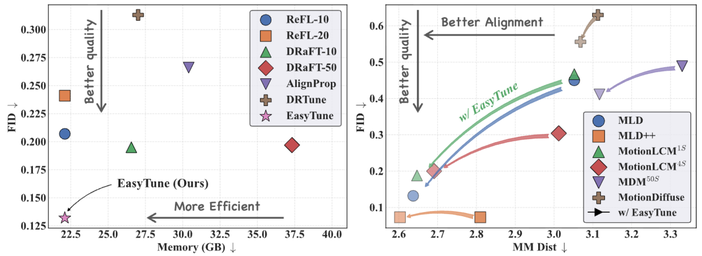
EasyTune: Efficient Step-Aware Fine-Tuning for Diffusion-Based Motion Generation

Xiaofeng Tan[1,2 *], Wanjiang Weng[1,2 *], Haodong Lei[1,2], Hongsong Wang[1,2 †]

[1] Southeast University, [2] Key Laboratory of New Generation Artificial Intelligence Technology [†] Corresponding Author [*] Equal Contributions

Introduction

Issue: Text-to-motion diffusion models trained on likelihood objectives misalign with downstream goals. While differentiable reward methods enable efficient preference fine-tuning without large-scale paired data, existing approaches suffer from (1) coarse-grained trajectory-level optimization, (2) O(T) memory overhead, and (3) vanishing gradients hindering early-step learning.



Key Observation: We theoretically and empirically identify that the root cause of these limitations is the recursive dependence between different steps in the denoising trajectory, leading to (1) O(T) memory complexity due to the entire computation graph, (2) delayed, sparse optimization signals, and (3) vanishing gradients.

TL;DR: We propose EasyTune, a reinforcement fine-tuning framework for diffusion models that decouples recursive dependencies and enables (1) dense and effective optimization, (2) memory-efficient training, and (3) fine-grained alignment.

🔥 ***Extension:*** We have released ***MotionRFT***, a unified reinforcement fine-tuning for text-to-motion generation, with comprehensive code and documentation.

Motivation: Rethink differentiable reward

Recursive Dependence

Given a pre-trained motion diffusion model parameterized by ϵ_θ, the optimization objective is to fine-tune θ to maximize the reward value $\mathcal{R}_\phi(\mathbf{x}_0^\theta, c)$, with the loss defined as:

$$\mathcal{L}(\theta) = -\mathbb{E}_{c\sim\mathcal{D}_\mathrm{T}, \mathbf{x}_0^\theta\sim\pi_\theta(\cdot|c)}\left[\mathcal{R}_\phi(\mathbf{x}_0^\theta, c)\right], \tag{1}$$

To compute the full gradient $\partial\mathcal{L}(\theta)/\partial\theta$, we unroll the $\partial\mathbf{x}_0^\theta/\partial\theta$ using Corollary 1 and substitute it into Eq. (3) resulting in (see proof in App. C.3):

$$\frac{\partial\mathcal{L}(\theta)}{\partial\theta} = -\mathbb{E}_{c\sim\mathcal{D}_\mathrm{T}, \mathbf{x}_0^\theta\sim\pi_\theta(\cdot|c)}\left[\frac{\partial\mathcal{R}_\phi(\mathbf{x}_0^\theta)}{\partial\mathbf{x}_0^\theta}\cdot\sum_{t=1}^{T}\underbrace{\left(\prod_{s=1}^{t-1}\frac{\partial\pi_\theta(\mathbf{x}_s^\theta, s, c)}{\partial\mathbf{x}_s^\theta}\right)}_{\text{tend to 0 when t is larger}}\underbrace{\left(\frac{\partial\pi_\theta(\mathbf{x}_t^\theta, t, c)}{\partial\theta}\right)}_{\text{optimizing t-th step}}\right]. \tag{5}$$

Memory Complexity, Vanishing Gradients

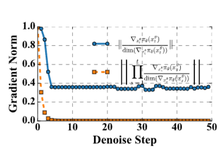


Fig. 4. Gradient norm with respect to denoising steps. Here, dim(·) denotes the gradient dimension.

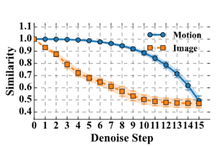


Fig. 5. Similarity between t-th step noised and clean motion.

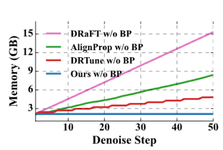
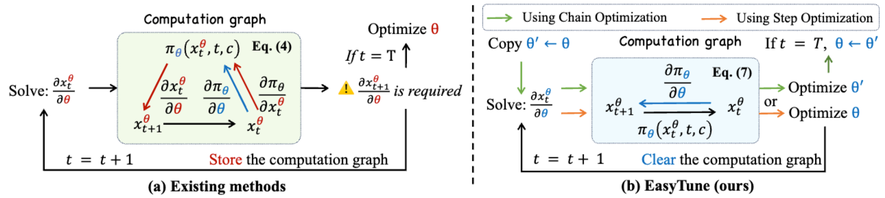


Fig. 6. Memory usage comparison. Here, "w/o BP" indicates memory measured without back-propagation.

Framework & Solution

Efficient Fine-Tune Method (EasyTune)



Step-aware Reward Model & Self-Preference Learning

Reward Model. Given a motion $\mathbf{x}$ and a text description c, the reward value is computed based on the similarity between the motion features $\mathbf{x}$ and text features c, denoted as:

$$\mathcal{R}_\phi(\mathbf{x}, c) = \mathcal{E}_\mathrm{M}(\mathbf{x})\cdot\mathcal{E}_\mathrm{T}(c)\cdot\tau, \tag{11}$$

where $\mathcal{E}_\mathrm{M}$ and $\mathcal{E}_\mathrm{T}$ are the motion and text encoders from the pre-trained retrieval model (Weng et al., 2025a), and τ is a trainable temperature parameter.

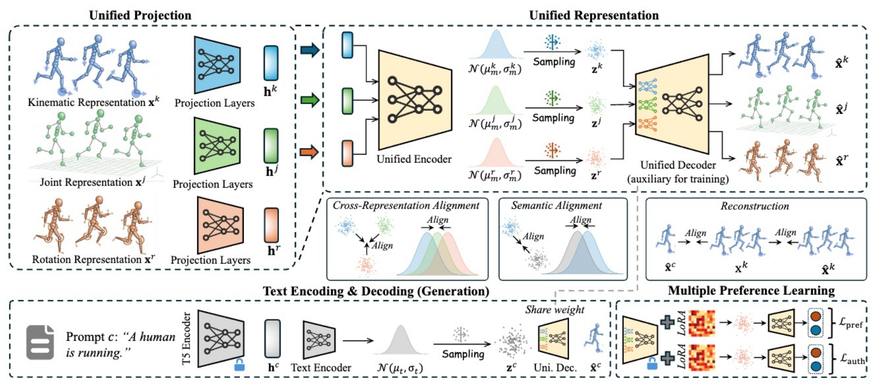
$$\mathcal{R}_\phi(\mathbf{x}_t, t, c) = \begin{cases}\mathcal{R}_\phi(\hat{\mathbf{x}}_0, 0, c), & \text{Only for ODE-based settings,}\\ \mathcal{R}_\phi(\mathbf{x}_t, t, c), & \text{For SDE- and ODE-based settings.}\end{cases} \tag{12}$$

To optimize the reward model ϕ, we minimize the KL divergence between them:

$$\mathcal{L}_\mathrm{SPL}(\phi) = \mathrm{D}_\mathrm{KL}(\mathcal{Q}\,\|\,\mathcal{P}) = \sum_{\mathbf{x}\in\{\mathbf{x}^\mathrm{w}, \mathbf{x}^\mathrm{l}\}}\mathcal{Q}(\mathbf{x}, c)\log\frac{\mathcal{Q}(\mathbf{x}, c)}{\mathcal{P}(\mathbf{x}, c)}. \tag{17}$$

MotionReward (Extension)

Unified Motion Reward Model for Multi-Representation & Multi-Reward



Contact

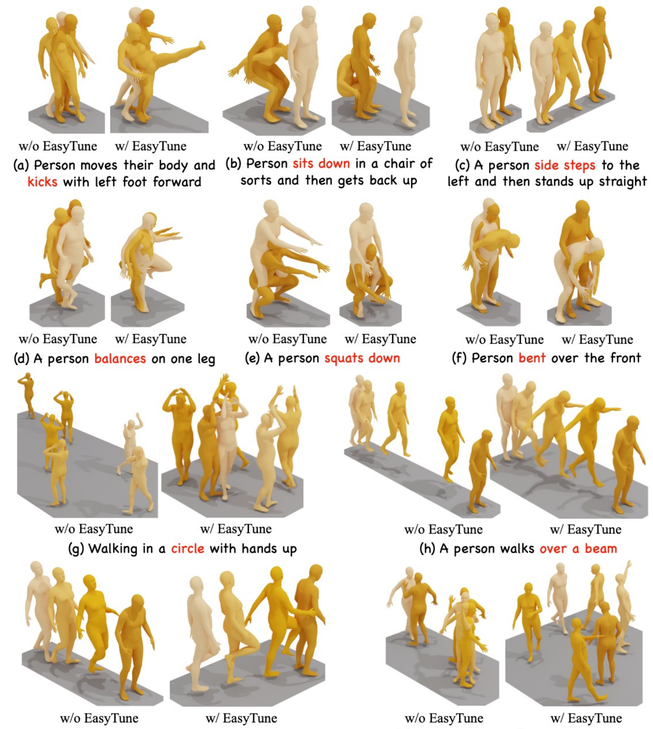
Email: txf0620@gmail.com
WeChat: txf_06_20

🔥 The first author is seeking Ph.D opportunity in 2027 Spring/Fall

First Author | Project Page (Extension) | Code | Project Page (This work)

Experiments

Qualitative Results



Quantitative Results

Method	R Precision ↑			FID ↓	MM Dist ↓	Diversity →	Memory (GB) ↓
	Top 1	Top 2	Top 3				
Real	0.511	0.703	0.797	0.002	2.974	9.503	-
MLD (Base Model)	$0.504^{\pm.002}$	$0.698^{\pm.003}$	$0.796^{\pm.002}$	$0.450^{\pm.011}$	$3.052^{\pm.009}$	$9.634^{\pm.064}$	15.21
w/ ReFL-10 (Clark et al., 2024)	$0.533_{+5.8\%}$	$0.720_{+3.2\%}$	$0.821_{+3.1\%}$	$0.207_{+54.0\%}$	$2.852_{-6.6\%}$	$10.129_{-0.495}$	**22.10**$_{+6.89}$
w/ ReFL-20 (Clark et al., 2024)	$0.528_{+4.8\%}$	$0.718_{+2.9\%}$	$0.815_{+2.1\%}$	$0.241_{+46.4\%}$	$2.883_{-5.5\%}$	$10.189_{-0.555}$	**22.10**$_{+6.89}$
w/ DRaFT-10 (Clark et al., 2024)	$0.565_{+12.1\%}$	$0.757_{+8.5\%}$	$0.846_{+6.3\%}$	$0.195_{+56.7\%}$	$2.703_{-11.4\%}$	$9.851_{-0.217}$	$26.56_{+11.35}$
w/ DRaFT-50 (Clark et al., 2024)	$0.528_{+4.8\%}$	$0.724_{+3.7\%}$	$0.819_{+2.9\%}$	$0.197_{+56.2\%}$	$2.872_{-5.9\%}$	$9.641_{-0.007}$	$37.32_{+22.11}$
w/ AlignProp (Prabhudesai et al., 2023)	$0.560_{+11.1\%}$	$0.753_{+7.9\%}$	$0.841_{+5.7\%}$	$0.266_{+40.9\%}$	$2.739_{-10.3\%}$	$9.877_{-0.243}$	$30.40_{+15.19}$
w/ DRTune (Wu et al., 2025a)	$0.549_{+8.9\%}$	$0.746_{+6.9\%}$	$0.836_{+5.0\%}$	$0.313_{+30.4\%}$	$2.795_{-8.4\%}$	$9.930_{-0.296}$	$27.01_{+11.80}$
w/ EasyTune (Ours, Step Optimization)	**0.581**$_{+15.3\%}$	**0.769**$_{+10.2\%}$	**0.855**$_{+7.4\%}$	**0.132**$_{+70.7\%}$	$2.637_{+13.6\%}$	**9.465**$_{+0.093}$	**22.10**$_{+6.89}$
w/ EasyTune (Ours, Chain Optimization)	$0.574_{+13.9\%}$	$0.766_{+9.7\%}$	$0.854_{+7.3\%}$	$0.172_{+61.8\%}$	**2.614**$_{+14.3\%}$	$9.348_{-0.024}$	$24.21_{+9.00}$

Method	R Precision ↑			FID ↓	MM Dist ↓	Diversity →
	Top 1	Top 2	Top 3			
Real	0.511	0.703	0.797	0.002	2.974	9.503
TM2T (Guo et al., 2022b)	$0.424^{\pm0.003}$	$0.618^{\pm0.003}$	$0.729^{\pm0.002}$	$1.501^{\pm0.017}$	$3.467^{\pm0.011}$	$8.589^{\pm0.076}$
T2M (Guo et al., 2022a)	$0.455^{\pm0.002}$	$0.636^{\pm0.003}$	$0.736^{\pm0.003}$	$1.087^{\pm0.002}$	$3.347^{\pm0.008}$	$9.175^{\pm0.002}$
MDM (Tevet et al., 2023)	$0.455^{\pm0.006}$	$0.645^{\pm0.007}$	$0.749^{\pm0.006}$	$0.489^{\pm0.047}$	$3.330^{\pm0.025}$	$9.920^{\pm0.083}$
T2M-GPT (Zhang et al., 2023a)	$0.492^{\pm0.003}$	$0.679^{\pm0.002}$	$0.775^{\pm0.002}$	$0.141^{\pm0.005}$	$3.121^{\pm0.009}$	$9.722^{\pm0.082}$
ReMoDiffuse (Zhang et al., 2023b)	$0.510^{\pm0.005}$	$0.698^{\pm0.006}$	$0.795^{\pm0.004}$	$0.103^{\pm0.004}$	$2.974^{\pm0.016}$	$9.018^{\pm0.075}$
AttT2M (Zhong et al., 2023)	$0.499^{\pm0.003}$	$0.690^{\pm0.002}$	$0.786^{\pm0.002}$	$0.112^{\pm0.006}$	$3.038^{\pm0.007}$	$9.700^{\pm0.090}$
MotionDiffuse (Zhang et al., 2024a)	$0.491^{\pm0.001}$	$0.681^{\pm0.001}$	$0.775^{\pm0.001}$	$0.630^{\pm0.001}$	$3.113^{\pm0.001}$	$9.410^{\pm0.049}$
MotionLCM (Dai et al., 2024)	$0.502^{\pm0.003}$	$0.698^{\pm0.002}$	$0.798^{\pm0.002}$	$0.304^{\pm0.012}$	$3.012^{\pm0.007}$	$9.607^{\pm0.066}$
MotionMamba (Zhang et al., 2024b)	$0.502^{\pm0.003}$	$0.693^{\pm0.002}$	$0.792^{\pm0.002}$	$0.281^{\pm0.011}$	$3.060^{\pm0.000}$	$9.871^{\pm0.084}$
CoMo (Huang et al., 2024)	$0.502^{\pm0.002}$	$0.692^{\pm0.007}$	$0.790^{\pm0.002}$	$0.262^{\pm0.004}$	$3.032^{\pm0.015}$	$9.936^{\pm0.066}$
ParCo (Zou et al., 2024)	$0.515^{\pm0.003}$	$0.706^{\pm0.003}$	$0.801^{\pm0.002}$	$0.109^{\pm0.005}$	$2.927^{\pm0.008}$	$9.576^{\pm0.088}$
SoPo (Tan et al., 2025)	$0.528^{\pm0.005}$	$0.722^{\pm0.004}$	$0.827^{\pm0.004}$	$0.174^{\pm0.005}$	$2.939^{\pm0.011}$	$9.584^{\pm0.074}$
MLD (Chen et al., 2023) (Base Model)	$0.504^{\pm0.002}$	$0.698^{\pm0.003}$	$0.796^{\pm0.002}$	$0.450^{\pm0.011}$	$3.052^{\pm0.009}$	$9.634^{\pm0.064}$
w/ EasyTune (Ours)	$0.581^{\pm0.003}_{+15.3\%}$	$0.769^{\pm0.002}_{+10.2\%}$	$0.855^{\pm0.002}_{+7.4\%}$	$0.132^{\pm0.005}_{+70.7\%}$	$2.637^{\pm0.007}_{+13.6\%}$	$\mathbf{9.465}^{\pm0.075}_{+0.09}$
MLD++ (Dai et al., 2025) (Base Model)	$0.548^{\pm0.004}$	$0.738^{\pm0.003}$	$0.829^{\pm0.002}$	$0.073^{\pm0.003}$	$2.810^{\pm0.008}$	$9.658^{\pm0.089}$
w/ EasyTune (Ours)	$\mathbf{0.591}^{\pm0.004}_{+7.8\%}$	$\mathbf{0.777}^{\pm0.002}_{+5.3\%}$	$\mathbf{0.859}^{\pm0.002}_{+3.6\%}$	$\mathbf{0.069}^{\pm0.003}_{+6.8\%}$	$\mathbf{2.592}^{\pm0.008}_{+7.8\%}$	$9.705^{\pm0.086}_{-0.06}$