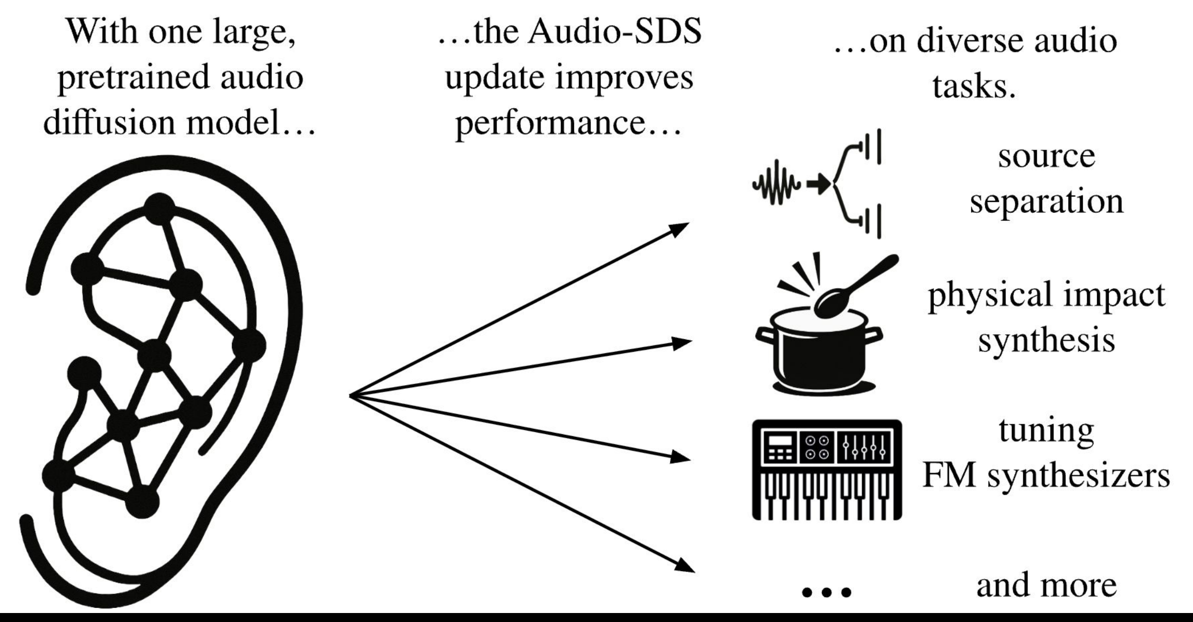
Score Distillation Sampling for Audio: Source Separation, Synthesis, and Beyond

Jessie Richter-Powell, Antonio Torralba, Jonathan Lorraine

ICML International Conference On Machine Learning · NVIDIA · MIT CSAIL

Introduction

- We adapt Score Distillation Sampling (SDS), originally for 3D generation [1], to audio models.



Our Method

- High-level idea for the SDS update:

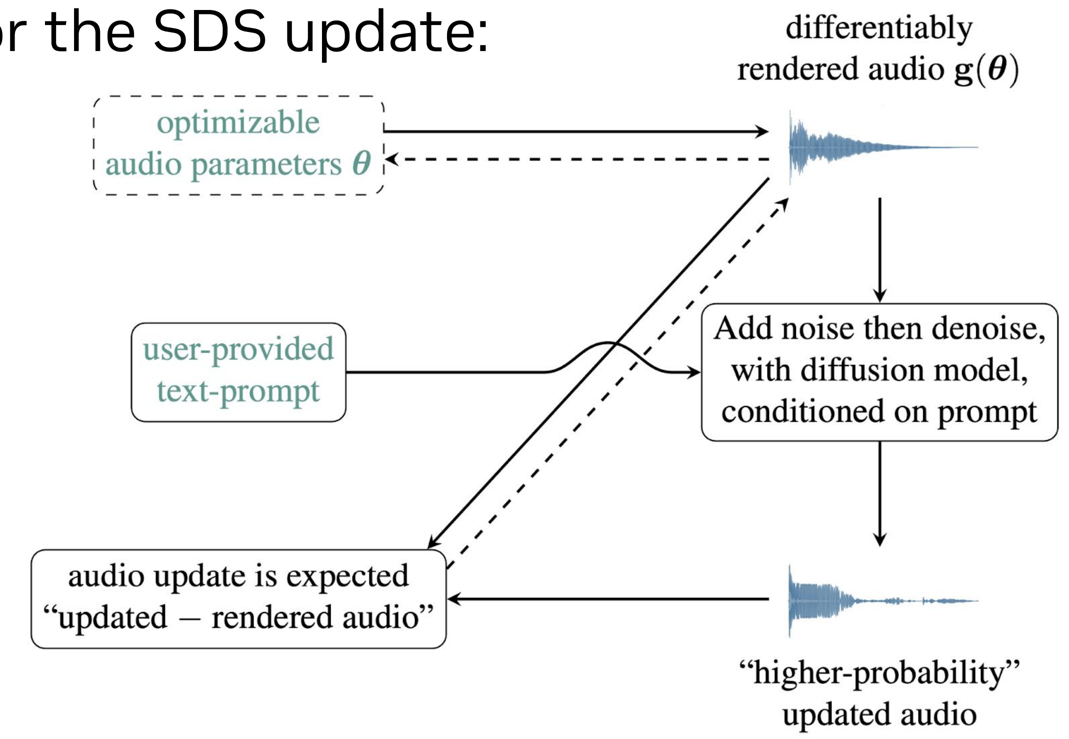
Finds parameters $\boldsymbol{\theta}$ that render audio $\mathbf{x}$ the diffusion model finds probable.

Rendered Audio:

$$\mathbf{x} = \mathbf{g}(\boldsymbol{\theta}, \boldsymbol{c})$$

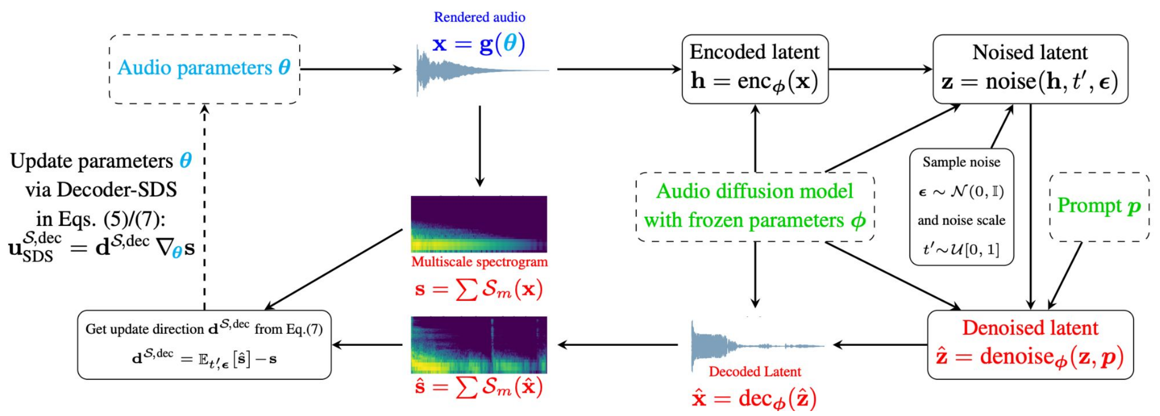
SDS update (roughly):

$$\mathbf{u}^{\mathrm{dec}}_{\mathrm{SDS}} = (\mathbb{E}[\hat{\mathbf{x}}] - \mathbf{x})\nabla_{\boldsymbol{\theta}}\mathbf{x}$$



- All the gory details for SDS and our audio-variant:

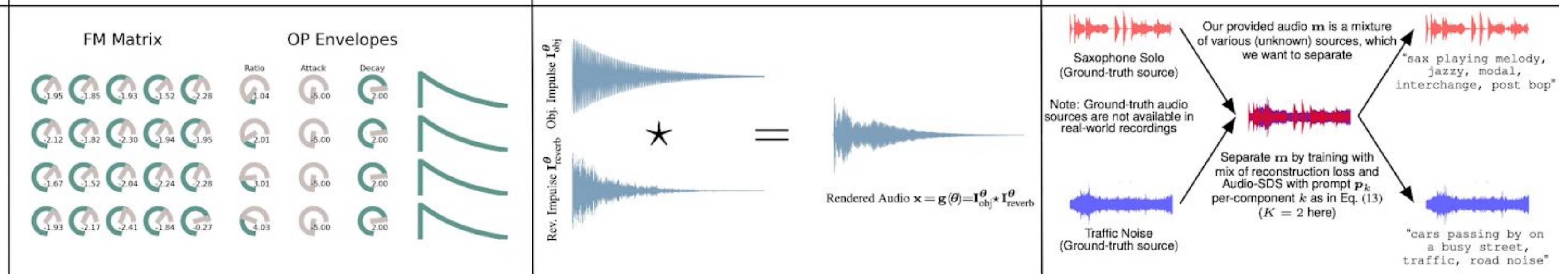
Full-SDS Update: $\mathbf{u}_{\mathrm{SDS}}(\boldsymbol{\theta};\boldsymbol{p}) = \mathbb{E}_{t',\boldsymbol{\epsilon},\boldsymbol{c}}\left[\omega(t')\left(\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\phi}}(\mathbf{z}(\boldsymbol{\theta},\boldsymbol{c}),t',\boldsymbol{p}) - \boldsymbol{\epsilon}\right)\nabla_{\boldsymbol{\theta}}\mathbf{z}(\boldsymbol{\theta},\boldsymbol{c})\right]$



Our SDS-variant with spectrogram emphasis:

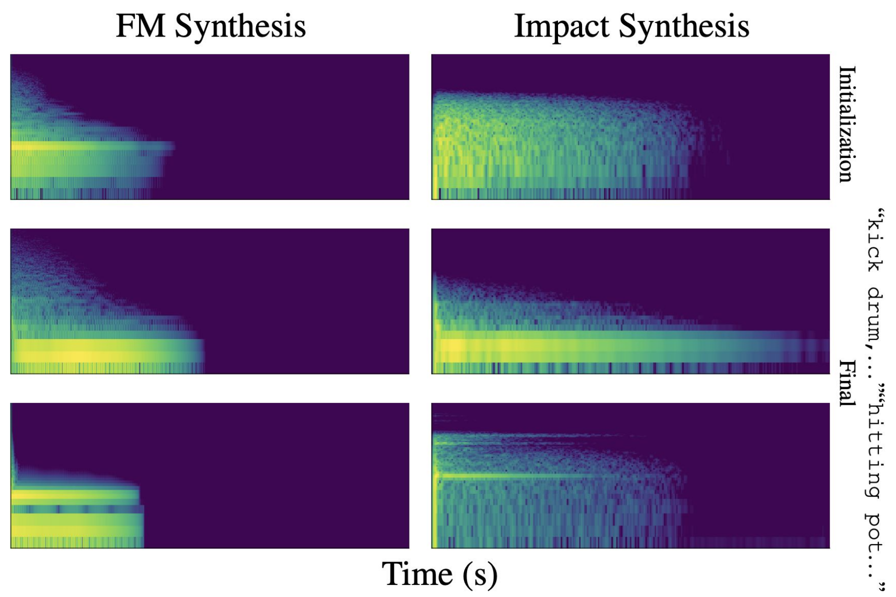
$$\mathbf{u}^{\mathcal{S},\mathrm{dec}}_{\mathrm{SDS}} = (\mathbb{E}[\hat{\mathbf{s}}] - \mathbf{s})\nabla_{\boldsymbol{\theta}}\mathbf{s}$$

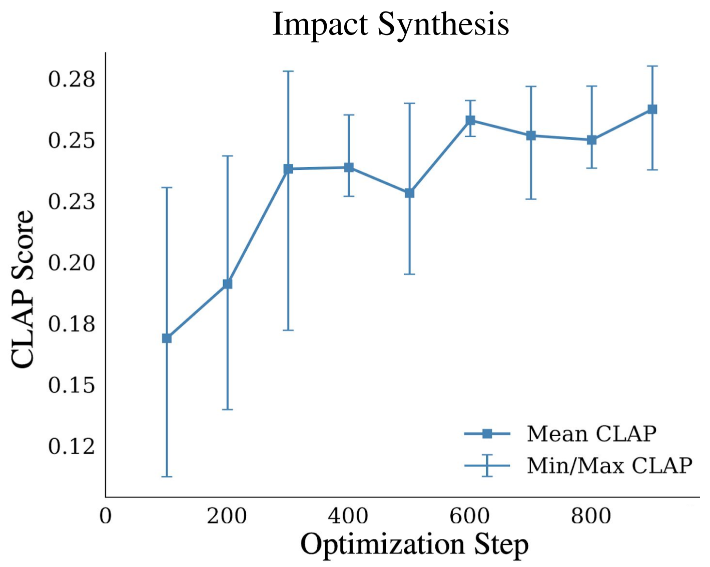
Our Proposed Tasks

Task	FM Synthesis	Impact Synthesis	Source Separation
Use-Case	Toy setup, to generate FM synthesizer settings consistent with prompts like "*kick drum, bass, reverb*"	Generate impacts consistent with prompts like "*hitting pot with wooden spoon*"	Prompt-conditional source separation for a source audio, such as separating out a "*sax …*" and "*cars …*" from a music recording on a road
Optimizable Parameters $\boldsymbol{\theta}$	Envelope params., fundamental freq., FM Matrix $\boldsymbol{\theta} = \{\alpha_v, \delta_v, \omega_v\}_{v=1}^{V}, \mathbf{A}$	Frequency, damping, amplitude of sinusoids $\boldsymbol{\theta} = \{\lambda_n, d_n, a_n, \tilde{\lambda}_n, \tilde{d}_n, \tilde{a}_n\}_{n=1}^{N}$	Latent / raw audio for each source $\boldsymbol{\theta} = \{\mathbf{x}_k\}_{k=1}^{K}$
Rendering Function $\mathbf{g}(\boldsymbol{\theta})$	Sine oscillators modulate each other's frequency $f_v(t) = \max(0, \min(t/(\alpha_v + 10^{-5}), \exp(\alpha_v - t)/\delta_v^2)(\delta_v - t - \alpha_v)/\delta_v)$ $\mathbf{u}_v[t] = \sin(t \cdot \omega_v + \langle \mathbf{A}_v, \mathbf{u}[t-1]\rangle) f_v(t)$ $\mathbf{g}(\boldsymbol{\theta})[t] = \langle \mathbf{A}_v, \mathbf{u}_{t-1}\rangle$	Convolution of impact, object and reverb impulse $\mathbf{I}^{\boldsymbol{\theta}}_{\mathrm{obj}}[t] = \sum_{n=1}^{N} a_n \exp(-d_n t) \cos(\lambda_n t)$ $\mathbf{I}^{\boldsymbol{\theta}}_{\mathrm{reverb}}[t] = \sum_{n=1}^{N} \tilde{a}_n \exp(-\tilde{d}_n t)\, F(\mathcal{W}(t), \tilde{\lambda}_n)$ $\mathbf{I}_{\mathrm{impact}}$ = Delta function $\mathbf{g}(\boldsymbol{\theta}) = \mathbf{I}_{\mathrm{impact}} \star \mathbf{I}^{\boldsymbol{\theta}}_{\mathrm{obj}} \star \mathbf{I}^{\boldsymbol{\theta}}_{\mathrm{reverb}}$	Simply the sum of audio over sources $\mathbf{g}(\boldsymbol{\theta}) = \sum_{k=1}^{K} \mathbf{x}_k$
Parameter Update	Audio-SDS	Audio-SDS	Reconstruction Loss Gradient + $\gamma \cdot$ Audio-SDS
Visualization			

Experimental Results

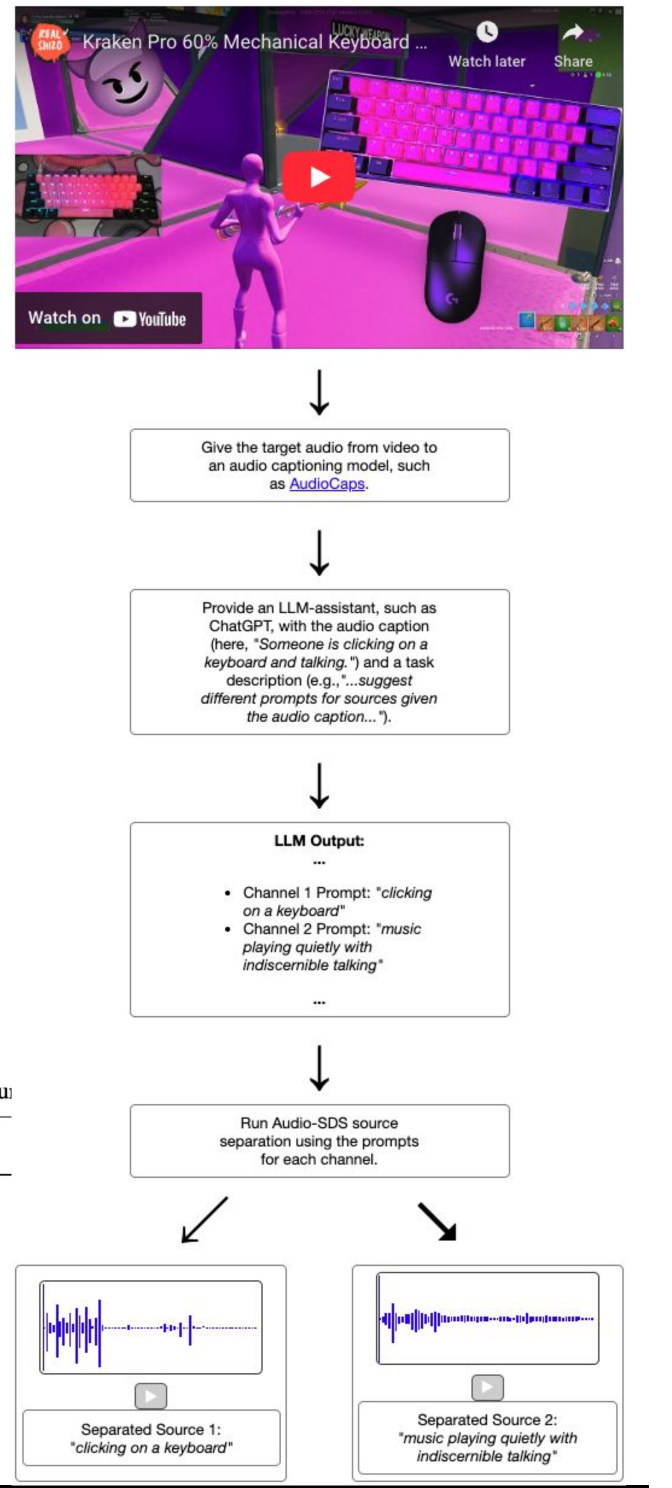
- We assess quality with audio-prompt alignment via CLAP
- FM Synthesis works for simple prompts
- Impact synthesis improves CLAP on impact-oriented prompts
- Ablations provided in the paper





Experimental Results

- **Prompt-guided Source Separation:** A user takes mixed audio and chooses a set of prompts for channels to split it.
- We improve SDR to ground truth and CLAP when ground truth unavailable.
- **Fully-automatic pipeline:** We take a video, caption it with a model, and provide that to an LLM-assistant suggesting source decompositions.



Mixture	SDRs for reconstructed (source$_1$, source$_2$, mixture) Initialization	Us
Traffic + Sax	$(-0.7, -5.2, 3.2)$	$(\mathbf{8.5}, \mathbf{8.1}, \mathbf{13.1})$
Bongo + Waves	$(1.2, -6.4, 3.9)$	$(\mathbf{1.2}, -\mathbf{2.1}, \mathbf{8.7})$
Pipes + Glass	$(-4.2, 0.0, 2.8)$	$(\mathbf{1.5}, \mathbf{6.5}, \mathbf{7.7})$
Clock + Bongo	$(-15.8, \mathbf{10.4}, 11.8)$	$(-\mathbf{10.4}, 3.6, \mathbf{13.9})$
Wind + Pipes	$(-6.1, 1.9, 5.8)$	$(-\mathbf{0.5}, \mathbf{8.6}, \mathbf{7.3})$

Mixture	CLAPs for reconstructed source$_1$ and source$_2$, then SDR for mixtu Initialization	Us
Traffic + Sax	$(0.17, 0.02, 3.2)$	$(\mathbf{0.2}, \mathbf{0.05}, \mathbf{13.1})$
Bongo + Waves	$(0.15, \mathbf{0.14}, 3.9)$	$(\mathbf{0.16}, 0.08, \mathbf{8.7})$
Pipes + Glass	$(\mathbf{0.25}, 0.27, 2.8)$	$(0.21, \mathbf{0.3}, \mathbf{7.7})$
Clock + Bongo	$(0.22, 0.24, 11.8)$	$(\mathbf{0.30}, \mathbf{0.34}, \mathbf{13.9})$
Wind + Pipes	$(0.25, 0.06, 5.8)$	$(0.25, \mathbf{0.09}, \mathbf{7.3})$

Our View of the Future

- Limitations to improve on:
  - **Clip-Length Budget:** We optimized on ≤10 s clips; minute-scale audio may have artifacts or blow up memory.
  - **Audio-Model Bias:** We use Stable Audio Open, so when this struggles, e.g., on rare instruments, speech, so do we.
- **Future:** Easily use one video + audio diffusion model with SDS-style updates for various tasks — impacts, lighting, cloth, fluids, and more.

Links

- Webpage in QR-code: research.nvidia.com/labs/toronto-ai/Audio-SDS/

- [1] Poole, Ben, et al. "DreamFusion: Text-to-3D using 2D Diffusion." The Eleventh International Conference on Learning Representations.