MoGeFi: Learning Geometric Feedback for Few-Step Human Motion Generation

Anonymous authors
Paper under double-blind review

Abstract

Interactive human motion generation calls for natural, text-aligned movement, accurate geometric control, and low latency. Precise control often relies on costly test-time refinement, while fast generators can leave substantial joint errors. We present MoGeFi, a framework that combines one- and few-step conditional motion generation with learned geometric feedback. The backbone learns finite-interval latent updates conditioned on text and optional joint-position constraints, without relying on a pretrained diffusion teacher. For controlled synthesis, proposal-based geometric feedback corrects the backbone's latent velocity using joint geometry estimated from the current state and refreshed at each sampling step. These corrections are learned through geometric and distributional supervision of the model's own short rollouts, enabling forward-only control without per-sample optimization. Extensive experiments on HumanML3D demonstrate that MoGeFi combines high motion quality, accurate geometric control, and interactive latency across multiple motion generation tasks. In sparse joint control, it achieves comparable FID and roughly halves trajectory and location error rates compared with MaskControl-Fast, while generating each sequence in approximately 61 ms. Ablations show that explicit geometric feedback and per-step proposal refresh improve constraint satisfaction. We also execute generated motions on a physical humanoid using an existing motion tracker. Our project page is available at `https://anonymous.4open.science/w/MoGeFi/`.

1 Introduction

Text-driven human motion generation offers an intuitive way to animate virtual characters and create motion references for embodied systems (Guo et al., 2022a; Luo et al., 2026). In interactive applications, motion may need to continue from an initial movement or reach specified joint positions at particular times, in addition to following a text description. A practical generator must combine natural, text-aligned motion and accurate geometric control with low latency to respond promptly to changing instructions and constraints.

Controllable motion models address these requirements by combining learned motion priors with joint conditioning and geometric guidance. OmniControl couples spatial and realism guidance during diffusion sampling (Xie et al., 2024), while MaskControl combines joint-conditioned masked generation with test-time optimization of token logits (Pinyoanuntapong et al., 2025). These methods use explicit geometric errors to improve constraint satisfaction, but repeated sampling and gradient-based refinement incur substantial online computation. OmniControl, for example, takes over 70 seconds per sequence (Tab. 2), making it difficult to respond promptly to changes in instructions or constraints.

One- and few-step latent models already support real-time generation from text and joint observations (Dong et al., 2026). MotionLCM and its V2 release obtain few-step generators through consistency distillation, transferring a pretrained diffusion model's motion prior to the accelerated sampler (Dai et al., 2024b;a). In sparse joint control, however, the fast MotionLCM-V2 baseline has higher FID and larger positional errors than refinement-based methods (Tab. 2). Its consistency latent tuning (CLT) variant optimizes the sampling latents using gradients from decoded joint errors. This improves control accuracy but brings end-to-end inference time back to over 10 s per sequence.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

Achieving high-quality motion generation and accurate joint control at interactive latency therefore remains challenging (Fig. 1).

We address this challenge with MoGeFi, a few-step conditional generation framework that learns to use explicit geometric feedback. Inspired by average-flow modeling (Geng et al., 2025; 2026), we train the backbone to predict finite-interval average velocities without relying on a pretrained diffusion teacher, enabling one- and few-step synthesis in a compact motion latent space. We jointly train the backbone and a conditioning pathway that preserves the spatio-temporal structure of joint observations and combines them with text and motion latents in a shared Transformer.

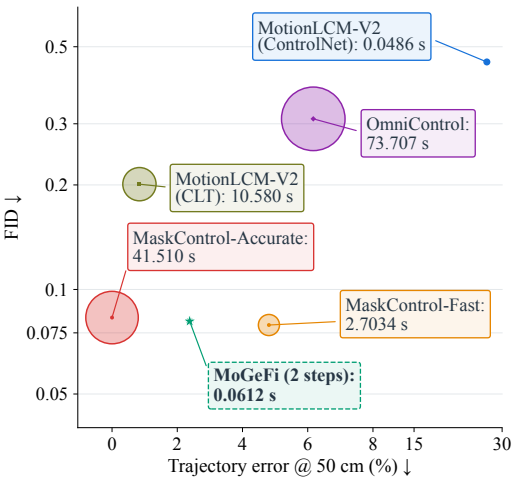

Figure 1: Generation quality, control accuracy, and inference time for sparse joint control on HumanML3D. Labels report inference time, with larger bubbles indicating longer runtimes.

For geometric control, the input specifies desired joint positions at selected motion frames. We introduce *proposal-based geometric feedback*: a frozen snapshot of the learned motion backbone serves as an observer, predicting a clean-motion proposal from the current latent state at each sampling step. A controller combines the proposal's decoded joint geometry and deviations from the input targets with the current state and backbone velocity to predict a latent-velocity correction. Refreshing the proposal keeps the feedback aligned with the evolving candidate.

We supervise finite-interval latent updates with a flow objective. Because feedback corrections alter subsequent sampling states, we also train on the model's own short rollouts: decoded joint losses penalize target deviations, while a representation-space Fréchet objective (Yang et al., 2026) matches complete-motion feature statistics to those of real motions. Inference uses only forward evaluations, with no per-sample optimization.

Experiments on HumanML3D (Guo et al., 2022a) demonstrate high-quality text-to-motion generation and accurate initial-motion and sparse joint control at interactive latency. Ablations show that explicit geometric feedback and proposal refresh improve constraint satisfaction. We also execute text-generated motions on a physical Unitree G1 using SONIC's motion tracker.

Our contributions are summarized as follows:

- We develop a conditional motion generation framework that learns finite-interval latent updates without relying on a pretrained diffusion teacher, enabling high-quality synthesis from text and spatio-temporal joint observations in one or a few steps.
- We introduce proposal-based geometric feedback that uses refreshed motion estimates to predict state-dependent latent-velocity corrections. Learned through the model's own short rollouts, these corrections improve joint constraint satisfaction without per-sample optimization at inference.
- Extensive experiments demonstrate that our framework combines high motion quality, accurate geometric control, and interactive latency across multiple motion generation tasks.

2 Related Work

Text-driven motion generation. Early approaches learn language–motion mappings through sequence prediction and latent-variable modeling (Ahuja & Morency, 2019; Guo et al., 2022a), while MotionCLIP uses pretrained vision–language representations to connect motion and text (Tevet et al., 2022). MotionDiffuse and MDM denoise motion sequences (Zhang et al., 2024; Tevet et al., 2023), whereas MLD performs diffusion in a learned continuous motion latent space (Chen et al., 2023). Another line represents motions as discrete tokens: TM2T and T2M-GPT use autoregressive

prediction (Guo et al., 2022b; Zhang et al., 2023a), while MoMask combines bidirectional masked prediction with residual quantization (Guo et al., 2024).

Efficient motion generation. Motion generation has been accelerated through both cheaper network evaluations and fewer sampling updates. FlashMo uses frequency-aware token pruning (Zhang et al., 2025), while EMDM learns denoising transitions over larger step sizes (Zhou et al., 2024). MotionLCM distills a pretrained latent diffusion model into a consistency model for one- and few-step synthesis (Dai et al., 2024b). Flow-based methods include Motion Flow Matching for ODE synthesis (Hu et al., 2023) and MotionFlow for fast latent-space synthesis (Dong et al., 2026).

Geometrically controllable motion generation. Beyond text, motion synthesis can be conditioned on joint locations, trajectories, and partial keyframes. PriorMDM adapts diffusion priors to end-effector constraints (Shafir et al., 2024), and CondMDI conditions motion in-betweening on sparse or dense keyframes (Cohan et al., 2024). GMD combines motion imputation with spatial guidance (Karunratanakul et al., 2023), while OmniControl couples analytical spatial guidance with learned realism guidance to control different joints (Xie et al., 2024). In latent space, MotionLCM trains a motion ControlNet using decoded joint supervision (Dai et al., 2024b), and MotionFlow incorporates a control encoder with motion-space supervision (Dong et al., 2026). Test-time refinement can also operate on the sampling variables: DNO optimizes initial diffusion noise by differentiating motion-space objectives through the sampling trajectory (Karunratanakul et al., 2024), and consistency latent tuning in MotionLCM-V2 optimizes sampling latents (Dai et al., 2024a). MaskControl combines a learned, joint-conditioned logits regularizer with test-time logits optimization (Pinyoanuntapong et al., 2025). MoGeFi learns latent-velocity corrections from proposal-based geometric feedback, enabling forward-only control without per-sample optimization.

3 METHOD

MoGeFi combines few-step conditional motion generation with proposal-based geometric feedback (Fig. 2). A conditional backbone predicts latent updates, while a controller uses the geometry of a current motion estimate to correct these updates.

3.1 TASK AND MOTION REPRESENTATION

Given a text description c and a length L, we generate a motion $\widehat{\mathbf{x}} \in \mathbb{R}^{L\times D}$, where D is the pose-feature dimension. We use the HumanML3D representation (Guo et al., 2022a) and recover world-space joint positions through $\mathcal{J}(\mathbf{x}) \in \mathbb{R}^{L\times J\times 3}$. Optional geometric observations are specified by

$$\mathcal{O} = (\mathbf{Y}, \mathbf{M}), \qquad \mathbf{Y} \in \mathbb{R}^{L\times J\times 3}, \quad \mathbf{M} \in \{0,1\}^{L\times J}, \tag{1}$$

where $M_{fj} = 1$ assigns joint j a target position $\mathbf{Y}_{fj}$ at frame f; unobserved entries and padding are excluded. Text-to-motion uses $\mathbf{M} = 0$, initial-motion control specifies a multi-joint prefix, and sparse joint control specifies sampled positions of a selected joint. We train task-specific models under this common formulation.

A pretrained motion VAE (Dai et al., 2024a) encodes each motion into N continuous tokens $\mathbf{z}_{\text{data}} \in \mathbb{R}^{N\times d_z}$. Its frozen decoder $\mathcal{D}$ maps latents back to motion. We write the conditions as $\mathcal{C} = (c, \mathcal{O}, \omega)$, with guidance strength ω. Frame index f describes motion time; continuous times r, t describe generation, from noise at $t = 1$ to clean motion at $t = 0$.

3.2 FEW-STEP CONDITIONAL GENERATION

Finite-interval motion updates. Inspired by average-flow modeling (Geng et al., 2025; 2026), we learn $\mathbf{u}_\theta(\mathbf{z}, r, t \mid \mathcal{C})$ to predict the *average velocity* over $[r, t]$. Starting from $\mathbf{s}_0 = \boldsymbol{\epsilon} \sim \mathcal{N}(0, I)$ on a grid $1 = t_0 > \cdots > t_K = 0$, we sample by

$$\mathbf{u}_k = \mathbf{u}_\theta(\mathbf{s}_k, t_{k+1}, t_k \mid \mathcal{C}), \qquad \mathbf{s}_{k+1} = \mathbf{s}_k - (t_k - t_{k+1})\mathbf{u}_k. \tag{2}$$

The output is $\widehat{\mathbf{x}} = \mathcal{D}(\mathbf{s}_K)$. One step covers the noise-to-motion interval; multiple steps subdivide it. The backbone is trained directly from motion data without diffusion-teacher distillation.

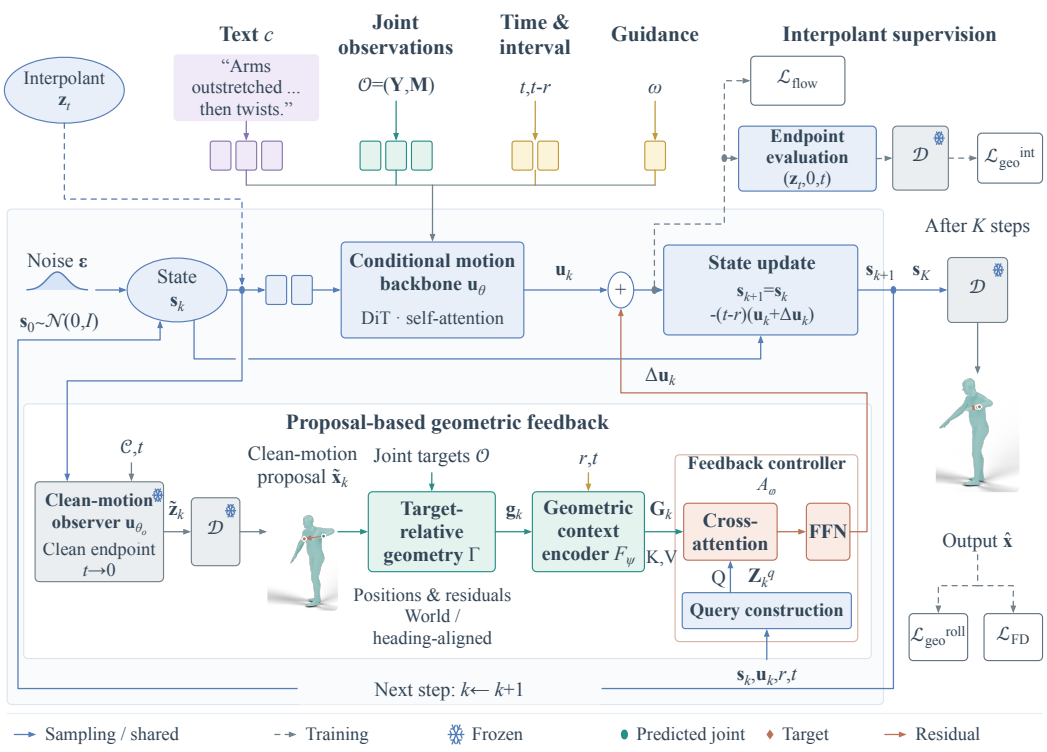

Figure 2: Overall architecture of MoGeFi. Few-step generation with proposal-based geometric feedback, conditioned on $\mathcal{C} = (c, \mathcal{O}, \omega)$. Motion and condition tokens interact in the conditional motion backbone. A frozen clean-motion observer predicts a clean-motion proposal; its geometry relative to the joint targets provides context for a feedback controller that corrects the base average velocity. Each update produces $\mathbf{s}_{k+1}$, which becomes the current state in the next iteration; the final state $\mathbf{s}_K$ is decoded after K steps. Dashed branches show training supervision: $\mathcal{L}_{\text{flow}}$ and $\mathcal{L}_{\text{geo}}^{\text{int}}$ use interpolant evaluations, while $\mathcal{L}_{\text{geo}}^{\text{roll}}$ and $\mathcal{L}_{\text{FD}}$ supervise complete rollouts. Snowflakes mark frozen parameters; the observer proposal is detached during training.

Representing geometric observations. We encode normalized joint targets and observation masks in a fixed joint order, together with frame-position embeddings:

$$\begin{aligned} \mathbf{a}_f &= P_{\mathcal{O}}\big(\mathrm{vec}\big[\{M_{fj}\mathcal{N}_j(\mathbf{Y}_{fj}), M_{fj}\mathbf{1}_3\}_{j=1}^{J}\big]\big) + \mathbf{p}_f, \\ \mathbf{T}_{\mathcal{O}} &= \mathrm{Enc}_{\mathcal{O}}(\mathbf{a}_{1:L}), \qquad \mathbf{C}_{\mathcal{O}} = \mathrm{Resample}(\mathbf{Q}_{\mathcal{O}}, \mathbf{T}_{\mathcal{O}}). \end{aligned} \tag{3}$$

Here $\mathcal{N}_j$ is fixed spatial normalization, and the mask distinguishes unobserved entries from valid zero coordinates. A temporal Transformer encodes the observed frames, and learned queries $\mathbf{Q}_{\mathcal{O}}$ aggregate them into a fixed number of observation tokens. A presence token indicates whether observations are supplied.

Joint semantic and geometric conditioning. The backbone uses DiT-style adaLN-Zero blocks (Peebles & Xie, 2023) with in-context conditioning (Geng et al., 2026). Frozen Sentence-T5 features (Ni et al., 2022) provide text tokens, while separate embeddings encode t, $t - r$, and $1 - \omega^{-1}$. Together with observation tokens and projected motion latents, they form

$$\mathbf{H}^0 = [P_z(\mathbf{z}); \mathbf{C}_c; \mathbf{C}_t; \mathbf{C}_{t-r}; \mathbf{C}_\omega; \mathbf{C}_{\text{pres}}; \mathbf{C}_{\mathcal{O}}] + \mathbf{P}_{\text{pos}}. \tag{4}$$

All valid tokens interact through self-attention; time and interval embeddings also modulate the blocks. The motion-token outputs predict the average velocity. We jointly train the observation encoder and backbone to learn updates conditioned on text and geometric targets. Guidance is learned through the objective in Sec. 3.4, allowing the backbone to predict the guided average velocity in a single evaluation without separately evaluating and combining conditional and null-text outputs.

3.3 PROPOSAL-BASED GEOMETRIC FEEDBACK

The backbone receives the current latent state and joint targets. We provide explicit geometric feedback by predicting and decoding a *clean-motion proposal* from this state and comparing its joint positions with the targets.

At sampling step k, let $t = t_k$ and $r = t_{k+1}$. Denote the proposal by $\widetilde{\mathbf{x}}_k$ and its target-relative geometric descriptors by $\mathbf{g}_k = \Gamma(\widetilde{\mathbf{x}}_k, \mathcal{O})$. A geometric context encoder F_ψ and state-conditioned feedback controller A_ϕ produce

$$\mathbf{G}_k = F_\psi(\mathbf{g}_k; r, t), \qquad \Delta\mathbf{u}_k = A_\phi(\mathbf{s}_k, \mathbf{u}_k, \mathbf{G}_k; r, t), \tag{5}$$

which modifies the candidate transition as

$$\mathbf{s}_{k+1} = \mathbf{s}_k - (t - r)(\mathbf{u}_k + \Delta\mathbf{u}_k). \tag{6}$$

A clean-motion reference for the current state. A frozen copy of a trained generator, with feedback disabled, serves as the *clean-motion observer* $\mathbf{u}_{\theta_o}$. It estimates the clean endpoint from $\mathbf{s}_k$:

$$\widetilde{\mathbf{z}}_k = \mathrm{sg}[\mathbf{s}_k - t\,\mathbf{u}_{\theta_o}(\mathbf{s}_k, 0, t \mid \mathcal{C})], \qquad \widetilde{\mathbf{x}}_k = \mathcal{D}(\widetilde{\mathbf{z}}_k), \tag{7}$$

where sg stops gradients during training. Predicting over $t \to 0$ yields a motion estimate in which joint targets can be evaluated, whereas the candidate advances only over $t \to r$. At each sampling step, we recompute the proposal from the current latent state, keeping the feedback tied to the evolving motion estimate.

World-space and heading-aligned geometry. We combine world-space and heading-aligned geometry to describe targets in both absolute and body-relative coordinates. Let $\widetilde{\mathbf{P}}_k = \mathcal{J}(\widetilde{\mathbf{x}}_k)$. The observed world-space residual is

$$\mathbf{E}^w_{k,fj} = M_{fj}(\mathbf{Y}_{fj} - \widetilde{\mathbf{P}}_{k,fj}). \tag{8}$$

For an observed non-root joint, let $\widetilde{\mathbf{p}}^{xz}_{k,f} = (\widetilde{p}^x_{k,f}, 0, \widetilde{p}^z_{k,f})^\top \in \mathbb{R}^3$ be the proposal's horizontal root translation. Together with the world-to-heading rotation $\widetilde{\mathbf{R}}_{k,f}$, it defines

$$\begin{aligned} \mathbf{y}^\ell_{k,fj} &= \widetilde{\mathbf{R}}_{k,f}(\mathbf{Y}_{fj} - \widetilde{\mathbf{p}}^{xz}_{k,f}), \qquad \widetilde{\mathbf{p}}^\ell_{k,fj} = \widetilde{\mathbf{R}}_{k,f}(\widetilde{\mathbf{P}}_{k,fj} - \widetilde{\mathbf{p}}^{xz}_{k,f}), \\ \mathbf{E}^\ell_{k,fj} &= M_{fj}(\mathbf{y}^\ell_{k,fj} - \widetilde{\mathbf{p}}^\ell_{k,fj}). \end{aligned} \tag{9}$$

Subtracting only horizontal root translation preserves height above the ground. The descriptor Γ combines root state, target and proposal positions, residuals, and observation availability. Target-dependent channels are masked where unobserved. The geometric context encoder applies temporal attention over all valid proposal frames, preserving context between sparse targets.

Conditioning the correction on the intended update. The same geometric discrepancy can require different corrections depending on the update already proposed by the backbone. We therefore form the controller queries from both the current state and base average velocity, together with the generation interval:

$$\begin{aligned} \mathbf{Z}^q_k &= \mathrm{LN}[P_s(\mathrm{LN}(\mathbf{s}_k)) + P_u(\mathrm{LN}(\mathbf{u}_k)) + \mathbf{e}(r, t)], \\ \mathbf{Z}^g_k &= P_g(\mathrm{LN}(\mathbf{G}_k)). \end{aligned} \tag{10}$$

The geometric context supplies keys and values to multi-head cross-attention (Vaswani et al., 2017):

$$\mathbf{A}^{\mathrm{attn}}_k = \mathrm{MHA}(Q = \mathbf{Z}^q_k,\ K = \mathbf{Z}^g_k,\ V = \mathbf{Z}^g_k). \tag{11}$$

Attention and feed-forward residual blocks map the retrieved context to $\Delta\mathbf{u}_k$. A zero-initialized output projection preserves the base update at the start of feedback training, and the correction is zero when observations are absent.

Multi-joint prefix observations. For initial-motion control, the context encoder combines root and joint streams with boundary tokens linking the observed prefix to the generated continuation. The resulting context feeds the same controller, with the proposal and correction rule unchanged. Architecture details are in Sec. S1.

With the default one-step observer, the controlled sampler (Alg. 1) requires K backbone evaluations, K observer evaluations, K proposal decodings, and one final decoding for K sampling steps. Sampling is forward-only, without per-sample optimization; text-only generation omits feedback.

3.4 LEARNING CONDITIONAL GENERATION AND FEEDBACK

We supervise both the generative updates and the motions they produce. Field learning uses *data–noise interpolants*, while output supervision evaluates clean predictions from these interpolants and complete *noise-started rollouts*. Let $\mathbf{u}_\Theta^{\mathrm{fb}} = \mathbf{u}_\theta + \Delta\mathbf{u}$ denote the feedback-augmented field, with $\Theta = (\theta, \psi, \phi)$.

Learning finite-interval conditional updates. For an encoded training motion and independent noise, we form $\mathbf{z}_t = (1-t)\mathbf{z}_{\mathrm{data}} + t\boldsymbol{\epsilon}$ with conditional velocity $\mathbf{v}_{\mathrm{data}} = \boldsymbol{\epsilon} - \mathbf{z}_{\mathrm{data}}$. The average-flow identity $\mathbf{v} = \mathbf{u} + (t-r)\, d\mathbf{u}/dt$ connects a finite-interval prediction to instantaneous velocity along the flow (Geng et al., 2025; 2026). With feedback enabled, we evaluate the Jacobian–vector product with the proposal descriptors held fixed and a detached state tangent $\mathbf{b}$. We use this relation to supervise the average field in instantaneous-velocity space:

Algorithm 1 Few-step sampling with MoGeFi

Require: Conditions $\mathcal{C} = (c, \mathcal{O}, \omega)$ with $\mathcal{O} = (\mathbf{Y}, \mathbf{M})$; grid $1 = t_0 > \cdots > t_K = 0$

$\mathbf{s}_0 \sim \mathcal{N}(0, I)$
for $k = 0, \ldots, K-1$ **do**
  $(t, r) \leftarrow (t_k, t_{k+1})$
  $\mathbf{u}_k \leftarrow \mathbf{u}_\theta(\mathbf{s}_k, r, t \mid \mathcal{C})$
  ▷ Guided average velocity
  $\Delta\mathbf{u}_k \leftarrow \mathbf{0}$
  if $\mathbf{M} \neq \mathbf{0}$ **then** ▷ Geometric feedback
    $\widetilde{\mathbf{z}}_k \leftarrow \mathrm{sg}[\mathbf{s}_k - t\, \mathbf{u}_{\theta_o}(\mathbf{s}_k, 0, t \mid \mathcal{C})]$
    $\widetilde{\mathbf{x}}_k \leftarrow \mathcal{D}(\widetilde{\mathbf{z}}_k)$ ▷ Clean-motion proposal
    $\mathbf{g}_k \leftarrow \Gamma(\widetilde{\mathbf{x}}_k, \mathcal{O})$ ▷ Target-relative geometry
    $\mathbf{G}_k \leftarrow F_\psi(\mathbf{g}_k; r, t)$
    $\Delta\mathbf{u}_k \leftarrow A_\phi(\mathbf{s}_k, \mathbf{u}_k, \mathbf{G}_k; r, t)$
    ▷ Latent-velocity correction
  end if
  $\mathbf{s}_{k+1} \leftarrow \mathbf{s}_k - (t-r)(\mathbf{u}_k + \Delta\mathbf{u}_k)$
end for
return $\mathcal{D}(\mathbf{s}_K)$

$$\mathbf{V}_\Theta = \mathbf{u}_\Theta^{\mathrm{fb}}(\mathbf{z}_t, r, t \mid \mathcal{C}) + (t-r)\,\mathrm{sg}\big[\partial_t \mathbf{u}_\Theta^{\mathrm{fb}} + (\partial_{\mathbf{z}} \mathbf{u}_\Theta^{\mathrm{fb}})\mathbf{b}\big]. \tag{12}$$

A training-only auxiliary instantaneous-velocity head $\mathbf{v}_\theta^{\mathrm{aux}}$ shares the backbone's early blocks. Both predictions are regressed to the same guidance-conditioned target:

$$\mathcal{L}_{\mathrm{flow}} = \mathbb{E}\big[a_u\, \mathrm{MSE}(\mathbf{V}_\Theta, \mathrm{sg}[\mathbf{v}_{\mathrm{tgt}}^\omega]) + \lambda_v a_v\, \mathrm{MSE}(\mathbf{v}_\theta^{\mathrm{aux}}, \mathrm{sg}[\mathbf{v}_{\mathrm{tgt}}^\omega])\big]. \tag{13}$$

Here MSE averages over valid latent entries. The target $\mathbf{v}_{\mathrm{tgt}}^\omega$ combines $\mathbf{v}_{\mathrm{data}}$ with conditional and null-text diagonal predictions, retaining geometric observations in both; it reduces to $\mathbf{v}_{\mathrm{data}}$ at $\omega = 1$. Sec. S1.2 details the target, tangent, and adaptive weights.

Supervising the geometry produced by latent updates. Condition tokens make targets available to the backbone; decoded supervision trains the model to realize those targets in joint space (Dai et al., 2024b). For a batch of motion predictions $\mathbf{x}'$, define

$$\ell_{\mathrm{geo}}(\mathbf{x}'; \mathcal{O}) = \frac{\sum_{i,f,j} M_{ifj} \|\mathcal{N}_j(\mathcal{J}(\mathbf{x}'_i)_{fj}) - \mathcal{N}_j(\mathbf{Y}_{ifj})\|_2^2}{\sum_{i,f,j} M_{ifj} + \varepsilon}. \tag{14}$$

At an interpolant, the clean prediction $\widehat{\mathbf{x}}^{\mathrm{int}} = \mathcal{D}[\mathbf{z}_t - t\mathbf{u}_\Theta^{\mathrm{fb}}(\mathbf{z}_t, 0, t \mid \mathcal{C})]$ gives $\mathcal{L}_{\mathrm{geo}}^{\mathrm{int}} = \mathbb{E}\ell_{\mathrm{geo}}(\widehat{\mathbf{x}}^{\mathrm{int}}; \mathcal{O})$.

During generation, later states result from earlier model updates rather than data interpolation. We therefore also train through the actual short sampling process. Let $\mathcal{S}_{\Theta;\theta_o}^{(K)}$ denote the sampler in Alg. 1, including final decoding, initialized with independent noise $\boldsymbol{\xi} \sim \mathcal{N}(0, I)$:

$$\widehat{\mathbf{x}}^{\mathrm{roll}} = \mathcal{S}_{\Theta;\theta_o}^{(K)}(\boldsymbol{\xi} \mid \mathcal{C}), \qquad \mathcal{L}_{\mathrm{geo}}^{\mathrm{roll}} = \mathbb{E}\ell_{\mathrm{geo}}(\widehat{\mathbf{x}}^{\mathrm{roll}}; \mathcal{O}). \tag{15}$$

Gradients pass through the candidate transitions and feedback networks, training corrections through their effect on final constraint satisfaction, while stopping at the observer proposals.

Supervising complete generated motions. We apply a distributional objective to complete rollouts in both text-only and controlled generation, supervising motion statistics beyond the observed joint targets. Following representation Fréchet training (Yang et al., 2026), we extract $\Phi(\widehat{\mathbf{x}}^{\mathrm{roll}})$ with the frozen motion encoder of the pretrained text–motion evaluator (Guo et al., 2022a) and match its feature statistics to those of real training motions:

$$\begin{aligned} D_{\mathrm{F}} &= \|\boldsymbol{\mu}_g - \boldsymbol{\mu}_r\|_2^2 + \mathrm{Tr}\Big[\boldsymbol{\Sigma}_g + \boldsymbol{\Sigma}_r - 2(\boldsymbol{\Sigma}_r^{1/2}\boldsymbol{\Sigma}_g\boldsymbol{\Sigma}_r^{1/2})^{1/2}\Big], \\ \mathcal{L}_{\mathrm{FD}} &= D_{\mathrm{F}}/(\mathrm{sg}[D_{\mathrm{F}}] + \varepsilon_{\mathrm{FD}}). \end{aligned} \tag{16}$$

The subscripts g and r denote generated and real feature statistics. Running first and second moments aggregate a broader sample window while retaining gradients through the current batch.

The combined objective is

$$\mathcal{L} = \lambda_{\text{flow}}\mathcal{L}_{\text{flow}} + \lambda_{\text{int}}\mathcal{L}_{\text{geo}}^{\text{int}} + \lambda_{\text{roll}}\mathcal{L}_{\text{geo}}^{\text{roll}} + \lambda_{\text{FD}}\mathcal{L}_{\text{FD}}. \tag{17}$$

The decoder and feature encoder remain frozen but transmit input gradients to the learned generator and feedback modules. Text-only training uses field and distributional supervision without geometric terms. Controlled models first learn conditional generation and then feedback. The observer can be updated using a frozen copy of the adapted generator. Feedback calibration then adapts the geometric context encoder and controller to this proposal source, with the backbone fixed and the training objective unchanged.

4 Experiments

4.1 Experimental Setup

Dataset and evaluation. We train and evaluate on the standard HumanML3D splits using its pretrained text–motion evaluator (Guo et al., 2022a). Generation and control evaluation follow MotionLCM and MaskControl (Dai et al., 2024b; Pinyoanuntapong et al., 2025). For control evaluation, Traj. Err. and Loc. Err. denote trajectory and location error rates (%) at a 50 cm threshold, while Avg. Err. denotes mean joint-position error in centimeters. We report end-to-end average inference time per sentence (AITS) on an RTX 4090 with batch size one. Qualitative figures report benchmark-average AITS.

Baselines. Our primary baselines are MotionLCM and MotionLCM-V2 (Dai et al., 2024a). We retain V2's pretrained motion VAE, frozen Sentence-T5 encoder, motion representation, and decoded joint-position supervision, while randomly initializing a task-specific backbone.

Implementation. We evaluate text-to-motion generation with one, two, and four steps and use two candidate steps by default for control, balancing motion quality, joint accuracy, and latency. Guidance is set to 5 for text-to-motion and 8 for control, with each model's weights and guidance fixed across sampling budgets. Setup details and additional comparisons are in Secs. S2 and S5.

4.2 Text-to-Motion Generation

Our one-step model achieves the lowest FID among the compared methods (Tab. 1), reaching 0.018 vs. 0.045 for MoMask and 0.072 for one-step MotionLCM-V2, at comparable latency to V2 (0.0226 vs. 0.0227 s). A second step improves R-Precision@1 from 0.515 to 0.530 with a small FID increase to 0.021; four steps provide little further retrieval gain. The two-step model also achieves higher Multimodality than MoMask and MotionLCM-V2, with Diversity close to real data. We also execute text-generated motions on a physical Unitree G1 using SONIC's whole-body tracker (Luo et al., 2026) (Fig. S4).

4.3 Controllable Motion Generation

Sparse joint control. We follow MaskControl's sparse single-joint evaluation protocol and report the arithmetic mean of the six per-joint results. Our two-step model reduces FID from 0.452 to 0.081 and mean joint-position error from 24.64 to 8.41 cm relative to MotionLCM-V2 ControlNet (Tab. 2), while both run in under 0.1 s. Compared with MaskControl-Fast, our model achieves comparable FID (0.081 vs. 0.079), roughly halves trajectory and location error rates, and reduces inference time from 2.70 to 0.061 s. MoGeFi follows the five head targets while capturing the crawling-to-standing sequence (Fig. 3).

Initial-motion control. Following MotionLCM, we constrain six joints over the first 25% of each sequence. Our two-step model reduces FID and mean joint-position error by approximately 67% and 50% relative to MotionLCM-V2 ControlNet, with similar text retrieval (Tab. 3). Compared with MaskControl-Fast, mean joint-position error decreases from 4.69 to 2.54 cm and inference time from 2.47 to 0.071 s.

Table 1: Text-to-motion generation on HumanML3D. In comparison tables, bold and underline indicate the best and second-best results, excluding real data. →: closer to real data is better.

Method	AITS (s)↓	R@1↑	FID↓	MM Dist↓	Div.→	MMod.↑
Real	–	0.511	0.002	2.974	9.503	–
MDM (Tevet et al., 2023)	5.9452	0.320	0.544	5.566	9.559	**2.799**
MLD (Chen et al., 2023)	0.1415	0.481	0.473	3.196	9.724	2.413
MoMask (Guo et al., 2024)	0.1068	0.521	0.045	2.958	9.675	1.241
StableMoFusion (Huang et al., 2024)	0.1560	**0.553**	0.098	2.770	9.748	1.774
MSQ (Wang et al., 2025)	–	0.483	0.063	3.094	9.497	1.290
MotionFlow (1 step) (Dong et al., 2026)	–	0.517	0.333	2.947	9.965	1.879
TCA-T2M (Wang et al., 2026)	–	0.517	0.068	2.951	9.485	1.103
MotionGPT3 (Flow) (Ban et al., 2026)	–	0.544	0.192	2.837	9.340	2.366
MotionLCM (4 steps) (Dai et al., 2024b)	0.0306	0.502	0.304	3.012	9.607	2.259
MotionLCM-V2 (1 step) (Dai et al., 2024a)	0.0227	0.546	0.072	2.767	9.577	1.858
MotionLCM-V2 (2 steps) (Dai et al., 2024a)	0.0256	0.551	0.049	**2.765**	9.584	1.833
MotionLCM-V2 (4 steps) (Dai et al., 2024a)	0.0330	**0.553**	0.056	2.773	9.598	1.758
MoGeFi (1 step)	**0.0226**	0.515	**0.018**	2.919	9.405	2.313
MoGeFi (2 steps)	0.0279	0.530	0.021	2.862	**9.507**	2.186
MoGeFi (4 steps)	0.0384	0.531	0.024	2.859	9.532	2.128

Table 2: Sparse joint control on HumanML3D. MoGeFi improves generation quality and control accuracy over MotionLCM-V2 ControlNet at sub-0.1-second latency. It matches MaskControl-Fast's FID, roughly halves trajectory and location error rates, and runs over 40× faster.

Method	AITS (s)↓	R@3↑	FID↓	Skating↓	Traj.↓	Loc.↓	Avg.↓
OmniControl (Xie et al., 2024)	73.707	0.693	0.310	0.0608	6.17	1.07	4.04
MaskControl-Fast (Pinyoanuntapong et al., 2025)	2.7034	0.808	**0.079**	0.0555	4.81	1.21	5.63
MaskControl-Accurate (Pinyoanuntapong et al., 2025)	41.510	0.805	0.083	**0.0545**	**0.00**	**0.00**	**0.72**
MotionLCM-V2 (ControlNet) (Dai et al., 2024a)	**0.0486**	0.783	0.452	–	27.40	13.15	24.64
MotionLCM-V2 (CLT) (Dai et al., 2024a)	10.580	**0.820**	0.201	–	0.84	0.05	2.31
MoGeFi (2 steps)	0.0612	0.809	0.081	0.0901	2.38	0.60	8.41

Table 3: Initial-motion control on HumanML3D, observing six joints over the first 25% of frames.

Method	AITS (s)↓	FID↓	R@3↑	Traj.↓	Loc.↓	Avg.↓
OmniControl (Xie et al., 2024)	72.657	2.328	0.557	33.62	3.22	9.77
MaskControl-Fast (Pinyoanuntapong et al., 2025)	2.4738	**0.060**	**0.825**	0.28	0.07	4.69
MaskControl-Accurate (Pinyoanuntapong et al., 2025)	–	**0.060**	0.819	**0.00**	**0.00**	**1.42**
MotionLCM (2 steps) (Dai et al., 2024b)	**0.0343**	0.397	0.759	19.60	1.43	10.92
MotionLCM-V2 (ControlNet, 2 steps) (Dai et al., 2024a)	0.0350	0.248	0.799	2.74	0.24	5.03
MoGeFi (2 steps)	0.0708	0.081	0.800	0.43	0.02	2.54



Figure 3: Qualitative comparison of sparse joint control with five head targets.

Table 4: Ablations of geometric feedback for sparse joint control with two sampling steps and guidance $\omega = 8$.

Variant	FID↓	Traj.↓	Avg.↓
(a) Geometric feedback			
Base backbone	0.143	42.05	34.89
Output-supervised backbone	0.148	14.34	16.28
Full model	**0.063**	**2.25**	**8.41**
(b) Feedback dynamics and FD			
Fixed, w/o FD	0.224	25.08	23.08
Refreshed, w/o FD	0.178	5.11	11.01
Refreshed, with FD	0.083	3.36	10.28
(c) Proposal refresh			
Initial reuse	0.069	27.70	25.67
Per-step refresh	0.063	2.25	8.41

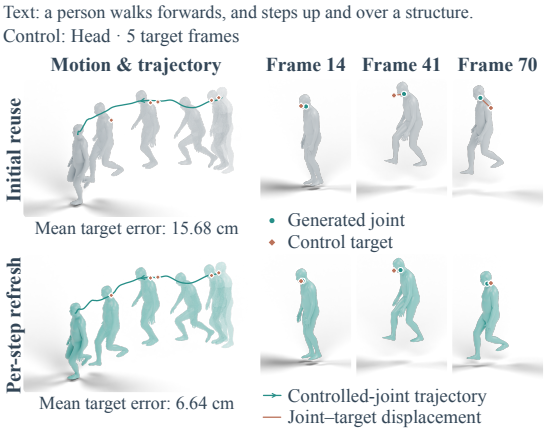

Figure 4: Effect of proposal refresh on sparse head control. Only the proposal-refresh policy changes; model weights, inputs, and sampling budget are fixed. Mean target error is measured over the five constrained frames.

4.4 Analysis of Geometric Feedback

We evaluate the ablations in Tab. 4 by sampling the controlled joint and observation density per sequence and computing metrics over the mixed-joint outputs. Full metrics and additional ablations are in Sec. S3.

(i) Feedback beyond output supervision. The output-supervised backbone uses the full model's loss terms and weights but omits the observer and feedback modules (block a). Output supervision reduces mean joint-position error from 34.89 to 16.28 cm; adding geometric feedback further reduces it to 8.41 cm while improving FID from 0.148 to 0.063. These results show that explicit feedback improves control beyond output supervision alone. **(ii) Proposal refresh during training.** We compare fixed-proposal and refreshed-proposal feedback under the same field and geometric supervision, including rollout geometric losses, without FD (block b). Fixed-proposal feedback reuses the initial proposal throughout each rollout, whereas refreshed-proposal feedback recomputes it from the current state. Refreshed-proposal feedback reduces mean joint-position error from 23.08 to 11.01 cm and FID from 0.224 to 0.178, showing the benefit of learning corrections with updated geometric context. **(iii) Distributional supervision.** With the observer fixed (block b), adding the Fréchet objective reduces FID from 0.178 to 0.083 and mean joint-position error from 11.01 to 10.28 cm. Distributional supervision therefore improves motion distribution quality while preserving control accuracy. The full model additionally uses observer replacement and feedback calibration. **(iv) Proposal refresh during sampling.** With the full model and guidance fixed, block c changes only the proposal-refresh policy; the controller runs at every sampling step. Initial-proposal reuse increases mean joint-position error from 8.41 to 25.67 cm, while FID changes only slightly (0.063 vs. 0.069). Per-step proposal refresh reduces the final head–target offsets compared with initial-proposal reuse (Fig. 4). Refreshing the proposal keeps the geometric feedback aligned with the evolving candidate and improves final constraint satisfaction.

5 Conclusion

We presented MoGeFi, a framework that combines few-step conditional generation with proposal-based geometric feedback learned through geometric and distributional supervision of short sampling rollouts. The framework achieves high-quality one-step text-to-motion generation and combines motion quality, accurate constraint following, and interactive latency in initial-motion and sparse joint control. The sparse-control analyses show that explicit feedback improves constraint following beyond output supervision alone, with further benefits from refreshing the motion estimate during sampling. Improving foot-contact consistency and extending control to richer geometric conditions are promising directions for future work.

AI USE STATEMENT

In this work, we used generative AI tools to assist with implementing and modifying research code. Additionally, we used these tools to polish and organize the manuscript, retrieve and organize related literature, check references, and assist with LaTeX formatting.

The authors reviewed the AI-assisted text and code, checked references against their sources, and validated code changes through testing and experimental checks. We take responsibility for the final content of this work, including its text, claims, code, and other artifacts produced with AI assistance.

REPRODUCIBILITY STATEMENT

Method and implementation details are provided in Secs. S1 and 3, and training and evaluation settings in Sec. S2. Additional analyses and comparisons are in Secs. S3 to S5.

REFERENCES

Chaitanya Ahuja and Louis-Philippe Morency. Language2pose: Natural language grounded pose forecasting. In *International Conference on 3D Vision*, pp. 719–728, 2019.

Jaymin Ban, JiHong Jeon, and SangYeop Jeong. From diffusion to flow: Efficient motion generation in MotionGPT3. In *ReALM-GEN Workshop at ICLR 2026*, 2026. URL https://openreview.net/pdf?id=EuuNOyt282.

Ling-Hao Chen, Shunlin Lu, Wenxun Dai, Zhiyang Dou, Xuan Ju, Jingbo Wang, Taku Komura, and Lei Zhang. Pay attention and move better: Harnessing attention for interactive motion generation and training-free editing. arXiv:2410.18977, 2024.

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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 18000–18010, 2023.

Setareh Cohan, Guy Tevet, Daniele Reda, Xue Bin Peng, and Michiel van de Panne. Flexible motion in-betweening with diffusion models. In *ACM SIGGRAPH 2024 Conference Papers*, pp. 1–9, 2024.

Wenxun Dai, Ling-Hao Chen, Yufei Huo, Jingbo Wang, Jinpeng Liu, Bo Dai, and Yansong Tang. MotionLCM-V2: Improved compression rate for multi-latent-token diffusion. Hugging Face Community Article, 2024a. URL https://huggingface.co/blog/wxDai/motionlcm-v2. Accompanying author-hosted report, "Real-time Controllable Motion Generation via Latent Consistency Model": https://dai-wenxun.github.io/files/MotionLCM_V2.pdf.

Wenxun Dai, Ling-Hao Chen, Jingbo Wang, Jinpeng Liu, Bo Dai, and Yansong Tang. MotionLCM: Real-time controllable motion generation via latent consistency model. In *European Conference on Computer Vision*, pp. 390–408, 2024b.

Kun Dong, Jian Xue, Xing Lan, Qingyuan Liu, and Ke Lu. MotionFlow: Efficient motion generation with latent flow matching. *IEEE Transactions on Multimedia*, 28:2325–2336, 2026. doi: 10.1109/TMM.2026.3651041.

Zhengyang Geng, Mingyang Deng, Xingjian Bai, J. Zico Kolter, and Kaiming He. Mean flows for one-step generative modeling. In *Advances in Neural Information Processing Systems*, volume 38, 2025. doi: 10.52202/085713-2534.

Zhengyang Geng, Yiyang Lu, Zongze Wu, Eli Shechtman, J. Zico Kolter, and Kaiming He. Improved mean flows: On the challenges of fastforward generative models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 30467–30476, 2026.

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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5142–5151, 2022a. doi: 10.1109/CVPR52688.2022.00509.

Chuan Guo, Xinxin Zuo, Sen Wang, and Li Cheng. TM2T: Stochastic and tokenized modeling for the reciprocal generation of 3d human motions and texts. In *European Conference on Computer Vision*, pp. 580–597, 2022b.

Chuan Guo, Yuxuan Mu, Muhammad Gohar Javed, Sen Wang, and Li Cheng. MoMask: Generative masked modeling of 3d human motions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 1900–1910, 2024.

Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. GANs trained by a two time-scale update rule converge to a local nash equilibrium. In *Advances in Neural Information Processing Systems*, volume 30, 2017.

Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. arXiv:2207.12598, 2022.

Vincent Tao Hu, Wenzhe Yin, Pingchuan Ma, Yunlu Chen, Basura Fernando, Yuki M. Asano, Efstratios Gavves, Pascal Mettes, Björn Ommer, and Cees G. M. Snoek. Motion flow matching for human motion synthesis and editing. arXiv:2312.08895, 2023.

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 *Proceedings of the 32nd ACM International Conference on Multimedia*, pp. 224–232, 2024. doi: 10.1145/3664647.3681657.

Korrawe Karunratanakul, Konpat Preechakul, Supasorn Suwajanakorn, and Siyu Tang. Guided motion diffusion for controllable human motion synthesis. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 2151–2162, 2023.

Korrawe Karunratanakul, Konpat Preechakul, Emre Aksan, Thabo Beeler, Supasorn Suwajanakorn, and Siyu Tang. Optimizing diffusion noise can serve as universal motion priors. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 1334–1345, 2024.

Zhengyi Luo, Ye Yuan, Tingwu Wang, Chenran Li, Fernando Castañeda, Sirui Chen, Zi-Ang Cao, Jiefeng Li, David Minor, Qingwei Ben, Jinhyung Park, David Sami, Zi Wang, Xingye Da, Runyu Ding, Cyrus Hogg, Lina Song, Edy Lim, Eugene Jeong, Tairan He, Haoru Xue, Wenli Xiao, Simon Yuen, Jan Kautz, Yan Chang, Umar Iqbal, Linxi Jim Fan, and Yuke Zhu. SONIC: Supersizing motion tracking for natural humanoid whole-body control. *Science Robotics*, 11(117): eaed4592, 2026. doi: 10.1126/scirobotics.aed4592.

Jianmo Ni, Gustavo Hernández Ábrego, Noah Constant, Ji Ma, Keith B. Hall, Daniel Cer, and Yinfei Yang. Sentence-T5 (ST5): Scalable sentence encoders from pre-trained text-to-text models. In *Findings of the Association for Computational Linguistics: ACL 2022*, pp. 1864–1874, 2022. doi: 10.18653/v1/2022.findings-acl.146.

William Peebles and Saining Xie. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 4195–4205, 2023.

Mathis Petrovich, Michael J. Black, and Gül Varol. TMR: Text-to-motion retrieval using contrastive 3d human motion synthesis. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 9488–9497, 2023.

Ekkasit Pinyoanuntapong, Muhammad Usama Saleem, Korrawe Karunratanakul, Pu Wang, Hongfei Xue, Chen Chen, Chuan Guo, Junli Cao, Jian Ren, and Sergey Tulyakov. MaskControl: Spatio-temporal control for masked motion synthesis. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 9955–9965, 2025.

Yonatan Shafir, Guy Tevet, Roy Kapon, and Amit H. Bermano. Human motion diffusion as a generative prior. In *International Conference on Learning Representations*, 2024.

Guy Tevet, Brian Gordon, Amir Hertz, Amit H. Bermano, and Daniel Cohen-Or. MotionCLIP: Exposing human motion generation to CLIP space. In *European Conference on Computer Vision*, pp. 358–374, 2022.

Guy Tevet, Sigal Raab, Brian Gordon, Yonatan Shafir, Daniel Cohen-Or, and Amit H. Bermano. Human motion diffusion model. In *International Conference on Learning Representations*, 2023.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30, 2017.

Hongsong Wang, Wenjing Yan, Qiuxia Lai, and Xin Geng. Temporal consistency-aware text-to-motion generation. *Visual Intelligence*, 4:7, 2026. doi: 10.1007/s44267-026-00110-8.

Zan Wang, Jingze Zhang, Yixin Chen, Baoxiong Jia, Wei Liang, and Siyuan Huang. Spatial-temporal multi-scale quantization for flexible motion generation. arXiv:2508.08991, 2025. URL https://arxiv.org/abs/2508.08991.

Yiming Xie, Varun Jampani, Lei Zhong, Deqing Sun, and Huaizu Jiang. OmniControl: Control any joint at any time for human motion generation. In *International Conference on Learning Representations*, 2024.

Jiawei Yang, Zhengyang Geng, Xuan Ju, Yonglong Tian, and Yue Wang. Representation fréchet loss for visual generation. arXiv:2604.28190, 2026.

Jianrong Zhang, Yangsong Zhang, Xiaodong Cun, Yong Zhang, Hongwei Zhao, Hongtao Lu, Xi Shen, and Ying Shan. Generating human motion from textual descriptions with discrete representations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 14730–14740, 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 *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 364–373, 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. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 46(6):4115–4128, 2024. doi: 10.1109/TPAMI.2024.3355414.

Zeyu Zhang, Yiran Wang, Danning Li, Dong Gong, Ian Reid, and Richard Hartley. FlashMo: Geometric interpolants and frequency-aware sparsity for scalable efficient motion generation. In *Advances in Neural Information Processing Systems*, volume 38, 2025.

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 *European Conference on Computer Vision*, pp. 18–38, 2024.

Supplementary Material

S1 Method and Implementation Details

We provide the full field objective, feedback descriptors, gradient paths, and feature-statistics estimator.

S1.1 Representation, Conditioning, and Network Roles

We use the 263-dimensional HumanML3D representation for a 22-joint skeleton (Guo et al., 2022a), comprising root motion, local joint positions, rotations, velocities, and foot contacts. World positions are recovered by integrating root motion and transforming relative positions. The frozen MotionLCM-V2 VAE supplies 16 latent tokens of dimension 32 (Dai et al., 2024a); the generative backbone is initialized independently.

The backbone uses 12 DiT blocks with hidden dimension 768 and 12 attention heads (Peebles & Xie, 2023). Frozen Sentence-T5 features (Ni et al., 2022) provide text conditioning. Separate embeddings encode time, interval, and $1-\omega^{-1}$; condition slots and motion latents interact bidirectionally. Time and interval additionally supply the adaLN-Zero modulation. The training-only auxiliary instantaneous-velocity head shares early blocks and uses a separate tail and output projection.

The joint-observation encoder concatenates masked coordinates with observation indicators, retains joint identity through a fixed ordering, and adds frame-position embeddings. A temporal Transformer encodes the frame tokens, excluding unobserved frames and padding as attention context. Learned queries aggregate the sequence through attention, a residual feed-forward block, and normalization. A presence embedding handles text-only inputs. The sparse-joint geometric context encoder retains every valid proposal frame, including frames with masked target channels; the multi-joint prefix context encoder pools root/joint sequences and preserves boundary tokens.

S1.2 Guidance-Conditioned Field Objective

We train the conditional motion field using the average-flow identity and instantaneous-space regression (Geng et al., 2025; 2026). Let $\mathbf{u}_\Theta^{\mathrm{fb}}$ denote the field including feedback when enabled. For an encoded data motion and independent Gaussian noise,

$$\mathbf{z}_t = (1-t)\mathbf{z}_{\mathrm{data}} + t\boldsymbol{\epsilon}, \qquad \mathbf{v}_{\mathrm{data}} = \boldsymbol{\epsilon} - \mathbf{z}_{\mathrm{data}}, \qquad 0 \le r \le t \le 1. \tag{S1}$$

The identity $\mathbf{v} = \mathbf{u} + (t-r)\, d\mathbf{u}/dt$ converts the average prediction to instantaneous-velocity space. To learn guidance-conditioned generation, we use the iMF guidance construction, based on conditional and null-text predictions as in classifier-free guidance (Ho & Salimans, 2022):

$$\mathbf{b}_c = \mathrm{sg}\big[\mathbf{u}_\Theta^{\mathrm{fb}}(\mathbf{z}_t, t, t \mid c, \mathcal{O}, \omega)\big], \tag{S2}$$

$$\mathbf{b}_\varnothing = \mathrm{sg}\big[\mathbf{u}_\Theta^{\mathrm{fb}}(\mathbf{z}_t, t, t \mid \varnothing, \mathcal{O}, 1)\big], \tag{S3}$$

$$\mathbf{v}_{\mathrm{tgt}}^{\omega} = \mathbf{v}_{\mathrm{data}} + (1-\omega^{-1})(\mathbf{b}_c - \mathbf{b}_\varnothing). \tag{S4}$$

Both branches retain the geometric observations; the null-text branch uses guidance 1. At $\omega = 1$, the target equals $\mathbf{v}_{\mathrm{data}}$. Learning this target makes guidance strength an input to the sampling field.

Let $\mathbf{b}$ denote the JVP state tangent. After warm-up, $\mathbf{b} = \mathbf{b}_c$; during warm-up, it interpolates from $\mathbf{v}_{\mathrm{data}}$ to that learned diagonal estimate. With feedback enabled, we construct the raw proposal descriptors $\bar{\mathbf{g}}$ once from the interpolant and treat them as fixed conditioning inputs to the local field evaluations. We apply the average-flow regression to this conditional field, holding r and $\bar{\mathbf{g}}$ fixed in the JVP:

$$\dot{\mathbf{u}}_\Theta^{\mathrm{fb}} = \partial_t \mathbf{u}_\Theta^{\mathrm{fb}}\big|_{\bar{\mathbf{g}}} + \Big(\partial_{\mathbf{z}} \mathbf{u}_\Theta^{\mathrm{fb}}\big|_{\bar{\mathbf{g}}}\Big)\,\mathbf{b}, \qquad \mathbf{V}_\Theta = \mathbf{u}_\Theta^{\mathrm{fb}}(\mathbf{z}_t, r, t \mid \mathcal{C}) + (t-r)\,\mathrm{sg}[\dot{\mathbf{u}}_\Theta^{\mathrm{fb}}]. \tag{S5}$$

This is the JVP along $(\mathbf{b}, 0, 1)$ for inputs $(\mathbf{z}, r, t)$. The derivative retains the explicit state and time dependence of the backbone and feedback modules, including the context encoder's time dependence. During sampling, the proposal descriptors are recomputed from the current state at each

step. The stop-gradient prevents differentiation through the derivative term when optimizing the regression. The auxiliary head is supervised against the same guided target:

$$\mathcal{L}_{\text{flow}} = \mathbb{E}\big[a_u \,\text{MSE}(\mathbf{V}_\Theta, \text{sg}[\mathbf{v}^{\omega}_{\text{tgt}}]) + \lambda_v a_v \,\text{MSE}(\mathbf{v}^{\text{aux}}_\theta, \text{sg}[\mathbf{v}^{\omega}_{\text{tgt}}])\big]. \tag{S6}$$

The MSE is averaged over valid latent entries. Let e_u and e_v denote the corresponding per-sample squared errors before weighting. The adaptive weights are

$$a_u = (\text{sg}[e_u] + 10^{-3})^{-p}, \qquad a_v = (\text{sg}[e_v] + 10^{-3})^{-p}, \tag{S7}$$

where p is the weighting exponent. The average field's diagonal prediction supplies the learned tangent; the auxiliary head receives additional regression supervision.

S1.3 Geometric Feedback Descriptors

Sparse single-joint control. The sparse feedback descriptor has 44 scalar channels per valid frame (Tab. S1). Its reference root is taken from the proposal, so the reference and proposal root positions coincide and the root-offset channels are zero.

Table S1: Geometric feedback descriptors for sparse joint control. Joint identity and generation/frame embeddings are supplied separately from these 44 channels.

Group	Channels	Contents
Reference root state	8	Position (3), planar velocity (2), yaw-quaternion components (2), heading change (1).
Proposal root and offset	8	Proposal position (3), yaw-quaternion components (2), reference-minus-proposal position (3).
World geometry	9	Target, proposal joint position, and positional residual (3 each).
Heading-aligned geometry	9	Target, proposal joint position, and residual in the heading-aligned frame (3 each).
Observation flags and context	10	Active axes (3), observed-point flag, valid-frame flag, density, world-residual norm, residual-based proximity score, heading-frame applicability, pelvis indicator.

The heading transformation removes horizontal root translation and preserves height. World and heading-aligned positions give the controller absolute and body-relative descriptions alongside root context. Pelvis control uses zero-valued heading-aligned channels.

Target coordinates are masked before they enter the descriptors. Residuals and the proximity-score channel are zero where unobserved; the latter is an exponential function of the world residual norm using a fixed positive scale. The valid-frame mask distinguishes sequence padding from a valid frame without a target. Observation density is the fraction of valid frames with an observed point. Position, velocity, heading-change, and residual channels use fixed physical scales before projection. A sparse-joint geometric context encoder adds joint identity, frame position, density, root-source identity, and generation-time embeddings. Its output preserves a valid-frame context sequence for the cross-attention controller.

Multi-joint initial-motion control. The multi-joint prefix context encoder uses root, joint, and boundary streams. Root features supply the generated body's global placement and motion and its relation to observed root positions. Joint features encode world-space and heading-aligned positions and per-frame displacement features, together with target/proposal discrepancies. Target displacements are computed only when both adjacent frames are observed. If $M_{f-1,j}M_{f,j} = 1$, the target finite difference is $\mathbf{Y}_{f,j} - \mathbf{Y}_{f-1,j}$; otherwise that target-dependent displacement channel is masked.

For joint tokens $\mathbf{T}^0 \in \mathbb{R}^{L \times J_c \times d_c}$, the encoder first connects a joint across time and then connects joints within a frame:

$$\mathbf{T}^1_{:,j} = \text{Enc}_{\text{time}}(\mathbf{T}^0_{:,j}), \qquad \mathbf{T}^2_{f,:} = \text{Enc}_{\text{joint}}(\mathbf{T}^1_{f,:}). \tag{S8}$$

This structure reflects the two relations in a multi-joint prefix: continuity of each controlled trajectory and coordination across the body. Learned attention pooling produces root context and separate temporal summaries for the controlled joints.

The boundary stream retains the final observed state, terminal observed velocity, and the proposal's first unobserved state and transition. Boundary-type and joint embeddings identify these three roles. After boundary encoding, the context is

$$\mathbf{G}_k^{\text{prefix}} = \left[\text{Pool}_r(\mathbf{T}^r); \{\text{Pool}_j(\mathbf{T}^2_{:,j})\}_{j=1}^{J_c}; \text{Enc}_b(\mathbf{T}^b)\right]. \tag{S9}$$

Both sparse-joint and multi-joint prefix context encoders use the candidate interval (r, t). Their inputs differ in task structure, but both yield context for the same state-conditioned latent-velocity correction.

Controller details. The controller queries depend on the current candidate and the update it is about to receive. We use the normalized state/velocity projections in Eq. (10), with an MLP embedding of $[t, r, t-r, tr]$. Let $\mathbf{Z}_k^q \in \mathbb{R}^{N\times d}$ be the resulting query inputs and $\mathbf{Z}_k^g \in \mathbb{R}^{P\times d}$ the independently normalized and projected context. For attention head h,

$$\begin{aligned} \mathbf{Q}_k^h &= \mathbf{Z}_k^q W_Q^h, \qquad \mathbf{K}_k^h = \mathbf{Z}_k^g W_K^h, \qquad \mathbf{V}_k^h = \mathbf{Z}_k^g W_V^h, \\ \mathbf{A}_k^{\text{attn},h} &= \text{softmax}\left(\frac{\mathbf{Q}_k^h (\mathbf{K}_k^h)^\top}{\sqrt{d_h}} + \mathbf{B}_{\text{mask}}\right)\mathbf{V}_k^h, \\ \mathbf{A}_k^{\text{attn}} &= \text{Concat}_h(\mathbf{A}_k^{\text{attn},h}) W_O. \end{aligned} \tag{S10}$$

Here d_h is the head dimension and $\mathbf{B}_{\text{mask}}$ excludes invalid context tokens. The remaining residual blocks are

$$\mathbf{B}_k = \text{LN}(\mathbf{Z}_k^q + \mathbf{A}_k^{\text{attn}}), \qquad \Delta\mathbf{u}_k = P_\Delta(\text{LN}[\mathbf{B}_k + \text{FFN}(\mathbf{B}_k)]). \tag{S11}$$

The normalization layers have separate parameters. The feed-forward network has two linear layers with GELU and dropout. The final projection P_Δ has no bias and is zero-initialized. Its output is gated to zero for samples without observations, and padded latent entries are masked. These operations produce one correction per latent token while allowing the geometric context to have a different length.

S1.4 Prediction Sources and Gradient Paths

The interpolant endpoint receives joint-target supervision. The observer proposal supplies geometric feedback. The noise-started rollout output receives distributional supervision and, for controlled generation, joint-target supervision. Tab. S2 summarizes their gradient paths.

Table S2: Trainable parameters, gradient paths, and inference components.

Path	Parameter treatment	Input-gradient treatment in rollout training	Inference
Conditional motion backbone	Learned during generator/feedback adaptation; fixed during calibration	Candidate-state gradients retained	Yes
Observer	Fixed snapshot	State-to-proposal branch stopped	Yes
Proposal decoding/geometry	Frozen decoder and deterministic construction	Stopped observer branch	Yes
Geometric context encoder and feedback controller	Learned in feedback adaptation and calibration	Gradients retained	Yes
Final candidate decoder	Frozen	Input gradients retained	Yes
Motion feature encoder	Frozen	Input gradients retained for current FD batch	No
Auxiliary instantaneous-velocity head	Training supervision only	Shared representation receives its training loss	No

During feedback calibration, candidate-state gradients pass through the frozen backbone and final decoder to the trainable feedback modules. The observer branch is evaluated with stop-gradient.

S1.5 Representation Fréchet Estimation

Let $\mathbf{h}_i = \Phi(\widehat{\mathbf{x}}_i^{\text{roll}})$ be features from the frozen T2TM motion encoder. For batch size B, define the first and second moments

$$\boldsymbol{\mu}_B = \frac{1}{B}\sum_i \mathbf{h}_i, \qquad \mathbf{S}_B = \frac{1}{B}\sum_i \mathbf{h}_i\mathbf{h}_i^\top. \tag{S12}$$

The EMA estimator maintains a wider statistical window than the current optimization batch (Yang et al., 2026):

$$\boldsymbol{\mu}_g^{(n)} = \beta\,\mathrm{sg}[\boldsymbol{\mu}_g^{(n-1)}] + (1-\beta)\boldsymbol{\mu}_B, \tag{S13}$$

$$\mathbf{S}_g^{(n)} = \beta\,\mathrm{sg}[\mathbf{S}_g^{(n-1)}] + (1-\beta)\mathbf{S}_B, \tag{S14}$$

$$\boldsymbol{\Sigma}_g^{(n)} = \mathbf{S}_g^{(n)} - \boldsymbol{\mu}_g^{(n)}(\boldsymbol{\mu}_g^{(n)})^\top. \tag{S15}$$

Only current features carry gradients; historical state is detached after each update. Separate estimator states are maintained for different rollout budgets. Real moments are computed once from the training split. The symmetric square-root expression in the main paper is used for the covariance term, and the loss is normalized by its detached value plus a numerical stabilizer.

The same formulation also supports the alternative TMR training features (Petrovich et al., 2023) examined in the representation comparison.

S2 Training and Evaluation Details

S2.1 Training Schedule

We first learn a conditional generator for each task. Text-to-motion uses distributional adaptation, while controlled models learn proposal-based feedback on sampled trajectories. Tab. S3 specifies the optimization schedule. The field and geometric objectives remain active when distributional supervision is introduced.

Table S3: Training budgets and objective schedules.

Training	Budget	Learning rate	Sampling and supervision
T2M adaptation	20k	10^{-6}	Rollouts in $\{1, 2, 4\}$; add distributional supervision.
Control adaptation	40k	10^{-4}	First 20k: rollouts in $\{1, 2, 4\}$ with field/geometric losses. Next 20k: four-step rollouts with distributional supervision.
Feedback calibration	5k	2.5×10^{-5}	Updated frozen observer; optimize feedback modules with the backbone fixed.

Loss weights. We use the weights in Tab. S4 for the objective in Eq. (17). Within the flow objective, the auxiliary velocity loss has weight $\lambda_v = 0.1$, and the adaptive weighting exponent is $p = 1$. For control training, per-sample flow losses are additionally scaled by 0.5 when geometric observations are present and by 1 otherwise.

Table S4: Loss weights across training stages.

Training stage	λ_{flow}	λ_{int}	λ_{roll}	λ_{FD}
T2M generation	1	0	0	0
T2M distributional adaptation	0.01	0	0	1
Feedback learning without FD	1	1	1	0
Feedback learning with FD	1	1	1	1
Feedback calibration	1	1	1	1

The pretrained motion VAE and text encoder remain frozen. For each task, weights and guidance are fixed across sampling budgets.

S2.2 Ablation Settings

The main feedback ablations use the random-joint control setting.

For the output-supervision comparison, we train a backbone-only model with the same field, interpolant geometric, rollout geometric, and distributional loss terms and weights as the full feedback model. The backbone-only variant retains text and joint conditioning and removes the observer, geometric context encoder, and controller. These are independently trained alternatives.

The fixed-proposal and refreshed-proposal feedback variants both use field and geometric objectives, including rollout geometric supervision, without FD. Fixed-proposal feedback retains the initial proposal within each rollout, while refreshed-proposal feedback recomputes it at every candidate step. The FD comparison uses refreshed-proposal feedback with the same observer. The inference-time refresh comparison fixes the final model and changes only the refresh policy; its controller executes at every step in both settings. All main ablations use two candidate steps and guidance 8.

S2.3 Tasks and Evaluation

We use the HumanML3D split, 263-dimensional representation, and pretrained T2TM matching network (Guo et al., 2022a). Initial-motion control follows MotionLCM (Dai et al., 2024b): positions of joints $[0, 10, 11, 15, 20, 21]$ (pelvis, feet, head, and wrists) are observed over the first quarter of each sequence. Sparse control follows OmniControl and MaskControl (Xie et al., 2024; Pinyoanuntapong et al., 2025), with observation densities of 1, 2, or 5 frames, 25%, or 100%. Pelvis-only control uses a separately trained model.

Method comparisons in Tabs. 2 and S14 report the arithmetic mean of six separate per-joint evaluations. For our results, each joint is evaluated over five full-test repetitions. Random-joint ablations and sampling-budget analyses instead sample the controlled joint and observation density per sequence and compute metrics over the mixed-joint outputs. Both settings use the same joint set and observation-density choices.

FID (Heusel et al., 2017), R-Precision, matching distance, Diversity, and Multimodality use the T2TM features. Diversity is compared with the real-motion reference; Multimodality measures variation under the same text. Control evaluation reports mean joint-position error in centimeters, trajectory error rate, and location error rate. Trajectory error rate is the percentage of sequences with any observed joint position more than 50 cm from its target; location error rate is the percentage of observed joint positions exceeding the same threshold. Foot skating is reported using the MaskControl protocol. Our tables report means over five sampling/evaluation repetitions.

S2.4 Comparison Sources

Baseline quality metrics are taken from the cited reports or evaluated using the released implementations. MotionLCM-V2 uses the authors' multi-latent-token release (Dai et al., 2024a); its ControlNet and test-time CLT configurations are separate rows. Its quality and control metrics in Tab. 2 are taken from the authors' report. We evaluate the released implementation using its official sampling configurations to obtain the per-joint results in Tab. S14, which are not provided in that report. The OmniControl and MaskControl-Fast entries in the latter table use the per-joint results in the MaskControl supplementary material (Pinyoanuntapong et al., 2025). MaskControl's initial-motion results are evaluated through its official implementation with the same six-joint prefix condition. We report its Fast, Medium, and Accurate settings.

S2.5 Inference Time

Following MLD (Chen et al., 2023), we measure AITS for all methods on a single RTX 4090 over 1,000 samples at batch size one, with CUDA synchronization per sample. The timed path includes online text encoding, condition encoding, candidate updates, observer predictions, and motion decoding. Model loading, quality evaluation, and saving are excluded. Baselines use the inference configurations of their official implementations.

With K candidate steps and observer-step allocation $\mathbf{n}^{\mathrm{obs}} = (n_0^{\mathrm{obs}}, \ldots, n_{K-1}^{\mathrm{obs}})$, the controlled sampler performs K candidate field evaluations and $\sum_k n_k^{\mathrm{obs}}$ observer field evaluations, plus K proposal

decodings and one final decoding. The main setting has $n_k^{\text{obs}} = 1$ throughout. This count explains how the number of observer integration steps changes computation while keeping the candidate budget fixed.

S3 Additional Analysis of Sparse Joint Control

We extend the main ablations in the random-joint control setting. Unless stated otherwise, we use two candidate steps, guidance $\omega = 8$, and one observer step per proposal.

S3.1 Complete Feedback and Supervision Results

The complete ablation results are reported in Tab. S5, including additional metrics and observer-replacement and feedback-calibration variants. Training settings are given in Sec. S2.2.

Table S5: Complete metrics for geometric feedback and supervision ablations in random-joint control.

Variant	FID↓	R@3↑	Div.→	Skate↓	Traj. err. (50 cm) (%)↓	Loc. err. (50 cm) (%)↓	Avg. err. (cm)↓
(a) Backbone-only baselines							
Base conditional backbone	0.143	0.823	9.689	0.0778	42.05	21.83	34.89
Output-supervised backbone	0.148	0.775	9.389	0.1173	14.34	3.71	16.28
(b) Feedback dynamics and distributional supervision							
Fixed-proposal feedback, w/o FD	0.224	0.800	9.561	0.0917	25.08	9.91	23.08
Refreshed-proposal feedback, w/o FD	0.178	0.801	9.452	0.0976	5.11	1.20	11.01
Refreshed-proposal feedback, with FD	0.083	0.810	9.461	0.1006	3.36	0.87	10.28
(c) Observer replacement and feedback calibration							
Observer replaced	0.084	0.808	9.448	0.1017	3.08	0.78	9.07
Full model (calibrated feedback)	**0.063**	**0.812**	**9.492**	**0.0905**	**2.25**	**0.59**	**8.41**

Fig. S1 compares the base conditional backbone, the output-supervised backbone, and the full model on a stepping-over example. Output supervision brings the head trajectory closer to the targets, while the full model further reduces the visible offsets and captures the stepping-over motion described by the text.

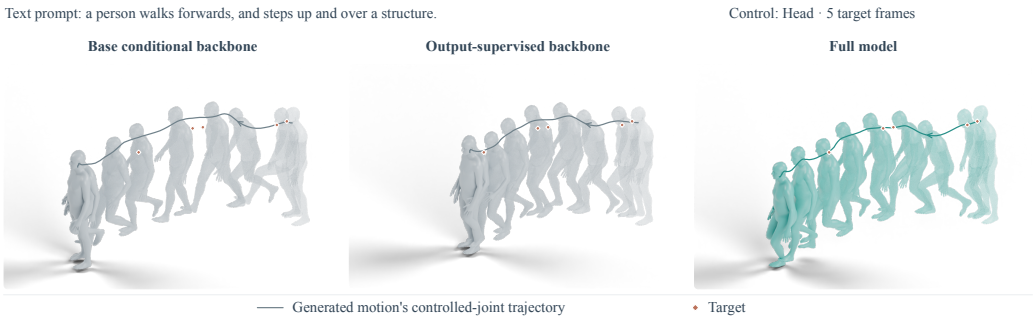

Figure S1: Effect of output supervision and geometric feedback on sparse head control. All variants use the same text, five head targets, initial noise, sampling budget, and guidance.

S3.2 Feedback Calibration after Observer Replacement

The observer-replacement and feedback-calibration variants in Tab. S5 start from the FD-trained generator in block b. Replacing the observer while keeping the candidate generator and feedback modules fixed reduces mean error from 10.28 to 9.07 cm with little change in FID. Calibrating the feedback modules to the new proposal source further improves both metrics, yielding the full model's 0.063 FID and 8.41 cm error.

S3.3 Effect of the Supervision Representation

We compare T2TM and TMR as the feature representation for FD supervision on pelvis and random-joint control (Tab. S6). The pelvis comparison uses joint-objective feedback training, while the random-joint comparison uses distributional adaptation before observer replacement. Across both tasks, TMR yields slightly lower FID and less foot skating, while T2TM achieves lower joint-position errors. We use T2TM as the default for its balance of generation quality and control accuracy across these tasks.

Table S6: Effect of the feature representation used for FD supervision.

FD features	FID↓	R@3↑	Div.→	Skate↓	Traj. err. (50 cm) (%)↓	Loc. err. (50 cm) (%)↓	Avg. err. (cm)↓
Pelvis control, 2 steps							
T2TM (Guo et al., 2022a)	0.100	0.820	9.600	0.1027	4.54	1.65	10.33
TMR (Petrovich et al., 2023)	0.097	0.812	9.494	0.0859	7.29	3.06	12.18
Random-joint control, 2 steps							
T2TM	0.083	0.810	9.461	0.1006	3.36	0.87	10.28
TMR	0.077	0.805	9.365	0.0815	6.94	1.82	12.76

S3.4 Alternative Feedback Training Strategies

We also enable distributional supervision from the start of feedback learning. Tab. S7 compares this joint-objective strategy with the default staged strategy at two and four steps. Both achieve competitive generation and control performance, with different strengths: joint-objective training yields lower FID, while staged training produces lower mean joint-position errors and fewer large target deviations. Joint-objective training is therefore an effective alternative; we retain staged training as the default for its stronger control accuracy.

Table S7: Joint-objective and staged feedback training for random-joint control. Within each strategy, the two- and four-step results use the same checkpoint.

Training	Steps	FID↓	R@3↑	Skate↓	Traj. err. (50 cm) (%)↓	Loc. err. (50 cm) (%)↓	Avg. err. (cm)↓
Staged (default)	2	0.0633	0.8116	0.0905	2.25	0.59	8.41
Joint-objective	2	0.0409	0.8050	0.0931	4.38	0.89	9.79
Staged (default)	4	0.0581	0.8226	0.1008	5.14	1.36	9.76
Joint-objective	4	0.0413	0.8245	0.0955	7.06	1.52	10.78

S4 Sampling and Inference Analysis

S4.1 Control Sampling Budgets

We vary the sampling budget of each final control model while keeping its weights and guidance fixed. For initial-motion control (Tab. S16), one update reaches 2.63 cm mean error. A second

update substantially improves FID and text matching while slightly reducing the error; four steps mainly improve FID, with similar joint accuracy and higher latency.

For pelvis-only and random-joint control (Tab. S8), two updates yield lower mean error and fewer large deviations, whereas four improve retrieval and slightly reduce FID. Across these settings, the additional generation budget primarily improves distributional quality and text matching. We use two steps as the default balance of control accuracy and latency.

Table S8: Effect of sampling budget on sparse joint control. Model weights, guidance ($\omega = 8$), and the one-step observer are fixed within each setting.

Setting	Steps	FID↓	R@3↑	Skate↓	Traj. err. (50 cm) (%)↓	Loc. err. (50 cm) (%)↓	Avg. err. (cm)↓	AITS (s)↓
Pelvis-only	2	0.110	0.818	0.1031	2.85	1.02	7.91	0.0644
Pelvis-only	4	0.093	0.824	0.0906	4.13	1.57	8.59	0.1068
Random-joint	2	0.063	0.812	0.0905	2.25	0.59	8.41	0.0612
Random-joint	4	0.058	0.823	0.1008	5.14	1.36	9.76	0.1057

S4.2 Guidance Strength

Tab. S9 varies guidance for both candidate and observer while fixing the networks, two candidate steps, and observer-step allocation $[1, 1]$. Stronger guidance improves FID but increases geometric residuals in the sparse-joint settings. For random-joint control, increasing ω from 1 to 15 changes FID from 0.150 to 0.048 and mean error from 6.58 to 9.36 cm. Performance changes smoothly around the default $\omega = 8$. Initial-motion error stays within 2.48–2.57 cm across the range, with FID and text retrieval improving as guidance increases.

S4.3 Proposal Refresh during Sampling

Tab. S10 compares initial-proposal reuse with per-step proposal refresh at two and four candidate steps. Each pair fixes the generator, observer, controller, guidance, and candidate grid; both policies execute the controller at every sampling step. With four steps, reusing the initial proposal gives 33.42 cm mean error, compared with 9.76 cm for per-step proposal refresh. Increasing the candidate budget therefore does not compensate for stale geometry. Refreshed proposals improve target following at both budgets, despite much smaller differences in FID.

S4.4 Allocating Observer Computation

At sampling step k, an observer integrates from the candidate time t_k to zero in n_k^{obs} steps on a scaled internal grid. We vary $\mathbf{n}^{\mathrm{obs}} = (n_0^{\mathrm{obs}}, \ldots, n_{K-1}^{\mathrm{obs}})$ while fixing the candidate grid, weights, and guidance. Each resulting proposal is decoded once.

With two candidate steps (Tab. S11), allocations $[2, 1]$ and $[1, 2]$ both take 0.0679 s, but their mean errors are 7.40 and 13.60 cm. Allocating the extra evaluation to the first proposal is more effective than allocating it to the second.

The four-step results show the same ordering (Tab. S12): an extra early observer step, $[2, 1, 1, 1]$, yields 9.57 cm error, compared with 11.81 cm for $[1, 1, 1, 2]$. Using more observer integration steps for every proposal lowers FID but worsens control relative to the all-one allocation. The default uses one step per proposal for low latency; targeted early observation offers an additional accuracy option.

Table S9: Effect of guidance strength with two sampling steps and one observer step per proposal.

ω	FID↓	R@3↑	Div.→	Skate↓	Traj. err. (50 cm) (%)↓	Loc. err. (50 cm) (%)↓	Avg. err. (cm)↓
Pelvis-only							
1	0.138	0.791	9.686	0.0986	1.33	0.45	6.08
4	0.135	0.817	9.704	0.1011	2.11	0.77	7.11
7	0.113	0.818	9.643	0.1027	2.70	0.96	7.75
8	0.110	0.818	9.627	0.1031	2.85	1.02	7.91
9	0.108	0.817	9.615	0.1035	2.98	1.08	8.03
11	0.105	0.815	9.595	0.1040	3.15	1.13	8.20
15	0.103	0.814	9.580	0.1045	3.37	1.22	8.42
Initial-motion control							
1	0.116	0.777	9.648	0.0817	0.43	0.02	2.48
4	0.097	0.796	9.624	0.0843	0.41	0.02	2.50
7	0.084	0.800	9.601	0.0860	0.43	0.02	2.53
8	0.081	0.800	9.595	0.0863	0.43	0.02	2.54
9	0.080	0.801	9.590	0.0866	0.42	0.02	2.55
11	0.078	0.801	9.582	0.0871	0.41	0.02	2.56
15	0.076	0.801	9.572	0.0876	0.41	0.02	2.57
Sparse joint control							
1	0.150	0.793	9.565	0.0888	1.20	0.35	6.58
4	0.102	0.810	9.570	0.0915	1.66	0.48	7.56
7	0.069	0.810	9.507	0.0909	2.10	0.58	8.23
8	0.063	0.812	9.492	0.0905	2.25	0.59	8.41
9	0.058	0.811	9.477	0.0903	2.40	0.62	8.59
11	0.052	0.809	9.455	0.0898	2.74	0.69	8.91
15	0.048	0.807	9.425	0.0895	3.37	0.88	9.36

Table S10: Effect of proposal refresh during sampling for random-joint control. Model weights are fixed, and the controller runs at every sampling step under both policies.

Steps	Refresh policy	FID↓	R@3↑	Div.→	Skate↓	Traj. err. (50 cm) (%)↓	Loc. err. (50 cm) (%)↓	Avg. err. (cm)↓
2	Initial reuse	0.069	0.817	9.592	0.0823	27.70	12.83	25.67
2	Per-step refresh	0.063	0.812	9.492	0.0905	2.25	0.59	8.41
4	Initial reuse	0.076	0.828	9.691	0.0902	38.46	20.32	33.42
4	Per-step refresh	0.058	0.823	9.543	0.1008	5.14	1.36	9.76

Table S11: Observer-step allocation for random-joint control with two candidate steps. Each entry of $\mathbf{n}^{\mathrm{obs}}$ specifies the number of observer integration steps at the corresponding sampling step.

Observer steps $\mathbf{n}^{\mathrm{obs}}$	FID↓	R@3↑	Div.→	Skate↓	Traj. err. (50 cm) (%)↓	Loc. err. (50 cm) (%)↓	Avg. err. (cm)↓	AITS (s)↓
$[1, 1]$	0.063	0.812	9.492	0.0905	2.25	0.59	8.41	0.0612
$[2, 1]$	0.059	0.807	9.488	0.0882	1.73	0.39	7.40	0.0679
$[1, 2]$	0.059	0.806	9.440	0.0954	8.80	3.49	13.60	0.0679
$[2, 2]$	0.055	0.801	9.421	0.0935	9.35	3.83	13.91	0.0722

Table S12: Observer-step allocation for random-joint control with four candidate steps. Each entry of $\mathbf{n}^{\text{obs}}$ specifies the number of observer integration steps at the corresponding sampling step.

Observer steps $\mathbf{n}^{\text{obs}}$	FID↓	R@3↑	Div.→	Skate↓	Traj. err. (50 cm) (%)↓	Loc. err. (50 cm) (%)↓	Avg. err. (cm)↓	AITS (s)↓
[1, 1, 1, 1]	0.058	0.823	9.543	0.1008	5.14	1.36	9.76	0.1057
[2, 1, 1, 1]	0.057	0.823	9.545	0.1005	5.04	1.28	9.57	0.1131
[1, 1, 1, 2]	0.056	0.821	9.526	0.1027	8.39	2.58	11.81	0.1122
[2, 2, 2, 1]	0.057	0.823	9.544	0.1005	4.23	1.03	9.12	0.1266
[2, 2, 2, 2]	0.054	0.822	9.521	0.1028	8.21	2.57	11.67	0.1304

S5 Additional Benchmark Results

S5.1 Extended Text-to-Motion Comparison

Tab. S13 extends the main text-to-motion comparison with additional methods and sampling budgets. It reports generation, text matching, and diversity metrics together with the available inference times.

Fig. S2 compares text-to-motion generation across varied actions. MoGeFi generates natural-looking poses that reflect the described actions, while maintaining inference latency close to MotionLCM-V2 and substantially below MoMask. These examples complement the quantitative results, illustrating the combination of motion quality and efficient generation.

Table S13: Complete text-to-motion comparison on HumanML3D. → indicates that closer to real data is better.

Method	AITS (s)↓	R@1↑	FID↓	MM Dist↓	Div.→	MMod.↑
Real	–	0.511	0.002	2.974	9.503	–
T2M (Guo et al., 2022a)	<u>0.0223</u>	0.457	1.067	3.340	9.188	2.090
TM2T (Guo et al., 2022b)	0.0234	0.424	1.501	3.467	8.589	<u>2.424</u>
MDM (Tevet et al., 2023)	5.9452	0.320	0.544	5.566	9.559	**2.799**
MLD (Chen et al., 2023)	0.1415	0.481	0.473	3.196	9.724	2.413
T2M-GPT (Zhang et al., 2023a)	0.2165	0.492	0.141	3.121	9.722	1.831
ReMoDiffuse (Zhang et al., 2023b)	0.2741	0.510	0.103	2.974	9.018	1.795
MotionDiffuse (Zhang et al., 2024)	7.8458	0.491	0.630	3.113	9.410	1.553
MoMask (Guo et al., 2024)	0.1068	0.521	0.045	2.958	9.675	1.241
StableMoFusion (Huang et al., 2024)	0.1560	**0.553**	0.098	2.770	9.748	1.774
MotionCLR (Chen et al., 2024)	0.2562	0.542	0.099	2.981	9.846	2.145
MSQ (Wang et al., 2025)	–	0.483	0.063	3.094	<u>9.497</u>	1.290
MotionFlow (1 step) (Dong et al., 2026)	–	0.517	0.333	2.947	9.965	1.879
MotionFlow (2 steps) (Dong et al., 2026)	–	0.515	0.315	2.943	9.917	1.961
MotionFlow (3 steps) (Dong et al., 2026)	–	0.521	0.294	2.918	9.926	1.907
TCA-T2M (Wang et al., 2026)	–	0.517	0.068	2.951	9.485	1.103
MotionGPT3 (Flow) (Ban et al., 2026)	–	0.544	0.192	2.837	9.340	2.366
MotionLCM (1 step) (Dai et al., 2024b)	**0.0217**	0.502	0.467	3.022	9.631	2.172
MotionLCM (2 steps) (Dai et al., 2024b)	0.0246	0.505	0.368	2.986	9.640	2.187
MotionLCM (4 steps) (Dai et al., 2024b)	0.0306	0.502	0.304	3.012	9.607	2.259
MotionLCM-V2 (1 step) (Dai et al., 2024a)	0.0227	0.546	0.072	<u>2.767</u>	9.577	1.858
MotionLCM-V2 (2 steps) (Dai et al., 2024a)	0.0256	<u>0.551</u>	0.049	**2.765**	9.584	1.833
MotionLCM-V2 (4 steps) (Dai et al., 2024a)	0.0330	**0.553**	0.056	2.773	9.598	1.758
MoGeFi (1 step)	0.0226	0.515	**0.018**	2.919	9.405	2.313
MoGeFi (2 steps)	0.0279	0.530	<u>0.021</u>	2.862	**9.507**	2.186
MoGeFi (4 steps)	0.0384	0.531	0.024	2.859	9.532	2.128

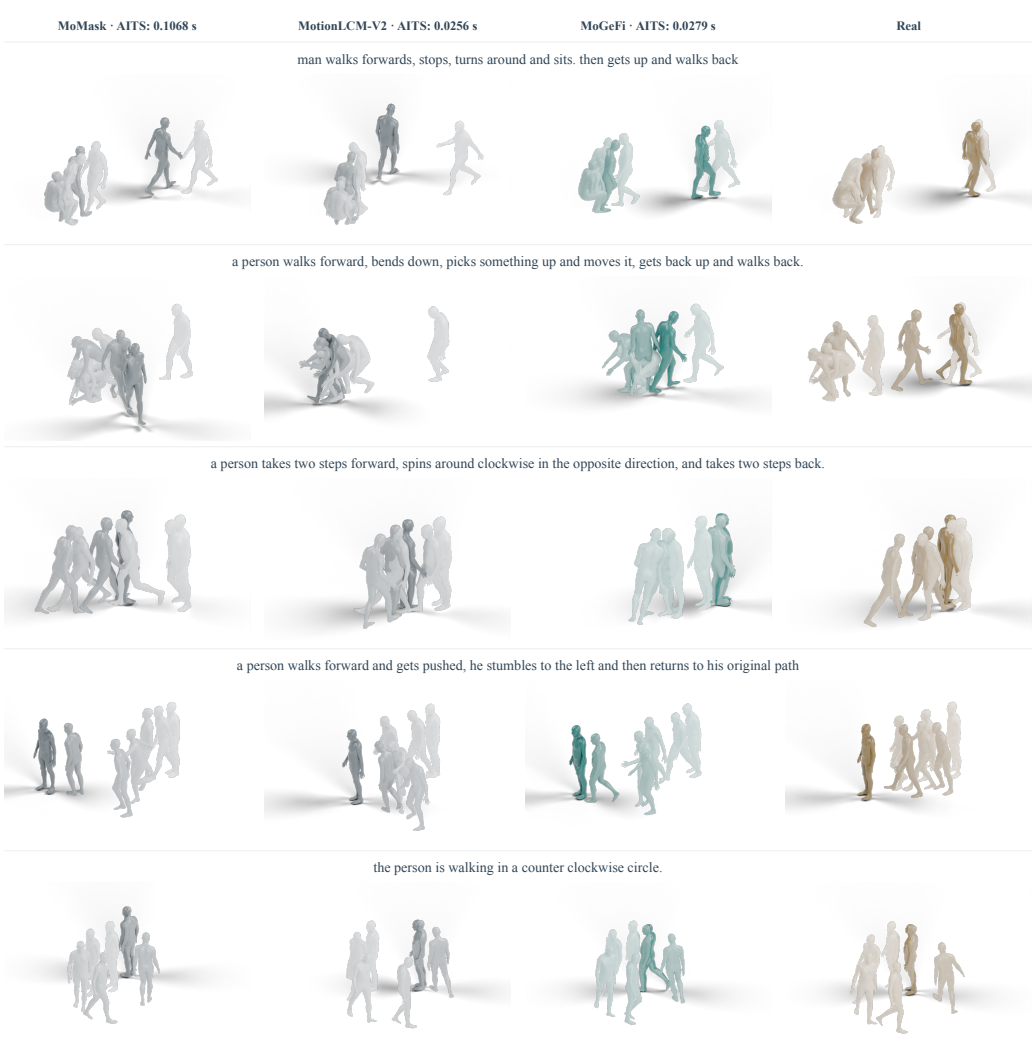

Figure S2: Qualitative comparison of text-to-motion generation. Poses progress from light to dark; Real denotes the reference motion from the dataset.

S5.2 Per-Joint Sparse Control Comparison

Tab. S14 compares performance for each controlled joint. Our two-step model achieves lower FID and mean joint-position error than the one-, two-, and four-step MotionLCM-V2 variants for every joint. It also yields the lowest trajectory and location error rates across all six joints among the compared methods. Relative to MaskControl-Fast, FID is lower for the head and wrists, while MaskControl-Fast has lower FID for the pelvis and feet and lower mean joint-position errors throughout.

Fig. S3 compares sparse joint control across different joints and actions. MoGeFi follows the specified targets more closely than MotionLCM-V2 in the displayed examples while retaining full-body poses consistent with the text. Together with the reported inference times, these comparisons illustrate accurate, text-consistent control at substantially lower latency than OmniControl and MaskControl-Fast.

S5.3 Pelvis-Only Control

We evaluate the pelvis-only model using the control protocol of OmniControl and MaskControl. Relative to MotionLCM-V2 ControlNet, our two-step model reduces FID from 0.393 to 0.110 and

Table S14: Per-joint sparse control on HumanML3D. Models are trained for control of all six joints and evaluated on one joint at a time. Average denotes the arithmetic mean of the six per-joint results.

Method	Joint	R@3↑	FID↓	Div.→	Skate↓	Traj. err. (50 cm) (%)↓	Loc. err. (50 cm) (%)↓	Avg. err. (cm)↓
OmniControl (Xie et al., 2024)	Pelvis	0.691	0.322	9.545	0.0571	4.04	0.85	**3.67**
MaskControl-Fast (Pinyoanuntapong et al., 2025)		0.806	**0.067**	9.453	**0.0552**	4.46	1.51	6.91
MotionLCM-V2 (1 step) (Dai et al., 2024a)		0.765	0.715	9.410	0.1183	12.83	6.16	17.60
MotionLCM-V2 (2 steps)		0.788	0.449	9.632	0.0870	14.46	7.14	18.18
MotionLCM-V2 (4 steps)		0.796	0.426	9.763	0.0778	14.58	7.44	18.34
MoGeFi (2 steps)		**0.811**	0.115	9.567	0.1108	**1.51**	**0.52**	7.16
OmniControl	Left foot	0.696	0.280	9.553	0.0692	5.94	0.94	**3.14**
MaskControl-Fast		0.806	**0.074**	9.450	**0.0561**	4.95	1.05	4.84
MotionLCM-V2 (1 step)		0.760	0.752	9.389	0.1056	20.29	8.58	21.15
MotionLCM-V2 (2 steps)		0.785	0.460	9.627	0.0839	23.72	10.41	22.23
MotionLCM-V2 (4 steps)		0.789	0.428	9.777	0.0762	24.35	10.76	22.58
MoGeFi (2 steps)		**0.814**	0.086	9.498	0.0938	**2.77**	**0.66**	8.16
OmniControl	Right foot	0.701	0.319	9.481	0.0668	6.66	1.20	**3.34**
MaskControl-Fast		0.808	**0.069**	9.416	**0.0566**	4.53	0.99	4.69
MotionLCM-V2 (1 step)		0.759	0.732	9.397	0.1046	20.33	8.67	21.12
MotionLCM-V2 (2 steps)		0.785	0.465	9.649	0.0839	23.37	10.36	22.17
MotionLCM-V2 (4 steps)		0.794	0.423	9.766	0.0765	24.31	11.01	22.60
MoGeFi (2 steps)		**0.811**	0.081	9.515	0.0981	**2.78**	**0.61**	7.96
OmniControl	Head	0.696	0.335	9.480	0.0556	4.22	0.79	**3.49**
MaskControl-Fast		0.810	0.080	9.411	**0.0555**	5.25	1.48	6.65
MotionLCM-V2 (1 step)		0.766	0.683	9.417	0.1170	16.43	7.74	20.63
MotionLCM-V2 (2 steps)		0.788	0.424	9.631	0.0868	18.28	8.82	21.33
MotionLCM-V2 (4 steps)		0.796	0.401	9.762	0.0781	18.89	9.24	21.63
MoGeFi (2 steps)		**0.811**	**0.056**	9.503	0.0673	**1.69**	**0.53**	8.06
OmniControl	Left wrist	0.680	0.304	9.436	0.0562	8.01	1.34	**5.29**
MaskControl-Fast		**0.809**	0.085	9.380	**0.0545**	4.67	1.08	5.34
MotionLCM-V2 (1 step)		0.768	0.638	9.448	0.1189	34.19	15.99	29.71
MotionLCM-V2 (2 steps)		0.793	0.384	9.648	0.0886	38.46	18.31	30.96
MotionLCM-V2 (4 steps)		0.802	0.370	9.774	0.0798	39.51	19.12	31.39
MoGeFi (2 steps)		0.803	**0.077**	9.359	0.0807	**2.71**	**0.61**	9.56
OmniControl	Right wrist	0.692	0.299	9.519	0.0601	8.13	1.27	**5.19**
MaskControl-Fast		**0.807**	0.095	9.387	**0.0549**	4.98	1.13	5.38
MotionLCM-V2 (1 step)		0.770	0.632	9.470	0.1195	34.78	16.35	29.78
MotionLCM-V2 (2 steps)		0.790	0.382	9.677	0.0895	38.61	18.45	31.09
MotionLCM-V2 (4 steps)		0.798	0.367	9.781	0.0798	39.98	19.61	31.65
MoGeFi (2 steps)		0.805	**0.069**	9.395	0.0897	**2.83**	**0.68**	9.58
OmniControl	Average	0.693	0.310	9.502	0.0608	6.17	1.07	**4.04**
MaskControl-Fast		0.808	**0.079**	9.416	**0.0555**	4.81	1.21	5.63
MotionLCM-V2 (1 step)		0.765	0.692	9.422	0.1140	23.14	10.58	23.33
MotionLCM-V2 (2 steps)		0.788	0.427	9.644	0.0866	26.15	12.25	24.32
MotionLCM-V2 (4 steps)		0.796	0.403	9.770	0.0780	26.93	12.86	24.70
MoGeFi (2 steps)		**0.809**	0.081	9.473	0.0901	**2.38**	**0.60**	8.41

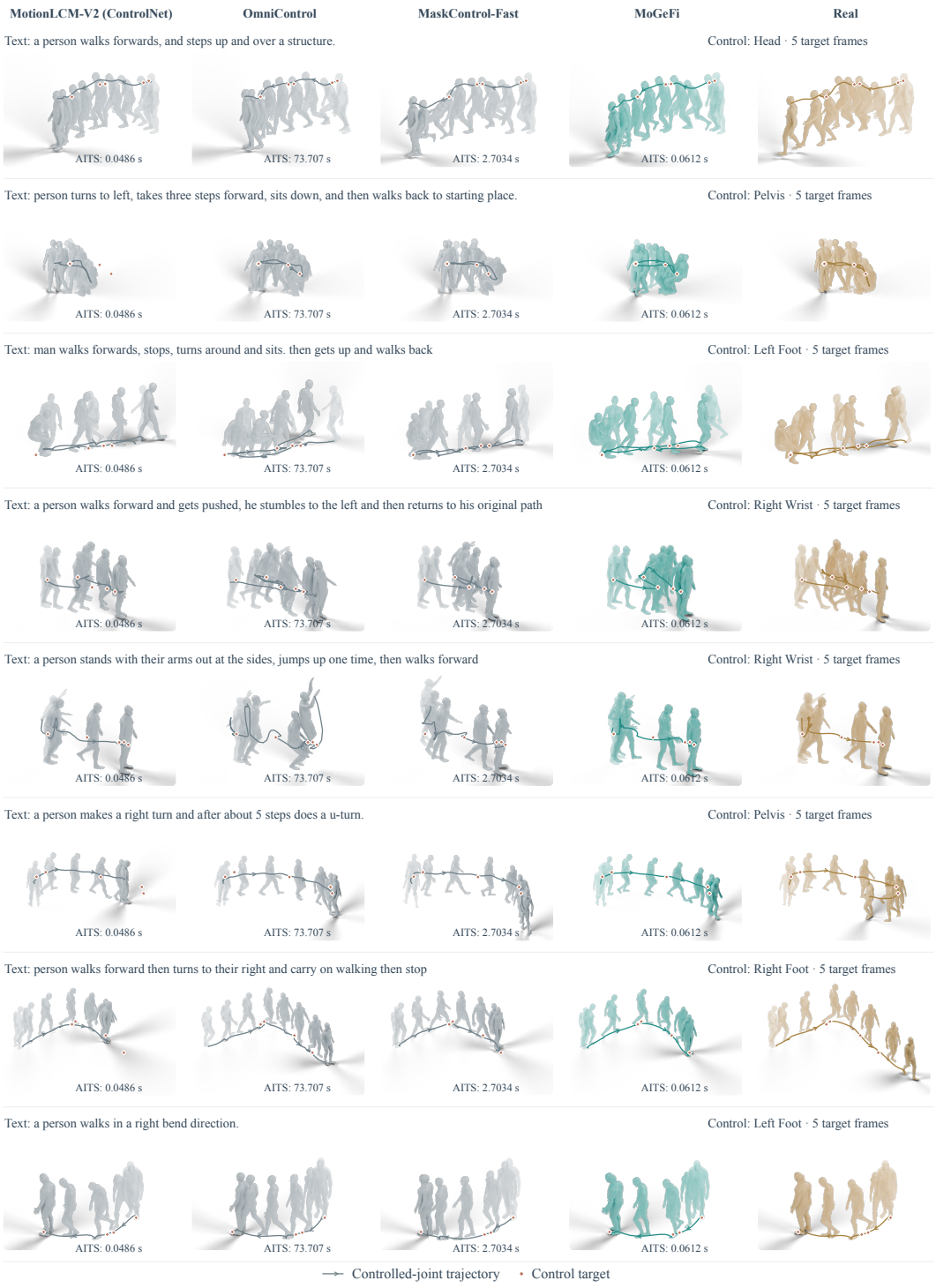

Figure S3: Qualitative comparison of sparse joint control with five targets per sequence. Real denotes the reference motion from the dataset.

mean error from 13.86 to 7.91 cm, with higher text retrieval (Tab. S15). These improvements in generation and root placement are obtained with a forward sampler taking 0.0644 s per motion. MaskControl's optimized configurations achieve smaller residuals at substantially higher inference cost.

Table S15: Pelvis-only control on HumanML3D.

Method	AITS (s)↓	R@3↑	FID↓	Div.→	Skate↓	Traj. err. (50 cm) (%)↓	Loc. err. (50 cm) (%)↓	Avg. err. (cm)↓
MDM (Tevet et al., 2023)	–	0.602	0.698	9.197	0.1019	40.22	30.76	59.59
GMD (Karunratanakul et al., 2023)	58.436	0.665	0.576	9.206	0.1009	9.31	3.21	14.39
PriorMDM (Shafir et al., 2024)	4.6274	0.583	0.475	9.156	0.0897	34.57	21.32	44.17
OmniControl (Xie et al., 2024)	73.707	0.687	0.218	9.422	**0.0547**	3.87	0.96	3.38
MaskControl (no optimization) (Pinyoanuntapong et al., 2025)	0.1953	0.802	0.128	–	0.0594	39.14	24.00	40.41
MaskControl-Fast (Pinyoanuntapong et al., 2025)	2.7034	0.808	**0.059**	9.444	0.0570	2.00	0.75	5.50
MaskControl-Medium (Pinyoanuntapong et al., 2025)	13.606	0.806	0.069	9.425	0.0568	0.05	**0.00**	1.24
MaskControl-Accurate (Pinyoanuntapong et al., 2025)	41.510	0.809	0.061	9.496	**0.0547**	**0.00**	**0.00**	**0.98**
MotionFlow (Dong et al., 2026)	–	0.765	0.364	9.928	0.0893	16.70	4.16	13.95
MotionFlow (+ latent optimization) (Dong et al., 2026)	–	0.774	0.360	9.944	0.0871	3.42	0.21	5.47
MotionLCM (Dai et al., 2024b)	–	0.752	0.531	9.253	–	18.87	7.69	18.97
MotionLCM-V2 (ControlNet) (Dai et al., 2024a)	**0.0486**	0.787	0.393	9.837	–	10.80	5.81	13.86
MotionLCM-V2 (CLT) (Dai et al., 2024a)	10.580	0.816	0.202	9.707	–	0.39	0.03	2.03
MoGeFi (2 steps)	0.0644	**0.818**	0.110	9.627	0.1031	2.85	1.02	7.91

S5.4 Extended Initial-Motion Comparison

Tab. S16 includes additional baseline configurations and our one-, two-, and four-step results for the six-joint prefix task. We analyze the effect of sampling budget in Sec. S4.1.

Table S16: Complete initial-motion control comparison on HumanML3D. Six joints are observed over the first 25% of each sequence.

Method	AITS (s)↓	FID↓	R@3↑	Traj. err. (50 cm) (%)↓	Loc. err. (50 cm) (%)↓	Avg. err. (cm)↓
OmniControl (Xie et al., 2024)	72.657	2.328	0.557	33.62	3.22	9.77
MaskControl-Fast (Pinyoanuntapong et al., 2025)	2.4738	**0.060**	**0.825**	0.28	0.07	4.69
MaskControl-Medium (Pinyoanuntapong et al., 2025)	14.124	0.061	0.824	**0.00**	**0.00**	1.54
MaskControl-Accurate (Pinyoanuntapong et al., 2025)	–	**0.060**	0.819	**0.00**	**0.00**	**1.42**
MotionLCM (1 step) (Dai et al., 2024b)	**0.0279**	0.419	0.756	19.88	1.47	11.27
MotionLCM (2 steps) (Dai et al., 2024b)	0.0343	0.397	0.759	19.60	1.43	10.92
MotionLCM (4 steps) (Dai et al., 2024b)	0.0462	0.444	0.753	20.89	1.72	11.40
MotionLCM-V2 (ControlNet, 1 step) (Dai et al., 2024a)	0.0288	0.300	0.789	2.74	0.23	5.05
MotionLCM-V2 (ControlNet, 2 steps) (Dai et al., 2024a)	0.0350	0.248	0.799	2.74	0.24	5.03
MotionLCM-V2 (ControlNet, 4 steps) (Dai et al., 2024a)	0.0493	0.236	0.802	2.77	0.24	5.04
MoGeFi (1 step)	0.0458	0.160	0.781	0.45	0.02	2.63
MoGeFi (2 steps)	0.0708	0.081	0.800	0.43	0.02	2.54
MoGeFi (4 steps)	0.1210	0.068	0.804	0.46	0.02	2.60

S5.5 Real-Robot Motion Transfer

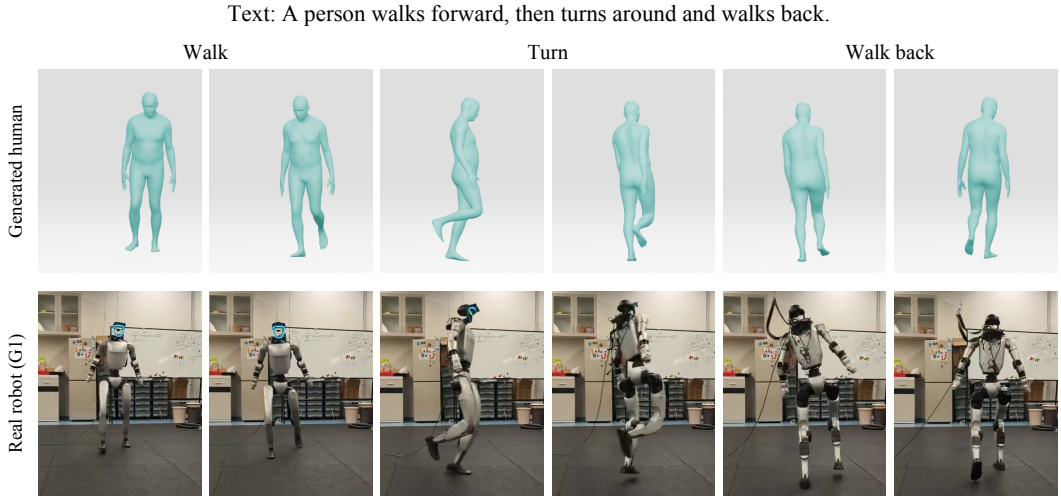

Figure S4: Walking and turning on a physical Unitree G1. Top: human motion generated by MoGeFi. Bottom: robot execution using SONIC's whole-body tracker.