Audio-sync Video Instance Editing with Granularity-Aware Mask Refiner

Supplementary Material

Haojie Zheng[1,2†] Shuchen Weng[2,3†] Jingqi Liu[1,2] Siqi Yang[5]
Boxin Shi[3,4‡] Xinlong Wang[2]

[1]School of Software and Microelectronics, Peking University [2]Beijing Academy of Artificial Intelligence
[3]State Key Lab of Multimedia Info. Processing, School of Computer Science, Peking University
[4]Nat'l Eng. Research Ctr. of Visual Tech., School of Computer Science, Peking University
[5]Institute for Artificial Intelligence, Peking University
{suimu, liujingqi}@stu.pku.edu.cn {scweng, wangxinlong}@baai.ac.cn
{yousiki, shiboxin}@pku.edu.cn

A. Appendix

A.1. Additional Application Scenarios

As illustrated in Sec. 5.3, we additionally present application scenarios in Fig. S1 to demonstrate the versatility of our framework.

Instance insertion. By specifying target regions within a user-provided video, AVI-Edit can generate the corresponding instance and its accompanying audio according to the text description, seamlessly composited with the background regions (Fig. S1 first row, inserting a car with its accompanying audio into an empty field).

Instance removal. When an unwanted instance is specified, AVI-Edit removes the target instance and its accompanying audio from all frames based on the text description, resulting in a clean background (Fig. S1 second row, removing a plane flying over the sea and its accompanying audio).

Audio-sync video generation. In the extreme case where the instance mask is expanded to cover the full video frame, AVI-Edit discards all video content and accompanying audios, becoming an audio-sync generation model according to the text description (Fig. S1 third row, generating an audio-sync video from the provided text description).

Long-duration video editing. To enable AVI-Edit to edit long-duration videos, we incorporate a conditioning strategy during training to unmask k randomly selected frames. During inference, we generate the initial segment from noise. Subsequent segments are then generated using the last k frames of the preceding clip as visual guidance, enabling the iterative editing of arbitrary-length videos (Fig. S1 last row, consistently editing an instance spanning 241 frames).

A.2. Precision Factor Details

As illustrated in Sec. 4.2, as user-provided instance masks are typically inaccurate (*e.g.*, bounding box), we introduce a precision factor p to indicate the mask granularity.

† Equal contribution.
‡ Corresponding author.

Table S1. The IoU scores of different degradation schedules.

Schedule	Linear	Constant	Instant
IoU (%) ↑	71.87	68.01	**76.23**

In real applications, users typically provide the maximal precision factor $p = P$ for bounding boxes, or manually specify an intermediate p value to provide precise control for the target instance region based on the provided mask. To simulate the inaccuracy of user-provided masks, we randomly degrade each precise instance mask to an arbitrary granularity level p during training (Eq. (5)).

We further provide the architecture details of the Granularity-Aware Mask Refiner (GAMR) in Fig. S2, which demonstrates the approach of precision factor injection. Specifically, the precision factor p is linearly encoded and added to the timestep embedding to produce the modulation parameters:

$$f^{\mathrm{p}} = \mathrm{Linear}(p) + \mathrm{Linear}(t), \tag{1}$$

$$(\gamma, \beta, \alpha) = \mathrm{Linear}(f^{p}), \tag{2}$$

where t is the timestep embedding and (γ, β, α) are estimated parameters for modulation. These parameters are then used to modulate the adaptive layer normalization (AdaLN) [7] followed by a self- or cross-attention:

$$h' = \mathrm{Attention}(h \odot (\gamma + 1) + \beta), \tag{3}$$

where h is the mask feature before modulation. The resulting feature is then further modulated via the gating mechanism [7] with a residual skip connection:

$$\hat{h} = h + \alpha \odot h_{\mathrm{out}}. \tag{4}$$

A.3. Iterative Mask Refinement Details

As illustrated in Sec. 4.2, the Granularity-Aware Mask Refiner (GAMR) is designed to iteratively refine the instance mask throughout the ODE solving process, guided by the

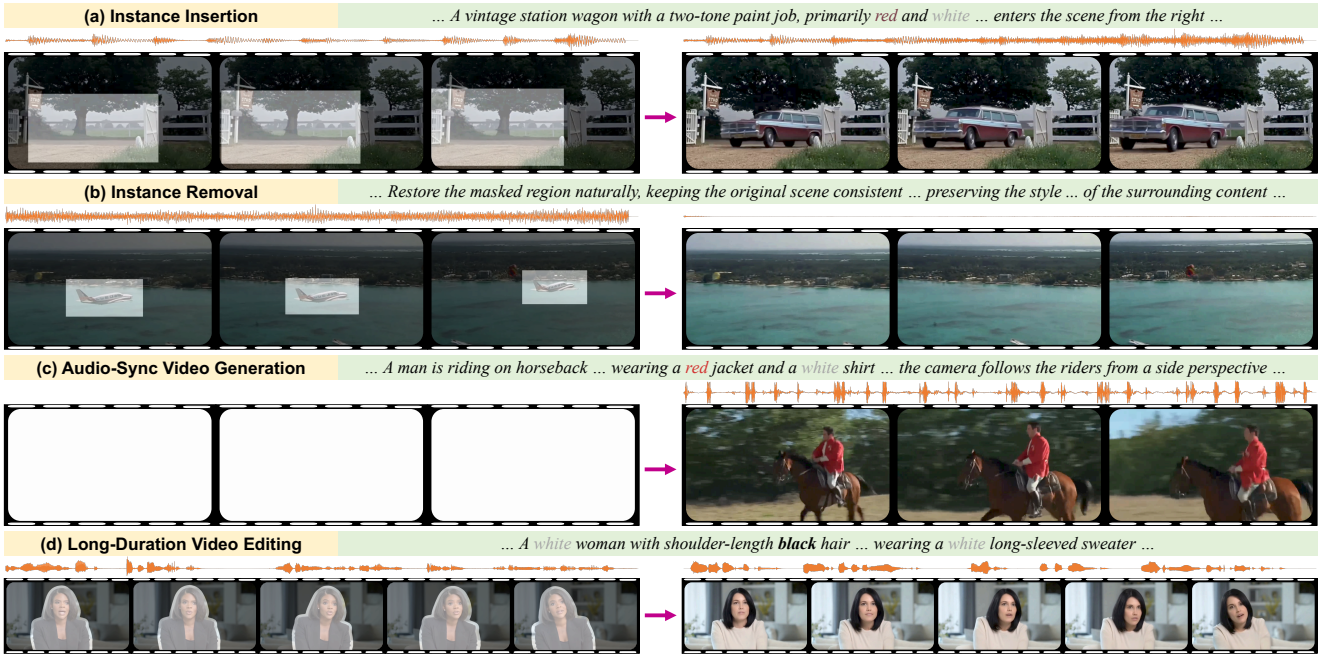

Figure S1. Additional application scenarios of AVI-Edit.

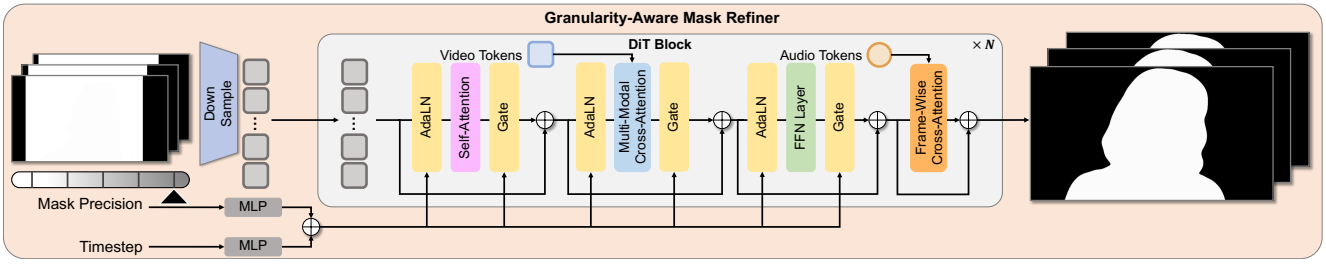

Figure S2. The detailed architecture of the granularity-aware mask refiner.

precision factor p and a predefined degradation schedule. We investigate three distinct schedules: *(i) Linear degradation.* The precision factor p decreases linearly from its initial value down to 0 throughout the inference steps. *(ii) Constant degradation.* The precision factor p remains fixed at its initial value throughout the inference process. *(iii) Instant degradation.* The precision factor p is set to its initial value only for the first step (*i.e.*, $k = 0$). For all subsequent steps (*i.e.*, $k \geq 1$), the factor is instantly set to $p = 1$, allowing only subtle mask refinement.

As shown in Fig. S3, the linear and constant schedules are generally suboptimal, because the mask is largely optimized during the initial step ($k = 0$). If p decays too slowly, the AMR is guided by an overly-coarse mask granularity. Consequently, the model struggles in predicting the target boundary, causing unnecessary shrinkage in the refined mask and leading to unnatural visual content (*e.g.*, the woman's face is rendered in profile to fit the overly small region). Therefore, the best schedule strategy is the simple yet effective instant degradation, where the first step quickly converges to the target region, while subsequent steps precisely refine the mask.

To validate this conclusion, we randomly sample 40 video clips from the test split of AVISET and calculate the Intersection over Union (IoU) [2] of the predicted masks. As presented in Tab. S1, the instant degradation strategy achieves the best score, demonstrating that its predicted masks best match the ground truth.

A.4. Self-Feedback Audio Agent Details

As introduced in Sec. 4.3, we design a dedicated audio agent to curate the accompanying audio. Since human speech separation is a mature technique that typically requires no text prompts, our prompt refinement loop primarily targets the more challenging non-speech separation and audio generation tasks. For clarity, we present a representative execution example in Fig. S4, illustrating the separate-generate-remix-rework pipeline, including plan initialization, separation, generation, remixing, and quality verification.

Judging MLLM. We adopt a judging MLLM to perform

Figure S3. Visual comparison of the mask refinement process under different degradation schedules. **Top:** Original videos with coarse masks. **Rows 2-4:** Editing results with evolving refined masks.

quality verification and evaluate the mixed audio across five dimensions: *(i) Separation accuracy* assesses whether the corresponding audio components are correctly retained from the original audio; *(ii) Generation accuracy* assesses whether the newly generated audio component matches the text description; *(iii) Acoustic harmony* assesses whether the remixed audio is acoustically harmonious; *(iv) Instruction adherence* assesses whether the remixed audio adheres to the text instruction and editing goal; and *(v) Audio fidelity* assesses whether the remixed audio sounds realistic. We prompt the MLLM to score each dimension on a 3-point scale (*i.e.*, 0: Unacceptable, 1: Borderline acceptable, 2: Perfectly acceptable). Only remixed audios with a total quality score exceeding a predefined threshold $\tau = 7$ are accepted. Otherwise, the audio is rejected, and the MLLM generates an improvement instruction to rework with an iterative refinement loop.

Remix. We implement remix as an additive operator without optimization that composites the separation and generation audio components, as a *prerequisite* for final results.

Efficiency. We quantitatively assess the operational efficiency of our self-feedback audio agent on a subset of 200 randomly sampled video clips from the test set. On average, the agent requires only 1.67 rework iterations per clip to achieve satisfactory results, with a total processing time averaging 69.9 seconds. Specifically, the initial editing plan generation requires 27.3 seconds, leaving 42.6 seconds dedicated to the iterative self-feedback loop.

A.5. Inference Costs

We evaluate the inference time and peak VRAM of our framework and the comparison methods, averaged over 100 clips using a single NVIDIA A100 GPU. As shown in Tab. S2, AVI-Edit operates over 3× faster than both AvED [5] and VACE-Foley [4, 10], while requiring significantly less peak VRAM than Ovi [6].

Table S2. The inference time and peak VRAM of comparison with state-of-the-art methods.

Method	AvED	Ovi	VACE-Foley	AVI-Edit
Time (s) ↓	1140.5	**232.6**	1003.6	311.3
VRAM (GB) ↓	37.6	59.6	**33.2**	36.2

A.6. Evaluation Metrics Details

As described in Sec. 5.1, we adopt seven metrics to quantitatively evaluate performance and present the results in Tab. 1 of the main paper. For clarification, we detail these metrics as follows:

- **FVD** assesses the video quality, calculating the Frechét video distance [11] between real and generated video features, extracted by the Inception Network.
- **IS** assesses the fidelity and diversity of generated frames, calculating the Inception score [9] using the softmax outputs of the Inception classifier.
- **FC** assesses the temporal consistency across video frames, calculating the cosine similarity between the embeddings of consecutive video frames, extracted by the CLIP visual encoder [8].
- **TC** assesses the text-video consistency, calculating the cosine similarity between the generated video and text embeddings extracted by the VideoCLIP-XL [12].
- **AC** assesses the audio-video consistency, calculating the cosine similarity between the generated audio and video embeddings extracted by the ImageBind [3].
- **Sync-C/D** are metrics that assess the audio-lip synchronization. They first calculate the Euclidean distance between audio embeddings and mouth-region video embeddings, extracted by the SyncNet [1]. Next, Sync-C is defined as the difference between the minimum and the median of these Euclidean distances, and Sync-D is defined as the minimum of all computed Euclidean distances.

A.7. Organization of Supplementary Video

We provide a supplementary video to dynamically showcase our audio-sync instance editing results. The video is

structured as follows: *(i)* **Representative application scenarios:** We demonstrate four representative editing scenarios to highlight the diverse and compelling applications of our framework (Fig. 1). *(ii)* **Additional application scenarios:** We showcase four extended application scenarios to further demonstrate the versatility of our framework (Fig. S1). *(iii)* **Optional control contexts:** We visualize the integration of optional control contexts (*i.e.*, scribble, pose, and reference image) to facilitate fine-grained editing (Fig. 5). *(iv)* **Comparison and ablation study:** Finally, we provide audio-sync instance editing comparisons against state-of-the-art methods (Fig. 3), followed by visual ablations to verify the effectiveness of each component (Fig. 4).

References

[1] Joon Son Chung and Andrew Zisserman. Out of time: automated lip sync in the wild. In *ACCV Workshops*, 2017. 3

[2] Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (voc) challenge. *IJCV*, 2010. 2

[3] Rohit Girdhar, Alaaeldin El-Nouby, Zhuang Liu, Mannat Singh, Kalyan Vasudev Alwala, Armand Joulin, and Ishan Misra. ImageBind: One embedding space to bind them all. In *CVPR*, 2023. 3

[4] Zeyinzi Jiang, Zhen Han, Chaojie Mao, Jingfeng Zhang, Yulin Pan, and Yu Liu. VACE: All-in-one video creation and editing. *arXiv:2503.07598*, 2025. 3

[5] Yan-Bo Lin, Kevin Lin, Zhengyuan Yang, Linjie Li, Jianfeng Wang, Chung-Ching Lin, Xiaofei Wang, Gedas Bertasius, and Lijuan Wang. Zero-shot audio-visual editing via cross-modal delta denoising. *arXiv:2503.20782*, 2025. 3

[6] Chetwin Low, Weimin Wang, and Calder Katyal. Ovi: Twin backbone cross-modal fusion for audio-video generation. *arXiv:2510.01284*, 2025. 3

[7] William Peebles and Saining Xie. Scalable diffusion models with transformers. In *ICCV*, 2023. 1

[8] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *ICML*, 2021. 3

[9] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training gans. In *NIPS*, 2016. 3

[10] Sizhe Shan, Qiulin Li, Yutao Cui, Miles Yang, Yuehai Wang, Qun Yang, Jin Zhou, and Zhao Zhong. HunyuanVideo-Foley: Multimodal diffusion with representation alignment for high-fidelity foley audio generation. *arXiv:2508.16930*, 2025. 3

[11] Thomas Unterthiner, Sjoerd Van Steenkiste, Karol Kurach, Raphael Marinier, Marcin Michalski, and Sylvain Gelly. Towards accurate generative models of video: A new metric & challenges. *arXiv:1812.01717*, 2018. 3

[12] Jiapeng Wang, Chengyu Wang, Kunzhe Huang, Jun Huang, and Lianwen Jin. VideoCLIP-XL: Advancing long description understanding for video clip models. *arXiv:2410.00741*, 2024. 3

Model Instruction

Vison Language Model

You are an expert visual captioner. You will be given a video, an instance mask, a summarized audio caption, and a user-provided text description. Your task is to edit the audio caption by removing the masked object's sound or to generate a new audio description that satisfies the user's intent.

(a) Audio Separation

First, analyze the provided mask, video, the original audio caption and the user-provided description to determine the removed instance and whether the removed instance is a human who is speaking or not.
Next, imagine the audio after removing that instance and generate a caption within 10 words, which should reflect the remaining audio after the instance is removed.
Output only two lines:
Line 1 — the remaining audio caption for separation.
Line 2 — "Human-speech" or "Non-speech", representing the sound category of the removed audio components.

(b) Audio Generation

First, analyze the provided mask, video, and the user-provided description to determine the new target instance and classify its intended sound category.
Next, based on the video scene and the user's description, generate a scene-appropriate audio caption suitable for audio generation.
Output only two lines:
Line 1 — the new audio caption for generation.
Line 2 — "Speech", "Music", or "Effect", representing the sound category of the generated audio components.

Judging Multi-Modal Language Model

You are an expert multimodal evaluator for an automatic audio editing system. Your goal is to evaluate the quality of the remixed audio and its adherence to a user's edit instruction. You should score the audio across five criteria and provide improved generative prompts if needed.
You will be given the following 7 inputs: 1. Input Video (v): The video to be edited; 2. Accompanying Audio (a): The accompanying audio with the video; 3. Input Mask (m): A video identifying the target instance to be edited; 4. Remixed Audio (a'): The final, edited audio; 5. Text Description (y): The high-level text instruction describing the desired edits; 6. Separation Prompt: The text prompt used by the system to identify and separate non-target audio components. 7. Generation Prompt: The text prompt used by the system to generate the new target audio components.
You are an extremely harsh and critical evaluator whose primary job is to find flaws. You must rigorously compare the Original Audio (a) and Remixed Audio (a'). Then, evaluate a' based on the inputs (v, m, y) by assigning an integer score (0=Unacceptable, 1=Borderline Acceptable, 2=Perfectly Acceptable) to each of the five criteria. Calculate the Total Score (max 10); if the Total Score is less than 8, you should provide improved versions of the Separation Prompt and Generation Prompt; otherwise, output the original prompts as the edit is successful.
The five evaluation criteria are: (i) Separation accuracy assesses whether the corresponding audio components are correctly retained from the original audio. (ii) Generation accuracy assesses whether the newly generated audio component matches the text description. (iii) Acoustic harmony assesses whether the remix audio is acoustically harmonious. (iv) Instruction adherence assesses whether the remixed audio adheres to the text instruction and editing goal. (v) Audio fidelity assesses whether the remixed audio sounds realistic.

Overall Process

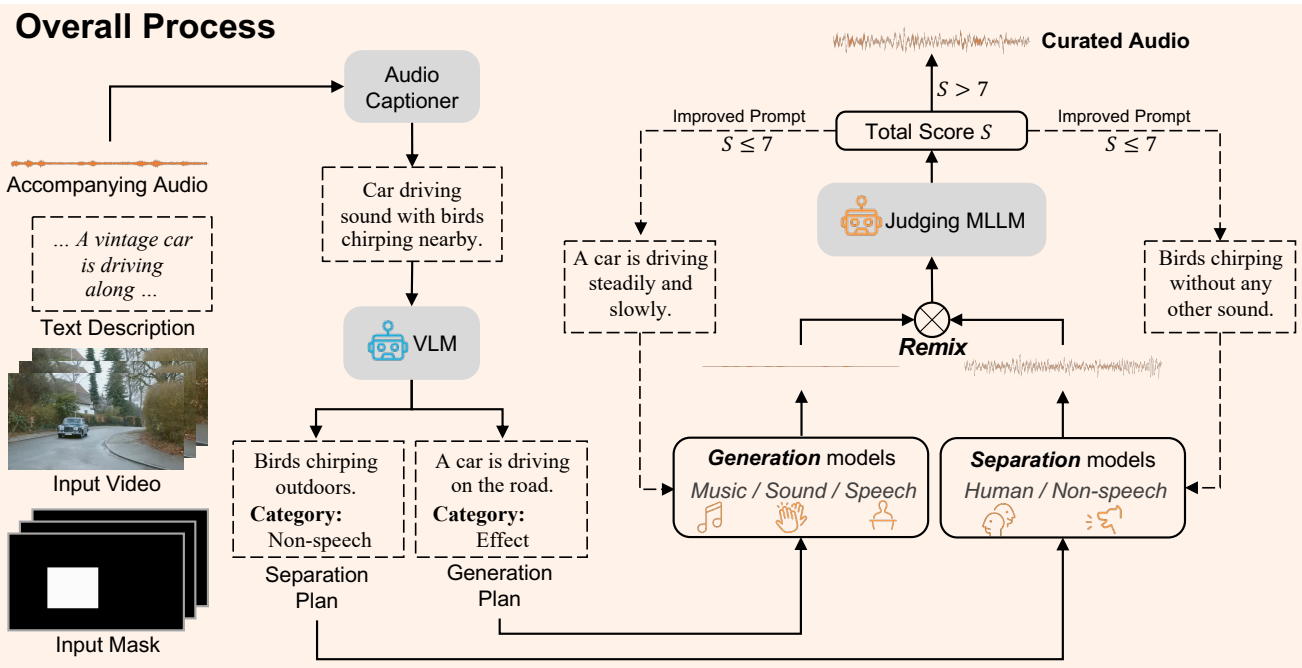

Figure S4. A representative example of the self-feedback audio agent.