Talking us into the Metaverse: Towards Realistic Streaming Speech-to-Face Animation

Salvador Bedredín Medina Maza

CMU-LTI-23-012

Language Technologies Institute
School of Computer Science
Carnegie Mellon University
5000 Forbes Ave., Pittsburgh, PA 15213
www.lti.cs.cmu.edu

Thesis Committee:
Alexander Hauptmann
Shinji Watanabe
David Mortensen
Iain Matthews (Epic Games)

Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy
In Language and Information Technologies

ABSTRACT

Speech-to-face animation aims to create a realistic visual representation of a person's face based on their voice. Developing realistic facial animations of a person from a speech signal pertains to the main challenges of accurately capturing and reproducing the face's complex motions, including the tongue's motion. In this work, we use a combination of speech-pathology study techniques and machine learning to analyze a person's speech patterns and map them onto a digital avatar. The ultimate objective of our research is to animate realistic avatars in real time. To achieve this goal, we introduce a large-scale dataset that comprises 2.55 hours of corresponding speech and motion data captured from the tongue, lips, and jaw using electromagnetic articulography, while the facial motion was captured through stereo video. As an initial step, we make an exploratory analysis of different deep-learning-based methods for accurate and generalizable speech-to-tongue animation. We evaluate several encoder-decoder network architectures and audio features ranging from traditional to self-supervised audio representations. The best model achieves a temporal mean error of 1.77 *mm* when predicting the tongue, lips, and jaw motion and delivers realistic animations on singing audio regardless of training the model only using neutral speech from a single actor. Although adept at tongue movement predictions, this approach was limited in facial animations, prompting the evolution towards the IMFT'23 dataset that captures intricate facial motion pairing 2.28 hours of audio, video, and facial, lips, jaw, and tongue 3D landmarks. Our proposed Phonetically Informed Speech-Animation Network (PhISANet), results in animations with a sub-millimeter mean vertex error. It does this by incorporating WavLM feature encodings and by pioneering the use of a Connectionist Temporal Classifier (CTC) through Multi-task Learning in the speech-to-animation field of study. PhISANet generalizes across voices from different ages, genders, and languages. Perception user studies confirm that a CTC offers superior animation accuracy and realism and perceptible improvements in tongue and lip animations. Furthermore, to ensure animation quality, we advocate using visual speech recognition networks, specifically the AV-HuBERT model, as a benchmark. This research pushes the boundaries of realistic speech-to-animation, emphasizing the promise of real-time applications and setting a precedent in the speech-to-animation field of study.

ACKNOWLEDGEMENTS

I am profoundly grateful for the exceptional opportunity to be part of the Carnegie Mellon University community. In this environment, I interacted with some of the world's brightest, most dedicated, and most hard-working individuals I have ever met. I extend my heartfelt thanks to my advisor, Alex Hauptmann, whose immeasurable support allowed me to explore diverse areas, continuously fueling my curiosity and providing me with an invaluable taste of scientific rigor through the unique experiences that only his project can offer. The freedom that he gave me allowed me to explore areas such as speech animation, which I have been working on in the last few years.

I want to express my deep appreciation to the faculty of the Language Technologies Institute for their remarkable guidance and teachings. In particular, I am indebted to Eric Nyberg for introducing me to CMU and Robert Frederking for engaging me in insightful discussions that ignited my passion for research. L.P. Morency, your expertise in Multimodal Machine Learning has been a revelation to me. I sincerely thank Shinji Watanabe and David Mortensen for their invaluable assistance and thought-provoking discussions during the final stages of my Ph.D., which significantly enriched the research I presented to you here.

I am acutely aware that this thesis would not have been possible without the introduction to the topic, the challenging opportunities, and the continuous academic and resource support provided by Iain Matthews. Iain, thank you so much for your unwavering support throughout these years and for welcoming me to the remarkable community you founded at Epic Games. I'd also like to express my gratitude to Epic for the invaluable opportunities provided during my internships and for their generous funding throughout the concluding years of my Ph.D. journey. Sarah Taylor, I am grateful for sharing your expertise in the area with me. Your guidance and collaboration at every step of the journey have helped us create one of the most advanced Speech-to-Animation models out there. Carsten Stoll, your well-written and easily usable tools were indispensable for the research presented in this thesis and are fundamental to this project. Mark Tiede and Kevin J. Mulhall, I deeply appreciate your assistance at the beginning of this journey and for aiding in the creation of a groundbreaking dataset that enabled the realistic animation of a face and, in particular, the tongue. Brendan Mulcahy, without your invaluable help, the award-winning demo would not have been possible. Denis Tome, thank you for your friendship and insightful discussions while guiding me through the meticulous details required for publishing research. Tianye Li, your suggestions and comments on the analysis and visualizations presented in this work were immensely valuable. Albert Mosella, I thank you for the final push, enjoyable times, and the music that accompanied the culmination of this endeavor.

My friends from CMU, I am incredibly grateful for your unwavering presence and for making this journey as enjoyable and fulfilling as possible. Ramon, Hiro, Chaitanya, Dheeraj, Eva, Shruti, Min, Vipul, Michael, Ting-Yao, Bernie, Lijun, Kevin, Wenhei, Liangke, Billy, Paco —thank you!

A big thanks to my Mexican squad who turned those lengthy nights into moments of laughter and support: Abraham, Vic, Santiago, Octavio, Rodrigo, and Anaya. As time goes on and despite the distance, I'm optimistic we'll keep making memories and sharing stories.

Val and Memo, since our paths crossed upon our return to Pittsburgh, your friendship and consistent support have played a key role in this significant chapter of my life. My deepest thanks to you both.

My family has been supporting me in the idea of pursuing a Ph.D. from beginning to end. Mom, Julie, thank you, thank you, thank you. I will always be grateful for your unconditional love and support. Lastly, Mariana, my love, I cannot thank you enough for your endless support, love, and encouraging words during the hardest moments that only a place like CMU can put you in. You have always been there through this endless journey, along with Luna, who has kept my sanity and forced me to touch grass even on the darkest nights during this Ph.D. journey. As this phase ends, I am filled with anticipation for what lies ahead as I continue working on exciting ideas ahead.

CONTENTS

1 Speech Animation **19**

1.1 Motivation . . . 19

1.2 Speech Driven Facial Animation . . . 21

1.3 Speech Articulator Modeling and Animation . . . 23

2 Tongue Mocap Speech Animation Dataset **25**

2.1 Data Capture . . . 26

2.2 Diphonic Analysis . . . 27

2.2.1 Relationship between mid and parasagittal sensors . . . 28

2.2.2 Tongue width and curvature . . . 30

2.2.3 Dynamics of Parasagittal Sensors . . . 30

2.3 Jaw Motion Analysis . . . 31

3 Speech-to-Tongue Animation: A First Approach **33**

3.1 Proposed Method: A Two-Stage Pipeline . . . 34

3.2 Stage One: Landmark Motion Prediction from Audio . . . 35

3.2.1 Audio Encoding . . . 35

3.2.2 Experiments . . . 37

3.2.3 Quantitative Evaluation . . . 39

3.2.4 Qualitative Evaluation . . . 41

3.3 Stage Two: Parametric Face Model Optimization . . . 42

3.3.1 Optimizing the Rig Parameters . . . 42

3.3.2 Results . . . 44

3.3.3 User Study . . . 45

3.4 Limitations . . . 46

4 End-to-End Speech Animation **48**

4.1 New Data Generation . . . 50

4.1.1 Stereo Reconstruction . . . 50

4.1.2 Visual landmarks and EMA alignment . . . 51

4.1.3 IMFT'23 dataset . . . 51

4.2 Baseline Model . . . 53

4.3 New Audio Feature Representations . . . 54
4.3.1 WavLM . . . 54
4.3.2 Whisper . . . 55
4.4 Exploring Rig Parameter Velocity and Acceleration Loss . . . 57
4.5 Inner Layers of Neural Audio Feature Representations . . . 58
4.6 Phonetically Informed Speech Animation Network via Multi-task Learning (PhISANet) . . . 59
4.6.1 Phone Classifier . . . 61
4.6.2 Connectionist Temporal Classifier . . . 63
4.7 Experimentation . . . 65
4.7.1 Velocity and Acceleration of the Rig Parameters . . . 67
4.7.2 Inner Layers of Transformer Encodings Results . . . 68
4.7.3 Phonetically Informed Speech Animation Network (PhISANet) . . . 70
4.7.4 Rig Parameter Evaluation . . . 72
4.7.5 Mesh Vertex Evaluation . . . 74
4.8 User Studies . . . 80
4.8.1 Ground Truth Study . . . 82
4.8.2 Audio Feature Study . . . 83
4.8.3 WavLM Study . . . 84
4.9 Visual Speech Recognition Evaluation . . . 86
4.10 Qualitative Analysis . . . 89
4.11 Discussion . . . 90

5 Towards Real-time Streaming Speech-Animation 93
5.1 Streaming Model Design Framework . . . 94
5.2 Audio Encoding Selection . . . 96
5.3 Streaming Audio Encoding . . . 97
5.4 Model . . . 98
5.4.1 Reincorporating the idea of Multi-task Learning CTC . . . 100
5.5 Qualitative Results . . . 102
5.6 Discussion . . . 102

6 Conclusions 104

7 Where do we go from here? 107
7.1 End-to-End Model . . . 107
7.2 Data Augmentation . . . 108
7.3 Future Data Capture . . . 108
7.4 More Complex Models . . . 109
7.5 Model Compression and Computational Reduction . . . 109
7.6 Audio Feature Representations . . . 110
7.7 Interactive Agents . . . 110
7.8 Final Words . . . 111

Bibliography 113

A Results Statistical Analysis **126**
A.1 Model Configuration Comparison 126
A.2 Audio Feature Comparison 128

List of Figures

2.1 Sensor configuration for capturing the tongue, lips, and jaw mocap dataset. 27
2.2 (a) and (b) Correlation of midsagittal tongue sensors to left and right parasagittal sensors shown for each diphone and axis. (c) Distance between the parasagittal sensors in [mm] and proxy curvature (BL-TB-BR) in the coronal plane for each diphone. 29
2.3 Data visualization of diphones /sɔ/ and /ik/. Sensors are color spheres. Sensor motion is represented by colored quivers (rainbow). The tongue pose is the mean of the mid-position of the second phone. . . . 31
2.4 Different views for Posselt's Envelope of Motion for both jaw sensors LI and LJ: midsagittal (purple) and parasagittal (blue) respectively. (a) Front view, (b) Sagittal view, (c) Top view, (d) Head-axis reference: X-axis describes the anteroposterior direction (back to front), Y-axis describes the mediolateral direction (right to left), and Z-axis describes the vertical direction (bottom to top). 32

3.1 The proposed method consists of a two-stage pipeline. The first stage predicts the EMA landmark positions from the audio signal. In the second stage, the rig parameters are computed by minimizing the distance between the predicted landmark positions and their counterparts on the mesh through a Quasi-Newtonian optimizer. Finally, the rig parameters are used by a rendering engine to show the animation results. 33
3.2 The diagram depicts the explored configuration space of an encoder-decoder architecture comprising encoders (E), latent representations (Z), and decoders (D). The encoder determines the definition of the latent space (Z), while the architecture of the decoder can be selected independently from the encoder. 34
3.3 Visualization of the audio features for audio sample 0734 with caption *"The sky in the west is tinged with orange-red."*. The heatmap colors are normalized for a proper comparison. The audio features explored on these experiments are: (a) Phones, (b) MFCC, (c) DeepSpeech2, (d) Wav2Vec-Z, and (e) Wav2Vec-C. 36
3.4 Landmark Prediction Error on the test set of the bidirectional 5-layered GRU across all audio encodings. The error values shown are the temporal mean sample error in *mm* for each landmark. 42
3.5 Visualization of 300 *ms* of predicted motion on a test sample vs. ground truth. Ground truth is in gray, and the predicted motion is in color according to the sensor. (a) Shows an example of a predicted long motion. (b) Shows an example of motion shortcuts on prediction visible on *Blade Left* and *Tongue Tip* sensors. We suggest watching the video to obtain a more comprehensive evaluation. 43
3.6 FACS-based face model landmark correspondence visualization before optimization. Green: mesh surface landmarks $C(\theta_t)_l$; Pink: target constraints $C(N)_l + \delta_l$ generated by the articulation decoder. . . 44
3.7 Sample frame from the final results video of our approach. It shows how we can animate an Epic Games' MetaHuman character using in- and out-of-domain audio samples. 45

3.8 Visualization of two frames from test samples. On the first row, we see the ground truth landmark locations. The second row displays the predicted landmark locations. The third row shows the corresponding frame from the video reference and the solved animation frame. 45
3.9 Pairwise preference perception user-study. Left: The user interface of the survey shown to the user for a given sample. Right: User study summarized results where *GT*: Ground-truth animation; *pred*: predicted animation; *mismatch*: tongue mismatched animation; *null*: nullified tongue animation. GT was preferred overall, but the predictions from our model were preferred over mismatched animations and nullified tongue motion animations. 46

4.1 End-to-End model estimates the sequence of animation parameters directly from raw input speech. . . 48
4.2 Comparison between first approach and end-to-end model 49
4.3 Facial Landmark Tracking and Reconstruction. Throughout the session, the facial landmarks and lips were meticulously tracked on both cameras using a state-of-the-art industrial-grade tracker. The obtained 2D landmarks were then accurately reconstructed in three-dimensional space utilizing the Direct Linear Transformation (DLT) triangulation. 51
4.4 The mean and standard deviation of the 67 rig parameters. The parameters are grouped and color-coded according to their function: tongue, lips , nose, jaw, and the rest of the mouth. No abnormal values are observed across the parameter set. 52
4.5 Region of the actor's mesh influenced by the 67 rig parameters utilized for articulation animation, constituting this research's focal point. 53
4.6 Visualization of the selected WavLM and Whisper audio features for audio sample 0734 with transcript *"The sky in the west is tinged with orange-red."*. Note the difference in the range of values between both encodings. The WavLM feature shows stable patterns during the silence at the end of the sequence, while the Whisper model shows some noisy patterns during the silence segments. 56
4.7 Fusion of all transformer encoder layers of WavLM and Whisper models for the end-to-end model. . . 60
4.8 Fusion of Wav2Vec Layers for the end-to-end model. 61
4.9 Multi-task learning speech animation model with a phone classifier auxiliary task. 61
4.10 Multi-task learning speech animation model with a Connectionist Temporal Classification layer to predict the corresponding phone sequence. 63
4.11 CTC distribution and marginalization example over the word *null*. 65
4.12 This diagram presents the layout of the experiments conducted to obtain the best end-to-end model. We performed four different experiments: (1) Velocity and Acceleration Analysis, (2) Audio Encoder Inner Layer Aggregation, (3) Phonetically Informed CTC, and (4) Phonetically Informed Phone Classifier. The best resulting model from each experiment was evaluated through a series of (5) User Studies, and a novel approach to evaluate the speech animation model through (6) Visual Speech Recognition. Each experiment was carried out for each of the audio feature representations explored in this study. 66
4.13 Figure demonstrating the impact of weighted velocity and acceleration on prediction errors in rig parameter sequences from audio data. Results are based on three audio feature extraction methods: Wav2Vec, WavLM, and Whisper. Errors are evaluated using L1, L2, and a Normalized error (mean of min-max normalized L1 and L2 errors). This figure shows different audio feature representations require different weightings in the loss function. 68

4.14 Inner layer relevance visualization of the WavLM model. Layers, 1, 6, and 11 are the most relevant for the Base model (a), while the Large model (b) shows a similar pattern but with higher relevance on the initial layers of the Transformer encoder. 69
4.15 Inner layer relevance visualization of the Whisper model. Layers, 1, 6, and 11 are the most relevant for the Small model (a), while the Medium model (b) shows a similar pattern but with higher relevance on the initial layers of the Transformer encoder. 70
4.16 Multi-task learning comparative of the phone classifier (*Clf*) and CTC auxiliary tasks using WavLM audio encoding. The MTL models were evaluated on every single epoch on the test set. Both improve over the baseline, and the CTC errors are lower than the *Clf* regardless of the weight. 71
4.17 The figures present the L1 (a) and L2 (b) errors for the *All-Enc CTC* models trained on different audio encodings-Wav2Vec, WavLM, and Whisper-, tested across varying λ_{CTC} weights. Each bar group corresponds to a specific λ_{CTC} weight. On both metrics, the minimum values are obtained when λ_{CTC} is 0.008 across all the models. 71
4.18 Rig parameter L1 ((a), (b), and (c)) and L2 ((d), (e), and (f)) errors distribution from models to predict animation parameters across three audio features: Wav2Vec, WavLM, and Whisper. Training methodologies include the use of final layer embedding (*enc*), a weighted mean of all phase embeddings (*all-enc*), and a multi-task approach that employs Connectionist Temporal Classification (CTC) on the All-Enc model to predict the phone sequences (*all-enc ctc*). 74
4.19 Lower face ((a), (b), and (c)), tongue ((d), (e), and (f)), and lips ((g), (h), and (i)) regions temporal mean vertex error (*mm*) distribution over the test set. Different model configurations were evaluated to predict animation parameters across three audio features: Wav2Vec, WavLM, and Whisper. Training methodologies include the use of final layer embedding (*enc*), a weighted mean of all phase embeddings (*all-enc*), and a multi-task approach that employs Connectionist Temporal Classification (CTC) on the AllEnc model to predict the phone sequences (*all-enc ctc*). 76
4.20 This figure presents a side-by-side comparison of the temporal mean vertex error for the *Enc*, *All-Enc*, and *All-Enc CTC* models, all trained on Wav2Vec features. Subfigure (a) displays the mean vertex error for the lower face, visualized from two perspectives: a front view and a 3/4 view. Subfigure (b) specifically highlights the temporal mean vertex error of the tongue mesh from these same two viewpoints. It is evident that the *All-Enc* model tends to increase the error around the lip area and the tongue's tip. In contrast, the *All-Enc CTC* model reduces the overall error. This comparison allows for an in-depth model performance analysis across distinct facial regions. 77
4.21 This figure offers a comparative visualization of the temporal mean vertex error across the *Enc*, *All-Enc*, and *All-Enc CTC* models, each trained on WavLM features. Subfigure (a) illustrates the mean vertex error for the lower face, captured from both front and 3/4 viewpoints. Subfigure (b) specifically emphasizes the mean vertex error within the tongue mesh, displayed from identical viewpoints. The graphics highlight the improvements achieved by aggregating all Transformer Encoder layers from WavLM (*All-Enc*), evidenced by the reduction in overall error across the lower face and tongue regions. While the gains realized from adding a CTC (*All-Enc CTC*) are minimal, they contribute to a discernible, albeit subtle, enhancement in performance. 78

4.22 This figure provides a detailed comparison of the temporal mean vertex error in the *Enc*, *All-Enc*, and *All-Enc CTC* models, all trained on Whisper audio features. Subfigure (a) portrays the mean vertex error for the lower face, as observed from front and 3/4 views. Subfigure (b) focuses on the mean vertex error for the tongue mesh, visualized from the same two perspectives. Interestingly, the error in the lower face region demonstrates a progressive reduction as we aggregate the Transformer Encoder layers (*All-Enc*) and subsequently apply regularization through a CTC (*All-Enc CTC*). However, the tongue region exhibits a distinct pattern, with the *All-Enc* model resulting in a higher error at the tongue tip, which is marginally reduced in the *All-Enc CTC* model. 79
4.23 Interface for the user studies. Participants were prompted to discern which animation—left or right—was better aligned with the audio they heard. Their task involved evaluating the correspondence of the animation with respect to the audio, with particular attention to the motion of the lips and tongue. Subsequently, they evaluated in a 5-star rating matching level between their preferred animation and the audio. 81
4.24 Visual Speech Recognition (VSR) WER and CER metrics for different model configurations and audio features. The models evaluated are Enc, All-Enc, Enc CTC, and All-Enc CTC. The audio features evaluated are Wav2Vec, WavLM, and Whisper. Lower values for WER and CER indicate better performance. Based on the results, the Enc configuration performs best with the Wav2Vec feature, while the Enc CTC configuration performs best with the WavLM and Whisper features. 85
4.25 WER comparison vs. L1 and L2 rig parameter errors across epochs for All-Enc CTC models utilizing WavLM and Whisper Audio Encodings. Notably, the WER trajectory mirrors the L1 and L2 error trends in both models, with some intermittent broad fluctuations upon convergence. 88
4.26 Predicted sequence by our top-performing model, *WavLM All-Enc CTC*, for the sample sentence "So he hides the mayonnaise.". The corresponding phone is presented on top of the tongue rendered from a lateral view, while the face is presented in a three-quarter view. The symbol '∅' stands for silence. The displayed frames are carefully chosen within the boundaries predicted for each phone by the *Montreal Forced Aligner*. Observe the synchronized movements of the lips, jaw, and tongue, which follow the expected pattern for each phone. 92

5.1 This figure illustrates the various stages and components of a streaming model. 95
5.2 Procedure for Real-time Streaming Audio Feature Extraction and Selection. Initially, Wav2Vec features are inferred from the pre-trained model using overlapping context windows. One frame is omitted to downsample from 100 FPS to 50 FPS. The final feature from each window is then chosen to construct the streaming features, factoring in the bias from context computation. Subsequently, a streaming model is trained using the pre-computed streaming features. 98

LIST OF TABLES

2.1 EMA sensors positions on the tongue, lips, and jaw. The placement is either Midsagittal (M) or Parasagittal (P). 28

3.1 Model architecture evaluation using different audio feature representations: Phonemes (Phone), MFCC, DeepSpeech2 (DS2), Wav2Vec c- (W2V-C) and z- (W2V-Z) features. Models were trained with 300 ms input windows of audio. The error is the temporal mean L2-norm in mm calculated through the test split. The number of parameters reported is the number of trainable parameters per architecture design. The inference time is the mean time over the test split measured as ms per one second of audio input. . 40

3.2 Model architecture evaluation using different audio feature representations. Models were trained with 1 s input windows of audio. The error is the temporal MSE in mm calculated through the validation split. The number of parameters reported is the number of trainable parameters per architecture design. The inference time is the mean time over the validation split measured as ms per second of audio input. . . 41

4.1 Overview of the pre-trained WavLM audio encoder models available. The table outlines each model's number of Transformer Encoder layers, feature encoding dimension, number of parameters (expressed in millions as 'M'), and the quantity of data (expressed in hours) used for pre-training each model variant: Base, Base+, and Large. 55

4.2 Overview of the pre-trained Whisper speech recognition models available. The table outlines each model's number of Transformer Encoder layers, feature dimensionality, number of parameters (expressed in millions as 'M'), and the quantity of data (expressed in hours) used for pre-training each model variant: Tiny, Base, Small, Medium, and Large. 56

4.3 List of appearing phonemes and their allophones in the IMFT'23 dataset obtained by a forced alignment using the audio and their corresponding transcripts through the Montreal Forced Aligner. 62

4.4 Best velocity and acceleration weighting results for each audio encoding. 69

4.5 Performance metrics (L1 and L2 Errors) for different audio encodings (Wav2Vec, WavLM, and Whisper) at various λ_{CTC} coefficients. The results suggest an optimal λ_{CTC} value of 0.008 for most encodings, with minor deviations for specific metrics. 72

4.6 Summary of various architecture designs explored in our experiments. Each design is distinguished by its method of encoding, whether it utilizes the last Transformer layer only (denoted as 'Enc') or combines all Transformer layers (denoted as 'All-Enc'), and its multi-task learning (MTL) strategy, which may involve a phone classifier ('Clf') or a Connectionist Temporal Classification (CTC) for aligning the phone sequence. 72

4.7 Comparison of different model configurations using Wav2Vec, WavLM, and Whisper audio encodings. The evaluation of model configurations focused on the rig parameter space, considering metrics such as mean temporal L1 and L2 error, as well as mean temporal vertex error in *mm*. Specifically, the evaluation examined the performance of the models in the lower face, lips, and tongue regions of the mesh. . . . 73
4.8 Rating scale provided to participants during the perception user study. Each rating corresponds to the degree of alignment between the animation and audio in the selected video from a pairwise comparison. 80
4.9 Transcripts of selected test samples highlighting coarticulation motions with emphasis on diphones featuring open and rounded vowels. 82
4.10 Results from a user study employing pairwise selection tests as a sanity check to verify the quality of the models against the ground truth. A set of *All-Enc CTC* models trained using the Wav2Vec, WavLM, and Whisper audio feature representations were compared versus the ground truth animations generated from the IMFT'23 test set. The evaluation of the selected animations was conducted on a 5-point star rating scale, where users were asked to assess whether the selected animation corresponded with the audio. The scale ranged from 1, indicating *Does not match the audio*, to 5, indicating *Perfectly matches the audio*. A rating of 4 was assigned to animations that *Matches the audio*. 83
4.11 Results from a user study employing pairwise selection tests across the Wav2Vec, WavLM, and Whisper audio feature encodings using an *All-Enc CTC* architecture consisting of an encoder-decoder model has multi-task training configuration with a CTC branch. The evaluation was conducted on a 5-point star rating scale, where users were asked to assess the extent to which the selected animation corresponded with the audio. The scale ranged from 1, indicating *Does not match the audio*, to 5, indicating *Perfectly matches the audio*. A rating of 4 was assigned to animations that *Matches the audio*. 84
4.12 Results from a user study employing pairwise comparison tests across various model configurations using WavLM encoding. The configurations under consideration include: (1) *Enc*, wherein solely the last layer's embedding was utilized, (2) *All-Enc*, which linearly combined the embeddings from all layers of the transformer encoder, and (3) *All-Enc CTC*, where the model underwent multi-task training with a CTC branch. The evaluation was conducted on a 5-point star rating scale, where users were asked to assess the extent to which the selected animation corresponded with the audio. The scale ranged from 1, indicating *Does not match the audio*, to 5, indicating *Perfectly matches the audio*. A rating of 4 was assigned to animations that *Matches the audio*. 85
4.13 Performance comparison of four different model configurations (Enc, All-Enc, Enc CTC, and All-Enc CTC) with three different audio feature representations (Wav2Vec, WavLM, and Whisper). The models were evaluated using two metrics: Word Error Rate (WER) and Character Error Rate (CER), computed by the AvHuBERT visual speech recognition model. The results are presented as mean ± standard deviation. The best results for each audio feature representation are highlighted in bold. 89

5.1 Profiling results of various audio encoder models on CPU and GPU for encoding 1 second of audio. The table contrasts the performance of three prominent models—Wav2Vec, WavLM, and Whisper—in various configurations. The measured times are presented in milliseconds (*ms*) to highlight the relative computational efficiency of each model. 97
5.2 A summary of the TCN model's performance metrics across varying numbers of layers. The table illustrates the relationship between the number of layers, receptive field (RF) in frames and milliseconds, RF latency, and the respective L1, L2, and temporal mean vertex errors (TMVE). Notably, the receptive field expands with additional layers, influencing the model's error metrics. 99

5.3 Performance evaluation of the CNN model across different numbers of layers and kernel sizes. The table presents the model's receptive field, RF latency, and associated error metrics. It's highlighted that increasing the layers influences the receptive field and the subsequent need for input padding, affecting the overall performance. 100
5.4 Performance metrics of the streaming model under varying λ_{CTC} regularization coefficients. The table showcases the relationship between the weighting coefficient of the auxiliary task and the resulting L1 and L2 error on the rig parameter space, as well as the temporal mean vertex error (TMVE). A saddle point in performance is observed at $\lambda_{CTC} = 0.004$. 101
5.5 Performance evaluation of the CNN trained with a CTC via MTL across different numbers of layers and kernel sizes. The table presents the model's receptive field, RF latency, and associated error metrics. It's highlighted that increasing the layers influences the receptive field and corrects when the receptive field surpasses the training window. There is an overall improvement when compared versus the non-CTC version of the models. 101

A.1 Comparison of p-values for the L1 error on the rig parameter space across three audio features: Wav2Vec, WavLM, and Whisper. The p-values represent the statistical significance when comparing between the *Enc*, *All-Enc*, and *All-Enc CTC* models. Values $p < 0.05$ are considered statistically significant. All comparisons in the table are statistically significant. 126
A.2 Comparison of p-values for the L2 error on the rig parameter space across three audio features: Wav2Vec, WavLM, and Whisper. The p-values represent the statistical significance when comparing between the *Enc*, *All-Enc*, and *All-Enc CTC* models. Values $p < 0.05$ are considered statistically significant. Comparisons in red are not statistically significant. 127
A.3 Comparison of p-values for the lower face temporal mean vertex error (TMVE) across three audio features: Wav2Vec, WavLM, and Whisper. The p-values represent the statistical significance when comparing between the *Enc*, *All-Enc*, and *All-Enc CTC* models. Values $p < 0.05$ are considered statistically significant. All comparisons in the table are statistically significant. 127
A.4 Comparison of p-values for the L1 error over rig parameters, evaluating the impact on the performance of different audio features (Wav2Vec, WavLM, and Whisper) across model configurations (*Enc*, *All-Enc*, and *All-Enc CTC*). Values $p < 0.05$ are considered statistically significant. Comparisons in red are not statistically significant. 128
A.5 Comparison of p-values for the L2 error over rig parameters, evaluating the impact on the performance of different audio features (Wav2Vec, WavLM, and Whisper) across model configurations (*Enc*, *All-Enc*, and *All-Enc CTC*). Values $p < 0.05$ are considered statistically significant. Comparisons in red are not statistically significant. 128
A.6 Comparison of p-values for the lower face temporal mean vertex error (TMVE), evaluating the impact on the performance of different audio features (Wav2Vec, WavLM, and Whisper) across model configurations (*Enc*, *All-Enc*, and *All-Enc CTC*). Values $p < 0.05$ are considered statistically significant. All comparisons in the table are statistically significant. 128

ACRONYMS

ASR Automatic Speech Recognition

CCNN Causal Convolutional Neural Network

CER Character Error Rate

CNN Convolutional Neural Network

CTC Connectionist Temporal Classification

cVAE Conditional Variational Auto-Encoder

DLT Direct Linear Transformation

DS2 DeepSpeech2

EMA Electromagnetic Articulography

EPG Electropalatography

FPS Frames Per Second

GRU Gated Recurrent Unit

LLM Large Language Model

LSTM Long Short-Term Memory

MFCC Mel-Frequency Cepstral Coefficients

MLP Multilayer Perceptron

MRI Magnetic Resonance Imaging

MSE Mean Squared Error

MTL Multi-task Learning

OOD Out-of-Domain

OOV Out-of-Vocabulary

PhiSANet Phonetically Informed Speech-Animation Network

PINN Physics Informed Neural Network

RNN Recursive Neural Network

SSL Self-Supervised Learning

TCN Temporal Convolutional Network

TMVE Temporal Mean Vertex Error

TTS Text-to-Speech

VSR Visual Speech Recognition

W2V Wav2Vec

WER Word Error Rate

OVERVIEW

Chapter 1: Speech Animation

In the first chapter, we introduce the problem of speech animation, discussing its relevance and providing a brief history of the field. We also look closely at the current approaches to animating a face from speech only, highlighting their pros and cons.

Chapter 2: Tongue Mocap Speech Animation Dataset

As we focus on animating the tongue from an input speech signal. In the second chapter, we present a novel tongue and face motion dataset that was captured to enable the training of data-driven models. This dataset was captured from a single actor uttering phonetically balanced sentences. We provide quantitative and qualitative analysis to verify its relevance and demonstrate how it differs from previous datasets.

This work was accepted at Interspeech'21 [105] and presented in a talk where we highlighted the importance of parasagittal sensor placement for capturing rich tongue motion.

Chapter 3: Speech-to-Tongue Animation

The third chapter delves into our first approach to solving the problem of animating the lips, tongue, and jaw with the new dataset. We propose a two-stage pipeline that consists of an Encoder-Decoder model and a quasi-Newtonian optimizer. The first stage of the pipeline converts the raw audio signal into an intermediate audio feature representation, which is then decoded into 3D EMA landmark positions. The second stage takes these predicted landmark positions as input and uses them as constraints in a 3D optimization step to find the best animation parameters for a 3D model.

This work was accepted at CVPR'22 [106] and won the Best Demo Award of the conference, where attendants could experience the results of our proposed method by animating 3D models using their voices.

Chapter 4: End-to-End Model

The chapter begins by discussing the two-stage pipeline solution for speech animation, which is explored in the previous chapter. However, this approach is found to be slow due to the computation of the Hessian for optimal rig parameter determination in the animation sequence.

To address this limitation, the chapter introduces a new dataset called IMFT'23, which extends the previous IMT'22 dataset by incorporating 3D facial information and animation parameters. This expanded dataset enables training neural networks specifically designed for the speech-to-animation task.

Furthermore, the chapter explores two robust and generalizable neural audio feature representations, WavLM and Whisper, and evaluates their effectiveness. Based on the findings, a novel architecture called PhISANet (Phonetically Informed Speech Animation Network) is proposed. PhISANet leverages a CTC layer to phonetically inform the network, resulting in realistic tongue, lip, and lower face motion in the generated animations. We also demonstrate the generalizability of PhISANet by effectively predicting animation parameters from various speech signals, including singing audio.

Chapter 5: Streaming Model

In this chapter, we focus on developing a real-time animation system for MetaHumans using the Wav2Vec model for audio encoding due to its efficient inference time. Through experiments, we examine the effects of lookahead and receptive field size on latency and performance, using the CNN and TCN as our primary testbed architectures. Our results highlight the importance of keeping the receptive field within the training sample durations for optimal performance. Building on insights from PhiSANet, we enhance our three-layer CNN model by adding a CTC auxiliary task to align phone sequences, leading to moderate performance improvements. Our efforts indicate the potential of a real-time speech-animation model that can produce facial and tongue animations at 30 FPS with minimal delay. Further refinements and optimizations of these models are essential for their practical application.

Chapter 6: Conclusions

We present an extensive summary and highlight the findings and contributions described in the previous Chapters.

Chapter 7: Where Do We Go from Here?

Our pioneering PhiSANet model, the first to utilize a Connectionist Temporal Classifier (CTC) through Multi-task Learning, marked a significant improvement in animation accuracy and realism. Despite these strides, potential enhancements remain. In this chapter, we discuss the immediate and long-term future work, including expanding data through voice cloning, capturing more intricate facial and oral movements, and adopting more complex and novel models like RWKV for fluid animation generation. Future prospects also encompass model compression for faster real-time applications and leveraging next-generation audio feature representations for richer animations. Moreover, integrating our findings with Large Language Models promises transformative applications for interactive agents, impacting sectors like customer service, healthcare, education, and retail, streamlining processes and elevating user experiences.

CHAPTER 1

SPEECH ANIMATION

The field of speech-to-animation has gained significant momentum in recent years, promising to revolutionize the landscape of digital communication. This innovative technology aims to construct dynamic and accurate virtual avatars that can reproduce visual speech patterns in real-time. The complexity of this endeavor is encapsulated in the accurate capture and recreation of intricate facial movements. In this work we present an investigation with a particular emphasis on the motion of the tongue. To achieve this, our approach employs a synergy of computer vision, machine learning, and speech pathology study techniques, providing a robust framework to extract salient speech features and map them onto a virtual avatar.

Achieving a significant degree of realism necessitates a comprehensive approach to speech-driven animation. This entails not only modeling the visible movements of the lips and face but also the more concealed, yet crucial, motions of the tongue. Despite its limited visibility during speech production, the tongue is indispensable during coarticulation and sound production, which constitutes an essential element for comprehensive and realistic animations.

The potential applications of this technology are widespread, spanning various sectors such as telecommunications, entertainment, and healthcare. Each of these domains benefits from enhanced interactions with AI agents and other users through lifelike avatars to foster a more natural and intuitive digital communication experience.

These considerations underpin the motivation driving this research, setting the stage for the subsequent discussion of the challenges and potential solutions within this sphere. Our guiding vision is to promote more authentic, engaging, and accessible digital experiences through advancements in speech-to-animation technology.

1.1 Motivation

The primary goal of this project is to pioneer a generative model that produces real-time realistic facial animations solely from speech. Such a system could facilitate engaging interactions with virtual characters and digital avatars with an ease never before experienced. Our aim is to transcend the current state-of-the-art in speech-to-animation by creating

natural coarticulation motions by focusing on the integration of realistic animation of the lower face and tongue into high-quality 3D characters.

Our focus on driving 3D characters, as opposed to employing a video synthesis approach, is strategic. This approach is particularly potent in the context of virtual and augmented reality scenarios—environments where 2D generative methodologies typically fail to produce adequate results.

By harnessing a 3D approach, we can generate content from new angles and perspectives, even those not originally captured from its source. This flexibility makes our approach a powerful tool for creating novel content, thereby opening up a myriad of opportunities for enhanced interactive experiences.

While facial tracking technologies indeed have made strides in creating realistic digital avatars from monocular video, as showcased in previous research [13, 43, 173], they often falter under less-than-ideal conditions. Practical scenarios, where the camera is not ideally positioned in front of the person controlling the virtual character, or where the lighting conditions are inadequate, pose significant challenges.

Furthermore, facial occlusions, such as accessories, clothing, or face masks; also significantly impact the accuracy of facial tracking. This impact extends to individuals with dense facial hair, where motion transfer to a character without similar features becomes particularly challenging to facial capture approaches.

To address these limitations and to enhance the overall immersion in virtual environments, we believe in the necessity of a real-time speech-to-animation solution. This approach would not only circumvent the constraints posed by traditional facial tracking and motion transfer methods, but it would also amplify the perceived realism of virtual avatars, thereby enriching the overall user experience in these environments.

A speech-animation solution can account for these scenarios where full face visibility is compromised as long as the voice remains constant. Leveraging this consistency, our speech-driven solution can ensure a seamless interactive experience in virtually any scenario, thus broadening the viability of avatar-based digital interactions.

Our focus is specifically on enhancing the animation of lips, jaw, and tongue; a vital detail often overlooked due to its assumed invisibility. However, we posit that these elements significantly contribute to animation realism and user immersion. By accurately generating jaw movements and inner mouth motions, we can render animations that mirror real-life speech articulation more precisely.

We found that such improvements in animating coarticulation substantially amplify viewer engagement and perception of virtual or human-driven animated characters in a wide range of interactive systems. This enhancement holds true irrespective of whether a human or an automated system drives the character. Integrating this level of realism is expected to unlock new potential in the immersive experience that digital characters bring.

Envisioning the practical implications of our speech-to-animation solution for realistic avatars, we identify a vast array of potential applications in sectors such as telecommunications, entertainment, and healthcare. One particularly salient application is within the realm of online gaming; a modern-day virtual playground where children commonly interact with strangers.

Today, real-time communication between players has been made possible by advancements in video conferencing technology. However, this reveals potential issues concerning privacy protection, particularly for younger users. This is where our innovative solution can prove applicable. By representing players through lifelike virtual characters that animate in sync with their speech, we can ensure a rich and immersive gaming experience while simultaneously safeguarding their visual identity.

In this way, our system provides a dual advantage. Not only does it significantly enhance the realism and interactivity of the gaming experience, but it also upholds a critical element of online safety. By refining the nuances of facial animation, particularly in the lower face and tongue, we believe our solution sets a new standard for digital interactions, one where engagement and privacy coexist seamlessly.

In the healthcare sector, natural and realistic facial animation development holds immense potential, particularly for senior patients. Enhanced interaction with a lifelike, conversational avatar could significantly improve their overall well-being. A critical factor contributing to the health deterioration in seniors is the lack of social interaction, often leading to depression. This emotional state not only weakens the immune system but may also trigger conditions like senile dementia.

This aspect is particularly significant to me due to personal experience. My father suffered from Alzheimer's disease, and interaction with patients suffering from this condition can be emotionally draining for caregivers, including healthcare workers and family members. Conversations often become repetitive and lack logical coherence, which can be mentally exhausting over time. Looking ahead, I envision interactive systems capable of conversing with patients, potentially slowing the disease's progress and providing respite for caregivers. If these systems could drive a realistic avatar bearing the likeness and voice of a familiar person to the patient, it could help them feel calmer and safer. This approach could significantly enhance the quality of life for patients and those caring for them.

In conclusion, this research's motivation is rooted in the belief that the future of digital interaction lies in the realm of speech-driven animation. By incorporating realistic lower facial movements into 3D characters, we can redefine digital interactions across various sectors. Whether it is enhancing the gaming experience in virtual spaces, safeguarding players' privacy, or providing comfort to patients and relief to caregivers in healthcare environments, the applications of this technology have significant ramifications across many sectors. My pursuit to innovate the current state-of-the-art in speech-to-animation is not just a technical endeavor, but a deeply personal one. Guided by the experiences of those closest to us, we are determined to unlock the potential of speech-driven animation, making digital interactions more realistic, engaging, and empathetic than ever before.

1.2 Speech Driven Facial Animation

Within the field of speech-to-animation, we distinguish two main groups where the methods fall in: a) speech-to-face video synthesis, and b) speech-to-3D animation.

Speech to face video synthesis. Recent GAN-based work [143, 155, 169] take an image-audio pair as input and generate videos of the person articulating the audio file. [73] learn a joint embedding of the target face and speech to generate an image in sync with the speech segment. The model combines audio and identity encoders to feed an image generator

and is trained on unlabeled videos in a self-supervised fashion. [170] propose a two-step approach that combines a facial landmark prediction network with a translation network to generate photo-realistic or cartoon images. [158] use a parametric 3D face model representation driven by audio features. The synthesized sequence is fed into a neural face renderer to generate a photorealistic video portrait akin to [81, 82]. This approach shows good generalization capability for different audio inputs, including synthesized speech. [129] build upon the work by [110, 151] and learn a direct mapping from input to target modalities that require no style or domain transfer.

Text to face video synthesis. Text-based editing approaches typically align phoneme labels (extracted from text) to audio data, either recorded [51], or synthesized [7]. The work presented in [51] builds on Deep Video Portraits [82] to craft synthetic videos from input text only. Only the mouth region is synthesized and composited with the rest of the face. [7] propose a multimodal approach that allows the inclusion of emotional content to generate expressive face animations. The emotions and visemes are parameterized by a face appearance model, which in turn synthesizes the animation from the input text.

None of the video-based methods target tongue animation but rather focus on generating images of speaking faces, i.e., they indirectly learn to synthesize tongue motion using the underlying distribution of the training data. Our approach predicts 3D landmarks, which allows for retargeting and rendering animation on different target asset types with no further modifications or model training, which is a major advantage in practical and 3D workflows.

Speech to 3D animation. A common approach is to map sound units, *e.g.*, phonemes, onto shape parameters to drive a parametric model capable of animating [39, 89, 103, 140, 152]. Another example is JALI [42], which is a procedural method to generate viseme units of mouth motions from corresponding phonemes. Viseme sequences are blended into co-articulated action units to animate a FACS-based face rig [34], which can be used across different 3D characters.

Several Deep Learning approaches [78, 149, 150] map audio embeddings into an animated face. However, their output may not integrate well into a standard animation workflow. The work in [171] addresses this by generating animator-centric speech animations. Specifically, a Long Short-Term Memory (LSTM) architecture learns to predict phoneme groups and geometric locations from JALI parameters to animate a face. The work by [90] generates realistic animations using a registered 3D mesh sequence with accompanying phoneme-labeled audio data and a face model with a mapping procedure. It builds upon [85] for 3D performance capture, using a regression model *e.g.* random forests [96], to synthesize face model parameters from a sequence of speech features. Previous work has introduced the first multilingual / mixed-lingual speech animation model [67]. Their work uses a bidirectional LSTM model trained on phonetic posteriorgrams (PPG) as input, where the speaker differences were equalized while preserving phoneme distribution and duration information. The model is trained on an expressive speech-to-face dataset to enhance animations outputting a sequence of animation parameters. Other work uses a neural network that acts as a sliding window regressor [150]. The network estimates animation parameters from a phonetic speech representation. The proposed overlapping estimations models the localized context and co-articulation resulting in more fluid animation at the price of dampening the coarticulation. The work from [153] evaluates different audio representations, *i.e.* MFCCs and LPCs, to estimate 3D face model controllers through a bidirectional LSTM.

1.3 Speech Articulator Modeling and Animation

Speech articulator models are a representation of human speech physiology, including the tongue, the jaw, the hyoid bone, and the vocal tract wall. Capturing a model for the complete apparatus (or just parts of it) requires sophisticated imagery, such as volumetric MRI data [37, 61, 166] or x-rays [14, 15]. Most approaches focus on modeling the tongue (and sometimes the jaw) for speech synthesis applications.

Many approaches in the literature focus on lips and facial deformations. Animating the mouth interior has often been neglected as this particular task is challenging due to the lack of data and generalized subject-independent models. Recent vision based generative animation approaches have shown compelling results using generative adversarial networks (GANs) [143, 155, 169], image-to-image translations [170], or neural rendering [81, 82, 149]. However, none of these methods output 3D animation directly but implicitly generate 2D image frames of speaking faces. Our application is to learn to synthesize 3D speech and tongue motion that can be used in existing 3D computer animation pipelines.

Tongue modeling dates back to [120] which modeled a two-dimensional surface projection of the tongue in the sagittal plane that ignores the intrinsic structure of the tongue and only accounts for geometric surface deformations. The bio-mechanical model by [159] models soft tissue deformations and non-linear geometric effects. The 2D physiological model proposed in [61] unifies the tongue, jaw, and laryngeal structures using a 2D finite-element simulation from MRI data of a single subject. The parameterized model in [86] describes the tongue's surface through B-splines by forming a grid of bi-cubic patches over 60 control points found on the top and under the tongue to produce realistic tongue shapes as described in [148]. The phone shapes were matched by manually setting the parameters of the tongue model and achieving a tongue animation by blending the shapes between phonemes.

Since then, representing the audio through symbolic sound units such as phonemes onto shape parameters is a general approach for speech animation [39, 89, 102, 103, 140, 152]. For instance, JALI [42] is a procedural method to generate viseme units of mouth shapes from phonemes. The viseme sequences are blended into co-articulated motions to animate a FACS-based face rig model [114]. The generated animation from these approaches can be transferred to different characters if they share a common rigging system.

Different approaches have also explored tongue animation from different input modalities. In [147], a tongue 3D model is animated directly from electromagnetic articulography (EMA) data. This approach does not include any audio processing as the animation is synchronized with the recorded audio from the EMA capture session through their open-sourced framework [146].

Tongue animation has also been achieved from ultrasound images. In [48], the authors explore animating the tongue from low-resolution imagery represented by EigenTongue [68] features and mapped into control parameters through a Gaussian mixture model. Later in [25], more realistic animations are obtained from ultrasound images using a snake contour extraction algorithm and driving a finite element model of the tongue, achieving animations at 21 FPS.

A multimodal end-to-end hidden Markov model (HMM) proposed by [145] is capable of synthesizing audio and generating tongue motion. Unlike previous work, their method replaces the midsagittal EMA data with tongue model parameters as the target articulatory representation. A follow-up multimodal approach [167] replaces the HMM with a bottleneck long-term recurrent convolutional network (BTRCNN). The network is trained on text and audio to predict

EMA positions as a proxy for tongue movement while considering embedded articulatory features while training the model.

Similar to our approach, other work also considers only speech audio as input. In [98], the input speech is represented as a sequence of phonemes which is mapped into EMA sensor positions through an HMM. The predicted articulatory movements control the deformations of a 3D tongue model. Similarly, in [100], a stacked restricted Boltzmann machine predicts EMA sensor positions from audio represented as mel frequency cepstral coefficients (MFCC). The predicted positions are fit to a volume-preserving model through a finite element method to generate animations. Zhu et. al. [172] also use MFCC as input features to solve articulatory inversion on EMA positions using a 2-layered bidirectional LSTM preceded by a linear projection of the audio features into the RNN. This model achieves state-of-the-art results on the MNGU0 dataset [130]. However, in [12], they demonstrate that gated recurrent unit (GRU) networks have a slight performance improvement over LSTM architectures since the GRU layers have fewer parameters making them less prone to over-fitting.

In this dissertation, we significantly advance the domain of animation generation from speech. Our main focus is to move beyond the traditional linguistically driven features, such as phonemes or Mel-Frequency Cepstral Coefficients (MFCCs). Instead, we embrace neural audio feature representations derived from models trained on thousands of hours of audio. The robustness and continuity of these representations make them ideal for speech-to-animation research, enabling model generalization across various speakers and exceptional performance on out-of-domain utterances when training data is limited or sourced from a single speaker. We also explore deep-learning architectures, following the examples set by previous studies [78, 150, 171], to convert these audio feature representations into animation format through an end-to-end approach, leveraging on innovations from the Automatic Speech Recognition (ASR) community.

We also significantly contribute by developing a realistic tongue animation for 3D characters, an innovation derived from data obtained from a medical setup typically used in speech pathology studies. This novel approach departs from the traditional reliance on procedurally generated animations, which heavily rely on the viseme concept limiting the animations to artificial estimations of coarticulation. This innovative leap lays a solid foundation for future research focused on creating models capable of accurately replicating human natural coarticulation, a possibility enabled by recent breakthroughs in speech pathology studies, computer vision, and machine learning.

With these advancements, we invite you to delve into the next chapter, where we elaborate on these innovations and their implications for the future of animation generation from speech.

CHAPTER 2

TONGUE MOCAP SPEECH ANIMATION DATASET

Capturing the motion of the tongue is an important aspect of speech research and has traditionally been achieved through X-ray imaging [58, 70]. However, this method poses health risks to actors due to prolonged exposure to radiation, making it impractical for large-scale data collection. Additionally, the jaw, teeth, and skull can obscure the view of the tongue, making it challenging to capture the full range of tongue motion. This is why in recent years, radiation-free alternative methods for capturing tongue motion have been practiced, such as magnetic resonance imaging (MRI), 3D ultrasound, and electromagnetic articulography (EMA).

Real-time MRI [26, 83, 160] and 3D ultrasound [20] are safer options, but the resulting imagery is unregistered, making it challenging to track specific points on the tongue over time. Additionally, these methods suffer from slow sampling rates. EMA [128] is a more invasive method, but it has several advantages over other methods. EMA can measure sensor position and orientation at fixed locations on the tongue with high spatial and temporal resolution and low error.

Articulatory data has been captured in different modalities from EPG, Laryngography, and EMA in a midsagittal configuration. The MOCHA-TIMIT [161] corpus is a phonetically balanced dataset of 460 sentences read by two British English speakers. EMA was also used for capturing tongue motion in [135] for 320 utterances of Austrian German speech, to construct the mngu0 dataset [131] which contains 1354 utterances. In [141], Dutch and English speakers recited a short phrase and isolated words, while in [18], 3 Italian speaker were captured reading 500 Italian sentences providing approximately 2 hours of speech. EMA sensors are generally placed midsagittally along the tongue for capturing 2D deformation of the tongue tip, body and dorsum [128]. Although the parasagittal motions of the tongue contribute to speech production, they are largely overlooked during data collection.

There has been some prior work that considered lateral tongue motion [164] to study the production of /l/ in Australian English with the aid of two parasagittal sensors acquired at a rate of 100 Hz. The work presented by [79] included one parasagittal sensor to examine the contribution of lateral motion on the production of alveolar consonants in vowel-consonant-vowel syllables. Their findings indicate that lateral motion is fundamental for articulating the sound /z/. Two parasagittal sensors were included in the capture by [64] and [107], who respectively studied the articulation of Czech

liquids in isolated nonsense words and English liquids in carrier sentences by Japanese speakers.

The work in [70] analyzed patterns of deformations of the midsagittal edge of the tongue in transitions between lingual segments from X-Ray images. An analysis of tongue motion during emotive speech revealed that the vertical motion of the tongue dorsum is dampened during sad speech [83]. A study of vowel-consonant-vowel syllables in [46] revealed that tongue width is largest for palatal plosives and fricatives as the tongue widens pressed against the hard palate and smallest for velar plosives and fricatives since the tongue body volume is largely retracted towards the velum. The work in [164] investigated tongue lateralization in the Australian production of /l/ and discovered that the lateral tongue is actively controlled rather than moving as a bi-product of tongue stretching. In [53], video recordings of the tongue during the articulation of an English passage revealed that bilateral movements are asymmetric, and one side of the tongue typically moves ahead of the other depending on the speaker.

Most previous work analyzes isolated or nonsensical words, and there has been very little research into the 3-D tongue motion during continuous speech production. An exception is the work in [72], which presented a statistical technique for identifying critical, dependent, and redundant roles played by the articulators during the production of the English phonemes in the MOCHA-TIMIT corpus. They found that fricatives and affricates required the most number of critical articulators, and none were identified for the alveolar /l/. They additionally observed that the articulatory system comprises three largely-independent components: the lip and jaw group, the tongue, and the velum.

We found Electromagnetic Articulography (EMA) to be a suitable method for capturing data for speech-to-tongue animation purposes due to its accuracy and high sample rates. EMA provides high-resolution data that allows for accurate tracking of the tongue's movement during speech, which is crucial for creating realistic animations. This made it an ideal choice for our research.

2.1 Data Capture

We collected a new tongue motion capture dataset for the speech animation task with additional parasagittal sensors is a valuable addition to the field of speech-to-face animation. The linguistic analysis in [105] demonstrates the importance of adding lateral sensors to describe richer tongue motion dynamics. The data was collected using a Carstens AG501 Electromagnetic Articulography (EMA) device [19], an invasive process but with ethical and health guidelines approved by the Institutional Review Board (IRB). The EMA device uses a configuration of ten sensors to acquire the motion of the tongue, jaw, and lips. These sensors are attached to the surface using medical-grade cyanoacrylate glue.

The actor sits below nine RF transmitters creating an electromagnetic field that energizes coils in the sensors. The resulting currents are processed to recover five degrees of freedom for each sensor: three for the 3D position (x, y, z) and two for rotation (azimuth and elevation). The EMA sensors were sampled at 250 Hz, and mono audio was synchronously recorded at a sampling rate of 48 kHz. This high-resolution data allows for a more accurate analysis of the tongue, jaw, and lip movements, which can be used to train and improve models for speech-to-face animation.

Five sensors were positioned on the tongue: the midsagittal dorsum, blade, tip, and left and right parasagittal sensors on the blade. The tongue tip sensor was positioned 5 mm behind the apex to avoid any damage to the actor's teeth. Two sensors are located on the lower jaw: one on the gingival margin at the medial incisors and one in a parasagittal location

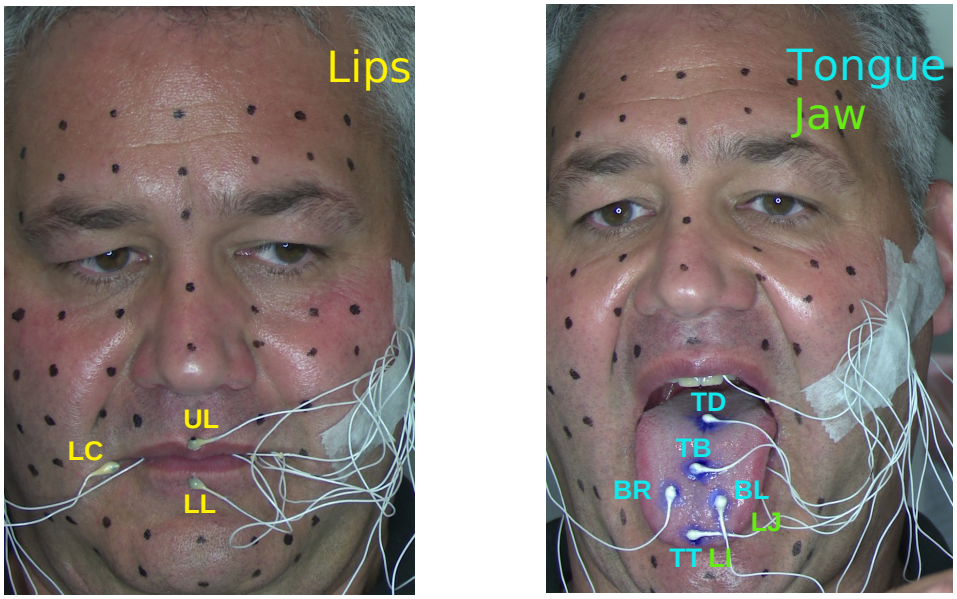


Figure 2.1: Sensor configuration for capturing the tongue, lips, and jaw mocap dataset.

between the canine and first premolar. Two more are placed midsagittally on the upper and lower lips at the vermilion border. The final sensor was placed at the right lip corner apex. To stabilize the speech articulator landmarks with respect to rigid head position, three additional sensors are positioned: one on the upper medial incisor and one each on the left and right mastoid process. The stabilization sensors capture rigid skull position and rotation over six degrees of freedom. A visualization of the sensor placements is shown in Figure 2.1 with the naming convention summarized in Table 2.1.

The data was recorded in a single 8-hour session by the actor reading a total of 2160 sentences. A subset of 720 sentences from the Harvard set [132] was repeated at both a regular and a fast pace. The remaining 1440 sentences come from the TIMIT dataset [52], which were uttered only at a regular pace due to time constraints for the capture session. The capture session resulted in a dataset formed by 2160 pairs of articulator mocap sequences and audio samples, being the first of its kind at this scale. Due to the long duration of the data capture session, utterance errors such as misreading and wrong correspondence were unavoidable. After manually inspecting each sample, we defined a subset of 1902 clean samples, which exclude reading errors and non-verbal gestures, giving a total of 2.55 hours of articulator mocap sequences paired with audio samples.

A synchronous HD reference video was captured from two cameras during the EMA capture session. The single actor was prepared with visual markers for visual analysis and to enable the reenactment of the face while solving for speech-to-face animation.

2.2 Diphonic Analysis

As a first approach to analyzing the data, we made a diphonic analysis to have insights into the motion captured in the data. A diphonic analysis is preferred over a phonemic analysis, as the former allows us to have a better insight of the coarticulation of the utterances emitted during the capture session. As a first step, we use the Montreal Forced

Table 2.1: EMA sensors positions on the tongue, lips, and jaw. The placement is either Midsagittal (M) or Parasagittal (P).

EMA Sensor	Position	Placement
TD	Tongue Dorsum	M
TB	Tongue Blade	M
BR	Tongue Blade Right	P
BL	Tongue Blade Left	P
TT	Tongue Tip	M
UL	Upper Lip	M
LC	Center Lip, Right Corner	P
LL	Lower Lip	M
LI	Jaw, Medial Incisors	M
LJ	Jaw, Canine & First Premolar	P

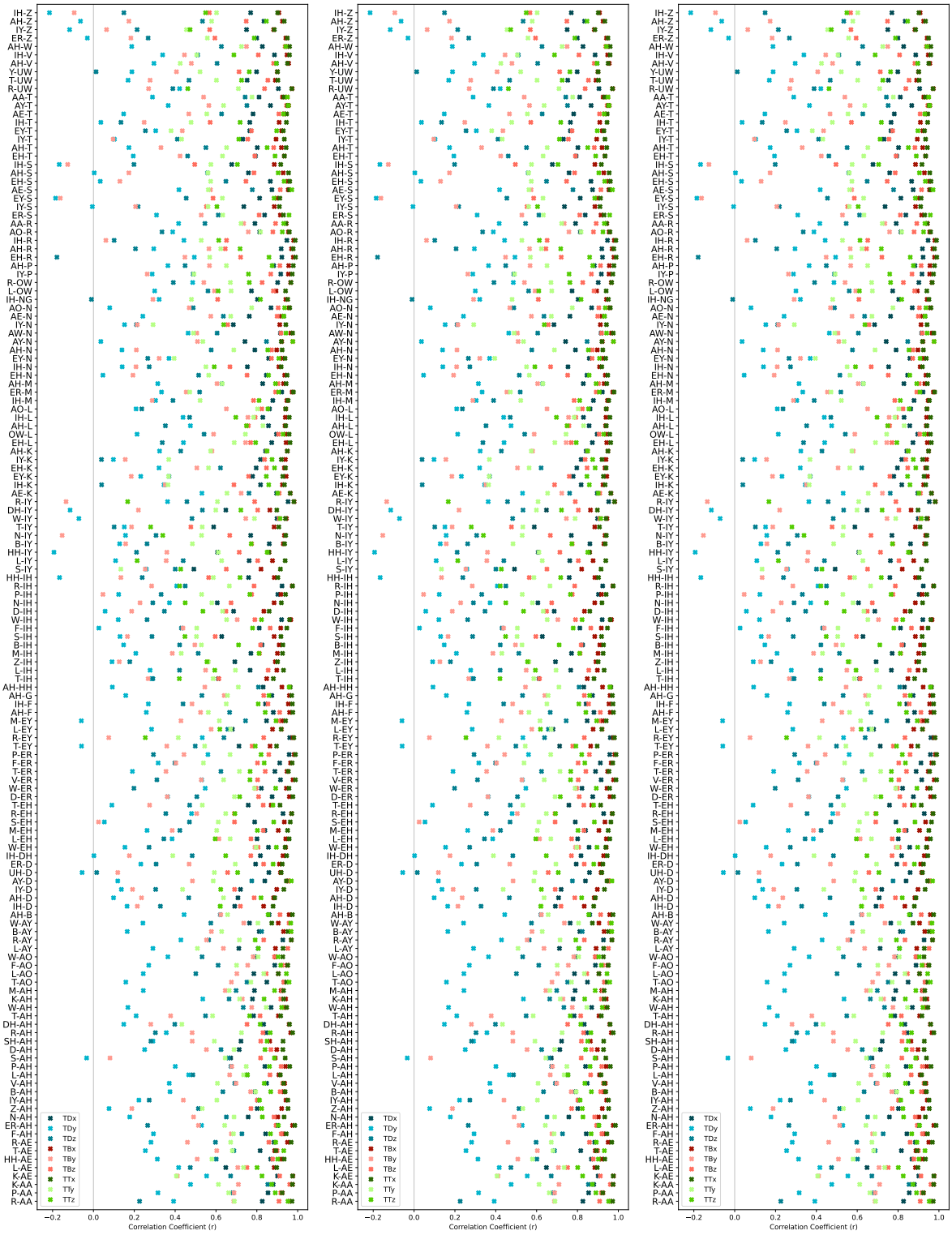
Aligner [104] to extract the diphone segments from the audio. For our analysis, we ignore the speech border diphones which include silence or non-speech segments. This first filtering of the data results in 1,158 diphones, from which 424 are consonant clusters. The remaining 734 diphones are distributed as follows: 305 vowel-consonant, 315 consonant-vowel, and 114 vowel-vowel. We filter out the consonant clusters and diphones with fewer than 86 examples resulting in 142 unique diphones, which cover 60.4% of the non-consonant cluster data.

2.2.1 Relationship between mid and parasagittal sensors

We first investigate the extent to which the parasagittal sensors deform with respect to the midsagittal sensors to identify the sounds where the parasagittal deformations are largely independent of the midsagittal motion. We select the Pearson correlation coefficient (r) for each midsagittal tongue sensor (TD, TB, TT) to each parasagittal sensor (BL, BR) independently for each of the x (anterior/posterior), y (left/right) and z (superior/inferior) to verify the correlation of the motion between the parasagittal sensors and the canonically used midsagittal sensors. The complete set of correlations for BL and BR are shown in Figures 2.2a and 2.2b.

We observe a high correlation of the parasagittal sensors with all the midsagittal sensors on the x-axis, demonstrating that during a regular speech diphone, the tongue's surface moves back and forth consistently. Moreover, we observe that the parasagittal sensors correlate most with the tongue tip, confirming the discoveries in [79], and are least correlated to the tongue blade and dorsum sensors in the coronal plane with a prominent difference on the y-axis for particular diphones. Specifically, we observe very low and slightly negative correlations with TD and TB in the coronal plane for the diphones that end with the alveolars /z/, /s/, /d/, or /n/. We find this effect to be less prominent for alveolar /t/. The same effect can be seen in diphones ending with the front unrounded vowels /i/ and /ɪ/.

The results suggest that the lateral tongue is actively controlled and does not move merely as a byproduct of midsagittal activity. Parasagittal sensors move independently of the mid-tongue sensors to the greatest extent in the coronal plane. This could indicate a) lateral curvature or b) a widening or narrowing of the superior surface to preserve tongue volume as it deforms.



(a) BL correlation **(b)** BR correlation **(c)** width and curvature

Figure 2.2: (a) and (b) Correlation of midsagittal tongue sensors to left and right parasagittal sensors shown for each diphone and axis. (c) Distance between the parasagittal sensors in [mm] and proxy curvature (BL-TB-BR) in the coronal plane for each diphone.

2.2.2 Tongue width and curvature

To analyze the width and curvature of the tongue we compute the 3-D Euclidean distance between the left and right parasagittal sensors as a proxy of the tongue width. On the other hand, the tongue curvature is computed as the Menger curvature [93] in the coronal plane for each diphone using three 2-D points corresponding to the y and z axes of BL, TB, and BR to determine the extent of the tongue roll and its relationship to the underlying speech. In this particular setup, a negative value represents a curled upward tongue surface, a positive value indicates a curled downward pose, and a zero value indicates a flat tongue. In Figure 2.3b, we visualize a diphone with slightly negative curvature showing a close to flat tongue, while in Figure 2.3d, we see an example with high positive curvature.

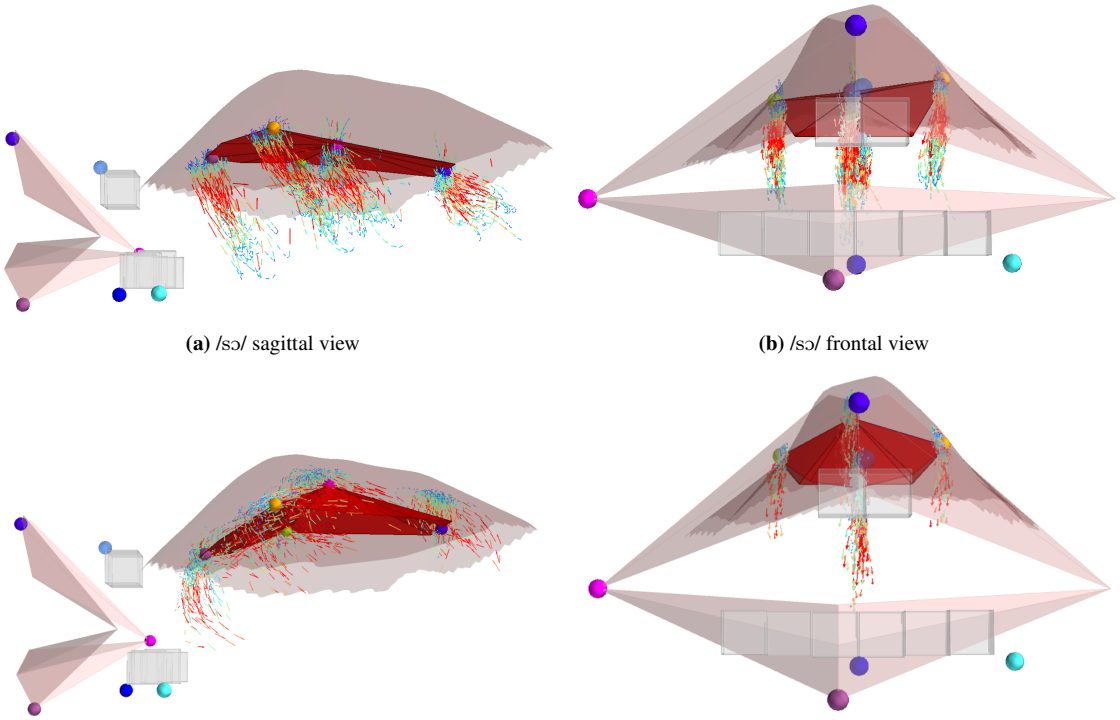
The means and standard deviations of tongue width and curvature for each diphone can be found in Figure 2.2c shown in ascending order of curvature. We generally observe that tongue width negatively correlates with curvature ($r = -0.384$). This is intuitive since the sensors become closer as the edges of the tongue curl up. At the top of the graph, we observe a cluster of diphones containing the velar consonants /k/, /g/, and /ŋ/ paired with vowels /i/, /ɪ/ and /ʌ/. These are associated with a relatively narrow tongue and large downward curvature of the lateral tongue. They are followed by a cluster of diphones containing the vowel /i/ with a range of consonant contexts that have diverse places of articulation. However, outliers appear when /i/ is spoken in the context of the alveolar fricatives /ʃ/ and /ʒ/, where we observe that the tongue curvature is approximately halved. The diphones that contain /ʃ/ and /ʒ/ appear towards the bottom of the graph, although /ʒ/ is distributed more uniformly throughout the lower half. The outliers are, therefore, the result of co-articulation that stems from transitioning between a flat or upwards-curled tongue to a downwards curvature and vice versa. This result indicates that parasagittal tongue motion is essential for producing these sounds.

2.2.3 Dynamics of Parasagittal Sensors

Our geometric analysis of the parasagittal sensors indicates the shape of the tongue, but tongue dynamics are lost. Figure 2.3 shows the frontal and sagittal view of diphones /sɔ/ and /ik/. The colored spheres represent all the EMA sensors. The images show the palate surface reconstruction. The lips and teeth are not a reconstruction from the data but serve as a reference for a better spatial understanding. The tongue's pose shown in the figures is the mean of the mid-poses from the diphone. The color-coded quivers represent the sensors' motion from all the data samples for the given set of diphones. The motion sequence is represented by colors of the rainbow, from violet to red, representing the motion from beginning to end, respectively. For example, we can observe how diphone/sɔ/ starts with the tongue tip close to the alveolar ridge (violet) followed by a rapid gesture that moves the tongue downwards (cyan) and back to a stationary position (red) in Figure 2.3a. Figure 2.3c, we can appreciate how the curved transition of /ik/ begins with a quick constriction on the palate and ends with a low frontal tongue pose. All the diphones images and videos can be found through this link [1] for better understanding and insights into the tongue sensor motion.

To gain insight into the tongue's motion statistics, we compute the peak velocities of the five tongue sensors for all diphone samples and calculate the mean of the velocities for each diphone class. In our analysis, we found that the diphones with alveolar and post-alveolar fricatives /z/, /s/, and /ʃ/ show low mean peak velocity below 40 mm/s due to the long periods in which the tongue remains stationary. Alternatively, the diphones with the highest velocities above 180 mm/s require an open or close movement of the jaw such as /ʌr/, /kɔ/, /ɑt/, and /ɑk/.

[1] https://salmedina.github.io/ContinuousTongueMotionAnalysis/



(a) /sɔ/ sagittal view

(b) /sɔ/ frontal view

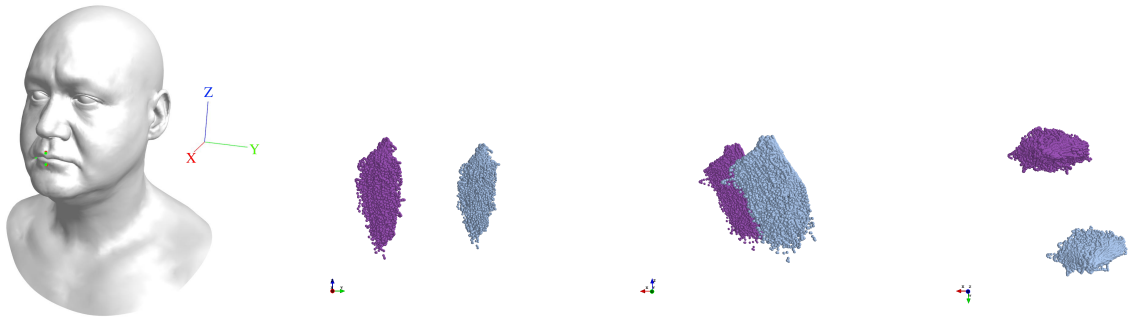
(c) /ik/ sagittal view

(d) /ik/ front view

Figure 2.3: Data visualization of diphones /sɔ/ and /ik/. Sensors are color spheres. Sensor motion is represented by colored quivers (rainbow). The tongue pose is the mean of the mid-position of the second phone.

2.3 Jaw Motion Analysis

In Figure 2.4, we visualized all the samples of both jaw sensors LI (midsagittal) and LJ (parasagittal) in a 3D scatter plot from three different views. As in can be seen, our captured data follows Posselt's Envelope of Motion (PEM) [123]. The PEM describes the range under which the jaw can move due to the physiological constraints based on the bones, muscles, and tendons that form the jaw. For context, we can see in Figure 2.4a that the reference frame of our captured data is the following: the X-axis describes the anteroposterior direction, the Y-axis the mediolateral direction, and the Z-axis the vertical direction. As carefully described in [174], the jaw motion from a frontal view usually follows a shield appearance as seen in Figure 2.4b. From a sagittal or lateral view, it follows a prolonged fang shape as shown in Figure 2.4c. While from a top view, the jaw motion follows a diamond shape as the one displayed in Figure 2.4d. Our visualization shows that our data follows such shapes with undefined edges. The main reason is that the actor did not do any extreme articulation during the capture session, as he only uttered regular sentences at a regular pace and fast pace with neutral emotion. This visualization supports that it would be appropriate to increase the variability of the gesticulations during a capture session in future work.



(a) Axis reference **(b)** Front View **(c)** Right Lateral View **(d)** Top View

Figure 2.4: Different views for Posselt's Envelope of Motion for both jaw sensors LI and LJ: midsagittal (purple) and parasagittal (blue) respectively. (a) Front view, (b) Sagittal view, (c) Top view, (d) Head-axis reference: X-axis describes the anteroposterior direction (back to front), Y-axis describes the mediolateral direction (right to left), and Z-axis describes the vertical direction (bottom to top).

CHAPTER 3

SPEECH-TO-TONGUE ANIMATION: A FIRST APPROACH

The first idea that came into our mind once we had cleaned the EMA data and verified its correctness was to predict the sequence of the tongue, lips, and jaw landmark positions from a speech signal by training a Sequence-to-Sequence model.

Sequence-to-sequence models have been widely used in natural language processing tasks, such as machine translation [109] and text summarization [139], and have also demonstrated their practicality for speech-related tasks like automatic speech recognition [11, 29, 69]. Using this approach, we aim to develop a model to input a speech signal and output a sequence of 3D landmark positions corresponding to the speaker's tongue, lips, and jaw movements. The EMA data provides accurate information on the position of the tongue and jaw, which can be used as a ground truth to train the model.

The main emphasis of this project is to develop a solution that generalizes well to out-of-domain input sources and that could be widely deployed without specialization or fine-tuning on a user-per-user basis. The flexibility of a pipeline

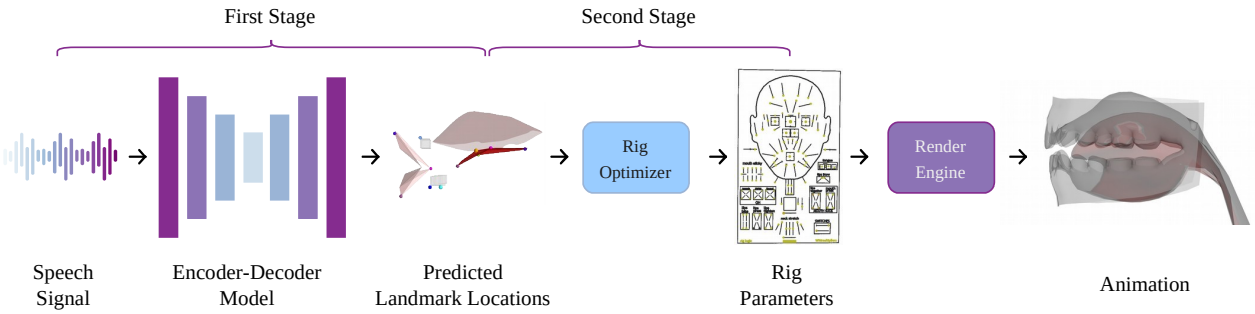


Figure 3.1: The proposed method consists of a two-stage pipeline. The first stage predicts the EMA landmark positions from the audio signal. In the second stage, the rig parameters are computed by minimizing the distance between the predicted landmark positions and their counterparts on the mesh through a Quasi-Newtonian optimizer. Finally, the rig parameters are used by a rendering engine to show the animation results.

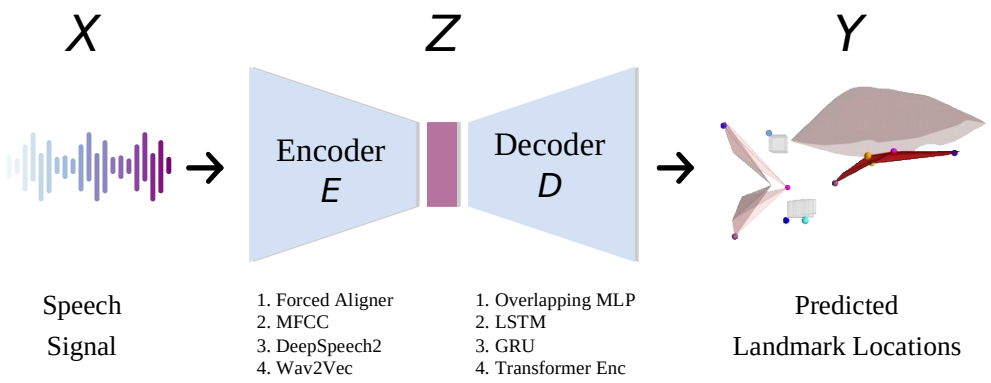


Figure 3.2: The diagram depicts the explored configuration space of an encoder-decoder architecture comprising encoders (E), latent representations (Z), and decoders (D). The encoder determines the definition of the latent space (Z), while the architecture of the decoder can be selected independently from the encoder.

solution would allow us to easily evaluate models that predict the landmark positions with different speech feature representations and select the one that best fits our needs. Additionally, we are interested in solutions that run in real-time with low latency for interactive applications.

3.1 Proposed Method: A Two-Stage Pipeline

Our proposed learning-based prediction pipeline consists of an encoder-decoder model followed by an optional rig-solving animation step as depicted in Figure 3.1. First, the input audio is encoded into a compressed latent feature representation by an *audio encoder*. Then a sequence of sparse landmark positions is predicted by an *articulation decoder*. Finally, these sparse points become the constraints of a *rig optimizer* module that identifies the optimal animation parameters to match corresponding mesh locations of the tongue, lips, and jaw on a rigged 3D model.

Formally, our dataset $\mathcal{D} = (\mathbf{X}, \mathbf{Y})$ is defined as the set of pairs where $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_n\}, \mathbf{x}_i \in \mathcal{X}$ denotes the set of audio input samples and $\mathbf{Y} = \{\mathbf{y}_1, \mathbf{y}_2, ..., \mathbf{y}_n\}, \mathbf{y}_i \in \mathcal{Y}$ is the corresponding sequence of EMA landmark positions. Each input audio $\mathbf{x}_i \in \mathbb{R}^T$ represents a one-dimension waveform consisting of T_i samples according to the duration of the audio and the sampling rate under which it was captured, while $\mathbf{y}_i \in \mathbb{R}^{S_i \times L \times 3}$ contains a series of S_i continuous frames of $L = 10$ 3D landmark positions.

As a first approach, we focus on finding the best model $E : \mathcal{X} \to \mathcal{Z}$ that encodes the input audio signal into a latent audio feature space $\mathcal{Z} \in \mathbb{R}^a$, where a is the dimensionality of the audio feature representation. The audio embeddings $\mathbf{z}_i \in \mathbb{R}^{S_i \times a}$ are later decoded by the articulation decoder $D : \mathcal{Z} \to \mathcal{Y}$ to predict a sequence of landmark positions $\mathbf{y}_i \in \mathbb{R}^{S_i \times L \times 3}$, expressed in the EMA stabilized pose space. Finally, the predicted landmark positions $\hat{\mathbf{y}}_i = D(E(\mathbf{x}_i))$ are mapped into the face mesh pose space $\mathcal{M}$ by applying a similarity transformation $\mathcal{A} : \mathcal{Y} \to \mathcal{M}$ resulting in the sequence of mesh constraints $m_i \in \mathbb{R}^{S_i \times L \times 3}$. A summary of the combinations of different encoders and decoders considered is shown in Figure 3.2.

3.2 Stage One: Landmark Motion Prediction from Audio

For the encoding stage, we explore different audio feature representations, ranging from traditional methods based on phonetic and frequency analysis representations, and explore more modern neural network-based features which were trained for ASR and for general purposes by means of SSL. Specifically, we explored the following audio features: phone representations, MFCC, DeepSpeech2, and Wav2Vec.

3.2.1 Audio Encoding

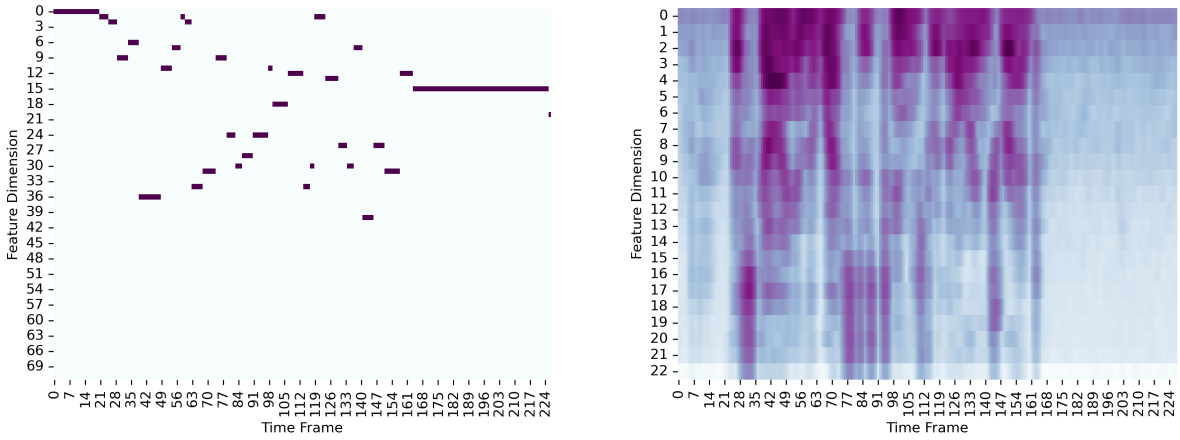
Phones: We perform a phonetic segmentation of the speech signal as used in [42, 90, 171]. We use phone representations that include the 39 phone representations from ARPAbet with the lexical stress variants of the actor's diction found in the dataset, yielding 79 phone representations.

MFCC: We also employ the widely used MFCC for speech processing tasks and convert the audio frequencies into a perceptually based logarithmic mel scale which is useful for characterizing human speech.

DeepSpeech2: We extract intermediate representations from the neural ASR model DeepSpeech2 (DS2) [6]. Specifically, we selected the output embedding from the Bi-LSTM layers as the latent audio feature representation to obtain embeddings with a higher generalization and avoid a bias towards the English character distribution learned at the pre-FC layer. In this way, we benefit from more generalized audio features that encode hundreds of hours of multi-speaker speech from the LibriSpeech dataset [113]. In this model, MFCCs are computed from the raw signal and serve as an input to a 5-layered bidirectional LSTM. The output passes through a fully connected layer, which classifies a character in the target language according to a Connectionist Temporal Classification (CTC) loss [55]. Then, the weighted vector from the CTC is passed to a beam search module, which assigns the most suitable transcription of the given speech signal. For speech-to-animation, we extract the features computed before the grapheme classification layers. This way, we use a generalized audio representation trained across speakers on hundreds of hours of speech from the LibriSpeech dataset [113].

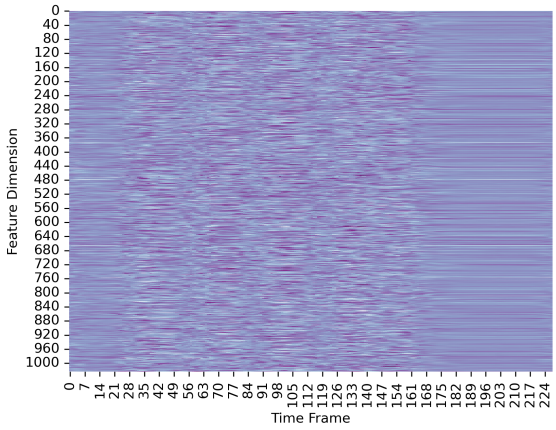
Wav2Vec-Z and -C: Wav2Vec [136] takes the raw audio waveform as input which is directly processed by two causal convolutional networks (CCNN). This SSL model was trained with the goal of obtaining a general representation of speech audio for any downstream application rather than a specific task, e.g., ASR. The input audio is fed into a CCNN that predicts a latent representation of the audio in the *z-features* (W2V-Z). A sequence of z-features in a larger window is then fed to the second CCNN to compute the contextual *c-features* (W2V-C).

The explored audio representations were visualized in Figure 3.3 as normalized heatmaps to obtain insights into their differences. An audio sample from the dataset, with the utterance *"The sky in the west is tinged with orange-red"*, was used to extract the audio features. The figure highlights the higher dimensionality and more complex patterns of the neural representations compared to the traditional methods. The changing patterns of all representations during the speech can be seen, as well as the almost constant pattern during the silent periods at the beginning and end of the audio.

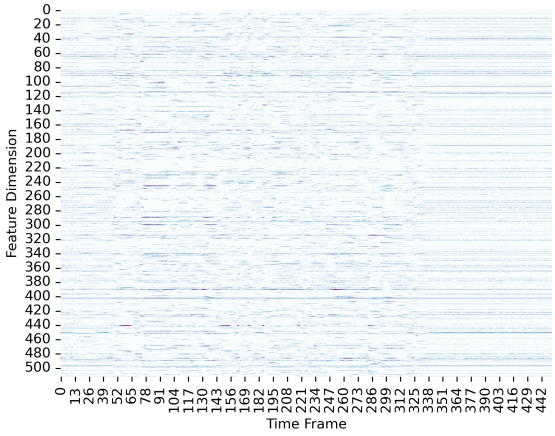


(a) Phones

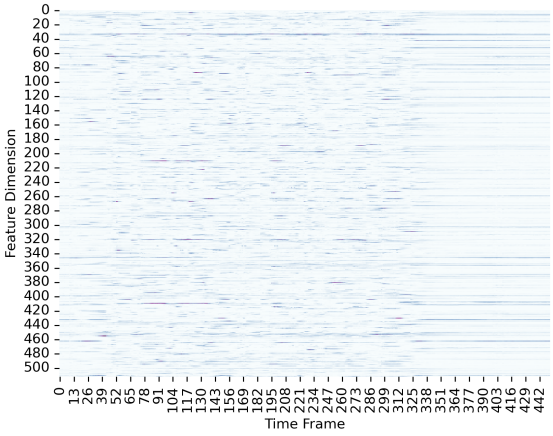
(b) MFCC



(c) DeepSpeech2



(d) Wav2Vec-Z



(e) Wav2Vec-C

Figure 3.3: Visualization of the audio features for audio sample 0734 with caption *"The sky in the west is tinged with orange-red."*. The heatmap colors are normalized for a proper comparison. The audio features explored on these experiments are: (a) Phones, (b) MFCC, (c) DeepSpeech2, (d) Wav2Vec-Z, and (e) Wav2Vec-C.

3.2.1.1 Articulation Decoding

Once the audio has been encoded into a latent space, the articulation decoder maps the sequence of encoded audio into a sequence of 3D speech articulator landmark positions. Different neural network architectures were evaluated for this task, from simple methods such as an Overlapping Multilayer Perceptron (MLP) [150] to models with higher complexity like Recurrent Neural Networks (RNNs) [62] and a Transformer architecture [154].

MLP: We implemented a simple sliding overlapping input and output window MLP as proposed by [150] with a slight modification. We enforce causal prediction by outputting the most recent value rather than predicting the middle frame. This avoids looking at information in the future and reduces the prediction latency of the model.

RNNs: The variations of LSTM and GRU tested in our experiments are: *a)* unidirectional, *b)* bidirectional, and *c)* their corresponding multi-staged variants with one, two and five layers.

Transformer: Our Transformer model [154] follows the work from [41] and [168] by neglecting the decoder layers and using only stacked encoder layers with Multi-head Self-Attention (MSA).

In our approach, the Transformer first encodes the position of the audio features in the sequence through learnable projection encodings. The positionally encoded audio embeddings serve as input to the Transformer Encoder layers. These layers are formed by a Multi-headed Self Attention (MSA) and an MLP, both starting with a Linear Normalization layer and ending with a residual connection. The MSA consists of multiple heads that compute in parallel a *scaled dot-product attention*, also known as the *Query-Key-Value* self-attention (QKV-SA), for the input sequence as described in Eq. 3.1. In a QKV-SA module, the score weights are based on pairwise similarities between the projected elements of the sequence of their query and key representations. The attention A requires a scaling factor to avoid vanishing gradients when the transformer embedding dimensionality d is large. The scaling factor is usually the size of the hidden dimension. The output of each head H in the MSA is the weighted sum of self-attention scores applied to the projected values of the sequence.

$$
\begin{aligned}
Q &= \mathbf{z}W_Q, \quad K = \mathbf{z}W_K, \quad V = \mathbf{z}W_V \\
A &= \text{softmax}\left(\frac{QK^T}{\sqrt{d}}\right) \\
H &= AV
\end{aligned}
\tag{3.1}
$$

3.2.2 Experiments

We perform an extensive evaluation of the audio feature encoding and articulation decoding architectures by training all combinations of encoding and decoding architectures.

Audio Encoding: In all of our experiments we down-sampled the audio to 16 kHz since DeepSpeech2 and Wav2Vec are trained for that specific sampling rate. DeepSpeech2 outputs an audio feature representation for every 20 ms in the audio. Therefore, we consider 20 ms as the common audio frame duration for the encoding of the input signal for all audio

encoding methods. This frame duration is also preserved in the articulation decoder networks and results in animation predictions generated at 50 frames per second.

We use the Montreal Forced Aligner [104] to obtain phone labels by aligning the transcripts from the recording session with the recorded audio. The resulting 72 class phone labels are sampled every 20 ms to make them comparable on a fairground with DeepSpeech2 and Wav2Vec features. Then, the sequence of phones extracted from the audio was one-hot encoded, to generate the sequence of phone representations.

Following [5] we compute MFCC features by separating the mel frequency spectrum into 27 bins with a Fast Fourier Transform on a 2080 Hz window, resulting in a sequence of 27-D feature vectors.

From DeepSpeech2, we extracted the 1024-D output from the 5-layered Bidirectional-LSTM (Bi-LSTM) and discarded the final English character classification layer.

In contrast, Wav2Vec's z- and c-features represent 10 ms of audio in a 512-D feature vector. To match the common frame duration, we concatenated two sequential feature vectors to represent 20 ms of audio in a 1024-D feature.

Articulation Decoding: All network architectures in this work were implemented using PyTorch [117]. The models were trained and tested by splitting the dataset of 1900 utterances into two groups: train and testing, in an 80/20 ratio.

We fine-tune the hyper-parameters of the models using the Hyperband algorithm [94] with Optuna [4]. In our experiments, we search for the optimal hidden layer size, learning rate, dropout rate, and the number of hidden layers. The search space for the different parameters is the following: hidden layer size [128 … 2048], learning rate [10^{-10} … 10^{-1}], and dropout rate [0.01 … 0.99]. For the MLP network, we search the number of hidden layers in the range [1 … 4].

The best set of hyper-parameters was selected by analyzing the results throughout all the architectures and selecting a set of reasonable values for all architectures to allow an equal comparison between the different audio encoders and articulation decoders. We utilize the mean squared error (MSE) over the predicted 3D landmark positions as the loss function for training. This decision was based on the fact that we are predicting ten 3D positions in the EMA reference space.

Model weights were optimized using the Adam optimizer [87] using $\beta_1 = 0.9$ and $\beta_2 = 0.99$. The initial learning rate was 10^{-5}, with a dropout rate equal to 0.25, and a batch size of 32. As the rest of the hyper-parameters, the best values were selected from the Optuna search results.

The MLP articulatory decoder consists of two 512-D hidden layers with a ReLU [3] activation function followed by a dropout layer [144] set to 0.25. A final linear layer predicts a linearized sequence of 3D positions of size $S \times L \times 3$. Where S is the output sequence length, L is the number of landmarks, and 3 for the x, y, and z axis in the EMA 3D space.

For the RNN models, both LSTM and GRU architectures were evaluated in unidirectional (forward in time) and bidirectional configurations. In all cases, we tested models with 1, 2, and 5 layers, and each RNN model outputs a

prediction for each 20 ms input feature.

We explored different configurations of Transformer depth and width, finding the best setup to be 4 encoder layers with 8 heads. The Transformer model is trained using Adam at an initial learning rate of 5×10^{-8}. A warm-up procedure using an L_1 loss was found to improve stability in the initial training stage. We then switched to an L_2 loss conditioned by an empirically determined threshold $\delta = 3$ to reach convergence.

The RNN and MLP models were trained for 40 epochs, while the Transformer models were trained for up to 100 epochs with an early stop criteria of 10.

All training samples were formed as windows of 15 audio feature representations in length (300 ms duration) sampled randomly from the training data. Audio input features were aligned with their corresponding EMA output landmark locations which were nearest-neighbor downsampled from 250 Hz to 50 Hz.

3.2.3 Quantitative Evaluation

In this section, we evaluate all the different configurations of audio feature encoders and articulation decoders. We define the sample error $e^{(i)}_{sample}$ as the L2-Norm of the estimated landmark positions with respect to the ground truth over the full sequence S_i, as shown in Eq. 3.2.

$$e^{(i)}_{sample} = \frac{1}{S_i} \sum_{s=1}^{S_i} \frac{1}{L} \sum_{l=1}^{L} ||\hat{y}^{(i)}_{s,l} - y^{(i)}_{s,l}||_2, \forall i. \tag{3.2}$$

Where $y^{(i)}_{s,l}$ denotes the position of the l^{th} landmark at time s for the i^{th} audio sample with duration of S_i audio frames, and $\hat{y}^{(i)}_{s,l}$ is the l^{th} landmark's position predicted by the articulation decoder at time s.

We present the results of our experiments in Table 3.1. This table provides a comprehensive overview of how well each model performed on the entire test set. We use the temporal mean sample error in mm as our primary metric to evaluate the overall performance of each model as described on 3.2. This metric captures the average distance between the predicted and ground truth through all the timesteps in the sequences of landmark positions, providing a clear and concise summary of each model's performance. Additionally, Table 3.1 breaks down the results by articulation decoder (rows) and audio feature encoder (columns), allowing us to compare the performance of different models and feature representations. This allows us to gain a deeper understanding of how the different components of our pipeline are impacting the overall performance of our models.

Analyzing the results, we see an improvement in the performance of the MLP architecture by widening the context of the input window and the output window. This architecture version is comparable to a single-layer GRU and LSTM network. However, the number of network parameters required for the MLP is greater when compared to RNN-based counterparts. The GRU architecture shows a slight improvement over the LSTM architecture, as seen in [12], due in part to the smaller amount of parameters in each layer, making it less susceptible to over-fitting.

Based on these results, we can appreciate how all the articulation decoders we presented can learn how to predict the pose to a reasonably low error. Notably, the LSTM and GRU models' performance improves as we increase their complexity

Table 3.1: Model architecture evaluation using different audio feature representations: Phonemes (Phone), MFCC, DeepSpeech2 (DS2), Wav2Vec c- (W2V-C) and z- (W2V-Z) features. Models were trained with 300 ms input windows of audio. The error is the temporal mean L2-norm in mm calculated through the test split. The number of parameters reported is the number of trainable parameters per architecture design. The inference time is the mean time over the test split measured as ms per one second of audio input.

Decoder\Feature	Phone	MFCC	DS2	W2V-C	W2V-Z	Num. Params	Inference [ms]	Latency [ms]
MLP 15:5	2.445	2.075	2.393	1.959	1.937	6.62×10^7	0.232	300
LSTM-1L	4.207	2.344	2.269	2.047	2.140	3.17×10^6	1.150	20
LSTM-2L	4.209	2.178	4.206	1.990	4.212	5.27×10^6	2.238	20
LSTM-5L	2.656	2.037	2.264	1.999	1.960	1.16×10^7	5.432	20
Bi-LSTM-1L	3.664	2.346	2.375	2.373	3.481	6.33×10^6	2.229	300
Bi-LSTM-2L	4.577	2.109	2.844	2.188	3.874	1.26×10^7	4.512	300
Bi-LSTM-5L	4.365	**1.912**	2.218	1.927	2.929	3.15×10^7	11.000	300
GRU-1L	4.150	2.290	2.250	1.949	2.071	2.38×10^6	1.144	20
GRU-2L	2.623	2.117	2.179	1.897	1.980	3.95×10^6	2.193	20
GRU-5L	2.661	2.006	2.184	1.916	1.954	8.68×10^6	5.339	20
Bi-GRU-1L	4.405	2.368	2.529	2.055	2.613	4.76×10^6	2.290	300
Bi-GRU-2L	3.143	1.953	2.947	1.932	2.513	9.48×10^6	4.439	300
Bi-GRU-5L	**2.341**	1.973	**2.058**	**1.757**	**1.784**	2.37×10^7	10.955	300
Transformer	2.368	2.283	2.168	1.935	1.942	5.045×10^7	3.515	300

by increasing the number of layers. Furthermore, our results show that bidirectional GRU and LSTM models' learning capability improves as they can look ahead in the sequence.

Audio Encoding: The results of our experiments show that deep-learning based audio features and MFCCs perform better than phone-based features, regardless of the choice of neural network architecture. While MFCC audio features exhibit better quantitative evaluation results compared to DS2, DS2 and W2V features demonstrate superior qualitative performance when generalizing to input speech from out-of-domain speakers. The video provided on the following link visually demonstrates these results, where we can observe that the DS2 and W2V features outperform the other audio features when dealing with out-of-domain speakers.

Both Wav2Vec feature variants show similar behavior, although the layered RNN architectures take more advantage of the c-features. Furthermore, there is a substantial improvement when moving from a 2-layer to a 5-layer version of the architecture in both the unidirectional and bidirectional versions of the RNNs. The best architecture from a test-set perspective consists of encoding the audio with Wav2Vec c-features and estimating the landmark positions using a bidirectional 5-layered GRU.

The same set of experiments was replicated with a training input window of 1000 ms as summarized in Table 3.2. The results are consistent with the results from training models using only a 300 ms input window. The overall performance slightly improved across all the models at the cost of a longer inference time, latency, and the number of parameters.

Table 3.2: Model architecture evaluation using different audio feature representations. Models were trained with 1 s input windows of audio. The error is the temporal MSE in mm calculated through the validation split. The number of parameters reported is the number of trainable parameters per architecture design. The inference time is the mean time over the validation split measured as ms per second of audio input.

Decoder \ Feature	Phone	MFCC	DeepSpeech2	W2V-C	W2V-Z	Num. Parameters	Inference [ms]
MLP 50:15	2.315	2.157	2.619	1.957	1.928	3.29×10^8	0.309
LSTM-1L	2.657	2.299	2.350	2.048	4.219	3.17×10^6	1.150
LSTM-2L	4.216	2.219	2.342	2.016	2.026	5.27×10^6	2.238
LSTM-5L	2.609	2.133	2.331	2.014	1.994	1.16×10^7	5.432
Bi-LSTM-1L	3.355	2.074	2.204	1.977	2.272	6.33×10^6	2.229
Bi-LSTM-2L	2.268	1.874	2.096	1.825	1.781	1.26×10^7	4.512
Bi-LSTM-5L	2.247	1.732	1.987	1.754	1.708	3.15×10^7	11.000
GRU-1L	4.195	2.213	2.283	1.943	2.001	2.38×10^6	1.144
GRU-2L	2.559	2.098	2.248	1.905	1.945	3.95×10^6	2.193
GRU-5L	2.570	2.003	2.235	1.908	1.943	8.68×10^6	5.339
Bi-GRU-1L	4.304	1.964	2.094	1.828	1.910	4.76×10^6	2.290
Bi-GRU-2L	2.206	1.784	2.091	1.744	1.714	9.48×10^6	4.439
Bi-GRU-5L	**2.179**	**1.660**	**1.935**	**1.684**	**1.648**	2.37×10^7	10.955
Transformer	2.349	2.393	2.139	1.926	2.044	5.049×10^7	3.552

Overall, the model trained with phonetic representation shows the highest error, followed by the model trained with DeepSpeech2 audio features. The models trained using MFCC and both Wav2Vec features follow a similar pattern across all the landmark predictions.

Landmark Error Analysis: To understand the performance of our model in predicting speech articulator landmarks, we analyzed the error of the predictions across all audio representations and summarized the results in Figure 3.4. The results show that the model performed best when predicting the position of the upper lip (UL), with an average error of 1.16 mm across all the audio encodings. This can be attributed to the fact that the upper lip's motion is not as wide as the tongue tip (TT), which had the least accurate predictions with an average error of 2.26 mm through all the audio feature representations.

In general, most of the error comes from predicting tongue positions (TD, TB, BR, BL & TT) and lower lip (LL) landmarks. These points experience the most significant motion during an utterance, making them challenging to predict accurately. Surprisingly, the jaw landmarks (LI, LJ) showed lower error, likely due to the jaw having a more rigid motion that occurs at a slower pace when compared to the elastic motion that the tongue and lower lip present in most utterances.

3.2.4 Qualitative Evaluation

To verify the results described in Table 3.1, we invite the reader to watch the supplementary video provided through this link for a visual comparison of the landmark sequence predictions are visible from a lateral view for the Overlapping MLP 15:5, 5-layered Bi-LSTM, 5-layered Bi-GRU, and Transformer Encoder models trained with the proposed audio encoders.

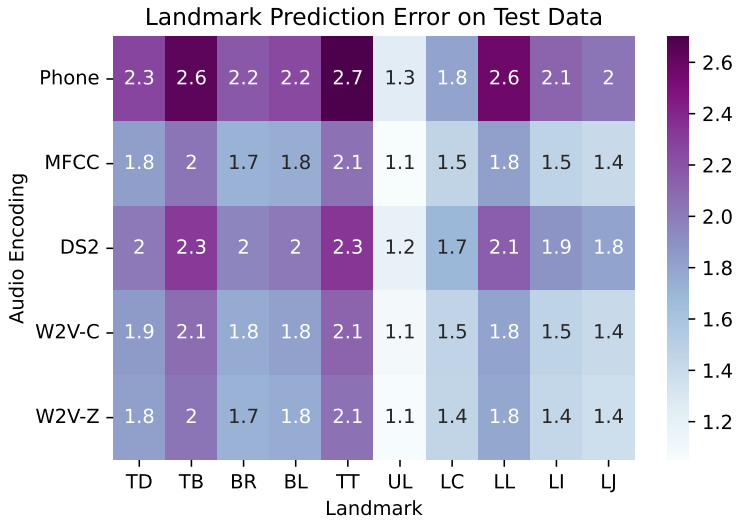


Figure 3.4: Landmark Prediction Error on the test set of the bidirectional 5-layered GRU across all audio encodings. The error values shown are the temporal mean sample error in *mm* for each landmark.

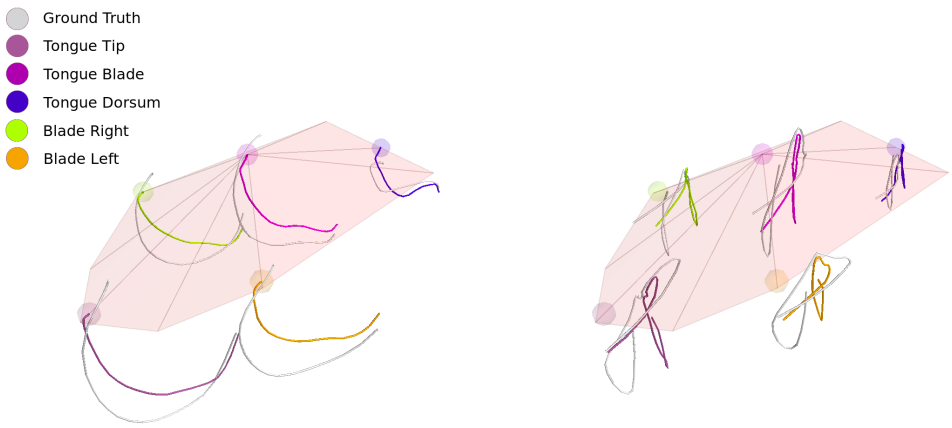
We found that all models have convincing performance while predicting *in-domain* audio from our dataset for both training and test samples. However, phone-based features did not perform as well as the other models with *out-of-domain* data, such as audio from other actors while they were speaking or even singing, as they require an ASR step and forced alignment, which is dependent on the language and prosody. In general, the DeepSpeech2 and Wav2Vec-based models performed similarly when the decoder was a multi-staged RNN or a Transformer model. In our analysis, we observe the architecture where the audio is encoded with Wav2Vec z-features and decoded into landmark positions through an MLP 15:5 shows compelling tongue motion results but exhibits temporal jitter on the predicted jaw motion. In general, the models with an MLP decoder present a certain level of jitter, which is not present in the models with an LSTM or GRU decoder.

We have provided a video demonstration to supplement the evaluation results presented in Figure 3.4. Analysis of the traces of the predicted motion of the landmarks, as shown in Figure 3.5a, reveals that the model is capable of capturing the coarse motion patterns of the articulator landmarks. However, in Figure 3.5b, we can see that the model learned to take shortcuts on fast and complex motions that take place within a small space. This occurs due to the model being trained only on a Mean Square Error loss, which leads the predicted sensor position to a local minimum. These findings highlight the need for future work to address these issues by adding more constraints to the learning loss. In general, the predicted motion shows an overall shift towards the front of the mouth. This shift can be attributed to the model's reliance on the initial position of the tongue at the beginning of the sequence.

3.3 Stage Two: Parametric Face Model Optimization

3.3.1 Optimizing the Rig Parameters

To demonstrate re-targeting to a final animation output rig, we use a high-quality artist-designed FACS-based [34, 45] 3D face model. The model and animation rig conforms to the MetaHuman standard [47] and closely resemble the actor. The output is represented as a triangle mesh $M = (V, F)$ defined by a set of vertices V that build the mesh faces F. The



(a) Long Motion **(b)** Prediction Shortcut

Figure 3.5: Visualization of 300 *ms* of predicted motion on a test sample vs. ground truth. Ground truth is in gray, and the predicted motion is in color according to the sensor. (a) Shows an example of a predicted long motion. (b) Shows an example of motion shortcuts on prediction visible on *Blade Left* and *Tongue Tip* sensors. We suggest watching the video to obtain a more comprehensive evaluation.

face model is controlled through a P-dimensional control parameter vector θ that deforms the mesh in a differentiable manner for any given frame in the animation. In our model, $P = 173$ for the whole face, of which nine parameters control the tongue and 12 move the jaw. We define $M(\theta_t)$ as the mesh posed by these control parameters for frame t. To estimate the pose of the mesh based on the predicted landmarks of the tongue, jaw, and lips, we first manually pre-define a correspondence between the transformed predicted landmark positions on the mesh coordinate system $\mathbf{v}_{\hat{\mathbf{y}}}^{(i)}$ to a set of points on mesh M as $C_t = \{f_l, b_l\}_{l=1}^{L}$, where $f_l \in F$ is a triangle index and b_l a barycentric coordinate defining a point on the triangle for the l^{th} landmark. In our case, $L = 10$ since we have five sensors on the tongue, two on the jaw, and three on the lips. These locations are shown in Figure 3.6.

We perform an initial alignment of the data from the EMA sensors to the mesh in a neutral pose by calculating the best similarity transform A that maps the points $\mathbf{v}_{\hat{\mathbf{y}}}^{(N)}$ into correspondences $C(\theta_N)$ for the neutral pose N described by parameters θ_N. While the geometry of our 3D face model is based on a 3D scan of the actor, the face geometry is not a perfect reconstruction. The teeth and tongue are adapted by an artist from a generic model and do not precisely align. To account for these minor differences, we calculate relative offsets $\delta_l = A(\mathbf{v}_{\hat{\mathbf{y}}}^{(l)} - \mathbf{v}_N^{(l)})$ between each landmark and the surface of the neutral 3D model.

The energy between the predicted landmarks at frame t and its corresponding point on the mesh is defined as:

$$e_{pose}(\theta_t) = \sum_{l=1}^{L} ||C(\theta_t)_l - (C(\theta_N)_l + \delta_l)||^2. \tag{3.3}$$

Our input data is sparse and asymmetric. There are three markers for the lips that capture the right side of the face and two for the jaw that cover the left side. For this reason, we enforce symmetry on both sides of the face on our parameter

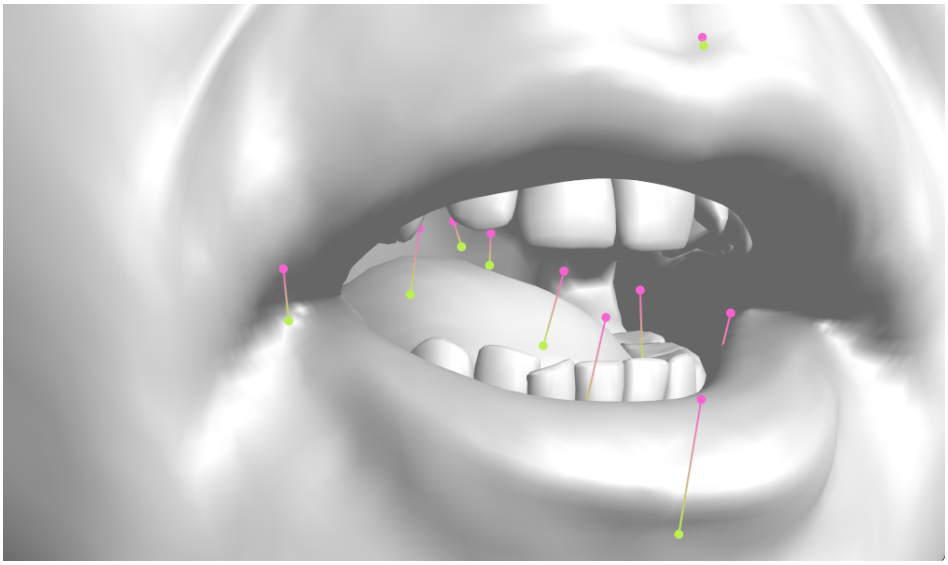


Figure 3.6: FACS-based face model landmark correspondence visualization before optimization. Green: mesh surface landmarks $C(\theta_t)_l$; Pink: target constraints $C(N)_l + \delta_l$ generated by the articulation decoder.

vector. In addition, we add an L1 regularization to our solver to ensure sparse activation for the model parameters:

$$e_{prior}(\theta_t) = \sum_{l=1}^{L} |\theta_l^{(t)}|. \tag{3.4}$$

Resulting in the following combined energy function:

$$e(\boldsymbol{\theta}) = e_{pose}(\boldsymbol{\theta}) + \alpha\ e_{prior}(\boldsymbol{\theta}). \tag{3.5}$$

We minimize $e(\boldsymbol{\theta})$ using an L-BFGS optimizer and initialize the parameters θ_t for frame t using the parameters from the previous frame θ_{t-1} for all T frames in the animation. The prior weight is $\alpha = 0.01$ for all results shown in this paper. Finally, no additional temporal smoothness priors were included in the optimization.

3.3.2 Results

We found that in the final animations on the mesh, the lips open and close correctly, which is remarkable considering that only three landmarks drive the mesh. No significant anomalies were observed in the motion of the jaw and tongue. We also compared the ground truth 3D landmark visualization against predicted landmark positions and ground-truth video. Frames from the generated animation are shown in Figure 3.8.

To comprehensively evaluate the effectiveness of our proposed pipeline, we tested the model with out-of-domain voices that were not included in the training phase. The test involved using markedly distinct out-of-domain sentences with different tempos, pitches, and tones to challenge the system further. We were pleased to observe that the model successfully generated animation results that closely corresponded to the audio input. We invite the reader to watch the supplementary video found on this link, and a sample frame from this video can be seen on Figure 3.7.

Figure 3.7: Sample frame from the final results video of our approach. It shows how we can animate an Epic Games' MetaHuman character using in- and out-of-domain audio samples.

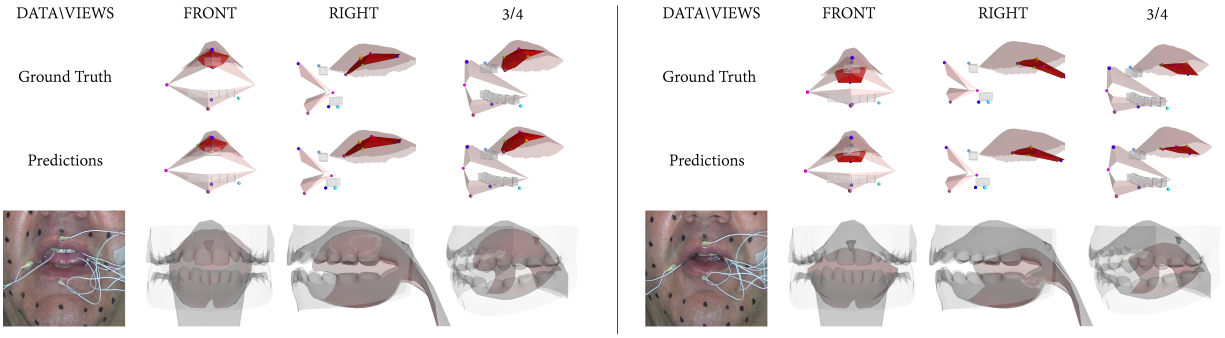
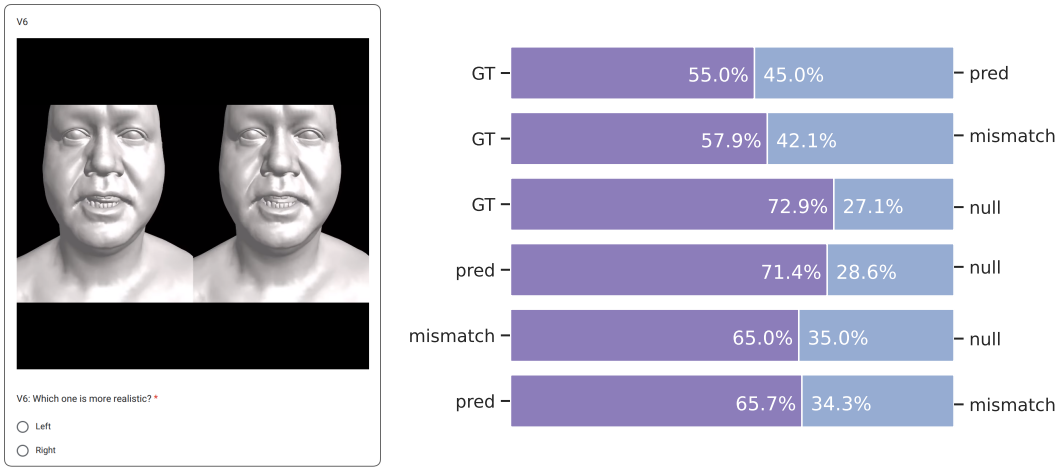


Figure 3.8: Visualization of two frames from test samples. On the first row, we see the ground truth landmark locations. The second row displays the predicted landmark locations. The third row shows the corresponding frame from the video reference and the solved animation frame.

We went further to test our model with singing audio samples, which the decoders were not trained on and were pleasantly surprised to see that the model could still generate plausible animation results. This robustness indicates that the proposed pipeline is capable of effectively capturing subtle nuances in voice inputs and translating them into accurate facial animations. These results demonstrate the potential of our approach to accurately animate a range of speaking styles beyond those seen during training.

3.3.3 User Study

We conducted a pairwise perceptual user study in which participants evaluated the quality of the generated animations using ten test sentences that ranged from visible to partially visible tongue motions. For each sentence, we created four animations using ground truth (*GT*), our best model's prediction (*pred*), a mismatch between lip and tongue animation (*mismatch*), and a nullified tongue motion from ground truth samples (*null*). We presented six pairwise comparison videos per sample, resulting in a total of 60 videos evaluated by 15 users. The participants were asked to choose the most realistic animation from each pair, as shown in Figure 3.9a. The study's results indicated that most users preferred



(a) Survey Inteface

(b) Results Barplot

Figure 3.9: Pairwise preference perception user-study. Left: The user interface of the survey shown to the user for a given sample. Right: User study summarized results where *GT*: Ground-truth animation; *pred*: predicted animation; *mismatch*: tongue mismatched animation; *null*: nullified tongue animation. GT was preferred overall, but the predictions from our model were preferred over mismatched animations and nullified tongue motion animations.

ground truth animations over others. Interestingly, users preferred an animated tongue over nullified tongue motion, even if the tongue's motion did not match the spoken sentence. The study also showed the consistency of our estimations, as users preferred our model's predictions over the mismatch animations. The study results are presented in Figure 3.9.

3.4 Limitations

The generated results prove that the dataset is limited in speech variability and expressiveness. Loud sounds, such as angry shouting in an angry manner, do not show a full extension of the jaw as one would expect from a preconception of the sound. A possible solution to this limitation is to diversify the data by further expanding our data and capturing speech and non-speech utterances with a wide range of emotions from more than one speaker by including actors of different genders and ages.

Finding the initial correspondence of the EMA landmarks in the 3D model is a one-time manual and a rather tedious process. Automating the alignment is a problem yet to be solved. This would help save hours of manpower when done with multiple characters as more data is captured from more actors.

While we can animate the lips and mouth using the ten landmarks captured to create our dataset, the expressiveness of these regions is limited by the 3D spatial sparsity of our data. As we only count with three landmarks for the lips, two are glued to the jaw, and five to the upper surface of the tongue. Unfortunately, adding many more sensors into the mouth is impossible as the passive electromagnetic sensors might interfere between them, and it has the potential to impede the actor's ability to speak clearly. Further research is required to be able to capture the full deformation of the tongue's surface at a better resolution with a high sample rate.

The current optimization stage of our model could be a slow process due to the estimation of the inverse Hessian matrix while optimizing for the best rig animation parameters per frame using L-BFGS. To address this limitation, a possible

solution is to train an end-to-end model that can predict the rig parameters directly from the raw audio signal. This would reduce the time needed for optimization.

CHAPTER 4

END-TO-END SPEECH ANIMATION

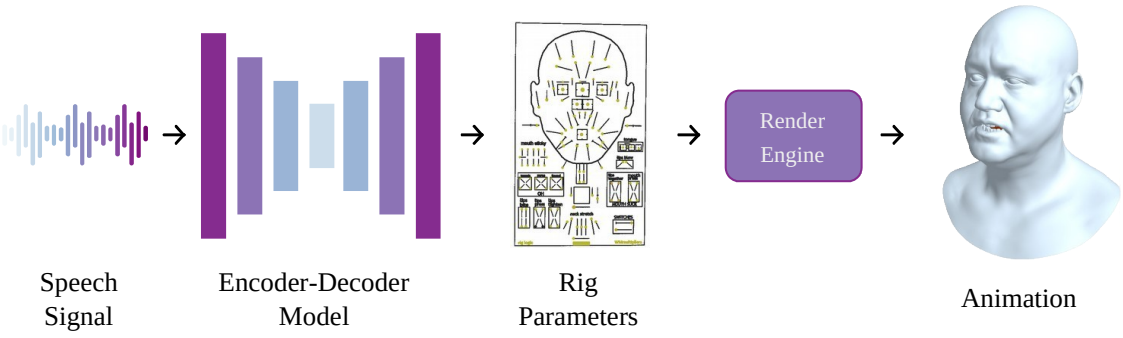
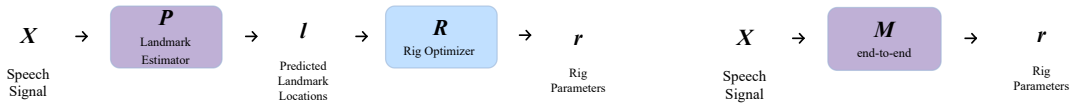


Figure 4.1: End-to-End model estimates the sequence of animation parameters directly from raw input speech.

In Chapter 3, we proposed a two-stage pipeline approach that successfully produced realistic animations. However, its processing speed is insufficient for real-time animation from an audio stream, which limits its practicality. To overcome this limitation, our next objective is to design a model that generates more accurate and realistic animations and does so more swiftly. We propose an end-to-end solution based on a Recurrent Neural Network architecture to achieve our task at hand. This model will be trained in an autoregressive scheme where the audio signal is mapped directly into the corresponding sequence of rig parameters required for generating the animations, as shown in Figure 4.1.

Adopting an end-to-end approach to the problem offers a key advantage: it mitigates the error propagation observed in the two-stage pipeline, as noted by [31]. This pipeline is illustrated in Figure 4.2a. Initially, the first stage, P, processes the speech signal X to predict the lips, tongue, and jaw landmark locations, l. This prediction incurs an error, ϵ_P, inherent to P (eq. 4.1). The predicted landmarks, l, are subsequently input into the rig optimizer, R, which computes the optimal set of rig parameters, r. This computation introduces an additional error, ϵ_R, native to the L-BFGS optimizer. However, the preceding error, ϵ_P, from the landmark predictor is further amplified by a constant, γ, as detailed in eq. 4.2. This augmentation raises the upper bound of the two-stage pipeline by a factor of γ relative to the error from the landmark



(a) Two-stage pipeline **(b)** End-to-End model

Figure 4.2: Comparison between first approach and end-to-end model

predictor, as outlined in eq. 4.3.

$$\mathcal{P}(x) = l + \epsilon_P \tag{4.1}$$

$$\mathcal{R}(\mathcal{P}(x)) = r + \gamma \cdot \epsilon_P + \epsilon_R \tag{4.2}$$

$$\Omega(\mathcal{R}(\mathcal{P}(x))) = \gamma \cdot \epsilon_P + \epsilon_R \tag{4.3}$$

In contrast, the training of an end-to-end model M, as portrayed in Figure 4.2b, utilizes a dataset D_M comprising pairs of corresponding audio samples x and sequences of rig parameters y (eq. 4.4). The rig parameters are computed through the rig optimizer R from captured ground truth landmarks. Consequently, the upper bound of uncertainty is limited to the error from the optimizer ϵ_R (eq. 4.6). This illustrates the end-to-end approach's potential to curtail the pipeline's overall error.

$$\mathcal{D}_M = \left\{(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), \dots\right\} \tag{4.4}$$

$$y^{(i)} = \mathcal{R}(l_{gt}^{(i)}) + \epsilon_R \tag{4.5}$$

$$\Omega(\mathcal{M}) = \epsilon_R \tag{4.6}$$

Moreover, given the recent advancements and optimizations in deep learning frameworks such as PyTorch [117] and ONNX [111], the resulting network is expected to provide faster inference times than the quasi-Newtonian optimizer previously employed.

This chapter comprehensively explores an end-to-end model to generate animation parameters from speech. We begin with a discussion of the IMT'22 dataset augmentation, where we incorporated visual information into the Electromagnetic Articulography (EMA) data, introducing the new IMFT'23 dataset. This enriched dataset provides the basis for training an end-to-end model capable of realistically animating facial and tongue movements.

We next focus on the model's design, specifically examining the effect of informing the model about the error of the velocity and acceleration of the rig parameter. We also consider new audio feature representations to see if Transformer-based models trained on thousands of hours of diverse audio can enhance our model's performance beyond Wav2Vec. As these models are built with a series of Transformer Encoder blocks, we look into the impact each of these internal layers has on generating better speech animations. We also test whether informing the network with phonetic information via multi-task learning can improve the creation of realistic speech animations. All concepts are thoroughly tested to find the most effective model configuration through a diverse set of metrics and evaluated by humans through a series of user studies.

Finally, as we generate more realistic animation, we propose a new way to evaluate speech animation models through visual speech recognition (VSR) or lip-reading models. For our evaluation purposes, we employ a state-of-the-art VSR model to measure the performance of our end-to-end from an automatic perception focus.

4.1 New Data Generation

In our initial approach, detailed in Chapter 3, the facial animation driven solely by the EMA-predicted landmarks resulted in pronounced tongue motion but lacked natural facial expressions, diminishing the effectiveness of the continuous tongue animation. To address this limitation, we leveraged the visual landmarks placed on the actor and the stereo vision recording obtained during the EMA capture session. These visual landmarks, as well as the lip contours, were tracked using an industrial-grade tracking service provided by Cubic Motion [108]. The tracking results for the frontal and lateral views are presented in Figure 4.3a and Figure 4.3b, respectively.

During the tracking process, we observed that a few landmarks exhibited noise due to occlusion issues. The articulograph device occasionally occluded the upper left facial landmarks, and the cables of the mouth interior sensors occluded some landmarks near the left lip corner. To ensure a more accurate mapping from the 2D image space to the 3D space, these noisy landmarks were disregarded during the subsequent 3D reconstruction process.

4.1.1 Stereo Reconstruction

Once the 2D landmarks were fully tracked in each video frame, we performed stereo reconstruction of the face and lips using the Direct Linear Transformation (DLT) method [1]. The DLT approach has been well-established and proven to capture human-related motion accurately [137]. To ensure the precision of the reconstruction, the video frames were fully synchronized, as indicated by the timestamps displayed atop Figures 4.3a and 4.3b at the frame level. The video was captured at a frame rate of 60 FPS.

Triangulation through DLT consists of estimating the 3D coordinates of points in a scene, given their 2D projections in two or more images captured from different viewpoints. The following equation can describe the relation between the 3D world points and their 2D image projections:

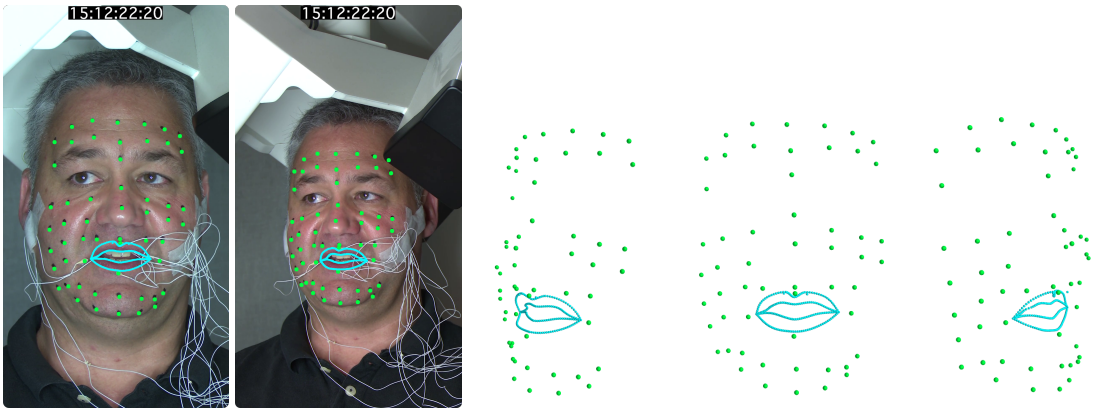
$$\mathbf{p} = \mathbf{PX} \tag{4.7}$$

Where $\mathbf{p}$ is the homogeneous coordinate of the 2D image point, $\mathbf{X}$ is the homogenous coordinate of the 3D world point, and $\mathbf{P}$ is the 3×4 camera projection matrix.

DLT provides a way to determine the camera projection matrix $\mathbf{P}$ from a set of known 3D world points and their corresponding 2D image projections. Given a sufficient number of point correspondences of at least 6 points, one can set up a system of linear equations to solve for the 12 unknowns in $\mathbf{P}$.

Once the camera projection matrices for both cameras have been obtained, it is possible to triangulate the 3D position of a point given its 2D projections in each camera frame. The core concept of triangulating is to find the intersection of the rays passing through the camera centers and the 2D image points.

This process is reiterated for every set of corresponding face and lips 2D landmarks. As a result, a 3D point cloud is created, representing the reconstructed face. Examples of these stereo-reconstructed landmarks can be viewed in the figures provided. Figure 4.3c showcases the right view of the 3D face and lips landmarks. Similarly, Figure 4.3d presents a frontal view, and Figure 4.3e displays the left view of the same landmarks.



(a) Front Tracking **(b)** Lateral Tracking **(c)** Right View **(d)** Front View **(e)** Left View

Figure 4.3: Facial Landmark Tracking and Reconstruction. Throughout the session, the facial landmarks and lips were meticulously tracked on both cameras using a state-of-the-art industrial-grade tracker. The obtained 2D landmarks were then accurately reconstructed in three-dimensional space utilizing the Direct Linear Transformation (DLT) triangulation.

4.1.2 Visual landmarks and EMA alignment

Following the reconstruction of facial landmarks, we manually identified corresponding barycentric points on the mesh. This process mirrored the methodology outlined for the tongue in the rig optimization (Section 3.3.1). Initially, we computed the optimal similarity transform, denoted as A_F, for the facial landmarks, utilizing the same neutral pose (N) considered for the EMA sensors. Subsequently, we mapped the 3D facial landmarks, including those associated with the lips, represented as $\mathbf{v}_F^N$ in the stereo reconstruction space, to their respective correspondences, denoted as $C(\theta_N)$, on the neutral pose (N) of the mesh (M). These correspondences were determined by the set of rig parameters (θ_N). Once the facial landmarks had been successfully mapped into the mesh space, we optimized the mesh using L-BFGS using the information obtained from the face, lips, and tongue correspondences.

Unlike our previous approach, the number of landmarks varied between the face and the lip contour. Specifically, we considered 51 landmarks for the face and 17 landmarks for the lip contour. This augmentation resulted in a total of 79 points ($L = 79$), including the 11 EMA landmarks. At this time, we considered the sensor placed on the incisor as an alignment constraint. While the L1 regularization term remained unaltered with the same (α) weight across all the landmarks, the lip contours served as 2D constraints as the error in their 3D reconstruction arises from the absence of distinguishable features along the contour, leading to discrepancies in the identification of landmarks between both camera views.

4.1.3 IMFT'23 dataset

After processing the visual features by tracking, stereo reconstructing, mapping, and matching the facial landmarks on mesh space and optimizing the mesh, we were able to create a new dataset named IMFT'23 formed by corresponding groups of audio samples, video samples, 2D landmarks on video frame space, 3D landmarks of the face, lips, and tongue

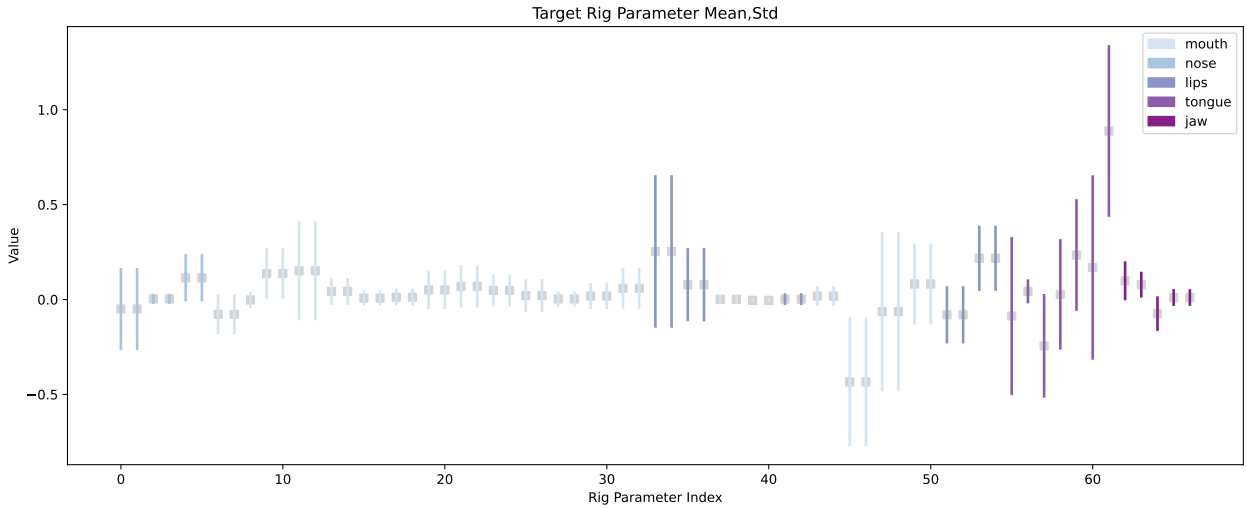


Figure 4.4: The mean and standard deviation of the 67 rig parameters. The parameters are grouped and color-coded according to their function: tongue, lips , nose, jaw, and the rest of the mouth. No abnormal values are observed across the parameter set.

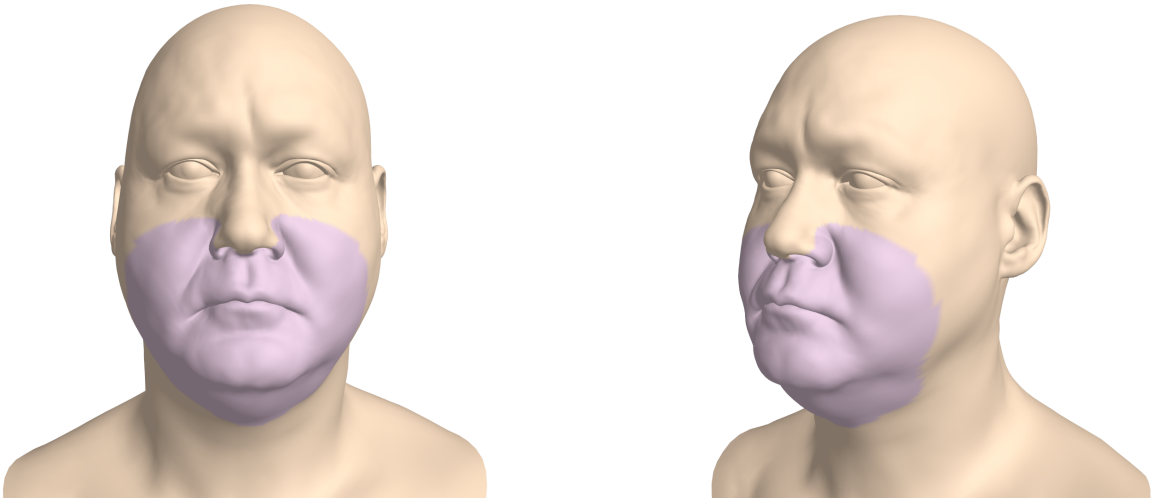
on mesh space, and sequences of rig parameters θ_i^k of the mesh M for each frame i of each sample k. Different from the IMT'22 dataset, the number of clean samples was reduced from 1902 to 1700 since the EMA session was not recorded with video in its entirety, resulting in a total duration of 2.28 hours of matching audio at 16 kHz with sequences of 2D and 3D features at 50 FPS.

The head mesh used in this study was derived from a high-quality 3D capture of the actor, comprising a total of 30,881 vertices. These vertices are manipulated by 173 rig parameters defined for the Epic Games MetaHuman model [47]. Each parameter is associated with a specific semantic meaning, such as chin position (up and down), left lid opening and closing, mouth opening and closing, etc. In this work, we decided to focus on designing an end-to-end model capable of generating animations as a sequence of rig parameters from the audio due to the fact that complex motions such as the ones present while coarticulating are compressed into a few floating values. By working with a parametrized model also enables makes the model capable of transferring the motion from our actor's model to other 3D characters that follow the same rigging design. Moreover, given the focus of our research on generating realistic articulations based on the speech signal, our research primarily targets the lower portion of the face and the inner mouth. As such, we opted to predict only a subset of 67 parameters that modify the position of 19,847 vertices, even though optimization could be performed for all 173 parameters.

In Figure 4.4, the mean values and standard deviations of these selected parameters are presented, where the parameters are grouped by their associated facial features and color-coded for clarity. As we can see, the parameters are found mainly in the $[-1, 1]$ range as this is the design of the parameters for a MetaHuman character rig. Interestingly, the tongue parameters exhibit higher mean values and greater variability when compared to the other groups. The tongue parameters group is followed, in descending order of value and deviation, by parameters for the lips, the nose, the jaw, and the remainder of the mouth. This detailed visual representation aids in our understanding of the behavior and distribution of these important parameters in face rigging.

In this work, we decided to focus on estimating the parameters that control the lower part of the face and the inner-mouth

elements such as the jaw and tongue. This subset encompasses approximately 64% of the total amount of vertices illustrated in Figure 4.5. One of the main goals of generating animations is to communicate regardless of the realism infused into it, as described by Parke and Waters [115]. Therefore, by only controlling the parameters that modify the regions of the face that present the coarticulatiion, empowers artists to exert control over the upper section of the face to convey emotions effectively, and we leave out from our target controls that allow the user to add more expresiveness to the motion of the mouth and cheeks. Moreover, we leave the control of upper face features such as gaze, eyebrows, and head motion out of scope from this work since their study requires a more comprehensive understanding of contextual information than the speech signal by itself.



(a) Front view **(b)** 3/4 view

Figure 4.5: Region of the actor's mesh influenced by the 67 rig parameters utilized for articulation animation, constituting this research's focal point.

For our final goal, we will train the end-to-end model $\mathcal{M}$ to take as input the audio signal $\mathbf{x}$ and predict the corresponding sequence of rig parameters θ that pose the mesh described as $\theta = \mathcal{M}(\mathbf{x})$.

4.2 Baseline Model

Our proposed framework is grounded on the encoder-decoder architecture, which has shown its effectiveness in the domain of speech animation. The distinctive aspect of our methodology is the transition from predicting EMA landmark locations to predicting a sequence of rig parameters, which drives the generation of animations.

Formally, our model processes an input audio signal $\mathbf{x}$, comprising N samples. The audio encoder model, denoted ENC, transforms this input into a latent space $\mathcal{Z}$ of dimensionality F, resulting in a sequence of latent representations $\mathbf{z} = [z_1, z_2, ..., z_S]$, where $\mathbf{z} \in \mathbb{R}^{S \times F}$.

Post the encoding process, a decoder model, denoted DEC, maps these latent representations of the audio features $\mathbf{z}$ into a sequence rig parameters $\mathbf{r} = [r_1, r_2, ..., r_S]$, where $\mathbf{r} \in \mathbb{R}^{S \times 67}$, representing the 67 parameters that animate the lower face of our target model.

These encoding and decoding processes can be formally described as:

$$\mathbf{z} = \textsc{Enc}(\mathbf{x}),$$
$$\mathbf{r} = \textsc{Dec}(\mathbf{z}).$$

The baseline model in our research employs Wav2Vec as the encoder ENC and a Gated Recurrent Unit (GRU) [30] as the decoder DEC. This model uses the encoder-decoder framework for rig parameter prediction, following recent trends in speech animation research [106, 119], which result in generalizable, reliable, and fluid animations.

4.3 New Audio Feature Representations

In the preceding chapter, our findings substantiated the superior performance of neural representations like Wav2Vec [136], particularly when compared to traditional forms of speech signal representation, such as MFCC and phone representations as the ones extracted from tools such as the Montreal Forced Aligner [104]. This outcome emphasizes the advantages of using advanced neural network models in speech processing. It confirms the potential for further enhancements in speech-to-animation by enabling the model to generalize through its strong representation power, which results from training on a vast amount of speech data from a highly diverse group of speakers.

In the next stage of our end-to-end model investigation, we expanded our exploration to include other neural network-based audio feature representations. We specifically analyzed models like WavLM [24] and Whisper [126], which, akin to Wav2Vec, are rooted in deep learning paradigms. As with Wav2Vec, we expect the representation of these models to improve the animation synthesis from any given audio. The core purpose of this exploratory study was to identify novel and more efficient techniques for modeling and interpreting speech data, thereby advancing one more step the state-of-the-art in speech-to-animation.

4.3.1 WavLM

WavLM, a model constructed based on the principles of HuBERT [65] and Wav2Vec 2.0 [10], is an innovation in the field of audio processing. The concept of masked speech prediction [40] inspires its design and incorporates an element of denoising, all while utilizing a Transformer-based model.

The architecture of the WavLM model comprises a convolutional encoder that processes 25 ms segments of audio at a sampling rate of 50 frames per second (FPS) with a stride of a 20 ms window. Following the conversion of the audio into convolutional embeddings, these are masked and fed into a series of Transformer Encoder layers. A key feature of this model is the inclusion of a gated relative position bias [28]. This mechanism allows the relative position bias to adapt, conditioning it based on the current speech content.

The strength of WavLM arises from its pre-training methodology, which capitalizes on a vast reservoir of unlabeled speech data that extends beyond traditional audiobook data found in resources like LibriSpeech [113]. The pre-training dataset encompasses 94k hours of public audio collected from various sources, such as audiobooks from the Libri-Light dataset [76], including podcasts and YouTube videos from the GigaSpeech dataset [21], and recordings from the European Parliament [156]. The pre-trained model can then be fine-tuned for specific downstream tasks.

Table 4.1: Overview of the pre-trained WavLM audio encoder models available. The table outlines each model's number of Transformer Encoder layers, feature encoding dimension, number of parameters (expressed in millions as 'M'), and the quantity of data (expressed in hours) used for pre-training each model variant: Base, Base+, and Large.

Model	Num. Enc. Layers	Feature Dim.	Num. Parameters	Num. hours Data
Base	12	768	94 M	960
Base+	12	768	94 M	94k
Large	24	1024	320 M	94k

The effectiveness of WavLM is evidenced by its performance on the SUPERB benchmark [163], where it outperformed other models on a total of 14 different tasks. These tasks span a broad spectrum of applications, including speaker identification, speaker diarization, out-of-domain automatic speech recognition, speech translation, emotion recognition, among many others. The impressive performance of WavLM across such a diverse range of tasks underscores the model's versatility and its potential use in the speech animation field.

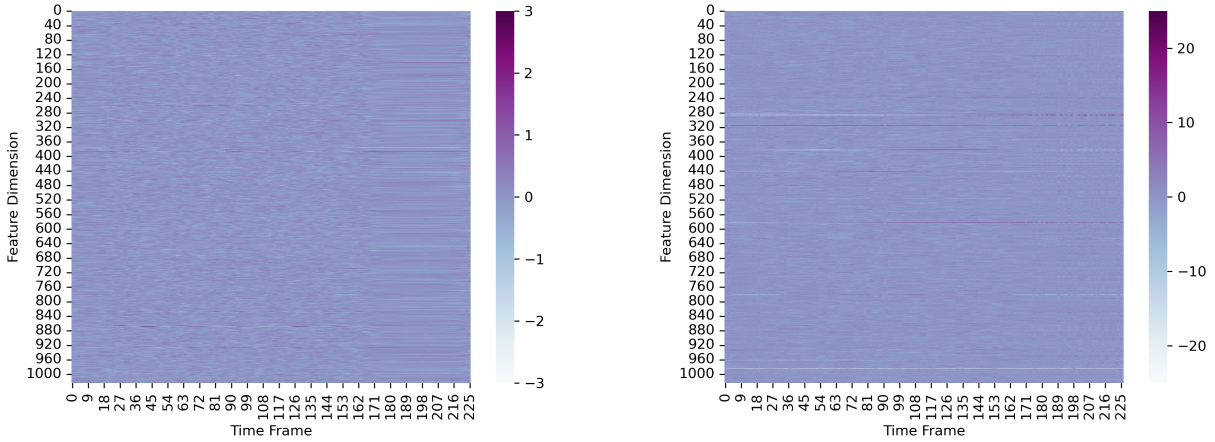
The authors have made available a collection of pre-trained models for the public to utilize in downstream tasks. A summary of these models can be found in Table 4.1, where we describe the number of Transformer Encoder layers per model, what is the dimensionality of the audio feature representations, the number of parameters and on how many hours the model was trained on. In our study, we focus on investigating the applicability of the *Large* model within the domain of speech animation as it performed the best on the SUPERB benchmark [24] and was trained on 94,000 hours of audio. We provide a visualization of how this model encodes a sample from the IMFT'23 dataset in Figure 4.6a. This sample features silence at both the beginning and the end, with speech occurring in the middle. The visualization reveals a stable pattern during the silent sections, contrasting with the irregular patterns that appear during speech.

4.3.2 Whisper

Whisper, an automatic speech recognition (ASR) system developed by OpenAI, harnesses an extensive dataset comprising 680,000 hours of undisclosed multilingual audio obtained from diverse sources on the web. This substantial corpus empowers Whisper with the ability to recognize diverse accents in spoken English and effectively mitigate the impact of significant background noise, such as loud music or mechanical sounds. Moreover, Whisper possesses the remarkable capability of performing simultaneous automatic speech recognition and translation into English.

The architecture of Whisper is based on an end-to-end vanilla encoder-decoder Transformer framework. The primary objective behind this design is to showcase the potential for substantial data volumes to significantly enhance the performance of simple models without necessitating complex design alterations. During audio processing, the input signal is segmented into 30-second intervals, which are subsequently transformed into log-Mel spectrograms. These spectrograms are then fed into the encoder, incorporating a convolutional feature subsampler analogous to WavLM. The intermediate features extracted by the encoder are subsequently fed into a Transformer encoder. The decoder is responsible for predicting the corresponding text captions and is equipped with a meticulously designed set of tokens enabling diverse tasks, including language identification, multilingual transcription, and speech-to-English translation.

Despite not being specifically fine-tuned for any particular dataset, Whisper exhibits remarkable robustness in zero-shot scenarios across a diverse range of datasets. Particularly noteworthy is Whisper's capacity to transcribe speech in the



(a) WavLM Large **(b)** Whisper Medium

Figure 4.6: Visualization of the selected WavLM and Whisper audio features for audio sample 0734 with transcript *"The sky in the west is tinged with orange-red."*. Note the difference in the range of values between both encodings. The WavLM feature shows stable patterns during the silence at the end of the sequence, while the Whisper model shows some noisy patterns during the silence segments.

original language, even in low-resource language scenarios, as well as its proficiency in translating speech into English. In fact, Whisper's performance surpasses that of state-of-the-art supervised models in English translation tasks.

Just like in contemporary research practices, the authors have contributed a range of pre-trained models for public use. These models are summarized in Table 4.2, echoing the format presented for WavLM. The authors have developed this diverse collection of models with the intention of catering to various applications and significantly enhancing speed in comparison to the largest model. In our experiments, we aim to explore the *Medium* model's application to our task, considering its resemblance to the largest WavLM model.

We present a visualization of feature encodings in Figure 4.6b. Unlike the encodings for Wav2Vec and WavLM, which are displayed in Figures 3.3e and 4.6a respectively, the silence segments in this model reveal some irregular patterns absent in the other encodings. Also notice the range of values of the encodings is an order of magnitude larger than the WavLM encodings.

Table 4.2: Overview of the pre-trained Whisper speech recognition models available. The table outlines each model's number of Transformer Encoder layers, feature dimensionality, number of parameters (expressed in millions as 'M'), and the quantity of data (expressed in hours) used for pre-training each model variant: Tiny, Base, Small, Medium, and Large.

Model	Num. Enc. Layers	Feature Dim.	Num. Parameters	Num. hours Data
Tiny	5	384	39 M	680k
Base	7	512	74 M	680k
Small	12	768	244 M	680k
Medium	25	1024	769 M	680k
Large	33	1280	1550 M	680k

4.4 Exploring Rig Parameter Velocity and Acceleration Loss

The pursuit of obtaining a performance improvement in our end-to-end model led us to consider an approach similar to the *Physics-Informed Networks* (PINN) [36], which emerged as a significant trend in 2018. This paradigm integrates physical laws into the training of machine learning models, as demonstrated by Raissi *et al*. [127], Smith *et al*. [142], and Cai *et al*. [17]. The strategy involves incorporating a physical term—such as the residual of a differential equation, for solving the Eikonal equation in 3D velocity structures without retraining the network for each velocity, or by governing the heat equation, momentum equation, and continuity equation through the loss to obtain the temperature and velocity fields in the domain. All of these approaches ensure the model predictions align with physical plausibility.

Research on neural networks to synthesize body and face animations, utilize various techniques such as direct vertex position prediction [9, 78], landmark prediction [106, 170, 171], estimating the motion vectors [23, 49], or directly the model parameters [22, 66, 121, 150] that modify the mesh. Our goal in developing an end-to-end model aligns with these approaches. These models commonly employ a reconstruction loss, typically minimizing the mean squared error (MSE), to train and reconstruct the target data. However, relying solely on frame-by-frame pose reconstruction may result in animations with noticeable jitter, as neighboring frame results are not fully considered during optimization. We have employed an RNN-based approach that leverages hidden states to carry information across frames, even incorporating future timesteps in bidirectional configurations to address this problem [106]. Nevertheless, we aim to investigate whether incorporating additional dynamic information of the parameters into the loss function can further enhance the network's performance and even further if all audio encodings perform the same with similar weightings for the velocity and acceleration loss.

In the literature, we often find claims that adding the velocity at which the target output changes improves the animation [16, 78] or through the error of an acceleration proxy of the predicted vertices with respect to the ground truth [22]. However, no support or justification has been given, nor the specifics of the weight values under which these loss terms are incorporated into the training of the network. Inspired by PINNs, we decided to explore the impact of adding the first and second derivatives of the rig parameters into our model and determine its value towards our task. Therefore, as a first approach, we set our loss as described in eq. 4.8, where besides a reconstruction loss, we added the velocity loss $\mathcal{L}_{vel}$ and acceleration loss $\mathcal{L}_{accel}$ terms, which are independently weighted with their own coefficients λ_{vel} and λ_{accel} respectively.

$$\mathcal{L} = \mathcal{L}_r + \lambda_{vel} \cdot \mathcal{L}_{vel} + \lambda_{accel} \cdot \mathcal{L}_{accel} \tag{4.8}$$

Through the reconstruction loss, we seek to minimize the MSE of the sequence S of the estimated rig parameters $\hat{\mathbf{y}}$ and the ground truth parameters $\mathbf{y}$ where $\hat{\mathbf{y}}, \mathbf{y} \in \mathbb{R}^{67}$ as described in eq. 4.9 .

$$\mathcal{L}_r = \frac{1}{|S|} \sum_{t=1}^{S} (\hat{\mathbf{y}}_t - \mathbf{y}_t)^2 \tag{4.9}$$

The first derivative, also referred to as the velocity, characterizes how the rig parameters change over time from one timestep to the next. This concept can be mathematically represented as shown in eq. 4.10, where the velocity at time t,

denoted by $\mathbf{v}_t$, is defined as the difference in rig parameter values between successive timesteps, $t+1$ and t.

$$\mathbf{v}_t = \mathbf{y}_{t+1} - \mathbf{y}_t \tag{4.10}$$

To ensure consistent motion over time, we introduce a velocity loss term, $\mathcal{L}_v$, defined in eq. 4.11. This term is calculated as the mean squared error (MSE) between the predicted sequence velocities ($\hat{\mathbf{v}}_t$) and the ground truth velocities ($\mathbf{v}_t$) across all timesteps in the sequence S.

$$\mathcal{L}_v = \frac{1}{|S|} \sum_{t=1}^{S-1} (\hat{\mathbf{v}}_t - \mathbf{v}_t)^2 \tag{4.11}$$

By incorporating the velocity loss term into our model, we aim to minimize the discrepancy between predicted and actual velocities, attempting to improve the motion consistency of the resulting animation by avoiding falling into any local minima in the rig parameter space.

Expanding upon the concept of velocity, we further incorporate the idea of acceleration, a measure that encapsulates the second-order dynamics of the rig parameters by quantifying instant changes in velocity between contiguous timesteps. As delineated in eq. 4.12, acceleration at time t, $\mathbf{a}_t$, is computed as the difference in velocity values between sequential timesteps, $t+1$ and t.

$$\mathbf{a}_t = \mathbf{v}_{t+1} - \mathbf{v}_t \tag{4.12}$$

We introduce an acceleration loss term, $\mathcal{L}_a$, analogous to the velocity loss term. Defined in eq. 4.13, this term is the mean squared error (MSE) between the predicted sequence accelerations ($\hat{\mathbf{a}}_t$) and the ground truth accelerations ($\mathbf{a}_t$) over all timesteps in the sequence S.

$$\mathcal{L}_a = \frac{1}{|S|} \sum_{t=1}^{S-1} (\hat{\mathbf{a}}_t - \mathbf{a}_t)^2 \tag{4.13}$$

By incorporating both velocity and acceleration loss terms, we aim to reduce discrepancies between predicted and actual motion parameters. However, it's important to approach the use of acceleration with care. As a second-order measure, acceleration can amplify the noise if the training data is inherently noisy. This amplified noise could introduce additional complexity to the network training process. Consequently, while acceleration provides valuable insights into motion dynamics, its potential to increase noise must be carefully managed during model training. In our experiments, we seek to determine if this measure is useful or not for improving a speech-to-animation model's performance.

4.5 Inner Layers of Neural Audio Feature Representations

Current models employed for large-scale audio feature representation exhibit a common architectural paradigm. Typically, these models either process raw audio input, as exemplified by WavLM [24], or preprocess the audio using a Short-Time Fourier Transform (STFT) as is the case with Whisper [126]. This audio data is subsequently transformed into a more manageable, compact, intermediate feature space utilizing a Convolutional Neural Network (CNN).

After the initial convolutional feature subsampling of the input signal, the resulting sequence of features undergoes

masked speech modeling, as proposed for the HuBERT model [65]. These features then enter a stack of Transformer-based encoders producing representational features suitable to solve various downstream tasks, a process rooted in strategic training methods that constitute the primary focus of current research in this field.

These features are further processed through a Transformer decoder during the training phase, a methodology observed in both Whisper and WavLM. Alternatively, a self-supervised approach is adopted, as in W2V-Bert [33]. By adopting this shared architectural framework, these models have made significant strides in the realm of audio feature representation.

In our pursuit to further explore these models for audio feature representation, we contemplated whether the information derived from the final layer ENC_N of a Transformer Encoder model with N layers suffices for speech animation models. For this reason, we posed ourselves the question of whether including some intermediate representations from the encoder could enhance the model's performance, leading to superior animations? Prompted by this hypothesis, we sought to ascertain the significance of the encoding layers' embeddings $\mathbf{z} = [z_1, z_2, ..., z_N]$ where $z_i = \text{ENC}_i(z_{i-1})$ by learning a weighted combination (eq. 4.15) of the N encodings through weights $\omega = [\omega_1, \omega_2, ..., \omega_N]$ via a softmax operation (eq. 4.14). By doing this, we aim to make an analysis of the importance of the layers through the normalized linear weights, and we also seek to enhance the model's convergence. This approach is illustrated in Figure 4.7 for the WavLM and Whisper models.

$$\omega_i = \frac{e^{z_i}}{\sum_{j=1}^{N} e^{z_j}} \tag{4.14}$$

$$z' = \sum_{i=1}^{N} \omega_i z_i \tag{4.15}$$

To ensure thoroughness, we mirrored this approach with Wav2Vec, which serves as our baseline audio representation despite being a fully convolutional model. The fusion of the $z-$ and $c-$ features via a softmax layer is depicted in Figure 4.8.

As suggested by [24], we anticipate that the initial layers encoding coarse sub-phoneme representations from the speech signal will exhibit high-importance values. In contrast, the final layers, which capture the subtleties of the audio, are expected to enhance performance on key tasks.

The exact significance of these layers, however, warrants further investigation. The results of this in-depth analysis will be presented further in the Experimentation section of this chapter. This will provide a comprehensive understanding of the role and impact of each layer in the audio feature representation models that we are studying.

4.6 Phonetically Informed Speech Animation Network via Multi-task Learning (PhISANet)

As we seek to keep on improving the model's performance, we conjectured whether phonetic information could similarly be utilized to augment network performance in generating animations from speech. A review of the Automatic Speech Recognition (ASR) field indicated that multi-task learning had been a successful approach in improving speech

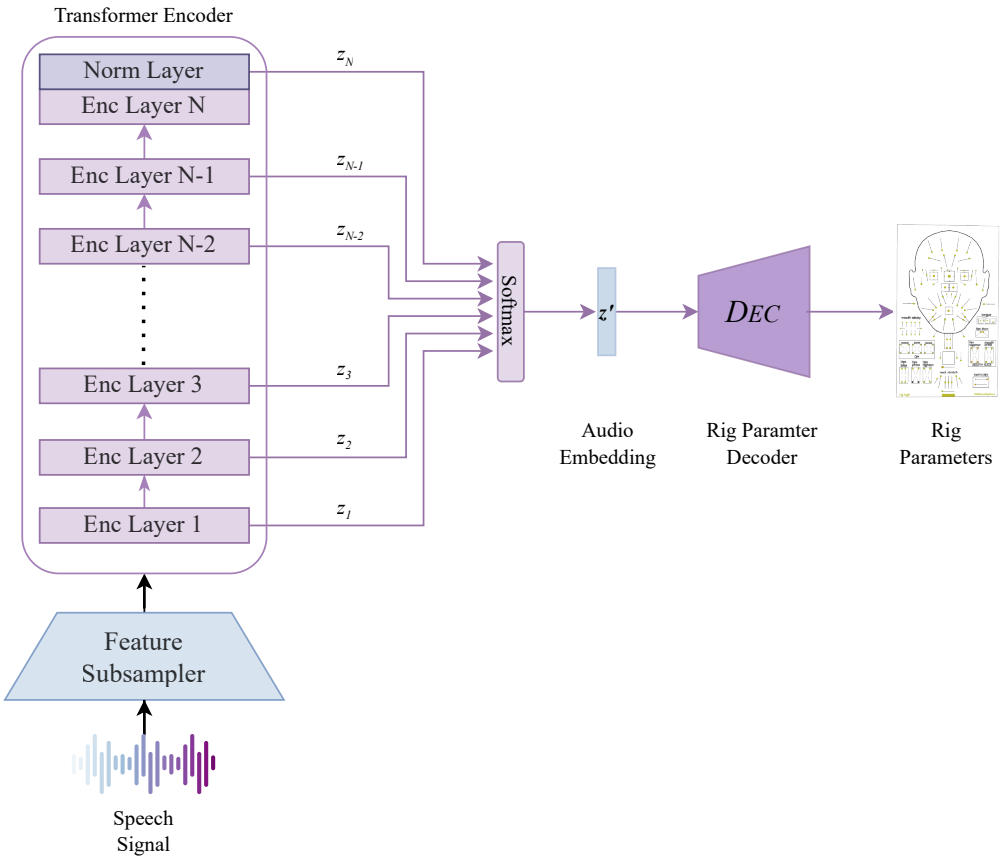


Figure 4.7: Fusion of all transformer encoder layers of WavLM and Whisper models for the end-to-end model.

recognition models.

For instance, Parveen and Green [116] leveraged multi-task learning to develop a robust ASR model based on Recurrent Neural Networks (RNNs), complemented by a noisy speech enhancement auxiliary task. Hinton *et al*. [60] adopted multi-task learning for acoustic modeling in ASR, simultaneously predicting phonemes and senones (context-dependent phonemes) via distinct branches of a Deep Belief Network (DBN). Kim *et al*. [84] introduced a joint Connectionist Temporal Classifier (CTC) attention-based end-to-end ASR system that utilized multi-task learning for sequence labeling and output sequence prediction, thereby significantly enhancing system performance. Heba *et al*. [59] addressed character-level speech recognition through multi-task learning, employing Consonant-Vowel (CV) recognition as an auxiliary task via Connectionist Temporal Classification (CTC). Finally, Chen *et al*. [27] improved multilingual ASR by incorporating hierarchical CTC objectives into an encoder-decoder model, postulating that language identification assists model convergence.

Informed by this work, our exploration comprises two paths to augment our model via multi-task learning. Firstly, we propose incorporating an auxiliary task to regularize the decoder—a classifier predicting the phone representation of the utterance at any given audio frame. The second approach seeks to regularize the audio feature decoder by predicting the phone sequence corresponding to the utterance via a CTC, paralleling the method proposed by Kim *et al*. [84].

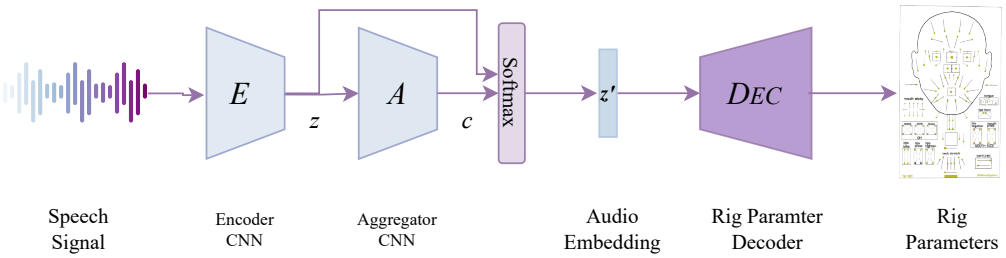


Figure 4.8: Fusion of Wav2Vec Layers for the end-to-end model.

4.6.1 Phone Classifier

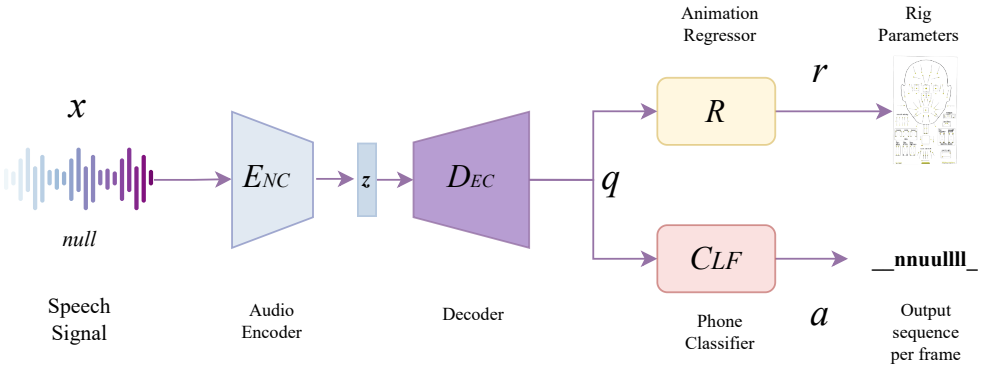


Figure 4.9: Multi-task learning speech animation model with a phone classifier auxiliary task.

Deploying a multilabel cross-entropy classifier is a robust mechanism to regularize the training process of the decoder. The primary objective of this strategy is to enhance the quality of regression towards the rig parameter space, as illustrated in Figure 4.9. The sequence of decoded features, denoted as $\mathbf{q} = [q_1, q_2, ..., q_S]$, is an S-length sequence where each q_i is an F-dimensional vector ($q_i \in \mathbb{R}^F$). Additionally, $\mathbf{a} = [a_1, a_2, ..., a_T]$ is a T-length sequence of tokens where $a_i \in V$. Specifically, V is the set of all the phones present in the dataset, as determined by the Montreal Forced Aligner [104], with a total count of 87 phones including silence for our particular case. In Table 4.3, we can find a list of all the phonemes and their allophones captured in the dataset.

The multi-label cross-entropy classifier is used to optimize our speech-to-rig parameters model, aiming to detect the presence of phones at the frame level. This approach significantly deviates from simply incorporating a phone classifier into a separate branch of the decoder. Instead of modeling the conditional probability, it directly classifies the decoded features $\mathbf{q}$ into phone labels.

The classifier CLF takes the decoded features $\mathbf{q}$ as inputs and outputs the phone labels $\mathbf{a}$ that best align with the input speech signal based on the trained model.

Table 4.3: List of appearing phonemes and their allophones in the IMFT'23 dataset obtained by a forced alignment using the audio and their corresponding transcripts through the Montreal Forced Aligner.

Phoneme	Allophones
/a/	a, aː, ɐ, ɑ, ɑː
/aɪ/	aɪ
/aʊ/	aʊ
/b/	b, bʲ
/c/	c, cʰ, cʷ
/d/	d, dʲ, d̪
/dʒ/	dʒ
/e/	e, eɪ
/f/	f, fʲ
/h/	h
/i/	i, iː, ɪ
/j/	j
/k/	k, kʰ, kʷ
/l/	l, ɬ
/m/	m, mʲ, m̩
/n/	n, ŋ, ɲ, n̩
/o/	o, oʊ
/p/	p, pʰ, pʲ
/s/	s
/t/	t, tʃ, tʰ, tʲ, tʷ, t̪
/u/	u, uː, ʉ, ʉː, ʊ
/v/	v, vʲ

Phoneme	Allophones
/w/	w
/z/	z
/æ/	æ
/ç/	ç
/ð/	ð
/ɒ/	ɒ, ɒː
/ɔ/	ɔ, ɔɪ
/ə/	ə, əʊ, ɚ
/ɛ/	ɛ, ɛː
/ɜ/	ɜ, ɜː, ɝ
/ɟ/	ɟ, ɟʷ
/ɡ/	ɡ
/ɬ/	ɬ
/ɹ/	ɹ
/ɾ/	ɾ, ɾʲ, ɾ̃
/ʃ/	ʃ
/ʎ/	ʎ
/ʒ/	ʒ
/ʔ/	ʔ
/θ/	θ
/ø/	ø

$$\mathbf{z} = \text{ENC}(\mathbf{x}) \tag{4.16}$$

$$\mathbf{q} = \text{DEC}(\mathbf{z}) \tag{4.17}$$

$$\mathbf{r} = \text{R}(\mathbf{q}) \tag{4.18}$$

$$\mathbf{a} = \text{CLF}(\mathbf{q}) \tag{4.19}$$

The loss of the classifier is the multi-label cross-entropy as expressed in eq. 4.20. During network training, this loss is incorporated into the loss for reconstruction, which computes the rig parameters dynamics described in eq. 4.8 weighted by λ_{CLF}.

$$\mathcal{L}_{CLF} = -\sum_{i=1}^{T} \mathbf{a}_i \cdot \log(\hat{\mathbf{a}}_i) \tag{4.20}$$

Here, $\mathbf{a}_i$ and $\hat{\mathbf{a}}_i$ denote the true and predicted labels, respectively, for each frame in the sequence. The sum runs over all frames, and the dot product calculates the cross-entropy for each frame, which is then summed to give the total multilabel

cross-entropy loss.

4.6.2 Connectionist Temporal Classifier

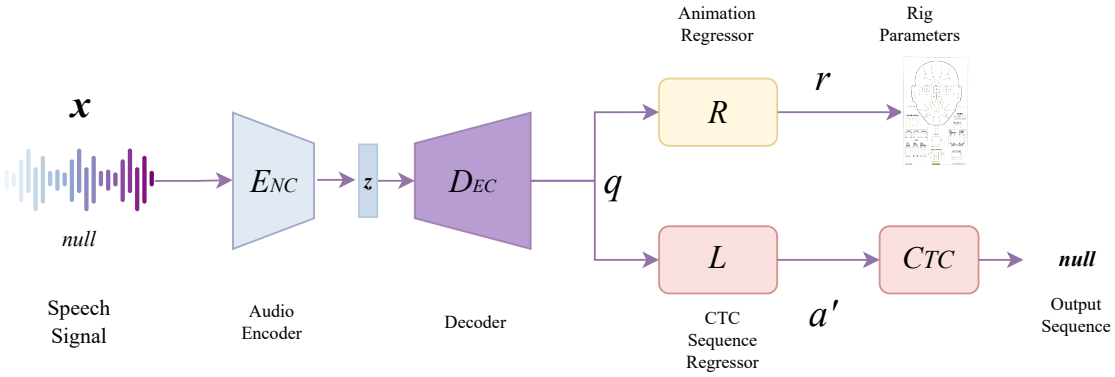


Figure 4.10: Multi-task learning speech animation model with a Connectionist Temporal Classification layer to predict the corresponding phone sequence.

Building on the same idea of multi-task learning as with the phone classifier branch, an alternative is to employ a Connectionist Temporal Classification (CTC) to regularize the decoder during training to improve the rig parameter sequence estimation. The model is depicted in Figure 4.10.

Our CTC-based model can be described as follows. First, an encoder ENC transforms the input speech signal **x** into a latent representation **z** as in eq. 4.21. Then, a decoder DEC takes this latent representation **z** and transforms it into a series of features **q** as in eq. 4.22. A regressor R solves for the main task by mapping the features **q** to rig parameters **r** as depicted in eq. 4.23. Additionally, on the auxiliary task branch, the phone sequence regressor L transforms **q** into a form suitable for the CTC network which maps to a feature space that considers a blank space as shown in eq. 4.24 and we will describe its use later in this section. Finally, a Connectionist Temporal Classification (CTC) is used to align the features **l** with the sequence of phones **a**.

$$\mathbf{z} = \text{ENC}(\mathbf{x}) \tag{4.21}$$

$$\mathbf{q} = \text{DEC}(\mathbf{z}) \tag{4.22}$$

$$\mathbf{r} = \text{R}(\mathbf{q}) \tag{4.23}$$

$$\mathbf{a} = \text{CTC}(\text{L}(\mathbf{q})) \tag{4.24}$$

To further understand how a CTC works, let us first define the decoded audio features $\mathbf{q} = [q_1, q_2, ..., q_T]$ where $(q_i \in \mathbb{R}^F)$ as they are F-dimensional vectors according to the dimensionality of the decoder. Also, let us define our vocabulary V as the set of phones found in our dataset through the Montreal Forced Aligner [104], and the sequence of phone tokens $\mathbf{a} = [a_1, a_2, ..., a_S]$ where $(a_i \in V)$ and the length S of the sequence of corresponding phones is shorter or equal to length T of the sequence of latent representations $(S \leq T)$. As one second of audio could be represented by $T = 50$

latent representations while only encompassing a sequence of $S = 5$ phones, for example.

The application of CTC [55] to our model enables the alignment between decoded features $\mathbf{q}$ and phone sequence $\mathbf{a}$. Distinct from the phone classifier, through the CTC, we seek to model the conditional probability $P(\mathbf{a}|\mathbf{q})$ as latent sequences per frame.

To achieve this, we use an intermediate label representation $\mathbf{a'} = [a'_1, a'_2, ..., a'_T]$ with repeated phone labels plus the addition of a blank symbol $(-)$ where $\mathbf{a'}_i \in V^{'}$ given that new intermediate label set is $V^{'} = V \cup \{-\}$. Note that the new label representations will match the sequence length of the intermediate representations $\mathbf{q}$. Through the CTC, we seek to maximize the probability distribution $P(\mathbf{a}|\mathbf{q})$ over all possible intermediate label sequences generated by $\Phi(\mathbf{a})$, as follows:

$$P(\mathbf{a'}|\mathbf{q}) = \prod_{t}^{T} P(a'_t|\mathbf{q}), a'_t \in V^{'} \tag{4.25}$$

$$P(\mathbf{a}|\mathbf{q}) = \sum_{\mathbf{a'} \in \Phi(\mathbf{a})} P(\mathbf{a'}|\mathbf{q}) \tag{4.26}$$

The function Φ plays a crucial role in estimating the posterior probability of a CTC, as it is designed to map the observed phone sequence $\mathbf{a}$ to all possible latent sequences $\mathbf{a'}$ that could be derived from $\mathbf{a}$.

The augmented label sequence $\mathbf{a'}$ is built by inserting blank symbols at the beginning and the end of the sequence and between the phones. For example, for the word *null*, its grapheme sequence is $\mathbf{a} = [n, u, l, l]$, while the augmented sequence is $\mathbf{a'} = [-, n, -, u, -, l, -, l, -]$. It is important to note how adding blank symbols $-$ into the sequence of phones allows us to distinguish different instances of the phone l even if they are contiguous.

In our case, since we are using an RNN as decoder DEC, the output $\mathbf{q}$ is linearly mapped through the *CTC Sequence Regressor* to a feature space of dimensionality $|V^{'}|$ where each output after a softmax operation can be interpreted as the probability of observing the modified label at each timestep t. Hence:

$$P(\mathbf{a'}|\mathbf{q}) = \prod_{t}^{T} P(a'_t|\mathbf{q}) \approx \prod_{t}^{T} \text{Softmax}(l_t) \tag{4.27}$$

Where l_t is the output of the CTC Sequence Regressor L. The CTC works under the assumption that the latent variables $\mathbf{q}$ are sufficient to determine the latent space $\mathbf{a'}$ at any frame, and the probability of the label sequence can be modeled as conditionally independent by the product of the network outputs.

In Figure 4.11 we show an example, inspired in a tutorial by Hannun, A. [57], of how we go from a sequence of decoder embeddings $\mathbf{q}$ into a distribution of over the outputs $\{n, u, l, \text{-}\}$ for each timestep as the result of the softmax operation of the mapping of embeddings through $L(\mathbf{q})$. Then, through $\Phi(\mathbf{a})$ we compute all the possible sequences along with their probabilities, which in turn are marginalized over the alignments to get the most probable sequence.

The alignment computation can be quite expensive if not programmed correctly; for the CTC this is solved through a

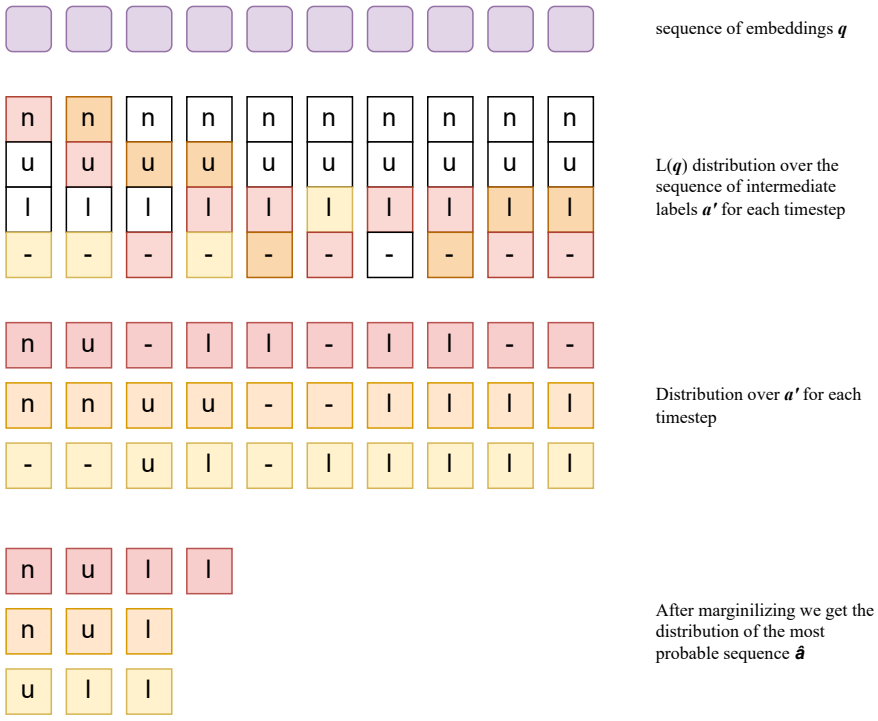


Figure 4.11: CTC distribution and marginalization example over the word *null*.

dynamic programming algorithm that merges two alignments if they have reached the same output at the same timestep based on the CTC probability from $L(\mathbf{q})$. The subsequence score is computed based on the previous timestep and only considers a probable merge for the previous token in the sequence or the second previous token in the sequence. This allows a fast merge and tractable score which can be employed for training the network.

Once the most probable sequence has been found, the CTC loss is calculated as the negative log-likelihood as described in eq. 4.28, and is added to the total loss (eq. 4.8) during the training of the network and controlled by a weight λ_{CTC}.

$$\mathcal{L}_{CTC} = -\log P(\mathbf{a}|\mathbf{q}) \tag{4.28}$$

The original publication by Graves *et al*. [55] provides a better understanding of the CTC, but the subsequent tutorial [56] provides an in-depth exploration of the CTC concept, including detailed derivations of the forward-backward algorithm and the CTC loss function.

4.7 Experimentation

In Figure 4.12, we delineate the series of experiments and evaluations conducted to obtain the most effective model for animation generation. The initial investigation stage involves a comprehensive grid search to probe the influence of incorporating the first and second derivatives of the rate of change of the rig parameters over time into the training loss. After this assessment, we scrutinize the significance of the audio encoder layers towards our task, thereby determining their relative importance. The next phase of our research introduces a novel approach of phonetically informing the

network via a multi-task learning framework, and we assess the extent to which this enhances the speech-animation end-to-end model's performance. After the model exploration, a series of user studies are conducted to gauge the human perception of the generated animations. Lastly, we propose using a state-of-the-art lip-reading network as a novel evaluative tool, examining the optimal models from each experiment to determine if user studies can be avoided. This stage aims to assess whether this alternative approach alone suffices to evaluate speech animation models.

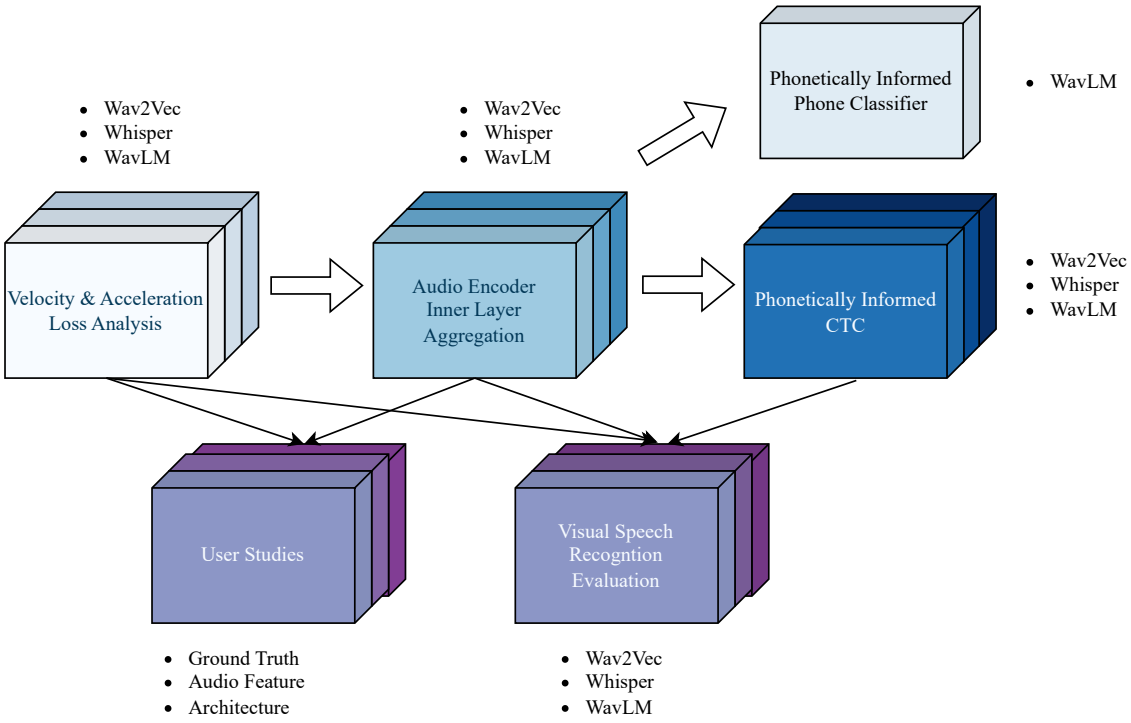


Figure 4.12: This diagram presents the layout of the experiments conducted to obtain the best end-to-end model. We performed four different experiments: (1) Velocity and Acceleration Analysis, (2) Audio Encoder Inner Layer Aggregation, (3) Phonetically Informed CTC, and (4) Phonetically Informed Phone Classifier. The best resulting model from each experiment was evaluated through a series of (5) User Studies, and a novel approach to evaluate the speech animation model through (6) Visual Speech Recognition. Each experiment was carried out for each of the audio feature representations explored in this study.

The audio encoder models used throughout all the experiments were the model originally released by the authors for Wav2Vec[1], WavLM[2], and Whisper[3]. We followed the same procedure during our experimentation with the first two-stage pipeline. We subsampled the Wav2Vec features from 100 FPs to 50 FPS, by concatenating the embeddings from two contiguous audio frames. We repeated this procedure as the WavLM and Whisper models deliver encodings every 20 ms.

All models in this study were developed using the PyTorch framework [117]. For the baseline model, the initial hyper-parameters were determined through the Hyper-band algorithm [94] provided by the Optuna framework [4]. The training and evaluation of the models were conducted on NVIDIA's Quadro RTX 8000 and A40 GPUs, each equipped with 48 GB of memory. It is noteworthy that the full memory capacity was never utilized in any of the training processes, and all models were trained using a single GPU.

[1] https://github.com/facebookresearch/fairseq/tree/main/examples/wav2vec
[2] https://github.com/microsoft/unilm/tree/master/wavlm
[3] https://github.com/openai/whisper

The decoder used in all the models was a bidirectional GRU [30], consisting of 5 layers, each containing 1,024 hidden units. In the baseline models, which include a pre-trained audio encoder and a decoder, the output from the GRU is projected into the rig parameter space using a linear layer. This mapping transforms the 1,024 units to 67 rig parameters. In all experiments conducted, the batch size was set to 32, and the maximum number of epochs was limited to 200. Interestingly, most models reached convergence within 100 epochs. Therefore, an early stop strategy was applied after 10 epochs, using the L2 error criterion. This approach was chosen because the models tended to overfit more quickly on the L1 error, even though the L1 error continued to decrease. The RAdam optimizer [97] was employed during training to find the optimal model, minimizing according to our proposed objective functions.

For this series of experiments, we employed the IMFT'23 dataset formed by a total of 1700 samples, which cover 2.28 hours of audio and animation parameters. The data was split into 80/20 for the training and test sets. The training samples were structured in an overlapping window fashion with input windows comprising 45 audio frames, equivalent to 900 ms, and a stride of 1 frame, or 20 ms. The output was subjected to min-max normalization, as the MetaHuman rig parameters varied within the range of either $[0, 1]$ or $[-1, 1]$, depending on the cases. For the linear layers and MLPs within the architecture, a consistent dropout rate of 0.1 was applied.

4.7.1 Velocity and Acceleration of the Rig Parameters

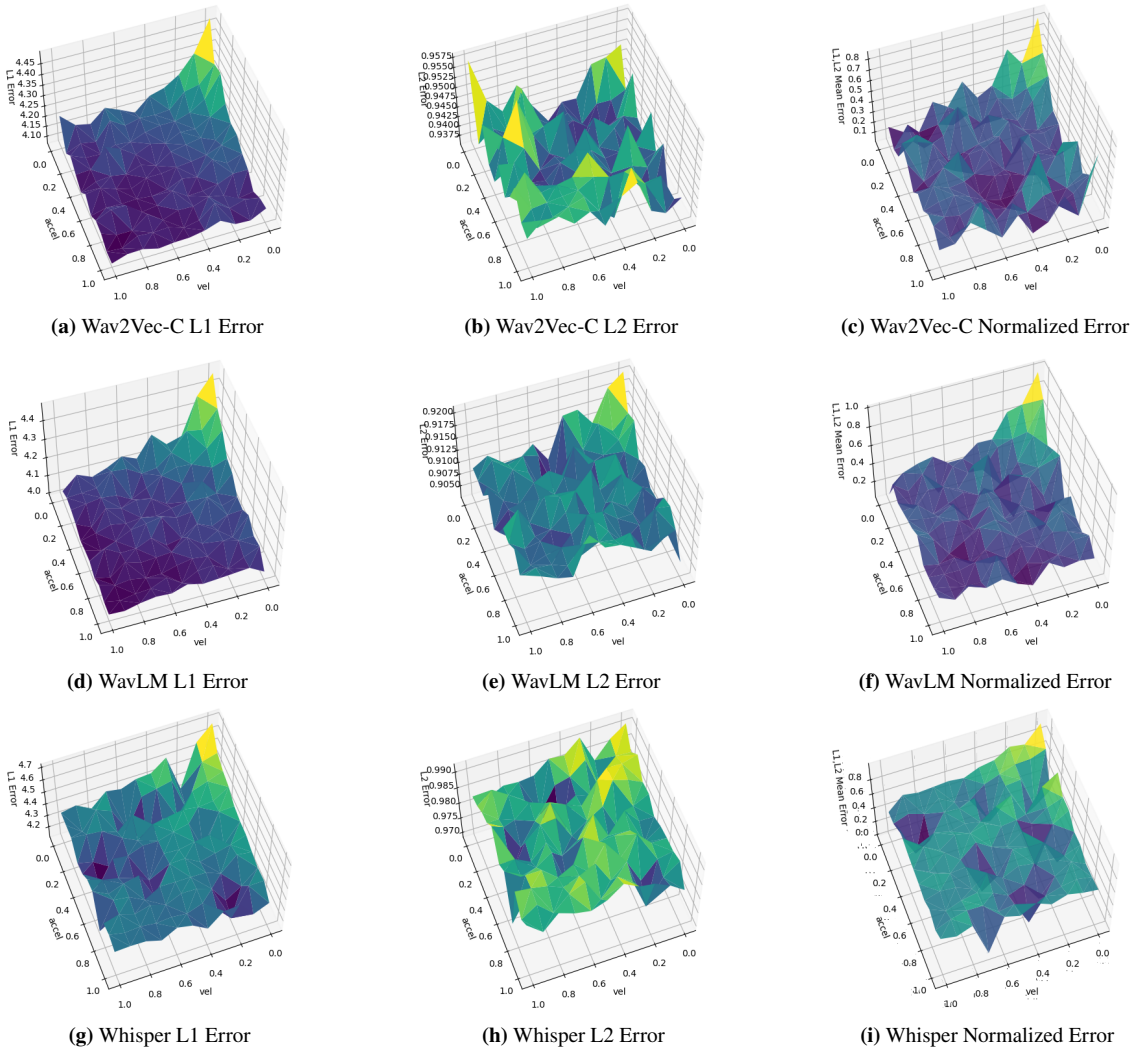
Our initial experiments aimed to investigate the effects of incorporating the first and second derivative of the rig parameters through a coarticulation sequence. In simpler terms, we integrated the error of the instantaneous velocity and acceleration of the predictions relative to the ground truth to empirically assess if this enhancement improves the quality of the generated animations. We incorporated these terms into the training loss as expressed in eq. 4.8 and conducted a grid search across the three audio encodings under investigation. The grid search varied the values of coefficients λ_{vel} and λ_{accel} within the range $[0, 1]$, at intervals of 0.1. In total, we trained 11 x 11 = 121 networks for each encoding, with the expectation of observing a reduction in the L1 and L2 errors on the predicted sequence of rig parameters.

The results of this study are presented in Figure 4.13. Each of the 3D surfaces was constructed by selecting the network with the lowest normalized error for each combination of the $(\lambda_{\text{vel}}, \lambda_{\text{accel}})$ values. The images in the first column illustrate that increasing both coefficients generally reduces the L1 error, except for the Whisper audio encoding. In the Whisper L1 error results, shown in Figure 4.13g, we observe distinct minima with lower values than those found at (1.0, 1.0), setting it apart from the Wav2Vec and WavLM counterparts.

The second column visualizes the behavior of the models based on the L2 Error. Generally, the optimization surfaces are irregular but display a concave form, suggesting that the error is reduced regardless of the chosen values for the velocity and acceleration loss coefficients. This trend can be more clearly seen in the third column of Figure 4.13, where we represent the normalized error by computing the mean of the min-max normalized L1 and L2 errors. While the Wav2Vec and WavLM models exhibit a similar downward slope from using only MSE error at (0.0, 0.0) as we increase both coefficients, the Whisper model exhibits distinct minima around coordinates (0.5, 0.2), (0.6, 0.6), and (0.8, 0.2) for λ_{vel} and λ_{accel}, respectively.

The optimal set of coefficient values is detailed in Table 4.4. The lowest L1 and L2 errors were achieved with the WavLM audio feature representation, specifically at $\lambda_{\text{vel}} = 0.4$ and $\lambda_{\text{accel}} = 0.3$, with values of 4.036 and 0.903, respectively. The model with the highest error utilized the Whisper features, with coefficient values of $\lambda_{\text{vel}} = 0.5$ and $\lambda_{\text{accel}} = 0.2$,

resulting in L1 and L2 errors of 4.119 and 0.968, respectively. While for the Wav2Vec-based model, the errors were 4.086 (L1) and 0.936 (L2) with $\lambda_{\text{vel}} = 0.1$ and $\lambda_{\text{accel}} = 0.8$. All models improved over their respective baseline, which were trained using only MSE loss. The *L1* error had a major impact by adding the instant velocity and acceleration terms, but the *L2* error also showed an improvement. These error values are arguably quite close to each other, indicating a relative consistency across the models.



(a) Wav2Vec-C L1 Error **(b)** Wav2Vec-C L2 Error **(c)** Wav2Vec-C Normalized Error

(d) WavLM L1 Error **(e)** WavLM L2 Error **(f)** WavLM Normalized Error

(g) Whisper L1 Error **(h)** Whisper L2 Error **(i)** Whisper Normalized Error

Figure 4.13: Figure demonstrating the impact of weighted velocity and acceleration on prediction errors in rig parameter sequences from audio data. Results are based on three audio feature extraction methods: Wav2Vec, WavLM, and Whisper. Errors are evaluated using L1, L2, and a Normalized error (mean of min-max normalized L1 and L2 errors). This figure shows different audio feature representations require different weightings in the loss function.

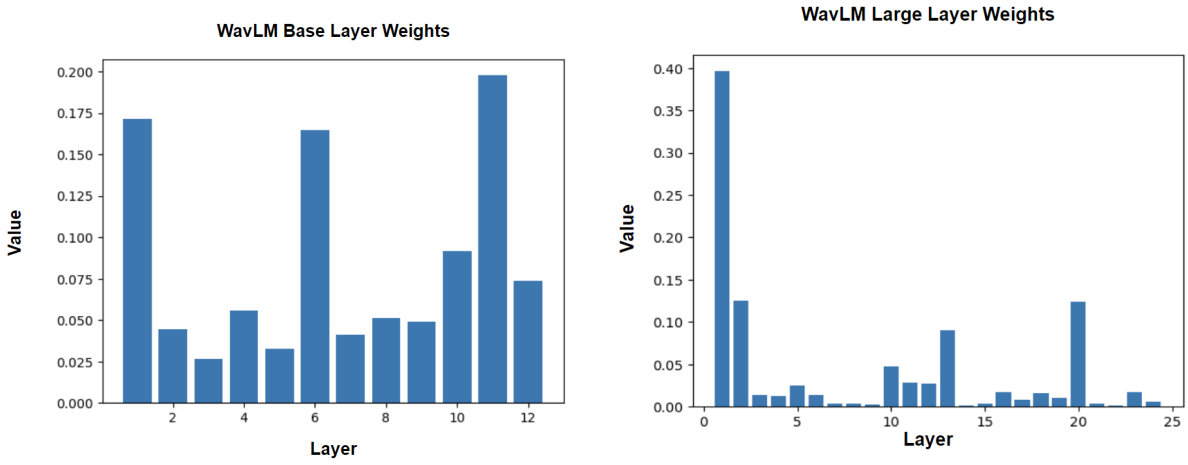
4.7.2 Inner Layers of Transformer Encodings Results

In this section, we investigated the significance of various layers within our audio encoder models. To achieve this, we designed a model that aggregates the encoder embeddings through a fusion layer described by eq. 4.14 and eq. 4.15. This layer, utilizing a softmax operation, calculates weights in the range of [0, 1], facilitating both interpretability and effective

Table 4.4: Best velocity and acceleration weighting results for each audio encoding.

Audio Feature	λ_{vel}	λ_{accel}	L1 error	L2 Error	Normalized Error
Wav2Vec	0.0	0.0	4.512	0.955	–
	0.1	0.8	4.086	0.936	0.020
WavLM	0.0	0.0	4.481	0.921	–
	0.4	0.3	**4.036**	**0.903**	0.039
Whisper	0.0	0.0	4.715	0.992	–
	0.5	0.2	4.119	0.968	0.000

model convergence. The results pertaining to the WavLM encoder model are illustrated in Figure 4.14. For the *Base* model, we found that layers 1, 6, and 11 were most influential, collectively accounting for about 53% of the total layer importance. While early layers were anticipated to be impactful due to their processing of broad speech characteristics, the prominence of the sixth layer was unexpected. Testing the *Large* model yielded similar patterns, where the sum of the weights from layers 1, 2, 13, and 20 covers 73.64% of the total relevance score. An original analysis provided by Chen *et al*. [24] indicates that intermediate layers are highly relevant when solving speaker identification tasks.



(a) Base model

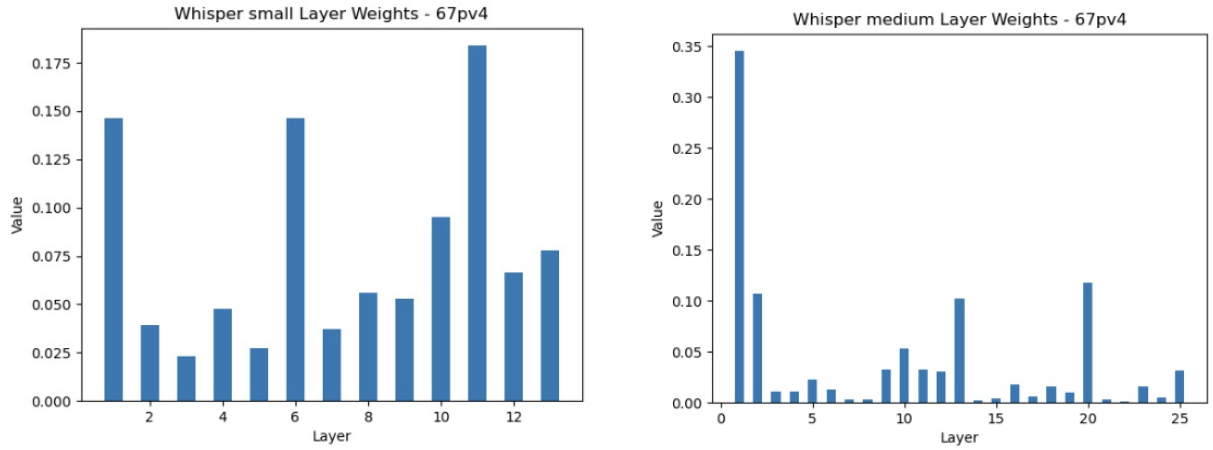
(b) Large model

Figure 4.14: Inner layer relevance visualization of the WavLM model. Layers, 1, 6, and 11 are the most relevant for the Base model (a), while the Large model (b) shows a similar pattern but with higher relevance on the initial layers of the Transformer encoder.

When conducting analogous experiments on the Whisper models, which bear resemblance in architecture to WavLM, we observed consistent patterns, as showcased in Figure 4.15.

To evaluate model performance in these experiments, we utilized L1, L2, and temporal mean vertex error (TMVE) metrics. The L1 error provides insights into parameter accuracy, the L2 error evaluates the smoothness of animations, and TMVE gives a perspective on the impact of the predicted parameters on the 3D mesh.

As depicted in Table 4.7, our Encoder-Decoder models, which were trained with the aggregated audio features, are labeled as *All-Enc*. These models generally outperformed models that were dependent solely on the encoding from the final



(a) Small model

(b) Medium model

Figure 4.15: Inner layer relevance visualization of the Whisper model. Layers, 1, 6, and 11 are the most relevant for the Small model (a), while the Medium model (b) shows a similar pattern but with higher relevance on the initial layers of the Transformer encoder.

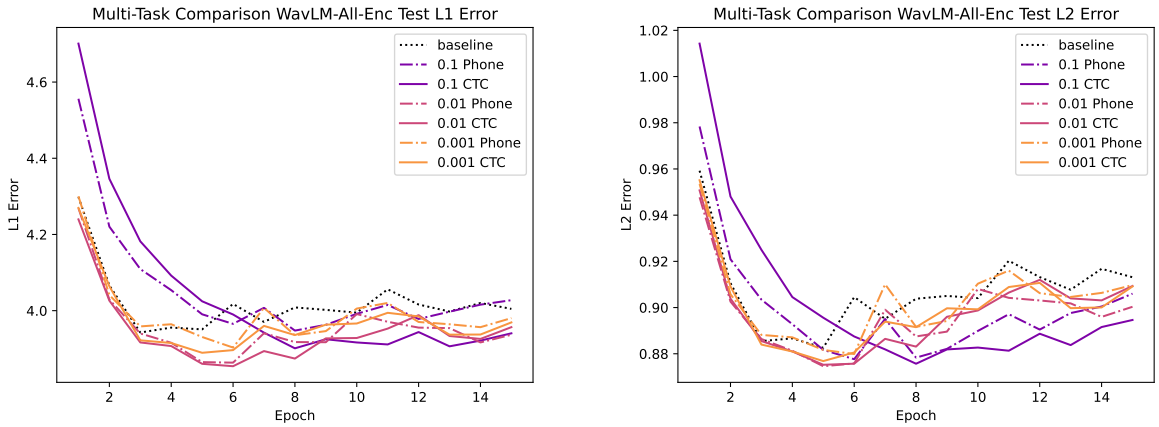
layer, which are denoted as *Enc*. However, with Wav2Vec, the results were mixed, suggesting that merging Wav2Vec's CCNN layers might not always be advantageous due to specifics in their self-supervised training methodology [136].

4.7.3 Phonetically Informed Speech Animation Network (PhISANet)

In our study, we strive to inform the animation parameter regressor by leveraging phonetic information. This integration is facilitated using multi-task learning (MTL) via two distinct methods: a phone classifier and a CTC. To determine the most effective approach for our objective, we embarked on a preliminary experiment. During this process, we briefly examined the coefficients that balance the MTL branch, particularly focusing on λ_{CLF} and λ_{CTC}. By adjusting the values of these MTL coefficients from the set $[0.1, 0.0, 0.001]$, we trained a series of multi-task learning models, evaluating all of them on our test set for every single epoch. This approach provided insights into their convergence behavior and allowed a comparative performance assessment. Based on our previous experiments, we decided to experiment with WavLM as it provided the best-performing model up to this point.

Referencing Figure 4.16, the results indicate that the CTC MTL outperforms the phone classifier in both L1 and L2 errors int rig parameter space, as seen in Figure 4.16a and Figure 4.16b, respectively. Importantly, it's evident that regardless of the auxiliary task chosen, both methods improve performance with a lower test error than the baseline—a standard encoder-decoder model. However, this prior gave us the insight to seek performance improvement from an MTL CTC perspective and make a thorough analysis of its impact in the speech animation realm.

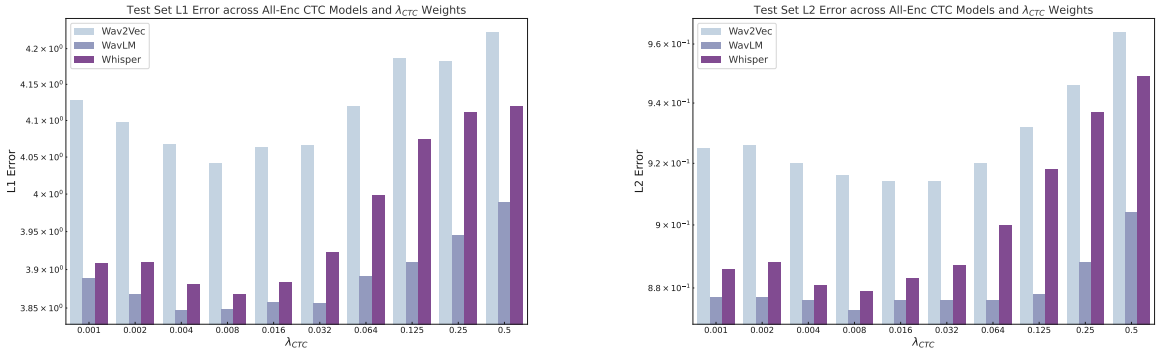
In our continued exploration of encoder-decoder models using a MTL CTC approach, we varied the coefficient λ_{CTC} that modulates the CTC auxiliary branch. This variation was done on an approximation to a logarithmic scale, ranging from 0.001 to 0.5. Table 4.5 summarizes the results, pinpointing $\lambda_{CTC} = 0.008$ as the optimal coefficient for the majority of the audio encodings under study. This trend is further visualized in Figure 4.17. While specific encodings like Wav2Vec and WavLM show marginal performance improvements at λ_{CTC} values of 0.016 and 0.004, respectively, the differences are minor in the scale of 0.001 and 0.002. Given the consistency of results at $\lambda_{CTC} = 0.008$, we have adopted this



(a) L1 Error **(b)** L2 Error

Figure 4.16: Multi-task learning comparative of the phone classifier (*Clf*) and CTC auxiliary tasks using WavLM audio encoding. The MTL models were evaluated on every single epoch on the test set. Both improve over the baseline, and the CTC errors are lower than the *Clf* regardless of the weight.

coefficient for subsequent experiments in this and the following chapters.



(a) L1 Error **(b)** L2 Error

Figure 4.17: The figures present the L1 (a) and L2 (b) errors for the *All-Enc CTC* models trained on different audio encodings- Wav2Vec, WavLM, and Whisper-, tested across varying λ_{CTC} weights. Each bar group corresponds to a specific λ_{CTC} weight. On both metrics, the minimum values are obtained when λ_{CTC} is 0.008 across all the models.

Based on our findings thus far, we are motivated to investigate further the optimal model configuration for producing enhanced animations. We intend to assess three distinct audio feature representations across various model configurations: an encoder-decoder that employs audio encoding from the final layer (denoted as *Enc*), another encoder-decoder leveraging a weighted mean of all encoder layers (termed *All-Enc*), and their respective variants when trained with an auxiliary CTC (labeled *Enc CTC* and *All-Enc CTC*). A concise summary of these configurations is provided in Table 4.6 for future reference.

Table 4.5: Performance metrics (L1 and L2 Errors) for different audio encodings (Wav2Vec, WavLM, and Whisper) at various λ_{CTC} coefficients. The results suggest an optimal λ_{CTC} value of 0.008 for most encodings, with minor deviations for specific metrics.

	Wav2Vec		**WavLM**		**Whisper**	
λ_{CTC}	**L1 Error**	**L2 Error**	**L1 Error**	**L2 Error**	**L1 Error**	**L2 Error**
0.5	4.223 ± 0.716	0.964 ± 0.173	3.989 ± 0.678	0.904 ± 0.163	4.120 ± 0.731	0.949 ± 0.177
0.25	4.182 ± 0.696	0.946 ± 0.170	3.945 ± 0.655	0.888 ± 0.157	4.111 ± 0.709	0.937 ± 0.172
0.125	4.186 ± 0.670	0.932 ± 0.162	3.909 ± 0.648	0.878 ± 0.156	4.074 ± 0.714	0.918 ± 0.172
0.064	4.119 ± 0.677	0.920 ± 0.165	3.891 ± 0.663	0.876 ± 0.158	3.999 ± 0.695	0.900 ± 0.165
0.032	4.066 ± 0.688	0.914 ± 0.165	3.856 ± 0.689	0.870 ± 0.164	3.923 ± 0.699	0.887 ± 0.166
0.016	4.064 ± 0.664	$\mathbf{0.914 \pm 0.159}$	3.858 ± 0.670	0.876 ± 0.161	3.884 ± 0.687	0.883 ± 0.162
0.008	$\mathbf{4.041 \pm 0.716}$	0.916 ± 0.171	3.848 ± 0.678	$\mathbf{0.873 \pm 0.164}$	$\mathbf{3.868 \pm 0.674}$	$\mathbf{0.879 \pm 0.161}$
0.004	4.068 ± 0.712	0.920 ± 0.170	$\mathbf{3.847 \pm 0.685}$	0.876 ± 0.165	3.881 ± 0.661	0.881 ± 0.159
0.002	4.098 ± 0.701	0.926 ± 0.168	3.868 ± 0.684	0.877 ± 0.165	3.910 ± 0.676	0.888 ± 0.161
0.001	4.128 ± 0.683	0.925 ± 0.163	3.889 ± 0.685	0.877 ± 0.164	3.908 ± 0.689	0.886 ± 0.164

Table 4.6: Summary of various architecture designs explored in our experiments. Each design is distinguished by its method of encoding, whether it utilizes the last Transformer layer only (denoted as 'Enc') or combines all Transformer layers (denoted as 'All-Enc'), and its multi-task learning (MTL) strategy, which may involve a phone classifier ('Clf') or a Connectionist Temporal Classification (CTC) for aligning the phone sequence.

Acronym	**Description**
Enc	Encoding from last Transformer Layer and decodes directly into rig parameters
All-Enc	Encoding by combining all Transformer Layers and decodes directly into rig parameters
Enc CTC	Encoding from the last Transformer layer, MTL with CTC aligning phone sequence
All-Enc CTC	Encoding by combining all Transformer layers, MTL with CTC aligning phone sequence

4.7.4 Rig Parameter Evaluation

Before we delve into the results found in our experiments, here is a brief disclaimer. For all the experiments reported in this and the following section, we conducted a pairwise t-test to determine the significant differences between the means of metrics reported. The confidence in the results is reported by their p-values, which, if $p < 0.05$, indicates a high level of statistical significance. Such low p-values provide strong evidence against the null hypothesis, suggesting that the differences in means between our groups are not due to random chance. Allowing us to have a high degree of confidence in our results. The full results are shown in Appendix A.

An additional evaluation of the models was conducted to investigate their behavior regarding L1 and L2 test errors. As illustrated in Figure 4.18, the Wav2Vec-based model does not gain any advantage from aggregating the embeddings from the *Feature Encoding* and *Feature Aggregator* CCNNs (denoted as *All-Enc*). In fact, the L1 error experiences an increase, while the mean L2 error and its standard deviation exhibit a marginal decrease.

Motivated by these findings, we explored a new series of experiments for the models utilizing the Wav2Vec feature. We trained models employing the Wav2Vec-Z features, derived from the first CCNN and incorporated a CTC branch into the model that utilizes Wav2Vec-C features (*Enc*) since these features are the output from the final layer. A summary of our

Table 4.7: Comparison of different model configurations using Wav2Vec, WavLM, and Whisper audio encodings. The evaluation of model configurations focused on the rig parameter space, considering metrics such as mean temporal L1 and L2 error, as well as mean temporal vertex error in *mm*. Specifically, the evaluation examined the performance of the models in the lower face, lips, and tongue regions of the mesh.

Audio Feature	Model	Rig L1	Rig L2	Lower Face Vtx	Lips Vtx	Tongue Vtx
Wav2Vec	z-feat	4.237±0.683	0.935±0.163	0.073±0.014	0.170±0.036	0.179±0.033
	Enc (c-feat)	4.151±0.652	0.938±0.163	0.072±0.014	0.164±0.036	0.180±0.036
	All-Enc	4.248±0.610	0.934±0.152	0.079±0.014	0.186±0.035	0.185±0.034
	Enc (c-feat) CTC	4.101±0.690	0.926±0.166	0.070±0.014	0.160±0.037	0.173±0.033
	All-Enc CTC	**4.049±0.603**	**0.911±0.149**	**0.069±0.014**	**0.158±0.035**	**0.173±0.032**
WavLM	Enc	4.033±0.655	0.906±0.168	0.076±0.013	0.175±0.035	0.166±0.034
	All-Enc	3.938±0.609	0.883±0.150	0.066±0.013	0.150±0.033	**0.156±0.031**
	All-Enc CTC	**3.831±0.596**	**0.869±0.150**	**0.065±0.013**	**0.149±0.033**	0.157±0.029
Whisper	Enc	4.120±0.731	0.969±0.178	0.077±0.015	0.172±0.034	0.190±0.034
	All-Enc	3.918±0.611	0.882±0.150	0.069±0.014	0.158±0.034	0.168±0.033
	All-Enc CTC	**3.851±0.601**	**0.876±0.149**	**0.066±0.014**	**0.149±0.035**	**0.163±0.029**

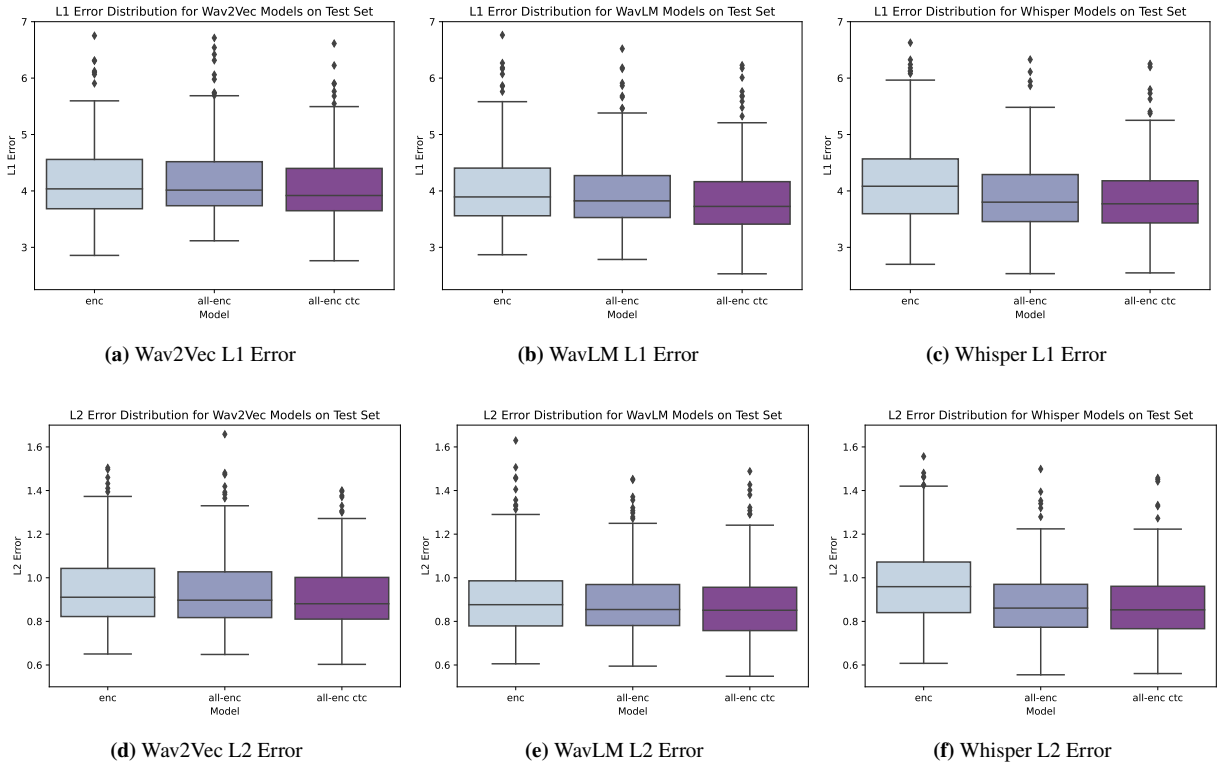
discoveries is presented in Table 4.7. Notably, the poorest performance across all metrics and audio representations was observed in the model trained on the $z-$features. Intriguingly, adding the CTC to the model trained on the C-features enhanced the model's performance. However, combining Wav2Vec features plus adding a CTC yielded the most favorable results. For the mean L1 error, a reduction from 4.151 ± 0.652 to 4.049 ± 0.603 was noted, corresponding to a decrease of 1.025. The mean L2 error declined by 0.027, from 0.938 to 0.911. All the comparisons on the results regarding the different configurations for the models with Wav2Vec have a p-value $< 10^{-6}$ except when comparing the L2 error of the *Enc* vs. the *All-Enc* models, which has a p-value of 0.319 indicating that they have a similar performance from that perspective.

For the WavLM-based models, improvements were consistently observed across both metrics as we combined the Transformer Encoder layers (*All-Enc*) and subsequently regularized the training through a CTC (*All-Enc CTC*). The L1 mean error was reduced from 4.033 to 3.831, a margin of 0.202, and the L2 error decreased from 0.906 to 0.869, a reduction of 0.037. An interesting finding was the WavLM *Enc* model demonstrated better performance than the Wav2Vec *All-Enc CTC* model. We can confidently assert these findings as their p-value is < 0.03 as shown in Table A.1 for the L1 error, and Table A.2 for the L2 error.

With respect to the models trained on the Whisper audio features, the principal enhancement was achieved by merging the embeddings from the Transformer layers, transitioning from *Enc* to *All-Enc*. This reduced the mean error by 0.202 and 0.087 for the L1 and L2 metrics, respectively. A further slight improvement was observed when training the model with a CTC, resulting in mean errors of 3.851 ± 0.601 and 0.876 ± 0.149 for the L1 and L2 errors, respectively. As with the WavLM results, from a statistical analysis, we can assert the trustworthiness of these findings as their p-value is < 0.03 in all cases. As with the WavLM-based models, we can find specific details on the statistical analysis in Table A.1 and Table A.2.

Analyzing the mean errors for L1 and L2, the *All-Enc CTC* model consistently outperformed across various audio

encodings. While the WavLM feature encoding marginally surpassed the performance of the Whisper-based model, the observed differences aren't statistically significant. A direct comparison between the *All-Enc* models using Whisper and WavLM yielded p-values of 0.444 and 0.591 for L1 and L2 errors, respectively. Incorporating the CTC auxiliary task, both errors reported identical p-values of 0.163. This suggests a comparable performance between the two features; however, the addition of the CTC branch distinctly enhances model performance at a confidence level of $p < 0.03$. It's also noteworthy that the Transformer-based feature encodings, namely WavLM and Whisper, significantly outpaced the Wav2Vec model, evidenced by a confidence value of $p < 10^{-5}$. All p-value results can be seen in Table A.4 for the L1 error and Table A.5 for the L2 metric.



(a) Wav2Vec L1 Error **(b)** WavLM L1 Error **(c)** Whisper L1 Error

(d) Wav2Vec L2 Error **(e)** WavLM L2 Error **(f)** Whisper L2 Error

Figure 4.18: Rig parameter L1 ((a), (b), and (c)) and L2 ((d), (e), and (f)) errors distribution from models to predict animation parameters across three audio features: Wav2Vec, WavLM, and Whisper. Training methodologies include the use of final layer embedding (*enc*), a weighted mean of all phase embeddings (*all-enc*), and a multi-task approach that employs Connectionist Temporal Classification (CTC) on the All-Enc model to predict the phone sequences (*all-enc ctc*).

4.7.5 Mesh Vertex Evaluation

Evaluating models based solely on rig parameter error provides a limited perspective, as it may overlook the impact of small errors on the overall animation. Such errors in rig parameters can lead to significant discrepancies in vertex placement, and conversely, minor vertex errors may not reflect underlying rig parameter inaccuracies. To address this, we assessed the models from a vertex error standpoint, identifying regions of improvement during animation generation from predicted rig parameters. Consequently, we adopted the temporal mean vertex error (TMVE) as our evaluation metric, as detailed in Eq. 4.29, to provide a more comprehensive assessment of the models.

$$\text{TMVE} = \frac{1}{N \cdot S \cdot K} \sum_{i=1}^{N} \sum_{j=1}^{S_i} \sum_{k=1}^{K} \left(\mathbf{v}_{ijk} - \hat{\mathbf{v}}_{ijk} \right)^2 \tag{4.29}$$

Where $\mathbf{v}_{ijk}$ represents the ground truth 3D vertex at the k-th position of the j-th timestep in the i-th sample, while $\hat{\mathbf{v}}_{ijk}$ denotes the corresponding predicted vertex by the model to be evaluated. Furthermore, N indicates the total number of samples present in the test set. The variable S_i represents the number of frames in sample I, and K corresponds to the number of vertices to be evaluated.

The vertex error was computed across all the affected vertices of the lower face, including those in the inner mouth, and also over isolated sub-regions, such as the lips and tongue, to gain a deeper understanding of the model's behavior. The distribution of the errors is summarized on the plots shown in the first, second, and third columns, which correspond to the models trained on Wav2Vec, WavLM, and Whisper features, respectively, while the first, second, and third rows display the results for the lower face, tongue, and lips regions in Figure 4.19.

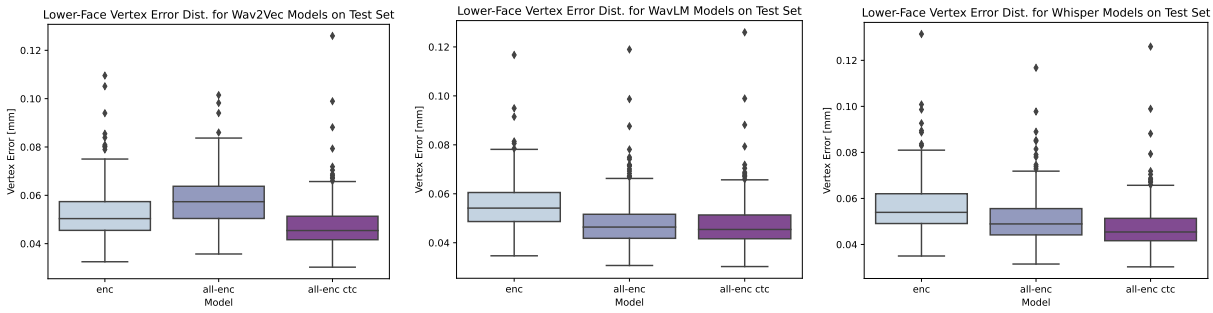
For the Wav2Vec results, the pattern aligns with the L1 and L2 mean errors, where the *All-Enc* configuration deteriorates the model's performance in all three vertex regions. This is vividly visualized in the mesh heatmap depicted in Figure 4.20, where the main error is localized to the lower lip and the front of the tongue, extending from the dorsum to the tip.

In the case of WavLM, the most significant improvement occurs when the encoding layers are combined (*All-Enc*), with a slight overall enhancement observed upon adding the CTC. This trend is also visible in the heatmap shown in Figure 4.21, and the details in Table 4.7 confirm a minor increase in TMVE on the tongue tip. In this video we show a randomly selected sample from the test set, it is noticeable how the model improves by combining the encoder layers and further on improves on the tongue error as we add a CTC.

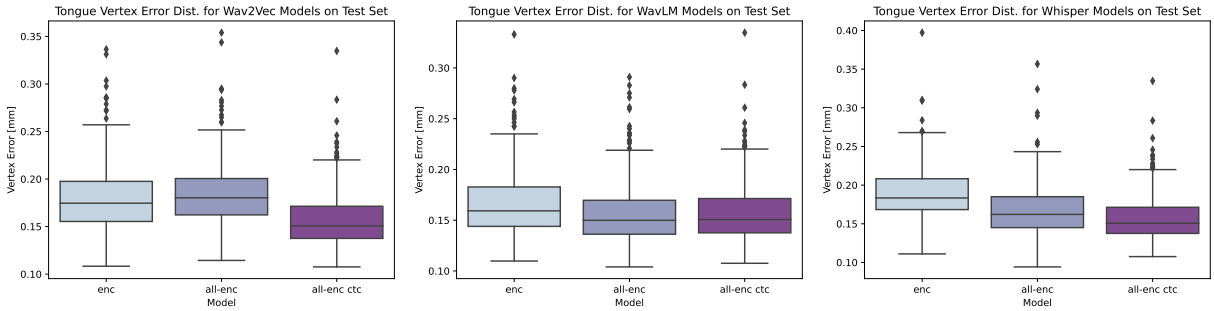
For the Whisper-based models, the improvement appears to be incremental as the transformer embeddings are combined (*All-Enc*) and further regularized with a CTC (*All-Enc CTC*). As evidenced in Figures 4.22a and 4.22b, the lower face consistently improves with the proposed method's aggregation with a particular focus on the chin and lower lip, which is more noticeable from the three quarter view perspective. However, the most substantial benefit is observed on the tongue tip region, where the TMVE on the tongue diminishes from being the worst-performing at 0.190 ± 0.034 to 0.163 ± 0.029. This represents a superiority over its Wav2Vec counterpart by a margin of 0.01. Though still trailing the WavLM *All-Enc CTC* in tongue placement, the error is comparable for the overall lower-face region and lips, with a difference in TMVE of just 0.001. In the following video, we can visualize the impact of such improvements in motion.

Examining vertex errors provides clear insights into the behavior of the models across different configurations and regions. Wav2Vec results emphasize the importance of careful feature selection, as some configurations can harm performance. The WavLM models consistently perform better, especially in the tongue region, showcasing their adaptability. In contrast, the Whisper-based models offer a balanced performance across all regions, focusing on minimizing errors in the tongue area. Furthermore, Wav2Vec indicates that combining the $z-$ and $c-$ features doesn't improve performance; it actually decreases it. However, when regularizing with a CTC, the model's performance improves compared to the baseline (*Enc*) configuration.

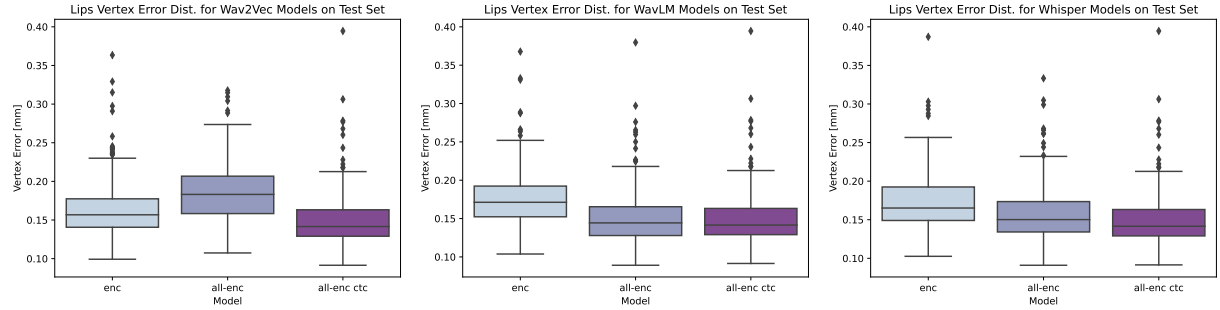
In our assessments on the TMVE, the results exhibit a high degree of statistical significance. For the majority of comparisons, the pairwise t-test yielded p-values below 10^{-4}, indicating robust confidence in the findings as shown in Tables A.3 when comparing the model configurations, and Table A.6 when comparing the different audio features. Nonetheless, as seen in more detail, when contrasting the *Enc* and *All-Enc CTC* configurations with distinct audio features, namely WavLM and Whisper, the p-values were recorded at 0.016 and 0.013, respectively. Despite being larger,



(a) Wav2Vec lower-face **(b)** WavLM lower-face **(c)** Whisper lower-face

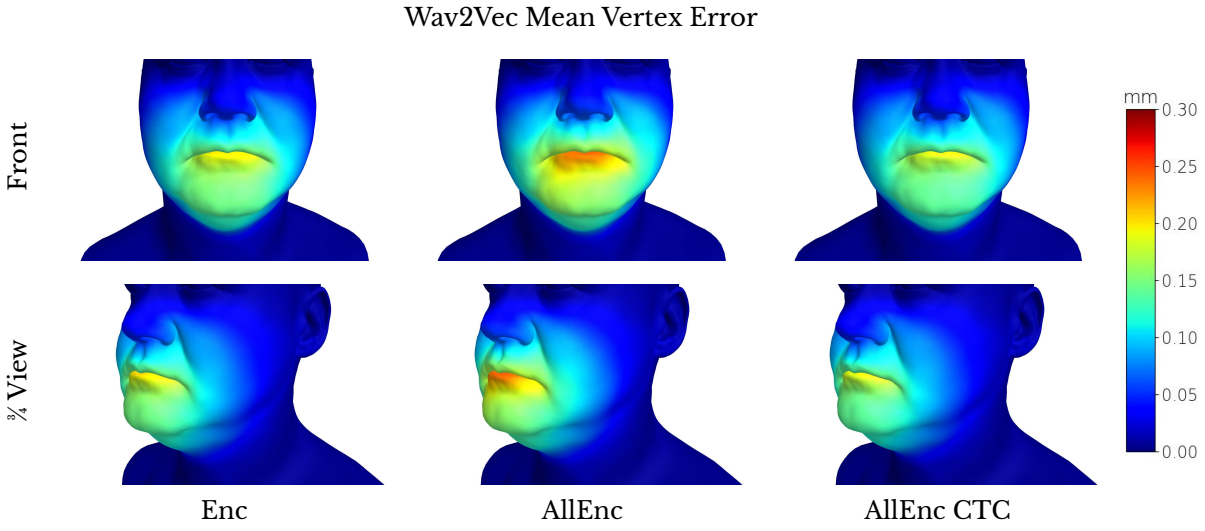


(d) Wav2Vec tongue **(e)** WavLM tongue **(f)** Whisper tongue

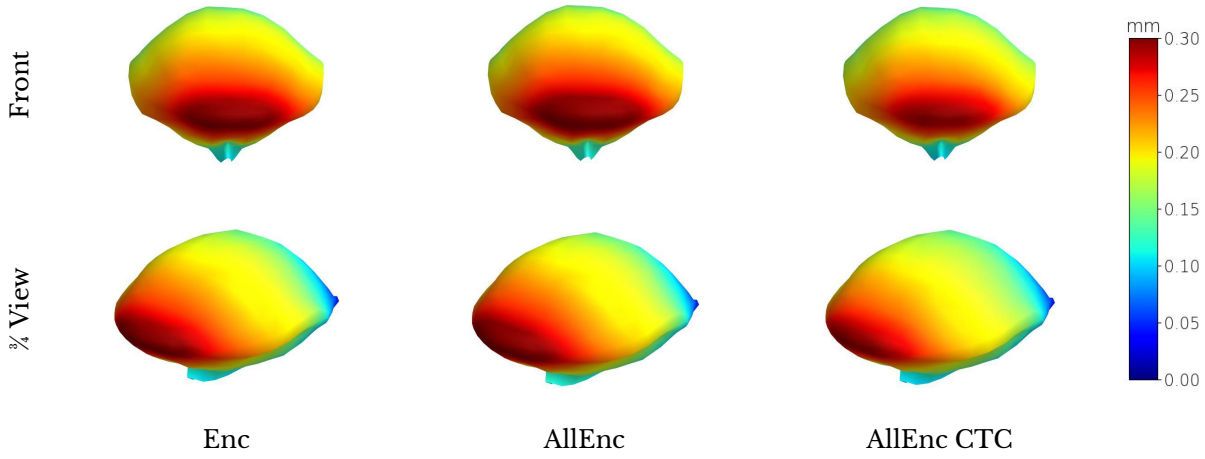


(g) Wav2Vec lips **(h)** WavLM lips **(i)** Whisper lips

Figure 4.19: Lower face ((a), (b), and (c)), tongue ((d), (e), and (f)), and lips ((g), (h), and (i)) regions temporal mean vertex error (*mm*) distribution over the test set. Different model configurations were evaluated to predict animation parameters across three audio features: Wav2Vec, WavLM, and Whisper. Training methodologies include the use of final layer embedding (*enc*), a weighted mean of all phase embeddings (*all-enc*), and a multi-task approach that employs Connectionist Temporal Classification (CTC) on the AllEnc model to predict the phone sequences (*all-enc ctc*).

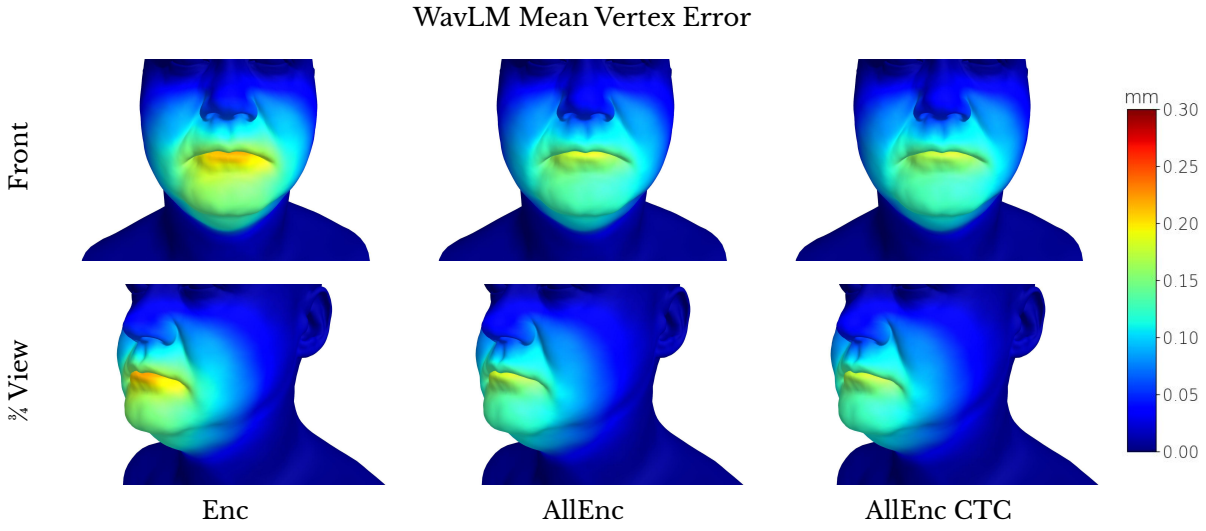


(a) Lower Face

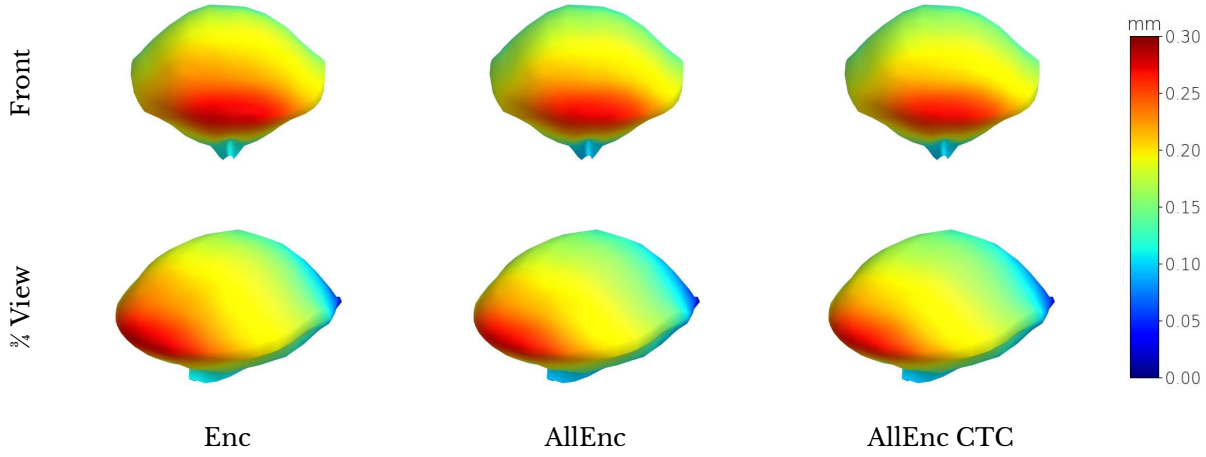


(b) Tongue

Figure 4.20: This figure presents a side-by-side comparison of the temporal mean vertex error for the *Enc*, *All-Enc*, and *All-Enc CTC* models, all trained on Wav2Vec features. Subfigure (a) displays the mean vertex error for the lower face, visualized from two perspectives: a front view and a 3/4 view. Subfigure (b) specifically highlights the temporal mean vertex error of the tongue mesh from these same two viewpoints. It is evident that the *All-Enc* model tends to increase the error around the lip area and the tongue's tip. In contrast, the *All-Enc CTC* model reduces the overall error. This comparison allows for an in-depth model performance analysis across distinct facial regions.

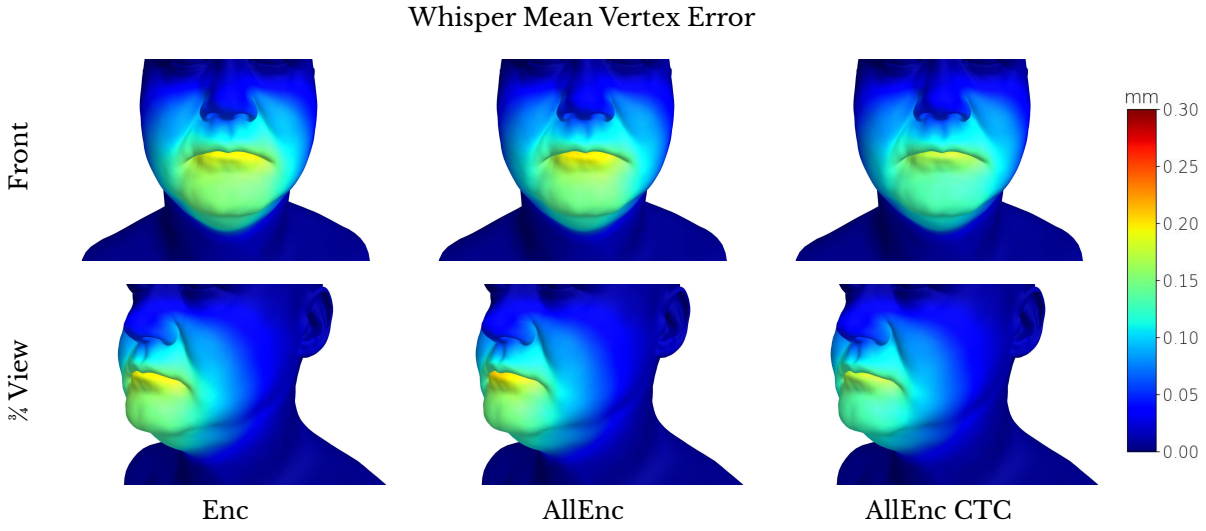


(a) Lower Face

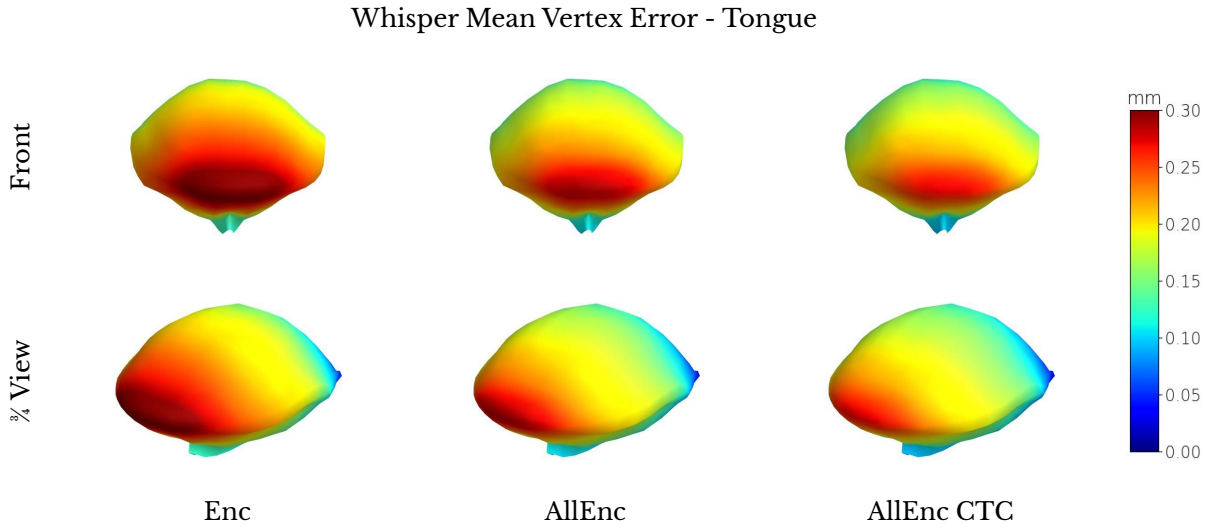


(b) Tongue

Figure 4.21: This figure offers a comparative visualization of the temporal mean vertex error across the *Enc*, *All-Enc*, and *All-Enc CTC* models, each trained on WavLM features. Subfigure (a) illustrates the mean vertex error for the lower face, captured from both front and 3/4 viewpoints. Subfigure (b) specifically emphasizes the mean vertex error within the tongue mesh, displayed from identical viewpoints. The graphics highlight the improvements achieved by aggregating all Transformer Encoder layers from WavLM (*All-Enc*), evidenced by the reduction in overall error across the lower face and tongue regions. While the gains realized from adding a CTC (*All-Enc CTC*) are minimal, they contribute to a discernible, albeit subtle, enhancement in performance.



(a) Lower Face



(b) Tongue

Figure 4.22: This figure provides a detailed comparison of the temporal mean vertex error in the *Enc*, *All-Enc*, and *All-Enc CTC* models, all trained on Whisper audio features. Subfigure (a) portrays the mean vertex error for the lower face, as observed from front and 3/4 views. Subfigure (b) focuses on the mean vertex error for the tongue mesh, visualized from the same two perspectives. Interestingly, the error in the lower face region demonstrates a progressive reduction as we aggregate the Transformer Encoder layers (*All-Enc*) and subsequently apply regularization through a CTC (*All-Enc CTC*). However, the tongue region exhibits a distinct pattern, with the *All-Enc* model resulting in a higher error at the tongue tip, which is marginally reduced in the *All-Enc CTC* model.

Table 4.8: Rating scale provided to participants during the perception user study. Each rating corresponds to the degree of alignment between the animation and audio in the selected video from a pairwise comparison.

Rating	Description
1	Does not match the audio
2	Barely matches the audio
3	Somewhat matches the audio
4	Matches the audio
5	Perfectly matches the audio

these p-values remain below the 5% threshold, further underscoring the reliability of the results discussed in this section.

4.8 User Studies

Generating animations from speech is a complex process that can be assessed both quantitatively and qualitatively. Our previous evaluations have effectively quantified the performance through error measurements in the rig parameter space and mesh vertex space. These metrics provide a detailed understanding of the models' behavior and accuracy. However, since the ultimate goal of our work is to create animations that are perceived as realistic by human viewers, implementing a human perception study becomes an essential next step. Quantitative evaluations, while indispensable, may not fully capture the nuanced aspects of realism that human observers can discern. Through human perception studies, we aim to identify the key factors and metrics that most significantly influence the acceptance and perceived quality of the generated animations. Such insights will not only validate our quantitative findings but also provide a more holistic perspective on the success of the models in creating authentic and engaging animations. By bridging the gap between quantitative measurements and human perception, we hope to enrich our understanding of what makes an animation truly convincing and lifelike.

We devised three user studies to evaluate our proposed models' effectiveness. Firstly, we compared the highest-performing models for each audio feature against animations derived from ground truth rig parameter sequences. Secondly, we evaluated the top models from each audio feature against each other. Lastly, we identified the highest-performing audio feature and compared various model designs within that category to ascertain which was more favorably received by users. We did not conduct any user study focused on the perception of the absence of tongue motion, as its significance has been established in our previous work [106].

The methodology for these user studies was centered around pairwise preference tests, where participants were presented with two videos from a sample from all possible combinations for each comparison required by the study. The user interface from one of the questions in the survey is shown in Figure 3.9a. Participants were instructed to choose the video from left, right, or none. From their selection, they were asked to rate the congruence of the selected animation with the audio on a 5-star scale shown in Table 4.8.

The user studies were conducted via the Prolific platform [125]. In an effort to ensure the most accurate assessment of the generated animations, we applied specific filtering criteria when selecting our participants. Given that our dataset exclusively encompasses English phrases, we limited the participation to native English speakers residing in

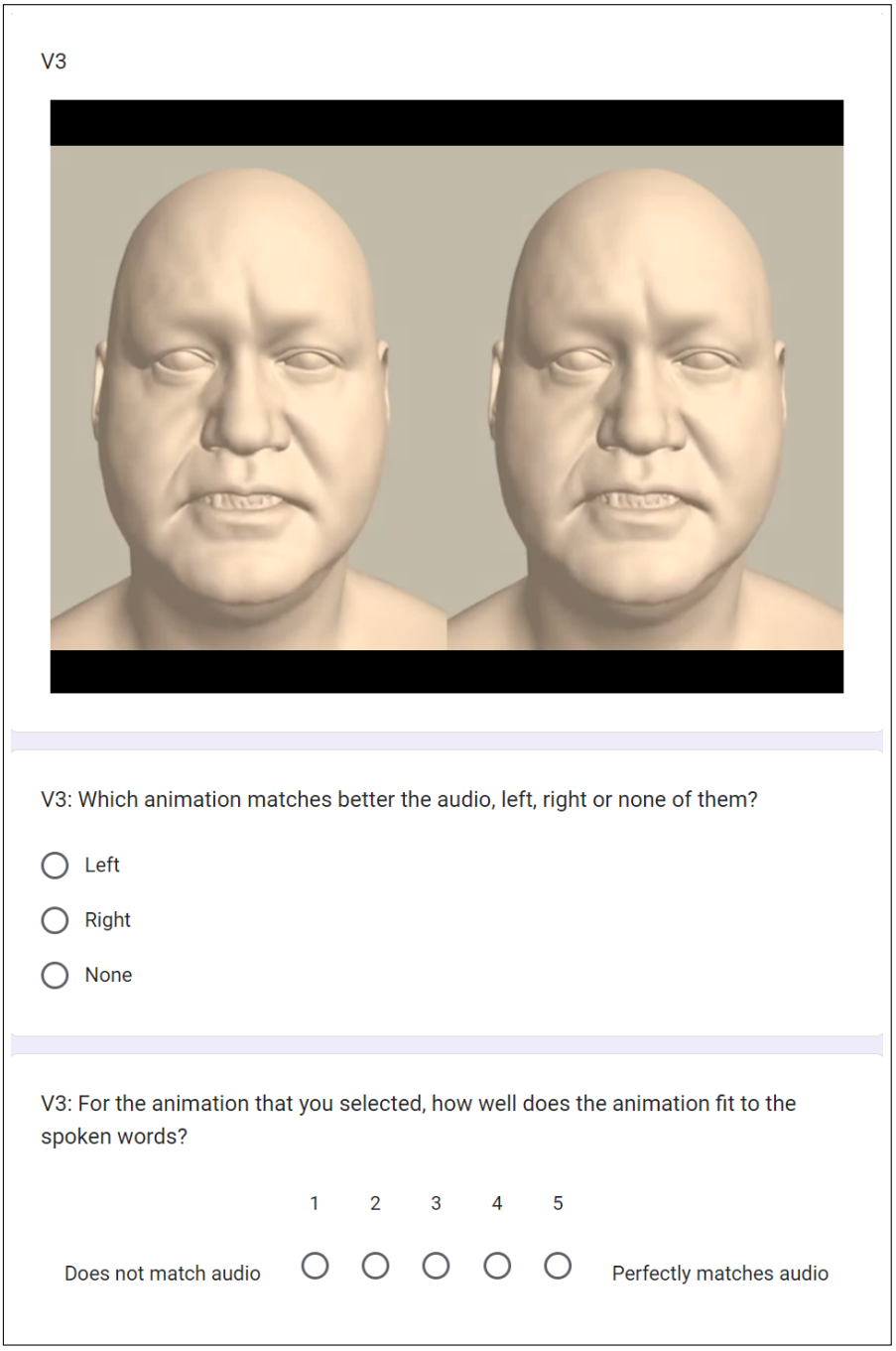


Figure 4.23: Interface for the user studies. Participants were prompted to discern which animation—left or right—was better aligned with the audio they heard. Their task involved evaluating the correspondence of the animation with respect to the audio, with particular attention to the motion of the lips and tongue. Subsequently, they evaluated in a 5-star rating matching level between their preferred animation and the audio.

English-speaking countries such as the United States of America, the United Kingdom, and Australia, among others. This strategic selection of participants ensured a more informed and precise evaluation of our animations.

The design of our user studies drew inspiration from the user studies conducted for the GENEA challenge [165], with

Table 4.9: Transcripts of selected test samples highlighting coarticulation motions with emphasis on diphones featuring open and rounded vowels.

Sample ID	Transcript
0922	Medieval society was based on hierarchies.
1016	Destroy every file related to my audits.
1141	The proof that you are seeking is not available in books.
1192	Dolphins are intelligent marine mammals.
1224	George is paranoid about a future gas shortage.
1466	They were shown how to advance against an enemy outpost atop a cleared ridge.
1625	For roasts, insert meat thermometer diagonally so it does not rest on bone.
1729	Several firms are merchandising enzyme preparation through feed manufacturers.
1837	Differences were related to social, economic, and educational backgrounds.
1846	Asked why, he replied primly: because that's no activity for a gentleman.

the specific aim of obtaining results that are statistically robust enough to enable a comprehensive evaluation of our findings. To facilitate this, videos were synthesized from each selected model for assessment, and they were meticulously rendered to allow for a side-by-side comparative analysis with corresponding counterparts. The first three videos shown to participants were random samples unrelated to the main test, designed to acclimate participants to the evaluation process. The data collected from these initial samples were not included in the study results. To avoid potential bias, the sequence and left-right positioning of the videos were randomized. Furthermore, three control videos featuring non-matching audio were added to the set. These videos, where the audio and animation durations were noticeably incongruent, ensured that participants were attentive throughout the duration of each video. They were asked to evaluate such videos by selecting the "None" option and rest the rating to one star. This methodology safeguarded against participant inattention and minimized bias in the study's results.

Final results were procured by collecting a total of 30 unblemished responses per user study, where all of the aforementioned criteria were successfully met. This accounted for 67% of the total surveys initiated through Prolific. The median time span for survey completion was recorded to be 18 minutes. This approach of carefully curating the responses ensured a meaningful and statistically significant conclusion. As with the video evaluations, this methodology was designed to uphold the integrity of the data collection process and maintain the consistency of the study's outcomes.

4.8.1 Ground Truth Study

Our initial study sought to determine whether users could differentiate between the predicted animations and the ground truth. We produced nine animations from the IMFT'23 dataset test split using both ground truth parameters and predictions from the All-Enc CTC model trained on each of the feature encodings: Wav2Vec, WavLM, and Whisper.

The selection of test samples considered an extensive array of coarticulation motions that users can easily observe and distinguish. Particular attention was given to diphones featuring open and rounded vowels in conjunction with dental consonants, plosives, and sibilants. Detailed transcripts of these samples are provided in Table 4.9.

Table 4.10 shows the user study results. Users consistently favored ground truth animations over predictions, regardless

Table 4.10: Results from a user study employing pairwise selection tests as a sanity check to verify the quality of the models against the ground truth. A set of *All-Enc CTC* models trained using the Wav2Vec, WavLM, and Whisper audio feature representations were compared versus the ground truth animations generated from the IMFT'23 test set. The evaluation of the selected animations was conducted on a 5-point star rating scale, where users were asked to assess whether the selected animation corresponded with the audio. The scale ranged from 1, indicating *Does not match the audio*, to 5, indicating *Perfectly matches the audio*. A rating of 4 was assigned to animations that *Matches the audio*.

Combination	Model	Preference (%)	Mean Rating
1	**Ground Truth**	**62.3**	**4.035±0.799**
	Wav2Vec	37.7	3.942±0.770
2	**Ground Truth**	**62.8**	**4.012±0.814**
	WavLM	37.2	3.922±0.681
3	**Ground Truth**	**57.2**	**4.025±0.886**
	Whisper	42.8	3.831±0.784

of the audio encoding employed. Predictions based on Wav2Vec and WavLM features were identifiable, where 62.3% and 62.8% of the users preferred the ground truth animations, respectively. However, users had more difficulty discerning results produced by the Whisper model, with only 57.2% of users favoring the ground truth animation. Notably, while the mean rating for the ground truth stood approximately at 4.0, the predicted animations' mean ratings were not far behind, with the lowest mean rating of 3.831 attributed to the Whisper-based model.

4.8.2 Audio Feature Study

In our second user study, we aimed to determine user preference for different audio encoding models, namely Wav2Vec, WavLM, and Whisper, all used in training an *All-Enc CTC* model. This model architecture linearly combines the embeddings from all the Transformer Encoder layers through a softmax, while using a CTC to regularize the decoder by aligning the phones to the input speech.

In contrast to the ground truth user study, this study presented participants with a mix of five samples within our test data and five OOD samples, all delivered through the voice of a male actor different from our data capture actor. This approach was taken to mitigate any negative bias arising from using our actor's mesh for the audio presentation.

The results summarized in Table 4.11 revealed several key insights. The WavLM and the Whisper models were generally preferred over the Wav2Vec-based model, by 69.6% and 60.7% of the participants, respectively. However, when it came to in-domain samples, Wav2Vec edged out a slight lead over the architectures encoding the audio through WavLM and Whisper with a preference of 51.7% and 55% preference which could be considered to deploy similar animations from a statistical point-of-view. However, for out-of-domain samples, the animations generated by encoding the audio with WavLM and Whisper were preferred over Wav2Vec by a large margin. Finally, when comparing the WavLM and Whisper models, users showed a slight preference for the WavLM model, with 55.6% favoring it over its Whisper counterpart. Most of the preference came from the OOD samples as 62.7% of users favored it over the Whisper model (37.3%), as the Whisper model was slightly more popular for in-domain samples, with 53.3% of participants favoring it.

The insights gathered from this study indicate that the models based on WavLM and Whisper exhibit a superior ability to generalize across varied contexts. Specifically, it was observed that WavLM's audio features were particularly effective

Table 4.11: Results from a user study employing pairwise selection tests across the Wav2Vec, WavLM, and Whisper audio feature encodings using an *All-Enc CTC* architecture consisting of an encoder-decoder model has multi-task training configuration with a CTC branch. The evaluation was conducted on a 5-point star rating scale, where users were asked to assess the extent to which the selected animation corresponded with the audio. The scale ranged from 1, indicating *Does not match the audio*, to 5, indicating *Perfectly matches the audio*. A rating of 4 was assigned to animations that *Matches the audio*.

		All		In Domain		Out of Domain	
Combination	**Model**	**Preference (%)**	**Mean Rating**	**Preference (%)**	**Mean Rating**	**Preference (%)**	**Mean Rating**
1	Wav2Vec	30.4	**3.756±1.007**	**51.7**	**4.097±0.777**	13.3	2.700±0.900
	WavLM	**69.6**	3.734±0.788	48.3	3.793±0.886	**86.7**	**3.708±0.738**
2	Wav2Vec	39.3	3.698±0.815	**55**	3.879±0.807	26.7	3.400±0.735
	Whisper	**60.7**	**3.780±0.765**	45	**4.000±0.816**	**73.3**	**3.673±0.715**
3	**WavLM**	**55.6**	**3.880±0.923**	46.7	**3.964±0.944**	**62.7**	**3.830±0.907**
	Whisper	44.4	3.833±0.778	**53.3**	3.938±0.704	37.3	3.714±0.839

in creating higher-quality animations when confronted with voices diverging from that of our original actor. This emphasizes the versatility and adaptability of the WavLM model in diverse scenarios.

4.8.3 WavLM Study

In a further analysis, we utilized results from a user study that compared different model designs. The study centered on the WavLM audio feature where the *All-Enc CTC* model was favored over the Wav2Vec and Whisper alternatives. Based on these findings, we expanded our comparison to include different architectures trained on WavLM.

In this third user study, we asked users to express their preferences among animations generated by the *Enc*, *All-Enc*, and *All-Enc CTC* models. Each participant was provided with five in-domain and five out-of-domain samples for assessment. As in our previous studies, we ensured the quality of our data by collecting 30 clean responses, which passed our safeguard protocols. The summarized results can be viewed in Table 4.12.

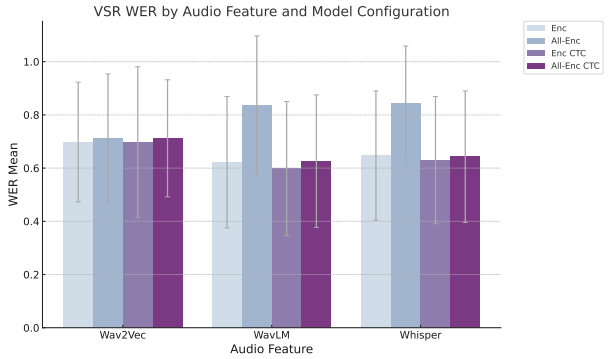
Unexpectedly, we observed that users preferred the animation from the *Enc* model over that of the *All-Enc* model, with 55% of users favoring the former. This outcome contradicts our prior analyses, where we examined the models from rig parameter and vertex error perspectives, which suggested the *All-Enc* model should be superior, as seen in Table 4.7.

Moreover, we found a consistent preference for the model regularized by a CTC that incorporates all the encoder layers from WavLM. Approximately 60% of users favored this model over non-regularized alternatives, with this preference persisting across both in-domain and out-of-domain samples. This finding suggests that adding a CTC through MTL can enhance the quality of the outcomes.

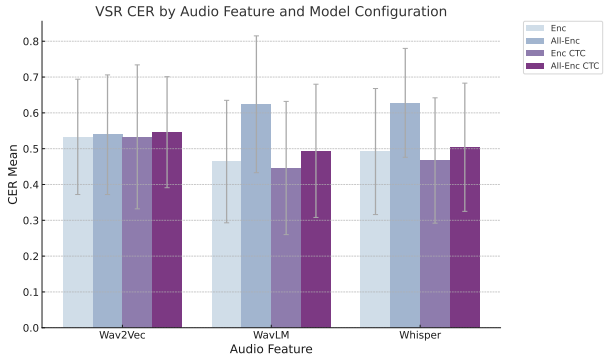
Overall, the study results yielded favorable mean ratings, particularly favoring out-of-domain (OOD) samples over in-domain ones. The average rating approximated to 4, signifying that users found the animations to be in good agreement with the corresponding audio. Notably, this is the same rating that was assigned to the ground truth in our initial user study. This suggests a high level of accuracy in the alignment of animations with their source audio across the model configurations examined bringing one more piece of evidence on the generalization of the model.

Table 4.12: Results from a user study employing pairwise comparison tests across various model configurations using WavLM encoding. The configurations under consideration include: (1) *Enc*, wherein solely the last layer's embedding was utilized, (2) *All-Enc*, which linearly combined the embeddings from all layers of the transformer encoder, and (3) *All-Enc CTC*, where the model underwent multi-task training with a CTC branch. The evaluation was conducted on a 5-point star rating scale, where users were asked to assess the extent to which the selected animation corresponded with the audio. The scale ranged from 1, indicating *Does not match the audio*, to 5, indicating *Perfectly matches the audio*. A rating of 4 was assigned to animations that *Matches the audio*.

		All		In Domain		OOD	
Combination	**Model**	**Preference (%)**	**Mean Rating**	**Preference (%)**	**Mean Rating**	**Preference (%)**	**Mean Rating**
1	**Enc**	**55.2**	**3.951**	**56.0**	3.831	**54.5**	**4.051**
	All-Enc	44.8	3.906	44.0	**4.02**	45.5	3.818
2	Enc	41.0	3.916	38.8	**3.933**	42.8	3.903
	All-Enc CTC	**59.0**	**3.981**	**61.2**	3.873	**57.2**	**4.072**
3	All-Enc	39.5	**4.0**	40.5	**3.957**	38.6	**4.036**
	All-Enc CTC	**60.5**	3.949	**59.5**	3.899	**61.4**	3.989



(a) Word Error Rate (WER)



(b) Character Error Rate (CER)

Figure 4.24: Visual Speech Recognition (VSR) WER and CER metrics for different model configurations and audio features. The models evaluated are Enc, All-Enc, Enc CTC, and All-Enc CTC. The audio features evaluated are Wav2Vec, WavLM, and Whisper. Lower values for WER and CER indicate better performance. Based on the results, the Enc configuration performs best with the Wav2Vec feature, while the Enc CTC configuration performs best with the WavLM and Whisper features.

4.9 Visual Speech Recognition Evaluation

User studies, although invaluable for assessing human perception, present challenges regarding time and financial investment. They demand meticulous design, deployment, and analysis, and it is common practice to compensate participants, with rates varying by country. These factors contribute to a process that does not easily scale in terms of response time or budget, limiting its efficiency.

Given these challenges, we explored the innovative idea of evaluating the model through a lip-reading or visual speech recognition (VSR) model. This approach aims to establish a perception metric in an automated manner, avoiding the need for extensive human-driven perception studies. Under the hypothesis that this method would function as anticipated, we could significantly reduce both time and budget costs.

While we recognize that this first approach may not fully replace traditional user studies, it holds promise as a proxy. By employing VSR-based evaluation, we could sift through and identify the models most likely to resonate with human evaluators. This hybrid approach combines the rigor of quantitative analysis with insights into human perception, providing a more agile and cost-effective way to advance our understanding of realistic speech-driven animation.

For this evaluation we relied on the state-of-the-art Audio-Visual Hidden Unit BERT (AV-HuBERT) model [138]. AV-HuBERT is a self-supervised model trained on unlabeled audio-visual speech data. The model consists of a video feature extractor, an audio feature extractor, and a Transformer backbone. Both extractors generate the representations of their corresponding modalitites at a frame level, which are concatenated and fused into audio-visual features. These features are taken by the Transformer module trained to perform masked predictions of the cluster assignment for each frame. Intially, the clusters are assigned based on the MFCC from the audio, but through the iterations, both modalities are considered. For the purpose of lip-reading the audio-visual features are decoded using a 4-gram language model trained on the LRS3 [2] dataset's training text.

The Lip-Reading Sentences 3 (LRS3) dataset is a large-scale publicly available dataset that collected video clips from TED talks as it ranged a diverse set of speaking individuals with accurate transcriptions, as provided by the organization, rather than transcriptions obtained automatically from ASR systems. Oxford's VGG group created the dataset to research visual speech recognition in mind. The dataset is formed by video clips with audio that contain a single speaking short sentences. The authors provide the precomputed mouth region-of-interest (ROI) crops and their corresponding transcriptions.

Our proposed method to evaluate the speech-animation models through AV-HuBERT is as follows. First, we predict the animation parameters directly from speech. Then, we rendered the videos in a front view to minimize the error from the transformation that AV-HuBERT does to analyze the mouth from a front view. Then, we extract the regions of interest (ROIs) and evaluate each sample through the AV-HuBERT. The pre-trained model used in this study is denominated *AV-HuBERT Large*, which was pre-trained with the LRS3 [2] and VoxCeleb2 [32] dataset and fine-tuned under the 433 hours from LRS3.

The metrics used to evaluate the models are the word error rate (WER) and character error rate (CER). Before computing these values by comparing the predictions from AV-HuBERT for our generated animations against the ground truth transcripts, we normalized the text by removing extra white spaces, converting all characters to lowercase, replacing

digits with words, removing punctuation, avoiding stemming, and keeping the stopwords as those are constantly present in the evaluation data.

As a first approach, we attempted to evaluate the 341 samples from our test set since we counted with the transcriptions from each of our audio samples. The results were unfavorable as the resulting test WER was 0.9577 with a maximum value of 2.66. After further investigation, we found that our data had a large out-of-vocabulary (OOV) ratio with respect to the byte-pair encoding (BPE) tokenizer that the pre-trained model used for predicting the transcriptions has a unigram granularity and a vocabulary size of 973 tokens. When analyzing our test split transcripts, we found an OOV of 0.6326. This is the reason why our mean WER was so high.

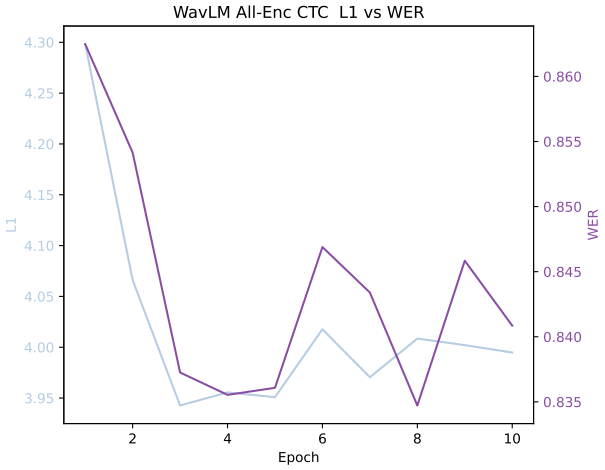
This discovery made us rethink the objective of evaluating through a VSR model. An automatic perception metric would allow us to measure the generated videos from the models if the model can read the lips properly. However, a secondary objective was to set up a benchmark under which the community could compare the results from their Speech-to-Animation models, as the field lacks a common ground to evaluate models and always recur to perception user studies. For this reason, we decided to evaluate the model using samples from the LRS3 dataset, and the objective would be two-fold: first, we would be able to overcome the OOV issue, and second, set up a benchmark for the community as these samples would be OOD of the rest of the researchers in this area of research.

To establish a benchmark, we opted to select test samples from the LRS3 dataset that have their stems present in the pre-trained BPE tokenizer. This decision was taken to avoid any potential OOV issues. From this selection, a total of 402 samples were identified, complete with both video and audio. We refined the beam search to achieve an equilibrium between insertions and deletions, which led to a maximum length of 100 and a beam width of 40. Every model included in our research was assessed by producing animations from the supplied audio and processing them through AV-HuBERT.

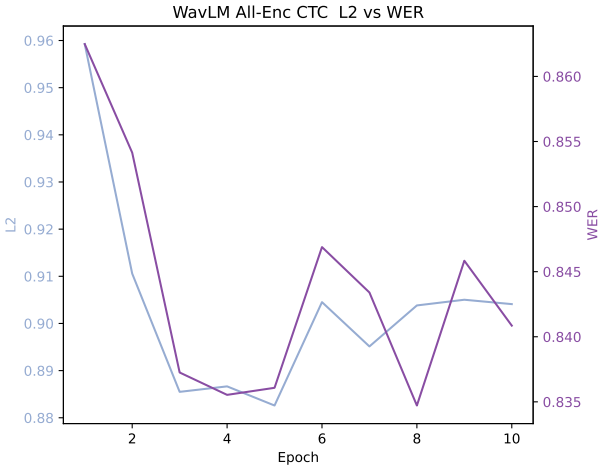
To measure the reliability of VSR metrics, particularly WER, we realized a comparative analysis against L1 and L2 errors over the course of a set of training epochs for the *All-Enc CTC* model for both WavLM and Whisper audio encodings. As illustrated in Figure 4.25, the WER trajectory closely mirrors the patterns observed in the L1 and L2 errors. Notably, as the training approaches convergence, the WER exhibits increased volatility with pronounced fluctuations. Furthermore, in the WavLM model, a distinct observation can be made: as L1 and L2 errors begin to diverge in later epochs, the VSR metric reaches its minimum value, registering a WER of 83.47

Once we have evaluated the WER metric with our best model, we decided to make a full study over all the configurations that have been studied up to this point, the results are summarized in Table 4.13. Upon initial observation, a consistent pattern emerged across all audio features for the model configurations. Specifically, the *All-Enc* configuration consistently ranked lowest in the VSR metrics. Contrary to our preceding evaluations on the L1, L2, and vertex error metrics, as well as the perception user studies, the *Enc* model exhibited superior performance. Motivated by this unexpected outcome, we introduced regularization to the *Enc* models using a CTC to determine its impact, positive or negative, on the model from a VSR perspective. Remarkably, the *Enc CTC* models consistently recorded the lowest WER and CER across all audio features.

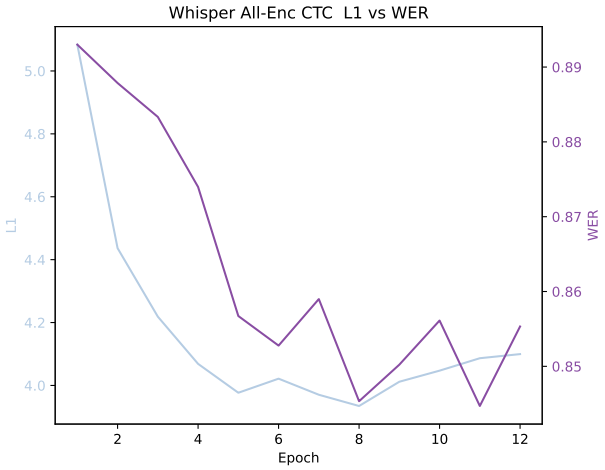
Such inconsistencies invited us to a deeper investigation. We rendered samples from the WavLM *Enc CTC* and found several notable observations. Broadly speaking, predictions from the *Enc CTC*, exemplified here, accurately capture the tongue's movements. However, there exists a noticeable discrepancy in lip motion, which exhibits a somewhat



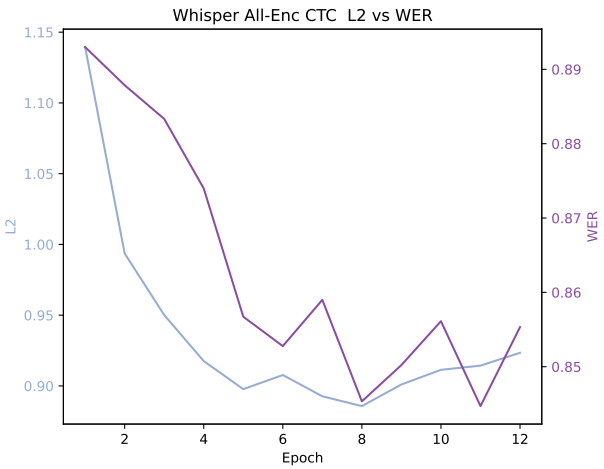
(a) WavLM L1 vs. WER



(b) WavLM L2 vs. WER



(c) Whisper L1 vs. WER



(d) Whisper L2 vs. WER

Figure 4.25: WER comparison vs. L1 and L2 rig parameter errors across epochs for All-Enc CTC models utilizing WavLM and Whisper Audio Encodings. Notably, the WER trajectory mirrors the L1 and L2 error trends in both models, with some intermittent broad fluctuations upon convergence.

Table 4.13: Performance comparison of four different model configurations (Enc, All-Enc, Enc CTC, and All-Enc CTC) with three different audio feature representations (Wav2Vec, WavLM, and Whisper). The models were evaluated using two metrics: Word Error Rate (WER) and Character Error Rate (CER), computed by the AvHuBERT visual speech recognition model. The results are presented as mean ± standard deviation. The best results for each audio feature representation are highlighted in bold.

Audio Feature	Model Configuration	WER	CER
Wav2Vec	**Enc**	**0.698 ± 0.225**	**0.533 ± 0.161**
	All-Enc	0.714 ± 0.247	0.539 ± 0.171
	Enc CTC	**0.698 ± 0.243**	**0.533 ± 0.176**
	All-Enc CTC	0.712 ± 0.240	0.546 ± 0.167
WavLM	Enc	0.622 ± 0.261	0.464 ± 0.191
	All-Enc	0.836 ± 0.214	0.624 ± 0.152
	Enc CTC	**0.598 ± 0.283**	**0.446 ± 0.201**
	All-Enc CTC	0.626 ± 0.252	0.494 ± 0.186
Whisper	Enc	0.647 ± 0.239	0.492 ± 0.175
	All-Enc	0.845 ± 0.220	0.628 ± 0.155
	Enc CTC	**0.630 ± 0.249**	**0.467 ± 0.186**
	All-Enc CTC	0.643 ± 0.247	0.504 ± 0.179

erratic movement in contrast to the results from *All-Enc CTC*, as demonstrated in this video. Shortly, a discernible low-frequency noise resulting in lower VSR metrics. An essential factor to acknowledge is that our original animations are generated at 50 FPS, while AV-HuBERT requires videos at 25 FPS. This difference raises the possibility that the motions during subsampling, visible in this video, may be accentuated, potentially misleading the lip-reading network.

4.10 Qualitative Analysis

Our comprehensive solution achieves lifelike tongue movements. An illustrative analysis can be seen in Figure 4.26, where we present a series of frames chosen based on the set of phones identified by the Montreal Forced Aligner. For a detailed inspection, the tongue is displayed from a side angle, while the face is shown from a three-quarter perspective. The sequence doesn't present any irregularities from the viewpoints of the tongue or the lower face. In fact, the tongue behaves as anticipated: resting on the mouth's floor during open vowels and situating itself between the teeth for dental consonants. It's noteworthy that for sibilants like /s/ and /z/, the tongue's tip doesn't elevate. A plausible explanation might be the limited resolution of the EMA sensors, hindering the capture of such specific positions.

To assess the best results from each audio encoding, we compared them side by side. We found that, in general, all the audio encodings generate plausible animations. However, the WavLM *All-Enc CTC* delivers the best results. When looking at a silence sample, the other two models do not keep the mouth close and generate short motions which are not expected during silence, as seen in this video. For in domain samples. No noticeable difference can be perceived, which correlates with the results from our user studies. The video of one of our test samples can be seen here.

The following out-of-domain sample demonstrates that the method works despite the age of the speaker. Once again, the WavLM *All-Enc CTC* delivers the best results, and it is quite remarkable how you can notice the change in motion between the kid's voice and the adult's voice. The Wav2Vec version of the model suffers from noisy predictions.

Another out-of-domain sample with a woman's voice can be found here. Notice how the motion is transferred regardless of gender, but there seems to be a small motion transfer problem that might be related to the female avatar's rig. This requires further investigation, as the tongue shows a constrained motion.

Our method generalizes enough that is capable of delivering singing animations despite having never seen a singing sample during training. A sample can be found here. This demonstrates how well our proposed method can be employed in practical applications such as video games and entertainment.

Finally, our model is capable of generating plausible animations across other languages such as German, Spanish, Japanese, and Mandarin. Demonstrating its versatility and potential use by a broad audience.

4.11 Discussion

In this chapter, we present the IMFT'23 dataset, a refined version of the IMT'22 dataset, augmented with face reconstruction derived from stereo video. This 3D reconstruction facilitates mesh fitting, subsequently yielding ground truth animation parameters. This enriched data set enabled the training of end-to-end models, predicting animation sequences directly from audio.

Empirically, we showcased that integrating predicted velocity and acceleration errors into the loss during the training of a speech-to-animation network significantly reduces errors in the rig parameter space. This approach yields superior outcomes compared to relying solely on an MSE loss.

Additionally, we investigated two innovative neural audio representations: WavLM and Whisper. Central to both is a Transformer backbone designed for estimating audio representations. Through a comprehensive analysis of their layers, we discerned that combining layer embeddings diminishes errors in both rig parameters and vertex space. Notably, certain layer activation patterns, especially in the mid-layers, were intriguing and called for further exploration. One potential implication of our findings is facilitating network distillation by highlighting the most relevant layers for the speech-animation task.

We then introduce PhISANet, a Phonetically Informed Speech Animation Network, an encoder-decoder model trained within a multi-task learning framework. Two auxiliary tasks were proposed. The first involves an phone classifier, aiming to categorize each audio frame into one of the 88 phones encompassing the phonemes and their corresponding allophones found through the Montreal Forced Aligner. The second task focuses on aligning audio frames with a sequence of phones for a specific window using a CTC. Preliminary investigations revealed the superiority of the CTC over the phone classifier. Comprehensive evaluations across rig parameter space, vertex space, and perception studies affirmed PhISANet's optimal performance when combining all transformer encoder layers from WavLM and training with a multi-task CTC. Result visualizations confirm PhISANet's robust generalization across diverse parameters such as age, gender, language, and even varied audio types like singing.

Furthermore, we proposed the novel concept of employing visual speech recognition systems for evaluating speech-to-animation models. Our exploration centered on AV-HuBERT, an audio-visual speech recognizer capable of processing solely visual input, effectively serving as a lip-reading network. Preliminary findings suggested a correlation between

model performance metrics in the rig parameter space. However, subsequent results deviated from our initial evaluations. Our observations indicated that while the top-performing models, as per AV-HuBERT, did not fully concur with our findings, the integration of a CTC did enhance model performance. This novel approach holds promise, potentially diminishing the need for extensive user studies by automatically computing perception scores from a lip-reading network. These results underscore the potential for continued research in this domain.

Lastly, user studies revealed an area for enhancement. The average rating hovered around ”Matches the audio,” falling short of the aspirational ”Perfectly matches the audio.” Further exploration determined that lip rotation accuracy remains an issue. This is attributed to lip deformations being primarily driven by lip contour tracking and the inherent motion presented by the MetaHuman rig upon mesh fitting to the reconstructed landmarks. This observation highlights the necessity for in-depth research, specifically targeting the precise capture of lip deformation.

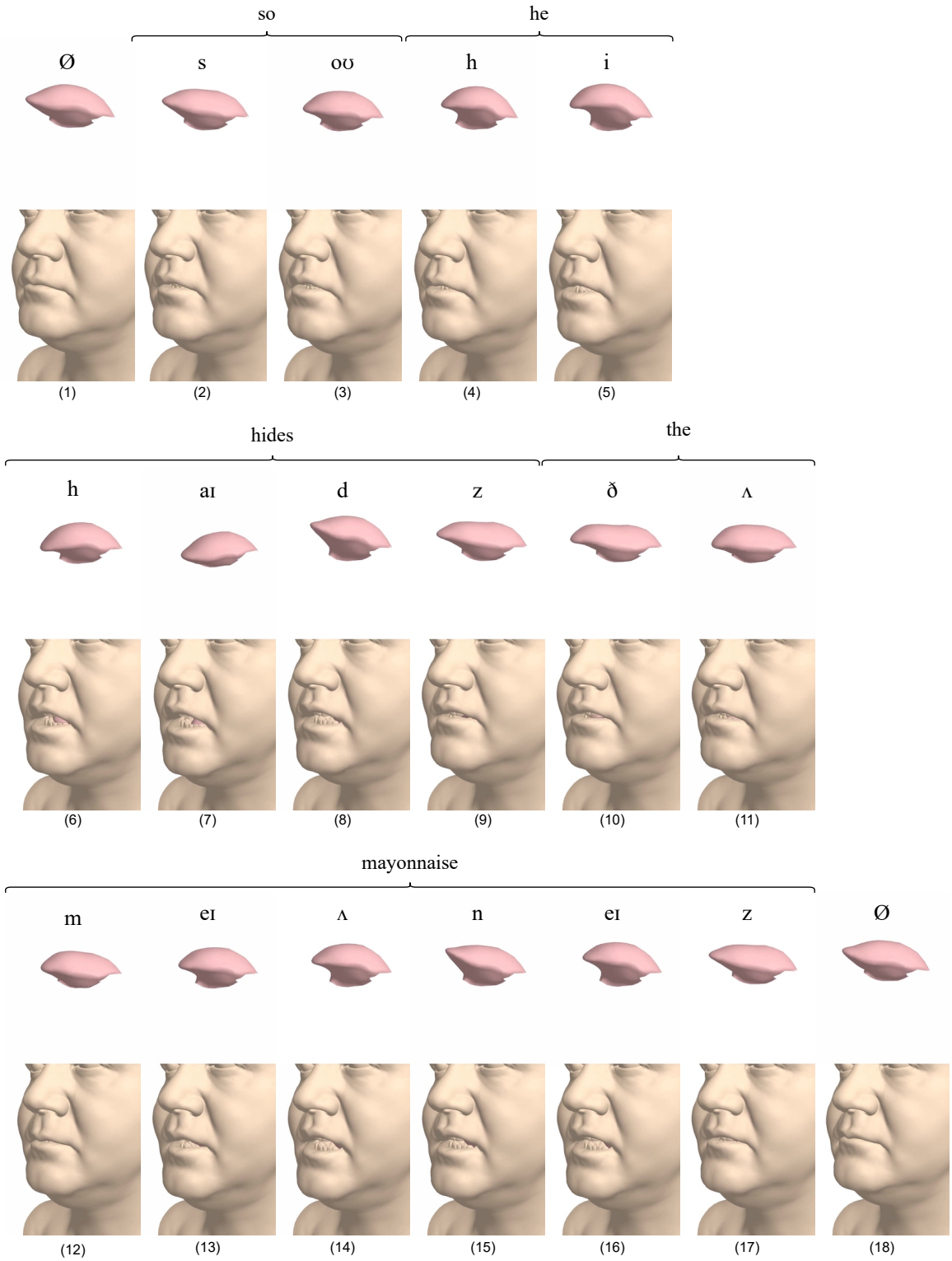


Figure 4.26: Predicted sequence by our top-performing model, *WavLM All-Enc CTC*, for the sample sentence "So he hides the mayonnaise.". The corresponding phone is presented on top of the tongue rendered from a lateral view, while the face is presented in a three-quarter view. The symbol '∅' stands for silence. The displayed frames are carefully chosen within the boundaries predicted for each phone by the *Montreal Forced Aligner*. Observe the synchronized movements of the lips, jaw, and tongue, which follow the expected pattern for each phone.

CHAPTER 5

TOWARDS REAL-TIME STREAMING SPEECH-ANIMATION

In the fields of entertainment, telecommunications, healthcare, and education, the prospect of a real-time speech-animation model holds transformative potential. The ability to animate a digital character *in tandem with speech* is a desired innovation. Although real-time speech animations using neural networks are not entirely novel as — Hong *et al.* [63] pioneered an MLP-based model that converted 7 audio frames into motion units for an iFACE solution with a mere 100 ms delay. Similarly, Axelsson [8] in his thesis detailed a model that generates lip motion from phonemes, subsequently classified into visemes. Malcangi *et al.* [101] presented an endeavor to predict lip motion, crafting a lip driver system that leverages both fuzzy logic and neural networks, demonstrating generalization across speakers and languages.

The quest for real-time, expressive speech animation has also seen developments via hybrid methodologies. Some have combined rule-based systems with predictive systems that categorize facial gestures over audio frames, ensuring a delay not exceeding 300 ms. An intriguing alternative to neural networks has been the deployment of Gaussian Mixture Models (GMM). Luo *et al.* [99], for instance, innovatively adapted GMMs to map audio to visual features. They ingeniously integrated prior visual features to avoid discontinuities in the following animation. In a study by Websdale *et al.* [157], the efficacy of lookahead information introduced during model prediction was explored. Their findings highlighted that their Bi-LSTM-based model maintained consistent performance for lookaheads of 70 ms or 170 ms. Moreover, they highlighted the ITU G.114 recommendation [71], which states that users don't perceive disruptions in telecommunication when delays remain below 200 ms. Taking a new approach, Pham *et al.* [122] introduced an end-to-end model. This model processes spectrograms using a CNN and subsequently, through an RNN, translates these intermediate outputs into micro facial action units, driving a 3D blendshape facial model. Impressively, their reported inference time from spectrograms to action units stands at approximately 5 ms, largely attributable to the efficiency of the convolutional layers.

Our objective differs from the preceding research as we aim to design a real-time solution based on our dataset. Notably, none of the prior works have attempted to animate the tongue in a real-time solution. Furthermore, the prevalent methodology leans towards generating animations based on visemes or action units, inevitably pushing these animations towards a synthetic spectrum. In contrast, our dataset facilitates the learning of natural lip motion by encompassing not just the lips but also the adjacent regions like the cheeks and the chin. As an initial exploration, we will harness the capabilities of convolutional layers, ensuring we remain within a 200 ms budget. Nevertheless, laying down a robust streaming model framework seems indispensable as groundwork for our ensuing research.

5.1 Streaming Model Design Framework

The discussion of real-time streaming models requires an understanding of the concepts of cold start, latency, and receptive field. These elements significantly influence the model design and weigh the advantages and disadvantages of each design decision in relation to these concepts.

Cold Start & Warm-up

A *cold start* in a streaming model refers to the initial phase when the model has not received any data from a live stream, such as an audio signal. The system enters into a warm-up period when the data starts filling up a data buffer but has not yet accumulated sufficient information for the model to start making meaningful predictions. As a matter of fact, the predictions during a warm-up period might be noisy and affect the system's performance. Common practice dictates that the output should be omitted during the warm-up period, making the system's response time equal to the warm-up period. Since a streaming model typically operates over a fixed window size of data or buffer, it requires a certain amount of data to fill this initial window before it can commence processing and generate accurate predictions based on the conditions under which the model was trained. The cold start phase presents challenges such as increased initial latency, proper model parameter initialization, initial data stream connection, and setting up an initial alignment of the incoming data within the processing window.

Latency

Latency in a streaming model refers to the time delay between the arrival of new data from the live stream and the expected output by the system or model. Several factors can influence this delay. For instance, the model's required lookahead, which indicates how much future data the model needs to make a prediction, is part of the total latency of the system. Another factor that adds to the total latency is the model's inference time for processing the incoming streaming data. Additional latency can also result from post-processing performed over the model's predictions, e.g., rendering the rig parameters predicted by a speech-to-animation streaming model.

Receptive Field

The *receptive field* in a streaming model refers to the range of input data that influences a particular output prediction. In other words, it represents the number of previous, current, and lookahead data points that the model takes into account when making a prediction for a given timestep t. The size and direction of the receptive field are critical design parameters. Determining the extent of past and future information can determine the accuracy of the model's prediction. A larger receptive field typically allows the model to capture more complex and long-term dependencies in the data, while a smaller receptive field focuses on more immediate and short-term patterns.

The interplay of cold start, latency, and receptive field is crucial in designing a streaming model or system, with the balance among these elements influencing the object of design's overall performance and responsiveness. For instance, the duration of the cold start period is affected by the receptive field of the model, as a larger receptive field requires more initial data for an accurate first prediction. Latency, or the delay in producing predictions from an incoming data stream, is influenced by factors such as the lookahead in the receptive field design. More lookahead data can improve model performance but may also increase the delay in delivering a prediction. Conversely, reducing these parameters may decrease both the cold start period and latency, but potentially at the expense of predictive accuracy. Therefore, designing a streaming model requires a comprehensive understanding of these concepts and carefully considering the trade-offs to meet the application's specific requirements, such as real-time responsiveness or the capability to model complex dynamics in the data stream.

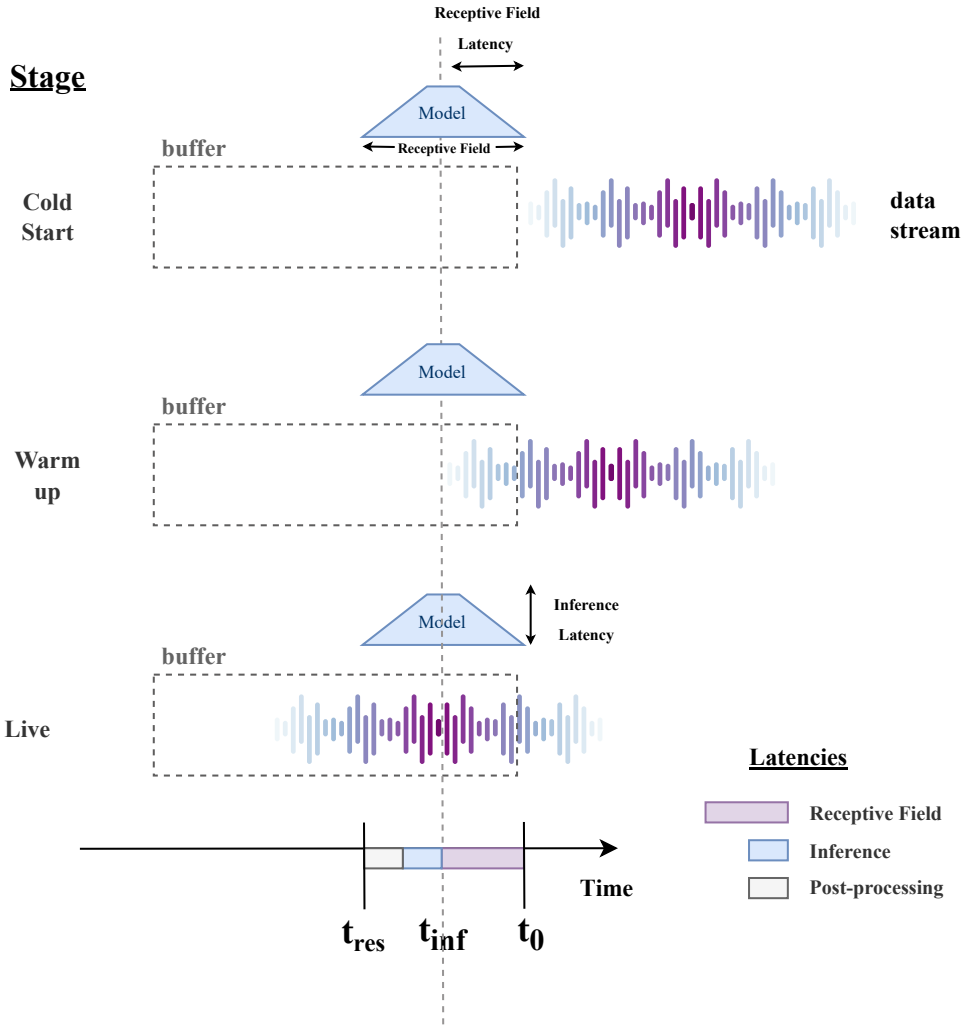


Figure 5.1: This figure illustrates the various stages and components of a streaming model.

To provide a clear understanding of the concepts, let's examine an example depicted in Figure 5.1. At the beginning our system has no data to process, hence no output can be delivered, at this point the system is in the *Cold Start* stage. As incoming data is streamed into our buffer, a specific amount of data must fill the buffer before processing by the designed streaming model. The period when the buffer is being filled is named as the *Warm Up* stage. During this stage,

the model's predictions may be erratic as the model might not be designed to overcome the issue of missing data. In system's design this is an important period where the model can be left *mute* and not output any response until the buffer is filled, or make it predict under some considerations. Once the buffer is filled with the appropriate amount of data, the system is said to be in the *Live* stage, and the prediction model can start processing the input stream to output predictions related to t_0.

In an ideal scenario, the model does not require a lookahead to make the right predictions at time t_0 or even forecast the output for time $t_0 + 1$. This type of model is commonly referred to as a causal model. A causal model would predict the output of the leading frame as new data enters into the buffer. However, practical experience often indicates that adding lookahead information enhances the system's performance, depending on the task. If the models were to operate causally, the latency would consist solely of the duration needed to obtain a single data frame for processing, thereby reducing the latency to one frame.

In our specific example, the model requires lookahead, introducing an inherent latency l_{rf} to the receptive field. It's important to recognize that the total latency of the system must account for not only l_{rf}, but also the inference latency l_{infer} and the post-processing latency l_{pp} if an additional step is needed to achieve our final task. The sum of all latencies results in the response time t_{res} of our system, which is the total time the system delivers a response related to the data from t_0.

5.2 Audio Encoding Selection

For the offline model, we explore different configurations based on three different audio encoding models: Wav2Vec, WavLM, and Whisper. Through a meticulous evaluation, we determined that the CTC-MTL model with WavLM encodings delivers the best animations from a quantitative and qualitative perspective. At the beginning of this research, these models were chosen based on their performance metrics and capability for audio encoding. However, transitioning to a real-time streaming application brings challenges, notably in processing speed and system latency.

Given our prior work with these models, the next step involves assessing their performance in a real-time setting. Key considerations guide this analysis:

- **Real-time Responsiveness**: Our offline model was effective, but real-time platforms emphasize processing speed. Profiling helps gauge the models' suitability for real-time requirements.

- **Hardware Synergy**: The offline model was developed for a specific hardware configuration. Real-time consumer platforms, however, have varying hardware needs. Evaluating performance on CPU and GPU platforms helps refine our model choice.

- **Scalability**: This study includes configurations not previously explored, specifically smaller versions of the WavLM and Whisper models. This provides insight into each configuration's performance, guiding our model selection for real-time applications.

For this analysis, we profiled each model using the PyTorch internal profiling tools, considering smaller versions of the

Table 5.1: Profiling results of various audio encoder models on CPU and GPU for encoding 1 second of audio. The table contrasts the performance of three prominent models—Wav2Vec, WavLM, and Whisper—in various configurations. The measured times are presented in milliseconds (*ms*) to highlight the relative computational efficiency of each model.

Model		CPU [ms]	GPU [ms]
Wav2Vec	large	209.019	7.427
WavLM	base	184.070	13.082
	large	169.044	19.743
Whisper	small	846.706	45.948
	medium	2104.000	128.550

WavLM and Whisper models than the ones evaluated for the offline model. The profiling was performed in a machine with an Intel(R) Xeon(R) Gold 6146 CPU @ 3.20GHz CPU and an NVIDIA Quadro RTX 8000 with 48GB of memory. The results, presented in Table 5.1, display the computational times required by each audio encoder model to process one second of audio on both CPU and GPU platforms.

The profiling results show that Wav2Vec (large) model has the fastest GPU processing time at 7.427 ms, while the Whisper (small) model has the quickest CPU processing time at 846.706 ms. It's worth noting that the Whisper (medium) model has the longest CPU time at 2.1 s but remains efficient on the GPU at 128.55 ms. Overall, GPU processing times are faster across all models. The Wav2Vec and WavLM models, in particular, have reduced encoding times, suggesting their suitability for real-time applications. Given these findings, we chose the Wav2Vec model for its speed and capability of processing directly the audio signal since the first layer without any pre-processing step as done by Whisper. The following sections will discuss the application and outcomes of using Wav2Vec in our real-time setting.

5.3 Streaming Audio Encoding

Using Wav2Vec features for real-time audio processing brings its own set of challenges. Primarily, the design of Wav2Vec introduces a temporal bias, mainly due to the group normalization (GroupNorm) [162] incorporated in each convolutional block. GroupNorm operates by normalizing specific groups of channels, as represented by the equation $y = \frac{x-\mu_g}{\sigma_g+\epsilon}$, where μ_g and σ_g are the mean and standard deviation of the group, respectively, and ϵ is a small constant. In Wav2Vec, the normalization considers the temporal progression of the audio, which means both the features and their occurrence in time influence the model's output. This inherent bias can complicate real-time audio processing. To counteract these effects, we've introduced a specialized feature selection and processing method depicted in Figure 5.2.

The receptive field of Wav2Vec spans approximately 890 milliseconds, dictating the minimum chunk of audio data needed to produce a relevant output. Given this, we designed a simulation where an audio stream of the same duration as our audio frames are captured and processed. Features are extracted from overlapping windows, each advanced by a stride equivalent to our audio frame duration, which is 20 ms. This method is pivotal for achieving our goal of minimal latency audio processing.

For training our streaming models, we extract a 3D tensor, T, from overlapping context windows. This tensor is defined as $T \in \mathbb{R}^{F \times C \times W}$, where F is the feature size, C is the context window size, and W represents the number of such

windows. The count of these context windows, W, is computed from the original sequence length S at 50 FPS, window size w, and stride s, using the formula $W = \left\lceil \frac{S-w}{s} \right\rceil + 1$. In our particular case, we set a window size $w = 45$ and a stride $s = 1$.

After obtaining the tensor T, a design choice was made to downsample the audio by skipping alternate frames, thus reducing the frame rate from 100 FPS to 50 FPS. This decision ensures the key features from Wav2Vec are preserved while facilitating real-time processing. Moreover, this approach maintains the original 512-D dimensionality, allowing for a more compact decoder model.

In the final step, we select all features from the initial context window and the last frame features from all subsequent context windows. This results in an audio feature $z_{rt} \in \mathcal{R}^{S \times F}$, which is then used for training the streaming decoder models.

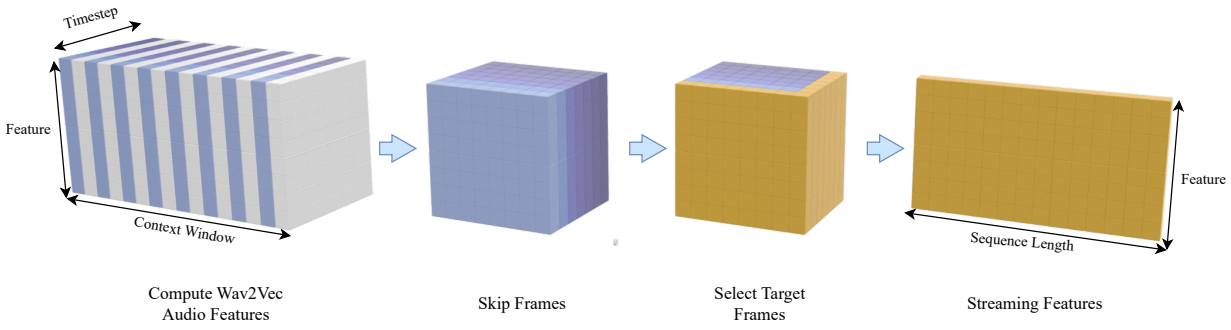


Figure 5.2: Procedure for Real-time Streaming Audio Feature Extraction and Selection. Initially, Wav2Vec features are inferred from the pre-trained model using overlapping context windows. One frame is omitted to downsample from 100 FPS to 50 FPS. The final feature from each window is then chosen to construct the streaming features, factoring in the bias from context computation. Subsequently, a streaming model is trained using the pre-computed streaming features.

5.4 Model

Since we are seeking a real-time solution that processes incoming audio streams as close as we can to the leading frame, we investigated CNN-based models due to their memory efficiency, speed, and overall robust performance when compared against a GRU. The goal of our first experiment was to evaluate the relationship between the required lookahead and the size of the receptive field to achieve smooth and consistent speech animations. For this reason, we trained two distinct CNN types: a conventional configuration with lookahead and a causal model, specifically, the Temporal Convolutional Network (TCN) [92]. The standard CNN is formed by blocks of one-dimensional convolutional layers, followed by a ReLU [3] activation and a GroupNorm [162] layer to enhance convergence and stability. The TCN, while sharing a similar configuration of building blocks, it differs from a regular CNN by having a constant kernel size and a dilation factor, enabling the model to widen its receptive field in a causal form as we add layers.

The receptive field size for the CNN, w_{CNN}, is usually calculated as:

$$w_{CNN} = 1 + \sum_{i=1}^{l} (k_i - 1) \times \prod_{j=1}^{i-1} s_j \tag{5.1}$$

Table 5.2: A summary of the TCN model's performance metrics across varying numbers of layers. The table illustrates the relationship between the number of layers, receptive field (RF) in frames and milliseconds, RF latency, and the respective L1, L2, and temporal mean vertex errors (TMVE). Notably, the receptive field expands with additional layers, influencing the model's error metrics.

Num. Layers	Hidden Size per Layer	RF [frames]	RF [ms]	RF Latency [frames]	RF Latency [ms]	L1 Error	L2 Error	TMVE
1	[256]	3	60	1	20	4.815 ± 0.681	1.079 ± 0.155	0.0857 ± 0.0774
2	[512, 256]	7	140	1	20	4.545 ± 0.674	1.052 ± 0.156	0.0810 ± 0.0729
3	[512, 256, 128]	15	300	1	20	4.534 ± 0.673	1.051 ± 0.157	**0.0802 ± 0.0721**
4	[512, 256, 256, 128]	31	620	1	20	4.533 ± 0.677	1.055 ± 0.160	0.0804 ± 0.0723
5	[512, 512, 256, 256, 128]	63	1260	1	20	**4.460 ± 0.705**	**1.039 ± 0.164**	0.0804 ± 0.0724

While the TCN's receptive field is given by:

$$w_{TCN} = 1 + 2 \times \sum_{i=0}^{l-1} d^i \times (k-1) \tag{5.2}$$

Where, l represents the number of layers, d is the dilation factor, k_i denotes the kernel size of the i^{th} layer, and s_j the stride of the j^{th} layer. In our experiments, for the TCN we held the dilation factor constant at $d = 2$ and the kernel size at $k = 2$ through all the layers. For both architectures, the stride was $s_i = 1$ through all the layers. We trained a variety of models with different numbers of layers to measure the impact of the receptive field and how the lookahead information influences the models' performance. Additionally, we aimed to discern the optimal receptive field size, ensuring that the time window effectively captures articulatory elements, aiding in generating accurate speech animations.

It is important to notice that the calculations described in eq. 5.1 and eq. 5.2 correspond to the receptive field while training the network. However, at inference time, since we want to deliver a prediction as close as possible to the leading frame, we only consider the last pass over the incoming sequence according to the kernel size of the first layer as the receptive field latency which is $\lceil k/2 \rceil$. This only applies to the conventional CNN model as the TCN's receptive field's latency remains the same regardless of the number of layers since is a causal model.

As the TCN model's receptive field doubles with each layer, we try to mimic a similar pattern by doubling the kernel size on the CNN model. This strategy has also proven to develop in good results as it is the main core idea of work like SGConv [95] and FFTNet [75].

Table 5.2 offers a comprehensive summary of the TCN model's performance when varying the number of layers. As anticipated, increasing layers in the causal model augments the receptive field. For instance, a single-layered TCN has a receptive field of 60 *ms*, which escalates to 300 *ms* and 620 *ms* for three-layer and four-layer networks, respectively. A notable observation is a similarity in performance between the three-layer and four-layer networks, especially given that their L1 and L2 errors are very close (4.534 ± 0.673 and 1.051 ± 0.157 for the three-layer network compared to 4.533 ± 0.677 and 1.055 ± 0.160 for the four-layer one). Introducing a fifth layer pushes the receptive field to 1.26 seconds, resulting in the lowest mean L1 and L2 errors of 4.460 ± 0.705 and 1.039 ± 0.164, respectively. However, the vertex error remains relatively unchanged across the models. Notably, the optimal temporal mean vertex error (TMVE) is achieved with the three-layer TCN at 0.0802 ± 0.0721. The model seems to achieve the best balance of performance and complexity when designed with three layers.

Table 5.3 shows the impact of layer variation of the CNN model, which increases the receptive field of the model,

Table 5.3: Performance evaluation of the CNN model across different numbers of layers and kernel sizes. The table presents the model's receptive field, RF latency, and associated error metrics. It's highlighted that increasing the layers influences the receptive field and the subsequent need for input padding, affecting the overall performance.

Num. Layers	Kernel Size per Layer	RF [frames]	RF [ms]	RF Latency [frames]	RF Latency [ms]	L1 Error	L2 Error	TMVE
1	[3]	3	60	2	40	4.660 ± 0.708	1.067 ± 0.163	0.0842 ± 0.0757
2	[7, 3]	9	180	4	80	4.402 ± 0.712	1.018 ± 0.164	0.0789 ± 0.0702
3	[15, 7, 3]	23	460	8	160	$\mathbf{4.275 \pm 0.703}$	$\mathbf{0.991 \pm 0.163}$	$\mathbf{0.0758 \pm 0.0676}$
4	[31, 15, 7, 3]	53	1060	16	320	4.297 ± 0.708	0.996 ± 0.165	0.0758 ± 0.0678

resulting in an increase in latency to deliver a prediction. The tendencies observed in the TCN as depicted in Table 5.2 are parallel for the CNN model. As the number of layers increases, so does the receptive field, consequently introducing more latency as denoted in the *RF Latency* column. Notably, the four-layer model, with a receptive field of 1060 ms, exceeds the 1 s subsampling window set for training. This necessitates the first layer to apply padding to the inputs, introducing artificial information to the network and adversely affecting performance metrics. The three-layer configuration stands out with an L1 error of 4.275 ± 0.703, L2 error of 0.991 ± 0.163, and TMVE of 0.0758 ± 0.0676. However, these values still lag behind the offline encoder-decoder model, which employs Wav2Vec $c-$features and a bidirectional GRU, as detailed in Table 4.7 from the previous chapter.

5.4.1 Reincorporating the idea of Multi-task Learning CTC

In Chapter 4, we highlighted the efficacy of regularizing the rig parameter sequence decoder in PhISANet using a CTC to align the phone sequences corresponding to the audio while training the speech-to-animation network. This regularization enhanced the model's performance, leading to better animations, as supported by a series of user studies. Building on this concept and the insights from the previous section, we explored whether a streaming model, like the one under examination, could similarly benefit from CTC regularization. Integrating the CTC into the training process is straightforward, given that the streaming model is trained on pre-computed streaming features. These features, detailed in Section 5.3, align seamlessly with the offline data.

For consistency in our approach, we conducted a parameter sweep over the weighting coefficient of the auxiliary task using identical value sets. These experiments yielded results similar to those observed for the offline model, as presented in Table 4.5). A notable observation was a saddle point occurring at $\lambda_{CTC} = 0.004$, resulting in a slight improvement over the original results.

Upon determining the optimal CTC coefficient, we revisited the CNN model experiment by setting them as a multi-task learning model with a CTC. While there was a modest enhancement across the board, it wasn't as significant as the improvements observed in the offline model, as detailed in Table 5.5. Notably, including the CTC helped mitigate the challenges previously encountered with the four-layer model. This adjustment led to the lowest L1 error of 4.238 ± 0.722, L2 error of 0.983 ± 0.170, and TMVE of 0.0755 ± 0.0676, making it the top performer in this set. However, despite these improvements, the four-layer model's latency of 320 ms renders it unsuitable for consumer-based applications due to noticeable delays of almost 1/3 of a second. On the other hand, the three-layer model, with its more manageable latency of 160 ms, an L1 error of 4.270 ± 0.704, L2 error of 0.990 ± 0.164, and TVME of 0.0767 ± 0.0684, offers a more practical solution for real-world applications.

Table 5.4: Performance metrics of the streaming model under varying λ_{CTC} regularization coefficients. The table showcases the relationship between the weighting coefficient of the auxiliary task and the resulting L1 and L2 error on the rig parameter space, as well as the temporal mean vertex error (TMVE). A saddle point in performance is observed at $\lambda_{CTC} = 0.004$.

λ_{CTC}	**L1 Error**	**L2 Error**	**TMVE**
0.5	4.403 ± 0.702	1.018 ± 0.164	0.0798 ± 0.0714
0.25	4.366 ± 0.691	1.009 ± 0.160	0.0787 ± 0.0703
0.125	4.333 ± 0.723	1.002 ± 0.169	0.0783 ± 0.0698
0.064	4.311 ± 0.691	0.995 ± 0.161	0.0771 ± 0.0687
0.032	4.296 ± 0.699	0.992 ± 0.163	0.0767 ± 0.0684
0.016	4.276 ± 0.709	0.990 ± 0.167	0.0769 ± 0.0685
0.008	4.273 ± 0.703	0.990 ± 0.165	0.0768 ± 0.0684
0.004	$\mathbf{4.270 \pm 0.704}$	$\mathbf{0.990 \pm 0.164}$	$\mathbf{0.0767 \pm 0.0684}$
0.002	4.270 ± 0.702	0.990 ± 0.165	0.0768 ± 0.0685
0.001	4.272 ± 0.701	0.990 ± 0.163	0.0789 ± 0.0702

Table 5.5: Performance evaluation of the CNN trained with a CTC via MTL across different numbers of layers and kernel sizes. The table presents the model's receptive field, RF latency, and associated error metrics. It's highlighted that increasing the layers influences the receptive field and corrects when the receptive field surpasses the training window. There is an overall improvement when compared versus the non-CTC version of the models.

Num. Layers	**Kernel Size per Layer**	**RF [frames]**	**RF [ms]**	**RF Latency [frames]**	**RF Latency [ms]**	**L1 Error**	**L2 Error**	**TMVE**
1	[3]	3	60	2	40	4.608 ± 0.700	1.058 ± 0.162	0.0842 ± 0.0756
2	[7, 3]	9	180	4	80	4.392 ± 0.705	1.016 ± 0.163	0.0794 ± 0.0708
3	[15, 7, 3]	23	460	8	160	4.270 ± 0.704	0.990 ± 0.164	0.0767 ± 0.0684
4	[31, 15, 7, 3]	53	1060	16	320	$\mathbf{4.238 \pm 0.722}$	$\mathbf{0.983 \pm 0.170}$	$\mathbf{0.0755 \pm 0.0676}$

5.5 Qualitative Results

For a comprehensive evaluation of our models, it's crucial not only to rely on metric analyses but also to inspect the generated outputs visually. Initially, we analyzed how a 4-layered TCN, a 3-layered CNN, and its CTC counterpart responded to white noise, which is analogous to silence for human perception. The visual outcomes are accessible via this link. Notably, the TCN model exhibits a consistent mouth motion, while the CNN models display lesser movement, resulting in a mildly open mouth and lower lip motion during silent intervals across all models. This behavior could present challenges while generating seamless animations.

Another visual examination was conducted using an in-domain sample from our test set. The video available here highlights the correct motion patterns across the three models. Although the TCN model displays undesired motions, the *CNN-CTC* model showcases more pronounced lip and tongue movements, especially during open vowels or diphones requiring lip protrusion. This phenomenon becomes particularly evident in an out-of-domain sample featuring an infant's voice.

Comparative analysis of animations from the three streaming models with a singing sample reveals that while jitter seems to be reduced, it is still present. Notably, the coarse movements of the lips, jaw, and tongue are as anticipated, though they are accompanied by constant unintended motions, breaking the illusion of the animation. This discrepancy becomes more evident when compared against our best-performing offline model, which integrates the combination of the encoder layers from WavLM Transformer and is regularized with a CTC, as seen in the same singing sample. The offline model's articulation is more refined, with a broader range of jaw and tongue movements, which are less evident in the streaming model.

5.6 Discussion

In this chapter, we focused on developing a real-time streaming solution for animating speech with lifelike characters, exemplified by MetaHumans [47]. Our choice to utilize the Wav2Vec model for audio encoding was influenced by its efficiency: it processes one-second context windows in just 7.427 *ms* on an NVIDIA Quadro RTX 8000 GPU. Notably, this speed is double that of the WavLM model.

Our experimental approach aimed to understand two key factors: the influence of lookahead and the impact of a model's receptive field size. To gain comprehensive insights, we compared the performance metrics of two well-established architectures: the CNN and the TCN. Our findings indicated that both lookahead and causal models, when augmented with additional layers, exhibited increased receptive fields and improved performance. However, it's imperative that the receptive field remains within the duration bounds of training samples. Exceeding this limit adversely affects performance, as evidenced by elevated L1 and L2 mean errors and the TMVE with the CNN.

Building on experience from PhiSANet, we further investigated whether multi-task learning—using a CTC auxiliary task to align allophonic sequences—could enhance the performance of the top-performing three-layer CNN with a 160 *ms* lookahead. This exploration yielded marginal improvements across all metrics.

Based on our visual assessments, it becomes evident that the animations generated by the TCN model exhibits significant

noise, rendering them unsuitable for practical applications. Incorporating a CTC with the CNN model improves the representation of specific phonetic motions, particularly while articulating rounded and open vowels, and a slightly improved tongue positioning. Nonetheless, the resultant animations, while improved, still contain a degree of noise that makes them challenging to deploy in industry-standard scenarios.

Based on the data presented in this chapter, we're optimistic about the potential of a real-time streaming speech-animation model capable of delivering fluid and natural facial and tongue motions at a rate of 30 FPS. The model's inference time is notably under 20 *ms*. Moreover, when considering the receptive field latency of 160 *ms* inherent to the design of our best-performing model in relation to the incoming audio signal and assuming frame rendering stays below ten milliseconds, it becomes evident that a model rendering speech animations at 30 FPS is within reach with a cumulative delay of 190 *ms* with respect to the incoming audio signal. However, further research and meticulous engineering are important to compress and optimize these models to be pushed into production.

CHAPTER 6

CONCLUSIONS

In this thesis, we proudly present and release the IMT'22 and IMFT'23 datasets [106]. The IMT'22 is a novel speech-to-tongue mocap dataset, which includes 2.55 hours of speech data and corresponding inner-mouth motion captured through an EMA device. This dataset is unique because it employs parasagittal sensors and the canonical midsagittal setup used in speech pathology studies. Our diphonic analysis [105] has confirmed the significance of these sensors in obtaining accurate 3D deformations of the tongue by analyzing its curvature and width. This dataset facilitates the continuous animation of the tongue in 3D models and, to some extent, drives the 3D model's face. Furthermore, it provides reliable data for training models that precisely estimate jaw motion - a critical component in speech-to-animation for realistic animations.

As a first approach, we developed a two-stage speech-to-animation pipeline, which served as a preliminary approach. The first stage involved training an encoder-decoder model to predict EMA landmarks from raw speech signals, which were subsequently transformed into the space of the 3D model of the actor who provided the data. In the second stage, we employed a second-order optimizer that minimized the distance between the transformed predicted landmarks and their corresponding landmarks on the 3D model. Our experiments showed that neural audio representations outperformed traditional audio feature representations like phone representations and MFCC for speech-to-animation purposes. Moreover, through an ablation study of different audio representations, we found that a Wav2Vec model for audio encoding and a five-layered bidirectional GRU for landmark prediction achieved the best results with a temporal MSE of less than 1.77 on our test set and proved to generalize across different voices across gender, age, and language mainly due to our transfer learning approach by using audio feature representations from self-supervised models trained on hundreds of hours of speech from a diverse set of speakers. To our knowledge, this is the first work to introduce neural audio representations into the speech-to-animation field.

We discovered a compelling insight while evaluating our two-stage pipeline using a pairwise preference user study. Participants preferred to observe an animated tongue during speech animations, even when the animation did not correspond precisely to the actual speech. Importantly, our predictions were still preferred over the mismatched animations and sometimes confused with animations generated from ground-truth data.

Our initial approach demonstrated the feasibility of generating animations using the IMT'22 dataset. However, due to significant computational demands during the second stage and the limited realism of solely using EMA sensors to drive facial animation, we sought to approach the problem from an end-to-end perspective to generate more realistic animations in less time. We created the IMFT'23 dataset to facilitate this approach, which builds on the IMT'22 by including visual data captured during EMA sessions. This dataset contains audio samples and 3D landmarks of the tongue, face, and lips. We generated a sequence of rig parameters per audio sample using the L-BFGS optimizer as we did with the EMA data. This allowed us to train end-to-end models capable of predicting the animation parameters directly from speech audio, which could then be fed into a rendering engine.

We comprehensively analyzed robust and recent neural audio representations trained on large-scale audio datasets comprising thousands of hours for various tasks to achieve the most optimal end-to-end model. In addition to our initial investigation on the applicability of Wav2Vec to Speech Animation, we explored other models, such as Microsoft's WavLM and OpenAI's Whisper. The results presented in this study demonstrate that any of the neural representations we explored can effectively map audio inputs to smooth and continuous animations for the lower face, including the tongue.

Traditionally, speech-to-animation models have often utilized Mean Squared Error (MSE) loss to train the models, incorporating either the instant velocity and/or acceleration of the mesh vertices or even a proxy of the acceleration without any analysis or justification for adding such terms. In this work, we conducted a comprehensive study that empirically validates the superiority of incorporating two weighted loss terms for the velocity and acceleration of the animation parameters instead of relying solely on MSE loss. Our experiments showed, for example, that the temporal mean L1 error on the rig parameters space with a relative improvement of approximately 10%, while the L2 approximately improves by 2% across all the audio features of this study, by just enforcing a weighted error of the predicted velocity and acceleration with respect to the ground truth to the training loss. We conclude that for any of the neural-based audio feature representations presented in this work, adding these terms delivers better results than just optimizing through the MSE loss.

Additionally, we introduced a novel model called PhISANet (Phonetically Informed Speech-to-Animation Network). PhISANet harnesses the richness of features from all the transformer encoder layers in modern audio feature representations and incorporates phonetic information through a Connectionist Temporal Classifier (CTC) to align the phone sequence extracted from audio using a forced aligner from our IMFT'23 dataset. Our experiments prove that the regularization of the decoder in PhISANet produces realistic and continuous speech animations for the lower face, exhibiting natural and plausible tongue motion with a sub-millimeter temporal mean vertex error on the mesh space. The addition of phonetic information into the network improves the motion on the lower lip and the region between the lower lip and the chin for all the models, regardless of the audio feature representation used to encode the speech signal, while also reducing the overall error of the tongue with a high emphasis on the tongue tip.

Through meticulously crafted user studies, we identified potential areas for enhancing the realism of generated data. On average, users felt that while animations created using ground truth aligned with the audio, they weren't flawless. Another study highlighted that animations produced by PhISANet, especially when augmented with WavLM audio features, were more compelling for out-of-domain samples with a preference greater than 60% over the models trained with Wav2Vec and Whisper. This indicates their potential for broader acceptance in real-world applications. Furthermore, this highlights the idea that animations derived from a model trained with a CTC auxiliary task are generally more favorably received than those without it, as the preference rate increased from 45.5% to 57.2% of the users.

As speech-only animations reach new levels of realism, there is a growing need for metrics that can effectively assess the plausibility and realism of these generated animations. To address this, we propose incorporating visual speech recognition networks, particularly lip-reading networks, to quantify the animations' authenticity and fidelity. This approach ensures that the generated animations closely align with the input utterances provided to the model. To establish a standardized benchmark for speech-to-animation models and facilitate meaningful comparisons between different methods, we advocate for adopting the state-of-the-art AvHuBERT model. Although our experiments were limited with a lower WER of 59.8% and CER at 44.6%, the results are compelling and open the door to keep on exploring and settling a benchmark that will provide a common ground for the community to evaluate and compare the different approaches to solve the problem of speech animation. Such a benchmark becomes increasingly essential as the field progresses toward more realistic animations.

In our concluding experiments, we showcased the potential of a real-time model that animates the lower facial region, including the lips, jaw, and tongue, from a continuous audio stream signal. We favored the Wav2Vec encoder for audio processing due to its speed advantage over both the WavLM and Whisper encoders. Through our tests, the CNN emerged as an adept model for generating speech animations, outpacing the GRU in terms of speed. While quantitative metrics suggest that this model doesn't quite match the performance of our top offline variant, a qualitative assessment of out-of-domain samples reveals the animations to be convincingly realistic during regular speech. Nonetheless, to improve the animation quality, further exploration is required. Potential improvements could stem from integrating attention mechanisms [154] into the CNN or delving into other causal architectures suitable for our final task of generating realistic-looking speech animations.

CHAPTER 7

WHERE DO WE GO FROM HERE?

In this study, we've adopted several strategies from the field of Automatic Speech Recognition (ASR), such as the utilization of large-scale pre-trained models, regularization of the model through a Connectionist Temporal Classification (CTC), and the application of Visual Speech Recognition (VSR) models to assess our model's performance using a quantifiable perception metric. However, this represents only a fraction of the full scope of possibilities in this research area. Given the constraints of a Ph.D. program, we are unable to explore all potential avenues.

In this final chapter, we will delve into potential pathways to advance the creation of realistic facial animations from speech. While we've made considerable progress in the preceding chapters, there remain aspects that demand further exploration and development. In the subsequent sections, we will discuss ideas that were not fully realized during this study and suggest directions for future research.

7.1 End-to-End Model

Data availability poses a significant constraint when training an end-to-end model that directly maps audio-to-face animation without relying on pre-trained audio feature representations. Our study achieved realistic and continuous results by leveraging robust audio feature representations and the IMFT'23 dataset, which comprises 2 hours and 17 minutes of paired audio and corresponding face/tongue motion data. However, this data is not enough to train an end-to-end model that generalizes as well as our current encoder-decoder approach since the data only has one speaker. To overcome this problem, we could train an end-to-end speech-to-face-animation model by bootstrapping a synthetic dataset generated by our current best model, taking as input the audio from a large corpus with a diverse set of voices and generating the corresponding sequence of rig parameters. Then, we would have enough data to train a generalizable end-to-end model. LibriSpeech [113] is a large-scale and commonly used dataset for speech-related tasks due to its clean alignment between speech and text and variability of the voices due to the speakers' performance while reading a book. Another potential dataset that could be used to bootstrap our model is VoxPopuli [156] formed by over 400 thousand hours of unlabeled speech from 23 languages taken from the European Parliament event recording. Including different languages could enrich our model and make it more robust to multilingual inputs. Nonetheless, it is worth

noting that this approach is limited by the upper bound of the best model and will transfer the inherited errors learned into the new end-to-end model.

7.2 Data Augmentation

Another interesting avenue to augment the IMFT'23 dataset is through voice cloning. Voice cloning is the process of synthesizing or replicating an individual's voice over a given input, such as text or speech. This field of research is filled with exciting results [74], [91], [133] where the synthetic voice sounds the same as the target voice to an untrained human ear. For our task, we could do voice cloning from speech to speech, keeping the pace and prosody to augment the data by generating new audio samples with a different voice from our original actor and assigning the corresponding face and inner mouth animation parameters.

7.3 Future Data Capture

The IMFT'23 dataset, discussed in Chapter 4, has demonstrated its value in creating authentic animations of the lower face and tongue. However, there is potential for further enhancement in future capture sessions. The limitations of the existing model could be overcome by collecting data from a more diverse group of actors—varying in gender, age, and spoken language. Future sessions could also focus on capturing facial and oral movements across a broad spectrum of emotions and extreme motion situations. This would provide the model with richer data to extrapolate from in extreme scenarios such as a Viking's fierce battle cry or an exuberant shout of victory in a video game.

Up to this point, we have achieved realistic lip surface deformations by tracking the vermillion border and integrating rotation data from the EMA sensors. Nonetheless, certain areas need enhancement, especially in ensuring the mouth closes fully and in the movement dynamics of the lower lip. As observed in this video, the lower lip in our dataset consistently protrudes more than that of the original actor, causing the lower teeth to be more visible than in natural scenarios. This deviation could result from our prevailing optimization approach or the solution of the MetaHuman rig parameters. Further investigation and refinement are essential to address these issues.

Moreover, we need to improve the mouth motion for non-speech audio, e.g., coughing, as seen on this video sample. Note how the lips follow a natural motion when they are about to open and close the mouth (00:12), as well as the jaw motion through the whole sequence. However, the motion of the cheeks and the tongue placement leave room for improvement since the animation does not reflect the constriction that occurs during an expectorating cough. Moreover, more complex lip motion, such as the one occurring during lips moistening with the tongue or pursing of the lips during a deep thought process, has yet to be captured and learned to be generated from their corresponding audio. A potential solution might be integrating markers similar to those used on the face into the 3D reconstruction process. However, this method, while an advancement, may not be ideal due to the sparsity of the points, necessitating further research from a computer vision perspective.

Furthermore, the inclusion of non-speech gestures into the audio-to-face animation dataset would add further richness. Nonverbal communication plays a pivotal role in human interaction [44], and accurately reproducing it based on non-speech audio cues can significantly augment the authenticity of animations. Initial steps in this direction could involve guiding eye gaze and synchronizing suitable head movements with the sound. For example, an exuberant shout might

be associated with a certain degree of eye closure and neck muscle tension. Likewise, an affirmative sound such as *mmm-hmm* or *uh-huh*, especially in Western English-speaking contexts, would typically be accompanied by a nod of the head. However, as highlighted by Fares *et al*. [50], it is crucial to acknowledge that effectively capturing the motion of the upper face and head necessitates both contextual information and low-level audio features. In the framework of a neural network paradigm, these can be extracted from a sentence embedding produced by a Language Model trained on vast amounts of data and a convolutional-based network instead of using the F0 formants.

7.4 More Complex Models

Collecting richer and more diverse data will require models of greater complexity. As we broaden the variance in voice and emotional speech, having more control over the type of animation being generated would be beneficial. An initial approach to consider is the use of a Conditional Variational AutoEncoder (cVAE), as suggested by Aylagas *et al*. [9]. In this context, the conditional state could encode a limited and predefined set of emotions. A progressive move in this direction would be to regulate the generated articulation—from subtle animation through regular motion to extreme gestures—based on the end user's requirements or application. Another approach to this problem is through a well-designed multi-task learning approach, different from what we have achieved through PhISANet; all tasks are considered as main tasks requiring vast experimentation and fine-tuning.

Recurrent Neural Networks (RNNs) have shown promise in generating fluid and continuous animations, although they are not without limitations. Overfitting can lead to the problem of generated animations from speech collapsing towards a mean face. Additionally, their design inherently benefits from a bidirectional pass, rendering them less suitable for streaming applications.

An innovative category of RNNs known as RWKV was recently introduced [118]. This model was designed to be trained similarly to a Transformer, but it uniquely operates without the need for any Attention, thus reducing the amount of memory required during training. At inference time, it behaves like an RNN and can implicitly handle an "infinite" context length due to the ingenious design of its time and channel mixing layers. Although its initial design targets the creation of Large Language Models, there is potential for RWKV to be employed in speech-to-animation tasks. This presents an exciting opportunity for future exploration.

7.5 Model Compression and Computational Reduction

The current real-time model suits solutions where the video is shown at 25 frames per second (FPS). However, modern games, video conferencing, and streaming applications usually run at 60 FPS or beyond.

Recent work on talking faces [80] has shown how to compress and speed up the Wav2Lip model [124] by reducing the number of parameters by a factor of 10 and achieving a 28-fold computational reduction. The main ideas are to remove residual connections, reduce the number of parameters by training a smaller model through multi-granular online knowledge distillation, and post-training quantization of the layers of the face decoder.

The authors of [80] found that quantizing all the layers in the decoder to INT8 or FP16 from FP32 deteriorates the model's performance. However, they found that layers closer to the final output with a larger precision, such as FP16 or

FP32, bring the best results. They found that a mixed model with the initial layers in the decoder with an INT8 precision and layers with an FP16 precision on the output block results in a model with a performance close to the original model, which was trained with a precision of FP32.

Quantizing the layers of the real-time model proposed in this work is an attractive avenue for future work. This would reduce the model's memory footprint on the machine and transmission costs whenever it is updated online in any application. It would also potentially result in a faster model, closing the gap for a solution closer to 60 FPS.

One thing to consider is that the streaming model presented in this work is formed by mainly three convolutional neural networks (CNNs), where the first two correspond to the Wav2Vec audio feature representation computation and the third CNN is the animation decoder formed by three convolutional and one self-attention layer, as described in Chapter 5. Based on the results presented by [80], it seems worth exploring the quantization or distillation of the Wav2Vec model since the animation decoder layers are closer to the output.

7.6 Audio Feature Representations

The training of neural networks with a large corpus to improve the performance of the models is well-known in the Computer Vision community. ImageNet [38] introduced this practice when a huge improvement occurred in image classification when they provided a large labeled corpus with a hierarchical structure that contains over 1.2 million images, and AlexNet [88] proved that CNNs could be trained with a corpus of that magnitude. Despite the years, this phenomenon continues to occur in more challenging tasks such as text-to-image generation with corpus such as LAION-5B with 5.8 billion samples open to the public and work like GigaGAN [77] have benefited from. A similar pattern is being replicated within the audio processing community, advancing from the popular LibriSpeech dataset formed by approximately 960 hours of book reading speech to models like Whisper, which was trained on over 680 thousand hours of audio of speech from several languages and non-speech audio from quiet to noisy environments. These audio feature representations have proven useful for the Speech-to-Animation field of study, as presented in this work. Without a doubt, newer feature representations yet to come in the following months on models trained on hundreds of thousands, or even millions of hours, and will be even more robust to noise, able to detect non-speech signals, and suitable to encode the prosody and emotion than what we currently have. This new feature representation will help us achieve a richer animation of the full. I believe the next generation of audio features will capture some sense of semantics if trained as done for AudioPaLM [133].

7.7 Interactive Agents

With the advent of Large Language Models (LLMs) like Bloom [134], Chat-GPT4 [112], and PaLM2 [54], the creation of imaginative, responsive, interactive agents that behave naturally is becoming more attainable. A common method for creating speech-based interactive agents would adopt a pipeline approach, as proposed by [35], but with the added element of an LLM. This process would entail four key stages:

1. An Automatic Speech Recognition (ASR) module to transcribe incoming speech.

2. An LLM to interpret the user's input and formulate a response.

3. A Text-To-Speech (TTS) module to generate a speech signal of the response.

4. A speech-to-animation module to visualize the interactive agent.

A pipeline as the one described is feasible with great quality results. However, a recently proposed model named AudioPaLM by Rubenstein *et al*. [133] shows that speech embeddings can be integrated with pre-trained language models to achieve end-to-end speech-to-speech tasks like simultaneous spoken language translation. One intriguing idea worth exploring is modifying this model class to simultaneously generate animation as the speech is being produced instead of making it a subsequent step after the speech signal has been created, as described above.

The advent of interactive agents promises to bring transforming changes across many sectors. For instance, these agents could revolutionize the restaurant booking system within customer service. By naturally understanding and responding to spoken language, they could optimize the reservation process and elevate the user experience.

In healthcare, interactive agents could serve as the first point of contact for patients seeking new services. By understanding and addressing patients' needs compassionately and responsively, these agents have the potential to bridge the divide between technology and personalized care.

The realm of education could also see a significant impact, particularly in the field of speech therapy. As an interactive and adaptive tool, these agents could assist children with speech impairments, facilitating their efforts to speak more effectively.

In the retail industry, in-store and online shopping experiences could be greatly enhanced. The agents could respond to customer queries and provide personalized recommendations, taking the shopping experience to unprecedented levels of convenience and personalization.

Furthermore, these agents could serve as effective mitigators of mood swings and repetitive tasks in call centers. By efficiently addressing customer inquiries and complaints, they would not only increase productivity but also enhance the quality of service.

7.8 Final Words

As we conclude this thesis, it is fitting to reflect on the direction and implications of this research. The primary goal of this work has been to advance the field of speech animation, focusing on animating realistic 3D characters that reflect natural human speech. By doing so, we aim to enable our digital selves in virtual worlds to transcend their applications, achieving a novel form of expression and interaction.

Our principal findings remark that speech-to-animation models can leverage pre-trained audio representation models enriched with immense amounts of data and adapt them to the specific task of animating 3D characters. This adaptation was made possible through the capture and processing of relevant data, such as the face, tongue, and jaw motions from different modalities.

At the time of writing this thesis, Large Language Models (LLMs) are witnessing a significant renaissance, revolutionizing how we interact with systems through natural language. This success is largely attributable to the sophisticated design of models capable of capturing language nuances and the availability of vast data resources. Initially limited to text only, but projects like DeepMind's Gemini continue to explore and incorporate various modalities, fostering an any-to-any modality paradigm. Concurrently, the research community is actively investigating ways to run these models locally without needing clusters of servers to benefit from these models. There is little doubt that global researchers will achieve this feat in due course.

Our research has demonstrated that it is possible to streamline the speech-animation pipeline. We have transitioned from a traditional approach—relying on the computation of phonemic representations from speech signals, aligning visemes to these phonemic representations, and interpolating between visemes to determine animation parameters—to a more direct and efficient method. This new approach animates a 3D head, including intricate inner mouth elements such as the tongue and jaw, solely from the speech signal.

A foundational premise of driving a 3D model, rather than rendering a 2D image directly, was the aspiration to generate novel and realistic views across various scenarios. Estimating the sequence of rig parameters and rendering in an engine such as Unreal Engine is one more step in our speech-to-animation pipeline, which could be reduced. For instance, recent progress in Neural Radiance Fields (NeRF) has shown that these models can capture the personal appearance of a head and generate novel views through input parameters. There's little doubt that diverse modalities like text or audio could drive these models in the near future.

The success of universal models, such as Gemini, prompts the expectation that we may achieve a comprehensive model capable of rendering realistic talking faces from any modality. This could encompass interpreting a person's spoken input, generating synchronized speech and visual rendering, or even replicating our tone and mannerisms when answering to the input either through text, voice, or video from known individuals. The possibilities are endless, profound, and promising, extending our understanding of human communication and opening new horizons for virtual interactions either driven by people or simulating agents.

It is clear we are only at the outset of a substantial journey into the future. The initial results from exploring this field have been promising, and the potential for further advancements is considerable. This early stage of end-to-end speech-to-animation development has the potential for further research.

With each stride, we broaden our knowledge and challenge the existing boundaries. We can expect this field to evolve and mature in both the near and distant future as we aim to develop increasingly sophisticated and realistically looking interactive agents. This path forward invites us to continue probing, discovering, and refining the ideas proposed in this work and the research community.

While the journey ahead may be extensive, each step brings us closer to a future where technology augments and enhances human interaction, reshaping how we communicate, learn, work, and live. As we look forward, let us do so with cautious optimism, prepared for the challenges and opportunities that undoubtedly lie ahead.

BIBLIOGRAPHY

[1] YI Adbel-Aziz. Direct linear transformation from comparator coordinates into object space in close-range photogrammetry. In *ASP Symp. Proc. on Close-Range Photogrammetry, American Society of Photogrammetry, Falls Church, 1971*, pages 1–18, 1971.

[2] Triantafyllos Afouras, Joon Son Chung, and Andrew Zisserman. Lrs3-ted: a large-scale dataset for visual speech recognition. *arXiv preprint arXiv:1809.00496*, 2018.

[3] Abien Fred Agarap. Deep learning using rectified linear units (relu). *arXiv preprint arXiv:1803.08375*, 2018.

[4] Takuya Akiba, Shotaro Sano, Toshihiko Yanase, Takeru Ohta, and Masanori Koyama. Optuna: A next-generation hyperparameter optimization framework. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '19, page 2623–2631, New York, NY, USA, 2019. Association for Computing Machinery.

[5] Simon Alexanderson, Gustav Henter, Taras Kucherenko, and Jonas Beskow. Style controllable speech-driven gesture synthesis using normalising flows. *Computer Graphics Forum*, 39(2):487–496, 05 2020.

[6] Dario Amodei, Sundaram Ananthanarayanan, Rishita Anubhai, Jingliang Bai, Eric Battenberg, Carl Case, Jared Casper, Bryan Catanzaro, Qiang Cheng, Guoliang Chen, Jie Chen, Jingdong Chen, Zhijie Chen, Mike Chrzanowski, Adam Coates, Greg Diamos, Ke Ding, Niandong Du, Erich Elsen, Jesse Engel, Weiwei Fang, Linxi Fan, Christopher Fougner, Liang Gao, Caixia Gong, Awni Hannun, Tony Han, Lappi Vaino Johannes, Bing Jiang, Cai Ju, Billy Jun, Patrick LeGresley, Libby Lin, Junjie Liu, Yang Liu, Weigao Li, Xiangang Li, Dongpeng Ma, Sharan Narang, Andrew Ng, Sherjil Ozair, Yiping Peng, Ryan Prenger, Sheng Qian, Zongfeng Quan, Jonathan Raiman, Vinay Rao, Sanjeev Satheesh, David Seetapun, Shubho Sengupta, Kavya Srinet, Anuroop Sriram, Haiyuan Tang, Liliang Tang, Chong Wang, Jidong Wang, Kaifu Wang, Yi Wang, Zhijian Wang, Zhiqian Wang, Shuang Wu, Likai Wei, Bo Xiao, Wen Xie, Yan Xie, Dani Yogatama, Bin Yuan, Jun Zhan, and Zhenyao Zhu. Deep speech 2: End-to-end speech recognition in english and mandarin. In *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48*, ICML'16, page 173–182, New York, NY, USA, 2016. JMLR.org.

[7] R. Anderson, B. Stenger, V. Wan, and R. Cipolla. Expressive visual text-to-speech using active appearance models. In *2013 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3382–3389, Los Alamitos, CA, USA, jun 2013. IEEE Computer Society.

[8] Andreas Axelsson and Erik Björhäll. Real time speech driven face animation, 2003.

[9] Monica Villanueva Aylagas, Hector Anadon Leon, Mattias Teye, and Konrad Tollmar. Voice2face: Audio-driven facial and tongue rig animations with cvaes. In *Computer Graphics Forum*, volume 41, pages 255–265. Wiley Online Library, 2022.

[10] Alexei Baevski, Yuhao Zhou, Abdelrahman Mohamed, and Michael Auli. wav2vec 2.0: A framework for self-supervised learning of speech representations. *Advances in neural information processing systems*, 33:12449–12460, 2020.

[11] Murali Karthick Baskar, Shinji Watanabe, Ramon Astudillo, Takaaki Hori, Lukáš Burget, and Jan Černockỳ. Semi-supervised sequence-to-sequence asr using unpaired speech and text. *arXiv preprint arXiv:1905.01152*, 2019.

[12] Théo Biasutto-Lervat and Slim Ouni. Phoneme-to-articulatory mapping using bidirectional gated rnn. In *INTERSPEECH*, 2018.

[13] Bernd Bickel, Manuel Lang, Mario Botsch, Miguel A Otaduy, and Markus H Gross. Pose-space animation and transfer of facial details. In *Symposium on Computer Animation*, pages 57–66, 2008.

[14] Peter Birkholz. Modeling consonant-vowel coarticulation for articulatory speech synthesis. *PloS one*, 8(4):e60603, 2013.

[15] Peter Birkholz, Dietmar Jackèl, and Bernd J Kroger. Construction and control of a three-dimensional vocal tract model. In *2006 IEEE International Conference on Acoustics Speech and Signal Processing Proceedings*, volume 1, pages I–I, USA, 2006. IEEE, IEEE.

[16] Matthew Brand. Voice puppetry. In *Proceedings of the 26th annual conference on Computer graphics and interactive techniques*, pages 21–28, 1999.

[17] Shengze Cai, Zhicheng Wang, Sifan Wang, Paris Perdikaris, and George Em Karniadakis. Physics-informed neural networks for heat transfer problems. *Journal of Heat Transfer*, 143(6):060801, 2021.

[18] Claudia Canevari, Leonardo Badino, and L. Fadiga. A new italian dataset of parallel acoustic and articulatory data. In *INTERSPEECH*, 2015.

[19] Carstens Medizinelektronik GmbH. 3D electromagnetic articulograph. https://www.articulograph.de.

[20] Sherman Charles and Steven M Lulich. Articulatory-acoustic relations in the production of alveolar and palatal lateral sounds in brazilian portuguese. *The journal of the Acoustical Society of America*, 145(6):3269–3288, 2019.

[21] Guoguo Chen, Shuzhou Chai, Guanbo Wang, Jiayu Du, Wei-Qiang Zhang, Chao Weng, Dan Su, Daniel Povey, Jan Trmal, Junbo Zhang, et al. Gigaspeech: An evolving, multi-domain asr corpus with 10,000 hours of transcribed audio. *arXiv preprint arXiv:2106.06909*, 2021.

[22] Haoyu Chen, Hao Tang, Nicu Sebe, and Guoying Zhao. Aniformer: Data-driven 3d animation with transformer. *arXiv preprint arXiv:2110.10533*, 2021.

[23] Kang Chen, Zhipeng Tan, Jin Lei, Song-Hai Zhang, Yuan-Chen Guo, Weidong Zhang, and Shi-Min Hu. Choreomaster: choreography-oriented music-driven dance synthesis. *ACM Transactions on Graphics (TOG)*, 40(4):1–13,

2021.

[24] Sanyuan Chen, Chengyi Wang, Zhengyang Chen, Yu Wu, Shujie Liu, Zhuo Chen, Jinyu Li, Naoyuki Kanda, Takuya Yoshioka, Xiong Xiao, Jian Wu, Long Zhou, Shuo Ren, Yanmin Qian, Yao Qian, Micheal Zeng, and Furu Wei. Wavlm: Large-scale self-supervised pre-training for full stack speech processing. *IEEE Journal of Selected Topics in Signal Processing*, 16:1505–1518, 2021.

[25] Shicheng Chen, Yifeng Zheng, Chengrui Wu, Guorui Sheng, Pierre Roussel, and Bruce Denby. Direct, near real time animation of a 3D tongue model using non-invasive ultrasound images. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4994–4998, Calgary, AB, Canada, 2018. IEEE, IEEE.

[26] Weiyi Chen, Dani Byrd, Shrikanth Narayanan, and Krishna S Nayak. Intermittently tagged real-time MRI reveals internal tongue motion during speech production. *Magnetic resonance in medicine*, 82(2):600–613, 2019.

[27] William Chen, Brian Yan, Jiatong Shi, Yifan Peng, Soumi Maiti, and Shinji Watanabe. Improving massively multilingual asr with auxiliary ctc objectives. In *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5. IEEE, 2023.

[28] Zewen Chi, Shaohan Huang, Li Dong, Shuming Ma, Bo Zheng, Saksham Singhal, Payal Bajaj, Xia Song, Xian-Ling Mao, Heyan Huang, et al. Xlm-e: Cross-lingual language model pre-training via electra. *arXiv preprint arXiv:2106.16138*, 2021.

[29] Chung-Cheng Chiu, Tara N Sainath, Yonghui Wu, Rohit Prabhavalkar, Patrick Nguyen, Zhifeng Chen, Anjuli Kannan, Ron J Weiss, Kanishka Rao, Ekaterina Gonina, et al. State-of-the-art speech recognition with sequence-to-sequence models. In *2018 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pages 4774–4778. IEEE, 2018.

[30] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014.

[31] Aritra Chowdhury, Malik Magdon-Ismail, and Bülent Yener. Quantifying contribution and propagation of error from computational steps, algorithms and hyperparameter choices in image classification pipelines. *ArXiv*, abs/1903.00405, 2019.

[32] Joon Son Chung, Arsha Nagrani, and Andrew Zisserman. Voxceleb2: Deep speaker recognition. In *Interspeech*, 2018.

[33] Yu-An Chung, Yu Zhang, Wei Han, Chung-Cheng Chiu, James Qin, Ruoming Pang, and Yonghui Wu. w2v-bert: Combining contrastive learning and masked language modeling for self-supervised speech pre-training. *2021 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pages 244–250, 2021.

[34] Jeffrey F. Cohn, Zara Ambadar, and Paul Ekman. Observer-based measurement of facial expression with the facial action coding system. *The handbook of emotion elicitation and assessment*, 1(3):203–221, 2007.

[35] Ronald A. Cole, Tim Carmell, Pam Connors, Michael W. Macon, Johan Wouters, Jacques de Villiers, Alice Tarachow, Dominic W. Massaro, Michael M. Cohen, Jonas Beskow, Jie Yang, Uwe Meier, Alexander H. Waibel,

Patrick Stone, Alice Davis, Chris Soland, and George E. Fortier. Intelligent animated agents for interactive language training. *ACM Sigcaph Computers and The Physically Handicapped*, pages 5–10, 1998.

[36] Salvatore Cuomo, Vincenzo Schiano Di Cola, Fabio Giampaolo, Gianluigi Rozza, Maziar Raissi, and Francesco Piccialli. Scientific machine learning through physics–informed neural networks: Where we are and what's next. *Journal of Scientific Computing*, 92(3):88, 2022.

[37] Jianwu Dang and Kiyoshi Honda. Construction and control of a physiological articulatory model. *The Journal of the Acoustical Society of America*, 115(2):853–870, 2004.

[38] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[39] Zhigang Deng and Ulrich Neumann. efase: Expressive facial animation synthesis and editing with phoneme-isomap controls. In *Proceedings of the 2006 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '06, pages 251–260, Aire-la-Ville, Switzerland, Switzerland, 2006. Eurographics Association.

[40] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[41] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *ArXiv*, abs/2010.11929:1–12, 2021.

[42] 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.

[43] Bernhard Egger, William AP Smith, Ayush Tewari, Stefanie Wuhrer, Michael Zollhoefer, Thabo Beeler, Florian Bernard, Timo Bolkart, Adam Kortylewski, Sami Romdhani, et al. 3d morphable face models—past, present, and future. *ACM Transactions on Graphics (TOG)*, 39(5):1–38, 2020.

[44] Paul Ekman. Emotions revealed: Recognizing faces and feelings to improve communication and emotional life.(2003). *Emotions Revealed: Recognizing Faces and Feelings to Improve Communication and Emotional Life. Xvii*, 267, 2003.

[45] Paul Ekman, Wallace V. Friesen, and Joseph C. Hager. *Facial Action Coding System: The Manual*. Paul Ekman Group, San Francisco, CA, USA, 2002.

[46] Olov Engwall. Combining MRI, EMA and EPG measurements in a three-dimensional tongue model. *Speech Communication*, 41(2):303–329, 2003.

[47] Epic Games. MetaHuman Creator. https://www.unrealengine.com/en-US/metahuman-creator.

[48] Diandra Fabre, Thomas Hueber, and Pierre Badin. Automatic animation of an articulatory tongue model from ultrasound images using gaussian mixture regression. In *Fifteenth Annual Conference of the International Speech Communication Association*, pages 2293–2297, Singapore, 2014. ISCA.

[49] Yingruo Fan, Zhaojiang Lin, Jun Saito, Wenping Wang, and Taku Komura. Faceformer: Speech-driven 3d facial animation with transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern*

Recognition, pages 18770–18780, 2022.

[50] Mireille Fares, Catherine Pelachaud, and Nicolas Obin. I-brow: Hierarchical and multimodal transformer model for eyebrows animation synthesis. In *International Conference on Human-Computer Interaction*, pages 435–452. Springer, 2023.

[51] 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.

[52] John S Garofolo, Lori F Lamel, William M Fisher, Jonathan G Fiscus, and David S Pallett. Darpa timit acoustic-phonetic continous speech corpus cd-rom. nist speech disc 1-1.1. *STIN*, 93:27403, 1993.

[53] Bryan Gick, Megan Keough, Oksana Tkachman, and Yadong Liu. Lateral bias in lingual bracing during speech. *The Journal of the Acoustical Society of America*, 144(3):1903–1903, 2018.

[54] Google. Palm 2 technical report. Technical report, Google, 2023.

[55] Alex Graves, Santiago Fernández, Faustino Gomez, and Jürgen Schmidhuber. Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks. In *Proceedings of the 23rd international conference on Machine learning*, pages 369–376, 2006.

[56] Alex Graves and Alex Graves. Connectionist temporal classification. *Supervised sequence labelling with recurrent neural networks*, pages 61–93, 2012.

[57] Awni Hannun. Sequence modeling with ctc. https://distill.pub/2017/ctc/, 2017. Accessed: August 24, 2023.

[58] Richard Harshman, Peter Ladefoged, and Louis Goldstein. Factor analysis of tongue shapes. *The Journal of the Acoustical Society of America*, 62(3):693–707, 1977.

[59] Abdelwahab Heba, Thomas Pellegrini, Jean-Pierre Lorré, and Régine André-Obrecht. Char+cv-ctc: Combining graphemes and consonant/vowel units for ctc-based asr using multitask learning. In *Interspeech*, 2019.

[60] Geoffrey Hinton, Li Deng, Dong Yu, George E Dahl, Abdel-rahman Mohamed, Navdeep Jaitly, Andrew Senior, Vincent Vanhoucke, Patrick Nguyen, Tara N Sainath, et al. Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *IEEE Signal processing magazine*, 29(6):82–97, 2012.

[61] H. Hirai. A physiological model of speech organs incorporating tongue-larynx interaction. *J Acoust. Soc. Jpn.(J)*, 52:918, 1995.

[62] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[63] Pengyu Hong, Zhen Wen, and Thomas S Huang. Real-time speech-driven face animation with expressions using neural networks. *IEEE Transactions on neural networks*, 13(4):916–927, 2002.

[64] Phil Howson and Alexei Kochetov. An EMA examination of liquids in czech. In *The Scottish Consortium for ICPhS 2015 (Ed.), Proceedings of the 18th International Congress of Phonetic Sciences*, 2015.

[65] Wei-Ning Hsu, Benjamin Bolte, Yao-Hung Hubert Tsai, Kushal Lakhotia, Ruslan Salakhutdinov, and Abdelrah-

man Mohamed. Hubert: Self-supervised speech representation learning by masked prediction of hidden units. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 29:3451–3460, 2021.

[66] Huirong Huang, Zhiyong Wu, Shiyin Kang, Dongyang Dai, Jia Jia, Tianxiao Fu, Deyi Tuo, Guangzhi Lei, Peng Liu, Dan Su, et al. Speaker independent and multilingual/mixlingual speech-driven talking head generation using phonetic posteriorgrams. In *2021 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)*, pages 1433–1437. IEEE, 2021.

[67] H. Huang, Zhiyong Wu, Shiyin Kang, Dongyang Dai, Jia Jia, Tianxiao Fu, Deyi Tuo, Guangzhi Lei, Peng Liu, Dan Su, Dong Yu, and H. Meng. Speaker independent and multilingual/mixlingual speech-driven talking head generation using phonetic posteriorgrams. *ArXiv*, abs/2006.11610:5, 2020.

[68] Thomas Hueber, Guido Aversano, Gérard Chollet, Bruce Denby, Gérard Dreyfus, Yacine Oussar, Pierre Roussel, and Maureen Stone. Eigentongue feature extraction for an ultrasound-based silent speech interface. In *2007 IEEE International Conference on Acoustics, Speech and Signal Processing-ICASSP'07*, volume 1, pages I–1245. IEEE, 2007.

[69] Hirofumi Inaguma, Yashesh Gaur, Liang Lu, Jinyu Li, and Yifan Gong. Minimum latency training strategies for streaming sequence-to-sequence asr. In *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6064–6068, 2020.

[70] Khalil Iskarous. Patterns of tongue movement. *Journal of Phonetics*, 33(4):363–381, 2005.

[71] T ITU. Recommendation g. 114, one-way transmission time. *Series G: Transmission Systems and Media, Digital Systems and Networks, Telecommunication Standardization Sector of ITU*, 2000.

[72] Philip JB Jackson and Veena D Singampalli. Statistical identification of articulation constraints in the production of speech. *Speech Communication*, 51(8):695–710, 2009.

[73] Amir Jamaludin, Joon Son Chung, and Andrew Zisserman. You said that?: Synthesising talking faces from audio. *International Journal of Computer Vision*, 127(11-12):1767–1779, 2019.

[74] Ye Jia, Yu Zhang, Ron Weiss, Quan Wang, Jonathan Shen, Fei Ren, Patrick Nguyen, Ruoming Pang, Ignacio Lopez Moreno, Yonghui Wu, et al. Transfer learning from speaker verification to multispeaker text-to-speech synthesis. *Advances in neural information processing systems*, 31, 2018.

[75] Zeyu Jin, Adam Finkelstein, Gautham J Mysore, and Jingwan Lu. Fftnet: A real-time speaker-dependent neural vocoder. In *2018 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pages 2251–2255. IEEE, 2018.

[76] J. Kahn, M. Rivière, W. Zheng, E. Kharitonov, Q. Xu, P. E. Mazaré, J. Karadayi, V. Liptchinsky, R. Collobert, C. Fuegen, T. Likhomanenko, G. Synnaeve, A. Joulin, A. Mohamed, and E. Dupoux. Libri-light: A benchmark for asr with limited or no supervision. In *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7669–7673, 2020. https://github.com/facebookresearch/libri-light.

[77] Minguk Kang, Jun-Yan Zhu, Richard Zhang, Jaesik Park, Eli Shechtman, Sylvain Paris, and Taesung Park. Scaling up gans for text-to-image synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern*

Recognition, pages 10124–10134, 2023.

[78] 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 Transactions on Graphics (TOG)*, 36(4):1–12, 2017.

[79] William F Katz, Sonya Mehta, Matthew Wood, and Jun Wang. Using electromagnetic articulography with a tongue lateral sensor to discriminate manner of articulation. *The Journal of the Acoustical Society of America*, 141(1):EL57–EL63, 2017.

[80] Bo-Kyeong Kim, Jaemin Kang, Daeun Seo, Hancheol Park, Shinkook Choi, Hyungshin Kim, and Sungsu Lim. A unified compression framework for efficient speech-driven talking-face generation. *arXiv preprint arXiv:2304.00471*, 2023.

[81] Hyeongwoo Kim, Mohamed Elgharib, Hans-Peter Zollöfer, Michael Seidel, Thabo Beeler, Christian Richardt, and Christian Theobalt. Neural style-preserving visual dubbing. *ACM Transactions on Graphics (TOG)*, 38(6):178:1–13, 2019.

[82] Hyeongwoo Kim, Pablo Garrido, Ayush Tewari, Weipeng Xu, Justus Thies, Matthias Niessner, Patrick Pérez, Christian Richardt, Michael Zollhöfer, and Christian Theobalt. Deep video portraits. *ACM Transactions on Graphics (TOG)*, 37(4):1–14, 2018.

[83] Jangwon Kim, Asterios Toutios, Sungbok Lee, and Shrikanth S Narayanan. Vocal tract shaping of emotional speech. *Computer Speech & Language*, 64:101100, 2020.

[84] Suyoun Kim, Takaaki Hori, and Shinji Watanabe. Joint ctc-attention based end-to-end speech recognition using multi-task learning. In *2017 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pages 4835–4839. IEEE, 2017.

[85] Taehwan Kim, Yisong Yue, Sarah L. Taylor, and Iain Matthews. A decision tree framework for spatiotemporal sequence prediction. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 577–586, New York, NY, United States, 2015. Association for Computing Machinery.

[86] Scott A. King and Richard E. Parent. A 3D parametric tongue model for animated speech. *The Journal of Visualization and Computer Animation*, 12(3):107–115, 2001.

[87] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, abs/1412.6980:11, 2014.

[88] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. *Commun. ACM*, 60(6):84–90, may 2017.

[89] Sumedha Kshirsagar and Nadia Magnenat-Thalmann. Visyllable based speech animation. *Comput. Graph. Forum*, 22(3):632–640, 2003.

[90] Felix Kuhnke and Jörn Ostermann. Visual speech synthesis from 3D mesh sequences driven by combined speech features. In *2017 IEEE International Conference on Multimedia and Expo (ICME)*, pages 1075–1080, Hong Kong, China, 2017. IEEE.

[91] Matt Le, Apoorv Vyas, Bowen Shi, Brian Karrer, Leda Sari, Rashel Moritz, Mary Williamson, Vimal Manohar,

Yossi Adi, Jay Mahadeokar, and Wei-Ning Hsu. Voicebox: Text-guided multilingual universal speech generation at scale. *ArXiv*, abs/2306.15687, 2023.

[92] Colin Lea, Michael D Flynn, Rene Vidal, Austin Reiter, and Gregory D Hager. Temporal convolutional networks for action segmentation and detection. In *proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 156–165, 2017.

[93] Jean-Christophe Léger. Menger curvature and rectifiability. *Annals of mathematics*, 149(3):831–869, 1999.

[94] Lisha Li, Kevin Jamieson, Giulia DeSalvo, Afshin Rostamizadeh, and Ameet Talwalkar. Hyperband: A novel bandit-based approach to hyperparameter optimization. *The Journal of Machine Learning Research*, 18(1):6765–6816, 2017.

[95] Yuhong Li, Tianle Cai, Yi Zhang, Deming Chen, and Debadeepta Dey. What makes convolutional models great on long sequence modeling? *arXiv preprint arXiv:2210.09298*, 2022.

[96] Andy Liaw and Matthew Wiener. Classification and regression by randomforest. *R news*, 2(3):18–22, 2002.

[97] Liyuan Liu, Haoming Jiang, Pengcheng He, Weizhu Chen, Xiaodong Liu, Jianfeng Gao, and Jiawei Han. On the variance of the adaptive learning rate and beyond. *ArXiv*, abs/1908.03265, 2019.

[98] Changwei Luo, Jun Yu, Xian Li, and Leilei Zhang. Hmm based speech-driven 3D tongue animation. In *2017 IEEE International Conference On Image Processing (ICIP)*, pages 4377–4381, Beijing, China, 2017. IEEE, IEEE.

[99] Changwei Luo, Jun Yu, and Zengfu Wang. Synthesizing real-time speech-driven facial animation. In *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4568–4572. IEEE, 2014.

[100] Ran Luo, Qiang Fang, Jianguo Wei, Wenhuan Lu, Weiwei Xu, and Yin Yang. Acoustic vr in the mouth: A real-time speech-driven visual tongue system. In *2017 IEEE Virtual Reality (VR)*, pages 112–121. IEEE, 2017.

[101] Mario Malcangi and Raffaele de Tintis. Audio based real-time speech animation of embodied conversational agents. In *International Gesture Workshop*, pages 350–360. Springer, 2003.

[102] Dominic W. Massaro, Jonas Beskow, Michael M. Cohen, Christopher L. Fry, and Tony Rodgriguez. Picture my voice: Audio to visual speech synthesis using artificial neural networks. In *Auditory-Visual Speech Processing*, pages 133–138, Santa Cruz, California, USA, August 1999. ISCA.

[103] Dominic W. Massaro, Ying Liu, Trevor H. Chen, and Charles Perfetti. A multilingual embodied conversational agent for tutoring speech and language learning. In *INTERSPEECH 2006 - ICSLP, Ninth International Conference on Spoken Language Processing*, pages 825–828, Pittsburgh, PA, USA, 2006. ISCA.

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

[105] Salvador Medina, Sarah Taylor, Mark Tiede, Alexander Hauptmann, and Iain Matthews. Importance of parasagittal sensor information in tongue motion capture through a diphonic analysis. *Interspeech 2021*, pages 3340–3344, 2021.

[106] Salvador Medina, Denis Tome, Carsten Stoll, Mark Tiede, Kevin Munhall, Alexander G Hauptmann, and Iain

Matthews. Speech driven tongue animation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 20406–20416, 2022.

[107] Jeff Moore, Jason Shaw, Shigeto Kawahara, and Takayuki Arai. Articulation strategies for English liquids used by Japanese speakers. *Acoustical Science and Technology*, 39(2):75–83, 2018.

[108] Cubic Motion. Animation services, 2023. Accessed: August 18, 2023.

[109] Graham Neubig. Neural machine translation and sequence-to-sequence models: A tutorial. *arXiv preprint arXiv:1703.01619*, 2017.

[110] Adam Polyak Noam Mor, Lior Wold and Yaniv Taigman. A universal music translation network. In *International Conference on Learning Representations (ICLR)*, page 10, La Jolla, CA, USA, 2019. ICLR.

[111] ONNX. Open Neural Network Exchange, 2023. Accessed: 2023-07-16.

[112] OpenAI. Gpt-4 technical report. Technical report, OpenAI, 2023.

[113] Vassil Panayotov, Guoguo Chen, Daniel Povey, and Sanjeev Khudanpur. Librispeech: an asr corpus based on public domain audio books. In *2015 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pages 5206–5210, South Brisbane, QLD, Australia, 2015. IEEE, IEEE.

[114] Frederic I. Parke and Keith Waters. *Computer Facial Animation*. A K Peters/CRC Press, New York, USA, 1996.

[115] Frederic I Parke and Keith Waters. *Computer facial animation*. CRC press, 2008.

[116] Shahla Parveen and Phil Green. Multitask learning in connectionist robust asr using recurrent neural networks. In *Eighth European Conference on Speech Communication and Technology*, 2003.

[117] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, volume 32, pages 8024–8035. Curran Associates, Inc., Red Hook, NY, USA, 2019.

[118] Bo Peng, Eric Alcaide, Quentin Anthony, Alon Albalak, Samuel Arcadinho, Huanqi Cao, Xin Cheng, Michael Chung, Matteo Grella, Kranthi Kiran GV, et al. Rwkv: Reinventing rnns for the transformer era. *arXiv preprint arXiv:2305.13048*, 2023.

[119] Ziqiao Peng, Haoyu Wu, Zhenbo Song, Hao Xu, Xiangyu Zhu, Hongyan Liu, Jun He, and Zhaoxin Fan. Emotalk: Speech-driven emotional disentanglement for 3d face animation. *arXiv preprint arXiv:2303.11089*, 2023.

[120] Joseph S Perkell. *A physiologically-oriented model of tongue activity in speech production.* PhD thesis, Massachusetts Institute of Technology, 1974.

[121] Mathis Petrovich, Michael J Black, and Gül Varol. Action-conditioned 3d human motion synthesis with transformer vae. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10985–10995, 2021.

[122] Hai Xuan Pham, Yuting Wang, and Vladimir Pavlovic. End-to-end learning for 3d facial animation from speech.

In *Proceedings of the 20th ACM International Conference on Multimodal Interaction*, pages 361–365, 2018.

[123] Ulf Posselt et al. Range of movement of the mandiblew. *The Journal of the American Dental Association*, 56(1):10–13, 1958.

[124] 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.

[125] Prolific. Prolific: World's largest community of research participants, 2023. Accessed: 2023-07-21.

[126] Alec Radford, Jong Wook Kim, Tao Xu, Greg Brockman, Christine McLeavey, and Ilya Sutskever. Robust speech recognition via large-scale weak supervision. *ArXiv*, abs/2212.04356, 2022.

[127] Maziar Raissi, Paris Perdikaris, and George Em Karniadakis. Machine learning of linear differential equations using gaussian processes. *Journal of Computational Physics*, 348:683–693, 2017.

[128] Teja Rebernik, Jidde Jacobi, Roel Jonkers, Aude Noiray, and Martijn Wieling. A review of data collection practices using electromagnetic articulography. *Laboratory Phonology: Journal of the Association for Laboratory Phonology*, 12(1), 2021.

[129] Alexander Richard, Colin Lea, Shugao Ma, Juergen Gall, Fernando de la Torre, and Yaser Sheikh. Audio- and gaze-driven facial animation of codec avatars. In *2021 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 41–50, Waikoloa, HI, USA, 2021. IEEE.

[130] Korin Richmond, Phil Hoole, and Simon King. Announcing the electromagnetic articulography (day 1) subset of the mngu0 articulatory corpus. In *INTERSPEECH*, 2011.

[131] K. Richmond, P. Hoole, and S. King. Announcing the electromagnetic articulography (day 1) subset of the mngu0 articulatory corpus. In *Interspeech 2011: 12th Annual Conference of the International Speech Communication Association*, pages 1505–1508, 2011.

[132] EH Rothauser. Ieee recommended practice for speech quality measurements. *IEEE Trans. on Audio and Electroacoustics*, 17:225–246, 1969.

[133] Paul K Rubenstein, Chulayuth Asawaroengchai, Duc Dung Nguyen, Ankur Bapna, Zalán Borsos, Félix de Chaumont Quitry, Peter Chen, Dalia El Badawy, Wei Han, Eugene Kharitonov, et al. Audiopalm: A large language model that can speak and listen. *arXiv preprint arXiv:2306.12925*, 2023.

[134] Teven Le Scao, Angela Fan, Christopher Akiki, Ellie Pavlick, Suzana Ilić, Daniel Hesslow, Roman Castagné, Alexandra Sasha Luccioni, François Yvon, Matthias Gallé, et al. Bloom: A 176b-parameter open-access multilingual language model. *arXiv preprint arXiv:2211.05100*, 2022.

[135] Dietmar Schabus, Michael Pucher, and Phil Hoole. The MMASCS multi-modal annotated synchronous corpus of audio, video, facial motion and tongue motion data of normal, fast and slow speech. In *LREC*, pages 3411–3416, 2014.

[136] Steffen Schneider, Alexei Baevski, Ronan Collobert, and Michael Auli. wav2vec: Unsupervised pre-training for speech recognition. In *INTERSPEECH*, pages 1–9, Graz, Austria, 2019. ISCA.

[137] Robert Shapiro. Direct linear transformation method for three-dimensional cinematography. *Research Quarterly. American Alliance for Health, Physical Education and Recreation*, 49(2):197–205, 1978.

[138] Bowen Shi, Abdelrahman Mohamed, and Wei-Ning Hsu. Learning lip-based audio-visual speaker embeddings with av-hubert. *arXiv preprint arXiv:2205.07180*, 2022.

[139] Tian Shi, Yaser Keneshloo, Naren Ramakrishnan, and Chandan K Reddy. Neural abstractive text summarization with sequence-to-sequence models. *ACM Transactions on Data Science*, 2(1):1–37, 2021.

[140] Eftychios Sifakis, Andrew Selle, Avram Robinson-Mosher, and Ronald Fedkiw. Simulating speech with a physics-based facial muscle model. In *Proceedings of the 2006 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '06, pages 261–270, Aire-la-Ville, Switzerland, Switzerland, 2006. Eurographics Association.

[141] Ganesh Sivaraman, Carol Y Espy-Wilson, and Martijn Wieling. Analysis of acoustic-to-articulatory speech inversion across different accents and languages. In *INTERSPEECH*, pages 974–978, 2017.

[142] Jonathan D Smith, Kamyar Azizzadenesheli, and Zachary E Ross. Eikonet: Solving the eikonal equation with deep neural networks. *IEEE Transactions on Geoscience and Remote Sensing*, 59(12):10685–10696, 2020.

[143] Yang Song, Jingwen Zhu, Dawei Li, Andy Wang, and Hairong Qi. Talking face generation by conditional recurrent adversarial network. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19*, pages 919–925, Macao, China, 7 2019. International Joint Conferences on Artificial Intelligence Organization.

[144] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958, 2014.

[145] Ingmar Steiner, Sébastien Le Maguer, and Alexander Hewer. Synthesis of tongue motion and acoustics from text using a multimodal articulatory database. *IEEE ACM Trans. Audio Speech Lang. Process.*, 25(12):2351–2361, 2017.

[146] Ingmar Steiner and Slim Ouni. Artimate: an articulatory animation framework for audiovisual speech synthesis. *ArXiv*, abs/1203.3574:1–4, 2012.

[147] Ingmar Steiner and Slim Ouni. Progress in animation of an ema-controlled tongue model for acoustic-visual speech synthesis. *ArXiv*, abs/1201.4080:1–8, 2012.

[148] Maureen Stone and Andrew Lundberg. Three-dimensional tongue surface shapes of english consonants and vowels. *The Journal of the Acoustical Society of America*, 99(6):3728–3737, 1996.

[149] Supasorn Suwajanakorn, Steven M. Seitz, and Ira Kemelmacher-Shlizerman. Synthesizing obama: learning lip sync from audio. *ACM Transactions on Graphics (TOG)*, 36(4):1–13, 2017.

[150] Sarah L. Taylor, Taehwan Kim, Yisong Yue, Moshe Mahler, James Krahe, Anastasio Garcia Rodriguez, Jessica Hodgins, and Iain Matthews. A deep learning approach for generalized speech animation. *ACM Transactions on Graphics (TOG)*, 36(4):1–11, 2017.

[151] Sarah L. Taylor, Moshe Mahler, Barry-John Theobald, and Iain Matthews. Dynamic units of visual speech. In

Proceedings of the 11th ACM SIGGRAPH/Eurographics conference on Computer Animation, pages 275–284, Postfach 2926,Goslar,Germany, 2012. Eurographics Association.

[152] Sarah L. Taylor, Moshe Mahler, Barry-John Theobald, and Iain Matthews. Dynamic units of visual speech. In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '12, pages 275–284, Postfach 2926, Goslar, Germany, 2012. Eurographics Association.

[153] Guanzhong Tian, Yi Yuan, and Yong Liu. Audio2face: Generating speech/face animation from single audio with attention-based bidirectional lstm networks. In *2019 IEEE International Conference on Multimedia Expo Workshops (ICMEW)*, pages 366–371, Shanghai, China, 2019. IEEE.

[154] Ashish Vaswani, Noam M. Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *ArXiv*, abs/1706.03762:1–12, 2017.

[155] Konstantinos Vougioukas, S. Petridis, and M. Pantic. End-to-end speech-driven facial animation with temporal gans. In *BMVC*, pages 1–12, Newcastle, UK, 2018. Springer.

[156] Changhan Wang, Morgane Riviere, Ann Lee, Anne Wu, Chaitanya Talnikar, Daniel Haziza, Mary Williamson, Juan Pino, and Emmanuel Dupoux. VoxPopuli: A large-scale multilingual speech corpus for representation learning, semi-supervised learning and interpretation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 993–1003, Online, Aug. 2021. Association for Computational Linguistics.

[157] Danny Websdale, Sarah L. Taylor, and Ben P. Milner. The effect of real-time constraints on automatic speech animation. In *Interspeech*, 2018.

[158] Xin Wen, Miao Wang, Christian Richardt, Ze-Yin Chen, and Shi-Min Hu. Photorealistic audio-driven video portraits. *IEEE Transactions on Visualization and Computer Graphics*, 26(12):3457–3466, 2020.

[159] Reiner Wilhelms-Tricarico. Physiological modeling of speech production: Methods for modeling soft-tissue articulators. *The Journal of the Acoustical Society of America*, 97(5):3085–3098, 1995.

[160] Jonghye Woo, Fangxu Xing, Maureen Stone, Jordan Green, Timothy G Reese, Thomas J Brady, Van J Wedeen, Jerry L Prince, and Georges El Fakhri. Speech map: A statistical multimodal atlas of 4D tongue motion during speech from tagged and cine MR images. *Computer Methods in Biomechanics and Biomedical Engineering: Imaging & Visualization*, 7(4):361–373, 2019.

[161] Alan Wrench. The mocha-timit articulatory database, 1999.

[162] Yuxin Wu and Kaiming He. Group normalization. In *Proceedings of the European conference on computer vision (ECCV)*, pages 3–19, 2018.

[163] 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, et al. Superb: Speech processing universal performance benchmark. *arXiv preprint arXiv:2105.01051*, 2021.

[164] Jia Ying, Jason A. Shaw, Christopher Carignan, Michael Proctor, Donald Derrick, and Catherine T. Best. Evidence for active control of tongue lateralization in australian english /l/. *Journal of Phonetics*, 86:101039, 2021.

[165] Youngwoo Yoon, Pieter Wolfert, Taras Kucherenko, Carla Viegas, Teodor Nikolov, Mihail Tsakov, and Gustav Eje Henter. The genea challenge 2022: A large evaluation of data-driven co-speech gesture generation. In *Proceedings of the 2022 International Conference on Multimodal Interaction*, ICMI '22, page 736–747, New York, NY, USA, 2022. Association for Computing Machinery.

[166] Jun Yu, Chen Jiang, and Zengfu Wang. A fast and precise speech-triggered tongue animation system by combining parameterized model and anatomical model. In *2016 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*, pages 1624–1627, Shenzhen, China, 2016. IEEE, IEEE.

[167] Lingyun Yu, Jun Yu, and Qiang Ling. Bltrcnn-based 3-d articulatory movement prediction: Learning articulatory synchronicity from both text and audio inputs. *IEEE Transactions on Multimedia*, 21:1621–1632, 2019.

[168] Ce Zheng, Sijie Zhu, Mat'ias Mendieta, Taojiannan Yang, Chen Chen, and Zhengming Ding. 3D human pose estimation with spatial and temporal transformers. *ArXiv*, abs/2103.10455:1–10, 2021.

[169] 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, Honolulu, Hawaii, USA, 2019. AAI.

[170] Yang Zhou, Xintong Han, Eli Shechtman, Jose Echevarria, Evangelos Kalogerakis, and Dingzeyu Li. Makelttalk: speaker-aware talking-head animation. *ACM Transactions on Graphics (TOG)*, 39(6):1–15, 2020.

[171] 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):1–10, 2018.

[172] Pengcheng Zhu, Lei Xie, and Yunlin Chen. Articulatory movement prediction using deep bidirectional long short-term memory based recurrent neural networks and word/phone embeddings. In *INTERSPEECH*, 2015.

[173] Michael Zollhöfer, Justus Thies, Pablo Garrido, Derek Bradley, Thabo Beeler, Patrick Pérez, Marc Stamminger, Matthias Nießner, and Christian Theobalt. State of the art on monocular 3d face reconstruction, tracking, and applications. In *Computer graphics forum*, volume 37, pages 523–550. Wiley Online Library, 2018.

[174] Gaspard Zoss, Derek Bradley, Pascal Bérard, and Thabo Beeler. An empirical rig for jaw animation. *ACM Transactions on Graphics (TOG)*, 37(4):1–12, 2018.

APPENDIX A

RESULTS STATISTICAL ANALYSIS

A.1 Model Configuration Comparison

In our study, we assessed models trained using embeddings from the last audio encoder layer (termed *Enc*), an aggregation of all audio features (*All-Enc*), and a multi-task learning model with a CTC auxiliary task focusing on the allophones of the input audio (*All-Enc CTC*). Our analysis spanned various audio features, including Wav2Vec, WavLM, and Whisper.

Our statistical analysis, presented in Table A.1 for the L1 metric, Table A.2 for the L2 metric, and in Table A.3 for the TMVE metric, show statistical significance for most of our results, with a confidence threshold of $p < 0.03$. However, one exception to note is the comparison between the *Enc* and *All-Enc* models when processing audio using Wav2Vec.

From this perspective, our claim regarding the performance of the WavLM *All-Enc CTC* remains valid and robust.

Table A.1: Comparison of p-values for the L1 error on the rig parameter space across three audio features: Wav2Vec, WavLM, and Whisper. The p-values represent the statistical significance when comparing between the *Enc*, *All-Enc*, and *All-Enc CTC* models. Values $p < 0.05$ are considered statistically significant. All comparisons in the table are statistically significant.

Comparison	**Wav2Vec**	**WavLM**	**Whisper**
Enc vs. All-Enc	2.133×10^{-7}	1.833×10^{-5}	1.705×10^{-58}
Enc vs. All-Enc CTC	4.257×10^{-8}	6.31×10^{-25}	4.345×10^{-71}
All-Enc vs. All-Enc CTC	7.305×10^{-46}	6.824×10^{-11}	8.859×10^{-9}

Table A.2: Comparison of p-values for the L2 error on the rig parameter space across three audio features: Wav2Vec, WavLM, and Whisper. The p-values represent the statistical significance when comparing between the *Enc*, *All-Enc*, and *All-Enc CTC* models. Values $p < 0.05$ are considered statistically significant. Comparisons in red are not statistically significant.

Comparison	**Wav2Vec**	**WavLM**	**Whisper**
Enc vs. All-Enc	*0.319*	6.189×10^{-5}	1.888×10^{-46}
Enc vs. All-Enc CTC	1.037×10^{-8}	4.375×10^{-13}	3.596×10^{-51}
All-Enc vs. All-Enc CTC	6.493×10^{-16}	0.029	0.024

Table A.3: Comparison of p-values for the lower face temporal mean vertex error (TMVE) across three audio features: Wav2Vec, WavLM, and Whisper. The p-values represent the statistical significance when comparing between the *Enc*, *All-Enc*, and *All-Enc CTC* models. Values $p < 0.05$ are considered statistically significant. All comparisons in the table are statistically significant.

Comparison	**Wav2Vec**	**WavLM**	**Whisper**
Enc vs. All-Enc	$2.747 \times 10^{-}40$	$1.692 \times 10^{-}68$	$4.032 \times 10^{-}33$
Enc vs. All-Enc CTC	$3.612 \times 10^{-}7$	$1.782 \times 10^{-}78$	$1.420 \times 10^{-}50$
All-Enc vs. All-Enc CTC	$1.107 \times 10^{-}52$	$5.275 \times 10^{-}5$	$1.953 \times 10^{-}13$

A.2 Audio Feature Comparison

We delved deeper into understanding the impact of the different audio features, specifically Wav2Vec, Whisper, and WavLM, across various model configurations (*Enc*, *All-Enc*, and *All-Enc CTC*). Our analysis revealed inconclusive results when contrasting the *All-Enc* and *All-Enc CTC* configurations trained using the WavLM and Whisper features, especially from the L1 and L2 error perspectives as shown in Table A.4 and Table A.5, respectively. This is attributed to p-values exceeding the 0.05 threshold shown in red in the tables.

User studies mirror these findings, showing comparable performance of these configurations on in-domain data. Both configurations rendered low errors during animation reconstruction. However, both the TMVE metric and user feedback indicate a slight edge for the WavLM *All-Enc CTC* model over models trained with Whisper audio features.

Table A.4: Comparison of p-values for the L1 error over rig parameters, evaluating the impact on the performance of different audio features (Wav2Vec, WavLM, and Whisper) across model configurations (*Enc*, *All-Enc*, and *All-Enc CTC*). Values $p < 0.05$ are considered statistically significant. Comparisons in red are not statistically significant.

Comparison	Enc	All-Enc	All-Enc CTC
Wav2Vec vs. WavLM	5.467×10^{-7}	1.256×10^{-37}	2.157×10^{-26}
Wav2Vec vs. Whisper	4.264×10^{-14}	1.541×10^{-47}	$7.221 \times 1-^{-30}$
WavLM vs. Whisper	5.227×10^{-36}	*0.444*	*0.163*

Table A.5: Comparison of p-values for the L2 error over rig parameters, evaluating the impact on the performance of different audio features (Wav2Vec, WavLM, and Whisper) across model configurations (*Enc*, *All-Enc*, and *All-Enc CTC*). Values $p < 0.05$ are considered statistically significant. Comparisons in red are not statistically significant.

Comparison	Enc	All-Enc	All-Enc CTC
Wav2Vec vs. WavLM	1.843×10^{-7}	2.712×10^{-18}	2.712×10^{-15}
Wav2Vec vs. Whisper	3.956×10^{-8}	2.400×10^{-23}	7.366×10^{-15}
WavLM vs. Whisper	7.363×10^{-26}	*0.591*	*0.163*

Table A.6: Comparison of p-values for the lower face temporal mean vertex error (TMVE), evaluating the impact on the performance of different audio features (Wav2Vec, WavLM, and Whisper) across model configurations (*Enc*, *All-Enc*, and *All-Enc CTC*). Values $p < 0.05$ are considered statistically significant. All comparisons in the table are statistically significant.

Comparison	Enc	All-Enc	All-Enc CTC
Wav2Vec vs. WavLM	1.368×10^{-11}	2.115×10^{-74}	5.247×10^{-23}
Wav2Vec vs. Whisper	2.698×10^{-16}	2.373×10^{-5}	8.277×10^{-18}
WavLM vs. Whisper	0.016	5.357×10^{-12}	0.013