Optimizing Diffusion Noise Can Serve As Universal Motion Priors
Appendix

A. Motion completion

Table A.1 shows the results of the motion completion task. The task is evaluated under the same setting as the motion refinement task in the main paper, except that the ground truth joint locations are given without added noise. The goal of the task is to generate the full-body motion given partial joint observations.

The results are consistent with the motion denoising experiment where DNO's performance scale with base model, DNO-GMD outperforms other baselines regarding MPJPE and FID, while HuMoR tends to produce smoother motions.

Table A.1. Motion completion results on a subset of HumanML3D [3] dataset. All experiments were run with $N = 300$ except DNO-MLD* which runs with 1,000 optimization steps. FIDs are computed against *Real*. The *Real*'s FIDs are computed against a hold-out set from the dataset. HuMoR* means we exclude the sequence when its optimization fails.

	MPJPE ↓ observed (cm)	FID ↓	Foot ↓ skating ratio	Jitter ↓
Real	0.0	0.50	0.08	0.50
Six joints				
HuMoR*	8.7	1.53	0.13	**0.17**
GMD	31.1	7.08	0.08	0.79
DNO-MDM	8.5	1.31	0.07	0.33
DNO-MLD*	11.0	0.67	0.10	1.29
DNO-GMD	**6.6**	**0.30**	0.07	0.92
Eight joints				
HuMoR*	8.4	1.22	0.13	**0.17**
GMD	29.8	7.06	0.08	0.79
DNO-MDM	8.7	0.98	0.07	0.34
DNO-MLD*	11.3	0.51	0.11	1.30
DNO-GMD	**6.6**	**0.12**	0.08	0.93
Ten joints				
HuMoR*	8.3	1.06	0.12	**0.18**
GMD	28.4	6.88	0.08	0.79
DNO-MDM	8.6	0.80	0.07	0.36
DNO-MLD*	11.3	0.49	0.11	1.31
DNO-GMD	**6.5**	**0.11**	0.07	0.93

B. Additional motion-related tasks

Under the DNO framework, the same method presented in Algorithm 1 can be adapted to many motion-related tasks without retraining the model. In this section, we present different settings of DNO for motion blending and motion in-betweening tasks. The qualitative results are presented in our supplementary video.

B.1. Motion Blending.

For motion blending, the goal is to smoothly transition from one distinct action to another. The inputs are two motion sequences and the expected output is a long motion that combines the two input motions together. With DNO, the problem can be formulated in the same manner as the motion refinement and completion task (Sec. 5.2), where the joint locations of the concatenated input motions are used as targets and the optimization is initialized from a random $\mathbf{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$. To facilitate a smooth blending between motions, we define a 10-frame window around the concatenated frame as a transition period where we drop all target joints. Consequentially, the model needs to fill in this transition according to its motion prior. We set the content criterion $\lambda_{\text{cont}} = 0.0, \lambda_{\text{decorr}} = 10^3$, the perturbation amount $\gamma = 0$, and the optimization step to 1000 for this task.

B.2. Motion In-betweening.

For motion in-betweening, the inputs are the starting pose and the ending pose, given by the location of each joint. The goal is to generate the in-between motion according to those two poses. Similar to motion blending and motion completion, this task can be formulated as an optimization with partial observation as targets. We use the same setting as in the motion blending task with the only difference being the number of target joints.

C. Why we do not report FID for motion editing.

As the motion Fréchet inception distance (FID) is a measurement *between two data distributions*, it requires a large number of samples in both datasets [2]. For the motion editing task, only one motion sequence exists before editing and only a few sequences exist after editing, thus there are not enough data points to measure a meaningful FID.

D. GMD implementation details

To compare with GMD [5], we use the released model with Emphasis projection and Dense gradient propagation for all tasks. The trajectory model is not used. When conditioned on the ground locations, we use the provided point-to-point imputing method until $t = 20$ as suggested in their experiments. The guidance is provided using the same criterion terms used in our method for all tasks. As GMD does not support editing while preserving the content, in the editing task, we instead provided the text prompt together with the target condition as inputs for the motion editing task. The

observed joints are used without text conditioning for noisy motion refinement and motion completion.

E. HuMoR implementation details

We use the officially released version of HuMoR [8] which uses both the pose prior and motion prior for evaluations. We note that the released model is trained on a subset of the AMASS dataset [6] at 30 FPS which does not entirely overlap with the 20 FPS sequences in the HumanML3D dataset [3]. The HuMoR code accepts the FPS number and does its interpolation to match the input with its learned motion prior. We also noticed that HuMoR optimization fails on some sequences in the test set, resulting in NaN error. We removed those sequences when computing the metrics for HuMoR.

F. SDEdit on motion refinement

We include SDEdit [7] results on the motion refinement task in Tab. F.1. We tried all possible hyperparameters t in 100 increments from 100-1000. Except for the very extreme values of $t = 1000$, SDEdit exhibits unrealistic motions affected by the presence of noise in the original motion representation with high FID and Jitter. At $t = 1000$, SDEdit becomes a normal DDPM generative process, and no original content is preserved. In all cases, SDEdit fails to preserve the original content suggested by very high MPJPE. Note that the high FID of 29.73 for $t = 1000$ comes from the fact that the motions generated from MDM without any text prompts are heavily biased toward simple motions, e.g. standing, which do not capture the wide range of possible motions in the HumanML3D dataset. We conclude that SDEdit is not an effective motion refinement method.

Table F.1. SDEdit [7] results on the motion refinement task (noise std. = 5 cm.). We used the default number of repetitions $k = 3$ in all of the following experiments.

	MPJPE ↓ observed (cm)	FID ↓	Foot ↓ skating ratio	Jitter ↓
All joints				
Real	0.0	0.48	0.08	0.50
Noisy	11.4	58.82	0.66	28.61
SDEdit (t=100)	346.3	36.10	0.12	3.12
SDEdit (t=200)	313.6	33.92	0.14	1.61
SDEdit (t=300)	288.7	32.47	0.14	1.11
SDEdit (t=400)	259.4	31.24	0.13	1.01
SDEdit (t=500)	226.3	30.53	0.12	0.86
SDEdit (t=600)	187.2	28.99	0.10	0.93
SDEdit (t=700)	150.2	27.76	0.09	1.48
SDEdit (t=800)	122.7	30.83	0.08	2.35
SDEdit (t=900)	79.2	19.09	0.06	1.33
SDEdit (t=1000)	68.2	29.73	0.00	0.04

F.1. Qualitative Results

Please check our **supplementary video** for qualitative results from DNO in all tasks including motion editing, refinement, blending, and in-betweening.

G. Differences from guided diffusion method

While both DNO and loss-guided or classifier-guidance diffusion methods [1, 5, 9, 11] can be used to produce motion samples with specific guidance objectives, these processes are completely different.

The loss-guided or classifier-guidance diffusion method (LGD) is a *sampling technique* that uses the gradient of a loss function to steer the trajectory of the diffusion sampling. The process is done in one full-chain sampling and outputs a *sample* that follows the guidance.

In contrast, DNO is a *latent optimization technique* where each optimization step involves a full-chain diffusion sampling. The output is a *latent code* whose decoded sample follows the guidance.

The differences between DNO and LGD also have the following practical implications:

1) Latent optimization (DNO) does not have approximation error during guidance because it operates on the exact output $\mathbf{x}_0$ from solving the full-chain diffusion process via an ODE solver, while in LGD, the loss criterion $\mathcal{L}(\cdot)$ is *approximately* computed on an expected $\hat{x} = \mathbb{E}[\mathbf{x}_0|\mathbf{x}_t]$ as explained in Eq. 7, 8 of [9] and [11] as follows:

$$\begin{aligned}\mathcal{L}(\mathbf{x}_t) &= \mathbb{E}_{p(\mathbf{x}_0|\mathbf{x}_t)}\mathcal{L}(\mathbf{x}_0) \\ &\approx \mathcal{L}(\hat{\mathbf{x}})\end{aligned}$$

The approximation error is severe when $\mathrm{Var}[\mathbf{x}_0|\mathbf{x}_t]$ is large, particularly near the beginning where $T \sim 1000$. This means the guidance is only effective near the end of the denoising process. Empirically, we observe that GMD [5] does not reach the targets as well compared to DNO (25.7 vs 9.1 MPJPE, Table 2).

2) The latent space can serve as universal priors for valid motions. DNO can answer the question "What is the closest valid motion to the input x?" by optimizing latent x_T to produce a valid motion x_0 that best matches the input x. GMD, an LGD method, is ineffective at generating valid motions from noisy inputs as shown in the refinement task in Table 2.

3) LGD cannot easily preserve content (Tab. 1). As editing in LGD is equivalent to new conditional sampling with the input motion, it is not obvious how to specify what aspects of the input motion are to be preserved and how to preserve them with LGD.

Most recent developments in diffusion image editing operate on the latent noise space with the help of the conditional inversion process [4, 10]. This direction further bolsters the merits of DNO as a latent approach for content

preservation. The latent space naturally provides smooth transitions between valid motions; samples that are close in latent space x_T are also likely to be close in motion space x_0. DNO enables content-preserving editing through minimal updates on the latent space and results in a minimal change in the input motion to fulfill the objectives.

As shown in the experiments, DNO enables a wide range of tasks that require precise control, motion prior, or content preservation, which cannot be effectively solved with LGD.

References

[1] Prafulla Dhariwal and Alex Nichol. Diffusion models beat GANs on image synthesis. 2021. 2

[2] Chuan Guo, Xinxin Zuo, Sen Wang, Shihao Zou, Qingyao Sun, Annan Deng, Minglun Gong, and Li Cheng. Action2motion: Conditioned generation of 3d human motions. In *Proceedings of the 28th ACM International Conference on Multimedia*, pages 2021–2029, 2020. 1

[3] 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*, pages 5152–5161, 2022. 1, 2

[4] Inbar Huberman-Spiegelglas, Vladimir Kulikov, and Tomer Michaeli. An edit friendly DDPM noise space: Inversion and manipulations. 2023. 2

[5] 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 (ICCV)*, pages 2151–2162, 2023. 1, 2

[6] Naureen Mahmood, Nima Ghorbani, Nikolaus F Troje, Gerard Pons-Moll, and Michael J Black. Amass: Archive of motion capture as surface shapes. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 5442–5451, 2019. 2

[7] Chenlin Meng, Yang Song, Jiaming Song, Jiajun Wu, Jun-Yan Zhu, and Stefano Ermon. SDEdit: Image synthesis and editing with stochastic differential equations. 2021. 2

[8] Davis Rempe, Tolga Birdal, Aaron Hertzmann, Jimei Yang, Srinath Sridhar, and Leonidas J Guibas. Humor: 3d human motion model for robust pose estimation. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 11488–11499, 2021. 2

[9] Jiaming Song, Qinsheng Zhang, Hongxu Yin, Morteza Mardani, Ming-Yu Liu, Jan Kautz, Yongxin Chen, and Arash Vahdat. Loss-guided diffusion models for plug-and-play controllable generation. In *International Conference on Machine Learning*, pages 32483–32498. PMLR, 2023. 2

[10] Bram Wallace, Akash Gokul, and Nikhil Naik. EDICT: Exact diffusion inversion via coupled transformations. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2023. 2

[11] Jiwen Yu, Yinhuai Wang, Chen Zhao, Bernard Ghanem, and Jian Zhang. Freedom: Training-free energy-guided conditional diffusion model. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 23174–23184, 2023. 2