NUMMOTION: PRECISE NUMERICAL CONTROL FOR 3D HUMAN MOTION GENERATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Existing generative 3D human motion models can generate realistic movements but struggle to meet numerical targets. This limits their use for controlled biomechanical comparisons, which require varying specific movement characteristics to prescribed values while minimizing unintended changes to other properties. We propose NumMotion, a lightweight adaptation framework for precise numerical control of movement characteristics in text-conditioned 3D human motion generation. NumMotion conditions a frozen generator on explicit numerical inputs through a compact adapter. Target fitting matches motion measurements to requested values, while Jacobian regularization encourages proportional responses to changed requests and penalizes unintended changes in other controlled properties. We introduce a benchmark based on NimblePhysics covering seven action categories, with two numerical controls per category, to evaluate simultaneous target success, cross-control interference, and motion quality. We demonstrate that NumMotion meets the requested numerical target in 94% of trials versus 8.8% for baseline MDM.

*A person walks **3 meters** forward while turning left by **45 degrees**.*



Baseline MDM
Distance: 1.89 m, turn: 83.01°

MDM+NumMotion
Distance: 2.96 m, turn: 46.59°

Figure 1: Walking 3 m with a 45° turn: baseline MDM and NumMotion.

1 INTRODUCTION

Text-conditioned 3D human motion models can generate diverse movements from descriptions such as “a person walks forward” or “a person performs a squat” (Tevet et al., 2023; Zhang et al., 2023; Guo et al., 2024). Many uses of these models, however, require a measurable property of the movement to take a specified value (Karunratanakul et al., 2023; Wan et al., 2024; Siy et al., 2026; Dai et al., 2026). A request to walk 3 m while turning left by 45° specifies both the action and two quantities that the generated motion must realize. Producing recognizable walking does not establish that either numerical requirement has been met. Figure 1 illustrates this distinction by comparing baseline MDM with its NumMotion adaptation.

Numerical control also requires the generated motion to respond predictably to changes in the requested values. If the requested walking distance increases while the turn angle remains fixed, the generated motion should cover the additional distance without changing the turn angle. We refer to unintended changes in other controlled movement properties as cross-control interference. This matters for controlled comparisons of movement because changing several properties at once

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.

can obscure the effect of the quantity a user intended to vary. Alignment of text with motion and distributional similarity assess correspondence with the text description (Petrovich et al., 2023) and resemblance to real motions (Heusel et al., 2017; Guo et al., 2022), respectively, but do not directly measure numerical accuracy or cross-control interference. Broad evaluations also identify fine grained instruction accuracy as a continuing difficulty (Yang et al., 2026). The problem therefore involves achieving requested numerical values and limiting cross-control interference while preserving motion quality. These requirements apply to feasible target combinations, since some movement properties cannot vary independently.

We propose NumMotion, a framework that trains numerical control in a pretrained motion generator through target fitting and Jacobian regularization. The pretrained generator remains fixed, while a compact adapter learns to incorporate requested movement quantities alongside the action description (cf. Shafir et al., 2024; Zhong et al., 2024). Target fitting measures each controlled property in a predicted motion and penalizes its difference from the requested value. After category specific adaptation, users provide an action description and explicit numerical targets to generate motion without a separate optimization procedure for each request. This provides a consistent interface for repeated variation of the predefined movement properties.

The central contribution of NumMotion is Jacobian regularization of the numerical control response. The response Jacobian describes how each measured movement property changes when a requested number changes. During training, we perturb one control at a time, encouraging the corresponding measurement to change by the requested amount while the other monitored property remains stable. Together with target fitting, this objective supervises both numerical accuracy and predictable responses to changing requests. Figure 2 summarizes the framework.

We evaluate NumMotion with a benchmark covering seven action categories and two numerical controls per category. We develop a kinematic verifier built on NimblePhysics (Werling et al., 2021) to measure the controlled numerical movement properties. The benchmark assesses numerical target success, cross-control interference, and motion quality. We compare DNO, OmniControl, and MaskControl with three pretrained generators and their NumMotion adaptations: MDM, T2M-GPT, and MoMask. These backbones represent three major text-to-motion generation paradigms: diffusion, autoregressive generation, and masked generation, respectively.

Our contributions are:

- **Jacobian regularization for numerical control.** We supervise the response of measured movement properties to requested quantities, encouraging proportional changes in the edited property and limiting cross-control interference. Combined with target fitting, this objective trains a compact numerical adapter.
- **Adaptation across three major paradigms.** We apply the same numerical adaptation framework to diffusion, autoregressive generation, and masked generation, evaluating MDM, T2M-GPT, and MoMask with separately trained adapters.
- **A measurable control benchmark.** We introduce seven action categories spanning locomotion, jumping, and actions of the upper and lower limbs, with explicit kinematic measurements of simultaneous target success, cross-control interference, and motion quality.

2 Related work

Text-to-motion generation. Text-to-motion models learn a generative prior over human movement from paired motion sequences and language descriptions (Petrovich et al., 2022; Tevet et al., 2022; Zhang et al., 2022; Chen et al., 2023). MDM performs diffusion directly on motion representations (Tevet et al., 2023). T2M-GPT generates discrete motion tokens autoregressively (Zhang et al., 2023), and MoMask combines masked token prediction with residual refinement (Guo et al., 2024). These approaches differ in both representation and generation procedure, but text conditioning alone does not guarantee that a requested numerical magnitude is realized (Goel et al., 2024; Huang et al., 2024; Shen et al., 2026). NumMotion conditions these pretrained generators on explicit quantities and supervises how their measured outputs respond to changes in those quantities.

Controllable motion generation. OmniControl combines spatial guidance and realism guidance to control joint positions at specified frames (Xie et al., 2024). Diffusion Noise Optimization (DNO)

optimizes a pretrained generator's initial noise by propagating gradients from motion space objectives through denoising (Karunratanakul et al., 2024; see also Liu et al., 2024, 2026). MaskControl introduces training time logits regularization and inference time logits optimization for masked motion models, using differentiable expectation sampling to accommodate discrete tokens (Pinyoanuntapong et al., 2025). DNO and MaskControl support differentiable objectives beyond prescribed joint trajectories and are therefore relevant baselines for scalar numerical constraints. NumMotion learns numerical control during adaptation, using Jacobian regularization to shape the response of motion measurements to changes in requested quantities. The resulting adapter conditions generation without a separate optimization loop for each request (cf. Dai et al., 2024; Siy et al., 2026).

Derivative supervision and numerical adaptation. Sobolev training supervises both function values and derivatives (Czarnecki et al., 2017), while orthogonal Jacobian regularization separates output changes induced by different latent dimensions (Wei et al., 2021). NumMotion regularizes the Jacobian from explicit numerical requests to measured movement properties. Its diagonal and off-diagonal penalties encourage proportional responses to edited requests and stability of the other monitored quantities. We implement numerical conditioning through FiLM feature modulation (Perez et al., 2018) and conditional low rank updates based on LoRA (Hu et al., 2022).

Evaluation and physical control. T2MBench evaluates 14 generators on 1,025 out-of-distribution prompts and identifies fine grained instruction accuracy as a persistent limitation (Yang et al., 2026). NumMotion complements this broad evaluation with explicit measurements of target error and cross-control interference. CLoSD combines a diffusion planner with a tracking controller in a closed simulation loop to perform tasks conditioned on text and targets (Tevet et al., 2025). NumMotion instead adapts a motion prior to match scalar movement properties and regulate their measured response. Physical simulation is not part of its generation loop.

3 NUMMOTION

NumMotion learns both requested movement quantities and the response to changing them. Target fitting matches measured values, while Jacobian regularization encourages proportional changes in the edited property and suppresses cross-control interference. A compact adapter supplies numerical conditioning, and a fixed kinematic verifier provides the measurements used for supervision. Figure 2 distinguishes training from generation with the adapted model. Sections 3.1 through 3.3 describe MDM. Section 3.4 explains the extension to other backbones.

3.1 NUMERICAL MOVEMENT CONDITIONS AND ADAPTER

Given a text action description s, numerical targets $c \in \mathbb{R}^K$, and control presence indicators $m \in \{0,1\}^K$, NumMotion generates a motion $X \in \mathbb{R}^{T\times d}$, where T is the number of frames and d the motion feature dimension. The generator has frozen parameters θ and trainable adapter parameters ϕ:

$$X = G_{\theta,\phi}(z, s, c, m). \tag{1}$$

Here z denotes generation randomness. The MDM implementation uses $K = 2$ controls and a separate adapter per category.

Numerical encoding. Normalized controls are represented by their scalar values and sine/cosine features, then combined with control presence indicators and passed through a small MLP to form a shared embedding e for the adapter. Presence indicators distinguish an omitted control from a requested zero. Omitted controls contribute no scalar or sine/cosine features. Implementation details are in Appendix B.

Feature modulation. Each transformer block projects e to its own scales and offsets. Featurewise linear modulation (FiLM, Perez et al., 2018) adjusts the block output a_ℓ:

$$[\gamma_\ell, \beta_\ell] = P_\ell e, \qquad h_\ell = (1 + \gamma_\ell) \odot a_\ell + \beta_\ell.$$

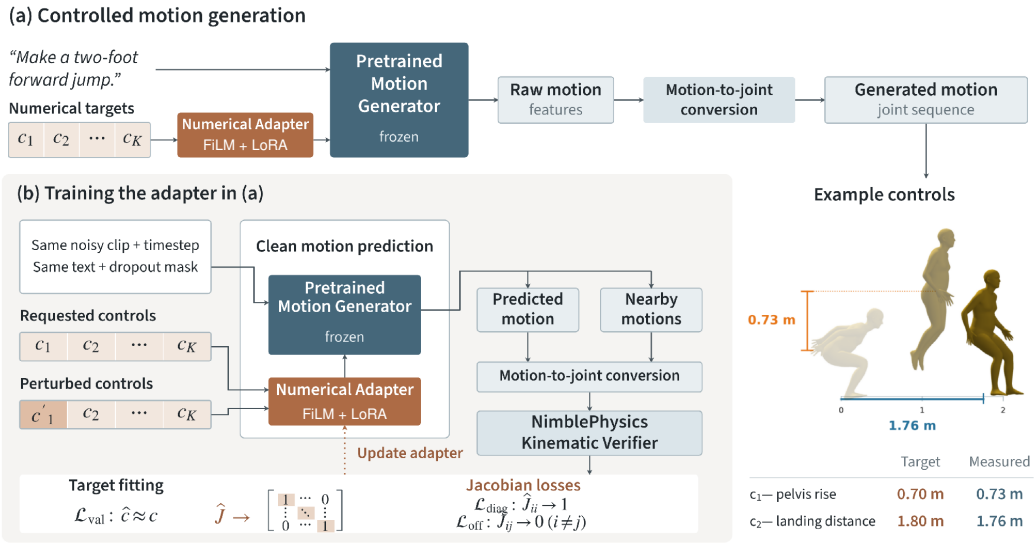


Figure 2: NumMotion generation (a) and Jacobian training (b). Matched perturbations encourage unit diagonal and zero off-diagonal responses. Target fitting uses the center request. The jump example compares requested and measured controls.

Here P_ℓ is a learned projection, γ_ℓ and β_ℓ are channelwise scale adjustments and offsets, h_ℓ is the modulated output, and $\odot$ denotes elementwise multiplication. The projections start at zero, so FiLM initially leaves the block output unchanged.

Conditional low rank updates. We also add rank-8 low rank adaptation (LoRA, Hu et al., 2022) to each block's attention output projection and both feedforward projections. For projection p, the effective weight is

$$W_{\text{eff},p} = W_{\text{frozen},p} + \frac{\alpha}{r} B_p C_p(c,t) A_p, \qquad r = 8.$$

The factors A_p and B_p are learned, and α sets the update scale. A router produces the r $\times$ r matrix C_p from the controls, diffusion timestep t, and projection location p. Appendix B gives the dimensions and configuration.

We train only the numerical encoder, FiLM projections, LoRA factors, and router. The generator, CLIP text encoder (Radford et al., 2021), verifier, and reference skeleton remain fixed.

3.2 Target fitting in measurement space

Each training clip provides an action description and two numerical targets measured from the source motion. At a randomly sampled diffusion timestep, we add noise to the clip and ask the adapted MDM to predict its clean motion. A fixed conversion from motion features to joints D maps the prediction to 3D joint positions, and the NimblePhysics verifier (Werling et al., 2021) measures the controlled properties. Write these measurements as $y \in \mathbb{R}^K$, where y_i corresponds to requested value c_i. Target fitting minimizes their scaled squared difference:

$$L_{val} = \sum_{i=1}^{K} m_i w_i \left(\frac{d_i\left(y_i, c_i\right)}{\sigma_i} \right)^2. \tag{2}$$

Here m_i selects the supplied controls, w_i weights each property, and $d_i\left(y_i, c_i\right)$ is the signed measurement error: measured minus requested, with angular differences wrapped where required. The training scales σ_i are 0.10 m for lengths and 10° for angles. These scales balance properties with different units and are separate from input normalization and evaluation tolerances. The squared penalty remains active inside the evaluation tolerance.

3.3 NUMERICAL RESPONSE REGULARIZATION

Target fitting constrains measured values at individual requests. Jacobian regularization also constrains their local response to control changes (cf. Peebles et al., 2020; Wei et al., 2021; Rhodes and Lee, 2021). For each training example, we evaluate the original two controls and positive and negative perturbations of each control, giving five clean motion predictions. All five use the same model weights, noisy motion, timestep, text, and dropout mask. Only the numerical adapter inputs change. This shared state isolates the response to the perturbed numerical control.

For a perturbation of control j by δ_j, let $y_i^{(j,+)}$ and $y_i^{(j,-)}$ be property i measured from predictions with that control increased and decreased, respectively. Each entry of the response matrix $\widehat{J} \in \mathbb{R}^{K\times K}$ estimates how property i changes with control j:

$$\widehat{J}_{ij} = \frac{d_i\left(y_i^{(j,+)}, y_i^{(j,-)}\right)}{2\delta_j}. \tag{3}$$

The diagonal penalty L_{diag} encourages $\widehat{J}_{ii} = 1$: changing a requested value should produce the same change in its measurement. The off-diagonal penalty L_{off} encourages $\widehat{J}_{ij} = 0$ for $i \neq j$: changing one control should leave the other monitored property stable.

Perturbations are adjusted near geometric boundaries, with one sided differences when needed. These constraints apply locally to feasible changes in the monitored properties.

3.4 TRAINING OBJECTIVE

The training objective combines target fitting with diagonal and off-diagonal response penalties:

$$L_{num} = 3.0L_{val} + 0.1L_{diag} + 0.1L_{off}. \tag{4}$$

In MDM, these losses supervise clean motion predictions at sampled diffusion timesteps, without differentiating through the complete inference trajectory. At inference, the requested controls condition the adapted model throughout ordinary sampling. The target fitting and response losses are not optimized for each request. Appendix B gives implementation details and the training algorithm.

Extension across model families. T2M-GPT and MoMask (Zhang et al., 2023; Guo et al., 2024) use the same numerical encoding, FiLM, and conditional rank-8 adaptation, with separate adapters for each backbone and category. T2M-GPT is adapted throughout its autoregressive transformer. MoMask is adapted in both its masked and residual token transformers. Their numerical losses supervise complete generated motions. Hard tokens are used in the forward pass, while a straight through Gumbel softmax estimator supplies gradients through generation and the frozen motion decoder (Jang et al., 2017). Matched requests share sampling noise when estimating the response Jacobian. The pretrained text encoders, transformers, and motion autoencoders remain frozen. Appendix B specifies the token model training paths and configurations.

4 A BENCHMARK FOR NUMERICAL MOTION CONTROL

4.1 TASKS AND MOTION REPRESENTATION

The benchmark contains seven categories: walking with turning, forward jump and landing, squat jump, kick, punch, reach, and throw. We select these categories to cover common movement patterns with distinct control demands: locomotion and turning, takeoff and landing, lower limb striking, and upper limb reaching, striking, and throwing. Their paired controls span whole body displacement and orientation, joint configuration, and limb placement at defined movement phases. Each category has two numerical controls, such as pelvis rise and landing distance for a forward jump. Appendix A gives the control definitions and data provenance.

Nonwalking adaptation episodes are drawn from HumanML3D (Guo et al., 2022). The T2M-GPT and MoMask walking corpora also include preexisting teacher trajectories generated by MDM. Motions use a canonical 22 joint representation at 20 frames per second. Source labels, selected

limbs, and event frames refer to the same extracted movement. Original recordings and their mirrored versions are grouped for adaptation splits. This does not imply that those recordings were absent from backbone pretraining.

4.2 Kinematic measurements and event validity

NimblePhysics provides articulated kinematics for the benchmark (Werling et al., 2021). For the token model implementations, nonwalking measurements use an articulated output with a fixed reference, while walking uses raw decoded joints in metres. Training and evaluation use the declared mapping for each category. These measurements do not certify physical feasibility. Appendix B specifies the output conversion.

A landing measurement uses the supported landing after the intended flight. Squat depth is measured during preparation, rather than during an airborne tuck. Reach and strike controls refer to the selected limb and intended outward phase. Missing or ambiguous required events are recorded as failures. Additional action checks assess complete phases, recovery, repeated strokes, and support.

4.3 Evaluation metrics

We report simultaneous target success for both controls. A request succeeds only when every required measurement is defined and each absolute error is at most 0.10 m for length controls or 10° for angular controls. Main tables report this success rate for both controls.

We measure numerical response by changing one requested value while holding the other control, text, and generation randomness fixed. Cross-talk denotes cross-control interference: changes in the control that was held fixed. Gain error measures deviation from a one-to-one response in the edited control. We aggregate absolute errors to avoid cancellation. FID assesses distributional quality (Heusel et al., 2017; Guo et al., 2022). We omit R-precision because requests within a category share identical captions and differ only in numerical targets, making caption based retrieval uninformative for this comparison. Penetration depth, contact sliding, and foot-skating rate characterize kinematic quality (Yuan et al., 2023). Quality metrics use all generated samples.

5 Experiments

5.1 Experimental setup

We evaluate three pretrained backbones (MDM, T2M-GPT, and MoMask) (Tevet et al., 2023; Zhang et al., 2023; Guo et al., 2024) and their NumMotion adaptations. A separate adapter is trained for each backbone and category. DNO, MaskControl, and OmniControl (Karunratanakul et al., 2024; Pinyoanuntapong et al., 2025; Xie et al., 2024) provide complementary controllable generation baselines. Each unadapted backbone receives the action and numerical request in text. DNO uses objectives corresponding to the numerical measurements, and OmniControl receives spatial constraints constructed from the request. MaskControl uses its supported control objective and logits optimization.

The comparison uses matched action descriptions, numerical requests, measurement definitions, and success thresholds. Within each backbone, random seeds are paired across conditions and ablations.

5.2 Numerical accuracy across categories and backbones

Table 1 compares simultaneous target success across seven categories and three generation paradigms. Each baseline and its NumMotion adaptation receive matched requests, so the paired comparison tests the effect of numerical conditioning within a fixed generator family. Success requires both controls to fall within their tolerances. Satisfying only the easier control therefore does not count as a successful sample.

Table 1: Success on both controls (%) across seven categories, using 0.10 m and 10° tolerances. Mean averages categories. Higher is better.

Method	Mean ↑	Walk	Forward jump	Squat jump	Kick	Punch	Reach	Throw
DNO	45%	100%	29%	35%	37%	38%	51%	25%
MaskControl	63%	100%	57%	65%	12%	64%	86%	57%
OmniControl	3%	2%	0%	0%	3%	5%	7%	4%
MDM	9%	2%	0%	0%	0%	14%	29%	17%
MDM+NumMotion	94%	100%	100%	100%	89%	86%	100%	83%
T2M-GPT	7%	2%	0%	0%	4%	12%	19%	12%
T2M-GPT+NumMotion	59%	94%	76%	82%	37%	43%	51%	30%
MoMask	11%	2%	0%	0%	0%	17%	36%	22%
MoMask+NumMotion	88%	98%	86%	95%	84%	82%	91%	80%

The category breakdown distinguishes performance on locomotion, jumping, and individual limb actions. The mean gives each category equal weight, while the individual columns expose variation that a single aggregate can obscure. Table 2 complements target success with response error and motion quality, separating agreement at a requested value from the behavior under changes in that request.

Table 2: Response error, motion quality, and inference time. Lower is better.

Method	Gain error ↓	FID ↓	Penetration (mm) ↓	Contact sliding (cm/s) ↓	Foot-skating (%) ↓	Inference (s/motion) ↓
Real reference	N/A	N/A	0.015	7.60	6.41	N/A
DNO	N/A	5.49	0.031	12.31	7.19	117.11
MaskControl	0.017	20.55	0.017	15.49	9.52	45.92
OmniControl	0.781	46.63	1.047	10.14	12.72	386.19
MDM	0.975	17.99	0.001	5.25	2.80	0.062
MDM+NumMotion	0.018	16.31	0.190	7.79	4.18	0.443
T2M-GPT	1.176	18.50	0.051	16.26	27.29	0.1203
T2M-GPT+NumMotion	1.090	16.41	0.018	16.54	26.69	0.3156
MoMask	1.051	12.23	0	13.93	5.93	0.132
MoMask+NumMotion	0.805	15.06	0.002	12.02	17.41	0.559

5.3 Component ablations on MDM

Table 3 isolates the contribution of Jacobian regularization on MDM and examines the adapter components. The variant without Jacobian regularization uses the numerical encoder, FiLM, rank-8 conditional LoRA, and target fitting. Only the diagonal and off-diagonal penalties are removed. We compare it with full NumMotion under matched data, training budgets, checkpoint selection, and inference settings. The remaining variant removes conditional LoRA.

These comparisons address different questions. Removing Jacobian regularization tests whether fitting requested values alone also produces proportional responses when a control changes. Removing LoRA tests the contribution of conditional weight updates alongside FiLM modulation. Gain error measures response scaling, whereas simultaneous success measures agreement with a particular request. Reading both metrics together distinguishes local response behavior from pointwise target fitting.

Table 3: MDM ablations averaged across seven categories.

Variant	Success on both controls (%) ↑	Gain error ↓
Full NumMotion	94%	0.018
Without Jacobian regularization	64%	0.944
Without LoRA	69%	0.580

5.4 Motion quality and qualitative comparisons

Table 2 evaluates motion quality over all generated samples, including numerical failures. FID measures distributional similarity, while penetration, contact sliding, and foot-skating describe distinct kinematic artifacts. These quantities complement numerical accuracy because a motion can satisfy its requested measurements while exhibiting undesirable contact or movement patterns.

Figures 3 and 4 show punch and kick with NumMotion across three backbones and with DNO and MaskControl, respectively. The pose sequences show how each motion approaches and leaves the measured configuration. This temporal context matters: matching an extension, angle, or height alone does not establish a recognizable action. The examples provide qualitative context for the numerical measurements rather than an estimate of action validity across the full evaluation set.



Figure 3: NumMotion punch (upper row) and kick (lower row) across three backbones. Labels give requested and achieved controls.

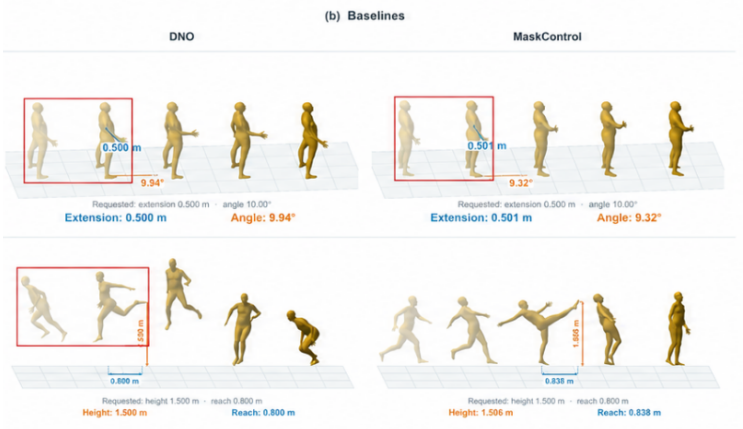


Figure 4: DNO and MaskControl on the same punch and kick requests. Sequences contextualize the measured configurations.

6 Discussion and limitations

Jacobian regularization trains the relationship between changing a numerical request and changing the resulting motion measurement. The matched MDM ablation tests the contribution of response penalties beyond target fitting. Comparisons across the three backbones assess predictable numerical control and motion quality. Adapter training incurs an upfront cost that can be reused across subsequent numerical requests.

Feasibility and measurement scope. Plausible individual control ranges do not guarantee feasible combinations. Success on the selected kinematic properties does not establish dynamic feasibility, causal isolation, or clinical benefit. Other properties, such as trunk posture, may change.

Training and coverage. The experiments cover seven categories with two controls each and three major generation paradigms. MDM response supervision uses individual clean motion predictions, whereas token models train on complete generations using biased straight through gradients. Extending the method to more controls increases the finite difference cost and requires suitable measurements.

7 Conclusion

NumMotion adapts pretrained motion generators for explicit numerical requests through target fitting and measurement space Jacobian regularization. Its central objective supervises how measured movement properties respond to control changes, encouraging proportional responses while limiting cross-control interference. We evaluate the framework across seven action categories and three major text-to-motion generation paradigms: diffusion, autoregressive generation, and masked generation.

AI use statement

AI tools assisted with prose editing, language refinement, and manuscript formatting.

Reproducibility statement

Sections 3 and 4 specify the method, measurements, and success thresholds. Appendix A provides control ranges and grid spacing. Appendix B describes adapter configurations and training settings for the token models. Code is not available with this draft.

References

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, 2023. URL https://arxiv.org/abs/2212.04048.

Wojciech M. Czarnecki, Simon Osindero, Max Jaderberg, Grzegorz Swirszcz, and Razvan Pascanu. Sobolev training for neural networks. In *Advances in Neural Information Processing Systems*, volume 30, 2017. URL https://arxiv.org/abs/1706.04859.

Minyue Dai, Ke Fan, Anyi Rao, Jingbo Wang, and Bo Dai. Controllable Text-to-Motion Generation via Modular Body-Part Phase Control. arXiv preprint arXiv:2603.19795, 2026. URL https://arxiv.org/abs/2603.19795.

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, 2024. URL https://arxiv.org/abs/2404.19759.

Purvi Goel, Kuan-Chieh Wang, C. Karen Liu, and Kayvon Fatahalian. Iterative Motion Editing with Natural Language. In ACM SIGGRAPH Conference Papers, 2024. URL https://arxiv.org/abs/2312.11538.

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*, 2022. URL https://openaccess.thecvf.com/content/CVPR2022/html/Guo_Generating_Diverse_and_Natural_3D_Human_Motions_From_Text_CVPR_2022_paper.html.

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*, 2024. URL https://arxiv.org/abs/2312.00063.

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. URL https://proceedings.neurips.cc/paper/2017/hash/8a1d694707eb0fefe65871369074926d-Abstract.html.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-Rank Adaptation of Large Language Models. In International Conference on Learning Representations, 2022. URL https://openreview.net/forum?id=nZeVKeeFYf9.

Yiming Huang, Weilin Wan, Yue Yang, Chris Callison-Burch, Mark Yatskar, and Lingjie Liu. CoMo: Controllable Motion Generation through Language Guided Pose Code Editing. In European Conference on Computer Vision, 2024. URL https://www.ecva.net/papers/eccv_2024/papers_ECCV/html/4203_ECCV_2024_paper.php.

Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with Gumbel-Softmax. In International Conference on Learning Representations, 2017. URL https://openreview.net/forum?id=rkE3y85ee.

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, 2023. URL https://arxiv.org/abs/2305.12577.

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*, 2024. URL https://arxiv.org/abs/2312.11994.

Hanchao Liu, Xiaohang Zhan, Shaoli Huang, Tai-Jiang Mu, and Ying Shan. Programmable Motion Generation for Open-Set Motion Control Tasks. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2024. URL https://arxiv.org/abs/2405.19283.

Hanchao Liu, Fang-Lue Zhang, Shining Zhang, Tai-Jiang Mu, and Shi-Min Hu. Towards Highly-Constrained Human Motion Generation with Retrieval-Guided Diffusion Noise Optimization. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2026. URL https://arxiv.org/abs/2605.08054.

William Peebles, John Peebles, Jun-Yan Zhu, Alexei A. Efros, and Antonio Torralba. The Hessian Penalty: A Weak Prior for Unsupervised Disentanglement. In European Conference on Computer Vision, 2020. URL https://arxiv.org/abs/2008.10599.

Ethan Perez, Florian Strub, Harm de Vries, Vincent Dumoulin, and Aaron Courville. FiLM: Visual reasoning with a general conditioning layer. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2018. URL https://arxiv.org/abs/1709.07871.

Mathis Petrovich, Michael J. Black, and Gül Varol. TEMOS: Generating Diverse Human Motions from Textual Descriptions. In European Conference on Computer Vision, 2022. URL https://mathis.petrovich.fr/temos/.

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, 2023. URL https://arxiv.org/abs/2305.00976.

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, 2025. URL https://arxiv.org/abs/2410.10780.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning Transferable Visual Models From Natural Language Supervision. In International Conference on Machine Learning, 2021. URL https://proceedings.mlr.press/v139/radford21a.html.

Travers Rhodes and Daniel D. Lee. Local Disentanglement in Variational Auto-Encoders Using Jacobian L1 Regularization. In Advances in Neural Information Processing Systems, volume 34, 2021. URL https://proceedings.neurips.cc/paper/2021/hash/bfd2308e9e75263970f8079115edebbd-Abstract.html.

Yonatan Shafir, Guy Tevet, Roy Kapon, and Amit H. Bermano. Human Motion Diffusion as a Generative Prior. In International Conference on Learning Representations, 2024. URL https://openreview.net/forum?id=dTpbEdN9kr.

Keming Shen, Bizhu Wu, Junliang Chen, Xiaoqin Wang, and Linlin Shen. FineXtrol: Controllable Motion Generation via Fine-Grained Text. In Proceedings of the AAAI Conference on Artificial Intelligence, volume 40, 2026. URL https://ojs.aaai.org/index.php/AAAI/article/view/37845.

Joshua Siy, Huakun Liu, Yutaro Hirao, Monica Perusquia-Hernandez, Hideaki Uchiyama, and Kiyoshi Kiyokawa. EMA: Effort Metric Attention for Anatomical Effort-Guided Human Motion Diffusion. In IEEE International Conference on Automatic Face and Gesture Recognition, 2026. URL https://arxiv.org/abs/2605.24566.

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, 2022. URL https://arxiv.org/abs/2203.08063.

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. URL https://openreview.net/forum?id=SJ1kSyO2jwu.

Guy Tevet, Sigal Raab, Setareh Cohan, Daniele Reda, Zhengyi Luo, Xue Bin Peng, Amit H. Bermano, and Michiel van de Panne. CLoSD: Closing the loop between simulation and diffusion for multi-task character control. In International Conference on Learning Representations, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/file/7319b7561ffe5e2f6419acd4a2f52d6b-Paper-Conference.pdf.

Weilin Wan, Zhiyang Dou, Taku Komura, Wenping Wang, Dinesh Jayaraman, and Lingjie Liu. TLControl: Trajectory and Language Control for Human Motion Synthesis. In European Conference on Computer Vision, 2024. URL https://www.ecva.net/papers/eccv_2024/papers_ECCV/html/5370_ECCV_2024_paper.php.

Yuxiang Wei, Yupeng Shi, Xiao Liu, Zhilong Ji, Yuan Gao, Zhongqin Wu, and Wangmeng Zuo. Orthogonal jacobian regularization for unsupervised disentanglement in image generation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 6721–6730, 2021. URL https://arxiv.org/abs/2108.07668.

Keenon Werling, Dalton Omens, Jeongseok Lee, Ioannis Exarchos, and C. Karen Liu. Fast and feature-complete differentiable physics engine for articulated rigid bodies with contact constraints. In *Robotics: Science and Systems*, 2021. URL https://www.roboticsproceedings.org/rss17/p034.html.

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. URL https://arxiv.org/abs/2310.08580.

Bin Yang, Rong Ou, Weisheng Xu, Jiaqi Xiong, Xintao Li, Taowen Wang, Luyu Zhu, Xu Jiang, Jing Tan, and Renjing Xu. T2MBench: A benchmark for out-of-distribution text-to-motion generation. *arXiv preprint arXiv:2602.13751*, 2026. URL https://arxiv.org/abs/2602.13751.

Ye Yuan, Jiaming Song, Umar Iqbal, Arash Vahdat, and Jan Kautz. PhysDiff: Physics-Guided Human Motion Diffusion Model. In Proceedings of the IEEE/CVF International Conference on Computer Vision, 2023. URL https://arxiv.org/abs/2212.02500.

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, 2023. URL https://openaccess.thecvf.com/content/CVPR2023/html/Zhang_Generating_Human_Motion_From_Textual_Descriptions_With_Discrete_Representations_CVPR_2023_paper.html.

Mingyuan Zhang, Zhongang Cai, Liang Pan, Fangzhou Hong, Xinying Guo, Lei Yang, and Ziwei Liu. MotionDiffuse: Text-Driven Human Motion Generation with Diffusion Model. arXiv preprint arXiv:2208.15001, 2022. URL https://arxiv.org/abs/2208.15001.

Lei Zhong, Yiming Xie, Varun Jampani, Deqing Sun, and Huaizu Jiang. SMooDi: Stylized Motion Diffusion Model. In European Conference on Computer Vision, 2024. URL https://arxiv.org/abs/2407.12783.

A BENCHMARK TASKS AND MEASUREMENT DEFINITIONS

Table A1 defines the seven task categories. Our T2M-GPT and MoMask implementations measure paired controls at the same selected action episode and anatomical side. Throwing describes wrist motion during a standing overarm throw. Its controls do not measure an object trajectory. Evaluation uses absolute error thresholds of 0.10 m for length controls and 10° for angles.

Table A1: Numerical controls and target ranges by category.

Category	Control 1	Control 2	Target ranges
Walking	Signed forward displacement (m)	Unwrapped heading change ($^\circ$)	Control 1: 0.5 to 3.0 m Control 2: -90° to $+90^\circ$
Jump and landing	Pelvis rise from setup (m)	Forward landing displacement (m)	Control 1: 0.2 to 0.7 m Control 2: 0.1 to 1.80 m
Squat jump	Pre takeoff mean knee flexion ($^\circ$)	Pelvis rise from setup (m)	Control 1: 50° to 120° Control 2: 0.1 to 0.8 m
Kick	Foot height above floor (m)	Horizontal pelvis to foot reach (m)	Control 1: 0.4 to 1.5 m Control 2: 0.4 to 0.9 m
Punch	Shoulder to wrist extension (m)	Torso relative punch azimuth ($^\circ$)	Control 1: 0.3 to 0.5 m Control 2: -40° to $+40^\circ$
Reach	Wrist height relative to shoulder (m)	Shoulder to wrist distance (m)	Control 1: -0.1 to 0.4 m Control 2: 0.3 to 0.5 m
Throw	Wrist height relative to shoulder (m)	Shoulder to wrist distance (m)	Control 1: 0.1 to 0.4 m Control 2: 0.3 to 0.5 m

Grid spacing is 0.1 m for length controls and 10° for angular controls. Infeasible throw and reach combinations are excluded.

Walking displacement and heading compare median initial and final five frame quantities. Distance is signed displacement along the initial facing direction, not accumulated path length. Forward jump distance uses the detected landing, and pelvis rise is measured from supported setup to the selected apex. Squat flexion is measured before takeoff. Punch azimuth uses a circular residual, while walking heading remains unwrapped and knee flexion is a bounded angle.

Adaptation uses action episodes extracted from HumanML3D. The token model walking corpus also includes preexisting teacher trajectories generated by MDM. Real recordings and their mirrored versions share a recording level split, while teacher trajectories use their seed group split. The source manifests identify the recordings, selected episodes, and assigned splits.

B IMPLEMENTATION AND TRAINING DETAILS

B.1 ADAPTER CONFIGURATION

Each numerical control is divided by its fixed normalization scale and clipped to its configured range. For a normalized value u_i, the scalar feature vector is

$$\psi(u_i) = [u_i,\ \sin(\pi u_i), \cos(\pi u_i), \sin(2\pi u_i), \cos(2\pi u_i),\\ \sin(4\pi u_i), \cos(4\pi u_i)].$$

The input concatenates $m_1\psi(u_1)$, $m_2\psi(u_2)$, and the two control presence indicators, giving 16 features. Features for an omitted control are zeroed before concatenation. Input normalization scales, target fitting scales, and evaluation tolerances serve different purposes and are recorded separately.

At each block, the same FiLM scale and offset vectors (Perez et al., 2018) are applied at every sequence position, including the conditioning token in MDM. The numerical encoder does not receive the diffusion timestep, so these coefficients remain fixed for a given request while MDM continues to use its existing timestep conditioning. Gradients propagate through the frozen computations to the adapter.

Notation. The shared embedding $e \in \mathbb{R}^{256}$ is projected separately at each transformer block. For a block width H and N sequence positions, block features are $N \times H$ matrices, FiLM scale and offset

are H dimensional vectors, and the bias free projection has shape $2H \times 256$. For a frozen projection with input width I and output width O, the LoRA factors A and B have shapes $r \times I$ and $O \times r$, and the conditional matrix C has shape $r \times r$, with r = 8. Subscripts identify the block or projection location.

Algorithm: NumMotion adapter training on MDM

Require: Category data, frozen pretrained generator, and numerical adapter parameters ϕ.

1: For each training step:

2: Sample a clip, action text, controls, timestep, and diffusion noise.

3: Form the original request and feasible positive and negative perturbations of each control.

4: Predict clean motions using the same noisy clip, timestep, text, and dropout mask.

5: Convert predictions to joints and measure the controls with NimblePhysics.

6: Compute target fitting and estimate the response Jacobian by finite differences.

7: Penalize deviations of J_{ii} from 1 and J_{ij} from 0 for $i \neq j$.

8: Update only the numerical encoder, FiLM, and conditional LoRA using the training objective.

9: End for.

10: Return the category specific adapter.

Target fitting uses the original request, while the perturbed requests supply the response estimates. Perturbations and finite differences respect the declared geometric boundaries. Angular differences wrap. Losses are averaged over the minibatch.

B.2 Adaptation of other generators

T2M-GPT and MoMask adapt the same types of transformer projections, but train numerical control through their complete token generation procedures. Table B1 summarizes the gradient paths. Both keep CLIP, native transformer weights, motion encoders and decoders, and codebooks frozen. Only the numerical encoder, FiLM heads, low rank factors, and router are learned.

Table B1: Backbone representations and adapter gradient paths.

Backbone	Generation family	Adapter sites / gradient path
MDM	Diffusion in motion space	FiLM and conditional LoRA, with clean prediction at one sampled noise level and differentiable joint conversion
T2M-GPT	Autoregressive discrete tokens	FiLM and rank-8 LoRA in all 18 blocks, with complete autoregressive generation and VQ decoding using straight through token gradients.
MoMask	Masked discrete tokens	FiLM and rank-8 LoRA in both token transformers, with 18 masked rounds, five residual levels, and RVQ decoding using straight through token gradients.

Token model adapters. T2M-GPT uses 18 blocks of width 1,024, with 11,288,160 trainable parameters per category. MoMask uses independent adapters in its eight block base and eight block residual transformers, each of width 384, totaling 3,766,528 parameters per category. Both use the 16 feature encoding and 16 $\rightarrow$ 256 $\rightarrow$ 256 SiLU MLP. Rank-8 updates affect attention outputs and both feedforward projections, with scale α = 16. The router uses control features, control presence, and a learned site embedding, without a diffusion timestep input.

Initialization and conditioning. The conditional router is a 33 $\rightarrow$ 64 $\rightarrow$ 64 $\rightarrow$ 64 MLP with SiLU hidden activations. Its outputs form an 8 $\times$ 8 core equal to the identity plus a bounded learned correction. Input factors use Kaiming uniform initialization, while output factors, FiLM heads, and the final router layer start at zero. Both adapters initially preserve the pretrained generator. Rows with no supplied controls bypass adaptation. Frozen modules remain in evaluation mode. Source

control dropout removes both controls together with probability 0.10, shared across all five response branches.

Discrete generation. Hard sampled token embeddings are used both for subsequent generation and for motion decoding. A biased straight through Gumbel softmax estimator supplies the backward path (Jang et al., 2017). It does not exactly differentiate discrete sampling or MoMask remasking decisions. No generation prefix is detached. Each central request and its four perturbations share text, duration, anatomical side, control presence, and sampling noise. T2M-GPT generates the requested token length autoregressively with EOS suppressed. MoMask runs 18 masked base rounds followed by five residual levels. Both decode and crop to the requested frame count.

Source and request supervision. Numerical losses measure complete motions generated for both source derived and manually specified requests. Manual requests have no paired source motion. Response penalties use measurement and control normalization, matching diagonal responses to one and off-diagonal responses to zero. Nonwalking boundary stencils use actual feasible spacings. Equation (4) specifies the training objective.

Training schedule. Each category is configured for 7,500 source updates followed by 2,500 updates mixing source examples and unpaired manual requests. Source batches contain 32 requests, while mixed batches contain 16 of each type with equal loss weights. Eight loss groups of four requests each are accumulated per update. The mixed phase restores the final source checkpoint and optimizer state.

Table B2: Token model training and sampling settings, shared unless specified.

Setting	Configuration
Optimizer	AdamW with a constant learning rate, weight decay 0.01, and gradient norm cap 1.0
Learning rate	0.00005 for nonwalking categories and 0.0001 for walking
Source / mixed phase seeds	20260914 / 2026092407
Sampling / relaxation / guidance	All 1.0, with MoMask top k threshold 0.9
Inference	One conditioned generation without target optimization or refinement
Training hardware	T2M-GPT: one A100 80 GB per category. MoMask: one A100 40 GB per category

Output conversion. Source features are converted from the stored normalization to each model's released VQ normalization (Zhang et al., 2023; Guo et al., 2024), and decoded features are converted back before measurement. A fixed unit conversion is applied once to recover joints in metres. Nonwalking outputs preserve decoded joint directions and root translation while using the fixed HumanML reference skeleton. Walking evaluation uses raw decoded joints. The numerical verifier uses a kinematic position map derived from Nimble with an equivalent float64 GPU gradient path. It does not simulate dynamics or contact forces.

B.3 Local interpretation

Let $F(c)$ collect the K measured properties obtained by predicting and decoding motion under fixed training inputs, and let J be its Jacobian with respect to the K requested controls. For a small feasible change Δc, the local approximation is $F(c+\Delta c) - F(c) \approx J\Delta c$. Diagonal entries near one produce proportional responses in the edited property, while off-diagonal entries near zero limit response in the other monitored property. This approximation is local and does not guarantee independent control over an arbitrary range. With central differences, K controls require $2K+1$ predictions per training example. The two control implementation uses five.