{2.10}$$

where $D$ is the discriminator, $\pi_E$ is the expert policy, and $\pi$ is the generator policy. The discriminator tries to maximize the likelihood of correctly identifying expert actions while minimizing the likelihood of incorrectly identifying generated actions. Conversely, the generator tries to fool the discriminator by producing actions that the discriminator cannot distinguish from the expert's actions.

## 2.4 3D Character Motion

3D character motion, which is a sequence with spatial and temporal dimensions to represent the animation of 3D characters. More specifically, we focus on the motion of articulated subjects, such as humans or humanoids. There are two perspectives on this modality: kinematic solutions and simulator-based (dynamic simulation) methods. Most of the kinematic solutions focus on directly modeling the spatial state, while simulator-based methods often additionally formulate second-order systems with hard constraints from the dynamic functions.

### 2.4.1 Rotation-based Motion Representation

The 3D skeletal character motion is a sequence of 3D poses continuously arranged along the temporal dimension. The 3D pose could be simply represented by a set of 3D key points or joint positions for humans. With regard to the skeletal motion, each pose is further regularized by a fixed skeleton tree as a hard constraint. To feasibly satisfy this constraint, the 3D skeletal pose could be parameterized $SE(3)$ transformations. To be specific, each rigid body (link) is described by the **Special Euclidean Group**:

$$SE(3) := \left\{ T := \begin{bmatrix} R & p \\ 0 & 1 \end{bmatrix} \mid R \in SO(3), p \in \mathbb{R}^3 \right\} \subset \mathbb{R}^{4\times 4} \tag{2.11}$$

where, $p$ is the translation of the link, the pose $T$ of a rigid body in the world frame specifies a transformation from the body frame to the world frame or its parent body frame.

For this rotation-based pose representation, given a kinematic chain composed of links and joints with multiple degree of freedom, we could use **Forward Kinematics** to find the position and orientation of any joints and the end-effector in the word frame when all the joint rotation parameters are known.

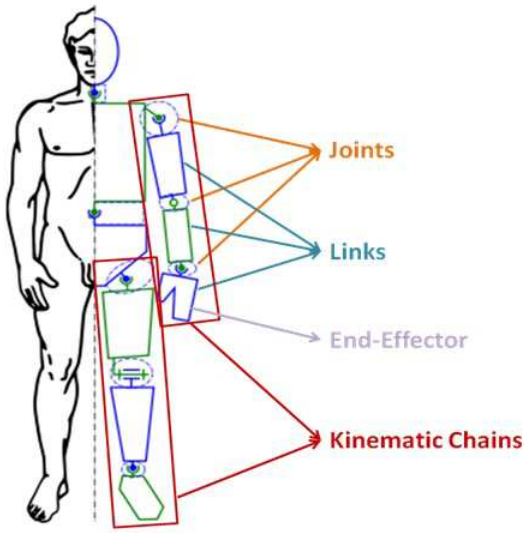



Figure 2.1: Example to illustrate Joints, Links, End-effector, Kinematic chains for 3D skeletal motion [23].

We give a brief introduction to key concepts of rotation-based motion representation, which could also refer to robotics and Fig. 2.1.

- Links: Links are rigid bodies that give structure to the articulated character.
- Joints: Joints are the movable parts (actuators) of the articulated character that connect the links of the character. Joints cause relative motion between adjacent links.
- Kinematic chains: Kinematic chain is the assembly of links connected by joints to produce a desired motion. Human arm is an example of a kinematic chain. Human body is a group of kinematic chains connected in series.
- Degree of Freedom (DOF): Degree of freedom in robotics is simply the total number of independent joints which can change the pose of the robot. If we take a human arm from shoulder to palm (fingers not included), arm has 7 DOF. For instance, the shoulder has 3 DOF, the elbow has 1 DOF, and the wrist has 3 DOF. However, for simplicity, every human joint has 3 DOF if not otherwise specified.
- End-Effector: The end-effector is the last link of the manipulator interacting with the environment in robotics.

### 2.4.2 Articulated Character Simulation

A physics engine is a quadratic program that provides an approximate simulation of certain physical systems, such as rigid body dynamics, soft body dynamics, and fluid dynamics [69]. We will focus on physics engines for rigid body dynamics, collision, and external force (e.g., gravity). Simulators play a key role in training physically plausible motion performers and robots. Popular physics simulator such Isaac Gym [66], MuJoCo [96], PyBullet [17], DART [45], Drake [92] use numerical methods to solve the dynamic functions with the hard constraint of collision. To be more specific, at every time step, assigned the

physics attributes (e.g. density, stiffness, and coefficient of friction) and given the current state of the subject (e.g. velocity, torques, and accelerations), the simulator updates the next subject state by solving the dynamic. Among all the physics simulators publicly available, Isaac Gym is currently the most efficient environment for online learning since it operates on GPU and supports parallel accelerations. This leads to blazing fast training times for complex simulated character control tasks on a single GPU with 2-3 orders of magnitude improvements compared to conventional online training that uses a CPU-based simulator and GPUs for neural networks [66].

Issac Gym uses the Temporal Gauss Seidel (TGS) [64] solver to compute the future states of objects in our physics simulation. The TGS solver uses the observation that sub-stepping a simulation with a single Gauss-Seidel solver iteration yields significantly faster convergence than running larger steps with more solver iterations. It folds this process efficiently into the iteration process, calculating the velocity at the end of each iteration and accumulating these velocities (scaled by $\delta t/N$, where $N$ is the number of iterations) into a per-body accumulated delta buffer. This delta buffer is projected onto the constraint Jacobians and added to the bias terms in the constraints. This approach adds only a few additional operations to a more traditional Gauss-Seidel solver, producing almost identical performance costs per-iteration. However, it achieves the same effect on convergence as having sub-stepped the simulation without the computational expense. With positional joint constraints, an additional rotational term is calculated for joint anchors to improve the handling of non-linear motion and avoid linearization artifacts [66].

# Chapter 3

# Related Work

Generating 3D motion akin human has been a long lasting problem in the cross-discipline of computer vision, animations and robotics. On one hand, recently, the emerging generative artificial intelligence techniques are revolutionizing this field. By modeling the distribution of large motion caption dataset, we are going to be able to generate motion more natural than ever before. On the other hand, involving physics constrains by controlling character motion in the physics simulator is intuitively one of the best solutions to guarantee physically plausible results. In this chapter, we first provide a brief overview of related techniques on latent representation, generative mask modeling, and retrieval augmented generation. Then we review the most closely related work in motion matching, generative human motion modeling, and simulated character control.

## 3.1 Latent Representation and Deep Quantization

Motion contains plenty of redundant features, particularly along the temporal dimension. This redundancy necessitates more efficient latent representations for network learning and drives the use of deep compression and quantization techniques. Deep Motion Signatures [2] learns semantically meaningful discrete motif words leveraging triplet contrastive learning. TM2T [31] starts applying vector quantized-VAE [100] to learn the mutual mapping between human motions and discrete tokens, where the autoencoding latent codes are

replaced with the selected entries from a codebook. T2M-GPT [111] further enhances the performance using EMA and code reset techniques. This is adopted in several other works such as PoseGPT [61] and MotionGPT [41], [114]. Nevertheless, the quantization process inevitably introduces errors, leading to suboptimal motion reconstruction. In the work of this thesis, MoMask, we adapt residual quantization [5], [68], [110], a technique used in neural network compression [22], [51], [52] and audio quantization [5], [102] which iteratively quantizes a vector and its residuals. This approach represents the vector as a stack of codes, enabling high-precision motion quantization.

## 3.2 Generative Masked Modeling

BERT [18] introduces masked modeling for language tasks that word tokens are randomly masked out with a fixed ratio, and then the bi-directional transformer learns to predict the masked tokens. Despite being a decent pre-trained text encoder, BERT cannot synthesize novel samples. In this regard, [11] proposes to mask the tokens with a variable and traceable rate that is controlled by a scheduling function. Therefore, new samples can be synthesized iteratively following the scheduled masking. MAGE [50] unifies representation learning and image synthesis using the masked generative encoder. Muse [10] extends this paradigm for text-to-image generation and editing. Magvit [109] suggests a versatile masking strategy for multi-task video generation. Inspired by these successes, we first introduce generative masked modeling for human motion synthesis in our work, MoMask.

## 3.3 Retrieval Augmented Generation

Retrieval Augmented Generation (RAG) has increasingly captured the attention of the Natural Language Processing (NLP) community, demonstrating exceptional performance in a variety of NLP tasks, such as dialogue systems[48], machine translation[27], text style transfer[106], etc. Compared to traditional pre-trained generation models, RAGs offer compelling advantages, such as straightforward knowledge acquisition, robust scalability, and reduced train-

ing costs. A critical component of RAMs is the retrieval metrics, often associated with similarity measurements. Dense-vector retrieval exploits pre-trained encoders like BERT to assess the similarity of latent space vectors [46], [49]. However, employing similarity-based metrics as a simple heuristic may not always yield optimal results, as the closest match in a hand-crafted feature space may not necessarily fit best within a specific content space scenario. Consequently, the ideal solution appears to be the integration of the retriever and its augmented model into a task-oriented, learnable entity. To facilitate the gradient flow from the retrieved value to the generated query, re-parameterization techniques are applied, such as weighted summing of retrieval candidate values, biased with relevance scores [6] linked to query-key similarity. In motion generation task, it is impractical to combine various motion clips using any additive approach. As such, we employ policy-gradient to train a query generator with specific task objectives. This approach enables our retrieval component to autonomously handle extensive and disordered databases and boosts control task responsiveness.

## 3.4 Motion Tracking

Numerous studies have explored the use of simple objectives or meticulously constructed heuristics in motion-tracking to replicate locomotion behaviors [19], [35]. To enhance animation realism, the recent focus has shifted to the incorporation of motion capture data [39], [57], where reference motions serve as constraints to promote character imitation. These constraints, typically defined as a tracking objective, aim to minimize the pose discrepancy between the simulated and reference motion [75], [101]. Reference motions require careful selection and synchronization with the simulated character. A phase variable, utilized in [47], [75], serves as an additional signal to generate a condensed set of reference poses, with each pose assigned a phase. Despite the success of phase-variable-based methods in replicating simple motions, they are still inapplicable for a wider range of motions that are difficult to synchronize. An alternative solution involves the ensemble of multiple independent

models [57], [75]. Contemporary methods handling larger motion datasets assign exact reference poses for the target frame as the objective rather than a motion phase[4], [14], [101], [104]. Consequently, the motion tracking problem simplifies to precise motion planning and straightforward mimicking. Nevertheless, identifying the most suitable action for a character to mimic in a given scenario remains challenging and necessitates extensive engineering efforts towards a sophisticated motion planner [4], [74]. Thus, these motion tracking-based methods may have their performance limited by the pre-designed motion planner and the quality of the MoCap database. In contrast, our work trains a motion expert retriever in tandem with the controller. The retriever, unlike previous methods, is automatically optimized with its own objective and the overarching system objective, making it task-oriented. We aim to develop a more integrated and efficient approach for controlling physics-based characters through the joint training of these components, even with extensive and chaotic MoCap databases.

## 3.5 Generative Human Motion Modeling

Recently, there has been a surge in motion generation research, exploring various domains such as motion prefix [59], [67], action class [9], [32], [61], [78], audio [25], [89], [99], [116], and texts [13], [30], [31], [79], [95]. Early works [1], [24], [40], [53], [81] commonly modeled motion generation deterministically, resulting in averaged and blurry motion outcomes. This issue is effectively addressed by stochastic models. GAN modeling is employed in [7], [103] for action-conditioned motion generation. Simultaneously, the temporal VAE framework and transformer architecture are utilized in works such as [33], [78].

For text-to-motion generation, T2M [30] extends the temporal VAE to learn the probabilistic mapping between texts and motions. Similarly, TEMOS [79] leverages Transformer VAE to optimize a joint variational space between natural language and motions, further extended by TEACH [3] for long motion compositions. MotionCLIP [94] and ohMG [55] model text-to-motion in an

unsupervised manner using the large pretrained CLIP [83] model.

The advent of diffusion models and autoregressive models has significantly transformed the field of motion generation. Diffusion methods train a network to gradually denoise the motion sequence, guided by a scheduled diffusion process [13], [42], [44], [60], [95], [99], [112]. While diffusion models show promising capabilities for generating high-quality motions, their inference efficiency remains a bottleneck, typically requiring hundreds of sampling steps. Various solutions, such as latent diffusion [13], [86] and the DDIM sampler [90], have been proposed to address this issue, but it remains an ongoing challenge. In autoregressive models [25], [31], [41], [111], [114], motions are first discretized into tokens via vector quantization [100], which are then modeled by expressive transformers similar to language models. This approach has become popular for topics such as action-to-motion [61], text-to-motion [31], [41], [111], [114], and dance generation [25], [89].

## 3.6 Simulated Character Motion Control

A growing number of studies are now focusing on integrating stochastic generative models with explicit physics awareness in motion control, predominantly through non-differentiable physics simulators. Harmonizing the statistical assumptions of novel generative models with sampling-based methods or reinforcement learning, however, remains challenging. Drawing from Generative Adversarial Imitation Learning (GAIL) principles [36], one strategy employs a discriminator to guide the system, aiming to align the character's motion distribution with the dataset [21], [77]. Despite this, such methods often fall prey to mode collapse, a common issue in GAN family models. While using conditional variables in the latent space can enhance latent capacity, training control policies or tuning the latent space remains challenging [20], [76], [93].

Following a similar latent representation approach, another set of methods employs Variational Autoencoders (VAEs) [43] to map the conditional distribution of the labeled dataset into a unified latent space [56], [105], [107], [108]. Here, a trained world model replaces the corresponding non-differentiable sim-

ulator, transforming the reinforcement learning problem into a fully differentiable supervised learning problem. This approach simplifies the task but could deviate significantly from the original problem depending on the precision of the world model.

Recently, diffusion models have also emerged in the motion control domain, leveraging their strong generative capabilities to model the distribution of state-action trajectories [98]. Some approaches use diffusion models as trajectory planners, cascading with an action controller to solve the inverse dynamics $p(a_1, ..., a_k|s_1, ..., s_k)$ [85]. Others use diffusion models to generate the current action through a few denoising steps, replacing common single forward policies [12], [15], [88].

# Chapter 4

# MoMask: Generative Masked Modeling for Text-to-Motion Generation

In this chapter, we introduce MoMask, a novel masked modeling framework for text-driven 3D human motion generation. In MoMask, a hierarchical quantization scheme is employed to represent human motion as multi-layer discrete motion tokens with high-fidelity details. Starting at the base layer, with a sequence of motion tokens obtained by vector quantization, the residual tokens of increasing orders are derived and stored at the subsequent layers of the hierarchy. This is consequently followed by two distinct bidirectional transformers. For the base-layer motion tokens, a Masked Transformer is designated to predict randomly masked motion tokens conditioned on text input at training stage. During generation (i.e. inference) stage, starting from an empty sequence, our Masked Transformer iteratively fills up the missing tokens; Subsequently, a Residual Transformer learns to progressively predict the next-layer tokens based on the results from current layer. Extensive experiments demonstrate that MoMask outperforms the state-of-the-art methods on the text-to-motion generation task, with an FID of 0.045 (vs e.g. 0.141 of T2M-GPT [111] in CVPR'23) on the HumanML3D dataset, and 0.228 (vs 0.514) on KIT-ML, respectively. MoMask can also be seamlessly applied in related tasks without further model fine-tuning, such as text-guided temporal inpainting.

## 4.1 Introduction

Generating 3D human motions from textual descriptions, aka text-to-motion generation, is a relatively new task that may play an important role in a broad range of applications such as video games, metaverse, and virtual reality & augmented reality. In the past few years, it has generated intensive research interests [13], [30], [31], [41], [44], [79], [95], [111], [112]. Among them, it has become popular to engage generative transformers in modeling human motions [25], [31], [41], [111]. In this pipeline, motions are transformed into discrete tokens through vector quantization (VQ), then fed into e.g. an autoregressive model to generate the sequence of motion tokens in an unidirectional order. Though achieving impressive results, these methods shares two innate drawbacks. To begin with, the VQ process inevitably introduces approximation errors, which imposes undesired limit to the motion generation quality. Moreover, the unidirectional decoding may unnecessarily hinder the expressiveness of the generative models. For instance, consider the following scenario: at each time step, the motion content is generated by only considering the preceding (rather than global) context; furthermore, errors will often accumulate over the generation process. Though several recent efforts using discrete diffusion models [44], [60] have considered to decode the motion tokens bidirectionally, by relying on a cumbersome discrete diffusion process, they typically require hundreds of iterations to produce a motion sequence.

Motivated by these observations, we propose a novel framework, MoMask, for high-quality and efficient text-to-motion generation by leveraging the residual vector quantization (RVQ) techniques [5], [68], [110] and the recent generative masked transformers [10], [11], [50], [109]. Our approach builds on the following three components. First, an RVQ-VAE is learned to establish precise mappings between 3D motions and the corresponding sequences of discrete motion tokens. Unlike previous motion VQ tokenizers [25], [31], [111] that typically quantize latent codes in a single pass, our hierarchical RVQ employs iterative rounds of residual quantization to progressively reduce quantization errors. This results in multi-layer motion tokens, with the base layer

serving to perform standard motion quantization, and the rest layers in the hierarchy capturing the residual coding errors of their respective orders, layer by layer. Our quantization-based hierarchical design is further facilitated by two distinct transformers, the Masked Transformer (i.e. M-Transformer) and Residual Transformer (R-Transformer), that are dedicated to generating motion tokens for the base VQ layer and the rest residual layers, respectively.

The M-Transformer, based on BERT [18], is trained to predict the randomly masked tokens at the base layer, conditioned on textual input. The ratio of masking, instead of being fixed [18], [34], is a scheduled variable that ranges from 0 to 1. During generation, starting from all tokens being masked out, M-Transformer produces a complete sequence of motion tokens within a small number of iterations. At each iteration, all masked tokens are predicted *simultaneously*. Predicted tokens with the highest confidence will remain unchanged, while the others are masked again and re-predicted in the next iteration. Once the base-layer tokens are generated, the R-Transformer ensues to progressively predict the residual tokens of the subsequent layer given the token sequence at current layer. Overall, the entire set of layered motion tokens can be efficiently generated within merely 15 iterations, regardless of the motion's length.

Our main contributions can be summarized as follows: First, our MoMask is the first generative masked modeling framework for the problem of text-to-motion generation. It comprises of a hierarchical quantization generative model and the dedicated mechanism for precise residual quantization, base token generation and residual token prediction. Second, our MoMask pipeline produces precise and efficient text-to-motion generation. Empirically, it achieves new state-of-the-art performance on text-to-motion generation task with an FID of 0.045 (*vs.* 0.141 in [111]) on HumanML3D and 0.204 (*vs.* 0.514 in [111]) on KIT-ML. Third, our MoMask also works well for related tasks, such as text-guided motion inpainting.

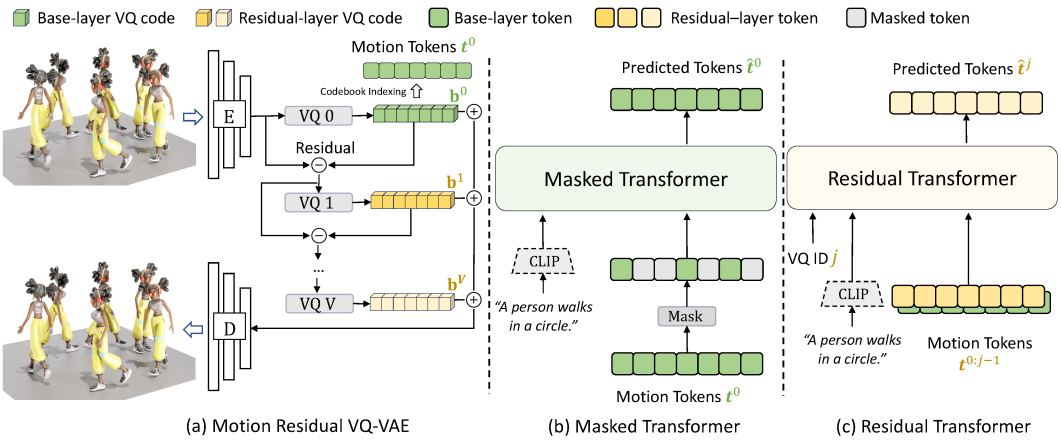



Figure 4.1: Approach overview of MoMask.

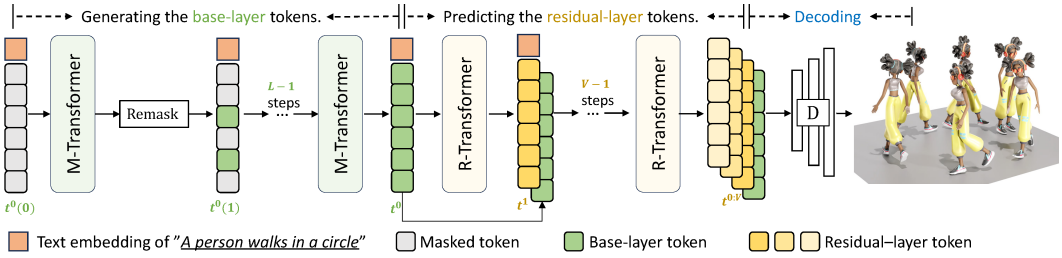



Figure 4.2: Inference process of MoMask.

## 4.2 Approach

Our goal is to generate a 3D human pose sequence $\mathbf{m}_{1:N}$ of length $N$ guided by a textual description $c$, where $\mathbf{m}_i \in \mathbb{R}^D$ with $D$ denoting the dimension of pose features. As illustrated in Fig. 4.1, our approach consists of three principle components: a residual-based quantizer that tokenizes motion sequence into multi-layer discrete tokens (Sec. 4.2.1), a masked transformer that generates motion tokens in the base layer (Sec. 4.2.2), and a residual transformer (Sec. 4.2.3) that predicts the tokens in the subsequent residual layers. (a) Motion sequence is tokenized through vector quantization (VQ), also referred to as the base quantization layer, as well as a hierarchy of multiple layers for residual quantization. (b) *Parallel* prediction by the Masked Transformer: the tokens in the base layer $t^0$ are randomly masked out with a variable rate, and then a text-conditioned masked transformer is trained to predict the masked

tokens in the sequence simultaneously. (c) Layer-by-layer *progressive* prediction by the Residual Transformer. A text-conditioned residual transformer learns to progressively predict the residual tokens $t^{j>0}$ from the tokens in previous layers, $t^{0:j-1}$. The inference process of generation is detailed in Sec. 4.2.4. Starting from an empty sequence $t^0(0)$, the M-Transformer generates the base-layer token sequence $t^0$ in $L$ iterations. Following this, the R-Transformer progressively predicts the rest-layer token sequences $t^{2:V}$ within $V-1$ steps.

### 4.2.1 Training: Motion Residual VQ-VAE

Conventional motion VQ-VAEs [31], [41], [111], [114] transform a motion sequence into one tuple of discrete motion tokens. Specifically, the motion sequence $\mathbf{m}_{1:N} \in \mathbb{R}^{N\times D}$ is firstly encoded into a latent vector sequence $\tilde{\mathbf{b}}_{1:n} \in \mathbb{R}^{n\times d}$ with downsampling ratio of $n/N$ and latent dimension $d$, using 1D convolutional encoder E; each vector is subsequently replaced with its nearest code entry in a preset codebook $\mathcal{C} = \{\mathbf{c}_k\}_{k=1}^{K} \subset \mathbb{R}^d$, known as quantization $\mathrm{Q}(\cdot)$. Then the quantized code sequence $\mathbf{b}_{1:n} = \mathrm{Q}(\tilde{\mathbf{b}}_{1:n}) \in \mathbb{R}^{n\times d}$ is projected back to motion space for reconstructing the motion $\hat{\mathbf{m}} = \mathrm{D}(\mathbf{b})$. After all, the indices of the selected codebook entries (namely *motion tokens*) are used as the alternative discrete representation of input motion. Though effective, the quantization operation $\mathrm{Q}(\cdot)$ inevitably leads to information loss, which further limits the quality of reconstruction.

To address this issue, we introduce residual quantization (RQ) as described in Fig. 4.1(a). In particular, RQ represents a motion latent sequence $\tilde{\mathbf{b}}$ as $V+1$ *ordered* code sequences, using $V+1$ quantization layers. Formally, this is defined as $\mathrm{RQ}(\tilde{\mathbf{b}}_{1:n}) = [\mathbf{b}^v_{1:n}]_{v=0}^{V}$, with $\mathbf{b}^v_{1:n} \in \mathbb{R}^{n\times d}$ denoting the code sequence at the $v$-th quantization layer. Concretely, starting from 0-th residual $\mathbf{r}^0 = \tilde{\mathbf{b}}$, RQ recursively calculates $\mathbf{b}^v$ as the approximation of residual $\mathbf{r}^v$, and then the next residual $\mathbf{r}^{v+1}$ as

$$\mathbf{b}^v = \mathrm{Q}(\mathbf{r}^v), \quad \mathbf{r}^{v+1} = \mathbf{r}^v - \mathbf{b}^v, \tag{4.1}$$

for $v = 0, ..., V$. After RQ, the final approximation of latent sequence $\tilde{\mathbf{b}}$ is the sum of all quantized sequences $\sum_{v=0}^{V} \mathbf{b}^v$, which is then fed into decoder D for

motion reconstruction.

Overall, the residual VQ-VAE is trained via a motion reconstruction loss combined with a latent embedding loss at each quantization layer:

$$\mathcal{L}_{rvq} = \|\mathbf{m} - \hat{\mathbf{m}}\|_1 + \beta \sum_{v=1}^{V} \|\mathbf{r}^v - \text{sg}[\mathbf{b}^v]\|_2^2, \tag{4.2}$$

where sg[·] denotes the stop-gradient operation, and $\beta$ a weighting factor for embedding constraint. This framework is optimized with straight-though gradient estimator [100], and our codebooks are updated via exponential moving average and codebook reset following T2M-GPT [111].

**Quantization Dropout.** Ideally, the early quantization layers are expected to restore the input motion as much as possible; then the later layers add up the missing finer details. To exploit the capacity of each quantizer, we adopt a *quantization dropout* strategy, which randomly disables the last 0 to $V$ layers with probability $q \in [0, 1]$ during training.

After training, each motion sequence $\mathbf{m}$ can be represented as $V + 1$ discrete motion token sequences $T = [t^v_{1:n}]_{v=0}^{V}$ where each token sequence $t^v_{1:n} \in \{1, ..., |\mathcal{C}^v|\}^n$ is the ordered codebook-indices of quantized embedding vectors $\mathbf{b}^v_{1:n}$, such that $\mathbf{b}^v_i = \mathcal{C}^v_{t^v_i}$ for $i \in [1, n]$. Among these $V + 1$ sequences, the first (i.e. base) sequence possesses the most prominent information, while the impact of subsequent layers gradually diminishes.

### 4.2.2 Training: Masked Transformer

Our bidirectional masked transformer is designed to model the base-layer motion tokens $t^0_{1:n} \in \mathbb{R}^n$, as illustrated in Figure 4.1(b). We first randomly masked out a varying fraction of sequence elements, by replacing the tokens with a special [MASK] token. With $\tilde{t}^0$ denoting the sequence after masking, the goal is to predict the masked tokens given text $c$ and $\tilde{t}^0$. We use CLIP [83] for extracting text features. Mathematically, our masked transformer $p_\theta$ is optimized to minimize the negative log-likelihood of target predictions:

$$\mathcal{L}_{mask} = \sum_{\tilde{t}_k^0=[\text{MASK}]} -\log p_\theta(t_k^0|\tilde{t}^0, c). \tag{4.3}$$

**Mask Ratio Schedule.** We adopt a cosine function $\gamma(\cdot)$ for scheduling the masking ratio following [10], [11]. Practically, the mask ratio is obtained by $\gamma(\tau) = \cos(\frac{\pi\tau}{2}) \in [0, 1]$, where $\tau \in [0, 1]$ that $\tau = 0$ means the sequence is completely corrupted. During training, the $\tau \sim \mathcal{U}(0, 1)$ is randomly sampled, and then $m = \lceil \gamma(\tau) \cdot n \rceil$ sequence entries are uniformly selected to be masked with $n$ denoting the length of sequence.

**Replacing and Remasking.** To enhance the contextual reasoning of the masked transformer, we adopt the remasking strategy used in BERT pretraining [18]. If a token is selected for masking, we replace this token with (1) [MASK] token 80% of the time; (2) a random token 10% of the time; and (3) an unchanged token 10% of the time.

### 4.2.3 Training: Residual Transformer

We learn a single residual transformer to model the tokens from the other $V$ residual quantization layers. The residual transformer has a similar architecture to the masked transformer (Sec. 4.2.2), except that it contains $V$ separate embedding layers. During training, we randomly select a quantizer layer $j \in [1, V]$ to learn. All the tokens in the preceding layers $t^{0:j-1}$ are embedded and summed up as the token embedding input. Taking the token embedding, text embedding, and RQ layer indicator $j$ as input, the residual transformer $p_\phi$ is trained to predict the $j$-th layer tokens in parallel. Overall, the training objective is:

$$\mathcal{L}_{res} = \sum_{j=1}^{V} \sum_{i=1}^{n} -\log p_\phi(t_i^j|t_i^{0:j-1}, c, j). \tag{4.4}$$

We also share the parameters of the $j$-th prediction layer and the $(j+1)$-th motion token embedding layer for more efficient learning.

### 4.2.4 Inference

As presented in Figure 4.2, there are three stages in inference. Firstly, starting from an empty sequence $t^0(0)$ that all tokens are masked out, we expect to generate the base-layer token sequence $t^0$ of length $n$ in $L$ iterations. Given the masked token sequence at $l$-th iteration $t^0(l)$, M-Transformer first predicts the probability distribution of tokens at the masked locations, and samples motion tokens with the probability. Then the sampled tokens with the lowest $\lceil \gamma(\frac{l}{L}) \cdot n \rceil$ confidences are masked again, and the other tokens will remain unchanged for the rest iterations. This new token sequence $t^0(l+1)$ is used to predict the token sequence at the next iteration until $l$ reaches $L$. Once the base-layer tokens are completely generated, the R-Transformer progressively predicts the token sequence in the rest quantization layers. Finally, all tokens are decoded and projected back to motion sequences through the RVQ-VAE decoder.

**Classifier Free Guidance.** We adopt classifier-free guidance (CFG) [10], [38] for the prediction of both M-Transformer and R-Transformer. During training, we train the transformers unconditionally $c = \emptyset$ with probability of 10%. During inference, CFG takes place at the final linear projection layer before softmax, where the final logits $\omega_g$ are computed by moving the conditional logits $\omega_c$ away from the unconditional logits $\omega_u$ with guidance scale $s$:

$$\omega_g = (1 + s) \cdot \omega_c - s \cdot \omega_u. \tag{4.5}$$

## 4.3 Experiments

Empirical evaluations are conducted on two widely used motion-language benchmarks, HumanML3D [30] and KIT-ML [80]. **HumanML3D** dataset collects 14,616 motions from AMASS [65] and HumanAct12 [32] datasets, with each motion described by 3 textual scripts, totaling 44,970 descriptions. This diverse motion-language dataset contains a variety of actions, including exercising, dancing, and acrobatics. **KIT-ML** dataset consists of 3,911 motions and 6,278 text descriptions, offering an small-scale evaluation benchmark. For both

| Datasets | Methods | R Precision↑ | | | FID↓ | MultiModal Dist↓ | MultiModality↑ |
|---|---|---|---|---|---|---|---|
| | | Top 1 | Top 2 | Top 3 | | | |
| Human ML3D | TM2T [31] | $0.424^{\pm.003}$ | $0.618^{\pm.003}$ | $0.729^{\pm.002}$ | $1.501^{\pm.017}$ | $3.467^{\pm.011}$ | $2.424^{\pm.093}$ |
| | T2M [30] | $0.455^{\pm.003}$ | $0.636^{\pm.003}$ | $0.736^{\pm.002}$ | $1.087^{\pm.021}$ | $3.347^{\pm.008}$ | $2.219^{\pm.074}$ |
| | MDM [95] | - | - | $0.611^{\pm.007}$ | $0.544^{\pm.044}$ | $5.566^{\pm.027}$ | $\mathbf{2.799}^{\pm.072}$ |
| | MLD [13] | $0.481^{\pm.003}$ | $0.673^{\pm.003}$ | $0.772^{\pm.002}$ | $0.473^{\pm.013}$ | $3.196^{\pm.010}$ | $2.413^{\pm.079}$ |
| | MotionDiffuse [112] | $0.491^{\pm.001}$ | $0.681^{\pm.001}$ | $0.782^{\pm.001}$ | $0.630^{\pm.001}$ | $3.113^{\pm.001}$ | $1.553^{\pm.042}$ |
| | T2M-GPT [111] | $0.492^{\pm.003}$ | $0.679^{\pm.002}$ | $0.775^{\pm.002}$ | $0.141^{\pm.005}$ | $3.121^{\pm.009}$ | $1.831^{\pm.048}$ |
| | ReMoDiffuse [113] | $0.510^{\pm.005}$ | $0.698^{\pm.006}$ | $0.795^{\pm.004}$ | $0.103^{\pm.004}$ | $2.974^{\pm.016}$ | $1.795^{\pm.043}$ |
| | MotionGPT [41] | $0.492^{\pm.003}$ | $0.681^{\pm.003}$ | $0.778^{\pm.002}$ | $0.232^{\pm.008}$ | $3.096^{\pm.008}$ | $2.008^{\pm.084}$ |
| | MoMask (base) | $0.504^{\pm.004}$ | $0.699^{\pm.006}$ | $0.797^{\pm.004}$ | $0.082^{\pm.008}$ | $3.050^{\pm.013}$ | $1.050^{\pm.061}$ |
| | **MoMask** | $\mathbf{0.521}^{\pm.002}$ | $\mathbf{0.713}^{\pm.002}$ | $\mathbf{0.807}^{\pm.002}$ | $\mathbf{0.045}^{\pm.002}$ | $\mathbf{2.958}^{\pm.008}$ | $1.241^{\pm.040}$ |
| KIT-ML | TM2T [31] | $0.280^{\pm.005}$ | $0.463^{\pm.006}$ | $0.587^{\pm.005}$ | $3.599^{\pm.153}$ | $4.591^{\pm.026}$ | $\mathbf{3.292}^{\pm.081}$ |
| | T2M [30] | $0.361^{\pm.005}$ | $0.559^{\pm.007}$ | $0.681^{\pm.007}$ | $3.022^{\pm.107}$ | $3.488^{\pm028}$ | $2.052^{\pm.107}$ |
| | MDM [95] | - | - | $0.396^{\pm.004}$ | $0.497^{\pm.021}$ | $9.191^{\pm.022}$ | $1.907^{\pm.214}$ |
| | MLD [13] | $0.390^{\pm.008}$ | $0.609^{\pm.008}$ | $0.734^{\pm.007}$ | $0.404^{\pm.027}$ | $3.204^{\pm.027}$ | $2.192^{\pm.071}$ |
| | MotionDiffuse [112] | $0.417^{\pm.004}$ | $0.621^{\pm.004}$ | $0.739^{\pm.004}$ | $1.954^{\pm.062}$ | $2.958^{\pm.005}$ | $0.730^{\pm.013}$ |
| | T2M-GPT [111] | $0.416^{\pm.006}$ | $0.627^{\pm.006}$ | $0.745^{\pm.006}$ | $0.514^{\pm.029}$ | $3.007^{\pm.023}$ | $1.570^{\pm.039}$ |
| | ReMoDiffuse [113] | $0.427^{\pm.014}$ | $0.641^{\pm.004}$ | $0.765^{\pm.055}$ | $\mathbf{0.155}^{\pm.006}$ | $2.814^{\pm.012}$ | $1.239^{\pm.028}$ |
| | MoMask (base) | $0.415^{\pm.010}$ | $0.634^{\pm.011}$ | $0.760^{\pm.005}$ | $0.372^{\pm.020}$ | $2.931^{\pm.041}$ | $1.097^{\pm.054}$ |
| | **MoMask** | $\mathbf{0.433}^{\pm.007}$ | $\mathbf{0.656}^{\pm.005}$ | $\mathbf{0.781}^{\pm.005}$ | $0.204^{\pm.011}$ | $\mathbf{2.779}^{\pm.022}$ | $1.131^{\pm.043}$ |

Table 4.1: Quantitative evaluation on the HumanML3D and KIT-ML test set of MoMask.

motion datasets, we adopt the pose representation from the work of T2M [30]. The datasets are augmented by mirroring, and divided into training, testing, and validation sets with the ratio of **0.8:0.15:0.05**.

**Evaluation metrics** from T2M [30] are also adopted throughout our experiments including: (1) *Frechet Inception Distance* (FID), which evaluates the overall motion quality by measuring the distributional difference between the high-level features of the generated motions and those of real motions; (2) *R-Precision* and *multimodal distance*, which gauge the semantic alignment between input text and generated motions; and (3) *Multimodality* for assessing the diversity of motions generated from the same text.

Though *multimodality* is indeed important, we stress its role as a secondary metric that should be assessed in the conjunction with primary performance metrics such as FID and RPrecision. Emphasizing multimodality without considering the overall quality of generated results could lead to optimization of models that produce random outputs for any given input.

**Implementation Details.** Our models are implemented using PyTorch. For the motion residual VQ-VAE, we employ resblocks for both the encoder and decoder, with a downscale factor of 4. The RVQ consists of 6 quantization layers, where each layer's codebook contains 512 512-dimensional codes. The quanti-

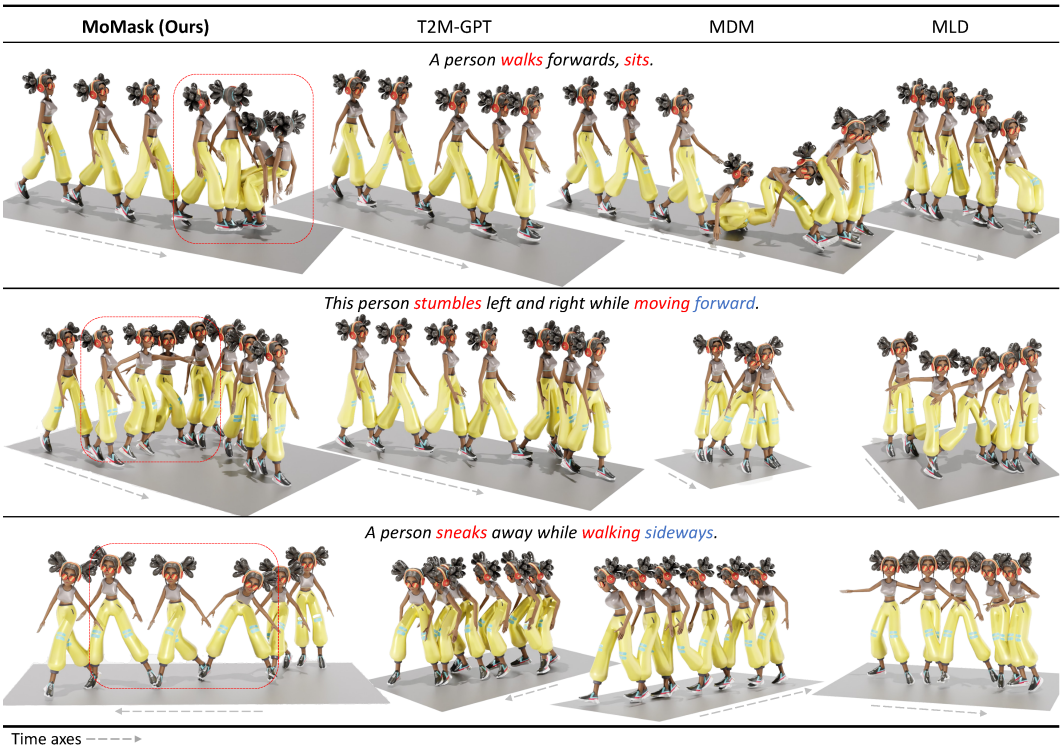
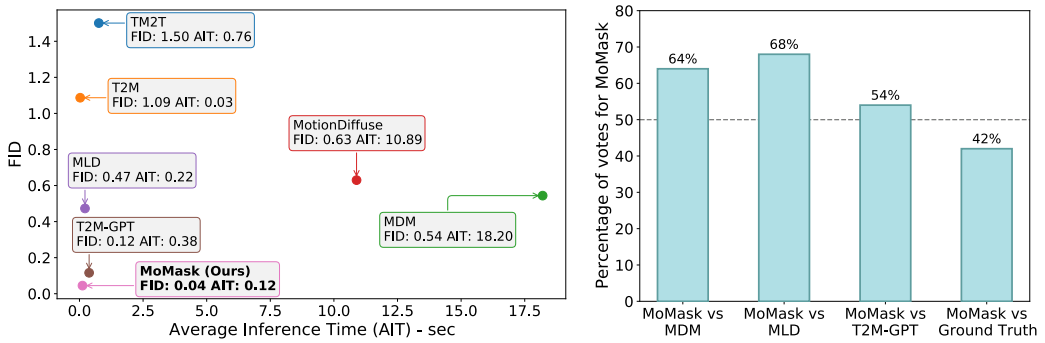



Figure 4.3: Visual comparisons between MoMask and previous SOTA methods.

zation dropout ratio $q$ is set to 0.2. Both the masked transformer and residual transformer are composed of 6 transformer layers, with 6 heads and a latent dimension of 384, applied to the HumanML3D and KIT-ML datasets. The learning rate reaches 2e-4 after 2000 iterations with a linear warm-up schedule for the training of all models. The mini-batch size is uniformly set to 512 for training RVQ-VAE and 64, 32 for training transformers on HumanML3D and KIT-ML, respectively. During inference, we use the CFG scale of 4 and 5 for M-Transformer and R-Transformer on HumanML3D, and (2, 5) on KIT-ML. Meanwhile, $L$ is set to 10 on both datasets.

### 4.3.1 Comparison to state-of-the-art approaches

We compare our approach to a set of existing state-of-the-art works ranging from VAE [30], diffusion-based models [13], [95], [113], to autoregressive models [31], [111].




(a) Comparisons on FID and Inference Cost

(b) HumanML3D User Study

Figure 4.4: (a) Comparison of inference time costs of MoMask. (b) User study results on the HumanML3D dataset of MoMask.

**Quantitative Comparisons.** Following previous practices [30], [95], each experiment is repeated 20 times, and the reported metric values represent the mean with a 95% statistical confidence interval. Additionally, we conduct experiments with MoMask exclusively generating the base-layer motion tokens, denoted as MoMask (base). Quantitative results for the HumanML3D and KIT-ML datasets are presented in Table 4.1. $\pm$ indicates a 95% confidence interval. MoMask (base) means that MoMask only uses base-layer tokens. **Bold** face indicates the best result, while underscore refers to the second best.

Overall, MoMask attains state-of-the-art performance on both datasets, demonstrating substantial improvements in metrics such as FID, R-Precision, and multimodal distance. For the suboptimal performance on KIT-ML dataset, we would like to point out that the leading model, ReMoDiffuse [113], involves more intricate data retrieval from a large database to achieve high-quality motion generation. Additionally, we observe that MoMask, even with the base-layer tokens alone, already achieves competitive performance compared to baselines, and the inclusion of residual tokens further elevates the results to a higher level.

In Figure 4.4(a), we evaluate the efficiency and quality of motion generation using various methods. All tests are conducted on the same Nvidia2080Ti. The closer the model is to the origin, the better. The inference cost is calculated as the average inference time over 100 samples on one Nvidia2080Ti

device. Comparing to baseline methods, MoMask positions itself more favorably between generation quality and efficiency.

**User Study.** We further conduct a user study on Amazon Mechanical Turk to validate our previous conclusions. This user study involves 42 AMT users with *master* recognition, with the side-by-side comparisons between MoMask and each of the state-of-the-art methods including MDM [95], MLD [13] and T2M-GPT [111]. We generate the 50 motions for each method using the same text pool from HumanML3D test set, and collect feedback from 3 distinct users for each comparison. The results are shown in Fig. 4.4(b), where each bar represents the preference rate of MoMask over the compared model. Overall, MoMask is preferred over the other models most of the time. The dashed line marks 50%. MoMask is preferred by users in most of the time, and even earns 42% of preference on par with ground truth motions.

**Qualitative Comparisons.** Figure 4.3 displays qualitative comparisons of our approach and MDM[95], MLD [13], and T2M-GPT [111]. The comparison is conducted by giving three distinct text descriptions from HumanML3D test-set. Only key frames are displayed. Compared to previous methods, MoMask generates motions with higher quality and better understanding of the subtle language concepts such as "*stumble*", "*sneak*", "*walk sideways*". MDM [95] usually generates overall semantically correct motions but fails to capture nuanced concepts such as "*sneak*" and "*sideways*". Though T2M-GPT [111] and MLD [13] have improved performance in this aspect, they still find it difficult to generate motions accurately aligned with the textual description. For example, in the bottom row, the motions from these two methods either forget to *walk sideways* (T2M-GPT [111]) or to *sneak away* (MLD [13]). Moreover, MLD [13] sometimes produces lifeless motions where the character slides around, as shown in the top row. In comparison, our method is able to generate high-quality motions faithful to the input texts.

### 4.3.2 Component Analysis

In Table 4.2, we comprehensively evaluate the impact of different design components in MoMask through various comparisons, showcasing the performance

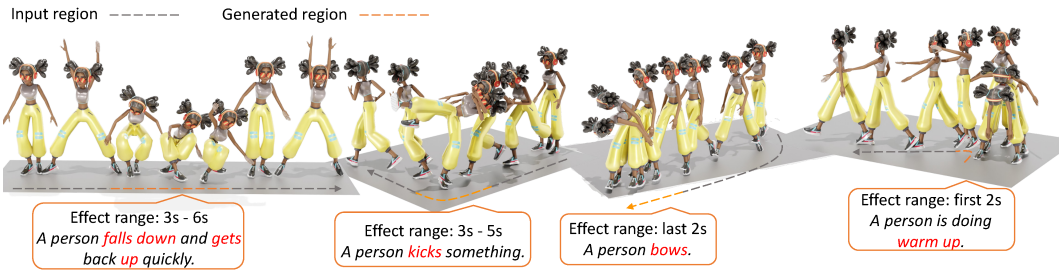



Figure 4.5: Examples of temporal inpainting for MoMask.

in both motion reconstruction and generation. We further analysis on residual quantization (RQ), quantization dropout (QDropout), and replacing & remasking (RRmask). $V$ and $q$ are the number of RQ and QDropout ratio, respectively. MPJPE is measured in millimeters. Generation results are based on 18 inference steps. Initially, we compare our approach with previous VQ-based motion generation methods [31], [44], [111] on the HumanML3D and KIT-ML datasets. Notably, M2DM [44] incorporates orthogonality constraints among all codebook entries to enhance VQ performance. Our residual design shows clearly superior performance when comparing with these single VQ-based approaches.

**Ablation.** In the ablation experiments, we observe that both residual quantization (RQ) and quantization dropout (QDropout) effectively contribute to the enhancement of motion quality in terms of both reconstruction and generation. Additionally, replacing-and-remasking strategy, as well as RQ, facilitates more faithful motion generation.

In Table 4.3, we conduct an ablation study by employing a *bidirectional* Transformer encoder (B) and a *unidirectional* Transformer decoder (U) separately as the backbone in our MoMask framework. We assess the performance of MoMask using a unidirectional Transformer Decoder as the backbone (Ours (U)). In this configuration, causal attention is employed, and the model can only attend to previous positions in the sequence. This stands in contrast to the design presented in the paper, which utilizes a bidirectional Transformer Encoder (Ours (B)). The experiment results highlight the significance of bidirectionality in the MoMask framework. Additionally, it's worth noting that

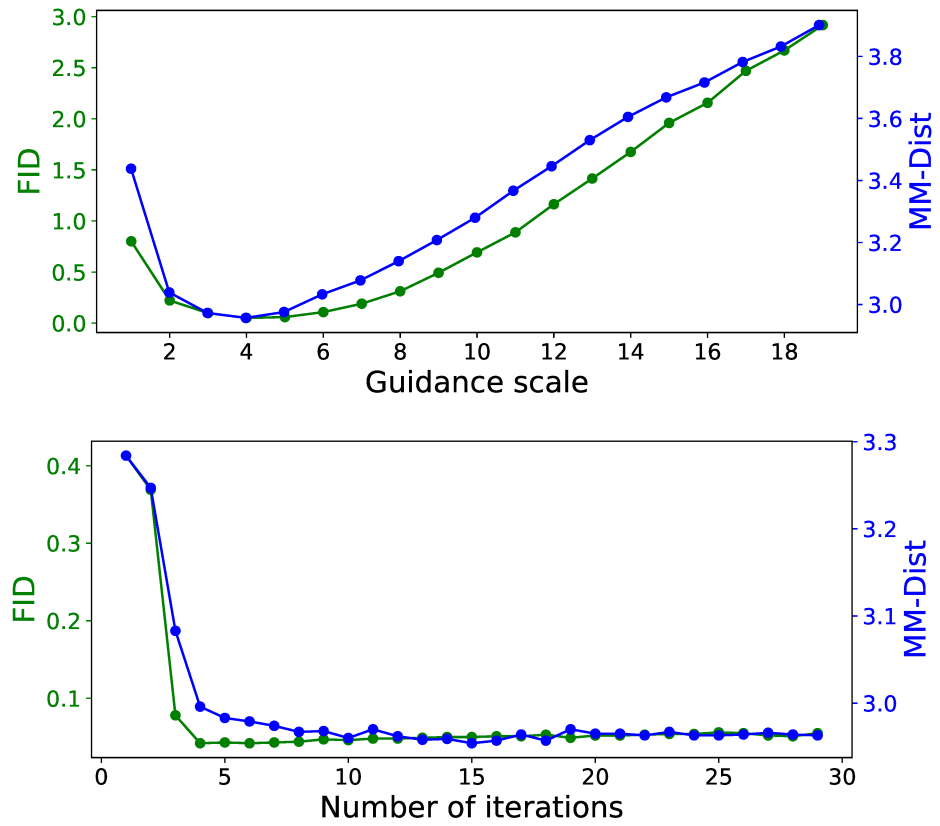



Figure 4.6: Evaluation sweep over guidance scale $s$ (top) and iteration numbers $L$ (bottom) in inference on MoMask.

the baseline of T2M-GPT [111] represents a state-of-the-art unidirectional approach.

**Number of Residual Layers ($V$).** In Tab. 4.2, we investigate RVQ with different numbers of quantization layers. Generally, more residual VQ layers result in more precise reconstruction, shown in Fig. 4.7, but they also increase the burden on the R-Transformer for residual token generation. We particularly observe that the generation performance starts to degrade with more than 5 residual layers. This finding emphasizes the importance of striking a balance in the number of residual layers for optimal performance.

**Quantization Dropout ($q$).** We also analyze the impact of quantization dropout ratio $q$ in Tab. 4.2. As we increase dropout probability from 0.2, the performance gains become marginal, or even converse. We speculate that frequent disabling quantization layers may disturb the learning of quantization

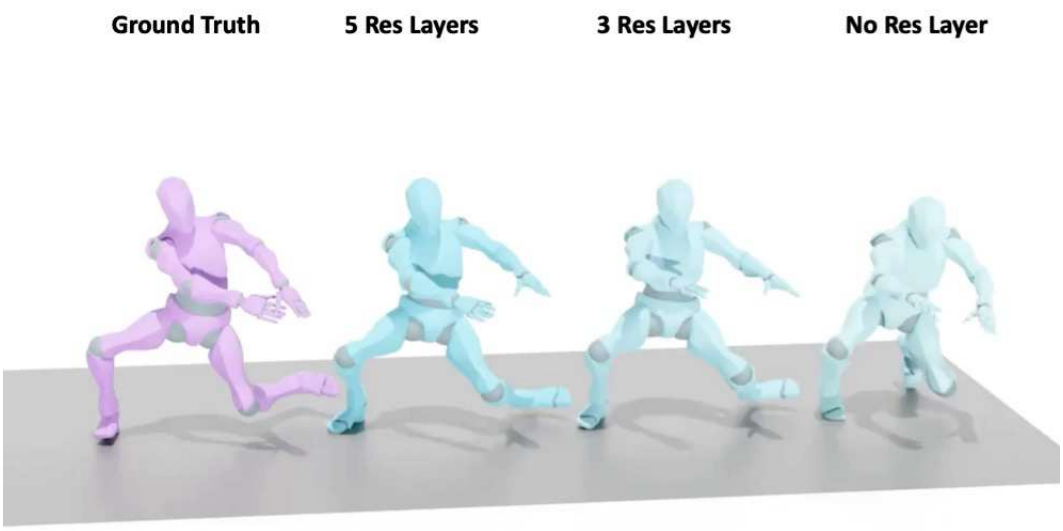



Figure 4.7: Reconstruction results using a different number of residual layers in RVQ.

models.

**Inference Hyper-parameters.** The CFG scale $s$ and the number of iterations $L$ are two crucial hyperparameters during the inference of masked modeling. In Fig. 4.6, we present the performance curves of FID and multimodality distance by sweeping over different values of $s$ and $L$. We initially find a accuracy-fidelity sweep spot around $s = 4$, meanwhile 10 iterations ($L = 10$) for masked decoding yield sufficiently good results. Moreover, several key observations emerge. Firstly, an optimal guidance scale $s$ for M-Transformer inference is identified around $s = 4$. Over-guided decoding may even inversely deteriorate the performance. Secondly, more iterations are not necessarily better. As $L$ increases, the FID and multimodality distance converge to the minima quickly, typically within around 10 iterations. Beyond 10 iterations, there are no further performance gains in both FID and multimodal distance. In this regard, our MoMask requires fewer inference steps compared to most autoregressive and diffusion models.

### 4.3.3 Application: Temporal Inpainting

In Fig. 4.5, we showcase the capability of MoMask in temporally inpainting a specific region in a motion sequence. Dark dash line indicates the range(s) where the motion content(s) is given by the reference sequence. Orange dash line indicates the range of motion content generated by MoMask, conditioned on the text prompt below. The region can be freely located in the middle, suffix, or prefix. Specifically, we mask out all the tokens in the region of interest and then follow the same inference procedure described in Sec. 4.2.4. For both tasks, our approach generates smooth motions in coherence with the given text descriptions. Additionally, we conduct a user study to quantitatively compare our inpainting results with those of MDM [95]. In this study, 40 samples are generated from both methods using the same motion and text input, and presented to users side-by-side. With 6 users involved, 68% of the results from MoMask are preferred over MDM.

## 4.4 Limitations

We acknowledge certain limitations of MoMask. Firstly, while MoMask excels in fidelity and faithfulness for text-to-motion synthesis, its diversity is relatively limited. We plan to delve into the underlying causes of this limitation in future work. Secondly, MoMask requires the target length as input for motion generation. This could be properly addressed by applying the text2length sampling [30] beforehand. Thirdly, akin to most VQ-based methods, MoMask may face challenges when generating motions with fast-changing root motions, such as *spinning*.

| Methods | Reconstruction | | Generation | |
|---|---|---|---|---|
| | FID↓ | MPJPE↓ | FID↓ | MM-Dist↓ |
| ***Evaluation on KIT-ML dataset*** | | | | |
| M2DM [44] | $0.413^{\pm.009}$ | - | $0.515^{\pm.029}$ | $3.015^{\pm.017}$ |
| T2M-GPT [111] | $0.472^{\pm.011}$ | - | $0.514^{\pm.029}$ | $3.007^{\pm.023}$ |
| **MoMask** | $\mathbf{0.112}^{\pm.002}$ | **37.2** | $\mathbf{0.228}^{\pm.011}$ | $\mathbf{2.774}^{\pm.022}$ |
| ***Evaluation on HumanML3D dataset*** | | | | |
| TM2T [31] | $0.307^{\pm.002}$ | 230.1 | $1.501^{\pm.017}$ | $3.467^{\pm.011}$ |
| M2DM [44] | $0.063^{\pm.001}$ | - | $0.352^{\pm.005}$ | $3.116^{\pm.008}$ |
| T2M-GPT [111] | $0.070^{\pm.001}$ | 58.0 | $0.141^{\pm.005}$ | $3.121^{\pm.009}$ |
| **MoMask** | $\mathbf{0.019}^{\pm.001}$ | **29.5** | $\mathbf{0.051}^{\pm.002}$ | $\mathbf{2.957}^{\pm.008}$ |
| *w/o* RQ | $0.091^{\pm.001}$ | 58.7 | $0.093^{\pm.004}$ | $3.031^{\pm.009}$ |
| *w/o* QDropout | $0.077^{\pm.000}$ | 39.3 | $0.091^{\pm.003}$ | $2.959^{\pm.008}$ |
| *w/o* RRemask | - | - | $0.063^{\pm.003}$ | $3.049^{\pm.006}$ |
| MoMask ($V$, 0) | $0.091^{\pm.001}$ | 58.7 | $0.093^{\pm.004}$ | $3.031^{\pm.009}$ |
| MoMask ($V$, 1) | $0.069^{\pm.001}$ | 54.6 | $0.073^{\pm.003}$ | $3.031^{\pm.008}$ |
| MoMask ($V$, 2) | $0.049^{\pm.002}$ | 46.0 | $0.072^{\pm.003}$ | $2.978^{\pm.006}$ |
| MoMask ($V$, 3) | $0.037^{\pm.001}$ | 42.5 | $0.064^{\pm.003}$ | $2.970^{\pm.007}$ |
| MoMask ($V$, 4) | $0.027^{\pm.001}$ | 35.3 | $0.069^{\pm.003}$ | $2.987^{\pm.007}$ |
| MoMask ($V$, 5) | $0.019^{\pm.001}$ | 29.5 | $\mathbf{0.051}^{\pm.002}$ | $\mathbf{2.962}^{\pm.008}$ |
| MoMask ($V$, 6) | $0.014^{\pm.001}$ | 26.7 | $0.076^{\pm.003}$ | $2.994^{\pm.007}$ |
| MoMask ($V$, 7) | $\mathbf{0.014}^{\pm.000}$ | **25.3** | $0.084^{\pm.004}$ | $2.968^{\pm.007}$ |
| MoMask ($q$, 0) | $0.077^{\pm.000}$ | 39.3 | $0.091^{\pm.003}$ | $2.959^{\pm.008}$ |
| MoMask ($q$, 0.2) | $\mathbf{0.019}^{\pm.001}$ | **29.5** | $\mathbf{0.051}^{\pm.002}$ | $\mathbf{2.957}^{\pm.008}$ |
| MoMask ($q$, 0.4) | $0.021^{\pm.000}$ | 30.2 | $0.082^{\pm.003}$ | $3.006^{\pm.007}$ |
| MoMask ($q$, 0.6) | $0.024^{\pm.000}$ | 33.2 | $0.053^{\pm.003}$ | $2.946^{\pm.006}$ |
| MoMask ($q$, 0.8) | $0.023^{\pm.000}$ | 33.4 | $0.083^{\pm.004}$ | $3.002^{\pm.008}$ |

Table 4.2: Comparison of MoMask RVQ design vs. motion VQs from previous works.

| Methods | R Precision↑ | | FID↓ | MMDist↓ |
|---|---|---|---|---|
| | Top 1 | Top 3 | | |
| **Ours (B)** | $\mathbf{0.521}^{\pm.002}$ | $\mathbf{0.807}^{\pm.002}$ | $\mathbf{0.045}^{\pm.002}$ | $\mathbf{2.958}^{\pm.008}$ |
| Ours (U) | $0.514^{\pm.003}$ | $0.805^{\pm.003}$ | $0.210^{\pm.008}$ | $3.002^{\pm.009}$ |

Table 4.3: Ablation of bidiretionality on HumanML3D.

# Chapter 5

# RACon: Retrieval-Augmented Simulated Character Locomotion Control

Current generative models, though able to generalize to diverse motions, often pose challenges to the responsiveness of end-users control. To address these issues, in this chapter we introduce **RACon**: **R**etrieval-**A**ugmented Simulated Character Locomotion **Con**trol. Our end-to-end hierarchical reinforcement learning method utilizes a retriever and a motion controller. The retriever searches motion experts from a user-specified database in a task-oriented fashion, which boosts the responsiveness to the user's control. The selected motion experts and the manipulation signal are then transferred to the controller to drive simulated character. In addition, a retrieval-augmented discriminator is designed to stabilize the training process. Our method surpasses existing techniques in both quality and quantity in locomotion control, as demonstrated in our empirical study. Moreover, by switching extensive databases for retrieval, it can adapt to distinctive motion types at run time.

## 5.1 Introduction

Recent advancements in the simulation of virtual character motion can be largely credited to the advent of physics-based deep reinforcement learning methods, such as those outlined in [21]. Despite these advancements, it remains a considerable challenge to create systems that allow end users to conve-

niently manipulate virtual character locomotion while simultaneously ensuring the motions generated are convincingly natural and realistic. Moreover, the system should be capable of seamlessly transitioning to new motion types in real-time.

Motion tracking has been utilized by several recent studies [57] to imitate a library of pre-existing 3D expert motion sequences. Often a motion planner is employed to synchronize expert motions with the character simulation over time, relying on manually constructed pose features [4]. Although these methods have been demonstrated to generate high-fidelity motions, their performance largely depends on the quality and quantity of the available expert motions. These are typically carefully curated and often limited in scope, resulting in mediocre generalization when faced with complex scenarios or large datasets. Their effectiveness diminishes further when dealing with novel motion types or styles [57], [75]. In response to these limitations, recent research has considered stochastic generative models such as GANs [21], [76], [77] and VAEs [56], [105], [107]. These models have produced impressive results, demonstrating their ability to generalize to unseen motion types or scenarios. However, as a black-box framework, these models pose a challenge for end-users who wish to explicitly determine or edit specific poses or skills [77], [107].

The recent successes of Retrieval-Augmented Models (RAM) in Natural Language Processing, showing case its straightforward knowledge acquisition and scalability, has inspired us to employ innovative retrieval and RAM techniques in our motion control context. Our work aims to enhance task responsiveness, generate natural and realistic animations, and alleviate the need to fine-tune a complex controller. Consequently, we propose an end-to-end retrieval-augmented (RA-) solution that utilizes hierarchical reinforcement learning (HRL) for task-oriented learnable retrieval and physics-based character control.

The workflow of our approach is as follows: The direction and velocity specified by the game-pad stick are combined with the current character state and fed into a retrieval policy. This policy predicts an adaptive query, which

then searches for the reference motion from an expert motion database. The controller receives this reference and the goal to generate driven signals for the character's actuators within the simulator. The retrieved and simulated motions are subsequently filtered through a RA-discriminator, adhering to the Generative Adversarial Imitation Learning (GAIL) strategy [36]. This step serves as a motion prior, ensuring the coherence of retrieved motions and the naturalness of simulated ones.

Components such as the goal, prior, and simulated state feedback are fed back to both the retrieval and control modules. Within this context, two reinforcement learning frameworks, task-oriented retrieval and embodied agent manipulation, are seamlessly integrated into our approach. Thanks to our RA-HRL design, our system can generate high-fidelity character motions and adapt to distinctive motion types without extensive professional tuning or selection. Meanwhile, it remains interpretable, providing evidence of retrieved reference motion from user-selected Motion Capture (MoCap) databases (as depicted in Fig. 1.1).

Our contributions are summarized below:

- An end-to-end integrated approach is proposed, which incorporates two RL frameworks as an HRL system, and a RA-discriminator as a training-time motion prior. A set of rewards is designed to jointly optimize the two policies at both the policy level and the system level. This leads to stable run-time performance, and alleviates the mode collapse issue typically found in GAIL models, scaling to large-scale datasets.

- Our approach is demonstrated to drive character locomotions of both high fidelity and diversity, outperforming the state-of-the-art methods both quantitatively and qualitatively. It is additionally capable of transiting to different motion types in-situ, in responding to end-user choices.

## 5.2 Approach

Our method, depicted in Fig.5.1, incorporates two frameworks: task-oriented retrieval Sec. 5.2.1 and embodied agent manipulation Sec. 5.2.2. Specifically, we embed the retrieval framework into the embodied agent manipulation framework, acting as a high-level 'manager' that provides 'guidance' to the controller policy. This arrangement operates within a hierarchical reinforcement learning structure, in response to the given goal signal $g$. The overarching learning objective is to follow the locomotion control $g$ and mimic human behavior, which is realized through the rewards $r^g$ and $r^{\text{pior}}$. The prior reward $r^{\text{pior}}$ is provided by a retrieval-augmented discriminator as detailed in Sec. 5.2.3. We optimize the system in accordance with the principles of GCRL. The forward process is shown in solid arrows, and the state feedback process is shown in dot arrows. Our system incorporates two frameworks, $\langle \pi^{\text{retr}}, \text{Env}^{\text{retr}} \rangle$ and $\langle \pi^{\text{ctrl}}, \text{Env}^{\text{phy}} \rangle$, working in the 'manager-worker' fashion. These two frameworks share the same goal signal $g$ and feedback state $s_{t+1}$ and are trained in an end-to-end manner with policy-level rewards (*i.e.* $\tilde{r}_t^{\text{g}}$, $r_t^{\text{ref}}$) and system-level rewards (*i.e.* $r_t^{\text{g}}$, $r_t^{\text{prior}}$). The optimization details of the retriever and the controller are elaborated in Sec. 5.2.1 and Sec. 5.2.2 respectively. In addition, the yellow dash arrows mark the workflow of retrieval-augmented motion discriminator described in Sec. 5.2.3 which provides a robust motion prior reward for both policies.

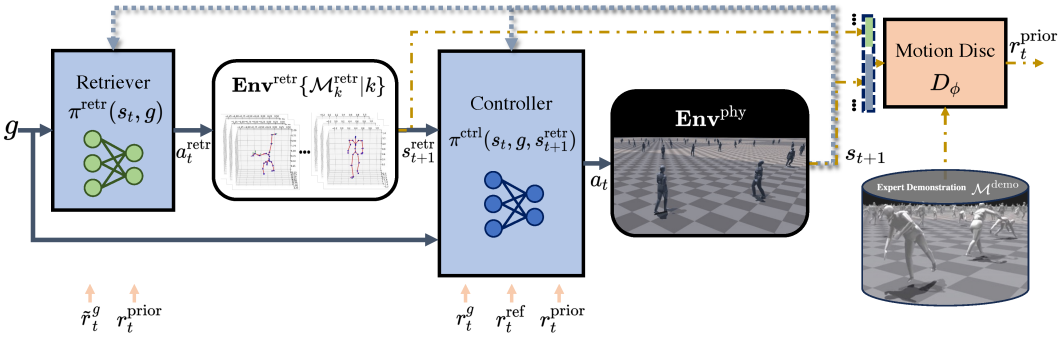



Figure 5.1: The pipeline of the proposed RACon HRL System.

### 5.2.1 Task-Oriented Learnable Retrieval (TOLR)

The idea of motion retrieval revolves around the periodic querying of the database to identify the motion clips that are best compatible in terms of fulfilling user control expectations. The periodically retrieved clips are stitched together to create a complete reference trajectory. Traditional methods like Motion Matching [4] employ hand-crafted features for retrieval, posing a challenge in determining which features are vital for optimal performance whenever there's a modification in the database or the representations (*e.g.*, DoFs). In contrast, our work introduces an innovative data-driven strategy that employs neural networks to formulate queries based on the current character state and task objectives. By harnessing the learning capabilities of neural networks, our approach can automatically identify significant features for motion retrieval, even within massive and disordered databases.

Specifically, TOLR framework illustrated in Fig. 5.2 is considered as a goal-augmented Markov decision process. The *environment* of retrieval is defined as the process of kNN-based $K, Q, V$ search where the given query vector $Q$ is employed to search for the most similar key vector $K$ to find the optimized value $V$. These keys are calculated from the motion clips in advance when building the databases at one time, which are constituted as libraries of key-value pairs in the form $K : V|\mathcal{M}^{\text{retr}}$. At run time, we first extract and compute the raw query in the same manner as the keys $K$, with features such as initial-frame root velocity, clip average velocity, end effector positions, etc. Then, the retriever policy is used to generate a weight vector, working as the adaptive importance of different features, which adjusts the raw query to be the input of the environment $Q$. Our retrieval environment can hold a set of databases $\{\mathcal{M}_k^{\text{retr}}\}$ of different motion types for the user to choose at run time. Given the action $a_t^{\text{retr}}$ as a query $Q$, the environment searches for the most similar key $K$ to index the most suitable value $V$, *i.e.* a motion clip $\tilde{s}_{t+1}$, from the database.

Additionally, the retrieval *environment* can integrate extra plug-in motion databases of preferred types in real-time to perform locomotion in unique

motion styles, which allows our system to execute diverse locomotion skills that are tractable for the end-user.

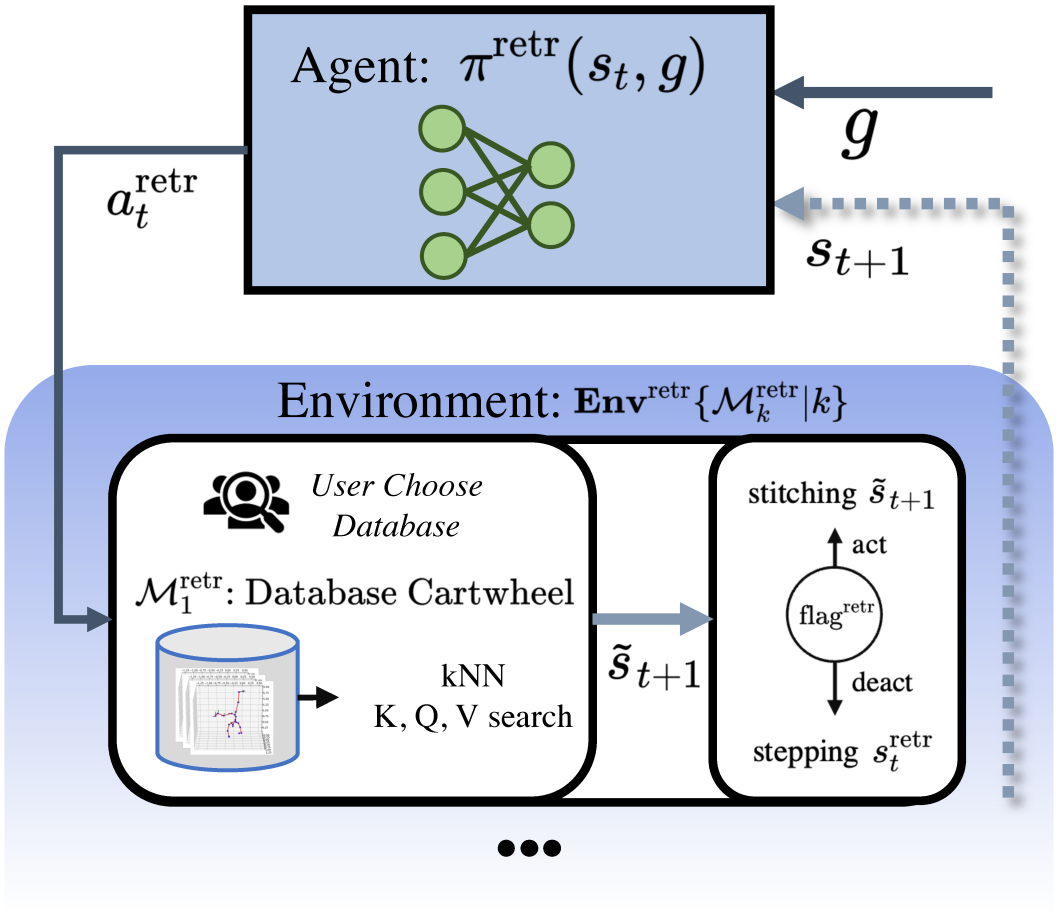



Figure 5.2: The RL framework of Task-Oriented Learnable Retrieval in RACon.

To achieve this, we train the system by randomly alternating retrieval databases of different motion types. We demonstrate this in Fig. 1.1 when incorporating an unseen database at test time.

Our optimization objective has a dual focus: maximizing the capability of the method to accomplish control goals, and enhancing the continuity between sequential retrieved motion clips. This is achieved by employing a goal reward and a prior reward, as follows:

$$r_t^{\text{retr}} = \tilde{w}^{\text{g}} \tilde{r}_t^{\text{g}} + w^{\text{prior}} r_t^{\text{prior}}. \tag{5.1}$$

Here $\tilde{r}_t^{\text{g}} = \exp\big(-d\big(g, \tilde{q}_t\big)\big)$, where $d(\cdot)$ is the function that evaluates the

distance between the current character state and the goal. Specifically, it is the combination of cosine similarity and norm difference of the retrieved trajectory velocity and expected counterpart. $\tilde{q}_t$ is the horizontal root velocity and rotation extracted from the retrieval state $s_t^{\text{retr}}$. $r_t^{\text{prior}}$ is determined by a discriminator, as explained in Sec. 5.2.3.

### 5.2.2 Simulated Character Control

Although TOLR provides a synchronous reference sub-goal $\hat{g}_t^{\text{sub}} \triangleq s_{t+1}^{\text{retr}}$ for the controller $\pi^{ctrl}\left(a_t^{\text{ctrl}}|s_t, g, s_{t+1}^{\text{retr}}\right)$, the learning objective here differs from a mere mimicry term that would require strict adherence to the reference motion and trajectory. We conceptualize our methodology as an *end effector constraint*, allowing the controller to learn the motion types from specific reference motions, without being trapped by them. Therefore, we incorporate only the end effector feature and root rotations of the reference motion to define the reference reward:

$$r_t^{\text{ref}} = w^{\text{h}} r_t^{\text{h}} + w^{\text{rrot}} r_t^{\text{rrot}} + w^{\text{ravel}} r_t^{\text{ravel}} + w^{\text{local}} r_t^{\text{loc}}, \tag{5.2}$$

where

$$\begin{aligned}
r_t^{\text{h}} &= \exp\left(-s^{\text{h}} \|b_t^{\text{h}} - \hat{b}_t^{\text{h}}\|^2\right), \\
r_t^{\text{rrot}} &= \exp\left(-s^{\text{rrot}} \|b_t^{\text{rrot}} \ominus \hat{b}_t^{\text{rrot}}\|^2\right), \\
r_t^{\text{ravel}} &= \exp\left(-s^{\text{ravel}} \|b_t^{\text{ravel}} - \hat{b}_t^{\text{ravel}}\|^2\right), \\
r_t^{\text{loc}} &= \exp\left(-s^{\text{ep}} \sum_{k=1}^{5} \|j_t^k - \hat{j}_t^k\|^2\right).
\end{aligned} \tag{5.3}$$

Here, $\ominus$ signifies the rotation difference. $r_t^{\text{h}}$ represents the root height reward, $r_t^{\text{rrot}}$ denotes the root rotation reward independent of yaw, $r_t^{\text{ravel}}$ is the root angular velocity reward relative to the root coordinate, and $r_t^{\text{loc}}$ measures the error of character body endpoints relative to the root coordinate. $\hat{\cdot}$ indicates the target extracted from expert reference $\hat{g}_t^{\text{sub}}$. We normalize the global horizontal information for root features $b^{\{\text{h,rrot,ravel}\}}$ by registering them to the origin. The endpoints of a body $\{j\}_{k=1}^{5}$ are used for the representation of the character end effector.

Our modified mimicry objective allows for some deviation from the original reference, as it doesn't require the character to replicate the rotation at every joint. While this flexibility might compromise the naturalness of the motion, it benefits the controller responsiveness by offering a more adaptable curriculum. This deviation can then be offset and enhanced through the RA- motion prior $r_t^{\text{prior}}$, as discussed in Sec. 5.2.3. Additionally, we incorporate the locomotion control goal into the reward definition for the controller policy:

$$r_t = w^{\text{g}} r_t^{\text{g}} + w^{\text{ref}} r_t^{\text{ref}} + w^{\text{prior}} r_t^{\text{prior}}, \tag{5.4}$$

where $r_t^{\text{g}} = \exp\big(-d\big(g, q_t\big)\big)$, similar to $\tilde{r}_t^{\text{g}}$, while $q_t$ is from the simulated state $s_t$.

### 5.2.3 Retrieval Augmented Adversarial Motion Prior

Given a motion dataset, our adversarial framework trains a motion discriminator to predict whether a state transition $\langle s_t, s_{t+1} \rangle$ originates from the real dataset or is a generated sample. Our generated samples can be either simulated state transitions, represented as $(s_t, s_{t+1}) \sim \mathcal{M}^{\pi^{\text{ctrl}}}$, or retrieved state transitions denoted as $(s_t, \tilde{s}_{t+1}) \sim \mathcal{M}^{\pi^{\text{retr}}}$. Here, $\tilde{s}_{t+1}$ is the state retrieved from databases based on the simulated state $s_t$. Our motion discriminator then provides a prior reward, $r_t^{\text{prior}}$, which fosters coherence for retrieved motions (Sec. 5.2.1) and naturalness for simulated motions (Sec. 5.2.2). Intriguingly, a composite generated sample can be formed as $(s_t, s_{t+1}, \tilde{s}_{t+2}) \sim \mathcal{M}^{\pi}$, as the retrieved anchor state $\tilde{s}_{t+1}$ can be stepped to produce the subsequent $\tilde{s}_{t+2}$, resulting in a retrieval-augmented term of simulated state transition. This integration of generated samples has proven empirically to enhance the system's performance. In what follows, we discuss the effect of this retrieval-augmented design, namely Retrieval Augmented Adversarial Motion Prior.

Previous GAIL-based methods, such as AMP [77], have shown limitations in terms of stability and efficiency. These methods typically require well-organized, high-quality, and limited-scale datasets, as the generation process might initiate with extremely outlier instances, potentially leading to early-stage model breakdown or unstable learning process. To mitigate these issues,

our design transforms the framework into a *Conditional GAIL*, augmenting the simulated sample features to $\langle s_t, s_{t+1}, \tilde{s}_{t+2} \rangle$ by utilizing the retrieved state $\tilde{s}_{t+2}$. More specifically, the simulated motion and the retrieved expert feature $\tilde{s}_{t+2}$ are concatenated to form a retrieval-augmented simulated sample. In the context of controller training, the RA-term can be seen as a pseudo-label condition drawn from real motion databases and ideally being coherent with the preceding simulated states $\langle s_t, s_{t+1} \rangle$. Empirically, one of the most notable characteristics of real motion samples is temporal smoothness. This property facilitates the discriminator's task in distinguishing fake from real samples and guides the synthesis of spatial pose more easily, by considering the consistency between the simulated states and the RA-term.

We define the training objective for the discriminator as:

$$\begin{aligned} &\arg\min_{\phi} \mathbb{E}_{(s_t, s_{t+1}, s_{t+2}) \sim \mathcal{M}^{\text{demo}}} \left[ \log D_\phi \left( s_t, s_{t+1}, s_{t+2} \right) \right] \\ &+ \mathbb{E}_{(s_t, s_{t+1}, \tilde{s}_{t+2}) \sim \mathcal{M}^{\pi}} \left[ \log \left( 1 - D_\phi \left( s_t, s_{t+1}, \tilde{s}_{t+2} \right) \right) \right] \\ &+ \frac{w^{\text{gp}}}{2} \mathbb{E}_{(s_t, s_{t+1}, s_{t+2}) \sim \mathcal{M}^{\text{demo}}} \| \nabla_\phi D_\phi \left( s_t, s_{t+1}, s_{t+2} \right) \|^2, \end{aligned} \tag{5.5}$$

and the motion prior reward for the two policies as:

$$r_t^{\text{prior}} = -\alpha \log \max \left[ 1 - D_\phi \left( s_t, s_{t+1}, \tilde{s}_{t+2} \right), \epsilon \right], \tag{5.6}$$

where $\alpha$ is a scaling factor to adjust the range of the reward. Here, $\mathcal{M}^{\text{demo}}$ represents the demonstration dataset of real motions, while $\mathcal{M}^{\pi}$ denotes the generated dataset. Given that $\tilde{s}_{t+2}$ is retrieved from the real motion database, it can be viewed as the posterior label condition for the simulated transition $\langle s_t, s_{t+1} \rangle$. Accordingly, we can recast the discriminator within the framework of a Conditional GAIL, formalizing it as $D_\phi \left( s_t, s_{t+1} | \tilde{s}_{t+2} \right)$. It's worth noting that the features could extend beyond three continuous states, incorporating $s_{t-i}$ and $\tilde{s}_{t+2+j}$ in implementation.

## 5.3 Experiments

### 5.3.1 System Workflow

An in-depth view of the training procedural steps in our proposed method is offered in Algorithm 1, detailed using pseudocode for the sake of clarity. This

structured overview elucidates the sequence of actions executed within our system, facilitating the reader's understanding of the underlying mechanics and process flow.

### 5.3.2 Network Architecture

We build all the neural networks, including $D_\phi, \pi^{\text{retr}}, \pi^{\text{ctrl}}$, $V^{\text{retr}}, V^{\text{ctrl}}$, with three-layered MLPs of hidden sizes [1024, 512] and size-specific output layer. Particularly, we set up the policy for retrieval a little different from the common. The architecture of our task-oriented learnable retrieval (TOLR) is depicted in Fig. 5.3. The feature extractor is non-differentiable that extracts and computes feature from motion clips or the current state. The pipeline at the bottom is the actual retrieval policy built by linear layers. As the key $K$ is part of the *environment*, which shouldn't be influenced by the agent, modifying the key through an agent policy — following the traditional setting to create the key and query via identical mapping — would conflict with our TOLR RL framework definition. Consequently, we calculate the key in advance when building databases (*i.e.* $\{K : V | \mathcal{M}^{\text{retr}}\}$) at one time, and exclusively use the policy to generate the query at run time. To preserve the essence of classical retrieval, we design the policy to generate adaptive weights to automatically tune the raw query into $Q$ through a Hadamard product, where the raw query is computed in the same manner as the key.

### 5.3.3 Motion Database

Our demonstration dataset of real motions, $\mathcal{M}^{\text{demo}}$, is sourced from AMASS [65], a comprehensive motion dataset that consolidates various MoCap data resources. We extract over 40,000 seconds of expert motion, unified to 30 fps. The databases for retrieval, $\{\mathcal{M}_k^{\text{retr}}\}$, are also constructed from AMASS. We filter common locomotion movements using category labels from BABEL [82], such as walking, running, turning, and so forth—which amount to approximately 49,000 motion clips, each with a duration of 0.5 seconds. Additionally, we build databases for cartwheel movements ($\sim$6590 clips) and zombie movements (142 clips), respectively. It should be noted that the zombie database is

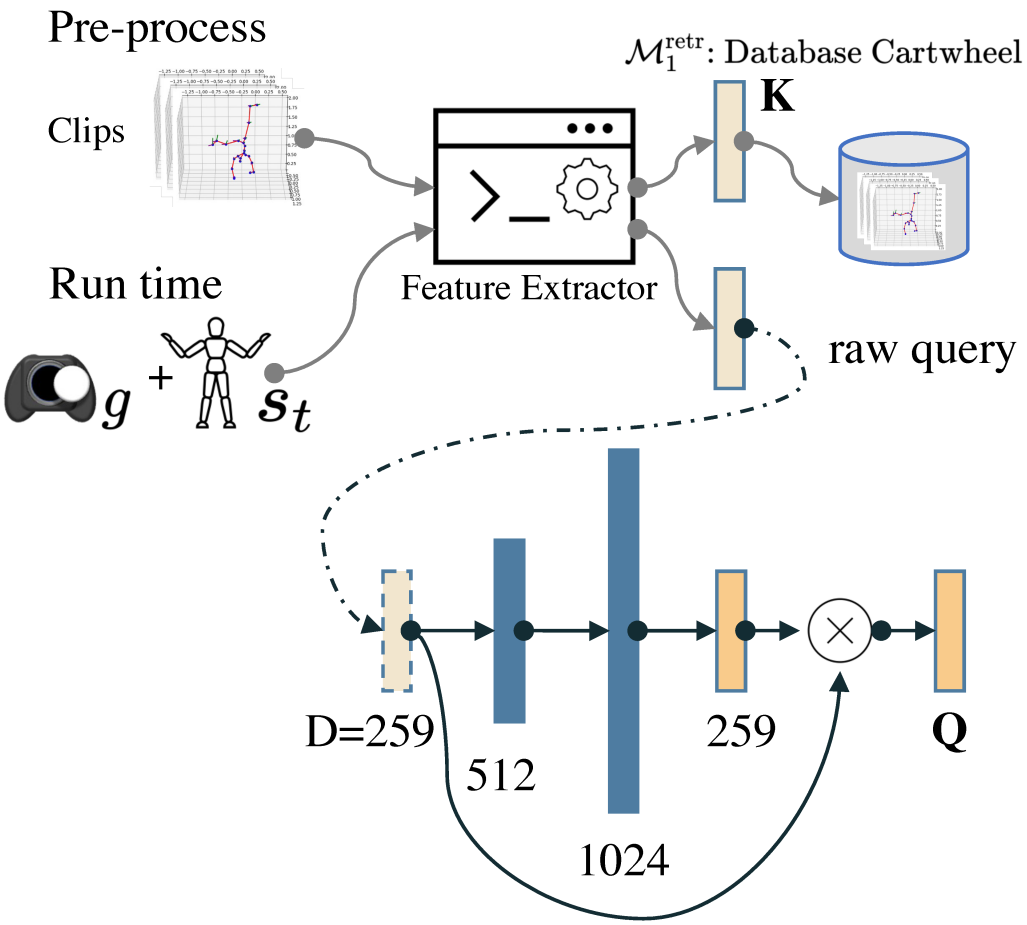



Figure 5.3: The architecture of TOLR in RACon.

solely employed for testing purposes, and its movement is limited to "standing still", to simulate a user-given plugin database at test time.

### 5.3.4 Implementation Details

To ensure a fair comparison and to effectively demonstrate the design of our proposed system, we build $D_\phi, \pi^{\text{retr}}, \pi^{\text{ctrl}}, V^{\text{retr}}, V^{\text{ctrl}}$ with three-layered MLPs of hidden sizes $[1024, 512]$. Both policies are trained with the proximal policy optimization (PPO) method. The discount factor $\gamma$ is set at 0.97, while the learning rate is established at $5 \times 10^{-5}$ with Adam as the optimizer. Our simulation is set up on Isaac Gym, with the fundamental system clock at 30 fps. The methods we are comparing are either properly adapted (*i.e.* AMP [77],

ASE [76]) or independently implemented (*i.e.* DReCon [4]) on the same platform and settings.

### 5.3.5 Simulated Character Model

The Character Model is composed of rigid bodies, a skeleton tree, and joints, each endowed with appropriate dynamic properties such as stiffness. This simulated character is constructed based on prior knowledge of human shape and articulation, as derived from SMPL. Additionally, by utilizing biomedical knowledge, we retarget the elbows and knees to serve as revolute joints with one degree of freedom (DoF). To summarize, our humanoid model contains 17 spherical joints, 4 revolute joints, and a total of 55 actuated DoFs. More details and illustrations would be in the supplementary files. This intricate character model facilitates more expressive and vivid performance while introduces a higher level of control complexity.

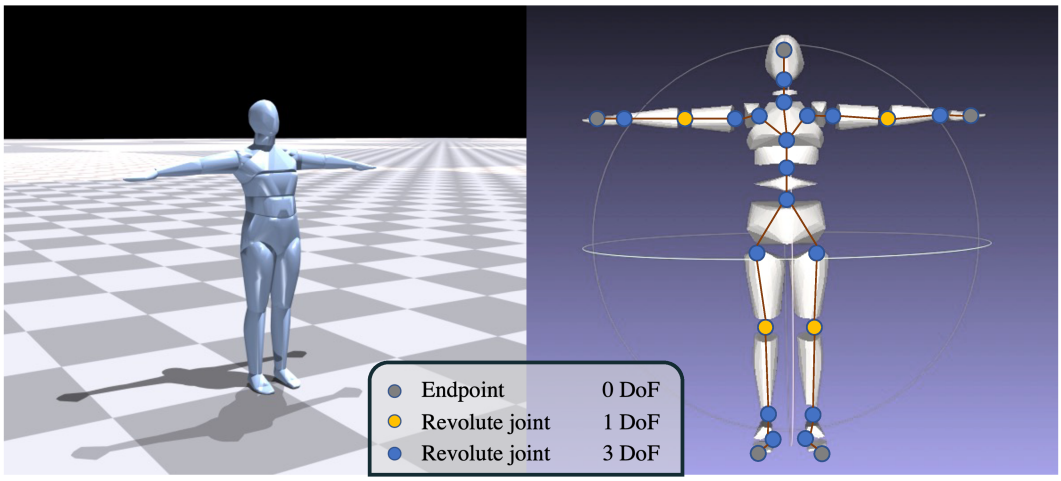



Figure 5.4: The simulated character model.

The character used is illustrated in T-pose in Fig. 5.4 in the simulator (left) and as a toy model (right) with the skeleton tree. Different from the previous method using a capsule-based body, we build an SMPL-convex-fitted body which is closer to real scenario. A fixed density of $985\,kg/m^3$, known as the average human body density, which is also assigned to the rigid bodies. Other properties (*e.g.* velocity and rotation constrain) can refer to our char-

acter model file. We count the Degrees of Freedom excluding the global root descriptor (*i.e.* position and rotation) which is of 6 DoFs but not actuatable. Compared with related work Tab. 5.1 on the simulated humanoid model used, our setup incorporates 55 DoFs simultaneously, which is a much more challenging context while enabling expressive and vivid animation. *In DReCon, 25 actuators are driven by controller policy, while the rest are under open-loop control [4].

Table 5.1: Comparison of the simulated character definition

| Method | Rigid Bodies | Degrees of Freedom |
|---|---|---|
| DReCon [4] | 23 | 25* (48) |
| AMP [77] | 13 | 28 |
| ASE [76] | 13 | 28 |
| Ours | 22 | 55 |

### 5.3.6 Baseline Implementation

For a fair comparison, we adapt the baseline models with minimal changes from their official implementations (except for DReCon [4]), training them on the same data splits. Specifically, our comparative experiment is confined to common locomotion tasks that only include walking, running, and turning, etc. This is because the baseline models are not designed to extend to a diverse range of locomotion skills.

**AMP** [77] & **ASE** [76]. For AMP and ASE which are already set on Isaac Gym [66], we modify the dim of I/O features within the network and environment to accommodate our character model. We also adjust the dataloader with minimum modification. However, we notice that the datasets they used likely have manually set sampling weights for each motion sequence. In our case, our MoCap dataset is too vast for manually tuning the sampling weights. Consequently, we employ uniform sampling weights for them. We also tune the performance following the paper, adjust the discriminator learning, *e.g.* extend the length of discriminator input sample from 2 to 10.

**DReCon** [4]. We re-implement DReCon to control our character model, following an unofficial implementation example written by *C++* and the pa-

per. We also manually tuning the performance following the paper. However, performance may vary due to the use of the simulator and open-loop control solver (*i.e.* Isaac PhysX engine) that differs from the Bullet engine [16] they used. Additionally, it depends on the significant engineering effort involved in empirical parameters, such as tuning motion matching.

### 5.3.7 State Representation and Measurement

**Feature Representations**. The representation of overall system state $s_t = \{b^{\text{h}} \in \mathbb{R}^1, b^{\text{rrot}} \in \mathbb{R}^6, b^{\text{vel}} \in \mathbb{R}^3, b^{\text{avel}} \in \mathbb{R}^3, b^{\text{jrot}} \in \mathbb{R}^{17\times6+4}, b^{\text{javel}} \in \mathbb{R}^{17\times3+4}, j \in \mathbb{R}^{5\times3}\} \in \mathbb{R}^{206}$, which is built from a set of features. These include: root height, rotation, linear velocity and angular velocity, represented in the character's local coordinate frame; local rotation and angular velocity of each joint; 3D positions of hands, feet and head, represented in the character's local coordinate frame. For spherical joints, we convert the raw representation of axis-angle to the 6D-rotation representation[115].

The representation of retrieved state $\tilde{s}_t = \{b^{\text{h}} \in \mathbb{R}^1, b^{\text{rrot}} \in \mathbb{R}^6, b^{\text{vel}} \in \mathbb{R}^3, b^{\text{avel}} \in \mathbb{R}^3, j \in \mathbb{R}^{5\times3}\} \in \mathbb{R}^{28}$. To clarify, $\tilde{s}_t$ is the value $V$ of our $KQV$ searching process. More specifically, it's a expert state sequence $\{\tilde{s}_t, \tilde{s}_{t+1}, \ldots, \tilde{s}_{t+15}\}$ starting from the anchor frame $\tilde{s}_t$.

The representation of raw query and $K$ contains $\{b^{\text{rrot}} \in \mathbb{R}^6, b^{\text{vel}} \in \mathbb{R}^3, b^{\text{avel}} \in \mathbb{R}^3, b^{\text{jrawrot}} \in \mathbb{R}^{17\times3+4}, g^{\text{xzdir}} \in \mathbb{R}^2, g^{\text{speed}} \in \mathbb{R}^1, g^{\text{facedir}} \in \mathbb{R}^2\} \in \mathbb{R}^{72}$, where $g^{\{\text{xzdir,speed,facedir}\}}$ can be calculated from user control signal and expert clips, corresponded to raw query and $K$. $Q$ is then generated by the retrieval policy from raw query as shown in Fig. 5.3

**Measurement Function**. The distance function $d(\cdot)$ to measure the goal reward is formed as follows:

$$\begin{aligned} d(\cdot) =& exp(-10\|b^{\text{speed}} - \hat{b}^{\text{speed}}\|^2 - 5\tan(\theta^{\text{yaw}} - \hat{\theta}^{\text{yaw}})) \\ &+ b^{\text{facedir}} \cdot \hat{b}^{\text{facedir}}, \end{aligned} \tag{5.7}$$

where $b^{\text{speed}}$ is the character root velocity along xOz plain and projected to the target velocity direction, "$\cdot$" is the dot product to calculate the cosine similarity of facing direction. We also provided the code in the supplement.

The distance function used for KQV searching is the dot product between keys. We utilize an off-the-shelf knn-based searcher library Scalable Nearest Neighbors (ScaNN) to perform the matching efficiently. We set the number of leaves (2,000), the number of leaves to search (100), training sample size (20,000), anisotropic quantization threshold (0.2) accordingly. It takes around 10 seconds to build the searching tree, depending on the size of the database, and less than 5 milliseconds to search the key for a given query.

### 5.3.8 Results

In the experiment results, we first qualitatively evaluate our method by performing Cartwheel and Common Locomotion, such as walk and run, and switching motion types at run time. Then, a comprehensive evaluation is conducted on the common locomotion task, compared with the state-of-the-art methods. Lastly, the ablation study for our RA-GAIL and TOLR is presented to further demonstrate the efficacy of our design.


Figure 5.5: Example results of locomotion control with skills of Cartwheel and Common Locomotion.

**Cartwheel and Common Locomotion Qualitative Results.** Controlling a simulated character in stable and balanced body states such as walk&run is comparatively straightforward since the center of mass (CoM) is typically

positioned between the feet. In contrast, performing a 'cartwheel' presents a more complex locomotion challenge.

Our results, demonstrating the transition between common locomotion and a cartwheel, are presented in Fig. 5.5 showing in 5 fps. The game-pad control signal is marked with a blue stick. The color of the simulated character marks different retrieval databases $\{\mathcal{M}_k^{\text{retr}}|k\}$ chosen by users at run time. (Light Grey: Common Locomotion. Brown: Cartwheel.) Initially, the character is guided by the reference retrieved from the common locomotion database. We then shift to the cartwheel database in $\mathbf{Env}^{\text{retr}}$ by simply switching the lookup dictionary, indicated by the color of brown. We provide further qualitative results in the supplementary video.

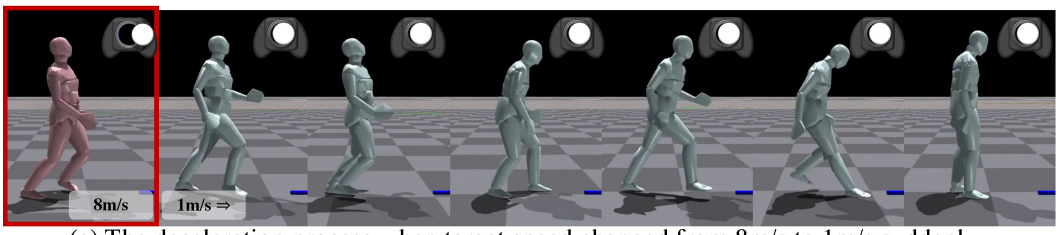
Without any additional planning required, our motion switching performance at run-time showcases the flexibility and adaptability of our system. Notably, our proposed approach showcases the extraordinary versatility of even a lightweight MLP controller, which can cover a diverse range of distinctive modes. Additionally, the simulated character is able to execute specific styles while simultaneously reaching the user-designated goal, further underscoring the responsiveness of our approach.

Table 5.2: Quantitative comparison on the basic locomotion setting for RACon.

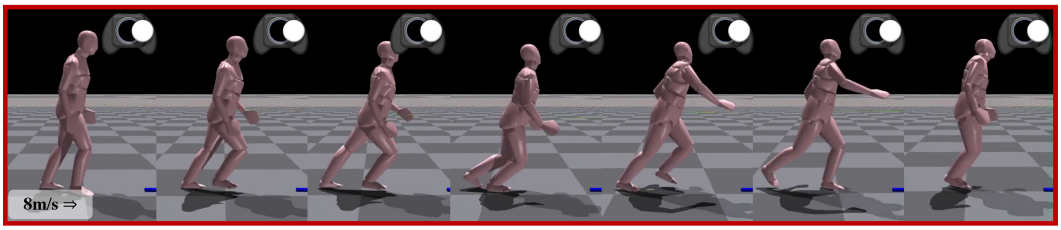
| Method | MVE(m/s) ↓ | TRate(%) ↓ | Len(%) ↑ | FID ↓ | MModality ↗ |
|---|---|---|---|---|---|
| Real Motion | - | - | - | 0.0113 | 1.0271 |
| DReCon (TOG'19) | 3.361 | 28.06 | 82.98 | 7.998 | **3.386** |
| AMP (TOG'21) | 3.551 | 5.00 | 96.01 | **3.453** | 1.096 |
| ASE (TOG'22) | 3.472 | 1.65 | 98.52 | 4.212 | 0.727 |
| Ours | **2.709** | **0.44** | **99.69** | **3.453** | 1.380 |
| Ours ($s^{\text{retr}}$) | 2.837 | - | - | - | - |

**Common Locomotion Quantitative Comparisons.** We show quantitative results of common locomotion control in Table 5.2, compared with DReCon [4], AMP [77] and ASE [76].

We employ various evaluation metrics including Mean Velocity Error (MVE), Termination Rate (TRate), Episode Length (Len), Fréchet Inception Distance (FID), and Multimodality (MModality).




(a) The deceleration process when target speed changed from 8m/s to 1m/s suddenly.




(b) The acceleration process when target speed changed from 1m/s to 8m/s suddenly.

Figure 5.6: Generalization ability and response latency of RACon.

The robustness of our proposed system is evidenced by the lowest TRate and highest Len. This accomplishment is partly due to the similar GAIL training scheme used by AMP, ASE, and our approach, which significantly surpasses DReCon. Furthermore, our method's use of the task-oriented retriever allows it to achieve the smallest MVE amongst the comparisons, suggesting that our proposal efficiently responds to the locomotion control signal. We evaluated intermediate retrieved states ($s^{\text{retr}}$) from our method, revealing a small MVE, which shows an automatic improvement in retrieval performance achieved by task-oriented optimization. Importantly, our reference mimic objective isn't tailored to exactly replicate the expert motion but instead to provide end effector information, thus enabling the system (Ours) to outperform the retrieved expert (Ours $s^{\text{retr}}$). The substantial absolute value of MVE can be attributed to latency stemming from the natural acceleration and deceleration process, as depicted in Fig. 5.6. We set an extreme case of high and low speed to 8m/s and 1m/s. The simulated character in red indicates that it is currently under the goal with a higher speed, while it switches to a lower speed when turning to green. (a) shows a spontaneous deceleration process, the character sticks out a foot to "brake"; (b) shows an acceleration process, the character leans forward to accelerate strongly. These cases demonstrate our superior

performance in typical locomotion control scenarios.

In terms of motion diversity and fidelity, a comprehensive assessment of MultiModality performance necessitates the consideration of the FID metric as well. The generated motion is expected to closely resemble real motion while demonstrating a comparable diversity. While DReCon can control motion with diverse appearance, it often results in unnatural outcomes with high FID.

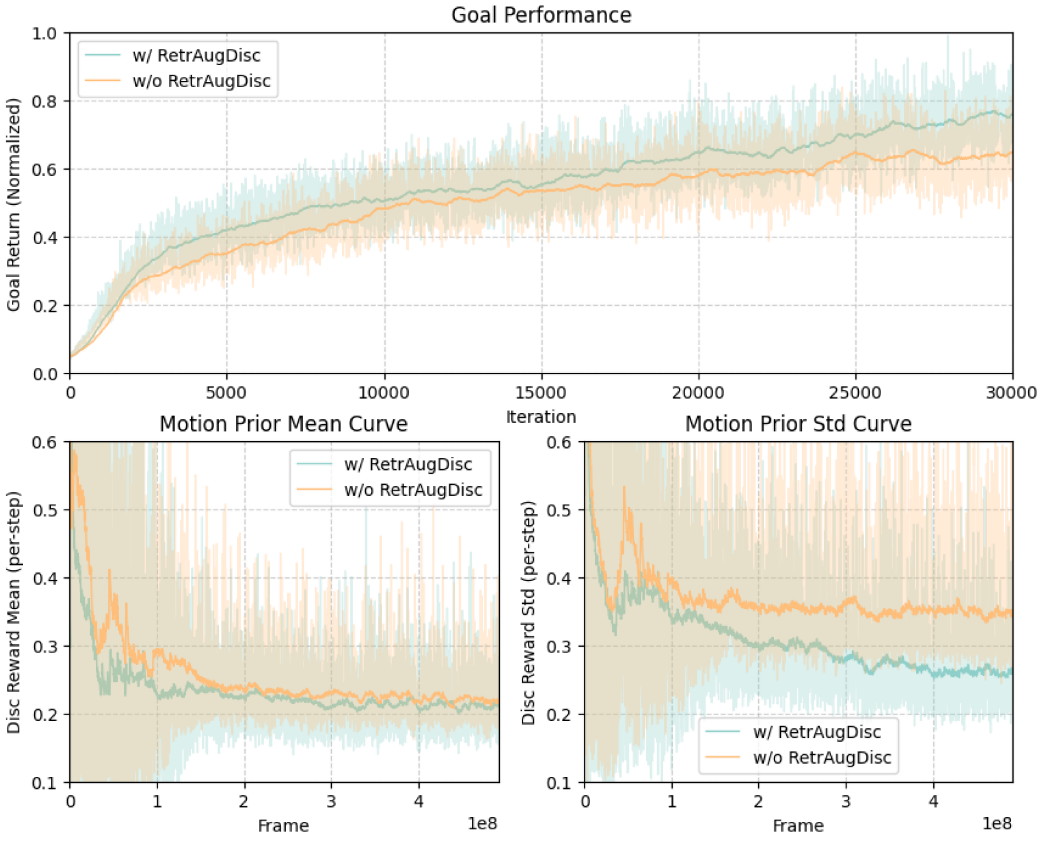



Figure 5.7: Ablation study on GAIL.

### 5.3.9 Hyperparameter

We list in Tab. 5.3 the hyperparameter used in our experiment for further reference. Each set of reward weights $\{w\}$ will be normalized to 1 when performing a weighted sum by default. $\{s\}$ are scaling factors when transforming errors into exponential reward values.

Table 5.3: The hyperparameter used in RACon experiment.

| Param | Value | Param | Value |
|---|---|---|---|
| $\tilde{w}^g$ | 0.2 | $w^{\text{g}}$ | 0.1 |
| $w^{\text{prior}}$ | 0.8 | $w^{\text{ref}}$ | 0.1 |
| $w^{\text{h}}$ | 1 | $w^{\text{prior}}$ | 0.8 |
| $w^{\text{rrot}}$ | 5 | $w^{\text{gp}}$ | 5 |
| $w^{\text{ravel}}$ | 3 | $\alpha$ | 2 |
| $w^{\text{local}}$ | 10 | $\gamma$ | 0.97 |
| $s^{\text{h}}$ | 2.5 | $\tau$ | 0.95 |
| $s^{\text{rrot}}$ | 1.25 | $lr$ | 5.0e-5 |
| $s^{\text{ravel}}$ | 7.5 | $lr_{\mathcal{D}}$ | 5.0e-3 |
| $s^{\text{ep}}$ | 75 | $i\|\langle s_{t+1-i}, \mathcal{D}\rangle$ | range(8) |
| $s^{\text{h}}$ | 0.2 | $j\|\langle s_{t+2+j}, \mathcal{D}\rangle$ | range(2) |
| $s^{\text{rot}}$ | 0.2 | retr step | 15 |

We show the additional quantitative results of common locomotion tasks for effect of hyper-parameter in Tab. 5.4.

**Effect of $\gamma$.** $\gamma$ is the reward discount factor which is of great important in reinforcement learning application, commonly range from 0.9 to 0.999. Our experiment shows that when $\gamma$ is large, the model gain the lowest failure rate while it struggles to maintain responsive control over the movement velocity. While small $\gamma$ leads to better response according to the smaller MVE but fails more. TRate refers to the long-term planning compared with the relative short-term velocity control. It is coherent with the common observation of $\gamma$ in reinforcement learning applications.

**Effect of $w^{\text{prior}}$.** $w^{\text{prior}}$ weighs how important to follow the motion-prior distribution. Small prior reward weights $w^{\text{prior}}$ would result in a high failure rate, which verifies the importance of prior reward for robustness. On the other side, large prior reward weights might diminish the effect of goal reward, showing unsatisfied MVE.

### 5.3.10 Ablation Study

**Dataset** We conducted additional experiments to evaluate the messiness of our used dataset AMASS [65], compared to the dataset CMU MoCap [8] commonly used in the baselines. Firstly, we include far more motion clips in our

Table 5.4: Effect of hyper-parameter and design for RACon.

| Method | MVE(m/s) ↓ | TRate(%) ↓ | Len(%) ↑ | FID ↓ | MModality ↗ |
|---|---|---|---|---|---|
| Dataset (AMASS[65]) | - | - | - | 0.0113 | 1.0271 |
| Dataset (CMU MoCap[8]) | - | - | - | 0.0097 | 0.6904 |
| Ours($s^{\text{retr}}$) | 2.837 | - | - | - | - |
| w/o learnable retriever | **2.387** | - | - | - | - |
| Ours ($\gamma = 0.97, w^{\text{prior}} = 0.8$) | 2.709 | 0.44 | 99.69 | 3.453 | 1.380 |
| w/o learnable retriever | 3.170 | 8.71 | 91.54 | 4.178 | 1.593 |
| w/o RA-GAIL | 2.610 | 4.37 | 95.37 | 3.770 | 1.716 |
| $\gamma = 0.99$ | 4.97 | **0.31** | **99.86** | **3.199** | 1.017 |
| $\gamma = 0.95$ | 2.683 | 6.47 | 95.20 | 3.972 | 1.892 |
| $w^{\text{prior}} = 0.1$ | 3.97 | 23.70 | 78.02 | 5.943 | **3.520** |
| $w^{\text{prior}} = 0.5$ | 2.713 | 0.64 | 99.29 | 3.738 | 1.439 |
| $w^{\text{prior}} = 1.0$ | 3.86 | 0.39 | 99.62 | 3.340 | 1.153 |

training setting compared with the delicately selected CMU MoCap used in the baselines original implementation[76], [77]. Secondly, Tab. 5.4 shows a larger diversity of the dataset we used according to the MModality. The higher FID score implies a relatively chaotic motion composition. This dataset setting builds a more challenging but practical scenario for applications.

**Module** The task-oriented learnable retriever (TOLR) policy is disabled to retrieve by the raw query. We also show in Tab. 5.4 the intermediate state of the system, $s^{\text{retr}}$. Using the hand-crafted raw query improves the MVE of $s^{\text{retr}}$. However, it's crucial to recognize that the task's primary concern revolves around the simulated state. The model without TOLR is mediocre in regard to the overall system performance.

We also evaluated the model trained with RA-GAIL or GAIL scheme. GAN-based method, modeling the overall distribution of the entire dataset—which aligns with the FID's motivation-results in a favorable FID score. Based on empirical evidence, controllers trained with GAIL may become stagnant, repeatedly generating the most stable motion clip, which is referred to as mode-collapse in GAN. It is more evident in the supplementary video that baseline methods suffer from replanting in the face of massive databases and complicated skeletal structures. In addition to the training curves presented in the manuscript, which is pivotal for illustrating the impact of RL training strategies, we also provide the corresponding quantitative results in Tab. 5.4. The

model trained without RA-GAIL is observed to gain a better MVE score than it's counterpart. It might be due to the replacement of $\tilde{r}^{\text{retr}}$ in $r^{\text{retr}}$. We cannot utilize the original prior term for training the retrieval policy, as it doesn't have any direct involvement in the outputs with GAIL. Instead, we introduce a slight smoothing term for compensation. Consequently, the retriever and the entire system may shift their emphasis towards the goal reward associated with MVE. The increased termination rate could potentially stem from this change or the potentially unstable training process of standard GAIL, as evident in the training curves in the manuscript. Due to the tightly integrated system and training strategy, it's not feasible to implement the system without some form of additional effect.

We provide further evidence of the strength of the retrieval-augmented adversarial prior, demonstrating its ability to promote more stable training. Fig. 5.7 presents the training curves of policies learned on a straightforward locomotion task (using the common locomotion database only), comparing those with and without RetrAugDisc. The Goal Performance shows the normalized overall goal return of an episode. The discriminator reward mean and std are per-step values. For the sake of consistency and simplicity in our experiments, we employ a standard unlearnable retriever to query the retrieved expert clips. The policy trained with a retrieval-augmented adversarial prior not only learns faster but also shows a higher goal return in the end. Our policy, trained with a retrieval-augmented adversarial prior, yields a smaller training reward variance, indicating a more stable training process.

## 5.4 Limitation

Our method may not perform well when faced with severe perturbations, and our model is not designed to recover from a fall, see Fig. 5.8. This is an issue that our model does not explore enough trajectory, *e.g.* falling down. In the future, it is possible to give the policy more chance to explore rare situations using empirical settings or novel RL approaches. We might also try to teach our retriever to recover using our task-oriented learning scheme. Another

issue is choppy stride animation. We observe unnatural, quick short steps when our model encounters situations such as sharp turns or imbalance. Our future plans include introducing penalties for these undesirable movements. The initial step towards this will be to identify the statistical features of these anomalies.

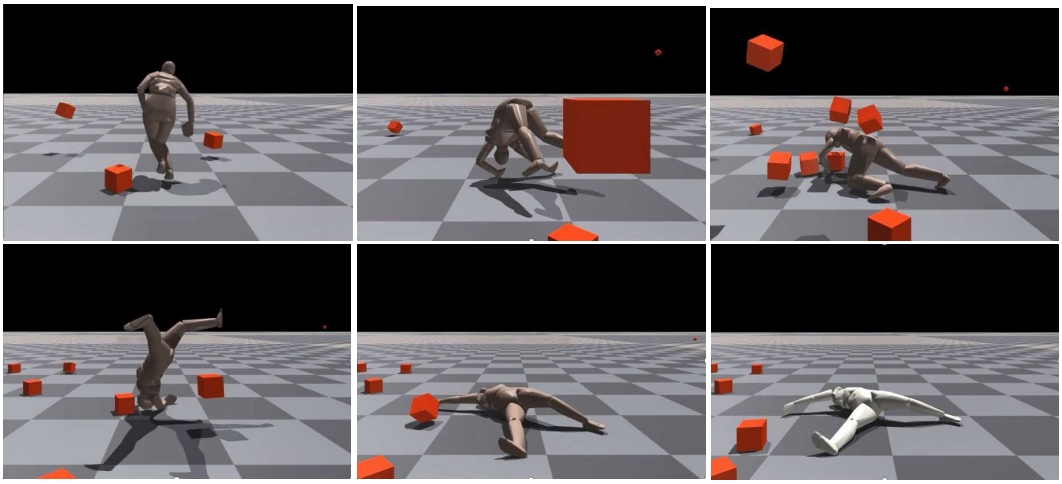


Figure 5.8: Failure Cases of RACon.

**Algorithm 1** Training with RA-HRL

**input** $\mathcal{M}^{\text{demo}}$: dataset of real motion demonstration
**input** $\{\mathcal{M}_k^{\text{retr}}|k\}$: $k$ databases of different motion types
$D_\phi \leftarrow$ initialize discriminator
$\pi^{\text{retr}} \leftarrow$ initialize retriever
$\pi^{\text{ctrl}} \leftarrow$ initialize controller
$V^{\text{retr}}, V^{\text{ctrl}} \leftarrow$ initialize value functions

**while** not done **do**
    $\mathcal{B}^{\text{retr}}, \mathcal{B}^{\text{ctrl}} \leftarrow \emptyset$ initialize experience buffers
    **for** trajectory $i = 1, ..., n$ **do**
        **for** time step $t = 0, ..., T-1$ **do**
            $\tilde{s}_{t+1} \leftarrow \langle \pi^{\text{retr}}(s_t, g), \mathbf{Env}^{\text{retr}}\{\mathcal{M}_k^{\text{retr}}|k\}\rangle$
            **if** flag$^{\text{retr}}$ **then**
                $s_{t+1}^{\text{retr}} \leftarrow \tilde{s}_{t+1}$
            **else**
                $s_{t+1}^{\text{retr}} \leftarrow$ stepping $s_t^{\text{retr}}$
            **end if**
            $s_{t+1} \leftarrow \langle \pi^{\text{ctrl}}(s_t, g, s_{t+1}^{\text{retr}}), \mathbf{Env}^{\text{phy}}\rangle$
            Calculate rewards $\{r_t^g, \tilde{r}_t^g, r_t^{\text{ref}}, r_t^{\text{prior}}\}$ accordingly
            $r_t^{\text{retr}} \leftarrow \tilde{w}^{\text{g}} \tilde{r}_t^{\text{g}} + w^{\text{prior}} r_t^{\text{prior}}$
            $r_t \leftarrow w^{\text{g}} r_t^{\text{g}} + w^{\text{ref}} r_t^{\text{ref}} + w^{\text{prior}} r_t^{\text{prior}}$
            record $(s_t; a_t^{\text{retr}}; g; r_t^{\text{retr}})$ in $\tau_i^{\text{retr}}$
            record $(s_t; a_t; g, s_{t+1}^{\text{retr}}; r_t)$ in $\tau_i^{\text{ctrl}}$
            store transitions $(s_t; s_{t+1}; \tilde{s}_{t+2})$ in $\mathcal{M}^\pi$
        **end for**
    **end for**
    store $\{\tau_i^{\text{retr}}\}_{i=1}^n, \{\tau_i^{\text{ctrl}}\}_{i=1}^n$ in $\mathcal{B}^{\text{retr}}, \mathcal{B}^{\text{ctrl}}$

    Update discriminator:
    **for** update step $= 1, ..., n$ **do**
        $m^{\text{demo}} \leftarrow$ sample batch $(s_t; s_{t+1}; s_{t+2})$ from $\mathcal{M}^{\text{demo}}$
        $m^\pi \leftarrow$ sample batch $(s_t; s_{t+1}; \tilde{s}_{t+2})$ from $\mathcal{M}^\pi$
        update $D_\phi$ according to Eq. (5.5) using $m^{\text{demo}}$ & $m^\pi$
    **end for**

    update $V^{\text{retr}}$ and $\pi^{\text{retr}}$ using data from $\mathcal{B}^{\text{retr}}$
    update $V^{\text{ctrl}}$ and $\pi^{\text{ctrl}}$ using data from $\mathcal{B}^{\text{ctrl}}$
**end while**

# Chapter 6

# Conclusion and Future Work

## 6.1 Discussion and Conclusion

In this thesis, we have explored the generation of 3D human character motion conditioned on text and game-pad-controlled simulated environments. By adopting generative modeling approaches, we developed intelligent 3D skeletal motion performers that learn from human motion knowledge. Our work emphasized the stochastic nature of motions, enabling the generation of lifelike character movements in both kinematic and physically constrained dynamics environments.

Our kinematic-based motion generation method, MoMask, established a state-of-the-art text-to-motion framework, demonstrating swift performance and inherent frame editing capabilities. MoMask features three advanced techniques: residual quantization for precise motion quantization, masked transformer, and residual transformer for high-quality and faithful motion generation. MoMask is efficient and flexible, achieving superior performance without extra inference burden, and effortlessly supporting temporal motion inpainting in multiple contexts. This approach highlighted the distinct characteristics of motion compared to natural language and images, and enhanced motion quality through a residual vector quantization variational autoencoder. It leverages cutting-edge techniques like masked transformers for generative motion modeling and residual vector quantization for a compact yet lossless representation of rotational skeletal motion continuity. We found that Diffusion Models aren't ideal for sequence generation, and while GPTs are powerful,

they can be overly complex. The masked transformer, acting as a nonlinear language model, proves to be both efficient and effective in our context. In terms of fidelity and efficacy, MoMask emerges as a potentially superior modeling paradigm for motion generation in current times.

Furthermore, our hierarchical reinforcement learning framework, RACon, for game-pad-controlled simulated character control successfully integrated task-oriented motion retrieval and a model-based controller in an end-to-end manner. This retrieval-augmented hierarchical reinforcement learning system is not only interpretable but also demonstrates exceptional generalizability across diverse motion types. It allowed for seamless character locomotion and real-time switching of movement styles using a game-pad. The retrieval-augmented generative adversarial imitation learning pipeline further enhanced training stability and performance. It represents a small step towards large-scale models for general motion control tasks by leveraging extensive motion datasets. The end-to-end trainable task-oriented retrieval also reduces the human effort required for feature crafting in conventional motion-matching pipelines. This advancement has the potential to positively influence a broad spectrum of applications, including virtual reality, animation, robotics, and more.

While our methods show promise, there are limitations that need to be addressed in future work. These include improving the robustness of motion generation in diverse and complex environments, and expanding the range of control signals beyond text and game-pad inputs. Nevertheless, the advancements presented in this thesis open new avenues for applications in augmented reality, virtual reality, and robotics, paving the way for more natural and versatile 3D character motion generation.

## 6.2 Future Work

Given that kinematic-based motion foundation models are within reach, future work in the field would be directed towards building large control models for simulated character manipulation.

One potential direction is expanding current models with more modalities and data [54], such as learning from large-scale video datasets or informative inertial sensor data sequences. This approach could lead to a more generalizable motion generation or control model.

Another direction is cascading kinematic-based motion foundation models with universal humanoid controllers, which precisely drive the simulated character to perform the given poses. The current state-of-the-art universal humanoid controllers [62], [63] can imitate almost all the motions in AMASS dataset [65] roughly. There are an increasing number of applications in the community making text-to-motion models with these unified controllers to perform different tasks as wished. The future work in this direction would be to enhance the control precision of the universal controllers/imitators and make end-to-end fine-tuning possible to comprehensively promote the complex system performance.

Ultimately, it could involve directly constructing large control models using advanced reinforcement learning paradigms. The efficiency gap between reinforcement learning and supervised learning is critical to their performance disparity. Recent advancements in reinforcement learning, particularly in offline imitation learning, could offer a promising path forward. The reinforcement learning problem could be conceptualized as a state-action trajectory planning problem [98], leveraging robust sequence modeling methods like trajectory transformers or diffusion models. However, action information is commonly unavailable. Acquiring action parameters corresponding to the state trajectory is sometimes challenging or time-consuming. Designing a better action-state trajectory for acquiring workflow would be helpful in future research. Furthermore, trying to get rid of action with state-only learning is also a possible direction. Additionally, though the modeling capacity of the diffusion model is larger than ever model before, the diffusion model requires much more time for inference due to its progressive denoising process. Moreover, the diffusion models' synchronized denoising process does not fit the principle of the Markov process or sequence problem well. Improving diffusion models to be specialized for motion control tasks or developing even suitable generative

models would be a future solution.

We anticipate that large control models will prove invaluable in developing policies for effectively controlling complex virtual reality characters and potentially even robots.

# References


[1] C. Ahuja and L.-P. Morency, "Language2pose: Natural language grounded pose forecasting," in *2019 International Conference on 3D Vision (3DV)*, IEEE, 2019, pp. 719–728.

[2] A. Aristidou, D. Cohen-Or, J. K. Hodgins, Y. Chrysanthou, and A. Shamir, "Deep motifs and motion signatures," *ACM Transactions on Graphics (TOG)*, vol. 37, no. 6, pp. 1–13, 2018.

[3] N. Athanasiou, M. Petrovich, M. J. Black, and G. Varol, "Teach: Temporal action composition for 3d humans," in *2022 International Conference on 3D Vision (3DV)*, IEEE, 2022, pp. 414–423.

[4] K. Bergamin, S. Clavet, D. Holden, and J. R. Forbes, "Drecon: Data-driven responsive control of physics-based characters," *ACM Transactions On Graphics (TOG)*, vol. 38, no. 6, pp. 1–11, 2019.

[5] Z. Borsos, R. Marinier, D. Vincent, *et al.*, "Audiolm: A language modeling approach to audio generation," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 2023.

[6] D. Cai, Y. Wang, H. Li, W. Lam, and L. Liu, "Neural machine translation with monolingual translation memory," *arXiv preprint arXiv:2105.11269*, 2021.

[7] H. Cai, C. Bai, Y.-W. Tai, and C.-K. Tang, "Deep video generation, prediction and completion of human action sequences," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 366–382.

[8] "Carnegie mellon university - CMU graphics lab - motion capture library." (), [Online]. Available: `http://mocap.cs.cmu.edu/faqs.php` (visited on 08/30/2023).

[9] P. Cervantes, Y. Sekikawa, I. Sato, and K. Shinoda, "Implicit neural representations for variable length human motion generation," in *European Conference on Computer Vision*, Springer, 2022, pp. 356–372.

[10] H. Chang, H. Zhang, J. Barber, *et al.*, "Muse: Text-to-image generation via masked generative transformers," *arXiv preprint arXiv:2301.00704*, 2023.

[11] H. Chang, H. Zhang, L. Jiang, C. Liu, and W. T. Freeman, "Maskgit: Masked generative image transformer," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 11 315–11 325.

[12] R. Chen, M. Shi, S. Huang, P. Tan, T. Komura, and X. Chen, "Taming diffusion probabilistic models for character control," *arXiv preprint arXiv:2404.15121*, 2024.

[13] X. Chen, B. Jiang, W. Liu, *et al.*, "Executing your commands via motion diffusion in latent space," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 18 000–18 010.

[14] N. Chentanez, M. Müller, M. Macklin, V. Makoviychuk, and S. Jeschke, "Physics-based motion capture imitation with deep reinforcement learning," in *Proceedings of the 11th ACM SIGGRAPH Conference on Motion, Interaction and Games*, 2018, pp. 1–10.

[15] C. Chi, S. Feng, Y. Du, *et al.*, "Diffusion policy: Visuomotor policy learning via action diffusion," *arXiv preprint arXiv:2303.04137*, 2023.

[16] E. Coumans, "Bullet physics simulation," in *ACM SIGGRAPH 2015 Courses*, ACM, 2015, p. 1.

[17] E. Coumans and Y. Bai, *Pybullet, a python module for physics simulation for games, robotics and machine learning*, `http://pybullet.org`, 2016–2021.

[18] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.

[19] J. Di Carlo, P. M. Wensing, B. Katz, G. Bledt, and S. Kim, "Dynamic locomotion in the mit cheetah 3 through convex model-predictive control," in *2018 IEEE/RSJ international conference on intelligent robots and systems (IROS)*, IEEE, 2018, pp. 1–9.

[20] Z. Dou, X. Chen, Q. Fan, T. Komura, and W. Wang, "C· ase: Learning conditional adversarial skill embeddings for physics-based characters," in *SIGGRAPH Asia 2023 Conference Papers*, 2023, pp. 1–11.

[21] A. Escontrela, X. B. Peng, W. Yu, *et al.*, "Adversarial motion priors make good substitutes for complex reward functions," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2022, pp. 25–32. DOI: `10.1109/IROS47612.2022.9981973`.

[22] S. Ferdowsi, S. Voloshynovskiy, and D. Kostadinov, "Regularized residual quantization: A multi-layer sparse dictionary learning approach," *arXiv preprint arXiv:1705.00522*, 2017.

[23] *Forward Kinematics — ROS Robotics — rosroboticslearning.com*, `https://www.rosroboticslearning.com/forward-kinematics`, [Accessed 07-06-2024].

[24] A. Ghosh, N. Cheema, C. Oguz, C. Theobalt, and P. Slusallek, "Synthesis of compositional animations from textual descriptions," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 1396–1406.

[25] K. Gong, D. Lian, H. Chang, *et al.*, "Tm2d: Bimodality driven 3d dance generation via music-text integration," *arXiv preprint arXiv:2304.02419*, 2023.

[26] I. Goodfellow, J. Pouget-Abadie, M. Mirza, *et al.*, "Generative adversarial nets," *Advances in neural information processing systems*, vol. 27, 2014.

[27] J. Gu, Y. Wang, K. Cho, and V. O. Li, "Search engine guided neural machine translation," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2018.

[28] C. Guo, Y. Mu, M. G. Javed, S. Wang, and L. Cheng, "Momask: Generative masked modeling of 3d human motions," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 1900–1910.

[29] C. Guo, Y. Mu, X. Zuo, *et al.*, "Generative human motion stylization in latent space," *arXiv preprint arXiv:2401.13505*, 2024.

[30] C. Guo, S. Zou, X. Zuo, *et al.*, "Generating diverse and natural 3d human motions from text," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 5152–5161.

[31] C. Guo, X. Zuo, S. Wang, and L. Cheng, "Tm2t: Stochastic and tokenized modeling for the reciprocal generation of 3d human motions and texts," in *European Conference on Computer Vision*, Springer, 2022, pp. 580–597.

[32] C. Guo, X. Zuo, S. Wang, *et al.*, "Action2motion: Conditioned generation of 3d human motions," in *Proceedings of the 28th ACM International Conference on Multimedia*, 2020, pp. 2021–2029.

[33] C. Guo, X. Zuo, S. Wang, *et al.*, "Action2video: Generating videos of human 3d actions," *International Journal of Computer Vision*, vol. 130, no. 2, pp. 285–315, 2022.

[34] K. He, X. Chen, S. Xie, Y. Li, P. Dollár, and R. Girshick, "Masked autoencoders are scalable vision learners," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 16 000–16 009.

[35] N. Heess, D. TB, S. Sriram, *et al.*, "Emergence of locomotion behaviours in rich environments," *arXiv preprint arXiv:1707.02286*, 2017.

[36] J. Ho and S. Ermon, "Generative adversarial imitation learning," *Advances in neural information processing systems*, vol. 29, 2016.

[37] J. Ho, A. Jain, and P. Abbeel, "Denoising diffusion probabilistic models," *Advances in Neural Information Processing Systems*, vol. 33, pp. 6840–6851, 2020.

[38] J. Ho and T. Salimans, "Classifier-free diffusion guidance," *arXiv preprint arXiv:2207.12598*, 2022.

[39] S. Hong, D. Han, K. Cho, J. S. Shin, and J. Noh, "Physics-based full-body soccer motion control for dribbling and shooting," *ACM Transactions on Graphics (TOG)*, vol. 38, no. 4, pp. 1–12, 2019.

[40] R. Huang, H. Hu, W. Wu, K. Sawada, M. Zhang, and D. Jiang, "Dance revolution: Long-term dance generation with music via curriculum learning," *arXiv preprint arXiv:2006.06119*, 2020.

[41] B. Jiang, X. Chen, W. Liu, J. Yu, G. Yu, and T. Chen, "Motiongpt: Human motion as a foreign language," *arXiv preprint arXiv:2306.14795*, 2023.

[42] J. Kim, J. Kim, and S. Choi, "Flame: Free-form language-based motion synthesis & editing," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 37, 2023, pp. 8255–8263.

[43] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," *arXiv preprint arXiv:1312.6114*, 2013.

[44] H. Kong, K. Gong, D. Lian, M. B. Mi, and X. Wang, "Priority-centric human motion generation in discrete latent space," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023, pp. 14 806–14 816.

[45] J. Lee, M. X. Grey, S. Ha, *et al.*, "Dart: Dynamic animation and robotics toolkit," *The Journal of Open Source Software*, vol. 3, no. 22, p. 500, 2018.

[46] K. Lee, M.-W. Chang, and K. Toutanova, "Latent retrieval for weakly supervised open domain question answering," *arXiv preprint arXiv:1906.00300*, 2019.

[47] S. Lee, M. Park, K. Lee, and J. Lee, "Scalable muscle-actuated human simulation and control," *ACM Transactions On Graphics (TOG)*, vol. 38, no. 4, pp. 1–13, 2019.

[48] P. Lewis, E. Perez, A. Piktus, *et al.*, "Retrieval-augmented generation for knowledge-intensive nlp tasks," *Advances in Neural Information Processing Systems*, vol. 33, pp. 9459–9474, 2020.

[49] H. Li, Y. Su, D. Cai, Y. Wang, and L. Liu, "A survey on retrieval-augmented text generation," *arXiv preprint arXiv:2202.01110*, 2022.

[50] T. Li, H. Chang, S. Mishra, H. Zhang, D. Katabi, and D. Krishnan, "Mage: Masked generative encoder to unify representation learning and image synthesis," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 2142–2152.

[51] Y. Li, W. Ding, C. Liu, B. Zhang, and G. Guo, "Trq: Ternary neural networks with residual quantization," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 35, 2021, pp. 8538–8546.

[52] Z. Li, B. Ni, W. Zhang, X. Yang, and W. Gao, "Performance guaranteed network acceleration via high-order residual quantization," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 2584–2592.

[53] A. S. Lin, L. Wu, R. Corona, K. Tai, Q. Huang, and R. J. Mooney, "Generating animated videos of human activities from natural language descriptions," *Learning*, vol. 2018, no. 1, 2018.

[54] J. Lin, A. Zeng, S. Lu, *et al.*, "Motion-x: A large-scale 3d expressive whole-body human motion dataset," *Advances in Neural Information Processing Systems*, vol. 36, 2024.

[55] J. Lin, J. Chang, L. Liu, *et al.*, "Being comes from not-being: Open-vocabulary text-to-motion generation with wordless training," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 23 222–23 231.

[56] H. Y. Ling, F. Zinno, G. Cheng, and M. Van De Panne, "Character controllers using motion vaes," *ACM Transactions on Graphics (TOG)*, vol. 39, no. 4, pp. 40–1, 2020.

[57] L. Liu, M. V. D. Panne, and K. Yin, "Guided learning of control graphs for physics-based characters," *ACM Transactions on Graphics (TOG)*, vol. 35, no. 3, pp. 1–14, 2016.

[58] M. Liu, M. Zhu, and W. Zhang, "Goal-conditioned reinforcement learning: Problems and solutions," *arXiv preprint arXiv:2201.08299*, 2022.

[59] Z. Liu, S. Wu, S. Jin, *et al.*, "Investigating pose representations and motion contexts modeling for 3d motion prediction," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 45, no. 1, pp. 681–697, 2022.

[60] Y. Lou, L. Zhu, Y. Wang, X. Wang, and Y. Yang, "Diversemotion: Towards diverse human motion generation via discrete diffusion," *arXiv preprint arXiv:2309.01372*, 2023.

[61] T. Lucas, F. Baradel, P. Weinzaepfel, and G. Rogez, "Posegpt: Quantization-based 3d human motion generation and forecasting," in *European Conference on Computer Vision*, Springer, 2022, pp. 417–435.

[62] Z. Luo, J. Cao, K. Kitani, W. Xu, *et al.*, "Perpetual humanoid control for real-time simulated avatars," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023, pp. 10 895–10 904.

[63] Z. Luo, J. Cao, J. Merel, *et al.*, "Universal humanoid motion representations for physics-based control," *arXiv preprint arXiv:2310.04582*, 2023.

[64] M. Macklin, K. Storey, M. Lu, *et al.*, "Small steps in physics simulation," in *Proceedings of the 18th Annual ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, 2019, pp. 1–7.

[65] N. Mahmood, N. Ghorbani, N. F. Troje, G. Pons-Moll, and M. J. Black, "Amass: Archive of motion capture as surface shapes," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2019, pp. 5442–5451.

[66] V. Makoviychuk, L. Wawrzyniak, Y. Guo, *et al.*, "Isaac gym: High performance gpu-based physics simulation for robot learning," *arXiv preprint arXiv:2108.10470*, 2021.

[67] W. Mao, M. Liu, M. Salzmann, and H. Li, "Learning trajectory dependencies for human motion prediction," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 9489–9497.

[68] J. Martinez, H. H. Hoos, and J. J. Little, "Stacked quantizers for compositional vector compression," *arXiv preprint arXiv:1411.2173*, 2014.

[69] I. Millington, *Game physics engine development*. CRC Press, 2007.

[70] T. M. Mitchell, "Generative and discriminative classifiers: Naive bayes and logistic regression," *Machine learning*, vol. 1, no. 2, pp. 1–17, 2010.

[71] V. Mnih, K. Kavukcuoglu, D. Silver, *et al.*, "Human-level control through deep reinforcement learning," *nature*, vol. 518, no. 7540, pp. 529–533, 2015.

[72] Y. Mu, S. Zou, K. Yin, *et al.*, "Racon: Retrieval-augmented simulated character locomotion control," *arXiv preprint arXiv:2406.17795*, 2024.

[73] A. Ng and M. Jordan, "On discriminative vs. generative classifiers: A comparison of logistic regression and naive bayes," *Advances in neural information processing systems*, vol. 14, 2001.

[74] S. Park, H. Ryu, S. Lee, S. Lee, and J. Lee, "Learning predict-and-simulate policies from unorganized human motion data," *ACM Transactions on Graphics (TOG)*, vol. 38, no. 6, pp. 1–11, 2019.

[75] X. B. Peng, P. Abbeel, S. Levine, and M. Van de Panne, "Deepmimic: Example-guided deep reinforcement learning of physics-based character skills," *ACM Transactions On Graphics (TOG)*, vol. 37, no. 4, pp. 1–14, 2018.

[76] X. B. Peng, Y. Guo, L. Halper, S. Levine, and S. Fidler, "Ase: Large-scale reusable adversarial skill embeddings for physically simulated characters," *ACM Transactions On Graphics (TOG)*, vol. 41, no. 4, pp. 1–17, 2022.

[77] X. B. Peng, Z. Ma, P. Abbeel, S. Levine, and A. Kanazawa, "Amp: Adversarial motion priors for stylized physics-based character control," *ACM Transactions on Graphics (TOG)*, vol. 40, no. 4, pp. 1–20, 2021.

[78] M. Petrovich, M. J. Black, and G. Varol, "Action-conditioned 3d human motion synthesis with transformer vae," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 10 985–10 995.

[79] M. Petrovich, M. J. Black, and G. Varol, "Temos: Generating diverse human motions from textual descriptions," in *European Conference on Computer Vision*, Springer, 2022, pp. 480–497.

[80] M. Plappert, C. Mandery, and T. Asfour, "The kit motion-language dataset," *Big data*, vol. 4, no. 4, pp. 236–252, 2016.

[81] M. Plappert, C. Mandery, and T. Asfour, "Learning a bidirectional mapping between human whole-body motion and natural language using deep recurrent neural networks," *Robotics and Autonomous Systems*, vol. 109, pp. 13–26, 2018.

[82] A. R. Punnakkal, A. Chandrasekaran, N. Athanasiou, A. Quiros-Ramirez, and M. J. Black, "Babel: Bodies, action and behavior with english labels," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 722–731.

[83] A. Radford, J. W. Kim, C. Hallacy, *et al.*, "Learning transferable visual models from natural language supervision," in *International Conference on Machine Learning*, PMLR, 2021, pp. 8748–8763.

[84] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, I. Sutskever, *et al.*, "Language models are unsupervised multitask learners," *OpenAI blog*, vol. 1, no. 8, p. 9, 2019.

[85] J. Ren, M. Zhang, C. Yu, X. Ma, L. Pan, and Z. Liu, "Insactor: Instruction-driven physics-based characters," *Advances in Neural Information Processing Systems*, vol. 36, 2024.

[86] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer, "High-resolution image synthesis with latent diffusion models," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 10 684–10 695.

[87] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv preprint arXiv:1707.06347*, 2017.

[88] Y. Shi, J. Wang, X. Jiang, and B. Dai, "Controllable motion diffusion model," *arXiv preprint arXiv:2306.00416*, 2023.

[89] L. Siyao, W. Yu, T. Gu, *et al.*, "Bailando: 3d dance generation by actor-critic gpt with choreographic memory," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 11 050–11 059.

[90] J. Song, C. Meng, and S. Ermon, "Denoising diffusion implicit models," *arXiv preprint arXiv:2010.02502*, 2020.

[91] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.

[92] R. Tedrake and the Drake Development Team, *Drake: Model-based design and verification for robotics*, 2019. [Online]. Available: `https://drake.mit.edu`.

[93] C. Tessler, Y. Kasten, Y. Guo, S. Mannor, G. Chechik, and X. B. Peng, "Calm: Conditional adversarial latent models for directable virtual characters," in *ACM SIGGRAPH 2023 Conference Proceedings*, 2023, pp. 1–9.

[94] G. Tevet, B. Gordon, A. Hertz, A. H. Bermano, and D. Cohen-Or, "Motionclip: Exposing human motion generation to clip space," in *European Conference on Computer Vision*, Springer, 2022, pp. 358–374.

[95] G. Tevet, S. Raab, B. Gordon, Y. Shafir, D. Cohen-Or, and A. H. Bermano, "Human motion diffusion model," *arXiv preprint arXiv:2209.14916*, 2022.

[96] E. Todorov, T. Erez, and Y. Tassa, "Mujoco: A physics engine for model-based control," in *2012 IEEE/RSJ international conference on intelligent robots and systems*, IEEE, 2012, pp. 5026–5033.

[97] F. Torabi, G. Warnell, and P. Stone, "Behavioral cloning from observation," *arXiv preprint arXiv:1805.01954*, 2018.

[98] T. E. Truong, M. Piseno, Z. Xie, and C. K. Liu, "Pdp: Physics-based character animation via diffusion policy," *arXiv preprint arXiv:2406.00960*, 2024.

[99] J. Tseng, R. Castellon, and K. Liu, "Edge: Editable dance generation from music," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 448–458.

[100] A. Van Den Oord, O. Vinyals, *et al.*, "Neural discrete representation learning," *Advances in Neural Information Processing Systems*, vol. 30, 2017.

[101] N. Wagener, A. Kolobov, F. V. Frujeri, R. Loynd, C.-A. Cheng, and M. Hausknecht, "Mocapact: A multi-task dataset for simulated humanoid control," *arXiv preprint arXiv:2208.07363*, 2022.

[102] C. Wang, S. Chen, Y. Wu, *et al.*, "Neural codec language models are zero-shot text to speech synthesizers," *arXiv preprint arXiv:2301.02111*, 2023.

[103] Z. Wang, P. Yu, Y. Zhao, *et al.*, "Learning diverse stochastic human-action generators by learning smooth latent transitions," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, 2020, pp. 12 281–12 288.

[104] J. Won, D. Gopinath, and J. Hodgins, "A scalable approach to control diverse behaviors for physically simulated characters," *ACM Transactions on Graphics (TOG)*, vol. 39, no. 4, pp. 33–1, 2020.

[105] J. Won, D. Gopinath, and J. Hodgins, "Physics-based character controllers using conditional vaes," *ACM Transactions on Graphics (TOG)*, vol. 41, no. 4, pp. 1–12, 2022.

[106] F. Xiao, L. Pang, Y. Lan, Y. Wang, H. Shen, and X. Cheng, "Transductive learning for unsupervised text style transfer," *arXiv preprint arXiv:2109.07812*, 2021.

[107] H. Yao, Z. Song, B. Chen, and L. Liu, "Controlvae: Model-based learning of generative controllers for physics-based characters," *ACM Transactions on Graphics (TOG)*, vol. 41, no. 6, pp. 1–16, 2022.

[108] H. Yao, Z. Song, Y. Zhou, T. Ao, B. Chen, and L. Liu, "Moconvq: Unified physics-based motion control via scalable discrete representations," *arXiv preprint arXiv:2310.10198*, 2023.

[109] L. Yu, Y. Cheng, K. Sohn, *et al.*, "Magvit: Masked generative video transformer," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 10 459–10 469.

[110] N. Zeghidour, A. Luebs, A. Omran, J. Skoglund, and M. Tagliasacchi, "Soundstream: An end-to-end neural audio codec," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 30, pp. 495–507, 2021.

[111] J. Zhang, Y. Zhang, X. Cun, *et al.*, "T2m-gpt: Generating human motion from textual descriptions with discrete representations," *arXiv preprint arXiv:2301.06052*, 2023.

[112] M. Zhang, Z. Cai, L. Pan, *et al.*, "Motiondiffuse: Text-driven human motion generation with diffusion model," *arXiv preprint arXiv:2208.15001*, 2022.

[113] M. Zhang, X. Guo, L. Pan, *et al.*, "Remodiffuse: Retrieval-augmented motion diffusion model," *arXiv preprint arXiv:2304.01116*, 2023.

[114] Y. Zhang, D. Huang, B. Liu, *et al.*, "Motiongpt: Finetuned llms are general-purpose motion generators," *arXiv preprint arXiv:2306.10900*, 2023.

[115] Y. Zhou, C. Barnes, J. Lu, J. Yang, and H. Li, "On the continuity of rotation representations in neural networks," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 5745–5753.

[116] Z. Zhou and B. Wang, "Ude: A unified driving engine for human motion generation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 5632–5641.

[117] B. D. Ziebart, A. L. Maas, J. A. Bagnell, A. K. Dey, *et al.*, "Maximum entropy inverse reinforcement learning.," in *Aaai*, Chicago, IL, USA, vol. 8, 2008, pp. 1433–1438.