MOGO: Residual Quantized Hierarchical Causal Transformer for High-Quality and Real-Time 3D Human Motion Generation

Dongjie Fu[1*]
dongjie.fu@mogo.ai

Tengjiao Sun[2*]
t.s.sun@soton.ac.uk

Pengcheng Fang[2*]
p.fang@soton.ac.uk

Xiaohao Cai[2]
x.cai@soton.ac.uk

Hansung Kim[2†]
h.kim@soton.ac.uk

[1]**Mogo AI, China**
[2]University of Southampton, United Kingdom
[*]Equal contribution [†]Corresponding author

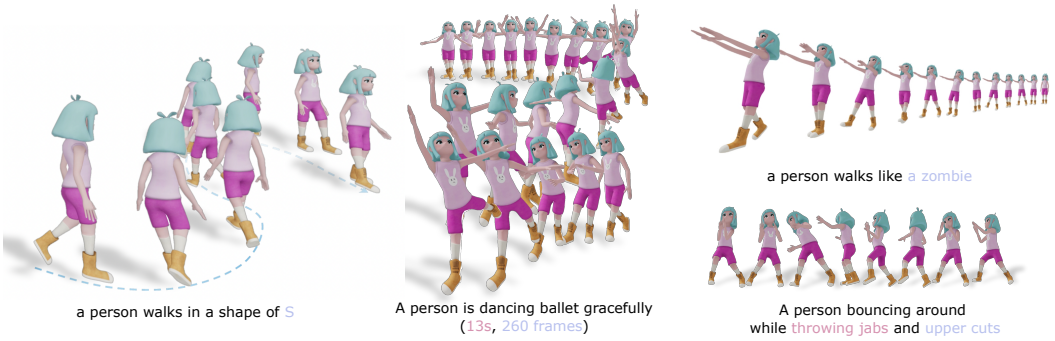


Figure 1: Illustration of MOGO, a GPT-style model for high-quality text-to-motion generation. The model supports open-vocabulary prompts and can generate long, lifelike 3D motion sequences from natural language inputs.

Abstract

Recent advances in transformer-based text-to-motion generation have led to impressive progress in synthesizing high-quality human motion. Nevertheless, jointly achieving high fidelity, streaming capability, real-time responsiveness, and scalability remains a fundamental challenge. In this paper, we propose MOGO (Motion Generation with One-pass), a novel autoregressive framework tailored for efficient and real-time 3D motion generation. MOGO comprises two key components: (1) MoSA-VQ, a motion scale-adaptive residual vector quantization module that hierarchically discretizes motion sequences with learnable scaling to produce compact yet expressive representations; and (2) RQHC-Transformer, a residual quantized hierarchical causal transformer that generates multi-layer motion tokens in a single forward pass, significantly reducing inference latency. To enhance semantic fidelity, we further introduce a text condition alignment mechanism that improves motion decoding under textual control. Extensive experiments on benchmark datasets including HumanML3D, KIT-ML, and CMP demonstrate that MOGO achieves competitive or superior generation quality compared to state-of-the-art transformer-

based methods, while offering substantial improvements in real-time performance, streaming generation, and generalization under zero-shot settings.

1 Introduction

Text-to-motion generation is a rapidly evolving research area with growing importance in virtual environments, such as gaming, AR/VR, and humanoid robotics [1]. Recent advances have leveraged large language model (LLM)-based techniques to generate high-quality 3D human motion from text descriptions, typically using vector-quantized variational autoencoders (VQ-VAE) and autoregressive decoding strategies [2–5]. Meanwhile, diffusion-based approaches have also demonstrated strong generation performance, particularly in terms of motion fidelity and diversity [6].

Despite these achievements, both diffusion- and LLM-based frameworks face practical limitations [7–10]. Diffusion models often rely on iterative refinement processes, which introduce significant inference latency and hinder their suitability for real-time or interactive applications [7]. On the other hand, LLM-based models, although autoregressive, typically involve large parameter sizes and long context dependencies, leading to high memory and computation costs that challenge deployment on lightweight or streaming scenarios [9]. A broader discussion of related research trends is provided in Section 4.

To address these issues, we propose *MOGO* (*M*otion *G*eneration with *O*ne-pass), a transformer-based autoregressive framework that directly predicts motion tokens in a single forward pass. Compared to diffusion-based models, MOGO enables streamable motion decoding without iterative sampling, significantly reducing inference time. Furthermore, unlike LLM-based motion generators, MOGO adopts a lightweight yet expressive transformer architecture with considerably fewer parameters—allowing for real-time token generation while maintaining high motion fidelity.

MOGO is composed of two key components: (1) *MoSA-VQ* (Motion Scale-Adaptive Residual VQ-VAE), which discretizes motion sequences via hierarchical quantization and learnable scaling, producing compact and adaptive representations [11]; and (2) *RQHC-Transformer* (Residual Quantized Hierarchical Causal Transformer), which autoregressively generates multi-layer motion tokens in a single pass [12]. As shown in Figure 1, this design allows for the synthesis of coherent and extended motion sequences while preserving streaming capability and generalization across diverse prompts and datasets.

Extensive experiments demonstrate that MOGO achieves superior results on both in-distribution and out-of-distribution scenarios, outperforming state-of-the-art LLM-based motion generators in quality, inference speed, and scalability—making it a robust and real-time-capable solution for practical text-to-motion generation.

Our main contributions:

- *MOGO*: A unified transformer-based framework for one-pass high-fidelity motion generation. It achieves generation quality comparable to state-of-the-art LLM- and diffusion-based methods, while significantly improving real-time efficiency and enabling streamable motion synthesis.
- *MoSA-VQ*: A motion discretization module that employs hierarchical quantization and learnable feature scaling for compact and expressive representations.
- *RQHC-Transformer*: A lightweight hierarchical causal transformer that autoregressively generates multi-layer motion tokens in one forward pass.

All code and demonstration pages will be made publicly available on our GitHub repository upon acceptance.

2 Methods

We introduce MOGO, a unified framework designed for efficient and expressive text-to-motion generation. Mogo could generate a high-quality motion sequence from a textual description, where each frame $\mathbf{m}_i \in \mathbb{R}^D$ represents a D-dimensional human pose. As shown in Figure 2, our MOGO

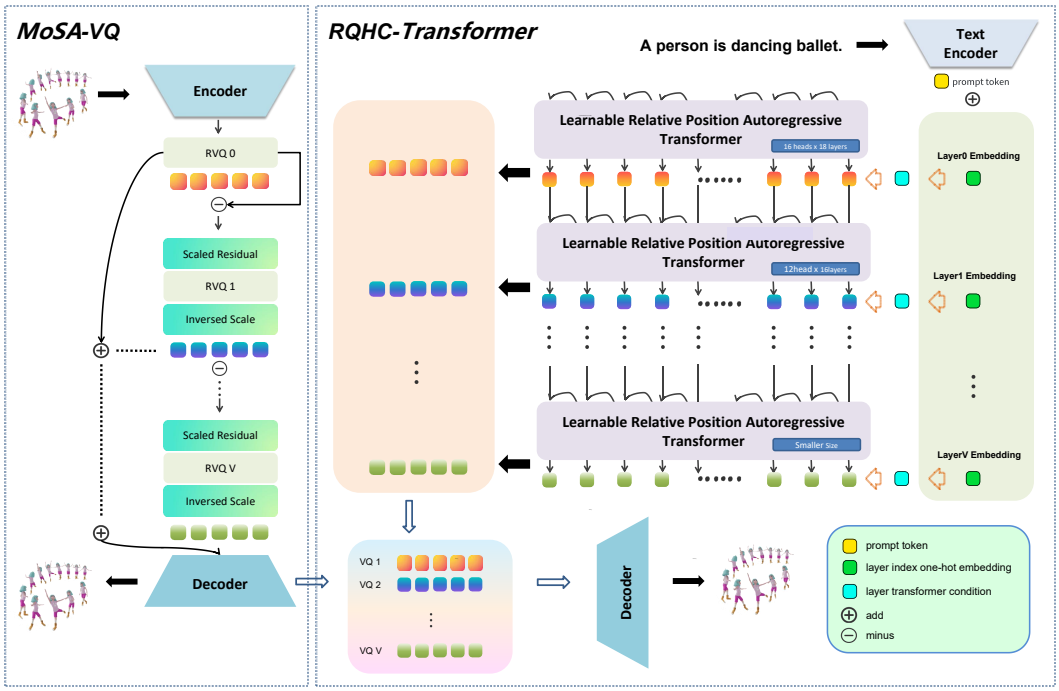


Figure 2: Overview of the proposed MOGO framework. *Left*: We first encode the motion sequence using a hierarchical MoSA-VQ module with learnable feature scaling, indexing a multi-level codebook. *Right*: Then, tokens are autoregressively generated by a RQHC-Transformer with relative positional attention. Finally, the predicted token sequence is quantized and decoded into a full 3D human motion sequence.

framework consists of two core components: (1) MoSA-VQ (left) that discretizes motion into multi-level tokens with high precision, and (2) RQHC-Transformer (right) leveraging hierarchical causal forward computation to generate motion tokens collectively in a streamlined manner (algorithms are shown in Appendix C). To enhance zero/few-shot performance, we incorporate text condition alignment (TCA) for improved motion decoding.

2.1 MoSA-VQ: Motion Scale-Adaptive Residual VQ-VAE

To effectively transform continuous human motion sequences into discrete representations suitable for autoregressive modeling, we adopt a residual vector quantized variational autoencoder (RVQ-VAE) architecture. This module, termed *MoSA-VQ*, serves as the foundation for learning compact yet expressive motion codes. Residual quantization enables progressive refinement of motion features across multiple levels, capturing both coarse and fine-grained temporal dynamics essential for high-fidelity generation.

However, previous RVQ-based approaches [2, 5] often rely on manually calibrated feature ranges, which limit their adaptability and quantization efficiency in complex motion settings. To address this, we introduce a novel *learnable feature scaling* mechanism, allowing the quantization space to dynamically adapt at each hierarchy level. This design not only eliminates the need for manual tuning but also improves reconstruction accuracy and codebook utilization.

Hierarchical Quantization with Feature Scaling. Motion encoder outputs a latent representation $\tilde{\mathbf{b}} \in \mathbb{R}^{n \times d}$, where n is the number of motion tokens and d is the feature dimension. At each level $l \in \{0, 1, \ldots, L\}$, we perform feature scaling before residual quantization. Let $\mathbf{r}^0 = \tilde{\mathbf{b}}$ be the residual feature at level 0. We first compute a scaled version of the residual feature:

$$\mathbf{r}^l_{\text{scaled}} = \|\mathbf{s}^l\| \cdot \mathbf{r}^l + \mathbf{b}^l, \tag{1}$$

where $\mathbf{s}^l \in \mathbb{R}^d$ and $\mathbf{b}^l \in \mathbb{R}^d$ are learnable scale and bias parameters. The input $\mathbf{r}^l$ is the residual feature passed from the previous quantization level and represents the information that has not yet been captured.

We then apply vector quantization to the scaled residual:

$$\mathbf{b}^l_{\text{scaled}} = \mathcal{Q}(\mathbf{r}^l_{\text{scaled}}), \tag{2}$$

where $\mathcal{Q}(\cdot)$ maps each input to the nearest codebook entry. This quantized output exists in the scaled feature space, so we invert the transformation to project it back into the original residual space:

$$\mathbf{b}^l = (\mathbf{b}^l_{\text{scaled}} - \mathbf{b}^l)/\|\mathbf{s}^l\|. \tag{3}$$

The recovered quantized vector $\mathbf{b}^l$ captures the portion of the motion representation explained at level v, and we update the residual for the next layer by subtracting this contribution:

$$\mathbf{r}^{l+1} = \mathbf{r}^l - \mathbf{b}^l. \tag{4}$$

This process repeats hierarchically across all levels, resulting in a set of quantized vectors $\{\mathbf{b}^0, \mathbf{b}^1, \ldots, \mathbf{b}^L\}$. The final quantized latent representation is reconstructed by summing contributions from all layers:

$$\hat{\mathbf{b}} = \sum_{l=0}^{L} \mathbf{b}^l, \tag{5}$$

which is subsequently passed to the decoder to generate the full motion sequence.

Training Objective. To guide the hierarchical quantization process toward informative and stable motion representations, we jointly optimize a multi-term objective designed to promote accurate reconstruction, effective codebook usage, and numerical stability. The overall loss function is defined as:

$$\mathcal{L}_{\text{rvq}} = \|\mathbf{m} - \hat{\mathbf{m}}\|_1 + \beta \sum_{l=1}^{L} \|\mathbf{r}^l - \text{sg}[\mathbf{b}^l]\|_2^2 + \gamma \sum_{l=0}^{L} \left(\|\mathbf{s}^l - 1\|_2^2 + \|\mathbf{b}^l\|_2^2 \right), \tag{6}$$

where $\text{sg}[\cdot]$ denotes the stop-gradient operator. The first term encourages accurate motion reconstruction, the second enforces codebook commitment, and the third stabilizes training by constraining the scale and bias parameters to remain close to identity mappings.

To preserve fine-grained temporal details throughout the encoding and decoding process, we employ stride-1 convolutions in both the encoder and decoder. This design avoids excessive downsampling, which can introduce high-frequency aliasing artifacts and degrade motion continuity.

Learnable scaling allows the quantization layers to focus on encoding residuals at different abstraction levels, with deeper layers specializing in finer motion refinements. Such a design increases the expressiveness and interpretability of the latent representation, contributing to higher reconstruction quality and better hierarchical decomposition of motion dynamics.

2.2 RQHC-Transformer: Residual Quantized Hierarchical Causal Transformer

multi-level token representation via MoSA-VQ, the next challenge is to generate these tokens from a textual prompt in a temporally coherent and semantically aligned manner. To this end, we propose the RQHC-Transformer, a specialized autoregressive model designed to synthesize motion token sequences across multiple quantization levels in a single unified framework.

Unlike prior transformer-based models that process motion tokens at a single resolution or require separate modules per layer [2], our RQHC-Transformer embraces the hierarchical structure of the quantized latent space. It sequentially predicts tokens from coarse to fine levels, enabling the model to first capture the overall motion structure and then refine it with increasing detail. This mirrors the residual nature of MoSA-VQ and preserves the interpretability of each quantization stage.

Input Construction. To generate the token sequence at quantization level l, we construct an input sequence $\mathbf{s}^l$ that conditions on both the textual prompt and the residual context from lower levels:

$$\mathbf{s}^l = [\mathbf{p} + \mathbf{q}_{\text{emb}}, \mathbf{t}_l^{1:n}], \tag{7}$$

where $\mathbf{p}$ is the CLIP-based text prompt embedding, and $\mathbf{q}_{\text{emb}}$ is the embedding representing the current quantization level. The sequence $\mathbf{t}_l^{1:n}$ is computed by aggregating token embeddings across all previous levels at each position:

$$\mathbf{t}_l^{1:n} = \left[\text{Embed}(t_l^1),\ \text{Embed}(t_l^2),\ \ldots,\ \text{Embed}(t_l^n)\right] \tag{8}$$

Here, n denotes the number of motion tokens (i.e., temporal steps) in the sequence. Each t_l^i is the i-th quantized token at layer l, and $\text{Embed}(\cdot)$ maps each token to its corresponding embedding vector.

Relative Positional Encodings. To effectively handle long motion sequences, we incorporate a relative positional encoding scheme into our causal attention layers. Compared to absolute positional encodings, this approach better preserves attention consistency for varying sequence lengths and enhances the model's ability to capture long-range dependencies.

Given an input token sequence, the attention score between token i and token j is computed as:

$$\mathbf{A}_{i,j}^{\text{rel}} = \mathbf{q}_i^\top \mathbf{k}_j + \mathbf{q}_i^\top \phi_{i-j} + u^\top \mathbf{k}_j + v^\top \phi_{i-j}, \tag{9}$$

where $\mathbf{q}_i$ and $\mathbf{k}_j$ are the query and key embeddings at position i and j, and ϕ_{i-j} is a learned relative position embedding, and u and v are learnable global bias vectors.

This formulation allows the model to incorporate relative distance information into the attention computation. The final output of the masked self-attention is:

$$\mathbf{a}_l^n = \text{Masked-Softmax}(\mathbf{A}_l^n)\mathbf{V}_l^n, \tag{10}$$

where $\mathbf{A}_l^n$ is the attention score matrix incorporating relative positional encoding at layer n of the l-th quantization level. $\mathbf{V}_l^n$ denotes the value matrix at the same layer.

Autoregressive Objective. We use a multi-layer conditional maximum likelihood objective:

$$\mathcal{L}_{\text{ce}} = -\frac{1}{B}\sum_{i=1}^{B}\sum_{j=1}^{T_i}\sum_{l=1}^{L}\log p_\theta(t_j^l \mid O_{i,j-1}^l, c). \tag{11}$$

Here, B is the batch size, T_i is the sequence length for the i-th sample, and L is the number of quantization levels. t_j^l denotes the ground truth discrete token at position j in layer l, while $O_{i,j-1}^l$ represents the model's predicted context (e.g., previously generated tokens or hidden states) up to position $j-1$ for sample i and layer l. The conditional input c encodes auxiliary information such as the text prompt, and $p_\theta(\cdot)$ denotes the model's predicted probability distribution parameterized by weights θ.

2.3 Text Condition Alignment for Improved Motion Decoding

Despite the RQHC-Transformer being trained on structured datasets such as HumanML3D, real-world user prompts at inference time often exhibit substantial variation in phrasing, granularity, and semantic clarity. This distribution shift between training and deployment introduces a critical challenge: the model may underperform on out-of-distribution inputs due to linguistic mismatch between test-time instructions and training-time prompts.

To mitigate this discrepancy, we propose TCA (text condition alignment), a lightweight inference-time module that reformulates raw user instructions into a training-compatible form without altering the model's weights. TCA operates by leveraging a pretrained language model $\mathcal{T}$, which serves as a prompt rewriting transformer to align free-form inputs with the motion model's learned linguistic prior.

We denote the space of unstructured user prompts as $\mathbb{U}$, which includes arbitrary, potentially noisy natural language descriptions. In contrast, $\mathbb{S}$ denotes the domain of structured, semantically aligned prompts used during training. Given a test-time input $\mathbf{c}_{\text{raw}} \in \mathbb{U}$, TCA produces an aligned instruction $\mathbf{c}_{\text{aligned}} \in \mathbb{S}$ by querying the rewriting model $\mathcal{T}$ under a style prior Ψ:

$$\mathbf{c}_{\text{aligned}} = \mathcal{T}(\mathbf{c}_{\text{raw}} \mid \Psi), \tag{12}$$

where Ψ encapsulates syntactic and semantic formatting patterns derived from the training corpus.

The aligned prompt $\mathbf{c}_{\text{aligned}}$ is then directly used as the conditioning input for the motion generation pipeline, replacing the original $\mathbf{c}_{\text{raw}}$. This inference-time alignment improves compatibility between the textual condition and the model's learned text-motion mapping, enabling more robust, coherent, and semantically accurate generation—particularly in zero- and few-shot settings. Since TCA operates independently of model parameters and requires no additional supervision, it offers a practical and efficient solution for open-domain deployment.

3 Experiments

We comprehensively evaluate our MOGO framework on text-to-motion generation tasks, focusing on motion quality, text-motion alignment, and generalization. Additional experimental results and ablation studies are provided in the appendix for further analysis.

3.1 Datasets and Evaluation Metrics

Datasets. We train and evaluate MOGO on HumanML3D [13] and KIT-ML [14]. HumanML3D contains 14,616 motion clips and 44,970 textual annotations, while KIT-ML includes 3,911 motion clips and 6,278 descriptions. Data splits follow T2M [13] with ratios of 0.8/0.15/0.05. To assess out-of-distribution generalization, we evaluate zero-shot performance on the CMP dataset [15], which includes 8,700 non-daily, combat-style motions. CMP provides three levels of text descriptions, and we use its test split. During CMP evaluation, prompt engineering is disabled to ensure fairness.

Evaluation Metrics. We adopt standard metrics from [13]. *FID* (Frechet Inception Distance) measures motion realism and serves as a primary evaluation metric. *R-Precision* evaluates text-motion alignment (Top-1, 2, 3). *MM-Dist* computes the distance between motion and text embeddings. *MultiModality* measures the variance of motions generated from the same text prompt.

3.2 Real-time Inference

To enhance zero- and few-shot generalization, we leverage GLM-4 [16] to rewrite user prompts that deviate from the HumanML3D style, aligning them more closely with the model's learned text-motion priors. Moreover, benefiting from MOGO's autoregressive design, motion tokens are generated in a streaming fashion—each token is predicted and emitted sequentially—enabling low-latency, frame-by-frame output without the need to wait for the entire sequence.

This architectural advantage allows MOGO to operate in real-time settings, achieving an inference speed of 30 frames per second on a single NVIDIA A100 GPU, making it well-suited for interactive applications such as virtual avatars, animation systems, and embodied agents.

3.3 Comparison to State-of-the-art Approaches

3.3.1 Reconstruction Quality of Motion Encoder

We compare the reconstruction fidelity of our MoSA-VQ with other state-of-the-art motion VAEs. As shown in Table 1, our encoder achieves the lowest FID on both HumanML3D and KIT-ML datasets.

3.3.2 Quantitative Results

As shown in Table 2, MOGO exhibits consistently strong performance across the HumanML3D, KIT-ML, and CMP (zero-shot) benchmarks. With the incorporation of our proposed TCA module, MOGO achieves leading results in several key metrics—particularly in terms of generation quality and diversity—while maintaining competitive performance across all others.

Table 1: Comparison of the reconstruction of our VAE Design vs. Motion VAEs from previous works.

HumanML3D		KIT-ML	
Method	**FID ↓**	**Method**	**FID ↓**
M2DM [17]	$0.063^{\pm 0.001}$	M2DM [17]	$0.413^{\pm 0.009}$
T2M-GPT [3]	$0.070^{\pm 0.001}$	T2M-GPT [3]	$0.472^{\pm 0.011}$
MoMask [2]	$0.019^{\pm 0.001}$	MoMask [2]	$0.112^{\pm 0.002}$
MMM [4]	$0.075^{\pm 0.001}$	MMM [4]	$0.641^{\pm 0.014}$
MOGO	$\mathbf{0.013}^{\pm 0.001}$	**MOGO**	$\mathbf{0.037}^{\pm 0.001}$

On in-distribution datasets like HumanML3D and KIT-ML, MOGO with TCA demonstrates top-tier performance across most metrics, though it exhibits a slight increase in FID compared to the base MOGO, the overall generation quality remains state-of-the-art, underscoring the robustness of our hierarchical architecture.

Table 2: Comparison with Motion Generation Models.

Datasets	Methods	R Precision↑			FID↓	MultiModal Dist↓	MultiModality↑
		Top 1	Top 2	Top 3			
HumanML3D	MotionDiffuse [18]	0.491±0.001	0.681±0.001	0.782±0.001	0.630±0.001	3.113±0.001	1.553±0.042
	T2M-GPT† [3]	0.491±0.003	0.680±0.002	0.775±0.002	0.116±0.004	3.118±0.011	1.856±0.011
	Fg-T2M [19]	0.492±0.002	0.683±0.003	0.783±0.003	0.243±0.019	3.109±0.007	1.614±0.049
	AttT2M† [5]	0.499±0.005	0.690±0.006	0.786±0.004	0.112±0.004	3.038±0.016	2.452±0.043
	MotionGPT† [20]	0.492±0.003	0.681±0.003	0.778±0.002	0.232±0.008	3.096±0.009	2.008±0.084
	MoMask [2]	0.521±0.002	0.713±0.002	0.807±0.002	0.045±0.002	2.958±0.008	1.241±0.040
	MMM [4]	0.504±0.002	0.696 ±0.003	0.794±0.004	0.080±0.004	2.998±0.007	1.226±0.035
	MOGO†	0.515±0.003	0.709±0.003	0.801±0.003	0.038±0.002	2.951±0.008	2.108±0.070
	MOGO with TCA†	0.527±0.007	0.722±0.008	0.827±0.012	0.064±0.003	2.849±0.003	2.344±0.037
KIT-ML	MotionDiffuse [18]	0.417±0.004	0.621±0.004	0.739±0.004	1.954±0.062	2.958±0.005	0.730±0.013
	T2M-GPT† [3]	0.416±0.006	0.627±0.006	0.745±0.006	0.514±0.029	3.007±0.023	1.570±0.039
	Fg-T2M [19]	0.418±0.005	0.626±0.004	0.745±0.004	0.571±0.047	3.114±0.015	1.019±0.029
	AttT2M† [5]	0.413±0.006	0.632±0.006	0.751±0.006	0.870±0.039	3.039±0.016	2.281±0.043
	MotionGPT† [20]	0.366±0.005	0.558±0.004	0.680±0.005	0.510±0.004	3.527±0.021	2.328±0.117
	MoMask [2]	0.433±0.007	0.656±0.005	0.781±0.005	0.204±0.011	2.779±0.022	1.131±0.043
	MMM [4]	0.381±0.005	0.590±0.006	0.718±0.005	0.429±0.019	3.146±0.019	1.105±0.026
	MOGO†	0.420±0.007	0.634±0.007	0.754±0.007	0.191±0.016	2.957±0.029	2.063±0.066
	MOGO with TCA†	0.447±0.023	0.668±0.016	0.801±0.007	0.313±0.009	2.849±0.007	2.273±0.073
CMP (zero-shot)	T2M-GPT† [3]	0.061±0.003	0.103±0.005	0.147±0.006	16.092±0.099	4.179±0.049	2.118±0.033
	AttT2M† [5]	0.065±0.004	0.109±0.008	0.147±0.008	18.403±0.071	4.048±0.017	2.208±0.019
	MotionGPT† [20]	0.050±0.002	0.094±0.002	0.133±0.003	15.654±0.183	4.431±0.021	5.535±0.259
	MoMask [2]	0.062±0.003	0.108±0.005	0.150±0.004	24.351±0.205	4.817±0.022	1.651±0.050
	MMM [4]	0.067±0.004	0.116±0.008	0.154±0.008	17.087±0.313	4.360±0.017	2.802±0.011
	MOGO†	0.071±0.003	0.124±0.004	0.183±0.004	10.388±0.171	3.847±0.029	4.562±0.066
	MOGO with TCA†	0.122±0.006	0.227±0.011	0.304±0.004	6.873±0.073	3.040±0.014	4.299±0.013

Red = Best, Blue = Second Best, orange = Third Best.

This advantage becomes more pronounced in the CMP zero-shot setting, where other transformer-based approaches often suffer from significant degradation. In contrast, MOGO with TCA achieves the best results in multiple metrics, highlighting its ability to generalize effectively to out-of-distribution prompts. These improvements stem from the synergy between our hierarchical generation framework and the inference-time input alignment mechanism introduced by TCA.

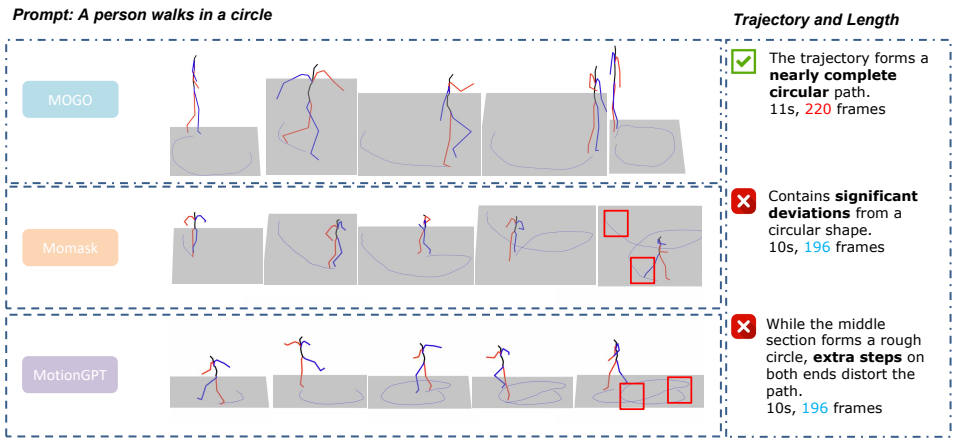
Furthermore, MOGO supports streaming token-wise generation, allowing for real-time inference without sacrificing output quality. This combination of strong generalization, high overall performance, and low-latency generation makes MOGO well-suited for practical deployment in open-ended, real-world motion generation scenarios.

3.3.3 Qualitative Comparison and Long-range Generation

Qualitative Comparison. As shown in Figure 3, our generated motions are smoother, more consistent and better aligned with input prompts compared to other methods (more results are shown in Appendix A, and videos are also provided in the supplementary material). Built upon a hierarchical architecture, MOGO enables longer motion generation without sacrificing coherence or realism, demonstrating competitive performance among transformer-based autoregressive models and strong generalization capabilities (more ablation studies are shown in Appendix).

4 Related Work

Human Motion Generation. Recent advances in human motion generation have enabled conditioning on modalities like text, audio, music, and images [1]. Early deterministic models [21, 22] often produced over-smoothed, unrealistic motions. To address this, stochastic approaches, such as GANs [23, 24] and VAE-based models [25, 26], improved motion diversity. Text-to-motion generation gained prominence with works like [13], which used temporal VAEs to model text-motion distributions. Recently, diffusion-based methods [27, 18, 28, 17, 8, 6] and transformer-based approaches [2, 3, 20] have led the field.



(a) Comparison on execution accuracy of long actions

(b) Comparison on execution completion of long actions

Figure 3: Qualitative comparison among our model, Momask [2], and MotionGPT [20].

LLM-Based Motion Generation Models. LLM-based architectures have become a cornerstone of text-to-motion generation, leveraging their ability to model sequential data and adapt to varied tasks [2–5, 20]. Models like MoMask [2] and MMM [4] use masked token modeling to produce high-quality motions but struggle with streaming output and generalization in low-data or out-of-distribution settings due to their bidirectional design. Conversely, autoregressive models like T2M-GPT [3] and AttT2M [5] enable sequential generation, making them suitable for real-time applications and scalable with larger datasets, though they often sacrifice some motion quality. Efforts like MotionGPT [20] integrate multimodal language modeling but face challenges in achieving high-fidelity motion outputs. Our MOGO framework addresses these issues by combining efficient encoding through the MoSA-VQ with a single-pass autoregressive transformer, ensuring both high-quality motion and streaming capabilities.

Hierarchical Transformers. Hierarchical transformer architectures excel in domains like NLP [12, 29], image generation [30], and vision tasks [31, 32]. By processing data at multiple abstraction levels, these models enhance representation capacity and scalability. For instance, Swin Transformers [31] support scalable high-resolution vision, while CogView2 [30] enables high-fidelity image synthesis. In motion generation, hierarchical transformers are underexplored, especially for autoregressive frameworks with residual quantization. Our RQHC-Transformer in MOGO leverages hierarchical modeling to efficiently process multi-layer motion tokens, improving quality and generalization.

Motion Tokenization. Discretizing continuous motion data into tokens v‘ia vector quantization is central to transformer-based motion generation. TM2T [33] introduced VQ-VAE to map motions to discrete sequences. T2M-GPT [3] enhanced token quality with exponential moving average and codebook reset techniques. AttT2M [5] improved quantization through body-part-aware encoding. MoMask [2] advanced this with residual vector quantization (RVQ), producing multi-level tokens for better reconstruction quality. Compared to prior work, our approach introduces learnable feature scaling into the residual vector quantization process, allowing each quantization level to adaptively adjust the magnitude of its residuals. This scaled RVQ mechanism stabilizes the residual distribution across quantization stages, mitigates the risk of later stages collapsing to noise, and ensures more effective codebook utilization. As a result, the model benefits from improved reconstruction fidelity, better token expressiveness, and enhanced training stability.

5 Limitations and Discussion

Despite MOGO's strong performance in real-time streaming and zero-shot generalization, several limitations remain.

Motion Editing. Due to its unidirectional autoregressive design, MOGO struggles with temporal editing tasks such as infilling or interpolation. This limits its use in scenarios like animation post-processing, where flexible motion manipulation is needed.

Generation Length. While relative positional encoding allows MOGO to generate up to 260 frames in cyclic motions, non-continuous sequences remain limited to around 196 frames. This is primarily due to weak temporal correlations in motion data, which hinder the use of segment-level memory mechanisms like Transformer-XL [34].

To address these issues, future work should focus on (1) enabling conditional adjustments after initial generation to support dynamic prompts, and (2) improving local condition alignment for seamless mid-sequence control. These enhancements could make MOGO more adaptable for interactive and editable motion generation.

6 Conclusion

We presented MOGO, a one-pass autoregressive framework for high-quality and real-time 3D human motion generation. By combining the MoSA-VQ module for scale-adaptive residual vector quantization with the RQHC-Transformer for hierarchical causal decoding, MOGO achieves efficient and temporally coherent motion synthesis. The integration of TCA mechanism further enhances semantic alignment and enables better generalization in zero- and few-shot scenarios. Our framework supports low-latency, frame-by-frame inference, making it well-suited for real-time and streamable applications. Extensive experiments demonstrate that MOGO outperforms prior transformer-based methods across multiple benchmarks in terms of generation quality and text-motion alignment. While current limitations remain in motion editing and handling complex non-cyclic sequences, MOGO provides a robust foundation for future research toward controllable, editable, and interactive motion generation.

References

[1] W. Zhu, X. Ma, D. Ro, H. Ci, J. Zhang, J. Shi, F. Gao, Q. Tian, and Y. Wang, "Human motion generation: A survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 46, no. 4, pp. 2430–2449, 2023.

[2] 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.

[3] J. Zhang, Y. Zhang, X. Cun, Y. Zhang, H. Zhao, H. Lu, X. Shen, and Y. Shan, "Generating human motion from textual descriptions with discrete representations," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2023, pp. 14 730–14 740.

[4] E. Pinyoanuntapong, P. Wang, M. Lee, and C. Chen, "Mmm: Generative masked motion model," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 1546–1555.

[5] C. Zhong, L. Hu, Z. Zhang, and S. Xia, "Attt2m: Text-driven human motion generation with multi-perspective attention mechanism," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023, pp. 509–519.

[6] 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.

[7] 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 (CVPR)*. New Orleans, LA, USA: IEEE, June 2022, pp. 10 684–10 695, arXiv:2112.10752.

[8] X. Chen, B. Jiang, W. Liu, Z. Huang, B. Fu, T. Chen, and G. Yu, "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.

[9] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell *et al.*, "Language models are few-shot learners," *Advances in neural information processing systems*, vol. 33, pp. 1877–1901, 2020.

[10] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu, "Exploring the limits of transfer learning with a unified text-to-text transformer," *Journal of machine learning research*, vol. 21, no. 140, pp. 1–67, 2020.

[11] A. van den Oord, O. Vinyals, and K. Kavukcuoglu, "Neural discrete representation learning," in *Proceedings of the 31st International Conference on Neural Information Processing Systems*. Long Beach, CA, USA: Curran Associates Inc., December 2017, pp. 6306–6315, arXiv:1711.00937.

[12] P. Nawrot, S. Tworkowski, M. Tyrolski, Ł. Kaiser, Y. Wu, C. Szegedy, and H. Michalewski, "Hierarchical transformers are more efficient language models," in *Findings of the Association for Computational Linguistics: NAACL 2022*, 2022, pp. 1559–1571.

[13] C. Guo, S. Zou, X. Zuo, S. Wang, W. Ji, X. Li, and L. Cheng, "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.

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

[15] Y. Liao, Y. Fu, Z. Cheng, and J. Wang, "Animationgpt: An aigc tool for generating game combat motion assets," https://github.com/fyyakaxyy/AnimationGPT, 2024.

[16] T. GLM, A. Zeng, B. Xu, B. Wang, C. Zhang, D. Yin, D. Zhang, D. Rojas, G. Feng, H. Zhao *et al.*, "Chatglm: A family of large language models from glm-130b to glm-4 all tools," *arXiv preprint arXiv:2406.12793*, 2024.

[17] 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.

[18] M. Zhang, Z. Cai, L. Pan, F. Hong, X. Guo, L. Yang, and Z. Liu, "Motiondiffuse: Text-driven human motion generation with diffusion model," *IEEE transactions on pattern analysis and machine intelligence*, vol. 46, no. 6, pp. 4115–4128, 2024.

[19] Y. Wang, Z. Leng, F. W. Li, S.-C. Wu, and X. Liang, "Fg-t2m: Fine-grained text-driven human motion generation via diffusion model," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023, pp. 22 035–22 044.

[20] B. Jiang, X. Chen, W. Liu, J. Yu, G. Yu, and T. Chen, "Motiongpt: Human motion as a foreign language," *Advances in Neural Information Processing Systems*, vol. 36, pp. 20 067–20 079, 2023.

[21] 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.

[22] 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.

[23] 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.

[24] Z. Wang, P. Yu, Y. Zhao, R. Zhang, Y. Zhou, J. Yuan, and C. Chen, "Learning diverse stochastic human-action generators by learning smooth latent transitions," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 34, no. 07, 2020, pp. 12 281–12 288.

[25] C. Guo, X. Zuo, S. Wang, X. Liu, S. Zou, M. Gong, and L. Cheng, "Action2video: Generating videos of human 3d actions," *International Journal of Computer Vision*, vol. 130, no. 2, pp. 285–315, 2022.

[26] 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.

[27] 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, no. 7, 2023, pp. 8255–8263.

[28] M. Zhang, X. Guo, L. Pan, Z. Cai, F. Hong, H. Li, L. Yang, and Z. Liu, "Remodiffuse: Retrieval-augmented motion diffusion model," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023, pp. 364–373.

[29] R. Pappagari, P. Żelasko, J. Villalba, Y. Carmiel, and N. Dehak, "Hierarchical transformers for long document classification," in *2019 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*. Singapore: IEEE, December 2019, pp. 838–844, arXiv:1910.10781.

[30] M. Ding, W. Zheng, W. Hong, and J. Tang, "Cogview2: Faster and better text-to-image generation via hierarchical transformers," *Advances in Neural Information Processing Systems*, vol. 35, pp. 16 890–16 902, 2022.

[31] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, and B. Guo, "Swin transformer: Hierarchical vision transformer using shifted windows," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2021, pp. 10 012–10 022.

[32] R. J. Chen, C. Chen, Y. Li, T. Y. Chen, A. D. Trister, R. G. Krishnan, and F. Mahmood, "Scaling vision transformers to gigapixel images via hierarchical self-supervised learning," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2022, pp. 16 144–16 155.

[33] 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.

[34] Z. Dai, Z. Yang, Y. Yang, J. Carbonell, Q. Le, and R. Salakhutdinov, "Transformer-xl: Attentive language models beyond a fixed-length context," in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Florence, Italy: Association for Computational Linguistics, July 2019, pp. 2978–2988.

A Generation Results

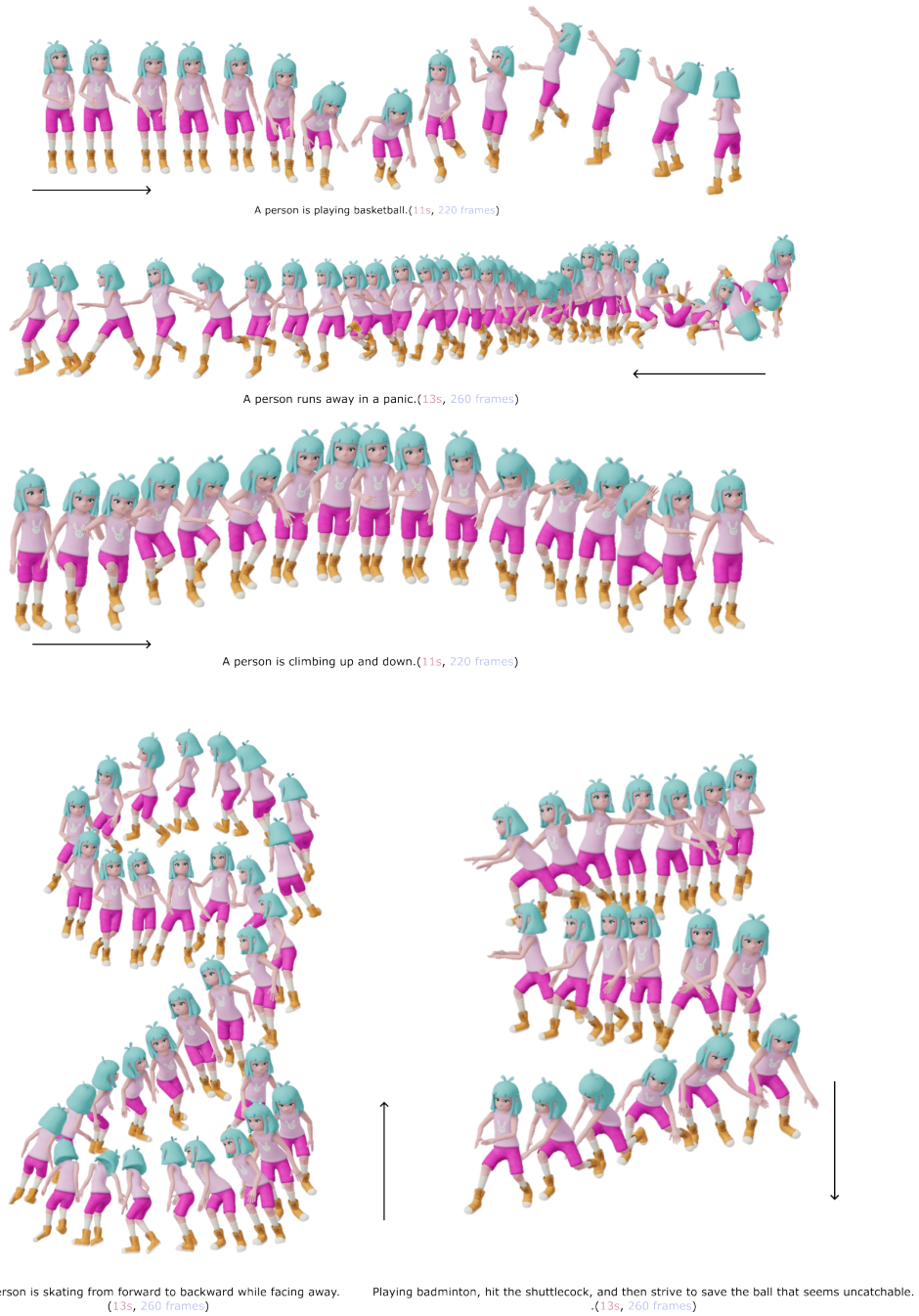


Figure 4: Demonstration of generative capabilities for open vocabulary and ultra-long sequences.

Stylized Motion Generation. Figure 4 illustrates the results of our model applied to stylized motion generation. Three representative action types are shown, each demonstrating the model's ability to preserve motion semantics while adapting to distinct style patterns. To further showcase the generality and diversity of our approach, we provide additional stylized motion sequences in MP4 format in the supplementary material.

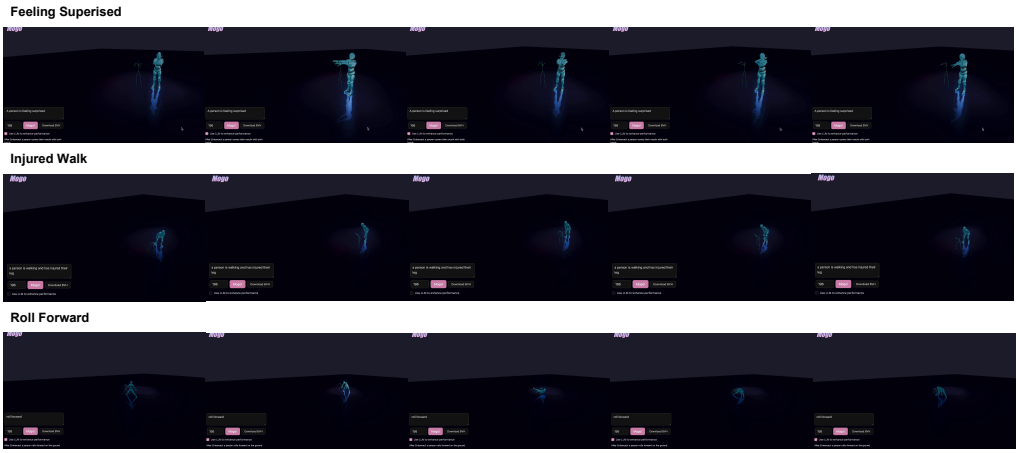


Figure 5: Generated motions rendered on skinned 3D characters.

Qualitative Comparison. Figures 6–9 present a qualitative comparison of motion generation results given the same text prompt, showcasing outputs among our model, Momask [2] and MotionGPT [20]. As observed, MOGO produces more complete and coherent motion sequences, particularly excelling in motion completeness and fidelity. For a more intuitive evaluation, we additionally provide corresponding video results in the supplementary material (MP4 format), which further highlight the differences in motion quality and realism.

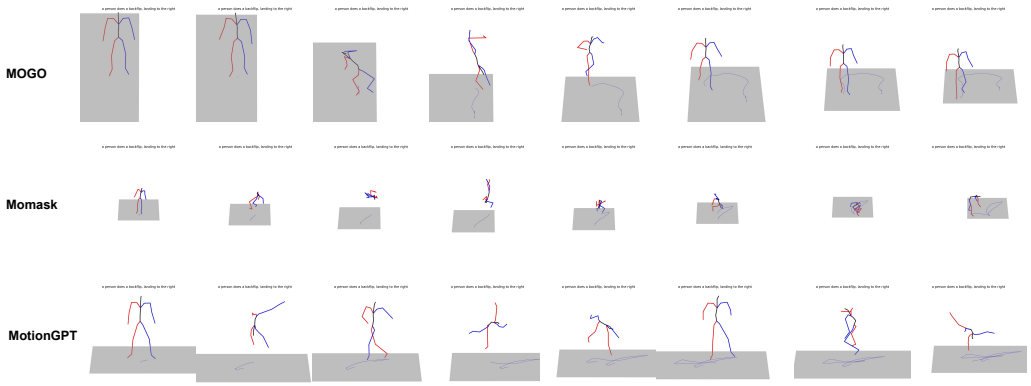


Figure 6: A person does a backflip, landing to the right.

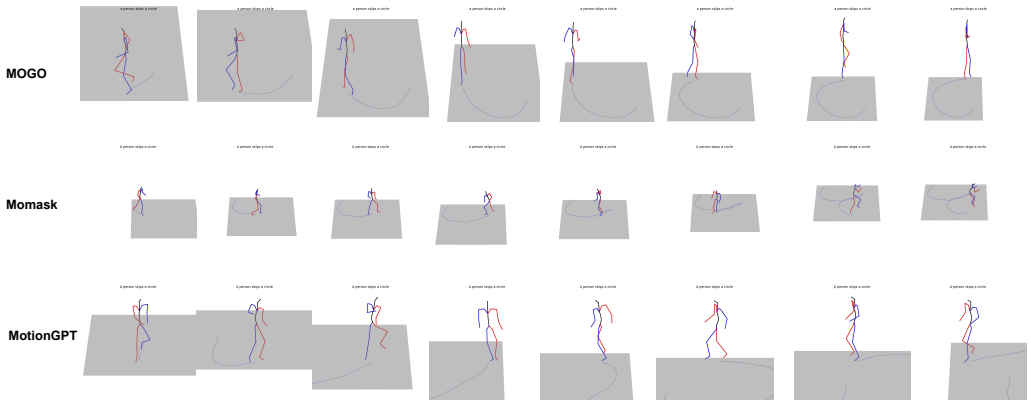


Figure 7: A person skips a circle.

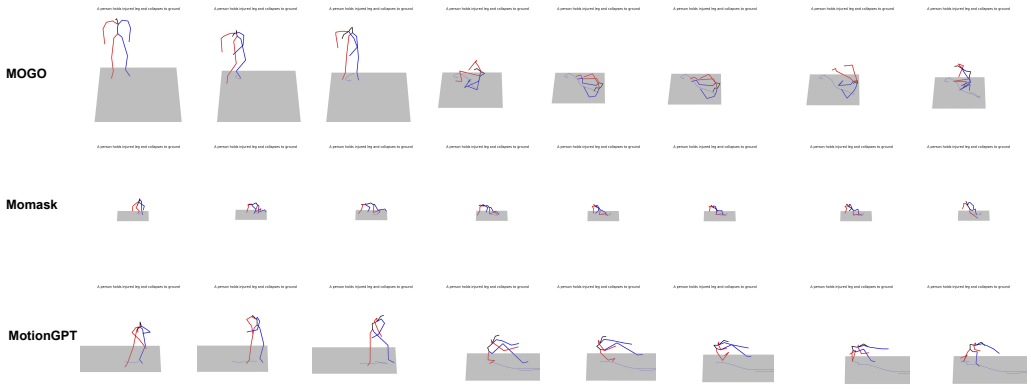


Figure 8: A person holds injured leg and collapses to ground

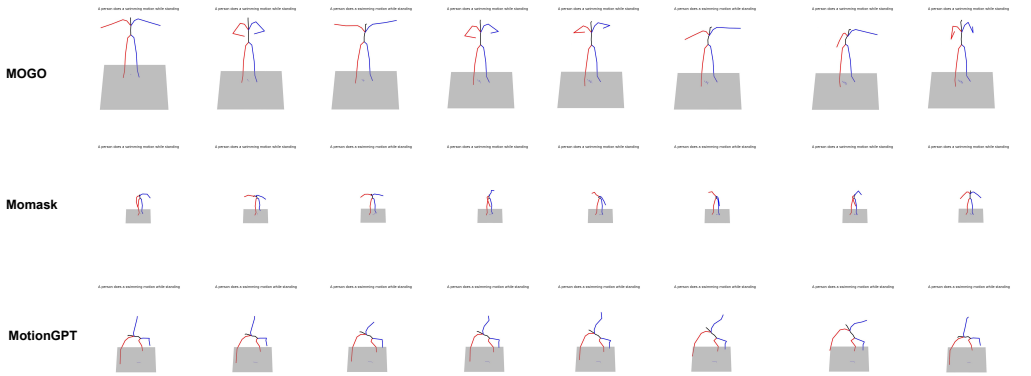


Figure 9: A person does a swimming motion while standing.

B Ablation Study

Impact of Learnable Scaling in MoSA-VQ. We investigate the role of the proposed learnable scaling mechanism in MoSA-VQ by comparing models with and without this design. In the baseline configuration, residual features at each quantization level are directly quantized without any adaptive normalization. In contrast, our full model applies a layer-specific affine transformation before quantization, allowing each level to adjust the residual space dynamically.

As shown in Table 3, incorporating learnable scale and bias parameters leads to consistent improvements across multiple evaluation metrics. The model demonstrates better reconstruction quality, enhanced alignment between motion and text, and stronger semantic consistency. These results highlight the benefit of introducing adaptive normalization in residual quantization, which facilitates more expressive and stable token representations.

Table 3: Ablation study on the use of learnable scaling

Codebook size	**w/ Learnable Scaling**		
	FID↓	Top1↑	MM-Dist↓
2048 × 512	0.052±0.005	0.498±0.002	2.997±0.008
4096 × 256	0.047±0.005	0.501±0.002	2.901±0.008
8192 × 128	**0.038**±0.003	**0.515**±0.007	2.951±0.003
Codebook size	**w/o Learnable Scaling**		
	FID↓	Top1↑	MM-Dist↓
2048 × 512	0.094±0.011	0.484±0.002	3.273±0.009
4096 × 256	0.091±0.004	0.493±0.004	3.451±0.017
8192 × 128	**0.079**±0.002	**0.502**±0.002	3.002±0.008

Effect of Quantization Depth in MoSA-VQ. We further explore how the number of residual quantization layers in MoSA-VQ affects the model's overall performance. As residual quantization hierarchically captures motion information from coarse to fine granularity, increasing the depth allows more detailed refinement of the motion representation.

Table 4: Study on the number of Layers in MoSA-VQ on HumanML3D [13] test set. **Bold** indicates the best FID result.

Layers	**Reconstruction**			**Generation**		
	FID↓	**Top1↑**	**MM-Dist↓**	**FID↓**	**Top1↑**	**MM-Dist↓**
1	0.067±0.001	0.508±0.002	2.997±0.007	0.068±0.009	0.469±0.002	3.138±0.006
2	0.046±0.001	0.504±0.001	2.984±0.010	0.053±0.008	0.467±0.003	3.114±0.002
3	0.035±0.001	0.508±0.003	2.980±0.004	0.044±0.005	0.498±0.002	2.997±0.008
4	0.023±0.001	0.505±0.003	2.989±0.006	0.045±0.005	0.501±0.002	2.982±0.002
5	0.017±0.001	0.511±0.001	2.980±0.003	0.041±0.005	0.498±0.002	2.977±0.002
6	0.013±0.001	0.510±0.001	2.982±0.006	**0.038**±0.003	0.515±0.002	2.951±0.003
7	**0.013**±0.001	0.503±0.002	2.993±0.007	0.040±0.003	0.513±0.007	2.943±0.003

As shown in Table 4, models with a shallow quantization depth tend to underperform, suggesting that limited hierarchy constrains the representational capacity of the latent code. As the number of layers increases, performance improves consistently across reconstruction fidelity, text-motion alignment, and multimodal distance. However, the gains tend to plateau beyond a certain depth, indicating diminishing returns. This analysis supports the use of a moderate number of residual layers as a trade-off between expressiveness and efficiency.

Codebook Size. We evaluate various codebook sizes on HumanML3D. As shown in Table 5, 8192×128 achieves the best FID for generation and competitive reconstruction.

Dataset Size. We evaluate MOGO's scalability by mixing HumanML3D with varying proportions of CMP data (Table 6). Without retraining the MoSA-VQ, generation quality improves significantly with increased data volume.

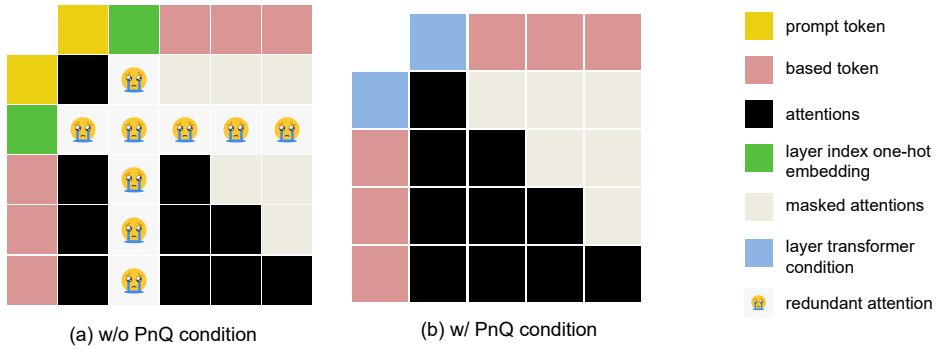


Figure 10: Comparison of attention patterns with and without the PnQ (Prompt-and-Quantization) condition. (a) Without PnQ, an extra layer token between the prompt and motion sequence causes redundant attention and weakens the alignment between text and motion. (b) With PnQ, the prompt token is directly fused with quantization conditions, improving attention efficiency and semantic alignment.

Table 5: Study on the number of codes in codebook on HumanML3D [13] test set. **Bold** indicates the best FID result.

Codebook Size	Reconstruction			Generation		
	FID↓	Top1↑	MM-Dist↓	FID↓	Top1↑	MM-Dist↓
512 × 512	0.022±0.001	0.508±0.003	2.997±0.007	0.203±0.009	0.469±0.002	3.138±0.006
1024 × 1024	0.015±0.001	0.511±0.002	2.984±0.010	0.114±0.008	0.467±0.003	3.114±0.009
2048 × 512	0.017±0.001	0.511±0.003	2.980±0.007	0.052±0.005	0.498±0.002	2.997±0.008
4096 × 256	0.019±0.001	0.510±0.002	2.989±0.006	0.047±0.005	0.501±0.002	2.901±0.008
8192 × 128	**0.013**±0.001	0.510±0.002	2.989±0.007	**0.038**±0.003	0.515±0.007	2.951±0.003

Input Condition. We compared the performance of input conditions by either adding the prompt token and layer token as a sequence prefix or not. As shown in Table 7, our experiments reveal that when without PnQ (adding prompt token to quantization layer token) is applied, the correlation between generated motion sequences and text (measured by R-Precision and MM Distance) decreases significantly. This may result from an additional layer token between action sequence tokens and prompt tokens during training, leading to some attention loss, as shown in Figure 10.

Table 6: The impact of varying dataset sizes on FID performance. Lower FID indicates better generation quality.

Epochs	w/ 0% CMP	w/ 50% CMP	w/ 100% CMP
50	0.671	0.601	0.501
80	0.437	0.379	0.284
100	0.347	0.263	**0.205**

Table 7: Ablation study on the use of the PnQ condition.

Codebook size	w/ PnQ		
	FID↓	Top1↑	MM-Dist↓
2048 × 512	0.052±0.005	0.498±0.002	2.997±0.008
4096 × 256	0.047±0.005	0.501±0.002	2.901±0.008
8192 × 128	**0.038**±0.003	**0.515**±0.007	2.951±0.003
Codebook size	**w/o PnQ**		
	FID↓	Top1↑	MM-Dist↓
2048 × 512	0.096±0.005	0.452±0.002	3.293±0.009
4096 × 256	0.091±0.005	0.459±0.002	3.141±0.007
8192 × 128	**0.085**±0.002	**0.482**±0.003	3.127±0.008

C Algorithm

The core training code structure of our MoSA-VQ model is illustrated below. The variable abbreviations used in the code are defined as follows:

Algorithm 1 MoSA-VQ: Hierarchical Residual Vector Quantization with Feature Scaling

Require: Motion encoder output $\tilde{\mathbf{b}} \in \mathbb{R}^{n \times d}$, levels L, learnable scales $\{\mathbf{s}^l\}_{l=0}^{L}$, biases $\{\mathbf{b}^l\}_{l=0}^{L}$, codebooks $\{\mathcal{Q}^l\}_{l=0}^{L}$

Ensure: Quantized latent representation $\hat{\mathbf{b}}$

Initialize residual $\mathbf{r}^0 \leftarrow \tilde{\mathbf{b}}$
for $l = 0$ to L **do**
  $\mathbf{r}^l_{\text{scaled}} \leftarrow \|\mathbf{s}^l\| \cdot \mathbf{r}^l + \mathbf{b}^l$ ▷ Feature scaling
  $\mathbf{b}^l_{\text{scaled}} \leftarrow \mathcal{Q}^l(\mathbf{r}^l_{\text{scaled}})$ ▷ Vector quantization
  $\mathbf{b}^l \leftarrow \frac{\mathbf{b}^l_{\text{scaled}} - \mathbf{b}^l}{\|\mathbf{s}^l\|}$ ▷ Inverse transform
  $\mathbf{r}^{l+1} \leftarrow \mathbf{r}^l - \mathbf{b}^l$ ▷ Update residual
end for
$\hat{\mathbf{b}} \leftarrow \sum_{l=0}^{L} \mathbf{b}^l$ ▷ Reconstruct quantized latent

The core training code structure of our RQHC-Tranformer is illustrated below. The variable abbreviations used in the code are defined as follows:

Algorithm 2 Motion Generation Algorithm

Input: p_texts, m_ids, m_len, lbls
bs, nt, r ← shape(m_ids)
p_logits ← EncodeText(p_texts)
p_logits ← CondEmb(p_logits)
m_ids ← mask_motion_token(m_ids[:, :-1, :])
tks ← TokEmb(m_ids)
for i = 0 to n_q - 1 **do**
  r_tks ← Reduce(tks, i)
  tks_st[i] ← r_tks
end for
all_layer_out ← []
for i = 0 to n_t - 1 **do**
  q_ids ← Fill(bs, i)
  q_oh ← EncodeQuant(q_ids)
  s_tks ← QuantEmb(q_oh)
  st_tks ← Concatenate(p_logits + s_tks, tks_st[i])
  ret, att ← Transformer(st_tks, lbls)
  layer_out ← Head(att)
  all_layer_out.append(layer_out)
end for
Output: out ← Stack(all_layer_out)
ce_loss, pred_id, acc ← CalcPerf(out, lbls, m_len)
return ce_loss, acc, pred_id, out