SoundVCM: Efficient Video-to-Audio Generation with Velocity Consistency Models

Tianze Luo
Tsinghua University
ltz22@mails.tsinghua.edu.cn

Xingchen Miao
Tsinghua University
miuxc22@mails.tsinghua.edu.cn

Yang Zhang
MIT-IBM Watson AI Lab
yang.zhang2@ibm.com

Lie Lu
Dolby Laboratories
llu@dolby.com

Chuang Gan
University of Massachusetts at Amherst
chuangg@cs.umass.edu

Abstract

High-fidelity video-to-audio synthesis is often computationally expensive, as state-of-the-art diffusion models require numerous iterative steps that impede real-time applications. To overcome this limitation, we introduce velocity consistency models built upon the Flow Matching framework, designed explicitly for few-step inference. Our core contribution lies in a novel training objective that learns a ground-truth average velocity field. This is achieved by supervising the model with a target that linearly interpolates its own velocity predictions at lower noise levels with the instantaneous velocity field given by the Flow Matching framework. We train our models from scratch, eliminating the need for a diffusion pre-training process. Crucially, we apply Classifier-Free Guidance (CFG) to our models during training, which substantially improves audio quality for few-step generation. Experimental results demonstrate that our model significantly reduces inference steps while achieving competitive generation fidelity compared with diffusion or Flow Matching models, bridging the critical gap between quality and efficiency for real-time video-to-audio synthesis. The codes and audio samples are available at https://luotianze666.github.io/files/SoundVCM.zip.

1. Introduction

Video-to-audio generation aims to synthesize semantically aligned and temporally synchronized audios from silent video clips. The inclusion of synchronized audio is crucial for enhancing the immersive experience of video, which has spurred significant research to advance video-to-audio generation performance [4, 38, 42, 45–47]. This work centers on *foley* audio generation, where we train models to synthesize sound effects and ambient audio based on visual events, rather than background music or speech.

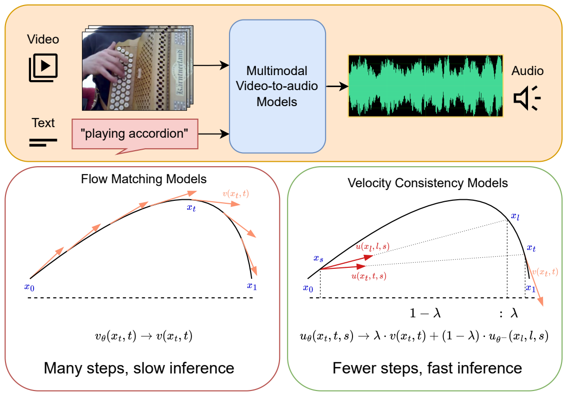
Video
Text
"playing accordion"
Multimodal Video-to-audio Models
Audio
Flow Matching Models
$v(x_t, t)$
x_t
x_0
x_1
$v_\theta(x_t, t) \to v(x_t, t)$
Many steps, slow inference
Velocity Consistency Models
x_l
x_t
x_s
$u(x_l, l, s)$
$u(x_t, t, s)$
$v(x_t, t)$
x_0
x_1
$1-\lambda$
$: \lambda$
$u_\theta(x_t, t, s) \to \lambda \cdot v(x_t, t) + (1-\lambda) \cdot u_{\theta^-}(x_l, l, s)$
Fewer steps, fast inference

Figure 1. As a multimodal video-to-audio model, SoundVCM takes visual input and optional text to predict audio aligned with both modalities. Rather than optimizing instantaneous velocity, our model is supervised with a linear interpolation of both instantaneous and average velocities. See Sec. 3.2 for details.

The advancement of deep generative models has led to various approaches for video-to-audio generation. Early explorations with Generative Adversarial Networks (GANs) [3, 13] aimed to generate synchronized audio, but these methods were often hampered by low perceptual quality and limited practical applicability. Subsequent works employed transformers for autoregressive generation [16, 39, 44], which improved generation quality but were constrained by high latency and still left room for advancement. More recently, latent diffusion models have emerged as the predominant paradigm for high-quality video-to-audio synthesis [4, 29, 38, 42, 46, 47]. The adoption of architectures like the Multimodal Diffusion Transformer (MMDiT) [6] and multimodal joint training has significantly boosted audio quality and temporal synchronization.

Despite their success, latent diffusion models introduce efficiency issues due to the iterative denoising process. This problem is compounded by Classifier-Free Guidance (CFG) [15], which doubles the computational cost at inference. To address this, Frieren [47] employs rectified flow and one-step distillation, while MF-MJT [50] uses the Meanflow [11] training objective for distillation-free one-step generation. However, their few-step generation quality still lags considerably behind that of conventional multi-step diffusion models, creating a notable performance gap.

In this work, we introduce SoundVCM, a novel velocity consistency model for few-step, high-fidelity video-to-audio generation. Our model learns the ground-truth average velocity field from the marginal velocity field within the Flow Matching framework, using a novel training target. This target, a linear interpolation between the marginal velocity and the model's prediction at lower noise, is computed via simple forward propagation. This design advantageously avoids Jacobian-Vector Products (JVPs) computations that are inefficient in frameworks like PyTorch and required by continuous Consistency Models [27] and Meanflow Models [11]. Furthermore, we integrate Classifier-Free Guidance (CFG) directly into the training of SoundVCM. This enhances sample quality while improving inference efficiency by eliminating duplicated runtime computations. On the VGGSound dataset [2], experiments demonstrate that SoundVCM, trained from scratch, generates audio of competitive quality with drastically fewer inference steps than traditional diffusion and Flow Matching models, underscoring our method's efficacy and efficiency.

2. Related Work

2.1. Video-to-Audio Generation

The challenge of generating synchronized audio from video has been approached from several perspectives. Early attempts sought to leverage the capabilities of large pre-trained text-to-audio (TTA) models as a foundation. Representative works in this category include T2AV [32], which leveraged an audio-visual ControlNet [52] to steer a base TTA model with visual information. FoleyCrafter [53] adopted a distinct strategy, feeding both visual and textual embeddings into its UNet generator and using cross-attention mechanisms alongside temporal controllers to direct the synthesis. A more direct and fruitful line of research focused on training dedicated video-to-audio models from scratch. This included foundational work with GANs [3, 13], autoregressive Transformers generating spectrograms [16, 39], and a significant advancement with latent diffusion models like Diff-Foley [29], which improved quality through contrastive pre-training. However, both of these early paradigms were ultimately constrained by the limited scale of available paired audio-video datasets, such as VGGSound [2], highlighting the need for a more robust and data-efficient architectural approach.

These limitations catalyzed the development of multimodal architectures, designed for a more profound and scalable fusion of information. This paradigm shift involves creating unified models that can jointly process different data types, representing the true solution path for high-quality video-to-audio synthesis. Efforts in this direction include developing sophisticated strategies to map features across modalities, such as projecting visual CLIP embeddings into an audio embedding space [45] or using comprehensive aligners like ImageBind and Seeing&Hearing [14, 49]. Further advancements have focused on specific challenges within this multimodal framework, such as enhancing generation efficiency with rectified flow matching [47] or capturing fine-grained motion details for superior temporal alignment [44]. The current state-of-the-art is represented by MMAudio [4], which pioneers a joint training strategy on both video-audio and text-audio data. This approach effectively expands the available training data and improves the model's cross-modal understanding, setting a new benchmark for generation quality. Following this architectural lineage, recent works such as HunyuanVideo-Foley [38] and Kling-Foley [46] have incorporated their own refinements upon this multimodal diffusion framework, also achieving highly competitive results and further validating this research direction.

2.2. Few-Step Generative Models

The pursuit of accelerated generation has made few-step sampling a central focus of modern research. The first approach involves distillation, where large, pre-trained multi-step models are compressed into more efficient, few-step versions [9, 30, 35, 37, 51, 55]. The second strategy centers on training fast samplers from the ground up without relying on a teacher model, a path notably exemplified by the family of consistency training methods (CMs, iCT, ECT, sCT) [10, 27, 40, 41].

Consistency Models (CMs) function by directly mapping corrupted inputs back to their original, clean state, enabling generation in one or very few steps [41]. Initially prominent in distillation frameworks [10, 28], CMs were later found trainable from scratch using consistency training techniques [40]. Subsequent improvements built upon this foundation: iCT streamlined the training process with more robust objective functions and teacher-free learning [40], while ECT bridged CMs with diffusion models through a continuous-time ordinary differential equation (ODE) perspective [10]. More recently, the sCT framework has enhanced the stability of continuous-time training, allowing for consistency models at the billion-parameter scale [27].

Recent research has advanced to handle flexible transitions between arbitrary points in time. Shortcut models, for example, use shared weights conditioned on noise levels and step sizes to accommodate both few- and multi-step sampling [7]. MeanFlow contributes an analysis based on interval-averaged velocities to clarify the relationship between averaged and instantaneous changes [11]. To mitigate variance and bypass multi-stage training, IMM matches moments across transitions within a single objective [54]. Stability and generalization have been improved by Flow-Anchored CMs, which regularize the learning process with a flow-matching anchor [33]. In the distillation domain, Align Your Flow merges CM and Flow Matching (FM) into continuous-time flow maps that perform well regardless of step count, further refined by autoguidance and adversarial tuning [34]. Finally, Transition Models offer a new perspective by reformulating generation based on exact finite-interval dynamics, creating a unified framework where quality consistently improves with an increased step budget [48].

3. Method

3.1. Flow Matching

We firstly outline the fundamentals of Flow Matching. The objective of Flow Matching is to learn a velocity field that maps a simple prior distribution to a complex data distribution. We designate the data distribution as $p(x_0)$ and the prior, a standard Gaussian distribution $\mathcal{N}(0, I)$, as $p(x_1)$.

A linear interpolation between a data sample x_0 and a noise sample x_1 defines the trajectory: $x_t = tx_1+(1-t)x_0$, where $x_t \in \mathbb{R}^n$ and $t \in [0, 1]$. Associated with this path is the marginal velocity field, formulated as the conditional expectation of the vector field pointing from x_0 to x_1:

$$v(x_t, t) = \mathbb{E}_{p(x_0,x_1|x_t)}[x_1 - x_0]. \tag{1}$$

The Flow Matching framework allows for the generation of new samples by first drawing a sample x_1 from the prior distribution $p(x_1)$. Subsequently, one solves the initial value problem (IVP) for the following velocity ordinary differential equation (ODE) from $t = 1$ down to $t = 0$:

$$\frac{\mathrm{d}X(t)}{\mathrm{d}t} = v(X(t), t), \quad X(1) = x_1. \tag{2}$$

To build a generative model, we can approximate the marginal velocity field with a neural network v_θ, parameterized by θ. The network is trained by minimizing the mean squared error between the predicted and true velocity fields:

$$\mathcal{L}_{\text{FM}}(\theta) = \mathbb{E}_{t,x_t} \left\| v_\theta(x_t, t) - v(x_t, t)\right\|^2 . \tag{3}$$

However, training with this objective is not feasible because the ground-truth velocity field $v(x, t)$ is a conditional expectation. To address this, the Conditional Flow Matching (CFM) framework gives a tractable loss function [24, 25]:

$$\mathcal{L}_{\text{CFM}}(\theta) = \mathbb{E}_{t,x_0,x_1,x_t} \left\| v_\theta(x_t, t) - (x_1 - x_0)\right\|^2 . \tag{4}$$

Optimizing $\mathcal{L}_{\text{CFM}}$ is equivalent to optimizing $\mathcal{L}_{\text{FM}}$. This loss is practical as it only requires sampling pairs of data and noise to define the regression targets for the neural network.

3.2. Velocity Consistency Models

Following previous works [12, 23, 33, 50], we define the average velocity field based on the marginal velocity field in the Flow Matching framework as follows:

$$u(x_t, t, s) = \frac{1}{s-t}\int_t^s v(x_\tau, \tau)\,\mathrm{d}\tau, \tag{5}$$

where $x_t \in \mathbb{R}^n$, $0 \le s < t \le 1$.

When $t = s$, the average velocity field equals the marginal velocity field, i.e., $u(x_t, t, t) = v(x_t, t)$. By approximating the ground-truth average velocity function with a neural network $u_\theta(x_t, t, s)$, we can avoid numerically solving the velocity ODE and achieve few-step or one-step data generation. We now consider how to construct effective training objectives for our model.

By the additive property of integrals, we have:

$$u(x_t, t, s)(s-t) = u(x_t, t, l)(l-t)+u(x_l, l, s)(s-l), \tag{6}$$

where $x_l = x_t + u(x_t, t, l)(l - t)$ and $0 \le s < l < t \le 1$.

Note that when l is sufficiently close to t, the average velocity field approximates the marginal velocity field, i.e., $u(x_t, t, l) = v(x_t, t) + O(l - t)$, assuming the average velocity field is continuously differentiable. We thus obtain the following approximation for $u(x_t, t, s)$:

$$\begin{aligned} u(x_t, t, s) &= \lambda v(x_t, t) + (1 - \lambda)u(x_l, l, s) \\ &+ O((l - t)\lambda), \quad \text{where } \lambda = \frac{l-t}{s-t} \in [0, 1]. \end{aligned} \tag{7}$$

This indicates that the average velocity field can be approximated by a linear combination of the marginal velocity field and a nearby average velocity field, with the approximation error becoming negligible when l is sufficiently close to t. In addition, when l is close to t, x_l can be approximated as:

$$\begin{aligned} x_l &= x_t + u(x_t, t, l)(l - t) \\ &= x_t + v(x_t, t)(l - t) + O((l - t)^2). \end{aligned} \tag{8}$$

Assuming the marginal velocity field is continuously differentiable, we combine Eq. (7) and Eq. (8) to obtain:

$$\begin{aligned} u(x_t, t, s) &= \lambda v(x_t, t) + (1 - \lambda)u(x_t \\ &+v(x_t, t)(l - t), l, s) + O((l - t)[(l - t) + \lambda]). \end{aligned} \tag{9}$$

This property directly enables us to construct a training objective for our model $u_\theta(x_t, t, s)$:

$$L_{\text{VCM}} = \mathbb{E}_{t,l,s,x_t}\Bigg[\frac{w(\text{MSE})}{D} \cdot \big\|u_\theta(x_t, t, s) \\ -[\lambda v(x_t, t) + (1-\lambda)u_{\theta^-}(\hat{x}_l, l, s)]\big\|_2^2\Bigg], \tag{10}$$

where $\hat{x}_l = x_t + v(x_t, t)(l - t)$, D is the data dimension, θ^- denotes the stop-gradient operation, MSE represents the raw mean square error, and $w(\text{MSE})$ is an adaptive loss coefficient to enhance model performance. When training from scratch, the marginal velocity field $v(x_t, t) = \mathbb{E}_{p(x_0,x_1|x_t)}[x_1 - x_0]$ is intractable; we therefore replace it with the random term $x_1 - x_0$ following the Flow Matching framework.

For the adaptive loss coefficient, following previous works [10, 11, 40], we employ:

$$w(\text{MSE}) = \frac{1}{(\text{MSE} + \epsilon)^p} \tag{11}$$

to improve loss robustness, where p controls the robustness level and ϵ is a small smoothing factor. When $p > 0$, this coefficient downscales gradients for data points with larger mean square errors within a batch. Consistent with prior work [10, 12, 23], this adaptive loss coefficient improves few-step generation quality in our model.

Note that the above loss function is defined for $s < t$ since $\lambda = \frac{l-t}{s-t}$. For $t = s$, since the average velocity field equals the marginal velocity field, we set $\lambda = 1$ to reduce our loss function to the flow matching loss. We observe that including a certain proportion of data points with $t = s$ improves performance, as the training target is more stable compared to the $s < t$ case, thereby accelerating model convergence. Besides, as shown in next section, including data points with $t = s$ is also important to apply CFG to our model. For more details related to our loss function, please refer to supplementary materials.

Regarding the sampling of time variables t, l, s during training: for the fixed ratio of data points with $t = s$, we sample t directly from a logit-normal distribution, $\text{sigmoid}(\mathcal{N}(\mu, \sigma))$. For the $s < t$ case, we first draw two samples from the same logit-normal distribution, then assign the smaller to s and the larger to t. Here, s is clamped to ensure $s < t - 10^{-4}$, avoiding excessively close values. Following experiments in Consistency Models [10, 40], we adopt an exponential decay schedule for the interval length in the consistency loss. We also determine λ using an exponential schedule:

$$\lambda = \lambda_{\text{init}} \left(\frac{\lambda_{\text{end}}}{\lambda_{\text{init}}}\right)^{S_{\text{now}}/S_{\text{total}}}, \tag{12}$$

where S_{now} and S_{total} denote the current and total training steps, respectively. This is equivalent to determining l as:

$$l = t + (s - t) \cdot \lambda_{\text{init}} \left(\frac{\lambda_{\text{end}}}{\lambda_{\text{init}}}\right)^{S_{\text{now}}/S_{\text{total}}}. \tag{13}$$

Similar to s, l is clamped to ensure $l < t - 10^{-4}$, since when l is too close to t, the difference between the two average velocities may be too small for effective learning.

3.3. Classifier-Free Guidance

Classifier-Free Guidance (CFG) [15] has become a standard technique for enhancing the fidelity of conditional generation in diffusion models. During training, the conditioning information is randomly discarded with a fixed probability, allowing the model to jointly learn both conditional and unconditional distributions. At inference, the conditional and unconditional predictions are linearly combined to amplify the guidance signal, thereby improving output quality.

A computational drawback of this approach is the doubled cost during inference, as the model must compute both conditional and unconditional predictions. In contrast, we integrate CFG into the training phase of our model, thereby avoiding extra inference-time computation. Specifically, in the Flow Matching framework, the conditional marginal velocity field for conditional generation is defined as:

$$v(x_t, t \mid c) = \mathbb{E}_{p(x_0,x_1|x_t,c)}[x_1 - x_0]. \tag{14}$$

By linearly combining this with the unconditional marginal velocity field, we obtain the guided marginal velocity field:

$$v_{\text{CFG}}(x_t, t, c) = wv(x_t, t \mid c) + (1 - w)v(x_t, t). \tag{15}$$

We want the model to learn the average of the ground-truth guided marginal velocity field. However, direct estimation of the unconditional velocity field is intractable. To address this, the unconditional field is learned by replacing the condition c with an empty condition ϕ with a probability of 0.1, and by setting $t = s$ for a subset of the data during training. For data points assigned ϕ and $s < t$, we compute the loss by feeding ϕ to the network and replacing the velocity term $v(x_t, t)$ in Eq. (10) with $x_1 - x_0$, following unconditional Flow Matching framework.

For data points with a valid condition c, we first predict the unconditional velocity field v_{uncond} using $u_\theta(x_t, t, t, \phi)$. We then compute the guided loss by feeding c to the network and replacing $v(x_t, t)$ in Eq. (10) with $w(x_1 - x_0) + (1 - w)v_{\text{uncond}}$, where w is the CFG scale.

The term $w(x_1-x_0)+(1-w)v_{\text{uncond}}$ can suffer from high variance due to the scaling factor w, which impairs training convergence and final performance. To mitigate this, we leverage the model's ability to directly predict the guided velocity field v_{guided} using $u_\theta(x_t, t, t, c)$, a prediction with

Algorithm 1 SoundVCM Guided Loss Computation

```
1:  Input: Neural network $u_\theta$, audio latent $x$, visual and textual conditions $c$, adaptive loss coefficient $(p, \epsilon)$, Velocity mix ratio $m$, Target combination coefficient $\lambda$
2:  Sample $t, s$ according to Sec. 3.2, $\epsilon \sim \mathcal{N}(0, I)$
3:  $x_t \leftarrow (1-t) \cdot x + t \cdot \epsilon$, $v_t \leftarrow \epsilon - x$
4:  $v_{\text{uncond}} \leftarrow u_{\theta^-}(x_t, t, t, \phi)$, $v_{\text{guided}} \leftarrow u_{\theta^-}(x_t, t, t, c)$
5:  $v_{\text{mix}} = m(wv_t + (1-w)v_{\text{uncond}}) + (1-m)v_{\text{guided}}$
6:  $l = t + (s-t)\lambda$, $\hat{x}_l = x_t + (l-t)v_{\text{mix}}$
7:  $u_{ts} = u_\theta(x_t, t, s, c)$, $u_{ls} = u_{\theta^-}(\hat{x}_l, l, s, c)$
8:  MSE $= \frac{1}{D}\|u_{ts} - (\lambda v_{\text{mix}} + (1-\lambda)u_{ls})\|_2^2$
9:  $w = \text{stop-gradient}\left(\frac{1}{(MSE+\epsilon)^p}\right)$
10: Loss = $w \cdot$ MSE
11: Output: Loss
```

Algorithm 2 SoundVCM Inference

```
1: Input: Neural network $u_\theta$, visual and textual conditions $c$, sampling steps $N$.
2: Sample initial noise: $x \sim \mathcal{N}(0, I)$
3: for $i = N-1$ to $0$ do
4:     $x = x - \frac{1}{N} \cdot u_\theta(x, \frac{i+1}{N}, \frac{i}{N}, c)$
5: end for
6: Output: $x$
```

much lower variance than the random term. We then introduce a mixed velocity

$$v_{\text{mix}} = m\big(w(x_1 - x_0) + (1-w)v_{\text{uncond}}\big) + (1-m)v_{\text{guided}} \tag{16}$$

to replace $v(x_t, t)$ in Eq. (10). Since the variance primarily stems from the stochastic term $x_1 - x_0$, a small velocity mix ratio m ($0 < m \leq 1$) effectively dampens the overall variance by incorporating the stable v_{guided} prediction.

3.4. Algorithm

The preceding sections have detailed our velocity consistency model's principle and its training-time CFG application. To crystallize these concepts, we provide the pseudocode for the guided loss computation and inference process below. For complete PyTorch implementation, please refer to our codes in supplementary materials.

3.5. Model Architecture

Figure 2 presents the overall framework of SoundVCM, a multimodal generative model built upon an enhanced Flux-style transformer. The model predicts the mean velocity on the space of latent variables, guided by video and text conditions as well as timestep embeddings. Here we outline its main components and explain our design choices.

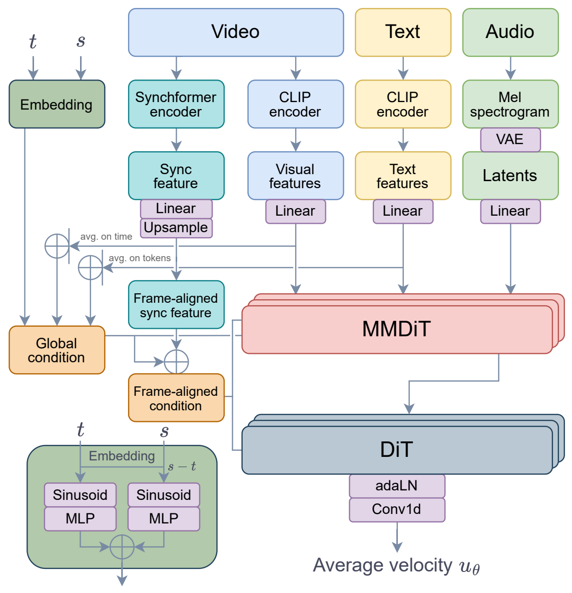

Figure 2. Model architecture, based on MMAudio framework. We modified the time embedding to support two-time conditioning. See the supplementary material for details.

3.5.1. Multimodal Transformer

Our model architecture follows the multimodal framework proposed in MMAudio [4], which unifies text, audio, and video streams within a multimodal transformer backbone. The design aims to capture fine-grained interactions across modalities while maintaining temporal and semantic coherence, crucial for foley generation. All modalities are represented as 1D token sequences, and they are projected to a unified hidden dimension before being processed by the transformer. Built upon the MMDiT structure [6], the model alternates between multimodal attention blocks, where different modalities interact through joint attention, and audio-only DiT layers that refine acoustic representations. The audio latent is then processed through adaptive layer normalization (adaLN) and Conv1d to predict the latent space's average velocity u_θ between two timesteps t and s. Visual and textual representations are pooled and combined with timestep embeddings to construct a global context vector. This vector provides global conditioning through adaLN in each MMDiT block, modulating each block via learned scale and bias transformations, which ensures shared global control across the entire sequence.

During inference, visual and textual modalities are passed through encoded conditions (if available) or a fixed empty vector (if missing), naturally adapting to any visual-textual combination for audio prediction. The audio latent is initialized as noise from a $\mathcal{N}(0, I)$ distribution, rather than

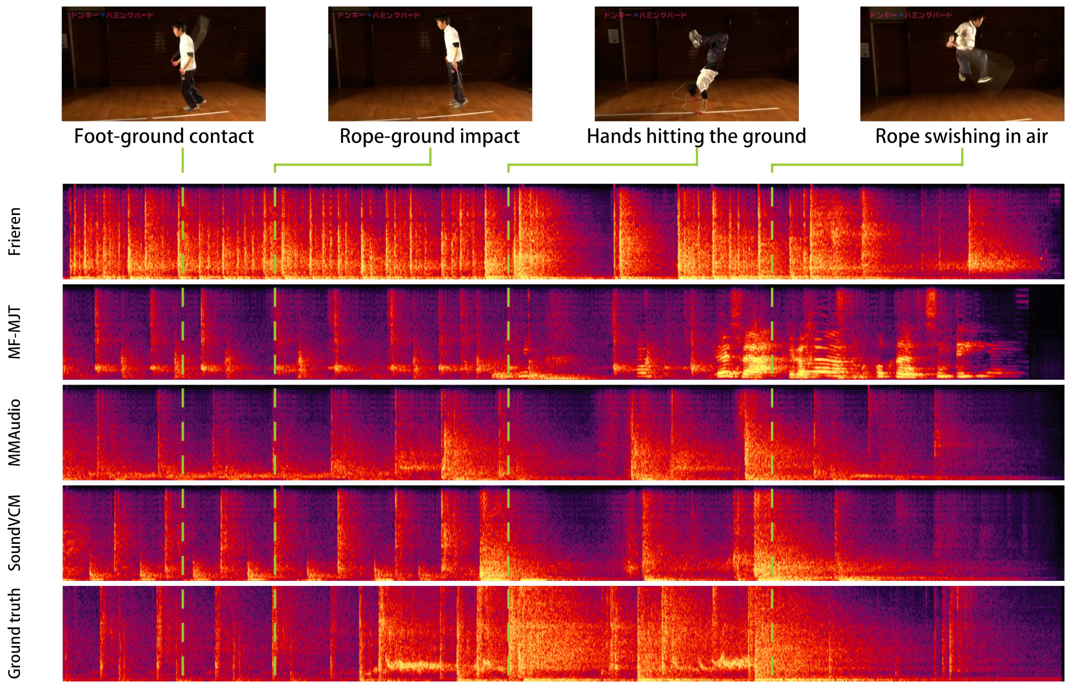

Figure 3. Mel-spectrogram comparison of different models on a VGGSound test clip. Frieren, MF-MJT, and SoundVCM use 1-NFE inference, whereas MMAudio uses 25×2 NFEs. Among all one-step models, ours achieves the best audio-visual alignment and produces semantically coherent audio, remaining competitive with the multi-step MMAudio model.

derived from data through a VAE.

3.5.2. Two-time embedding

In the MMAudio architecture, the conditioning mechanism employs a single-time sinusoidal embedding to encode temporal information. To support average velocity expression. we extend this by introducing a two-time embedding scheme. This method explicitly incorporates both the current time t and the relative offset $(s-t)$, where s is the next time step. This formulation allows the model to capture transitions between successive time points, providing richer temporal cues for smoother interpolation and improved consistency across latent transitions. The embedding technique follows the approach outlined in MeanFlow [12]. This method naturally degenerates to predicting instantaneous velocities when $s=t$, in which case the second embedding deals with a fixed offset 0.

4. Experiments

4.1. Datasets

For training, we use two types of datasets to support our multimodal model: video-text-audio and text-audio. During training, we balance the contribution of both types by adjusting their sampling weights, aiming for a roughly equal amount of exposure to each modality. The two types of datasets are randomly mixed, and for the text-audio samples the video features are set to a fixed empty vector. Before training, we preprocess the features of video, text, and audio to reduce I/O overhead on training. Our dataset setup follows MMAudio [4] to ensure a fair comparison. Below are the datasets used for training and evaluation:

VGGSound [2] The only video-text-audio dataset used, containing ∼200,000 10 s video clips. Each clip is paired with a text label for classification, which we directly use as the text input [4, 18, 26]. We use only the first 8 s of each video for both training and evaluation.

AudioCaps [20] A text-audio dataset containing ∼50,000 10 s audios with manually annotated text descriptions. We use only the first 8 s of each audio. While a 2.0 version with ∼100,000 clips exists, we restrict our use to the ∼50,000 clips from the original version for consistency.

Clotho [5] A text-audio dataset with ∼4,000 audios of varying lengths. We split each audio into several non-overlapping 8 s clips. Each audio has five human-annotated text descriptions, meaning five text-audio pairs per clip.

WavCaps [31] A collection of text-audio datasets weakly labeled using ChatGPT. Data sources include FreeSound, BBC Sound Effects, AudioSet SL, and SoundBible; we only use the first three, following MMAudio.

4.2. Evaluation Metrics

We evaluate our model across four dimensions: distribution matching, audio quality, semantic alignment, and tempo-

Table 1. Video-to-audio evaluation results on VGGSound [2] test set. At NFE column, $\times 2$ means the model adopts Classifier-Free Guidance during inference, so the total function evaluation time is doubled. $\spadesuit$: Evaluated using official code and checkpoints. *Frieren* includes three variants: no reflow, reflow and reflow+distill. Reflow supports few-step inference, while only reflow+distill can be evaluate in 1-NFE. $\heartsuit$: Results reported by MF-MJT [50]. As no public code is available, we list only the metrics provided in the paper. $\clubsuit$: These models does not support text inputs during inference. $\diamondsuit$: Results reported by MMAudio [4]. We include them directly because our evaluation strictly follows the same `av-benchmark` protocol.

Model	NFE $\downarrow$	Params	FD$_{\text{VGG}}$ $\downarrow$	FD$_{\text{PANNs}}$ $\downarrow$	FD$_{\text{PaSST}}$ $\downarrow$	KL$_{\text{PANNs}}$ $\downarrow$	KL$_{\text{PaSST}}$ $\downarrow$	IS$_{\text{PANNs}}$ $\uparrow$	IB $\uparrow$	DeSync $\downarrow$
ReWaS$^{\diamondsuit}$ [18]	200×2	619 M	1.79	17.54	141.38	2.87	2.82	8.51	14.82	1.062
V2A-Mapper$^{\clubsuit\diamondsuit}$ [45]	200×2	229 M	0.84	8.40	84.57	2.69	2.56	12.47	22.58	1.225
Seeing&Hearing$^{\diamondsuit}$ [49]	30×2	415 M	5.40	24.58	219.01	2.26	2.30	8.58	**33.99**	1.204
V-AURA$^{\clubsuit}$ [44]	-	695 M	2.88	14.80	218.50	2.42	2.07	10.08	27.64	0.654
VATT$^{\diamondsuit}$ [1]	16×2	415 M	2.77	10.63	131.88	**1.48**	**1.41**	11.90	25.00	1.195
FoleyCrafter$^{\diamondsuit}$ [53]	25×2	1.22 B	2.51	16.24	140.09	2.30	2.23	15.68	25.68	1.225
Frieren$^{\spadesuit\clubsuit}$ (no reflow) [47]	25×2	159 M	1.34	11.45	106.10	2.73	2.86	12.25	22.78	0.851
MMAudio$^{\diamondsuit}$ [4]	25×2	157 M	0.79	5.22	70.19	1.65	1.59	14.44	29.13	**0.483**
MF-MJT$^{\heartsuit}$ [50]	25	157 M	1.13	5.87	-	1.59	-	**16.55**	28.22	0.57
Frieren$^{\spadesuit\clubsuit}$ (reflow+distill)	1	159 M	1.83	16.48	142.99	2.54	2.64	8.51	21.92	0.841
Frieren$^{\spadesuit\clubsuit}$ (reflow)	4	159 M	2.23	14.40	142.09	2.59	2.67	7.98	22.15	0.833
MF-MJT$^{\heartsuit}$	1	157 M	1.46	11.14	-	1.87	-	9.39	21.78	0.86
SoundVCM (ours)	1	157 M	0.88	5.95	78.90	1.78	1.76	12.54	25.31	0.632
	4		**0.74**	**5.08**	**69.75**	1.66	1.58	13.62	27.32	0.553

ral alignment. These metrics together assess both the perceptual quality of the generated audio and its correspondence to the input video and text. All evaluations follow the `av-benchmark` implementation provided by MMAudio to ensure full comparability with the baseline.

Distribution Matching This dimension measures how closely the generated audio matches the feature distribution of the ground truth. We report two standard distances: Fréchet Distance (FD) and Kullback–Leibler (KL) divergence. FD is computed using embeddings from VGGish [8], PANNs [21], and PaSST [22], while KL divergence is computed using PANNs and PaSST classifiers.

Audio Quality The quality of generated audio is evaluated using the Inception Score (IS) [36] with a PANNs classifier.

Semantic Alignment To evaluate semantic correspondence between video and audio, we use ImageBind [14] to extract their features. The cosine similarity forms the IB-score.

Temporal Alignment Temporal synchrony between audio and video is quantified using the DeSync score predicted by the Synchformer [17] module, which estimates the degree of misalignment between the two modalities.

4.3. Training Settings

The model is optimized with AdamW using a peak learning rate of 1×10^{-4} and a linear warm-up of 1,000 steps. The learning rate is decayed to 1×10^{-5} at 240k steps and further to 1×10^{-6} at 270k steps, with training proceeding for a total of 300k steps under a global batch size of 512. We use $\beta_1 = 0.9$, $\beta_2 = 0.95$, and a weight decay of 1×10^{-6}. Model parameters are additionally stabilized using Post-Hoc EMA [19] with $\sigma_{\text{rel}} = 0.05$. Training is performed in mixed precision (`bf16`) on 4 H100 GPUs and takes around 30 hours of wall-clock time.

In each training iteration, the batch is divided between velocity learning $(t = s)$ and consistency learning $(s < t)$ with a ratio of $r = 0.75$. The logit-normal distribution is set to $\text{sigmoid}(\mathcal{N}(-0.4, 1))$ to sample time points. On-training classifier-free guidance is applied only for $t \in [0, 0.8]$, with a guidance scale of $w = 4$ and a null-conditioning probability of 0.1. The velocity mix ratio is set to $m = 0.75$. We employ an adaptive loss scaling strategy with parameters $p = 1$ and $\epsilon = 0.01$. Finally, we use a simple time-decay schedule with $\lambda_{\text{init}} = 1/10$ and $\lambda_{\text{end}} = 1/500$.

4.4. Video-to-Audio Results

Tab. 1 compares our model's video-to-audio generation results with other few-step and multi-step models on the VGGSound [2] test set, which contains around 15K video-audio pairs. Among few-step models, both MF-MJT [50] and SoundVCM adopts a MMAudio-like network. While MF-MJT combines it with MeanFlow [12] training scheme, SoundVCM performs velocity consistency model training.

Across all evaluation metrics, SoundVCM delivers consistently strong performance. Our model achieves the best Fréchet Distance (FD) among all compared methods, outperforming not only all few-step models but also multi-

Table 2. Ablation studies.

Variant	$\text{FD}_{\text{VGG}} \downarrow$	$\text{FD}_{\text{PANNs}} \downarrow$	$\text{FD}_{\text{PaSST}} \downarrow$	$\text{KL}_{\text{PANNs}} \downarrow$	$\text{KL}_{\text{PaSST}} \downarrow$	$\text{IS}_{\text{PANNs}} \uparrow$	IB $\uparrow$	DeSync $\downarrow$
$w = 1$	1.63	12.40	102.63	2.05	2.01	7.93	21.02	0.807
$w = 2$	0.98	7.93	84.69	1.84	1.79	10.30	23.91	0.687
$w = 3$	**0.84**	6.49	80.13	**1.79**	**1.75**	11.60	25.00	0.641
$w = 4$	0.86	**6.08**	**79.35**	**1.79**	**1.75**	**12.34**	**25.31**	**0.633**
$m = 0.25$	**0.88**	**5.95**	**78.90**	**1.78**	1.76	**12.54**	**25.31**	**0.632**
$m = 0.5$	**0.88**	6.36	82.27	1.81	1.78	11.99	25.00	0.650
$m = 0.75$	0.93	6.46	85.72	1.79	**1.75**	11.59	24.80	**0.632**
$m = 1$	0.96	6.78	89.15	1.82	1.78	10.92	24.37	0.644
$r = 0$	9.51	30.76	279.95	2.82	2.85	3.69	12.20	0.960
$r = 0.25$	0.94	8.38	87.41	1.94	1.91	10.78	22.61	0.720
$r = 0.5$	0.89	6.62	82.93	1.83	1.80	11.96	24.46	0.659
$r = 0.75$	**0.88**	**5.95**	**78.90**	**1.78**	**1.76**	**12.54**	**25.31**	**0.632**
$p = 0$	5.36	20.03	204.34	2.16	2.08	5.48	17.80	0.683
$p = 0.5$	1.00	6.60	86.06	1.79	1.76	10.99	24.93	**0.609**
$p = 1$	**0.88**	**5.95**	**78.90**	**1.78**	**1.76**	**12.54**	**25.31**	0.632
$p = 1.5$	0.93	8.57	90.30	1.89	1.88	10.75	23.33	0.719
$\lambda_{\text{end}} = 1/250$	0.87	6.09	79.36	1.79	1.76	12.34	25.21	0.633
$\lambda_{\text{end}} = 1/500$	0.88	**5.95**	**78.90**	**1.78**	1.76	**12.54**	**25.31**	**0.632**
$\lambda_{\text{end}} = 1/1000$	**0.86**	6.18	80.26	**1.78**	**1.74**	12.36	**25.31**	0.635

step diffusion baselines. Our model also performs the best among all few-step models on KL divergence. In terms of Inception Score (IS), SoundVCM shows a clear advantage within the few-step category. Its performance surpasses several multi-step models. On the IB metric, SoundVCM substantially outperforms all few-step models and most multi-step baselines. Note that Seeing&Hearing [49] explicitly optimizes the IB objective during training, which explains its unusually high IB scores. For temporal synchronization, SoundVCM behaves similarly to MMAudio in leveraging sync features during training. As a result, SoundVCM obtains stronger sync performance than most baselines. Although MF-MJT also incorporates MMAudio's sync features, our approach captures noticeably more temporal structure, leading to better alignment.

4.5. Ablation Study

We conduct extensive ablation studies to analyze the impact of key hyperparameters on the performance of SoundVCM, and the results are summarized in Table 2.

CFG Scale w. Consistent with observations in diffusion models, the Classifier-Free Guidance (CFG) scale w significantly influences the model's output quality. Our experiments reveal that performance progressively improves as the CFG scale is increased, indicating that a stronger guidance signal is beneficial within the tested range.

Velocity Mix Ratio m. The velocity mix ratio m balances deterministic and stochastic elements in the velocity field. Our experimental results indicate that a smaller value of m consistently enhances performance. This can be attributed to the suppression of the stochastic term, which in turn reduces randomness and accelerates model convergence.

$t = s$ Data Ratio r. We investigate the effect of incorporating standard flow matching data points (where $t = s$) during training. The inclusion of this data notably improves both training stability and overall model performance. The optimal results are achieved with a high ratio of $r = 75\%$, highlighting the importance of flow matching loss.

Adaptive Loss Scale p. The choice of the adaptive loss scale p determines the robustness of the loss function. We found that a moderately robust loss yields the best results. The raw mean square error, corresponding to $p = 0$, leads to significantly inferior performance. In contrast, $p = 1$ provides the optimal balance of robustness, leading to the best overall performance.

Value of λ_{end}. The hyperparameter λ_{end} governs the convergence rate of λ towards zero. A larger λ_{end} results in a slower but smoother learning of the average velocity, while a smaller value leads to a faster but more unstable learning process. Our experiments show that a moderate value of $1/500$ achieves relatively better performance.

5. Conclusion

In this paper, we introduced SoundVCM, a novel velocity consistency model designed for efficient, high-fidelity video-to-audio synthesis. By introducing a novel training objective under the Flow Matching framework, our approach learns a ground-truth average velocity field, eliminating the need for costly diffusion pre-training or Jacobian-Vector Product computations. Integrating Classifier-Free Guidance further improves audio quality in few-step inference. Experiments on VGGSound demonstrate that SoundVCM bridges the gap between generation quality and efficiency, achieving competitive audio fidelity with far fewer inference steps—a key step toward real-time video-to-audio applications.

References

[1] Hassan Akbari, Liangzhe Yuan, Rui Qian, Wei-Hong Chuang, Shih-Fu Chang, Yin Cui, and Boqing Gong. Vatt: Transformers for multimodal self-supervised learning from raw video, audio and text. *Advances in neural information processing systems*, 34:24206–24221, 2021. 7

[2] Honglie Chen, Weidi Xie, Andrea Vedaldi, and Andrew Zisserman. Vggsound: A large-scale audio-visual dataset. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 721–725. IEEE, 2020. 2, 6, 7, 3

[3] Peihao Chen, Yang Zhang, Mingkui Tan, Hongdong Xiao, Deng Huang, and Chuang Gan. Generating visually aligned sound from videos. *IEEE Transactions on Image Processing*, 29:8292–8302, 2020. 1, 2

[4] Ho Kei Cheng, Masato Ishii, Akio Hayakawa, Takashi Shibuya, Alexander Schwing, and Yuki Mitsufuji. Mmaudio: Taming multimodal joint training for high-quality video-to-audio synthesis. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 28901–28911, 2025. 1, 2, 5, 6, 7, 3

[5] Konstantinos Drossos, Samuel Lipping, and Tuomas Virtanen. Clotho: An audio captioning dataset. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 736–740. IEEE, 2020. 6

[6] Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorenz, Axel Sauer, Frederic Boesel, et al. Scaling rectified flow transformers for high-resolution image synthesis. In *Forty-first international conference on machine learning*, 2024. 1, 5

[7] Kevin Frans, Danijar Hafner, Sergey Levine, and Pieter Abbeel. One step diffusion via shortcut models. *arXiv preprint arXiv:2410.12557*, 2024. 3

[8] Jort F Gemmeke, Daniel PW Ellis, Dylan Freedman, Aren Jansen, Wade Lawrence, R Channing Moore, Manoj Plakal, and Marvin Ritter. Audio set: An ontology and human-labeled dataset for audio events. In *2017 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pages 776–780. IEEE, 2017. 7

[9] Zhengyang Geng, Ashwini Pokle, and J Zico Kolter. One-step diffusion distillation via deep equilibrium models. *Advances in Neural Information Processing Systems*, 36: 41914–41931, 2023. 2

[10] Zhengyang Geng, Ashwini Pokle, William Luo, Justin Lin, and J Zico Kolter. Consistency models made easy. *arXiv preprint arXiv:2406.14548*, 2024. 2, 4

[11] Zhengyang Geng, Mingyang Deng, Xingjian Bai, J Zico Kolter, and Kaiming He. Mean flows for one-step generative modeling. *arXiv preprint arXiv:2505.13447*, 2025. 2, 3, 4

[12] Zhengyang Geng, Mingyang Deng, Xingjian Bai, J Zico Kolter, and Kaiming He. Mean flows for one-step generative modeling. *arXiv preprint arXiv:2505.13447*, 2025. 3, 4, 6, 7

[13] Sanchita Ghose and John J Prevost. Foleygan: Visually guided generative adversarial network-based synchronous sound generation in silent videos. *IEEE Transactions on Multimedia*, 25:4508–4519, 2022. 1, 2

[14] 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 *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 15180–15190, 2023. 2, 7

[15] Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598*, 2022. 2, 4

[16] Vladimir Iashin and Esa Rahtu. Taming visually guided sound generation. *arXiv preprint arXiv:2110.08791*, 2021. 1, 2

[17] Vladimir Iashin, Weidi Xie, Esa Rahtu, and Andrew Zisserman. Synchformer: Efficient synchronization from sparse cues. In *ICASSP 2024-2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5325–5329. IEEE, 2024. 7

[18] Yujin Jeong, Yunji Kim, Sanghyuk Chun, and Jiyoung Lee. Read, watch and scream! sound generation from text and video. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 17590–17598, 2025. 6, 7

[19] Tero Karras, Miika Aittala, Jaakko Lehtinen, Janne Hellsten, Timo Aila, and Samuli Laine. Analyzing and improving the training dynamics of diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 24174–24184, 2024. 7

[20] Chris Dongjoo Kim, Byeongchang Kim, Hyunmin Lee, and Gunhee Kim. 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, 2019. 6

[21] Qiuqiang Kong, Yin Cao, Turab Iqbal, Yuxuan Wang, Wenwu Wang, and Mark D Plumbley. Panns: Large-scale pretrained audio neural networks for audio pattern recognition. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 28:2880–2894, 2020. 7

[22] Khaled Koutini, Jan Schlüter, Hamid Eghbal-Zadeh, and Gerhard Widmer. Efficient training of audio transformers with patchout. *arXiv preprint arXiv:2110.05069*, 2021. 7

[23] Xiquan Li, Junxi Liu, Yuzhe Liang, Zhikang Niu, Wenxi Chen, and Xie Chen. Meanaudio: Fast and faithful text-to-audio generation with mean flows. *arXiv preprint arXiv:2508.06098*, 2025. 3, 4

[24] Yaron Lipman, Ricky TQ Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. *arXiv preprint arXiv:2210.02747*, 2022. 3

[25] Xingchao Liu, Chengyue Gong, and Qiang Liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. *arXiv preprint arXiv:2209.03003*, 2022. 3

[26] Xiulong Liu, Kun Su, and Eli Shlizerman. Tell what you hear from what you see-video to audio generation through text. *Advances in Neural Information Processing Systems*, 37:101337–101366, 2024. 6

[27] Cheng Lu and Yang Song. Simplifying, stabilizing and scaling continuous-time consistency models. *arXiv preprint arXiv:2410.11081*, 2024. 2

[28] Simian Luo, Yiqin Tan, Longbo Huang, Jian Li, and Hang Zhao. Latent consistency models: Synthesizing high-resolution images with few-step inference. *arXiv preprint arXiv:2310.04378*, 2023. 2

[29] Simian Luo, Chuanhao Yan, Chenxu Hu, and Hang Zhao. Diff-foley: Synchronized video-to-audio synthesis with latent diffusion models. *Advances in Neural Information Processing Systems*, 36:48855–48876, 2023. 1, 2

[30] Weijian Luo, Tianyang Hu, Shifeng Zhang, Jiacheng Sun, Zhenguo Li, and Zhihua Zhang. Diff-instruct: A universal approach for transferring knowledge from pre-trained diffusion models. *Advances in Neural Information Processing Systems*, 36:76525–76546, 2023. 2

[31] Xinhao Mei, Chutong Meng, Haohe Liu, Qiuqiang Kong, Tom Ko, Chengqi Zhao, Mark D Plumbley, Yuexian Zou, and Wenwu Wang. Wavcaps: A chatgpt-assisted weakly-labelled audio captioning dataset for audio-language multimodal research. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 32:3339–3354, 2024. 6

[32] Shentong Mo, Jing Shi, and Yapeng Tian. Text-to-audio generation synchronized with videos. *arXiv preprint arXiv:2403.07938*, 2024. 2

[33] Yansong Peng, Kai Zhu, Yu Liu, Pingyu Wu, Hebei Li, Xiaoyan Sun, and Feng Wu. Flow-anchored consistency models. *arXiv preprint arXiv:2507.03738*, 2025. 3

[34] Amirmojtaba Sabour, Sanja Fidler, and Karsten Kreis. Align your flow: Scaling continuous-time flow map distillation. *arXiv preprint arXiv:2506.14603*, 2025. 3

[35] Tim Salimans and Jonathan Ho. Progressive distillation for fast sampling of diffusion models. *arXiv preprint arXiv:2202.00512*, 2022. 2

[36] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training gans. *Advances in neural information processing systems*, 29, 2016. 7

[37] Axel Sauer, Dominik Lorenz, Andreas Blattmann, and Robin Rombach. Adversarial diffusion distillation. In *European Conference on Computer Vision*, pages 87–103. Springer, 2024. 2

[38] 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 preprint arXiv:2508.16930*, 2025. 1, 2

[39] Roy Sheffer and Yossi Adi. I hear your true colors: Image guided audio generation. In *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5. IEEE, 2023. 1, 2

[40] Yang Song and Prafulla Dhariwal. Improved techniques for training consistency models. *arXiv preprint arXiv:2310.14189*, 2023. 2, 4

[41] Yang Song, Prafulla Dhariwal, Mark Chen, and Ilya Sutskever. Consistency models. In *International Conference on Machine Learning*, 2023. 2

[42] Zeyue Tian, Yizhu Jin, Zhaoyang Liu, Ruibin Yuan, Xu Tan, Qifeng Chen, Wei Xue, and Yike Guo. Audiox: Diffusion transformer for anything-to-audio generation. *arXiv preprint arXiv:2503.10522*, 2025. 1

[43] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017. 2

[44] Ilpo Viertola, Vladimir Iashin, and Esa Rahtu. Temporally aligned audio for video with autoregression. In *ICASSP 2025-2025 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5. IEEE, 2025. 1, 2, 7

[45] Heng Wang, Jianbo Ma, Santiago Pascual, Richard Cartwright, and Weidong Cai. V2a-mapper: A lightweight solution for vision-to-audio generation by connecting foundation models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 15492–15501, 2024. 1, 2, 7

[46] Jun Wang, Xijuan Zeng, Chunyu Qiang, Ruilong Chen, Shiyao Wang, Le Wang, Wangjing Zhou, Pengfei Cai, Jiahui Zhao, Nan Li, et al. Kling-foley: Multimodal diffusion transformer for high-quality video-to-audio generation. *arXiv preprint arXiv:2506.19774*, 2025. 1, 2

[47] Yongqi Wang, Wenxiang Guo, Rongjie Huang, Jiawei Huang, Zehan Wang, Fuming You, Ruiqi Li, and Zhou Zhao. Frieren: Efficient video-to-audio generation network with rectified flow matching. *Advances in Neural Information Processing Systems*, 37:128118–128138, 2024. 1, 2, 7, 3

[48] Zidong Wang, Yiyuan Zhang, Xiaoyu Yue, Xiangyu Yue, Yangguang Li, Wanli Ouyang, and Lei Bai. Transition models: Rethinking the generative learning objective. *arXiv preprint arXiv:2509.04394*, 2025. 3

[49] Yazhou Xing, Yingqing He, Zeyue Tian, Xintao Wang, and Qifeng Chen. Seeing and hearing: Open-domain visual-audio generation with diffusion latent aligners. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7151–7161, 2024. 2, 7, 8

[50] Xiaoran Yang, Jianxuan Yang, Xinyue Guo, Haoyu Wang, Ningning Pan, and Gongping Huang. Meanflow-accelerated multimodal video-to-audio synthesis via one-step generation. *arXiv preprint arXiv:2509.06389*, 2025. 2, 3, 7

[51] Tianwei Yin, Michaël Gharbi, Richard Zhang, Eli Shechtman, Fredo Durand, William T Freeman, and Taesung Park.

One-step diffusion with distribution matching distillation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 6613–6623, 2024. 2

[52] Lvmin Zhang, Anyi Rao, and Maneesh Agrawala. Adding conditional control to text-to-image diffusion models. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 3836–3847, 2023. 2

[53] Yiming Zhang, Yicheng Gu, Yanhong Zeng, Zhening Xing, Yuancheng Wang, Zhizheng Wu, and Kai Chen. Foleycrafter: Bring silent videos to life with lifelike and synchronized sounds. *arXiv preprint arXiv:2407.01494*, 2024. 2, 7

[54] Linqi Zhou, Stefano Ermon, and Jiaming Song. Inductive moment matching. *arXiv preprint arXiv:2503.07565*, 2025. 3

[55] Mingyuan Zhou, Huangjie Zheng, Zhendong Wang, Mingzhang Yin, and Hai Huang. Score identity distillation: Exponentially fast distillation of pretrained diffusion models for one-step generation. In *Forty-first International Conference on Machine Learning*, 2024. 2

SoundVCM: Efficient Video-to-Audio Generation with Velocity Consistency Models

Supplementary Material

A. Learning Objective Analysis

In this section, we provide an analysis demonstrating that minimizing this learning objective enables the neural network $u_\theta(x_t, t, s)$ to correctly approximate the ground truth average velocity field $u(x_t, t, s)$.

Recall that our learning objective is defined as:

$$L_{\text{VCM}} = \mathbb{E}_{t,l,s,x_t}\left[\frac{w(\text{MSE})}{D} \cdot \Big\| u_\theta(x_t, t, s) - [\lambda v(x_t, t) + (1-\lambda) u_{\theta^-}(\hat{x}_l, l, s)] \Big\|_2^2\right], \tag{17}$$

We firstly consider the case where $t = s$. By definition, $u(x_t, t, t) = v(x_t, t)$. Since we sample a certain proportion of data points with $t = s$ during training and set $\lambda = 1$, our model effectively learns the instantaneous velocity field via standard flow matching loss.

Next, we consider the case where $s < t$. Let $l_i = s + (t - s)\frac{i}{N}$ for $i \in \{0, 1, \cdots, N\}$ be $N + 1$ discrete time points such that $l_0 = s$ and $l_N = t$. Letting $\hat{x}_{l_N} = x_t$, we can iteratively define the Euler approximate trajectory:

$$\begin{aligned} \hat{x}_{l_{i-1}} &= \hat{x}_{l_i} + v(x_{l_i}, l_i)(l_{i-1} - l_i) \\ &= \hat{x}_{l_i} + v(x_{l_i}, l_i)\frac{s-t}{N}, \quad i \in \{1, 2, \cdots, N\}. \end{aligned} \tag{18}$$

This represents an approximate ODE trajectory generated by an Euler solver using the ground truth instantaneous velocity field $v(x_t, t)$. In contrast, the points on the ground truth trajectory are governed by the ground truth average velocity field $u(x_t, t, s)$:

$$\begin{aligned} x_{l_{i-1}} &= x_{l_i} + u(x_{l_i}, l_i, l_{i-1})(l_{i-1} - l_i) \\ &= x_{l_i} + u(x_{l_i}, l_i, l_{i-1})\frac{s-t}{N}, \quad i \in \{1, 2, \cdots, N\}. \end{aligned} \tag{19}$$

We assume that the ground truth average velocity function is continuously differentiable with globally bounded derivatives. Using the property $u(x_t, t, t) = v(x_t, t)$ for all $t \in [0, 1]$, we have:

$$\begin{aligned} x_{l_{N-1}} &= x_t + u(x_t, t, l_{N-1})\frac{s-t}{N} \\ &= x_t + v(x_t, t)\frac{s-t}{N} + O\left(\left(\frac{s-t}{N}\right)^2\right) \\ &= \hat{x}_{l_{N-1}} + O\left(\left(\frac{s-t}{N}\right)^2\right). \end{aligned} \tag{20}$$

Proceeding iteratively, we derive:

$$\begin{aligned} x_{l_{N-2}} &= x_{l_{N-1}} + u(x_{l_{N-1}}, l_{N-1}, l_{N-2})\frac{s-t}{N} \\ &= x_t + v(\hat{x}_{l_{N-1}}, l_{N-1})\frac{s-t}{N} + 2O\left(\left(\frac{s-t}{N}\right)^2\right) \\ &= \hat{x}_{l_{N-2}} + 2O\left(\left(\frac{s-t}{N}\right)^2\right). \end{aligned} \tag{21}$$

Continuing this process until $l_0 = s$, we obtain:

$$\begin{aligned} x_s = x_{l_0} &= \hat{x}_{l_0} + N \times O\left(\left(\frac{s-t}{N}\right)^2\right) \\ &= \hat{x}_s + O\left(\frac{(s-t)^2}{N}\right). \end{aligned} \tag{22}$$

This result confirms that, as the Euler solver is a first-order method, the solution $\hat{x}_s$ deviates from the true solution x_s with an error of magnitude $O(\frac{(s-t)^2}{N})$.

Now, assume that after training, the model satisfies an error bound e_{max} for arbitrary $0 \le s < l < t \le 1$ and $x_t \in \mathbb{R}^D$:

$$\left\| u_\theta(x_t, t, s) - \left[\frac{l-t}{s-t} v(x_t, t) + \left(1 - \frac{l-t}{s-t}\right) u_\theta(\hat{x}_l, l, s)\right] \right\|_2 \le e_{max}. \tag{23}$$

Multiplying by $|s - t|$, this inequality implies:

$$\big\| u_\theta(x_t, t, s)(s-t) - [(l-t)v(x_t, t) + (s-l)u_\theta(\hat{x}_l, l, s)] \big\|_2 \le e_{max}(s-t). \tag{24}$$

Combining this with the time points l_i defined previously, for a single step we have:

$$\big\| [u_\theta(x_t, t, s)(s-t) - (l_{N-1} - t)v(x_t, t)] - (s - l_{N-1})u_\theta(\hat{x}_{l_{N-1}}, l_{N-1}, s) \big\|_2 \le e_{max}(s-t). \tag{25}$$

We can iteratively apply Equation 24 to Equation 25 to obtain:

$$\begin{aligned} &\left\| u_\theta(x_t, t, s)(s-t) - \sum_{i=0}^{N-1} (l_i - l_{i+1}) v(\hat{x}_{l_{i+1}}, t) \right\|_2 \\ &\le \sum_{i=0}^{N-1} e_{max}(s-t) = N e_{max}(s-t). \end{aligned} \tag{26}$$

By the definition of $\hat{x}_s = \hat{x}_{l_0}$ and the approximate trajectory construction, we know that:

$$\begin{aligned}\sum_{i=0}^{N-1}(l_i - l_{i+1})v(\hat{x}_{l_{i+1}}, t) &= \hat{x}_{l_0} - \hat{x}_{l_N} = \hat{x}_s - \hat{x}_t \\ &= x_s - x_t + O\left(\frac{(s-t)^2}{N}\right).\end{aligned} \tag{27}$$

Substituting this back into Equation 26, we have:

$$\begin{aligned}&\left\|u_\theta(x_t, t, s)(s-t) - (x_s - x_t)\right\|_2 \\ &\qquad\leq Ne_{max}(s-t) + O\left(\frac{(s-t)^2}{N}\right).\end{aligned} \tag{28}$$

Recalling that $x_s = x_t + (s-t)u(x_t, t, s)$, we divide by $(s-t)$ to obtain the final bound:

$$\left\|u_\theta(x_t, t, s) - u(x_t, t, s)\right\|_2 \leq Ne_{max} + O\left(\frac{s-t}{N}\right). \tag{29}$$

During training, we set $\lambda = \frac{l-t}{s-t}$, which essentially corresponds to a discretization scheme where $N \approx O(\frac{1}{\lambda})$. As $\lambda \to 0$ (implying N becomes large), the remainder term $O(\frac{s-t}{N})$ decreases. Therefore, if the model's training loss (e_{max}) and λ are sufficiently small, the derived bound ensures that our model u_θ accurately approximates the ground truth velocity field $u(x_t, t, s)$. This inequality also suggests that selecting an excessively small λ is suboptimal, as the growth of N may outpace the reduction in e_{max}. This observation aligns with our ablation study, where $\lambda_{\text{end}} = \frac{1}{500}$ yields better performance than both $\lambda_{\text{end}} = \frac{1}{250}$ and $\lambda_{\text{end}} = \frac{1}{1000}$.

B. Network Details

Our model backbone is primarily built upon MMAudio [4]. For completeness and to contextualize the simplified architectures discussed in Section 3.5, we provide the detailed specifications of our model below.

B.1. Model Conditioning

We utilize averaged text and visual features to condition the DiT blocks. Specifically, after linear projection, the text features t_f undergo average pooling across the token dimension, while the visual features v_f are average-pooled across the time dimension. Subsequently, we compute the global condition c_g, which is applied to the MMDiTs, by summing the time embedding and the output of an MLP processing the combined features: $\text{MLP}\left(\text{avg}_{\text{token}}(t_f) + \text{avg}_{\text{time}}(v_f)\right)$. This global condition is further added to the frame-aligned synchronization feature to yield the frame-aligned sync conditioning c_f, which serves as input to both the MMDiT and DiT modules.

B.2. Sinusoidal Frequency

Our implementation of sinusoidal embeddings adheres to the standard positional encoding principles used in Transformers [43], where each timestep is mapped to oscillating components of distinct frequencies:

$$\begin{aligned}\text{PE}(t, 2i) &= \cos(t/10000^{2i/d}), \\ \text{PE}(t, 2i+1) &= \sin(t/10000^{2i/d}).\end{aligned} \tag{30}$$

However, the MMAudio-v2 implementation sets `max_period` $= 1$, effectively collapsing the frequency spectrum. This results in nearly constant embeddings for neighboring timesteps, drastically reducing the model's capacity to represent temporal distances or phase relationships. While such compression may remain tolerable for discrete diffusion steps, it proves detrimental to consistency-based models, which rely on smooth, continuous evolution within the latent space.

In our formulation, we restore a larger period (`max_period` $= 10^4$), thereby reinstating a broad frequency basis and preserving temporal resolution across the embedding space. This modification enables the model to capture both fine-grained local variations and long-term consistency between timesteps. Empirically, we observe that utilizing a large `max_period` stabilizes the velocity field estimation and significantly improves temporal coherence in the generated outputs.

B.3. Feature Projection

Audio, text, and video features are processed by distinct projection modules before serving as inputs to the MMDiT blocks. Following MMAudio, we project audio latents using a `Conv1d(kernel=7, pad=3)` layer, followed by an SELU activation and a `ConvMLP(kernel=3, pad=1)` module. Text features derived from the CLIP text encoder are mapped directly using a linear layer followed by an MLP. Conversely, visual features from the CLIP visual encoder are processed via a linear layer and a `ConvMLP(kernel=3, pad=1)` module. Additionally, we project synchronization features using an architecture similar to the audio projector, replacing only the final layer with a `ConvMLP(kernel=3, pad=1)`. We employ `Conv1d` and `ConvMLP` in these components as they capture temporal relationships within features more effectively than standard MLPs.

C. Additional Visualization

In video-to-audio generation, especially for multi-shot scenarios such as concert recordings, the model must maintain temporal and semantic consistency across visual segments. When the camera cuts from a wide orchestral shot to close-ups of individual percussionists, the audio should remain

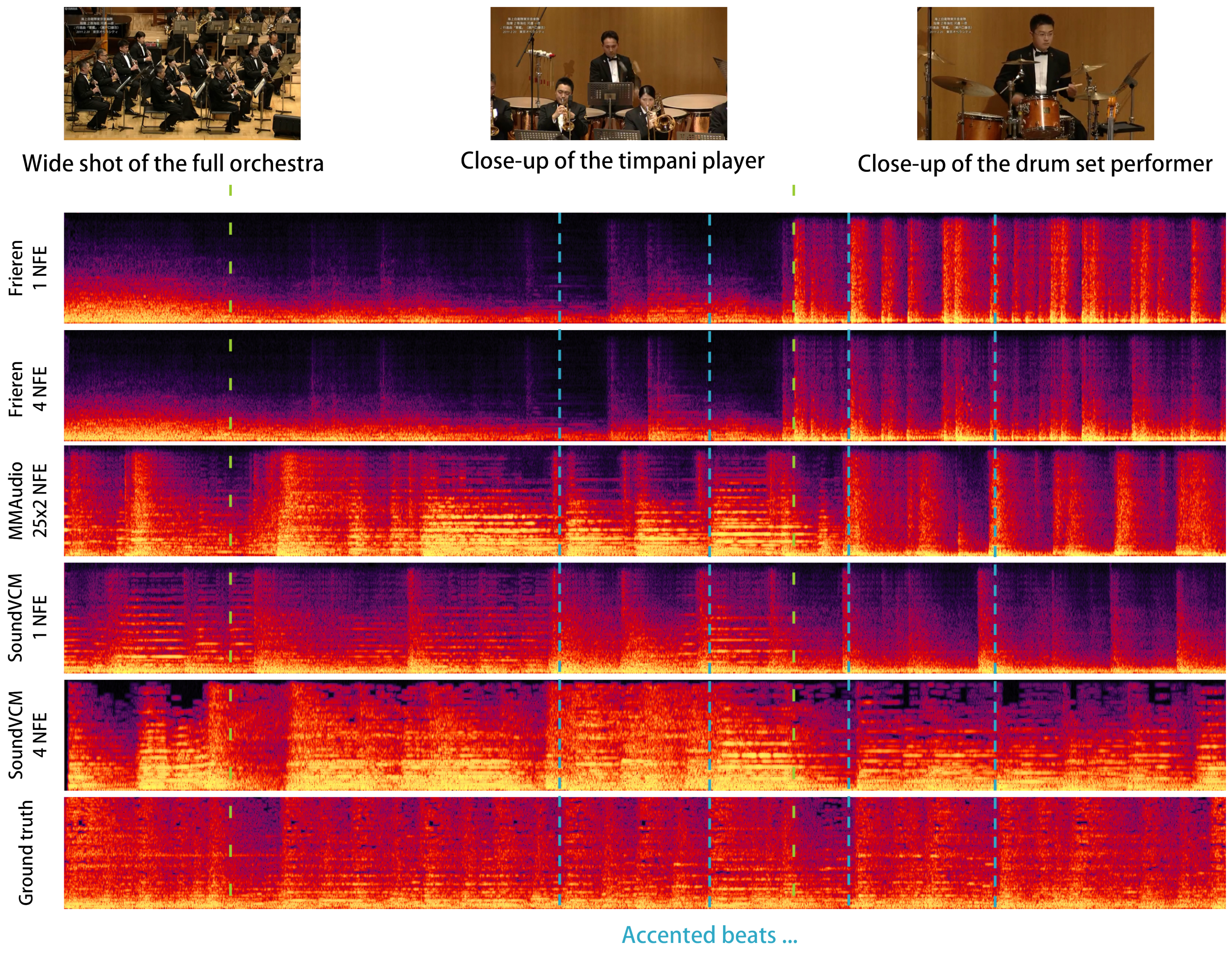

Figure 4. Mel-spectrogram comparison on a VGGSound test clip. Prompt: *playing timpani*. Yellow-green dashed lines indicate visual segment boundaries; blue dashed lines highlight accented beats.

globally coherent rather than respond to each view independently. Although text prompts in VGGSound [2] (*e.g.*, "playing timpani") sometimes capture only local actions, textual guidance still plays a crucial role: text-free models such as Frieren [47] fail to generate reasonable audio in the first two segments and overreact to the third close-up.

Figure 4 compares the mel-spectrograms from several models across three visual segments: an orchestral wide shot, a timpani close-up, and a drum-set close-up. Despite the unified nature of the performance, most baselines introduce discontinuities—melodic components appear in the first two segments but vanish in the third, which collapses into isolated drum hits. This reflects a misinterpretation of shot changes as alterations in the underlying sound source.

Our SoundVCM (4-NFE) achieves the best cross-segment coherence, preserving harmonic structure across all segments and correctly interpreting the shots as different viewpoints of a continuous event. The 1-step version remains relatively consistent as well, though the third segment becomes more percussive; it is still noticeably more stable than other baselines.

Finally, we analyze rhythmic alignment using the accented beats marked by blue dashed lines. Both SoundVCM (1-step and 4-step) and MMAudio [4] (25×2 steps) closely adhere to the ground-truth beat pattern, particularly in segments 2 and 3 where percussion actions are visually explicit. Models that effectively utilize visual timing cues demonstrate clearer and more accurate beat placement, with SoundVCM exhibiting the strongest overall alignment across segments.