

BETTER CONDITIONING, BETTER MOTION: GENERATION-ORIENTED TEXT REPRESENTATION LEARNING FOR MOTION GENERATION

Anonymous authors

Paper under double-blind review

ABSTRACT

Text-to-motion generation aims to synthesize realistic human motion sequences from natural language descriptions. Existing methods typically use a text encoder separately pretrained on proxy objectives. However, such encoders struggle to capture subtle dynamic nuances, leading to inadequate text representations for motion generation. We argue that effective text representations for motion generation should be generation-oriented, i.e., optimized to better support the synthesis of high-fidelity, text-aligned motion sequences. Accordingly, we propose LeGO, a co-optimization framework that Learns Generation-Oriented text representations for motion generation and derives LeGO-CLIP, a motion-specialized text encoder. Specifically, LeGO adapts a pretrained CLIP text encoder into LeGO-CLIP by jointly optimizing it with a motion diffusion model, allowing the encoder to receive differentiable feedback from generation quality through two semantic consistency constraints: a motion consistency constraint and a text consistency constraint. Comprehensive experiments demonstrate the effectiveness of our method from three perspectives: 1) Performance: It achieves state-of-the-art results on HumanML3D, KIT-ML and SnapMoGen benchmarks; 2) Efficiency: It significantly accelerates convergence, requiring up to 8x fewer training iterations; and 3) Transferability: LeGO-CLIP serves as a plug-and-play replacement and consistently improves the performance of diverse motion generation models. The source code will be made publicly available.

1 INTRODUCTION

Text-to-motion generation (T2M) (Tevet et al., 2023; Zhang et al., 2024a; 2023a; Chen et al., 2023) aims to synthesize realistic human motion sequences from natural language descriptions, with wide-ranging applications in animation (Hong et al., 2022), gaming (Holden et al., 2017), and robotics (Nishimura et al., 2020). Despite its vast potential, the task remains challenging because it depends not only on the generator itself, but also on whether the conditioning text representation can faithfully encode subtle motion-relevant semantics such as speed, direction, and action intensity.

The majority of existing T2M methods (Tevet et al., 2023; Zhang et al., 2024b; Huang et al., 2024a; Liang et al., 2024; Chen et al., 2023; Jin et al., 2024; Meng et al., 2025; Zhang et al., 2023a; Pinyoanuntapong et al., 2023; Guo et al., 2024; Zou et al., 2024; Wan et al., 2024; Huang et al., 2024b; Gupta et al., 2026) adopt a general text encoder such as CLIP (Radford et al., 2021) to extract text representations. However, a fundamental misalignment exists: CLIP is trained on static image-text pairs, which conflicts with the temporal and dynamic requirements of motion generation. To address this limitation, several recent studies (Maldonado et al., 2025; Li et al., 2025) propose pretraining specialized text encoders on text-motion pairs, primarily employing contrastive learning to align text and motion embeddings.

While these specialized text encoders have significantly improved text-motion matching (Maldonado et al., 2025; Li et al., 2025), they are still optimized as standalone representation learners under proxy objectives such as retrieval-oriented contrastive learning. This creates an objective mismatch: the encoder is optimized for cross-modal alignment, but not for producing representations that directly facilitate high-fidelity motion synthesis. As a result, better retrieval alignment does not necessarily

translate into better generation-oriented conditioning. Fundamentally, text representations for motion generation must be capable of interpreting complex instructions while remaining highly sensitive to subtle dynamic nuances. Yet, encoders trained on separate proxy objectives lack this critical sensitivity. As illustrated in Fig. 1, both the general CLIP (Radford et al., 2021) and the specialized LAMP (Li et al., 2025) encoders exhibit high cosine similarities between text pairs differing by only one dynamic keyword (e.g., “slowly” vs “quickly”), suggesting that proxy-trained encoders may preserve coarse semantic alignment while failing to encode the fine-grained distinctions that significantly affect downstream motion generation.

We therefore argue that the central bottleneck in T2M is not merely the lack of a motion-specialized encoder, but the lack of a generation-oriented learning objective for text representations in T2M. Specifically, an optimal text representation should be learned according to its usefulness for downstream motion synthesis, i.e., whether it enables the generator to produce motions that are both realistic and faithful to fine-grained dynamic instructions.

To realize this generation-oriented view, we propose LeGO, a co-optimization framework that **L**earns **G**eneration-Oriented text representations directly from motion generation quality. LeGO adapts a pretrained CLIP text encoder into LeGO-CLIP by allowing the encoder to receive differentiable feedback from the downstream generator, rather than training it separately with a proxy objective. We instantiate this adaptation in a parameter-efficient manner using LoRA (Hu et al., 2022), which preserves CLIP’s pretrained priors while enabling stable specialization for motion generation. Specifically, LeGO quantifies this generation quality by introducing two semantic consistency constraints on generated motions: a motion consistency constraint that measures whether the predicted motion preserves the semantics of the ground-truth motion, and a text consistency constraint that measures whether the predicted motion remains aligned with the conditioning text. These constraints serve not merely as regularizers, but as differentiable supervision that backpropagates through the generator to refine the conditioning text representation. As a result, LeGO-CLIP learns generation-oriented representations that directly support high-fidelity motion generation while retaining CLIP’s pretrained linguistic priors. Notably, because LeGO is a training-time strategy, replacing the vanilla CLIP encoder with LeGO-CLIP incurs no additional computational overhead during inference.

An important consequence of generation-oriented representation learning is that the learned encoder is not tied to a single generator architecture. If LeGO-CLIP indeed captures text semantics that are more effective for motion generation, it should transfer as a plug-and-play replacement across different motion generators. We verify this by integrating LeGO-CLIP into VQ-VAE-based (e.g., MoMask (Guo et al., 2024)) and latent-diffusion-based (e.g., SALAD (Hong et al., 2025)) generators, where it consistently improves performance without changing the remaining training setup.

Moreover, stronger conditioning representations should reduce the burden on the generator to infer motion-relevant semantics from weak text features. Consistent with this hypothesis, replacing vanilla CLIP with LeGO-CLIP leads to markedly faster convergence and stronger data efficiency across baselines. As illustrated in Fig. 2, replacing the vanilla CLIP encoder with LeGO-CLIP yields superior performance and faster convergence in terms of training iterations across baselines. With LeGO-CLIP, MDM and MoMask reach comparable performance with 4x and 8x fewer training iterations, respectively. These results underscore the effectiveness of LeGO-CLIP as a general-purpose solution for advancing high-fidelity

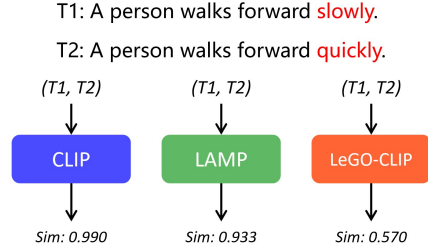


Figure 1: Only one keyword differs in paired texts. Cosine similarities of the embeddings from CLIP (Radford et al., 2021), LAMP (Li et al., 2025), and LeGO-CLIP are calculated.

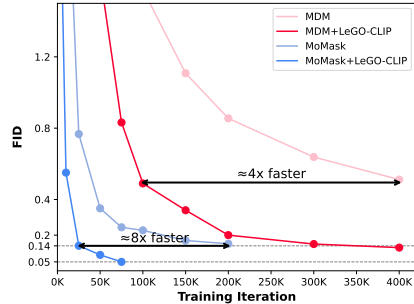


Figure 2: Our method remarkably improves the performance of two representative T2M models (MDM (Tevet et al., 2023) and MoMask (Guo et al., 2024)). Beyond performance gains, our method exhibits superior convergence efficiency, achieving results comparable to baselines with 4x to 8x fewer iterations. MoMask metric is evaluated using only M-Transformer (Guo et al., 2024).

text-to-motion generation and support our central assertion that textual representations play a critical role in T2M.

Our main contributions can be summarized as follows:

- We identify an objective mismatch in text-to-motion generation: text encoders optimized with proxy objectives do not necessarily learn representations suitable for downstream motion synthesis.
- We propose LeGO, a generation-oriented co-optimization framework that adapts the text encoder using differentiable feedback from generated motion quality, thereby learning motion-specialized text representations beyond standalone proxy training.
- Extensive quantitative and qualitative experiments on HumanML3D, KIT-ML, and SnapMoGen demonstrate the effectiveness of LeGO and LeGO-CLIP.

2 RELATED WORKS

Recent research in text-conditioned motion generation has primarily focused on two directions: (1) improving generative paradigms and (2) enhancing the modeling of human motion dynamics.

Improving Generative Paradigms. According to the generative paradigm, mainstream text-to-motion generation methods can be primarily categorized into vector-quantized variational autoencoders (VQ-VAE)-based methods (Van Den Oord et al., 2017) and diffusion-based methods (Ho et al., 2020). T2M-GPT (Zhang et al., 2023a) introduces a GPT-style autoregressive framework to predict motion sequences autoregressively. Inspired by MaskGIT (Chang et al., 2022), MMM (Pinyoanuntapong et al., 2023) employs a masked modeling approach to synthesize motion in parallel. Building on the vanilla VQ, MoMask (Guo et al., 2024) integrates residual vector quantization with multi-level codebooks, significantly reducing quantization errors in motion VQ-VAE. Concurrently, diffusion-based models have profoundly reshaped the landscape. MDM (Tevet et al., 2023) and MotionDiffuse (Zhang et al., 2024a) pioneered the use of diffusion processes for text-to-motion tasks by denoising representations iteratively. MLD (Chen et al., 2023) projects the diffusion process into a latent space to achieve superior generation fidelity and efficiency. Furthermore, several studies (Li et al., 2024; Zhou et al., 2024) investigate acceleration techniques that expedite motion diffusion. Despite these extensive architectural innovations, a critical oversight persists across all aforementioned paradigms. They typically adopt the vanilla CLIP (Radford et al., 2021) text encoder to extract kinematic textual representations, operating under the implicit, unverified assumption that CLIP’s static, object-centric alignments are suitable for driving dynamic motion generation.

Enhancing Modeling of Human Motion Dynamics. The second predominant research direction concentrates on formulating more accurate representations of human motion dynamics. Some methods (Zou et al., 2024; Zhong et al., 2023) focus on modeling spatiotemporal cues using a body-part-aware network. Furthermore, MoGenTS (Yuan et al., 2024) explicitly preserves the joint dimension and significantly reduces quantization errors. KinMo (Zhang et al., 2025) proposes a Joint-Kinematic Motion Representation to explicitly capture joint-level interactions. SALAD (Hong et al., 2025) introduces specialized spatial attention and temporal attention into a latent diffusion model to model the joint features and temporal features independently. TriC-Motion (Cao et al., 2026) proposes a unified tri-domain framework that uses spatial-temporal-frequency-domain modeling to capture comprehensive motion features. Although these methods enhance the modeling of human motion dynamics, they still rely on standard pre-trained CLIP text encoders, leaving the synthesized motions struggling to accurately reflect the fine-grained dynamic nuances of the text prompts.

Specialized Text Encoder for Motion Generation. Several works design specialized text encoders for text-to-motion generation. For example, MoCLIP (Maldonado et al., 2025) adapts the vanilla CLIP via contrastive learning and distillation on motion datasets. Similarly, LAMP (Li et al., 2025) pretrains a pair of text and motion encoders using four proxy tasks (e.g., contrastive learning) and several auxiliary classifier heads. However, as revealed by Fig. 1, these retrieval-oriented objectives may not be inherently semantic-sensitive; they often overlook the fine-grained dynamic cues essential for accurate motion synthesis. Consequently, these separately trained encoders may not provide optimal text representations for downstream motion generators. Instead of separately training text encoders using these proxy objectives, we argue that the optimal text representations should be generation-oriented. To this end, we introduce a co-optimization framework that enables the autonomous adaptation of the text encoder driven directly by the quality of the generated motions.

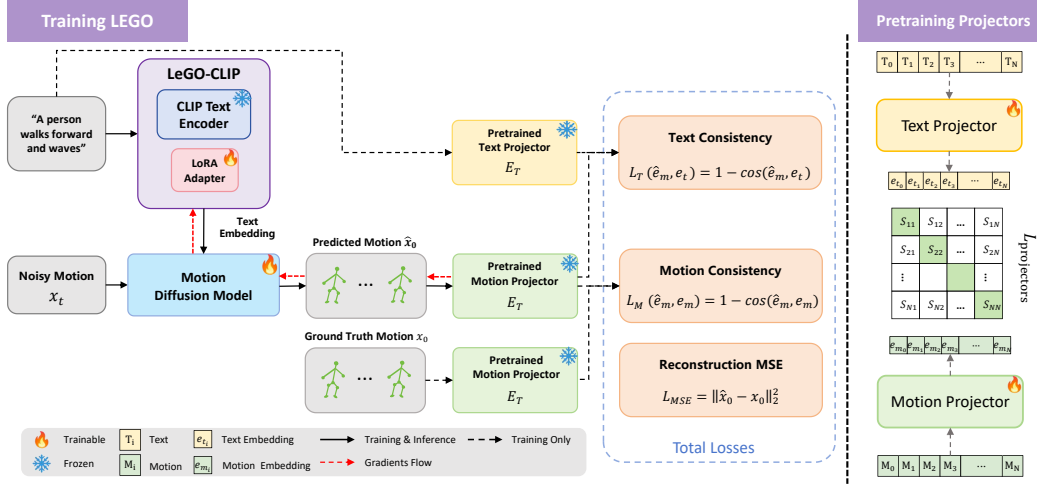


Figure 3: **Overview of LeGO framework.** (Left) LeGO is used to co-optimize LeGO-CLIP and a motion diffusion model. Its training objectives include the basic mean square error (MSE) loss, the text consistency constraint L_T , and the motion consistency constraint L_M . (Right) Before training LeGO, a pair of text and motion projectors is trained using InfoNCE and cosine similarity losses.

3 METHODOLOGY

Given a text prompt c as a condition, our goal is to generate a motion sequence $x \in \mathbb{R}^{N \times D}$, where N is the sequence length, and D is the pose dimension. In this section, we present LeGO, a co-optimization framework for learning generation-oriented text representations for motion generation. The key idea is to transform motion generation quality into differentiable supervision for the text encoder, so that the text representation is optimized according to its usefulness for downstream synthesis rather than a standalone proxy objective. In this sense, the semantic consistency constraints are the mechanism by which LeGO learns generation-oriented text representations from output-side generation feedback. To realize this idea, we first construct a shared semantic space for text and motion using pretrained projectors (Sec. 3.2), then derive generation-quality supervision via semantic consistency constraints (Sec. 3.3), and finally instantiate text encoder adaptation with LoRA (Sec. 3.4).

3.1 PRELIMINARIES

Diffusion models (Ho et al., 2020) are generative models that learn to synthesize high-quality data from random noise. They operate through a two-phase framework: a diffusion process that gradually adds noise to corrupt the original data, and a denoising process that iteratively denoises noisy data to reconstruct the sample.

Diffusion process. Starting with a motion sequence x_0 sampled from the training data distribution, the diffusion process transforms x_0 into noisy data $\{x_t\}_{t=1}^T$ with varying noise levels, where T is the maximum diffusion process step. This process is formulated as $x_t = \sqrt{1 - \beta_t}x_{t-1} + \sqrt{\beta_t}\epsilon$, where β_t is the noise scheduler and ϵ is gaussian noise. A notable property of the forward process is that it admits sampling x_t at an arbitrary timestep t in one step:

$$x_t = \sqrt{\bar{\alpha}_t}x_0 + \sqrt{1 - \bar{\alpha}_t}\epsilon, \quad \bar{\alpha}_t = \prod_{s=1}^t (1 - \beta_s). \quad (1)$$

Denoising process. The denoising process models the conditional distribution $p_\theta(\hat{x}_0|x_T, c)$ based on the text prompt c and obtains the synthesized motion $\hat{x}_0$. This was achieved using a denoising network to denoise x_t to $\hat{x}_0$, iteratively changing t from T to 1. The adoption of x_0 -prediction allows us to directly apply differential constraints on the estimated motion $\hat{x}_0$.

3.2 TRAINING TEXT AND MOTION PROJECTORS

Most existing text-to-motion generation methods rely primarily on the mean squared error (MSE) loss, which provides only low-level frame-wise supervision and is insufficient for evaluating motion generation quality at the semantic level. To optimize text representations using generation quality, we therefore construct a shared semantic space in which the conditioning text, the predicted motion, and the ground-truth motion can be compared at the representation level. Specifically, we train a text projector E_T and a motion projector E_M to map the conditional text c , the predicted motion $\hat{x}_0$, and the ground-truth motion x_0 into corresponding embeddings e_c , $\hat{e}_m$, and e_m , respectively. We enforce two key properties in this space: (1) embeddings of paired text-motion samples should be highly aligned, and (2) embeddings of unmatched samples within a mini-batch should remain highly discriminative.

Specifically, we build the text and motion projectors using Gated Recurrent Units (GRUs) following T2M (Guo et al., 2022) for training efficiency. The text projector takes the conditional text and its length as input, and employs a GRU layer to extract sequential features, with the final hidden state serving as the resulting text embedding. Similarly, the motion projector maps the motion sequence into a corresponding motion embedding using the same GRU-based architecture.

To optimize these projectors, we apply an InfoNCE loss (Oord et al., 2018) and a cosine similarity loss. The InfoNCE loss facilitates cross-modal alignment by pulling paired text-motion samples closer while pushing apart unpaired ones within a batch. In parallel, the cosine similarity loss is imposed on paired text-motion samples and explicitly maximizes their angular similarity, thereby reinforcing fine-grained alignment of semantically corresponding samples.

Given a batch of N paired text-motion embeddings $(e_{t_1}, e_{m_1}), \dots, (e_{t_N}, e_{m_N})$, we define the pairs with $i = j$ as positive samples and those (e_{t_i}, e_{m_j}) with $i \neq j$ as negative samples. We obtain the similarity matrix by computing the pairwise cosine similarities for all pairs of embeddings in the batch, where each element in this matrix is denoted as $S_{ij} = \cos(e_{t_i}, e_{m_j})$. The loss function to train the text and motion projectors can be formulated as follows:

$$\mathcal{L}_{\text{projectors}} = -\frac{1}{2N} \sum_i \left(-\log \frac{\exp(S_{ii}/\tau)}{\sum_j \exp(S_{ij}/\tau)} - \log \frac{\exp(S_{ii}/\tau)}{\sum_j \exp(S_{ji}/\tau)} \right) + \frac{1}{N} \sum_i S_{ii}. \quad (2)$$

3.3 TRAINING WITH DIFFERENTIAL CONSISTENCY CONSTRAINTS

With the above pretrained text and motion projectors, we now formulate motion generation quality as two complementary semantic consistency constraints: a Motion Consistency Constraint ($\mathcal{L}_M$) and a Text Consistency Constraint ($\mathcal{L}_T$). $\mathcal{L}_M$ evaluates whether the generated motion remains semantically consistent with the ground-truth motion, thereby capturing structural and perceptual characteristics beyond frame-wise reconstruction. $\mathcal{L}_T$ evaluates whether the generated motion remains semantically aligned with the conditioning text, thereby measuring its faithfulness to the textual instruction. The pretrained projectors are kept frozen when computing both consistency constraints. Together, these two constraints provide high-level semantic supervision that complements the standard MSE loss, and their gradients are backpropagated through the generator to adapt the text encoder toward generation-oriented representations. We provide further intuitive insights into these consistency constraints in the Appendix.

Motion Consistency Constraint is designed as a cosine similarity loss in the embedding space during training. Let x_0 denote the ground-truth motion sequence and $\hat{x}_0$ the generated motion produced by the diffusion model. Both sequences are projected into an embedding space by the motion projector E_M : $\hat{e}_m = E_M(\hat{x}_0)$ and $e_m = E_M(x_0)$. Thus, $\mathcal{L}_M(\hat{e}_m, e_m)$ is formulated as follows:

$$\mathcal{L}_M(\hat{e}_m, e_m) = 1 - \cos(\hat{e}_m, e_m) = 1 - \frac{\hat{e}_m \cdot e_m}{\|\hat{e}_m\| \|e_m\|}. \quad (3)$$

Text Consistency Constraint is also designed as a cosine similarity loss in the embedding space, providing cross-modal semantic supervision. We project both the generated motion $\hat{x}_0$ and the input textual description c into the embedding space using the text projector E_T and the motion projector E_M , respectively: $\hat{e}_m = E_M(\hat{x}_0)$ and $e_t = E_T(c)$. Thus, $\mathcal{L}_T(\hat{e}_m, e_t)$ is formulated as follows:

$$\mathcal{L}_T(\hat{e}_m, e_t) = 1 - \cos(\hat{e}_m, e_t) = 1 - \frac{\hat{e}_m \cdot e_t}{\|\hat{e}_m\| \|e_t\|}. \quad (4)$$

Table 1: **Quantitative Comparisons on the HumanML3D Dataset.** We separate all methods in this table into diffusion-based methods and VQ-based methods. $\pm$ indicates a 95% confidence interval. **Bold** face indicates the best result, while underscore refers to the second best.

	Methods	R-Precision $\uparrow$			FID $\downarrow$	MM Dist $\downarrow$	MultiModality $\uparrow$	Diversity $\rightarrow$
		Top 1	Top 2	Top 3				
VAE/VQ	T2M-GPT (Zhang et al., 2023a)	0.492 $\pm$.003	0.679 $\pm$.002	0.775 $\pm$.002	0.141 $\pm$.005	3.121 $\pm$.009	1.831 $\pm$.048	9.722 $\pm$.082
	MMM (Pinyoanuntapong et al., 2023)	0.504 $\pm$.003	0.696 $\pm$.003	0.794 $\pm$.002	0.080 $\pm$.003	2.998 $\pm$.007	1.164 $\pm$.041	9.411 $\pm$.058
	MoMask (Guo et al., 2024)	0.521 $\pm$.002	0.713 $\pm$.002	0.807 $\pm$.002	0.045 $\pm$.002	2.958 $\pm$.008	1.241 $\pm$.040	-
	BAMM (Pinyoanuntapong et al., 2024)	0.525 $\pm$.002	0.720 $\pm$.003	0.814 $\pm$.003	0.055 $\pm$.002	2.919 $\pm$.008	1.687 $\pm$.051	9.717 $\pm$.089
	KinMo (Zhang et al., 2025)	0.529 $\pm$.003	0.722 $\pm$.002	0.817 $\pm$.002	0.050 $\pm$.003	2.907 $\pm$.009	1.313 $\pm$.041	9.684 $\pm$.063
	MoMask+MoCLIP (Maldonado et al., 2025)	0.533 $\pm$.003	0.730 $\pm$.002	0.823 $\pm$.001	0.047 $\pm$.002	2.868 $\pm$.006	1.242 $\pm$.040	9.619 $\pm$.082
	LAMP (Li et al., 2025)	0.557$\pm$.003	0.751$\pm$.002	0.843$\pm$.001	0.032$\pm$.002	2.759$\pm$.007	-	<u>9.571$\pm$.069</u>
	Motion-R1 (Ouyang et al., 2026)	0.515 $\pm$.003	0.719 $\pm$.002	0.818 $\pm$.002	0.201 $\pm$.004	2.854 $\pm$.010	2.317 $\pm$.105	10.026 $\pm$.075
	MoMask+MATE (Han et al., 2026)	<u>0.550$\pm$.002</u>	<u>0.737$\pm$.002</u>	<u>0.832$\pm$.002</u>	<u>0.040$\pm$.002</u>	<u>2.811$\pm$.007</u>	1.369 $\pm$.036	9.516$\pm$.092
Diffusion	MotionDiffuse (Zhang et al., 2024a)	0.491 $\pm$.001	0.681 $\pm$.001	0.782 $\pm$.001	0.630 $\pm$.001	3.113 $\pm$.001	1.553 $\pm$.042	9.410 $\pm$.049
	MDM (Tevet et al., 2023)	0.455 $\pm$.006	0.645 $\pm$.007	0.749 $\pm$.006	0.489 $\pm$.047	3.330 $\pm$.025	2.290 $\pm$.070	<u>9.446$\pm$.102</u>
	MLD (Chen et al., 2023)	0.481 $\pm$.003	0.673 $\pm$.003	0.772 $\pm$.002	0.473 $\pm$.013	3.196 $\pm$.010	2.413 $\pm$.079	9.724 $\pm$.082
	ReMoDiffuse (Zhang et al., 2023b)	0.510 $\pm$.005	0.698 $\pm$.006	0.795 $\pm$.004	0.103 $\pm$.004	2.974 $\pm$.016	1.795 $\pm$.043	9.018 $\pm$.075
	StableMoFusion (Huang et al., 2024a)	0.553 $\pm$.003	0.748 $\pm$.002	0.841 $\pm$.002	0.098 $\pm$.003	-	1.774 $\pm$.051	9.748 $\pm$.092
	SALAD (Hong et al., 2025)	0.581 $\pm$.003	0.769 $\pm$.003	0.857 $\pm$.002	<u>0.076$\pm$.002</u>	2.649 $\pm$.009	1.751 $\pm$.062	9.696 $\pm$.096
	TriC-Motion (Cao et al., 2026)	<u>0.612$\pm$.006</u>	<u>0.806$\pm$.005</u>	<u>0.885$\pm$.004</u>	0.285 $\pm$.042	<u>2.465$\pm$.042</u>	-	9.434 $\pm$.089
	LeGO(Ours)	0.627$\pm$.004	0.811$\pm$.003	0.888$\pm$.002	0.039$\pm$.001	2.382$\pm$.010	1.184 $\pm$.042	9.497$\pm$.115

Table 2: **Quantitative Comparisons on SnapMoGen test set.** The text encoder used by each method is indicated in parentheses after its name. We separate all methods in this table into diffusion-based methods and VQ-based methods.

	Methods	R Precision $\uparrow$			FID $\downarrow$	CLIP Score $\uparrow$	MModality $\uparrow$
		Top 1	Top 2	Top 3			
	Real motions	0.940 $\pm$.001	0.976 $\pm$.001	0.985 $\pm$.001	0.001 $\pm$.000	0.837 $\pm$.000	-
VAE/VQ	T2M-GPT (T5-base) (Zhang et al., 2023a)	0.618 $\pm$.002	0.773 $\pm$.002	0.812 $\pm$.002	32.62 $\pm$.087	0.573 $\pm$.001	<u>9.172</u> $\pm$.181
	MoMask (T5-base) (Guo et al., 2024)	0.777 $\pm$.002	0.888 $\pm$.002	0.927 $\pm$.002	17.40 $\pm$.051	0.664 $\pm$.001	8.183 $\pm$.184
	MoMask++ (T5-base) (Guo et al., 2025)	<u>0.802</u> $\pm$.001	<u>0.905</u> $\pm$.002	<u>0.938</u> $\pm$.001	15.06 $\pm$.065	<u>0.685</u> $\pm$.001	7.259 $\pm$.180
	ScaleMoGen (T5-base) (Hwang et al., 2026)	0.807 $\pm$.004	0.908 $\pm$.003	0.941 $\pm$.003	<u>16.35</u> $\pm$.086	0.693 $\pm$.001	9.399 $\pm$.669
Diffusion	MDM (CLIP) (Tevet et al., 2023)	0.334 $\pm$.002	0.469 $\pm$.002	0.549 $\pm$.002	42.48 $\pm$.083	0.360 $\pm$.002	20.78 $\pm$.885
	MDM (T5-base) (Tevet et al., 2023)	0.503 $\pm$.002	0.653 $\pm$.002	0.727 $\pm$.002	57.78 $\pm$.092	0.481 $\pm$.001	13.41 $\pm$.231
	MDM (BERT) (Tevet et al., 2023)	0.795 $\pm$.002	0.892 $\pm$.002	0.929 $\pm$.003	22.65 $\pm$.069	0.650 $\pm$.001	<u>15.53</u> $\pm$.945
	StableMoFusion (T5-base) (Huang et al., 2024a)	0.679 $\pm$.002	0.823 $\pm$.002	0.888 $\pm$.002	27.80 $\pm$.063	0.605 $\pm$.001	9.064 $\pm$.138
	MARDM (T5-base) (Meng et al., 2025)	0.659 $\pm$.002	0.812 $\pm$.002	0.860 $\pm$.002	26.87 $\pm$.131	0.602 $\pm$.001	9.812 $\pm$.287
	SALAD (T5-base) (Hong et al., 2025)	0.742 $\pm$.002	.875 $\pm$.003	0.922 $\pm$.002	23.24 $\pm$.091	0.658 $\pm$.002	9.238 $\pm$.175
	CMDM (BERT) (Yu et al., 2026)	0.831 $\pm$.004	<u>0.926</u> $\pm$.003	<u>0.958</u> $\pm$.002	14.45 $\pm$.074	<u>0.702</u> $\pm$.001	9.521 $\pm$.196
	LeGO(CLIP)	<u>0.832</u> $\pm$.001	0.924 $\pm$.001	0.956 $\pm$.001	17.64 $\pm$.056	0.668 $\pm$.001	12.83 $\pm$.592
	LeGO(BERT)	0.911 $\pm$.001	0.956 $\pm$.001	0.980 $\pm$.001	<u>16.64</u> $\pm$.036	0.724 $\pm$.001	10.45 $\pm$.379

3.4 CO-OPTIMIZING LEGO-CLIP WITH LORA

To adapt the pretrained CLIP text encoder for motion generation without updating all its parameters, we use Low-Rank Adaptation (LoRA) (Hu et al., 2022) to reduce the memory cost of fine-tuning. Specifically, we insert learnable low-rank adapters into the CLIP text encoder and jointly optimize these adapters with the motion diffusion model. The resulting text encoder, denoted as LeGO-CLIP, receives gradients from the proposed semantic consistency constraints through the generator, enabling generation-oriented text representation learning while keeping the original CLIP parameters frozen. After training, LeGO-CLIP can be used as a frozen replacement for vanilla CLIP in other motion generation frameworks, as evaluated in Sec. 4.2.

3.5 OVERALL TRAINING OBJECTIVE

The overall training objective of our LeGO can be formulated as follows:

$$\mathcal{L} = \mathcal{L}_{mse}(x_0, \hat{x}_0) + \lambda_M \mathcal{L}_M + \lambda_T \mathcal{L}_T, \quad (5)$$

where $\mathcal{L}_{mse}$ denotes the MSE loss between the predicted motion and the ground-truth motion. λ_M and λ_T control the strengths of $\mathcal{L}_M$ and $\mathcal{L}_T$, respectively. We conduct ablation studies on hyperparameters in Appendix K.

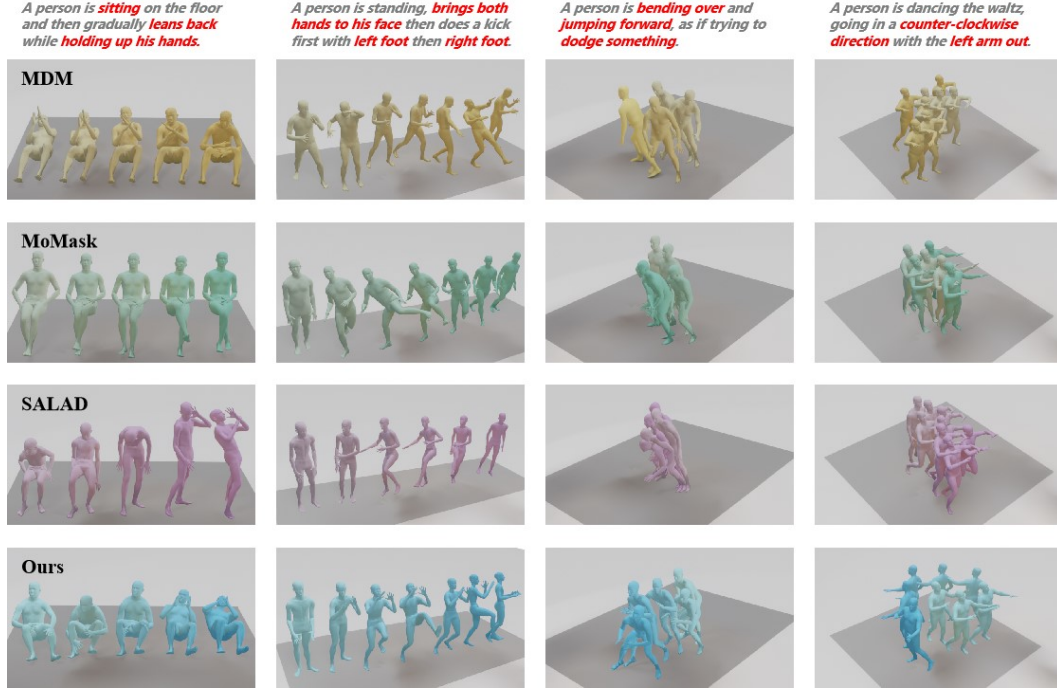


Figure 4: **Qualitative Comparisons between MDM (Tevet et al., 2023), MoMask (Guo et al., 2024), SALAD (Hong et al., 2025), and ours.** Action-relevant keywords are highlighted in red.

4 EXPERIMENTS

LeGO-CLIP is realized through the LeGO framework. We evaluate the performance of LeGO and the derived LeGO-CLIP, respectively. We conduct comprehensive experiments on two mainstream text-to-motion datasets, namely HumanML3D (Guo et al., 2022) and KIT-ML (Plappert et al., 2016), following the evaluation protocol proposed by T2M (Guo et al., 2022). We provide a detailed introduction to metrics in Appendix. Detailed experimental settings are provided in Appendix D.

4.1 COMPARISON TO STATE-OF-THE-ART METHODS

Quantitative comparisons with previous methods on the HumanML3D (Guo et al., 2022) are presented in Table 1, and those on the KIT-ML dataset (Plappert et al., 2016) are presented in Appendix E. Our method surpasses existing approaches and achieves state-of-the-art performance in both FID and R-Precision across the HumanML3D and KIT-ML benchmarks. These results underscore the significance of learning generation-oriented text representations discussed in Sec. 1. Notably, compared to MDM with the same backbone diffusion model, our method reduces FID from 0.489 to 0.039 (90% $\uparrow$) and boosts R-Precision from 0.749 to 0.888. Although TriC-Motion (Cao et al., 2026) is the only recent work that also achieves a similar R-Precision level (0.88) via joint spatial-temporal-frequency modeling, its FID (0.285) is inferior to ours (0.039). LeGO also achieves state-of-the-art zero-shot performance on the HumanML3D and KIT-ML benchmarks using the essential motion representation (Meng et al., 2025). Please refer to Appendix A.

Qualitative Comparisons. Figure 4 presents a qualitative comparison between our LeGO and three representative baselines: MDM (Tevet et al., 2023), MoMask (Guo et al., 2024), and SALAD (Hong et al., 2025). As observed, MDM follows the text prompt roughly and fails to execute fine-grained details such as “*dodge something*”. MoMask and SALAD can synthesize relatively fluent motions; however, they still struggle to capture specific actions like “*sitting on the floor*” and “*jumping forward*”. In contrast, our method demonstrates a superior understanding of complex text instructions. For instance, in the second sample of Figure 4, LeGO precisely executes intricate details like “*bring both hands to his face*”. For additional visualization results, please refer to the supplementary videos.

Table 3: **Transferability Experiments of LeGO-CLIP on HumanML3D Dataset.**

Methods	R Precision $\uparrow$			FID $\downarrow$	MM Dist $\downarrow$	MultiModality $\uparrow$
	Top 1	Top 2	Top 3			
MDM (Tevet et al., 2023)	0.455 $\pm$.006	0.645 $\pm$.007	0.749 $\pm$.006	0.489 $\pm$.047	3.330 $\pm$.025	2.290 $\pm$.070
MDM+LeGO-CLIP	0.539$\pm$.003	0.726$\pm$.004	0.820$\pm$.005	0.101$\pm$.005 80% $\uparrow$	2.737$\pm$.013	1.083 $\pm$.052
MoMask (Guo et al., 2024)	0.521 $\pm$.002	0.713 $\pm$.002	0.807 $\pm$.002	0.045 $\pm$.002	2.958 $\pm$.008	1.241 $\pm$.040
MoMask+MoCLIP (Maldonado et al., 2025)	0.533 $\pm$.003	0.730 $\pm$.002	0.823 $\pm$.001	0.047 $\pm$.002	2.868 $\pm$.006	1.242 $\pm$.040
MoMask+MATE (Han et al., 2026)	0.550 $\pm$.002	0.737 $\pm$.002	0.832 $\pm$.002	0.040$\pm$.002	2.811 $\pm$.007	1.369 $\pm$.036
MoMask+LeGO-CLIP	0.554$\pm$.003	0.751$\pm$.002	0.842$\pm$.003	0.040$\pm$.002 15% $\uparrow$	2.771$\pm$.009	1.165 $\pm$.044
SALAD (Hong et al., 2025)	0.581 $\pm$.003	0.769 $\pm$.003	0.857 $\pm$.002	0.076 $\pm$.002	2.649 $\pm$.009	1.751 $\pm$.062
SALAD+LeGO-CLIP	0.590$\pm$.003	0.779$\pm$.002	0.864$\pm$.001	0.061$\pm$.003 20% $\uparrow$	2.625$\pm$.007	1.400 $\pm$.056

4.2 TRANSFERABILITY OF LEGO-CLIP TO EXISTING MOTION GENERATORS

Our LeGO-CLIP learns generation-oriented text representations that can be readily transferred to other motion generation frameworks, thereby boosting their performance. To demonstrate this transferability, we replace the vanilla CLIP with LeGO-CLIP trained on the HumanML3D dataset for three state-of-the-art models: MDM (Tevet et al., 2023) (standard diffusion-based), MoMask (Guo et al., 2024) (VQ-VAE-based), and SALAD (Hong et al., 2025) (latent-diffusion-based), and train them from scratch. For each model, we keep the original architecture, training schedule, and inference setting unchanged, so that the performance difference can be attributed to the text encoder replacement. For fair comparison, all text encoders in Table 3 are trained on the HumanML3D training split. As shown in Table 3, LeGO-CLIP consistently improves all three generators in terms of FID and R-Precision, with up to 80% relative FID reduction. These results demonstrate that LeGO-CLIP learns transferable text representations that benefit diverse motion generation frameworks.

Notably, although LeGO-CLIP is optimized through the sentence-level $[\text{CLS}]$ token during LeGO training, its improvement is not limited to sentence-level conditioning. SALAD relies on word-level embeddings for text conditioning, yet replacing its text encoder with LeGO-CLIP still brings clear performance gains.

4.3 SEMANTIC SENSITIVITY ANALYSIS

Given the high-dimensional complexity of text embeddings, it is challenging to quantify how well a text encoder represents global sentence semantics. Instead, we propose evaluating the discriminative ability of text encoders in capturing motion-relevant fine-grained nuances (e.g., dynamic keywords). We term this evaluation semantic sensitivity analysis.

Experimental Setup. We conduct the semantic sensitivity analysis on HumanML3D. We define a set of paired keywords that represent opposing or distinct dynamic characteristics, including speed (*slowly/quickly*), direction (*forward/backward*, *clockwise/counter-clockwise*, and *left/right*), and action intensity (*walk/run*). We traverse all 4,384 text files in the test set and build a total of 11,205 adversarial text pairs by replacing keywords with their counterparts (e.g., replacing “left” with “right”).

We evaluate three text encoders: CLIP (Radford et al., 2021), LAMP (Li et al., 2025), and our LeGO-CLIP. We use them to extract embeddings from these adversarial text pairs and compute the pairwise cosine similarities. In this adversarial setting, a high cosine similarity indicates that the encoder maps distinct text descriptions into nearly identical representations, potentially leading to the omission of subtle motion differences. Conversely, a lower cosine similarity indicates that the encoder successfully projects them into distinguishable embeddings.

Table 4: **Mean and Standard Deviation** of cosine similarity of the vanilla CLIP (Radford et al., 2021), LAMP (Li et al., 2025), and our LeGO-CLIP.

Text Encoder	Mean	Std
CLIP (Radford et al., 2021)	0.9815	0.0144
LAMP (Li et al., 2025)	0.9001	0.1362
LeGO-CLIP (Ours)	0.7546	0.1756

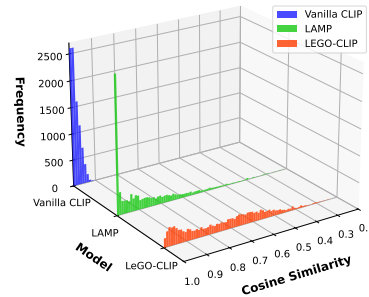
Figure 5: **3D Frequency Histogram** of cosine similarities of vanilla CLIP (Radford et al., 2021), LAMP (Li et al., 2025), and LeGO-CLIP.

Table 5: Ablation of Proposed Components on HumanML3D.

Ablation	Method	Consistency Constraints	Tuning CLIP	FID ↓	R-Precision ↑ (Top-3)	Skating ↓ Ratio
①	baseline			0.489 $\pm$.047	0.749 $\pm$.006	0.095 $\pm$.0009
②	with consistency constraints	✓		0.098 $\pm$.002	0.838 $\pm$.002	0.069 $\pm$.0008
③	with LoRA tuning		LoRA	0.566 $\pm$.004	0.802 $\pm$.002	0.107 $\pm$.0007
④	Ours (full model)	✓	LoRA	0.039$\pm$.001	0.888$\pm$.002	0.061$\pm$.0007
⑤	Ours with full tuning	✓	Full	0.073 $\pm$.002	0.873 $\pm$.002	0.063 $\pm$.0007

Results. The statistical results are summarized in Table 4. Vanilla CLIP exhibits limited sensitivity to subtle semantic changes, as indicated by its high mean similarity and low standard deviation. In contrast, LeGO-CLIP achieves a substantially lower mean similarity and a higher standard deviation, suggesting that it better separates text pairs with fine-grained motion-relevant differences. The histogram in Fig. 5 further shows that Vanilla CLIP similarities are highly concentrated above 0.9, while LAMP still retains many high-similarity pairs. By comparison, LeGO-CLIP produces a much broader distribution, indicating stronger discriminability across adversarial text pairs. For additional analysis across different text lengths, please refer to Appendix F.

We note that this semantic sensitivity analysis provides *input-side* evidence, i.e., it evaluates whether adversarial prompts are distinguishable in the text embedding space. To further verify whether such distinctions are reflected in generated motions, we conduct extensive *output-side* controllability experiments in Appendix B, where motions are generated from adversarial text pairs under identical noise. These results further demonstrate that our method follows fine-grained textual instructions more faithfully during motion generation.

4.4 ABLATION STUDIES ON PROPOSED COMPONENTS

We conduct several ablation experiments to evaluate each component’s contribution, including: 1) the baseline without any proposed component, 2) applying consistency constraints only, 3) tuning CLIP only, and 4) applying consistency constraints while co-optimizing CLIP. We introduce Skating Ratio to evaluate the physical fidelity, following (Karunratanakul et al., 2023) and (Xie et al., 2024). We present the quantitative results in Table 5. Ablation ② reveals that even without finetuning CLIP, the consistency constraints still regularize the motion diffusion model to produce better motions. The strong performance of the consistency-only variant confirms that semantic consistency constraints provide an effective generation-oriented supervision signal. Ablation ③ shows that solely finetuning CLIP without consistency constraints substantially deteriorates the generation quality. We attribute this degradation to shortcut learning, where CLIP may drift away from meaningful embeddings when optimized only through the reconstruction loss. In contrast, in Ablation ④, combining the consistency constraints with CLIP tuning yields significant improvement. This indicates that such supervision can further guide LeGO-CLIP toward generation-oriented text representations. We adopt LoRA for CLIP adaptation, which is more parameter-efficient and performs better than full-parameter fine-tuning, as shown by Ablation ⑤.

5 CONCLUSION

In this paper, we identify a key limitation of text encoders pretrained on proxy objectives for text-to-motion generation: they often fail to capture fine-grained motion semantics required by the generation task. To address this issue, we propose LeGO, a co-optimization framework that derives LeGO-CLIP, a motion-specialized text encoder, by training it together with a motion diffusion model under differentiable semantic consistency constraints. Extensive experiments on HumanML3D and KIT-ML demonstrate that LeGO achieves state-of-the-art performance, improves data efficiency, and transfers effectively across diverse motion generation frameworks. **Limitation.** One limitation of LeGO is that training relies on two additional projector modules. Although these projectors are lightweight, they still introduce extra training time and optimization complexity. Reducing this overhead, or designing projector-free alternatives, is an important direction for future work. **Broader Impact.** This work suggests that representation learning for generative tasks should be aligned with generation quality rather than solely with proxy pretraining objectives. We hope this perspective can benefit future research on motion generation and other multimodal generative problems. To the best of our knowledge, this work has no obvious negative social impacts.

AI USE STATEMENT

In this work, we used generative AI tools for assisting with code implementation, editing, completion, and debugging. We have not used generative AI tools for generating synthetic datasets, developing theoretical models or conceptual frameworks, formulating mathematical claims, providing critical ingredients for or assisting in writing proofs, proposing or refining research hypotheses, designing or providing feedback on research methodology or experiments, assisting with translation, cleaning or reformatting datasets, supporting qualitative or thematic data analysis, or interpreting experimental results, and the remaining required disclosure tasks are not applicable to this work. Additionally, we used generative AI tools for drafting and editing parts of the manuscript, improving grammar, clarity, organization, and readability, and assisting with general software code creation or editing. We have reviewed all AI-assisted work. All AI-assisted manuscript text was carefully reviewed and revised by the authors to ensure factual and technical accuracy and to preserve the authors' intended meaning. All AI-assisted code was manually inspected, executed, tested, and validated by the authors before being used in the experiments. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

To support reproducibility, we provide detailed descriptions of the dataset construction, model architecture, implementation details, training procedure, and evaluation protocols in the main paper and appendix. Implementation details are included in the supplementary material. Upon acceptance, we will publicly release the datasets, source code, pretrained models, and data-processing scripts to facilitate reproduction and future research.

REFERENCES

- Yiyang Cao, Yunze Deng, Ziyu Lin, Bin Feng, Xinggang Wang, Wenyu Liu, Dandan Zheng, and Jingdong Chen. Tric-motion: Tri-domain causal modeling grounded text-to-motion generation. In *ICLR*, 2026.
- Huiwen Chang, Han Zhang, Lu Jiang, Ce Liu, and William T Freeman. Maskgit: Masked generative image transformer. In *CVPR*, 2022.
- Xin Chen, Biao Jiang, Wen Liu, Zilong Huang, Bin Fu, Tao Chen, and Gang Yu. Executing your commands via motion diffusion in latent space. In *CVPR*, 2023.
- Chuan Guo, Shihao Zou, Xinxin Zuo, Sen Wang, Wei Ji, Xingyu Li, and Li Cheng. Generating diverse and natural 3d human motions from text. In *CVPR*, 2022.
- Chuan Guo, Yuxuan Mu, Muhammad Gohar Javed, Sen Wang, and Li Cheng. Momask: Generative masked modeling of 3d human motions. In *CVPR*, 2024.
- Chuan Guo, Inwoo Hwang, Jian Wang, and Bing Zhou. Snapmogen: Human motion generation from expressive texts. In *Advances in Neural Information Processing Systems*, 2025.
- Prerit Gupta, Shourya Verma, Ananth Grama, and Aniket Bera. Unified multi-modal interactive & reactive 3d motion generation via rectified flow. In *ICLR*, 2026.
- Ke Han, Yueming Lyu, and Nicu Sebe. Motion-aligned word embeddings for text-to-motion generation. In *ICLR*, 2026.
- Yannan He, Garvita Tiwari, Xiaohan Zhang, Pankaj Bora, Tolga Birdal, Jan Eric Lenssen, and Gerard Pons-Moll. Molingo: Motion-language alignment for text-to-motion generation. In *CVPR*, 2026.
- Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598*, 2022.
- Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *NIPS*, 2020.
- Daniel Holden, Taku Komura, and Jun Saito. Phase-functioned neural networks for character control. *ACM Transactions on Graphics*, 2017.

- Fangzhou Hong, Mingyuan Zhang, Liang Pan, Zhongang Cai, Lei Yang, and Ziwei Liu. Avatarclip: Zero-shot text-driven generation and animation of 3d avatars. *ACM Transactions on Graphics*, 2022.
- Seokhyeon Hong, Chaelin Kim, Serin Yoon, Junghyun Nam, Sihun Cha, and Junyong Noh. Salad: Skeleton-aware latent diffusion for text-driven motion generation and editing. In *CVPR*, pp. 7158–7168, 2025.
- Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, et al. Lora: Low-rank adaptation of large language models. In *ICLR*, 2022.
- Yiheng Huang, Hui Yang, Chuanchen Luo, Yuxi Wang, Shibiao Xu, Zhaoxiang Zhang, Man Zhang, and Junran Peng. Stablemofusion: Towards robust and efficient diffusion-based motion generation framework. In *MM*, pp. 224–232, 2024a.
- Yiming Huang, Weilin Wan, Yue Yang, Chris Callison-Burch, Mark Yatskar, and Lingjie Liu. Como: Controllable motion generation through language guided pose code editing. In *ECCV*, 2024b.
- Inwoo Hwang, Hojun Jang, Bing Zhou, Jian Wang, Young Min Kim, and Chuan Guo. Scalemogen: Autoregressive next-scale prediction for human motion generation. In *ECCV*, 2026.
- Peng Jin, Yang Wu, Yanbo Fan, Zhongqian Sun, Wei Yang, and Li Yuan. Act as you wish: Fine-grained control of motion diffusion model with hierarchical semantic graphs. In *NIPS*, 2024.
- Korrawe Karunratanakul, Konpat Preechakul, Supasorn Suwajanakorn, and Siyu Tang. Guided motion diffusion for controllable human motion synthesis. In *ICCV*, pp. 2151–2162, 2023.
- Tianyu Li, Calvin Qiao, Guanqiao Ren, KangKang Yin, and Sehoon Ha. Aamdm: Accelerated auto-regressive motion diffusion model. In *CVPR*, 2024.
- Zhe Li, Weihao Yuan, Yisheng He, Lingteng Qiu, Shenhao Zhu, Xiaodong Gu, Weichao Shen, Yuan Dong, Zilong Dong, and Laurence T Yang. Lamp: Language-motion pretraining for motion generation, retrieval, and captioning. In *ICLR*, 2025.
- Han Liang, Jiacheng Bao, Ruichi Zhang, Sihan Ren, Yuecheng Xu, Sibe Yang, Xin Chen, Jingyi Yu, and Lan Xu. Omg: Towards open-vocabulary motion generation via mixture of controllers. In *CVPR*, 2024.
- Gabriel Maldonado, Armin Danesh Pazho, Ghazal Alinezhad Noghre, Vinit Katariya, and Hamed Tabkhi. Moclip motion-aware fine-tuning and distillation of clip for human motion generation. In *CVPR*, pp. 2931–2941, 2025.
- Zichong Meng, Yiming Xie, Xiaogang Peng, Zeyu Han, and Huaizu Jiang. Rethinking diffusion for text-driven human motion generation. In *CVPR*, 2025.
- Yusuke Nishimura, Yutaka Nakamura, and Hiroshi Ishiguro. Long-term motion generation for interactive humanoid robots using gan with convolutional network. In *Companion of the ACM/IEEE Int. Conf. Hum.-Robot Interact.*, pp. 375–377, 2020. doi: 10.1145/3371382.3378386.
- Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.
- Runqi Ouyang, Haoyun Li, Zhenyuan Zhang, Xiaofeng Wang, Zeyu Zhang, Zheng Zhu, Guan Huang, Sirui Han, and Xingang Wang. Motion-rl: Enhancing motion generation with decomposed chain-of-thought and rl binding. In *ICLR*, 2026.
- Mathis Petrovich, Michael J Black, and Gül Varol. Temos: Generating diverse human motions from textual descriptions. In *ECCV*, 2022.
- Mathis Petrovich, Michael J Black, and Gül Varol. Tmr: Text-to-motion retrieval using contrastive 3d human motion synthesis. In *ICCV*, 2023.
- Ekkasit Pinyoanuntapong, Pu Wang, Minwoo Lee, and Chen Chen. Mmm: Generative masked motion model. In *CVPR*, 2023.

- Ekkasit Pinyoanuntapong, Muhammad Usama Saleem, Pu Wang, Minwoo Lee, Srijan Das, and Chen Chen. Bamm: Bidirectional autoregressive motion model. In *ECCV*, pp. 172–190. Springer, 2024.
- Matthias Plappert, Christian Mandery, and Tamim Asfour. The KIT motion-language dataset. *IEEE Trans. Big Data*, 4(4):236–252, dec 2016.
- Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *ICML*, 2021.
- Guy Tevet, Sigal Raab, Brian Gordon, Yoni Shafir, Daniel Cohen-or, and Amit Haim Bermano. Human motion diffusion model. In *ICLR*, 2023.
- Aaron Van Den Oord, Oriol Vinyals, et al. Neural discrete representation learning. In *NIPS*, 2017.
- Weilin Wan, Zhiyang Dou, Taku Komura, Wenping Wang, Dinesh Jayaraman, and Lingjie Liu. Tlcontrol: Trajectory and language control for human motion synthesis. In *ECCV*, 2024.
- Yiming Xie, Varun Jampani, Lei Zhong, Deqing Sun, and Huaizu Jiang. Omnicontrol: Control any joint at any time for human motion generation. In *ICLR*, 2024.
- Qing Yu, Akihisa Watanabe, and Kent Fujiwara. Causal motion diffusion models for autoregressive motion generation. In *CVPR*, 2026.
- Weihao Yuan, Yisheng He, Weichao Shen, Yuan Dong, Xiaodong Gu, Zilong Dong, Liefeng Bo, and Qixing Huang. Mogents: Motion generation based on spatial-temporal joint modeling. *NIPS*, 2024.
- Jianrong Zhang, Yangsong Zhang, Xiaodong Cun, Shaoli Huang, Yong Zhang, Hongwei Zhao, Hongtao Lu, and Xi Shen. T2m-gpt: Generating human motion from textual descriptions with discrete representations. In *CVPR*, 2023a.
- Mingyuan Zhang, Xinying Guo, Liang Pan, Zhongang Cai, Fangzhou Hong, Huirong Li, Lei Yang, and Ziwei Liu. Remodiffuse: Retrieval-augmented motion diffusion model. In *ICCV*, 2023b.
- Mingyuan Zhang, Zhongang Cai, Liang Pan, Fangzhou Hong, Xinying Guo, Lei Yang, and Ziwei Liu. Motiondiffuse: Text-driven human motion generation with diffusion model. *PAMI*, 2024a.
- Mingyuan Zhang, Huirong Li, Zhongang Cai, Jiawei Ren, Lei Yang, and Ziwei Liu. Finemogen: Fine-grained spatio-temporal motion generation and editing. In *NIPS*, 2024b.
- Pengfei Zhang, Pinxin Liu, Hyeonwoo Kim, Pablo Garrido, and Bindita Chaudhuri. Kinmo: Kinematic-aware human motion understanding and generation. In *ICCV*, 2025.
- Chongyang Zhong, Lei Hu, Zihao Zhang, and Shihong Xia. Attt2m: Text-driven human motion generation with multi-perspective attention mechanism. In *ICCV*, 2023.
- Wenyang Zhou, Zhiyang Dou, Zeyu Cao, Zhouyingcheng Liao, Jingbo Wang, Wenjia Wang, Yuan Liu, Taku Komura, Wenping Wang, and Lingjie Liu. Emdm: Efficient motion diffusion model for fast and high-quality motion generation. In *ECCV*, 2024.
- Qiran Zou, Shangyuan Yuan, Shian Du, Yu Wang, Chang Liu, Yi Xu, Jie Chen, and Xiangyang Ji. Parco: Part-coordinating text-to-motion synthesis. In *ECCV*, 2024.

The Appendix provides:

1. Zero-shot performance on essential motion representations proposed by MARDM (Meng et al., 2025) in Sec. A.
2. Output-side evidence of keyword-conditioned motion control in Sec. B.
3. Keyword control under compositional text prompts in Sec. C
4. Experimental settings in Sec. D.
5. Quantitative comparisons on the KIT-ML dataset in Sec. E.
6. Deeper analysis on the Relationship Between Text Length and Cosine Similarity in Sec. F.
7. Data efficiency of our method compared with the previous baselines in Sec. G.
8. Analysis and mitigation of potential evaluator coupling in Sec. H.
9. The retrieval performance of the projectors in Sec. I.
10. We further provide empirical analysis to justify our choice of GRU-based projectors in Sec. J.
11. Ablation studies on hyperparameters of consistency constraints in Sec. K
12. User studies in Sec. L.
13. The performance with extremely few training samples in Sec. M.
14. The evolution of metrics during training in Sec. N.
15. More quantitative results on the SnapMoGen dataset in Sec. ??.

A ZERO-SHOT EVALUATION ON ESSENTIAL MOTION REPRESENTATIONS

MARDM (Meng et al., 2025) highlights that the standard motion representations used in benchmarks like HumanML3D and KIT-ML contain highly redundant features, which may introduce distribution inconsistencies within diffusion-based frameworks and lead to insufficient learning. To address this problem, they propose using an **essential motion representation**, preserving the local positions and disregarding rotations and velocities. It includes $\text{num_joints} \times 3 + 1$ dimensions, resulting in 67 dimensions for HumanML3D and 64 for KIT-ML.

In this section, we aim to demonstrate that LeGO will not be affected by such a problem. Additionally, to further validate that the performance gains of LeGO stem from a genuine prompt understanding of dynamic motion semantics, we conduct a zero-shot evaluation on this essential-representation-based benchmark. Specifically, our LeGO model is trained to generate motions in the standard 263-dimensional representation, while for evaluation we directly take the first 67 dimensions of the generated motions and feed them into the MARDM evaluator, without any additional post-finetuning. As reported in Table 6, LeGO still maintains state-of-the-art performance in this **zero-shot** setting, surpassing the methods directly trained on essential representation by a large margin. This result highlights the importance of learning the generation-oriented text representations and the robustness of the learned text representations.

B OUTPUT-SIDE EVIDENCE OF KEYWORD-CONDITIONED MOTION CONTROL

While Table 4 provides **input-side** evidence by evaluating whether adversarial prompts are separated in the text embedding space, it does not directly verify whether such distinctions survive the generation process and manifest in generated motions. We therefore complement it with **output-side** evidence through a controllability analysis: for each adversarial prompt pair, we generate motions under identical noise and measure whether the resulting motions change along the expected physical dimension. This provides a stronger behavioral test of fine-grained text understanding.

Experimental Setup. We compare two models under identical seeds and noise: **MDM** and **LeGO**. The text encoders of both models are built upon the pretrained CLIP text encoder: MDM conditions the generator on the frozen vanilla CLIP text embeddings, while LeGO uses LeGO-CLIP, which is adapted from the same CLIP text encoder by our generation-oriented co-optimization framework. For

Framework	Methods	R-Precision $\uparrow$			FID $\downarrow$	MM Dist $\downarrow$	MModality $\uparrow$	CLIP-score $\uparrow$	
		Top 1	Top 2	Top 3					
HumanML3D	VQ	T2M-GPT (Zhang et al., 2023a)	0.470 $\pm$.003	0.659 $\pm$.002	0.758 $\pm$.002	0.335 $\pm$.003	3.505 $\pm$.017	2.018 $\pm$.053	0.607 $\pm$.005
		MMM (Pinyoanuntapong et al., 2023)	0.487 $\pm$.003	0.683 $\pm$.002	0.782 $\pm$.001	0.132 $\pm$.004	3.359 $\pm$.009	1.241 $\pm$.073	0.635 $\pm$.003
		MoMask (Guo et al., 2024)	0.490 $\pm$.004	0.687 $\pm$.003	0.786 $\pm$.003	0.116 $\pm$.006	3.353 $\pm$.010	1.263 $\pm$.079	0.637 $\pm$.003
		MoLingo (He et al., 2026)	0.544 $\pm$.002	0.739 $\pm$.002	0.832 $\pm$.002	0.066 $\pm$.003	-	1.226 $\pm$.063	0.686 $\pm$.001
	(AR) Diffusion	MDM-50Step (Tevet et al., 2023)	0.440 $\pm$.007	0.636 $\pm$.006	0.742 $\pm$.004	0.518 $\pm$.032	3.640 $\pm$.028	3.604 $\pm$.031	0.578 $\pm$.003
		MotionDiffuse (Zhang et al., 2024a)	0.450 $\pm$.006	0.641 $\pm$.005	0.753 $\pm$.005	0.778 $\pm$.005	3.490 $\pm$.023	3.179 $\pm$.046	0.606 $\pm$.004
		MLD (Chen et al., 2023)	0.461 $\pm$.004	0.651 $\pm$.004	0.750 $\pm$.003	0.431 $\pm$.014	3.445 $\pm$.019	3.506 $\pm$.031	0.610 $\pm$.003
		ReMoDiffuse (Zhang et al., 2023b)	0.468 $\pm$.003	0.653 $\pm$.003	0.754 $\pm$.005	0.883 $\pm$.021	3.414 $\pm$.020	2.703 $\pm$.154	0.621 $\pm$.003
		MARDM-DDPM (Meng et al., 2025)	0.492 $\pm$.006	0.690 $\pm$.005	0.790 $\pm$.005	0.116 $\pm$.004	3.349 $\pm$.010	2.470 $\pm$.053	0.637 $\pm$.005
		MARDM-SiT (Meng et al., 2025)	0.500 $\pm$.004	0.695 $\pm$.003	0.795 $\pm$.003	0.114 $\pm$.007	3.270 $\pm$.009	2.231 $\pm$.071	0.642 $\pm$.002
		SALAD* (Hong et al., 2025)	0.554 $\pm$.005	0.748 $\pm$.008	0.839 $\pm$.008	0.139 $\pm$.009	2.981 $\pm$.028	-	0.654 $\pm$.003
		LeGO*(Ours)	0.566 $\pm$.004	0.759 $\pm$.002	0.847 $\pm$.001	0.059 $\pm$.005	2.846 $\pm$.007	1.206 $\pm$.151	0.674 $\pm$.001
KIT	VQ	T2M-GPT (Zhang et al., 2023a)	0.359 $\pm$.007	0.553 $\pm$.007	0.690 $\pm$.013	0.593 $\pm$.053	3.765 $\pm$.046	1.798 $\pm$.157	0.651 $\pm$.005
		MMM (Pinyoanuntapong et al., 2023)	0.363 $\pm$.005	0.569 $\pm$.006	0.724 $\pm$.006	0.478 $\pm$.034	3.629 $\pm$.028	1.455 $\pm$.106	0.660 $\pm$.003
		MoMask (Guo et al., 2024)	0.369 $\pm$.005	0.588 $\pm$.005	0.731 $\pm$.005	0.411 $\pm$.026	3.577 $\pm$.021	1.309 $\pm$.058	0.669 $\pm$.002
		MDM (Tevet et al., 2023)	0.333 $\pm$.012	0.561 $\pm$.009	0.689 $\pm$.009	0.585 $\pm$.043	4.002 $\pm$.033	1.681 $\pm$.107	0.605 $\pm$.007
	(AR) Diffusion	MotionDiffuse (Zhang et al., 2024a)	0.344 $\pm$.009	0.536 $\pm$.007	0.658 $\pm$.007	3.845 $\pm$.087	4.167 $\pm$.054	1.774 $\pm$.217	0.626 $\pm$.006
		MLD (Chen et al., 2023)	0.351 $\pm$.007	0.536 $\pm$.007	0.658 $\pm$.007	0.492 $\pm$.047	3.746 $\pm$.044	1.803 $\pm$.164	0.646 $\pm$.006
		ReMoDiffuse (Zhang et al., 2023b)	0.356 $\pm$.004	0.572 $\pm$.007	0.706 $\pm$.009	1.725 $\pm$.053	3.735 $\pm$.036	1.928 $\pm$.127	0.665 $\pm$.005
		MARDM-DDPM (Meng et al., 2025)	0.375 $\pm$.006	0.597 $\pm$.008	0.739 $\pm$.006	0.340 $\pm$.020	3.489 $\pm$.018	1.479 $\pm$.078	0.681 $\pm$.003
		MARDM-SiT (Meng et al., 2025)	0.387 $\pm$.006	0.610 $\pm$.006	0.749 $\pm$.006	0.242 $\pm$.014	3.374 $\pm$.019	1.312 $\pm$.053	0.692 $\pm$.002
		SALAD* (Hong et al., 2025)	0.387 $\pm$.011	0.609 $\pm$.012	0.749 $\pm$.005	0.350 $\pm$.028	3.375 $\pm$.040	-	0.691 $\pm$.001
		LeGO*(Ours)	0.405 $\pm$.005	0.637 $\pm$.005	0.767 $\pm$.005	0.146 $\pm$.008	3.251 $\pm$.016	1.121 $\pm$.050	0.702 $\pm$.001

Table 6: Zero-Shot Evaluation on Essential Motion Representations on HumanML3D and KIT-ML datasets. “*” denotes zero-shot performance.

each adversarial prompt pair, both models are evaluated using identical random seeds and initial noise, so that the measured differences mainly reflect the effect of textual conditioning rather than sampling randomness. We construct six categories of adversarial prompt pairs covering motion-relevant keywords, including *left/right*, *quickly/slowly*, *forward/backward*, *counter-clockwise/clockwise*, *left hand/right hand*, and *run/walk*, as listed in Table 7. Each category contains 5 prompt pairs. We generate 20 motions per prompt, so that each side of a category contributes $5 \times 20 = 100$ motions, forming 100 matched motion pairs for each category.

Common Metrics. We report the results of the six adversarial categories separately in the following sub-experiments. Across these sub-experiments, we use two common metrics, **Diff** and **Success Rate**. **Diff** denotes the difference between the two group means of the measured physical quantity. For example, in the *quickly/slowly* experiment, it is the difference between the mean root speeds of the two sides. **Success Rate** is used for the four categories whose measured physical quantity has a natural zero point, namely *left/right*, *forward/backward*, *counter-clockwise/clockwise*, and *left hand/right hand*. It is defined as the percentage of matched motion pairs in which both sides satisfy the expected sign. For example, in the *left/right* experiment, a matched pair is counted as successful only if the “left” motion has a positive lateral deviation and the “right” motion has a negative lateral deviation. Finally, the Success Rate measures the successful fraction of these 100 pairs. **Pairwise Success Rate** is used for the two speed-related categories, namely *quickly/slowly* and *run/walk*, because their measured quantity does not have a signed correctness criterion. We therefore report the mean speed of each side and the **speed ratio** between them; a larger ratio indicates stronger separation between the two speed-related prompts. For example, in the *quickly/slowly* experiment, a pair is counted as successful if the “quickly” motion is faster than the matched “slowly” motion under the same seed.

In the following, we present the six categories one by one, together with the category-specific metric used to evaluate each keyword pair.

Experiment 1: left vs. right. The example prompt is “A person walks toward left/right”. We measure the signed lateral deviation of the net root displacement with respect to the body-left axis at the first frame, in degrees; positive is leftward and negative is rightward. As shown in the table below, both models get the sign right, but LeGO shows a larger separation between left and right as well as a higher Success Rate.

Model	left	right	Diff	Success Rate
MDM	+50.3°	−15.4°	65.7°	80.5
LeGO	+64.2°	−73.9°	138.1°	90.4

Experiment 2: quickly vs. slowly. The example prompt is “A person walks quickly/slowly.” We measure the average root speed in m/s, discarding frames whose instantaneous root speed is below 0.1 m/s as static. As shown in the table below, according to both Diff and the speed ratio, LeGO clearly shows a larger separation between quickly and slowly, and its quickly and slowly motions differ greatly in speed.

Model	quickly	slowly	Diff	Speed ratio	Pairwise Success Rate
MDM	0.491 m/s	0.439 m/s	0.052	1.12×	49.0
LeGO	1.259 m/s	0.508 m/s	0.751	2.48×	95.0

Experiment 3: forward vs. backward. The example prompt is “a person walks forward / backward”. We measure the per-frame alignment between the root displacement and the body-facing direction, in $[-1, +1]$, where +1 means the subject moves along its facing direction and −1 means opposite to it. As shown in the table below, according to the Diff of both models, both of them distinguish forward from backward well, but LeGO wins decisively on the sample-level accuracy.

Model	forward	backward	Diff	Success Rate
MDM	+0.949	−0.610	1.559	91.6
LeGO	+0.912	−0.848	1.760	100.0

Experiment 4: counter-clockwise vs. clockwise. The example prompt is “a person walks in a circle counter-clockwise / clockwise”. We measure the accumulated signed change of the body-facing direction divided by the duration, i.e., the mean angular velocity in deg/s; positive is counter-clockwise and negative is clockwise. As shown in the table below, MDM fails particularly on the “clockwise” side: its mean angular velocity remains positive under the clockwise prompt, indicating that it still generates counter-clockwise motions when asked to move clockwise. Consequently, its Success Rate is only 55.5%, close to random guessing.

Model	counter-clockwise	clockwise	Diff	Success Rate
MDM	+36.4	+15.5 (×)	20.9	55.5
LeGO	+32.3	−36.2	68.6	97.0

Experiment 5: left hand vs. right hand. The example prompt is “a person is walking with his left / right arm raised”. We measure the height difference between the two wrists in meters; positive means the left hand is higher. As shown in the table below, both models perform very well in this experiment.

Model	left hand	right hand	Diff	Success Rate
MDM	+0.969 m	−1.159 m	2.128 m	100.0
LeGO	+1.071 m	−1.083 m	2.153 m	100.0

Experiment 6: run vs. walk. The example prompt is “a person runs / walks forward”. Unlike Experiment 2, we measure the average speed of the two feet relative to the root, in m/s, because some models learn “running in-place” motions, with active leg movements but little global root displacement. As shown in the table below, both models perform well, but LeGO shows a larger separation between run and walk in the foot speed relative to the root.

Model	run	walk	Diff	Speed ratio	Pairwise Success Rate
MDM	1.536 m/s	0.791 m/s	0.745	1.94×	82.0
LeGO	2.007 m/s	0.850 m/s	1.157	2.36×	92.0

C KEYWORD CONTROL UNDER COMPOSITIONAL TEXT PROMPT

A single keyword can often be resolved by a model that only partially encodes the text, as long as no other constraint competes with it. To test whether keyword understanding survives composition, we start from the base prompt “a person raises his left hand while walking counter-clockwise quickly” and replace exactly one keyword on the B side. We report two metrics. **Success Rate** measures whether the replaced keyword is correctly reflected in the generated motion. **Compositional Success Rate** measures whether the two unchanged constraints are preserved simultaneously. Both metrics are computed over 100 motion pairs.

Experiment 1: replacing the hand. The prompt pair is “a person raises his **left/right** hand while walking counter-clockwise quickly”. We measure the height difference between the two wrists in meters; positive means the left hand is higher. As shown in the table below, both models place the two sides on the correct side of zero on average, but LeGO separates them by a large difference and reaches higher compositional success rate.

Model	left hand	right hand	Diff	Success Rate	Comp. Success Rate
MDM(CLIP)	+0.233 m	−0.329 m	0.562 m	67.5	34.5
LeGO(Ours)	+0.959 m	−1.060 m	2.020 m	100.0	71.0

Experiment 2: replacing the rotation direction. The prompt pair is “a person raises his left hand while walking **counter-clockwise/clockwise** quickly”. We measure the accumulated signed change of the body-facing direction divided by the duration, i.e., the mean angular velocity in deg/s; positive is counter-clockwise and negative is clockwise. As shown in the table below, LeGO produces a clearer separation between the two rotation directions and achieves higher success rates, while MDM(CLIP) shows only a weak response to the replaced rotation keyword.

Model	counter-clockwise	clockwise	Diff	Success Rate	Comp. Success Rate
MDM(CLIP)	+19.7	+15.8 (×)	3.9	47.5	37.5
LeGO(Ours)	+31.30	−12.65	43.95	81.5	65.0

Experiment 3: replacing the speed. The prompt pair is “a person raises his left hand while walking counter-clockwise **quickly/slowly**”. We measure the average root speed in m/s. Since speed is the replaced keyword, the compositional success rate measures whether the hand and rotation constraints are still preserved after changing the speed word. As shown in the table below, LeGO maintains the unchanged constraints while producing a clearer speed separation, whereas MDM(CLIP) is more likely to lose the hand or rotation constraint when the speed keyword is replaced.

Model	quickly	slowly	Diff	Speed ratio	Paired rate	Comp. Success Rate
MDM(CLIP)	0.374 m/s	0.316 m/s	0.058	1.18×	49.0	55.0
LeGO(Ours)	1.024 m/s	0.596 m/s	0.428	1.72×	88.0	93.5

Conclusion and Limitations. LeGO consistently outperforms MDM in compositional text following. Across all three compositional experiments, LeGO-based methods achieve clearly higher compositional accuracy than the MDM baseline, demonstrating a stronger capability to jointly control multiple motion attributes. In addition, compositional text-prompt-based motion generation remains challenging. Although LeGO-based methods improve over MDM, the overall compositional accuracy can be further improved. We believe compositional text-prompt-based motion generation is an important direction for future work.

D EXPERIMENTAL SETTINGS

For the backbone of the motion diffusion model, we use an 8-layer Transformer decoder. The text embeddings extracted by the text encoder are injected into each layer via cross-attention. The

Category	Prompt A/B
left / right	A person walks toward left/right A person walks to the left/right A person turns left/right and keeps walking A man steps to his left/right side A person is walking to the left/right direction
quickly / slowly	A person walks quickly/slowly A person walks forward quickly/slowly A person is walking at a fast/slow pace A man walks very quickly/slowly A person quickly/slowly moves forward
forward / backward	A person faces forward and walks forward/backward A person is facing forward and walks forward/backward A person walks forward/backward while facing forward A person is facing forward and steps forward/backward A person faces forward and walks forward/backward in a straight line
counter-clockwise / clockwise	A person walks in a circle counterclockwise/clockwise A person turns counterclockwise/clockwise A person walks counterclockwise/clockwise A man rotates his body counterclockwise/clockwise A person moves along a counter-clockwise/clockwise path
left hand / right hand	A person walks forward while raising their left/right arm A person walks forward and raises their left/right hand up high A person is walking with his left/right arm raised A person walks while holding their left/right hand up in the air A man walks forward and lifts his left/right arm up
run / walk	A person runs/walks forward The person runs/walks forward A man runs/walks forward A person runs/walks in a straight line The person runs / walks forward slowly

Table 7: Adversarial prompt pairs used in the directional controllability analysis.

maximum diffusion step is set to 50 during both training and inference. The training process is executed on a single NVIDIA 4090 GPU. On the HumanML3D benchmark, we use an initial learning rate of $1e-4$, reduce it by 0.1 at 300,000 iterations, and set the max training iteration to 600,000. Batch size is set to 64. On the KIT-ML benchmark, we use the same learning rate schedule and max training iteration as HumanML3D. Batch size is set to 32. We also use classifier-free guidance (CFG) (Ho & Salimans, 2022) to achieve a better trade-off between motion quality and text alignment. We simply set the CFG weight to 2.5 for both HumanML3D and KIT-ML benchmarks. For finetuning LeGO-CLIP, the LoRA rank is set to 16 across all modules without extensive hyperparameter tuning. Before LeGO training, we pretrain the text and motion projectors on the training split. This pretraining takes only approximately 1.5 hours on a single NVIDIA RTX 4090.

E QUANTITATIVE COMPARISONS ON THE KIT-ML DATASET

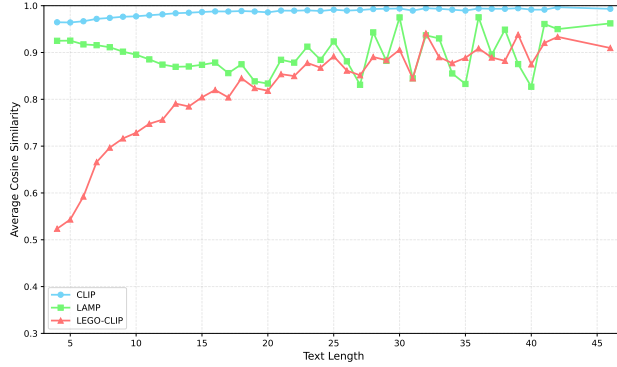
Table 8 presents the quantitative comparisons between our LeGO and previous methods on the KIT-ML dataset (Plappert et al., 2016), following the evaluation protocol proposed by T2M (Guo et al., 2022). LeGO surpasses the existing approaches and achieves state-of-the-art performance in FID, R-Precision, and MM Dist, which is consistent with the observations on HumanML3D.

Table 8: Quantitative Comparisons on the KIT-ML Dataset.

Methods	R Precision $\uparrow$			FID $\downarrow$	MM Dist $\downarrow$	MultiModality $\uparrow$	Diversity $\rightarrow$
	Top 1	Top 2	Top 3				
VAE/VQ	T2M (Guo et al., 2022)	0.361 $\pm$.005	0.559 $\pm$.007	0.681 $\pm$.007	3.022 $\pm$.107	3.488 $\pm$.028	2.052 $\pm$.107
	T2M-GPT (Zhang et al., 2023a)	0.416 $\pm$.006	0.627 $\pm$.006	0.745 $\pm$.006	0.514 $\pm$.029	3.007 $\pm$.023	1.570 $\pm$.039
	MMM (Pinyoanuntapong et al., 2023)	0.404 $\pm$.005	0.621 $\pm$.004	0.744 $\pm$.004	0.316 $\pm$.028	2.977 $\pm$.019	1.232 $\pm$.032
	MoMask (Guo et al., 2024)	0.433 $\pm$.007	0.656 $\pm$.005	0.781 $\pm$.005	0.204 $\pm$.011	2.779 $\pm$.022	1.131 $\pm$.043
	BAMM (Pinyoanuntapong et al., 2024)	0.438 $\pm$.009	0.661 $\pm$.009	0.788 $\pm$.005	0.183 $\pm$.013	2.723 $\pm$.026	1.609 $\pm$.065
	LAMP (Li et al., 2025)	0.479 $\pm$.006	0.691 $\pm$.005	0.826 $\pm$.005	0.141 $\pm$.013	2.704 $\pm$.018	-
	Motion-R1 (Ouyang et al., 2026)	0.431 $\pm$.003	0.638 $\pm$.002	0.761 $\pm$.003	0.287 $\pm$.004	3.196 $\pm$.040	2.262 $\pm$.014
	MoMask+MATE (Han et al., 2026)	0.443 $\pm$.006	0.669 $\pm$.005	0.798 $\pm$.007	0.197 $\pm$.015	2.732 $\pm$.014	1.683 $\pm$.041
Diffusion	MotionDiffuse (Zhang et al., 2024a)	0.417 $\pm$.004	0.621 $\pm$.004	0.739 $\pm$.004	1.954 $\pm$.062	2.958 $\pm$.005	0.730 $\pm$.013
	MDM (Tevet et al., 2023)	0.471 $\pm$.004	0.657 $\pm$.004	0.765 $\pm$.004	0.322 $\pm$.018	2.802 $\pm$.022	-
	MLD (Chen et al., 2023)	0.390 $\pm$.008	0.609 $\pm$.008	0.734 $\pm$.007	0.404 $\pm$.027	3.204 $\pm$.027	2.192 $\pm$.071
	ReMoDiffuse (Zhang et al., 2023b)	0.427 $\pm$.014	0.641 $\pm$.004	0.765 $\pm$.006	0.155 $\pm$.006	2.814 $\pm$.012	1.239 $\pm$.028
	StableMoFusion (Huang et al., 2024a)	0.445 $\pm$.006	0.660 $\pm$.005	0.782 $\pm$.004	0.258 $\pm$.029	-	1.362 $\pm$.062
	SALAD (Hong et al., 2025)	0.477 $\pm$.006	0.711 $\pm$.005	0.828 $\pm$.005	0.296 $\pm$.012	2.585 $\pm$.016	1.004 $\pm$.040
	LeGO(ours)	0.494 $\pm$.005	0.727 $\pm$.005	0.846 $\pm$.005	0.161 $\pm$.010	2.513 $\pm$.018	1.631 $\pm$.062

F RELATIONSHIP BETWEEN TEXT LENGTH AND COSINE SIMILARITY

To further investigate the impact of text length on the encoders' discriminative ability, we plot a line chart in Figure 6 illustrating the relationship between text length (x-axis) and the average cosine similarity of adversarial text pairs (y-axis) across three text encoders: Vanilla CLIP (Radford et al., 2021), LAMP (Li et al., 2025), and LeGO-CLIP. Each data point in the figure represents the mean cosine similarity calculated at a specific text length.



As observed, both Vanilla CLIP and LeGO-CLIP generally exhibit a positive correlation: as the text length increases, the cosine similarity rises overall. This aligns with the expected phenomenon of semantic dilution, where the abundant shared context in longer descriptions inherently overshadows the difference introduced by a single keyword substitution. However, the curve for Vanilla CLIP remains excessively high across all lengths (consistently above 0.95), indicating a severe lack of sensitivity to dynamic keyword alterations. In contrast, the curve for the LAMP encoder fluctuates around 0.9 but lacks a regular trend. For instance, as the text length increases from 4 to 13, its average cosine similarity paradoxically decreases. This irregular behavior suggests that LAMP's embeddings are not systematically sensitive to keyword modifications and risk producing inadequate global semantic representations.

Figure 6: Relationship between text length and cosine similarity.

Most importantly, while LeGO-CLIP also exhibits the length-dependent similarity increase due to semantic dilution, its curve consistently remains well below that of Vanilla CLIP across all lengths. This clearly demonstrates that LeGO-CLIP preserves the fine-grained yet critical dynamic instructions.

Furthermore, to provide a more intuitive understanding of this phenomenon, we present a detailed table (Table 13) that divides similarity intervals from 0.2 to 1.0 (with a step size of 0.1) and showcases representative adversarial text pairs for each interval.

G DATA EFFICIENCY OF GENERATOR TRAINING

We compare the data efficiency of the text-to-motion generator training stage between LeGO and three representative methods (MDM (Tevet et al., 2023), MoMask (Guo et al., 2024), and SALAD (Hong et al., 2025)). For methods with pretrained modules, these modules are trained once on the full HumanML3D training split before low-data generator training. Specifically, MoMask and SALAD pretrain their VQ-VAE/VAE modules on the full training split, while LeGO pretrains its text and motion projectors on the full training split. LeGO's projector pretraining is lightweight, *taking only*

approximately 1.5 hours on a single NVIDIA RTX 4090, and introduces no inference overhead. For LeGO, only the motion generator and the LoRA adapters of LeGO-CLIP are trained on each low-data subset. MDM is a one-stage diffusion model and thus has no separate pretrained module.

For generator training, each method uses 500, 1,000, 2,000, 4,000, 6,000, and 8,000 motion-text pairs, respectively. Their FID and R-Precision are plotted as curves in Figure 7. MDM achieves competitive FID scores with 1,000 and 2,000 training samples, but its overall R-Precision remains relatively low at around 70%. MoMask exhibits steady improvements in both metrics as the training size increases from 500 to 8,000, but it requires more paired samples to learn the text-to-motion mapping effectively. SALAD consistently achieves the second-highest R-Precision, while its FID is less satisfactory when the generator training set is smaller than 2,000 samples. In contrast, under the same low-data generator-training setting, LeGO achieves competitive FID scores and the highest R-Precision across all subset sizes. In particular, even when the generator and LoRA adapters are trained with only 500 motion-text pairs, LeGO attains the best FID and R-Precision simultaneously. These results indicate that the fixed text and motion projectors provide effective generation-oriented semantic supervision, enabling more sample-efficient training of the text-to-motion generator.

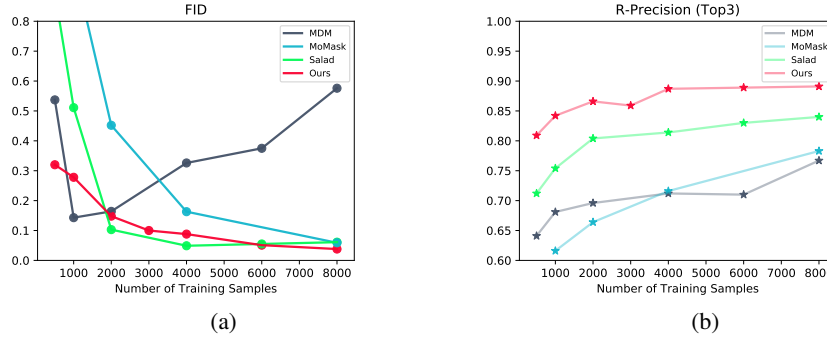


Figure 7: The FID and R-Precision curves under different generator-training subset sizes. For methods with pretrained modules, MoMask/SALAD use full-data pretrained VQ-VAE/VAE modules, while LeGO uses full-data pretrained text/motion projectors; the text-to-motion generators are trained with the corresponding subsets. The entire training sample size of HumanML3D is 23,384.

H ANALYSIS AND MITIGATION OF POTENTIAL EVALUATOR COUPLING

Projector–evaluator relationship on HumanML3D and KIT-ML. We adopt GRU-based projectors mainly for training efficiency, as analyzed in Sec. J. Sharing a broad architecture family, however, does not mean that our projectors are coupled with the official evaluators. First, our projectors are randomly initialized and trained from scratch: they do not share weights, checkpoints, or pretrained parameters with the official evaluators. Second, they are trained only on the training split and never access the validation or test split, so that the consistency supervision introduces no evaluation leakage. Third, they are optimized with an objective different from the distance-based evaluation protocol. Instead of forcing paired text and motion embeddings to be as close as possible in a retrieval space, we provide high-level semantic supervision while allowing text and motion embeddings to preserve their own modality-specific information, as detailed in Sec. O.

Different projector architectures yield consistent improvements. To verify that the improvement of LeGO is not merely caused by evaluator coupling, we replace our projectors with two alternatives: the official evaluators and TMR (Petrovich et al., 2023), a representative text-motion retrieval model. As shown in Table 9, using the official evaluators as projectors leads to clearly worse performance, especially in FID. Using TMR as projectors still achieves strong results, but it remains slightly worse than our projectors and is substantially slower to train, as analyzed in Appendix J. These results indicate that LeGO benefits from semantic consistency supervision rather than directly optimizing the evaluation space.

Table 9: Comparison of different projectors on HumanML3D.

Projectors	R@Top1 $\uparrow$	R@Top2 $\uparrow$	R@Top3 $\uparrow$	FID $\downarrow$	MM Dist $\downarrow$
TMR	0.620 $\pm$.002	0.808 $\pm$.002	0.879 $\pm$.002	0.045 $\pm$.002	2.390 $\pm$.011
Our projectors	0.627$\pm$.004	0.811$\pm$.003	0.888$\pm$.002	0.039$\pm$.001	2.382$\pm$.010
Official evaluators	0.611 $\pm$.003	0.803 $\pm$.002	0.871 $\pm$.001	0.103 $\pm$.003	2.407 $\pm$.011

Projector-evaluator difference on SnapMoGen. The SnapMoGen setting further reduces the concern of evaluator coupling. Our training-time projectors differ from the official SnapMoGen evaluation-time models in two key aspects: (1) **Input motion representation.** The official SnapMoGen evaluators operate on the first 148 dimensions of the motion representation, which mainly encode root and joint rotation information in the embedding space. In contrast, when training our projectors and computing the text consistency and motion consistency constraints, we use the full 292-dimensional motion representation (excluding the foot contact label), following the same design choice as on HumanML3D and KIT-ML, because our goal is to learn the overall similarity of motions rather than the sole part involved in the evaluation protocol. Therefore, the raw motion contents used by the official evaluators and our projectors are different. (2) **Network architecture.** The official SnapMoGen evaluators are Transformer-based, whereas we still use GRU-based projectors to keep training efficient. Thus, in SnapMoGen, our projectors differ from the official evaluators in both input motion representation and network architecture.

Design insight of consistency constraints. The two constraints are designed as generation-oriented semantic supervision rather than as a direct optimization of evaluation metrics, as detailed in Sec. O: the motion consistency constraint encourages the generated motion to preserve the high-level semantics of the ground-truth motion instead of strictly matching every joint coordinate frame by frame, and the text consistency constraint encourages the generated motion to remain semantically aligned with the input text while allowing multiple valid motions to correspond to the same textual description.

Plug-and-play evidence of LeGO-CLIP. LeGO-CLIP can be directly plugged into three different motion generators and consistently improves them under unchanged training settings; under the same MoMask generator it also outperforms existing pretrained text encoders such as MoCLIP and MATE. We refer to Sec. 4.2 and Table 3 for the quantitative results. These plug-and-play results suggest that LeGO-CLIP provides generally useful conditioning representations for motion generation, rather than representations overfitted to a specific evaluator.

Input-side and output-side evidence. Beyond evaluator-based metrics, LeGO-CLIP shows stronger sensitivity to motion-critical keyword substitutions in the input space (Sec. 4.3), and single-keyword replacement (Appendix B) and compositional prompt experiments (Appendix C) further show that LeGO-based models generate motions that better follow the modified textual attributes. These are measured by physical properties of generated motions, such as moving direction, speed, angular velocity, and hand height, rather than by learned evaluator embeddings, supporting the effectiveness of LeGO-CLIP beyond evaluator-based metrics.

Zero-shot evaluation on the essential 67-dimensional benchmark. As reported in Sec. A, our model trained with the standard 263-dimensional HumanML3D representation still achieves the best zero-shot performance on the 67-dimensional essential-motion benchmark without any additional finetuning. This indicates that the learned representation and generator are not merely overfitted to the standard 263-dimensional evaluation protocol, and can generalize to a different motion representation and evaluation setting.

Summary. Overall, the potential evaluator-coupling concern is mitigated from multiple perspectives: our projectors differ from the official evaluators in supervision objective and training data, and, on SnapMoGen, in both input representation and network architecture; replacing our projectors with official evaluators performs worse, while using TMR still validates the generality of our consistency-supervision design; and evidence beyond the standard evaluator, including plug-and-play transfer (Sec. 4.2), input-side sensitivity analysis (Sec. 4.3), output-side physical evaluation (Sec. B),

compositional prompt tests (Sec. C), and zero-shot evaluation on the 67-dimensional benchmark (Sec. A), all supports that LeGO improves motion generation beyond evaluator-specific alignment.

I THE RETRIEVAL PERFORMANCE OF PROPOSED TEXT PROJECTOR AND MOTION PROJECTOR.

To illustrate the effectiveness of our text and motion projectors, we evaluate them on the text-to-motion and motion-to-text retrieval tasks. We adopted the commonly used metrics Recall (denoted as $R@k$ for brevity) to compare, which include $R@1,2,3,5,10$. Higher Recall values indicate better performance. The retrievals are conducted on the test sets of the HumanML3D and KIT-ML benchmarks. Our results for both text-motion and motion-text retrieval are summarized in Table 10.

Methods	Text-Motion Retrieval (%) $\uparrow$					Motion-Text Retrieval (%) $\uparrow$				
	$R@1 \uparrow$	$R@2 \uparrow$	$R@3 \uparrow$	$R@5 \uparrow$	$R@10 \uparrow$	$R@1 \uparrow$	$R@2 \uparrow$	$R@3 \uparrow$	$R@5 \uparrow$	$R@10 \uparrow$
HumanML3D	TEMOS (Petrovich et al., 2022)	42.40	53.52	61.14	70.96	84.15	39.96	53.49	61.79	72.40
	T2M (Guo et al., 2022)	52.48	71.05	80.65	89.66	96.58	52.00	71.21	81.11	89.87
	TMR (Petrovich et al., 2023)	67.16	81.32	86.81	91.43	95.36	67.97	81.20	86.35	91.70
	LAMP (Li et al., 2025)	67.18 ± 0.5	81.9 ± 0.4	87.04 ± 0.3	92.0 ± 0.2	95.73 ± 0.2	68.02 ± 0.3	82.1 ± 0.3	87.5 ± 0.3	92.2 ± 0.3
	Ours	69.15 ± 0.04	84.43 ± 0.02	90.21 ± 0.02	94.89 ± 0.02	98.35 ± 0.01	69.0 ± 0.03	84.77 ± 0.02	90.52 ± 0.03	95.26 ± 0.02
KIT-ML	TEMOS (Petrovich et al., 2022)	43.88	58.25	67.00	74.00	84.75	41.88	55.88	65.62	75.25
	T2M (Guo et al., 2022)	42.25	62.62	75.12	87.50	96.12	39.75	62.75	73.62	86.88
	TMR (Petrovich et al., 2023)	49.25	69.75	78.25	87.88	95.00	50.12	67.12	76.88	88.88
	LAMP (Li et al., 2025)	52.5 ± 0.7	74.8 ± 0.5	84.7 ± 0.5	92.7 ± 0.3	97.6 ± 0.3	54.0 ± 0.005	75.3 ± 0.5	84.4 ± 0.4	92.2 ± 0.2
	Ours	51.33 ± 0.06	73.13 ± 0.04	83.62 ± 0.06	92.96 ± 0.06	98.04 ± 0.02	52.54 ± 0.010	74.16 ± 0.005	84.73 ± 0.006	93.57 ± 0.04

Table 10: Text-motion (**left**) and motion-text (**right**) retrieval benchmark on the HumanML3D and KIT-ML.

J WHY DO WE USE GRU-BASED PROJECTORS?

We further clarify our choice of using GRU-based projectors in LeGO. A natural alternative is to adopt a Transformer-based design, such as the projector architecture used in TMR, which demonstrates convincing retrieval performances. In our work, we intentionally choose GRU-based projectors because they provide a substantially more lightweight solution while remaining sufficient for bridging the text–motion representation gap.

Table 11 compares the training efficiency and memory cost of different designs. Relative to the vanilla MDM baseline, our method with GRU-based projectors reduces the training throughput from 9.80 to 5.85 iterations per second and increases the GPU memory usage from 5.6G to 9.5G under batch size 64. In contrast, replacing the projectors with TMR further reduces the throughput to 3.01 iterations per second and increases the memory usage to 15G. Therefore, although both variants introduce additional overhead beyond the base generator, the GRU-based design is considerably more efficient than the Transformer-based TMR in both speed and memory.

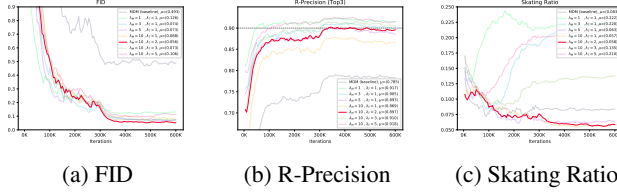
Method	Training Speed (iter/s) $\uparrow$	GPU Memory (G) $\downarrow$
MDM	9.80	5.6
Ours (GRU Projector)	5.85	9.5
Ours (TMR Projector)	3.01	15.0

Table 11: Efficiency comparison of different projector architectures. We report training throughput (iterations per second) and GPU memory usage under batch size 64.

K ABLATIONS ON HYPERPARAMETERS OF CONSISTENCY CONSTRAINTS

LeGO involves only two hyperparameters, λ_M and λ_T . We empirically choose different combinations of λ_M and λ_T , with λ_M ranging from 1 to 10 and λ_T from 1 to 5. We train several experiments from scratch with the same training settings introduced in Sec. D. We evaluate each model on the

HumanML3D validation set, and the variation curves of FID, R-precision, and Skating ratio are shown in Figure 8, where the mean of each metric after training becomes stable is reported top right in the legend. The quantitative results of FID, R-precision, and Skating ratio at the point where FID is the lowest are reported in Table 12. It is observed that all the combinations outperform the baseline MDM by a large margin, further suggesting the importance of allowing the text encoder to learn optimal text embeddings autonomously. As λ_M increases from 1 to 10, the overall FIDs and Skating ratio become better. Increasing the value of λ_T can lead to better text-motion matching precision, while the FID and Skating ratio become worse. In summary, λ_M determines the motion quality and realism, and λ_T determines the textual alignment of motion. Since we consider motion quality to be the most important (FIDs and Skating ratio), we choose $\lambda_M = 10$ and $\lambda_T = 2$ as our default setting (the red line in Figure 8).



Method	FID ↓	R-precision ↑ (Top-3)	Skating Ratio ↓
MDM (baseline)	0.357	0.712	0.096
$\lambda_M = 1, \lambda_T = 1$	0.099	0.917	0.225
$\lambda_M = 3, \lambda_T = 1$	0.047	0.905	0.227
$\lambda_M = 5, \lambda_T = 1$	0.048	0.893	0.064
$\lambda_M = 10, \lambda_T = 1$	0.052	0.867	0.055
$\lambda_M = 10, \lambda_T = 2$	0.039	0.888	0.062
$\lambda_M = 10, \lambda_T = 3$	0.054	0.905	0.135
$\lambda_M = 10, \lambda_T = 5$	0.081	0.927	0.208

Figure 8: Ablation studies of the hyperparameter λ_M and λ_T . The curves of FID, R-Precision, and Skating Ratio during training. μ denotes the mean value after 400K iterations. The red line denotes our default setting.

Table 12: Ablation studies of the hyperparameter λ_M and λ_T .

L USER STUDY

We further conduct a user study to compare the realism and semantic alignment of our method with previous methods: MDM (Tevet et al., 2023), MoMask (Guo et al., 2024), and SALAD (Hong et al., 2025). We randomly sample 20 texts for sampling. 25 participants were asked to rank videos generated by four methods based on their realism and semantic alignment. The ranking was collected through the web-based interface shown in Figure 10, where each participant watched the four videos generated from the same text description and ranked them independently according to realism and semantic alignment. Results in Figure 9 show that our method clearly outperforms others in both realism and semantic alignment.

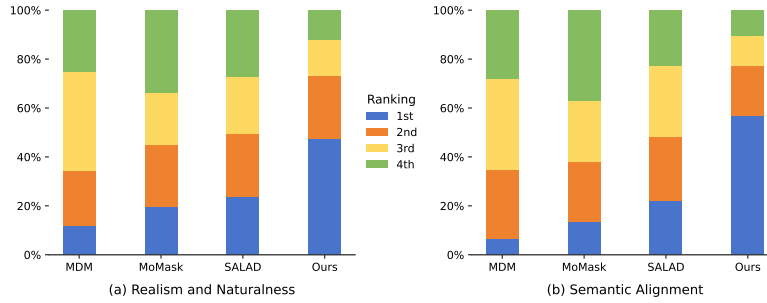


Figure 9: User study comparing our method with MDM, MoMask, and SALAD.

M THE PERFORMANCE WITH EXTREMELY FEW TRAINING SAMPLES

In this section, we evaluate LeGO and MDM under a limited-data training setting. We still use the whole test set for evaluation, including 4383 samples. For the training scheduler, we train the networks with a maximum iteration of 200K. The initial learning rate is $1e-4$ and degrades by 0.1 at 100K iterations.

Performances comparison under 10% training samples. Here we plot the FIDs and R-Precision when using only 2000 training samples (approximately 10%) in Figure 11. We observe that MoMask

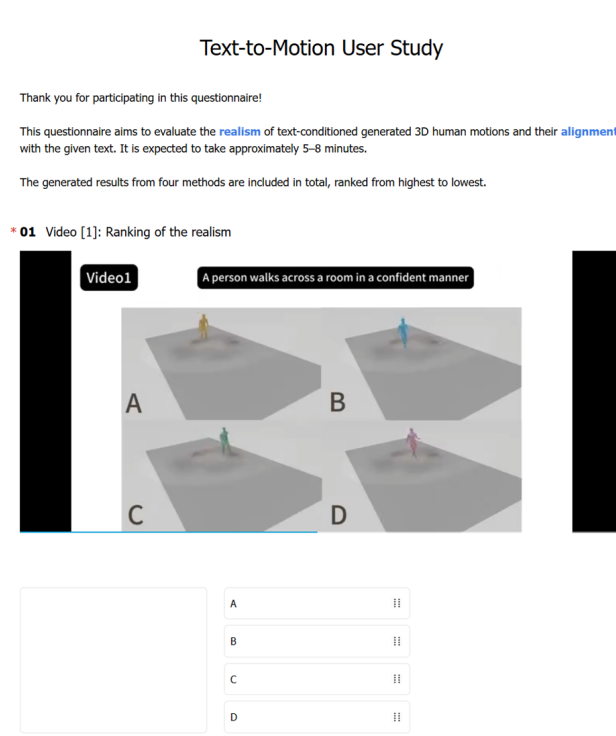


Figure 10: The user study interface. For each text description, the four videos generated by different methods are presented anonymously as A, B, C, and D, and the participant ranks them from the highest to the lowest according to realism and semantic alignment.

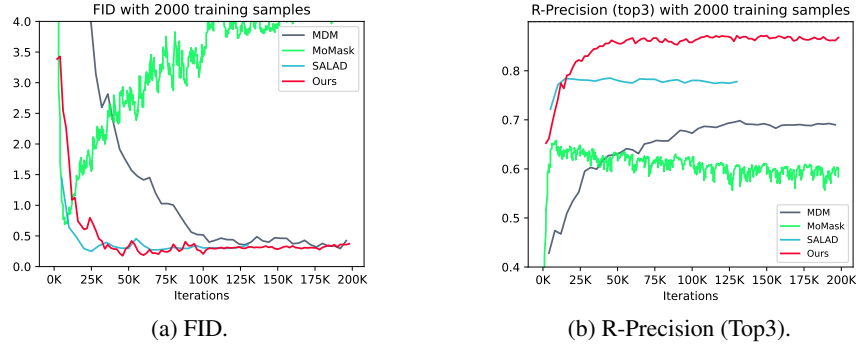


Figure 11: The FID and R-Precision curves of MDM (Tevet et al., 2023), MoMask (Guo et al., 2024), SALAD (Hong et al., 2025), and our method, trained with 2000 training samples (approximately 10%). The size of the training set of HumanML3D is 23,384.

exhibits severe overfitting as training proceeds, whereas the three diffusion-based methods (MDM, SALAD, and our LeGO) remain stable. We attribute this to the discrete tokenization used in VQ-VAE models, which limits representation flexibility and magnifies the impact of data scarcity. When samples are few, the pretrained codebook is insufficiently covered and the model quickly memorizes the training data. By contrast, diffusion-based methods model motions in a continuous space with noise perturbation and iterative denoising, which naturally act as a form of regularization and make them more robust under limited data.

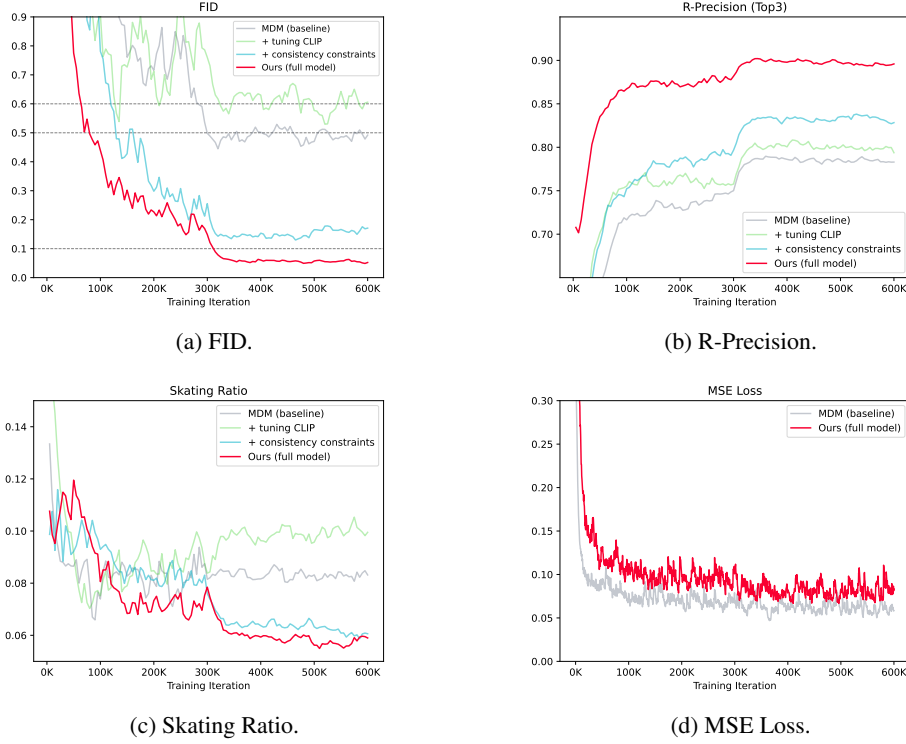


Figure 12: Evolution of metrics during training.

N EVOLUTION OF METRICS DURING TRAINING.

To provide a more comprehensive understanding of our proposed components, the evolution of FID, R-Precision, Skating Ratio, and MSE loss during training is shown in Figure 12. Each component improves performance by a clear and stable margin rather than fluctuation.

O INTUITIVE INSIGHT INTO CONSISTENCY CONSTRAINTS

Element-wise MSE loss biases the model toward memorizing each joint frame-by-frame, while neglecting higher-level semantic patterns in motion sequences. For example, when learning a walking motion, minimizing MSE encourages the network to reproduce the exact positions of each joint at every frame, but a walking motion should still be recognized as such even with slight variations in body joints.

The insight into Motion Consistency Constraint. We believe it is not optimal to judge the closeness between GT and predicted motion solely using MSE during training. Even though several joints in motion slightly deviate from the GT at some frames, as long as the overall motion remains visually similar to GT, then this motion should not be assigned a significant loss value. We should provide the network with some degree of freedom to search for similar motions to GT. Thus, we consider judging the similarity in the embedding space.

The insight into Text Consistency Constraint. The text conditions are coarse-grained, and there is no strict one-to-one correspondence between text and GT motion. Many possible motions can match the text, rather than just the GT. Therefore, we consider allowing the network to learn the consistency within a high-level text and motion embedding space.

P EVALUATION METRICS

We adopt the metrics proposed by T2M (Guo et al., 2022) to evaluate the quantitative performance of our method: (1) *Fréchet Inception Distance (FID)* evaluates the quality of generated motions by measuring the distributional difference between the generated motions and real motions. (2) *R-Precision* and *Multimodal Distance* evaluate the semantic alignment between input text and generated motions. *R-Precision* is the motion-retrieval precision between texts and generated motions, and the *Multimodal Distance* calculates the distance between motions and texts. (3) *Diversity* indicates the variance in generated motions, and *MultiModality*, which assesses the diversity of motions generated from the same text description, are also adopted as secondary metrics. They are considered less important than the generation quality and alignment with the input text. We emphasize that FID and R-precision are the primary metrics, whereas Diversity and MultiModality are secondary metrics to be assessed in conjunction with the primary metrics (Guo et al., 2024; Hong et al., 2025). (4) *Foot skating ratio* is a proxy for the physical plausibility of motions, following (Karunratanakul et al., 2023; Xie et al., 2024). It measures the proportion of frames in which either foot skids more than a certain distance (2.5 cm) while maintaining contact with the ground (foot height < 5 cm).

Similarity Interval	Cosine Similarity			Keyword	Text
	CLIP	LAMP	LeGO-CLIP		
[0.2,0.3)	0.948	0.940	0.281	forward	a person walks forward and stops
	0.970	0.879	0.255	backward	a man walks backwards for a few steps.
	0.968	0.951	0.299	backward	walking backwards and stopping.
	0.971	0.452	0.295	forward	a person walks forward , then turns to the right, then continues to walk forward .
[0.3,0.4)	0.977	0.889	0.376	forward	a man takes 5 steps in a forward direction
	0.988	0.762	0.358	slowly	slowly swinging the arms forward as the body walks slowly .
	0.938	0.956	0.379	walks	a person walks forward then walks up something.
	0.974	0.946	0.398	slowly	a person walking slowly forward
[0.4,0.5)	0.944	0.921	0.439	walking	a person walking slowly forward with confidence
	0.965	0.922	0.471	running	person is running in place and then stands still
	0.938	0.785	0.499	forward	a person walks forward with his arms at his side slowly.
	0.940	0.947	0.473	walks	a person walks steadily straight ahead.
[0.5,0.6)	0.979	0.789	0.514	forward	the person walks forward with their right hand raised, then sways with both hands raised.
	0.976	0.655	0.598	forward	a person takes two steps forward , then walks sideways three steps, then walks forward diagonally and to the left three steps.
	0.990	0.933	0.570	slowly	a figure walks forward slowly
	0.945	0.794	0.545	walks	a person walks while touching something with his left hand.
[0.6,0.7)	0.984	0.767	0.609	runs	someone runs forward quickly, swinging their arms back and forth in front of them.
	0.984	0.810	0.616	forward	the man walks forward then turns and returns back
	0.975	0.862	0.648	forward	a person walks b then steps onto an object before stepping back down again, steps forward some more, before turning around and going back again.
	0.985	0.887	0.683	backward	a person takes a quick step backwards and to their right.
[0.7,0.8)	0.985	0.700	0.710	walking	a person bounces while walking forward, stops, holding something in his right hand lifts it up to his face and then lowers it
	0.995	1.000	0.767	left	a person throws an object with his left hand, then holds both hands up over his head
	0.988	0.942	0.788	forward	a person walks forward , bends down to pick something up off the ground, and walks right to place the object down in a different location.
	0.986	1.000	0.799	left	a person throws with the left arm.
[0.8,0.9)	0.989	0.889	0.808	forward	person walks forward , bends down, grabs two beverages, and pours them into one container.
	0.997	1.000	0.871	right	a person swaying his/her leg a certain motion and raising their right arm
	0.998	0.989	0.894	slowly	a man stands with his legs parted and slowly rotates his upper torso a few times, then starts to rotate his hips.
	0.987	0.998	0.895	left	a man puts his left hand to his right shoulder and then his right hand to his left shoulder and stretches his shoulder.

Table 13: Samples of text descriptions in each similarity interval of LeGO-CLIP. The selection of samples in each similarity interval is based on LeGO-CLIP’s similarity.