LipSync3D: Data-Efficient Learning of Personalized 3D Talking Faces from Video using Pose and Lighting Normalization

Avisek Lahiri[1,2*†] Vivek Kwatra[1*] Christian Frueh[1*] John Lewis[1] Chris Bregler[1]
[1]Google Research [2]Indian Institute of Technology Kharagpur
{avisek,kwatra,frueh,jplewis,bregler}@google.com

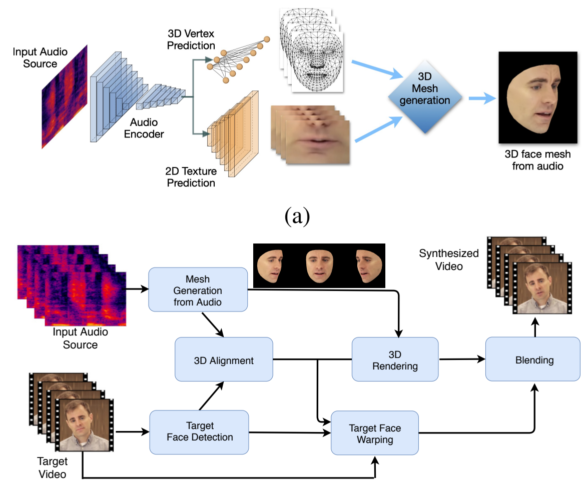
Abstract

In this paper, we present a video-based learning framework for animating personalized 3D talking faces from audio. We introduce two training-time data normalizations that significantly improve data sample efficiency. First, we isolate and represent faces in a normalized space that decouples 3D geometry, head pose, and texture. This decomposes the prediction problem into regressions over the 3D face shape and the corresponding 2D texture atlas. Second, we leverage facial symmetry and approximate albedo constancy of skin to isolate and remove spatio-temporal lighting variations. Together, these normalizations allow simple networks to generate high fidelity lip-sync videos under novel ambient illumination while training with just a single speaker-specific video. Further, to stabilize temporal dynamics, we introduce an auto-regressive approach that conditions the model on its previous visual state. Human ratings and objective metrics demonstrate that our method outperforms contemporary state-of-the-art audio-driven video reenactment benchmarks in terms of realism, lip-sync and visual quality scores. We illustrate several applications enabled by our framework.

1. Introduction

"Talking head" videos, consisting of closeups of a talking person, are widely used in newscasting, video blogs, online courses, etc. Other applications that feature talking faces prominently are face-to-face live chat, 3D avatars and animated characters in games and movies. We present a deep learning approach to synthesize 3D talking faces (both photorealistic and animated) driven by an audio speech signal. We use speaker-specific videos to train our model in a data-efficient manner by employing 3D facial tracking. The resulting system has multiple applications, including video editing, lip-sync for dubbing of videos in a new language, personalized 3D talking avatars in gaming, VR and CGI, as well as compression in multimedia communication.

The importance of talking head synthesis has led to a variety of methods in the research literature. Many recent techniques [6, 7, 40, 43, 28, 30] use the approach of regressing facial motion from audio, employing it to deform one or more reference images of the subject. These approaches can inherit the realism of the reference photos, however, the results do not accurately reproduce 3D facial articulation and appearance under general viewpoint and lighting variations. Another body of research predicts 3D facial meshes from audio [38, 13, 19, 11]. These approaches are directly suitable for VR and gaming applications. However, visual realism is often restricted by the quality of texturing. Some recent approaches [32, 33, 14] attempt to bridge the gap by combining 3D prediction with high-quality rendering, but are only able to edit fixed target videos that they train on.



(a)

(b)

Figure 1: Flow diagram of our approach to (a) generate a dynamically textured 3D face mesh from audio, and (b) insert the generated face mesh into a target video to create a synthesized talking head video from new audio input.

Our work encompasses several of the scenarios mentioned above. We can use 3D information to edit 2D video, including novel videos of the same speaker not seen during training. We can also drive a 3D mesh from audio or text-to-speech (TTS), and synthesize animated characters by predicting face blendshapes. Next, we highlight some of our key design choices.

Personalized models: We train personalized speaker-specific models, instead of building a single universal model to be applied across different people. While universal mod-

*Equal contribution
†Work done while an intern at Google

els like Wav2Lip [30] are easier to reuse for novel speakers, they need large datasets for training and do not adequately capture person-specific idiosyncrasies [5]. Personalized models like ours and NVP [33] produce results with higher visual fidelity, more suitable for editing long speaker-specific videos. Additionally, our model can be trained entirely using a single video of the speaker.

3D pose normalization: We use a 3D face detector [20] to obtain the pose and 3D landmarks of the speaker's face in the video. This information allows us to decompose the face into a normalized 3D mesh and texture atlas, thus decoupling head pose from speech-induced face deformations, *e.g*. lip motion and teeth/tongue appearance.

Lighting normalization: We design a *novel* algorithm for removing spatial and temporal lighting variations from the 3D decomposition of the face by exploiting traits such as facial symmetry and albedo constancy of the skin. This lighting normalization removes another confounding factor that can otherwise affect the speech-to-lips mapping.

Data-efficient learning: Our model employs an encoder-decoder architecture that computes embeddings from audio spectrograms, and decodes them to predict the decomposed 3D geometry and texture. Pose and lighting normalization allows us to train this model in a data-efficient manner. The model complexity is greatly reduced, since the network is not forced to disentangle unrelated head pose and lighting changes from speech, allowing it to synthesize high quality lip-sync results even from short training videos (2-5 minutes long). Lighting normalization allows training and inference illumination to be different, which obviates the need to train under multiple lighting scenarios. The model predicts *3D talking faces* instead of just a 2D image, even though it learns just from video, broadening its applicability. Finally, pose and lighting normalization can be applied in a backward fashion to align and match the appearance of the synthesized face with novel target videos. See Figure 1 for an overview of our approach.

Our **key technical contributions** are:

- A method to convert arbitrary talking head video footage into a normalized space that decouples 3D pose, geometry, texture, and lighting, thereby enabling data-efficient learning and versatile high-quality lip-sync synthesis for video and 3D applications.
- A novel algorithm for normalizing facial lighting in video that exploits 3D decomposition and face-specific traits such as symmetry and skin albedo constancy.
- To our best knowledge, this is the first attempt at disentangling pose and lighting from speech via data pre-normalization for personalized models.
- An easy-to-train auto-regressive texture prediction model for temporally smooth video synthesis.
- Human ratings and objective metrics suggest that our method outperforms contemporary audio-driven video reenactment baselines in terms of realism, lip-sync and visual quality scores.

2. Related Work

Audio-driven 3D Mesh Animation: These methods generate 3D face models driven by input audio or text, but do not necessarily aim for photorealism. In [38], the authors learn a Hidden Markov Model (HMM) to map Mel-frequency Cepstral Coefficients (MFCC) to PCA model parameters. Audio features are mapped to Jali [13] coefficients in [44]. In [19], the authors learn to regress to 3D vertices of a face model conditioned on input audio spectrograms and simultaneously disambiguate variations in facial expressions unexplained by audio. In [18], the authors learn to regress blendshapes of a 3D face using the combined audio-visual embedding from a deep network. VOCA [12] pre-registers subject-specific 3D mesh models using FLAME [26] and then learns (using hours of high quality 4D scans) an offset to that template based on incoming speech, represented with DeepSpeech [15] features.

Audio-driven Video Synthesis: These methods aim to generate visually plausible 2D talking head videos, conditioned on novel audio. In [6], an audio-visual correlation loss is used to match lip shapes to speech, while maintaining the identity of the target face. In [7], a two-stage cascaded network is used to first predict 2D facial landmarks from audio, followed by target frame editing conditioned upon these landmarks. In [36], the authors leverage a temporal GAN for synthesizing video conditioned on audio and a reference frame. They improve it further in [37] via a specialized lip-sync discriminator. In contrast to our approach, the above methods fail to produce full-frame outputs; instead they generate normalized cropped faces, whose lips are animated based on input audio and a reference frame.

Among efforts on full-frame synthesis, Video Rewrite [5] was a pioneering work. It represented speech with phonetic labels and used exemplar-based warping for mouth animation. Speech2Vid [8] learns a joint embedding space for representing audio features and the target frame, and uses a shared decoder to transform the embedding into a synthesized frame. X2Face [40] learns to drive a target frame with the head pose and expression of another source video, and it can optionally be also driven by an audio to animate a target frame. A framework to translate an input speech to another language and then modify the original video to match it is presented in [23]. Recently, Wav2Lip [30] reported appreciable lip-sync performance by using a powerful offline lip-sync discriminator [9] as an expert to train their generator. While currently this is one of the best universal models, it lacks

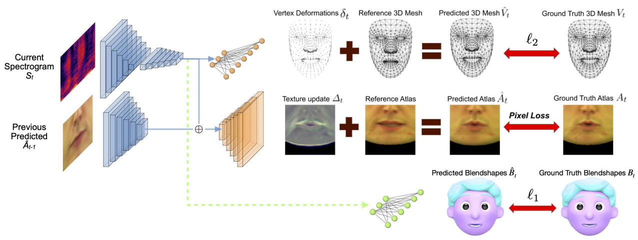


Figure 2: Joint prediction pipeline: geometry and texture models have dedicated decoders but share the audio encoder. The texture model also depends on the previously predicted atlas. Optionally, the audio embedding can drive a 3D CGI character via a blendshape coefficients decoder. Please enlarge to see details.

the visual fidelity of speaker-specific models.

Some recent works [32, 33, 31] have focused on 3D model guided video synthesis. In [32] an RNN regresses audio to mouth shape, producing convincing results on President Obama. The approach required very extensive training data however (17 hours). In [33], the DeepSpeech RNN is used to map input speech to audio expression units which then drive a blendshapes-based 3D face model. Finally, a neural renderer [22] is used to render the face model with the audio expressions. Since neural renderer training depends on target illumination, the methods leveraging such rendering [31, 33] suffer from the need for retraining if inference-time lighting conditions change. On the contrary, our method seamlessly adapts to novel lighting.

Text-based Video Editing: In [14], the authors present a framework for text based editing of videos (TBE). They first align written transcripts to audio and track each frame to create a face model. During edit operations, a (slow) viseme search is done to find best matching part of training video. This method needs a time-aligned transcript and around one hour of recorded data, and is mostly suitable for for small edits. Our method, on the other hand, relies on just the audio signal and can synthesize videos of unrestricted length.

Actor-driven Video Synthesis: [34, 22] present techniques for generating and dubbing talking head videos by transferring facial features, such as landmarks or blendshape parameters, from a different actor's video. These techniques generate impressive results, however they require a video of a surrogate actor to drive synthesis. We emphasize that our approach uses only audio or text-to-speech (TTS) as the driving input, and does not require any actors for dubbing. It is therefore fundamentally different from these methods.

3. Method

We now describe the various components of our approach including data extraction and normalization, neural network architecture and training, and finally, inference and synthesis. Figure 2 shows an overview of our model.

We extract the audio channel from the training video and transform it into frequency-domain spectrograms. These

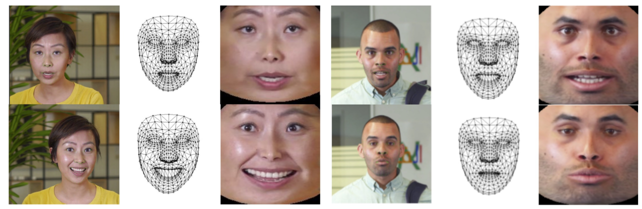


Figure 3: Pose normalization of training data. For each subject– Left: input frames with detected features (see zoomed in); Middle: normalized vertices and triangle mesh; Right: texture atlas which acts as ground truth for texture prediction.

spectrograms are computed using Short-time Fourier transforms (STFT) with a Hann window function [39], over 30ms wide sliding windows that are 10ms apart. We align these STFTs with video frames and stack them across time to create a 256×24 complex spectrogram image, spanning 240ms centered around each video frame. Our model predicts the face geometry, texture, and optionally, blendshape coefficients, for each frame based on the audio spectrogram.

The face in the video is tracked using a 3D face landmark detector [20], resulting in 468 facial features, with the depth (z-component) predicted using a deep neural network. We refer to these features as vertices, which are accompanied by a predefined triangulated face mesh with fixed topology.

3.1. Normalizing Training Data

We preprocess the training data to eliminate the effects of head movement and lighting variations, and work with normalized facial geometry and texture. Both training and inference take place in this normalized space.

3.1.1 Pose normalization

For pose normalization, we first select one frame of the input video as a reference frame, and its respective 3D face feature points as reference vertices. The choice of frame is not critical; any frame where the face is sufficiently frontal is suitable. Using the reference vertices, we define a reference cylindrical coordinate system (similar to [4]) with a vertical axis such that most face vertices are equidistant to the axis. We then scale the face size such that the eyes and nose project to fixed locations on this reference cylinder.

Next, for each frame of the training video, we stabilize the rigid head motion (see [3, 24]) to provide a registered 3D mesh suitable for training our geometry model. Specifically, we approximately align the vertices of the upper, more rigid parts of the face with corresponding vertices in the normalized reference using Umeyama's algorithm [35] and apply the estimated rotation $\mathbf{R}$, translation $\mathbf{t}$ and scale c to all tracked vertices $\mathbf{v}$ as $\hat{\mathbf{r}} = c\mathbf{R}\mathbf{v} + \mathbf{t}$.

We use these normalized vertices, along with the cylindrical mapping defined above, to create a pose-invariant, *frontalized* projection of the face texture for each video frame (including the reference frame). Mapping the face vertices to the reference cylinder creates a set of 2D texture

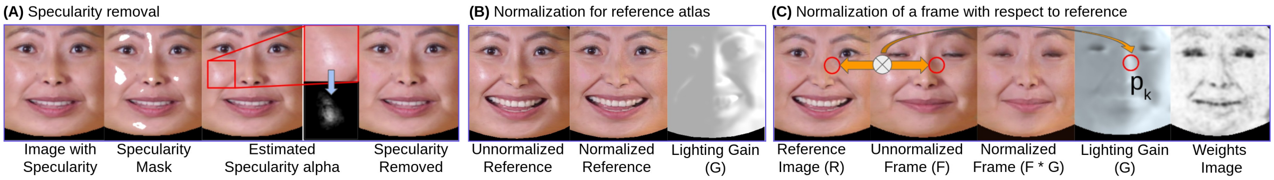


Figure 4: Steps of our proposed lighting normalization during training. **(A:)** First step is to specularity removal from an input frame. **(B:)** Second step is self normalization of the reference atlas. **(C:)** Finally, any given training frame is normalized with respect to the pre-normalized reference atlas of step B.

coordinates for the face's surface, which are used to *unroll* its texture. We warp the triangles associated with these coordinates from the source frame onto the texture domain, resulting in a 256×256 *texture atlas* that resembles a frontal view of the face, but with the non-rigid features like the lips and mouth moving with the speech. Figure 3 demonstrates the effect of normalization; the head pose is removed, but the moving lip shapes and mouth interior are preserved.

3.1.2 Lighting normalization

We normalize the frontalized texture atlas to remove lighting variations, which are mostly caused by head motion or changing illumination. Our lighting normalization algorithm works in two phases. It first exploits facial symmetry to normalize the reference atlas R *spatially*, removing specularities and lighting variations that run across the face. It then performs a *temporal* normalization across video frames that transforms each frame's atlas F to match the illumination of R. The resulting atlases have a more uniform *albedo*-like appearance, that stays consistent across frames.

We first describe the temporal normalization algorithm, as it is a core component also used during spatial normalization. This algorithm assumes that the two textures F and R are pre-aligned geometrically. However, any non-rigid facial movements, *e.g*. from speech, can result in different texture coordinates, and consequently, misalignments between R and F. Hence, we first warp R to align it with F's texture coordinates, employing the same triangle-based warping algorithm used for frontalization.

Given the aligned R and F, we estimate a mapping that transforms F to match the illumination of R. This mapping is composed of a smooth multiplicative pixel-wise gain G in the luminance domain, followed by a global channel-wise gain and bias mapping $\{a, b\}$ in the RGB domain. The resulting normalized texture F^n is obtained via the following steps: (1) $(F_y, F_u, F_v) = \mathbf{RGBtoYUV}(F)$; (2) $F_y^l = G * F_y$; (3) $F^l = \mathbf{YUVtoRGB}(F_y^l, F_u, F_v)$; (4) $F^n = aF^l + b$.

Gain Estimation: To estimate the gain G, we observe that a pair of corresponding pixels at the same location k in F and R should have the same underlying appearance, modulo any change in illumination, since they are in geometric alignment (see Figure 4(C)). This *albedo constancy* assumption, if perfectly satisfied, yields the gain at pixel k as $G_k = R_k/F_k$. However, we note that (a) G is a smoothly varying illumination map, and (b) albedo constancy may be occasionally violated, *e.g*. in non-skin pixels like the mouth, eyes and nostrils, or where the skin deforms sharply, *e.g*. the nasolabial folds. We account for these factors by, firstly, estimating G_k over a larger patch p_k centered around k, and secondly, employing a robust estimator that weights pixels based on how well they satisfy albedo constancy. We formulate estimating G_k as minimizing the error:

$$\mathbf{E}_k = \sum_{j \in p_k} W_j \| R_j - G_k * F_j \|^2, \tag{1}$$

where W is the per-pixel weights image, and solve it using iteratively reweighted least squares (IRLS). In particular, we initialize the weights uniformly, and then update them after each (i^{th}) iteration as:

$$W_k^{i+1} = \exp\left(\frac{-\mathbf{E}_k^i}{T}\right), \tag{2}$$

where T is a temperature parameter. The weights and gain converge in 5-10 iterations; we use $T = 0.1$ and a patch size of 16×16 pixels for 256×256 atlases. Figure 4(C) shows example weights and gain images. Pixels with large error $\mathbf{E}_k$ get low weights, and implicitly interpolate their gain values from neighboring pixels with higher weights.

To estimate the global color transform $\{a, b\}$ in closed form, we minimize $\sum_k W_k \| R_k - aF_k - b\|^2$ over all pixels, with W_k now fixed to the weights estimated above.

Reference Atlas Normalization using Facial Symmetry: We first estimate the gain G^m between the reference R and its mirror image R', using the algorithm described above. This gain represents the illumination change between the left and right half of the face. To obtain a reference with uniform illumination, we compute the symmetrized gain $G^s = \max(G^m, G^{m'})$, where $G^{m'}$ is the mirror image of G^m, *i.e*. for every symmetric pair of pixels, we make the darker pixel match the brighter one. The normalized reference is then $R^n = G^s * R$, as shown in Figure 4(B). Note that our weighting scheme makes the method robust to inherent asymmetries on the face, since any inconsistent pixel pairs will be down-weighted during gain estimation, thereby preserving those asymmetries.

Specularity Removal: We remove specularities from the face before normalizing the reference and video frames, since they are not properly modeled as a multiplicative gain, and also lead to duplicate specularities on the reference due to symmetrization. We model specular image formation as:

$$I = \alpha + (1 - \alpha) * I_c, \tag{3}$$

where I is the observed image, α is the specular alpha map and I_c is the underlying *clean* image without specularities. We first compute a mask, where $\alpha > 0$, as pixels whose minimum value across RGB channels in a smoothed I exceeds the 90^{th} percentile intensity across all skin pixels in I. The face mesh topology is used to identify and restrict computation to skin pixels. We then estimate a *pseudo* clean image $\tilde{I}_c$ by hole-filling the masked pixels from neighboring pixels, and use it to estimate $\alpha = (I - \tilde{I}_c)/(1 - \tilde{I}_c)$. The final clean image is then $I_c = (I - \alpha)/(1 - \alpha)$. Note that our soft alpha computation elegantly handles any erroneous over-estimation of the specularity mask (see Figure 4(A)). The above method is specifically tailored for stabilized face textures and is simple and effective, thus we do not require more generalized specularity removal techniques [42].

3.2. Joint Prediction Model and Training Pipeline

In this section we describe the framework for learning a function $\mathbf{F}$ to jointly map from domain S of audio spectrograms to the domains V of *vertices* and A of texture *atlases*: $\mathbf{F} : S \rightarrow V \times A$, with $V \in \mathbb{R}^{468 \times 3}$ and $A \in \mathbb{R}^{128 \times 128 \times 3}$, where for the purpose of prediction, we crop the texture atlas to a 128×128 region around the lips, and only predict these cropped regions. The texture for the upper face is copied over from the reference, or target video frames, depending upon the application. We follow an encoder-decoder architecture for realizing $\mathbf{F}(\cdot)$, as shown in Figure 2. It consists of a shared encoder for audio, but separate dedicated decoders for geometry and texture. However, the entire model is trained jointly, end-to-end.

Audio encoder: The input at time instant t is a complex spectrogram, $S_t \in \mathbb{R}^{256 \times 24 \times 2}$. Our audio encoder — and face geometry prediction model — is inspired by the one proposed in [19], in which the vertex positions of a fixed-topology face mesh are also modified according to an audio input. However, while [19] used formant preprocessing and autocorrelation layers as input, we directly use complex spectrograms S_t. Each S_t tensor is passed through a 12 layer deep encoder network, where the first 6 layers apply 1D convolutions over frequencies (kernel 3×1, stride 2×1), and the subsequent 6 layers apply 1D convolution over time (kernel 1×3, stride 1×2), all with leaky ReLU activation, intuitively corresponding to phoneme detection and activation, respectively. This yields a latent code $L_t^s \in \mathbb{R}^{N_s}$.

Geometry decoder: This decoder maps the latent audio code L_t^s to vertex *deformations* δ_t, which are added to the reference vertices V_r to obtain the predicted mesh $\hat{V}_t = V_r + \delta_t$. It consists of two fully connected layers with 150 and 1404 units, and linear activations, with a dropout layer in the middle. The resulting output is 468 vertices ($1404 = 468 \times 3$ coordinates). As proposed in [19], we initialize the last layer using PCA over the vertex training data. Further, we impose ℓ_2 loss on the vertex positions: $\mathbf{R_{geo}} = \|V_t - \hat{V}_t\|_2$, where V_t are ground-truth vertices.

Texture decoder: This decoder maps the audio code L_t^s to a texture atlas *update* (difference map) Δ_t which is added to the reference atlas A_r to obtain the predicted atlas, $\hat{A}_t = A_r + \Delta_t$. It consists of a fully connected layer to distribute the latent code spatially, followed by progressive up-sampling using convolutional and interpolation layers to generate the 128×128 texture update image (see supplementary material). We impose an image similarity loss between the predicted and ground-truth atlas A_t: $\mathbf{R_{tex}} = d(A_t, \hat{A}_t)$, where d is a visual distance measure. We tried different variants of $d(\cdot)$ including the ℓ_1 loss, Structural Similarity Loss (SSIM), and Gradient Difference Loss (GDL) [27] and found SSIM to perform the best.

Blendshapes decoder: To animate CGI characters using audio, we optionally add another decoder to our network that predicts *blendshape* coefficients B_t in addition to geometry and texture. For training, these blendshapes are derived from vertices V_t by fitting them to an existing blendshapes basis either via optimization or using a pre-trained model [25]. We use a single fully connected layer to predict coefficients $\hat{B}_t$ from audio code L_t^s, and train it using ℓ_1 loss $\mathbf{R_{bs}} = \|B_t - \hat{B}_t\|_1$ to encourage sparse coefficients.

3.2.1 Auto-regressive (AR) Texture Synthesis:

Ambiguities in facial expressions while speaking (or silent) can result in temporal jitters. We mitigate these by incorporating memory into the network. Rather than using RNNs, we condition the current output of the network (A_t) not only on S_t but also on the previous predicted atlas $\hat{A}_{t-1}$, encoding it as a latent code vector $L_{t-1}^a \in \mathbb{R}^{N_a}$. L_t^s and L_{t-1}^a are combined and passed to the texture decoder to generate the current texture $\hat{A}_t$ (Figure 2). This appreciably improves the temporal consistency of synthesized results. We can train this AR network satisfactorily via *Teacher Forcing* [41], using previous ground truth atlases. The resulting network $\mathbf{F}$ is trained end-to-end, minimizing the combined loss $\mathbf{R} = \mathbf{R_{tex}} + \alpha_1 \mathbf{R_{geo}} + \alpha_2 \mathbf{R_{bs}}$, where $\alpha_1 = 3.0$ and $\alpha_2 = 0.3$ (when enabled). We used hyperparameter search to determine the latent code lengths, $N_s = 32$ and $N_a = 2$.

3.3. Inference and Synthesis

Textured 3D mesh: During inference, our model predicts geometry and texture from audio input. To convert it to a textured 3D mesh, we project the predicted vertices onto the reference cylinder, and use the resulting 2D locations as texture coordinates. Since our predicted texture atlas is defined on the same cylindrical domain, it is consistent with the computed texture coordinates. The result is a fully textured 3D face mesh, driven by audio input (Figure 1a).

Talking head video synthesis: The pose and lighting normalization transforms (Section 3.1) are invertible, *i.e.* one can render the synthesized face mesh in a different pose

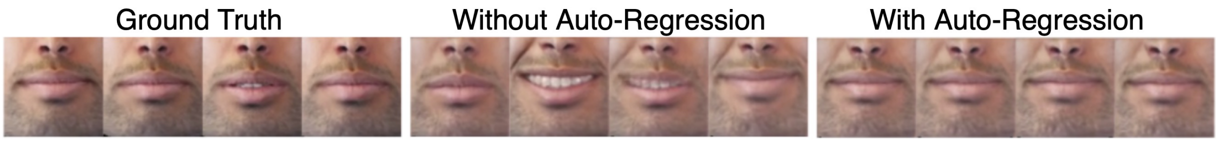


Figure 5: Benefits of proposed auto-regressive (AR) prediction. Left: Four consecutive frames when the subject was silent. Middle: Prediction without AR. Right: Prediction with AR. In absence of AR, the model fluctuates between different visual states, while the AR substantially improves temporal stability.

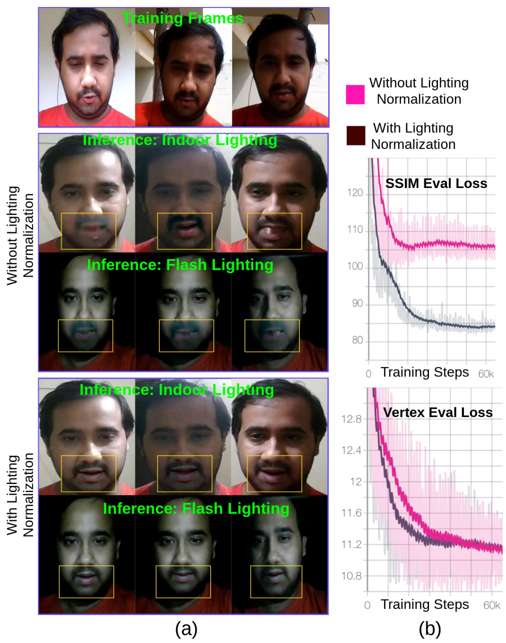


Figure 6: **(a:)** Benefits of the proposed lighting normalization. Top row shows representative training frames in a sunny outdoor setting while we conduct inference under two novel lighting settings which have not been used in training. Note that the proposed lighting normalization enables realistic synthesis under new lighting while absence of lighting normalization yields degraded outputs. **(b:)** Plot of SSIM loss (texture prediction) and vertex loss (geometry prediction) on the evaluation set. Even though both models result in similar lip shapes, the lower SSIM loss of the lighting-normalized model boosts the visual realism and overall lip-sync quality.

under novel lighting, which allows us to procedurally blend it back into a different target video (Figure 1b). Specifically, we warp the textured face mesh to align it with the target face, then apply our lighting normalization algorithm in reverse, *i.e.* on the warped texture, using the target face as reference. One caveat is that the target frame's area below the chin may not align with the warped synthesized face, due to inconsistent non-rigid deformations of the jaw. Hence, we pre-process each target frame by warping the area below the original chin to match the expected new chin position. To avoid seams at border areas, we gradually blend between the original and new face geometry, and warp the original face in the target frame according to the blended geometry.

Cartoon rendering: For stylized visualizations, we can create a cartoon rendering of the textured mesh (or video), by combining bilateral filtering with a line drawing of the facial features. In particular, we identify nose, lips, cheeks and chin contours in the synthesized face mesh, and draw them prominently over the filtered texture or video frame (see supplementary material).

CGI Characters: Models trained with the blendshapes decoder also output blendshape coefficients that can drive a CGI character. We combine these predicted blendshapes (that generally affect the lips and mouth) with other blendshapes, such as those controlling head motion and eye gaze, to create lively real-time animations.

4. Experiments

Our training and inference pipelines were implemented in Tensorflow [1], Python and C++. We trained our models with batch sizes of 128 frames, for 500-1000 epochs, with each epoch spanning the entire training video. Sample training times were between 3-5 hours, depending on video length (usually 2-5min). Average inference times were 3.5ms for vertices, 31ms for texture and 2ms for blendshapes, as measured on a GeForce GTX 1080 GPU. Our research-quality code for blending into target videos takes 50-150ms per frame, depending on the output resolution.

4.1. Ablation Studies

Benefit of Auto-Regressive Prediction: The auto-regressive texture prediction algorithm stabilizes mouth dynamics considerably. In Figure 5, we show that without auto-regression, the model can produce an unrealistic jittering effect, especially during silent periods.

Benefit of Lighting Normalization: We use a short training video (~4 minutes) recorded in an outdoor setting but with varying illumination. However, during inference, we select two novel environments: a) indoor lighting with continuous change of lighting direction, and b) a dark room with a face illuminated by a moving flash light. Some representative frames of models trained with and without lighting normalization are shown in Figure 6(a). Without lighting normalization, the model produces disturbing artifacts around the lip region, exacerbated by the extreme changes in illumination. However, with normalized lighting, the model adapts to widely varying novel illumination conditions. This ability to edit novel videos of the same speaker *on-the-fly* without needing to retrain for new target illumination is a significant benefit. In contrast, neural rendering based approaches [33] require retraining on each new video, because they map 3D face models directly to the facial texture in video without disentangling illumination.

We also visualize the loss curves on held out evaluation sets in Figure 6(b). With lighting normalization, the SSIM loss (used for texture generation) saturates at a much lower

Figure 7: Qualitative comparison on subjects from GRID, CREMA-D and TCD-TIMIT against IJCV'19 and CVPR'19 (latter only available on GRID). Our model is capable of seamlessly blending back into the video instead of animating a normalized cropped frame as in IJCV'19 and CVPR'19..

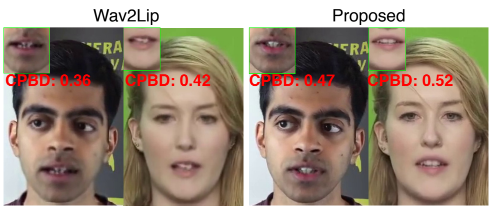


Figure 8: Comparison with Wav2Lip [30]. Our model generates higher resolution outputs (evident by higher CPBD metric [29]) with fewer artifacts compared to Wav2Lip. Examples are provided in accompanying video.

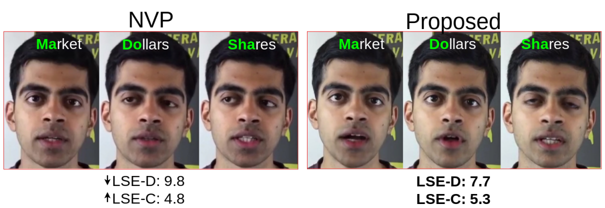


Figure 9: Comparison with NVP [33]. We show that a sequence generated by our method usually has better lip dynamics compared to NVP. The observation is also supported by LSE-D (lower is better) and LSE-C (higher is better) metrics [30] for our model. Examples are provided in accompanying video. Best viewed **zoomed** in.

value than without normalization. This supports our hypothesis that lighting normalization results in more data-efficient learning, since it achieves a better loss with the same amount of training data. The vertex loss (responsible for lip dynamics) is similar for both models, because lighting normalization does not directly affect the geometry decoder, but overall lip-sync and visual quality are improved.

4.2. Comparison: Self-reenactment

We objectively evaluate our model under the self-reenactment setting (audio same as target video), since it allows us to have access to ground truth facial information. We show experiments with three talking head datasets: GRID [10], TCD-TIMIT [16] and CREMA-D [21].

Comparing Methods: We perform quantitative comparisons against state-of-the-art methods whose models/results are publicly available: CVPR'19 [7], IJCV'19 [37], CVPR-W [36]. It is difficult to do an apples-to-apples comparison, since we use personalized models while other techniques use a universal model. However, we minimize this gap by testing on the same 10 subjects (details in supplementary) from each of the 3 datasets used in IJCV'19 and CVPR'19, and employing the official evaluation frameworks of these papers. We also compare against other prior methods, but reuse the results already reported by CVPR'19 or IJCV'19.

Evaluation Metrics: We follow the trend in recent papers [7, 37, 36], which use **SSIM** (Structural Similarity Index) as a reconstruction metric, **LMD** (Landmark Distance) on mouth features as a shape similarity metric, **CPBD** (Cumulative Probability Blur Detection) [29] as a sharpness metric and **WER** (word error rate) as a content metric to evaluate the correctness of words from reconstructed videos. Following [37], we use a LipNet model [2] pre-trained for lip-reading on GRID dataset [10].

Observations: We report the metrics in Figure 10 (left). On LMD and WER, which capture lip-sync, our model is significantly better than any competing method. Also, in terms of reconstruction measures (SSIM, CPBD), our model almost always performs better. CVPR-W and IJCV'19 have a better (though comparable) CPBD on GRID, but it is a low-resolution dataset. On higher resolution TCD-TIMIT and CREMA-D, our CPBD is the best. We also show qualitative comparisons in Figure 7. Note that we synthesize full frame videos, while CVPR'19 and IJCV'19 only generated normalized face crops at a resolution of 128×128, and 96×128 respectively. Thus our method is more suitable for practical video applications.

4.3. Comparison: Audio-Driven Video Dubbing

In this section we focus on *'audio-driven'* video dubbing where the driving audio is different from the target video.

User Study: We conducted a user study to quantitatively compare our lip-sync and perceptual quality against the state-of-the-art audio-driven frameworks of Wav2Lip, NVP, IJCV'19 and TBE. In the study, 35 raters were each shown 29 sample clips consisting of synthetic and real videos. For competing methods, we used their released videos or generated results with their pre-trained models. The raters were asked three questions: Q1) Is the video real or fake? Q2) Rate lip-sync quality on a 3-point discrete scale. Q3) Rate visual quality on a 5-point discrete scale. We report the Mean Opinion Scores (MOS) of the questions in Figure 10 (right). As is evident, among the competing methods our method receives the most favorable user ratings.

Comparison with Wav2Lip [30]: Unlike other image-based methods, Wav2Lip can paste back the generated face on background video. However, compared to our model, the outputs from Wav2Lip are of low resolution. Also, at

Self-reenactment Metrics

Methods	GRID				TCD-TIMIT			CREMA-D		
	SSIM(↑)	LMD(↓)	CPBD(↑)	WER(↓)	SSIM	LMD	CPBD	SSIM	LMD	CPBD
CVPR-W'19	0.84	-	**0.27**	25%	0.69	-	0.25	-	-	-
IJCV'19	0.81	1.32	0.26	23%	0.73	1.81	0.30	0.66	1.70	0.21
CVPR'19	0.81	1.37	0.17	70%	—	—	—	—	—	—
BMVC'17	0.72	—	0.25	58%	0.65	—	0.21	0.70	—	0.21
Chen *et al.* (ECCV'18)	0.73	1.73	—	—	—	—	—	—	—	—
Wiles *et al.* (ECCV'18)	0.75	1.60	—	—	—	—	—	—	—	—
Proposed	**0.94**	**0.80**	0.25	**18%**	**0.91**	**1.57**	**0.40**	**0.91**	**1.33**	**0.26**

MOS: Audio Driven Video Dubbing

Method	Is Real ? (% Yes)	Lip-Sync (1-3))	Visual Quality (1-5)
Real	97.6	2.95±0.03	4.55±0.13
IJCV'19	60.7	2.45±0.11	2.49±0.17
Wav2Lip	34.4	1.72±0.12	3.35±0.20
NVP	44.4	1.80±0.13	3.75±0.19
TBE	50.7	2.17±0.17	4.07±0.26
Proposed	**71.6**	**2.46±0.10**	**4.10±0.15**

Figure 10: **Left:** Self-reenactment performance comparison against state-of-the-art benchmarks of CVPR-W'19 [36], IJCV'19 [37], CVPR'19 [7], BMVC'17 [8], Chen *et al.* [6] and Wiles *et al.* [40]. Pre-trained LipNet (for WER) is available only on GRID. Authors of [7] released checkpoint for GRID only. (↑):Higher is better. (↓):Lower is better. Best results are marked in bold. **Right:** Mean Opinion Scores of user study. The statistical significance of these differences in ratings is confirmed by ANOVA with Tukey post-hoc tests. Please see the supplementary material for details.

high resolution, Wav2Lip produces significant visual artifacts (see Figure 8) and lip-sync starts to degrade.

Comparison with NVP [33]: The lip-sync and dynamics of our model are generally better than NVP. The lip movements of NVP are clearly muted compared to our model, as seen in representative frames in Figure 9(a).

5. Applications

Speech/Text-to-Video: We can create or edit talking head videos for education, advertisement, and entertainment by simply providing new audio transcripts. **"Actor-free" video translation:** while *'actor-driven'* video translation techniques [22, 34] generally require a professional actor to record the entire translated audio and video, our *'actor-free'* approach does not need video, and can be driven by either recorded audio, TTS, or voice cloning [17]. **Voice controlled Avatars:** Our model's blendshapes output can be used to animate CGI characters in real-time, allowing low-bandwidth voice-driven avatars for chat, VR, and games without the need for auxiliary cameras. **Assistive technologies:** Voice-driven 3D faces can support accessibility and educational applications, *e.g.* personified assistants and cartoon animations for visualizing pronunciation.

6. Limitations and Conclusion

Facial expressions: We do not explicitly handle facial expressions, though our model may implicitly capture correlations between expressions and emotion in the audio track. **Strong movements in the target video:** When synthesized faces are blended back into a target video, emphatic hand or head movement might seem out of place. This has not proved to be a problem in our experiments. **Processing speed:** Our research-quality code, running at highest quality, is slightly slower than real-time.

We have presented a data efficient yet robust end-to-end system for synthesizing personalized 3D talking faces, with applications in video creation and editing, 3D gaming and CGI. Our proposed pose and lighting normalization decouples non-essential factors such as head pose and illumination from speech and enables training our model on a relatively short video of a single person while nevertheless generating high quality lip-sync videos under novel ambient lighting. We envision that our framework is a promising stepping stone towards personalized audio-visual avatars and AI-assisted video content creation.

7. Ethical Considerations

Our technology focuses on world-positive use cases and applications. Video translation and dubbing have a variety of beneficial and impactful uses, including making educational lectures, video-blogs, public discourse, and entertainment media accessible to people speaking different languages, and creating personable virtual "assistants" that interact with humans more naturally.

However, we acknowledge the potential for misuse, especially since audiovisual media are often treated as veracious information. We strongly believe that the development of such generative models by *good actors* is crucial for enabling preemptive research on fake content detection and forensics, which would allow them to make early advances and stay ahead of actual malicious attacks. Approaches like ours can also be used to generate counterfactuals for training provenance and digital watermarking techniques.

We also emphasize the importance of acting responsibly and taking ownership of synthesized content. To end that, we strive to take special care when sharing videos or other material that have been synthesized or modified using these techniques, by clearly indicating the nature and intent of the edits. Finally, we also believe it is imperative to obtain consent from all performers whose videos are being modified, and be thoughtful and ethical about the content being generated. We follow these guiding principles in our work.

8. Acknowledgments

We would like to thank all the performers who graciously allowed us to use their videos for this work, including D. Sculley, Kate Lane, Ed Moreno, Glenn Davis, Martin Aguinis, Caile Collins, Laurence Moroney, Paige Bailey, Ayush Tewari and Yoshua Bengio. We also thank the performers and creators of external, public datasets, as well as all participants of our user study. We would also like to thank our collaborators at Google and DeepMind: Paul McCartney, Brian Colonna, Michael Nechyba, Avneesh Sud, Zachary Gleicher, Miaosen Wang, Yi Yang, Yannis Assael, Brendan Shillingford, and Yu Zhang.

References

[1] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dan Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org. 6

[2] Yannis M Assael, Brendan Shillingford, Shimon Whiteson, and Nando De Freitas. Lipnet: End-to-end sentence-level lipreading. *arXiv preprint arXiv:1611.01599*, 2016. 7

[3] Thabo Beeler and Derek Bradley. Rigid stabilization of facial expressions. *ACM Trans. Graph.*, 33(4), July 2014. 3

[4] J. Booth and S. Zafeiriou. Optimal UV spaces for facial morphable model construction. In *2014 IEEE International Conference on Image Processing (ICIP)*, pages 4672–4676, 2014. 3

[5] Christoph Bregler, Michele Covell, and Malcolm Slaney. Video Rewrite: driving visual speech with audio. In *Siggraph*, volume 97, pages 353–360, 1997. 2

[6] Lele Chen, Zhiheng Li, Ross K Maddox, Zhiyao Duan, and Chenliang Xu. Lip movements generation at a glance. In *ECCV*, pages 520–535, 2018. 1, 2, 8

[7] Lele Chen, Ross K Maddox, Zhiyao Duan, and Chenliang Xu. Hierarchical cross-modal talking face generation with dynamic pixel-wise loss. In *CVPR*, pages 7832–7841, 2019. 1, 2, 7, 8

[8] Joon Son Chung, Amir Jamaludin, and Andrew Zisserman. You said that? In *BMVC*, 2017. 2, 8

[9] J. S. Chung and A. Zisserman. Out of time: automated lip sync in the wild. In *Workshop on Multi-view Lip-reading, ACCV*, 2016. 2

[10] 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, November 2006. 7

[11] Daniel Cudeiro, Timo Bolkart, Cassidy Laidlaw, Anurag Ranjan, and Michael J Black. Capture, learning, and synthesis of 3D speaking styles. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 10101–10111, 2019. 1

[12] Daniel Cudeiro, Timo Bolkart, Cassidy Laidlaw, Anurag Ranjan, and Michael J Black. Capture, learning, and synthesis of 3d speaking styles. In *CVPR*, pages 10101–10111, 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):127, 2016. 1, 2

[14] Ohad Fried, Ayush Tewari, Michael Zollhöfer, Adam Finkelstein, Eli Shechtman, Dan B Goldman, Kyle Genova, Zeyu Jin, Christian Theobalt, and Maneesh Agrawala. Text-based editing of talking-head video. *ACM Transactions on Graphics (TOG)*, 38(4):1–14, 2019. 1, 3

[15] Awni Hannun, Carl Case, Jared Casper, Bryan Catanzaro, Greg Diamos, Erich Elsen, Ryan Prenger, Sanjeev Satheesh, Shubho Sengupta, Adam Coates, et al. Deep speech: Scaling up end-to-end speech recognition. *arXiv preprint arXiv:1412.5567*, 2014. 2

[16] Naomi Harte and Eoin Gillen. TCD-TIMIT: An audio-visual corpus of continuous speech. *IEEE Transactions on Multimedia*, 17(5):603–615, 2015. 7

[17] Wei-Ning Hsu, Yu Zhang, Ron Weiss, Heiga Zen, Yonghui Wu, Yuxuan Wang, Yuan Cao, Ye Jia, Zhifeng Chen, Jonathan Shen, Patrick Nguyen, and Ruoming Pang. Hierarchical generative modeling for controllable speech synthesis. In *International Conference on Learning Representations*, 2019. 8

[18] Ahmed Hussen Abdelaziz, Barry-John Theobald, Paul Dixon, Reinhard Knothe, Nicholas Apostoloff, and Sachin Kajareker. Modality dropout for improved performance-driven talking faces. In *Proceedings of the 2020 International Conference on Multimodal Interaction*, pages 378–386, 2020. 2

[19] Tero Karras, Timo Aila, Samuli Laine, Antti Herva, and Jaakko Lehtinen. Audio-driven facial animation by joint end-to-end learning of pose and emotion. *ACM Trans. Graph.*, 36(4):94:1–94:12, July 2017. 1, 2, 5

[20] Yury Kartynnik, Artsiom Ablavatski, Ivan Grishchenko, and Matthias Grundmann. Real-time facial surface geometry from monocular video on mobile GPUs. In *Third Workshop on Computer Vision for AR/VR, Long Beach, CA*, 2019. 2, 3

[21] Michael K Keutmann, Samantha L Moore, Adam Savitt, and Ruben C Gur. Generating an item pool for translational social cognition research: methodol-

ogy and initial validation. *Behavior research methods*, 47(1):228–234, 2015. 7

[22] H. Kim, P. Garrido, A. Tewari, W. Xu, J. Thies, N. Nießner, P. Pérez, C. Richardt, M. Zollhöfer, and C. Theobalt. Deep Video Portraits. *ACM Transactions on Graphics 2018 (TOG)*, 2018. 3, 8

[23] Prajwal KR, Rudrabha Mukhopadhyay, Jerin Philip, Abhishek Jha, Vinay Namboodiri, and CV Jawahar. Towards automatic face-to-face translation. In *Proceedings of the 27th ACM International Conference on Multimedia*, pages 1428–1436, 2019. 2

[24] Mathieu Lamarre, J.P. Lewis, and Etienne Danvoye. Face stabilization by mode pursuit for avatar construction. In *Image and Vision Computing*, pages 1–6. IEEE, 2018. 3

[25] J.P. Lewis, Ken Anjyo, Taehyun Rhee, Mengjie Zhang, Frédéric H. Pighin, and Zhigang Deng. Practice and theory of blendshape facial models. In Sylvain Lefebvre and Michela Spagnuolo, editors, *Eurographics - State of the Art Reports*. Eurographics Association, 2014. 5

[26] 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 (TOG)*, 36(6):194, 2017. 2

[27] Michael Mathieu, Camille Couprie, and Yann LeCun. Deep multi-scale video prediction beyond mean square error. *ICLR*, 2016. 5

[28] Gaurav Mittal and Baoyuan Wang. Animating face using disentangled audio representations. In *WACV*, 2019. 1

[29] Niranjan D Narvekar and Lina J Karam. A no-reference perceptual image sharpness metric based on a cumulative probability of blur detection. In *International Workshop on Quality of Multimedia Experience*, pages 87–91. IEEE, 2009. 7

[30] 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. 1, 2, 7

[31] Linsen Song, Wayne Wu, Chen Qian, Chen Qian, and Chen Change Loy. Everybodys talkin: Let me talk as you want. *arXiv preprint*, arXiv:, 2020. 3

[32] Supasorn Suwajanakorn, Steven M Seitz, and Ira Kemelmacher-Shlizerman. Synthesizing Obama: learning lip sync from audio. *ACM Transactions on Graphics (TOG)*, 36(4):95, 2017. 1, 3

[33] Justus Thies, Mohamed Elgharib, Ayush Tewari, Christian Theobalt, and Matthias Nießner. Neural voice puppetry: Audio-driven facial reenactment. In *ECCV*, pages 716–731. Springer, 2020. 1, 2, 3, 6, 7, 8

[34] Justus Thies, Michael Zollhofer, Marc Stamminger, Christian Theobalt, and Matthias Nießner. Face2face: Real-time face capture and reenactment of rgb videos. In *CVPR*, pages 2387–2395, 2016. 3, 8

[35] Shinji Umeyama. Least-squares estimation of transformation parameters between two point patterns. *IEEE Trans. Pattern Anal. Mach. Intell.*, 13(4):376–380, 1991. 3

[36] Konstantinos Vougioukas, Stavros Petridis, and Maja Pantic. End-to-end speech-driven realistic facial animation with temporal gans. In *CVPR Workshops*, pages 37–40, 2019. 2, 7, 8

[37] Konstantinos Vougioukas, Stavros Petridis, and Maja Pantic. Realistic speech-driven facial animation with GANs. *IJCV*, pages 1–16, 2019. 2, 7, 8

[38] Lijuan Wang, Wei Han, Frank K Soong, and Qiang Huo. Text driven 3D photo-realistic talking head. In *Twelfth Annual Conference of the International Speech Communication Association*, 2011. 1, 2

[39] Wikipedia. Short-time Fourier transform. `https://en.wikipedia.org/wiki/Short-time_Fourier_transform`, 2019. 3

[40] Olivia Wiles, A Sophia Koepke, and Andrew Zisserman. X2face: A network for controlling face generation using images, audio, and pose codes. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 670–686, 2018. 1, 2, 8

[41] R. J. Williams and D. Zipser. A learning algorithm for continually running fully recurrent neural networks. *Neural Computation*, 1(2):270–280, 1989. 5

[42] Qingxiong Yang, Jinhui Tang, and Narendra Ahuja. Efficient and robust specular highlight removal. *IEEE transactions on pattern analysis and machine intelligence*, 37(6):1304–1311, 2014. 5

[43] Hang Zhou, Yu Liu, Ziwei Liu, Ping Luo, and Xiaogang Wang. Talking face generation by adversarially disentangled audio-visual representation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 9299–9306, 2019. 1

[44] Yang Zhou, Zhan Xu, Chris Landreth, Evangelos Kalogerakis, Subhransu Maji, and Karan Singh. Visemenet: Audio-driven animator-centric speech animation. *ACM Transactions on Graphics (TOG)*, 37(4):161, 2018. 2