A Study of 2D-Augmented 3D Speech-Driven Facial Animation

Anonymous CVPR submission

Paper ID 10034

Abstract

Speech-driven 3D facial animation is a challenging problem plagued by ill-posed data that is increasingly expensive to generate and sparse. Most existing works do not attempt to fix this and instead focus on model-based changes which, in recent years, have seen great improvements. In this work, we alternatively explore how 2D signals can be used to augment existing speech-driven 3D facial animation techniques to address the issues the aforementioned issues. We first present a pseudo-labeling technique for generating 3D animation data from 2D videos and introduce 3D-MEAD, a 3D version of the popular MEAD [39] dataset. We also present a novel 2D photometric loss based on techniques used in facial reconstruction to better align 3D facial animations in 2D space and introduce useful regularization. We finally argue against the conventional one-hot encoding for speaker style conditioning and introduce improved speaker embeddings that can be used in a zero-shot manner. We demonstrate how these contributions garner as high as 9.0% *improvements over existing state-of-the-art and present further directions for future work.*

1. Introduction

Building speech-driven generative models that can produce high-fidelity 3D animations is an important problem within AI that has broad applications to fields such as gaming, virtual reality, film production, and online communication. In recent years, manually labeled datasets such as VOCASET [3] and BIWI [16] have accelerated the development of models such as VOCA [3], FaceFormer [15], and CodeTalker [42], which have produced increasingly better animations. Despite these advancements, this field still is encumbered by two problems: First, labeled datasets such as VOCASET and BIWI are extremely costly to obtain, requiring expensive and cumbersome motion capture equipment. Second, the mapping of speech (audio) signals to high-dimensional 3D data (facial meshes) is an ambiguous one-to-many problem — where one speech input can map to more than one compatible animation or style (e.g. variations in emotion, expression, etc.) — which can often lead to unexpressive and low-fidelity animations due to an averaging of motions.

In this paper, we aim to take steps toward addressing the aforementioned issues and explore how 2D signals can be used to augment the 3D task of speech-driven facial animation. To that end, we explore various independent studies and propose a data generation technique, improved speaker style embeddings, and a new photometric loss, and explore how these components can be used to augment the performance of existing state-of-the-art models. For this purpose, we employ the previous state-of-the-art models FaceFormer and CodeTalker, and show that when augmented with our techniques, we can achieve higher-fidelity animations.

Specifically, we develop and demonstrate a method for reconstructing 3D facial animations from 2D videos. We apply this method on the popular MEAD [39] dataset and build *3D-MEAD*. This technique obviates the need for expensive motion-capturing software, and we show how this additional data can be used to augment existing training pipelines and improve performance. Using this augmented data, we are able to achieve state-of-the-art results in a joint training setting.

An issue with current models in speech-driven 3D facial animation is that they are encumbered by one-hot speaker style embeddings, and the choice of speaker ID in evaluation can greatly affect performance. This limits the ability of models to generalize to unseen speakers and makes the choice of speaker ID very important when applying these models in practice. To address this issue, we extended the current state-of-the-art models, namely FaceFormer and CodeTalker, to use learned speaker embeddings, which enable them to be trained on an arbitrary number of speakers and be used in inference with new, unseen speakers. We further demonstrate how these new models outperform their one-hot encoding counterparts.

Finally, in addition to augmenting data, we also investigate how inverse rendering techniques can be used to develop a 2D photometric loss to better align the 3D facial animations in 2D, a novel contribution to this field. As these models will ultimately be used to generate 2D animations

for user consumption, alignment in 2D is important. We show how models trained with this loss perform better, producing higher-fidelity animations.

The contributions of this work are as follows.

- A new 3D dataset (3D-MEAD) [1], that is generated from 2D videos, obviating the need for expensive and cumbersome motion capture equipment.
- A novel joint training method that utilizes the combination of high-quality, motion-captured data (VOCASET) and pseudo-generated data (3D-MEAD).
- A 2D photometric loss using inverse rendering to better align animations in 2D image space.
- Improved learned speaker embeddings that better capture the nuances of speech for improved style conditioning.

2. Related Work

The methods developed in this work relate to several other research fields, namely 3D facial reconstruction and speech-driven 3D facial animation. As an aside, many existing works in speech-driven 2D facial animation (talking heads) exist [1, 5–7, 10, 20, 22, 23, 27, 29, 31, 32, 34, 38, 40] however, we do not cover them in detail here.. The primary difference between 2D talking heads and 3D approaches is that, although both will ultimately be rendered to video, 2D approaches cannot be integrated into game engines or any other 3D virtual or metaverse environment. We review relevant works as follows.

2.1. 3D Reconstruction From 2D Videos

Many single-view 3D face reconstructions methods exist [4, 9, 12, 17, 19, 26, 43, 46], relying on classifiers with ResNet [21] backbones to predict the parameters of a 3D morphable model, pose, texture and displacement maps, and camera information. Recently, methods using synthetic datasets have been explored [4, 12], resulting in improved facial alignment. Aside from classifiers, other approaches rely on using a 3D model prior, photometric consistency losses, and sparse keypoints [36, 47]. More recent work [41] introduces a two-stage approach with denser key points. Our approach to 3D reconstruction from 2D videos builds off this work. The primary concerns with all these methods are (a) the models themselves present a bottleneck in that they can only process high-level features to predict the face alignment, and (b) they rely on manually defined key points and cannot do per-vertex deformations. In our facial reconstruction method used to generate 3D-MEAD, we explore dense face tracking that does not rely on predefined key points.

[1]This dataset will be made publicly available

2.2. Speech-driven 3D facial animation.

Although many earlier methods [11, 13, 14, 24, 37] focusing on predefined facial rigs and rules exist, we focus on data-driven approaches here. VOCA [8] casts 3D facial animation as a regression problem and employs audio feature extraction models to better map from speech to animation. They also propose the popular VOCASET dataset, generated using motion capture equipment, and capture 12 unique speakers. MeshTalk [33] proposes a method for decoupling audio-correlated and audio-uncorrelated information. FaceFormer [15] proposes a transformer-based approach with a pretrained Wav2Vec2.0 [2] audio encoder to better model the long-term dependencies of speech, leading to greater fidelity. CodeTalker [42] builds from FaceFormer and casts the problem as a code query task with a similar transformer backbone. Similar to VOCA and FaceFormer, CodeTalker uses one-hot encodings during training for speaker-style conditioning. Our proposed improved learned speaker embeddings contrasts this method by being able to adapt to new, unseen speakers, in a zero-shot manner. Imitator [35] proposes a pre-trained style-agnostic transformer, which is subsequently optimized for speaker-specific styling based on short reference videos. Imitator also proposes a lip contact loss which guides the model to emphasize lip closure towards the end of a sentence. Although they don't use one-hot encodings, their method still requires short training videos to properly condition on a new speaker. Recently, [44] proposed a cross-modal semi-supervised framework, learning a common latent space from speech and image domains, learning to map speech to image, and finally transforming those images into meshes. They show promising zero-shot performance on VOCASET. This work primarily differs from ours in that we predict meshes in an end-to-end manner, and our dataset generation method only requires videos without artist intervention.

3. Methodology

3.1. Problem Formulation

We formulate the problem of speech-driven 3D facial animation as a sequence-to-sequence (seq2seq) problem as in FaceFormer [15]. See Fig. 1 for a visual. Given speech signal $\mathcal{X}$ and ground truth vertex positions $\Gamma_{1:T} = \{\upsilon_1, \ldots, \upsilon_T\}$, $\upsilon_t \in \mathbb{R}^{V\times 3}$, where V is the number of vertices in the face mesh and T is the number of frames in the sequence, the goal is to translate $\mathcal{X}$ into a sequence of vertex positions $\hat{\Gamma}_{1:T}$ that matches $\Gamma_{1:T}$ as closely as possible. In practice, $\Gamma_{1:T}$ is the change in vertex positions, the vertex offsets, rather than the absolute position, from a given template mesh $h \in \mathbb{R}^{V\times 3}$. This reframing of the problem both normalizes the data and enables training a model that can animate any given template mesh as

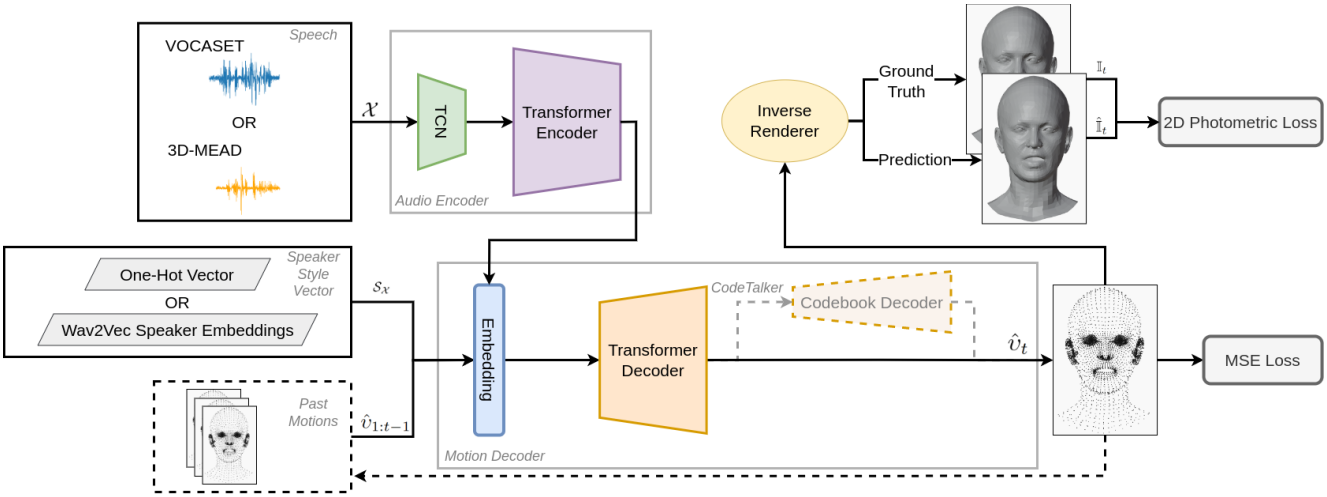


Figure 1. The data flow through the generalized model architecture of our 3D speech-drive facial animation methods. The primary difference between the FaceFormer and CodeTalker backbone, as visualized here, is that CodeTalker's Transformer decoder predicts motion codes that are fed to the codebook decoder, while FaceFormer directly predicts vertex offsets. To augment these baslines, We propose three methods visualized here: first, we introduce a 2D photometric loss to regularize mesh predictions. Second, we use a pre-trained speaker recognition model to generate speaker embeddings (Wav2Vec Speaker Embeddings), replacing the traditional one-hot encoding. Third, we train our model using an augmented dataset (3D-MEAD) jointly with the original dataset (VOCASET).

$H_{1:T} = \{v_1 + h, \ldots, v_T + h\}$.

In addition to the speech signal, the model conditions on a speaker style vector used to represent speaker identities $\mathcal{S}_\mathcal{X} \in \mathbb{R}^{D_S}$. This style vector usually takes the form of a one-hot vector, where $D_\mathcal{S}$ is the number of speakers, however, it can be any vector used to represent a speaker and the specific audio input. We can then formally define a model Φ, that maps input speech to vertex offsets, as

$$\hat{v}_t = \Phi_\theta(\hat{v}_{1:t-1}, \mathcal{S}_\mathcal{X}, \mathcal{X}) \quad (1)$$

where θ is the model parameters and $\hat{v}_t \in \hat{\Gamma}_{1:T};\ t \in [T]$

3.2. Generating Data from 2D Videos

Conventionally, 3D facial animation datasets like VOCASET and BIWI require expensive and cumbersome motion capture equipment, making generating such data very time-consuming and expensive. As a result, existing works [3, 15, 42, 44] have been restricted to these two datasets. To alleviate this issue, we propose a new, cost-effective method for generating 3D facial animation data from 2D videos.

Given a 2D video sequence $\mathcal{I} = \{\imath_1, \ldots, \imath_T\}$, $\imath_t \in \mathbb{R}^{W \times H}$, we employ an in-house dense-landmark prediction model to generate ground-truth facial mesh sequences $\Gamma^*_{1:T} = \{v^*_1, \ldots, v^*_T\}$. This model is similar to [41] and we train it on the Facescape [43] dataset.

We apply our in-house dense-landmark prediction model to the popular audio-visual dataset for emotional talking-faces, MEAD [39] and generate 3D-MEAD. MEAD is a multi-view talking-face video corpus with 43 English speakers, speaking 40 unique sequences with 8 different emotions. For the purposes of this work, similar to VOCASET and BIWI, we focus only on the *neutral* emotion. We split training, validation, and testing sets into 27, 8, and 8 speakers, yielding 1080, 320, and 320 animation sequences, respectively. We also generate a training subset of only 8 speakers from the same set of 27 speakers for certain studies. In all subsets, there is an equal (when possible) split of female and male speakers.

3.3. Speech-driven 3D Facial Animation

In our experiments, we employ FaceFormer [15] and CodeTalker [42] as our two transformer-based backbone models. We briefly describe both models below, but defer to the original papers for greater detail.

3.3.1 Models

Both FaceFormer and CodeTalker use transformer decoders to predict vertex offsets to produce animations (see Fig. 1 for an illustration of the data flow). Both models can be generalized to four primary components: (1) an audio encoder, (2) a style embedding layer, (3) a vertex embedding layer, and (4) a transformer-based vertex decoder. The primary difference between the two is that while FaceFormer predicts vertex offsets for each vertex in the mesh in an end-to-end manner, CodeTalker breaks the problem into two stages: First, a codebook of motion primitives is learned using a variational autoencoder on the target dataset. Second, with the codebook and its decoder frozen, the transformer

Figure 2. Input videos and corresponding 3D facial mesh reconstructions for our method applied on 3D-MEAD.

decoder learns to predict the motion codes of the codebook in order to generate vertex offsets. Both models share similar performance, with CodeTalker having a slight edge, and we show in Sec. 4 how both models can be augmented in different ways to improve on that performance.

Audio encoder: To encode speech inputs $\mathcal{X}$, we use the Wav2Vec 2.0 [2] model. Wav2Vec 2.0 is a generalized model that features an audio feature extractor and transformer-based encoder. The audio feature extractor is composed of temporal convolutions networks (TCN) and processes raw waveform into feature vectors that the encoder converts to contextualized speech representations. Wav2Vec's output is resampled to match the sampling frequency (16 kHz) of both datasets used in this work. As in previous works [15, 42], during training, the TCN and encoder are initialized and frozen with pretrained Wav2Vec 2.0 weights, and a randomly initialized linear projection layer is added on top.

Embedding layers. Both the style embedding layer and vertex embedding layer are linear layers, with a feature size of 64. CodeTalker originally explored a feature size of 1000, however, we found experimentally that results were better with a feature size of 64, as originally proposed by FaceFormer.

Vertex Decoder. The transformer-based vertex decoder is equipped with causal self-attention and cross-modal attention. The causal self-attention is used to learn inter-frame dependencies conditioned on the past motion sequences, while the cross-modal attention aligns audio and motion modalities. Formally, the recursive process of the decoder can be defined as

$$\hat{v}_t = D_{\text{cross-model}}(E_{\text{speech}}(\mathcal{X}), \mathcal{S}_{\mathcal{X}}, \hat{v}_{1:t-1}) \quad (2)$$

where E_{speech} denotes the audio encoder and $D_{\text{cross-modal}}$ is the cross-modal motion decoder.

3.3.2 Improved Speaker Style Embeddings

Previous methods [3, 8, 15, 42] conditioned on speaker-styles using one-hot encodings for speaker identification. A learnable linear layer in the network decoder is then used to map this encoding into style feature vector $\mathbb{Z}_{\mathcal{X}} \in \mathbb{R}^{64}$. This method has two primary drawbacks. First, the model has no way of conditioning on speakers not seen during training. This has the added drawback of forcing the user to choose a speaker code during inference, which can greatly affect the performance of the model. Second, this learned style embedding is fixed and does not have the capability to reflect the nuances of the current input audio signal, such as emotion, volume, pace, etc., all of which can greatly affect the final animation.

To address these issues, we propose to map speaker styles to a common feature space using the input speech signal directly rather than a one-hot encoding. That is, given $\mathcal{X}$, we wish to extract a latent code $\mathbb{A}(\mathcal{S}) = W_{\mathcal{X}}$ using a learned audio encoder $\mathbb{A}(\mathcal{S})$ that better represents the speaker as well as the nuances of their speech. As before, a learnable linear layer is then used to map this latent code into style feature vector $\mathbb{Z}_{\mathcal{X}} \in \mathbb{R}^{64}$.

Our requirements for $\mathbb{A}$ are that the extracted latent code should (1) isolate unique speakers while (2) simultaneously relating types of speech (emotion, pace, volume). To achieve this, we make use of Wav2Vec 2.0 [2] fine-tuned on the SUPERB Speaker Identification Task [45] which utilizes the VoxCeleb1 dataset [30]. We show a t-SNE plot of speaker embeddings speakers and emotions from VOCASET and MEAD in Fig. 3. This figure demonstrates how those requirements are met, separating speakers while also relating styles of speech — in this case emotions — together. We show in Sec. 4 how this new speaker style embedding garners greater expressiveness and performance.

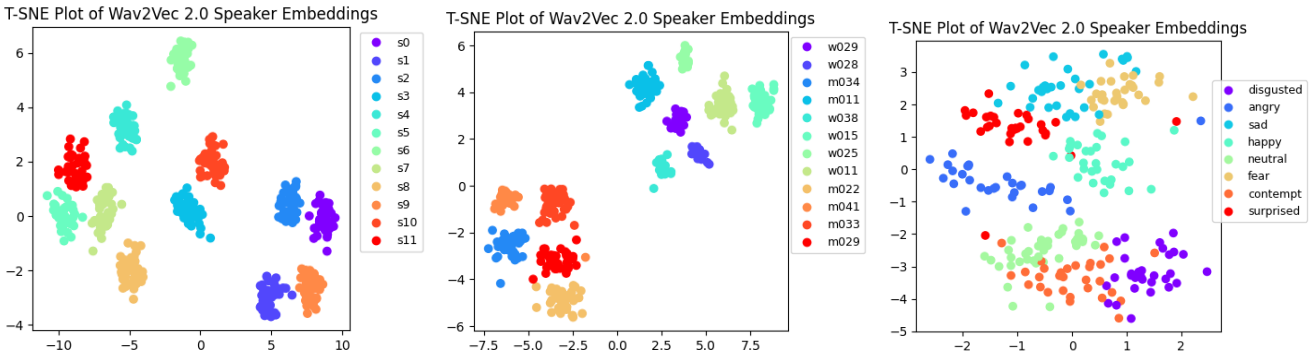
3.4. Training Objectives

Mean Squared Error (MSE) Loss. The primary training objective is the MSE between predicted and ground truth decoder outputs (vertex offsets). Formally, this is defined as

$$\mathcal{L}_{\text{MSE}}(\hat{\Gamma}_{1:T}, \Gamma_{1:T}) = \| \hat{\Gamma_{1:T}} - \Gamma_{1:T} \|_2^2$$

where

$$\| \hat{\Gamma_{1:T}} - \Gamma_{1:T} \|_2^2 = \sum_{t=1}^{T} \sum_{n=1}^{V} \| \hat{v}_{t,n} - v_{t,n} \|^2 \quad (3)$$



(a) VOCASET (b) MEAD (c) MEAD Emotions

Figure 3. T-SNE plots of Wav2Vec 2.0 speaker embeddings for (a) all 12 speakers in VOCASET, (b) 12 random MEAD speakers, and (c) various emotion-labeled speeches for a randomly selected speaker in the MEAD dataset. These plots demonstrate how these speaker embeddings are able to separate speaker identities and sufficiently differentiate emotions within the same speaker.

where V is the number of 3D vertices in the set of vertex offsets (i.e. $v_t \in \mathbb{R}^{V\times 3}$).

FaceFormer-based models are trained without teacher forcing, guided by the MSE loss ($\mathcal{L}_{\text{MSE}}$) between the predicted sequence $\hat{\Gamma}_{1:T}$ and the ground truth $\Gamma_{1:T}$, while CodeTalker-based models are trained with teacher forcing, guided by same predicted sequence MSE ($\mathcal{L}_{\text{MSE}}$) loss as well as an MSE loss between the predicted motion sequences features $\mathbb{Z}_{\Gamma_{1:T}}$ and their quantized features $\mathbb{Z}^q_{\Gamma_{1:T}}$ from the CodeTalker codebook. This additional loss is introduced as a regularization method to the network and was shown to improve performance. We defer to [42] for more details.

2D photometric loss. We introduce the constraint of training with a photometric loss ($\mathcal{L}_{\text{PHO}}$), similar to DECA [18]. Specifically, for each predicted frame $\hat{v}_t$ in the predicted sequence, we inverse render the corresponding mesh into an $W \times H$ sized image $\hat{\mathbb{I}}_t$. Similarly, we inverse render the corresponding ground truth frames v_t into $\mathbb{I}_t$. Given this sequence of images, the photometric loss computes the error between these images as

$$\mathcal{L}_{\text{PHO}} = \sum_{t=1}^{T} \parallel V_{\mathbb{I}} \odot (\hat{\mathbb{I}}_t - \mathbb{I}_t) \parallel_{1,1}, \tag{4}$$

where $V_{\mathbb{I}}$ is a mask that removes the background to isolate the face, and $\odot$ is the Hadamard product. Occluding the background in this manner is important to focus on the facial features and not dilute the changes between prediction and ground truth. The aim of this loss is to introduce additional regularization and ensure that the rendered predictions align with the rendered ground truth.

Final losses. For FaceFormer baselines,

$$\mathcal{L}_{\text{FaceFormer}} = \alpha_\Gamma \mathcal{L}_{\text{MSE}}(\hat{\Gamma}_{1:T}, \Gamma_{1:T}) \tag{5}$$

with $\alpha_\Gamma = 1$ to match the original implementation. For CodeTalker baselines,

$$\begin{aligned}\mathcal{L}_{\text{CodeTalker}} =& \alpha_\Gamma \mathcal{L}_{\text{MSE}}(\hat{\Gamma}_{1:T}, \Gamma_{1:T}) + \\ & \alpha_{\mathbb{Z}} \mathcal{L}_{\text{MSE}}(\mathbb{Z}_{\Gamma_{1:T}}, \mathbb{Z}^q_{\Gamma_{1:T}})\end{aligned} \tag{6}$$

with $\alpha_\Gamma = \alpha_{\mathbb{Z}} = 1$ to match the original implementation. Finally, when training with photometric loss,

$$\mathcal{L}^{\text{2D}}_{\text{FaceFormer}} = \alpha_\Gamma \mathcal{L}_{\text{MSE}}(\hat{\Gamma}_{1:T}, \Gamma_{1:T}) + \alpha_{\text{2D}} \mathcal{L}_{\text{PHO}} \tag{7}$$

with $\alpha_\Gamma = 1$ and $\alpha_{\text{2D}} = 1 \times 10^{-7}$, and

$$\begin{aligned}\mathcal{L}^{\text{2D}}_{\text{CodeTalker}} =& \alpha_\Gamma \mathcal{L}_{\text{MSE}}(\hat{\Gamma}_{1:T}, \Gamma_{1:T}) + \\ & \alpha_{\mathbb{Z}} \mathcal{L}_{\text{MSE}}(\hat{\mathbb{Z}}, \mathbb{Z}) + \alpha_{\text{2D}} \mathcal{L}_{\text{PHO}}\end{aligned} \tag{8}$$

with $\alpha_\Gamma = \alpha_{\mathbb{Z}} = 1$ and $\alpha_{\text{2D}} = 1 \times 10^{-7}$. We explain the loss weights in greater detail in Sec. 4.

4. Experiments

Datasets. We utilize the popular VOCASET [8] to train and test different methods in our experiments, as well as the 3D-MEAD dataset introduced in Sec. 3.2. Both contain 3D facial animations paired with English utterances. VOCASET contains 255 unique sentences, which are partially shared among different speakers, yielding 480 animation sequences from 12 unique speakers. Those 12 speakers are split into 8 unique training, 2 unique validation, and 2 unique testing speakers. Each sequence is captured at 60 fps, resamples to 30 fps, and ranges between 3 and 4 seconds. We use the same training, validation, and testing splits as VOCA and FaceFormer, which we similarly refer to as VOCA-Train, VOCA-Val, and VOCA-Test. For 3D-MEAD, there are 43 unique speakers, where each speaker has 40 unique sequences, yielding a total of 1680 sequences. We randomly split the dataset into 27, 8, and 8

Model	Best LVE ($\times 10^{-5}$ mm) ↓	% imp.
FaceFormer	3.194	—
CodeTalker	3.137	—
FaceFormer^{2D}	3.102	+1.1%
FaceFormer^{W2V}	2.940	+6.2%
CodeTalker^{W2V}	3.050	+2.8%
CodeTalker^{Joint}	2.854	+9.0%

Table 1. A comparison of our models on the VOCA-Test dataset w.r.t. the Best LVE (Lip Vertex Error). "Best" refers to the best tested model from the three seeds used to train the same model. Each% improvement is measured based on a comparison with the best baseline model (CodeTalker).

Model	Mean LVE ($\times 10^{-5}$ mm) ↓	% imp.
FaceFormer	3.252 ± 0.050	—
CodeTalker	3.312 ± 0.281	—
FaceFormer^{2D}	3.172 ± 0.097	+2.4%
FaceFormer^{W2V}	3.056 ± 0.158	+6.0%
CodeTalker^{W2V}	3.084 ± 0.033	+5.2%
CodeTalker^{Joint}	3.012 ± 0.159	+7.4%

Table 2. A comparison of our models on the VOCA-Test dataset w.r.t. Mean LVE. Mean is calculated as the average testing results over the three seeds used to train each model. % improvement is measured based on a comparison with the best *mean* baseline model (FaceFormer).

training, validation, and test speakers. We refer to each split as 3D-MEAD-Train, 3D-MEAD-Val, 3D-MEAD-Test. We additionally subsample 3D-MEAD-Train to generate a 3D-MEAD-Train-8 dataset containing only 8 training speakers, similar to VOCASET-Train. In both datasets, face meshes are composed of 5023 vertices.

Hyper-parameters. We replicate the experimental settings proposed in FaceFormer [15], using the Adam [25] optimizer with a learning rate $\eta = 1 \times 10^{-4}$, batch size of 1, and period set to 30. Where necessary, the audio encoder weights are initialized with the pre-trained Wav2Vec 2.0 [2] weights. Each model is trained for 100 epochs and tested using the final weights after 100. A full hyper-parameter breakdown can be found in Appendix A

Multiple seeds. Unlike other works, we train each experiment three times using the same fixed seeds of $[0, 420, 666]$ and report the average. We found early on that good performance is often heavily dependent on the random seed used during training and SOTA can be achieved by cherry-picking the right seed. Therefore, we report both the average and best-performing models for a more accurate comparison to baselines. We subsequently discuss the results of seed choice and argue for a change in how future works should report performance.

Baselines. We compare our methods against the two state-of-the-art models FaceFormer and CodeTalker. As done in their respective original works, for testing on unseen subjects, we condition FaceFormer and CodeTalker on all training speakers and average their results.

Measuring performance. Following previous works, we report quantitative performance using the **L**ip **V**ertex **E**rror (LVE) to measure lip synchronization with ground truth. As far as we know, this is the only widely used metric for this task. LVE calculates the mean over all frames of the maximal L2 error of all lip vertices. As previous works do not explicitly define this metric, we do so here as follows. For a single frame in the n^{th} predicted test sample sequence $\hat{v_{n,t}} \in [\hat{\Gamma}^n_{1:T}]$, the maximal L2 error of lip vertices V_{lips} is

$$\ell_2^{lip}(\hat{v}_t) = \max_{v \in V_{lips}} \left[\sum_{i=1}^{3} (v^{v,i} - \hat{v}^{v,i})^2 \right]. \quad (9)$$

The LVE for the entire test set of predicted sequences $\{\hat{\Gamma}^1_{1:T}, \ldots, \hat{\Gamma}^N_{1:T}\}$ is then

$$LVE = \frac{1}{N * T} \sum_{n=1}^{N} \sum_{t=1}^{T} \ell_2^{lip}(\hat{v}_{n,t}) \quad (10)$$

4.1. Photometric Loss Experiment

Our first experiment involves augmenting the baseline models (with one-hot speaker style encodings) with the photometric loss proposed in Sec. 3.4. That is, we train and test FaceFormer^{2D} and CodeTalker^{2D} using $\mathcal{L}^{2D}_{\text{FaceFormer}}$ and $\mathcal{L}^{2D}_{\text{CodeTalker}}$, respectively. Experimentally, MSE loss generally ranges on the order of magnitude of 1×10^{-7}, while photometric losses generally range on the order of magnitude of 1×10^{1}. As the photometric loss is meant to regularize the network and not drown out the MSE loss, we use an α_{2D} of $1e-7$ to scale the loss to an acceptable range. Further improvements may come from a more comprehensive hyper-parameter optimization study, which we leave as future work.

We see in Tab. 1 and Tab. 2 that introducing this photometric loss in FaceFormer^{2D} garners a 1.1% improvement for the best LVE and a 2.4% improvement for mean LVE on VOCA-Test. CodeTalker^{2D} on the other hand does not see similar benefits, and performance is even harmed. We hypothesize that this is due to CodeTalker's use of the additional $\mathcal{L}_{\text{MSE}}(\mathbb{Z}_{\Gamma_{1:T}}, \mathbb{Z}^q_{\Gamma_{1:T}})$ loss. As this loss is also meant to act as a regularizer, the additional photometric loss over-regularizes the network. Perhaps additional hyper-parameter tuning of $\alpha_{\mathbb{Z}}$ and α_{2D} is required in this context to better balance the regularization, and we leave this exploration as future work. Nonetheless, FaceFormer^{2D}'s empirical results support our hypothesis that the 2D photometric

loss could promote alignment in 2D image space, yielding a performance boost.

4.2. Improved Speaker Embeddings

Our second experiment involves training our models with improved speaker style embeddings introduced in Sec. 3.3.2. We experiment with replacing the one-hot encodings in both baselines, labeled FaceFormer$^{\text{W2V}}$ and CodeTalker$^{\text{W2V}}$. From Tab. 1 and Tab. 2, FaceFormer$^{\text{W2V}}$ outperforms the state-of-the-art by 6.2% for best LVE and 6.0% for mean LVE. Similarly, CodeTalker$^{\text{W2V}}$ achieves greater performance than baselines, outperforming it by 2.8% for best LVE and 5.2% for mean LVE. These results validate our motivation to expose the model to richer speaker representations, yielding better audio mapping to 3D mesh animation.

4.3. Joint Training with 3D-MEAD

In our third experiment, we perform joint training with our generated dataset, 3D-MEAD. To focus on the dataset, we do not use the 2D photometric loss or improved speaker style embeddings and use the baseline models directly. Similarly, we use the 3D-MEAD-Train-8 split to match the number of training speakers in VOCASET to avoid a dataset imbalance. This results in a larger one-hot encoding of dimension 16 instead of 8.

We experiment with 3 different models, namely FaceFormer$^{\text{Joint}}$ (naive), CodeTalker$^{\text{Joint}}$ (naive), and CodeTalker$^{\text{Joint}}$ (ours). The *naive* label comes from naively training the baselines on this combined dataset. Inherently, the meshes in 3D-MEAD and VOCASET are not aligned with each other, and the distribution of face motions will be different given the differences in generation (we show examples in Appendix B). This misalignment will introduce additional ambiguities and confusion in the model that hinders performance. To alleviate this alignment issue, we constrain CodeTalker to only predict vertex offsets aligned with VOCASET by training its stage-1 codebook only on VOCASET and then using the combined dataset to train the stage-2 decoder. We denote that model with the *ours* label. We can see from Tab. 3 that our best-performing model is CodeTalker$^{\text{Joint}}$ (ours), resulting in a 9.0% improvement over the baseline. In contrast, Faceformer$^{\text{Joint}}$ (naive) and CodeTalker$^{\text{Joint}}$(naive) achieve -40.6% and -19.9%, respectively, highlighting how important data-alignment is. Importantly, we mention here that CodeTalker's architecture enables it to handle this alignment more easily than FaceFormer, where additional data post-processing steps would need to be made in order for FaceFormer to properly take advantage of this additional data. Despite FaceFormer's baseline exhibiting great Mean LVE performance over CodeTalker, this is a feature it lacks.

Model	Best LVE ↓ ($\times 10^{-5}$ mm)	% imp.
FaceFormer	3.194	—
CodeTalker	3.137	—
FaceFormer$^{\text{Joint}}$ (naive)	4.410	-40.6%
CodeTalker$^{\text{Joint}}$ (naive)	3.761	-19.9%
CodeTalker$^{\text{Joint}}$ (ours)	2.854	+9.0%

Table 3. Ablation study of joint-training on the VOCA-Test dataset w.r.t. Best LVE. % improvement is measured based on comparison with the best baseline model (CodeTalker).

4.4. User Study

Model	Dataset	Realism↑	LipSync↑
CodeTalker$^{\text{Joint}}$ vs. GT	VOCA	47.57	50.00
CodeTalker$^{\text{W2V}}$ (VOCA) vs. CodeTalker$^{\text{W2V}}$ (3D-MEAD)	BEAT	53.40	50.00

Table 4. A/B tested user study results on VOCA-Test and BEAT. We report the percentage of answers where A is preferred over B.

We conduct a qualitative user study similar to [15, 42] to perform two comparative studies: First, to evaluate how well our best model CodeTalker$^{\text{Joint}}$ approximates the ground truth (GT), we compare it on VOCA-Test. Second, to evaluate the quality of 3D-MEAD, we compare a CodeTalker$^{\text{W2V}}$ model trained only on VOCA-Train against a CodeTalker$^{\text{W2V}}$ model trained only on 3D-MEAD-Train and test both in a zero-shot manner on the BEAT [28] dataset. BEAT is a large-scale audio-to-gesture dataset, but for this work, we only focus on using a subsample of its audio inputs, truncated to ten-second clips.

Similar to [42], we adopt A/B tests for each comparison in terms of realistic facial animations and lip sync [2]. For both VOCA-Test and BEAT, 32 speech samples are randomly selected, and for models with one-hot encodings, each training speaker is evenly distributed among these 32 samples, yielding at least 4 speech samples per speaker ID. This ensures a fair comparison of models across speaker IDs. This results in 128 A vs. B pairs (32 samples $\times$ 4 comparisons) for the first study, and 32 A vs. B pairs for the second. Each pair is judged by at least 3 different participants, and in total, we collect 480 comparisons.

We tabulate the percentage of A/B testing in Tab. 4 and show that our CodeTalker$^{\text{Joint}}$ model is in fact equally preferred to the ground truth on lip sync, and only slightly less preferred on realism. This first study justifies that

[2]We found from early focus groups that sometimes, it's practically impossible to select one pair over the other, so a third *"I don't know"* option is made available.

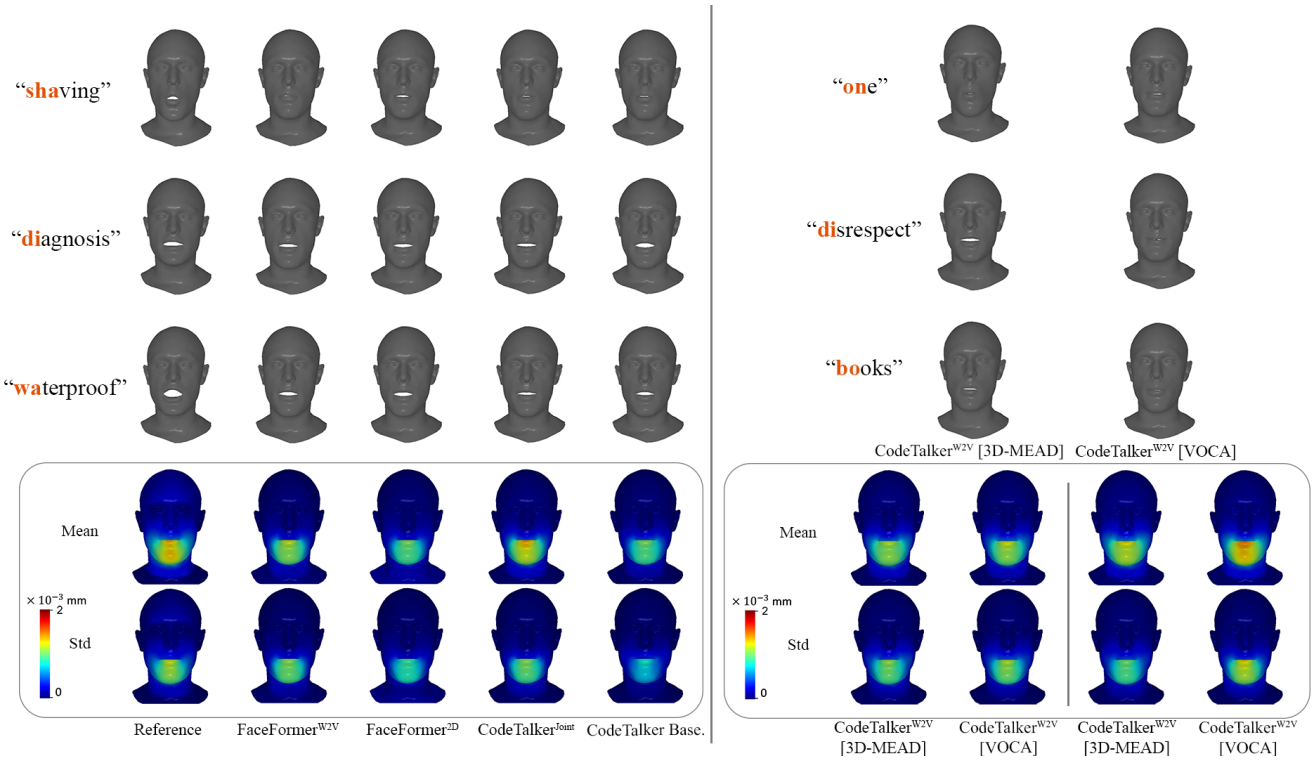


Figure 4. A visual comparison of predicate facial animations by different methods on VOCA-Test (left) and a comparison of zero-shot facial animations on BEAT with CodeTalker^{W2V} trained on 3D-MEAD or VOCASET (right). The upper grey renders show the keyframes for different parts-of-speech, while the section below visualizes the temporal statistics (mean and standard deviation) of adjacent-frame motion variations within a single sequence.

our method of joint training yields high-fidelity animations. Similarly, Tab. 4 also shows how a model trained exclusively on 3D-MEAD is very competitive to one trained on VOCA-Train in a zero-shot setting, which highlights the quality of our pseudo-generated dataset.

4.5. On Seeds and LVE

As seen in Tab. 1 and Tab. 2, while CodeTalker baseline exhibits better best-LVE over FaceFormer, FaceFormer has better mean-LVE across the three seeds. CodeTalker and other models also exhibit quite a large standard deviation in results, which opens the door for cherry-picked results in the future that could potentially mask the true performance of these models. As a result, we argue for a more fair comparison by reporting averaged results across multiple runs or seeds, as is commonly done in other AI fields and as was done in this work.

5. Conclusion

We explored how 2D signals can be used to augment the task of 3D speech-driven facial animation. We presented a new 2D photometric loss for better mesh regularization and alignment in 2D image space, improved speaker style embeddings, and techniques for 3D mesh generation from 2D videos to augment training data, all of which garnered improvements in their own rights. A limitation of our work lies in the lack of merging of these methods together, which is not a trivial task. As each method introduces new variations in learning and regularization, naively combining them does not yield benefits above those presented in this work, and we leave that exploration as future work. Nonetheless, our independent studies showed great improvements over baselines and good qualitative comparisons to ground truth.

References

[1] Mohammed M Alghamdi, He Wang, Andrew J Bulpitt, and David C Hogg. Talking head from speech audio using a pre-trained image generator. In *Proceedings of the 30th ACM International Conference on Multimedia*, pages 5228–5236, 2022. 2

[2] Alexei Baevski, Yuhao Zhou, Abdelrahman Mohamed, and Michael Auli. wav2vec 2.0: A framework for self-supervised learning of speech representations. In Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin, editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, De-*

cember 6-12, 2020, virtual, 2020. 2, 4, 6, 1

[3] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *European conference on computer vision*, pages 213–229. Springer, 2020. 1, 3, 4

[4] Zenghao Chai, Tianke Zhang, Tianyu He, Xu Tan, Tadas Baltrušaitis, HsiangTao Wu, Runnan Li, Sheng Zhao, Chun Yuan, and Jiang Bian. Hiface: High-fidelity 3d face reconstruction by learning static and dynamic details, 2023. 2

[5] Lele Chen, Guofeng Cui, Celong Liu, Zhong Li, Ziyi Kou, Yi Xu, and Chenliang Xu. Talking-head generation with rhythmic head motion. In *European Conference on Computer Vision*, pages 35–51. Springer, 2020. 2

[6] Lele Chen, Zhiheng Li, Ross K Maddox, Zhiyao Duan, and Chenliang Xu. Lip movements generation at a glance. In *Proceedings of the European conference on computer vision (ECCV)*, pages 520–535, 2018.

[7] Joon Son Chung and Andrew Zisserman. Out of time: automated lip sync in the wild. In *Computer Vision–ACCV 2016 Workshops: ACCV 2016 International Workshops, Taipei, Taiwan, November 20-24, 2016, Revised Selected Papers, Part II 13*, pages 251–263. Springer, 2017. 2

[8] Daniel Cudeiro, Timo Bolkart, Cassidy Laidlaw, Anurag Ranjan, and Michael J. Black. Capture, learning, and synthesis of 3d speaking styles. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, pages 10101–10111. Computer Vision Foundation / IEEE, 2019. 2, 4, 5

[9] Radek Danecek, Michael J. Black, and Timo Bolkart. Emoca: Emotion driven monocular face capture and animation, 2022. 2

[10] Dipanjan Das, Sandika Biswas, Sanjana Sinha, and Brojeshwar Bhowmick. Speech-driven facial animation using cascaded gans for learning of motion and texture. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXX 16*, pages 408–424. Springer, 2020. 2

[11] José Mario De Martino, Léo Pini Magalhães, and Fábio Violaro. Facial animation based on context-dependent visemes. *Computers & Graphics*, 30(6):971–980, 2006. 2

[12] Yu Deng, Jiaolong Yang, Sicheng Xu, Dong Chen, Yunde Jia, and Xin Tong. Accurate 3d face reconstruction with weakly-supervised learning: From single image to image set. In *IEEE Computer Vision and Pattern Recognition Workshops*, 2019. 2

[13] Pif Edwards, Chris Landreth, Eugene Fiume, and Karan Singh. Jali: an animator-centric viseme model for expressive lip synchronization. *ACM Transactions on graphics (TOG)*, 35(4):1–11, 2016. 2

[14] Tony Ezzat and Tomaso Poggio. Miketalk: A talking facial display based on morphing visemes. In *Proceedings Computer Animation'98 (Cat. No. 98EX169)*, pages 96–102. IEEE, 1998. 2

[15] Yingruo Fan, Zhaojiang Lin, Jun Saito, Wenping Wang, and Taku Komura. Faceformer: Speech-driven 3d facial animation with transformers. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2022, New Orleans, LA, USA, June 18-24, 2022*, pages 18749–18758. IEEE, 2022. 1, 2, 3, 4, 6, 7

[16] Gabriele Fanelli, Jürgen Gall, Harald Romsdorfer, Thibaut Weise, and Luc Van Gool. A 3-d audio-visual corpus of affective communication. *IEEE Trans. Multim.*, 12(6):591–598, 2010. 1

[17] Yao Feng, Haiwen Feng, Michael J. Black, and Timo Bolkart. Learning an animatable detailed 3d face model from in-the-wild images. *CoRR*, abs/2012.04012, 2020. 2

[18] Yao Feng, Haiwen Feng, Michael J Black, and Timo Bolkart. Learning an animatable detailed 3d face model from in-the-wild images. *ACM Transactions on Graphics (ToG)*, 40(4):1–13, 2021. 5

[19] Jianzhu Guo, Xiangyu Zhu, Yang Yang, Yang Fan, Zhen Lei, and Stan Li. *Towards Fast, Accurate and Stable 3D Dense Face Alignment*, pages 152–168. 11 2020. 2

[20] Yudong Guo, Keyu Chen, Sen Liang, Yong-Jin Liu, Hujun Bao, and Juyong Zhang. Ad-nerf: Audio driven neural radiance fields for talking head synthesis. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5784–5794, 2021. 2

[21] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition, 2015. 2

[22] Xinya Ji, Hang Zhou, Kaisiyuan Wang, Qianyi Wu, Wayne Wu, Feng Xu, and Xun Cao. Eamm: One-shot emotional talking face via audio-based emotion-aware motion model. In *ACM SIGGRAPH 2022 Conference Proceedings*, pages 1–10, 2022. 2

[23] Xinya Ji, Hang Zhou, Kaisiyuan Wang, Wayne Wu, Chen Change Loy, Xun Cao, and Feng Xu. Audio-driven emotional video portraits. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 14080–14089, 2021. 2

[24] Gregor A Kalberer and Luc Van Gool. Face animation based on observed 3d speech dynamics. In *Proceedings Computer Animation 2001. Fourteenth Conference on Computer Animation (Cat. No. 01TH8596)*, pages 20–251. IEEE, 2001. 2

[25] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In Yoshua Bengio and Yann LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. 6, 1

[26] Biwen Lei, Jianqiang Ren, Mengyang Feng, Miaomiao Cui, and Xuansong Xie. A hierarchical representation network for accurate and detailed face reconstruction from in-the-wild images, 2023. 2

[27] Borong Liang, Yan Pan, Zhizhi Guo, Hang Zhou, Zhibin Hong, Xiaoguang Han, Junyu Han, Jingtuo Liu, Errui Ding, and Jingdong Wang. Expressive talking head generation with granular audio-visual control. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3387–3396, 2022. 2

[28] Haiyang Liu, Zihao Zhu, Naoya Iwamoto, Yichen Peng, Zhengqing Li, You Zhou, Elif Bozkurt, and Bo Zheng. Beat: A large-scale semantic and emotional multi-modal dataset for conversational gestures synthesis. In *European Conference on Computer Vision*, pages 612–630. Springer, 2022. 7

[29] Xian Liu, Yinghao Xu, Qianyi Wu, Hang Zhou, Wayne Wu, and Bolei Zhou. Semantic-aware implicit neural audio-

driven video portrait generation. In *European Conference on Computer Vision*, pages 106–125. Springer, 2022. 2

[30] Arsha Nagrani, Joon Son Chung, Weidi Xie, and Andrew Zisserman. Voxceleb: Large-scale speaker verification in the wild. *Computer Speech & Language*, 60:101027, 2020. 4

[31] Youxin Pang, Yong Zhang, Weize Quan, Yanbo Fan, Xiaodong Cun, Ying Shan, and Dong-ming Yan. Dpe: Disentanglement of pose and expression for general video portrait editing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 427–436, 2023. 2

[32] KR Prajwal, Rudrabha Mukhopadhyay, Vinay P Namboodiri, and CV 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*, pages 484–492, 2020. 2

[33] Alexander Richard, Michael Zollhöfer, Yandong Wen, Fernando De la Torre, and Yaser Sheikh. Meshtalk: 3d face animation from speech using cross-modality disentanglement. In *2021 IEEE/CVF International Conference on Computer Vision, ICCV 2021, Montreal, QC, Canada, October 10-17, 2021*, pages 1153–1162. IEEE, 2021. 2

[34] Shuai Shen, Wanhua Li, Zheng Zhu, Yueqi Duan, Jie Zhou, and Jiwen Lu. Learning dynamic facial radiance fields for few-shot talking head synthesis. In *European Conference on Computer Vision*, pages 666–682. Springer, 2022. 2

[35] Balamurugan Thambiraja, Ikhsanul Habibie, Sadegh Aliakbarian, Darren Cosker, Christian Theobalt, and Justus Thies. Imitator: Personalized speech-driven 3d facial animation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 20621–20631, 2023. 2

[36] Justus Thies, Michael Zollhöfer, Marc Stamminger, Christian Theobalt, and Matthias Nießner. Face2face: Real-time face capture and reenactment of rgb videos, 2020. 2

[37] Ashish Verma, Nitendra Rajput, and L Venkata Subramaniam. Using viseme based acoustic models for speech driven lip synthesis. In *2003 IEEE International Conference on Acoustics, Speech, and Signal Processing, 2003. Proceedings.(ICASSP'03).*, volume 5, pages V–720. IEEE, 2003. 2

[38] Konstantinos Vougioukas, Stavros Petridis, and Maja Pantic. Realistic speech-driven facial animation with gans. *International Journal of Computer Vision*, 128:1398–1413, 2020. 2

[39] Kaisiyuan Wang, Qianyi Wu, Linsen Song, Zhuoqian Yang, Wayne Wu, Chen Qian, Ran He, Yu Qiao, and Chen Change Loy. Mead: A large-scale audio-visual dataset for emotional talking-face generation. In *ECCV*, August 2020. 1, 3

[40] Suzhen Wang, Lincheng Li, Yu Ding, and Xin Yu. One-shot talking face generation from single-speaker audio-visual correlation learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 2531–2539, 2022. 2

[41] Erroll Wood, Tadas Baltrusaitis, Charlie Hewitt, Matthew Johnson, Jingjing Shen, Nikola Milosavljevic, Daniel Wilde, Stephan Garbin, Chirag Raman, Jamie Shotton, Toby Sharp, Ivan Stojiljkovic, Tom Cashman, and Julien Valentin. 3d face reconstruction with dense landmarks, 2022. 2, 3

[42] Jinbo Xing, Menghan Xia, Yuechen Zhang, Xiaodong Cun, Jue Wang, and Tien-Tsin Wong. Codetalker: Speech-driven 3d facial animation with discrete motion prior. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2023, Vancouver, BC, Canada, June 17-24, 2023*, pages 12780–12790. IEEE, 2023. 1, 2, 3, 4, 5, 7

[43] Haotian Yang, Hao Zhu, Yanru Wang, Mingkai Huang, Qiu Shen, Ruigang Yang, and Xun Cao. Facescape: a large-scale high quality 3d face dataset and detailed riggable 3d face prediction, 2020. 2, 3

[44] Peiji Yang, Huawei Wei, Yicheng Zhong, and Zhisheng Wang. Semi-supervised speech-driven 3d facial animation via cross-modal encoding. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 21032–21041, 2023. 2, 3

[45] Shu-Wen Yang, Po-Han Chi, Yung-Sung Chuang, Cheng-I Jeff Lai, Kushal Lakhotia, Yist Y. Lin, Andy T. Liu, Jiatong Shi, Xuankai Chang, Guan-Ting Lin, Tzu-Hsien Huang, Wei-Cheng Tseng, Ko-tik Lee, Da-Rong Liu, Zili Huang, Shuyan Dong, Shang-Wen Li, Shinji Watanabe, Abdelrahman Mohamed, and Hung-yi Lee. SUPERB: speech processing universal performance benchmark. *CoRR*, abs/2105.01051, 2021. 4

[46] Xiangyu Zhu, Xiaoming Liu, Zhen Lei, and Stan Z. Li. Face alignment in full pose range: A 3d total solution. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 41(1):78–92, jan 2019. 2

[47] Wojciech Zielonka, Timo Bolkart, and Justus Thies. Towards metrical reconstruction of human faces, 2022. 2

A. Hyper-Parameters

The audio encoder weights are initialized with the pre-trained Wav2Vec 2.0 [2] weights. The audio encoder itself is a 12-layer transformer encoder with model dimensionality of 768. A linear project layer sits on top of the audio encoder, converting the 768-dimensional audio output to a 64-dimensional speech representation.

The vertex encoder and style-embedder are fully-connected layers with 64 outputs. The transformer decoder is a 1 or 6 layer decoder for FaceFormer or CodeTalker, respectively. For both CodeTalker and FaceFormer, the biased causal and cross-modal multi-headed self-attention are 4-headed with dimensionality of 64. A fully-connected layer with 15069 or 1024 outputs is used encoder outputs for FaceFormer or CodeTalker, respectively. CodeTalker's outputs map to the codebook, hence the difference output sizes.

We replicate the experimental settings proposed in FaceFormer [15], using the Adam [25] optimizer with a learning rate $\eta = 1 \times 10^{-4}$, batch size of 1, and period set to 30. Each model is trained for 100 epochs and tested using the final weights after 100.

B. Misalignment in 3D-MEAD

VOCASET is built using expensive motion capture equipment that can accurately generate neutral template meshes, which in turn are used during training to get the vertex offsets of the animations, which the model learns to predict. The issue with 3D-MEAD is that neutral templates do not exist, and nor is there video or images from which to generate a neutral mesh, so we are left with trying to generate neutral template meshes by disentangling the expressions from the animated sequences and approximating a neutral mesh. This approximate prediction of neutral meshes, generated from animated videos, is not perfect, and any inaccuracies in that prediction will trickle down when subtracting the template to generate vertex offsets.

Figure 5 demonstrates the issue well, where parts of the mesh appear pinched or raised, particularly around the eyebrows. This pinching effect arises from inaccurate neutral mesh template predictions that skew the vertex offset predictions when trained on 3D-MEAD. Similarly, it is not trivial to simply combine these datasets, as this misalignment will undoubtedly confuse the model, and make it very hard for the model to fit to the data well. Fixing this misalignment between neutral meshes in generated datasets is a challenging problem, which we leave as future work.

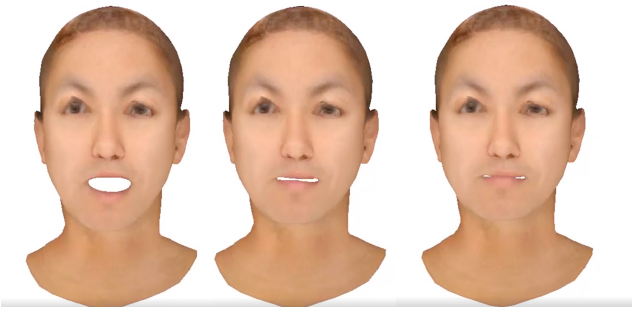


Figure 5. The misalignment of 3D-MEAD templates, with FaceFormer trained on 3D-MEAD and zero-shot to VOCASET.