DiFlowDubber: Discrete Flow Matching for Automated Video Dubbing via Cross-Modal Alignment and Synchronization

Supplementary Material

Summary

This appendix contains additional materials for the paper *"DiFlowDubber: Discrete Flow Matching for Automated Video Dubbing via Cross-Modal Alignment and Synchronization"*. The appendix is organized as follows:

- Section A presents dataset details, including LibriTTS for zero-shot TTS pre-training and the Chem and GRID benchmarks for video dubbing.
- Section B provides implementation details, covering preprocessing, model architectures for both the zero-shot TTS pre-training stage and the video dubbing adaptation stage, as well as training configurations.
- Section C describes the baseline methods used for comparison.
- Section D reports additional qualitative results, including alignment visualizations of the *Synchronizer* and mel-spectrogram comparisons with baseline systems.
- Section E discusses the limitations of the proposed method.

A. Dataset Details

We adopt a two-stage training process. In the zero-shot TTS pre-training stage, we train on the LibriTTS dataset, while in the video dubbing adaptation stage, we use the Chem and GRID benchmarks.

LibriTTS. The LibriTTS [22] is a large-scale multi-speaker English speech corpus designed for text-to-speech (TTS) research. It contains recordings from numerous speakers with diverse accents, vocal styles, and speaking rates. Each utterance is carefully segmented and aligned with its corresponding text transcript. In our work, we use 470 hours of the LibriTTS dataset as the text-speech corpus for the zero-shot TTS pre-training stage.

Chem. The Chem dataset [10] consists of classroom lecture videos featuring a chemistry instructor, collected from publicly available YouTube recordings [10]. It spans approximately nine hours of content and is segmented at the sentence level to facilitate precise dubbing alignment. The dataset contains 6,082 samples for training, 50 for validation, and 196 for testing.

GRID. The GRID corpus [6] is a standard benchmark for evaluating multi-speaker video dubbing systems. It features recordings from 33 distinct speakers, each contributing 1,000 short English utterances. All recordings were captured under controlled studio conditions with a consistent visual background. The dataset includes 32,670 training instances and 3,280 test instances.

B. Implementation Details

B.1. Preprocessing

We use FACodec [11] as the audio tokenizer, detokenizer, and speaker extractor. Audio is sampled at 16 kHz and tokenized using FACodec at 80 tokens/s, while videos are sampled at 25 FPS, resulting in a fixed 5:16 length ratio between video frames and speech tokens. Each lip-cropped frame is resized to 96×96 pixels. The numbers of quantizers are $m = 1$ for prosody, $n = 2$ for content, and $k = 3$ for acoustic tokens, each with a vocabulary size of 1024. To obtain the ground-truth alignment matrices for video-text and speech-text alignment, we first extract the duration of each phoneme using the Montreal Forced Aligner (MFA) [15]. Based on these durations, we construct the video-text alignment matrix by multiplying each phoneme's duration by the video frame rate (25 FPS) to determine the number of frames corresponding to each phoneme. Similarly, the speech-text alignment matrix is constructed by multiplying the phoneme durations by the token rate (80 tokens/s) to obtain the number of speech tokens aligned with each phoneme.

B.2. Model Details

B.2.1. Zero-shot TTS Pre-training Stage

Content Modeling. Following conventional duration-based alignment modules [12, 18, 19], our architecture consists of a *Phoneme Encoder*, a *Duration Predictor*, a *Length Regulator*, and *Feed-Forward Transformer* (FFT) layers. We model the content by adapting existing duration-alignment mechanisms to the discrete token modeling setting. Specifically, we employ the duration predictor from [18] to estimate the duration (i.e., the number of tokens) for each input phoneme. Each phoneme is then duplicated by the length regulator based on the predicted durations and passed through 2 FFT blocks to refine the representations. Each FFT block contains 4 attention heads, a hidden size of 256, an output dimension of 768, convolutional filter sizes of 1024 with kernel sizes $[9, 1]$, a dropout rate of 0.2, and a maximum sequence length of 5000. After the FFT blocks, the *Content Predictor* employs two FFT blocks with the same configuration to generate two textual representations that capture progressively richer features through the stacked FFT layers. The final output consists of two branches: the first is projected through a linear layer to obtain the *final hidden state*, while the second is projected

onto the vocabulary space to produce *logits* representing the distribution of content tokens.

Discrete Flow Matching Overview. The objective of Discrete Flow Matching is to transform source samples $\mathbf{x}_0 \sim p$ into target samples $\mathbf{x}_1 \sim q$. In our formulation, the source distribution corresponds to all-mask tokens `[MASK]`, while the target distribution $\mathbf{x}_1 = [\mathbf{x}_{p,1}; \mathbf{x}_{a,1}] \in \mathbb{N}^{(m+k)\times L}$ is factorized into prosodic $\mathbf{x}_{p,1} \in \mathbb{N}^{m\times L}$ and acoustic $\mathbf{x}_{a,1} \in \mathbb{N}^{k\times L}$ components, enabling structured joint learning. Following [8], we employ a monotonic scheduler $\kappa_t \in [0, 1]$ with boundary conditions $\kappa_0 = 0$ and $\kappa_1 = 1$, where $t \in [0, 1]$ denotes continuous time. In our implementation, we set $\kappa_t = t^2$. This scheduler progressively interpolates between the source and target distributions, smoothly transitioning from $\mathbf{x}_0$ to $\mathbf{x}_1$ as κ_t increases. We then construct a conditional probability path, referred to as the *mixture path*, which linearly interpolates between the source and target distributions: $p_t(\mathbf{x}^i|\mathbf{x}_0, \mathbf{x}_1) = (1 - \kappa_t)\, p_0(\mathbf{x}^i|\mathbf{x}_0) + \kappa_t\, p_1(\mathbf{x}^i|\mathbf{x}_1)$. This path defines an evolution of discrete states governed by a probability velocity field $\mathbf{u}_t$, expressed as:

$$\mathbf{u}_t^i(\mathbf{x}^i, \mathbf{x}_t) = \frac{\dot{\kappa}_t}{1 - \kappa_t}\Big[p_{1|t}(\mathbf{x}^i|\mathbf{x}_t, \mathbf{c}; \theta) - p_t(\mathbf{x}^i|\mathbf{x}_t)\Big], \quad (1)$$

where $\dot{\kappa}_t$ denotes the time derivative of the scheduler, θ represents the learnable parameters of the probability denoiser, and $p_{1|t}(\cdot|\mathbf{x}_t, \mathbf{c}; \theta)$ is the posterior distribution of $\mathbf{x}_1$ given the partially corrupted sequence $\mathbf{x}_t$ and the conditioning context $\mathbf{c}$.

Denoiser Architecture. We use a Diffusion Transformer (DiT) [17] as the denoiser in our discrete flow matching framework. The DiT consists of 8 layers with a hidden size of 768 and 8 attention heads, and is enhanced with rotary position embeddings (RoPE) [20]. The time step is embedded using an MLP that maps it to a 768-dimensional vector, while the speaker embedding, originally of dimension 256, is linearly projected to the same 768-dimensional space. The global conditioning vector $\mathbf{c}_g$ is then formed by summing the resulting the time embedding $\mathbf{t}$ and the speaker embedding $\mathbf{s}$ projected via an MLP, which are used to condition the Adaptive Layer Normalization (AdaLN) layers. We employ a long skip connection from the input to the output of the final Transformer block. The final prediction stage performs a non-affine layer normalization followed by AdaLN modulation and a linear projection. The global conditioning vector is processed through a SiLU-activated [7] MLP to generate shift and scale parameters that modulate the normalized hidden states. The final linear projection produces an output of $(1 + 3) \times 768$ dimensions, corresponding to one prosody quantizer and three acoustic quantizers.

Contextual Modeling. We describe the construction of the input to the *Denoiser*. The input consists of two parts: the conditioning input, denoted as $\mathbf{e}_{\text{cond}}$, and the current denoising target, $\mathbf{e}_t$ at time step t.

For $\mathbf{e}_{\text{cond}}$, given a reference speech sample $\mathbf{S}^r$, we use FACodec [11] to extract discrete speech tokens along with a speaker embedding, yielding $\mathbf{x}_p^r \in \mathbb{N}^{m\times L'}$, $\mathbf{x}_c^r \in \mathbb{N}^{n\times L'}$, $\mathbf{x}_a^r \in \mathbb{N}^{k\times L'}$, and $\mathbf{s} \in \mathbb{R}^{D_s}$, representing the prosody, content, and acoustic token sequences, and the speaker embedding, respectively. Here, m, n, and k denote the number of Residual Vector Quantization (RVQ) codebooks for prosody, content, and acoustic representations, each with a vocabulary size of v. L' is the token sequence length, and D_s is the speaker embedding dimension. We pass $\mathbf{x}_p^r$ and $\mathbf{x}_a^r$ to dedicated embedders to encode them into latent representations $\mathbf{e}_p^r \in \mathbb{R}^{m\times L'\times D}$ and $\mathbf{e}_a^r \in \mathbb{R}^{k\times L'\times D}$. In the zero-shot TTS setting, which aims to mimic the speaker style of the reference speech, we omit the content tokens $\mathbf{x}_c^r$ and replace them with *zero embeddings*. The final conditioning vector is defined as $\mathbf{e}_{\text{cond}} = [\mathbf{e}_p^r; \mathbf{0}; \mathbf{e}_a^r] \in \mathbb{R}^{(m+n+k)\times L'\times D}$.

For $\mathbf{e}_t$, given the current denoising target tokens $\mathbf{x}_t = [\mathbf{x}_{p,t}; \mathbf{x}_{a,t}] \in \mathbb{N}^{(m+k)\times L}$, we feed $\mathbf{x}_{p,t} \in \mathbb{N}^{m\times L}$ and $\mathbf{x}_{a,t} \in \mathbb{N}^{k\times L}$ into the dedicated embedders to obtain latent representations: $\mathbf{e}_{p,t} = \mathbf{E}_p(\mathbf{x}_{p,t}) \in \mathbb{R}^{m\times L\times D}$ and $\mathbf{e}_{a,t} = \mathbf{E}_a(\mathbf{x}_{a,t}) \in \mathbb{R}^{k\times L\times D}$. Since speech rhythm also depends on linguistic content [25], we incorporate the content information from the *final hidden state* of the content modeling module, denoted as $\tilde{\mathbf{h}}_c$. The final denoising target is then defined as $\mathbf{e}_t = [\mathbf{e}_{p,t}; \tilde{\mathbf{h}}_c; \mathbf{e}_{a,t}] \in \mathbb{R}^{(m+n+k)\times L\times D}$.

Finally, the complete input to the *Denoiser* is constructed by concatenating $\mathbf{e}_{\text{cond}}$ and $\mathbf{e}_t$ along the temporal dimension: $\mathbf{e} = [\mathbf{e}_{\text{cond}}; \mathbf{e}_t] \in \mathbb{R}^{(m+n+k)\times(L'+L)\times D}$. The input $\mathbf{e}$ is permuted to shape $\mathbb{R}^{(L'+L)\times(m+n+k)D}$, projected to $\mathbb{R}^{(L'+L)\times D}$, and fed into the *Denoiser*. The resulting output is then passed through a final transformation layer comprising layer normalization, AdaLN-based modulation conditioned on $\mathbf{c}_g$, and a linear projection, yielding features of shape $\mathbb{R}^{(L'+L)\times(m+k)D}$. We discard the conditioning portion and permute the result to obtain the final hidden representation in $\mathbb{R}^{(m+k)\times L\times D}$, which is then passed through a dedicated MLP to produce the posterior distributions over prosody and acoustic tokens in $\mathbb{R}^{(m+k)\times L\times v}$.

Total Loss. The overall training objective in this stage consists of three components. First, we optimize the *Duration Predictor* using the Mean Squared Error loss on the logarithmic scale, denoted as $\mathcal{L}_d$. Second, $\mathcal{L}_c$ corresponds to the content modeling loss, computed as the Cross-Entropy between the predicted logits and the target content sequence. Third, we employ the discrete flow matching loss, $\mathcal{L}_{\text{DFM}}$ (described in the main paper). The overall loss is formulated as follow:

$$\mathcal{L}_{\text{TTS}} = \lambda_1\mathcal{L}_d + \lambda_2\mathcal{L}_c + \lambda_3\mathcal{L}_{\text{DFM}}, \quad (2)$$

where $\lambda_1 = 0.5$, $\lambda_2 = 1.0$, and $\lambda_3 = 1.0$ are the weighting coefficients that balance the contribution of each term.

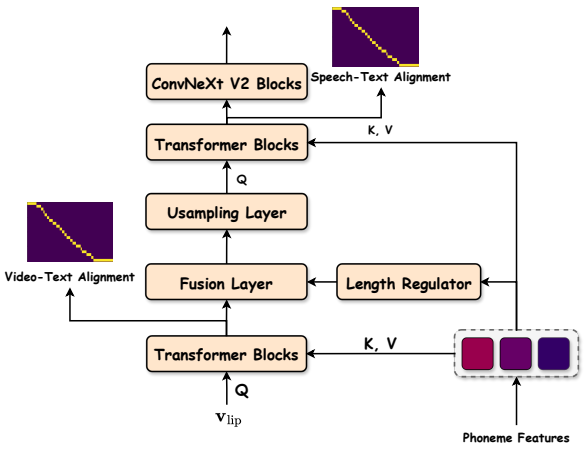

Figure 1. The detailed architecture of *Synchronizer*.

B.2.2. Video Dubbing Adaptation Stage

FaPro Module. The architecture is composed of a learnable upsampling layer, a ConvNeXt V2 [21] encoder stack, layer normalization, and a Transformer decoder. The upsampling layer applies linear interpolation followed by 4 sets of learnable convolutional transformations. Each set consists of a 1D convolution (kernel size 3, stride 1, padding 1), a group normalization layer with 1 group, and a Mish activation [16] function. A final 1D convolution projects the features to the target hidden dimension of 256. The upsampled features are then processed by a stack of 8 ConvNeXt V2 [21] blocks, each with hidden dimension 256 and intermediate dimension 1024. Every block contains a depthwise 1D convolution with kernel size 7, followed by layer normalization and 2 linear projections. Between these projections, we apply a GELU activation [9] and a Global Response Normalization (GRN) module, which stabilizes the feature scale by normalizing the response of each channel with respect to its global L2 magnitude. A residual connection is added around each block, enabling stable and efficient learning of temporal dynamics across long facial sequences. After the ConvNeXt V2 [21] encoder stack, a layer normalization layer is applied before forwarding the features to one Transformer decoder that serves as the *Prosody Predictor*, built on the FFT block architecture. The decoder uses 4 attention heads, a hidden dimension of 256, and a feed-forward expansion ratio of 4, and operates with a maximum sequence length of 5000.

Synchronizer Module. As shown in Figure 1, the module consists of 2 Transformer stacks, a learnable upsampling layer, a light fusion network, a ConvNeXt V2 [21] encoder stack, and two layer normalization layers. We use 8 Transformer blocks for both video-text and speech-text alignment, where the attention mechanism follows a Monotonic Multi-Head Attention [14] formulation with 4 attention heads, a hidden dimension of 256, and an intermediate dimension of 1024. The learnable upsampling layer and the ConvNeXt V2 [21] encoder stack use the same architecture and hyperparameters as those in the *FaPro* module. The light fusion network is implemented as a linear projection layer.

B.3. Training Details

The training is conducted in 2 stages:

Zero-shot TTS Pre-training. We train on 470 hours of the LibriTTS dataset [22] using 4 NVIDIA A100 80GB GPUs with a batch size of 16 for a total of 300k training steps. The learning rate is set to 10^{-4} with a 10% warm-up ratio and a weight decay of 0.01, optimized using the AdamW optimizer [13]. The weighting coefficients λ_1, λ_2, and λ_3 are defined as in Equation 2.

Video Dubbing Adaptation. Training is performed on the Chem [10] and GRID [6] datasets using 4 NVIDIA A100 80GB GPUs with batch sizes of 16 and 64, for 150 and 200 epochs, respectively. We use the AdamW optimizer [13] with a learning rate of 10^{-3}, a weight decay of 0.01, and a 5% warm-up ratio. The weights for the loss $\mathcal{L}_{\text{Dubbing}}$ (defined in the main paper) are set to $\lambda_1 = 1.0$, $\lambda_2 = 0.1$, $\lambda_3 = 0.1$, $\lambda_4 = 1.0$, $\lambda_5 = 0.001$, and $\lambda_6 = 0.001$.

C. Baselines Details

We compare our model with previous dubbing systems, including:

- **V2C-Net** [1] is the baseline model for the V2C task, which generates speech from text while conditioning on both reference audio and video. It fuses visual emotion cues with text and voice identity to produce speech that matches the target speaker and reflects video-driven affect.
- **HPMDubbing** [3] is a hierarchical prosody modeling framework that bridges video features (lip, face, scene) and speech prosody to improve emotional alignment and timing in video dubbing.
- **StyleDubber** [4] switches dubbing learning from the frame level to the phoneme level, using a multimodal style adaptor and phoneme-guided lip aligner to jointly model pronunciation style and visual emotion while maintaining lip-sync.
- **Speaker2Dubber** [23] is a two-stage dubbing method that first pre-trains a phoneme encoder on large-scale TTS data to learn clear pronunciation, then adapts it to video dubbing with prosody and duration consistency learning.
- **ProDubber** [24] introduces prosody-enhanced acoustic pre-training and acoustic-disentangled prosody adapting to fully leverage TTS pre-training, improving acoustic quality and prosody control in dubbing.
- **EmoDubber** [5] is an emotion-controllable dubbing architecture that uses lip-related prosody aligning, pronunciation enhancing, and a flow-based emotion controller

to achieve high-quality lip sync, intelligibility, and user-specified emotion type and intensity.

D. Additional Qualitative Results

D.1. Alignment Visualization of Synchronizer

Figure 2 visualizes the attention maps learned by the *Synchronizer* module. The left panel shows the video-text alignment between lip-frame features and phoneme embeddings, while the right panel shows the speech-text alignment between discrete speech tokens and phonemes. In both cases, the attention concentrates along a clear diagonal, indicating monotonic and fine-grained temporal alignment. This confirms that the Synchronizer successfully learns to align visual and speech streams with the textual content, providing a reliable cross-modal bridge that underpins the strong lip synchronization and pronunciation accuracy observed in our experiments.

D.2. Mel-spectrogram Visualization

We provide additional qualitative comparisons in Figure 3. For 4 representative samples, we visualize the mel-spectrograms of the ground-truth speech, our DiFlowDubber, and all baseline methods. The red bounding boxes highlight regions where different models exhibit noticeable differences in speech quality. Across all samples, our method produces spectra most similar to the ground-truth, preserving clear harmonic structure and temporal dynamics. The voiced regions in our results align well with the visual boundaries, whereas baseline systems often exhibit temporal drift and over-smoothed harmonics, resulting in degraded synchronization.

D.3. Expressive Metrics for Prosody Validation

To effectively validate that the DFPA module generates diverse yet globally consistent prosody under the guidance of the FaPro module, we conduct additional evaluations using a set of expressive prosody metrics that directly measure pitch modeling accuracy and emotional consistency:

- **Gross Pitch Error (GPE)** [2]: Measures the percentage of voiced frames where the relative error between the fundamental frequency F0 of synthesized speech and the ground truth exceeds a predefined threshold (typically 20%), indicating major pitch deviations.
- **Voicing Decision Error (VDE)** [2]: Computes the percentage of frames with incorrect voiced/unvoiced decisions relative to the reference, reflecting rhythmic consistency.
- **F0 Frame Error (FFE)** [2]: Combines GPE and VDE, representing the percentage of frames with either gross pitch error or incorrect voicing decisions, thereby summarizing overall F_0 modeling accuracy.
- **Emotion Similarity (Emo-SIM)**: Measures emotional consistency by computing the cosine similarity between emotion embeddings extracted from the synthesized and reference speech using a pretrained emotion recognition model. In our experiments, we adopt the Emo2Vec[1] model for embedding extraction.

Table 1. Evaluation of prosodic expressiveness and emotional consistency on the Chem dataset (Setting 2.0).

Model	FFE↓	GPE↓	VDE↓	Emo-SIM↑
HPMDubbing	0.535	0.473	0.289	0.979
StyleDubber	0.583	0.493	0.360	0.976
Speaker2Dubber	0.639	0.519	0.408	0.979
ProDubber	0.653	0.562	0.436	0.959
EmoDubber	0.426	0.408	0.220	0.977
DiFlowDubber (ours)	**0.395**	**0.361**	**0.209**	**0.983**

Since the global prosody prior is derived from facial expressions, the target expressive intent is implicitly encoded in the corresponding ground-truth speech. Therefore, we use the ground-truth audio directly as the reference when computing these metrics, enabling us to measure how faithfully the synthesized speech aligns with the intended prosodic and emotional characteristics. As shown in Table 1, we conduct the evaluation under Setting 2.0 to simulate a more challenging scenario for expressive dubbing. The results show that DiFlowDubber achieves the best performance across all expressive metrics. Specifically, our model significantly reduces pitch and voicing errors compared to strong baselines like EmoDubber (0.395 vs. 0.426 FFE), while achieving the highest emotional consistency (0.983 Emo-SIM). These findings confirm that leveraging facial dynamics as a prosody prior effectively guides the DFPA module to generate speech that is both prosodically and emotionally consistent with the visual content.

E. Limitations

Although our proposed method significantly enhances the quality of generated dubbing, it still faces certain limitations. The framework depends on the third-party FACodec [11], from which it inherits certain constraints. In future work, we plan to adapt our system to operate with alternative codec models. Furthermore, the current design does not fully meet expectations in voice cloning. Developing more effective approaches to mimic speaker timbre from audio in real-world video dubbing remains an open area for improvement.

References

[1] Qi Chen, Mingkui Tan, Yuankai Qi, Jiaqiu Zhou, Yuanqing Li, and Qi Wu. V2c: Visual voice cloning. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 21210–21219, 2022. 3

[2] Wei Chu and Abeer Alwan. Reducing f0 frame error of f0 tracking algorithms under noisy conditions with an un-

[1] https://github.com/ddlBoJack/emotion2vec

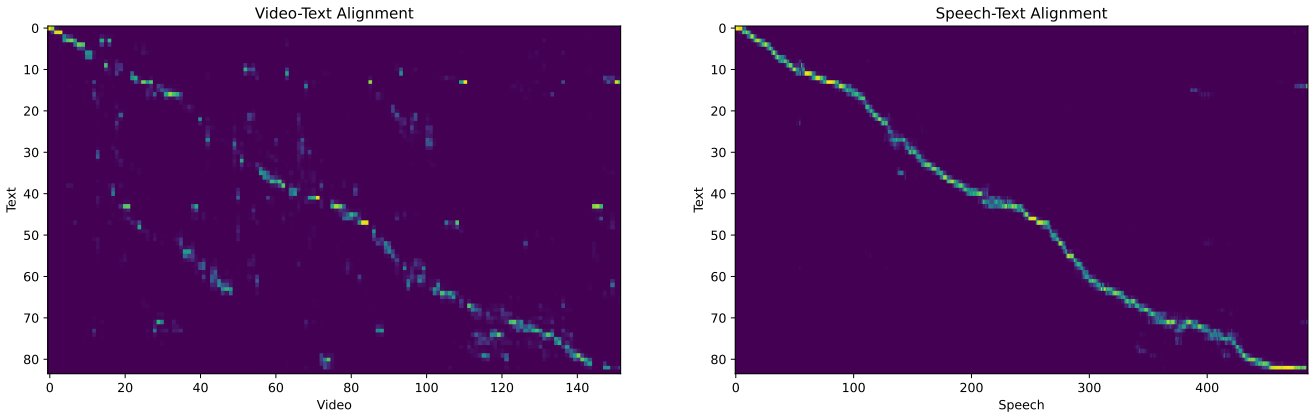

Figure 2. Alignment visualization of the *Synchronizer*. Left: Video-Text alignment matrix. Right: Speech-Text alignment matrix.

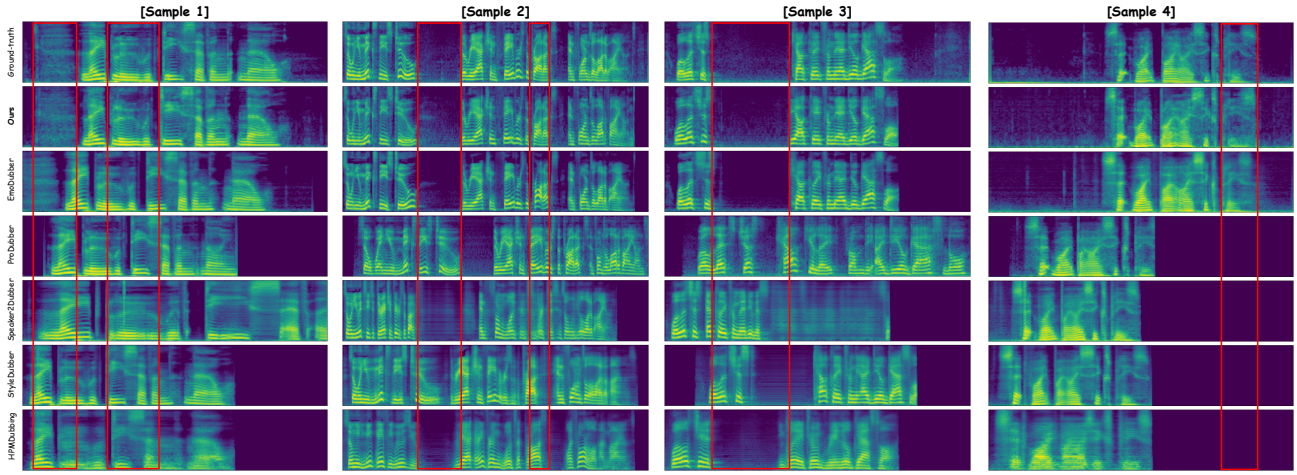

Figure 3. Qualitative comparison of mel-spectrograms. Our method produces spectra closest to the ground-truth, with clearer harmonics, cleaner pauses, and better temporal synchronization than baseline methods.

voiced/voiced classification frontend. In *Proceedings of the 2009 IEEE International Conference on Acoustics, Speech and Signal Processing*, page 3969–3972, USA, 2009. IEEE Computer Society. 4

[3] Gaoxiang Cong, Liang Li, Yuankai Qi, Zheng-Jun Zha, Qi Wu, Wenyu Wang, Bin Jiang, Ming-Hsuan Yang, and Qingming Huang. Learning to dub movies via hierarchical prosody models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14687–14697, 2023. 3

[4] Gaoxiang Cong, Yuankai Qi, Liang Li, Amin Beheshti, Zhedong Zhang, Anton Hengel, Ming-Hsuan Yang, Chenggang Yan, and Qingming Huang. StyleDubber: Towards multi-scale style learning for movie dubbing. In *Findings of the Association for Computational Linguistics ACL 2024*, pages 6767–6779, 2024. 3

[5] Gaoxiang Cong, Jiadong Pan, Liang Li, Yuankai Qi, Yuxin Peng, Anton van den Hengel, Jian Yang, and Qingming Huang. Emodubber: Towards high quality and emotion controllable movie dubbing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 15863–15873, 2025. 3

[6] Martin Cooke, Jon Barker, Stuart Cunningham, and Xu Shao. An audio-visual corpus for speech perception and automatic speech recognition. *The Journal of the Acoustical Society of America*, 120(5):2421–2424, 2006. 1, 3

[7] Stefan Elfwing, Eiji Uchibe, and Kenji Doya. Sigmoid-weighted linear units for neural network function approximation in reinforcement learning. *Neural networks*, 107: 3–11, 2018. 2

[8] Itai Gat, Tal Remez, Neta Shaul, Felix Kreuk, Ricky T. Q. Chen, Gabriel Synnaeve, Yossi Adi, and Yaron Lipman. Discrete flow matching. In *Advances in Neural Information Processing Systems*, pages 133345–133385. Curran Associates, Inc., 2024. 2

[9] Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*, 2016. 3

[10] Chenxu Hu, Qiao Tian, Tingle Li, Wang Yuping, Yuxuan Wang, and Hang Zhao. Neural dubber: Dubbing for videos according to scripts. In *Advances in Neural Information*

Processing Systems, pages 16582–16595. Curran Associates, Inc., 2021. 1, 3

[11] Zeqian Ju, Yuancheng Wang, Kai Shen, Xu Tan, Detai Xin, Dongchao Yang, Eric Liu, Yichong Leng, Kaitao Song, Siliang Tang, Zhizheng Wu, Tao Qin, Xiangyang Li, Wei Ye, Shikun Zhang, Jiang Bian, Lei He, Jinyu Li, and Sheng Zhao. NaturalSpeech 3: Zero-shot speech synthesis with factorized codec and diffusion models. In *Proceedings of the 41st International Conference on Machine Learning*, pages 22605–22623. PMLR, 2024. 1, 2, 4

[12] Sungwon Kim, Kevin J. Shih, Rohan Badlani, Joao Felipe Santos, Evelina Bakhturina, Mikyas T. Desta, Rafael Valle, Sungroh Yoon, and Bryan Catanzaro. P-flow: A fast and data-efficient zero-shot TTS through speech prompting. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. 1

[13] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019. 3

[14] Xutai Ma, Juan Miguel Pino, James Cross, Liezl Puzon, and Jiatao Gu. Monotonic multihead attention. In *International Conference on Learning Representations*, 2020. 3

[15] Michael McAuliffe, Michaela Socolof, Sarah Mihuc, Michael Wagner, and Morgan Sonderegger. Montreal forced aligner: Trainable text-speech alignment using kaldi. In *Interspeech 2017*, pages 498–502, 2017. 1

[16] Diganta Misra. Mish: A self regularized non-monotonic neural activation function. *arXiv preprint arXiv:1908.08681*, 2019. 3

[17] William Peebles and Saining Xie. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 4195–4205, 2023. 2

[18] Yi Ren, Yangjun Ruan, Xu Tan, Tao Qin, Sheng Zhao, Zhou Zhao, and Tie-Yan Liu. Fastspeech: Fast, robust and controllable text to speech. In *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 2019. 1

[19] Yi Ren, Chenxu Hu, Xu Tan, Tao Qin, Sheng Zhao, Zhou Zhao, and Tie-Yan Liu. Fastspeech 2: Fast and high-quality end-to-end text to speech. In *International Conference on Learning Representations*, 2021. 1

[20] Jianlin Su, Murtadha Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. Roformer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568:127063, 2024. 2

[21] Sanghyun Woo, Shoubhik Debnath, Ronghang Hu, Xinlei Chen, Zhuang Liu, In So Kweon, and Saining Xie. Convnext v2: Co-designing and scaling convnets with masked autoencoders. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 16133–16142, 2023. 3

[22] Heiga Zen, Viet Dang, Rob Clark, Yu Zhang, Ron J. Weiss, Ye Jia, Zhifeng Chen, and Yonghui Wu. Libritts: A corpus derived from librispeech for text-to-speech. In *Interspeech 2019*, pages 1526–1530, 2019. 1, 3

[23] Zhedong Zhang, Liang Li, Gaoxiang Cong, Haibing Yin, Yuhan Gao, Chenggang Yan, Anton van den Hengel, and Yuankai Qi. From speaker to dubber: Movie dubbing with prosody and duration consistency learning. In *Proceedings of the 32nd ACM International Conference on Multimedia, MM 2024, Melbourne, VIC, Australia, 28 October 2024 - 1 November 2024*, pages 7523–7532. ACM, 2024. 3

[24] Zhedong Zhang, Liang Li, Chenggang Yan, Chunshan Liu, Anton van den Hengel, and Yuankai Qi. Prosody-enhanced acoustic pre-training and acoustic-disentangled prosody adapting for movie dubbing. In *Proceedings of the Computer Vision and Pattern Recognition Conference (CVPR)*, pages 172–182, 2025. 3

[25] Jialong Zuo, Shengpeng Ji, Minghui Fang, Mingze Li, Ziyue Jiang, Xize Cheng, Xiaoda Yang, Chen Feiyang, Xinyu Duan, and Zhou Zhao. Rhythm controllable and efficient zero-shot voice conversion via shortcut flow matching. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 16203–16217, Vienna, Austria, 2025. Association for Computational Linguistics. 2