Supplemental Material

We provide additional ablation studies regarding flame baselines, landmark conditioning, dataset creation and model architecture in Section A, network architecture details in Section B, metric evaluation in Section C, and additional details regarding dataset creation in Section D. For a visual comparisons of all experiments and ablation studies, we request readers to watch the supplemental video.

A. Additional Ablation Studies

Baselines trained on FLAME fittings of our data. We report results on our hybrid test set with audio from Vocaset, Ours, and LJSpeech. We additionally train baselines on our dataset (using FLAME fittings, as they don't operate on NPHM space) and show results on the same hybrid test set in Tab. 1. Emote and EmoTalk require emotional correspondences and MeshTalk is incompatible with the FLAME topology; thus, a comparison with them is not possible. Our method outperforms baselines on lip-sync (lowest LSE-D), while significantly improving quality (highest FIQA/VQA).

Method	LSE-D ↓	FIQA ↑	VQA ↑
FaceFormer	12.1391	38.64	0.4206
CodeTalker	11.8442	39.44	0.4218
Imitator	11.7402	37.83	0.4492
FaceDiffuser	12.7644	37.65	0.3826
Ours (Flame)	11.6839	38.53	0.4868
Ours (w/o diffusion)	11.3217	42.81	0.5469
Ours (Full)	**11.2737**	**45.75**	**0.6145**

Table 1. All methods, including baselines were trained on our dataset. The baselines and Ours (Flame) were trained with FLAME fittings, and bottom two rows with NPHM fittings. Our Flame baseline also outperforms other methods in lip sync and video quality. Our full model synthesizes highest-quality animations (highest VQA), well-matches real face qualities (highest FIQA), while performing better lip sync (lowest LSE-D).

Additional Conditioning. FaceTalk can also be flexibly adapted to other temporal conditioning signals, such as face landmarks. In Fig 1, we replace input audio with 3D face landmarks extracted by FLAME [7], and train a landmark-conditioned model to synthesize faithful expressions.

Expression Fitting. To evaluate the temporal consistency of our optimized expression sequences, we report Mean Absolute Error (MAE), Root Mean Square Error (RMSE) between adjacent frames and Lip Sync Error Distance(LSE-D) in Tab. 2. We further visualize auto-correlation between the neighboring frames in Fig 2. We notice that our temporal regularization helps stabilize the high-frequency jitter in the fitted expression codes.

Model Architecture Ablations. We ablate different design choices to infuse diffusion timestamps into the model. We experiment with design choices shown in Fig 3 to infuse diffusion timestamps into the model and show results corresponding to these design choices in Tab. 3. We argue that it is critical to carefully inject diffusion timestamps into the model to obtain high-fidelity and diverse results. Using FiLM layers to inject diffusion timestamps outperforms the rest of the design choices, both in producing better lip articulation as well as synthesizing much more diverse results.

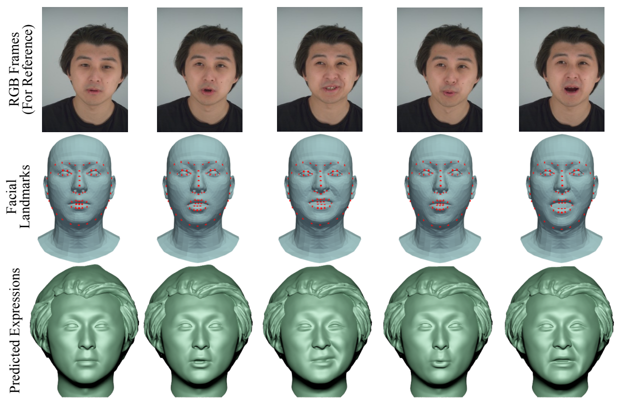


Figure 1. Landmark conditioned expression generation. FaceTalk can also generate accurate expressions (bottom) conditioned on face landmarks (middle), matching the original RGB capture (top).

Method	MAE ↓	RMSE ↓	LSE-D ↓
w/o temporal loss	0.01422	0.01978	10.9221
w/ temporal loss	**0.00248**	**0.00623**	**10.5478**

Table 2. Expression Fitting. Our temporal huber loss helps to stabilize the inter-frame jitter while effectively matching the mouth poses with the audio signal.

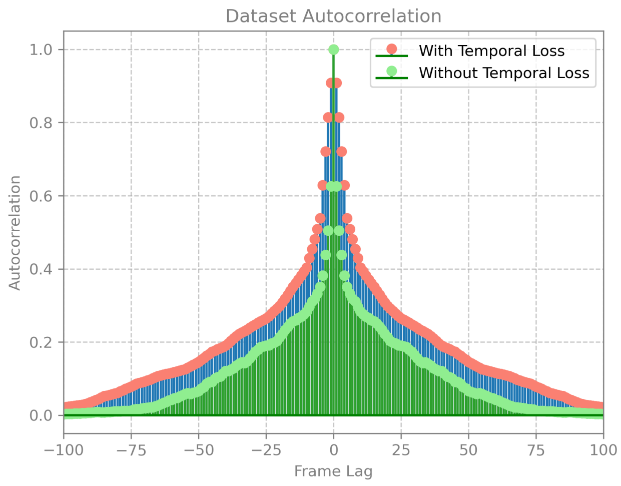


Figure 2. Expression Fitting. Note that with temporal loss, the method achieves a high correlation between neighboring frames.

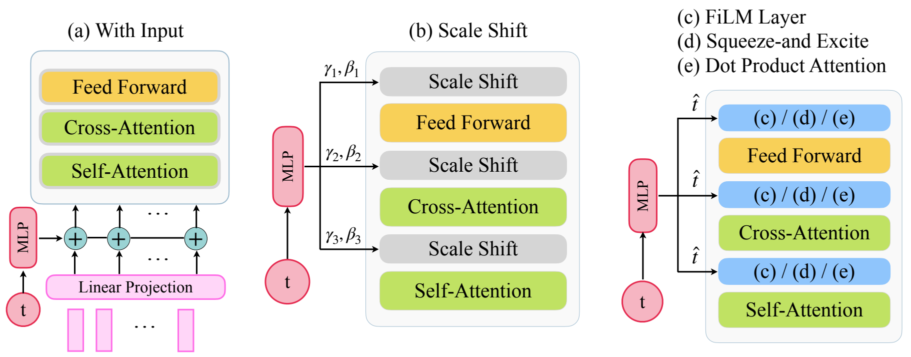


Figure 3. Architecture Design Choices to infuse diffusion timestamp into the model. (a)With Input denotes that diffusion timestamp is simply added to the input expression codes. (b) Scale-Shift modulates the intermediate features after Self-Attention (SA), Cross Attention (CA) and FeedForward (FF) layers of the decoder block with diffusion timestamp, via the corresponding scale and shift parameters γ_i, β_i. (c), (d) and (e) inject the diffusion timestamp into the decoder block with one-layer FiLM [10], Squeeze-and-Excite [5] and vanilla dot-product attention respectively.

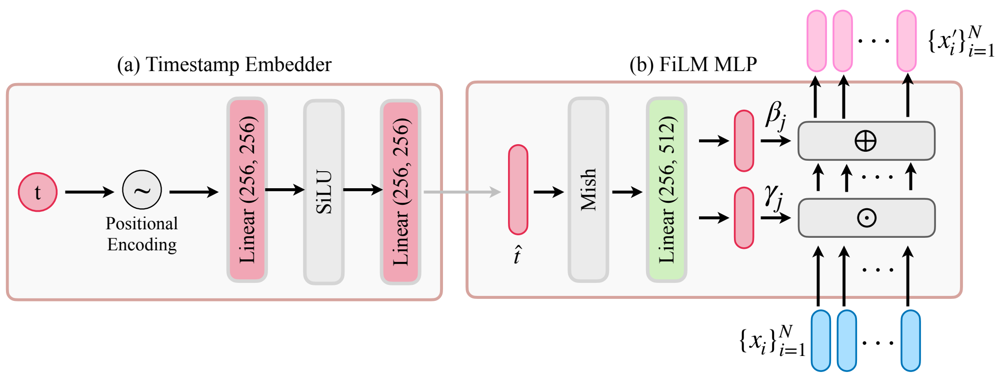


Figure 4. (a) Utilizing a 2-layer MLP architecture with SiLU activation, we project the diffusion timestamp t into the latent space of our expression decoder. The timestamp is first embedded using sinusoidal positional encoding, resulting in a d-dimensional timestamp vector and passed through the MLP layers. (b) FiLM layers are employed to inject the processed diffusion timestamp into the model. The FiLM MLP, featuring Mish activation followed by a linear layer, produces scale γ_j and shift β_j parameters. These parameters modulate incoming temporal expression features which serves as input for subsequent layers.

Our Method	LSE-D ↓	Diversity ↑
Input	12.029	0.006
Scale-Shift	11.9575	0.204
SqueezeExcite [5]	11.7586	0.029
Dot Prod. Attention	12.83	2.009e-7
FiLM [10]	**11.2737**	**0.340**

Table 3. Ablation Study: Effectiveness of different components to fuse diffusion timestamp into the model. Adding it to the input produces muted mouth motion. Using Scale-Shift or Squeeze-Excite layers can perform better lip articulation however produce much less diverse results. Dot Product attention excessively modulates the incoming features, nearly collapsing to a constant expression. Using FiLM layer not only produces better lip articulation but can synthesize much more diverse results.

B. Architecture and Training Details

FaceTalk is implemented in the Pytorch Lightning framework [4, 9]. For mesh extraction, we use Python-based Marching Cubes library [11] and BlenderProc [2] for rendering figures.

Expression Decoder. Our expression decoder consists of a stack of four transformer-based decoder blocks with FiLM [10] layers sandwiched between Multihead Self-Attention, Multihead Cross-Attention and Feedforward Layers, as shown in Fig. 3(c). We use a stacked multi-head transformer decoder model with latent dimension of 256. For Multihead Self- and Cross-Attention layers, we use eight heads and set the dimension to 1024 for each transformer decoder block. The linear projection layer at input projects from the NPHM expression space (200-dim.) to the latent space of our expression decoder (256-dim.), and processes them through the stack of decoder blocks and finally use another linear layer to project the learnt expressions from our latent space (256-dim.) back to NPHM expression space (200-dim).

Timestamp Embedder. To project diffusion timestamp into the latent space of our expression decoder, we use a 2-layer MLP architecture with SiLU [3] activation. The diffusion timestamp $\mathbf{t}$ is first embedded via sinusoidal positional embedding to create d-dimensional timestamp vector as:

$$\mathrm{PE}(t, 2i) = \sin\left(\frac{t}{10000^{2i/d}}\right)$$

$$\mathrm{PE}(t, 2i+1) = \cos\left(\frac{t}{10000^{2i/d}}\right),$$

where t is the position and i is the dimension index. Next, this is passed through the MLP layers and processed timestamp $\hat{\mathbf{t}}$ is obtained. This is shown in Fig 4(a).

FiLM MLP. We use FiLM layers to inject the processed diffusion timestamp $\mathbf{t}$ into the the model. The FiLM MLP consists of a Mish activation [8] followed by a linear layer. The linear layer outputs scale γ_j and shift β_j parameters, which modulates the incoming temporal expression features as:

$$\{x'_i\}_{i=1}^{N} = \left(\{x_i\}_{i=1}^{N} \odot \gamma_j\right) \oplus \beta_j, \tag{1}$$

where $\{x'_i\}_{i=1}^{N}$ refers to the processed expression features, which are input to the next layers. This is shown in Fig. 4(b).

C. Metric Evaluation

Lip Synchronization To evaluate lip synchronization of the generated mouth expressions with the audio signal, we use LSE-D (Lip Sync Error Distance) [12]. Specifically, this involves feeding rendered grayscale face crops and the corresponding audio signal into a pre-trained SyncNet [1] to evaluate how close the acoustic signal matches the phonetic movements. The facial movements are encoded as grayscale crops of only the facial region, and the audio signal is represented as MFCC power spectrum. These are then passed into the pretrained SyncNet backbone [1] and the pairwise distance is evaluated, as shown in Fig. 5.

Diversity. To calculate the visual fidelity, we report the standard GAN metrics KID and FID on the grayscale mouth region crops for all the methods, however, it cannot fully capture how diverse the generated expressions are for a given audio signal. We, thus, evaluate the diversity score D in the latent expression space of NPHM model. For the diversity, we measure how much the generated expression codes diversify for the same audio input. Given a set

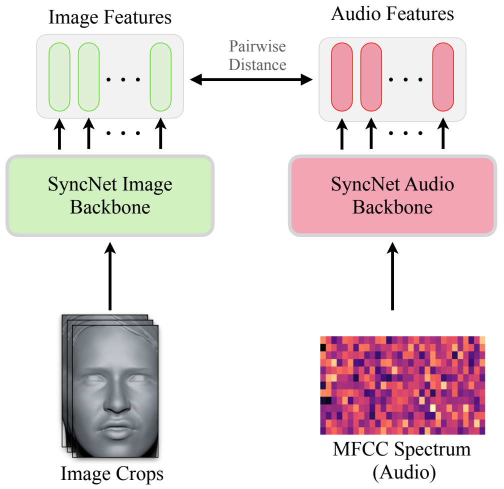


Figure 5. The grayscale image crops are passed to the pretrained Syncnet image backbone to extract image features, and audio features, represented as MFCC power spectrum, are extracted via pretrained Syncnet audio backbone. Finally, the pairwise distance between image and audio features are calculated to compute lip synchronization.

$S = \{S_1, S_2, ...S_K\}$ of generated expression codes (generated from different random noises) for the same audio signal, we calculate the pairwise distance within each set over the entire test dataset as:

$$D = \frac{1}{|N| \times |K|} \sum_{i=1}^{N} \sum_{j=1}^{K} \sum_{l=1, l\neq j}^{K} \|\mathbf{e}_{i,j} - \mathbf{e}_{i,l}\|_2, \tag{2}$$

where $|N|$ refers to the number of audio signals in the test set. $|K|$ refers to the number of generated expression codes for the same audio signal, and $\mathbf{e}$ refers to the synthesized expression codes.

User Study. To evaluate the fidelity based on human perceptual evaluation, we performed a user study with 40 participants. The users were given a carefully crafted set of instructions to evaluate (a) Overall Animation Quality (b) Lip Synchronization and (c) Realism in Facial Movements. The users were asked to assess eight different anonymous methods (including FaceTalk), shown in Fig 6 on these three parameters.

In the course of the study, participants were presented with these questions to focus on different aspects of 3D facial animation, shown in Fig 7. For every question, participants were instructed to meticulously evaluate the provided methods and select the option that best aligned with their judgment. For the first question evaluating overall quality, participants were instructed to consider factors such as visual appeal, clarity, and general impression, and to choose the method number that they believed demonstrates the highest overall quality. Moving on to the second ques-

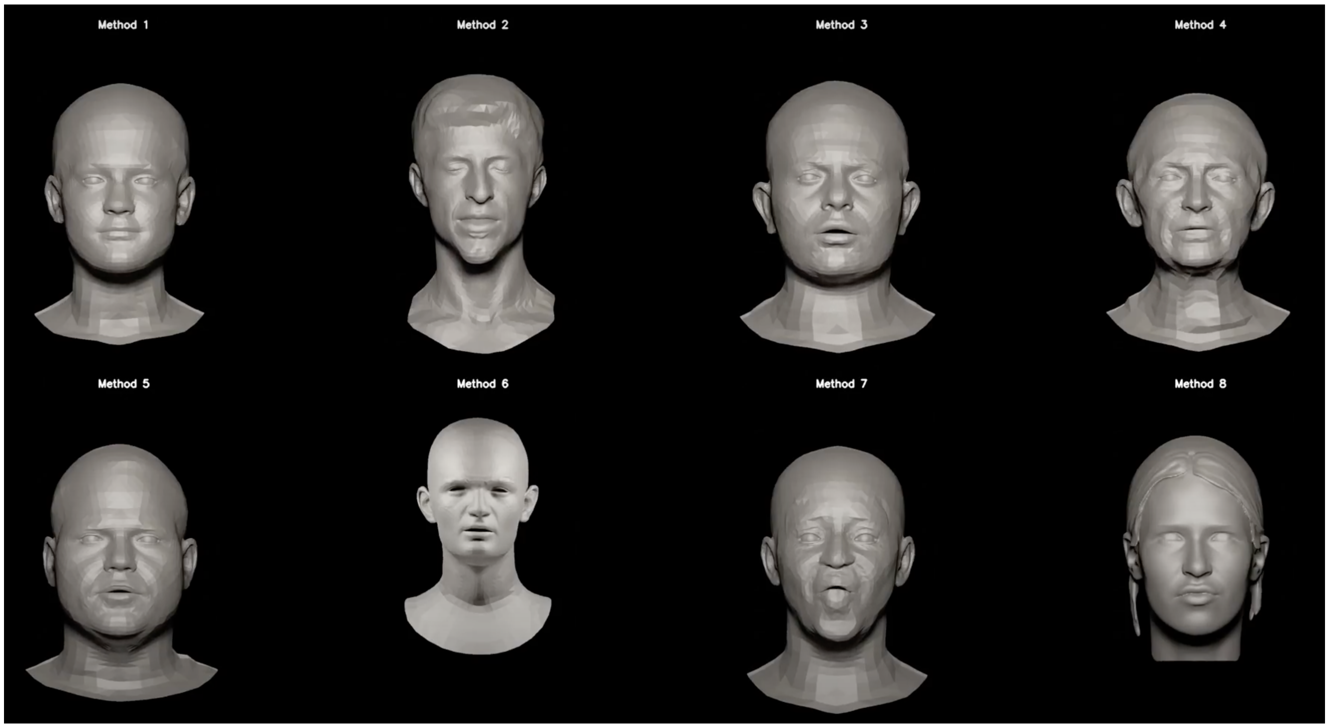
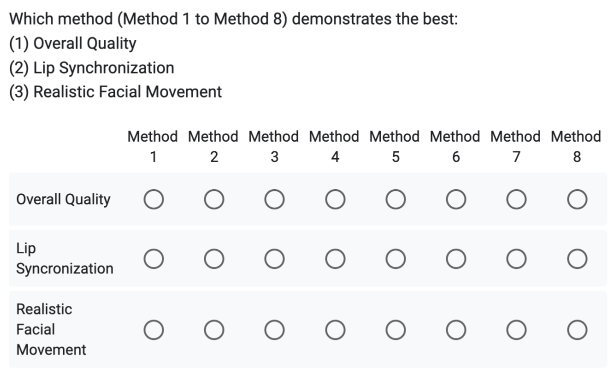


Figure 6. Different methods shown to the users during perceptual evaluation. Method names were anonymized to avoid bias towards a particular method.



Which method (Method 1 to Method 8) demonstrates the best:
(1) Overall Quality
(2) Lip Synchronization
(3) Realistic Facial Movement

	Method 1	Method 2	Method 3	Method 4	Method 5	Method 6	Method 7	Method 8
Overall Quality	○	○	○	○	○	○	○	○
Lip Syncronization	○	○	○	○	○	○	○	○
Realistic Facial Movement	○	○	○	○	○	○	○	○

Figure 7. Questions asked during the user study to assess the quality of different methods. Users were asked to select one of the eight methods for each of the three questions based on which one they believe exhibits best results.

tion, participants were directed to evaluate the lip synchronization of each 3D facial animation method. They were prompted to pay close attention to how well the lip movements aligned with the spoken words or sounds. Participants were reminded to select only one option that, in their judgment, exhibited the best lip synchronization. Lastly, the third question honed in on evaluating the realistic facial movement of each 3D facial animation method. Participants were instructed to consider the naturalness and persuasiveness of facial expressions and movements and to choose the method number that, in their opinion, demonstrates the most realistic facial movement. Again, participants were reminded to select only one option per question throughout the study.

D. Dataset Creation

In the work, we leverage the Nersemble dataset [6] to create the paired audio-NPHM expression dataset. The Nersemble dataset consists of multi-view recordings of people speaking with corresponding audio, for identities present in the shape space of NPHM model. The full dataset creation process is explained in the following steps.

Backprojection. Given the multi-view frames and camera calibrations from Nersemble dataset, using the estimated depth and normals, we first backproject them to 3D to obtain 3D points and 3D landmarks. A depth mask based on valid depth values (between 0 and 1.4) is used during the process. To include only the valid pixels from the estimated depth map, we use 2D facial segmentation masks to include points only for the facial area. The valid depth values and corresponding screen coordinates are extracted based on the depth mask. Screen coordinates are converted to canonical camera coordinates and then to camera coordinates. The resulting camera coordinates are then transformed to world

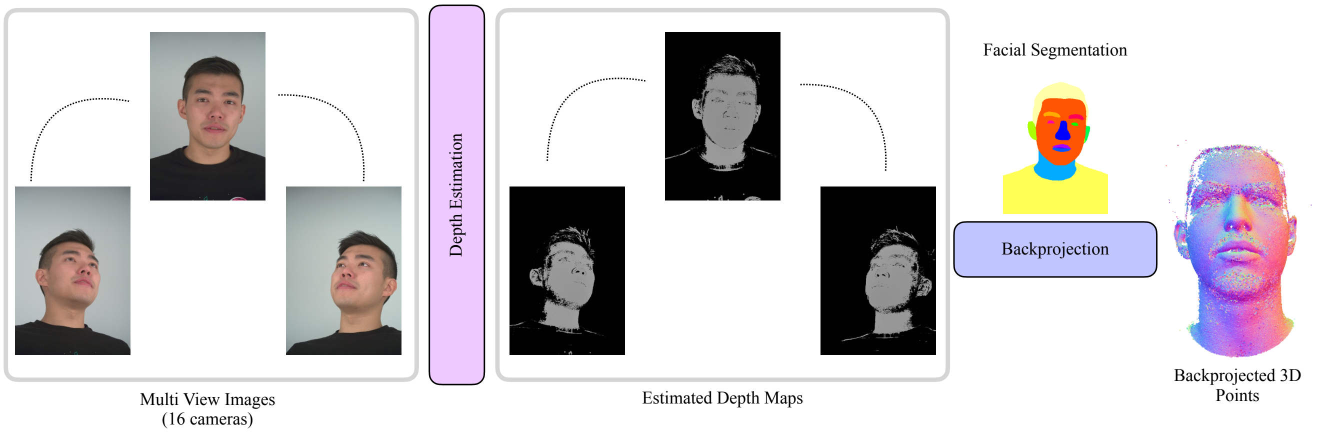


Figure 8. Given a set of multi-view images, we first use COLMAP to estimate depth maps, and backproject these to 3D space only for the pixels corresponding to facial area by leveraging facial segmentation mask. These steps are performed for all the frames in the sequence.

coordinates using the extrinsic parameters. Normal vectors are extracted based on the depth mask and transformed to world coordinates using the rotation part of the extrinsic parameters. This is shown in Fig 8.

Flame Fitting. Next, we align the Flame template mesh [7] to the 3D landmarks obtained above from multi-view data. We compute the rigid transformation $[R_{rigid}, t_{rigid}]$ between the FLAME template landmarks and the valid multi-view landmarks. Outliers are filtered based on the Euclidean distance between the transformed FLAME template landmarks and the original multi-view landmarks. A threshold of 0.020 is used to determine valid points. After filtering outliers, a similarity transformation is recomputed to obtain the transformation scale factor, rotation matrix, and translation vector as $[s_{sim}, R_{sim}, t_{sim}]$.

Initializing Flame parameters $\mathcal{F} = [s, R, t, \theta_{shape}, \theta_{exp}]$ with $[s_{sim}, R_{sim}, t_{sim}]$ obtained above and zero vectors for shape θ_{shape} and expression blendshapes θ_{exp}, we optimize flame parameters $\mathcal{F}$ for entire sequence using an Adam optimizer for 2500 steps, with overall loss $\mathcal{L}_{total}$.

We use several geometric and temporal regularizers to obtain accurate flame fittings. Specifically, $\mathcal{L}_{lmk}$ measures residuals between predicted flame landmarks $L_{flame} \in \mathbb{R}^{68\times3}$ and input landmarks from multi-view stereo $L_{mv} \in \mathbb{R}^{68\times3}$, normalized over all frames N of the sequence as:

$$\mathcal{L}_{lmk} = \frac{1}{N}\sum_{i=1}^{N}\left\|L^i_{flame} - L^i_{mv}\right\|_1 \tag{3}$$

We use different weights for different regions (jaw, eye, mouth). We also use geometric loss $\mathcal{L}_{geo}$ between predicted Flame vertices $V_{flame} \in \mathbb{R}^{5023\times3}$ and backprojected 3D points $P_{mv} \in \mathbb{R}^{K\times3}$, considering both point-to-point and point-to-plane distances. The point-to-point distance is defined as:

$$\mathcal{L}_{point} = \frac{1}{N}\sum_{i=1}^{N}\left\|V^i_{flame} - P^i_{mv_{match}}\right\|_1, \tag{4}$$

and point-to-plane distance is defined as:

$$\mathcal{L}_{plane} = \frac{1}{N}\sum_{i=1}^{N}\left\|V^i_{flame} - P^i_{mv_{match}} \cdot N^i_{mv_{match}}\right\|_1, \tag{5}$$

where $V^i_{flame} \in \mathbb{R}^{5023\times3}$ refers to the predicted Flame vertices for the i^{th} frame, $P^i_{mv_{match}} \in \mathbb{R}^{5023\times3}$ and $N^i_{mv_{match}} \in \mathbb{R}^{5023\times3}$ refer to the points and corresponding normals closest to V^i_{flame}. The geometric loss $\mathcal{L}_{geo}$ is defined as:

$$\mathcal{L}_{geo} = 0.1.\mathcal{L}_{point} + 0.9.\mathcal{L}_{plane}. \tag{6}$$

Next, we employ parameter regularization to penalize large values of shape, expression, and rigid transformation parameters.

$$\mathcal{L}^{reg}_{shape} = \|\theta_{shape}\|_2 \tag{7}$$

$$\mathcal{L}^{reg}_{exp} = \frac{1}{N}\sum_{i=1}^{N}\|\theta^i_{exp}\|_2 \tag{8}$$

$$\mathcal{L}^{reg}_{rigid} = \frac{1}{N}\sum_{i=1}^{N}\left(\frac{\|R^i\|_2}{2\pi} + \|t^i\|_2 + \|s^i\|_2\right). \tag{9}$$

The overall regularization $\mathcal{L}_{reg}$ is then defined as:

$$\mathcal{L}_{reg} = \mathcal{L}^{reg}_{shape} + \mathcal{L}^{reg}_{exp} + \mathcal{L}^{reg}_{rigid}. \tag{10}$$

Finally, we leverage smoothness loss $\mathcal{L}_{smooth}$ to penalize changes in expression and rigid transformations between consecutive frames:

$$\mathcal{L}_{exp}^{smooth} = \frac{1}{N}\sum_{i=2}^{N} \|\theta_{exp}^{i} - \theta_{exp}^{i-1}\|_2 \tag{11}$$

$$\mathcal{L}_{rigid}^{smooth} = \frac{1}{N}\sum_{i=2}^{N} \left(\frac{\|R^i - R^{i-1}\|_2}{2\pi} + \|t^i - t^{i-1}\|_2 \right). \tag{12}$$

The total optimization loss is then defined as:

$$\begin{aligned}\mathcal{L}_{\text{total}} = \lambda_{lmk}\mathcal{L}_{lmk} + \lambda_{geo}\mathcal{L}_{geo} + \lambda_{reg}\mathcal{L}_{reg} \\ + \lambda_{smooth}\mathcal{L}_{smooth}.\end{aligned} \tag{13}$$

Audio Processing The audio captured by the Nersemble dataset contains a lot of background noise and speaking volume of the person is very low. We first amplify the audio signal by increasing it by 20 dB (deciBels), however this adds a lot of white noise to the audio signal. We then process the audio signal to remove this added noise by using the NoiseReduce library [13].

References

[1] J. S. Chung and A. Zisserman. Out of time: automated lip sync in the wild. In *Workshop on Multi-view Lip-reading, ACCV*, 2016. 3

[2] Maximilian Denninger, Dominik Winkelbauer, Martin Sundermeyer, Wout Boerdijk, Markus Knauer, Klaus H. Strobl, Matthias Humt, and Rudolph Triebel. Blenderproc2: A procedural pipeline for photorealistic rendering. *Journal of Open Source Software*, 8(82):4901, 2023. 2

[3] Stefan Elfwing, Eiji Uchibe, and Kenji Doya. Sigmoid-weighted linear units for neural network function approximation in reinforcement learning, 2017. 3

[4] William Falcon et al. Pytorch lightning. *GitHub. Note: https://github. com/PyTorchLightning/pytorch-lightning*, 3 (6), 2019. 2

[5] Jie Hu, Li Shen, Samuel Albanie, Gang Sun, and Enhua Wu. Squeeze-and-excitation networks, 2019. 2

[6] Tobias Kirschstein, Shenhan Qian, Simon Giebenhain, Tim Walter, and Matthias Nießner. Nersemble: Multi-view radiance field reconstruction of human heads. *ACM Trans. Graph.*, 42(4), 2023. 4

[7] Tianye Li, Timo Bolkart, Michael. J. Black, Hao Li, and Javier Romero. Learning a model of facial shape and expression from 4D scans. *ACM Transactions on Graphics, (Proc. SIGGRAPH Asia)*, 36(6):194:1–194:17, 2017. 1, 5

[8] Diganta Misra. Mish: A self regularized non-monotonic activation function, 2020. 3

[9] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. 2017. 2

[10] Ethan Perez, Florian Strub, Harm de Vries, Vincent Dumoulin, and Aaron Courville. Film: Visual reasoning with a general conditioning layer, 2017. 2

[11] pmneila. Pymcubes. `https://github.com/pmneila/PyMCubes`, 2020. 2

[12] K R Prajwal, Rudrabha Mukhopadhyay, Vinay P. Namboodiri, and C.V. Jawahar. A lip sync expert is all you need for speech to lip generation in the wild. In *Proceedings of the 28th ACM International Conference on Multimedia*, page 484–492, New York, NY, USA, 2020. Association for Computing Machinery. 3

[13] Tim Sainburg. timsainb/noisereduce: v1.0, 2019. 6