Appendix for
SPACE 🚀: Speech-driven Portrait Animation with Controllable Expression

Anonymous ICCV submission

Paper ID 11139

A. Network architecture and training

A.1. Speech2Landmarks

Figure 1 shows the high-level architecture of the Speech2Landmarks (S2L) model. The main components include an audio encoder, landmark encoder, and the LSTM regressor that autoregressively predicts the face landmarks frame-by-frame.

The audio encoder is a 12-layer convolutional network with 1024 1D filters. Each layer uses leaky ReLU activations and BatchNorm. The first 4 layers use symmetrical filters spanning 3 time-steps. The last 8 layers use causal filters spanning 5 time-steps. The goal of this asymmetrical padding is to have the audio encoder focus more on the past and have fewer time-steps look-ahead, thus enabling the use in real-time applications.

There are two landmark encoders. For the 68 3DDFA landmarks we use an 8-layer fully-connected network with 1024 hidden units with leaky ReLU activations and BatchNorm. For the 52 MTCNN eye landmarks we use a 4-layer fully-connected network with 1024 hidden units with leaky ReLU activations and BatchNorm. Both the encoders are multi-layer fully connected networks.

The regressor is a 2-layer LSTM with 1024 hidden units. We also found that using a learned initialization for the hidden state leads to better initial outputs. For this purpose we use a 4-layer fully connected network which takes the initial time-step of all inputs:

$$\mathrm{h}_0 = \mathrm{MLP}(\mathrm{face}_0,\, \mathrm{a}_0) \tag{1}$$

A.2. Pose Generation

Figure 2 depicts the high-level architecture of our Pose Generation network (PoseGen). The encoder is a 2-layer bi-LSTM with 256 hidden units. The latent z is 64 dimensional. The decoder is a 2-layer LSTM with 256 hidden units.

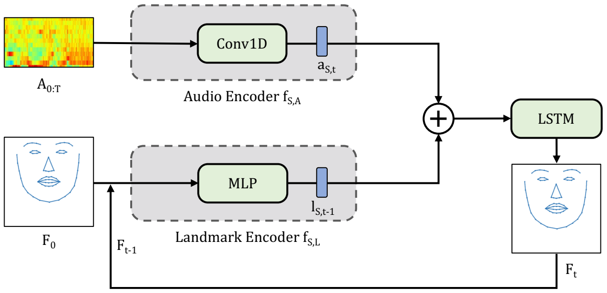


Fig. 1: **Speech2Landmarks architecture.** Speech2Landmarks takes as input the normalized face landmarks F_0, and the MFCC features of the speech recording $A_{0:T}$. It is an autoregressive model that takes as input the predicted face landmarks from previous time step, and audio features for the current time step.

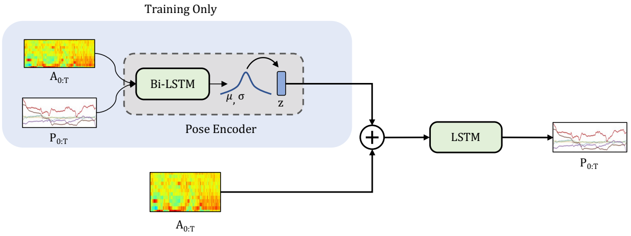


Fig. 2: **PoseGen architecture.** PoseGen is a variational autoencoder which jointly encoders audio and pose sequences. The decoder subsequently reconstructs the pose sequence conditioned on the input audio.

A.3. Landmarks2Latents

Figure 3 shows the high-level architecture of the Landmarks2Latents (L2L) model. The main components include an audio encoder, a face landmark encoder, a latent keypoint encoder, and an LSTM regressor that autoregressively predicts the latent landmarks frame-by-frame.

The audio encoder, face landmarks encoders, and the regressor follow the same architecture as Speech2Landmarks.

The source encoder encodes the latent keypoints from the

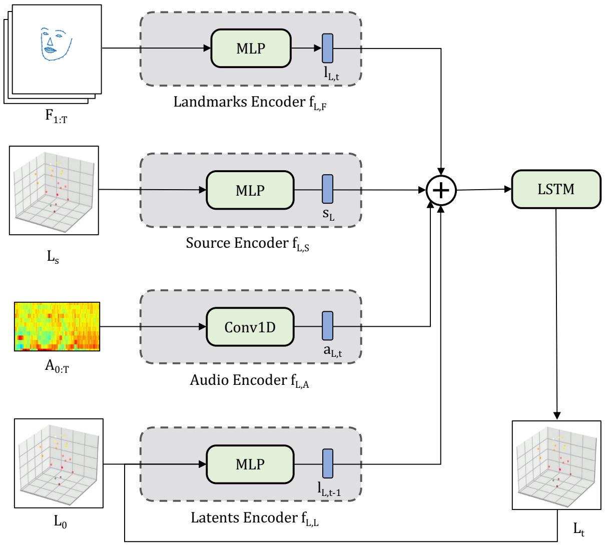


Fig. 3: **Landmarks2Latents architecture.** Landmarks2Latents is the final autoregressive component of SPACE. It is trained to transform facial landmarks to self-supervised keypoints learned by the talking head synthesis model. Landmarks2Latents takes as input the full face landmarks aligned with the image, which in turn are aligned with the keypoints. It also uses the input speech as an additional source of information. At a particular time step, the model takes the previous keypoints encodings, the current landmarks, audio features and source image keypoint encodings.

source image. It is a 4-layer fully connected network with 1024 hidden units with leaky ReLU activations and BatchNorm. The latents encoder encodes the latent keypoints of the previous time-step. It is an 8-layer fully connected network with 1024 hidden units with leaky ReLU activations and BatchNorm.

A.4. Video Generation

For details about face-vid2vid, please visit the project page: `https://nvlabs.github.io/face-vid2vid/`.

A.5. Training details

Each network is trained using the Adam optimizer with a batch size of 256. We use a learning schedule which warms up the learning rate from 1e−5 to 5e−4 over 10000 steps. Then we use a cosine schedule to decay the learning rate to 0 over 1 million steps. We find our networks converge within 200k steps.

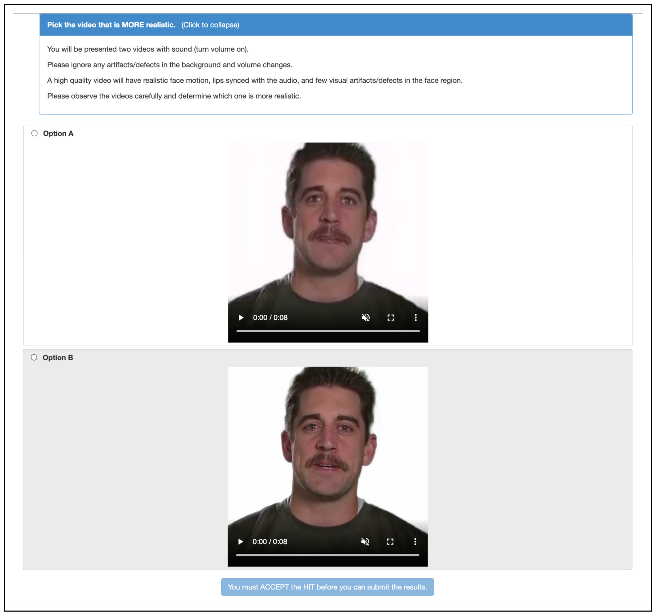
B. Data preprocessing

Since VoxCeleb1 and VoxCeleb2 datasets were captured in the wild, there is a large variation in the quality of videos due to different recording devices, camera viewpoints, head poses, occlusions, *etc*. We found it essential to filter the data to remove outliers and obtain a clean subset for training our models. First, we remove any video where a state-of-the-art face detection algorithm fails to detect a face in any of its frames. This is because we assume facial landmarks are available for all the frames in a video. If a face detector cannot detect a face in a frame, then we will either fail to extract the facial landmarks from the frame or have unreliable facial landmarks from the frame. Neither is desirable. We also remove a video with a large temporal difference between facial landmarks in any pair of its neighboring frames. Such a case often occurs in an edited video where the view transitions from one subject to another. In addition, we remove videos with large head rotations and camera zoom motions. We track the scale of the face landmarks and use the difference between the minimum and maximum scales as the criterion for filtering. We also remove videos with head rotations greater than 45 degrees along any axis. We use a pre-trained hand tracker to remove any videos where hands are detected.

For the Ryerson and MEAD datasets, we use a face detector to crop and resize the full size videos to 512×512. MEAD contains videos from various camera perspectives: front, up, down, left 30 degrees, right 30 degrees, left 60 degrees and right 60 degree. We discard the left 60 degrees and right 60 degrees videos.

B.1. Test Set details

Our test set was chosen to have a diverse set of initial poses. Using head pose estimation, we found that our test set has ∼45% data with rotations less than 15 degrees, ∼35%



Pick the video that is MORE realistic. (Click to collapse)

You will be presented two videos with sound (turn volume on).

Please ignore any artifacts/defects in the background and volume changes.

A high quality video will have realistic face motion, lips synced with the audio, and few visual artifacts/defects in the face region.

Please observe the videos carefully and determine which one is more realistic.

Fig. 4: **Amazon Mechanical Turk interface.** We ask users to choose the more realistic video, with a focus on the face and mouth regions.

data with rotations between 15 and 30 degrees, and ∼20% data with rotations between 30 and 45 degrees.

C. Evaluation

Figure 4 shows the interface of our user study. As described in the main text, a user is presented with a pair of videos from randomized models. Each pair consists of videos generated using the same reference image and audio with different models. The user is asked to pick the video they prefer in terms of face and mouth motions.

D. Controllability

One of the key features of our method is allowing control over certain facial motions which are difficult in the latent keypoint domain by utilizing face landmarks as an intermediate representation. We manipulate landmarks predicted by the S2L network before we feed it to the L2L network to achieve the desired affect on the output video. In this section we explain in more detail how we implement certain controls such as eye blinks and gaze shifting. While we demonstrate the ability to control eye landmarks, the nature of our method allows for controlling other landmarks manually such as the eyebrows or the mouth. Such capability is vital for use in content creation settings.

D.1. Eye blinks

We insert blinks following a two-step process. First, we determine where in the audio to insert blinks. In our implementation we do this by sampling time steps following a Poisson distribution with a given blink-per-minute. After determining where in the video to insert blinks we perform linear interpolation to overlap the upper eyelid landmarks with the lower eyelid and back to their original position across 4 frames which corresponds to 133 ms.

D.2. Eye gaze

Similarly, we can control the eye gaze. A user can specify the trajectory of the iris by shifting their landmarks' positions in the 2D plane and the output reflects the desired iris position. In the supplementary videos we show how we are able to move the iris in the both the x and y axis.