One-Stage Multi-Task Instruction-Guided 3D Spatial Audio Editing

Anonymous ACL submission

Abstract

Spatial audio editing modifies an existing soundfield according to a user's instruction while preserving the rest of the scene. Unlike conventional audio editing, it must reason jointly about audio events, spatial information, dynamic changes, and environmental information in first-order Ambisonic (FOA) waveforms. Existing language-guided editors mainly target conventional audio or rely on sequential operations, and therefore do not directly support one-stage editing for complex 3D spatial instructions. We present SPATIALEDIT, the first one-stage multi-task framework for instruction-guided 3D FOA spatial audio editing. We build paired FOA supervision from open-source speech and sound-effect corpora using controllable room simulation, covering more than ten single-operation and compound tasks across the four editing axes. To handle this heterogeneous edit space, SPATIALEDIT uses Spatial Edit Mixture-of-Experts (SE-MoE) with dual-level routing, selecting task-aware expert combinations for compound instructions and frame-level routed/null experts for local edit decisions. We further introduce Spatial Preference Optimization (SPO), a Direct Preference Optimization (DPO)-based alignment objective with edit-specific negative targets, and adopt staged training to improve natural-language grounding. Experiments show that SPATIALEDIT achieves better editing quality than existing general audio editors and spatial audio baselines across all tasks. Spatial audio editing demos can be found at https://spatial-edit.github.io/.

1 Introduction

Spatial audio is increasingly used in immersive media, including virtual and augmented reality, telepresence, games, and film production (Zhang et al., 2017; Poeschl et al., 2013; Broderick et al., 2018). It turns audio into a scene: sources occupy locations, move over time, and interact with the surrounding space. This makes editing especially useful. In production, a scene is often close to the desired result but still needs a targeted change to its content, source layout, motion, or room impression. Editing should therefore operate on an existing spatial scene, applying only the requested change instead of rebuilding the entire sound field.

Natural-language instructions provide a convenient interface for this setting, since users can describe the intended edit without specifying a full target scene. Progress in audio-language models and spoken dialogue benchmarks has advanced audio-text representation learning and acoustic-language understanding (Hu et al., 2025b; Cheng et al., 2025), and language-guided audio editing methods have made clear progress for conventional audio, where a model must ground the target event, modify it, and avoid drifting unedited regions (Wang et al., 2023; Manor and Michaeli, 2024; Jia et al., 2024). Spatial editing, however, is not just the same problem with more channels. As shown in Figure 1, an instruction may target audio events, spatial information, dynamic changes, or environmental information, and these factors are coupled in the waveform. Moving a source farther away, for example, can change its energy, directionality, reverberation, and masking with other sources. Meanwhile, unedited sources should preserve their content, trajectories, and room-acoustic context. These requirements make instruction-guided spatial audio editing distinct from general audio editing.

Currently, spatial audio editing still faces two main challenges. 1) **One-stage generation.** Existing language-guided editing methods can synthesize an edited waveform, but they mainly target monaural or conventional audio operations and do not treat 3D spatial variables as editable controls (Wang et al., 2023; Manor and Michaeli, 2024; Jia et al., 2024). SmartDJ is closer to spatial editing, but it operates on stereo audio and decomposes a high-level request into sequential atomic operations

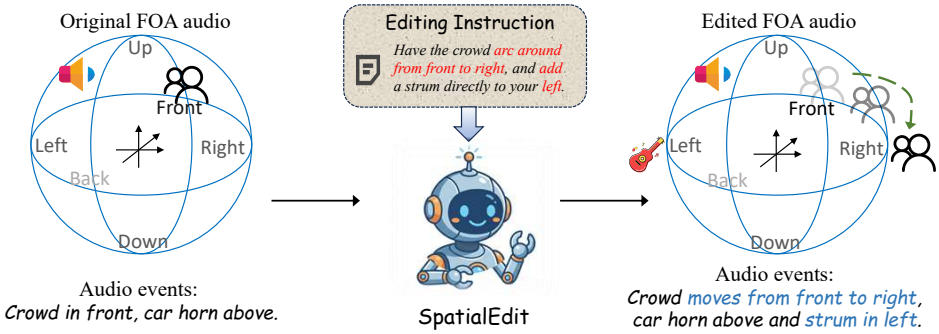

Figure 1: Overview of SPATIALEDIT, which follows natural-language instructions to edit FOA audio, such as moving a crowd from the front to the right and adding a strum on the left while preserving unchanged events.

(Lan et al., 2025). Such a pipeline does not directly learn the joint acoustic consequence of a compound spatial instruction, and errors may accumulate across stages. 2) **Task coverage.** Spatialization and spatial audio generation methods model directionality, localization, or video-conditioned spatial sound (Richard et al., 2021; Leng et al., 2022; Li et al., 2024; Pan et al., 2025; Liu et al., 2025; Kim et al., 2025), and SALM provides representation-level spatial-language manipulation (Hu et al., 2025a). These methods are not waveform-level editors for existing first-order Ambisonic (FOA) scenes. To our knowledge, prior work has not conducted FOA editing covering difficult 3D and dynamic operations, including audio-event edits, 3D relocation, dynamic trajectories, and environmental changes.

To address these challenges, we present SPATIALEDIT, the first one-stage multi-task framework for instruction-guided 3D FOA spatial audio editing. We build paired FOA supervision from open-source speech and sound-effect corpora using controllable room simulation, covering over ten single-operation and compound tasks across audio events, spatial information, dynamic changes, and environmental information. On the model side, we introduce Spatial Edit Mixture-of-Experts (SE-MoE), which uses dual-level routing to choose task-aware expert combinations for compound instructions and frame-level routed/null experts for local edit decisions, improving one-stage edit quality across heterogeneous tasks and events. We further introduce Spatial Preference Optimization (SPO), a Direct Preference Optimization (DPO)-based alignment objective with edit-specific negative targets, to improve instruction following for challenging spatial edits. Finally, we adopt a staged training strategy: text-to-FOA pretraining first builds semantic grounding for spatial audio, and single-operation and compound-edit training then gradually teach the model one-stage editing for complex compound tasks, thereby improving its understanding of natural-language instructions. Experimental results show that SPATIALEDIT achieves better editing quality than existing general audio editors and spatial audio baselines across all tasks.

Our contributions are summarized as follows:

- We introduce SPATIALEDIT, the first one-stage multi-task framework for instruction-guided 3D FOA spatial audio editing, supported by paired FOA supervision over single-operation and compound tasks.
- We propose SE-MoE with dual-level routing, which combines instruction-level task experts with frame-level routed/null experts to improve one-stage edit quality across heterogeneous tasks and audio events.
- We introduce SPO, a DPO-based alignment objective with edit-specific negative targets, and a staged training strategy that improves natural-language grounding for complex spatial edits.
- Experimental results show that SPATIALEDIT achieves better editing quality than existing baselines across all tasks.

2 Related Work

2.1 Audio Editing

Audio editing aims to modify an existing recording while preserving the parts that are not targeted by the edit. Recent text-to-audio models provide strong generative priors for open-domain sound synthesis (Kreuk et al., 2022; Huang et al., 2023; Liu et al., 2023, 2024), and language-guided editing methods build on such priors to generate an edited output from an input clip and a textual

condition. AUDIT constructs triplets of input audio, instruction, and edited audio, and trains a latent diffusion model for adding, dropping, replacement, inpainting, and super-resolution (Wang et al., 2023). ZETA and ZEUS explore zero-shot editing with DDPM inversion, using either text prompts or unsupervised semantic directions (Manor and Michaeli, 2024), while AudioEditor adapts diffusion-inversion techniques for training-free editing with attention to precise edits and preservation (Jia et al., 2024). SmartDJ is closest to our setting: it extends editing to stereo scenes and uses an audio-language-model planner to decompose high-level requests into sequential atomic operations executed by a diffusion editor (Lan et al., 2025). These works establish important editing paradigms, but their edit spaces remain largely semantic or stereo-level. They do not jointly model 3D spatial attributes, dynamic trajectories, and room-acoustic transformations as editable targets. In contrast, SPATIALEDIT treats audio events, spatial information, dynamic changes, and environmental information as a coupled edit space and directly generates the final edited 3D scene in one stage.

2.2 Spatial Audio Generation

Spatial audio generation and spatialization aim to synthesize audio with directional and environmental cues. Early learning-based systems studied spatial sound from visual or geometric context, including visual sound localization and audio-visual navigation environments (Gao and Grauman, 2019; Chen et al., 2020, 2022a). Binaural and text-guided spatialization methods convert mono or content-conditioned audio into spatial audio by modeling listener- or text-specified localization cues (Richard et al., 2021; Leng et al., 2022; Li et al., 2024; Pan et al., 2025); recent video-to-spatial audio systems further generate spatial audio from panoramic or silent video inputs (Liu et al., 2025; Kim et al., 2025). Another related line studies spatial audio representations and spatial audio-language alignment: spatial datasets and SELD-style annotations provide event and direction labels (Shimada et al., 2023), and SALM builds structured semantic and spatial embeddings that enable directional manipulation in representation space (Hu et al., 2025a). These methods are useful for spatial understanding, localization, spatialization, or generation, but they do not directly solve instruction-guided editing of an existing 3D sound scene. SPATIALEDIT instead takes an existing spatial mixture, applies a requested audio-event, spatial, dynamic, or environmental edit, and preserves the rest of the scene.

3 Method

3.1 Problem formulation

Let $a_{\text{src}} \in \mathbb{R}^{4\times T}$ and $a_{\text{tgt}} \in \mathbb{R}^{4\times T}$ denote the source and target first-order Ambisonic (FOA) audio, respectively, where the four channels encode the spatial soundfield and T is the waveform length. Given a natural-language editing instruction c, such as *Insert a motorcycle sound behind on the left*, our goal is to learn an editor

$$\hat{a}_{\text{tgt}} = F_\theta(a_{\text{src}}, c), \tag{1}$$

so that the generated audio $\hat{a}_{\text{tgt}}$ follows the requested edit while preserving unrelated sound events, spatial cues, and room characteristics.

Spatial editing can change what sounds, where it is located, how it evolves over time, or what acoustic space it occupies. We group the tasks along four axes:

- **Audio events**: add, remove, replace, extract, enhance, or attenuate a sound event.
- **Spatial information**: edit the spatial attributes of a source, including azimuth, elevation, distance, and source layout.
- **Dynamic changes**: apply time-varying gain changes or dynamic spatial trajectories, such as angular motion and distance motion.
- **Environmental information**: change room-acoustic conditions such as perceived room size and reverberation while keeping the scene content and source layout consistent unless otherwise instructed.

These axes are coupled in the waveform. A compound instruction should therefore be learned as a direct source-to-target transformation rather than as a chain of separately executed operations. SPATIALEDIT follows this formulation and directly predicts the final edited FOA scene in one stage.

3.2 Data construction

One-stage editing requires paired supervision in which the target already contains the complete acoustic consequence of the instruction. We construct such supervision from open-source speech and sound-effect corpora, including AudioCaps

(Kim et al., 2019), FSDKaggle2019 (Fonseca et al., 2019), PicoAudio (Xie et al., 2024), LibriSpeech (Panayotov et al., 2015), and Spatial LibriSpeech (Sarabia et al., 2023). For each example, we sample two to four audio events to form a source scene and synthesize its target scene according to the requested edit. Controllable room simulation provides FOA rendering with specified directions, distances, motion patterns, and room-acoustic conditions (Scheibler et al., 2018).

Each training item is a triplet $(a_{\text{src}}, c, a_{\text{tgt}})$ consisting of a source FOA waveform, an editing instruction, and the final target FOA waveform. The target is rendered after all requested changes have been applied, so the model can learn compound edits without observing intermediate operations. The task set covers more than ten single-operation and compound tasks across audio events, spatial information, dynamic changes, and environmental information. We generate about 125K triplets for each single-operation task and 250K triplets for compound editing; each example is approximately 8.9 seconds long. To reduce dependence on a fixed instruction style, we generate three paraphrases for each edit instruction using Gemini 2.5 Pro (Comanici et al., 2025). Additionally, in the test set, we include 50 examples for each single-operation edit type, and separately synthesize 50 examples each for two-operation, three-operation, and four-operation edits. The test set and training set do not share original audio clips to prevent data leakage.

The same data also supports the staged training curriculum described below. Before editing training, we decompose each source-target pair into scene-level FOA-caption examples, where the caption describes a complete spatial scene. These pairs initialize the model with text-to-FOA generation ability and with language grounding for events, spatial layout, motion, and room acoustics. Editing triplets are then used for single-operation training and, finally, compound-edit training.

3.3 Model overview

Editing FOA waveforms directly would require the model to synthesize long four-channel signals while preserving fine spatial cues. We instead edit in a compact latent space, where the generator can focus on the requested transformation rather than low-level waveform reconstruction. As shown in Figure 2, SPATIALEDIT contains an FOA encoder, an instruction encoder, and a flow-matching latent editor equipped with Spatial Edit Mixture-of-Experts (SE-MoE). Following recent FOA generation models (Liu et al., 2025), we fine-tune a four-channel FOA VAE to map spatial audio into a continuous latent space. The encoder maps source and target FOA waveforms into latent representations:

$$z_{\text{src}} = \mathcal{E}(a_{\text{src}}), \quad z_{\text{tgt}} = \mathcal{E}(a_{\text{tgt}}), \tag{2}$$

where $z_{\text{src}}, z_{\text{tgt}} \in \mathbb{R}^{L \times d}$, L is the latent length, and $d = 128$ is the latent dimension. The instruction c is encoded by a pretrained language encoder and injected into the editor through cross-attention.

We use a Diffusion Transformer backbone (Peebles and Xie, 2023) because spatial editing requires coordination across events, temporal regions, and FOA channels. Since z_{src} and the noisy target latent have the same temporal length, we concatenate them along the feature dimension before the Transformer input projection. The self-attention layers can then learn frame-level correspondences between the source scene and the latent being generated, while cross-attention supplies the instruction condition. The editor is trained with conditional flow matching (Lipman et al., 2023), which matches the one-stage source-to-target supervision by learning a vector field from noise to the target edited latent. Let $z_1 = z_{\text{tgt}}$ be the target latent and $z_0 \sim \mathcal{N}(0, I)$ be a Gaussian noise latent. For a sampled timestep $t \sim \mathcal{U}(0, 1)$, we define

$$z_t = (1 - t)z_0 + tz_1, \quad v = z_1 - z_0. \tag{3}$$

We optimize the following flow-matching objective to regress the target velocity:

$$\mathcal{L}_{\text{FM}} = \mathbb{E}_{z_0,t}\left[\|v - v_\theta(z_t, t, z_{\text{src}}, c)\|_2^2\right]. \tag{4}$$

The source latent gives the vector field an explicit preservation condition, while the instruction specifies the intended edit.

3.4 Spatial Edit MoE with dual-level routing

A single shared feed-forward module must cover very different editing behaviors: inserting an event, relocating a source, changing a trajectory, or modifying room acoustics. This is especially brittle for compound instructions, where several edit types interact. We introduce Spatial Edit Mixture-of-Experts (SE-MoE), a MoE module (Shazeer et al., 2017) in the flow-matching editor, to separate task-level routing from local edit routing. As illustrated in Figure 2, SE-MoE replaces the feed-forward block in every Transformer layer.

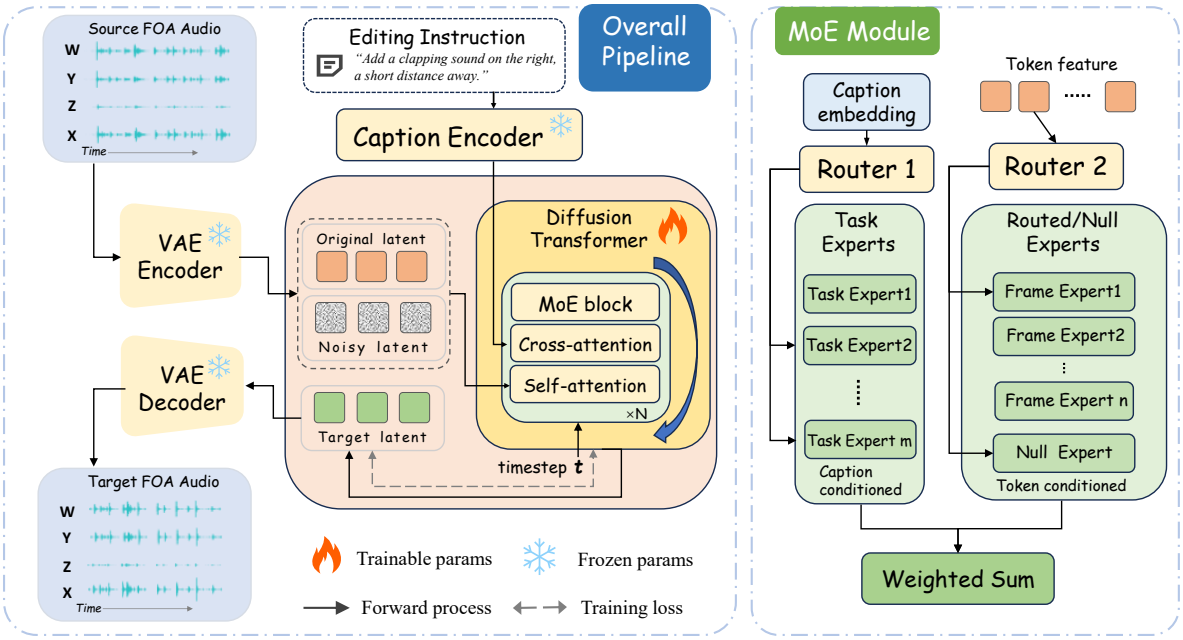

Figure 2: Overview of SpatialEdit. The left panel shows the overall training and inference pipeline, and the right panel shows the SE-MoE module with instruction-level task routing and token-level routed/null experts.

SE-MoE contains three types of experts. Task experts are selected from the instruction representation and remain fixed for all latent frames in the same sample, giving task-aware expert combinations for different edit families and compound instructions. Routed experts are selected independently for each latent token, allowing local adaptation to edited regions and sound events. Null experts return zero tensors, allowing unchanged regions to avoid unnecessary transformations.

Given a latent hidden state x_i at frame token i, the frame-level router computes $p_i = \mathrm{softmax}(W_g(x_i + W_t e_t))$, where e_t is the flow-matching timestep embedding. Rather than using a fixed top-k gate, we use top-p routing and select the smallest expert set $\mathcal{S}_i$ whose cumulative probability exceeds a threshold ρ. Simple or unchanged frames can therefore activate fewer experts, while complex edited regions can use more experts.

For task-level routing, the instruction states are pooled with learned task queries to obtain instruction-conditioned expert probabilities $q(c)$. We apply the same top-p selection to obtain the task expert set $\mathcal{T}(c)$ and normalize the selected probabilities as β_e. The final SE-MoE output combines frame-level routed experts and instruction-level task experts:

$$\text{SE-MoE}(x_i) = \sum_{e \in \mathcal{S}_i \cap \mathcal{E}_r} \alpha_{i,e} f_e^r(x_i) + \sum_{e \in \mathcal{T}(c)} \beta_e f_e^t(x_i) \tag{5}$$

where $\mathcal{E}_r$ is the routed expert set, f_e^r and f_e^t denote routed and task experts, respectively, and $\alpha_{i,e}$ is the normalized frame-level routing weight. Null experts may be selected by the frame-level router, but they contribute zero output and are omitted from the summation.

We use a lightweight routing regularizer, denoted as $\mathcal{L}_{\text{moe}}$, to prevent degenerate expert usage. Details are provided in the Appendix B.1.

3.5 Training strategy

Training directly on complex editing triplets asks the model to learn spatial audio generation, language grounding, and edit execution at the same time. A failure on a compound instruction may therefore come from poor scene understanding rather than from the edit operation itself. We use a staged curriculum to reduce this ambiguity. First, text-to-FOA pretraining uses the scene-level FOA-caption pairs described above, giving the model basic spatial audio generation ability and grounding for audio events, spatial information, dynamic changes, and environmental information.

After pretraining, we initialize the editing model from the text-to-FOA model and fine-tune it on edit-

ing triplets $(a_{\text{src}}, c, a_{\text{tgt}})$. We first train on single-operation edits from the four editing axes so that the model learns atomic editing abilities. We then add multi-operation editing data for joint training. A compound edit may change several factors at the same time, such as replacing a source while moving it farther away or changing the room size while preserving all sound events. We supervise the model with the final target scene rather than a sequence of intermediate edits. This one-stage formulation avoids error accumulation and encourages the model to learn the joint acoustic effect of compound spatial edits.

3.6 Spatial Preference Optimization

Supervised flow matching specifies the target, but it does not explicitly contrast the target with plausible editing failures. In practice, the model can still copy the source audio, modify the wrong event, place an event in the wrong direction, or produce an implausible spatial transformation. We address these failures with Spatial Preference Optimization (SPO), a Direct Preference Optimization (DPO)-based objective (Rafailov et al., 2023; Majumder et al., 2024; Wang et al., 2025), built from edit-specific negative targets.

For each edit type, we design negative samples that reflect typical model errors. For example, for extraction, the negative sample is generated by either extracting the wrong event or performing no operation. These two cases respectively discourage event misunderstanding and simple source copying. Similar negative designs are used for other edit types and are detailed in the appendix. In SPO data construction, wrong-edit negatives and no-operation negatives are generated at a ratio of $2:1$. For each edit type, we construct 10K SPO preference examples, with an average duration of approximately 9.17 seconds.

Let z^+ and z^- denote the positive and negative target latents. For the policy model θ, we compute the per-sample flow-matching losses

$$\ell_\theta^+ = \ell_\theta(z^+, z_{\text{src}}, c), \quad \ell_\theta^- = \ell_\theta(z^-, z_{\text{src}}, c). \quad (6)$$

A frozen reference model θ_{ref} gives the corresponding losses ℓ_{ref}^+ and ℓ_{ref}^-. Since lower flow loss indicates a higher preference, we define

$$\Delta = \beta\left[(\ell_\theta^- - \ell_\theta^+) - (\ell_{\text{ref}}^- - \ell_{\text{ref}}^+)\right], \quad (7)$$

and optimize $\mathcal{L}_{\text{SPO}} = -\log\sigma(\Delta)$.

We keep the positive supervised loss:

$$\mathcal{L}_{\text{align}} = \mathcal{L}_{\text{SPO}} + \lambda_{\text{pos}}\ell_\theta^+ + \lambda_{\text{moe}}\mathcal{L}_{\text{moe}}. \quad (8)$$

During SPO training, the positive and negative samples share the same noise and timestep, so the preference comparison focuses on the target edit rather than sampling randomness.

3.7 Inference strategy

At inference time, the source audio and instruction need different guidance strengths. The source condition should preserve unrelated content and spatial context, while the instruction condition should enforce the requested change. A single CFG scale on the joint condition cannot control these two effects separately. We therefore use a staircase classifier-free guidance (CFG) strategy (Ho and Salimans, 2022). With condition dropout, each solver step can evaluate three velocity predictions:

$$\begin{aligned} v_\varnothing &= v_\theta(z_t, t, \varnothing, \varnothing), \\ v_{\text{src}} &= v_\theta(z_t, t, z_{\text{src}}, \varnothing), \\ v_{\text{all}} &= v_\theta(z_t, t, z_{\text{src}}, c). \end{aligned} \quad (9)$$

Here $v_\varnothing$ is the unconditional prediction, v_{src} uses only the input audio, and v_{all} uses both the input audio and instruction. The guided velocity is

$$\tilde{v}_\theta = v_\varnothing + s_{\text{src}}(v_{\text{src}} - v_\varnothing) + s_{\text{inst}}(v_{\text{all}} - v_{\text{src}}), \quad (10)$$

where s_{src} controls source preservation and s_{inst} controls edit strength. The staircase path $\varnothing \rightarrow z_{\text{src}} \rightarrow (z_{\text{src}}, c)$ first anchors generation to the source scene and then applies the instruction as a residual edit. Compared with a flat two-scale CFG formulation used in multi-condition editing (Brooks et al., 2023), this separation makes the guidance easier to tune and reduces conflicts between preservation and instruction following. After integrating the guided flow from $t = 0$ to $t = 1$, the final edited FOA waveform is decoded as $\hat{a}_{\text{tgt}} = \mathcal{D}(\hat{z}_{\text{tgt}})$.

4 Experiments

4.1 Experimental Setup

Metrics We evaluate audio editing quality using both semantic and spatial metrics. Following prior audio generation and editing works (Lan et al., 2025; Wang et al., 2023), we report FD, FAD, KL, and LSD. FD and FAD measure distributional distance in pretrained audio embedding

spaces, KL measures event-level class-probability mismatch, and LSD measures spectral distortion between generated and reference audio. We further report CLAP score (Elizalde et al., 2023) to evaluate instruction following, computed as the cosine similarity between the editing instruction and the generated audio in the CLAP embedding space.

For spatial fidelity, we evaluate all methods in a common two-channel stereo space to ensure a fair comparison with stereo spatial audio editing baselines. Since our model directly generates FOA audio, we render both our FOA outputs and references into stereo audio using the same fixed FOA-to-stereo decoder before metric computation. Baseline methods are evaluated using their stereo outputs under the same stereo metric pipeline. Following the stereo spatial metrics used in SmartDJ (Lan et al., 2025), we calculate GCC MSE (GCC) based on Generalized Cross-Correlation with Phase Transform (GCC-PHAT), and use StereoCRW (Chen et al., 2022b) to produce stereo audio features to evaluate CRW MSE (CRW) and Fréchet Stereo Audio Distance (FSAD).

Baselines Since no prior method directly edits FOA audio, we decode each FOA result into a stereo proxy when comparing with stereo baselines. The FOA signals already contain the PyRoom-simulated room response, and no additional RIR filtering is applied. We compare with AudioEditor (Jia et al., 2024), ZETA (Manor and Michaeli, 2024), SDEdit (Meng et al., 2022), and SmartDJ (Lan et al., 2025). For AudioEditor, we replace its original Auffusion backend (Xue et al., 2024) with the BEWO/SpatialSonic spatial-audio generator (Sun et al., 2025) to support binaural outputs. For SDEdit and ZETA, we follow SmartDJ and use Stable Audio Open (Evans et al., 2025) as the generative backbone. For SmartDJ, we use its original sequential setting and feed the decomposed operations to the editor one by one.

Implementation Details We first convert the audio editing data into FOA-caption pairs to train a base text-to-FOA model. The base model is trained for 300K steps on 975K samples, enabling it to generate description-following FOA. We then initialize the editing model from the base model and train it on editing data for 500K steps, followed by 2 epochs of SPO post-training. During training, the source audio and text condition are independently replaced with empty inputs with a probability of 10%, enabling the unconditional, source-only, and fully conditioned branches used by staircase CFG.

4.2 Main results

Objective evaluation Table 1 reports the average objective results over all edit types and multi-task editing. Our method achieves the best performance across efficiency, semantic quality, reconstruction quality, and spatial fidelity. In terms of inference speed, our model edits a sample in 1.17s on average, faster than all baselines. This is mainly because SpatialEdit performs one-stage latent editing, while several baselines require many iterative diffusion sampling or sequential editing operations. Compared with the strongest baseline results, our model is better in FD, FAD, KL and LSD. These gains indicate higher-quality edited audio, better event consistency, and lower spectral distortion. Our method also obtains the highest CLAP score, suggesting stronger alignment with editing instructions.

For spatial fidelity, our method consistently outperforms all baselines on GCC, CRW, and FSAD. In particular, the lower GCC error shows that the directional structure encoded in FOA channels is better preserved after editing. The improvement on CRW further indicates more accurate stereo spatial cues after FOA-to-stereo decoding, while the best FSAD score suggests that our method produces spatial representations closer to the target distribution. These results show the advantage of directly modeling FOA audio instead of relying only on stereo-domain editing.

Table 2 further evaluates motion-related editing operations. Compared with SmartDJ, our method achieves substantial gains on both angle motion and distance motion. For angle motion, GCC is reduced from 23.25 to 1.15 and CRW from 50.16 to 14.84. For distance motion, GCC is reduced from 16.45 to 1.74 and CRW from 28.66 to 11.32. These results show that our model is especially effective for dynamic spatial edits, where accurate temporal changes in direction and distance are required.

Human evaluations We further conduct pairwise human evaluations with three annotators. For each test case, annotators are given the source audio, the editing instruction, and two edited results, and are asked to choose the better one in terms of audio quality and instruction-audio alignment. The HP column in Table 1 reports the preference ratio between each baseline and our method. Our method is preferred over ZETA, AudioEditor, SDEdit, and

Table 1: Objective evaluation results averaged over edit types. HP denotes the pairwise human preference split between each baseline and our method in terms of audio quality and instruction-audio alignment.

Method	Speed	FD↓	FAD↓	KL↓	LSD↓	CLAP↑	GCC↓	CRW↓	FSAD↓	HP
ZETA	56.26s	14.84	4.03	4.43	3.29	0.17	51.85	22.12	0.78	17.88%-82.12%
AudioEditor	119.97s	8.66	3.19	2.46	2.89	0.19	28.19	44.50	0.55	25.67%-74.33%
SDEdit	32.28s	8.60	2.03	2.41	2.72	0.18	24.19	25.48	0.40	23.30%-76.70%
SmartDJ	4.23s	8.47	2.87	2.69	2.56	0.17	20.32	33.07	0.32	38.44%-61.56%
Ours	**1.17s**	**5.47**	**1.16**	**1.67**	**1.07**	**0.21**	**12.39**	**18.80**	**0.31**	-

Table 2: Objective comparison on motion-related editing operations. Dis Motion means Distance Motion, indicating the distance of a sound event gradually changes.

Operation	Method	FD↓	FAD↓	KL↓	LSD↓	GCC↓	CRW↓	FSAD↓
Angle Motion	SmartDJ	6.70	2.40	1.92	2.36	23.25	50.16	0.29
	Ours	3.41	1.23	0.66	0.80	1.15	14.84	0.23
Dis Motion	SmartDJ	11.10	1.88	3.18	2.77	16.45	28.66	0.33
	Ours	5.54	0.93	1.25	0.86	1.74	11.32	0.22

Table 3: Ablation study of different model components.

Model	Variation	FD↓	FAD↓	KL↓	LSD↓	CLAP↑
Pretraining	w/o pretrain	8.49	3.15	2.78	2.33	0.16
MoE block	w/o MoE	6.84	2.72	2.34	1.66	0.21
	w/o task expert	6.21	2.14	1.98	1.24	0.18
SPO post-training	w/o SPO	5.94	1.82	1.85	1.10	0.21

SmartDJ. The human evaluation results are consistent with the objective metrics and further confirm the perceptual quality and instruction-following ability of our method.

4.3 Ablations

Ablation of pretraining As shown in Table 3, removing FOA-caption pretraining and training directly on the editing dataset leads to clear degradation across all metrics. FD increases from 5.47 to 8.49, KL increases from 1.67 to 2.78, and CLAP decreases from 0.21 to 0.16. This indicates that text-to-FOA pretraining provides a strong initialization for FOA generation and improves both semantic alignment and reconstruction quality before editing-specific training.

Ablation of SE-MoE with dual-level routing We further analyze the effectiveness of the proposed SE-MoE module. Removing the entire MoE block leads to degradation on FD, KL, and LSD, showing that a single shared feed-forward module is less effective for modeling heterogeneous spatial editing operations. This confirms the benefit of expert specialization for different edit patterns and temporal regions. We then remove only the task experts while keeping the frame-level routed and null experts. This variant performs better than the model without MoE, indicating that token-level routing is useful for local temporal adaptation. However, it is worse than the full model. This suggests that caption-conditioned task experts provide complementary instruction-level specialization. Overall, the ablation verifies that both frame-level routed experts and task-level experts are important for effective FOA spatial audio editing.

Ablation of SPO post-training Finally, we evaluate the effect of the proposed SPO post-training stage by removing it from the full training pipeline, and it is consistently worse than the full model in FD, FAD, KL, and LSD. This indicates that supervised training alone does not fully suppress plausible editing failures, such as source copying, wrong-event modification, or incorrect spatial placement. After SPO post-training, the model better distinguishes the desired target edit from edit-specific negative targets. The improvement is especially reflected in distributional and event-level metrics. These results show that SPO provides complementary preference-level supervision beyond the positive flow-matching objective and improves perceptual editing quality and instruction following.

5 Conclusion

We present SPATIALEDIT, a one-stage framework for instruction-guided 3D FOA spatial audio editing. Using paired FOA editing supervision built with controllable room simulation, SPATIALEDIT supports diverse edits over audio events, spatial cues, temporal dynamics, and environmental properties. The model combines latent flow-matching editing, SE-MoE, and SPO to handle heterogeneous instructions and improve preference alignment. Experiments show consistent improvements over general audio editors and spatial audio baselines across objective and human preference metrics. These results highlight the promise of directly modeling FOA spatial audio for controllable 3D audio creation.

Limitations

Although SPATIALEDIT shows strong performance on instruction-guided FOA spatial audio editing, several limitations remain. First, our paired supervision is constructed with controllable room simulation. This provides accurate source-target alignment for training, but may not fully capture the acoustic complexity of real recordings, such as microphone characteristics, background noise, occlusion, and highly irregular room responses. Extending the framework to real recorded FOA data is an important direction for future work.

Second, while our dataset covers diverse editing operations over audio events, spatial information, dynamic changes, and environmental information, the current scenes are still limited in duration and complexity. Most examples contain a small number of sound events and are approximately 10 seconds long. Editing longer scenes with denser event mixtures, overlapping trajectories, and more open-ended instructions may require stronger long-context modeling and more diverse supervision.

Ethical considerations

Our training data is constructed from open-source speech and sound-effect corpora, together with controllable room simulation. We use these data for modeling spatial editing operations rather than for imitating specific speaker identities. Nevertheless, speech data may contain human voices, and future extensions to real recorded FOA data should carefully consider consent, privacy, and dataset licensing. The simulated-data construction also reduces the need to collect private real-world spatial recordings.

References

James Broderick, Jim Duggan, and Sam Redfern. 2018. The importance of spatial audio in modern games and virtual environments. In *2018 IEEE games, entertainment, media conference (GEM)*, pages 1–9. IEEE.

Tim Brooks, Aleksander Holynski, and Alexei A. Efros. 2023. Instructpix2pix: Learning to follow image editing instructions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 18392–18402.

Changan Chen, Unnat Jain, Carl Schissler, Sebastia Vicenc Amengual Gari, Ziad Al-Halah, Vamsi Krishna Ithapu, Philip Robinson, and Kristen Grauman. 2020. Soundspaces: Audio-visual navigation in 3d environments. pages 17–36.

Changan Chen, Carl Schissler, Sanchit Garg, Philip Kobernik, Alexander Clegg, Paul Calamia, Dhruv Batra, Philip Robinson, and Kristen Grauman. 2022a. Soundspaces 2.0: A simulation platform for visual-acoustic learning. *Advances in Neural Information Processing Systems*, 35:8896–8911.

Ziyang Chen, David F Fouhey, and Andrew Owens. 2022b. Sound localization by self-supervised time delay estimation. In *European Conference on Computer Vision*, pages 489–508. Springer.

Xize Cheng, Ruofan Hu, Xiaoda Yang, Jingyu Lu, Dongjie Fu, Shengpeng Ji, Rongjie Huang, Boyang Zhang, Tao Jin, Zhou Zhao, and 1 others. 2025. Voxdialogue: Can spoken dialogue systems understand information beyond words? In *International Conference on Learning Representations*, volume 2025, pages 288–303.

Gheorghe Comanici, Eric Bieber, Mike Schaekermann, Ice Pasupat, Noveen Sachdeva, Inderjit Dhillon, Marcel Blistein, Ori Ram, Dan Zhang, Evan Rosen, and 1 others. 2025. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities. *arXiv preprint arXiv:2507.06261*.

Benjamin Elizalde, Soham Deshmukh, Mahmoud Al Ismail, and Huaming Wang. 2023. Clap learning audio concepts from natural language supervision. In *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5. IEEE.

Zach Evans, Julian D Parker, CJ Carr, Zack Zukowski, Josiah Taylor, and Jordi Pons. 2025. Stable audio open. In *ICASSP 2025-2025 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5. IEEE.

Eduardo Fonseca, Manoj Plakal, Frederic Font, Daniel P. W. Ellis, and Xavier Serra. 2019. Audio tagging with noisy labels and minimal supervision. In *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2019 Workshop (DCASE2019)*.

Ruohan Gao and Kristen Grauman. 2019. 2.5 d visual sound. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 324–333.

Jonathan Ho and Tim Salimans. 2022. Classifier-free diffusion guidance. *Preprint*, arXiv:2207.12598.

Jinbo Hu, Yin Cao, Ming Wu, Zhenbo Luo, and Jun Yang. 2025a. Salm: Spatial audio language model with structured embeddings for understanding and editing. *Preprint*, arXiv:2507.16724.

Ruofan Hu, Yan Xia, Minjie Hong, Jieming Zhu, Bo Chen, Xiaoda Yang, Minghui Fang, and Tao Jin. 2025b. Vela: Scalable embeddings with voice large language models for multimodal retrieval. *arXiv preprint arXiv:2506.14445*.

Rongjie Huang, Jiawei Huang, Dongchao Yang, Yi Ren, Luping Liu, Mingze Li, Zhenhui Ye, Jinglin Liu, Xiang Yin, and Zhou Zhao. 2023. Make-an-audio: Text-to-audio generation with prompt-enhanced diffusion models. In *International Conference on Machine Learning*, pages 13916–13932. PMLR.

Yuhang Jia, Yang Chen, Jinghua Zhao, Shiwan Zhao, Wenjia Zeng, Yong Chen, and Yong Qin. 2024. Audioeditor: A training-free diffusion-based audio editing framework. *Preprint*, arXiv:2409.12466.

Chris Dongjoo Kim, Byeongchang Kim, Hyunmin Lee, and Gunhee Kim. 2019. Audiocaps: Generating captions for audios in the wild. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 119–132.

Jaeyeon Kim, Heeseung Yun, and Gunhee Kim. 2025. Visage: Video-to-spatial audio generation. *arXiv preprint arXiv:2506.12199*.

Felix Kreuk, Gabriel Synnaeve, Adam Polyak, Uriel Singer, Alexandre Défossez, Jade Copet, Devi Parikh, Yaniv Taigman, and Yossi Adi. 2022. Audiogen: Textually guided audio generation. *arXiv preprint arXiv:2209.15352*.

Zitong Lan, Yiduo Hao, and Mingmin Zhao. 2025. Guiding audio editing with audio language model. *Preprint*, arXiv:2509.21625.

Yichong Leng, Zehua Chen, Junliang Guo, Haohe Liu, Jiawei Chen, Xu Tan, Danilo Mandic, Lei He, Xiangyang Li, Tao Qin, and 1 others. 2022. Binauralgrad: A two-stage conditional diffusion probabilistic model for binaural audio synthesis. *Advances in Neural Information Systems*, 35:23689–23700.

Zhaojian Li, Bin Zhao, and Yuan Yuan. 2024. Tas: Personalized text-guided audio spatialization. In *Proceedings of the 32nd ACM International Conference on Multimedia*, pages 9029–9037.

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matthew Le. 2023. Flow matching for generative modeling. In *International Conference on Learning Representations*.

Haohe Liu, Zehua Chen, Yi Yuan, Xinhao Mei, Xubo Liu, Danilo Mandic, Wenwu Wang, and Mark D Plumbley. 2023. Audioldm: Text-to-audio generation with latent diffusion models. *arXiv preprint arXiv:2301.12503*.

Haohe Liu, Yi Yuan, Xubo Liu, Xinhao Mei, Qiuqiang Kong, Qiao Tian, Yuping Wang, Wenwu Wang, Yuxuan Wang, and Mark D Plumbley. 2024. Audioldm 2: Learning holistic audio generation with self-supervised pretraining. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 32:2871–2883.

Huadai Liu, Tianyi Luo, Kaicheng Luo, Qikai Jiang, Peiwen Sun, Jialei Wang, Rongjie Huang, Qian Chen, Wen Wang, Xiangtai Li, and 1 others. 2025. Omniaudio: Generating spatial audio from 360-degree video. *arXiv preprint arXiv:2504.14906*.

Navonil Majumder, Chia-Yu Hung, Deepanway Ghosal, Wei-Ning Hsu, Rada Mihalcea, and Soujanya Poria. 2024. Tango 2: Aligning diffusion-based text-to-audio generations through direct preference optimization. In *Proceedings of the 32nd ACM International Conference on Multimedia*, pages 564–572.

Hila Manor and Tomer Michaeli. 2024. Zero-shot unsupervised and text-based audio editing using DDPM inversion. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 34603–34629. PMLR.

Chenlin Meng, Yutong He, Yang Song, Jiaming Song, Jiajun Wu, Jun-Yan Zhu, and Stefano Ermon. 2022. Sdedit: Guided image synthesis and editing with stochastic differential equations. In *International Conference on Learning Representations*.

Tianrui Pan, Jie Liu, Zewen Huang, Jie Tang, and Gangshan Wu. 2025. In-the-wild audio spatialization with flexible text-guided localization. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1989–2001.

Vassil Panayotov, Guoguo Chen, Daniel Povey, and Sanjeev Khudanpur. 2015. Librispeech: An asr corpus based on public domain audio books. In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5206–5210. IEEE.

William Peebles and Saining Xie. 2023. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 4195–4205.

Sandra Poeschl, Konstantin Wall, and Nicola Doering. 2013. Integration of spatial sound in immersive virtual environments an experimental study on effects of spatial sound on presence. In *2013 IEEE virtual reality (VR)*, pages 129–130. IEEE.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Stefano Ermon, Christopher D. Manning, and Chelsea Finn. 2023. Direct preference optimization: Your language model is secretly a reward model. In *Advances in Neural Information Processing Systems*, volume 36, pages 53728–53741.

Alexander Richard, Dejan Markovic, Israel D Gebru, Steven Krenn, Gladstone Alexander Butler, Fernando Torre, and Yaser Sheikh. 2021. Neural synthesis of binaural speech from mono audio. In *International Conference on Learning Representations*.

Miguel Sarabia, Elena Menyaylenko, Alessandro Toso, Skyler Seto, Zakaria Aldeneh, Shadi Pirhosseinloo,

Luca Zappella, Barry-John Theobald, Nicholas Apostoloff, and Jonathan Sheaffer. 2023. Spatial librispeech: An augmented dataset for spatial audio learning. In *Proceedings of INTERSPEECH 2023*.

Manolis Savva, Abhishek Kadian, Oleksandr Maksymets, Yili Zhao, Erik Wijmans, Bhavana Jain, Julian Straub, Jia Liu, Vladlen Koltun, Jitendra Malik, and 1 others. 2019. Habitat: A platform for embodied ai research. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 9339–9347.

Robin Scheibler, Eric Bezzam, and Ivan Dokmanić. 2018. Pyroomacoustics: A python package for audio room simulation and array processing algorithms. In *2018 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pages 351–355. IEEE.

Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. 2017. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. *Preprint*, arXiv:1701.06538.

Kazuki Shimada, Archontis Politis, Parthasaarathy Sudarsanam, Daniel A Krause, Kengo Uchida, Sharath Adavanne, Aapo Hakala, Yuichiro Koyama, Naoya Takahashi, Shusuke Takahashi, and 1 others. 2023. Starss23: An audio-visual dataset of spatial recordings of real scenes with spatiotemporal annotations of sound events. *Advances in neural information systems*, 36:72931–72957.

Peiwen Sun, Sitong Cheng, Xiangtai Li, Zhen Ye, Huadai Liu, Honggang Zhang, Wei Xue, and Yike Guo. 2025. Both ears wide open: Towards language-driven spatial audio generation. In *International Conference on Learning Representations*.

Yuancheng Wang, Zeqian Ju, Xu Tan, Lei He, Zhizheng Wu, Jiang Bian, and Sheng Zhao. 2023. Audit: Audio editing by following instructions with latent diffusion models. *Preprint*, arXiv:2304.00830.

Zehan Wang, Ke Lei, Chen Zhu, Jiawei Huang, Sashuai Zhou, Luping Liu, Xize Cheng, Shengpeng Ji, Zhenhui Ye, Tao Jin, and 1 others. 2025. T2a-feedback: Improving basic capabilities of text-to-audio generation via fine-grained ai feedback. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 23535–23547.

Zeyu Xie, Xuenan Xu, Zhizheng Wu, and Mengyue Wu. 2024. Picoaudio: Enabling precise timestamp and frequency controllability of audio events in text-to-audio generation. *Preprint*, arXiv:2407.02869.

Jinlong Xue, Yayue Deng, Yingming Gao, and Ya Li. 2024. Auffusion: Leveraging the power of diffusion and large language models for text-to-audio generation. *arXiv preprint arXiv:2401.01044*.

Wen Zhang, Parasanga N Samarasinghe, Hanchi Chen, and Thushara D Abhayapala. 2017. Surround by sound: A review of spatial audio recording and reproduction. *Applied Sciences*, 7(5):532.

A Dataset Details

A.1 Detailed implementation of spatial audio

We construct an editing training dataset from open-source speech and sound-effect corpora, including AudioCaps (Kim et al., 2019), FSDKaggle2019 (Fonseca et al., 2019), PicoAudio (Xie et al., 2024), LibriSpeech (Panayotov et al., 2015), and Spatial LibriSpeech (Sarabia et al., 2023). For each example, we sample two to four audio events to form a source scene and synthesize its target scene according to the requested edit.

We employ PyRoomAcoustics (Scheibler et al., 2018) for controllable room simulation of FOA spatial audio. Specifically, the simulation allows us to explicitly control the source direction, source-listener distance, motion trajectory, and room reflection conditions. We do not adopt Habitat(Savva et al., 2019) for rendering, as its complex scene geometry may cause occlusions in direct sound propagation and introduce strong multipath reflections, leading to inaccurate directional cues at the listener position. In contrast, PyRoomAcoustics provides a more direct and controllable simulation framework, and is therefore used for generating our spatial audio data. The default room size in our dataset is $[21, 21, 10]$ meters, corresponding to length, width, and height, respectively.

We render FOA spatial audio using PyRoomAcoustics. For each event, we construct a shoebox room and synthesize a room impulse response from the source position to a microphone placed at the listener(the center of the room). The RIR captures the direct path, early reflections, and late reverberation determined by room geometry, wall absorption, and reflection order. We then encode the center RIR into first-order Ambisonics using analytic directional gains derived from the source-listener unit vector. The dry event signal is convolved with the resulting four-channel FOA RIR and mixed into the scene timeline. Unlike binaural rendering, which synthesizes separate left and right ear responses, our rendering produces an Ambisonic sound field in the W,Y,Z,X channel order, allowing the model to learn editable spatial cues in a FOA representation.

Direction description Directional information is one of the most important distinctions between FOA and conventional mono audio. In our dataset, we define ten spatial directions: right, front, left, back, front-right, front-left, back-left, back-right, above, and below. The first eight correspond to azimuth angles of $[0°, 90°, 180°, 270°, 45°, 135°, 225°, 315°]$, while the above and below directions indicate that the source is located vertically relative to the listener, without being restricted to directly overhead or directly beneath. To introduce variability, after specifying the azimuth and distance, we randomly sample a point within a small spherical region centered at this nominal position as the final source location, while keeping the primary directional category fixed.

A.2 Basic edit actions

In this subsection, we explain the meaning of each single-step edit action, as well as how the corresponding training targets and negative samples for preference learning are constructed.

Add Add a new sound event to the original audio. The corresponding negative samples either add an incorrect event or perform no operation by copying the original audio.

Remove Remove a specified sound event from the original audio while preserving the remaining sound events and spatial information. The corresponding negative samples either remove an incorrect event or perform no operation by copying the original audio.

Replace Replace a specified sound event in the original audio with a new sound event. The corresponding negative samples either replace an incorrect event or perform no operation by copying the original audio.

Extract Extract the specified sound event from the original audio and remove the other sound events. The corresponding negative samples either extract an incorrect event or perform no operation by copying the original audio.

Volume increase/Volume fade Increase or decrease the volume of a specified sound event in the original audio while keeping the other sound events unchanged as much as possible. The corresponding negative samples either operate on an incorrect event or perform no operation.

Angle change Change the angular position of a specified sound event in the original audio. The corresponding negative samples either move the event to an incorrect angle, apply the angle change to an incorrect event, or perform no operation.

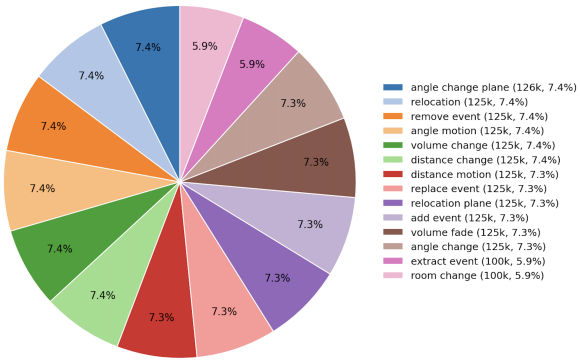

Figure 3: Overview of the training data distribution. Most edit types contain about 125K examples.

Distance change Change the distance of a specified sound event in the original audio. The corresponding negative samples either apply the distance change to an incorrect event or perform no operation.

Room change Change the room size or reverberation condition of the original audio while keeping the sound event content and spatial layout unchanged as much as possible. We define four room-size categories by length, width, and height: small room [2, 2, 2], medium room [6, 6, 4], large room [21, 21, 10], and extra-large room [100, 100, 100], where the extra-large room is regarded as an open-space scene. The corresponding negative samples either perform no room change or change the scene to an incorrect room condition.

Angle motion Move a specified sound event dynamically from an initial direction to a target direction in the original audio. The corresponding negative samples either place the event directly at the target position without motion, move it toward an incorrect direction, or perform no operation.

Distance motion Move a specified sound event dynamically from an initial distance to a target distance in the original audio. The corresponding negative samples either keep the event static, move it to an incorrect position or with an incorrect distance trajectory, or perform no operation.

The data distribution of different single-operation edit actions is shown in Figure 3.

A.3 Prompt for edit instructions

We use Gemini 2.5 Pro to generate the instructions required for editing. The prompt used is as follows:

Prompt for Editing Instruction Generation

```
You are a precise spatial audio editor. Given
a specific edit operation, write a short,
natural English spoken instruction.
STATIC edits: ADD, REMOVE, REPLACE, VOLUME
CHANGE, DISTANCE CHANGE, ANGLE CHANGE,
RELOCATION.
DYNAMIC edits: DISTANCE MOTION, ANGLE MOTION,
VOLUME FADE.
FILTER edit: EXTRACT.
ENVIRONMENT edit: ROOM CHANGE.
For DYNAMIC edits, the caption MUST convey
continuous motion/change, not a static state.
For STATIC edits, describe the resulting
state, not motion.
For ROOM CHANGE, describe the change of room
size.
Each caption must be 10-25 words, one sentence.
Output exactly 3 lines, one caption per line,
no numbering, no quotes.

Room: {room}
Scene before editing:
{original}
Edit: {edit_detail}
Write 3 different English instructions for
this exact edit.
```

B Model Details

B.1 SE-MoE routing regularization

This section gives the routing regularizer used for SE-MoE. The main text keeps only the compact notation $\mathcal{L}_{\text{moe}}$ because these terms stabilize expert usage but are not the central modeling contribution.

For the frame-level router, let $u^r_{b,i,e}$ be the routing logit for expert e at latent token i of sample b, and let

$$p^r_{b,i,e} = \frac{\exp(u^r_{b,i,e})}{\sum_{e'} \exp(u^r_{b,i,e'})} \tag{11}$$

be the corresponding routing probability. The selected top-p expert set is denoted by $\mathcal{S}_{b,i}$. For the task-level router, let $u^t_{b,e}$ and $q_{b,e}$ denote the task-router logit and probability for expert e, and let $\mathcal{T}_b$ be the selected task expert set.

We use a load-balancing loss to avoid routing collapse. For the frame-level router, expert importance and expert load are

$$I^r_e = \sum_{b,i} p^r_{b,i,e}, \qquad L^r_e = \sum_{b,i} \mathbb{I}[e \in \mathcal{S}_{b,i}]. \tag{12}$$

For the task-level router, they are

$$I^t_e = \sum_{b} q_{b,e}, \qquad L^t_e = \sum_{b} \mathbb{I}[e \in \mathcal{T}_b]. \tag{13}$$

Given a vector x, define

$$\mathrm{CV}^2(x) = \frac{\mathrm{Var}(x)}{\mathrm{Mean}(x)^2 + \epsilon}. \tag{14}$$

The balancing term is

$$\mathcal{L}_{\text{bal}} = \text{CV}^2(I^r)+\text{CV}^2(L^r)+\text{CV}^2(I^t)+\text{CV}^2(L^t). \tag{15}$$

We also use a router z-loss to keep routing logits numerically stable:

$$\begin{aligned}\mathcal{L}_z =& \frac{1}{|\mathcal{B}|L}\sum_{b,i}\left(\log\sum_e \exp(u^r_{b,i,e})\right)^2 \\ &+ \frac{1}{|\mathcal{B}|}\sum_{b}\left(\log\sum_e \exp(u^t_{b,e})\right)^2,\end{aligned} \tag{16}$$

where $|\mathcal{B}|$ is the batch size and L is the latent length.

For the null expert, we use the synthetic edit metadata to obtain a binary temporal mask $m_{b,i}$, where $m_{b,i} = 1$ indicates that latent token i overlaps an edited region and $m_{b,i} = 0$ indicates an unchanged region. Let $e_\varnothing$ denote the null expert. The null-routing loss is

$$\begin{aligned}\mathcal{L}_{\text{null}} = -\frac{1}{|\mathcal{B}|L}\sum_{b,i}\big[&(1-m_{b,i})\log(p^r_{b,i,e_\varnothing}+\epsilon) \\ &+ m_{b,i}\log(1-p^r_{b,i,e_\varnothing}+\epsilon)\big].\end{aligned} \tag{17}$$

This term encourages unchanged regions to use the null expert while discouraging edited regions from routing to it.

The final regularizer used in training is

$$\mathcal{L}_{\text{moe}} = \lambda_{\text{bal}}\mathcal{L}_{\text{bal}} + \lambda_z\mathcal{L}_z + \lambda_{\text{null}}\mathcal{L}_{\text{null}}. \tag{18}$$

B.2 Negative targets for SPO

SPO uses edit-specific negative targets rather than arbitrary corrupted audio. We construct two broad types of negatives. A *wrong-edit* negative applies a plausible but incorrect edit, such as changing the wrong event, moving a source to the wrong direction, using the wrong motion trajectory, or applying the wrong room-acoustic condition. A *no-operation* negative keeps the source scene unchanged. During SPO data construction, wrong-edit and no-operation negatives are mixed with a ratio of $2:1$, so the model is trained both to avoid semantically wrong edits and to avoid copying the source when an edit is required.