Taming Diffusion Models for Audio-Driven Co-Speech Gesture Generation ——Supplementary Material——

Lingting Zhu[1*] Xian Liu[2*] Xuanyu Liu[1] Rui Qian[2] Ziwei Liu[3] Lequan Yu[1†]
[1]The University of Hong Kong [2]The Chinese University of Hong Kong
[3]S-Lab, Nanyang Technological University
{ltzhu99, u3008631}@connect.hku.hk, lqyu@hku.hk,
{alvinliu, qr021}@ie.cuhk.edu.hk, ziwei.liu@ntu.edu.sg

Overview of Supplementary Material

In this supplementary material, we first provide the design and property of Diffusion Gesture Stabilizer in Section A. Then, we further show more implementation details in Section B. And some additional experiments are reported in Section C.

A. More Details of Diffusion Gesture Stabilizer

A.1. Design of Annealed Strategies

In denoising diffusion probabilistic models (DDPMs), we iteratively sample $\boldsymbol{x}_{t-1} \sim p_\theta(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t, \boldsymbol{c})$ at the inference as

$$\boldsymbol{x_{t-1}} = \frac{1}{\sqrt{\alpha_t}}\left(\boldsymbol{x_t} - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\hat{\boldsymbol{\epsilon}}_\theta\right) + \sigma_t \boldsymbol{z},$$

where $\boldsymbol{z} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$. At each timestep in the reverse process, $\boldsymbol{z}$ helps to increase the diversity of the generation.

However, the naive denoising strategy inevitably introduces a critical in our method. The dimensions of latent space consist of temporal dimension and pose dimension in our method to produce high-fidelity pose sequences in a non-autoregressive manner; thus, receiving the same treatment on two dimensions as on static image data can harm temporal consistency. In **DiffGesture**, we propose a Diffusion Gesture Stabilizer to eliminate the negative effect via constraining the temporal-unaware noise $\boldsymbol{z}$ at each timestep. We empirically find that given the trained diffusion model (with no extra training expense), with the Diffusion Gesture Stabilizer of the annealed strategies, DiffGesture produces better results, especially on those failure cases with clear temporal jittering ($\sim 5\%$, w/o Stabilizer).

In detail, our solution is to devise the temporal-aware noise $\boldsymbol{z}(t) = \{\boldsymbol{z}_i(t)\}_{i=1}^N$, where $\boldsymbol{z}_i(t) \in \mathbb{R}^C$. It is clear that $\boldsymbol{z}_i(t)$ independently follows $\mathcal{N}(\mathbf{0}, \boldsymbol{I})$ for $i \in \{1, \ldots, N\}$ and does not have relations with timestep t in the naive denoising strategy. A first attempt can be removing the temporal dimension of $\boldsymbol{z}(t)$ to make $\boldsymbol{z}_i(t) \equiv \boldsymbol{z}_0$ where $\boldsymbol{z}_0 \in \mathbb{R}^C$ follows $\mathcal{N}(\mathbf{0}, \boldsymbol{I})$. Thus, all the temporal variation is removed in each timestep t of the reverse process. However, this attempt fails due to an intuitive observation: values on a certain dimension tend to be similar after some denoising steps with the $\boldsymbol{z}_0$. As the diffusion model is trained with noisy data x_t with the predefined diffusion process, it fails to generalize well on a new noisy data domain and predict the noise from the noisy data of another noise prototype.

Therefore, we propose a **Thresholding** strategy where we remove the variation in the temporal dimension after denoising to a certain extent of a threshold timestep. In detail, we set a time threshold t_0, and then use the same $\boldsymbol{z} \in \mathbb{R}^{N\times C}$ in the naive sampling strategy for $t > t_0$ and set $\boldsymbol{z} = \{\boldsymbol{z}_0\}_{i=1}^N$ for $t \leq t_0$, where $z_0 \in \mathbb{R}^C$ follows $\mathcal{N}(\mathbf{0}, \boldsymbol{I})$ which do not introduce variation in the temporal dimension. Therefore, given the similar noisy data $\boldsymbol{x}_t$, the diffusion model could predict the noise, and $\boldsymbol{z}(t)$ would not introduce temporal variation when $t \leq t_0$.

Additionally, We further propose a smooth version of hard thresholding annealed strategy, **Smooth Sampling**. With Thresholding, the process is transitioned from high variance and entropy (hot) to low variance and entropy (cold) *at* the

*Equal contribution.
†Corresponding author.

threshold t_0, while in Smooth Sampling, we wish the transition is smooth. We first sample $\boldsymbol{z}_0(t) \sim \mathcal{N}(\mathbf{0}, \sigma_{a1}^2(t)\boldsymbol{I})$ only once for each timestep t in the reverse process, then given $\boldsymbol{z}_0(t)$, we sample $\boldsymbol{z}_i(t)|\boldsymbol{z}_0(t) \sim \mathcal{N}(\boldsymbol{z}_0(t), \sigma_{a2}^2(t)\boldsymbol{I})$ for $i \in \{1, \ldots, N\}$, where $\sigma_{a1}^2(t) + \sigma_{a2}^2(t) = 1$. Since given $\boldsymbol{z}_0(t)$, the group variance $\boldsymbol{z}(t) = \{\boldsymbol{z}_i(t)\}_{i=1}^N$ is controlled by $\sigma_{a2}(t)$, setting $\sigma_{a2}(t)$ a non-decreasing function (*i.e.*, setting $\sigma_{a1}(t)$ as a non-increasing function) temporal variance annealing is achieved, and therefore we can use the annealed sampling strategies smoothly controlling the trade-off between diversity and temporal consistency. Besides, we further prove that with our strategies, the sampled $\boldsymbol{z}_i(t)$ follows the standard Normal distribution, and the proof can be found in Section A.2. Obviously, Thresholding can be viewed as a special case of Smooth Sampling by setting $\sigma_{a2}(t)$ as a stepping function.

A.2. Property of Smooth Sampling

Property 1 *With the annealed noise sampling strategy of Smooth Sampling, the $\boldsymbol{z}_i(t)$ follows the standard Normal distribution $\mathcal{N}(\mathbf{0}, \boldsymbol{I})$.*

Proof 1 *According to Bayes' theorem,*

$$
\begin{aligned}
& p(\boldsymbol{z}_i(t)) = \int p(\boldsymbol{z}_i(t)|\boldsymbol{z}_0(t)) p(\boldsymbol{z}_0(t)) \mathrm{d}\boldsymbol{z}_0(t) \\
= \; & \frac{1}{2\pi\sigma_{a1}\sigma_{a2}} \int \exp\left(-\frac{\boldsymbol{z}_0^2(t)}{2\sigma_{a1}^2}\right) \exp\left(-\frac{(\boldsymbol{z}_0(t) - \boldsymbol{z}_i(t))^2}{2\sigma_{a2}^2}\right) \mathrm{d}\boldsymbol{z}_0(t) \\
= \; & \frac{1}{2\pi\sigma_{a1}\sigma_{a2}} \int \exp\left(-\frac{\sigma_{a1}^2\boldsymbol{z}_i^2(t) + \boldsymbol{z}_0^2(t) - 2\sigma_{a1}^2\boldsymbol{z}_0(t)\boldsymbol{z}_i(t)}{2\sigma_{a1}^2\sigma_{a2}^2}\right) \mathrm{d}\boldsymbol{z}_0(t) \\
= \; & \frac{1}{2\pi\sigma_{a1}\sigma_{a2}} \exp\left(-\frac{\sigma_{a1}^2\boldsymbol{z}_i^2(t) - (\sigma_{a1}^2\boldsymbol{z}_i(t))^2}{2\sigma_{a1}^2\sigma_{a2}^2}\right) \int \exp\left(-\frac{(\boldsymbol{z}_0(t) - \sigma_{a1}^2\boldsymbol{z}_i(t))^2}{2\sigma_{a1}^2\sigma_{a2}^2}\right) \mathrm{d}\boldsymbol{z}_0(t) \\
= \; & \frac{1}{2\pi\sigma_{a1}\sigma_{a2}} \exp\left(-\frac{(\sigma_{a1}^2 - \sigma_{a1}^4)\boldsymbol{z}_i^2(t)}{2\sigma_{a1}^2(1 - \sigma_{a1}^2)}\right) \int \exp\left(-\frac{(\boldsymbol{z}_0(t) - \sigma_{a1}^2\boldsymbol{z}_i(t))^2}{2\sigma_{a1}^2\sigma_{a2}^2}\right) \mathrm{d}\boldsymbol{z}_0(t) \\
= \; & \frac{1}{2\pi\sigma_{a1}\sigma_{a2}} \exp\left(-\frac{\boldsymbol{z}_i^2(t)}{2}\right) \cdot \sqrt{2\pi}\sigma_{a1}\sigma_{a2} \\
= \; & \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{\boldsymbol{z}_i^2(t)}{2}\right).
\end{aligned}
$$

Note that $\sigma_{a1}^2(t) + \sigma_{a2}^2(t) = 1$ holds and the integral of the normal distribution probability density function is 1.

B. More Implementation Details

Audio Encoder E_a. We extract audio features directly from raw audio data. We follow the implementations from [6]. The audio encoder consists of a series of Convolution, BatchNorm, and LeakyReLU. For the first Convolution, we set the stride to 5 and the padding to 1600. For the remaining Convolution, the stride is 6, and the padding is 0. For LeakyReLU, the negative slope is 0.3. All the detailed operations and the corresponding feature dimensions are shown in Table 1. As a result, 32-D feature vectors can be extracted, and the number of feature vectors (i.e., channel size of the feature map) is the same as the number of the motion frames.

Block	Shape	Operation
Input	1×36267	Unsqueeze
Conv Block-1	16×7891	Conv1d, BN, LeakyReLU
Conv Block-2	32×1313	Conv1d, BN, LeakyReLU
Conv Block-3	64×217	Conv1d, BN, LeakyReLU
Conv Block-4	32×34	Conv1d

Table 1. **Details of the operations in Audio Encoder and the shapes of audio feature maps.**

Diffusion Audio-Gesture Transformer. Our Transformer-based network consists of Linear Embedding, Transformer Blocks, and MLP. Through a linear layer FC (dim_{input}, dim_{hidden}), where dim_{input} and dim_{hidden} represent the input dimension

and the hidden dimension, we project the input data into a hidden space. We inject the time embedding to the transformer through expanding and concatenation, where the time embedding can be represented as $[\beta_t, \sin(\beta_t), \cos(\beta_t)]$ following [4]. Here the time information t is first represented in the variance β_t, then the basis of β_t are used. We concatenate the corrupted gesture x_t, the time embedding, and the context information (*i.e.,* the initial poses and audio features). Following former works [3, 6], the shape of the inial poses is $num_{pose} \times (dim_{pose} + 1)$, where num_{pose} denotes the number of ground truth poses, 34 for both two datasets, dim_{pose} denotes the dimension of poses, 27 for TED Gesture and 126 for TED Expressive, and an indicator logit is set to 1 for the use of initial poses. The initial poses are copied from the pseudo ground truth, and the unknown poses are set to zero. Therefore, $dim_{input} = 2 * dim_{pose} + 3 + 1 + 32$ based on the above inputs, where 3, 1, 32 represents the dimension of the time embedding, indicator logit, and the dimension of audio feature. For the transformer blocks, we use 8 Blocks, which are implemented in PyTorch Image Models (timm) [5]. And the input shape of the transformer blocks is $num_{pose} \times dim_{hidden}$. We set the number of heads to 8 in the multi-head self-attention blocks and the MLP ratio to 4 in Feed-Forward Networks. Finally, we use a simple 2-layer MLP to map the latent hidden space of the Transformer to the pose space.

Pose Auto-Encoder. Following Fréchet Inception Distance (FID) [2], which is widely leveraged to evaluate the image generation quality, Yoon *et al.* [6] proposes a similar evaluation metric Fréchet Gesture Distance (FGD) to evaluate gesture quality is. It leverages a pre-trained feature extractor to extract latent features of the gestures, then calculates the Fréchet distance between the distributions of the latent feature space of real and generated gestures. In our evaluation, we use the pre-trained auto-encoder provided from [6] and use it to evaluate FGD on TED Gesture. For TED Expressive, we train a new auto-encoder on this dataset following [3] and use this one to evaluate FGD on TED Expressive.

C. Additional Experiments

C.1. Exploitation of Other Context Information

In this work, we simplify the co-speech gestures generation and focus on the speech audio to drive the gestures generation while not using the text information as the input. We have shown the effectiveness with exhaustive experiments in the paper. Here, we also want to show more experiments to illustrate the ability to leverage various context information of our method. Specifically, we simply replace the audio feature in the context information with other formats of context information, like text information and speaker identity information. To show the results on various formats of context information, we train and test our method on the combination of context information and investigate all the eight settings. The results are reported in Table 2. And for the text information and speaker identity, following the pipeline in [7], we use a temporal convolutional network(TCN) [1] to exploit text information and an embedding mapping along with linear layers to map a speaker ID to a style embedding space. Then we simply concatenate multiple information before the Diffusion Audio-Gesture Transformer.

Context Input Logits			TED Gesture [6]			TED Expressive [3]		
			FGD ↓	BC ↑	Diversity ↑	FGD ↓	BC ↑	Diversity ↑
Ground Truth			0	0.698	108.525	0	0.703	178.827
audio	text	speaker	-	-	-	-	-	-
✗	✗	✗	12.542	0.745	92.536	25.768	0.682	166.075
✗	✗	✓	10.275	0.760	91.925	27.639	0.665	161.176
✗	✓	✗	3.474	0.704	103.546	3.137	0.707	182.137
✓	✗	✗	1.506	0.699	106.722	2.600	0.718	182.757
✗	✓	✓	3.462	0.695	102.378	3.587	0.715	175.697
✓	✗	✓	1.999	0.671	106.227	4.637	0.710	172.461
✓	✓	✗	2.685	0.726	101.986	2.089	0.723	184.422
✓	✓	✓	2.637	0.710	101.861	5.697	0.719	162.989

Table 2. **The quantitative results on TED Gesture [6] and TED Expressive [3].** We show all the eight settings corresponding to the different combinations of context modalities, where the setting 100 is what we investigate in this work.

In Table 2, it can be shown that our method produces great results when there is rich information, *i.e.,* excluding the first two rows, 000 and 001, where logit 1 denotes the modality is used and the logit 0 denotes the modality leaves unused.

With DiffGesture, it is simple yet effective to leverage other context information with the same architecture, which shows the scalability of our framework. Since we do not further investigate how to fuse multi-modality features and simply concatenate them in the input level, it is reasonable that case 111 does not produce the best metrics. We will explore how to effectively fuse multiple conditional information for high-fidelity generation in future work.

C.2. Ablation Studies on Different Architectures

We use a Transformer-based Network to serve the diffusion network, which aims to predict the Gaussian Noise accurately. Here we show the results of adopting different architectures in Table 3. We use the multilayered bidirectional gated recurrent unit (GRU) network in [6]. All the context inputs are kept the same as our method, *i.e.,* the initial poses, the feature of the audio, and the time embedding are concatenated before the network. For the 1D Conv network, we simply use multiple 1D Conv blocks consisting of Conv, bn, and ReLU, to upsample and downsample the inputs. It is observed that while our network achieves excellent performances, the GRU produces suboptimal results and 1D Conv cannot be well-trained in our setting. The results indicate that applying diffusion models in the audio-driven conditional generation is a non-trivial task, especially for non-static temporal data generation.

Network	FGD ↓	BC ↑	Diversity ↑
1D Conv on D_a	70.523	0.467	45.89
GRU on D_a	14.343	0.658	98.472
Transformer on D_a	**1.506**	**0.699**	**106.722**
1D Conv on D_b	95.410	0.475	132.250
GRU on D_b	17.452	0.680	172.168
Transformer on D_b	**2.600**	**0.718**	**182.757**

Table 3. **Ablation studies on different architectures.** We compare the performances of different diffusion-based backbones on TED Gesture (D_a) and TED Expressive (D_b).

References

[1] Shaojie Bai, J Zico Kolter, and Vladlen Koltun. An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. *arXiv preprint arXiv:1803.01271*, 2018. 3

[2] 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 I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. 3

[3] Xian Liu, Qianyi Wu, Hang Zhou, Yinghao Xu, Rui Qian, Xinyi Lin, Xiaowei Zhou, Wayne Wu, Bo Dai, and Bolei Zhou. Learning hierarchical cross-modal association for co-speech gesture generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10462–10472, June 2022. 3

[4] Shitong Luo and Wei Hu. Diffusion probabilistic models for 3d point cloud generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2021. 3

[5] Ross Wightman. Pytorch image models. https://github.com/rwightman/pytorch-image-models, 2019. 3

[6] Youngwoo Yoon, Bok Cha, Joo-Haeng Lee, Minsu Jang, Jaeyeon Lee, Jaehong Kim, and Geehyuk Lee. Speech gesture generation from the trimodal context of text, audio, and speaker identity. *ACM Transactions on Graphics (TOG)*, 39(6):1–16, 2020. 2, 3, 4

[7] Youngwoo Yoon, Woo-Ri Ko, Minsu Jang, Jaeyeon Lee, Jaehong Kim, and Geehyuk Lee. Robots learn social skills: End-to-end learning of co-speech gesture generation for humanoid robots. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 4303–4309. IEEE, 2019. 3