SoundCTM: Uniting Score-based and Consistency Models for Text-to-Sound Generation

Koichi Saito[1] **Dongjun Kim**[2] **Takashi Shibuya**[1] **Chieh-Hsin Lai**[1]
Zhi Zhong[3] **Yuhta Takida**[1] **Yuki Mitsufuji**[1,3]
[1]Sony AI [2]Stanford University [3]Sony Group Corporation
Koichi.Saito@sony.com

Abstract

Recent high-quality diffusion-based sound generation models can serve as valuable tools for sound content creators. However, despite producing high-quality sounds, these models often suffer from slow inference speeds. This drawback burdens creators, who typically refine their sounds through trial and error to align sounds with their artistic intentions. To address this issue, we introduce Sound Consistency Trajectory Models (SoundCTM). Our model enables flexible transitioning between high-quality 1-step sound generation and superior sound quality through multi-step generation. This allows creators to initially control sounds with 1-step samples before refining them through multi-step generation. We reframe original CTM's training framework and introduce a novel feature distance by utilizing the teacher's network for a distillation loss. Additionally, while distilling classifier-free guided trajectories, we train conditional and unconditional student models simultaneously and interpolate between these models during inference. SoundCTM achieves both promising 1-step and multi-step real-time sound generation. Audio samples are available at https://anonymus-soundctm.github.io/soundctm/.

1 Introduction

Sound content is essential for multimedia works such as video games, music, and films. Sound creators create sounds such as footsteps, dragon roaring, and environmental sounds by mixing and splicing digital sounds or by recording physical items. To enhance immersive experiences, they strive to produce flexible, diverse, and high-quality sound content tailored to each project.

Sound generation models have a potential to be valuable tools for sound creators. Recent diffusion-based Text-to-Sound (T2S) models [26, 27, 8, 14] have shown promising results. Despite producing high-quality sounds, these models suffer from slow inference speeds due to the iterative sampling in Diffusion Models (DMs) [33, 35, 16]. Sound creators must iterate to ensure the generated sounds align with their creative intentions, a process requiring them to listen to each sample individually. The slow inference adds a significant burden and time to the creative process. Thus, addressing slow inference makes sound generation models more appealing to sound creators.

An initial attempt to address slow inference for DM-based sound generation models is ConsistencyTTA [2], which uses Consistency Distillation (CD) [36] on T2S Latent Diffusion Models (LDMs) [30]. However, in practice, ConsistencyTTA focuses on 1-step generation, losing the flexibility of balancing sample quality with the number of sampling steps (see Table 2). Even if ConsistencyTTA is trained sufficiently well, achieving to demonstrate the trade-off between sample quality and the number of sampling steps, the generated contents cannot be preserved across different sampling steps (see Figure 1). This occurs even if the same initial noise and conditions are used since a deterministic sampling cannot be applied to ConsistencyTTA due to its training regime that learns

38th Conference on Neural Information Processing Systems (NeurIPS 2024).

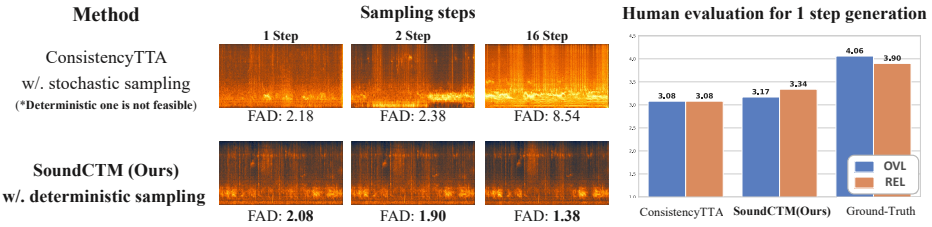


Figure 1: SoundCTM offers both 1-step high quality and multi-step higher quality sound generation and can preserve generated contents across different number of sampling steps. Additionally, 1-step generation of SoundCTM shows better overall audio quality (OVL) and relevance to text prompts (REL) than that of ConsistencyTTA [2].

anytime-to-zero-time jumps. This is another crucial issue for creators who need accurate, high-quality samples after finding suitable conditions for the models (see other examples in Figure 9).

In this paper, we present the ***Sound Consistency Trajectory Model*** (**SoundCTM**), a novel T2S model that offers flexible switching between 1-step high-quality sound generation and higher-quality multi-step generation. This is achieved through a training framework that learns anytime-to-anytime jumps and employs deterministic sampling as proposed in Consistency Trajectory Models (CTMs) [19]. Built on CTM's framework, we address the limitations of the current CTM training framework, which heavily depends on domain-specific components to achieve notable generation performance. Specifically, we propose a novel feature distance that uses a teacher's network as a feature extractor for distillation loss (see Section 3.1). Furthermore, we distill classifier-free guided text-conditional trajectories with Classifier-Free Guidance (CFG) [12] as a new condition for student models and train both text-conditional and unconditional student models, simultaneously. During inference, we introduce a new hyperparameter to interpolate the text-conditional and unconditional student models (see Section 3.2 and Algorithm 1).

In the experiments, SoundCTM achieves the state-of-the-art 1-step generation performance without any fine-tuning or auxiliary neural networks. In addition, under multi-step sampling, SoundCTM shows the flexible trade-off between the number of sampling steps and its generation quality.

2 Preliminary

Diffusion Models Let p_{data} denote the data distribution. In DMs, the data variable $\mathbf{x}_0 \sim p_{\text{data}}$ is generated through a reverse-time stochastic process [1] defined as $\mathrm{d}\mathbf{x}_t = -2t\nabla \log p_t(\mathbf{x}_t)dt + \sqrt{2t}\,\mathrm{d}\bar{\mathbf{w}}_t$ from time T to 0, where $\bar{\mathbf{w}}_t$ is the standard Wiener process in reverse-time. The marginal density $p_t(\mathbf{x})$ is obtained by encoding $\mathbf{x}_0$ along with a fixed forward diffusion process, $\mathrm{d}\mathbf{x}_t = \sqrt{2t}\,\mathrm{d}\mathbf{w}_t$, initialized by $\mathbf{x}_0$, where $\mathbf{w}_t$ is the standard Wiener process in forward-time. Song et al. [35] present the deterministic counterpart of the reverse-time process, called the *Probability Flow* Ordinary Differential Equation (PF ODE), given by

$$\frac{\mathrm{d}\mathbf{x}_t}{\mathrm{d}t} = -t\nabla \log p_t(\mathbf{x}_t) = \frac{\mathbf{x}_t - \mathbb{E}_{p_{t|0}(\mathbf{x}|\mathbf{x}_t)}[\mathbf{x}|\mathbf{x}_t]}{t},$$

where $p_{t|0}(\mathbf{x}|\mathbf{x}_t)$ is the probability distribution of the solution of the reverse-time stochastic process from time t to 0, initiated from $\mathbf{x}_t$. Practically, the denoiser function $\mathbb{E}_{p_{t|0}(\mathbf{x}|\mathbf{x}_t)}[\mathbf{x}|\mathbf{x}_t]$ [6] is estimated by a neural network $D_{\boldsymbol{\phi}}$, obtained by minimizing a Denoising Score Matching (DSM) loss [37, 35] $\mathbb{E}_{\mathbf{x}_0,t,p_{0|t}(\mathbf{x}|\mathbf{x}_0)}[\|\mathbf{x}_0 - D_{\boldsymbol{\phi}}(\mathbf{x},t)\|_2^2]$, where $p_{0|t}(\mathbf{x}|\mathbf{x}_0)$ is the transition probability from time 0 to t, initiated with $\mathbf{x}_0$. Given the trained denoiser, the empirical PF ODE is given by

$$\frac{\mathrm{d}\mathbf{x}_t}{\mathrm{d}t} = \frac{\mathbf{x}_t - D_{\boldsymbol{\phi}}(\mathbf{x}_t,t)}{t}. \tag{1}$$

DMs can generate samples by solving the empirical PF ODE, initiated with $\mathbf{x}_T$.

Text-Conditional Sound Generation with Latent Diffusion Models LDM-based T2S models [26, 27, 8] generate audio matched to textual descriptions by first obtaining the latent counterpart of the data variable $\mathbf{z}_0$ through the reverse-time process conditioned by text embedding $\mathbf{c}_{\text{text}}$. This latent variable $\mathbf{z}_0$ is then converted to $\mathbf{x}_0$ using a pretrained decoder $\mathcal{D}$. During the training phase, $D_{\boldsymbol{\phi}}(\mathbf{z},t,\mathbf{c}_{\text{text}})$ is trained by minimizing the DSM loss.

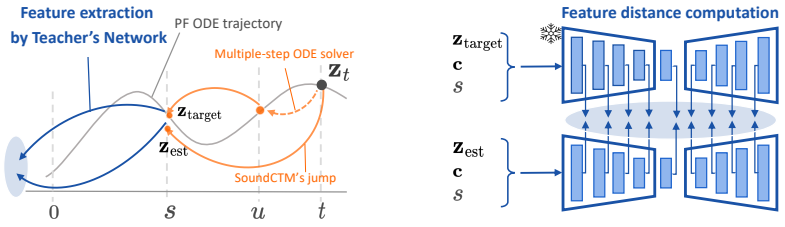


Figure 2: Illustrations of SoundCTM's two predictions $\mathbf{z}_{\text{target}}$ and $\mathbf{z}_{\text{est}}$ at time s with an initial value $\mathbf{z}_t$ and the feature extraction by the teacher's network for the CTM loss shown within the blue ellipse area. The conditional embedding $\mathbf{c}$ and time s are also input to the feature extractor.

Consistency Trajectory Models CTMs predict both infinitesimally small step jump and long step jump of the PF ODE trajectory. $G(\mathbf{x}_t, t, s)$ is defined as the solution of the PF ODE from initial time t to final time $s \leq t$ and G is estimated by $G_{\boldsymbol{\theta}}$ as the neural jump. To train $G_{\boldsymbol{\theta}}$, two s-predictions are compared: one from a teacher $\boldsymbol{\phi}$ and the other from a student $\boldsymbol{\theta}$ as:

$$G_{\boldsymbol{\theta}}(\mathbf{x}_t, t, s) \approx G_{\mathtt{sg}(\boldsymbol{\theta})}\big(\mathtt{Solver}(\mathbf{x}_t, t, u; \boldsymbol{\phi}), u, s\big), \tag{2}$$

where $\mathtt{Solver}(\mathbf{x}_t, t, s; \boldsymbol{\phi})$ is the pre-trained PF ODE in Eq. (1), a random $u \in [s, t)$ determines the amount of teacher information to distill, and $\mathtt{sg}$ is the exponential moving average (EMA) stop-gradient $\mathtt{sg}(\boldsymbol{\theta}) \leftarrow \mathtt{stopgrad}(\mu \mathtt{sg}(\boldsymbol{\theta}) + (1-\mu)\boldsymbol{\theta})$.

3 SoundCTM

To address the challenges of achieving fast, flexible, and high-quality T2S generation, we introduce SoundCTM. Since this paper primarily illustrates the method using LDM-based T2S models as the teacher model, the student model is trained to estimate the neural jump $G_{\boldsymbol{\theta}}$ as:

$$G_{\boldsymbol{\theta}}(\mathbf{z}_t, \mathbf{c}_{\text{text}}, t, s) \approx G_{\mathtt{sg}(\boldsymbol{\theta})}\big(\mathtt{Solver}(\mathbf{z}_t, \mathbf{c}_{\text{text}}, t, u; \boldsymbol{\phi}), \mathbf{c}_{\text{text}}, u, s\big), \tag{3}$$

where $\mathbf{z}_t$ is the latent counterpart of $\mathbf{x}_t$, and $\mathtt{Solver}(\mathbf{z}_t, \mathbf{c}_{\text{text}}, t, s; \boldsymbol{\phi})$ is the numerical solver of the pre-trained text-conditional PF ODE by following Eq. (2). To quantify the dissimilarity between $G_{\boldsymbol{\theta}}$ and $G_{\mathtt{sg}(\boldsymbol{\theta})}$, we propose a new feature distance in Section 3.1.

3.1 Teacher's Network as Feature Extractor for Distillation Loss

We propose a new training framework that **utilizes the teacher's network as a feature extractor** and measures the feature distance d_{teacher} between $G_{\boldsymbol{\theta}}$ and $G_{\mathtt{sg}(\boldsymbol{\theta})}$ in Eq. (3), as illustrated in Figure 2. The main benefit of using d_{teacher} is that it yields better performance compared with using the l_2 distance in the $\mathbf{z}$ domain computed at either 0-time or s-time, as demonstrated in Table 1.

We define d_{teacher} between two predictions $G_{\boldsymbol{\theta}}$ and $G_{\mathtt{sg}(\boldsymbol{\theta})}$ as follows:

$$d_{\text{teacher}}(G_{\mathtt{sg}(\boldsymbol{\theta})}, G_{\boldsymbol{\theta}}, \mathbf{c}, t) = \sum_{m=1}^{M} \|\text{TN}_{\boldsymbol{\phi},m}(G_{\mathtt{sg}(\boldsymbol{\theta})}, \mathbf{c}, t) - \text{TN}_{\boldsymbol{\phi},m}(G_{\boldsymbol{\theta}}, \mathbf{c}, t)\|_2^2, \tag{4}$$

where $\text{TN}_{\boldsymbol{\phi},m}(\cdot, \mathbf{c}, t)$ denotes the channel-wise normalized output feature of the m-th layer of the pretrained teacher's network, conditioned by time t and embedding $\mathbf{c}$. This approach is feasible since noisy latents are input to the teacher's network during teacher's training.

3.2 CFG Handling for SoundCTM

To manage CFG, which plays a crucial role in text-conditional diffusion models [12, 8, 26], in SoundCTM, we propose distilling the classifier-free guided PF ODE trajectory scaled by ω, uniformly sampled from the range $[\omega_{\text{min}}, \omega_{\text{max}}]$ during training, and using ω as a new condition in the student network, defined as:

$$G_{\boldsymbol{\theta}}(\mathbf{z}_t, \mathbf{c}_{\text{text}}, \omega, t, s) = \frac{s}{t}\mathbf{z}_t + \left(1 - \frac{s}{t}\right) g_{\boldsymbol{\theta}}(\mathbf{z}_t, \mathbf{c}_{\text{text}}, \omega, t, s). \tag{5}$$

To summarize Eq. (3) and Eq. (5), the distillation loss for SoundCTM is formulated as:

$$\mathcal{L}_{\text{CTM}}^{\text{Sound}}(\boldsymbol{\theta};\boldsymbol{\phi}) := \mathbb{E}_{t\in[0,T]}\mathbb{E}_{s\in[0,t]}\mathbb{E}_{u\in[s,t)}\mathbb{E}_{\omega}\mathbb{E}_{\mathbf{z}_0}\mathbb{E}_{\mathbf{z}_t|\mathbf{z}_0}\Big[d_{\text{teacher}}\big(\mathbf{z}_{\text{target}}(\mathbf{z}_t,\mathbf{c}_{\text{text}},\omega,t,u,s),\mathbf{z}_{\text{est}}(\mathbf{z}_t,\mathbf{c}_{\text{text}},\omega,t,s),\mathbf{c}_{\text{text}},s\big)\Big], \tag{6}$$

where

$$\mathbf{z}_{\text{target}}(\mathbf{z}_t,\mathbf{c}_{\text{text}},\omega,t,u,s) := G_{\text{sg}(\boldsymbol{\theta})}(\texttt{Solver}(\mathbf{z}_t,\mathbf{c}_{\text{text}},\omega,t,u;\boldsymbol{\phi}),\mathbf{c}_{\text{text}},\omega,u,s),$$
$$\mathbf{z}_{\text{est}}(\mathbf{z}_t,\mathbf{c}_{\text{text}},\omega,t,s) := G_{\boldsymbol{\theta}}(\mathbf{z}_t,\mathbf{c}_{\text{text}},\omega,t,s),$$

$\texttt{Solver}(\mathbf{z}_t,\mathbf{c}_{\text{text}},\omega,t,u;\boldsymbol{\phi}) := \omega\texttt{Solver}(\mathbf{z}_t,\mathbf{c}_{\text{text}},t,u;\boldsymbol{\phi}) + (1-\omega)\texttt{Solver}(\mathbf{z}_t,\varnothing,t,u;\boldsymbol{\phi})$, $\varnothing$ is an unconditional embedding, and $\omega \sim U[\omega_{\text{min}},\omega_{\text{max}}]$, respectively. To achieve better generation performance, the two auxiliary losses, the DSM loss and an adversarial loss [10], are used in the original CTM. For SoundCTM, we use the DSM loss , defined as:

$$\mathcal{L}_{\text{DSM}}^{\text{Sound}}(\boldsymbol{\theta}) = \mathbb{E}_{t,\mathbf{z}_0,\omega}\mathbb{E}_{\mathbf{z}_t|\mathbf{z}_0}[\|\mathbf{z}_0 - g_{\boldsymbol{\theta}}(\mathbf{z}_t,\mathbf{c}_{\text{text}},\omega,t,t)\|_2^2]. \tag{7}$$

Note that the DSM loss serves to improve the accuracy of approximating the small jumps during the training. Summing Eqs. (6) and (7), SoundCTM is trained with the following objective:

$$\mathcal{L}(\boldsymbol{\theta}) := \mathcal{L}_{\text{CTM}}^{\text{Sound}}(\boldsymbol{\theta};\boldsymbol{\phi}) + \lambda_{\text{DSM}}\mathcal{L}_{\text{DSM}}^{\text{Sound}}(\boldsymbol{\theta}), \tag{8}$$

where λ_{DSM} is a scaling weight for $\mathcal{L}_{\text{DSM}}^{\text{Sound}}$. Algorithm 2 summarizes SoundCTM's training framework.

ν-interpolation for SoundCTM's Sampling For SoundCTM's inference, we introduce ν-interpolation, which linearly interpolates between the text-conditional student models $G_{\boldsymbol{\theta}}(\mathbf{z}_t,\mathbf{c}_{\text{text}},\omega,t,s)$ and unconditional student models $G_{\boldsymbol{\theta}}(\mathbf{z}_t,\varnothing,\omega,t,s)$. In contrast to the previous sampling methods for the diffusion-based distillation models [19, 2], we also use $G_{\boldsymbol{\theta}}(\mathbf{z}_t,\varnothing,\omega,t,s)$ as highlighted in blue Algorithm 1.

Algorithm 1 SoundCTM's Inference

Require: Hyperparameter ν, text condition $\mathbf{c}_{\text{text}}$, CFG scale ω, hyperparameter of CTM's γ-sampling γ

1: Sample $\mathbf{z}_{t_0}$ from prior distribution
2: **for** $n = 0$ to $N-1$ **do**
3: $\quad\tilde{t}_{n+1} \leftarrow \sqrt{1-\gamma^2}t_{n+1}$
4: $\quad\mathbf{z}_{\tilde{t}_{n+1}} \leftarrow \nu G_{\boldsymbol{\theta}}(\mathbf{z}_{t_n},\mathbf{c}_{\text{text}},\omega,t_n,\tilde{t}_{n+1})$
5: $\qquad+(1-\nu)G_{\boldsymbol{\theta}}(\mathbf{z}_{t_n},\varnothing,\omega,t_n,\tilde{t}_{n+1})$
6: $\quad\mathbf{z}_{t_{n+1}} \leftarrow \mathbf{z}_{\tilde{t}_{n+1}} + \gamma t_{n+1}\boldsymbol{\epsilon}$
7: **end for**
8: **Return** $\mathbf{z}_{t_N}$

4 Experiments

We evaluate SoundCTM on the AudioCaps dataset [18], which contains 47,289 pairs of 10-second audio samples and human-written text descriptions for the training set and 957 pairs for the testset. All audio samples are downsampled to 16 kHz. We adopt TANGO [8] as the teacher model trained with EDM's variance exploding formulation [16]. We use deterministic sampling ($\gamma = 0$ in algorithm 1) and evaluate the model performance with student EMA rate $\mu = 0.999$. We also conduct the human evaluation in Appendix C.1.

Evaluation Metrics We use four objective metrics: the Frechet Audio Distance (FAD_{vgg}) [17] between the extracted embeddings by VGGish, the Kullback-Leibler divergence (KL_{passt}) between the outputs of PaSST [22], a state-of-the-art audio classification model, the Inception Score (IS_{passt}) [31] using the outputs of PaSST, and the CLAP score [39]. We refer the detail explanations of these metrics in Appendix B.1.

Table 1: Performance comparisons on AudioCaps test set. † denotes the results tested by us with provided checkpoint.

Model	# of steps	$\text{FAD}_{\text{vgg}}\downarrow$	$\text{KL}_{\text{passt}}\downarrow$	$\text{IS}_{\text{passt}}\uparrow$	CLAP $\uparrow$
Teacher Models					
TANGO-ConsistencyTTA (reported) (Teacher of ConsistencyTTA w/. DDIM)	200	1.91	-	-	-
TANGO-EDM (Teacher of Ours w/. Heun)	40	1.71	1.28	8.11	0.46
Student Models					
ConsistencyTTA [2]	1	2.58	1.33†	6.85†	0.41†
SoundCTM w/. l_2 (0-time step)	1	2.43	1.28	6.87	0.42
SoundCTM w/. l_2 (s-time step)	1	2.45	1.28	6.83	0.42
SoundCTM w/. d_{teacher}	1	**2.17**	**1.27**	**7.18**	**0.43**

Effectiveness of Utilizing Teacher's Network as Feature Extractor We first evaluate the efficacy of utilizing the teacher's network as a feature extractor for the distillation loss by comparing the following cases: 1) the l_2 at 0-time step, 2) the l_2 at s-time step, and 3) the d_{teacher} at s-time step. We report the results of 1-step generation with $\omega = 3.5$ and $\nu = 1.0$ for inference. As shown in Table 1, the results of the d_{teacher} case is better than the others. This result indicates that using d_{teacher} allows the student to distill the trajectory more accurately than using l_2.

Table 2: Performance comparisons on AudioCaps test set. Bold and underlined scores indicate the best and second-best results. † denotes the results tested by us using the provided checkpoints by the authors, as not all metrics are provided in each paper.

Model	# of sampling steps	ω	ν	$\text{FAD}_{\text{vgg}} \downarrow$	$\text{KL}_{\text{passt}} \downarrow$	$\text{IS}_{\text{passt}} \uparrow$	CLAP ↑
Diffusion Models							
AudioLDM-Large-FT [26]	200	3.0	-	1.96	1.59	-	-
AudioLDM 2-AC-Large [27]	200	3.5	-	<u>1.42</u>	**0.98**	-	-
AudioLDM 2-Full-Large [27]	200	3.5	-	1.86	1.64	-	-
TANGO† [8]	200	3.0	-	1.64	1.31	6.35	0.44
TANGO w/. Heun Solver (Our teacher model)	40	3.5	-	1.71	1.28	<u>8.11</u>	**0.46**
Distillation Models							
ConsistencyTTA [2](reported)	1	5.0	-	2.58	-	-	-
ConsistencyTTA† (tested by us)	1	5.0	-	2.67	1.33	6.85	0.41
	2	5.0	-	3.18	1.34	7.12	0.38
ConsistencyTTA-CLAP-FT (reported)	1	3.0	-	2.18	-	-	-
ConsistencyTTA-CLAP-FT† (tested by us)	1	3.0	-	2.22	1.35	6.95	0.41
	2	3.0	-	2.38	1.32	7.15	0.40
SoundCTM(Ours)	1	3.5	1.0	2.08	1.26	7.13	0.43
	2	3.5	1.0	1.90	1.24	7.26	<u>0.45</u>
	4	3.5	1.0	1.72	1.22	7.37	<u>0.45</u>
	8	3.0	1.5	1.45	1.20	7.98	**0.46**
	16	3.0	2.0	**1.38**	<u>1.19</u>	**8.24**	**0.46**

Additionally, we compare SoundCTM's performance with the 1-step generation of ConsistencyTTA [2]. The results of SoundCTM's 1-step generation trained with l_2 at 0-time step is slightly better than those of ConsistencyTTA's 1-step generation. This performance gap is likely due to differences in the teacher models. In the context of 1-step generation, the differences in the training framework between ConsistencyTTA and SoundCTM using l_2 at the 0-time step do not fundamentally contribute to the variations in their 1-step generation performance. On the other hand, using d_{teacher} further boosts SoundCTM's performance compared with using l_2. This indicates that SoundCTM with d_{teacher} surpasses ConsistencyTTA even considering of the teacher difference.

Performance Comparison with Other T2S Models We compare SoundCTM's performance with other T2S models under both 1-step and multi-step generation. We employ AudioLDM-L-FT [26], AudioLDM2-L [27], TANGO [8], and ConsistencyTTA as our baseline models. Table 2 presents the quantitative results. All the DM-based models use DDIM sampling [34] except for our teacher model that uses 2nd order Heun Solver [16]. Under the 1-step generation setting, SoundCTM shows the best performance in all evaluation metrics. This achievement is the obtained from using d_{teacher} as discussed earlier. Note that ConsistencyTTA-CLAP-FT conducts extra fine-tuning to maximize the CLAP score using CLAP network after its CD training.

Under the multi-step case, there are clear trade-offs between the performance improvements and the number of sampling steps. We highlight these results are attributed to SoundCTM's deterministic sampling ($\gamma = 0$) and anytime-to-anytime jump training framework, which cannot be achieved with the CD framework and its lack of deterministic sampling capability.

Generation Time Since one of our goals is to achieve fast high-quality generation, we verified that our model can conduct fast generation by measuring the inference time and real-time factors (RTFs) on a single NVIDIA RTX A6000. For reference, we also measure those of the original TANGO with a batch size of one. As shown in Table 3, SoundCTM can achieve real-time generation on a GPU with 16-step and on an Intel Xeon CPU with 2-step.

Table 3: Inference speed and RTF. RTF < 1.0 indicates real-time generation.

Method	# of steps	Batch size	Speed [sec.]	RTF ↓
TANGO (Diffusion models)	200	1	24	2.4
SoundCTM on GPU	16	1	**2.43**	**0.24**
SoundCTM on CPU	2	1	**6.93**	**0.69**

5 Conclusion

We propose SoundCTM that achieves not only high-quality 1-step generation but also significantly improves sample quality by increasing the number of sampling steps with a single model. Our new training framework contributes to its performance. Furthermore, our frameworks, which does not rely on domain-specific components, has the potential to extend the applicability of CTM to a wider range of domains while maintaining both its fundamental methodology and impressive performance.

References

[1] Brian. D. O. Anderson. Reverse-time diffusion equation models. *Stochastic Processes and their Applications*, 12:313–326, 1982.

[2] Yatong Bai, Trung Dang, Dung Tran, Kazuhito Koishida, and Somayeh Sojoudi. Accelerating diffusion-based text-to-audio generation with consistency distillation. *arXiv preprint arXiv:2309.10740*, 2023.

[3] Keunwoo Choi, Jaekwon Im, Laurie Heller, Brian McFee, Keisuke Imoto, Yuki Okamoto, Mathieu Lagrange, and Shinosuke Takamichi. Foley sound synthesis at the dcase 2023 challenge. *In arXiv e-prints: 2304.12521*, 2023.

[4] Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Yunxuan Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, Albert Webson, Shixiang Shane Gu, Zhuyun Dai, Mirac Suzgun, Xinyun Chen, Aakanksha Chowdhery, Alex Castro-Ros, Marie Pellat, Kevin Robinson, Dasha Valter, Sharan Narang, Gaurav Mishra, Adams Yu, Vincent Zhao, Yanping Huang, Andrew Dai, Hongkun Yu, Slav Petrov, Ed H. Chi, Jeff Dean, Jacob Devlin, Adam Roberts, Denny Zhou, Quoc V. Le, and Jason Wei. Scaling instruction-finetuned language models, 2022.

[5] Marco Comunità, Zhi Zhong, Akira Takahashi, Shiqi Yang, Mengjie Zhao, Koichi Saito, Yukara Ikemiya, Takashi Shibuya, Shusuke Takahashi, and Yuki Mitsufuji. Specmaskgit: Masked generative modeling of audio spectrograms for efficient audio synthesis and beyond. *arXiv preprint arXiv:2406.17672*, 2024.

[6] Bradley Efron. Tweedie's formula and selection bias. *Journal of the American Statistical Association*, 106:1602 – 1614, 2011.

[7] Zach Evans, CJ Carr, Josiah Taylor, Scott H. Hawley, and Jordi Pons. Fast timing-conditioned latent audio diffusion. *Proc. International Conference on Machine Learning (ICML)*, 2024.

[8] Deepanway Ghosal, Navonil Majumder, Ambuj Mehrish, and Soujanya Poria. Text-to-audio generation using instruction tuned llm and latent diffusion model. *Proc. ACM Int. Conf. on Multimedia. (ACMMM)*, 2023.

[9] Sang gil Lee, Wei Ping, Boris Ginsburg, Bryan Catanzaro, and Sungroh Yoon. Bigvgan: A universal neural vocoder with large-scale training. In *Proc. International Conference on Learning Representation (ICLR)*, 2023.

[10] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. *Proc. Advances in Neural Information Processing Systems (NeurIPS)*, 2014.

[11] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016.

[12] Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. *In arXiv e-prints: 2207.12598*, 2022.

[13] Jiawei Huang, Yi Ren, Rongjie Huang, Dongchao Yang, Zhenhui Ye, Chen Zhang, Jinglin Liu, Xiang Yin, Zejun Ma, and Zhou Zhao. Make-an-audio 2: Temporal-enhanced text-to-audio generation. *arXiv preprint arXiv:2305.18474*, 2023.

[14] Rongjie Huang, Jiawei Huang, Dongchao Yang, Yi Ren, Luping Liu, Mingze Li, Zhenhui Ye, Jinglin Liu, Xiang Yin, and Zhou Zhao. Make-an-audio: Text-to-audio generation with prompt-enhanced diffusion models. *Proc. International Conference on Machine Learning (ICML)*, 2023.

[15] Vladimir Iashin and Esa Rahtu. Taming visually guided sound generation. In *British Machine Vision Conference (BMVC)*, 2021.

[16] Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. Elucidating the design space of diffusion-based generative models. In *Proc. Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

[17] Kevin Kilgour, Mauricio Zuluaga, Dominik Roblek, and Matthew Sharifi. Fréchet audio distance: A metric for evaluating music enhancement algorithms. *arXiv preprint arXiv:1812.08466*, 2019.

[18] Chris Dongjoo Kim, Byeongchang Kim, Hyunmin Lee, and Gunhee Kim. AudioCaps: Generating Captions for Audios in The Wild. In *NAACL-HLT*, 2019.

[19] Dongjun Kim, Chieh-Hsin Lai, Wei-Hsiang Liao, Naoki Murata, Yuhta Takida, Toshimitsu Uesaka, Yutong He, Yuki Mitsufuji, and Stefano Ermon. Consistency trajectory models: Learning probability flow ode trajectory of diffusion. In *Proc. International Conference on Learning Representation (ICLR)*, 2024.

[20] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *Proc. International Conference on Learning Representation (ICLR)*, 2017.

[21] Jungil Kong, Jaehyeon Kim, and Jaekyoung Bae. Hifi-gan: Generative adversarial networks for efficient and high fidelity speech synthesis. In *Proc. Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

[22] Khaled Koutini, Jan Schlüter, Hamid Eghbal-zadeh, and Gerhard Widmer. Efficient training of audio transformers with patchout. In *Interspeech 2022, 23rd Annual Conference of the International Speech Communication Association, Incheon, Korea, 18-22 September 2022*, pages 2753–2757, 2022.

[23] Felix Kreuk, Gabriel Synnaeve, Adam Polyak, Uriel Singer, Alexandre Défossez, Jade Copet, Devi Parikh, Yaniv Taigman, and Yossi Adi. Audiogen: Textually guided audio generation. *Proc. International Conference on Learning Representation (ICLR)*, 2023.

[24] Rithesh Kumar, Prem Seetharaman, Alejandro Luebs, Ishaan Kumar, and Kundan Kumar. High-fidelity audio compression with improved rvqgan. In *Proc. Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

[25] Mark Levy, Bruno Di Giorgi, Floris Weers, Angelos Katharopoulos, and Tom Nickson. Controllable music production with diffusion models and guidance gradients. *arXiv preprint arXiv:2311.00613*, 2023.

[26] Haohe Liu, Zehua Chen, Yi Yuan, Xinhao Mei, Xubo Liu, Danilo Mandic, Wenwu Wang, and Mark D Plumbley. AudioLDM: Text-to-audio generation with latent diffusion models. *Proc. International Conference on Machine Learning (ICML)*, 2023.

[27] Haohe Liu, Qiao Tian, Yi Yuan, Xubo Liu, Xinhao Mei, Qiuqiang Kong, Yuping Wang, Wenwu Wang, Yuxuan Wang, and Mark D. Plumbley. AudioLDM 2: Learning holistic audio generation with self-supervised pretraining. *IEEE/ACM Trans. Audio, Speech, Lang. Process.*, 2024.

[28] Liyuan Liu, Haoming Jiang, Pengcheng He, Weizhu Chen, Xiaodong Liu, Jianfeng Gao, and Jiawei Han. On the variance of the adaptive learning rate and beyond. In *Proc. International Conference on Learning Representation (ICLR)*, April 2020.

[29] Zachary Novack, Julian McAuley, Taylor Berg-Kirkpatrick, and Nicholas J. Bryan. Ditto: Diffusion inference-time t-optimization for music generation. *Proc. International Conference on Machine Learning (ICML)*, 2024.

[30] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[31] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training gans. In *Proc. Advances in Neural Information Processing Systems (NeurIPS)*, 2016.

[32] Abraham. Savitzky and M. J. E. Golay. Smoothing and differentiation of data by simplified least squares procedures. *Analytical Chemistry*, 36(8):1627–1639, 1964.

[33] Jascha Sohl-Dickstein, Eric A. Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *Proc. International Conference on Machine Learning (ICML)*, 2015.

[34] J. Song, C. Meng, and S. Ermon. Denoising diffusion implicit models. *Proc. International Conference on Learning Representation (ICLR)*, 2021.

[35] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *Proc. International Conference on Learning Representation (ICLR)*, 2021.

[36] Yang Song, Prafulla Dhariwal, Mark Chen, and Ilya Sutskever. Consistency models. *Proc. International Conference on Machine Learning (ICML)*, 2023.

[37] Pascal Vincent. A connection between score matching and denoising autoencoders. *Neural Computation*, 23(7):1661–1674, 2011.

[38] Shih-Lun Wu, Chris Donahue, Shinji Watanabe, and Nicholas J. Bryan. Music controlnet: Multiple time-varying controls for music generation. *IEEE/ACM Trans. Audio, Speech, Lang. Process.*, 2024.

[39] Yusong Wu*, Ke Chen*, Tianyu Zhang*, Yuchen Hui*, Taylor Berg-Kirkpatrick, and Shlomo Dubnov. Large-scale contrastive language-audio pretraining with feature fusion and keyword-to-caption augmentation. In *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP*, 2023.

[40] Jiwen Yu, Yinhuai Wang, Chen Zhao, Bernard Ghanem, and Jian Zhang. Freedom: Training-free energy-guided conditional diffusion model. *Proc. IEEE International Conference on Computer Vision (ICCV)*, 2023.

[41] Alon Ziv, Itai Gat, Gael Le Lan, Tal Remez, Felix Kreuk, Alexandre Défossez, Jade Copet, Gabriel Synnaeve, and Yossi Adi. Masked audio generation using a single non-autoregressive transformer. *Proc. International Conference on Learning Representation (ICLR)*, 2024.

A Related Work

Masked Audio Token Modeling AudioGen [23], MAGNeT [41], and SpecMaskGIT [5] utilize masked generative sequence modeling of discrete audio tokens for T2S generation. These models exhibit a trade-off between sample quality and inference speed due to their token prediction methods. AudioGen provides better sample quality but requires much more time for sample generation due to its autoregressive prediction. In contrast, MAGNeT and SpecMaskGIT offer faster inference through its non-autoregressive prediction.

Diffusion-based Models Recent literature [14, 13, 26, 27, 8, 7] and competitions [3] report that LDM-based sound generation models outperform other approaches such as GANs. For fast generation, ConsistencyTTA [2] and Stable Audio [7] employ efficient strategies. Stable Audio, for instance, compresses a 95-second waveform signal to a latent representation, allowing its LDM to generate the representation through a multi-step reverse-time process, unlike other models that compress only 10-second audio signals.

Training-free Controllable Generation Levy et al. [25] demonstrate training-free controllable music generation using loss-based guidance [40]. Their approach defines a task-specific loss, such as for music continuation and infilling, and incorporates the gradient of this loss into the reverse-time diffusion process. Novack et al. [29] also present controllable generation by optimizing initial noisy latents based on loss computed in the target condition space. During optimization, diffusion models perform a reverse-time process to obtain a clean signal, which is projected onto the condition space by a predefined differentiable feature extractor. Post-optimization, the model generates a music signal from the optimized initial latent.

B Experimental Details

B.1 Details of T2S Generation

Training Details For teacher's training, we use $\sigma_{\text{data}} = 0.25$ and time sampling $t \sim \mathcal{N}(-1.2, 1.2^2)$ by following EDM's training manner. We mostly follow the original CTM's training setup for student training. We utilize the EDM's skip connection $c_{\text{skip}}(t) = \frac{\sigma_{\text{data}}^2}{t^2+\sigma_{\text{data}}^2}$ and output scale $c_{\text{out}}(t) = \frac{t\sigma_{\text{data}}}{\sqrt{t^2+\sigma_{\text{data}}^2}}$ for $g_{\boldsymbol{\theta}}$ modeling as

$$g_{\boldsymbol{\theta}}(\mathbf{z}_t, \mathbf{c}_{\text{text}}, \omega, t, s) = c_{\text{skip}}(t)\mathbf{z}_t + c_{\text{out}}(t)\text{NN}_{\boldsymbol{\theta}}(\mathbf{z}_t, \mathbf{c}_{\text{text}}, \omega, t, s),$$

where $\text{NN}_{\boldsymbol{\theta}}$ refers to the actual neural network output. We initialize the student's $\text{NN}_{\boldsymbol{\theta}}$ with $\boldsymbol{\phi}$ except for student model's s-embedding and ω-embedding structure. Following the original CTM, we employ adaptive weighting with $\lambda_{\text{DSM}} = \frac{\|\nabla_{\boldsymbol{\theta}_L} \mathcal{L}_{\text{CTM}}^{\text{Sound}}(\boldsymbol{\theta};\boldsymbol{\phi})\|}{\|\nabla_{\boldsymbol{\theta}_L} \mathcal{L}_{\text{DSM}}^{\text{Sound}}(\boldsymbol{\theta})\|}$, where θ_L is the last layer of the student's network.

We use 8×NVIDIA H100 (80G) GPUs and a global batch size of 64 for the training. We choose t and s from the N-discretized timesteps to calculate $\mathcal{L}_{\text{CTM}}^{\text{Sound}}$. For $\mathcal{L}_{\text{DSM}}^{\text{Sound}}$ calculation, we opt to use 50% of time sampling $t \sim \mathcal{N}(-1.2, 1.2^2)$. For the other half time, we first draw sample from $\xi \sim [0, 0.7]$ and transform it using $(\sigma_{\max}^{1/\rho} + \xi(\sigma_{\min}^{1/\rho} - \sigma_{\max}^{1/\rho}))^{\rho}$. We apply Exponential Moving Average (EMA) to update $\texttt{sg}(\boldsymbol{\theta})$ by

$$\texttt{sg}(\boldsymbol{\theta}) \leftarrow \texttt{stopgrad}(\mu\texttt{sg}(\boldsymbol{\theta}) + (1-\mu)\boldsymbol{\theta}).$$

Throughout the experiments, for student training, we use $N = 40$, $\mu = 0.999$, $\sigma_{\min} = 0.002$, $\sigma_{\max} = 80$, $\rho = 7$, RAdam optimizer [28] with a learning rate of 8.0×10^{-5}, and $\sigma_{\text{data}} = 0.25$. We set the maximum number of ODE steps as 39 during training and we use Heun solver for $\texttt{Solver}(\mathbf{z}_t, \mathbf{c}_{\text{text}}, \omega, t, u; \boldsymbol{\phi})$. To obtain the results shown in Table 2 and Figure 4, SoundCTM was trained for 30 K iterations. For other results, models were trained for 18 K iterations. We also utilized TANGO's data augmentation [8, Sec. 2.3] during student training.

The network architecture and dataset for the teacher's training remained unchanged from the original ones. Our teacher model (TANGO) and SoundCTM share the overall model architecture, comprising the VAE-GAN [26], the HiFiGAN vocoder [21] as $\mathcal{D}$ and the Stable Diffusion UNet architecture (SD-1.5), which consists of 9 2D-convolutional ResNet [11] blocks as D_{ϕ}. The UNet has a total of 866 M

Table 4: Subjective evaluation on AudioCaps testset

Method	# of sampling steps	OVL ↑	REL ↑
Ground-truth	-	4.06	3.90
ConsistencyTTA [2]	1	3.08	3.08
SoundCTM w/. l_2 (0-time step)	1	**3.18**	3.28
SoundCTM w/. $d_{\textbf{teacher}}$	1	3.17	**3.34**

parameters, and the frozen FLAN-T5-Large text encoder [4] is used. The UNet employs 8 latent channels and a cross-attention dimension of 1024. For $\mathcal{D}$ in SoundCTM, we used the AudioLDM1 checkpoint by following TANGO.

Evaluation Details For large-NFE sampling, we follow the EDM's and the CTM's time discretization. Namely, if we draw n-NFE samples, we divide $[0, 1]$ with n points and transform it (say ξ) to the time scale by $(\sigma_{\max}^{1/\rho} + (\sigma_{\min}^{1/\rho} - \sigma_{\max}^{1/\rho})\xi)^{\rho}$. However, we emphasize the time discretization for both training and sampling is a modeler's choice.

We use four objective metrics: the Frechet Audio Distance (FAD_{vgg}) [17] between the extracted embeddings by VGGish, the Kullback-Leibler divergence (KL_{passt}) between the outputs of PaSST [22], a state-of-the-art audio classification model, the Inception Score (IS_{passt}) [31] using the outputs of PaSST, and the CLAP score[1]. The lower FAD indicates better audio quality of the generated audio. The KL measures how semantically similar the generated audio is to the reference audio. The IS evaluates sample diversity. The CLAP score demonstrates how well the generated audio adheres to the given textual description.

Algorithm 2 SoundCTM's Training

Require: Probability of unconditional training p_{uncond}

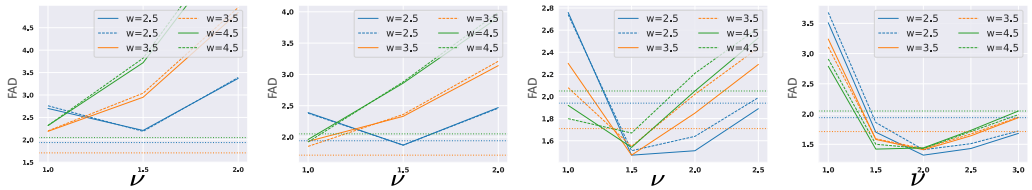
repeat
  Sample $(\mathbf{x}_0, \mathbf{c}_{\text{text}})$ from p_{data}
  Calculate $\mathbf{z}_0$ through $\mathcal{E}(\mathbf{x}_0)$
  $\mathbf{c}_{\text{text}} \leftarrow \varnothing$ with p_{uncond}
  Sample $\boldsymbol{\epsilon} \sim \mathcal{N}(0, I)$
  Sample $t \in [0, T]$, $s \in [0, t]$, $u \in [s, t)$
  Sample $\omega \sim U[\omega_{\min}, \omega_{\max}]$
  Calculate $\mathbf{z}_t = \mathbf{z}_0 + t\boldsymbol{\epsilon}$
  Calculate $\texttt{Solver}(\mathbf{z}_t, \mathbf{c}_{\text{text}}, \omega, t, u; \boldsymbol{\phi})$
  Calculate $\mathbf{z}_{\text{target}}(\mathbf{z}_t, \mathbf{c}_{\text{text}}, \omega, t, u, s)$
  Calculate $\mathbf{z}_{\text{est}}(\mathbf{z}_t, \mathbf{c}_{\text{text}}, \omega, t, s)$
  Update $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \frac{\partial}{\partial \boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta})$
until converged

C Additional Experiments on Text-to-Sound Generation Task

C.1 Human Evaluation

For further demonstration of the effectiveness of using teacher's network as a feature extractor, which we propose in Section 3.1 and quantitatively evaluate in Table 1, we also conduct a human evaluation. We ask 15 human evaluators to assess the generated audio samples from both overall audio quality (OVL) and relevance to the text prompts (REL). We present 20 samples per model to the participants and asked them to rate the samples on a five-point scale ranging from 5 (Excellent quality) to 1 (Bad quality). The text prompts for the generated samples are randomly picked from AudioCaps testset. We use ConsistencyTTA, SoundCTM w/. l_2 (0-time step), and Ground-truth samples as baselines.

[1]We use the "630k-audioset-best.pt" checkpoint from https://github.com/LAION-AI/CLAP



(a) 1-step generation (b) 2-step generation (c) 8-step generation (d) 16-step generation

Figure 3: FAD for various ω and ν. The solid, dashed, dotted lines indicate the FAD of the students trained with $\omega \sim U[2.0, 5.0]$, the students trained with $\omega \sim U[1.5, 7.0]$, and the teachers.

The results in Table 4 indicate that, firstly, SoundCTM outperforms ConsistencyTTA. Moreover, using d_{teacher} improves sample quality in terms of the adherence to text descriptions. One possible reason for this is that d_{teacher} effectively incorporates textual information as a condition when calculating the distance (see Eq. (4) and Figure 2).

C.2 Influence of Architecture Difference for Feature Extractors

We compare the results of using the teacher's entire network with those of using the first half of it, as shown in Table 5. The results are not significantly different, indicating that there is potential that the model can reduce memory consumption by using only part of the teacher's network without performance degradation.

Table 5: Performance comparison on different architectures for feature extractors

Network architecture	# of steps	$\text{FAD}_{\text{vgg}} \downarrow$	$\text{IS}_{\text{passt}} \uparrow$	CLAP ↑
First half of UNet	1	2.18	7.15	0.43
	4	1.79	7.11	0.44
Entire UNet	1	2.17	7.18	0.43
	4	1.76	7.14	0.45

C.3 Influence of ω and ν for SoundCTM's Sampling

We examine the influence of both ω and ν on SoundCTM's performance. In Figure 3, we compare SoundCTM trained with $\omega \sim U[2.0, 5.0]$ to $\omega \sim U[1.5, 7.0]$ across various ω and ν values during inference. Overall, there is no significant difference in FAD between $\omega \sim U[2.0, 5.0]$ (solid line) and $\omega \sim U[1.5, 7.0]$ (dashed line). This suggests that precisely pre-defining the range of ω for the student training is not necessary. A wider range of ω can be used for student training when prior knowledge of the teacher's dynamics of ω is absent, and high-quality generation can be achieved by adjusting ω and ν during inference.

The students show the better FAD than the best FAD of teachers through ν-interpolation during SoundCTM's sampling. This result could be interpreted as ν allows for generating samples that are more favorable to the FAD in the conditional domain, as the pairs between $\mathbf{c}_{\text{text}}$ and the samples $\mathbf{x}_0$ given the condition are many-to-many.

C.4 Preliminary Experiments of Employing Adversarial Loss for Student Training

Although the original CTMs on image domain demonstrate impressive 1-step image generation performance, this performance heavily depends on an adversarial loss (GAN loss) [10] (See [19, Fig.12]). However, obtaining performance improvements via the GAN loss requires careful selection of a discriminator.

In fact, in our preliminary experiments, even though we employed the several off-the-shelf discriminators in the audio domain [9, 24, 15] they did not lead better performance (we report one of the preliminary results of using GAN loss in Table 6. In this preliminary experiment, we use the discriminators from DAC [24]. Along with the lack of performance improvement from using the GAN loss, there is also a significant increase in memory consumption.). Therefore, in this work, we do not utilize the GAN loss for SoundCTM's training. That said, developing new GAN setups tailored for large-scale conditional sound generation tasks might be worth exploring as future work.

Table 6: Quantitative evaluation of one of the preliminary experiments with and without using GAN loss. Memory consumption is measured with a batch size of two per GPU. DAC's discriminators [24] are used in this experiment. We could not see any benefits of using GAN loss in this setup.

Method	# of steps	$\text{FAD}_{\text{vgg}} \downarrow$	$\text{KL}_{\text{passt}} \downarrow$	$\text{IS}_{\text{passt}} \uparrow$	CLAP ↑	Memory consumption for training
w/o. GAN	2	2.43	1.29	7.50	0.42	46782 MiB
	4	2.28	1.21	7.63	0.43	
w/. GAN	2	2.91	1.37	6.29	0.37	64492 MiB
	4	2.37	1.35	7.53	0.41	

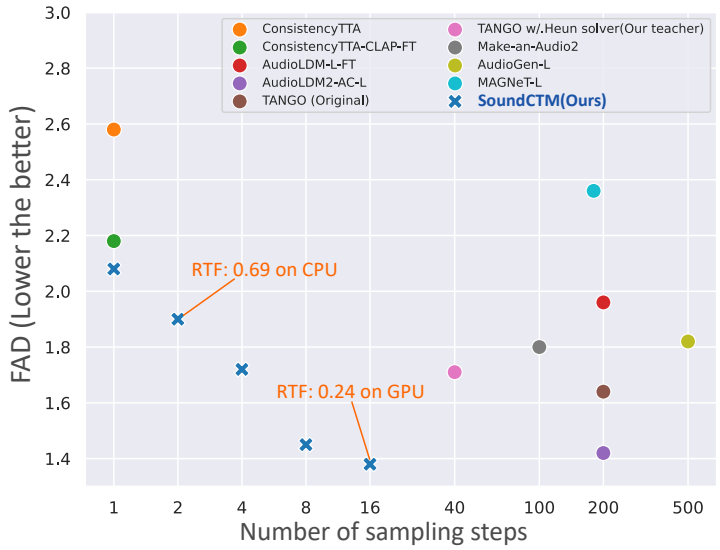


Figure 4: Performance comparison on AudioCaps testset. Real-time factors (RTFs) are measured on a single NVIDIA RTX A6000 and Intel Xeon CPU. SoundCTM can switch 1-step generation and multiple-step generation.

D Towards Training-free Controllable T2S Generation with SoundCTM

D.1 Methodology

Loss-based Guidance Framework for SoundCTM In DMs, the loss-based guidance method involves an additional update $\mathbf{z}_{t-1} = \mathbf{z}_{t-1} - \rho_t \nabla_{\mathbf{z}_t} \mathcal{L}(f(\hat{\mathbf{z}}_0(\mathbf{z}_t)), \mathbf{y}_{\text{condition}})$ during sampling, where $\hat{\mathbf{z}}_0(\mathbf{z}_t)$ is a clean estimate derived from $\mathbf{z}_t$ using Tweedie's formula [6], and ρ_t is a learning rate. We propose replacing $\hat{\mathbf{z}}_0(\mathbf{z}_t)$ with $\mathbf{z}_{0|t} = G_{\boldsymbol{\theta}}(\mathbf{z}_t, \mathbf{c}_{\text{text}}, \omega, t, 0)$ and conducting loss-based sampling within SoundCTM's γ-sampling, as shown in Algorithm 3. We denote the sampling timesteps as $T = t_0 > \cdots > t_N = 0$.

Test-time Optimization-based Framework for SoundCTM In addition, we explore SoundCTM's capabilities for controllable sound generation without additional training. In the music domain, DITTO [29] that optimizing an initial noise latent $\mathbf{z}_T$ of DMs achieves decent performance for training-free controllable music generation. Specifically, $\mathbf{z}_T$ is optimized by

$$\mathbf{z}_T^* = \arg\min_{\mathbf{z}_T} \mathcal{L}(f(\mathbf{x}_0), \mathbf{y}_{\text{condition}}), \tag{9}$$

where $\mathbf{z}_T^*$ is the optimized output, $f(\cdot)$ is a differentiable feature extractor that converts $\mathbf{x}_0$ into the same space as the target condition $\mathbf{y}_{\text{condition}}$, and $\mathbf{x}_0 = \mathcal{D}(\mathbf{z}_0)$. Although DITTO shows promising performance, it requires substantial time to generate a sample since $\mathbf{x}_0$ and $\mathbf{z}_0$ is given by the N-timestep reverse diffusion process in each optimization iteration[2]. Novack et al. [29] also report the performance of a loss-based guidance method [40, 25] as a baseline, which is another major method

[2]In their study, 20 diffusion steps are conducted per iteration, and the number of optimization steps ranges from 70 to 150, resulting in a total of approximately $1,400$ to $3,000$ diffusion steps.

for controllable generation, using the same loss $\mathcal{L}(f(\mathbf{x}_0), \mathbf{y}_{\text{condition}})$, and DITTO outperforms the guidance-based method.

To explore SoundCTM's capability for controllable generation in a training-free manner, we propose two new frameworks for SoundCTM based on $\mathbf{z}_T$-optimization and loss-based guidance. Firstly, we dramatically accelerate $\mathbf{z}_T$-optimization by utilizing SoundCTM's 1-step generation, as shown in Algorithm 4, making each iteration N times faster than DITTO.

Algorithm 3 SoundCTM's Loss-based Guidance Framework

Require: ν, $\mathbf{c}_{\text{text}}$, $\mathbf{y}_{\text{condition}}$, ω, γ, ρ_{t_n}

Start from $\mathbf{z}_{t_0}$
for $n = 0$ to $N-1$ **do**
  $\tilde{t}_{n+1} \leftarrow \sqrt{1-\gamma^2}t_{n+1}$
  Denoise $\mathbf{z}_{\tilde{t}_{n+1}} \leftarrow \nu G_{\boldsymbol{\theta}}(\mathbf{z}_{t_n}, \mathbf{c}_{\text{text}}, \omega, t_n, \tilde{t}_{n+1})$
    $+(1-\nu)G_{\boldsymbol{\theta}}(\mathbf{z}_{t_n}, \varnothing, \omega, t_n, \tilde{t}_{n+1})$
  $\mathbf{z}_{t_N|t_n} = G_{\boldsymbol{\theta}}(\mathbf{z}_{t_n}, \mathbf{c}_{\text{text}}, \omega, t_n, t_N)$
  $\hat{\mathbf{y}}_{\text{condition}} = f(\mathcal{D}(\mathbf{z}_{t_N|t_n}))$
  $\mathbf{z}_{\tilde{t}_{n+1}} = \mathbf{z}_{\tilde{t}_{n+1}} - \rho_{t_n}\nabla_{\mathbf{z}_{t_n}}\mathcal{L}(\hat{\mathbf{y}}_{\text{condition}}, \mathbf{y}_{\text{condition}})$
  Diffuse $\mathbf{z}_{t_{n+1}} \leftarrow \mathbf{z}_{\tilde{t}_{n+1}} + \gamma t_{n+1}\boldsymbol{\epsilon}$
end for
Return $\mathbf{z}_{t_N}$

Algorithm 4 SoundCTM's Optimization-based Training-free Controllable Generation Framework

Require: ν, $\mathbf{c}_{\text{text}}$, ω, $\mathbf{y}_{\text{condition}}$, γ, Learning rate ρ_{t_n}

Sample $\mathbf{z}_{t_0} \sim \mathcal{N}(0, I)$
// Run optimization
for $k = 0$ to $K-1$ **do**
  Denoise $\mathbf{z}_{t_N} \leftarrow G_{\boldsymbol{\theta}}(\mathbf{z}_{t_0}, \mathbf{c}_{\text{text}}, \omega, t_0, 0)$
  $\hat{\mathbf{y}}_{\text{condition}} = f(\mathcal{D}(\mathbf{z}_{t_N}))$
  Update $\mathbf{z}_{t_0} \leftarrow \mathbf{z}_{t_0} - \rho_{t_n}\nabla_{\mathbf{z}_{t_0}}\mathcal{L}(\hat{\mathbf{y}}_{\text{condition}}, \mathbf{y}_{\text{condition}})$
end for
// Run generation from optimized $\mathbf{z}^*_{t_0}$
Start from $\mathbf{z}^*_{t_0}$
for $n = 0$ to $N-1$ **do**
  $\tilde{t}_{n+1} \leftarrow \sqrt{1-\gamma^2}t_{n+1}$
  $\mathbf{z}_{\tilde{t}_{n+1}} \leftarrow \nu G_{\boldsymbol{\theta}}(\mathbf{z}_{t_n}, \mathbf{c}_{\text{text}}, \omega, t_n, \tilde{t}_{n+1})$
    $+(1-\nu)G_{\boldsymbol{\theta}}(\mathbf{z}_{t_n}, \varnothing, \omega, t_n, \tilde{t}_{n+1})$
  $\mathbf{z}_{t_{n+1}} \leftarrow \mathbf{z}_{\tilde{t}_{n+1}} + \gamma t_{n+1}\boldsymbol{\epsilon}$
end for
Return $\mathbf{z}_{t_N}$

By leveraging the anytime-to-anytime jump capability, SoundCTM can achieve both fast $\mathbf{z}_T$-optimization with 1-step sampling and multiple-step controllable generation with loss-based guidance within a single model. This is not possible with either a single Consistency Model (CM) [36]-based T2S model or a DM-based T2S model.

D.2 Training-Free Sound Intensity Control

To validate the proposed framework for training-free controllable T2S generation with SoundCTM, we conduct sound intensity control [29, 38]. This task adjusts the dynamics of the generated sound to match a given target volume line or curve. We follow the experimental protocol from DITTO [29] to control the decibel (dB) volume line or curve of the generated samples.

Experimental Settings We define $f(\mathbf{x}_0) := w * 20\log 10(\text{RMS}(\mathbf{x}_0))$ in Eq. (9), where w represents the smoothing filter coefficients of a Savitzky-Golay filter [32] with a 1-second context window over

Table 7: Quantitative results of sound intensity control on SoundCTM

# of steps	$\mathbf{z}_T$ optimization	Loss guidance	γ	MSE↓	FAD_{vgg} ↓	CLAP ↑
Default 16-step T2S Generation	✗	✗	0	231.9	2.08	0.47
1	✓	✗	0	**6.57**	4.94	0.34
16	✓	✗	0	60.2	3.65	0.38
16	✓	✗	0.2	50.4	3.71	<u>0.41</u>
16	✗	✓	0	18.5	**3.04**	<u>0.41</u>
16	✗	✓	0.2	14.2	<u>3.58</u>	**0.42**
16	✓	✓	0	<u>10.7</u>	4.34	0.37

the frame-wise value, and the RMS is the root mean squared energy of the generated sound. The target condition $\mathbf{y}_{\text{condition}}$ is a dB-scale target line or curve. We perform 70 iterations for $\mathbf{z}_T$-optimization following DITTO's settings. Note that for the loss-based guidance framework, the $\mathbf{z}_T$-optimization is not conducted, which is much efficient. We set $\gamma = 0$ for sampling and evaluate the model performance with student EMA rate $\mu = 0.999$. We use 200 audio-text pairs from the AudioCaps testset for each of the 6 different types of $\mathbf{y}_{\text{condition}}$ as shown in Figure 5 (a) and Figure 7 (a). We evaluate our framework using the mean squared error (MSE) between the target and obtained $\mathbf{y}_{\text{condition}}$ to measure the accuracy of dynamics control, the FAD between generated samples and the entire AudioCaps testset, and the CLAP score.

During the SoundCTM's optimization-based framework, we use Adam [20] with a learning rate of 1.0. We also tested learning rates of 1.0×10^{-1}, 1.0×10^{-2}, and 5.0×10^{-3} by following DITTO. However, we cannot obtain better results than the case using 1.0. For the time-dependent learning rate ρ_t in SoundCTM's loss-based guidance framework, we use the overall gradient norm by following DITTO. We set $\nu = 1$ for 1-step generation with $\mathbf{z}_T$-optimization, $\nu = 2$ for both 16-step generation with $\mathbf{z}_T$-optimization and 16-step loss-based guidance, and $\omega = 3.5$ for all the settings. The same values of $\nu = 2$ and $\omega = 3.5$ are used for the default 16-step T2S generation. For time discretization in multi-step generation, we use the same scheme as in T2S generation evaluation.

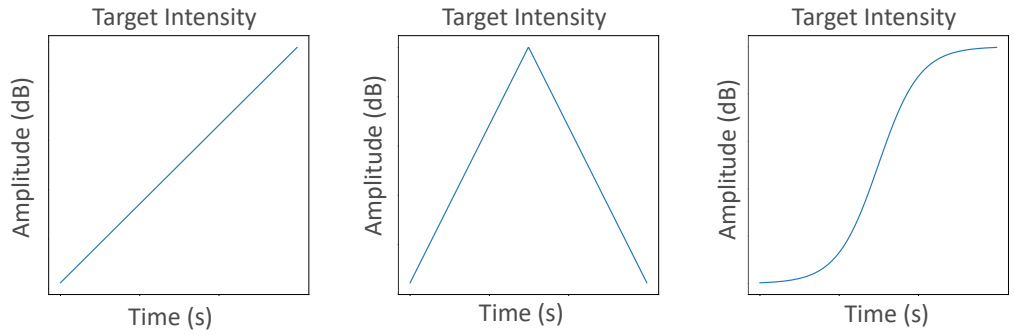
Experimental Results In Table 7, we compare results with various settings for SoundCTM's controllable generation framework, as DITTO is not open-sourced. We demonstrate the cases of using 1). only the optimization, 2). only the loss-based guidance, and 3). the loss-based guidance after the optimization. We also report the case where we use stochastic sampling ($\gamma = 0.2$).

Firstly, overall, both $\mathbf{z}_T$-optimization and loss-based guidance frameworks effectively control intensity as indicated by the MSE results and the intensity curve shown in Figures 5 to 8. However, the FAD and CLAP scores are worse than those of the default T2S generation case. This is likely due to a degradation in auditory quality and a shift in the volume-conditioned audio distribution away from that of the AudioCaps testset.

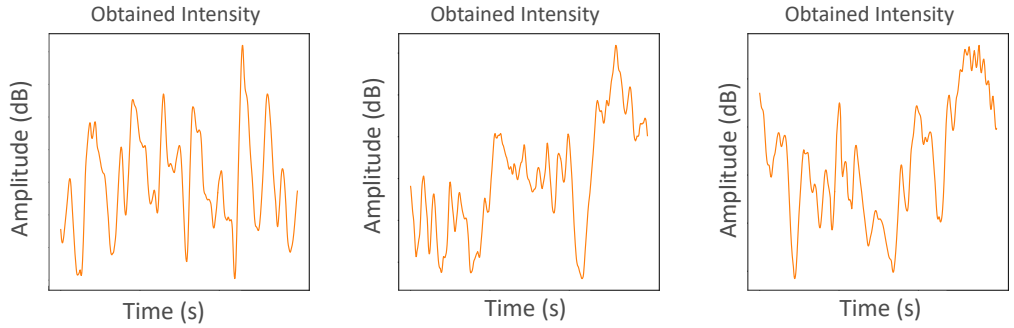
Except for the MSE of 1-step with $\mathbf{z}_T$-optimization case, the loss-based guidance method outperforms $\mathbf{z}_T$-optimization method. Under 16-step cases, the loss-based guidance method shows better results than those of the optimization-based method across all the metrics. Interestingly, this finding contrasts with DITTO's report that the optimization-based method outperforms the guidance-based one in terms of the MSE [29, Table 3]. The difference can be attributed to the performance improvement of the loss-based guidance method by using $G_{\boldsymbol{\theta}}(\mathbf{z}_t, \mathbf{c}_{\text{text}}, \omega, t, 0)$ instead of $\hat{\mathbf{z}}_0(\mathbf{z}_t)$, which provides a more accurate estimate of $\mathbf{z}_{0|t}$, resulting in more effective guidance during sampling.

Stochastic sampling ($\gamma = 0.2$) did not yield significant performance improvement. Integrating the loss-based guidance framework with $\mathbf{z}_T$-optimization showed the better MSE, consistent with DITTO's findings, but resulted in the much worse FAD and CLAP scores compared with only using the loss-based guidance.

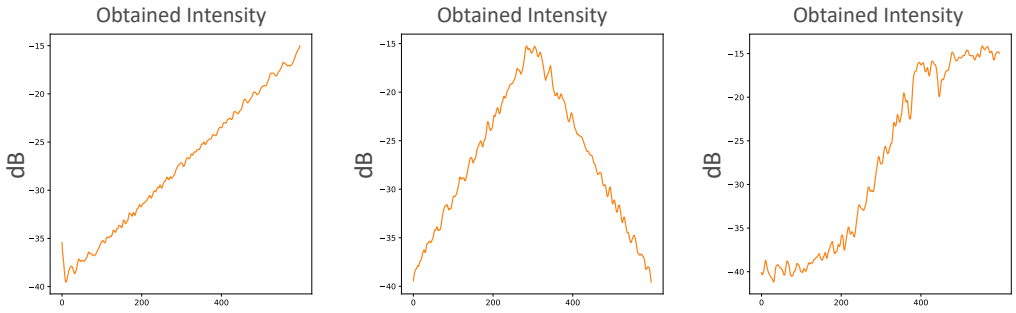
Considering both the qualitative results in Figures 5 to 8 and the quantitative results in Table 7, the loss-based guidance is the effective strategy for training-free controllable generation with SoundCTM.



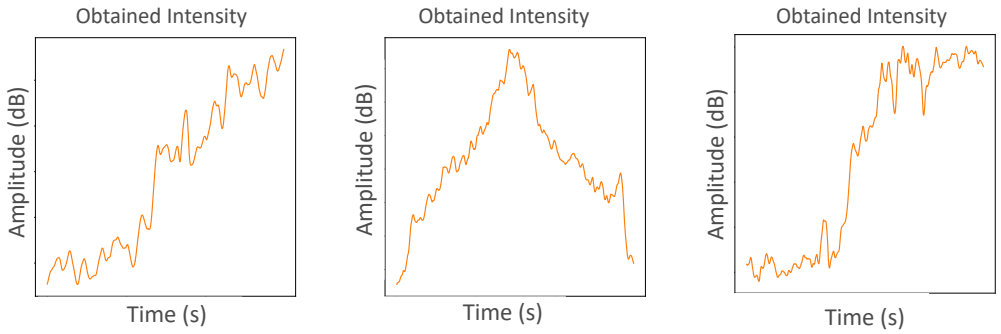
(a) Target intensities



(b) Default T2S generation

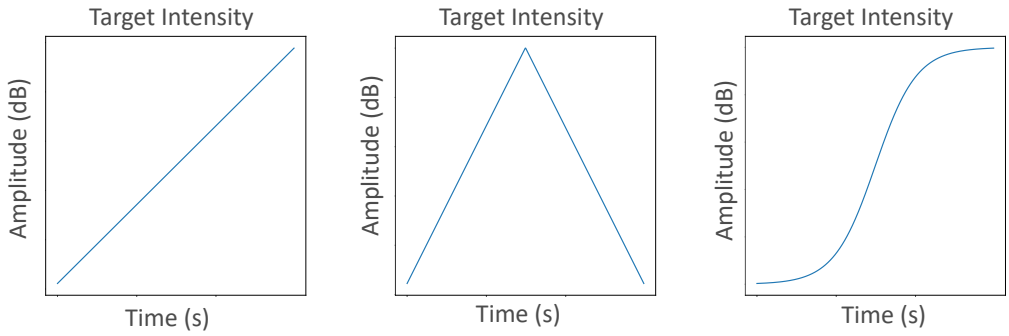


(c) $\mathbf{z}_T$-optimization with 1-step generation

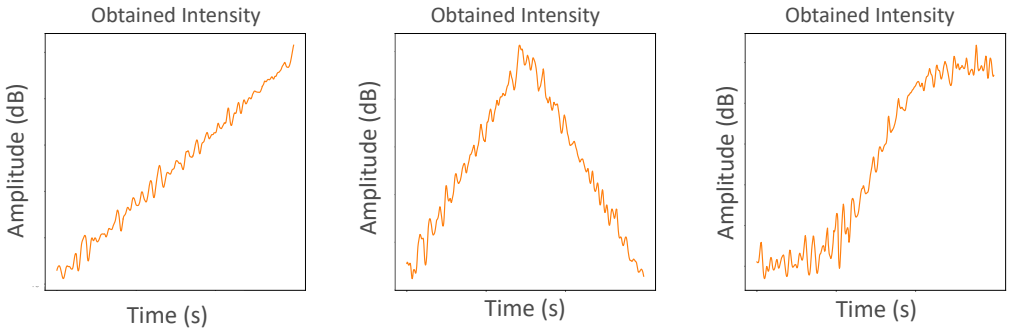


(d) $\mathbf{z}_T$-optimization with 16-step generation

Figure 5: Target sound intensities and obtained intensities. We use the same text prompt within each column and different prompts across different columns. Note that we use 70 iterations for $\mathbf{z}_T$-optimization.

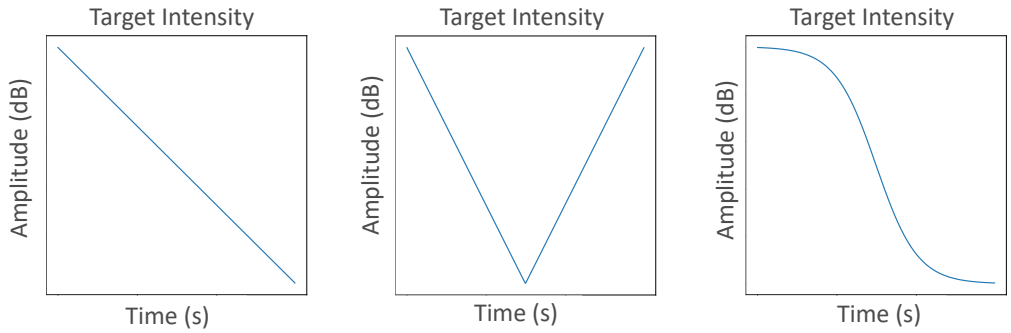


(a) Target intensities

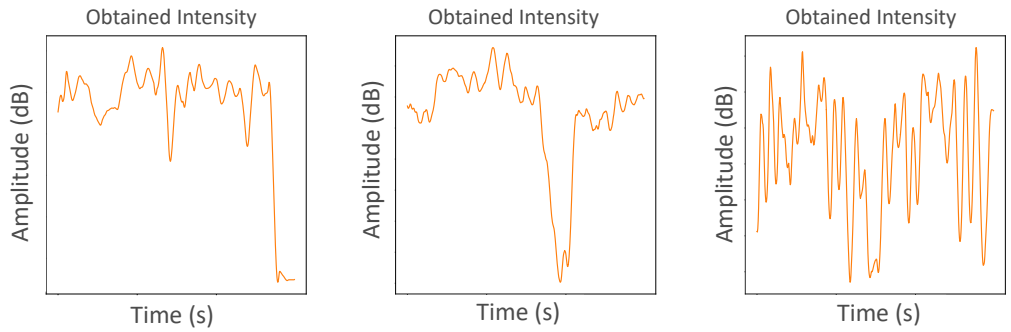


(b) Loss-based guidance with 16-step generation

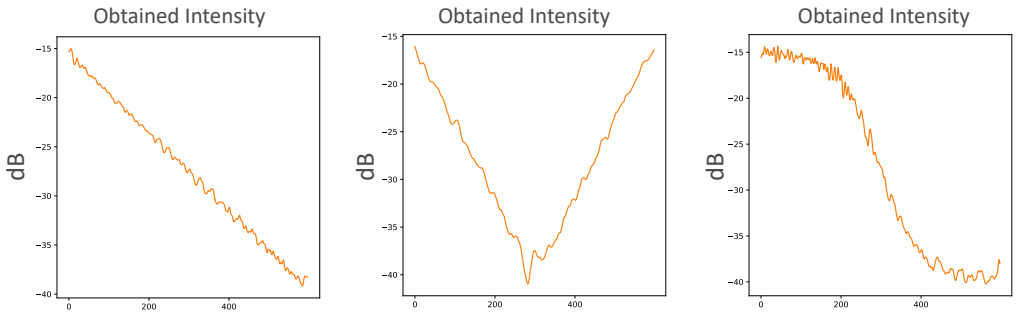
Figure 6: Target sound intensities and obtained intensities. We use same text prompts within each column and different prompt for each different column.



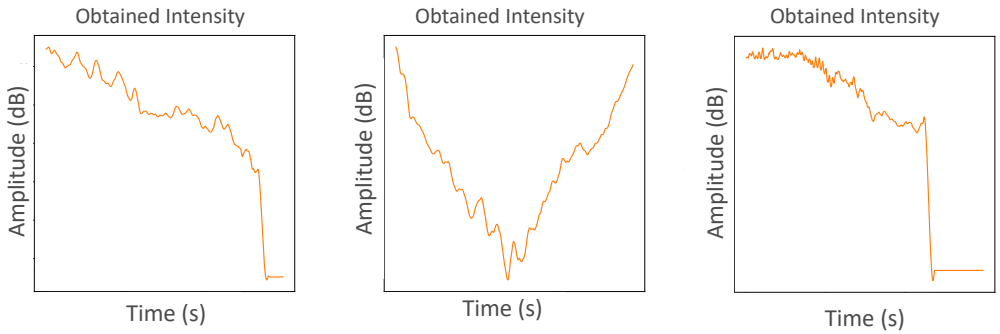
(a) Target intensities



(b) Default T2S sound generation

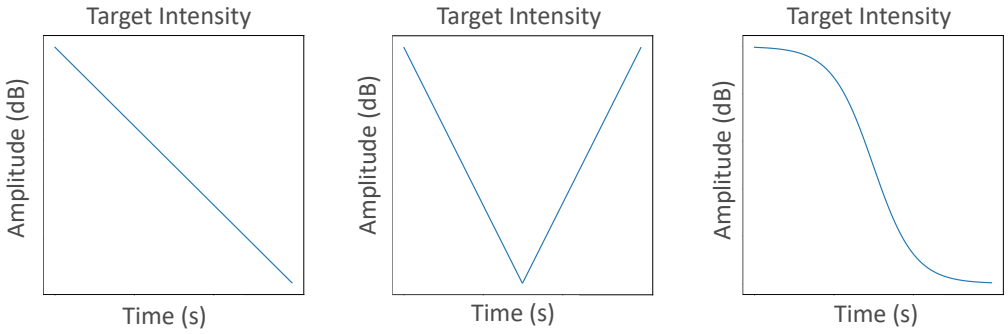


(c) $\mathbf{z}_T$-optimization with 1-step generation

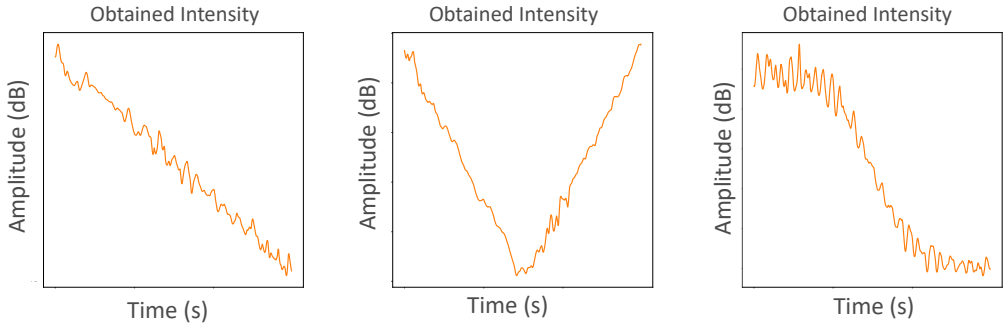


(d) $\mathbf{z}_T$-optimization with 16-step generation

Figure 7: Target sound intensities and obtained intensities. We use the same text prompt within each column and different prompts across different columns. Note that we use 70 iterations for $\mathbf{z}_T$-optimization.

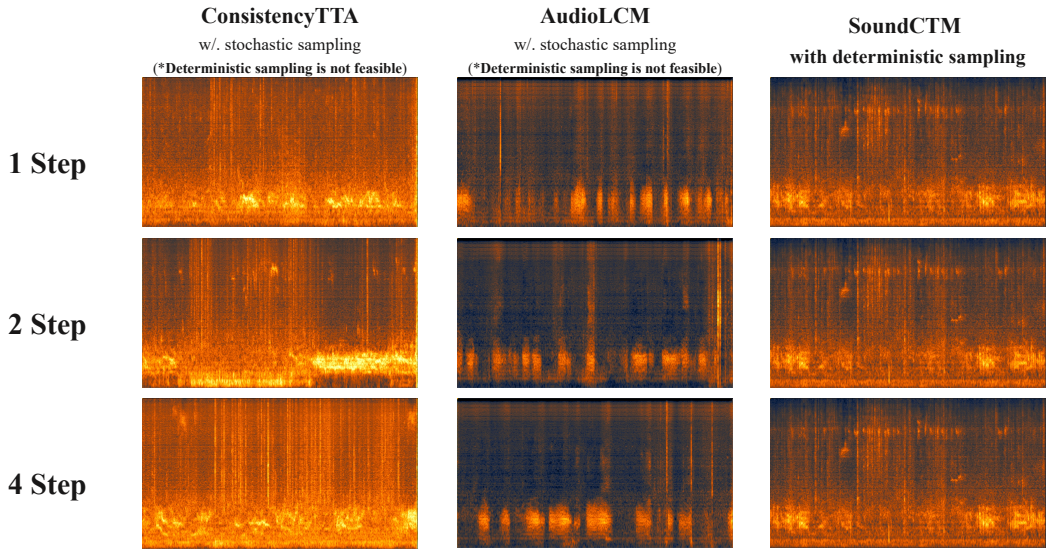


(a) Target Intensities

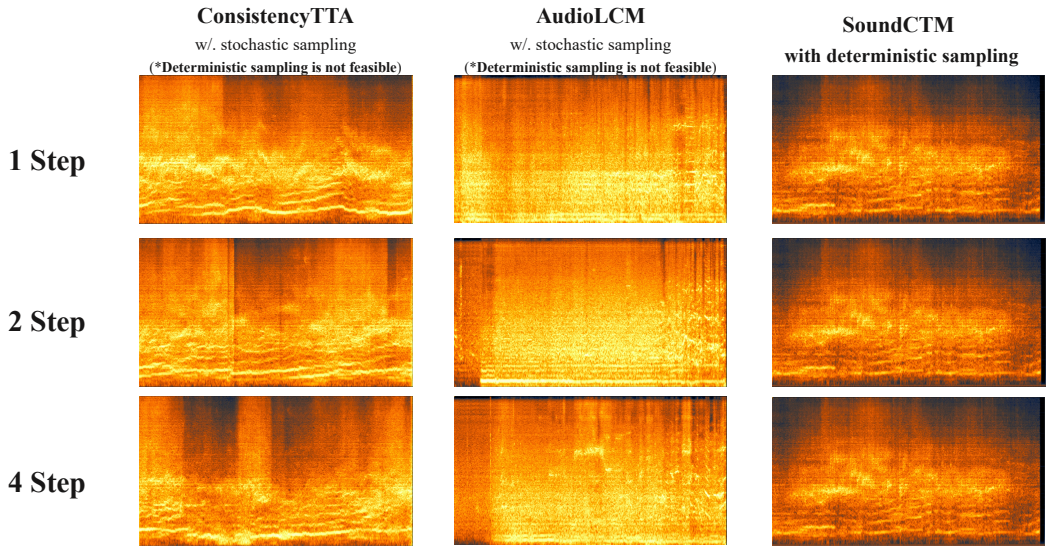


(b) Loss-based guidance with 16-step generation

Figure 8: Target sound intensities and obtained intensities. We use the same text prompt within each column and different prompts across different columns.



(a) Input text prompt: "Thunder claps, and hard rain falls and splashes on surfaces."



(b) Input text prompt: "Birds cooing and rustling."

Figure 9: Visualization of spectrograms for generated samples using 16-step, 2-step, and 1-step generation with ConsistencyTTA [2] and SoundCTM. As ConsistencyTTA, a CD-based model, inherently does not support deterministic sampling, the content of the generated samples varies when increasing the number of sampling steps, even when using the same initial noise and text prompts. This variability makes it challenging for sound creators to control the output. In contrast, SoundCTM with deterministic sampling ($\gamma = 0$) is able to maintain consistent content as the number of sampling steps increases.