VISUAL SPEECH SYNTHESIS BY LEARNING JOINT PROBABILISTIC MODELS OF AUDIO AND VIDEO

A thesis submitted to the University of Manchester
for the degree of Doctor of Philosophy
in the Faculty of Engineering and Physical Sciences

2012

By
Salil Prashant Deena
School of Computer Science

Contents

Abstract **13**

Declaration **14**

Copyright **15**

Acknowledgements **18**

Abbreviations **20**

Notations **22**

1 Introduction **25**

1.1 Overview 25

1.1.1 Models of the Face 26

1.1.2 Models of Audio-visual Mapping 27

1.2 Realistic Speech-driven Facial Animation 27

1.2.1 Challenges 28

1.2.2 Research Problems 28

1.3 Research Aims 29

1.4 Thesis Contributions 30

1.5 Thesis Structure 31

1.6 Publications 32

2 Background **33**

2.1 Human Speech 33

2.1.1 Human Speech Production 34

2.1.2 Human Speech Perception 35

2.1.3 Speech Primitives 36

2.1.4 Audio-visual Mapping 38

2.1.5 Coarticulation 39

2.2 Animating Faces . . . 43
2.2.1 Manual and Heuristic Facial Animation . . . 43
2.2.2 Data-driven Facial Animation . . . 44
2.2.3 Other Forms of Facial Animation . . . 47
2.2.4 Modelling Coarticulation . . . 48
2.3 The Uncanny Valley . . . 48
2.4 Applications of Visual Speech Synthesis . . . 50
2.4.1 Cinema . . . 50
2.4.2 Gaming . . . 51
2.4.3 Human-Computer Interaction . . . 51
2.4.4 Speech Therapy . . . 52
2.4.5 Medicine . . . 53
2.4.6 Internet and Communication . . . 53
2.5 Chapter Summary . . . 54

3 Data Acquisition and Processing 55
3.1 Modelling Faces . . . 56
3.1.1 Flexible Models in Computer Vision . . . 56
3.1.2 Principal Component Analysis . . . 57
3.1.3 Active Appearance Model . . . 59
3.1.4 Multidimensional Morphable Model . . . 62
3.1.5 3D Morphable Model . . . 63
3.1.6 Discussion . . . 63
3.2 Speech Processing . . . 64
3.2.1 The Fourier transform . . . 66
3.2.2 Linear Predictive Coding . . . 66
3.2.3 Line Spectral Frequencies . . . 67
3.2.4 Mel-frequency Cepstral Coefficients . . . 68
3.2.5 RASTA-PLP . . . 69
3.2.6 Discussion . . . 70
3.3 Data Corpora . . . 70
3.4 Visual Processing on Data Corpora . . . 71
3.4.1 Visual Normalisation . . . 74
3.5 Audio Processing on Data Corpora . . . 76
3.5.1 Synchronisation of Audio and Visual Parameters . . . 77
3.5.2 Comparing Speech Parameterisation Methods . . . 77
3.5.3 Discussion . . . 81
3.6 Chapter Summary . . . 83

4 State-Space Model for Audio-visual Mapping **84**
4.1 Probabilistic Graphical Models 84
4.2 Latent Variable Models 86
4.2.1 Linear Subspace Models 86
4.2.2 Gaussian Mixture Model 87
4.2.3 Hidden Markov Model 89
4.2.4 Linear Dynamical System 92
4.2.5 Gaussian Process Dynamical Model 94
4.3 Shared Latent Variable Models 102
4.3.1 Canonical Correlation Analysis 102
4.3.2 Coupled Hidden Markov Model 105
4.3.3 Shared Linear Dynamical System 106
4.3.4 Shared Gaussian Process Dynamical Model 107
4.4 Synthesis using the SGPDM 111
4.4.1 Point Optimisation 113
4.4.2 Sequence Optimisation 113
4.4.3 Initialisation of Latent Points 113
4.5 Audio-visual Mapping using SGPDM 115
4.5.1 Model Selection 116
4.5.2 Choice of Speech Parameterisation 124
4.5.3 Choice of Audio-visual Synchronisation Method 125
4.5.4 Limitations of SGPDM 125
4.5.5 Discussion 127
4.6 Chapter Summary 127

5 Switching State-Space Model for Audio-visual Mapping **129**
5.1 Switching State-space Models 129
5.1.1 Switching Linear Dynamical System 130
5.1.2 Deterministic Process Dynamical System 132
5.1.3 Switching Shared Gaussian Process Dynamical Model 134
5.2 Modelling Phonetic Context 137
5.2.1 Variable Length Markov Model 138
5.2.2 Language Modelling using VLMM 142
5.2.3 VLMMs of Behaviour 143
5.2.4 Phonetic Context Modelling using VLMM 145
5.3 Learning SSGPDMs of Audio and Video 149
5.3.1 SSGPDM with Phonetic Contexts as Switching States 150
5.3.2 SSGPDM with Phonemes as Switching States 151
5.4 Synthesis using the Switching SGPDM 155

5.4.1 Sequential Optimisation . . . 156
5.4.2 Sequential Prediction . . . 156
5.4.3 Smoothness Constraint . . . 156
5.4.4 Leading and Trailing Pauses . . . 158
5.4.5 Modelling Coarticulation . . . 158
5.5 Discussion . . . 159
5.6 Chapter Summary . . . 160

6 Evaluation **161**
6.1 Evaluation Methods for Visual Speech Synthesis . . . 161
6.2 Objective Evaluation . . . 162
6.2.1 Error Measures . . . 163
6.2.2 Correlation Measures . . . 164
6.2.3 Visual Feature Trajectories Comparison . . . 164
6.2.4 Automated Lip-reading . . . 165
6.2.5 Experiments . . . 166
6.2.6 Discussion . . . 170
6.3 Subjective Evaluation . . . 174
6.3.1 Scoring Tests . . . 174
6.3.2 Realism Tests . . . 175
6.3.3 Human Lip-reading Tests . . . 176
6.3.4 Statistical Hypothesis Testing . . . 177
6.3.5 Eye Blinks . . . 179
6.3.6 Experiments . . . 180
6.3.7 Discussion . . . 191
6.4 Chapter Summary . . . 191

7 Conclusion and Future Directions **192**
7.1 Thesis Summary . . . 193
7.2 Limitations of the Proposed Method . . . 195
7.3 Future Directions . . . 197

Bibliography **200**

A Fast ICA Algorithm **228**
A.1 Negentropy . . . 229
A.2 Pre-processing . . . 230
A.2.1 Centering . . . 230
A.2.2 Whitening . . . 230
A.3 FastICA for one unit . . . 230

A.4 FastICA for several units 231

B AAM Modes of Variation **232**

C ICA Modes of Variation **239**

D Animation Frames **242**

Word Count: 55318

List of Tables

2.1 Phoneme to viseme mapping for both BEEP and CMU phonemes. . . . 38

6.1 SGPDM quantitative evaluation results for LIPS and DEMNOW datasets. 169
6.2 SSGPDM quantitative evaluation results for LIPS and DEMNOW datasets. 169
6.3 Summary of probabilistic models for visual speech synthesis. 173
6.4 Distribution of the participants of the subjective tests in terms of age and sex. 183
6.5 Distribution of the participants of the subjective tests in terms of hours of computer usage and native vs. non-native speaker. 183
6.6 MOS scores for perceptual test. 186
6.7 Comparison of intelligibility scores between native and non-native speakers. 190

List of Figures

1.1 Overview of facial modelling approaches used in visual speech synthesis. 26
1.2 Overview of data-driven techniques for audio-visual mapping [77]. 27
1.3 Overview of the proposed method. 32

2.1 The speech production system [291]. 35
2.2 Representation of anticipatory and carryover coarticulation as overlapping gestures [114]. 40
2.3 The Uncanny Valley [215]. 50
2.4 Expressive facial animation [269]. 52
2.5 Expressive facial animation [43]. 52
2.6 The TNT facial animation system [191]. 54

3.1 (a) Markup points of facial landmarks. (b) Delaunay triangulation of landmarks. 60
3.2 Face reconstructed from the Basel Face Model [235]. 64
3.3 Triangular filter-banks based on the Mel scale. 68
3.4 Critical-band filter-banks based on the Bark scale. 70
3.5 (a) A frame from the LIPS corpus. (b) AAM markup points. (c) Delaunay triangulation. (d) AAM reconstruction. 72
3.6 (a) A frame from the DEMNOW corpus. (b) AAM markup points. (c) Delaunay triangulation. (d) AAM reconstruction. 73
3.7 Mean AAM trajectories for a given LIPS sequence before and after normalisation. The baseline before and after normalisation is also shown. . 74
3.8 Mean AAM trajectories for a given DEMNOW sequence before and after normalisation. 76
3.9 Mean LPC trajectories for a given LIPS sequence. 78
3.10 Mean LPC trajectories for a given DEMNOW sequence. 78
3.11 Mean LSF trajectories for a given LIPS sequence. 79
3.12 Mean LSF trajectories for a given DEMNOW sequence. 80
3.13 Mean MFCC trajectories for a given LIPS sequence. 80
3.14 Mean MFCC trajectories for a given DEMNOW sequence. 81

3.15 Mean RASTA-PLP trajectories for a given LIPS sequence. 82
3.16 Mean RASTA-PLP trajectories for a given DEMNOW sequence. 82

4.1 Notation used for graphical model: circles - continuous variables, squares - discrete variables, shaded - observed variables, unshaded - hidden variables. 85
4.2 Graphical model for probabilistic principal component analysis and Gaussian process latent variable model. 87
4.3 Graphical model for Gaussian mixture model. 87
4.4 Graphical model for a hidden Markov model. 89
4.5 Graphical model for linear dynamical system and Gaussian process dynamical model. 93
4.6 Latent spaces on MFCC or AAM data: Left - MFCC with phoneme labels, Right - AAM with viseme labels: (a) MFCC GPLVM, (b) AAM GPLVM, (c) MFCC GPLVM with MLP back-constraints, (d) AAM GPLVM with MLP back-constraints, (e) MFCC GPDM, (f) AAM GPDM, (g) MFCC GPDM with MLP back-constraints, (h) AAM GPDM with MLP back-constraints. 103
4.7 Graphical model for probabilistic canonical correlation analysis and shared Gaussian process latent variable model. 104
4.8 Graphical model for coupled hidden Markov model. 105
4.9 Graphical model for shared linear dynamical system and shared Gaussian process dynamical model. 107
4.10 Shared latent spaces on AAM and MFCC data: Left - Phoneme Labels, Right - Viseme Labels: (a) and (b) SGPLVM, (c) and (d) SGPLVM with KBR back-constraints with respect to MFCC, (e) and (f) SGPDM, (g) and (h) SGPDM with KBR back-constraints with respect to MFCC. . . 112
4.11 Hidden Markov model initialisation of latent points. 115
4.12 Varying sparse approximations: (a) LIPS (b) DEMNOW. 118
4.13 Varying latent space initialisation methods: (a) LIPS (b) DEMNOW. . 119
4.14 Varying latent space dimensionality: (a) LIPS (b) DEMNOW. 119
4.15 Varying dynamical GP RBF kernel inverse width: (a) LIPS (b) DEMNOW. 120
4.16 Varying likelihood and dynamics bias: (a) LIPS (b) DEMNOW. 121
4.17 Varying KBR back-constraints RBF kernel inverse width: (a) LIPS (b) DEMNOW. 122
4.18 Varying MLP back-constraints number of hidden layers: (a) LIPS (b) DEMNOW. 123
4.19 Comparing different latent space initialisation methods during SGPDM synthesis: (a) LIPS (b) DEMNOW. 123

4.20 Comparing SGPDM synthesis results for different speech parameterisations: (a) LIPS (b) DEMNOW. 124
4.21 Comparing SGPDM synthesis results for different audio-visual synchronisation methods: (a) LIPS (b) DEMNOW. 126

5.1 Graphical model for switching linear dynamical system and deterministic process dynamical system. 131
5.2 Graphical model for switching shared Gaussian process dynamical model. 135
5.3 VLMM training procedure according to Guyon and Pereira [130]: (a) Build prefix tree. (b) Build PST from prefix tree. (c) Convert PST to PFSA. 141
5.4 A hypothetical PFSA for a second-order VLMM on a sequence of phonemes. 146
5.5 LIPS - Perplexity tests on VLMM trained on: (a) repeating phonemes (b) non-repeating phonemes. 147
5.6 DEMNOW - Perplexity tests on VLMM trained on: (a) repeating phonemes (b) non-repeating phonemes. 148
5.7 LIPS - Occupancy of VLMM states for VLMM trained on: (a) repeating phonemes (b) non-repeating phonemes. 149
5.8 DEMNOW - Occupancy of VLMM states for VLMM trained on: (a) repeating phonemes (b) non-repeating phonemes. 150
5.9 SGPDM for VLMM states of the LIPS corpus: (a) VLMM state b oy oy oy oy oy oy. (b) VLMM state l ey ey ey ey ey. (c) VLMM state y y uw uw uw. 152
5.10 SGPDM for VLMM states of the DEMNOW corpus: (a) VLMM state P AY AY AY AY AY AY. (b) VLMM state P EY EY EY EY. (c) VLMM state R AY AY AY AY AY AY. 153
5.11 SGPDM for phonetic states of the LIPS corpus: (a) Phoneme state ah. (b) Phoneme state k. (c) Phoneme state eh. 154
5.12 SGPDM for Phoneme states of the DEMNOW corpus: (a) Phoneme state AW. (b) Phoneme state G. (c) Phoneme state OW. 155

6.1 Comparing ground truth and synthethic trajectories for: Voice Puppetry [32], shared LDS [186], Coupled HMM [312] and SGPDM for three test LIPS sequences. 167
6.2 Comparing ground truth and synthethic trajectories for: Voice Puppetry [32], shared LDS [186], Coupled HMM [312] and SGPDM for three test DEMNOW sequences. 168

6.3 Comparing ground truth and synthethic trajectories for the: Voice Puppetry [32], DPDS [102], Coupled HMM [312] and SSGPDM for three test LIPS sequences. 171
6.4 Comparing ground truth and synthethic trajectories for the: Voice Puppetry [32], DPDS [102], Coupled HMM [312] and SSGPDM for three test DEMNOW sequences. 172
6.5 Process of pasting eye blinks: (a) Original image with no eye blinks. (b) Image from which eye blinks are to be pasted from. (c) Shape alignment using generalised Procrustes analysis. (d) Segmented image without eye blinks. (e) Segmented image with eye blinks pasted. 181
6.6 Visual speech synthesis experiment - Main page. 182
6.7 Visual speech synthesis experiment - Popup page for user profile. 183
6.8 Visual speech synthesis experiment - Turing test. 185
6.9 Turing test results. 185
6.10 Visual speech synthesis experiment - Perceptual test. 186
6.11 Perceptual results for *quality of mouth articulation*. 187
6.12 Perceptual results for *naturalness*. 187
6.13 Perceptual results for *agreeableness*. 187
6.14 Visual speech synthesis experiment - Intelligibility test. 190
6.15 Intelligibility scores. 190

7.1 The BIWI 3D Audiovisual Corpus of Affective Communication [106]. . . 199

D.1 Synthesis frames a sequence of the LIPS dataset with the utterance: "An arch in Barbara's garden was heart shaped" with BEEP phonetic labels underneath each frame. 243
D.2 Synthesis frames for a sequence of the DEMNOW dataset with the utterance: "On Wednesday Jill Carroll's sister Katie" with CMU phonetic labels underneath each frame. 244

List of Algorithms

1 Active appearance model search 61
2 The Viterbi algorithm for the HMM 90
3 Training algorithm for prediction suffix tree (PST). 140
4 Backtracking to infer non-overlapping VLMM states 142
5 SSGPDM training algorithm for overlapping switching states 152
6 SSGPDM training algorithm for non-overlapping switching states 154
7 Sequential optimisation of latent points. 156
8 Sequential prediction of latent points. 157

Abstract

Visual speech synthesis deals with synthesising facial animation from an audio representation of speech. In the last decade or so, data-driven approaches have gained prominence with the development of Machine Learning techniques that can learn an audio-visual mapping. Many of these Machine Learning approaches learn a generative model of speech production using the framework of probabilistic graphical models, through which efficient inference algorithms can be developed for synthesis.

In this work, the audio and visual parameters are assumed to be generated from an underlying latent space that captures the shared information between the two modalities. These latent points evolve through time according to a dynamical mapping and there are mappings from the latent points to the audio and visual spaces respectively. The mappings are modelled using Gaussian processes, which are non-parametric models that can represent a distribution over non-linear functions. The result is a non-linear state-space model. It turns out that the state-space model is not a very accurate generative model of speech production because it assumes a single dynamical model, whereas it is well known that speech involves multiple dynamics (for e.g. different syllables) that are generally non-linear. In order to cater for this, the state-space model can be augmented with switching states to represent the multiple dynamics, thus giving a switching state-space model. A key problem is how to infer the switching states so as to model the multiple non-linear dynamics of speech, which we address by learning a variable-order Markov model on a discrete representation of audio speech. Various synthesis methods for predicting visual from audio speech are proposed for both the state-space and switching state-space models.

Quantitative evaluation, involving the use of error and correlation metrics between ground truth and synthetic features, is used to evaluate our proposed method in comparison to other probabilistic models previously applied to the problem. Furthermore, qualitative evaluation with human participants has been conducted to evaluate the realism, perceptual characteristics and intelligibility of the synthesised animations. The results are encouraging and demonstrate that by having a joint probabilistic model of audio and visual speech that caters for the non-linearities in audio-visual mapping, realistic visual speech can be synthesised from audio speech.

Declaration

No portion of the work referred to in this thesis has been submitted in support of an application for another degree or qualification of this or any other university or other institute of learning.

Copyright

To all those who supported me in my journey,

and the invisible hand that made it happen.

Carry On

It's easy to fight when everything's right,
And you're mad with the thrill and the glory;
It's easy to cheer when victory's near,
And wallow in fields that are gory.
It's a different song when everything's wrong,
When you're feeling infernally mortal;
When it's ten against one, and hope there is none,
Buck up, little soldier, and chortle:
  Carry on! Carry on!
There isn't much punch in your blow.
You're glaring and staring and hitting out blind;
You're muddy and bloody, but never you mind.
  Carry on! Carry on!
You haven't the ghost of a show.
It's looking like death, but while you've a breath,
  Carry on, my son! Carry on!
And so in the strife of the battle of life
It's easy to fight when you're winning;
It's easy to slave, and starve and be brave,
When the dawn of success is beginning.
But the man who can meet despair and defeat
With a cheer, there's the man of God's choosing;
The man who can fight to Heaven's own height
Is the man who can fight when he's losing.
  Carry on! Carry on!
Things never were looming so black.
But show that you haven't a cowardly streak,
And though you're unlucky you never are weak.
  Carry on! Carry on!
Brace up for another attack.
It's looking like hell, but – you never can tell:
  Carry on, old man! Carry on!
There are some who drift out in the deserts of doubt,
And some who in brutishness wallow;
There are others, I know, who in piety go
Because of a Heaven to follow.
But to labour with zest, and to give of your best,
For the sweetness and joy of the giving;
To help folks along with a hand and a song;
Why, there's the real sunshine of living.
  Carry on! Carry on!
Fight the good fight and true;
Believe in your mission, greet life with a cheer;
There's big work to do, and that's why you are here.
  Carry on! Carry on!
Let the world be the better for you;
And at last when you die, let this be your cry:
  Carry on, my soul! Carry on!

(Robert William Service)

Acknowledgements

The last four years at The University of Manchester have been truly amazing thanks to a number of people. First and foremost, I can never thank enough my supervisor, Dr. Aphrodite Galata who inspired me to undertake doctoral research in Computer Vision and who provided me with the training to attain the level of critical thinking required for a PhD. For your patience, understanding, guidance and much more - thank you. I am thankful to my advisor Dr. Ernie Hill who has been a great guide in helping me understand what a PhD involves in the initial stages. Special thanks to all members of the AIG group who turned out to be great buddies - Shaobo, Francho, Martin, Xin, Raza and others. The vibrant community in the School of Computer Science was more than I could ever ask for: Geoffrey, Chenjuan, Luke, Yuanjing, Adam, Demian, Wuxiang, Rishi, Franck, James, Paris, Isuru and many others - our stimulating discussions over lunch and coffee and my awkward attempts to practice Mandarin were fun times. Being part of the mentoring team within the school was an amazing experience thanks to Jasmin, Alex, Grace, Eleni, Isuru and Farzaneh who made such an excellent team. I would also like to thank Toby Howard for introducing me to public engagement at the Manchester Science Festival, with which I was involved for the last three years of my PhD.

Thanks to Prof. Neil Lawrence for the publicly available Gaussian process software that this thesis builds on. I would also like to thank Dr. Gwenn Englebienne and Dr. Barry-John Theobald for their publicly available audio-visual corpora used in this thesis.

A big thank you to my family who has also been a source of support and guidance throughout, generously offering help, encouragement and financial support in difficult times. To all my friends who were there for me at different times and locations - I am truly blessed to have you. My mentor, T.S. Vijayaraghavan in Mauritius - how lucky I was to have you at the right place and time and I only now start to grasp the significance of the transformation you spawned in my life. To all the teachers I encountered throughout my life - you whetted my thirst for knowledge and gave me the means to quench it using the tools of reason and intuition.

I gratefully acknowledge the studentship from The University of Manchester for my

postgraduate studies which made it all possible. I am also thankful to the wonderful support provided by the EPS faculty to research students and in particular to Dee-Ann Johnson for several exciting graduate development workshops. This list will not be complete without mentioning Dr. Elzbieta Pekalska for her personal development workshop in my first year, for reinforcing my understanding of the fundamental nature of consciousness and for being an inspiring friend. I was lucky to have been allowed to take part in the Manchester Leadership Programme for Researchers by the excellent career service at the university. Special mention to the team that I was part of: Kawther, Ben, Philip, David and Delali - I liked the way that we formed a synergy that was larger than the sum of our individualities in order to work on the community project.

I have come to see the PhD journey as a character-building process, refining thinking, strenghening will and determination and also broadening the horizons. I am grateful to everyone who has been part of this journey in one form or the other and apologise if I have omitted any names for the sake of brevity.

Abbreviations

3DMM	3D Morphable Model
AAM	Active Appearance Model
ACC	Average Correlation Coefficient
ACM	Active Contour Model
ADF	Assumed Density Filtering
AMSE	Average Mean Squared Error
ANN	Artificial Neural Network
ANOVA	Analysis-of-Variance
ARD	Automatic Relevance Determination
ASM	Active Shape Model
AVSR	Audio-Visual Speech Recognition
CCA	Canonical Correlation Analysis
CHMM	Coupled Hidden Markov Model
CODE	Compositional Gradient Descent
CV	Consonant-Vowel
DPDS	Deterministic Process Dynamical System
DCT	Discrete Cosine Transform
DFT	Discrete Fourier Transform
DTC	Deterministic Training Conditional
EC	Expectation Correction
EM	Expectation-Maximisation
EP	Expectation Propagation
FA	Factor Analysis
FAP	Face Animation Parameters
FDP	Face Definition Parameters
FFT	Fast Fourier Transform
FITC	Fully Independent Training Conditional
GP	Gaussian Process
GPA	Generalised Procrustes Analysis
GPLVM	Gaussian Process Latent Variable Model
GPDM	Gaussian Process Dynamical Model
GMM	Gaussian Mixture Model
HCI	Human-Computer Interaction
HMM	Hidden Markov Model
ICA	Independent Component Analysis
ICIA	Inverse-Compositional Image Alignment
KBR	Kernel-Based Regression

KNN	K-nearest neighbour
LDS	Linear Dynamical System
LLE	Locally Linear Embedding
LP	Linear Prediction
LPC	Linear Predictive Coding
LSF	Line Spectral Frequencies
MAE	Maximum Absolute Error
MAP	Maximum-a-Posteriori
MCMC	Markov Chain Monte Carlo
MCTE	Minimum Converted Trajectory Error
MDS	Multi-Dimensional Scaling
MGE	Minimum Generation Error
MLLR	Maximum Likelihood Linear Regression
MSE	Mean Squared Error
MFCC	Mel-Frequency Cepstral Coefficients
ML	Maximum Likelihood
MLP	Multi-Layer Perceptron
MMM	Multidimensional Morphable Model
MOS	Mean Opinion Score
NCCA	Non-Consolidating Component Analysis
NTSC	National Television System Committee
PAL	Phase Alternating Line
PCA	Principal Component Analysis
PFSA	Probabilistic Finite State Automaton
PCCA	Probabilistic Canonical Correlation Analysis
PPCA	Probabilistic Principal Component Analysis
PDM	Point Distribution Model
PST	Prediction Suffix Tree
PITC	Partially Independent Training Conditional
PLP	Perceptually Linear Prediction
RASTA-PLP	Relative Spectral Perceptual Linear Prediction
RBF	Radial Basis Function
RMSE	Root Mean Squared Error
SAD	Sum of Absolute Differences
SCG	Scaled Conjugate Gradient
SGPLVM	Shared Gaussian Process Latent Variable Model
SGPDM	Shared Gaussian Process Dynamical Model
SLDS	Switching Linear Dynamical System
SNR	Signal-to-Noise Ratio
SSGPDM	Switching Shared Gaussian Process Dynamical Model
SSM	Stochastic Segment Model
SVD	Singular Value Decomposition
SVM	Support Vector Machine
TTS	Text-to-Speech
VC	Vowel-Consonant
VCV	Vowel-Consonant-Vowel
VHML	Virtual Human Markup Language
VLMM	Variable Length Markov Model
VLHMM	Variable Length Hidden Markov Model

Notations

General Notation

Symbol	Meaning
$\mathbb{N}$	The set of natural numbers
$\mathbb{R}$	The set of real numbers
$\mathbb{C}$	The set of complex numbers
t	Time
T	Number of frames in a sequence
a	Lower case letters for scalar values
$\mathbf{x}$	Bold lower case letters for vectors
I	Upper case letters for images
$\mathbf{X}$	Bold upper case letters for matrices
$\mathbf{I}$	Identity matrix
δx	Small change in x
x_i	i^{th} dimension of variable $\mathbf{x}$
$\mathbf{x}_i$	i^{th} data vector (or row) from matrix $\mathbf{X}$
$\mathbf{x}_{:,j}$	j^{th} column from matrix $\mathbf{X}$
$x_{i,j}$	i^{th} row and j^{th} column from matrix $\mathbf{X}$
x_t	Vector $\mathbf{x}$ at time t
$\{\mathbf{x}_n\}_{n=1}^{N}$	Set of vectors $[\mathbf{x}_1, \ldots, \mathbf{x}_N]^T$
$\{\mathbf{x}_t\}_{t=1}^{T}$	Set of time-series vectors $[\mathbf{x}_1, \ldots, \mathbf{x}_T]^T$
$\triangle$	Difference vector
$\frac{dy}{dx}$	Derivative of x with respect to y
$\frac{\partial y}{\partial x}$	Partial derivative of x with respect to y
$f(x)$	Function f of x
$diag(x_1, \ldots x_N)$	The diagonal matrix with diagonal elements given by $x_1, \ldots x_N$
$tr(\mathbf{X})$	Trace of matrix $\mathbf{X}$
$det(\mathbf{X})$	Determinant of matrix $\mathbf{X}$
$\mathbf{x}^T$	Transpose of vector $\mathbf{x}$
$\lvert\mathbf{x}\rvert$	Cardinality of vector $\mathbf{x}$
$\parallel \mathbf{x} \parallel$	Euclidean or L_2-norm of vector $\mathbf{x}$
$< \mathbf{x}, \mathbf{y} >$	Inner product between two vectors: $\mathbf{x}$ and $\mathbf{y}$
$\delta_{x,y}$	Kronecker delta between x and y. $\delta_{x,y} = 1$ if $x = y$ and 0 otherwise

Probability and Statistics, Information Theory

Symbol	Meaning
$p(\mathbf{x})$	Probability of $\mathbf{x}$
$p(\mathbf{x}, \mathbf{y})$	Probability of $\mathbf{x}$ and $\mathbf{y}$
$p(\mathbf{x}\|\mathbf{y})$	Probability of $\mathbf{x}$ given $\mathbf{y}$
$\mathbb{E}[\mathbf{x}]$	Expectation of $\mathbf{x}$
μ	Mean of univariate variable
σ	Variance of univariate or multivariate variable
$\boldsymbol{\mu}$	Mean of multivariate variable
$\boldsymbol{\Sigma}$	Covariance matrix of multivariate variable
$\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$	Gaussian distribution with mean $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$
$D_{KL}(P\|\|Q)$	Kullback-Leibler divergence between probability distributions P and Q

Facial Modelling

Symbol	Meaning
$\mathbf{s}$	Shape vector
$\mathbf{g}$	Texture vector
$\mathbf{c}$	Combined shape and texture vector
$\bar{\mathbf{s}}$	Mean of shape vector $\mathbf{s}$
$\bar{\mathbf{g}}$	Mean of texture vector $\mathbf{g}$
$\bar{\mathbf{c}}$	Mean of combined shape and texture vector $\mathbf{b}$
$\mathbf{P_s}$	Eigenvectors of shape vector $\mathbf{s}$
$\mathbf{P_g}$	Eigenvectors of texture vector $\mathbf{g}$
$\mathbf{P_c}$	Eigenvectors of combined shape and texture $\mathbf{b}$
$\mathbf{b_s}$	Shape principal components
$\mathbf{b_g}$	Texture principal components
$\mathbf{b_c}$	Combined shape and texture principal components

Speech Processing

Symbol	Meaning
$F(z)$	Fourier transform of function f of m
$f(m)$	Inverse Fourier transform of function F of z
$\mathrm{DFT}(X)$	Discrete Fourier transform of set of complex numbers $X = \{X_1, \ldots, X_N\}$
$\mathrm{DCT}(X)$	Discrete cosine transform of set of real numbers $X = \{X_1, \ldots, X_N\}$

Probabilistic Generative Models

Symbol	Meaning
s_k	Starting probability for state k
m_k	Mixing probability for state k
π	Latent state. $\pi \in \{1 \dots K\}$
π_t	Latent state of frame t
ϵ	Noise term
ϵ_{π_t}	Noise term for state π_t of frame t
$\mathbf{A}$	Transition matrix
$\mathbf{A}_{\pi_t}$	Transition matrix for state π_t of frame t
$\mathbf{B}$	Observation matrix for audio data
$\mathbf{B}_{\pi_t}$	Observation matrix for audio data for state π_t of frame t
$\mathbf{C}$	Observation matrix for visual data
$\mathbf{C}_{\pi_t}$	Observation matrix for visual data for state π_t of frame t

Variable Length Markov Model

Symbol	Meaning
Q	Finite set of model states
Σ	Set of tokens
τ	Transition function
γ	Output probability function
s	Probability over initial states
w	Subsequence of length N
σw	The suffix of word w

Gaussian Processes

Symbol	Meaning
$\mathbf{y}$	D-dimensional vector of data points
$\mathbf{x}$	d-dimensional vector of latent points
$\mathbf{Y}$	Set of data points $\mathbf{Y}$=$[\mathbf{y}_1, \dots, \mathbf{y}_N]^T$
$\mathbf{X}$	Set of latent points $\mathbf{X}$=$[\mathbf{x}_1, \dots, \mathbf{x}_N]^T$
D	Dimensionality of data point $\mathbf{y}_i$
d	Dimensionality of latent point $\mathbf{x}_i$
$k(\mathbf{x}, \mathbf{x}')$	Kernel function
$\mathbf{K}$	Kernel matrix
$\mathbf{\Phi}$	Gaussian Process hyperparameters
h	Transition function
h_{π_t}	Transition function for state π_t of frame t
f	Observation function for audio data
f_{π_t}	Observation function for audio data for state π_t of frame t
g	Observation function for visual data
g_{π_t}	Observation function for visual data for state π_t of frame t

Chapter 1

Introduction

> A journey of a thousand miles begins with a single step.
>
> Lao Tzu

1.1 Overview

Visual speech synthesis involves generating synthetic talking heads uttering human speech such that the facial movements and expressions synchronise with the speech. It has been widely studied over several decades both in academia and industry and draws the interest of computer scientists, linguists, animators and even psychologists. Some of its applications include cinema, computer games, Human-Computer Interaction (HCI), education and even medicine.

Historically, the area emerged in the 1970's with geometrical 3D models of the face that could be morphed using control points to achieve the desired facial configuration [231]. Animation was then achieved through keyframing, i.e. the desired animation timeline was segmented into prototypes representing the basic units of facial movements. These keyframes were represented by control points of the facial model and animation was achieved by interpolating between the keyframes. An animator would analyse the audio containing the desired utterance, and would produce the segmentation accordingly. This was a very labour-intensive process and subject to errors and mismatches. The 1990's saw the emergence of rule-based techniques that were based on the premise that the rules that govern facial movements could be manually handcrafted such that these could be used to infer the facial configurations given a representation of speech [54, 17, 239]. The last decade or so has seen the advent of the data-driven approach to facial animation, where the rules governing facial movements from speech are learnt automatically from data using Machine Learning techniques [32, 105, 42, 102].

Data-driven approaches are mostly generative, i.e. they try to model the process that generated the speech and once the parameters of this model have been estimated during training, the same model can then be used in synthesis mode on novel input speech. In this thesis, our aim is to adopt a data-driven approach, modelling the generative process of speech using Machine Learning techniques and using the learnt model to synthesise visual speech from audio.

Data-driven approaches can be further categorised by the ways that the face is modelled and by how the mapping from audio to visual speech is achieved. We now present a brief overview of these two categorisations.

1.1.1 Models of the Face

When choosing the model of the face to use for visual speech synthesis, the application domain needs to be considered. If flexibility and manual control of facial configurations is needed, a 3D model of the face can be used [42, 321, 92]. However the main limitation of 3D models of the face is that it is difficult to attain a high level of realism and expensive equipment is required to capture the geometry of the face. On the other hand, 2D image-based methods have had success in achieving high levels of realism [34, 32, 105, 279]. Appearance-based methods [160, 60] are a subset of image-based methods that perform analysis on images to recover structure such as facial landmarks but can also synthesise novel images from a compact set of parameters.

Figure 1.1 shows an illustration of the categorisation of facial modelling approaches.

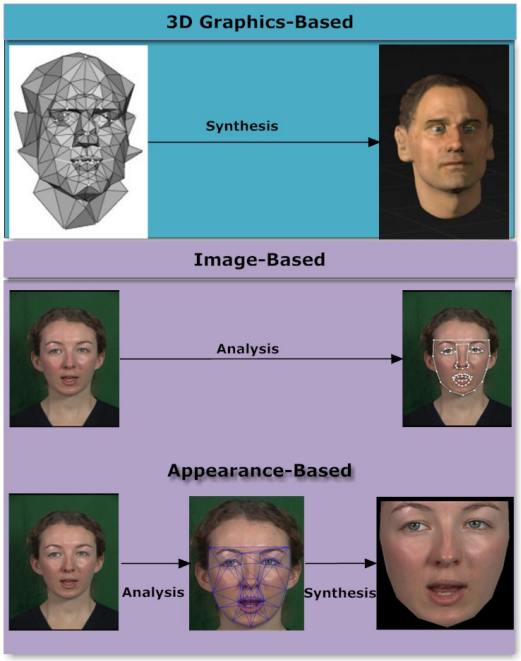


Figure 1.1: Overview of facial modelling approaches used in visual speech synthesis.

1.1.2 Models of Audio-visual Mapping

Deng and Neumann [77] distinguish between two types of data-driven techniques to audio-visual mapping: sample-based approaches and learning-based approaches. Sample-based approaches were dealt with in the PhD theses of Cosatto [61], Ypsilos [319] and Deng [76] and the focus is on using a large corpus of facial animations to find concatenative units that can then be reordered to match a target utterance. Learning-based approaches, on the other hand aim at using Machine Learning techniques to learn a mapping from audio data to visual data and use the mapping to predict visual data from audio data. They have been used in the PhD theses of Beskow [17], Ezzat [104], Theobald [278], Cosker [63], Lehn-Schiøler [185], Englebienne [101] and Hofer [142]. The data-driven approach is illustrated in Figure 1.2. Sample-based methods require storage of the whole image corpus but can yield very realistic facial animation [191, 302]. Learning-based approaches, on the other hand, provide a compact statistical representation of facial behaviour and offer more flexibility in terms of adapting the statistical models to new identities as well as modelling other modalities of speech such as head and eye movements and expressions.

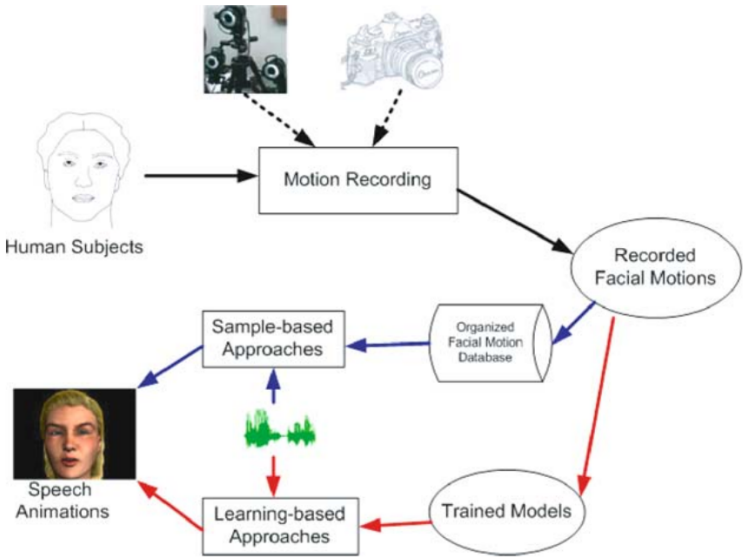


Figure 1.2: Overview of data-driven techniques for audio-visual mapping [77].

1.2 Realistic Speech-driven Facial Animation

One of the goals of facial animation is to achieve a high level of realism so that it can be used in demanding applications such as computer games and cinema. There are two ways to define and measure realism:

Photorealism is a measure of the *static realism* of facial images and a photorealistic facial animation implies that the synthesised facial images look like photographs.

Videorealism is a measure of the *dynamic realism* of facial animation and a video-realistic facial animation implies that the facial motions (lips, teeth, tongue, eyes, etc.) seem realistic and look like it is an actual person speaking. The motions need not only be smooth but they have to be physically plausible.

1.2.1 Challenges

Realistic facial animation driven by speech is challenging for the following reasons:

- The mapping from sounds to facial configurations is many-to-many, i.e. multiple sounds might map to a given facial configuration and multiple facial configurations might correspond to a given sound.
- The mapping from audio to visual speech is highly dependent on context, i.e. the facial configurations corresponding to a given sound depend on the sounds that come before and afterwards. The phenomenon is called coarticulation and is dealt with in greater depth in Chapter 2.
- The animation has to be smooth and respect the dynamics of the face. Humans are highly sensitive to the way faces look and behave and slight imperfections in facial appearance and movements are highly noticeable and might lead to repulsion in humans.

1.2.2 Research Problems

The following highlights some of the areas that currently draw research interest in visual speech synthesis:

- **Realism**: Realism remains a major challenge in visual speech synthesis. The choice of methods used for facial modelling and audio-visual mapping greatly influences the level of realism achieved. 2D image-based and appearance-based techniques of facial modelling have achieved the highest levels of realism [104, 279, 191, 302]. The reason is because 3D graphics-based techniques look artificial and cartoon-like unless very dense facial meshes are used. Dense facial meshes however, require a very fine level of control to mimick reality, which is possible only if a large corpus is recorded using expensive motion capture techniques. 2D appearance-based approaches on the other hand only require high quality video recording and facial behaviour can be controlled using statistical parameters representing the modes of facial variation [59, 160].
- **Transferability**: In practical applications such as human-computer interaction (HCI), games or cinema, it is not expected to have high quality data for all identities for which speech animation is to be carried out. It is thus highly

desirable to be able to adapt both the facial models and the models of audio-visual mapping to novel identities using a limited amount of data for the new person. Transferable speech animation was demonstrated by Chang and Ezzat [47] and remains a challenging area of research, due to the multiple types of facial models that can be used and also because of the difficulty in being able to transfer the speaking style and idiosyncracies of a particular person using a limited amount of data.

- **Expressiveness**: Research in psychology has shown that human communication is primarily non-verbal [311]. Thus, speech animation that looks inexpressive would be repulsive to humans and might even illicit responses characteristic of the Uncanny Valley [215] (refer to Chapter 2 Section 2.3 for more details). Expressive visual speech synthesis thus needs to be able to incorporate both non-verbal gestures such as eye blinks, pauses and breaths, as well as facial expressions that convey emotions such as happiness, anger, sadness, surprise, despair, etc. However, expressive cues depend both on the language content and the emotional tone of the audio speech [81], which makes expressive visual speech synthesis a very challenging problem. Moreover, the expressive and articulatory aspects of speech are interdependent [18], which requires the joint modelling of these different modalities of speech. Expressive speech animation remains an open problem and draws attention from both industry and academia.

1.3 Research Aims

In this thesis, we focus on the realism aspect of visual speech synthesis using a 2D appearance-based approach to facial modelling and a learning-based approach to audio-visual mapping. We use a 2D appearance-based facial modelling approach because of the higher level of realism that is achievable as compared to 3D facial models and also because appearance models can allow the extension of our work to expressive and transferable speech animation by adapting the facial model. We choose a learning-based approach because it also can allow extension to expressive and transferable speech animation by exploiting the statistical nature of the audio-visual mapping. Specifically, learning-based approaches can allow the integration of additional speech modalities such as expressions and non-verbal gestures.

Our aim is to improve the state-of-the-art in the area of learning-based speech-driven facial animation. We focus on the audio-visual mapping problem, aiming to automatically capture and model the non-linear relationship between audio and visual dynamics during speech by jointly modelling audio and visual speech using a shared latent space. The shared latent space provides a non-linear embedding of the shared

information between audio and visual speech, which are physiologically coupled modalities and thus highly correlated. In so doing, we aim to achieve more articulate and realistic facial animation compared to previous methods. We focus less on performance and real-time issues and thus the application domains that this work best fits into are those that do not require real-time performance. Potential applications include: cinema, speech therapy, surgical planning as well as virtual actors, tutors and anchors. More applications of visual speech synthesis are given in Chapter 2.

1.4 Thesis Contributions

The following gives an overview of the main contributions of this thesis.

- **Explicitly model the non-linearities in audio-visual mapping**: Audio-visual mapping is highly non-linear because of ambiguities in both the audio and visual domains [207]. Previous methods have used either linear approximations [185] or piecewise linear methods to model the non-linearities [32, 312, 102, 326, 304]. In this thesis, we explicitly model the non-linearities in audio-visual mapping using non-linear Gaussian processes (GPs) [251]. We show that this approach results in speech animation that better matches ground truth as compared to the linear or piecewise linear methods.
- **Use both discrete and continuous audio to predict facial behaviour**: Visual speech synthesis methods can be either driven by a discrete speech representation [32, 104, 102] or using continuous speech parameters [185, 312, 326, 304]. The first allows a principled approach to modelling coarticulation by exploiting the structure of language but discards prosodic information in the speech signal. The converse is true for the latter approach. In this thesis, we aim to combine the advantages of both discrete and continuous audio representations by using them to predict visual speech. In particular, we learn a language model on discrete phonemes in order to automatically identify the commonly occuring segments of speech. We model the audio and visual data corresponding to each of those segments jointly and devise synthesis algorithms to predict visual data from audio data whilst explicitly taking forward and backward context of speech into account. We perform experiments to compare the effectiveness of different audio speech parameterisation techniques in predicting visual speech and also investigate the effect of explicitly modelling phonetic context in our method by comparing the synthesis results against a closely related method that does not model phonetic context [102].
- **Realistic and articulate speech animation**: In this thesis, we use non-parametric models called Gaussian processes (GPs) [251] to address the problem of under-articulation obtained previously using parametric models [32, 185,

312, 102]. We perform quantitative evaluation to compare our proposed method against previously proposed parametric models. We also perform subjective tests to assess the level of realism of synthetic as compared to ground truth animations. Experiments are also performed to test whether or not our animations fall into the Uncanny Valley [215] as well as the effect of upper face movements such as eye blinks on the perception of visual speech. We also aim to achieve a high level of intelligibility that is comparable to ground truth videos, which we test using a human lip-reading test.

1.5 Thesis Structure

Chapter 2 reviews human speech production, perception and structure, examines the phenomenon of coarticulation in detail and looks at existing techniques for visual speech synthesis. Moreover, applications of visual speech synthesis are also considered.

Chapter 3 reviews facial modelling and speech processing techniques. The data corpora used in this work are then described, followed by the techniques we use to parameterise the audio and visual streams. Different speech parameterisation methods and audio-visual synchronisation methods are considered.

Chapter 4 presents various state-space methods that have been used in visual speech synthesis and introduces the shared Gaussian process dynamical model (SGPDM) [98] to jointly model audio and visual parameters. Both training and inference for the SGPDM as well as synthesis techniques for the SGPDM are described. Experiments to perform model selection and to choose the best audio parameterisation and audio-visual synchronisation techniques are also dealt with.

Chapter 5 deals with different switching state-space methods that have been used for visual speech synthesis and describes a novel extension of the SGPDM, called the switching SGPDM (SSGPDM) [48]. Both training and inference techniques for the SSGPDM are discussed. An in-depth description of the variable length Markov model (VLMM) [130], used to infer variable-order switching states, is presented, as well as its training and inference algorithms. Two synthesis algorithms proposed for the SSGPDM and applied to audio-visual mapping are also described.

Chapter 6 presents both objective and subjective evaluation results.

Chapter 7 concludes the thesis with a summary of its contributions, limitations and directions for future work.

In this thesis, “they” will be used to refer to either he or she.

Figure 1.3 illustrates an overview of the proposed method for visual speech synthesis.

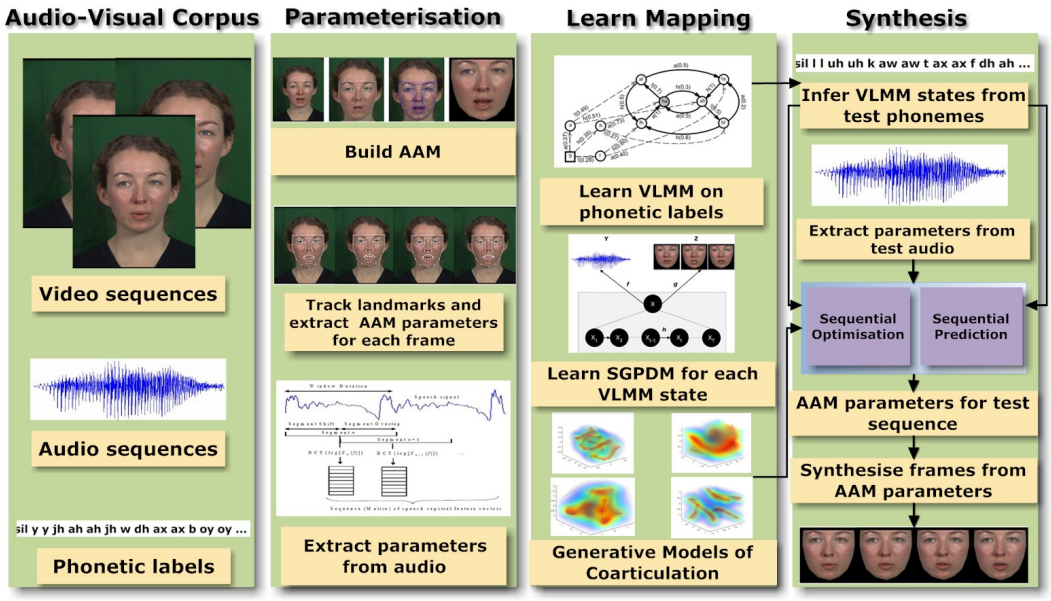


Figure 1.3: Overview of the proposed method.

1.6 Publications

The publications that have resulted from this thesis are:

- Salil Deena and Aphrodite Galata. Speech-driven facial animation using a shared Gaussian process latent variable model. In *ISVC'09: Proc. of the International Symposium on Visual Computing*. Springer, 2009
- Salil Deena, Shaobo Hou, and Aphrodite Galata. Visual speech synthesis by modelling coarticulation dynamics using a non-parametric switching state-space model. In *ICMI-MLMI'10: Proc. of the International Conference on Multimodal Interfaces and Workshop on Machine Learning for Multimodal Interaction*. ACM, 2010

Chapter 2

Background

> Man cannot discover new oceans unless he has the courage to lose sight of the shore.
>
> André Gide

This chapter presents some background on human speech and on its multimodal nature, namely its audio and visual realisations. Audio speech is produced by the speech production system and the facial gestures that accompany it can be viewed as forming part of the speech production process. However, on the perceptual side, these two aspects of speech are decoupled. Humans need to interpret both channels to disambiguate speech. Both the speech production process and the speech perception process are covered in this chapter. The phonological aspect of speech is then presented, followed by a description of the phenomenon of coarticulation. A review of techniques that have been used to generate synthetic visual speech from audio speech is also given. We also discuss the Uncanny Valley, which is an important finding in the perception of near-realistic facial animation. Finally, we present some applications of visual speech synthesis.

2.1 Human Speech

Human speech is produced by the speech production system and is perceived by the speech perception system. The following presents some details on the speech production and perception systems as well as on the audio and visual components of speech and how they relate to each other. The dynamical nature of speech (coarticulation) is then dealt with in terms of various theories of speech production as devised by linguists.

2.1.1 Human Speech Production

Human speech production is a complex process produced by the speech production system, which comprises of: the lungs, trachea, larynx (vocal cords) and pharyngeal, oral and nasal cavities. This is illustrated in Figure 2.1. The vocal tract is composed of the pharyngeal and oral cavities whilst the nasal tract constitutes the nasal cavity. The following gives a description of the role of the different parts of the speech production system according to Huang et al. [151].

- *Lungs and trachea*: Air is exhaled by the lungs and passes through the trachea or windpipe.
- *Vocal cords*: Responsible for producing either *voiced* sounds, when the vocal folds are held close together and vibrate against one another; or *unvoiced* sounds, when the vocal folds are too slack to vibrate periodically. The resonance of the vocal tract is also known as formants.
- *Soft Palate (Velum)*: Acts as a valve, which when open allows passage of air through the nasal cavity.
- *Hard Palate*: A rigid structure, which when the tongue presses against it during speech, consonants are produced.
- *Tongue*: A flexible structure that presses against the palate to produce consonants.
- *Teeth*: Rigid structures that also help in certain consonant production in conjunction with the tongue.
- *Lips*: Involved in both vowel and consonant production. Lip rounding can spread and affect vowel quality whilst lip closing is involved in producing certain consonants.

The following information is conveyed by speech [291].

- *Acoustic phonetic signals* - The elementary speech units from which larger speech units are formed.
- *Prosody* - The rhythms of speech which help to create intonation, indicate boundaries in segments of speech, link sub-phrases and clarify intention.
- *Gender information* - This is conveyed by the pitch.
- *Age* - Conveyed by the condition of the vocal tract as well as the pitch.
- *Accent* - Conveyed by a combination of changes in pronunciation as well as systematic changes in formants, pitch, intonation, duration, emphasis and stress.
- *Speaker's identity and health* - Conveyed by the physical characteristics of the person's vocal tract.
- *Emotion* - Conveyed by pitch, stress and intonation.

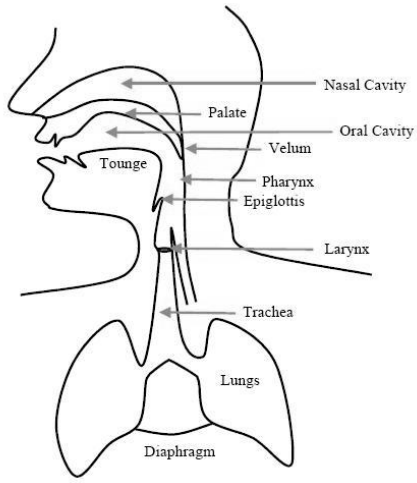


Figure 2.1: The speech production system [291].

2.1.2 Human Speech Perception

The human speech perception system comprises of the auditory system, the visual system and the brain. The ear transforms acoustic pressure into a mechanical vibration pattern, represented by a series of pulses, which is then transmitted by the auditory nerve. The intensity of the sound accounts for the perception of loudness whilst its fundamental frequency is perceived as pitch [151]. Tones of the same intensity but different pitch have different perceived loudness. In addition, the fundamental frequency also gives the perception of voice quality or *timbre*, which varies across genders and age groups. The human perception of pitch varies approximately linearly with respect to the actual pitch up till the pitch is about 500 Hz, above which it varies logarithmically [295]. A similar relationship exists between perceived loudness and the intensity of speech [151]. The visual perception of speech alone is less informative than audio, as evidenced by the fairly low accuracy scores of human lip reading tests [137]. However, McGurk and MacDonald [207] showed that ambiguities result in speech perception if a video of an utterance is dubbed with a different sound. A more detailed discussion of the McGurk effect is given in Section 2.1.4. More recently, Chandrasekaran and Ghazanfar [46] showed that because light travels faster than sound, what we read from the lips makes us anticipate the sound that will be produced and that mismatches between the sound and the lips result in different brain activations which negatively affect speech perception. These findings strongly support the importance of synchronisation between sounds and lip movements in visual speech synthesis.

2.1.3 Speech Primitives

Speech consists of the audio and visual components that both complement each other in creating perceptible speech. The audio part of speech has been traditionally studied by linguists and phonologists. Since the advent of audio-visual communication, a lot of research interest has gone into studying the visual aspects of speech by analysing how different articulators such as the tongue, lips and facial muscles move during speech production. The basic unit of audio speech is called the *phoneme* and the corresponding unit of visual speech is called the *viseme*.

Phoneme

In most languages, the basic unit of distinct sounds possible is called a phoneme. Different languages have different phonetic units and there are even variations within a particular language, depending on the accents. In speech recognition, pronunciation dictionaries are used to map words to their corresponding phonemes. For example, the pronunciation dictionary used for British English is called the British English Pronunciation Dictionary (BEEP) [116], which comprises of 44 phonemes. The pronunciation dictionary used for American English is the CMU Pronunciation Dictionary [308], which is based on a subset of the ARPABET table [169], and it defines 39 phonemes. The same word may map to different phonemes, depending on the pronunciation dictionary being used. For example, the word "on" in British English would result in the phonemes /OH/ /N/, whereas for American English, that would be /AH/ /N/, due to variations in the pronunciation.

The ARPABET table comprises of 48 phonemes, which can be categorised as: vowels, diphthongs, semivowels, nasal consonants, africatives, unvoiced fricatives and voiced fricatives [247, 175]. A description of each of these, as well as examples of corresponding phonemes from the ARPABET table are now given:

- *Vowels* are produced by an open larynx with no constriction of air pressure above the glottis. The vowel phonemes consist of: /AA/, /AE/, AH/, /AO/, /EH/, /IY/, /IH/, /UH/ and /UW/. Formant frequencies encode all the information required for humans to distinguish between vowels.
- *Diphthongs* consist of a gliding monosyllabic speech sound that starts at or near the articulatory position for one vowel and moves to or towards the position for another. There are five diphthongs in American English, namely: /AY/, /AW/, /EY/, /ER/, /OY/, /OW/.
- *Semivowels* are characterised by a gliding transition in the vocal tract area function between adjacent phonemes. They have a vowel-like nature, but are transitional towards other phonemes. The acoustic realisations of those sounds are

strongly influenced by context. /W/, /L/, /R/ and /Y/ are examples of semivowels.

- *Nasal consonants* are sounds produced when the vocal tract is totally constricted and instead, the sound emerges out of the nasal tract. The nasals comprise: /M/, /N/ and /NG/.
- *Unvoiced fricatives* are produced when the vocal tract is excited by a steady air flow, which becomes turbulent in the region of a constriction in the vocal tract. Examples are: /HH/, /F/, /S/ and /SH/.
- *Voiced fricatives* are the counterparts of the unvoiced fricatives with the difference that two excitation sources are involved in their production rather than one. /V/, /TH/, /DH/, /Z/ and /ZH/ are examples of voiced fricatives.
- *Affricatives* occur when a fricative occurs immediately after a constriction of air. /JH/ and /CH/ are affricatives.
- *Voiced and unvoiced stops* are produced by a building up of pressure behind a total constriction in the oral tract and then suddenly releasing the pressure. They comprise of: /B/, /D/, /G/, /P/, /T/ and /K/.

Syllable

Phonemes, when grouped together form syllables. Groups of syllables form words and a sequence of words forms a sentence. In phonology, the next unit after the phoneme is called a morpheme, which by definition, is a group of phonemes which has semantic meaning. Syllables, on the other hand, are groups of phonemes which do not necessarily have a semantic meaning.

Viseme

The viseme is the visual counterpart of the phoneme. Visemes are the facial configurations that result when pronouncing the different phonemes [110]. Because of occlusions that occur in the mouth when we perceive phonemes, there are much fewer viseme classes. Until recently, there was no standardised set of visemes as is the case for phonemes [49], but most researchers have agreed on 13-14 visemes [183, 312]. The MPEG-4 standard [216] has introduced various standards related to facial animation. This includes the definition of 14 visemes as well as the specification of standards for the animation of 3D face models by defining face definition parameters (FDP) and facial animation parameters (FAP).

Visyllable

In the same way as for phonemes, groups of visemes form visyllables. A visyllable is thus the visual realisation of a syllable.

MPEG-4 Viseme	BEEP Phoneme	CMU Phoneme
1	/p/, /b/, /m/	/P/, /B/, /M/
2	/f/, /v/	/F/, /V/
3	/dh/, /th/	/DH/, /TH/
4	/d/, /t/	/D/, /T/
5	/g/, /hh/, /k/, /w/	/G/, /HH/, /K/, /W/
6	/ch/, /jh/, /sh/, /zh/	/CH/, /JH/, /SH/, /ZH/
7	/s/, /z/	/S/, /Z/
8	/l/, /n/, /ng/,	/L/, /N/, /NG
9	/r/, /y/	/R/, /Y/
10	/aa/, /ae/, /ah/, /ao/, /ay/	/AA/, /AE/, /AH/, /AO/, /AY/
11	/ax/, /ea/, /eh/, /er/, /ey/	/EH/, /ER/, /EY/
12	/ia/, /ih/, /iy/	/IH/, /IY/
13	/oh/, /ow/, /oy/	/OW/, /OY/
14	/aw/, /ua/, /uh/, /uw/	/AW/, /UH/, /UW/

Table 2.1: Phoneme to viseme mapping for both BEEP and CMU phonemes.

2.1.4 Audio-visual Mapping

There is no simple one-to-one mapping from phonemes to visemes due to occlusions and ambiguities in visual perception that result from distance between the speaker and the listener. The mapping from phoneme classes to viseme classes is many-to-one, or, according to some classifications, many-to-many [183], because phonemes belong to multiple viseme classes and vice-versa. Different researchers give different phoneme to viseme groupings [273, 49, 183, 321, 312, 208] and the main reason is the lack of standardisation for visemes, and also because different languages have different phonetic alphabets. Table 2.1 shows the mapping from phonemes in the BEEP [116] and CMU phone sets [308] to MPEG-4 visemes [216], based on the categorisation given in [273].

The mapping between the audio realisation of a phoneme and the visual rendering of a viseme is also many-to-many, due to variations in the ways of pronuncing a phoneme and also in the visual appearance of a viseme. In addition, these are further complicated by coarticulation (refer to Section 2.1.5).

McGurk Effect

The McGurk effect [207] is a very important finding in the field of speech perception, which shows that when viewers are shown a video dubbed with a different audio to that being utterred, a third utterance is perceived. For example, when shown a video uttering /ga/, dubbed with a sound of /ba/, /da/ is perceived. It was also found that when the subjects are asked to close their eyes, the correct perception of the sound is restored. This experiment has been replicated on various occasions [73, 86, 197] with different audio and visual combinations revealing the multimodality of audio-visual mapping. Thus, on a perceptual level, many visual representations map to a given sound and similarly, many audio configurations of speech map to a given visual representation, depending on the clarity of the visual and audio channels.

2.1.5 Coarticulation

Coarticulation is a physical phenomenon that arises in speech production, where the sounds as well as lips configurations that occur during the utterance of a phoneme are conditioned on the phonemes that occurred before (backward coarticulation) as well as the phonemes that are coming next (forward coarticulation). It arises because the speech articulators need to transition from the current positions to the next configuration and thus there is a blurring at the boundaries of the phonetic units. As an example, consider the utterance of the phoneme /ih/ in "milk" and "sit". In the former case, the *nasal* /m/ phoneme preceeding the /ih/ would cause some lip-rounding during the utterance of /ih/, which in turn has to transition towards the *semi-vowel* /l/ before reaching the *stop* /k/. The shape of the mouth during the utterance of /ih/ is more elongated vertically. In the case of "sit", the /ih/ phoneme is encapsulated between a *fricative* and a *stop*, making the occurence of /ih/ of shorter duration and more elongated horizontally. The visual appearance would differ in the two cases, because of the preceeding and next phonemes that occur, thus making speech production a highly context-bound process. The phenomenon of coarticulation was first brought forward in phonetics by Menzerath and de LacerdaIn [209], as a theory superseding the previous theory of positional sounds, which hypothesised that each phoneme had a "target" place of articulation, which is modified to accomodate the next position, given by the next phoneme. Menzerath and de LacerdaIn [209] used kymographs[1] to study air flow measurements that occur during the production of German labial consonants and vowel sequences. Their findings rejected the view that there were stable articulatory positions that were to be reached, but rather proposed two major principles explaning coarticulation: "Koarticulation" and "Steuerung". According to "Koarticulation", it was found that articulators already prepare for the following sounds during the production of a preceding segment and that this preparation becomes as early as possible. According to the "Steuerung" (steering in English), it was found that vowel articulations were heavily influenced by the following consonants in certain types of syllables. Their conclusion was that each utterance is a complex overlap of simulaneous movements and instead of having fixed articulations with transitions representing coarticulation, *all articulation is coarticulation*.

Hardcastle and Hewlett [133] argue that coarticulation, in addition to being a requirement of the articulatory mechanism, might also arise as a response to facilitate the perception of speech. This observation was based on the fact that coarticulation accounts for the information about a particular segment of speech to extend beyond its boundaries, which gives rise to parallel processing of speech movements, resulting in a

[1] An instrument for recording variations in pressure, as of the blood, or in tension, as of a muscle, by means of a pen or stylus that marks a rotating drum.

faster processing of speech. Due to the phenomenon of coarticulation, the perceiver is able to anticipate what segment of speech is coming next and similarly use coarticulatory effects extending from previous segments to disambiguate what is being said at present.

Anticipatory and Preservatory Coarticulation

The prevailing view about the way speech is produced is based on two principles, namely: *economy* and *plasticity* [190]. The first principle of *economy* states that the speaker adapts their articulation effort to match the perceiver's ability according to the least cost involved and the second principle of *plasticity* states that speech movements are purpose-driven, i.e. articulators can be either hypo- or hyper-articulated depending on the needs of the situation, but both aim at conveying the required meaning to the listener. Accordingly, both *anticipatory* or backward coarticulation and *preservatory* or *carryover* or forward coarticulation serve these two principles with the aim conveying the appropriate meaning using the least cost. Fowler and Saltzman [114] explain the mechanism of anticipatory and carryover coarticulation in terms of gesture coproduction, with the activation of a gesture increasing and decreasing smoothly in time, thus having a bearing on the vocal tract shape and the acoustic signal. This is illustrated in Figure 2.2, which shows the production of three gestures associated with speech production, each delimited by vertical lines. In this case, gesture 2 is predominant but the weaker influence of the following gesture 3 gives rise to anticipatory coarticulation and the weaker influence of the gesture 1 gives rise to carryover coarticulation. The influence of each gesture changes as we move from predominant gesture 2 to gesture 3, with gesture 2 now accounting for carryover coarticulation and the next gesture 4 accounting for anticipatory coarticulation.

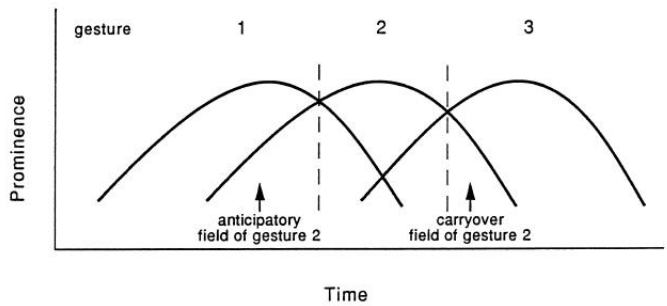


Figure 2.2: Representation of anticipatory and carryover coarticulation as overlapping gestures [114].

Inter-language Differences

Different languages exhibit different properties with respect to coarticulation. For example, English tends to be more *anticipatory* in nature with articulators positioning themselves in anticipation of future phonemes whilst French and Italian tend to be more *preservatory* with articulator positions being more dominated by sounds already produced [175]. Other differences have been found by Öhman [224], who studied coarticulation in Vowel-Consonant-Vowel (VCV) segments. It was found that for V_1CV_2 utterances in English and Swedish, articulators begin moving towards V_2 near the end of V_1, before the consonant boundary is reached. However, this effect was not found in Russian, which was attributed to weak Vowel-to-Vowel coarticulation in that language. It was also found by André-Pierre Benguerel [5] that languages such as French have coarticulatory effects extending to up to seven preceeding phonetic segments in certain sequences and these dynamical effects vary across languages. Hardcastle and Hewlett [133] raised the issue of whether differences in coarticulation between languages are a result of independent properties of the language itself or rather due to derivatives of other properties of the language, which are more generalisable. Lubker and Gay [194] proposed that inter-language differences are explained by language-specific articulatory or phonetic requirements. This conclusion was based on the observation that lip-rounding between a pair of vowels in Swedish is longer and more precise than in American English. This is explained by the fact that languages such as Swedish have a very crowded vowel space, which might result in perceptual confusion among vowels and thus more emphasis is needed to convey the required meaning to the perceiver.

Coarticulation Models

Models of coarticulation are used by speech production theorists to have a representation of speech production that explains the physical and acoustic aspects of speech [133]. These models of coarticulation need to explain both the *temporal* and *spatial* aspects of coarticulation. The former deal with the extent to which coarticulatory effects extend backwards and forwards in time when the articulators are not subject to competing demands from adjacent segments. The latter deal with what happens when articulatory structures are subject to competing articulatory and coarticulatory demands. The different mainstream models of coarticulation are now outlined:

The *look-ahead model*, proposed by Kozhevnikov and Chistovich [172] and Daniloff and Hammarberg [68] tries to explain anticipatory coarticulation in Consonant-Vowel (CV) syllables based on the principle that commands to produce the vowel are issued simultaneously with all the consonants preceding the vowel, thus predicting a high level of coarticulation in CV syllables with little or no coarticulation in other syllables such as Vowel-Consonant (VC). This was in contradication with studies showing high

coarticulatory effects in VC syllables [214].

The *coarticulation resistance model* [24] suggests that coarticulatory effects in Vowel-to-Consonant (VC) syllables experience resistance and tend to decrease gradually and also vary according to the boundary phonemes. These variations in coarticulatory effects can be explained by a rule that assigns coarticulatory resistance depending on the boundary phonemes. They also proposed that the resistance coefficients vary across languages.

The *window model* was proposed by Keating [165] and accounts for spatial and temporal aspects of coarticulation, as well as for differences observed between segments in a given language and across languages. The model specifies a particular range of legal values, called a *window* for each phonetic parameter (articulatory or acoustic), in a given coarticulatory segment. The exact width of a window is derived for each language from information on the maximum amount of contextual variability observed in the speech for that language.

The *coproduction model*, first formulated by Fowler et al. [113] attempts to bridge the gap between the cognitive and physical aspects of language. In particular, this model aims at accounting for the dynamics and kinematics of articulators in speech movements using a *task-dynamical* model. According to the coproduction theory, context-independent gestures translate into both the spatial and temporal aspects of vocal tract constrictions for each phoneme. The gestures can be represented by a score, which determines the movement of the articulators in the vocal tract. The *time-locked model* [15] is a variant of the *coproduction model* and hypothesises that the start of movement of a given articulator is independent of the preceding phone string length but begins at a fixed time before the start of the segment with which it is associated. Thus, the model is able to predict the starting time towards a given articulatory gesture prior to the start of the segment in which it is found, depending on the speaking rate.

The *gestural model* was suggested by Löfqvist [193] and the concept of overlapping dominance functions was used to explain anticipatory and preservatory coarticulation. According to that model, a speech segment has dominance over vocal articulators with a pattern of increase and decrease that overlaps with dominance functions of adjacent segments. Thus, a form of blending is required at the overlaps. This model goes even further to say that each articulator has a separate dominance function for a given segment. This model extends the coproduction model by having explicit dominance functions to model the overlapping gestures. The *gestural model* of coarticulation is able to better explain coarticulatory effects in Vowel-Consonant-Vowel (VCV) segments because independent overlapping commands are issued for the VC segment and the CV segment with greater effects of Vowels on Consonants and both gestures overlapping to produce the VCV segment [107]. Thus, it provides a better approximation to VCV

coarticulation than the *look-ahead model* and the *coarticulation resistance model*. The gestural model [193] of coarticulation was used by Cohen and Massaro [54] for visual speech synthesis.

The *hybrid model* was proposed by Perkell and Cohen [240] as a trade-off between the *look-ahead model* and the *time-locked model* following observations by Bladon and Al-Bamerni [23] that English speakers used both a one-stage opening gesture and a two-stage gesture in coarticulation of the velum. Perkell and Cohen [240] reported similar observations in lip-rounding movements. They thus proposed the *hybrid model* to account for these two phases.

The modelling of coarticulation in synthetic visual speech will be discussed in Section 2.2.4.

2.2 Animating Faces

This section presents a review of techniques used for facial animation. These methods can be categorised as keyframe-based, heuristic-based, data-driven and other variants. A more detailed taxonomy of speech-synchronised facial animation techniques is given in [230].

2.2.1 Manual and Heuristic Facial Animation

Parke [231] was the first to achieve facial animation using keyframes of a 3D geometric model. The face was constructed of polygonal surfaces and was manipulated through the use of parameters which controlled interpolation, translation, rotation and scaling of the various facial features. Animation was achieved by first creating keyframes that represented the different facial configurations or visemes. These facial configurations were created by the manipulation of fewer than 10 parameters. Finally, interpolation between the keyframes resulted in trajectories of parameters that could be mapped to the facial model to create speech-synchronised facial animation.

Instead of using geometrical models of the face for keyframe animation, anatomical models that simulate the bones and muscles of the face have also been used [242, 275, 307, 184, 161, 264, 199]. These models include mass-spring systems to model muscle deformation [242, 161]. Alternatively, a vector approach of facial mesh deformation can be used, where motion fields are used to represent muscle activation [307]. Yet another approach is the layered spring meshes [275, 184], that extend the mass-spring structure to connected mesh layers, thus resulting in a more accurate modelling of facial behaviour. More recently, a physics-based muscle model of the face has been used in conjunction with motion capture data to automatically learn the muscle activation parameters [264]. For these physics-based models of the face, keyframing was used to

interpolate muscle activation parameters to generate facial animation.

Keyframe animation has also been explored with the more recent 3D morphable models (3DMM) [25], which are powerful statistical appearance-based facial models that can generate photorealistic animation [26]. However, they fail to achieve video-realism because they do not explicitly model the inner structures of the mouth and eyes.

Heuristic methods either define a set of rules to model coarticulation [17, 239] or using dominance functions [54, 93]. Edge and Maddock [94] and Lazalde et al. [181] proposed a constraint-based visual speech synthesis method that by using a physically-constrained model of the face and optimising an objective function to generate speech-synchronised facial animation driven by visemes. These methods require a considerable amount of domain knowledge as well as trial and error to get the rules, dominance functions and constraints right. However many and precise the heuristics that are hand-crafted, they at best only approximate the mechanism of speech generation and fail to achieve very realistic results.

2.2.2 Data-driven Facial Animation

Data-driven approaches rely on the idea that the rules governing facial animation can be learnt automatically given several examples of a talking face. Facial and audio data are represented as parameters and Machine Learning techniques can be used to learn a mapping between the audio and visual parameters. This mapping can then be used to predict visual parameters given the audio parameters.

Data-driven approaches can be further categorised in terms of the input used to generate facial animation as well as on the technique used to achieve audio-visual mapping. The input can be either text or speech parameters.

Text-driven facial animation is driven by a text input in a given language and synthesis involves text-to-speech (TTS) synthesis, followed by a mapping of the underlying phonemes to visual speech [95, 191, 301, 244, 208, 192, 205, 158].

Speech-driven facial animation involves mapping continuous speech parameters or the discrete phonemes onto the face. Audio data can be represented using phonemes which are obtained by phonetically aligning audio data to phonemes [317, 150]. Speech-driven facial animation here refers both to phoneme-driven and continuous speech-driven facial animation.

The mainstream approaches to speech-driven facial animation are: sample-based, learning-based and hybrid approaches [77].

Sample-based Methods

Sample-based approaches are mostly phoneme-driven and aim at finding commonly occuring fragments of speech, followed by a search algorithm that finds fragments best matching the target utterance. Sample-based approaches need to solve the problem of finding units of visual speech that are to be reordered, a problem known as *unit selection*. In Bregler et al. [34], the fragments were chosen as triphones, which are groups of three phonemes. Image frames were first labelled using eigenpoint tracking [66]. This was followed by warping each image to a standard reference frame. New animation was achieved by reordering frames from the training corpus to match the target audio. The criterion for reordering the frames was based on a similarity measure between the triphones in novel and training data. Finally, the images were warped from the reference frame to the target shape, aligned to the shape of the synthesis frame and morphed together. Cao et al. [42] used a greedy graph search algorithm to synthesise novel speech animation from motion graphs by matching test audio features with those of motion graphs and using phonetic information to minimise jumps, thus accounting for coarticulation. In Kshirsagar and Magnenat-Thalmann [174], the fragments were syllables which were extracted automatically from a phonetic stream using a syllabification algorithm. Cosatto et al. [62] introduced variable length audio-visual units to visual speech synthesis. The units were found by running Viterbi search through a graph that connects phonetic units with transition links encoding a *concatenation cost* and each node having a *target cost* that measures the similarity of each unit with the unit to be synthesised. Ma et al. [196] used a similar approach where variable length units are computed for a given test utterance using Viterbi search through a trellis that represents the allowed transitions between phonemes as well as between utterances that are not part of the same sentence. A transition is allowed between two different utterances only if the connecting ends belong to the same viseme category. Edge et al. [95] also used variable-length unit selection at the level of phones, syllables, words and sentences using an algorithm that tries to maximise the length of fragments whilst balancing the similarity of phonetic timing and the similarity of contexts with the target utterance. Deng et al. [79, 81] learnt diphone and triphone coarticulation models, which were incorporated into a motion synthesis algorithm that selects the optimal units for generating a new utterance. Tao et al. [270] adopted a hybrid approach where a HMM-based unit selection method was used for visual speech synthesis and Gaussian mixture models (GMMs) were used for synthesis of facial expressions for expressive speech animation. Edge and Hilton [91] and Edge et al. [92] adopted a novel metric for measuring concatenative cost, based on wavelet decomposition, for 3D visual speech synthesis. Melenchón et al. [208] combined visual unit selection with geodesic interpolation to generate smooth facial animation that explicitly models emphasis. Liu

and Ostermann [192] proposed Pareto optimisation [327] for unit selection in concatenative visual speech synthesis. Mattheyses et al. [205] computed both audio and visual target costs and together with concatenation costs between frames, a search was made in the training database to find the best audio and visual segments matching a target phonetic sequence. The advantage of sample-based approaches is that the fidelity of the animations is higher due to the use of original images as compared to images generated from appearance models that can get blurred. However, sample-based methods require storage of the whole visual corpus to synthesise facial animation and are not amenable to adaptation of facial models and speech models for transferable speech animation [47].

Learning-based Methods

Learning-based approaches learn statistical models to model the relationship between visual parameters and either audio parameters or phonemes or both. Gaussian mixture models (GMMs) have been used to synthesise speech animation in [148, 249, 49, 105, 141, 326]. Yamamoto et al. [314] introduced hidden Markov models (HMMs) to visual speech synthesis. The method involved training context-independent audio HMMs with phonemes as states, which were then used to decode test audio using the Viterbi algorithm [293]. This was followed by a table look-up process to convert the HMM states to corresponding lip parameters with a look-ahead mechanism to model forward coarticulation. HMMs have also been used in the works of Brand [32], Choi et al. [50], Lee and Yook [183], Cosker et al. [64], Fu et al. [117], Li and Shum [188], Wang et al. [298], Govokhina et al. [125], Xie and Liu [312], Tao et al. [270], Bailly et al. [10], Wang et al. [303] and Wang et al. [304]. A linear dynamical system (LDS) was used in the works of Saisan et al. [258] and Lehn-Schiøler et al. [186] to jointly model audio and visual parameters. Englebienne et al. [102] used a variant of the switching linear dynamical system (SLDS) [122], called deterministic process dynamical system (DPDS) to model visual data while audio data was modelled using a HMM. Both models were coupled by the phonemes, which represented the states of the HMM as well as the switching states of the SLDS. During synthesis, the discrete phonetic labels need to be inferred from the speech signal followed by the generation of the most likely visual parameters for the phoneme sequence.

We adopt a learning-based approach in this work mainly because it allows for a compact representation of facial data as well as the adaptation of both facial and speech models to different identities [47, 280]. More details on specific learning-based methods applied to visual speech synthesis and related to this thesis are given in Chapters 4 and 5.

Hybrid Methods

More recently, some researchers have tried to combine sample-based and learning-based approaches. Edge et al. [92] first used a sample-based unit selection method to determine the closest unit to each segment in the test sequence. The selected units were then used to train a state-based model for each transition of phonemes in the test sequence. The Viterbi algorithm [293] was used to determine an optimal path through the trained model, which was used to generate a smooth trajectory through the visual features. Wang et al. [303] adopted a different approach where the first step consisted of first using a learning-based approach to train a HMM of lip movements and using the trained model to generate a trajectory of lip movements from speech parameters in the maximum-likelihood (ML) sense. The generated trajectory was used as a guide to select an optimal sequence of mouth images from the training database, which was then integrated with the whole face. This has the advantage that the blurring resulting from having appearance models of the face is not present in the synthesised video, which takes images from the original corpus.

Audio Representation

Several techniques for speech-driven animation first use audio data to infer discrete states and then drive the animation using these discrete state labels [34, 32, 105, 102]. Such approaches, however, discard prosodic information in the speech signal such as speech rate, emphasis and intonation. There are other techniques that use only continuous speech features to predict visual features of a talking head [186, 326, 303, 304] by having a joint probabilistic model of audio and video. These approaches, however, ignore the structure of language that is encoded in a discrete representation of speech. Combining both approaches results in using the full information of audio speech to synthesise visual speech and has been used by Cao et al. [42] and Edge et al. [92] through the use of sample-based visual speech synthesis. Our aim in this work is to do the same using learning-based approaches.

2.2.3 Other Forms of Facial Animation

There are also other variants of facial animation, such as: performance-driven facial animation, facial animation transferring and facial gesture generation [77]. Performance-driven facial animation concerns capturing the facial movements of a human actor and remapping it to a virtual avatar [51, 2, 3, 149, 309]. Facial animation transferring deals with transferring speech movements or facial expressions between facial models [198, 294]. Facial gesture generation aims at synthesising facial expressions on a talking head to reflect different emotive states such as happiness, sadness, anger, surprise, etc.

[52, 43, 81, 297].

2.2.4 Modelling Coarticulation

There are three broad approaches to the modelling of coarticulation in visual speech synthesis: heuristic methods, search methods and generative methods. These are closely related to the techniques for speech-driven facial animation but are presented here to highlight how they model coarticulation.

Heuristic methods include rule-based approaches [17, 239] that define a set of rules to model coarticulation, as well as fitting dominance functions to facial data in order to model the dynamics of segments of speech [54, 93]. The model of Cohen and Massaro [54] was inspired from the *gestural model* [193] of coarticulation discussed in Section 2.1.5.

Search methods aim at reordering frames from the training data to match the test phonetic stream. Search approaches need to identify concatenative units from the training data to match the test sequence. The units can be of fixed length, such as diphones and triphones [34, 79, 81] or they can be of variable length [62, 174, 196].

Generative methods aim at learning a model of the process which generates the data. They are mostly based on GMM [148, 249, 49, 105, 141, 326], HMM [314, 32, 50, 183, 64, 117, 188, 125, 298, 312, 270, 303, 304], LDS [258, 186] and SLDS [102]. However, some of them have an implicit model of coarticulation in the trajectory synthesis algorithm [32, 105, 64, 326]. Lehn-Schiøler et al. [186] and Englebienne et al. [102] have an explicit model of only backward coarticulation because the state vector for the current frame is predicted from that of the previous frame in the synthesis. Govokhina et al. [125] proposed a trajectory formation method to represent the *task-dynamical model* [113] of coarticulation. The gestural score influencing the movement of facial articulators was computed using both context-dependent and context-independent phoneme HMMs. Then, a trajectory formation model was used to execute this gestural score by moving the articulatory parameters shaping the vocal tract. The strength of this method is that it is data-driven, as compared to the heuristic method of Cohen and Massaro [54], which requires manual tuning.

In this work, we adopt a generative approach to modelling coarticulation by explicitly modelling speech dynamics, taking into account both the audio and visual modalities and exploiting structure embedded in natural language.

2.3 The Uncanny Valley

The Uncanny Valley [215] is an important finding that needs to be taken into account when trying to achieve realistic facial animation. It postulates that the response

elicitated in humans increases as the level of realism of the robot or virtual character increases, only up to a point where the artificial nature of the latter is still clearly evident. After that point, there is a "Valley" which is characterised by strong repulsion as the level of realism increases. The repulsion is due to a certain level of "eeriness" that arises when the viewer subconsciously associates the virtual character or robot to a corpse or "zombie". After the Valley, there is a rapid increase in acceptance as the realism gets to the point where the virtual character or robot is indistinguishable from reality. This is illustrated in Figure 2.3. The Uncanny Valley plays a very important role in robotics, particularly *socially interactive robots* because the aim is to build robots that can interact in a life-like manner with humans in order to support the latter in day-to-day tasks [111]. Thus, commercial robot makers need to make sure that their robots do not fall within the Valley, in order for the robots to have a favourable response from users. More recently, Hodgins et al. [138] did several experiments to investigate the factors that contribute to a virtual character eliciting an Uncanny Valley response by investigating the emotional response of participants in the presence of facial anomalies. They found that very slight facial anomalies cause an unfavourable emotional response, thus hinting that they fall into the Uncanny Valley. Tinwell et al. [283] found that lack of facial expressions in the upper parts of the face during speech accentuates the Uncanny Valley effect due to a face that looks emotionless. However, it must be pointed out that although there is strong scientific plausibility for the Uncanny Valley, there is yet much more to be done to understand the effect experimentally [230].

Several researchers have attempted to build socially acceptable robots or facial animation that cross the Valley. Experiments done by Hanson et al. [132] demonstrate very favourable response for a humanoid robot that is able to simulate human-like facial expressions and is equipped with very sophisticated machine perception technologies and Natural Language Processing. Recently, the *Emily Project* has attempted to develop a performance-driven facial animation method that claims to push the animation quality beyond the Uncanny Valley [3]. The facial model was acquired from an actress by using a high-resolution digital scanner, where several stereo photographs of the face under different lighting conditions were used to capture the face's geometry and reflectance. In addition, a plaster cast was used to model the teeth of the actress, which was incorporated into the model. This created a very detailed model of the face that was able to model various subtleties in facial expressions.

The effect of the Uncanny Valley on speech-driven facial animation has not been investigated. The main reason is the lack of expressiveness in image or appearance-based speech animation. In this thesis, we aim to investigate the Uncanny Valley effect in speech animation synthesised using our method. More details of these experiments are given in Chapter 6.

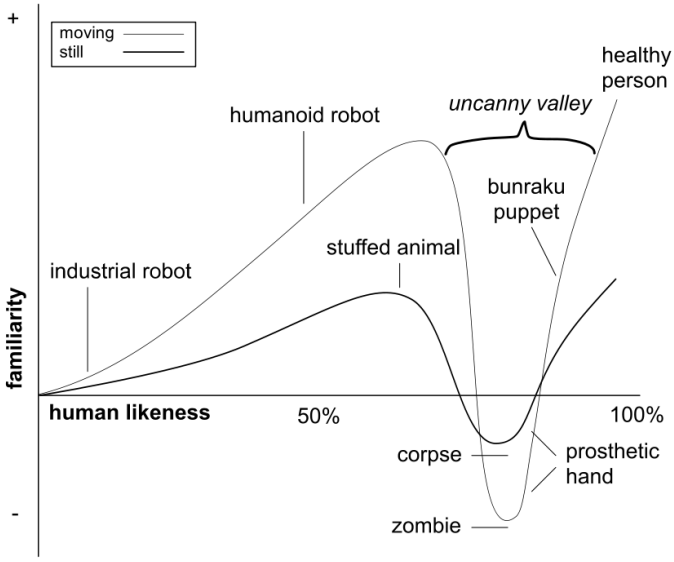


Figure 2.3: The Uncanny Valley [215].

2.4 Applications of Visual Speech Synthesis

Visual speech synthesis has many applications in the fields of: cinema, gaming, Human-Computer Interaction (HCI), speech therapy, medicine as well as internet and communication. These areas will now be covered briefly.

2.4.1 Cinema

Facial animation has been an important feature of computer-generated imagery (CGI) films over the past two decades. The first full-feature CGI movie, *Toy Story* had facial animation generated by modelling each muscle of the human face [276] without trying to achieve high levels of realism. Later movies such as *Matrix Reloaded* used photogrammetry [212] to acquire a dense shape model of the face in order to attain a high level of realism [28]. The deformation of the face was achieved by using optical flow [35]. Other CGI movies such as Final Fantasy: The Spirits Within (2001), Final Fantasy VII Advent Children (2005) and The Polar Express (2004) tried to achieve a very high level of realism by using motion capture technology and a very detailed modelling of the face and hair. More recently, in the 2010 movie *Avatar*, animation was achieved by using motion capture techniques through the placement of markers on the face and mapping the animation to a synthetic character using performance-driven facial animation [187]. As speech-driven facial animation techniques mature, character animation in cinema could be more speech-driven, thus reducing the need for motion capture of each utterance in the movie. Applications that could result from this include automatic movie dubbing as well as animating characters that are no longer alive, such

as in the movie *Forest Gump* [213].

Recently, the Manchester-based company, *ImageMetrics*[2] has been working on the Emily Project [2, 3], which uses performance-driven facial animation to create facial animation which looks highly realistic. The facial models were acquired using stereo cameras. *ImageMetrics* has provided facial animation solutions for various movies such as *Harry Potter*.

2.4.2 Gaming

Facial animation is widely used in the games industry to create lifelike avatars that either engage in conversations as part of the game or generate visual speech dynamically based on the gameplay. A possible application of the latter is in sports games that generate live commentary such as *FIFA* or *Need for Speed* by *EA Games*. Games that generate live audio commentaries typically use concatenative speech synthesis, where segments of speech from a corpus are extracted, concatenated and blended to generate the desired utterance. The same approach could be adopted for visual speech synthesis, where videos of pre-recorded utterances are recorded and then blended to generate a seamless animation based on the audio commentary that is being generated.

Various approaches for facial animation can be applied to gaming, such as: performance animation, keyframing, physics-based animation as well as other methods such as scripting and procedural animation [167]. Recently, *ImageMetrics* has been providing performance-driven facial animation solutions to various games such as *Grand Theft Auto* [252]. In addition, the Edinburg-based company *Speech Graphics*[3] has been offering speech-driven facial animation solutions for the games industry.

2.4.3 Human-Computer Interaction

One of the aims of HCI is to provide an intuitive and natural way of interacting with a computer. Humans are trained from birth to interact with the human face and having an equivalent on the computer can help novices learn computing much more easily. Moreover, facial animation is deemed to become an important component of virtual environments and immersive 3D virtual worlds. In this respect, various researchers have investigated the use of speech animation methods to virtual worlds with the key requirement being real-time synthesis [159, 269]. A further requirement of speech animation for HCI is expressiveness. The animation needs to convey emotions through facial expressions in order to provide a natural interface. Figures 2.4 and 2.5 illustrates expressive facial animation achieved by various researchers.

[2] http://www.image-metrics.com/
[3] http://www.speech-graphics.com/

Figure 2.4: Expressive facial animation [269].

Figure 2.5: Expressive facial animation [43].

Researchers at Curtin University have developed the Virtual Human Markup Language (VHML) as a markup language for artificial talking heads [203]. This language has been applied to the creation of avatars by Carretero et al. [44] and has been verified and validated by Gustavsson et al. [128].

A promising application of facial animation is to create novel HCI based on characters or agents [230]. Such agents can assist in simple computer tasks such as helping to navigate the internet and, ultimately, should be able to speak to the user, behave in real time and become social user interfaces that would supplement graphical user interfaces (GUI). In 2011, IBM unveiled its intelligent computer called Watson [109] that has natural language processing abilities and could take questions from humans and respond in a human-like manner. It also won the Jeopardy competition, beating human participants[4]. Augmenting such an agent with facial animation could greatly enhance intelligent natural language interfaces.

2.4.4 Speech Therapy

Facial animation technologies and in particular speech animation can be applied to speech therapy for helping speech-impaired people to learn how to pronounce words properly and speak. The *Speak As You See (SAYS)*[5] is a software product that uses 3D computer graphics to create realistic animation in order to help people with speech

[4] http://www.ibmwatson.com
[5] http://www.learningtechnologiesinternational.com/product.html

disabilities. It allows the user to focus on the level of communication that is to be emphasised to the patient. For instance, the user can be shown only specific articulators, such as the tongue or both the mouth and tongue, moving in a particular utterance. Such levels of customisation have been found to be very beneficial in speech therapy [121].

2.4.5 Medicine

Applications of facial animation in medicine have been made in the field of craniofacial surgery [123, 305]. In particular, by using facial models, assistance can be provided to craniofacial surgical planning and facial tissue surgical simulation ahead of the surgery. Such applications require a muscle and tissue model of the face, which the surgeons can work on to plan their surgery. Following that, the face model can be animated to see if the articulations are correct for the new face structure. Speech animation can help to automate the tests by providing an audio corpus in order for validation to be done by comparing it against what the animation should look like. Such models have already been used to validate the facial appearance of the resulting face against expected results in view of surgical planning [166].

2.4.6 Internet and Communication

The prevalence of multimedia content on the internet has seen an unprecedented growth in recent years. Increasingly, the web is becoming a platform for authoring new multimedia content and for seamlessly integrating television, music and video together with user-generated content such as comments and messages on wikis and social networking websites such as *Facebook* and *Twitter*. In such a dynamic environment, it is very easy for users to be overwhelmed by information and the need for users to find their way through massive amounts of information becomes paramount. In this respect, an interface that integrates facial animation with text-to-speech generation could be of great help. One example of the integration of facial animation with the web browsing experience has been achieved with the TNT facial animation system [191]. This system provides a plug-in with the web browser that converts all the text to speech and maps the speech to a talking face that leads the user through the web page in an intuitive way. The system is also provided in a standalone program that works both with input text and speech, as illustrated in Figure 2.6.

Speech animation can be used in online telephony systems such as *Skype*, where the incoming audio is mapped to facial animation for low-bandwidth communication. The software *CrazyTalk*[6] allows a user to take a single photo of themselves and from automatically detected landmark points builds a facial model. It then allows them

[6] http://www.reallusion.com/crazytalk/

Figure 2.6: The TNT facial animation system [191].

to either enter a text or a recorded voice sentence as input and maps the speech to the facial model to generate synthetic visual speech. A plugin of the software for *Skype* already exists that allows the animation of a single photograph of the person talking from incoming speech audio. A system of this kind would be particularly useful in translingual communication, where the speaker at the incoming end speaks in a language that is translated and communicated in a different language on the receiving end. Faruquie et al. [108] proposed a framework for achieving translingual facial animation using a speech recognition unit to extract phonetic information from the translated speech and mapping it to facial expressions.

2.5 Chapter Summary

We have presented an in-depth description of the different aspects of human speech. The phenomenon of coarticulation has also been dealt with as well as mainstream models of coarticulation in speech production theories. A review of techniques to synthesise visual speech from audio speech was then presented followed by techniques to model coarticulation in visual speech synthesis. Finally, we discussed the Uncanny Valley effect and covered some applications of visual speech synthesis.

Chapter 3

Data Acquisition and Processing

> Measure what is measurable, and
> make measurable what is not so.
>
> Galileo Galilei

A data-driven approach to visual speech synthesis requires a database of audio-visual recordings of a talking face that captures the different phonetic combinations in the language being used. The data needs to be processed both in the audio and visual domains in order to obtain a representation that is suitable for use with Machine Learning algorithms. We begin by reviewing techniques for facial parameterisation and speech feature extraction. The data corpora used in this work are then presented, followed by details of visual and audio processing. There is usually a mismatch between the audio and visual frame rates due to the requirement of an auditory window in which the speech signal is stationary. This results in a higher frequency of audio as compared to visual parameterisation. We investigate different methods to synchronise audio and visual speech parameters. Moreover, both the audio and visual features should be normalised to retain only speech-related content. Specifically the following should be normalised in our data: lighting and pose variations in the visual domain, variations in the audio data capture such as the distance between the speaker and the microphone and expressive cues in both the audio and visual domains. This requirement arises particularly in our case where audio and visual parameters are jointly modelled, because non-speech related content will affect the predictive ability of the joint models that couple audio and video. We thus deal with various techniques for audio and visual normalisation.

3.1 Modelling Faces

In order to synthesise visual speech, a way of representing faces as a compact set of parameters is needed. This can be done by using appearance models which can be used both for analysis and synthesis of faces. In this work, we restrict ourselves to 2D appearance models, which have been shown to generate more realistic results [104, 278, 63, 47, 101] than 3D appearance models, which require high quality 3D data from laser scans [25, 320, 316] or photogrammetry [2, 3]. These methods need to separately model the inner details of the mouth [26] or alternatively use texture maps, which require accurate dense correspondences between the 3D face shape and texture across frames, in order to avoid blurring and distortion when the texture is mapped to the face mesh [321]. Others using 3D appearance models discard the modelling of inner mouth details [218] with detrimental effects on the levels of realism of the final animation. A description of the main appearance models and their variants is now presented.

3.1.1 Flexible Models in Computer Vision

Appearance models need to be trained on example images and they parameterise the face by projecting the shape and texture of the images to a lower-dimensional subspace, defined by principal component analysis (PCA). They should allow accurate description of different facial configurations (analysis), as well as allow synthesis of novel faces by extrapolating on examples given in the training set. Active appearance models (AAMs) [59, 60] are statistical models of both shape and texture, thus providing a powerful representation for the analysis and synthesis of faces. Before AAMs, shape and texture models were developed separately. Kass et al. [164] introduced the idea of *flexible models* in Computer Vision called active contour models (ACMs) or "snakes" that could be used to snap onto nearby edges through an energy-minimising spline function. The seminal work on statistical models of shape was by Cootes et al. [57], where PCA was applied to inter-point distances between annotations and provided a compact parametric description of shape variability. This was extended to the point distribution model (PDM) [271], where the variability in shape was represented by a mean shape and modes of variation given by the standard deviations in each PCA dimension of the shape parameters. Active shape models (ASMs) [58], also known as "smart snakes" were then introduced, mirroring the idea of ACMs for tracking landmarks on novel images of the same object category, with the difference that the PDM was used as a constraint to restrict the shape range.

Similar statistical models of texture were also independently developed. Sirovich and Kirby [265] and Kirby and Sirovich [168] were among the first to apply PCA to find

a lower-dimensional representation of face images. Craw and Cameron [67] extended this idea by warping faces to a reference shape before applying PCA, thus giving a shape-free texture representation. This led to Turk and Pentland [288] developing the *Eigenface* model, an extension of the model of Kirby and Sirovich [168], which was applied to facial classificiation.

Cootes et al. [59] combined the modelling of both shape and texture using PCA, leading to the active appearance model (AAM). We first present an overview of PCA, which is used in most of the appearance models discussed in this chapter.

3.1.2 Principal Component Analysis

PCA [236] is a widely used technique in statistics, science and engineering to find a lower dimensional representation of high dimensional data. The way PCA works is by decorrelating a set of N correlated variables. This is done by rotating the axes of the data in order to provide better alignment with the modes of variation of the data. In addition, the axes are reordered by decreasing significance so that the first axis represents the direction of highest variance and each next axis is orthonormal to the previous and corresponds to smaller variance than the previous axis but larger than the next axis.

Given some data $\mathbf{X} = \{\mathbf{x}_n\}_{n=1}^N$, PCA finds a linear basis $\mathbf{P} \in \mathbb{R}^{D\times d}$ that can be used to project a given data point $\mathbf{x}_i$ to PCA parameters $\mathbf{b}_i$ according to Eqn.3.1, where $\bar{\mathbf{x}}$ is the mean vector given by $\bar{\mathbf{x}} = \frac{\sum_{i=1}^N \mathbf{x}_i}{N}$. Thus, the D-dimensional data $\mathbf{X}$ is projected onto a lower-dimensional subspace $\mathbf{B} = \{\mathbf{b}_n\}_{n=1}^N$ of dimension d, where $d < D$. Reconstruction of the original data from PCA coefficients is done according to Eqn.3.2.

$$\mathbf{b}_i = \mathbf{P}^T(\mathbf{x}_i - \bar{\mathbf{x}}) \tag{3.1}$$

$$\mathbf{x}_i = \bar{\mathbf{x}} + \mathbf{P}\mathbf{b}_i \tag{3.2}$$

The basis $\mathbf{P}$ is obtained by first computing the covariance matrix $\mathbf{C}$ according to Eqns.3.3 and 3.4, where $\mathbf{Z} = \mathbf{X} - \mathbb{E}[\mathbf{X}]$.

$$\mathbf{C} = \frac{1}{N}\sum_{i=1}^N (\mathbf{x}_i - \bar{\mathbf{x}})^T(\mathbf{x}_i - \bar{\mathbf{x}}) \tag{3.3}$$

$$= \frac{1}{N}\mathbf{Z}\mathbf{Z}^T \tag{3.4}$$

As $\mathbf{C}$ is square and symmetric, it can be decomposed using Singular Value Decomposition (SVD) to obtain its eigenvalues $\mathbf{\Lambda} = [\lambda_1, \ldots, \lambda_D]^T$ and corresponding eigenvectors $\mathbf{P} = [\mathbf{p}_1, \ldots, \mathbf{p}_D]^T$ according to Eqn.3.5, thus solving the eigenvector equation 3.6.

$$\mathbf{C} = \mathbf{P}\mathbf{\Lambda}\mathbf{P}^T \tag{3.5}$$

$$\mathbf{Z}\mathbf{Z}^T\mathbf{p}_i = \lambda_i\mathbf{p}_i \tag{3.6}$$

The eigenvectors are ordered according to decreasing order of eigenvalues, $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_D$ such that the first principal component corresponds to the highest variation in the data and the last principal component the least variation. Out of the D axes of variation of the data, only a small number d would capture most of the variance in the data and the others would be axes of lower variation that can sometimes be attributed to noise. If we want to retain a certain percentage p of the variance of the data, d components can be retained such that: $\sum_{i=1}^{d}\lambda_i \geq \frac{p}{100}\sum_{i=1}^{D}\lambda_i$, thus giving the retained eigenvectors as $\mathbf{P} = [\mathbf{p}_1, \ldots, \mathbf{p}_d]^T$. These eigenvectors, also known as the principal component coefficients or loadings, can be used to obtain a lower dimensional representation of the data according to Eqn.3.1.

PCA for Images

Images are high dimensional with a 100×100 RGB colour image having $D = 100 \times 100 \times 3 = 30,000$ dimensions. Computing a covariance matrix on such high dimensional data is usually intractable. This problem arises both in the Eigenfaces [288] model and AAMs [60] where $N < D$ training images are used to find the principal components of texture variation. An elegant solution to this problem comes from the observation that the rank of the covariance matrix is limited by the number of training examples N. This entails that eigenvector decomposition can be computed on a $N \times N$ matrix $\mathbf{S}$ given by Eqn.3.7.

$$\mathbf{S} = \frac{1}{N}\sum_{i=1}^{N}(\mathbf{x}_i - \bar{\mathbf{x}})(\mathbf{x}_i - \bar{\mathbf{x}})^T \tag{3.7}$$

$$= \frac{1}{N}\mathbf{Z}^T\mathbf{Z} \tag{3.8}$$

The eigenvector equation then becomes:

$$\frac{1}{N}\mathbf{Z}^T\mathbf{Z}\mathbf{p}_i = \lambda_i\mathbf{p}_i \tag{3.9}$$

Pre-multiplying both sides by $\mathbf{Z}$ gives the following, where $\mathbf{v}_i = \mathbf{Z}\mathbf{p}_i$.

$$\frac{1}{N}\mathbf{Z}\mathbf{Z}^T\mathbf{v}_i = \lambda_i\mathbf{v}_i \tag{3.10}$$

In order to determine the eigenvectors, both sides of Eqn.3.10 need to be further pre-multiplied by $\mathbf{Z}^T$ to give:

$$(\frac{1}{N}\mathbf{Z}^T\mathbf{Z})(\mathbf{Z}^T\mathbf{v}_i) = \lambda_i(\mathbf{Z}^T\mathbf{v}_i) \tag{3.11}$$

This implies that $\mathbf{Z}^T\mathbf{v}_i$ is an eigenvector of $\mathbf{S}$ and the corresponding eigenvalue is λ_i. These eigenvectors, however need to be normalised in order to become orthonormal

according to:

$$\mathbf{p}_i = \frac{1}{(N\lambda_i)^{1/2}}\mathbf{Z}^T\mathbf{v}_i \tag{3.12}$$

Thus, the approach mentioned above first needs to perform an eigendecomposition of $N^{-1}\mathbf{Z}\mathbf{Z}^T$, before computing the eigenvectors in the original data space according to Eqn.3.12.

3.1.3 Active Appearance Model

The Active Appearance Model [59, 96, 60] provides an integrated framework for modelling both the shape and texture using PCA, as well as an algorithm for tracking facial landmarks on novel faces, after a training phase, where a linear mapping is learnt between shape displacements and texture differences. The term Active Appearance Model is mostly used in reference to the tracking algorithm, although it may also be used for the combined shape and texture model. The AAM allows any face to be represented using a compact set of parameters, which can be used to regenerate the original face.

AAM Training

In order to build an AAM, a set of images is required, which are manually annotated with landmark points, as shown in Figure 3.1a. The landmarks for each image need to be aligned to the mean shape using Generalised Procrusted Analysis (GPA) [126] and concatenated into a single vector $\mathbf{s}$. The texture for each image, $\mathbf{g}_{im}$, which is either gray-scale pixel intensities or RGB colour values, needs to be warped to the mean shape to give a shape-free image. The warping is done by performing a piecewise affine warp, where the shape is decomposed into triangles using Delaunay triangulation [74], as shown in Figure 3.1b. Each triangle in the original shape is then warped to the corresponding triangle in the mean shape. The texture needs to be normalised in order to minimise the effect of global lighting variation. This is done by mean-centering around zero and setting the Euclidean norm to one according to Eqn.3.13, where $\boldsymbol{\mu}_{im}$ is the mean of the vector $\mathbf{g}_{im}$ and $\mathbf{1}$ is a vector of ones.

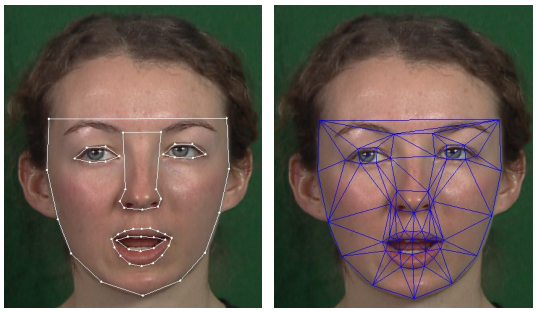
$$\mathbf{g} = \frac{\mathbf{g}_{im} - \boldsymbol{\mu}_{im}\mathbf{1}}{\| \mathbf{g}_{im} - \boldsymbol{\mu}_{im}\mathbf{1} \|} \tag{3.13}$$

The shape and texture are then idependently projected to PCA parameters according to:

$$\mathbf{b}_s = \mathbf{P}_s^T(\mathbf{s} - \bar{\mathbf{s}}) \tag{3.14}$$

$$\mathbf{b}_g = \mathbf{P}_g^T(\mathbf{g} - \bar{\mathbf{g}}) \tag{3.15}$$

where $\mathbf{b}_s$ are the shape parameters, $\mathbf{b}_g$ are the texture parameters, $\bar{\mathbf{s}}$ is the mean



(a) (b)

Figure 3.1: (a) Markup points of facial landmarks. (b) Delaunay triangulation of landmarks.

shape, $\bar{\mathbf{g}}$ is the mean texture, $\mathbf{P}_s$ are the shape eigenvectors and $\mathbf{P}_g$ are the texture eigenvectors.

The shape parameters $\mathbf{b}_s$ and texture parameters $\mathbf{b}_g$ are concatenated to form a single vector $\mathbf{b}$ according to Eqn.3.16, where $\mathbf{W}_s$ is a diagonal matrix of weights for each shape parameter, and is used to account for the difference in units between the shape parameters, $\mathbf{b}_s$ and the texture parameters, $\mathbf{b}_g$.

$$\mathbf{b} = \begin{bmatrix} \mathbf{W}_s \mathbf{b}_s \\ \mathbf{b}_g \end{bmatrix} = \begin{bmatrix} \mathbf{W}_s \mathbf{P}_s^T (\mathbf{s} - \bar{\mathbf{s}}) \\ \mathbf{P}_g^T (\mathbf{g} - \bar{\mathbf{g}}) \end{bmatrix} \tag{3.16}$$

Finally, PCA is applied to the combined parameters to give the AAM parameters $\mathbf{c}$ according to:

$$\mathbf{c} = \mathbf{P}_c^T \mathbf{b} \tag{3.17}$$

AAM Synthesis

In order to generate a face image, the shape $\mathbf{s}$ and texture $\mathbf{g}$ are reconstructed from the AAM parameters $\mathbf{c}$ as follows:

$$\mathbf{s} = \bar{\mathbf{s}} + \mathbf{P}_s \mathbf{W}_s^{-1} \mathbf{P}_{c,s} \mathbf{c}, \quad \mathbf{g} = \bar{\mathbf{g}} + \mathbf{P}_g \mathbf{P}_{c,g} \mathbf{c}, \quad \mathbf{P}_c = \begin{bmatrix} \mathbf{P}_{c,s} \\ \mathbf{P}_{c,g} \end{bmatrix} \tag{3.18}$$

The texture is then "unnormalised" by making its mean and Euclidean norm equal to the average of these two values for all the images before normalisation. Finally, the texture is warped from the mean shape to the reconstructed shape and rendered on an image.

AAM Search

The Active Appearance Model algorithm [59] is able to fit the AAM to novel facial images after performing linear regression between the displacement in AAM parameters,

$\delta\mathbf{c}$, and the corresponding displacement in texture, $\delta\mathbf{g}$, in order to obtain a linear regression matrix $\mathbf{R}$, which maps texture differences to differences in AAM parameters:

$$\delta\mathbf{c} = \mathbf{R}\delta\mathbf{g} \tag{3.19}$$

In addition to perturbations in the AAM parameters, small displacements in 2D position, scale and orientation are also modelled and included in the regression [59]. However, in order to keep the notation simple, they are regarded as extra elements of the vector $\delta\mathbf{c}$.

The search algorithm works by refining the AAM parameters until the texture reconstructed from the AAM parameters is statistically similar to the original texture of the image. At each iteration, a step is made towards the optimal AAM parameters, by exploiting the direction of convergence given by the linear matrix $\mathbf{R}$. The tracking algorithm is given in Algorithm 1. Once the optimal AAM parameters, $\mathbf{c}_{opt}$, are found, the tracked shape can be obtained according to Eqn.3.18.

Algorithm 1 Active appearance model search

Input: Image I
Output: Optimal AAM parameters $\mathbf{c}_{opt}$
Initialise $\mathbf{c}_{opt}$ by scanning through I using a rectangular window
Set $E_0 \leftarrow \infty$
Set $E_1 \leftarrow \infty$
Set ϵ to an infinitesimal value
while $|E_1 - E_0| > \epsilon$ **do**
  Obtain the normalised texture, $\mathbf{g}_s$ at the current estimate $\mathbf{c}_{opt}$
  Evaluate the error vector $\delta\mathbf{g}_0 = \mathbf{g_s} - \mathbf{g_m}$
  Evaluate the current error $E_0 = |\delta\mathbf{g}_0|^2$
  Compute the predicted displacement, $\delta\mathbf{c} = \mathbf{R}\delta\mathbf{g}_0$
  Set $k \leftarrow 1$
  Set $\mathbf{c}_1 \leftarrow \mathbf{c}_0 - k\delta\mathbf{c}$
  Sample the normalised texture at this new prediction and calculate the new error vector, $\delta\mathbf{g}_1$
  Evaluate the new error $E_1 = |\delta\mathbf{g}_1|^2$
  if $E_1 < E_0$ **then**
    Accept the new estimate $\mathbf{c}_1$ and Set $\mathbf{c}_{opt} \leftarrow \mathbf{c}_1$
  else
    Try at $k = 1.5, k = 0.5, k = 0.25$
  end if
end while

The original AAM formulation [59] for learning the regression matrix is not very computationally efficient. In Cootes et al. [60], a more efficient approach is derived by taking a first-order Taylor expansion of the difference texture vector, $\delta\mathbf{g}(\mathbf{c})$ at AAM parameter $\mathbf{c}$.

$$\mathbf{g}(\mathbf{c}^* + \delta\mathbf{c}) = \mathbf{g}(\mathbf{c}^*) + \frac{\partial \mathbf{g}(\mathbf{c}^*)}{\partial \mathbf{c}} \partial \mathbf{c} \tag{3.20}$$

Computation of the Jacobian matrix $\frac{\partial \mathbf{g}(\mathbf{c}^*)}{\delta \mathbf{c}}$ is less computationally expensive than performing linear regression. The matrix $\mathbf{R}$ is then computed from the Jacobian matrix as follows:

$$\mathbf{R} = \left(\frac{\partial \mathbf{g}^T}{\partial \mathbf{c}} \frac{\partial \mathbf{g}}{\partial \mathbf{c}} \right)^{-1} \frac{\partial \mathbf{g}^T}{\partial \mathbf{c}} \tag{3.21}$$

Further efficiency gains in both training and fitting have been achieved using the inverse-compositional image alignment (ICIA) algorithm [11, 204] and the compositional gradient descent (CODE) algorithm [4]. However, because efficiency was not the primary focus of the AAM parameterisation in our work, the AAM model used in this thesis made use of the first-order approximation through the Jacobian matrix.

3.1.4 Multidimensional Morphable Model

The multidimensional morphable model (MMM) was proposed by Jones and Poggio [160] and used in visual speech synthesis in [105]. Instead of using a *vector space* representation of texture as in the AAM, the MMM uses a *morph space*, where pixel flow and pixel appearance are used to represent shape and texture. In his PhD thesis, Ezzat [104] argues that a morph space representation achieves a higher level of realism than a vector space representation because the modelling of pixel flow ensures smooth and realistic mouth transitions between mouth configurations.

As opposed to the AAM that requires manual annotations of facial landmarks, the MMM uses optical flow to compute a dense correspondence between an input image and a reference image, which is used to compute perspective warp parameters that are then used to register the input image to the reference. This is based on the assumption that head movements are kept minimal between frames. After normalisation, a set of prototype images are chosen by clustering PCA parameters of all the images from the corpus. This is followed by computing the optical flow vectors that morph the reference image to each prototype image. The shape component for an input image is given by a linear combination of those computed optical flow vectors that best approximate the correspondence map between the reference image and the input image. For the texture component, the optical flow vectors that transform each prototype image into the input image are computed. This is followed by warping each prototype image to the input image, based on these computed optical flow vectors. The texture parameters for the input image are then given by the linear combination of the warped images that best generate the input image. In practice, obtaining the shape and texture parameters for a given input image is an optimisation problem that can be solved using gradient descent. However, Ezzat et al. [105] proposed a solution where the shape parameters

are solved in closed-form. This approach uses *flow concatenation*, where the shortest path in a connected graph of training images is computed from the reference image to each prototype image, using Dijkstra's algorithm [85], followed by a concatenation of optical flow vectors along this shortest path. This procedure leads to an analytic solution for the computation of the shape parameters for an input image. Finding the texture parameters is then reduced to a constrained optimisation problem that can be solved using quadratic programming [29].

The main disadvantages of the MMM are that: 1) it cannot handle large pose variations and 2) warping using the correspondence maps leads to holes in the target image. The latter problem can be solved by using a "hole filling" procedure that is based on linear interpolation between neighbouring pixels [105].

3.1.5 3D Morphable Model

The 3D morphable model (3DMM), proposed by Blanz and Vetter [25] is a 3D statistical model of the face. The requirement for building such a model is to have high quality 3D laser scans of faces. Blanz and Vetter [25] use *Cyberware*^{TM} scans which provide both geometric (shape) and texture data. This data is mapped to a lower dimensional manifold using PCA to give $\{\alpha, \beta, \rho\}_{k=1}^{N}$ for N laser scans, where α are the shape parameters, β are the texture parameters and ρ represents the camera position. In Blanz and Vetter [25], 200 heads of young adults, consisting of 100 men and 100 women were used to build the 3DMM. One requirement on this database of scans is that dense correspondence exists between them, i.e. the shape vectors need to be aligned. This alignment is achieved using 3D optical flow.

The 3DMM can be used to match both new 2D facial images as well as new 3D scans of people not seen in the database. For matching 2D images, an *analysis-by-synthesis* approach is used, whereby the parameters of the model are varied iteratively and rendered into a 2D image, until the error between the generated image and the input image is minimised. This is, however, an ill-posed problem because multiple possible matches might exist for one facial image due to occlusions. A probabilistic formulation was instead adopted for fitting the 3D morphable model to an image, based on stochastic gradient descent, that helps to avoid local minima [25].

The Basel Face Model [235] is a database of faces that have been fitted using the 3DMM, which can be used for face recognition. Figure 3.2 shows a face reconstructed from a sample of the Basel Face Model 3DMM parameters and rendered into an image.

3.1.6 Discussion

Three approaches to modelling faces have been presented. The AAM [60] is a 2D statistical model of shape and texture that uses a vector-space representation obtained

Figure 3.2: Face reconstructed from the Basel Face Model [235].

using PCA. The AAM requires hand-annotated images for training the model and can handle variations in pose. The MMM [160] provides a morph-space representation of faces in 2D using optical flow. No hand-annotated images are required but the MMM cannot handle large pose variations. Moreover, synthesis of novel images can lead to holes that need to be filled [105]. The 3DMM is a statistical model of 3D shape and texture that requires a database of 3D laser-scan data for training. It provides a combined vector and morph-space representation [25] of faces. However, it requires dense correspondence between the training facial data which is achieved using optical flow. Moreover, the 3DMM is not ideal for videorealism because the inner structures of the mouth have to be separately modelled, which compromises the level of realism achieved during facial animation [26]. In this work, we have used the AAM for facial parameterisation because our data has pose variations and also because our goal is to achieve both photorealism and videorealism.

3.2 Speech Processing

In this work, we jointly model audio and visual features. Speech thus needs to be represented as a set of continuous parameters that are aligned to the visual frames. Four speech parameterisation techniques have been considered in this work, namely: linear predictive coding (LPC), line spectral frequencies (LSF), Mel-frequency cepstral coefficients (MFCC) and relative spectral-perceptual linear prediction (RASTA-PLP). In this section, we describe the Fourier transform, which is used in many of the speech processing techniques, as well as each of the four speech parameterisations.

There are two main models of speech production that are used for parameterising speech. The *source-filter* model [151] of speech production attempts to accurately model how the speech signal is produced by the speech production system. Such models, which include LPC and LSF, aim at being able to reconstruct the original speech accurately and are thus well suited for speech coding. They model speech as being produced by a source or excitation, which is modulated by a filter to yield audible

speech. The *perceptually-motivated* methods, which include MFCC and RASTA-PLP, aim to accurately model how speech is perceived by the speech perception system and are thus better suited for discrimation between phonemes. The excitation component is usually discarded in the latter category of methods and thus the parameterisation is more compact. However, the excitation component or fundamental frequency can be retained and used together with perceptually-motivated speech parameters in text-to-speech synthesis systems [324, 272].

Speech is digitised at a given sampling rate, f_s, which consists of measuring the continuous signal at every $1/f_s$ seconds. The Nyquist-Shannon sampling theorem [220, 261] states that in order for a bandlimited signal[1] to be reconstructed accurately from the sampled version, the sampling rate should be more than twice the maximum frequency.

For speech parameterisation, a sliding window is used to represent the part of the signal that is taken into account for analysis at each time point. However, using a rectangular window of time t_w, also known as a Dirichlet window [246], will introduce spurious high-frequency components at the window edges. This can be overcome by having a soft window boundary, such as the Hamming window [131], which is based on the cosine function and thus attenuates the discontinuities at the edges. In addition, the window used is typically overlapping so as to capture dynamical properties of speech. The *window size* refers to the size of the auditory window used at any time point whilst the *hop size* refers to the length of the overlap when moving from one time point to the next. Speech coding and feature extraction methods typically require a window size of $10 - 25$ ms, where the speech signal is stationary. In this short time period, the speech signal can be regarded as a stationary process, i.e. the joint distribution of a sequence of measurements $x_{1+l}, x_{2+l}, \ldots, x_{k+l}$ is the same for all k, independent of the lag l. In order to statisfy this requirement, speech is typically parameterised with a window of 25 ms with overlaps of 10 ms between the windows, resulting in a frequency of 100 Hz.

In speech coding and feature extraction methods, a normalisation step called *pre-emphasis* [151] is usually necessary in order to obtain an equal distribution of the signal energy across the frequency spectrum. The pre-emphasis step results in raising the intensity of the signal proportional to its frequency, thus emphasising higher frequency components, which tend to have lower intensities as compared to their low frequency counterparts. This is done because the spectrum for voiced segments has more energy at lower frequencies than higher frequencies. Boosting of high-frequency energy gives more information to the acoustic model and thus improves phoneme discrimination ability.

The pre-emphasis is done by applying a first-order difference filter as given by Eqn.3.22.

[1] A signal which has a maximum frequency

$$\mathbf{x}'_t = \mathbf{x}_t - k\mathbf{x}_{t-1} \tag{3.22}$$

where $\mathbf{x}_t$ is the speech sample at time index t, $\mathbf{x}'_t$ is the first-order derivative and k is a value between 0.9 and 1, which controls the strength of the pre-emphasis.

3.2.1 The Fourier transform

The Fourier transform [112] is used in the *perceptually-motivated* methods of speech parameterisation and is thus described here.

The Fourier transform transforms a given signal from the time domain to the frequency domain. The Fourier transform of an integrable function $f : \mathbb{R} \to \mathbb{C}$ is given by Eqn.3.23, where z represents the frequency components.

$$F(z) = \int_{-\infty}^{\infty} f(m)e^{-2\pi imz}dm \tag{3.23}$$

The inverse Fourier transform [243] is given by:

$$f(m) = \int_{-\infty}^{\infty} F(z)e^{2\pi imz}dz \tag{3.24}$$

The discrete Fourier transform (DFT) [243] is a variant of the Fourier transform that finds the frequency domain representation of a finite segment of the input function that has been discretised through sampling from a continuous function. Taking the complex numbers $y_0, \ldots, y_{N-1}$ to be discrete samples from the function f, the DFT results in another sequence of N complex numbers $Y_0, \ldots, Y_{N-1}$, which is computed according to the formula:

$$Y(k) = \sum_{n=0}^{N-1} y_n e^{-\frac{2\pi i}{N}kn} \quad k = 0, \ldots, N-1 \tag{3.25}$$

The fast Fourier transform (FFT) [36] is an efficient algorithm to compute the DFT and its inverse.

The discrete Cosine transform (DCT) [1] is the equivalent of the DFT for real numbers. It transforms a finite number of real numbers into a sum of cosine functions oscillating at different frequencies.

3.2.2 Linear Predictive Coding

Linear predictive coding (LPC) [200, 202] is based on the principle that the speech signal consists of a source signal or excitation produced by the glottal chords, that gets transformed as it passes through a series of tubes that represent the articulators of the vocal tract or mouth. The series of tubes are also called poles, hence LPC is said to be based on an *all-pole* model of speech production. The LPC coefficients represent the formants or resonances of each tube and can be estimated by first estimating $x(m)$ as a linear combination of the previous P speech frames (an autoregressive process) using

linear prediction (LP) parameters, a_k, plus an additive term $e(m)$ that represents the excitation or source component.

$$x(m) = \sum_{k=1}^{P} a_k x(m-k) + e(m) \tag{3.26}$$

The term $\sum_{k=1}^{P} a_k x(m-k)$ represents the filter or effect of the vocal tract and speech articulators on the speech. The LPC coefficients are computed by sliding a window and computing the LP parameters for each window.

The LPC parameters are estimated by solving for the autoregressive coefficients. Methods that have been successfully applied include: the covariance method [7], the autocorrelation method [200, 201] and the lattice formulation [38, 155].

In the frequency domain, Eqn.3.26 becomes:

$$X(z) = \frac{E(z)}{1 - \sum_{k=1}^{P} a_k z^{-k}} = \frac{E(z)}{A(z)} \tag{3.27}$$

$E(z)$ represents the source part of the signal and $\frac{1}{A(z)}$ represents the filter part.

LPC can be used to resynthesise the original speech and is thus suited for low bandwidth communication such as Voice-over-IP.

More details on LPC can be found in [151].

3.2.3 Line Spectral Frequencies

Line spectral frequencies (LSF) [154] are a variant of LPC, particularly suited for transmission over a channel, such as a communication network. This is because for transmission vector quantisation needs to be performed and LPC is not very robust to quantisation noise. Instead, LSF is used, which involves decomposing the LP polynomial $A(z)$ into $P(z)$ and $Q(z)$ and finding their roots:

$$P(z) = A(z) + z^{-(p+1)} A(z^{-1}) \tag{3.28}$$

$$Q(z) = A(z) - z^{-(p+1)} A(z^{-1}) \tag{3.29}$$

where $P(z)$ corresponds to the vocal tract with the glottis closed and $Q(z)$ with the glottis open. The LSF are the roots of $P(Z)$ and $Q(Z)$ and occur in two symmetrical pairs $\pm v$. LSF parameters are preferred over LPC for speech coding for two reasons. First, LPC coefficients do not quantise well, i.e. small quantisation error may lead to large spectral distortion. Secondly, LPC coefficients do not interpolate well, i.e. it is hard to predict the coefficients in between those computed at two different times. LSF are robust to quantisation because they provide a frequency domain representation and quantisation can incorporate spectral features that are known to be important in speech perception. Moreover, frame-by-frame interpolation of LSF results in smooth spectral changes because of its close relationship to formant frequencies.

Refer to [151] for more details on how LSF are computed.

3.2.4 Mel-frequency Cepstral Coefficients

Mel-frequency cepstral coefficients (MFCCs) are computed by first computing the short-time power spectrum of a window of the speech signal, $x(m)$. This is done using the discrete Fourier transform (DFT):

$$X(k) = DFT\{x(m)\} \tag{3.30}$$

The frequency components are then warped to a filter-bank of M triangular filters based on the logarithmic Mel scale [296], which closely mirrors the human auditory perception system. The filter-bank of triangular filters corresponding to the Mel scale is shown in Figure 3.3. The filters are overlapping and are equally spaced along the Mel scale.

Cepstral coefficients are obtained from a Discrete Fourier Transform (DFT) of the logarithm of short-term power spectrum, represented by the M filter outputs. Since the spectrum of the speech signal is real and symmetric, a Discrete Cosine Transform (DCT) is used instead:

$$C(n) = DCT\{ln[|X(k)|]\} \tag{3.31}$$

where $C(n)$ are the cepstral coefficients.

The cepstrum is a good discriminant between vowels and consonants. Lower index cepstral coefficients represent the filter part of speech whilst the higher indexed coefficients represent the excitation (source) component. In speech recognition, typically only the first 13 cepstral coefficients are used, which correspond to the filter part of speech whilst discarding the source.

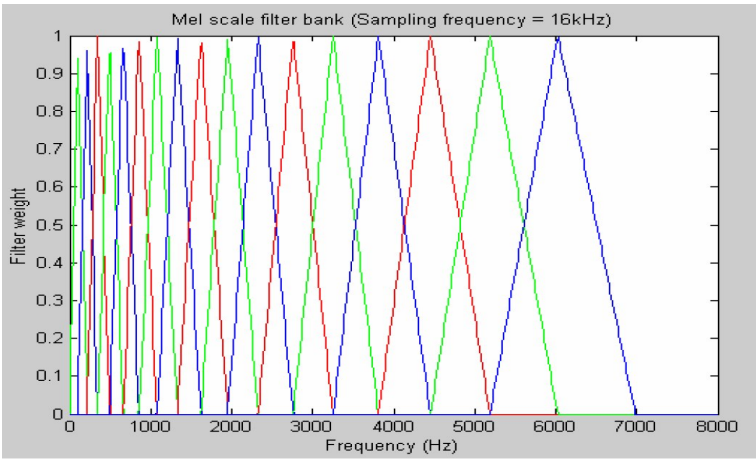


Figure 3.3: Triangular filter-banks based on the Mel scale.

MFCC feature extraction tries to accurately model how humans perceive speech and thus provides good discrimination between phonemes. They are used as the front-end of choice for speech recognition [247].

More details on MFCC are given in [151].

3.2.5 RASTA-PLP

The perceptual linear prediction (PLP) approach proposed by Hermansky [135] unifies LP models with cepstral analysis. A discrete Fourier transform (DFT) is first applied on a window of speech to compute the short-term power spectrum.

$$X(k) = DFT\{x(m)\} \tag{3.32}$$

The spectrum is then transformed to the Bark scale [329], which has similarities with the Mel scale [296], but uses M critical-band filters [329] rather than triangular filters. The filter-bank of critical-band filters corresponding to the Bark scale is shown in Figure 3.4. This is followed by an equal loudness pre-emphasis that approximates the sensitivity of human hearing at different frequencies. For cases where the Nyquist frequency [220], f, is above 5KHz, the following equation gives equal loudness weight, E, from the angular frequency, $w = 2\pi f$:

$$E = \frac{(w^2 + 56.8 \times 10^6)w^4}{(w^2 + 6.3 \times 10^6)^2(w^2 + 0.38 \times 10^9)^2(w^6 + 9.58 \times 10^{26})^2} \tag{3.33}$$

Each filter output in the filter-bank is weighted according to the equal loudness weight:

$$X_e(k) = E_k X(k) \quad 1 \leq k \leq M \tag{3.34}$$

The output is then compressed according to the equal-loudness power curve [151] to approximate the non-linear relationship between the intensity of a sound and its perceived loudness.

$$X_c(k) = (X_e(k))^{0.33} \quad 1 \leq k \leq M \tag{3.35}$$

The auditory spectrum is smoothed using linear prediction (LP) and the inverse discrete Fourier transform (IDFT) is applied. Finally, cepstral analysis is performed to obtain the PLP coefficients. The method has similarities with MFCC but with additional steps to better model the perceptual characteristics of speech.

RASTA-PLP, proposed by Hermansky et al. [136], is more robust to linear spectral distortions than PLP, because each frequency channel of PLP is band-pass filtered [131]. This attenuates effects that are due to additive[2] and convolutional[3] noise. It has also been shown that the RASTA-PLP features favour speaker-independence [134].

A more in-depth discussion of RASTA-PLP is given in [136].

[2] Additive noise is produced by the environment and is uncorrelated with the original speech signal.

[3] Convolutional noise is produced when the speech signal passes through a linear distortion channel such as the telephone line.

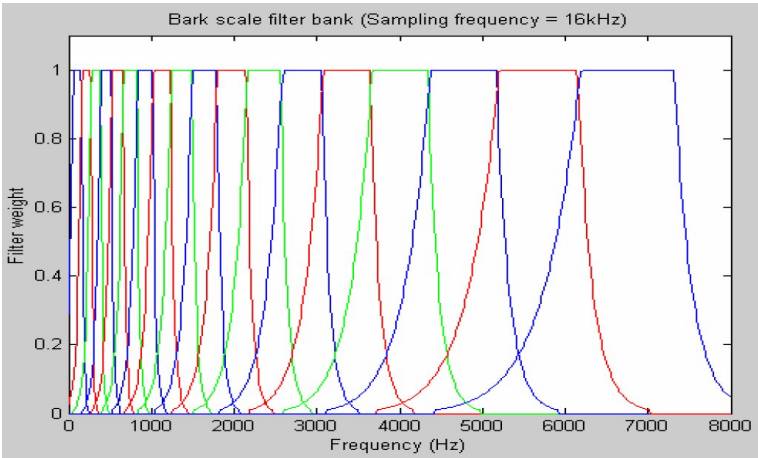


Figure 3.4: Critical-band filter-banks based on the Bark scale.

3.2.6 Discussion

We have presented different approaches to representing speech that are commonly used both in speech coding and speech feature extraction. Specifically, source-filter methods, which include LPC and LSF, and perceptually-motivated methods, namely MFCC and RASTA-PLP, have been dealt with. Source-filter methods are better suited for text-to-speech (TTS) synthesis whilst perceptually-motivated methods are more appropriate for speech recognition [144]. In Chapter 4, we present experiments to compare the effectiveness of these four different parameterisations for visual speech synthesis. We next describe the data corpora used in this work as well as details of audio and visual parameterisations on the data corpora.

3.3 Data Corpora

Two data corpora are used in this work: we call the first LIPS [281] and the second DEMNOW [101]. Two data corpora are used in this work because we want to test our proposed method both on a dataset that is phonetically-balanced (LIPS) as well as to a dataset that is closer to natural speech (DEMNOW).

The LIPS corpus (originally named LIPS08 corpus) [281] consists of 278 high quality sequences featuring a female British subject speaking sentences from the Messiah corpus [278], each sentence being of approximately of $3 - 6$ seconds. This corpus was made available for the *LIPS 2008 Visual Speech Synthesis Challenge* held at the Interspeech conference in Australia in 2008. The corpus consists of image frames of size 576×720 sampled at a rate of 50 frames-per-second (fps) according to the PAL standard [156]. A frame from the LIPS corpus is shown in Figure 3.5a. The corpus was downloaded from the website: `http://www.lips2008.org/`.

The DEMNOW corpus (originally named DemocracyNow! corpus) [101] consists of 803 sequences featuring a female American anchor giving news presentations, which was downloaded from http://gwenn.dk/demnow/. The sequences were extracted from newscasts of about an hour, featuring news presentations as well as panel discussions and interactive sessions. However, Englebienne [101] manually segmented the original videos into short sequences of about $3 - 10$ seconds, where the presenter is speaking whilst fully facing the camera and each sequence being delimited by leading and trailing silences or breaths. The original corpus consists of frames extracted at 29.97 fps according to the American NTSC standard [219], cropped around the face region and converted to grayscale. Because we want to retain colour information in this work, we had to take some additional steps to obtain the frames in colour. First, using the start and end frames for each sequence, obtained from *Avidemux*[4] project files in the original corpus, we re-extracted each video sequence from the original newscast videos. Frames were then extracted from the videos at 29.97 fps using the open-source tool *ffmpeg*[5]. The original frame size is 576×432, but is cropped to 288×302, which is the region where the face is located. A frame from the DEMNOW corpus is shown in Figure 3.6a.

Audio data in the form of WAV files at a bit rate of 44.1 KHz has also been made available for each corpora, as well as the phonetic annotation for each frame. The audio for each sequence of each corpus has been aligned to the corresponding video. The LIPS corpus was phonetically aligned with the HTK speech recognition software [318] using the BEEP phonetic dictionary [14], because it features a British speaker. The phonetic labels are given in terms of the timings as they occur in the video and this is processed to align the phonemes with each frame of the sequence, based on the visual frame rate of 50 fps. The DEMNOW corpus phonetic alignment was done using HTK [318] with the CMU pronunciation dictionary [308], because Amy Goodman, the news presenter, is American. The speech features used for phonetic alignment with HTK were MFCCs computed at 100Hz. Thus, the phonetic labels were obtained at 100Hz. In order to obtain phoneme labels synchronised with the visual frames, a voting scheme was used where a window of length equal to the ratio between the auditory frequency and the visual frequency was scanned through the data and the mode phoneme that labelled each visual frame in the window was chosen [101]. Thus, each visual frame in the DEMNOW corpus is labelled with a corresponding phoneme.

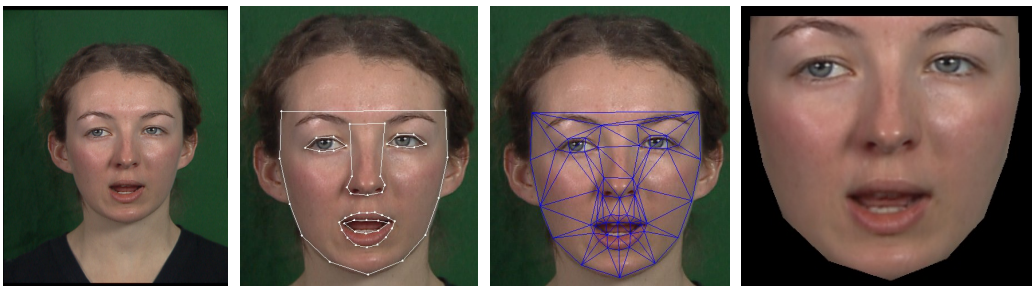
3.4 Visual Processing on Data Corpora

We use the active appearance model (AAM) [60] for visual processing, because of its ability to handle large pose variations that arise in the DEMNOW corpus and also

[4]Avidemux is an open-source video editing software: http://avidemux.org/
[5]Ffmpeg is a tool to record, convert and stream audio and video: http://www.ffmpeg.org/

because there is no need to deal with holes that arise in the synthesis stage of the multidimensional morphable model (MMM) [105].

For visual processing of the LIPS corpus, we downsampled the data to 25 fps by skipping every other frame in order to obtain a manageable corpus size. In order to train the AAM, we selected 184 prototype images by randomly choosing 4 frames from each of the 44 phonemes of the BEEP phonetic alphabet [14] plus silence and breath. This has been done automatically using a Matlab script. 56 markup points were then placed around the face, lips and nose in each of the prototype images (Figure 3.5b). The number of points was chosen after some manual experimentation in AAM model building and various trials in facial landmarking until decent reconstructions were obtained from AAM parameters. An AAM was built on the shapes and images by first aligning the shapes using generalised Procrustes analysis [126] and then computing a mean shape. As part of the AAM building process, the texture sampled from the convex hull of the shape for each prototype is warped to the mean shape using a piecewise affine warp algorithm. Figure 3.5c shows the Delaunay triangulation [74] of the shape vertices that needs to be performed for the piecewise affine warp. PCA is applied to the shape and texture separately and then again to the concatenated shape and texture PCA parameters. By retaining 99% of the variance of both the shape, texture and combined PCA, a 33-dimensional vector of AAM parameters is obtained.

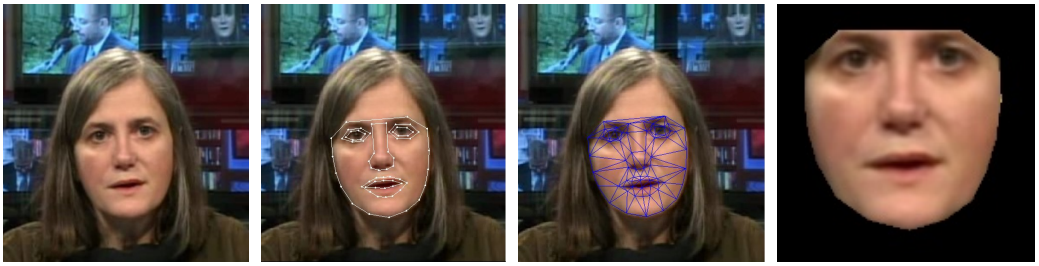


(a) (b) (c) (d)

Figure 3.5: (a) A frame from the LIPS corpus. (b) AAM markup points. (c) Delaunay triangulation. (d) AAM reconstruction.

Visual processing of the DEMNOW dataset using AAM is similar to the approach used for the LIPS corpus. We randomly selected 4 frames from each of the 39 phonemes of the CMU phonetic alphabet [308] plus silence and breath, giving a total of 164 frames for training the AAM. Because the DEMNOW corpus was processed after the LIPS corpus and having acquired experience in building AAMs for the LIPS corpus, more landmark points were chosen in order to have a smoother facial boundary. 69 landmark points were thus placed around the face, mouth, nose and eyes as shown in Figure 3.6b. The AAM was then built using the same procedure as outlined for the LIPS corpus.

The Delaunay triangulation [74] for a DEMNOW shape is shown in Figure 3.6c. By retaining 99% of the variance of both the shape, texture and combined PCA, a 24-dimensional vector of AAM parameters is obtained.



(a) (b) (c) (d)

Figure 3.6: (a) A frame from the DEMNOW corpus. (b) AAM markup points. (c) Delaunay triangulation. (d) AAM reconstruction.

For obtaining the AAM parameters for each frame, the AAM search algorithm (refer to Section 3.1.3) was applied to every sequence from the LIPS and DEMNOW corpora. The AAM tracking algorithm works as follows: For the first frame in the sequence, the image is scanned in rectangular grids from left to right, top to bottom, in order to locate the face region. The width and height of the rectangular grid is set to the width and height of the mean shape plus an offset. In each region, the texture is extracted from the mean shape within the grid and the difference image ΔI between the extracted texture and the mean image is computed. ΔI will be smaller when scanning near the face region. The region that gives the lowest ΔI is chosen. The initial shape is set as the mean shape and the AAM search algorithm is used to refine the AAM parameters until it matches the face. For subsequent frames in the sequence, the shape is initialised with the shape of the previous frame. When ΔI becomes too large for a given frame, which can happen as a result of drift, re-initialisation by the procedure mentioned for initialising the first frame is carried out.

After tracking, the shape, texture and combined parameters are projected to the corresponding retained eigenvectors, in order to obtain the AAM parameters for each frame. Reconstruction of an image is done by first reconstructing the combined shape and texture PCA coefficients from its AAM parameters. This is followed by projecting the shape and texture PCA parameters to the data space. Finally, the texture is warped from the mean shape to the reconstructed shape. A frame reconstructed from AAM parameters for the LIPS and DEMNOW corpora are shown in Figures 3.5d and 3.6d respectively.

Appendix B shows one and two standard deviations around the mean for each mode of variation of AAM parameters both for the LIPS and DEMNOW corpora.

3.4.1 Visual Normalisation

In this section, the techniques to retain only speech-related content in the visual domain are described. Different methods have been used for LIPS and DEMNOW due to the nature of the datasets and the chronology of the visual processing done for each dataset.

Mean-centering AAM Parameters

The pose variations in the LIPS corpus arise as a result of discrepancies in the orientation of the speaker with respect to the camera. Thus, there is a fixed pose per sequence with little or no head movements within that particular sequence. Such pose variations can be removed by first computing the mean of the parameters for each mode of variation for a given sequence and then subtracting that mean from the corresponding parameters, as illustrated in Eqn.3.36, where i is a given mode of variation and $\bar{\mathbf{c}}_i$ is the mean of the ith mode of variation across all frames in the sequence.

$$\mathbf{c}_i = \mathbf{c}_i - \bar{\mathbf{c}}_i \tag{3.36}$$

Figure 3.7 illustrates the mean AAM trajectories for a given sequence from the LIPS corpus, before and after normalisation. The parameters in the normalised data are varying around the zero baseline as opposed to the unnormalised data, where the baseline is an offset below zero.

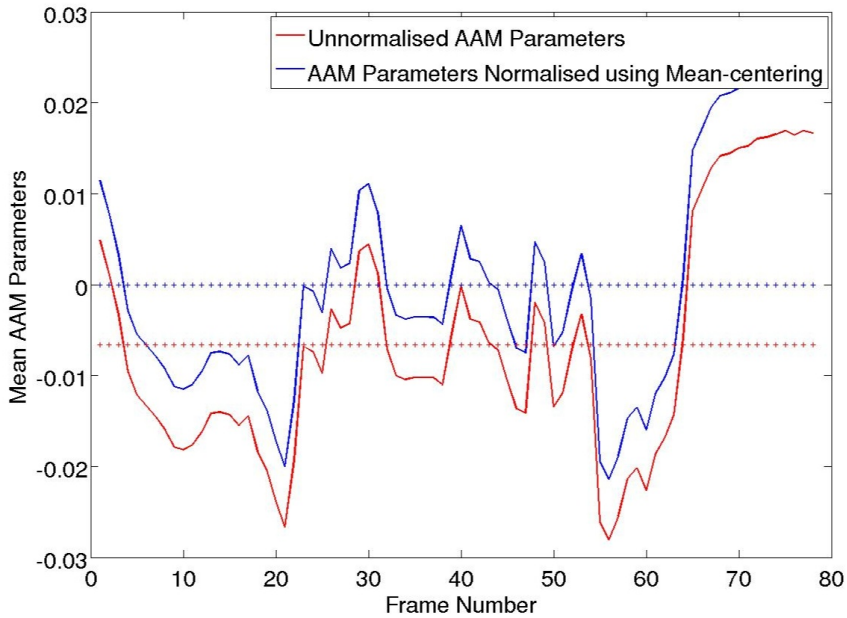


Figure 3.7: Mean AAM trajectories for a given LIPS sequence before and after normalisation. The baseline before and after normalisation is also shown.

Independent Component Analysis

The DEMNOW dataset contains much more variability than the LIPS corpus. There are sequences where the face is partially occluded with hair, and within a particular sequence, there are variations in pose and expression. Thus, the normalisation technique mentioned for the LIPS corpus does not apply here. This is because the LIPS corpus contains a lot of variability across sequences but not within a particular sequence. Such variability can be easily dealt with by subtracting the offset from the AAM modes of variation. For the DEMNOW corpus, we resorted to a different normalisation technique using independent component analysis (ICA) [56]. We drew inspiration from the use of ICA by Cao et al. [41] to separate content from style for facial motion capture data. In practice, it is possible to use ICA to perform visual normalisation for the LIPS dataset. However, the experiments for LIPS were carried out before those of DEMNOW and the mean substraction method was found to work well for the LIPS dataset.

ICA finds a basis that produces components that are independent, while PCA produces components that are uncorrelated. Eqn.3.37 shows the decomposition found by PCA whilst Eqn.3.38 shows the decomposition found by ICA. $\bar{\mathbf{b}}$ is the mean of the data, $\mathbf{P}_b$ are the principal components and $\mathbf{c}$ are the PCA parameters. In the case of ICA, the PCA parameters are further decomposed into ICA parameters $\mathbf{u}$, using a linear basis $\mathbf{A}$. The ICA basis is computed so as to maximise the independence and non-Gaussianity of the independent components $\mathbf{u}$.

$$\mathbf{b} = \bar{\mathbf{b}} + \mathbf{P}_b\mathbf{c} \tag{3.37}$$

$$\mathbf{b} = \bar{\mathbf{b}} + \mathbf{P}_b\mathbf{A}\mathbf{u} \tag{3.38}$$

We have used the FastICA algorithm [152] to find the independent components from AAM parameters. After the ICA basis $\mathbf{A}$ is found, the ICA parameters $\mathbf{u}$ are computed from the AAM parameters as $\mathbf{u} = \mathbf{A}^{-1}\mathbf{c}$. By visualising the modes of variation for each independent component, we identify components pertaining to style and set them to zero. The AAM PCA parameters are then reconstructed from the ICA parameters, according to $\mathbf{c} = \mathbf{A}\mathbf{u}$, giving normalised AAM parameters with uniform pose and expressive cues removed. Details on the FASTICA algorithm are given in Appendix A.

Figure 3.8 illustrates the mean AAM trajectories for a given sequence from the DEMNOW corpus before and after normalisation, which illustrates that unlike the LIPS corpus, there is no fixed offset that can be substracted for normalisation, given that there are a lot of pose and expression variations within a particular sequence. However, the normalised AAM parameters still vary around a baseline of zero, just as for the LIPS dataset.

Appendix C shows one and two standard deviations around the mean for each mode

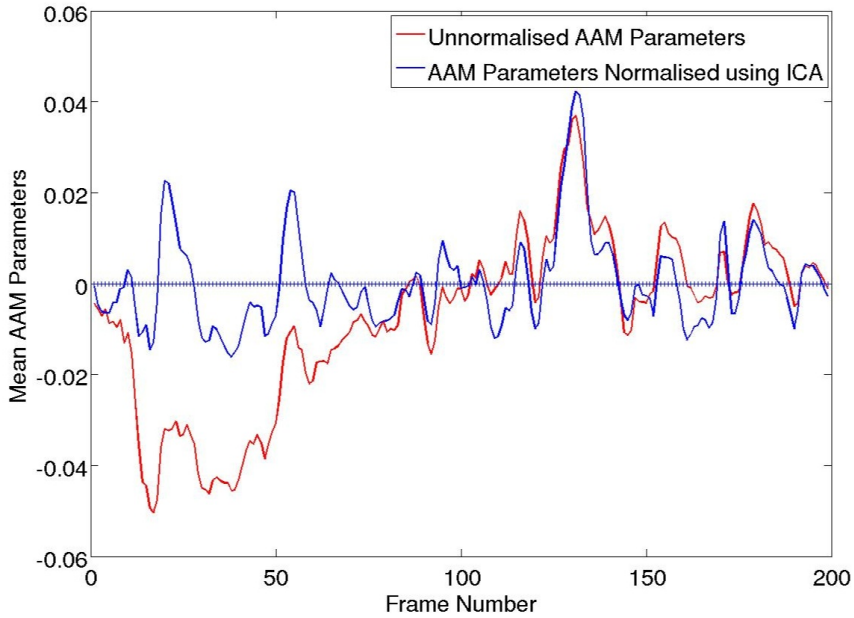


Figure 3.8: Mean AAM trajectories for a given DEMNOW sequence before and after normalisation.

of variation of ICA parameters for the DEMNOW corpus.

3.5 Audio Processing on Data Corpora

In this work, we aim to investigate the reliability of different speech codecs and feature extraction methods in visual speech synthesis. In particular, we use the source-filter codecs LPC and LSF as well as the MFCC and RASTA-PLP features, which are perceptually-motivated. In order to satisfy the requirement of having a window where the speech signal is stationary, a window size of 25ms and a hop size of 10ms is typically used (refer to Section 3.2), resulting in an audio processing frequency of 100Hz. This results in a mismatch with the visual processing rate of 25 fps or 29.97 fps used for the LIPS and DEMNOW corpora respectively. As a result, the speech parameters need to be downsampled to match the visual frame rate.

Theobald and Wilkinson [279] conducted experiments to investigate the effect of increasing the window size in speech parameterisation to 40 ms in order to match the visual frame rate of 25 fps. It was found that the larger window size resulted in smoother speech features that have higher linear correlation with AAM features, as compared to the correlation between upsampled visual features to match the speech parameterised at 100 Hz, and the speech parameters. This is because speech parameters at 100 Hz have more rapidly changing properties compared to the movement of articulators in the visual domain.

We have used 20 parameters to represent the LPC, LSF, MFCC and RASTA-PLP features. This was done in order to have the audio vector dimensionality as being comparable to the visual vector dimensionality. This was found to be important when learning our proposed joint probabilistic model of audio and video (refer to Chapter 4 Section 4.3.4).

3.5.1 Synchronisation of Audio and Visual Parameters

In this work, we investigate three approaches to matching the audio features to visual features. The first approach consists of using an auditory window of 50 ms and a hop window of 40 ms in order to obtain speech features at 25 Hz for the LIPS corpus. For the DEMNOW corpus, an auditory window of 50 ms and a hop window of 33 ms is used in order to obtain the speech features at 29.97 Hz.

The second and third approaches consist of parameterising the speech at 100 Hz using an auditory window of 25 ms and a hop window of 10 ms and downsampling to match the visual features using polyphase quadrature filtering [256] and median filtering [6], respectively. In polyphase quadrature filtering, the input signal is mapped to the frequency domain into an equidistant number of sub-bands. These sub-bands are downsampled by a factor equal to the ratio between the visual and audio frequencies by maintaining the number of samples per second the same, i.e. using critical sampling. Finally, the signal is mapped back to the time domain. In median filtering, the signal is divided into windows of length equal to the ratio between the audio and visual frequencies and the median value in each window is chosen.

3.5.2 Comparing Speech Parameterisation Methods

In this section, we visualise the mean trajectories of the different speech parameterisation techniques. For each speech parameterisation, we plot the speech features processed at the same frequency as the visual frame rate, as well as the features downsampled from 100Hz using median filtering and polyphase quadrature filtering. As a preprocessing step to each of the speech parameterisation techniques, a preemphasis filter with k = 0.97 (refer to Section 3.2) is applied, as suggested by Young et al. [318].

LPC

The order of the autoregressive process (refer to Section 3.2.2) for LPC was chosen to be 20, giving a 20-dimensional vector of LPC parameters.

Figures 3.9 and 3.10 show the mean trajectories of the LPC parameters for a sequence of the LIPS and DEMNOW corpora respectively, together with the mean trajectories of the downsampled parameters. It can be seen that the LPC parameters

processed at the visual frame rate are smoother than the downsampled parameters. This is because using a larger non-stationary window requires a more general autoregressive process, thus leading to smoothing of the parameters.

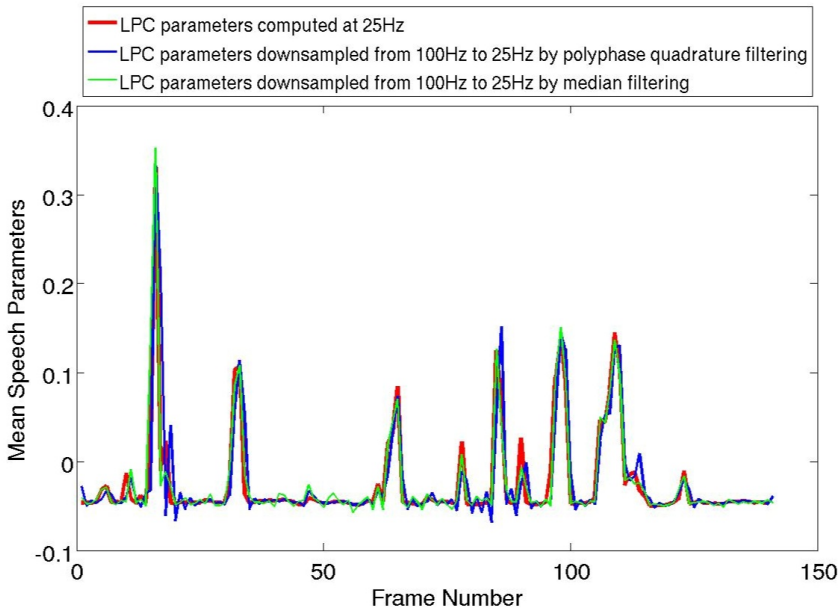


Figure 3.9: Mean LPC trajectories for a given LIPS sequence.

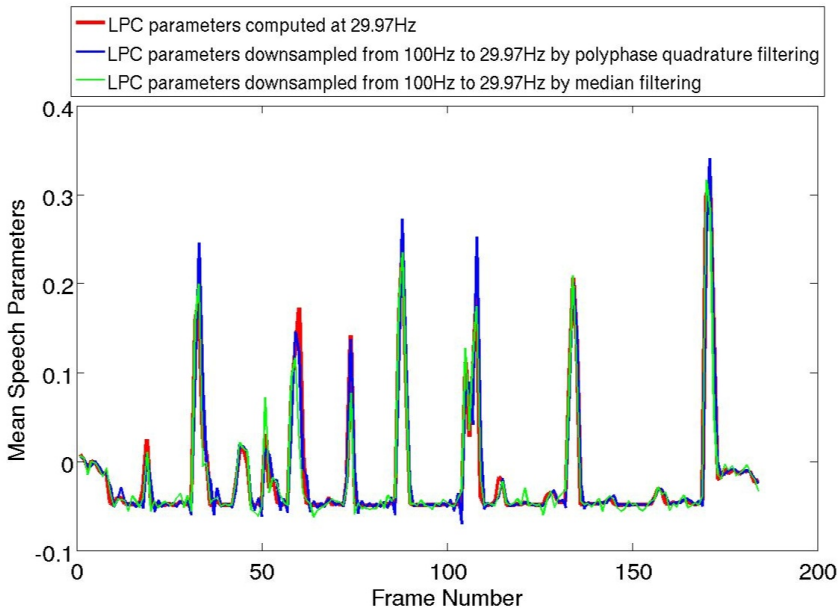


Figure 3.10: Mean LPC trajectories for a given DEMNOW sequence.

LSF

The 20 LPC parameters are mapped to LSF coefficients using the method described in Section 3.2.3, giving another 20-dimensional vector of LSF parameters.

Figures 3.11 and 3.12 show the mean trajectories of the LSF parameters for a sequence of the LIPS and DEMNOW corpora respectively, together with the mean trajectories of the downsampled parameters. From the visualisations, there is not a big difference between the trajectories of the speech parameters computed at the visual frame rate and those of the downsampled parameters. This is because the differences in LPC coefficients between the different representations are attenuated when converting to a frequency domain representation for LSF.

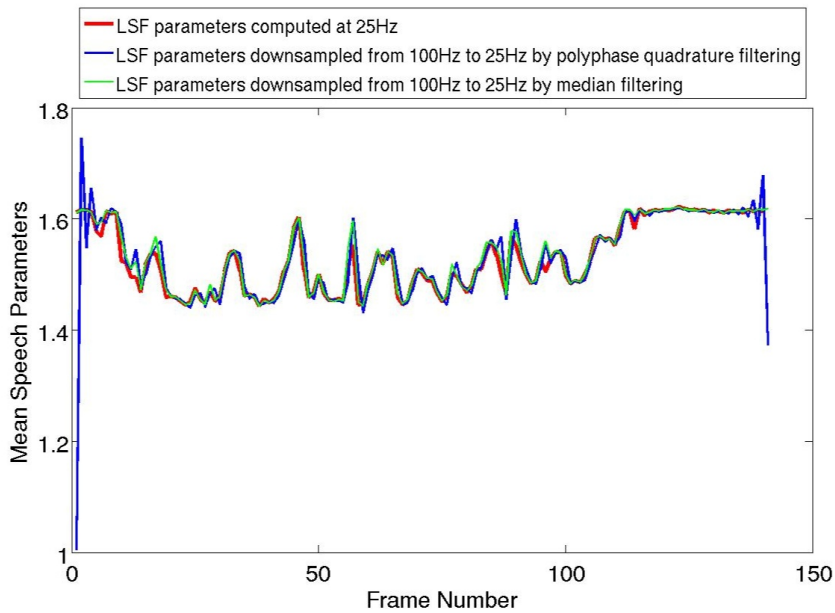


Figure 3.11: Mean LSF trajectories for a given LIPS sequence.

MFCC

MFCC features are extracted from the speech data as described in Section 3.2.4. 30 triangular filter-banks, warped according to the Mel scale, are used and 21 cepstral features are computed. The first ceptral feature, which corresponds to the the 0th-order coefficient is ignored, because it can be regarded as a collection of average energies of all frequency bands and contains speaker-specific information [241]. This results in a 20-dimensional vector of MFCC features.

Figures 3.13 and 3.14 show the mean trajectories of the MFCC parameters for a sequence of the LIPS and DEMNOW corpora respectively, together with the mean trajectories of the downsampled parameters. Due to the non-stationary window used

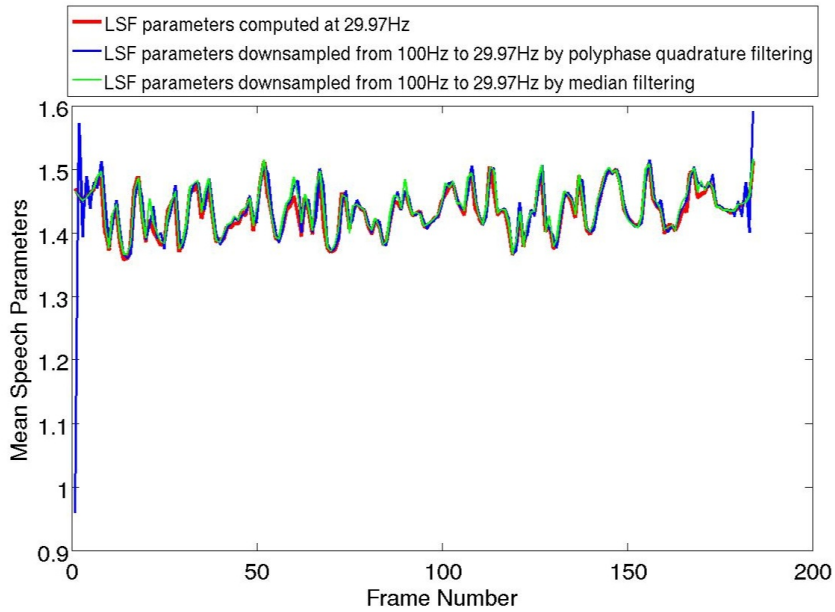


Figure 3.12: Mean LSF trajectories for a given DEMNOW sequence.

for MFCC parameters processed at the visual frame rate, their trajectories are smoother than downsampled MFCC parameters.

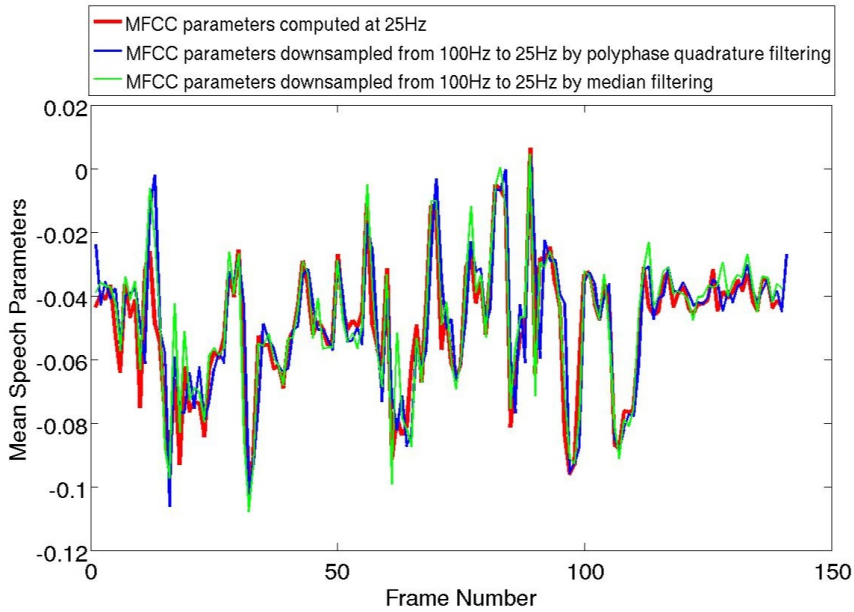


Figure 3.13: Mean MFCC trajectories for a given LIPS sequence.

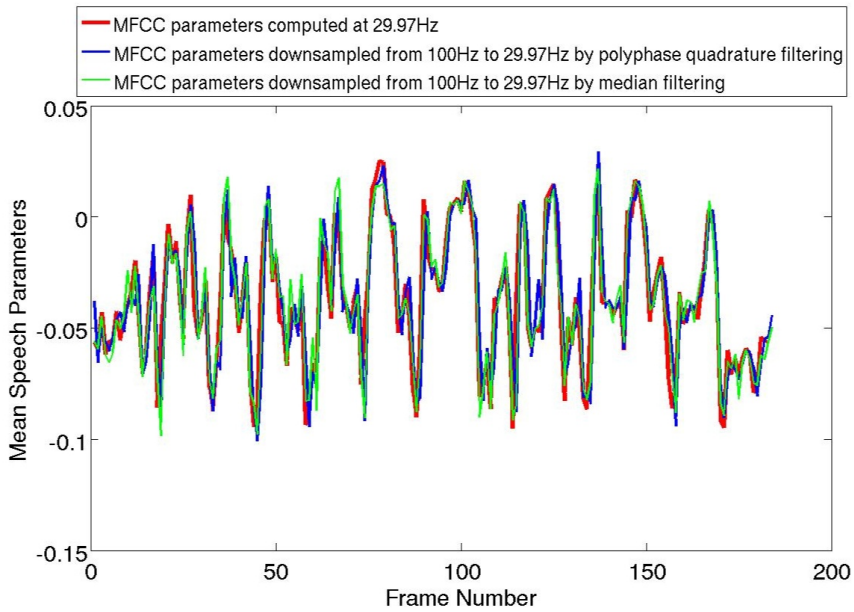


Figure 3.14: Mean MFCC trajectories for a given DEMNOW sequence.

RASTA-PLP

We compute RASTA-PLP features, as described in Section 3.2.5, using 30 critical-band filters warped according to the Bark scale. Just as for the other speech parameterisation techniques, 20 RASTA-PLP features are computed.

Figures 3.15 and 3.16 show the mean trajectories of the RASTA-PLP parameters for a sequence of the LIPS and DEMNOW corpora respectively, together with the mean trajectories of the downsampled parameters. The RASTA-PLP parameters processed at the visual frame rate are smoother than those downsampled due to the non-stationary window used for the former category.

3.5.3 Discussion

From the plots of the speech parameter trajectories, it is shown that the difference between the downsampled and lower-frequency LPC and LSF parameters is not significant except that downsampling results in a more "wavy" pattern, which indicates more detail preservation at the temporal scale. However, for MFCC and RASTA-PLP, the parameters obtained using a larger window are much more smoothed out as compared to the downsampled parameters. This is because a larger auditory window results in a representation that averages out the non-stationary properties of the signal in that window. In the next chapter, a probabilistic method for coupling audio and visual parameters is presented. Specific experiments are conducted to investigate which speech

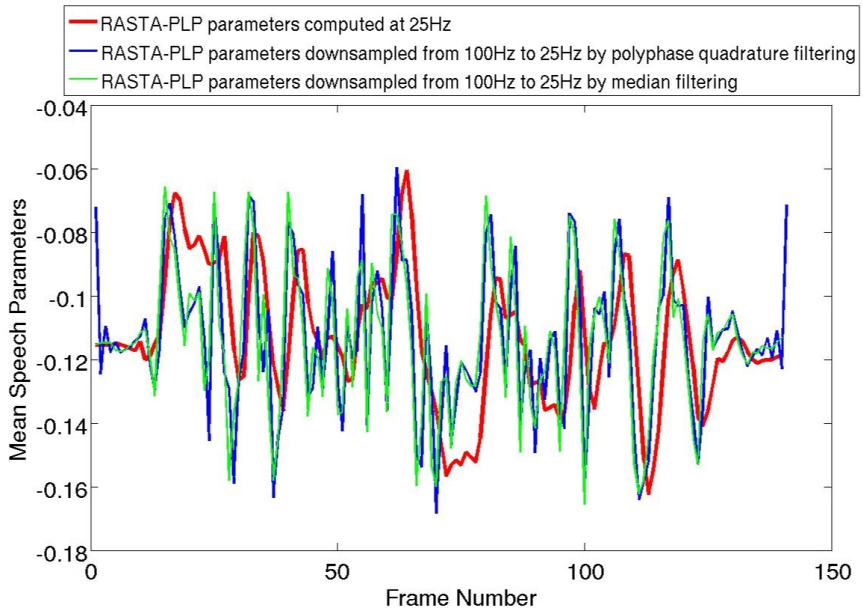


Figure 3.15: Mean RASTA-PLP trajectories for a given LIPS sequence.

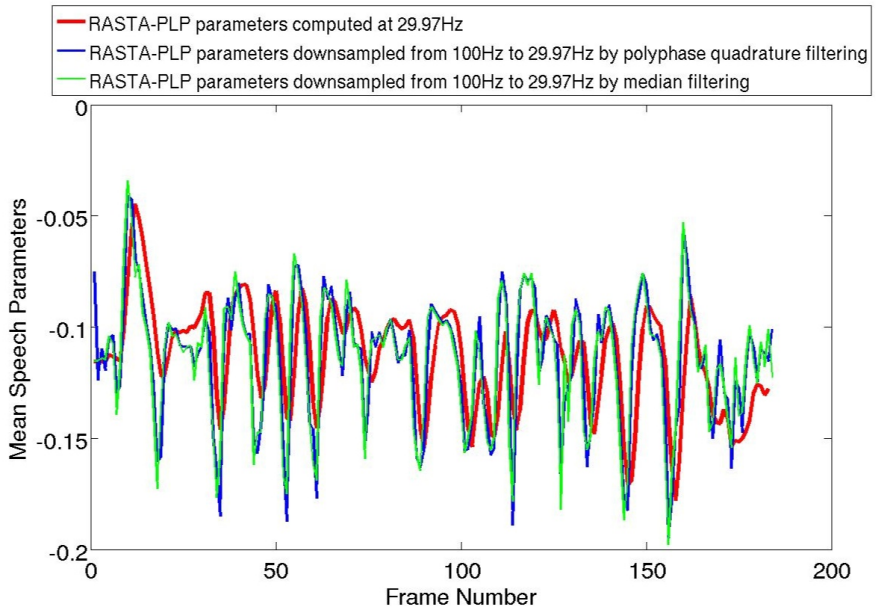


Figure 3.16: Mean RASTA-PLP trajectories for a given DEMNOW sequence.

parameterisation method and which audio-visual synchronisation method is most effective at predicting visual parameters of a talking face. It is known that facial articulators move at a slower rate than the vocal cords [151]. Thus, in the next chapter, we aim to investigate whether smooth audio speech features obtained using a non-stationary window are better predictors of visual speech as compared to audio speech processed using a stationary window and downsampled to match the visual rate.

3.6 Chapter Summary

This chapter presented a review of different facial modelling and speech processing methods. It then dealt with a description of the data corpora used in this work as well as techniques for extracting audio and visual parameters in order that they are both synchronised with each other. The active appearance model (AAM) has been used for extracting visual features. These parameters need to be normalised in order to retain only speech-related content whilst excluding style-related content such as pose variations and expressions. We proposed two techniques for visual normalisation, namely mean-centering of AAM parameters and style-content separation using Independent Component Analysis (ICA). The applicability of these two methods depend whether the variations in visual data occur within or between sequences in the two data corpora. For audio parameterisation, we compute LPC, LSF, MFCC and RASTA-PLP parameters. The parameters are computed at 100 Hz and then downsampled to match the visual frame rate using polyphase quadrature filtering and median filtering. We also compute speech features to match the visual frame rate for each corpora by varying the window size and hop size. It is found from plots of the speech parameter trajectories, that features computed from the latter method are smoother but might potentially discard some salient features.

Chapter 4

State-Space Model for Audio-visual Mapping

> In the field of observation, chance favours only the prepared mind.
>
> Louis Pasteur

In this chapter, we present a non-linear state-space model that can be used to jointly model the audio and visual data of a talking face. Using the probabilistic graphical model framework, various latent variable models are presented, highlighting their previous application to visual speech synthesis. This is followed by a treatment of some shared latent variable models that can couple two data streams through a shared latent space. We then present our shared latent variable model using Gaussian processes, giving rise to the shared Gaussian process dynamical model (SGPDM). We show how the SGPDM can be used for audio-visual mapping and present several experiments to optimise the free parameters of the model. Finally, experiments are done to investigate which audio parameterisation method and which audio-visual synchronisation method best predict visual from audio parameters using the SGPDM.

A preliminary version of some of the work in this chapter appeared in [70].

4.1 Probabilistic Graphical Models

Probabilistic graphical models use a graph-based representation as the basis for compactly encoding a complex distribution over a high-dimensional space [171]. In this work, they are used to represent the generative model of speech.

In a graphical model, a vertex or node represents the variables and the arcs or edges represent probabilistic dependencies between the variables. There are two main types of graphical models: directed graphical models, also known as Bayesian networks, and

undirected graphical models, also known as Markov random fields. Bayesian networks are useful for representing causal relationships between random variables and thus can be used to model generative processes. Markov random fields are applicable to the representation of soft constraints between variables [22]. Since we deal with the former case, we limit ourselves to the use of directed graphs. In Bayesian networks, the presence of a link from variable $\mathbf{X}$ to $\mathbf{Y}$ signifies a conditional dependency between $\mathbf{Y}$ and $\mathbf{X}$. The lack of links is more useful because it translates to conditional independence properties.

The notation used is as follows: circles represent continuous variables and squares represent discrete variables. Observed variables are shaded and hidden variables are unshaded. This is illustrated in Figure 4.1.

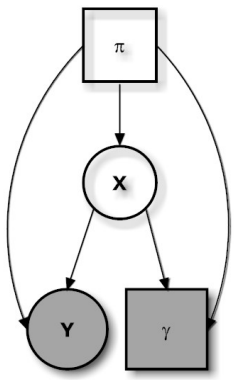


Figure 4.1: Notation used for graphical model: circles - continuous variables, squares - discrete variables, shaded - observed variables, unshaded - hidden variables.

The likelihood of the model shown in Figure 4.1 is given by:

$$p(\mathbf{Y}, \mathbf{X}, \boldsymbol{\gamma}, \boldsymbol{\pi}) = p(\mathbf{Y}|\mathbf{X}, \boldsymbol{\pi})p(\boldsymbol{\gamma}|\mathbf{X}, \boldsymbol{\pi})p(\mathbf{X}|\boldsymbol{\pi})p(\boldsymbol{\pi}) \tag{4.1}$$

If there were no link from $\mathbf{X}$ to $\mathbf{Y}$, the likelihood would become:

$$p(\mathbf{Y}, \mathbf{X}, \boldsymbol{\gamma}, \boldsymbol{\pi}) = p(\mathbf{Y}|\boldsymbol{\pi})p(\boldsymbol{\gamma}|\mathbf{X}, \boldsymbol{\pi})p(\mathbf{X}|\boldsymbol{\pi})p(\boldsymbol{\pi}) \tag{4.2}$$

Thus the absence of links conveys information about the conditional independence properties in the model.

The formulation of the likelihood allows us to train the model by maximising the likelihood with respect to the model parameters.

Graphical models allow us to do inference about the hidden variables given some observations using the sum and product rules of probability. Inference usually involves marginalising over some variables we are not interested in, which involves summations in the discrete case and integrations in the continuous case. By rearranging the equations involved in marginalisation and taking into account the ways the distribution factorises, we can make the computations much more efficient than relying on a naive

variable-elimination method. By exploiting the graphical model structure, these efficient inference algorithms can be formulated as message passing between nodes [22]. The generalised method for performing inference in graphical models is called the sum-product algorithm [173], which relies on converting the graphical model to a factor graph. Belief propagation is a more general algorithm for any type of graphical model including those with loops [315]. These methods provide a framework to specify any graphical model and have a generalised way of performing inference. However, the inference steps for specific graphical models can also be derived independently using the sum-product algorithm.

4.2 Latent Variable Models

The probabilistic methods adopted in this work can be broadly categorised as latent variable models because they involve the discovery of the underlying structure of the data, which can in some cases be represented as the evolution of a state-space. The states are continuous in our case but can also be discrete. In order to motivate the state-space models proposed in this thesis, we present other latent variable models that have been previously applied to visual speech synthesis.

4.2.1 Linear Subspace Models

Although linear subspace models have not been directly applied to visual speech synthesis, they have been used to initialise the latent space in our proposed joint probabilistic models of audio and video. They thus deserve a treatment of their own.

In linear subspace models, the latent space $\mathbf{X} = \{\mathbf{x}_n\}_{n=1}^N \in \mathbb{R}^{d\times N}$ is related to mean-centered observed data $\mathbf{Y} = \{\mathbf{y}_n\}_{n=1}^N \in \mathbb{R}^{D\times N}$ through a linear mapping $\mathbf{W} \in \mathbb{R}^{d\times D}$ that is corruped by noise:

$$\mathbf{Y} = \mathbf{W}\mathbf{X} + \boldsymbol{\mu} + \epsilon \tag{4.3}$$

where $\boldsymbol{\mu}$ allows having non-zero means in the model and ϵ is a noise term.

The model given by Eqn.4.3 represents a reformulation of principal component analysis (PCA) as a latent variable model and has been called probabilistic principal component analysis (PPCA) [285] in the case where the noise follows an isotropic Gaussian distribution. The graphical model for PPCA is shown in Figure 4.2.

The likelihood of PPCA is given by:

$$p(\mathbf{Y}, \mathbf{X}|\mathbf{W}, \boldsymbol{\mu}, \sigma^2) = p(\mathbf{Y}|\mathbf{X}, \mathbf{W}, \boldsymbol{\mu}, \sigma^2)p(\mathbf{X}) \tag{4.4}$$

where $p(\mathbf{Y}|\mathbf{X}, \mathbf{W}, \boldsymbol{\mu}, \sigma^2) \sim \mathcal{N}(\mathbf{W}\mathbf{X} + \boldsymbol{\mu}, \sigma^2\mathbf{I})$ and $p(\mathbf{X})$ is a prior distribution on $\mathbf{X}$.

By placing a Gaussian prior over the latent points $\mathbf{X}$ and marginalising the latent points, the mapping $\mathbf{W}$ as well as $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}^2$ can be solved for in closed-form using

maximum-likelihood as shown by Tipping and Bishop [285].

In this work, PPCA is one of the methods used to initialise latent spaces for our proposed non-linear shared latent variable models (refer to Section 4.5.1).

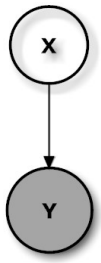


Figure 4.2: Graphical model for probabilistic principal component analysis and Gaussian process latent variable model.

4.2.2 Gaussian Mixture Model

In a Gaussian mixture model (GMM), the latent states $\boldsymbol{\pi} \in \{1, \dots K\}$ are discrete and the observed variable $\mathbf{Y}$ is a mixture of Gaussian distributions:

$$p(\mathbf{Y}, \boldsymbol{\pi} | \mathbf{m}, \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \prod_{n=1}^{N} \sum_{k=1}^{K} m_k \mathcal{N}(\mathbf{y}_n | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \tag{4.5}$$

where m_k is the mixing coefficient for state k and $\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k$ are the means and covariances of the Gaussians for state k. GMMs can be used to segment data into clusters of Gaussian distributions. The graphical model of the GMM is shown in Figure 4.3.

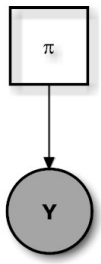


Figure 4.3: Graphical model for Gaussian mixture model.

Training

The aim of GMM training is to estimate the parameters of the Gaussian clusters that constitute the GMM as well as the mixing coefficients. The parameters of the GMM are estimated by maximum-likelihood (ML) using the expectation-maximisation (EM) algorithm [75]. In the E-step, the probability of data point $\mathbf{y}_n$ belonging to state k is computed as γ_{nk}, which is also known as the occupancy matrix.

$$\gamma_{nk} = \frac{m_k \mathcal{N}(\mathbf{y}_n | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_{j=1}^{K} m_k \mathcal{N}(\mathbf{y}_n | \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)} \tag{4.6}$$

In the M-step, the GMM parameters, $\boldsymbol{\mu}_k$, $\boldsymbol{\Sigma}_k$ and m_k are re-estimated based on the occupancy matrix calculated. The E and M steps are repeated until the likelihood, given by Eqn.4.5 converges.

Inference

For inference, each point needs to be assigned to a state, corresponding to the cluster index. This is done by maximising $\boldsymbol{\gamma}_n$, which is the probability of data point $\mathbf{y}_n$ given the states, with respect to k.

$$\pi_n = \arg\max_k (\boldsymbol{\gamma}_n) \tag{4.7}$$

where $\boldsymbol{\gamma}_n = p(\mathbf{y}_n | \boldsymbol{\pi})$.

Application to Visual Speech Synthesis

GMMs have been used for speech animation in the work of Ezzat et al. [105] and Ezzat [104]. The clusters corresponded to phonemes obtained by performing forced phonetic alignment on audio data. As a result, EM [75] was not necessary for training but instead, the means and covariances of each phonetic state were estimated from the visual data, represented by multidimensional morphable model (MMM) [160] parameters. Because the GMM does not model dynamics between successive frames, an interpolation algorithm was proposed to generate smooth animation parameters from a stream of phonetic labels corresponding to a test sequence. The algorithm was based on the minimisation of an objective function consisting of a target term and a regularisation term to ensure smoothness. The objective function is given by:

$$E = (\mathbf{Y} - \boldsymbol{\mu})^T \mathbf{D} \boldsymbol{\Sigma}^{-1} \mathbf{D} (\mathbf{Y} - \boldsymbol{\mu}) + \lambda \mathbf{Y}^T \mathbf{W}^T \mathbf{W} \mathbf{Y} \tag{4.8}$$

where λ is the coefficient of the regularising term, $\mathbf{D}$ is duration-weighting matrix for emphasising shorter phonemes and de-emphasising longer ones, $\mathbf{W}$ is a band-diagonal matrix to ensure smoothness, $\boldsymbol{\mu}$ is a concatenated vector of the means of all phonemes and $\boldsymbol{\Sigma}$ is a block-diagonal matrix of the covariance for all the phonemes and $\mathbf{Y}$ consists of the shape or appearance parameters of the MMM, which were treated separately.

The proposed method has been shown to generate very realistic-looking animations and attracted a lot of media attention. However, one of its limitations is that some aspects of the model have to be adjusted to lead to acceptable results. For example, if the means and covariances of visual parameters for each phoneme are computed automatically, the resulting animations are under-articulated. Instead, these have to be adjusted using gradient descent learning [21]. Moreover, the regularisation term λ

and the order of the smoothness term $\mathbf{W}$ (number of times it is multiplied to itself) need to be optimised through cross-validation.

More recently, another application of GMMs to visual speech synthesis was proposed by Zhuang et al. [326]. An audio-visual joint GMM was trained on a concatenation of audio and visual data. A minimum converted trajectory error (MCTE) approach was then proposed to address the issue of under-articulation. The MCTE method uses a generalised probabilistic descent algorithm to minimise the conversion error of the visual parameters according to the input speech. This model, however, has no explicit dynamical constraints but instead uses dynamical speech features to address coarticulation. Hidden Markov models are one way to include dynamical information in the generative model.

4.2.3 Hidden Markov Model

In terms of its graphical representation, the hidden Markov model (HMM) is similar to the GMM with the exception that there is a dynamical mapping from the previous state to the next. This dynamical mapping is given in the form of transition probabilities, $p(\pi_t|\pi_{t-1}, \mathbf{A})$. The transition probability matrix $\mathbf{A}$ stores the transition probabilities from each state to each other state. The likelihood of the model is given by:

$$p(\mathbf{Y}, \boldsymbol{\pi}|\mathbf{s}, \mathbf{m}, \boldsymbol{\mu}, \boldsymbol{\Sigma}, \mathbf{A}) = p(\pi_1|\mathbf{s}) \left[\prod_{t=2}^{T} p(\pi_t|\pi_{t-1}, \mathbf{A}) \right] \prod_{t=1}^{T} \sum_{k=1}^{K} m_k \mathcal{N}(\mathbf{y}_t|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \tag{4.9}$$

where s_k is the starting probability for state k. The graphical model for a HMM is shown in Figure 4.4.

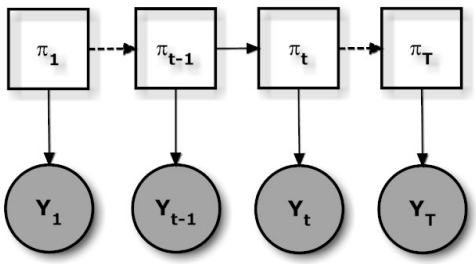


Figure 4.4: Graphical model for a hidden Markov model.

Training

The parameters of the HMM are also estimated using the EM algorithm [75]. The E-step involves the computation of γ_{tk} and $\xi_{j,tk}$. $\xi_{j,tk}$ is the probability of data point $\mathbf{y}_{t-1}$ being in state j and data point $\mathbf{y}_t$ being in state k. γ_{tk} and $\xi_{j,tk}$ are computed using the forward-backward algorithm [248], which is derived from the sum-product

algorithm [173]. In the M-step, the parameters of the HMM are re-estimated based on the calculated values for γ_{tk} and $\xi_{j,tk}$. The E and M steps are repeated until convergence of the likelihood given by Eqn.4.9.

Inference

For inferring the states $\boldsymbol{\pi}^* = \{\pi_t^*\}_{t=1}^T$ given a sequence of observation $\hat{\mathbf{Y}} = \{\hat{\mathbf{y}}_t\}_{t=1}^T$, a dynamic programming algorithm called the Viterbi algorithm [293] is used, which optimises the states from the joint likelihood between the observations and the states:

$$\boldsymbol{\pi}^* = \arg\max_{\hat{\boldsymbol{\pi}}} p(\hat{\mathbf{Y}}, \hat{\boldsymbol{\pi}}) \tag{4.10}$$

The Viterbi algorithm can be derived by replacing the *sum* in the sum-product algorithm with a maximisation, leading to the max-sum algorithm [22]. This can be solved efficiently using dynamic programming comprising two stages. The first consists of a forward pass through the sequence where for each observation $\hat{\mathbf{y}}_t$ and every possible state $\hat{\pi}_t$, the probability of the state sequence that ends in $\hat{\pi}_t$ and maximises the joint probability of the state and observation sequences up to time t, is computed. The state at time $t-1$ that maximises the probability to each state at time t is stored. In the second stage, a backtracking step is done when the last time frame T is reached. Starting with the end state π_T^* that gives the highest probability, the states along the path to the first state π_1^* are traced back, giving the optimal path through the states. The steps of the Viterbi algorithm are summarised in Algorithm 2.

Algorithm 2 The Viterbi algorithm for the HMM

Input: Test sequence $\hat{\mathbf{Y}} = \{\hat{\mathbf{y}}_t\}_{t=1}^T$
Output: Inferred states $\boldsymbol{\pi}^* = \{\pi_t^*\}_{t=1}^T$
Initialisation:
$Q_1(\hat{\pi}_1) \leftarrow p(\hat{\pi}_1)p(\hat{\mathbf{y}}_1|\hat{\pi}_1),\ \hat{\pi}_1 \in \{1 \dots K\}$
$S_1(\hat{\pi}_1) \leftarrow 0$
Iteration:
$Q_t(\hat{\pi}_t) \leftarrow p(\hat{\mathbf{y}}_t|\hat{\pi}_t) \max_{\hat{\pi}_{t-1}} \big(Q_{t-1}(\hat{\pi}_{t-1})p(\hat{\pi}_t|\hat{\pi}_{t-1})\big),\ \hat{\pi}_t \in \{1 \dots K\}, 2 < t \leq T$
$S_{t-1}(\hat{\pi}_{t-1}) \leftarrow \arg\max_{\hat{\pi}_{t-1}} \big(Q_{t-1}(\hat{\pi}_{t-1})p(\hat{\pi}_t|\hat{\pi}_{t-1})\big),\ \hat{\pi}_t \in \{1 \dots K\}, 2 < t \leq T$
Termination
$\pi_T^* \leftarrow \arg\max_{S_T} S_T(\hat{\pi}_T)$
Backtracking
$\pi_t^* \leftarrow Q_{t+1}(\hat{\pi}_{t+1}),\ t = T-1, \dots, 1$

Application to Visual Speech Synthesis

Brand [32] applied HMMs to speech animation. Facial features were represented by facial landmarks tracked on the face of the speaker. Audio data was represented as a combination of LPC and RASTA-PLP features. An entropic HMM [33], which includes

a prior for model sparsity was trained using maximum-a-posteriori (MAP) estimation on training data comprising of facial features and their first-order derivatives. The Gaussian clusters underlying the HMM were then remapped to the audio data by using the occupancy matrix $\boldsymbol{\gamma}$ computed from the visual data, to reestimate the means and covariances of each cluster. Taking $\mathbf{Y} = \{\mathbf{y}_t\}_{t=1}^T$ to be the audio features and $\mathbf{Z} = \{\mathbf{z}_t\}_{t=1}^T$ to be the visual features. The means and variances of the audio clusters, $\boldsymbol{\mu}_{y,k} and \boldsymbol{\Sigma}_{y,k}$, respectively, are estimated as follows:

$$\boldsymbol{\mu}_{y,k} = \frac{\sum_{t=1}^T \gamma_{tk}\mathbf{y}_t}{\sum_{t=1}^T \gamma_{tk}} \tag{4.11}$$

$$\boldsymbol{\Sigma}_{y,k} = \frac{\sum_{t=1}^T \gamma_{tk}(\mathbf{y}_t - \boldsymbol{\mu}_k)(\mathbf{y}_t - \boldsymbol{\mu}_k)^T}{\sum_{t=1}^T \gamma_{tk}} \tag{4.12}$$

The re-mapping is based on the assumption that the HMM model reflects the underlying structure of speech, which is catered for by using the entropic priors that maximise the model's compactness. The remapped audio HMM can be used to infer the states for novel audio data using the Viterbi algorithm. Because the observations in each state of a HMM are conditionally independent from each other, there is no model of local dynamics, and thus a trajectory algorithm is necessary for synthesis. The state indices are used in conjunction with the visual HMM to generate a smooth trajectory in the space of visual parameters using an "inverse Viterbi" algorithm. The algorithm is derived in the ML sense by differentiating the log likelihood with respect to the visual features (and first-order derivatives), $\mathbf{z}_t$, and setting it to zero. This results in a block-banded system of linear equations which can be solved in closed-form using LU-decomposition [124]. The approach described above has two main limitations. The first one is that the remapping of the visual HMM to the audio space assumes that both the visual and audio parameters have the same underlying structure. Using entropic HMMs [33] helps to enforce this constraint because sparsity of the models are ensured. However, from our experiments, it was found that the re-mapping does not always work as expected with a lot of the underlying clusters overlapping with each other when remapped to the audio space. The second limitation is that the trajectory synthesis for mapping cluster indices to continuous visual trajectories tends to oversmooth the animation, leading to under-articulation.

Another limitation of HMMs is that they exhibit state conditional independence, i.e. the observations at time t are conditionally independent from previous observations and previous states given the current state. This property makes HMMs suitable for performing recognition from data that exhibits continuous time-series evolution, for e.g. speech recognition. However, they are not ideal for synthesis. A trajectory HMM [325], which explicitly encodes the relationship between static and dynamic parameters

has been proposed when synthesis is required. The trajectory HMM is the state-of-the-art method in text-to-speech synthesis (TTS) [324]. Recently, there has been increased interest in adopting a similar approach to visual speech synthesis [125, 140, 142, 10]. The main disadvantage of HMM-based speech synthesis is that the audio or visual parameters generated are over-smoothed, resulting in under-articulation [262]. Wang et al. [304] addressed this problem by proposing a Minimum Generation Error (MGE) HMM-based visual speech synthesis approach. Context-dependent HMMs were trained by first using decision tree clustering on fixed phonetic contexts, in order to cater for sparsity problem when having contexts of fixed length. The MGE approach addresses the under-articulation problem in HMM-based speech synthesis by adapting its parameters such that speech trajectories generated by the HMM match ground truth trajectories more closely. This approach was used in conjunction with sample-based visual speech synthesis by guiding unit selection using the generated trajectory [302] and won the LIPS visual speech synthesis challenge [281]. In this thesis, we propose an alternative way to deal with the under-articulation problem by using non-parametric models of Machine Learning which explicitly model speech dynamics. However, we limit ourselves to *learning-based approaches* and attempt to solve research problems related to audio-visual mapping at the expense of having lower fidelity animations than can be obtained using *sample-based approaches*.

4.2.4 Linear Dynamical System

The linear dynamical system (LDS) is more appropriate for synthesis because the states are continuous and there is a dynamical mapping from the previous state to the next. This gives rise to state-space equations that generate the observed data from the states. The LDS has a similar graphical model to the HMM (Figure 4.5) with the difference that the latent states are continuous rather than discrete. The LDS is suitable in applications where a hidden process generates a state sequence, of which a transformed representation is observable. For example, in the case of a moving robot, the hidden process is the locomotion of the robot actuators and the observations consist of a stream of images. The LDS can be used to track the robot coordinates by inferring the hidden state sequence from the set of images.

The LDS is a generative model in which the observations $\{\mathbf{y}_t\}_{t=1}^T$ are generated from the states $\{\mathbf{x}_t\}_{t=1}^T$ through the following state-space equations:

$$\mathbf{x}_1 = \boldsymbol{\mu} + \mathbf{u} \tag{4.13}$$

$$\mathbf{x}_t = \mathbf{A}\mathbf{x}_{t-1} + \boldsymbol{\nu} + \mathbf{v}_t \tag{4.14}$$

$$\mathbf{y}_t = \mathbf{B}\mathbf{x}_t + \mathbf{w}_t \tag{4.15}$$

where $\mathbf{A}$ is the prediction matrix that maps previous states to future states, $\mathbf{B}$ is

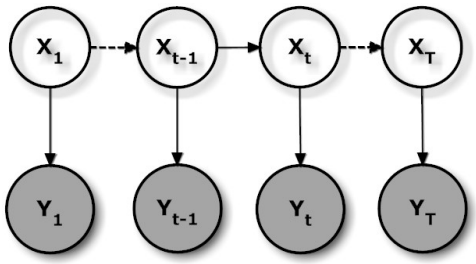


Figure 4.5: Graphical model for linear dynamical system and Gaussian process dynamical model.

the observation matrix that transforms latent states to observations, $\boldsymbol{\mu}$ and $\boldsymbol{\nu}$ are fixed offsets for the initial and subsequent states respectively. $\mathbf{u}$, $\mathbf{v}_t$ and $\mathbf{w}_t$ are the noise parameters following Gaussian distributions as given below:

$$\mathbf{u} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Lambda}) \tag{4.16}$$

$$\mathbf{v}_t \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}) \tag{4.17}$$

$$\mathbf{w}_t \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Gamma}) \tag{4.18}$$

The LDS parameters are $\boldsymbol{\theta} = \{\mathbf{A}, \mathbf{B}, \boldsymbol{\Sigma}, \boldsymbol{\Gamma}, \boldsymbol{\Lambda}, \boldsymbol{\mu}, \boldsymbol{\nu}\}$. The likelihood of the LDS is given by:

$$p(\mathbf{Y}, \mathbf{X}|\boldsymbol{\theta}) = p(\mathbf{x}_1|\boldsymbol{\mu}, \boldsymbol{\Lambda}) \prod_{t=2}^{T} p(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{A}, \boldsymbol{\Sigma}, \boldsymbol{\nu}) p(\mathbf{y}_t|\mathbf{x}_t, \mathbf{B}, \boldsymbol{\Gamma}) \tag{4.19}$$

Training

The aim of LDS training is to estimate the model parameters, $\boldsymbol{\theta}$. These parameters can be estimated using ML through the EM-algorithm [75]. In the E-step, local posterior marginals for the latent variables need to be determined, which can be done using the sum-product algorithm [173]. In the M-step, the parameters are re-estimated using the computed marginals. The E and M steps are repeated until convergence of the likelihood as given by Eqn.4.19.

Inference

The Kalman filter algorithm [163] is used for inference. The Kalman filter consists of two steps: predict and update. In the predict step, the current estimated state, $\hat{\mathbf{x}}_t$ is used to predict the next state, $\hat{\mathbf{x}}_{t+1}$ using the prediction matrix $\mathbf{A}$. This is followed by using the observation matrix $\mathbf{B}$ to obtain $\hat{\mathbf{y}}_{t+1}$. In the update step, the actual observation $\mathbf{y}_{t+1}$ is used to correct the estimated state $\hat{\mathbf{x}}_{t+1}$.

Application to Visual Speech Synthesis

The LDS has been applied to visual speech synthesis in [258], where the trajectories of visual parameters were represented as a linear transformation of a state-space consist of the concatenation of a deterministic and a stochastic audio speech component. The deterministic speech component can be viewed as the speaker-independent part of the audio speech whilst the stochastic speech component was used to cater for the subtleties associated with a given speaker. Both the deterministic process for speech parameters and the stochastic process for non-speech related parameters was learnt automatically from the training data. The parameters of the deterministic component were estimated using a subspace identification procedure, whereas the parameters of the stochastic component were estimated using a dynamical independent component analysis (ICA) algorithm. For synthesis, the audio speech parameters were used as input to the state-space equations and the system was evolved forward in time before being mapped to the visual space to obtain the corresponding synthetic facial motion trajectories. This evolution of the state-space involved feeding the speech parameters to the deterministic component and drawing random samples from the stochastic component. This method assumes that the speaker-independent part of speech is linear and the non-linearities associated with speaker-specific characteristics were modelled using ICA with a non-Gaussian distribution. However, it also assumes that the visual parameters are related to the speech parameters through a linear transformation, which can be highly approximate and thus reduce the quality of the synthesised visual speech. A better approach at modelling both the audio and visual components is to treat both components as being generated from a shared latent space. This is the approach we adopt and will be discussed in Section 4.3.

4.2.5 Gaussian Process Dynamical Model

The latent variable models discussed so far are *parametric* models, which means that the model can be described using a compact set of parameters and the training data can be discarded when making predictions on test data. Another family of latent variable models make use of the training data when extrapolating on test data and are referred to as *non-parametric* models. Although such models are more expensive in time and space complexity both for training and prediction, they can offer a richer representation of the graphical model and yield more accurate predictions. The Gaussian process latent variable model (GPLVM) [177] is obtained by taking a dual approach to probabilistic PCA [285], where the prior is placed over the mapping from the latent space to the data space, followed by marginalising the mapping and optimising the latent points. By using a Gaussian process (GP) [251] prior over the mapping, non-linear mappings can be incorporated. The Gaussian process dynamical model (GPDM) [299] extends

the GPLVM by having a non-linear dynamics mapping from one latent point to the next and thus has similarities with the linear dynamical system (LDS), except that the mappings are non-linear GPs. The GPDM extended to two observation spaces results in the shared Gaussian process dynamical model (SGPDM) [263] and is the approach we adopt to jointly model audio and visual parameters. Gaussian processes (GPs), which form the basis of the models in question, are first explored before delving into the GPLVM and then the GPDM.

Gaussian Processes

A Gaussian process (GP) is a generalisation of a Gaussian distribution on finite random variables to random functions represented by infinite index sets. By definition, a GP is a collection of random variables, any finite number of which have a joint Gaussian distribution [251]. A GP is completely specified by its mean function and covariance function:

$$f(\mathbf{x}) \sim \mathcal{GP}(m(\mathbf{x}), k(\mathbf{x}, \mathbf{x}')) \tag{4.20}$$

where the mean function and covariance function are defined as:

$$m(\mathbf{x}) = \mathbb{E}[f(\mathbf{x})] \tag{4.21}$$

$$k(\mathbf{x}, \mathbf{x}') = \mathbb{E}[f(\mathbf{x} - m(\mathbf{x}))f(\mathbf{x}' - m(\mathbf{x}'))] \tag{4.22}$$

The covariance function characterises how the GP varies about its mean function and represents how the outputs vary as a function of the inputs. The class of valid covariance functions is the same as the class of Mercer kernels [210] and thus, the covariance function is usually referred to as the kernel. The squared exponential or radial basis function (RBF) kernel is widely used in the GP literature [251, 266, 250]. The covariance function for the RBF kernel is given by:

$$k(\mathbf{x}_i, \mathbf{x}_j) = \alpha \exp\big(-\frac{\gamma}{2}(\mathbf{x}_i - \mathbf{x}_j)^T(\mathbf{x}_i - \mathbf{x}_j)\big) \tag{4.23}$$

where α is the variance of the kernel and γ is its scale or inverse width.

In practical applications, we do not have any prior knowledge about the mean function $m(\mathbf{x})$, so it is conveniently set to zero.

GPs provide an elegant framework for regression that does not need to have any parametric assumptions about the function that is to be fitted to the data. Given a set of univariate output points, $\mathbf{y} = \{y_n\}_{n=1}^N$ and a set of multivariate input points, $\mathbf{X} = \{\mathbf{x}_n\}_{n=1}^N$, we want to fit a function f to the data so that:

$$y_n = f(\mathbf{x}_n) + \epsilon \quad \epsilon \sim \mathcal{N}(0, \beta^{-1}) \tag{4.24}$$

where β is the precision or inverse variance.

A GP prior is placed on f with zero mean function and covariance function given

by the RBF kernel. By incorporating the noise term in Eqn.4.24 in the RBF kernel, the following kernel is obtained:

$$k(\mathbf{x}_i, \mathbf{x}_j) = \alpha \exp\big(-\frac{\gamma}{2}(\mathbf{x}_i - \mathbf{x}_j)^T(\mathbf{x}_i - \mathbf{x}_j)\big) + \beta^{-1}\delta_{i,j} \tag{4.25}$$

where $\delta_{i,j}$ is the Kronecker delta between i and j.

A marginal likelihood is formulated by integrating over f:

$$p(\mathbf{y}|\mathbf{X}, \mathbf{\Phi}) = \int p(\mathbf{y}|f)p(f|\mathbf{X}, \mathbf{\Phi})\mathrm{d}f \tag{4.26}$$

where $\mathbf{\Phi}$ are the parameters of the GP, which in the case of the RBF kernel include the hyperparameters of the kernel and the variance of the noise term: $\mathbf{\Phi} = [\alpha, \gamma, \beta]$. The GP parameters are obtained using maximum likelihood, typically using gradient-based optimisation methods such as conjugate gradient optimisation [124].

$$\mathbf{\Phi} = \arg\max_{\mathbf{\Phi}} p(\mathbf{y}|\mathbf{X}, \mathbf{\Phi}) \tag{4.27}$$

Once the regression model has been learnt, it can be used to predict function values y_* at previous unseen input points $\mathbf{x}_*$. The predictive distribution of y_* is a Gaussian distribution:

$$p(y_*|\mathbf{x}_*, \mathbf{X}, \mathbf{y}) \sim \mathcal{N}(\mu_*, \sigma_*^2) \tag{4.28}$$

The parameters of the Gaussian distribution can be obtained from the joint distribution between $\mathbf{y}$ and y_*.

$$\begin{bmatrix} \mathbf{y} \\ y_* \end{bmatrix} \sim \left(\mathbf{0}, \begin{bmatrix} k(\mathbf{X}, \mathbf{X}) + \beta^{-1}\mathbf{I} & k(\mathbf{X}, \mathbf{x}_*) \\ k(\mathbf{x}_*, \mathbf{X}) & k(\mathbf{x}_*, \mathbf{x}_*) + \beta^{-1} \end{bmatrix} \right) \tag{4.29}$$

Taking $\mathbf{K} = k(\mathbf{X}, \mathbf{X})$ and $\mathbf{k} = k(\mathbf{X}, \mathbf{x}_*) = k(\mathbf{x}_*, \mathbf{X})^T$, the mean and variance of the predictive distribution can be derived from Eqn.4.29 by conditioning the joint distribution on the training data. The following are the resulting mean and variance of the predictive distribution on test input points.

$$\mu_* = \mathbf{k}^T(\mathbf{K} + \beta^{-1}\mathbf{I})^{-1}\mathbf{y} \tag{4.30}$$

$$\sigma_* = k(\mathbf{x}_*, \mathbf{x}_*) - \mathbf{k}^T(\mathbf{K} + \beta^{-1})^{-1}\mathbf{k} \tag{4.31}$$

In practical applications, however, the output variables would be multivariate, i.e. comprising of vector-valued data, $\mathbf{Y} = \{\mathbf{y}_n\}_{n=1}^N$. The above GP framework can be modified slightly to accommodate multivariate outputs. Instead of having an independent GP for each output, which is computationally very expensive, the output channels can be assumed to be identically distributed, such that the likelihood function is modelled as a product of independent GPs with a shared covariance function (shared hyperparameters $\mathbf{\Phi}$):

$$p(\mathbf{Y}|\mathbf{X}, \mathbf{\Phi}) = \prod_{d=1}^{D} p(\mathbf{y}_{:,d}|\mathbf{X}, \mathbf{\Phi}) \tag{4.32}$$

The predictive distribution, follows a normal distribution given by:

$$p(\mathbf{y}_*|\mathbf{x}_*, \mathbf{X}, \mathbf{Y}) \sim \mathcal{N}(\boldsymbol{\mu}_*, \sigma_*^2\mathbf{I}) \tag{4.33}$$

where $\boldsymbol{\mu}_* = \mathbf{k}^T(\mathbf{K} + \beta^{-1}\mathbf{I})^{-1}\mathbf{Y}$.

Gaussian Process Latent Variable Model

The Gaussian process latent variable model (GPLVM) is a non-linear dimensionality reduction technique proposed by Lawrence [177, 178], and it has the same graphical model as probabilistic PCA (Figure 4.2). Given some observed mean-centered data, $\mathbf{Y} = \{\mathbf{y}_n\}_{n=1}^N \in \mathbb{R}^{N\times D}$, the goal is to obtain the latent points, $\mathbf{X} = \{\mathbf{x}_n\}_{n=1}^N \in \mathbb{R}^{N\times d}$, where $d < D$. The relationship between the data points and the latent points is the same as for GP regression as given by Eqn.4.24. Just like for GP regression, a GP prior is placed over the mapping f, which leads to the likelihood function of Eqn.4.32. By taking the log of the likelihood equation, we obtain the following log-likelihood, which is the sum of D log likelihoods:

$$\ln p(\mathbf{Y}|\mathbf{X}, \boldsymbol{\Phi}) = -\frac{DN}{2}\ln(2\pi) - \frac{D}{2}\ln|\mathbf{K}| - \frac{1}{2}tr(\mathbf{K}^{-1}\mathbf{Y}\mathbf{Y}^T) \tag{4.34}$$

By fixing d and choosing an appropriate initialisation for the latent points $\mathbf{X}$ and the hyperparameters $\boldsymbol{\Phi}$, training the GPLVM involves maximising the log likelihood of Eqn.4.34:

$$\{\mathbf{X}, \boldsymbol{\Phi}\} = \arg\max_{\mathbf{X},\boldsymbol{\Phi}} \ln p(\mathbf{Y}|\mathbf{X}, \boldsymbol{\Phi}) \tag{4.35}$$

The optimisation is performed using scaled conjugate gradient (SCG) [124] descent and each iteration involves an inversion of the kernel matrix $\mathbf{K}$, which makes the computational complexity $O(N^3)$ in the number of data points N. This makes the GPLVM intractable for large datasets. Various sparsification methods have been proposed to deal with this limitation in the GP regression framework, with a unified view presented by Quiñonero-Candela and Rasmussen [245]. These have been incorporated in Neil Lawrence's GP toolbox[1]. In Section 4.5.1, we compare the effectiveness of the different sparse approximation methods in the application of shared Gaussian process dynamical models (SGPDMs) to visual speech synthesis. The three main sparse approximation techniques that have been incorporated in the GPLVM are now described.

Sparse Approximations

The sparse approximations [179] involve augmenting the function values at the training points, $\mathbf{F} \in \mathbb{R}^{N\times D}$, and the function values at the test points, $\mathbf{F}_* \in \mathbb{R}^{\infty\times D}$, with a set of $k \ll N$ inducing variables, also known as active points, $\mathbf{U}_* \in \mathbb{R}^{k\times D}$. Given

[1] http://staffwww.dcs.shef.ac.uk/people/N.Lawrence/gpsoftware.html

that the likelihood of GPs factorises across columns, the training, test and inducing variables can be treated separately for each data dimension, such that: $\mathbf{f} \in \mathbb{R}^{N\times 1}$, $\mathbf{f}_* \in \mathbb{R}^{\infty\times 1}$ and $\mathbf{u}_* \in \mathbb{R}^{k\times 1}$. The joint distribution between the training and test function values becomes:

$$p(\mathbf{f}, \mathbf{f}_*) = \int p(\mathbf{f}, \mathbf{f}_*|\mathbf{u})p(\mathbf{u}) \tag{4.36}$$

where a GP prior is placed over the inducing variables:

$$p(\mathbf{u}) \sim \mathcal{GP}(\mathbf{0}, \mathbf{K}_{\mathbf{u},\mathbf{u}}) \tag{4.37}$$

The unification framework of Quiñonero-Candela and Rasmussen [245] assumes a conditional independence between the training and test function values, given the inducing variables:

$$p(\mathbf{f}, \mathbf{f}_*, \mathbf{u}) = p(\mathbf{f}|\mathbf{u})p(\mathbf{f}_*|\mathbf{u})p(\mathbf{u}) \tag{4.38}$$

where the training conditional is given by:

$$p(\mathbf{f}|\mathbf{u}) \sim \mathcal{GP}(\mathbf{K}_{\mathbf{f},\mathbf{u}}\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-\mathbf{1}}\mathbf{u}, \mathbf{K}_{\mathbf{f},\mathbf{f}} - \mathbf{K}_{\mathbf{u},\mathbf{u}}^{-\mathbf{1}}\mathbf{K}_{\mathbf{u},\mathbf{f}}) \tag{4.39}$$

The various sparse approximation methods use different approximations to the training conditional in order to make training and inference tractable.

The deterministic training conditional (DTC) uses a deterministic approximation to the training conditional where the GP is given by only a mean function with variance set to zero.

The fully independent training conditional (FITC) uses a conditional independence assumption for the training conditional with a *diagonal* covariance.

The partially independent training conditional (PITC) uses a conditional independence assumption for the training conditional with a *block diagonal* covariance.

The number of inducing variables, k, is set manually and is commonly set to 100 [179].

Scaled GPLVM

Grochow et al. [127] proposed a scaled GPLVM, where a scaling parameter w_d is applied to each dimension d of the observed data, so as to balance the contribution of each output dimension in the likelihood function. This is particularly useful if some dimensions have large variances, as compared to others. When dealing with human motion capture data as in [127], some dimensions represent the position of the avatar whilst others represent the joint angles. To balance the weight of each representation in the likelihood function, scaling is applied to the output dimensions. The log-likelihood function then becomes:

$$\ln p(\mathbf{Y}|\mathbf{X}, \mathbf{\Phi}) = -\frac{DN}{2}\ln(2\pi) - \frac{D}{2}\ln|\mathbf{K}| - \frac{1}{2}\sum_{d=1}^{D} w_d^2(\mathbf{Y}_d^T\mathbf{K}^{-1}\mathbf{Y}_d) \tag{4.40}$$

Initialisation

The optimisation of Eqn.4.35 is ill-posed because infinitely many solutions exist if there are no constraints on $\mathbf{X}$ and $\mathbf{\Phi}$. In practice, the training algorithm has to proceed with a suitable initialisation of both the latent points and the kernel hyperparameters. Initialisation of the kernel hyperparameters is a matter of guesswork in the original GPLVM and some researchers have dealt with this by placing non-informative priors on both the kernel hyperparameters and the latent points [290]. In the original formulation of the GPLVM, PPCA was used to initialise the latent space but for some datasets that have highly non-linear structure, PPCA initialisation leads to the GPLVM training algorithm becoming stuck in local minima that does not recover the true embedded space [178]. In such cases, spectral dimensionality reduction techniques such as Isomap [274] can be used for initialisation.

Back-constraints

The GPLVM adopts a generative approach to dimensionality reduction because the mapping f is from the latent space to the data space. This approach ensures that locality in the latent space is preserved in the data space. Spectral dimensionality reduction techniques such as multi-dimensional scaling (MDS) [27], Isomap [274], locally linear embedding (LLE) [257] etc., compute a mapping from the data space to the latent space, which ensures that locality in the data space is preserved in the latent space. Such locality is important if the latent space needs to be clustered or for classification, requiring different classes be separable on the latent space. Lawrence and Quiñonero-Candela [180] introduced the back-constrained GPLVM that enforces the distance preservation constraint of spectral methods by having a parametric mapping b from the data space to the latent space:

$$\mathbf{y}_n = b(\mathbf{x}_n, \boldsymbol{\theta}) \tag{4.41}$$

where $\boldsymbol{\theta}$ are the parameters of the parametric mapping, which can be the weights of a multi-layer perceptron (MLP) [21], the hyperparameters of a kernel-based regression (KBR) [22] model, or any alternative mapping.

Dynamics

The GPLVM can be temporally constrained by introducing a dynamical mapping on the latent space through an autoregressive function mapping h:

$$\mathbf{x}_t = h(\mathbf{x}_{t-1}) + \boldsymbol{\epsilon}_{dyn} \tag{4.42}$$

The mapping h can be a GP and the resulting model is called a Gaussian process dynamical model (GPDM), introduced by Wang et al. [299].

The log-likelihood function for the autoregressive dynamics is given by:

$$\ln p(\mathbf{X}|\mathbf{\Phi}_{dyn}) = \ln\left(p(\mathbf{x}_1)\right) - \frac{(N-1)d}{2}\ln(2\pi) - \frac{d}{2}|\mathbf{K}_X| - \frac{1}{2}tr(\mathbf{K}_X^{-1}\mathbf{X}_{out}\mathbf{X}_{out}^T) \quad (4.43)$$

where d denotes the dimensionality of the latent space, $\mathbf{X}$. $\mathbf{X}_{in} = \{\mathbf{x}_n\}_{n=1}^{N-1}$, $\mathbf{X}_{out} = \{\mathbf{x}_n\}_{n=2}^{N}$, $\mathbf{K}_X$ is an RBF kernel matrix constructed from $\mathbf{X}_{in}$, and $p(\mathbf{x}_1)$ is an isotropic Gaussian prior.

A "RBF+linear" kernel [299] is used for the dynamics GP, so as to cater for subsequences of behaviour that are approximately linear.

$$k_X(\mathbf{x}_i, \mathbf{x}_j) = \alpha_1 \exp\big(-\frac{\gamma}{2}(\mathbf{x}_i - \mathbf{x}_j)^T(\mathbf{x}_i - \mathbf{x}_j)\big) + \alpha_2\mathbf{x}_i^T\mathbf{x}_j + \beta^{-1}\delta_{i,j} \quad (4.44)$$

The log-likelihood of the GPDM is a sum of the log-likelihood of the GPLVM and that of the dynamics GP:

$$\ln p(\mathbf{Y}, \mathbf{X}|\mathbf{\Phi}, \mathbf{\Phi}_{dyn}) = \ln p(\mathbf{Y}|\mathbf{X}, \mathbf{\Phi}) + \ln p(\mathbf{X}|\mathbf{\Phi}_{dyn}) \quad (4.45)$$

Urtasun et al. [290] introduced an exponent to the likelihood of the dynamics GP in order to balance the weight of the GPLVM and dynamics likelihoods. In log space, this exponent becomes a constant that is multiplied to the log-likelihood of the dynamics GP.

$$\ln p(\mathbf{Y}, \mathbf{X}|\mathbf{\Phi}, \mathbf{\Phi}_{dyn}) = \ln p(\mathbf{Y}|\mathbf{X}, \mathbf{\Phi}) + \lambda_{dyn}\ln p(\mathbf{X}|\mathbf{\Phi}_{dyn}) \quad (4.46)$$

where $\lambda_{dyn} = \frac{D}{d}$. This balances the log-likelihood of the dynamics GP with that of the GPLVM by compensating for the difference in the dimensionality between $\mathbf{Y}$ and $\mathbf{X}$, thus giving equal importance to each term in the log-likelihood. Urtasun et al. [290] has shown that this gives smoother latent spaces for human motion capture data.

The dynamical model can be modified slightly to account for multiple sequences $\{\mathbf{Y}_n\}_{n=1}^N$ with associated latent sequences $\{\mathbf{X}_n\}_{n=1}^N$, as demonstrated by Urtasun et al. [290] and Wang et al. [300]. This is done by concatenating together all the frames for each sequence within the GP likelihood (Eqn.4.34) for the observation GP mapping f. For the dynamical mapping h, the first frame of each sequence from $\mathbf{X}_{out}$ is omitted and the last frame for each sequence is omitted from the kernel matrix $\mathbf{K}_X$ in Eqn.4.43. A sequence delimiter vector is used to store the indices of the last frame of each sequence, to be used by the GPDM training algorithm to identify the sequence boundaries from the training data.

Training

Training the GPLVM is done by optimising the log-likelihood function in Eqn.4.34 with respect to the latent points and the hyperparameters of the GP. The data needs to be mean-centered first, such that $\mathbf{Y}_d = \{\mathbf{y}_{n,d} - \mu_d\}_{n=1}^N$, where $\mu_d = \frac{\sum_{n=1}^N \mathbf{y}_{n,d}}{N}$.

In case a back-constraint is used, the optimisation is done for the parameters of the parametric mapping that maps data points to latent points, without needing to optimise the latent points directly, because the latent points are a function of the data points. If a dynamical model is used, the likelihood function in Eqn.4.45 is optimised with respect to the hyperparameters of the GPLVM and dynamics GPs, and the latent points.

If the latent space is 2D or 3D, a visualisation can be obtained for the latent points as well as the likelihood space around the latent points [177]. Brighter regions mean that the GP predictions at those regions are very certain, having low variance, and the opposite is true for darker regions.

Inference

Given a sequence of data points, $\hat{\mathbf{Y}} = \{\hat{\mathbf{y}}\}_{t=1}^{T}$, a learnt GPDM model can be used to infer the latent points, $\hat{\mathbf{X}} = \{\hat{\mathbf{x}}\}_{t=1}^{T}$. This is done by optimising the joint log-likelihood for a sequence of data points and their latent points under the dynamical model, with the hyperparameters of the GPLVM and dynamics GP as well as the training data and latent points given:

$$\mathbf{X}^* = \arg\max_{\hat{\mathbf{X}}} \ln p(\hat{\mathbf{Y}}, \hat{\mathbf{X}} | \mathbf{Y}, \mathbf{X}, \mathbf{\Phi}, \mathbf{\Phi}_{dyn}) \tag{4.47}$$

where $\mathbf{X}^*$ are the optimised latent points. The optimisation is done using SCG, similar as for training.

Experiments on Audio and Visual Data

We trained 2D GPLVM models on AAM and MFCC data for a sequence of the DEMNOW corpus. Figures 4.6a and 4.6b show the latent spaces for MFCC and AAM data respectively. The left plots show the audio latent spaces with different colours and symbols used for different phoneme labels. The right plots, on the other hand, show visual latent spaces with viseme labels, obtained from phoneme labels according to Table 2.1. The phonemes and visemes tend to cluster on the latent space, as can be seen in the figures. Figures 4.6c and 4.6d show the latent spaces when an MLP back-constraint has been used. Using back-constraints tends to move similar phonemes and visemes closer on the latent space, leading to further clustering. Figures 4.6e and 4.6f show the latent spaces when using a GPDM model with autoregressive dynamics. In the plots, lines are used to join points that are adjacent to each other temporally. Figures 4.6g and 4.6h show the latent spaces when using a GPDM model together with MLP back-constraints.

From these plots, it can be seen that training a dynamical model on audio data leads to a lot of jumps in the space, mostly because of the highly non-linear structure of speech. Thus, the dynamical model cannot constrain spatially dissimilar points to

be close on the latent space, even though they are temporally close. For the visual space, however, AAM parameters tend to have a smoother variation through time, due to locally linear substructures, and thus, the dynamical model results in a space that better preserves temporal relationships. Using back-constraints tends to enforce spatial locality by moving similar phonemes and visemes closer on the latent space. For the audio space, points that are temporally close may not be spatially close and vice-versa. This can result in further jumps in the latent space from one point to the next. For the visual space, spatial and temporal locality seem to match, resulting in a smoother path through the latent points when back-constraints and dynamics are both used.

4.3 Shared Latent Variable Models

The latent variable models considered in the previous section are for a single multivariate variable. In our case, we have two representations of human speech, namely the audio and visual parameters. We are thus interested in learning shared latent variable models, whereby the audio and visual parameters are generated from a shared latent space. In this section, we describe two parametric latent variable models, namely probabilistic canonical correlation analysis (PCCA) and the shared space LDS, which is an extension of the LDS described in the previous section. We then describe a shared latent space extension of the GPDM model, called the shared GPDM (SGPDM), which is the non-parametric state-space model that we propose to jointly model audio and visual data.

4.3.1 Canonical Correlation Analysis

Canonical correlation analysis (CCA) [145] can be used to find directions of maximal correlation between two variables $\mathbf{Y}$ and $\mathbf{Z}$.

Given two sets of mean-centered variables, $\mathbf{Y} = \{\mathbf{y}_n\}_{n=1}^N \in \mathbb{R}^{d_Y \times N}$ and $\mathbf{Z} = \{\mathbf{z}_n\}_{n=1}^N \in \mathbb{R}^{d_Z \times N}$, canonical correlation analysis (CCA) is a technique proposed by Hotelling [145] to learn two sets of basis vectors $\mathbf{W} = \{\mathbf{w}_i \in \mathbb{R}^{d_Y \times 1}\}_{i=1}^d$ and $\mathbf{V} = \{\mathbf{v}_i \in \mathbb{R}^{d_Z \times 1}\}_{i=1}^d$, where $d \leq min(d_Y, d_Z)$, such that the projections $\hat{\mathbf{Y}} = \{\hat{\mathbf{y}}_{:,i} = \mathbf{Y}^T\mathbf{w}_i\}_{i=1}^d$ and $\hat{\mathbf{Z}} = \{\hat{\mathbf{z}}_{:,i} = \mathbf{Z}^T\mathbf{v}_i\}_{i=1}^d$ are maximally correlated. The correlation coefficients are given by:

$$\rho_i = \frac{< \hat{\mathbf{y}}_{:,i}, \hat{\mathbf{z}}_{:,i} >}{\| \hat{\mathbf{y}}_{:,i} \| \| \hat{\mathbf{z}}_{:,i} \|} \tag{4.48}$$

where $\rho_1 > \rho_2 \ldots > \rho_d$, and $(\mathbf{w}_i, \mathbf{v}_i)$ are the canonical vectors.

This is formulated as a constraint optimisation problem:

$$\begin{aligned} &\arg\max_{\mathbf{w}_i, \mathbf{v}_i} \mathbf{w}_i^T \mathbf{Y}^T \mathbf{Z} \mathbf{v}_i \\ \text{subject to} \quad &\mathbf{w}_i^T \mathbf{Y}^T \mathbf{Z} \mathbf{w}_i = \mathbf{v}_i^T \mathbf{Y}^T \mathbf{Z} \mathbf{v}_i = 1 \end{aligned} \tag{4.49}$$

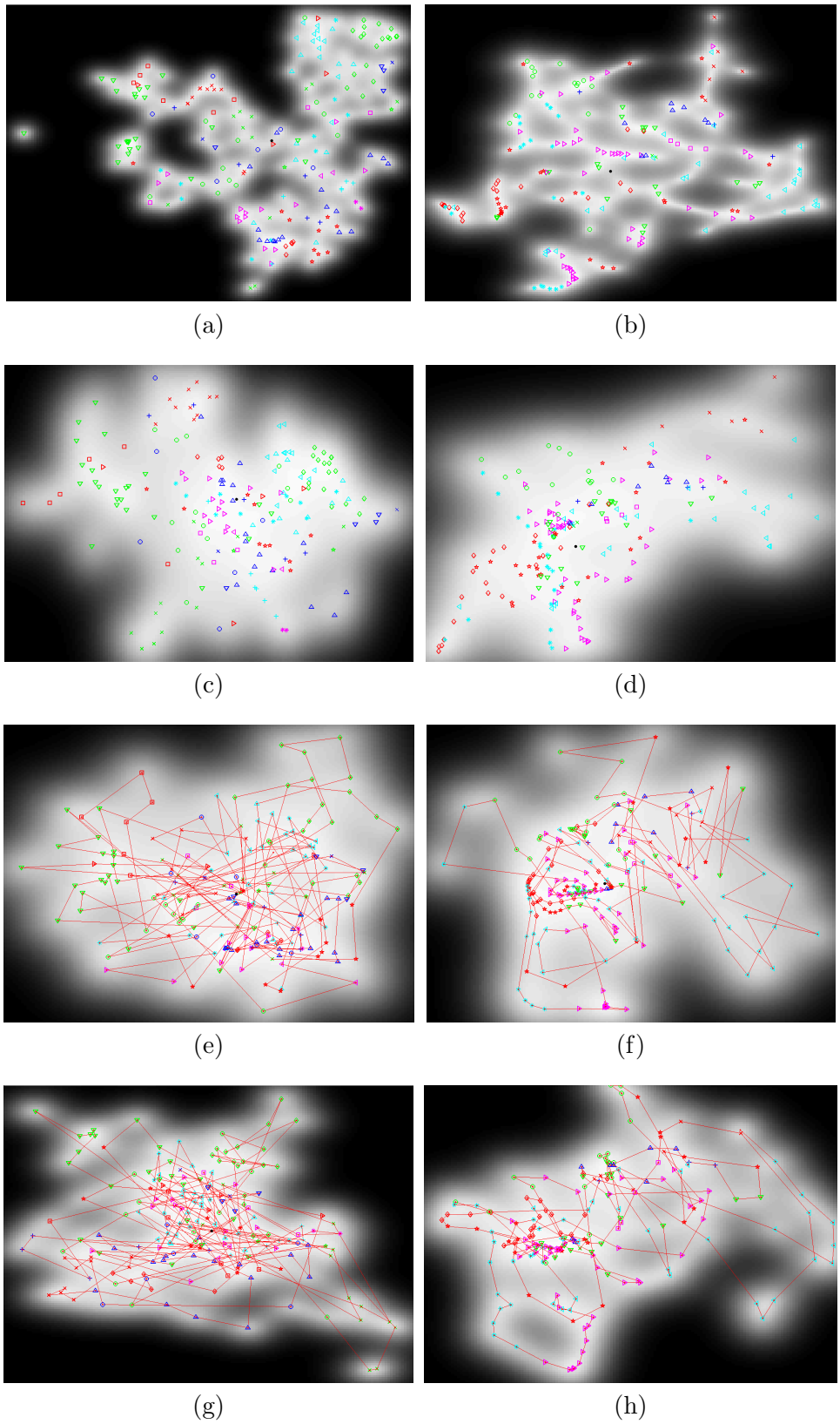


Figure 4.6: Latent spaces on MFCC or AAM data: Left - MFCC with phoneme labels, Right - AAM with viseme labels: (a) MFCC GPLVM, (b) AAM GPLVM, (c) MFCC GPLVM with MLP back-constraints, (d) AAM GPLVM with MLP back-constraints, (e) MFCC GPDM, (f) AAM GPDM, (g) MFCC GPDM with MLP back-constraints, (h) AAM GPDM with MLP back-constraints.

The maximisation of Eqn.4.49 leads to the following eigenvector equations:

$$(\mathbf{Y^T Y})^{-1}\mathbf{Y^T Z}(\mathbf{Z^T Z})^{-1}\mathbf{Y^T Z}\mathbf{w}_i = \lambda_i^2 \mathbf{w}_i \tag{4.50}$$

$$(\mathbf{Z^T Z})^{-1}\mathbf{Z^T Y}(\mathbf{Y^T Y})^{-1}\mathbf{Z^T Y}\mathbf{v}_i = \lambda_i^2 \mathbf{v}_i \tag{4.51}$$

Bach and Jordan [8] have given a shared latent representation of CCA called probabilistic CCA (PCCA) by adopting a probabilistic formulation, with the probabilistic graphical model shown in Figure 4.7. The mappings $\mathbf{W}^z$ and $\mathbf{W}^y$ are now from the latent space to the data spaces. The likelihood is given by:

$$p(\mathbf{Y}, \mathbf{Z}, \mathbf{X}|\mathbf{W}^y, \mathbf{W}^z, \boldsymbol{\mu}^z, \boldsymbol{\mu}^z, \boldsymbol{\Sigma}^z, \boldsymbol{\Sigma}^y) = p(\mathbf{Y}|\mathbf{X}, \mathbf{W}^y, \boldsymbol{\mu}^y, \boldsymbol{\Sigma}^y)p(\mathbf{Z}|\mathbf{X}, \mathbf{W}^z, \boldsymbol{\mu}^z, \boldsymbol{\Sigma}^z)p(\mathbf{X}) \tag{4.52}$$

where:

$$p(\mathbf{X}) \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_d) \tag{4.53}$$

$$p(\mathbf{Z}|\mathbf{X}, \mathbf{W}^z, \boldsymbol{\mu}^z, \boldsymbol{\Sigma}^z) \sim \mathcal{N}(\mathbf{W}^z\mathbf{X} + \boldsymbol{\mu}^z, \boldsymbol{\Sigma}^z) \tag{4.54}$$

$$p(\mathbf{Y}|\mathbf{X}, \mathbf{W}^y, \boldsymbol{\mu}^y, \boldsymbol{\Sigma}^y) \sim \mathcal{N}(\mathbf{W}^y\mathbf{X} + \boldsymbol{\mu}^y, \boldsymbol{\Sigma}^y) \tag{4.55}$$

The ML estimates of the canonical coefficients can be found in closed-form with more details in [8].

In this work, PCCA has been used as one of the initialisation methods to initialise the latent space when training our proposed joint probabilistic models of audio and video (refer to Section 4.5.1).

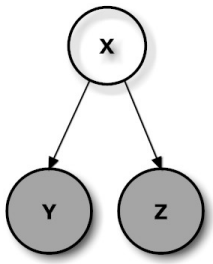


Figure 4.7: Graphical model for probabilistic canonical correlation analysis and shared Gaussian process latent variable model.

Application to Visual Speech Synthesis

Theobald and Wilkinson [279] used a combination of linear regression and CCA to model the audio and visual parameters of a talking face with the aim of synthesising facial animation. The limitation of CCA when applied to visual speech synthesis is that it does not incorporate dynamics and thus fails to model coarticulation. In order to cater for this, Theobald and Wilkinson [279] appended four frames to the left and right of each audio and visual frame prior to modelling with CCA.

4.3.2 Coupled Hidden Markov Model

The coupled hidden Markov model (CHMM) was introduced by Brand et al. [30] for modelling interacting processes where two representations of an underlying process, $\mathbf{Y} = \{\mathbf{y}_t\}_{t=1}^T$ and $\mathbf{Z} = \{\mathbf{z}_t\}_{t=1}^T$, are assumed to share a set of coupled states $\boldsymbol{\pi}^y \times \boldsymbol{\pi}^z$ according to the graphical model shown in Figure 4.8. The coupled states can be factorised into states for each interacting process, $\boldsymbol{\pi}^y = \{\pi_t^y\}_{t=1}^T$ and $\boldsymbol{\pi}^z = \{\pi_t^z\}_{t=1}^T$ respectively. Thus, a coupled HMM can be regarded as a collection to two separate HMM chains coupled through cross-time and cross-chain conditional probabilities. A CHMM defines four transition probability matrices: $p(\boldsymbol{\pi}_t^y|\boldsymbol{\pi}_{t-1}^y, \mathbf{A})$, $p(\boldsymbol{\pi}_t^z|\boldsymbol{\pi}_{t-1}^z, \mathbf{B})$, $p(\boldsymbol{\pi}_t^y|\boldsymbol{\pi}_{t-1}^z, \mathbf{C})$ and $p(\boldsymbol{\pi}_t^z|\boldsymbol{\pi}_{t-1}^y, \mathbf{D})$. The likelihood of the CHMM is given in Eqn.4.56.

$$\begin{aligned} p(\mathbf{Y}, \mathbf{Z}, \boldsymbol{\pi}^y, \boldsymbol{\pi}^z|\boldsymbol{\Theta}) &= p(\pi_1^y|\mathbf{s}^y)p(\pi_1^z|\mathbf{s}^z) \\ &\times \left[\prod_{t=2}^{T} p(\pi_t^y|\pi_{t-1}^y, \mathbf{A})p(\pi_t^z|\pi_{t-1}^z, \mathbf{B})p(\pi_t^y|\pi_{t-1}^z, \mathbf{C})p(\pi_t^z|\pi_{t-1}^y, \mathbf{D})\right] \\ &\times \left[\prod_{t=1}^{T}\sum_{k=1}^{K^y} m_k^y \mathcal{N}(\mathbf{y}_t|\boldsymbol{\mu}_k^y, \boldsymbol{\Sigma}_k^y)\sum_{k=1}^{K^z} m_k^z \mathcal{N}(\mathbf{z}_t|\boldsymbol{\mu}_k^z, \boldsymbol{\Sigma}_k^z)\right] \end{aligned} \tag{4.56}$$

where $\boldsymbol{\Theta} = [\mathbf{s}^y, \mathbf{s}^z, \mathbf{m}^y, \mathbf{m}^z, \boldsymbol{\mu}^y, \boldsymbol{\Sigma}^y, \boldsymbol{\mu}^z, \boldsymbol{\Sigma}^z, \mathbf{A}, \mathbf{B}, \mathbf{C}, \mathbf{D}]$

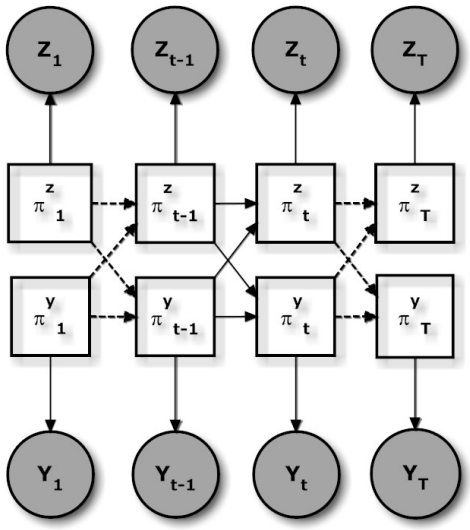


Figure 4.8: Graphical model for coupled hidden Markov model.

Training

The CHMM can be trained using the EM algorithm [75] to maximise the likelihood of Eqn.4.56. The E-step involves running the forward-backward [248] algorithm on both chains to obtain marginal distributions over the states and transitions. The M-step then re-estimates all the parameters of the CHMM.

Inference

Just like for the HMM, the Viterbi algorithm [293] can be used to perform inference either on the chains separately, by treating each chain as a separate HMM or using the coupled states. Moreover, both streams can be used to infer the coupled states using a modified Viterbi algorithm described in [31].

Application to Visual Speech Synthesis

A coupled HMM was used by Xie and Liu [312] to model the audio and visual streams separately with coupled states having a mapping on both data streams. A Baum-Welch audio-visual inversion algorithm was presented to predict visual from audio parameters, inspired from the HMM inversion method of Choi et al. [50]. This approach bypasses the need for an interpolation from sub-optimal Viterbi states by instead using the full occupancy of the states in synthesis. However, there is no dynamics model in the CHMM and the synthesis technique does not explicitly model coarticulation. In Xie and Liu [312] coarticulation was modelled by using speech features concatenated with their velocities and accelarations. Since the Baum-Welch audio-visual inversion method uses the full audio information for prediction, only backward context was taken into account.

4.3.3 Shared Linear Dynamical System

The linear dynamical system (LDS) presented in Section 4.2.4 has been extended by Lehn-Schiøler et al. [186] to cater for two data spaces generated from a shared latent space. Lehn-Schiøler et al. [186] applied this framework to model audio and visual modalities of a talking face. Taking $\mathbf{Y} = \{\mathbf{y}_t\}_{t=1}^T$ to be the audio data and $\mathbf{Z} = \{\mathbf{z}_t\}_{t=1}^T$ to be visual data, such that the audio and visual data are synchronised, the graphical model of the shared LDS is shown in Figure 4.9 and the state-space equations are:

$$\mathbf{x}_1 = \boldsymbol{\mu} + \mathbf{u} \tag{4.57}$$

$$\mathbf{x}_t = \mathbf{A}\mathbf{x}_{t-1} + \boldsymbol{\nu} + \mathbf{v}_t \tag{4.58}$$

$$\mathbf{y}_t = \mathbf{B}\mathbf{x}_t + \mathbf{w}^{\mathbf{y}}{}_t \tag{4.59}$$

$$\mathbf{z}_t = \mathbf{C}\mathbf{x}_t + \mathbf{w}^{\mathbf{z}}{}_t \tag{4.60}$$

with the noise parameters following Gaussian distributions as follows:

$$\mathbf{u} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Lambda}) \tag{4.61}$$

$$\mathbf{v}_t \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}) \tag{4.62}$$

$$\mathbf{w}^{\mathbf{y}}{}_t \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Gamma}^{\boldsymbol{y}}) \tag{4.63}$$

$$\mathbf{w}^{\mathbf{z}}{}_t \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Gamma}^{\boldsymbol{z}}) \tag{4.64}$$

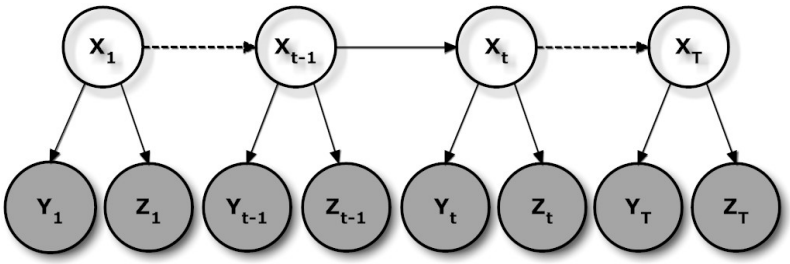


Figure 4.9: Graphical model for shared linear dynamical system and shared Gaussian process dynamical model.

Training

The shared LDS parameters, $\boldsymbol{\theta} = \{\mathbf{A}, \mathbf{B}, \mathbf{C}, \boldsymbol{\Sigma}, \boldsymbol{\Gamma}^{\boldsymbol{y}}, \boldsymbol{\Gamma}^{\boldsymbol{z}}, \boldsymbol{\Lambda}, \boldsymbol{\mu}, \boldsymbol{\nu}\}$, can be estimated using the EM algorithm [75], just like for the LDS.

Inference

Given a stream of test data, $\{\hat{\mathbf{y}}_t\}_{t=1}^T$ or $\{\hat{\mathbf{z}}_t\}_{t=1}^T$, Kalman filtering [163] can be used to infer the latent states, $\{\hat{\mathbf{x}}_t\}_{t=1}^T$.

Application to Visual Speech Synthesis

In Lehn-Schiøler [185] and Lehn-Schiøler et al. [186], $\mathbf{Y}$ was taken to be the audio parameters and $\mathbf{Z}$ was taken to be the visual parameters. After training, the shared latent points $\mathbf{X}$ were inferred from the model using Kalman filtering on the audio LDS factored from the shared LDS. The factored LDS corresponds to Eqn.4.57, Eqn.4.58 and Eqn.4.59. From the latent states, the visual data can be reconstructed as: $\hat{\mathbf{z}}_t = \mathbf{C}\hat{\mathbf{x}}_t$. The advantage of the shared LDS is that synthesis can be performed if the states are known by making use of the state-space equations without needing a trajectory synthesis algorithm as in the case of the HMM [32]. Compared to the CHMM [312], the shared LDS has an explicit model of backward coarticulation due to the autoregressive dynamics. However, both the dynamics and the mapping from the states to the observations are linear, thus not catering for the non-linear dynamics of speech [12] as well as the non-linear mapping from audio to visual speech. The shared Gaussian process dynamical model (SGPDM) addresses these limitations.

4.3.4 Shared Gaussian Process Dynamical Model

The shared Gaussian process dynamical model (SGPDM) [98] is a non-parametric and non-linear dynamical system with two observation spaces instead of one as is the case for the GPDM [299]. Instead of having linear mappings as in the case of the

shared LDS, the SGPDM models the prediction and observation mappings as non-linear Gaussian processes. Given two data spaces, $\mathbf{Y} = \{\mathbf{y}_t\}_{t=1}^T$ and $\mathbf{Z} = \{\mathbf{z}_t\}_{t=1}^T$ and assuming they are generated from a shared latent space, $\mathbf{X} = \{\mathbf{x}_t\}_{t=1}^T$, the state-space equations of the SGPDM are:

$$\mathbf{x}_1 \sim \mathcal{N}(\mathbf{0}, \beta_{dyn}^{-1}\mathbf{I}) \tag{4.65}$$

$$\mathbf{x}_t = h(\mathbf{x}_{t-1}) + \boldsymbol{\epsilon}_{dyn} \quad \boldsymbol{\epsilon}_{dyn} \sim \mathcal{N}(\mathbf{0}, \beta_{dyn}^{-1}\mathbf{I}) \tag{4.66}$$

$$\mathbf{y}_t = f(\mathbf{x}_t) + \boldsymbol{\epsilon}_y \quad \boldsymbol{\epsilon}_y \sim \mathcal{N}(\mathbf{0}, \beta_Y^{-1}\mathbf{I}) \tag{4.67}$$

$$\mathbf{z}_t = g(\mathbf{x}_t) + \boldsymbol{\epsilon}_z \quad \boldsymbol{\epsilon}_z \sim \mathcal{N}(\mathbf{0}, \beta_Z^{-1}\mathbf{I}) \tag{4.68}$$

where f and g are GP mappings from the latent space $\mathbf{X}$ to the observation spaces $\mathbf{Y}$ and $\mathbf{Z}$, respectively, and h is the GP mapping for the autoregressive dynamics. $\boldsymbol{\epsilon}_y$, $\boldsymbol{\epsilon}_z$ and $\boldsymbol{\epsilon}_{dyn}$ are the noise terms for the observation and dynamical GPs, which are drawn from a zero-mean Gaussian with isotropic covariance matrix. The variance of the noise terms are given as the inverse of precision.

The graphical model of the SGPDM is the same as for the shared LDS and is shown in Figure 4.9.

The joint log-likelihood of the resulting model is given by:

$$\ln p(\mathbf{Y}, \mathbf{Z}, \mathbf{X}|\boldsymbol{\Phi}) = \ln p(\mathbf{Y}|\mathbf{X}, \boldsymbol{\Phi}_Y) + \ln p(\mathbf{Z}|\mathbf{X}, \boldsymbol{\Phi}_Z) + \ln p(\mathbf{X}|\boldsymbol{\Phi}_{dyn}) \tag{4.69}$$

where $\boldsymbol{\Phi} = [\boldsymbol{\Phi}_Y, \boldsymbol{\Phi}_Z, \boldsymbol{\Phi}_{dyn}]$ is a concatenation of the hyperparameters of the GPs for $\mathbf{Y}$, $\mathbf{Z}$ and the dynamics.

Shared Gaussian Process Latent Variable Model

The shared Gaussian process latent variable model, originally proposed by Shon et al. [263], is an extension of the GPLVM to two variables instead of one but without the dynamical model of the SGPDM. The original SGPLVM of Shon et al. [263] dealt with learning correspondences between images of two different objects, with the orientation being similar for any paired images. Moreover, it was also applied to robotic imitation, which involves learning a correspondence between 3D poses of human motion capture and that of robot actuators. Ek et al. [98, 99] applied the framework for inferring 3D pose from 2D silhouettes and introduced dynamical models to cater for sequences of pose and silhouettes that are temporally aligned.

Just as for the GPLVM, scaling can be applied to the output dimensions of each data space, so as to give equal weight to each dimension in the likelihood function (refer to Section 4.2.5). In this work, we use scaling of outputs when jointly modelling audio and visual data, because the different audio and visual parameter dimensions have different variances.

Initialisation of Latent Points

In the original formulation of the SGPLVM, Shon et al. [263] initialised the latent space, $\mathbf{X}$, as the average of the first d principal components of $\mathbf{Y}$ and $\mathbf{Z}$, where d is the latent dimensionality being used for the SGPLVM. In Ek et al. [99], an extension of Kernel CCA [176] called non-consolidating component analysis (NCCA) was introduced that learns a shared latent space, as well as private latent spaces for both $\mathbf{Y}$ and $\mathbf{Z}$, so as to retain the variance corresponding to each data space. The variance corresponding to the data space of the test data can be used during inference, thus resolving ambiguities that arise due to one-to-many mappings from the test data space to the inferred data space. These ambiguites arise when mapping from silhouettes to 3D pose. For example in silhouettes, cases when the left foot is forward and the right foot backward are indistinguishable from those where the right foot is forward and the left foot backward.

In this work, we use averaged PPCA subspaces between audio and visual data, as well as the PCCA subspace obtained from both spaces, as the latent space initialisation method for training the SGPDM. We perform experiments to compare both approaches in Section 4.5.1.

Likelihood Bias

We propose a likelihood bias to account for $\mathbf{Y}$ and $\mathbf{Z}$ having different dimensionalities, which results in different scales in the likelihoods $P(\mathbf{Y}|\mathbf{X}, \mathbf{\Phi}_Y)$ and $P(\mathbf{Z}|\mathbf{X}, \mathbf{\Phi}_Z)$. This can result in the SGPDM model being biased towards the variable with the higher dimensionality because the contribution of that variable in the log-likelihood (Eqn.4.69) is higher than the other variable. In order to account for this, we adopt an approach similar to the dynamics balancing of Urtasun et al. [290], where an exponent was placed on the dynamics likelihood of the GPDM in order to balance the weight of the dynamics likelihood with the GPLVM likelihood. We introduce a likelihood bias to give equal weight to each data space according to the following rules, where $\lambda_Y = \frac{D_Z}{D_Y}$ and $\lambda_Z = \frac{D_Y}{D_Z}$.

if $D_Y > D_Z$ **then**

$\ln p(\mathbf{Y}, \mathbf{Z}, \mathbf{X}|\mathbf{\Phi}) = \ln p(\mathbf{Y}|\mathbf{X}, \mathbf{\Phi}_Y) + \lambda_Z \ln p(\mathbf{Z}|\mathbf{X}, \mathbf{\Phi}_Z) + \ln p(\mathbf{X}|\mathbf{\Phi}_{dyn})$

end if

if $D_Z > D_Y$ **then**

$\ln p(\mathbf{Y}, \mathbf{Z}, \mathbf{X}|\mathbf{\Phi}) = \lambda_Y \ln p(\mathbf{Y}|\mathbf{X}, \mathbf{\Phi}_Y) + \ln p(\mathbf{Z}|\mathbf{X}, \mathbf{\Phi}_Z) + \ln p(\mathbf{X}|\mathbf{\Phi}_{dyn})$

end if

Back-constraints

Just like for the GPLVM, back-constraints can be used for the SGPLVM by having a parametric mapping from either $\mathbf{Y}$ or $\mathbf{Z}$ to $\mathbf{X}$. Having mappings from both $\mathbf{Y}$ and $\mathbf{Z}$ would lead to two different back-projections for $\mathbf{X}$, which is undesirable.

In Ek et al. [98], where a SGPLVM was used to model 3D pose and 2D silhouettes, a back-constraint with respect to pose data was used in order to enforce a one-to-one mapping from the latent space to the pose space, and thus a one-to-many mapping from silhouette space to 3D pose space. This helped to resolve ambiguities in the inference of 3D pose from silhouettes.

Training

The model is trained by placing GP priors over f, g and h and optimising the log-likelihood in Eqn.4.69 with respect to the latent points and the hyperparameters:

$$\{\mathbf{X}, \mathbf{\Phi}\} = \arg\max_{\mathbf{X},\mathbf{\Phi}} \ln p(\mathbf{Y}, \mathbf{Z}, \mathbf{X}|\mathbf{\Phi}) \tag{4.70}$$

Just as for the GPDM, each data space is first mean-centered before training the model. Extension for multiple sequences is the same as for the GPDM (refer to Section 4.2.5).

Inference

One approach to the inference problem for the SGPDM is Bayesian inference, making use of Bayes' theorem, which can be stated as follows:

$$\text{Posterior} = \frac{\text{Likelihood} \times \text{Prior}}{\text{Evidence}} \tag{4.71}$$

Taking $\hat{\mathbf{Y}}$ to be a vector of test data, $\hat{\mathbf{Z}}$ to be the vector of data to be inferred and $\hat{\mathbf{X}}$ to be the shared latent space, we are interested in the predictive distribution $p(\hat{\mathbf{Z}}|\hat{\mathbf{Y}})$. First, the predictive distribution of the latent points given the test data is inferred using Bayes theorem:

$$p(\hat{\mathbf{X}}|\hat{\mathbf{Y}}) = \frac{p(\hat{\mathbf{Y}}|\hat{\mathbf{X}})p(\hat{\mathbf{X}})}{\int p(\hat{\mathbf{Y}}|\hat{\mathbf{X}})p(\hat{\mathbf{X}})d\hat{\mathbf{X}}} \tag{4.72}$$

where $p(\hat{\mathbf{Y}}|\hat{\mathbf{X}})$ is given by the GP mapping from the latent space to the first observation space and $p(\hat{\mathbf{X}})$ is the prior on the latent space.

The conditional distribution $p(\hat{\mathbf{Z}}|\hat{\mathbf{Y}})$ is then obtained by integrating over the latent points whilst multiplying the conditional distribution of $\hat{\mathbf{Z}}$ given the latent points and the predictive distribution of the latent points given $\hat{\mathbf{Y}}$:

$$p(\hat{\mathbf{Z}}|\hat{\mathbf{Y}}) = \int p(\hat{\mathbf{Z}}|\hat{\mathbf{X}})p(\hat{\mathbf{X}}|\hat{\mathbf{Y}}) \tag{4.73}$$

where $p(\hat{\mathbf{Z}}|\hat{\mathbf{X}})$ is given by the GP mapping from the latent space to $\hat{\mathbf{Z}}$.

In principle, the inference involved in Eqn.4.72 could be done by deriving forward and backward recursions using the sum-product algorithm for graphical models [173]. However, this is intractable both for the SGPDM and approximate inference algorithms have recently been proposed for Gaussian process dynamical systems [170, 72, 289].

Instead of a Bayesian approach, Ek et al. [98, 99] adopted a maximum likelihood (ML) approach, where a point estimate of $\hat{\mathbf{X}}$ was found by using gradient descent with an initial estimate of $\mathbf{X}^*$ obtained using an appropriate initialisation method.

$$\mathbf{X}^* = \arg\max_{\hat{\mathbf{X}}} p(\hat{\mathbf{Y}}|\hat{\mathbf{X}}) \tag{4.74}$$

$\mathbf{Z}^*$ can then be obtained by taking the expectation of the distribution $p(\hat{\mathbf{Z}}|\hat{\mathbf{X}})$, where $\hat{\mathbf{X}} = \mathbf{X}^*$:

$$\mathbf{Z}^* = \mathbb{E}[p(\hat{\mathbf{Z}}|\hat{\mathbf{X}} = \mathbf{X}^*)] \tag{4.75}$$

This is the approach we adopt in this work. More details on inference for the SGPDM are given in Section 4.4.

Experiments on Audio and Visual Data

Figures 4.10a and 4.10b show 2D SGPLVM spaces, trained on AAM and MFCC data, for a sequence of the LIPS corpus. The left plot has colours for phoneme labels and the right plot has colours for viseme labels. Figures 4.10c and 4.10d show the latent space when a KBR back-constraint is placed with respect to audio data. Figures 4.10e and 4.10f show the SGPDM latent space when an autoregressive dynamics is used. Figures 4.10g and 4.10h show the SGPDM latent space when a KBR back-constraint and an autoregressive dynamics are both used.

Compared to GPDM latent spaces, SGPDM latent spaces obtained from audio and visual data reveal a more definite path unfolding through time, because both audio and visual information of a talking face are consolidated with each other, to give a latent representation of the dynamics of a talking face. Thus when audio and visual data are consolidated with each other to produce a shared latent space, there are less bifurcations in the latent space as compared to when having only a latent space of audio, due to the constraints from the visual space.

4.4 Synthesis using the SGPDM

For the SGPDM, we have a single dynamical model as well as two GP mappings from the latent space to each observation space ($\mathbf{Y}$ and $\mathbf{Z}$). If we want to infer $\hat{\mathbf{Z}}$ from $\hat{\mathbf{Y}}$, first an estimate $\mathbf{X}^*$ of the latent points $\hat{\mathbf{X}}$ needs to be performed with respect to both the observation GP for $\mathbf{Y}$ and the dynamical GP. The second observation space $\hat{\mathbf{Z}}$ can then be obtained using the mean prediction of the corresponding GP:

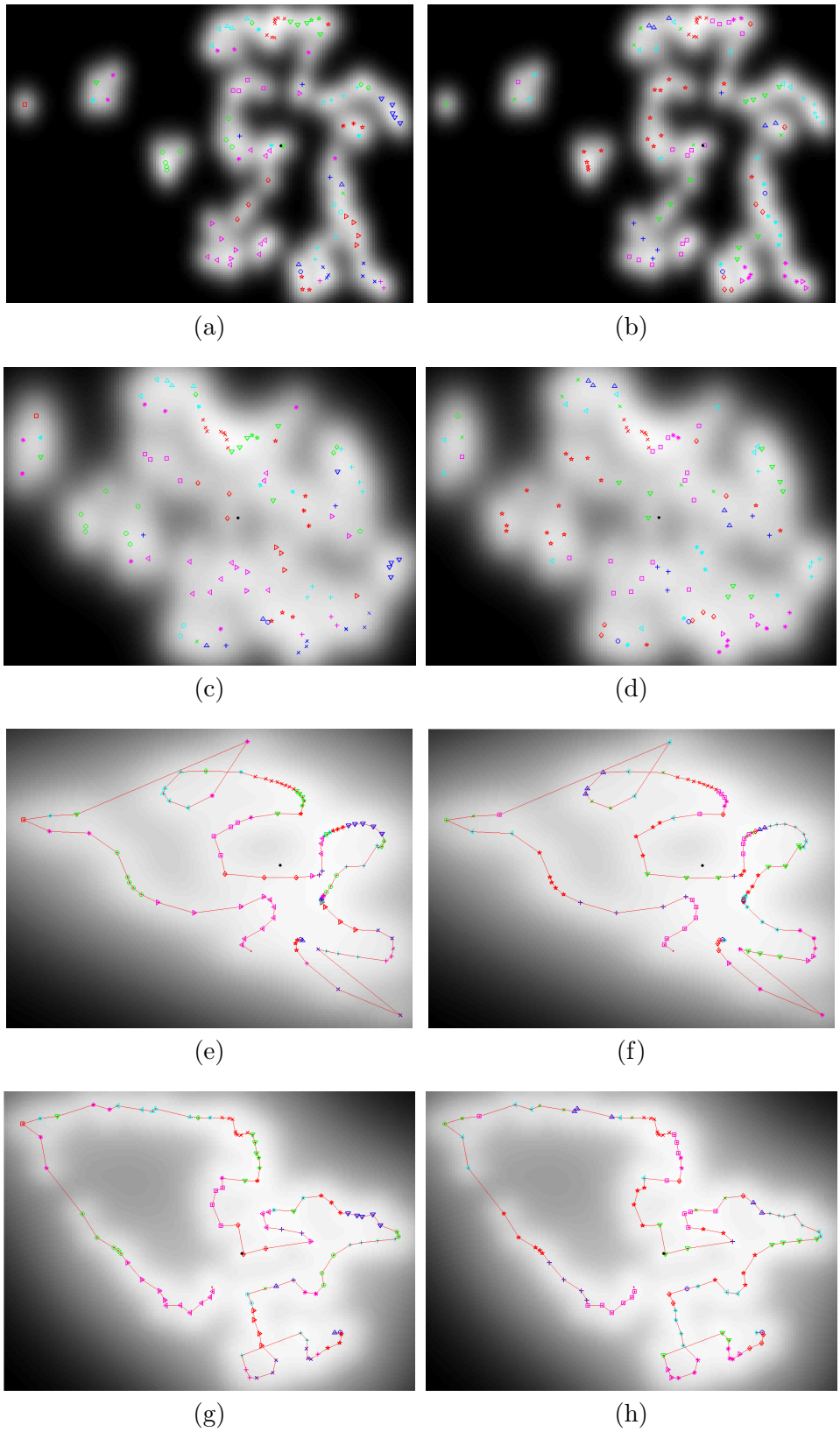


Figure 4.10: Shared latent spaces on AAM and MFCC data: Left - Phoneme Labels, Right - Viseme Labels: (a) and (b) SGPLVM, (c) and (d) SGPLVM with KBR back-constraints with respect to MFCC, (e) and (f) SGPDM, (g) and (h) SGPDM with KBR back-constraints with respect to MFCC.

$$\mathbf{Z}^* = k_Z(\mathbf{X}^*, \mathbf{X})^T \mathbf{K}_Z^{-1} \mathbf{Z} \tag{4.76}$$

where $\mathbf{Z}^*$ is an estimate of $\hat{\mathbf{Z}}$, k_Z is the kernel function for the visual GP, $\mathbf{K}_Z$ is the kernel matrix of the observation GP for $\mathbf{Z}$, computed using its corresponding training data, $\mathbf{X}$ are the latent points for the training data and $\mathbf{Z}$ are the training data points for the second observation space.

Next, we consider the inference of $\mathbf{X}^*$ from $\hat{\mathbf{Y}}$. The inference requires that we have an initialisation for the latent points because of the highly multimodal nature of the likelihood function with multiple local optima. Different initialisation techniques are considered in Section 4.4.3.

4.4.1 Point Optimisation

If we have a SGPLVM instead of a SGPDM, i.e. there is no dynamical model on the latent space, then each latent point is conditionally independent from each other and $\mathbf{X}^*$ can be inferred using a point optimisation, i.e. treating each point as being independent.

$$\mathbf{x}^* = \arg\max_{\hat{\mathbf{x}}} p(\hat{\mathbf{y}}|\hat{\mathbf{x}}, \mathbf{Y}, \mathbf{X}, \mathbf{\Phi}_Y) \tag{4.77}$$

where $\mathbf{\Phi}_Y$ are the hyperparameters of the first observation GP.

This optimisation can be carried out using scaled conjugate gradient (SCG) optimisation [124] similar to that used for GPLVM training.

4.4.2 Sequence Optimisation

For the SGPDM, the independence assumption on the latent points is removed and instead the inferred latent points should be constrained to respect the dynamics, as given by the dynamical GP. This can be done by formulating a joint likelihood between the sequence of observations and their latent points using the dynamical model. The optimisation then becomes:

$$\mathbf{X}^* = \arg\max_{\hat{\mathbf{X}}} p(\hat{\mathbf{Y}}, \hat{\mathbf{X}}|\mathbf{Y}, \mathbf{X}, \mathbf{\Phi}_Y, \mathbf{\Phi}_{dyn}) \tag{4.78}$$

where $\mathbf{\Phi}_{dyn}$ are the hyperparameters of the dynamical GP.

This optimisation is again carried out using scaled conjugate gradient (SCG) optimisation [124].

4.4.3 Initialisation of Latent Points

The following three methods can be used for the initialisation of latent points $\mathbf{X}_{init}$ from test data $\hat{\mathbf{Y}}$ prior to point or sequence optimisation.

Nearest-neighbour Initialisation

Because the SGPDM is a non-parametric method that makes predictions using training data, we can find the index m of the training data point that is closest to each test data point $\hat{\mathbf{y}}$, in terms of Euclidean distance.

$$m = \underset{n \in 1 \ldots N}{\arg\min} \operatorname{dist}(\hat{\mathbf{y}}, \mathbf{y}_n) \tag{4.79}$$

where $\operatorname{dist}(\hat{\mathbf{y}}, \mathbf{y}_n)$ is the Euclidean distance between $\hat{\mathbf{y}}$ and $\mathbf{y}_n$.

The corresponding training latent point $\mathbf{x}_m$ is then chosen as the initialisation for $\hat{\mathbf{y}}$.

The advantage of this method is that it performs an initialisation that minimises the distance between the test and training audio data. However, it does not take dynamics into account.

Back-constraint Initialisation

If a back-constraint b is used with respect to the test observation space, the latent space initialisation can be obtained using the back-constraint mapping:

$$\mathbf{X}_{init} = b(\hat{\mathbf{Y}}, \mathbf{W}) \tag{4.80}$$

where $\mathbf{W}$ are the parameters of the back-constraint mapping.

The back-constraint initialisation is very fast because it relies on a parametric mapping. However, its effectiveness depends on how well the parametric mapping maps audio data to latent points, which is contingent on the parametric assumptions used. Moreover, it also does not take dynamics into account.

Hidden Markov Model Initialisation

The HMM initialisation was proposed by Ek et al. [100]. The training latent points $\mathbf{X} = \{\mathbf{x}\}_{n=1}^{N}$ of the SGPDM are taken to be the states of a HMM and each state is associated with the test data $\hat{\mathbf{Y}} = \{\hat{\mathbf{y}}\}_{n=1}^{N}$. The transition log likelihood is computed as the GP point likelihood between each latent point and every other latent point:

$$\mathcal{L}_{i,j}^{\text{dyn}} = p(\mathbf{x}_i | \mathbf{x}_j) \tag{4.81}$$

The observation log likelihood is obtained by computing the GP point likelihood between each test data point and each of the training latent points, which are the states of the HMM:

$$\mathcal{L}_{i}^{\text{obs}} = p(\hat{\mathbf{y}} | \mathbf{x}_i) \tag{4.82}$$

This results in a trellis shown in Figure 4.11. The optimal sequence of latent points $\mathbf{X}_{init}$ is obtained from the Viterbi algorithm in log space. This can be thought of as choosing a set of latent points from the training set that best match both the test data

and the dynamical model. To speed computation when the number of training data points for the SGPDM is high, a subset of the points can be randomly chosen instead.

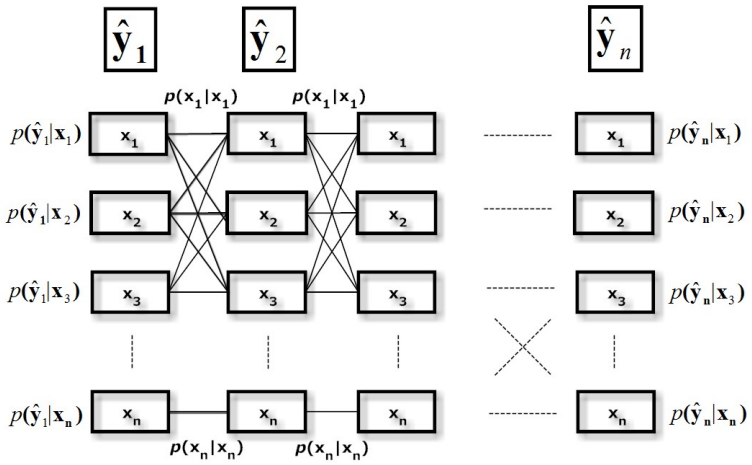


Figure 4.11: Hidden Markov model initialisation of latent points.

The HMM initialisation takes dynamics into account, thus leading to a smoother initialisation. However, the computation of the observation and dynamical likelihoods is computationally expensive. As a result, it is necessary to use a subset of training data, which can compromise the results.

Experiments that compare the different latent space initialisation methods in synthesis are presented in Section 4.5.1.

4.5 Audio-visual Mapping using SGPDM

The SGPDM can be used to couple audio and visual data through a shared latent space that represents the evolution of an underlying state-space. This approach makes sense because the audio and visual aspects of speech are highly correlated and thus can be modelled using a shared state-space. In addition, the advantage of using the SGPDM as opposed to the HMM is that the state-space is continuous and thus provides a richer representation, bypassing the need to interpolate between states to generate visual data. As compared to the shared LDS, the SGPDM offers the advantage that the observation and dynamical mappings are non-linear GPs as opposed to linear matrices. The dynamics of speech are highly non-linear [12] and thus a shared LDS only offers a linear approximation to the dynamics. Furthermore, the SGPDM can be used to cater for the many-to-one mapping between phonemes and visemes. Ek et al. [98] handled the one-to-many mapping between silhouettes and pose by placing a back-constraint with respect to the pose. This constrained the pose and latent space to have a one-to-one mapping, thus allowing a one-to-many mapping from silhouettes to pose. It also allowed the handling of ambiguity when a given silhouette corresponds to one or more

possible poses. For audio-visual mapping, the correspondence is many-to-one in the simplified case and many-to-many if noise is introduced, allowing for variations in the audio realisation of a phoneme and the visual realisation of a viseme. Thus, a back-constraint is placed with respect to audio. In Section 4.5.1, we perform experiments that compare quantitative results of visual features generated using the SGPDM with and without back-constraints. In addition, we compare two back-constraint methods, namely KBR [22] and MLP [21] and investigate the effect of varying their parameters.

The SGPDM model, however, has a large number of free parameters that can be adjusted. In the next section, we present model selection experiments to obtain the optimal parameters. In Section 4.5.2, we present experiments to determine which audio parameterisation method yields the best prediction of visual parameters using the SGPDM. In Section 4.5.3, we fix the audio parameterisation and present experiments to choose the best audio-visual synchronisation method.

A recently developed Bayesian GPLVM [286] removes the need for model selection by placing non-informative priors over the GP hyperparameters and computing posteriors over these. Moreover, it removes the need for initialisation of the latent space in training by treating the latent space as nuisance parameters that need to be marginalised. Furthermore, by using an automatic relevance determination (ARD) [22] kernel, the inherent dimensionality of the latent space can be determined in the Bayesian formulation. However, at the time that the work in this thesis was being carried out, the Bayesian GPLVM was not yet available and thus we used the maximum likelihood formulation [177, 100]. As a result, we performed model selection experiments to determine the optimal latent dimensionality, latent space initialisation methods and other free parameters in the model.

4.5.1 Model Selection

The model selection experiments described below involve a training set of 50 sequences and a validation set of 20 sequences, such that the training and validation sets are non-overlapping. The SGPDMs are first trained on the training set followed by performing inference using only audio from the validation set with the sequence optimisation method described in Section 4.4.2. The synthesised AAM features are statistically compared against the ground truth features. We use the average correlation coefficient or ACC (refer to Chapter 6 Section 6.2.1) to compare ground truth against synthesised AAM features because we found it to correspond to the visual plausibility of the facial animations. The ACC plots shown for each of the experiments include a 95% confidence interval as error bars.

The experiments proceed in a sequential manner whereby in the initial ones no dynamics and back-constraints are involved. We then introduce dynamics followed by

the introduction of back-constraints.

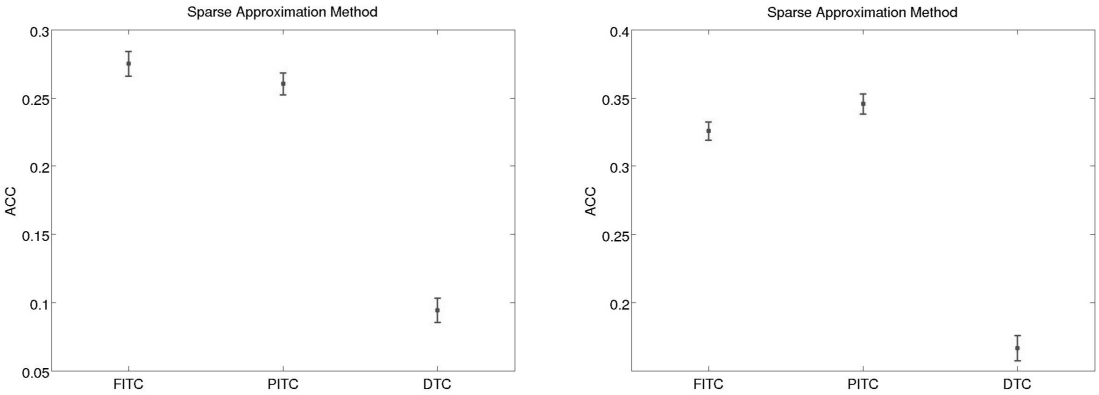
All experiments involve using a likelihood and a dynamics bias (in case a dynamical model is used), except in experiment 5 where we investigate the effect of using biases vs. not using the biases. In experiments 1 to 7, the latent space for synthesis was initialised using nearest-neighbour comparison of test audio features against training audio features (refer to Section 4.4.3). Experiment 8 compares the different initialisation methods in synthesis, in order to determine which one gives the best results. The reason why the latent space initialisation for synthesis is the last experiment is that it involves using the back-constraint initialisation (refer to Section 4.4.3), thus requiring that back-constraints are introduced.

We use normalised AAM features for visual representation and MFCC parameters downsampled using median filtering for the audio representation in the experiments. Once we obtain the optimal model parameters, we then fix these parameters and perform experiments to determine the best audio representation and audio-visual synchronisation method in Sections 4.5.2 and 4.5.3 respectively. The order of the experiments was chosen because the optimal audio representation and audio-visual mapping experiments would not be reliable if the models being used to perform these experiments do not have optimal parameters.

Experiment 1 - Sparse Approximations

In the first experiment, we train SGPDMs without back-constraints and dynamics and fix the latent space dimension to 6, which we found to give decent reconstructions of both the audio and visual spaces from the latent space (Experiment 3 will vary the latent space dimensionality). We vary the sparse approximations: FITC, PITC and DTC using k = 100 active points, as suggested by Lawrence [179].

The results are shown in Figure 4.12a for LIPS and Figure 4.12b for DEMNOW. The results show that FITC and PITC are comparable whilst DTC gives far worse results. FITC gives slightly better results than PITC for LIPS whilst for DEMNOW PITC gives slightly better results than FITC. The better performance of FITC and PITC as compared to DTC can be explained by the fact that DTC uses a variance of zero for the training conditional whereas FITC and PITC use a diagonal and block diagonal approximation to the training conditional, respectively, thus providing a richer form of approximation [179]. FITC and PITC have similar performances with slight differences between the two datasets. As a result, we fix the sparse approximations to FITC for LIPS and PITC for DEMNOW in the next experiments.



(a) (b)

Figure 4.12: Varying sparse approximations: (a) LIPS (b) DEMNOW.

Experiment 2 - Latent Space Initialisation in Training

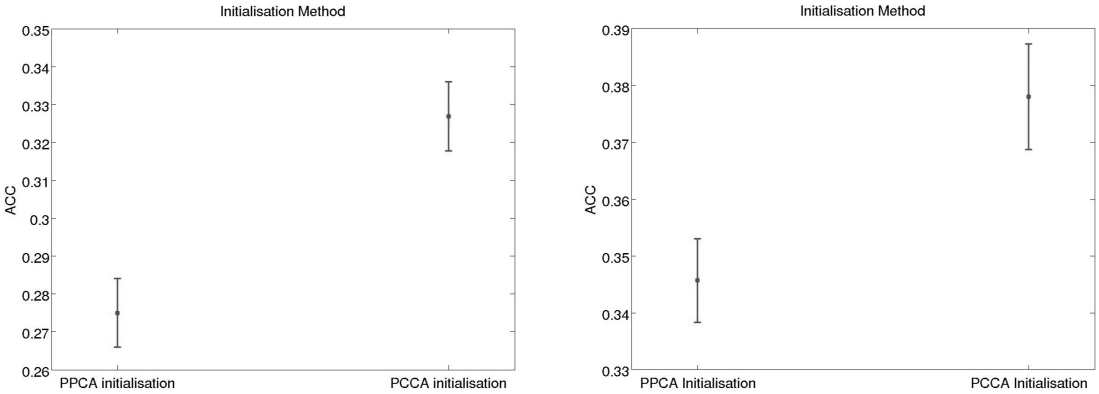
In this experiment, we again do not use back-constraints and dynamics and investigate two initialisations methods to initialise the latent points prior to training, namely: PPCA and PCCA. The latent space is again set to 6 for the same reason stated in the previous experiment.

The results are shown in Figure 4.13a for LIPS and Figure 4.13b for DEMNOW. For both LIPS and DEMNOW, PCCA initialisation gives better results than PPCA. This can be explained by the fact that PCCA finds a latent space that maximises the correlation between the two datasets and thus better mappings are learnt from the latent space to each data space using GPs in the SGPDM, as compared to using PPCA that finds separate latent spaces for each data space, which need to be averaged. In the next experiments, we fix the initialisation method to PCCA.

Experiment 3 - Latent Space Dimensionality

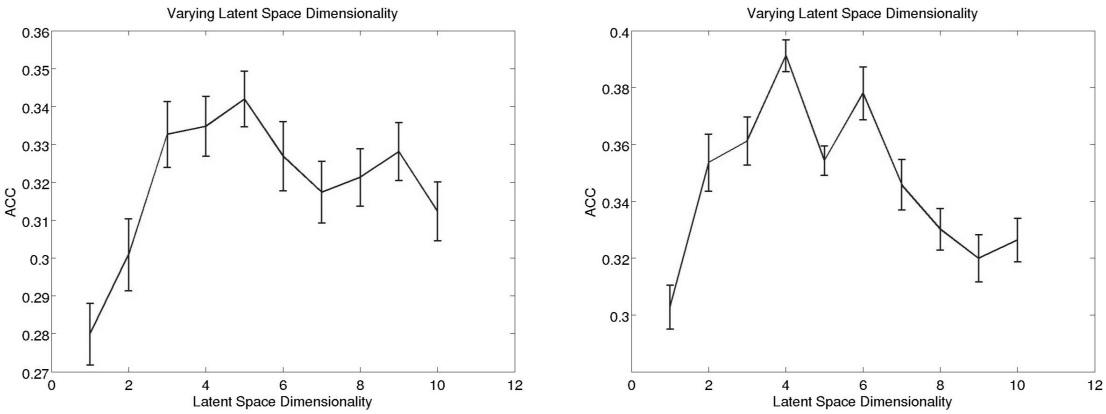
This experiment involves varying the latent space dimensionality. The latent space is varied from 1 to 10 because higher latent spaces would affect the training time adversely when introducing dynamics and back-constraints in future experiments.

Figures 4.14a and 4.14b show the results for LIPS and DEMNOW, respectively. The optimal latent space is found to be 5 for LIPS and 4 for DEMNOW. The experiment reveals the intrinsic dimensionality of data to be 5 and 4 respectively for LIPS and DEMNOW and when higher dimensions are used worse results are obtained due to overfitting. In subsequent experiments, we fix the latent spaces accordingly.



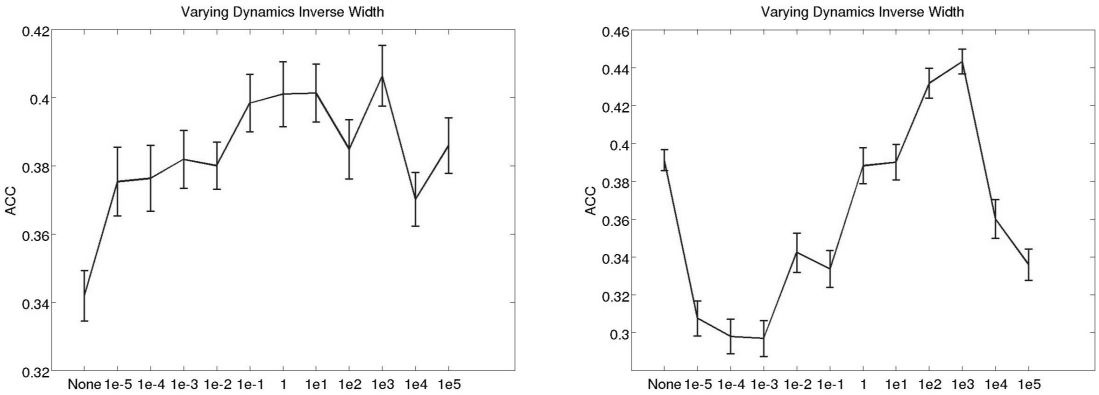
(a) (b)

Figure 4.13: Varying latent space initialisation methods: (a) LIPS (b) DEMNOW.



(a) (b)

Figure 4.14: Varying latent space dimensionality: (a) LIPS (b) DEMNOW.



(a) (b)

Figure 4.15: Varying dynamical GP RBF kernel inverse width: (a) LIPS (b) DEMNOW.

Experiment 4 - Dynamical GP Hyperparameters

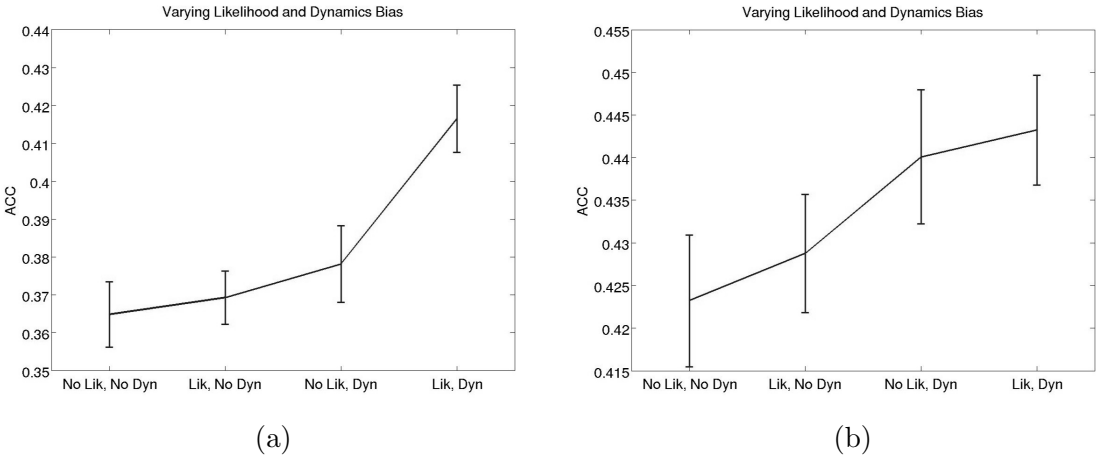
In this experiment, we introduce dynamics and vary the inverse width parameter γ of the dynamics kernel in Eqn.4.44. The first model in the experiment does not include dynamics and is included for comparison.

Figure 4.15a shows the results for LIPS, with an optimal $\gamma = 1000$. Figure 4.15b shows the results for DEMNOW, with an optimal $\gamma = 1000$. For both LIPS and DEMNOW, using dynamics with the optimal inverse width parameter gives better results than when not using dynamics.

Experiment 5 - Likelihood and Dynamics Bias

This experiment involves investigating the effects of using likelihood and dynamics biases. We use four categories of models: No likelihood and dynamics bias, likelihood bias and no dynamics bias, no likelihood bias and dynamics bias, likelihood bias and dynamics bias.

Figures 4.16a and 4.16b show the results for LIPS and DEMNOW respectively. Using a likelihood bias yields better results than when not using it. The use of dynamics bias also results in better results than both not using any bias and using likelihood bias alone. Using both likelihood and dynamics biases yields the best results, which is consistent with our hypothesis that balancing the likelihoods of each data space and dynamics in the training gives rise to a better model. We thus use both a likelihood and dynamics bias for future experiments.



(a) (b)

Figure 4.16: Varying likelihood and dynamics bias: (a) LIPS (b) DEMNOW.

Experiment 6 - KBR Back-constraint Hyperparameters

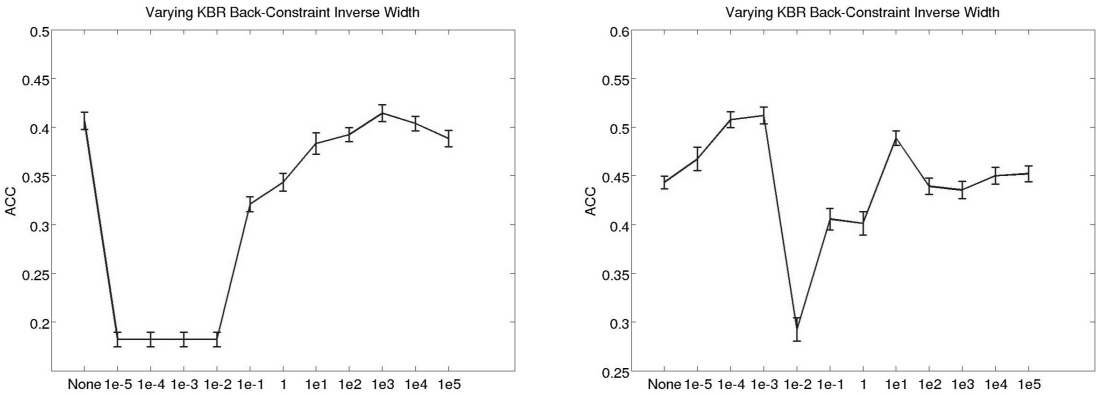
In this experiment, we introduce KBR back-constraints with respect to audio, using an RBF kernel and we vary the inverse width of the kernel. The first model in the experiment does not involve back-constraints and is included in order to investigate whether using KBR back-constraints brings any benefit to the results.

The results show that for both LIPS and DEMNOW, using a KBR back-constraint yields better results than not using it, which supports the claim that having a proper back-constraint allows us to model the many-to-one from phonemes to visemes. The optimal inverse width for LIPS is found to be 1000 whilst for DEMNOW, it is 0.001. The difference for the two datasets can be explained by different kernel responses of the KBR back-constraint to each dataset, which can arise due to different dynamic ranges of the data.

Experiment 7 - MLP Back-constraint Parameters

This experiment involves the introduction of an MLP back-constraint with respect to audio data and varying the number of hidden layers of the MLP [22]. The first model in the experiment does not use any back-constraints and is included to determine whether the MLP back-constraint brings any benefit to audio-visual mapping using the SGPDM.

The results in Figure 4.18a show that for LIPS, the MLP back-constraints lead to worse results than not using back-constraints at all, with the results getting better as the number of hidden layers is increased. The reasons for this are not very clear but could be due to competition between the back-constraints and the dynamics. The dynamics try to constrain temporally close points to be close on the latent space and the MLP



(a) (b)

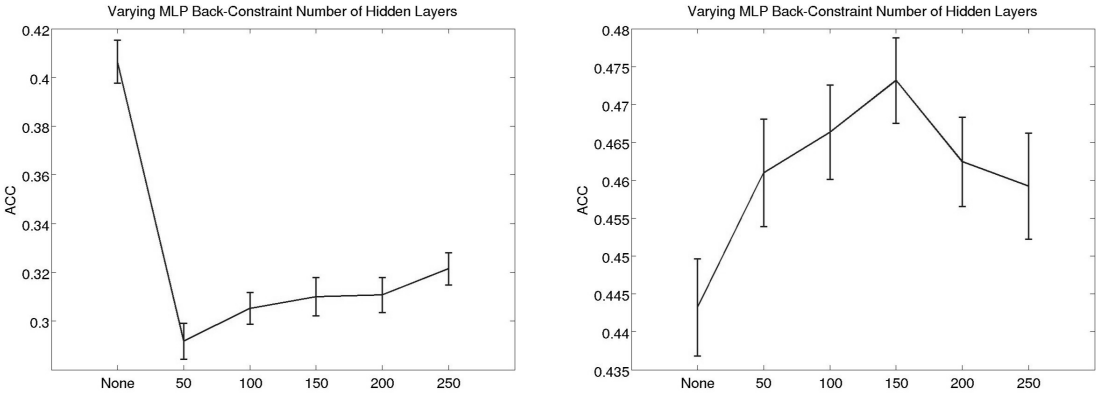
Figure 4.17: Varying KBR back-constraints RBF kernel inverse width: (a) LIPS (b) DEMNOW.

back-constraint trying to move spatially distant points far apart on the latent space, even though they are temporally close. A balance between the dynamics and back-constraints is necessary and this seems better enforced using the KBR back-constraints in the LIPS data. For DEMNOW, on the other hand, using MLP back-constraints yields better results than not using it with the optimal number of hidden layers being 150, but the optimal results obtained using KBR back-constraints are still better. We thus use KBR back-constraints in future experiments with the SGPDM.

Experiment 8 - Latent Space Initialisation in Synthesis

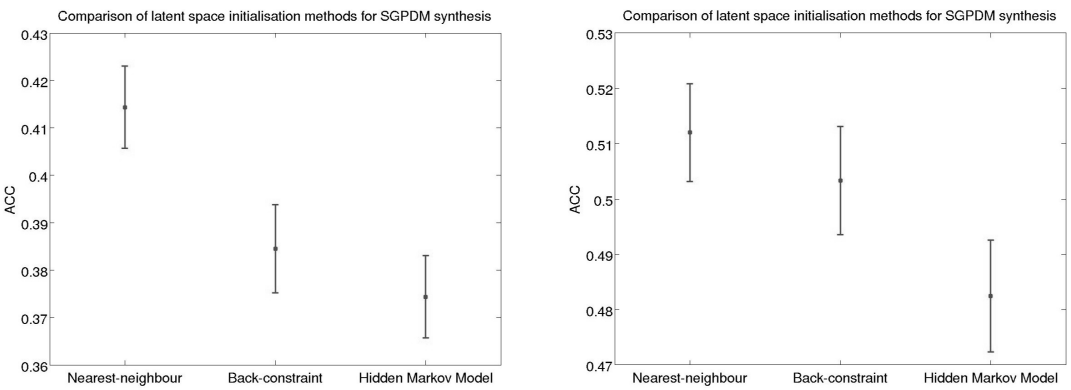
In this experiment, a KBR back-constraint and an autoregressive dynamics are both used with parameters set according to the optimal parameters found in the previous experiments. We vary the latent space initialisation methods used in synthesis, as described in Section 4.4.3.

Figure 4.19a shows the results for LIPS and Figure 4.19b show the corresponding results for DEMNOW. The nearest-neighbour initialisation method is found to give the best results, followed by the back-constraint initialisation. The HMM initialisation gives the worst results which can be explained by the fact that only a subset of the training points are used to build the HMM, thus leading to a sub-optimal initialisation on test data. The initialisation using the back-constraint mapping is limited by how well the back-constraint is able to map audio parameters to the latent space using its optimised parameters. The nearest-neighbour initialisation chooses the training latent points that correspond to the training audio parameters best matching the test audio parameters in terms of Euclidean distance, and thus gives the best performance. We thus use the nearest-neighbour initialisation method for synthesis in the SGPDM.



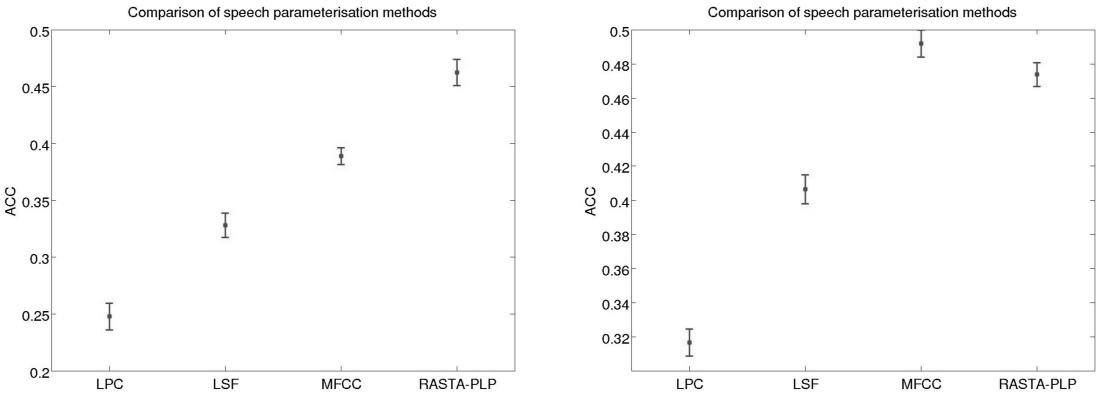
(a) (b)

Figure 4.18: Varying MLP back-constraints number of hidden layers: (a) LIPS (b) DEMNOW.



(a) (b)

Figure 4.19: Comparing different latent space initialisation methods during SGPDM synthesis: (a) LIPS (b) DEMNOW.



(a) (b)

Figure 4.20: Comparing SGPDM synthesis results for different speech parameterisations: (a) LIPS (b) DEMNOW.

4.5.2 Choice of Speech Parameterisation

This experiment is aimed at finding which speech parameterisation technique is best for visual speech synthesis using the SGPDM. The model selection experiments made use of MFCC downsampled using median filtering. In this experiment, we use LPC, LSF, MFCC and RASTA-PLP parameters processed at 25Hz for LIPS and 29.97Hz for DEMNOW (refer to Chapter 3 Section 3.5). We use the optimal model parameters found in the previous section and a KBR back-constraint with respect to audio as well as an autoregressive dynamical model on the latent space. We use the same 50 sequences for training and 20 sequences for validation, as in the model selection experiments, and compute the ACC of the synthesised AAM features with respect to ground truth.

Figures 4.20a and 4.20b show the results for LIPS and DEMNOW respectively. From the plots, it can be seen that RASTA-PLP gives the best results for LIPS whilst MFCC gives the best results for DEMNOW. The reasons for this do not seem obvious, but upon inspection of the plots of RASTA-PLP parameters in Chapter 3 Section 3.5.2, it can be observed that they produce a smoother trajectory than MFCC parameters. The LIPS corpus involves a British speaker reading sentences from the Messiah corpus [278] under controlled conditions, and tends to be hypo-articulated. On the other hand, the DEMNOW corpus involves an American speaker giving news presentations with a faster speaking rate and is thus hyper-articulated. As a result, the slower-varying RASTA-PLP parameters are more highly correlated with the smoother visual parameters of the LIPS corpus and thus are better able to predict the visual parameters. The opposite is true for DEMNOW with the MFCC parameters being more highly correlated with the less smooth AAM parameters.

It is clear from the above experiments that perceptually-motivated methods like

MFCC and RASTA-PLP outperform source-filter methods like LPC and LSF in speech animation. The same phenomenon is observed in speech recognition with perceptually-motivated features capturing more information that helps to discriminate between phonemes [69]. Researchers have also compared the performance of source-filter and perceptually-motivated methods in speech animation [162, 279] and showed the superiority in performance of the latter category of methods. In our experiments, the speech was not subject to much additive and convolutional noise (refer to Chapter 3 Section 3.2.5). RASTA-PLP is the speech feature of choice in noisy environments [134] and outperforms MFCC features in noisy speech recognition due to the latter's lack of robustness to noise [9, 217]. RASTA-PLP is also useful in applications that favour speaker-independence [217].

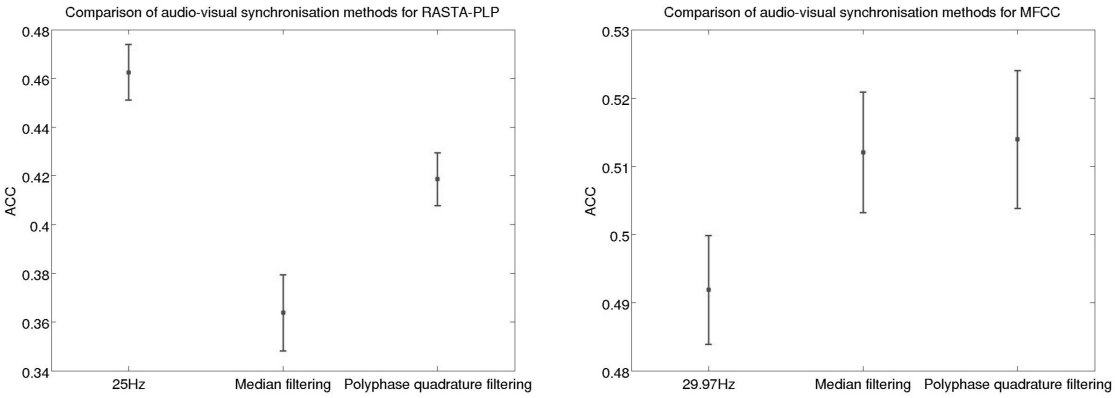
4.5.3 Choice of Audio-visual Synchronisation Method

This experiment is similar to the previous one but instead of varying the speech parameterisation method, we vary the audio-visual synchronisation method (refer to Chapter 3 Section 3.5.1). We use RASTA-PLP for LIPS and MFCC for DEMNOW and experiment with the three synchronisation methods namely: audio parameters processed at the same rate as visual parameters, audio parameters downsampled from 100Hz using median filtering [6] and polyphase quadrature filtering [256].

Figures 4.21a and 4.21b show the results for LIPS and DEMNOW respectively. It can be deduced from the plots that RASTA-PLP processed 25Hz gives the best results for LIPS, whilst for DEMNOW, MFCC parameters downsampled from 100Hz using polyphase quadrature filtering gives the best results. The reason again seems to have to do with the smoothness of the audio features. Theobald and Wilkinson [279] showed that using a larger auditory window tends to smooth out the audio parameters and thus downsampling from 100Hz retains more of the coarseness of the original speech parameters. For LIPS, the smoother RASTA-PLP at 25Hz correlates better with the slower-varying AAM parameters whilst for DEMNOW, the downsampled MFCC parameters retain more information on the temporal scale than MFCC parameters processed at 29.97Hz, and thus correlate better with the faster-varying AAM parameters, hence explaining their better predictive abilities using the SGPDM.

4.5.4 Limitations of SGPDM

The SGPDM model is effective at modelling the generative model of two representations of the same process, which in our case are the audio and visual components of speech. However, since the SGPDM is a non-parametric model, the size of the model grows with the data. Although sparse approximation techniques reduce the



(a) (b)

Figure 4.21: Comparing SGPDM synthesis results for different audio-visual synchronisation methods: (a) LIPS (b) DEMNOW.

time and space complexity of training and inference, training the model becomes intractable on modern computers if the amount of training data exceeds a few thousand frames. Moreover, the phenomenon of coarticulation results in a wide range of highly non-linear dynamics, which when modelled using a single dynamical model yields an over-generalised predictive model.

In the work of Lehn-Schiøler [185], a shared linear dynamical system (shared LDS) was used as a generative model of speech. The shared LDS has the same graphical representation as the SGPDM except for being a parametric model with linear observation and dynamical mappings. Englebienne [101] proposed a more powerful model by augmenting the linear dynamical system (LDS) with switching states, where each state corresponds to the visual speech dynamics of individual phonemes. The resulting model is the switching linear dynamical system (SLDS). However, Englebienne [101] showed that the parametric assumptions in the SLDS are not suitable for visual speech synthesis and simplified the model to obtain a model called the deterministic process dynamical system (DPDS). In the next chapter, we present a non-parametric switching state-space model, which is obtained by augmenting the SGPDM with switching states, thus accounting for the multiple dynamics in speech. As compared to the SLDS and DPDS which are models of only visual speech, the proposed switching SGPDM (SSGPDM) is a model of both audio and visual speech. Moreover, the SSGPDM is a non-parametric model that does not suffer from parametric assumptions in models like the SLDS, which has been shown to perform poorly on visual speech synthesis [101].

4.5.5 Discussion

The SGPDM is a powerful non-parametric method that can be used to couple audio and video streams of a talking face using non-linear dynamical and observation functions modelled using GPs. Its advantages over HMMs are that: 1) the state-space is continuous and therefore bypasses the need to interpolate from a discrete state sequence to synthesise visual data and 2) a shared latent space is used to represent the generative process of speech and maps to both the audio and visual modalities, which is more principled than training a HMM on visual data and remapping it to audio data as done by Brand [32]. Its advantage over the shared LDS of Lehn-Schiøler et al. [186] is that non-linear GPs are used to model the observation and dynamical functions, which better models the non-linear relationship between audio and visual speech.

However, these advantages come at the cost of increased space and time complexity. For a training set of N frames, the SGPDM has $O(N^3)$ space complexity. The time complexity for each iteration of the training and inference algorithm is $O(N^3)$ without sparse approximations and $O(k^2N)$ when using sparse approximations with k active points. On the other hand, for a HMM with K states and dimensionality of the visual data being D_Z, the space complexity for training is $O(K^2 + D_Z^2)$, whilst the time complexity for each iteration is $O(N^2K)$. For inference using the HMM, the time complexity of the Viterbi algorithm is $O(\hat{N}^2K)$, where $\hat{N}$ is the number of frames in the test sequence whilst the time complexity of the trajectory synthesis algorithm of Brand [32] is $O(\hat{N}D_Z^3)$. For training a shared linear dynamical system on audio data of dimensionality D_Y and visual data of dimensionality D_Z on N frames of data, the space complexity is $O(D^2)$, where $D = max(D_Y, D_Z)$. The time complexity is $O(dN^2)$ for each iteration, where d is the dimensionality of the latent space. The inference time complexity of the LDS is $O(d\hat{N}^2)$ where $\hat{N}$ is the number of frames in the test sequence. Thus, the SGPDM comes with a much higher computational complexity than parametric state-space models. In addition, in the absence of a Bayesian formulation, model selection has to be carried out to optimise the free parameters in the model.

4.6 Chapter Summary

This chapter introduced various probabilistic models used in visual speech synthesis, using the framework of graphical models. Gaussian processes which are a powerful non-parametric way of representing distributions over functions, were introduced. We then described the Gaussian process dynamical model (GPDM), which is a state-space model using Gaussian process dynamical and observation models. We also presented a review of shared latent variable models in order to motivate the shared Gaussian process dynamical model (SGPDM), which allows the coupling of two observation spaces using

non-linear mappings. We then presented synthesis techniques for the SGPDM. Model selection experiments as well as experiments to determine the optimal speech processing techniques were finally dealt with.

Chapter 5

Switching State-Space Model for Audio-visual Mapping

> No army can stop an idea whose time has come.
>
> Victor Hugo

This chapter presents a more powerful generative model for audio-visual mapping by addressing the limitations of the SGPDM. The resulting model is a switching state-space model that is obtained by augmenting the SGPDM with switching states, resulting in the switching SGPDM (SSGPDM). We first present other switching state-space models that have been previously applied to visual speech synthesis, followed by a description of the SSGPDM. We then deal with a technique for inferring switching states from streams of phonemes by using a variable length Markov model (VLMM) to find phonetic contexts. Using phonetic contexts as switching states allow the modelling of coarticulation. Experiments are presented for determining the maximum memory length to use with the VLMM as well as whether to train the VLMM on phonemes with or without repetitions. We also compare states of SSGPDMs trained with phonetic contexts and phonemes as switching states using volume rendering visualisation. Finally, we present synthesis techniques that we have developed to predict visual from audio data with the SSGPDM.

A preliminary version of this chapter appeared in [71].

5.1 Switching State-space Models

Switching state-space models allow switches or jumps in the state-space evolution, which may be due to transitions to a different regime with different dynamics. They are thus appropriate to model multiple dynamics in speech. The first switching state-space

model used for modelling speech for the purpose of speech recognition was the stochastic segment model (SSM) [84], where a separate LDS was trained for each phoneme. The SSM however assumes that each switching state is conditionally independent from each other. A generalisation of this model with conditional dependencies between switching states is the SLDS. Englebienne et al. [102] and Englebienne [101] applied the SLDS and a modified version called the deterministic process dynamical system (DPDS) to visual speech synthesis. We first describe the SLDS and the DPDS before presenting our proposed switching state-space model, obtained by augmenting the SGPDM with switching states. The resulting model is called the switching SGPDM (SSGPDM) and is described in Section 5.1.3.

5.1.1 Switching Linear Dynamical System

The SLDS was first introduced in the Machine Learning community by Ghahramani and Hinton [122]. Taking $\mathbf{Y} = \{\mathbf{y}_t\}_{t=1}^T$ to be observed data, $\mathbf{X} = \{\mathbf{x}_t\}_{t=1}^T$ to be hidden continuous states and $\boldsymbol{\pi} = \{\pi_t\}_{t=1}^T$ to be discrete hidden states, the state-space equations of the SLDS are:

$$\mathbf{x}_1 = \boldsymbol{\mu}_{\pi_1} + \mathbf{u}_{\pi_1} \tag{5.1}$$

$$\mathbf{x}_t = \mathbf{A}_{\pi_t}\mathbf{x}_{t-1} + \boldsymbol{\nu}_{\pi_t} + \mathbf{v}_{\pi_t} \tag{5.2}$$

$$\mathbf{y}_t = \mathbf{B}_{\pi_t}\mathbf{x}_t + \mathbf{w}_{\pi_t} \tag{5.3}$$

where the switching state at time t takes one of K distinct values, $\pi_t \in \{1 \dots K\}$ and each switching state consists of a different initial offset $\boldsymbol{\mu}_{\pi_t}$, a process offset $\boldsymbol{\nu}_{\pi_t}$, transition matrix $\mathbf{A}_{\pi_t}$, observation matrix $\mathbf{B}_{\pi_t}$ and noise terms:

$$\mathbf{u}_{\pi_t} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Lambda}_{\pi_t}) \tag{5.4}$$

$$\mathbf{v}_{\pi_t} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}_{\pi_t}) \tag{5.5}$$

$$\mathbf{w}_{\pi_t} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Gamma}_{\pi_t}) \tag{5.6}$$

The graphical model of the SLDS is shown in Figure 5.1. The SLDS is suitable for modelling data that have different switching regimes, each having linear dynamics. For example, Ghahramani and Hinton [122] applied the SLDS to model sleep apnea data, which is characterised by at least two regimes: no breathing and gasping breathing. The SLDS has also been applied to: speech recognition [255, 254, 211], the modelling the dancing behaviour of honey bees [222, 115] as well as human body tracking [232].

Training

The parameters of the model, $\boldsymbol{\theta} = \{\mathbf{A}_{\boldsymbol{\pi}}, \mathbf{B}_{\boldsymbol{\pi}}, \boldsymbol{\Sigma}_{\boldsymbol{\pi}}, \boldsymbol{\Gamma}_{\boldsymbol{\pi}}, \boldsymbol{\Lambda}_{\boldsymbol{\pi}}\, \boldsymbol{\mu}_{\boldsymbol{\pi}}, \boldsymbol{\nu}_{\boldsymbol{\pi}}\}$, can be learnt using maximum likelihood through the EM algorithm [75]. However, Ghahramani and Hinton [122] showed that the E-step, which involves computing posterior distributions

over the hidden states, is intractable because the posterior distribution over $\mathbf{X}$ is a Gaussian mixture with S^T terms. Thus, a variational inference [22] approximation to the E-step was proposed by Ghahramani and Hinton [122], where the posterior distribution over $\mathbf{X}$ is approximated using potential functions, thus making inference tractable. Other researchers have proposed different deterministic approximations such as: expectation propagation (EP) [328] and expectation correction (EC) [13], which are both variants of assumed density filtering (ADF) [206] and involve collapsing the intractable posterior to either a Gaussian or a mixture of Gaussians with a smaller number of components. Both EP and EC involve a forward pass (filtering) and a backward pass (smoothing) through the graphical model using the sum-product algorithm [173]. The key difference between EP and EC is that in EP, the backward messages correspond to conditional likelihoods, whilst in EC, they correspond to posterior distributions that result from a conditional independence assumption in the inference equations, making EC more numerically stable [13]. Sampling-based approaches for approximate inference, such as Rao-Blackwellised Gibbs sampling [254], data-driven Markov Chain Monte Carlo (MCMC) [223], iterative MCMC [87] and sequential MCMC based on particle filtering [88], have also been explored. Earlier approaches included an approximate Viterbi algorithm [234, 233] for computing a posterior over the hidden states.

Inference

The problem of inference is closely related to the E-step of the EM training algorithm, where the goal is to infer both $\hat{\boldsymbol{\pi}} = \{\hat{\pi}_t\}_{t=1}^T$ and $\hat{\mathbf{X}} = \{\hat{\mathbf{x}}_t\}_{t=1}^T$ given a sequence of observations, $\hat{\mathbf{Y}} = \{\hat{\mathbf{y}}_t\}_{t=1}^T$. Inference involves either filtering or smoothing or both and can leverage on deterministic approximations, sampling-based methods or approximate Viterbi.

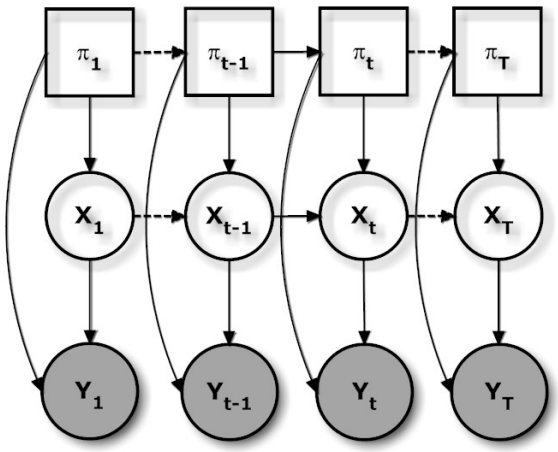


Figure 5.1: Graphical model for switching linear dynamical system and deterministic process dynamical system.

Application to Visual Speech Synthesis

The switching linear dynamical system (SLDS) was applied to visual speech synthesis by Englebienne et al. [102] and Englebienne [101], where phonemes were used as the switching states and visual data were the observations. Given a test sequence of phonetic labels $\hat{\boldsymbol{\pi}} = \{\hat{\pi}_t\}_{t=1}^T$, the most likely sequence of AAM parameters can be obtained by setting $\hat{\mathbf{x}}_1 = \boldsymbol{\mu}_{\hat{\pi}_1}$ and iterating for $t > 1$:

$$\hat{\mathbf{x}}_t = \mathbf{A}_{\hat{\pi}_t}\hat{\mathbf{x}}_{t-1} + \boldsymbol{\nu}_{\hat{\pi}_t} \tag{5.7}$$

$$\hat{\mathbf{y}}_t = \mathbf{B}_{\hat{\pi}_t}\hat{\mathbf{x}}_t \tag{5.8}$$

However, it was found that the SLDS in its original form is over-parameterised, i.e. it has too many degrees of freedom. As a result, it is prone to overfitting due to its flexibility. Englebienne et al. [102] found that the SLDS explains the data very well but does a poor job at synthesis. In particular, it was found that having a stochastic latent process with a Gaussian noise term might lead to a poor estimation of the process that generated the data, if the noise in the data is skewed. This leads to an accumulation of errors in the continuous states during synthesis, resulting in poor animation.

5.1.2 Deterministic Process Dynamical System

Englebienne et al. [102] introduced several simplifications in the SLDS in order to have a more constrained switching state-space model for visual speech synthesis. Specifically, the noise term in the latent process was eliminated, resulting in a deterministic process. The linear prediction matrices, $\mathbf{A}_{\pi_t}$, were made diagonal, due to the AAM parameters being uncorrelated as a result of PCA. Moreover, the noise term $\mathbf{w}_t$ was shared across all switching states with its covariance $\boldsymbol{\Gamma}$ constrained to be diagonal. Finally, the linear mapping $\mathbf{B}_{\pi_t}$ from the continuous states to the observations was removed. This leads to the following state-space equations:

$$\mathbf{x}_1 = \boldsymbol{\mu}_{\pi_1} \tag{5.9}$$

$$\mathbf{x}_t = \mathbf{A}_{\pi_t}\mathbf{x}_{t-1} + \boldsymbol{\nu}_{\pi_t} \tag{5.10}$$

$$\mathbf{y}_t = \mathbf{x}_t + \mathbf{w}_t \tag{5.11}$$

where $\mathbf{w}_t \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Gamma})$.

The graphical model is same as for the SLDS and is shown in Figure 5.1.

Training

Training the DPDS involves finding the parameters $\boldsymbol{\theta} = \{\mathbf{A}_{\boldsymbol{\pi}}, \boldsymbol{\mu}_{\boldsymbol{\pi}}, \boldsymbol{\nu}_{\boldsymbol{\pi}}, \boldsymbol{\Gamma}\}$. When the switching state labels are unknown, the standard EM algorithm [75] cannot be used for training the model because the E-step requires the enumeration of an exponential number of states, just like for the SLDS. However, in Englebienne [101] the discrete states

represented the phonetic labels which are known thus leading to no hidden variables in the likelihood function. As a result, the EM algorithm was not required. Instead, a combination of gradient-based optimisation and closed-form solution was proposed to infer the parameters $\boldsymbol{\theta}$ of the DPDS. For estimating $\mathbf{A}_{\pi_t}$, a gradient-based optimisation was carried out and at each iteration, the other parameters $\boldsymbol{\mu}_{\pi_t}, \boldsymbol{\nu}_{\pi_t}$ and $\boldsymbol{\Gamma}$ were estimated in closed-form by solving a system of linear equations using the current estimate of $\mathbf{A}_{\pi_t}$. The iterations were continued until convergence of the log-likelihood. This combination of gradient-based optimisation and solving of a linear system of equations has been shown to markedly improve the rate of convergence, as opposed to when all the parameters are estimated using gradient-based optimisation [102, 101].

Inference

Exact inference of the continuous latent states in the DPDS is intractable when the discrete states are unknown, just as for the SLDS, because the enumeration of S^T state assignments is required for a sequence of length T. Because the latent process is deterministic, the continuous states are exactly determined by the discrete states, which would lead to singularities in expectation propagation (EP) [328] or expectation correction (EC) [13] equations. Instead, Englebienne [101] proposed an approximate Viterbi algorithm to solve for the discrete states, $\hat{\boldsymbol{\pi}} = \{\hat{\pi}_t\}_{t=1}^T$, given a sequence of observations, $\hat{\mathbf{Y}} = \{\hat{\mathbf{y}}_t\}_{t=1}^T$. The proposed algorithm is analogous to computing the distribution of the latent variable as a Gaussian mixture with zero covariance and keeping the most likely mixture element at each time step, followed by backtracking to obtain the optimal states for the whole sequence.

Application to Visual Speech Synthesis

In Englebienne et al. [102] and Englebienne [101], the DPDS was applied to model the dynamics of facial data, represented by AAM parameters. The audio data was processed separately by using a HMM to obtain a phonetic transcription from MFCC parameters. The audio data was processed at 100Hz whilst the visual data was processed at 29.97Hz. After phonetic alignment, the phoneme labels were shrunk to match the visual parameters by choosing the most frequent phoneme in a given window, corresponding to one frame of AAM parameters. The DPDS was then trained on the AAM parameters with phonemes as the switching states. For synthesis, a phoneme stream obtained from phonetically aligning the test audio was used to obtain the most likely sequence of AAM parameters using the DPDS. Given a sequence of phonetic labels $\hat{\boldsymbol{\pi}} = \{\hat{\pi}_t\}_{t=1}^T$, the most likely sequence of AAM parameters can be obtained by setting $\hat{\mathbf{y}}_1 = \boldsymbol{\mu}_{\hat{\pi}_1}$ and iterating for $t > 1$:

$$\hat{\mathbf{y}}_t = \mathbf{A}_{\hat{\pi}_t}\hat{\mathbf{y}}_{t-1} + \boldsymbol{\nu}_{\hat{\pi}_t} \tag{5.12}$$

The DPDS has been shown to generate perceptually more realistic facial animations than the SLDS and the Voice Puppetry of Brand [32] in [102, 101]. However, it has some limitations. First, only backward or *preservatory* coarticulation is modelled using the DPDS. Indeed, synthesis does not take into account future phonemes, which can be an advantage from an application perspective, but the animations do exhibit sudden bursts of mouth movements because of the non-modelling of *anticipatory* coarticulation. This still results in acceptable animations for languages such as English that are more preservatory than anticipatory (refer to Chapter 2 Section 2.1.5). Secondly, the animation is completely driven by discrete phonemes at the expense of discarding prosodic information in the speech signal. Thus, if the speaker is trying to put some emphasis in their speech, retaining prosodic information from the speech signal would allow the facial expressions to reflect that. These limitations can be addressed by: 1) using higher-order Markov models to model phonetic context, thus accounting for forward and backward coarticulation and 2) learning a joint model of audio and video. The next switching state-space model that we propose tries to address these issues.

5.1.3 Switching Shared Gaussian Process Dynamical Model

Switching state-space models like the SLDS [122] have parametric assumptions that might lead to overfitting as discussed in the previous sections. An alternative is to use a non-parametric model, where the training data is used in inference, leading to synthesis results that more closely match ground truth. Non-parametric models also retain the full variance of the training data instead of having a compact parametric representation. Parametric models often under-estimate the predictive variance, as will be shown in Chapter 6. The underestimation of predictive variance with parametric models has also been reported in text-to-speech (TTS) synthesis [262]. The switching shared Gaussian process dynamical model (SSGPDM) is a non-parametric switching state-space model proposed by Chen et al. [48] and is obtained by augmenting the SGPDM (refer to Chapter 4 Section 4.3.4) with switching states, in order to cater for multiple dynamics in the data. In Chen et al. [48], the SSGPDM was applied to jointly modelling silhouettes and 3D pose data of complex behaviours such as salsa dancing, which involve dynamics with different switching regimes.

The SSGPDM is an extension of the SGPDM where multiple SGPDMs are indexed by switching states $\boldsymbol{\pi} = \{\pi_n\}_{n=1}^{N}$. The state-space equations are:

$$\texttt{if } t = 1 \texttt{ or } \pi_t \neq \pi_{t-1}$$

$$\mathbf{x}_t \sim \mathcal{N}(\mathbf{0}, \beta_{dyn_{\pi_t}}^{-1}\mathbf{I}) \tag{5.13}$$

$$\texttt{else if } \pi_t = \pi_{t-1}$$

$$\mathbf{x}_t = h_{\pi_t}(\mathbf{x}_{t-1}) + \boldsymbol{\epsilon}_{dyn_{\pi_t}} \quad \boldsymbol{\epsilon}_{dyn_{\pi_t}} \sim \mathcal{N}(\mathbf{0}, \beta_{dyn_{\pi_t}}^{-1}\mathbf{I}) \tag{5.14}$$

$$\texttt{end if}$$

$$\mathbf{y}_t = f_{\pi_t}(\mathbf{x}_t) + \boldsymbol{\epsilon}_{y_{\pi_t}} \quad \boldsymbol{\epsilon}_{y_{\pi_t}} \sim \mathcal{N}(\mathbf{0}, \beta_{Y_{\pi_t}}^{-1}\mathbf{I}) \tag{5.15}$$

$$\mathbf{z}_t = g_{\pi_t}(\mathbf{x}_t) + \boldsymbol{\epsilon}_{z_{\pi_t}} \quad \boldsymbol{\epsilon}_{z_{\pi_t}} \sim \mathcal{N}(\mathbf{0}, \beta_{Z_{\pi_t}}^{-1}\mathbf{I}) \tag{5.16}$$

where f_{π_t} and g_{π_t} are the observation mappings for state π_t and h_{π_t} is the corresponding dynamical mapping. The noise is heteroscedastic, i.e. each switching state π_t has a different observation and dynamical noise variance, which are given as the inverse of precision. In the SSGPDM, the dynamics are dictated by the switching states. When the next switching state is the same as the previous, the dynamical mapping h_{π_t} maps the previous latent point $\mathbf{x}_{t-1}$ to the next point $\mathbf{x}_t$. If a new state is encountered, i.e. $\pi_t \neq \pi_{t-1}$, then the new latent point is sampled from a Gaussian distribution corresponding to the dynamical noise term of switching state π_t, with mean $\mathbf{0}$ and variance $\beta_{dyn_{\pi_t}}^{-1}\mathbf{I}$.

The state-space equations of the SSGPDM are more similar to the stochastic segment model [84] than the SLDS [122] because continuous states are not propagated across the discrete switching states. Instead, whenever a new switching state is encountered in the state-space equations, the continuous states need to be sampled from the noise term indexed by the new switching state. If the switching states are known, SSGPDM training reduces to training a separate SGPDM for each switching state. The graphical model for the SSGPDM is shown in Figure 5.2.

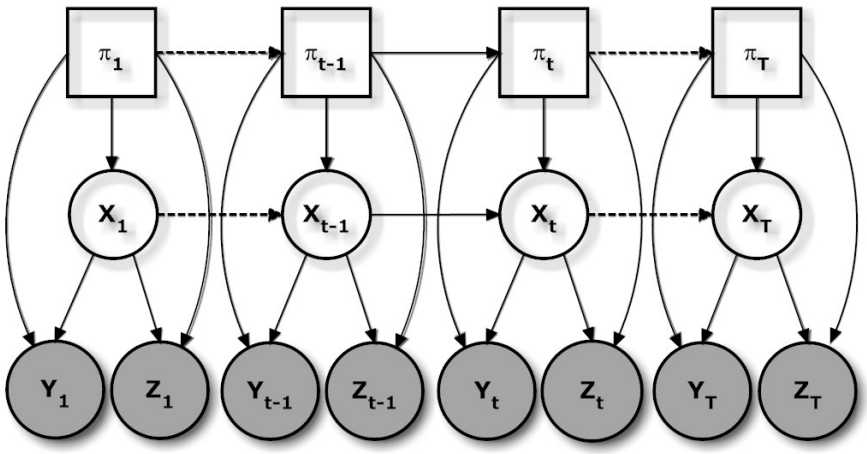


Figure 5.2: Graphical model for switching shared Gaussian process dynamical model.

Training

Training a SSGPDM is an ill-posed problem because the number of ways of segmenting the data into switching states increases with the number of states. An EM algorithm [75] can be devised, whereby in the E-step, the current parameters of the model are used to estimate an optimal segmentation, followed by the M-step where the segmentation is used to estimate the optimal parameters of each model. This approach was proposed for the stochastic segment model (SSM) [226] and by Chen et al. [48] for the SSGPDM, although no experiments were conducted to evaluate the quality of the segmentation by Chen et al. [48]. For the purpose of this thesis, we assume that the phonetic labels of speech are available and we can use either the phonemes or phonetic-contexts as switching states. An efficient algorithm for finding commonly occuring phonetic contexts is presented in Section 5.2.

Given two aligned data streams, $\mathbf{Y} = \{\mathbf{y}_t\}_{t=1}^T$ and $\mathbf{Z} = \{\mathbf{z}_t\}_{t=1}^T$, together with a corresponding aligned set of discrete labels, $\boldsymbol{\pi} = \{\pi_t\}_{t=1}^T$, the SSGPDM can be trained by grouping frames belonging to each switching state together and modelling them using a SGPDM. In section 5.3.1, we present a training algorithm for the SSGPDM on audio and visual data, where the switching states correspond to phonetic contexts. Section 5.3.2 presents a training algorithm for the SSGPDM where the switching states correspond to phonemes.

Inference

Chen et al. [48] proposed a particle filtering approach to jointly infer the switching states and the continuous latent space in view of predicting 3D pose from 2D silhouettes. Particle filtering is not ideally suited for visual speech synthesis because it would result in jitter in the results, thus affecting realism. If the switching states are known, we can locally optimise the continuous latent states for each switching state. Given a sequence of observations, $\hat{\mathbf{Y}} = \{\hat{\mathbf{y}}_t\}_{t=1}^T$, and corresponding labels, $\hat{\boldsymbol{\pi}} = \{\hat{\pi}_t\}_{t=1}^T$, the goal is to infer the continuous latent states, $\hat{\mathbf{X}} = \{\hat{\mathbf{x}}_t\}_{t=1}^T$. Inference can be done in a sequential manner, whereby the frames are traversed from $t = 1$ to $t = T - 1$. If a given state occupies only one frame, the dynamical GP is not taken into account in inference and a point optimisation is carried out (refer to Chapter 4 Section 4.4.1). Otherwise, the frames are concatenated until the state $\pi_{t+1} \neq \pi_t$. The frames in the subsequence are then used in the sequence optimisation method described in Chapter 4 Section 4.4.2, using the observation GP for f_{π_t} and the dynamical GP for h_{π_t}.

The same procedure applies when the sequence of observations given is $\hat{\mathbf{Z}} = \{\hat{\mathbf{z}}_t\}_{t=1}^T$, except that the observation GP for g_{π_t} is used in the optimisation.

From the inferred states, $\hat{\mathbf{X}} = \{\hat{\mathbf{x}}\}_{t=1}^T$, we can obtain either $\hat{\mathbf{Y}} = \{\mathbb{E}[f_{\pi_t}(\hat{\mathbf{x}}_\mathbf{t})]\}_{t=1}^T$ or $\hat{\mathbf{Z}} = \{\mathbb{E}[g_{\pi_t}(\hat{\mathbf{x}}_\mathbf{t})]\}_{t=1}^T$.

More details on our proposed inference algorithms for the SSGPDM applied to visual speech synthesis are given in Section 5.4.

Suitability of SSGPDM to Visual Speech Synthesis

The SSM [84], SLDS [122] and DPDS [102] are all parametric switching state-space models, because following training, the model is represented as a set of parameters that best describe the training data and the training data can be discarded when making predictions. On the other hand, the SSGPDM is a non-parametric switching state-space model because the training data needs to be used to make predictions. This leads to increased time and space complexity both for training and synthesis. However, as will be shown in Chapter 6, more accurate predictions are obtained using a non-parametric model. Another advantage offered by the SSGPDM is that the observation and dynamical mappings are non-linear Gaussian processes, allowing the modelling of coarticulation dynamics exhibiting non-linear structure. Rosti and Gales [255] showed that the SLDS with phonemes as switching states performs poorly at speech recognition as compared to HMMs. The reason suggested was that the linear state-space evolution is not suitable to model the non-linear dynamics of speech [12]. The SSGPDM addresses this by having multiple non-linear dynamics to represent the non-linearities in speech and the non-linear mapping from audio to visual speech.

5.2 Modelling Phonetic Context

A key question with the SSGPDM is how to segment the data into switching states. Since we know the phonetic labels for each frame of aligned audio and visual data, one possibility is to treat the phonemes as switching states. However, this approach presents the same limitation as that of Englebienne [101] in that forward coarticulation is not taken into account.

An alternative is to use phonetic contexts as switching states. However, this leads to the problem of how to partition the phonetic contexts. We could use a triphone model, as done by Bregler et al. [34], by taking the current phoneme plus a phoneme to the left and one to the right. Kshirsagar and Magnenat-Thalmann [174] adopted a segmentation of phonemes into syllables using a syllabification algorithm and then used a concatenative approach for reordering segments for visual speech synthesis. Pei and Zha [237, 238] model the dynamics of visual parameters within each syllable using a GPDM, resulting in a dynamical model for visyllables. Ostendorf et al. [227] identified phonetic contexts using k-means clustering of segments of audio data in order to group audio segments with similar context together, so as to model the clustered segments using the stochastic segment model. This approach is less accurate than

identifying phonetic contexts using phonetic labels because audio data can be noisy and the effectiveness depends on number of clusters chosen.

Yet another alternative is to use fixed-order higher order Markov models on phonetic labels, but this results in an exponential number of states, many of which are sparsely observed in the data thus requiring a large amount of data to robustly estimate their parameters. The way this is dealt with in speech recognition and text-to-speech synthesis is to cluster fixed-length contexts using decision trees [221, 323, 322] and model these clusters together by tying parameters of context-dependent HMMs. Increasing the context length would lead to a sharp increase in the number of states that have no representation from the data. The decision tree clustering would then need to group a lot of states together that might not necessarily have the same dynamics. Modelling each cluster using a SGPDM would lead to an over generalised model that fails to take into account the specificities of each context.

In this work, we use a variable length Markov model (VLMM) [130] to model phonetic context. The VLMM is an alternative higher-order Markov model where the order is variable, i.e. only states that have enough support from the data are retained. Thus, the VLMM would find commonly occuring phonetic contexts in the data and each context can then be modelled with a SGPDM, resulting in a SSGPDM. This avoids the problem of having an over-generalised model when clustering phonetic contexts of fixed-length. The next section gives a description of the VLMM and its applications to language modelling as well as to behaviour modelling in Computer Vision.

5.2.1 Variable Length Markov Model

Variable length Markov models (VLMMs) [130, 253] are a powerful extension of Nth-order Markov models which allow the memory length or order to vary locally.

A VLMM of order N generally contains fewer states than an equivalent Nth-order Markov model as higher-order states not supported by the training data are automatically pruned from the model during training, and is therefore more efficient space-wise. It presents an advantage over fixed-order Markov models in its ability to locally optimise the length of memory required for prediction. The result is a more flexible and efficient representation, having the ability to capture higher-order temporal dependencies in parts of the data and lower-order dependencies elsewhere.

A VLMM can be formulated as a probabilistic finite state automaton (PFSA), specified by $\mathcal{M} = (Q, \Sigma, \tau, \gamma, s)$, where Σ is a set of tokens representing the finite alphabet of the VLMM and Q is a finite set of model states. Each VLMM state corresponds to a string of tokens of at most length $N + 1$, representing the memory in the conditional transition distribution of the VLMM. The transition function τ, the next symbol probability function γ and the probability distribution over the initial

states s, are given as follows:

$$\tau \;:\; Q \times \Sigma \rightarrow Q \tag{5.17}$$

$$\gamma \;:\; Q \times \Sigma \rightarrow [0,1] \tag{5.18}$$

$$s \;:\; Q \rightarrow [0,1] \tag{5.19}$$

Thus, a VLMM of order N can be trained on a stream of discrete sequences of symbols from Σ, resulting in a predictive model that can predict a symbol σ using a previous string of symbols or context w of maximum length N. The VLMM can predict the transition from a state σ to $\texttt{suffix}(\sigma w)$, where $\texttt{suffix}(\sigma w)$ is a suffix of state σw, according to the transition function τ. The observation function γ specifies how likely it is for a state in Q to emit a symbol in Σ. The training procedure for the VLMM is described next.

Training

Training an Nth-order Markov model involves simply computing the frequencies of all strings of length $N+1$ from the training sequence. On the other hand, in order to train a VLMM of maximum order N, we consider w to be a prefix of length $N-1$ that can be used to predict the next character σ' according to an estimate $\hat{p}(\sigma'|w)$ of $p(\sigma'|w)$. If $\hat{p}(\sigma'|\sigma w)$ is significantly different from $\hat{p}(\sigma'|w)$, then adding the character σ to w helps better predict σ'. The decision criterion used by Guyon and Pereira [130] and Ron et al. [253] is the Kullback-Leibler divergence between the next-character distributions for the different prefixes weighted by the prior distribution of σw.

$$\Delta H(\sigma w, w) = \hat{p}(\sigma w) \sum_{\sigma'} \hat{p}(\sigma'|\sigma w) \log \frac{\hat{p}(\sigma'|\sigma w)}{\hat{p}(\sigma'|w)} \tag{5.20}$$

If $\Delta H(\sigma w, w)$ exceeds a given threshold ϵ_s, then the longer memory σw is retained, otherwise σw is pruned and the suffix w is used instead.

The VLMM training algorithm involves computing estimates of $p(\sigma_n|\sigma_1\sigma_2\ldots\sigma_{n-1})$ and $p(\sigma_1\sigma_2\ldots\sigma_n)$ for values in the range of $1 \leq n \leq N+1$. The estimates used are given by:

$$\hat{p}(\sigma_n|\sigma_1\sigma_2\ldots\sigma_{n-1}) = \frac{\nu(\sigma_1\sigma_2\ldots\sigma_n)}{\nu(\sigma_1\sigma_2\ldots\sigma_{n-1})} \tag{5.21}$$

$$\hat{p}(\sigma_1\sigma_2\ldots\sigma_n) = \frac{\nu(\sigma_1\sigma_2\ldots\sigma_n)}{\nu_0} \tag{5.22}$$

where $\nu(\sigma_1\sigma_2\ldots\sigma_n)$ is the number of occurences of string $\sigma_1\sigma_2\ldots\sigma_n$ in the training data and ν_0 is the total length of the training sequence.

In order to count strings of length $n = (1, 2, \ldots, N+1)$, a prefix tree of depth $N+1$ is grown by sliding a window of fixed length $N+1$ along training sequences and adding strings $\sigma_1\sigma_2\ldots, \sigma_{N+1}$ appearing in the window to the tree, from root to the leaves. The branches of the tree represent the characters whilst the nodes represent the

number of counts encountered for a string represented by the traversal from the root to that node. Every time a given branch is attained by entering a string $\sigma_1\sigma_2\ldots\sigma_n$, the counter $\nu(\sigma_1\sigma_2\ldots\sigma_n)$ associated with that branch is incremented. An example of a prefix tree as given by Ron et al. [253] is shown in Figure 5.3a. After the prefix tree is built, it is pruned by removing least visited nodes. A branch i is pruned if the following condition is satisfied:

$$\frac{\nu_i}{\nu_0} \leq \epsilon_p \tag{5.23}$$

where ν_i is the counter of branch i, ν_0 is the root counter and ϵ_p is a threshold value.

The next step of the VLMM training involves building a prediction suffix tree (PST). In a PST, the strings are entered in reverse order, as opposed to a prefix tree. The branches of the tree represent characters whilst the nodes have no specific representation. The node reached by accepting string $\sigma_{n-1}\sigma_{n-2}\ldots\sigma_1$ is associated with the prefix tree probability $\hat{p}(\sigma_n|\sigma_1\sigma_2\ldots\sigma_{n-1})$. Using a PST, it is possible to obtain the longest suffix of $\sigma_{-\infty}\ldots\sigma_0\sigma_1\sigma_2\ldots\sigma_{n-1}$ that provides the best probability estimate for the next character σ_n. Algorithm 3 gives the steps for building a PST, given a set of strings W, which are initialised as: $W = \{\sigma|\sigma \in \Sigma \text{ and } \hat{p}(\sigma) > \epsilon_s\}$

Algorithm 3 Training algorithm for prediction suffix tree (PST).

Input: A set of strings W, a set of tokens Σ, the maximum depth of the PST d_s, thresholds ϵ_p and ϵ_s
Output: The trained PST
Initialise the PST as a single root node.
while $W \neq \emptyset$ **do**
  Pick any $w \in W$ and remove it from W
  if $\Delta H(w, \texttt{suffix}(w)) \geq \epsilon_s$ **then**
    Add w to the PST by growing all necessary nodes
    if $|w| < d_s$ **then**
      For every $\sigma \in \Sigma$, add σw to W if $\hat{p}(\sigma w) > \epsilon_p$
    end if
  end if
end while

In order to determine $\Delta H(w, \texttt{suffix}(w)) \geq \epsilon_p$ (Eqn.5.20) according to the counts in Eqn.5.21 and Eqn.5.22, the pruned prefix tree is used by starting from the root node and entering the characters in w. The node reached gives the number of occurences of w whilst the ancestor of the node gives the counts for $\texttt{suffix}(w)$. A string w not meeting the criterion $\Delta H(w, \texttt{suffix}(w)) \geq \epsilon_s$ is not ruled out because its future descendants might meet the selection criterion. Thus, the descendants σw of w are added to W. The PST corresponding to the prefix tree in Figure 5.3a is shown in Figure 5.3b.

The final stage of training involves converting the suffix tree to a PFSA representing

the trained VLMM. Each node in the VLMM corresponds to a node in the PST. However, the label of each VLMM state is obtained by a string read backwards from the corresponding PST node to the root. A state w in the VLMM has an outgoing arc to state $\texttt{suffix}(w\sigma')$ with probability $\tau(\sigma', w) = \hat{p}(\sigma'|w)$, where $\texttt{suffix}(w\sigma')$ is the longest suffix of $w\sigma'$, obtained from the PST. The outgoing arcs are added repeatedly for each node of the PST until all nodes have been visited. In addition, the conversion from PST to PFSA involves the estimation of the initial probability function over the states, s, and the next symbol probability function, γ. More details on the conversion are given in [253]. Figure 5.3c shows a PFSA obtained from the PST in Figure **??**.

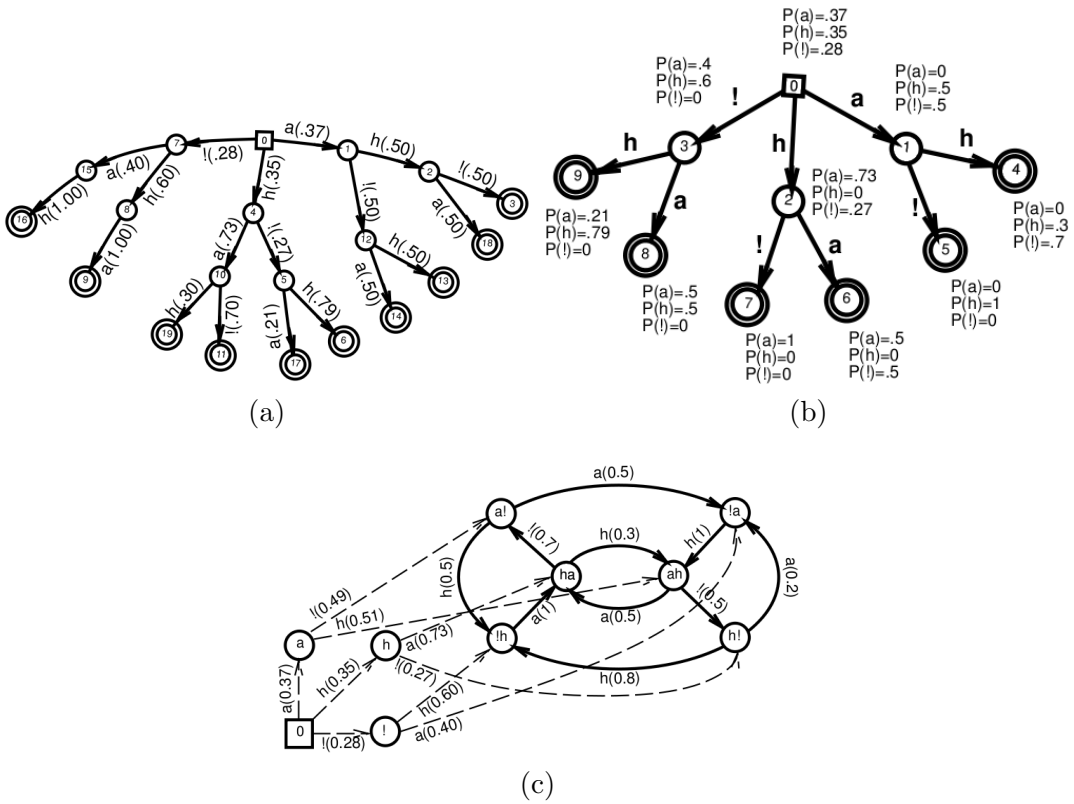


Figure 5.3: VLMM training procedure according to Guyon and Pereira [130]: (a) Build prefix tree. (b) Build PST from prefix tree. (c) Convert PST to PFSA.

Inference

Given a sequence of characters from the alphabet Σ, the aim is to infer the VLMM states for the sequence. The PFSA of the VLMM is traversed beginning with the start state and moving to the next VLMM state by applying the transition function in Eqn.5.17 to the current state and the incoming character. This is repeated until all the characters in the test sequence have been processed. Occasionally when traversing

the PFSA, a VLMM state can be reached that does not have a transition function for the incoming character, in which case, the algorithm moves back to the start state and forgets all previous memory. This assigns a VLMM state to each frame. However, because each VLMM state encodes a context of maximum length N, the assignment of VLMM states to the previous frames representing the context leads to multiple states being assigned to each frame, thus resulting in overlapping. For the purpose of synthesis with the SSGPDM, it is desirable that the assignment of VLMM states to frames is non-overlapping.

In order to assign each frame to non-overlapping VLMM states, we propose a backtracking method as given in Algorithm 4. This is particularly useful in synthesis using the SSGPDM with VLMM switching states, because of the sequential nature of the inference algorithms, requiring that each frame be assigned to one discrete state. The backtracking algorithm starts from the last VLMM state reached and marks all previous frames up to the length of that VLMM state with the index of the last state. Then, it moves to the frame before the last (in reverse order) of the newly marked frames, takes the VLMM state of that frame and marks all previous frames up to the length of that VLMM state with the index of that state. This is repeated until the first frame is reached.

Algorithm 4 Backtracking to infer non-overlapping VLMM states

Input: Overlapping VLMM states $\{\pi_t\}_{t=1}^{T}$, set of strings corresponding to VLMM states $Q = \{q_n\}_{n=1}^{K}$
Output: Non-overlapping VLMM states $\{\hat{\pi}_t\}_{t=1}^{T}$
Let π_T be the VLMM state index of the last frame (T) of the sequence, and π_t be the VLMM state index of the tth frame.
$t \leftarrow T$
while $t \geq 1$ **do**
  $M_{\pi_t} \leftarrow \texttt{Length}(q_{\pi_t})$
  $\hat{\pi}_{t-M_{\pi_t}+1:t} \leftarrow \pi_t$
  $t \leftarrow t - M_{\pi_t}$
end while

5.2.2 Language Modelling using VLMM

Language has a lot of structure that can be modelled effectively using VLMMs. Ron et al. [253] were the first to introduce the VLMM and applied it to the correction of corrupted text taken from the Bible. A dynamic programming algorithm that gives the optimal sequence of characters given a corrupted sequence was presented for that purpose. Guyon and Pereira [130] designed a linguistic postprocessor using VLMMs and introduced the prefix tree for counting the number of string occurences in the text. The model was trained on various corpora and the compression effectiveness of the

model was measured using the cross-entropy (entropy for short) as well as a lower-bound on the entropy called the intrinsic entropy. In speech recognition, a related measure called the perplexity [147] has been widely used. Given a language source $\mathcal{L}$ that produces character strings w according to a probability distribution $P_{\mathcal{L}}(w)$, and a model $\mathcal{M}$ of $\mathcal{L}$, having a probability distribution $P_{\mathcal{M}}(w)$, the cross-entropy of model $\mathcal{M}$ with respect to the actual distribution $P_{\mathcal{L}}(w)$, is a measure of the predictive ability of the model and is given by:

$$H_{\mathcal{M}}^{\mathcal{L}} = -\sum_{w} P_{\mathcal{L}}(w) \log P_{\mathcal{M}}(w) \tag{5.24}$$

The perplexity can be estimated from the cross-entropy according to:

$$B_{\mathcal{M}}^{\mathcal{L}} = 2^{H_{\mathcal{M}}^{\mathcal{L}}} \tag{5.25}$$

To measure the cross-entropy for a corpus of N characters $\{\sigma_n\}_{n=1}^{N}$ modelled with a VLMM and having states s, the following estimate is used:

$$\hat{H}_{\mathcal{M}}^{\mathcal{L}}(\text{per character}) = -\frac{1}{N}\sum_{n=1}^{N} \log P_{\mathcal{M}}(\sigma_n | s_{n-1}) \tag{5.26}$$

Hu et al. [147] compare various variable-order Markov models including the VLMM by training them on different text corpora. An additional measure called the best-path entropy was introduced, which measures the degree of non-determinism in the models. The best-path entropy corresponds to the entropy computed along the optimal state sequence and is given by:

$$\hat{H}_{\mathcal{M}}^{\mathcal{L}}(R) = -\frac{1}{N}\sum_{n=1}^{N} \log P_{\mathcal{M}}(R|O) \tag{5.27}$$

where R is a given test string and O is the optimal state sequence.

The experiments of Hu et al. [147] showed that the VLMM achieves a far superior compression compared to other higher-order Markov models, as measured by the cross-entropy. However, the VLMM was also found to exhibit higher ambiguity and inferior performance on smaller text corpora.

5.2.3 VLMMs of Behaviour

VLMMs have also been applied to learning models of human behaviour from image sequences in Computer Vision, because behaviours such as exercise, ballet dancing, hand movements and facial behaviour exhibit both short-term and long-term temporal structure. Using a VLMM provides an efficient representation of behaviour at different temporal scales as compared to a first-order Markov model which captures only short-term dependencies and higher-order Markov models which result in a lot of states that are only sparsely observed, if at all.

Galata et al. [118] introduced VLMMs to the Computer Vision community, where features were first extracted from an exercise sequence and vector-quantised, so as to represent the behaviour as a time-series sequence of discrete states. Two techniques were proposed to learn a behaviour model using VLMMs. In the first, the VLMM was trained on the discrete states and subsequently used for synthesis of new behaviour, as well as prediction of future behaviour from a history. A Hermite interpolation [37] method was proposed to interpolate between the discrete states in the feature space for synthesis. In the second method, a hierarchical memory mechanism was proposed that involved learning a VLMM at a higher level of abstraction, thus capturing higher-order dependencies. In Galata et al. [119], an interaction model between cars in a traffic sequence was learnt using VLMMs, using a representation of interaction behaviour as a discrete alphabet. New sequences of interaction behaviour could be generated by sampling from the behaviour model.

Liang et al. [189] applied VLMMs to learn a model of exercise behaviour on silhouette data with shape context [16] features extracted. The VLMM was used for recognition, where the movement primitives were recognised from a set of silhouette images by associating a discrete label to the shape context features of the test data and then using a Viterbi algorithm to estimate the optimal state sequence as predicted by the VLMM, given the observed discrete label sequence. This has similarities to the correction of corrupted text proposed by Ron et al. [253].

Caillette et al. [39, 40] applied VLMMs to tracking of ballet sequences in 3D through the use of particle filtering. Training data consisted of 3D motion capture of ballet dancing, represented as joint angles and augmented with first-order derivatives. The training data was clustered into primitive motion units and each frame was assigned to the closest cluster. A VLMM was then learnt on the state indices without removing repetitions of the same state. For tracking a test sequence, image frames were segmented and a volumetric reconstruction of the detected person was performed, followed by the fitting of Gaussian blobs to the articulated human figure. The particle filtering framework consisted of first propagating global dynamics to predict the next VLMM state and propagating local dynamics for the next joint angle configuration in case the next VLMM state is the same as the previous. The local dynamics were propagated using the first-order derivatives of the joint angles in the current cluster. Particles were weighted against the image evidence by fitting blobs to the particle configurations and using KL-divergence to measure the distance between the particle blobs and the blobs fitted to the image evidence, from which the weights of each particle were derived. Hou et al. [146] improved on that method by using a back-constrained GPLVM (refer to Chapter 4 Section 4.2.5) to perform dimensionality reduction of the feature space and

using a clustering algorithm that takes into account the uncertainty in the reduced-dimensional feature space. Stefanov et al. [267, 268] applied a similar framework to hand-tracking with the difference that there was no local dynamics model and the particle evaluation was performed using feature points detected using a Hough transform rather than blobs. To propagate the particles in case the next VLMM state is the same as the previous, Gaussian noise was added to the previous configuration and the particles were weighted in terms of the distance between the particles' feature points mapped to the image plane and the feature points from the image evidence.

Bettinger et al. [19, 20] used VLMMs represented as a PST rather than as a PFSA to learn a model of facial behaviour that could be applied to synthesise novel facial behaviour. The data consisted of video frames of a person shaking their head. The facial images were represented as AAM parameters. The feature space was then segmented into sub-trajectories with similar sub-trajectories grouped together using PCA. In order to do this, it was necessary that all sub-trajectories be encoded with the same number of points. This was done by interpolating all the sub-trajectories using cubic splines and homegeneously re-sampling them to a given number of points. A VLMM was then learnt on the sub-trajectory groupings, represented as discrete states. To generate new sequences given a history of generated sub-trajectory groups, first the longest possible memory encoded in the VLMM tree was found. The probability of generating a new sub-trajectory group could be read directly from the tree, if it was encoded in the tree. After having fetched the probabilities of generation for each sub-trajectory group, samples were drawn from this set of probabilities in order to generate the next sub-trajectory group. All the sub-trajectories generated were then concatenated, giving a sequence of AAM parameters for a synthetic facial behaviour. This method thus can generate novel facial behaviour by taking into account both short-term and long-term dependencies but is not appropriate for synthesising speech-synchronised facial animation because the audio component was not modelled.

5.2.4 Phonetic Context Modelling using VLMM

In this work, we train a VLMM on the stream of phonemes corresponding to the utterances in the training data in order to identify the switching states of our proposed switching state-space model. Figure 5.4 shows a hypothetical VLMM of maximum order two on a sequence of phonemes from the BEEP corpus.

We train a separate VLMM on the stream of phonemes from the LIPS and DEMNOW corpora respectively. For LIPS, we use 250 sequences for training and 28 sequences for testing. For DEMNOW, we use 550 sequences for training and 100 sequences for testing. The training and test sequences are non-overlapping.

VLMMs can either be trained on a stream of phonemes with repetitions or without

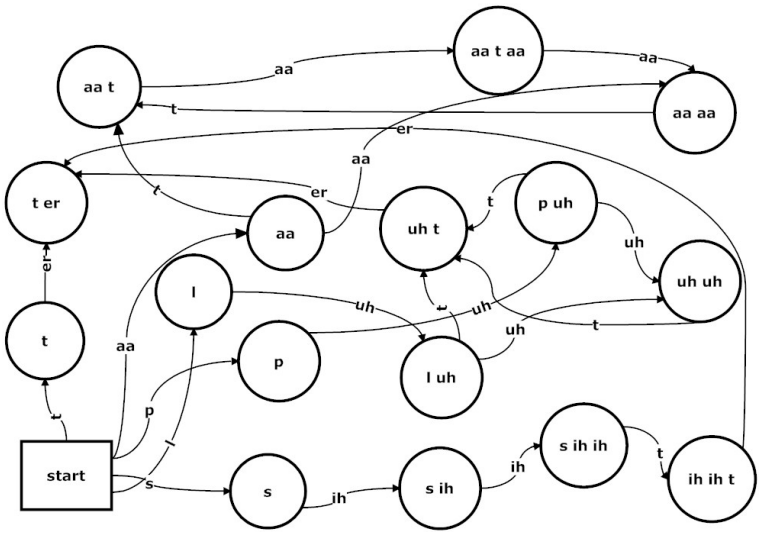


Figure 5.4: A hypothetical PFSA for a second-order VLMM on a sequence of phonemes.

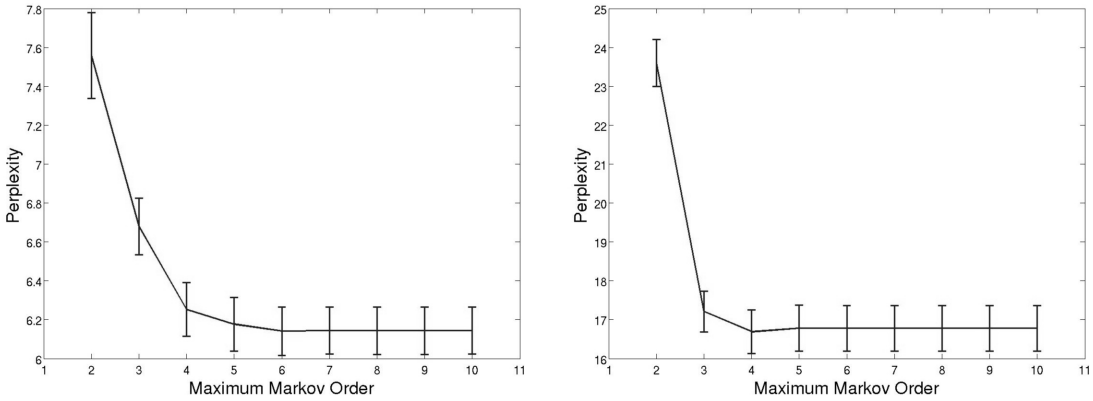
repetitions. For example, Liang et al. [189] trained a VLMM on a non-repeating sequence of behaviour prototypes whilst Caillette et al. [39], Hou et al. [146] and Stefanov et al. [268] trained VLMMs on a sequence of behaviour prototypes with repetitions retained. The advantage offered by not using repetitions is that longer-order temporal dependencies can be captured. However, it can result in a lot of states that are only sparsely observed in the data.

In our case, training a VLMM on repeating phonemes means that we can model all subsequences corresponding to a given VLMM state using an autogressive dynamical model, where the next latent state is predicted from the previous. If the VLMM is trained on non-repeating phonemes, then each subsequence will be of different length and the dynamical model to be used needs to take a time index as input to predict the latent state at that time.

In order to use the VLMM to find switching states for the SSGPDM, the segmentation has to be such that each state has enough data to train a SGPDM. We now present experiments to determine the optimal order of the VLMM . We also perform experiments to determine whether to use a VLMM trained on repeating or non-repeating phonemes. In particular, we aim to determine whether a VLMM trained on non-repeating phonemes leads to high occupancy across the states as compared to training a VLMM on repeating phonemes. The occupancy across the states would give a good indication of the predictive power of the corresponding SGPDM.

Perplexity Tests

The aim of the perplexity tests is to investigate the VLMM order to use on phonemes. This is done by observing how the perplexity (refer to Section 5.2.2) varies with the



(a) (b)

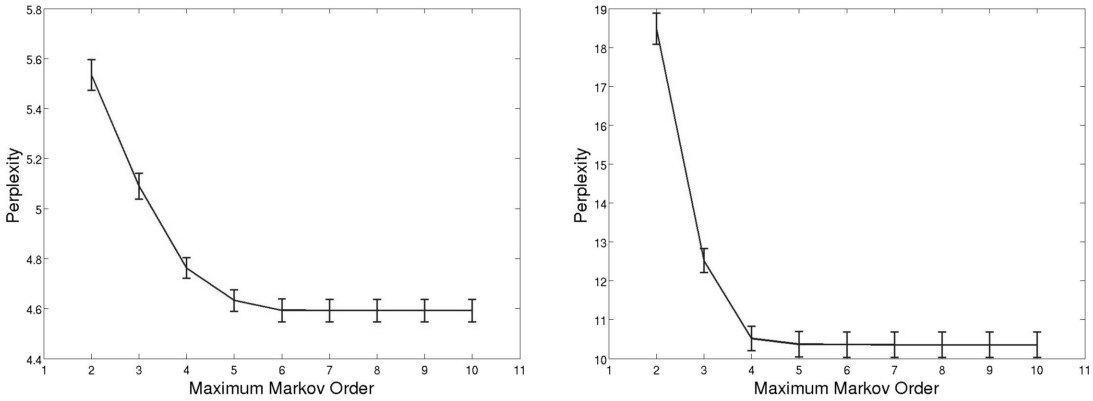
Figure 5.5: LIPS - Perplexity tests on VLMM trained on: (a) repeating phonemes (b) non-repeating phonemes.

model order. When there is no further drop in perplexity as the model order increases, this means that further increasing the order does not allow the VLMM to capture longer contexts because these do not occur frequently enough in the data. In the perplexity tests, we train VLMMs on the training set and compute the perplexity on the test set. This is repeated whilst increasing the maximum model order from 2 to an upper limit L. Perplexity plots are then generated by plotting perplexity against the maximum model order.

In our case, we vary the maximum model order from 1 to 10. The perplexity test for VLMMs trained on LIPS repeating phonemes is shown in Figure 5.5a and that for non-repeating LIPS phonemes is shown in Figure 5.5b. The corresponding plots for DEMNOW are shown in Figures 5.6a and 5.6b respectively. The plots show that for a VLMM trained on LIPS with repeating phonemes, the optimal maximum model order is 6, where the perplexity reaches the minimal value and then stays constant afterwards. This means that contexts of higher order do not occur frequently enough in the data. For a VLMM trained on LIPS with non-repeating phonemes, the optimal maximum model order is 4. For a VLMM trained on DEMNOW repeating phonemes, the optimal maximum model order is also 6 whilst on DEMNOW non-repeating phonemes, the optimal maximum model order is 5. These experiments give us the optimal memory length to use when training the phonetic VLMMs for the LIPS and DEMNOW corpora.

VLMM States Occupancy

The aim of this experiment is to find the occupancy of VLMM states for VLMMs trained on repeating and non-repeating phonemes. High occupancy per state is desirable, because it would lead to greater predictive power of our proposed switching



(a) (b)

Figure 5.6: DEMNOW - Perplexity tests on VLMM trained on: (a) repeating phonemes (b) non-repeating phonemes.

state-space model. The experiment involves training a separate VLMM for repeating and non-repeating phonetic streams for LIPS and DEMNOW and comparing the occupancy of the VLMM states. The maximum memory order of the VLMM is set to the optimal value found in the perplexity tests for each category.

A VLMM trained on repeating phonemes from the LIPS corpus yields 2136 states with 935 states having occupancy ≤ 5.

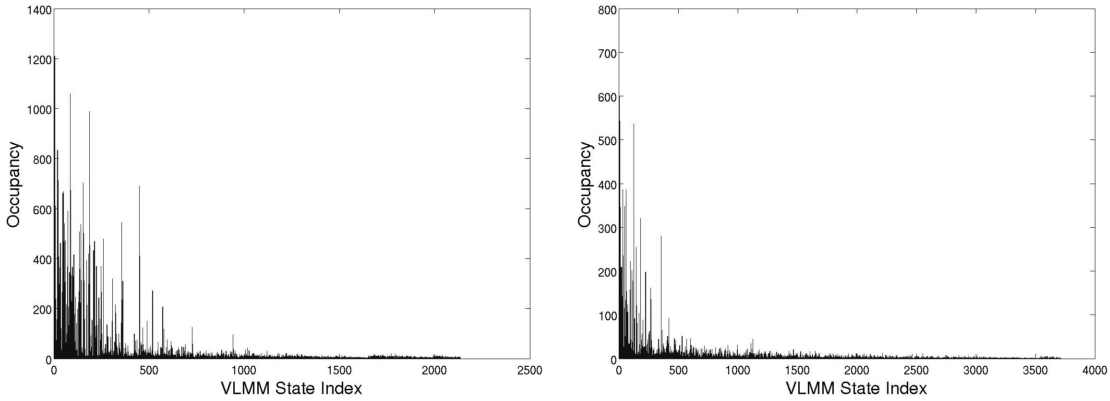
A VLMM trained on non-repeating phonemes from the LIPS corpus yields 3717 states with 3055 states having occupancy ≤ 5.

A VLMM trained on repeating phonemes from the DEMNOW corpus yields 822 states with only one state having occupancy ≤ 5.

A VLMM trained on non-repeating phonemes from the DEMNOW corpus yields 2187 states with 633 states having occupancy ≤ 5.

Figure 5.7a shows the occupancy of VLMM states for a VLMM trained on repeating LIPS phonemes, with the x-axis showing the VLMM state index and the y-axis showing the VLMM occupancy. Figure 5.7b shows the corresponding plot for a VLMM trained on non-repeating LIPS phonemes. The corresponding plots for DEMNOW are shown in Figures 5.8a and 5.8b respectively.

This experiment shows that training a VLMM on repeating phonemes results in high occupancy across a large number of VLMM states whilst a VLMM on non-repeating phonemes results in a larger number of states with the occupancy per state much lower and a larger number of states with very low occupancy. In our case, we want each state to have a high occupancy because that would lead to the SGPDM model for that state to have higher predictive power. As a result, we decide to use VLMMs trained on repeating phonemes.



(a) (b)

Figure 5.7: LIPS - Occupancy of VLMM states for VLMM trained on: (a) repeating phonemes (b) non-repeating phonemes.

The plots also reveal that the VLMM trained on DEMNOW repeating phonemes produces fewer states with higher occupancy as compared to LIPS. This is to be expected because the LIPS corpus [281] consists of utterances of sentences from a controlled list, aimed to maximise the coverage of different possible phonetic contexts whereas the DEMNOW corpus [102] consists of sentences from a newscast, which is closer to natural language. Thus, we expect to find less contexts with higher frequency of occurence per context in the DEMNOW corpus.

5.3 Learning SSGPDMs of Audio and Video

Given the switching states, the SSGPDM models can be trained on audio and visual data by training a separate SGPDM model for each switching state. The free parameters for each SGPDM are set to the optimal parameters obtained from the model selection experiments presented in Chapter 4 Section 4.5.1.

The visual data consists of normalised AAM parameters for both the LIPS and DEMNOW datasets. The audio data consists of RASTA-PLP features processed at 25Hz for LIPS data and MFCC features downsampled from 100Hz to 29.97Hz using polyphase quadrature filtering for DEMNOW data. These speech features have been found to give the best predictions of visual features for LIPS and DEMNOW, respectively, from experiments done in Sections 4.5.2 and 4.5.3 in Chapter 4.

In the SGPDM model of audio and visual data presented in Chapter 4, a back-constraint was used to model the many-to-one mapping from phonemes to visemes. By having a back-constraint, a one-to-one mapping was constrained between the audio data and the latent points, thus catering for the fact that multiple audio can map to similar

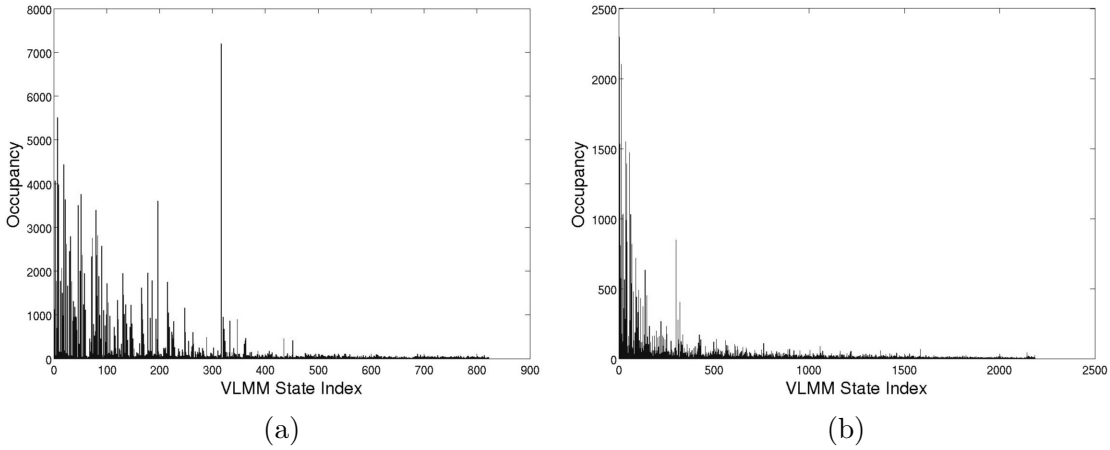


Figure 5.8: DEMNOW - Occupancy of VLMM states for VLMM trained on: (a) repeating phonemes (b) non-repeating phonemes.

visual data. In the SSGPDM, we have separate SGPDMs to model phonetic contexts. Because fewer phonemes are involved in each SGPDM comprising the SSGPDM, there are less ambiguities arising from the many-to-one relationship between phonemes and visemes. We thus do not use back-constraints in the SSGPDM. This also has the advantage in reducing the time and storage requirements of the model training.

In our experiments, we found that if the number of frames in a SGPDM exceeds a certain amount, the reconstructions of the visual and audio data from the latent space deviate substantially from ground truth. This is because the limited number of active points in the sparse approximations [179] fail to capture the whole distribution of the data. Moreover, the higher the number of frames in a given model, the more time it takes for the model to train and the larger storage required to store the model. As has been observed by some researchers [290, 300], the GPDM has a good generalisation ability with a limited amount of training data as opposed to parametric models such as the linear dynamical system (LDS). We found that a model having 1000 frames results in good reconstructions in the audio and visual spaces without compromising the predictive ability on new data. As a result, as we scan through the sequences in the training set to extract subsequences or data points pertaining to each switching state, we stop adding further data if the number of frames already added exceeds 1000.

5.3.1 SSGPDM with Phonetic Contexts as Switching States

Once the phonetic contexts have been found using the VLMM, an SSGPDM can be trained by grouping together audio and visual data corresponding to each VLMM state and modelling these together using a SGPDM. Given a set of VLMM states consisting of strings, $Q = \{q_n\}_{n=1}^{N}$ such that both the phonemes, σ_t, and the tokens,

q_n, come from a finite alphabet Σ. The state-space consists of overlapping segments, where in each segment, the string defined by the sequence of phonemes $\{\sigma_t\}$ within that segment corresponds to a string q_n. In that case, for each token q_n of length L, a window of length L is scanned across the phonemes to find string matches of that token. In case a match occurs, the corresponding frames from $\mathbf{Y}$ and $\mathbf{Z}$ are grouped together as subsequences, using a sequence delimiter vector to identify the subsequence boundaries (refer to Chapter 4 Section 4.2.5). If q_n is of length 1, a SGPLVM model without dynamics is trained on the grouped data whilst on the other hand, a SGPDM model is trained on the multiple sequences.

If instead of a single sequence, multiple data sequences $\{\mathbf{Y}_n\}_{n=1}^S$ and $\{\mathbf{Z}_n\}_{n=1}^S$ are available, the grouping of frames for each state is done over all the sequences available.

The pseudocode for the SSGPDM training algorithm for overlapping switching states is given in Algorithm 5.

Generative Coarticulation Models

The SSGPDM with phoneme VLMM switching states yields generative models of coarticulation because each SGPDM captures the dynamics of phonetic contexts. Figures 5.9a, 5.9b and 5.9c show the visualisation of SGPDMs from the SSGPDM with phoneme VLMM switching states trained on LIPS data and Figures 5.10a, 5.10b and 5.10c show the visualisations for DEMNOW. Only the first three dimensions are shown together with the likelihood space using Jon Conti's Matlab volume rendering codes[1]. The visualisations show the trajectories of the joint latent space of audio and visual data for the same phonetic context. The variations correspond to stylistic differences in the audio and visual realisations of the phonetic contexts. Regions close to the training data are darker, due to the higher likelihood associated with the likelihood decreasing in regions where there is no training data. It is to be noted that the dynamics also leads to smooth paths with no jumps which might be due to the sparsity of the models and the limited variability (due to context) being modelled by a single SGPDM.

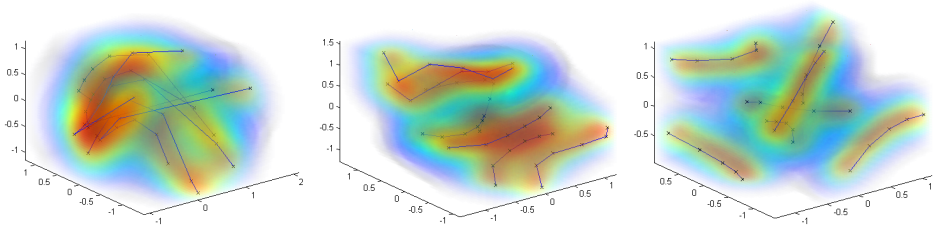
5.3.2 SSGPDM with Phonemes as Switching States

The SSGPDM with phonetic contexts as switching states explicitly models both forward and backward coarticulation (refer to Section 5.4.5). We want to investigate what effect the use of phonetic contexts as switching states has as compared to using phonemes as switching states, which was the approach adopted in the DPDS of Englebienne et al. [102]. In Chapter 6, we evaluate SSGPDMs with both phonetic contexts and phonemes as switching states.

[1] http://www.mathworks.com/matlabcentral/fileexchange/4927-vol3d-m-vol3dtool-m

Algorithm 5 SSGPDM training algorithm for overlapping switching states

Input: $\{\mathbf{Y}_n\}_{n=1}^{S}$, $\{\mathbf{Z}_n\}_{n=1}^{S}$, phonetic stream $\{\boldsymbol{\sigma}_n\}_{n=1}^{S}$, set of strings $Q = \{q_n\}_{n=1}^{K}$ and set of VLMM alphabets, Σ
Output: Trained SGPDM models, $SGPDM\{n\}_{n=1}^{K}$
for $i := 1 : K$ **do**
  $y\_train\{i\} \leftarrow \{\}$
  $z\_train\{i\} \leftarrow \{\}$
  $seq\_delimiter\{i\} \leftarrow \{\}$
  $L \leftarrow length(q_i)$
  for $j := 1 : S$ **do**
    $\mathbf{Y} \leftarrow \mathbf{Y}_j$
    $T \leftarrow length(\mathbf{Y})$
    for $k := 1 : T - L + 1$ **do**
      $test\_str \leftarrow \Sigma_{\{\sigma_t\}_{t=k}^{k+L-1}}$
      if $test\_str = q_i$ **then**
        Add $\{\mathbf{Y}_t\}_{t=k}^{k+L-1}$ to $y\_train\{i\}$
        Add $\{\mathbf{Z}_t\}_{t=k}^{k+L-1}$ to $z\_train\{i\}$
        $len\_sequence \leftarrow len\_sequence + L$
        if $L > 1$ **then**
          Add $len\_sequence$ to $seq\_delimiter\{i\}$
        end if
      end if
    end for
  end for
  if $seq\_delimiter\{i\} = \{\}$ **then**
    Train SGPLVM (without dynamics) *model* for state i using $y\_train\{i\}$ and $z\_train\{i\}$
  else
    Train SGPDM *model* for state i using $y\_train\{i\}$ and $z\_train\{i\}$ and $seq\_delimiter\{i\}$
  end if
  $SGPDM\{i\} \leftarrow model$
end for



(a) (b) (c)

Figure 5.9: SGPDM for VLMM states of the LIPS corpus: (a) VLMM state b oy oy oy oy oy oy. (b) VLMM state l ey ey ey ey ey. (c) VLMM state y y uw uw uw.

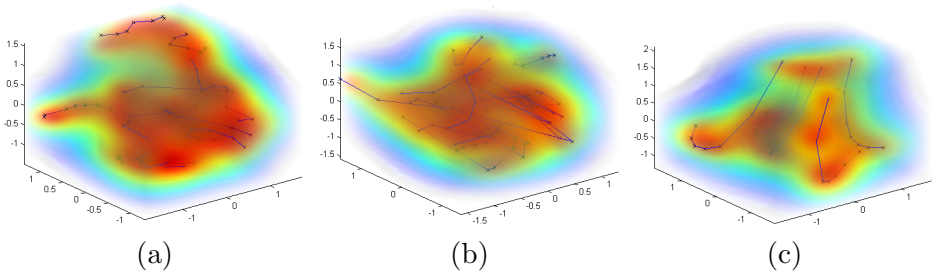


Figure 5.10: SGPDM for VLMM states of the DEMNOW corpus: (a) VLMM state P AY AY AY AY AY AY AY. (b) VLMM state P EY EY EY EY. (c) VLMM state R AY AY AY AY AY AY.

The following describes how to train a SSGPDM with phonemes as switching states. In our audio-visual data corpora, each frame is labelled with a phoneme. We can thus group audio and visual data corresponding to each phoneme together and model them using a SGPDM. This gives rise to a phoneme SSGPDM with the switching states non-overlapping. The pseudocode for the SSGPDM training algorithm for non-overlapping switching states is given in Algorithm 6.

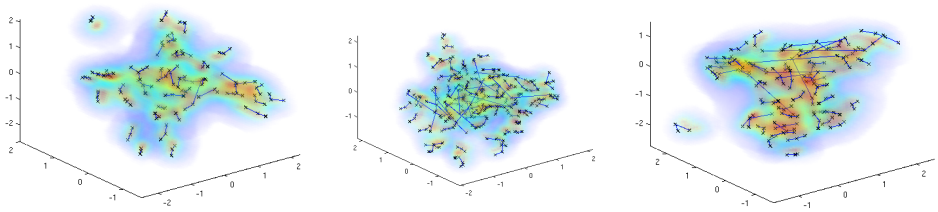
Generative Phonetic Models

Figures 5.11a, 5.11b and 5.11c show the visualisations of SGPDMs from the phoneme SSGPDM trained on LIPS data. Figures 5.12a, 5.12b and 5.12c show the visualisations of SGPDMs from the phoneme SSGPDM trained on DEMNOW data. Only the first three dimensions are visualised together with the likelihood space using the volume rendering codes. The visualisations clearly depict that the lengths of phonetic subsequences are shorter as compared to those found using the VLMM in Figures 5.9 and 5.10. Moreover, there are more variations between different latent trajectories of phoneme subsequences as compared to those of phoneme VLMM subsequences in Figures 5.9 and 5.10. This effect can be explained by coarticulation. In the case of phonemes, the different subsequences have different audio and visual realisations, depending on the context. On the other hand, phoneme VLMM subsequences already encapsulate the context of phonemes and thus there are fewer variations, which might be attributed to style. The phoneme SSGPDM visualisations also reveal more jumps in the latent trajectories, which are due to the larger variability being modelled by a single SGPDM as compared to when phonetic contexts are used as switching states.

Algorithm 6 SSGPDM training algorithm for non-overlapping switching states

Input: $\{\mathbf{Y}_n\}_{n=1}^{S}$, $\{\mathbf{Z}_n\}_{n=1}^{S}$ and $\{\boldsymbol{\pi}_n\}_{n=1}^{S}$ where $\pi \in \{1 \dots K\}$
Output: Trained SGPDM models, $SGPDM\{n\}_{n=1}^{K}$
for $i := 1 : K$ **do**
  for $j := 1 : S$ **do**
    $y\_train\{i\} \leftarrow \{\}$
    $z\_train\{i\} \leftarrow \{\}$
    $seq\_delimiter\{i\} \leftarrow \{\}$
    $T \leftarrow length(\mathbf{Y}_j)$
    for $t := 1 : T$ **do**
      if $\pi_t = i$ **then**
        $len\_sequence \leftarrow 0$
        if $\pi_t = \pi_{t+1}$ **then**
          Add $\mathbf{Y}_j[t]$ to $y\_train\{i\}$
          Add $\mathbf{Z}_j[t]$ to $z\_train\{i\}$
          $len\_sequence \leftarrow len\_sequence + 1$
        else
          Add $len\_sequence$ to $seq\_delimiter\{i\}$
        end if
      end if
    end for
  end for
  Train SGPDM *model* for state i using $y\_train\{i\}$, $z\_train\{i\}$ and $seq\_delimiter\{i\}$

  $SGPDM\{i\} \leftarrow model$
end for



(a) (b) (c)

Figure 5.11: SGPDM for phonetic states of the LIPS corpus: (a) Phoneme state ah. (b) Phoneme state k. (c) Phoneme state eh.

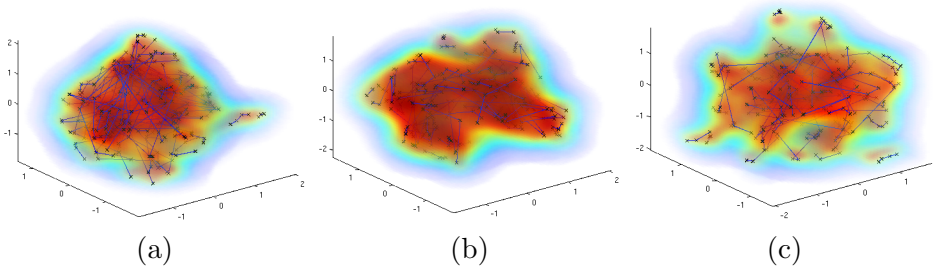


Figure 5.12: SGPDM for Phoneme states of the DEMNOW corpus: (a) Phoneme state AW. (b) Phoneme state G. (c) Phoneme state OW.

5.4 Synthesis using the Switching SGPDM

The synthesis methods we propose for the SSGPDM are sequential, i.e. inference proceeds from beginning to end of the test audio data. It is assumed that the phonetic labels for the test sequence are known and we can then use the method described in Section 5.2.1 to infer the VLMM states from the phoneme sequence. We can then assign each frame to a non-overlapping VLMM state using Algorithm 4. Two scenarios are considered for the synthesis of visual from audio data. The first one assumes that the whole audio is available at the beginning and adopts a batch method. We call it sequential optimisation and more details are given in Section 5.4.1. The second one assumes that the audio data is arriving in an online fashion. We call this method sequential prediction with more details given in Section 5.4.2.

For both methods, an initial estimate of the latent points $\mathbf{X}_{init}$ is found using a nearest-neighbour comparison of the test audio features against the training audio features in the current SGPDM model. The nearest-neighbour initialisation method was found to give the best results in model selection experiments in Chapter 4 Section 4.5.1. In both algorithms, $\mathbf{Y}_{\pi_t}$ and $\mathbf{X}_{\pi_t}$ are the training audio data and latent points, respectively, of the SGPDM model corresponding to the state π_t.

Once the latent points are obtained, the visual features, $\mathbf{Z}^* = \{\mathbf{z}_t^*\}_{t=1}^T$ can be obtained from the mean prediction of the visual observation GP corresponding to the VLMM state at frame t, according to Eqn.5.28.

$$\mathbf{z}_t^* = k_{Z_{\pi_t}}(\mathbf{x}_t^*, \mathbf{X}_{\pi_t})^T \mathbf{K}_{Z_{\pi_t}}^{-1} \mathbf{Z}_{\pi_t} \tag{5.28}$$

where $k_{Z_{\pi_t}}$ is the kernel function for the visual GP of state π_t and $\mathbf{K}_{Z_{\pi_t}}$ is the kernel matrix for the visual GP of state π_t, computed using the corresponding training visual data, and $\mathbf{Z}_{\pi_t}$ is the training visual data of the SGPDM model for state π_t.

5.4.1 Sequential Optimisation

Algorithm 7 describes a sequential optimisation algorithm which assumes the entire test sequence is available from the start, so that synthesis can be performed in an offline manner. Here, the latent point for each frame is locally optimised based on the current SGPDM model, which depends on the current state. If the current state is occupied by only the current frame, a point optimisation (refer to Chapter 4 Section 4.4.1) is carried out, otherwise a sequence optimisation (refer to Chapter 4 Section 4.4.2) is carried out.

Algorithm 7 Sequential optimisation of latent points.

Input: Non-overlapping VLMM states $\{\hat{\pi}_t\}_{t=1}^T$ and audio $\{\hat{\mathbf{y}}_t\}_{t=1}^T$ for test sequence
Output: Inferred latent points $\{\mathbf{x}_t^*\}_{t=1}^T$
$t \leftarrow 1$
while $t \leq T$ **do**
  if $\hat{\pi}_{t+1} \neq \hat{\pi}_t$ **then**
    Point Optimisation - Eqn(1):
    $\mathbf{x}_t^* \leftarrow \arg\max_{\hat{\mathbf{x}}} p(\hat{\mathbf{y}}_t|\hat{\mathbf{x}}, \mathbf{Y}_{\hat{\pi}_t}, \mathbf{X}_{\hat{\pi}_t}, \mathbf{\Phi}_{Y_{\hat{\pi}_t}})$
  else
    $t_i \leftarrow t$
    $\hat{\pi}_s \leftarrow \hat{\pi}_t$
    while $\hat{\pi}_{t+1} = \hat{\pi}_t$ **do**
      $t \leftarrow t + 1$
    end while
    $t_j \leftarrow t$
    Sequence Optimisation - Eqn(2):
    $\mathbf{X}_{t_i:t_j}^* \leftarrow \arg\max_{\hat{\mathbf{X}}} p(\hat{\mathbf{Y}}_{t_i:t_j}, \hat{\mathbf{X}}_{t_i:t_j}|\mathbf{Y}_{\hat{\pi}_s}, \mathbf{X}_{\hat{\pi}_s}, \mathbf{\Phi}_{Y_{\hat{\pi}_s}}, \mathbf{\Phi}_{dyn_{\hat{\pi}_s}})$
  end if
end while

5.4.2 Sequential Prediction

If the assumption is that the data is coming in a sequential manner, then Algorithm 8 allows the prediction of the next frame from the previous frame. This is done by using a point optimisation (refer to Chapter 4 Section 4.4.1) on the first frame of a given state that might span several frames. The dynamical GP is then used to predict the next frames for that state. This is repeated for all audio frames. The sequential prediction algorithm is best suited for real-time applications although the point optimisation might result in some latency, depending on the number of iterations used.

5.4.3 Smoothness Constraint

The algorithms above consider the switching states to be independent. In order to ensure the proper modelling of backward coarticulation, continuity has to be enforced

Algorithm 8 Sequential prediction of latent points.

Input: Non-overlapping VLMM states $\{\hat{\pi}_t\}_{t=1}^{T}$ and audio $\{\hat{\mathbf{y}}_t\}_{t=1}^{T}$ for test sequence
Output: Inferred latent points $\{\mathbf{x}_t^*\}_{t=1}^{T}$
$t \leftarrow 1$
$\mathbf{x}_t^* \leftarrow \arg\max_{\hat{\mathbf{x}}} p(\hat{\mathbf{y}}_t|\hat{\mathbf{x}}, \mathbf{Y}_{\hat{\pi}_t}, \mathbf{X}_{\hat{\pi}_t}, \mathbf{\Phi}_{Y_{\hat{\pi}_t}})$
while $t \leq T-1$ **do**
  if $\hat{\pi}_{t+1} \neq \hat{\pi}_t$ **then**
    Point Optimisation - Eqn(3):
    $\mathbf{x}_{t+1}^* \leftarrow \arg\max_{\hat{\mathbf{x}}} p(\hat{\mathbf{y}}_{t+1}|\hat{\mathbf{x}}_{t+1}, \mathbf{Y}_{\hat{\pi}_{t+1}}, \mathbf{X}_{\hat{\pi}_{t+1}}, \mathbf{\Phi}_{Y_{\hat{\pi}_{t+1}}})$
  else
    $\mathbf{x}_{t+1}^* \leftarrow h_{\hat{\pi}_t}(\mathbf{x}_t^*)$
  end if
  $t \leftarrow t+1$
end while

in the visual features even when switching from the SGPDM of the current VLMM state to the SGPDM of the next state. Each SGPDM has a local dynamical model that caters for the coarticulatory dynamic of the current speech segment. An additional smoothness constraint needs to be enforced when switching from one SGPDM to the next. To deal with this, we introduce an additional term in the likelihood that is to be optimised to find latent points. This is done by formulating a joint likelihood between the test audio features of the current frame and the visual features of the previous frame. Given the latent point of the previous frame, $\mathbf{x}_{t-1}^*$, the term $p(\hat{\mathbf{z}}_{t-1}|\mathbf{x}_{t-1}^*, \mathbf{Z}_{\hat{\pi}_{t-1}}, \mathbf{X}_{\hat{\pi}_{t-1}}, \mathbf{\Phi}_{Z_{\hat{\pi}_{t-1}}})$ is multiplied to the likelihood function of **Eqn(1)** and **Eqn(2)** in Algorithm 7 as well as **Eqn(3)** in Algorithm 8. For **Eqn(1)**, the likelihood function to be optimised becomes:

$$\begin{aligned} &p(\hat{\mathbf{y}}_t, \hat{\mathbf{z}}_{t-1}|\hat{\mathbf{x}}_t, \mathbf{x}_{t-1}^*, \mathbf{Y}_{\hat{\pi}_t}, \mathbf{X}_{\hat{\pi}_t}, \mathbf{Z}_{\hat{\pi}_{t-1}}, \mathbf{X}_{\hat{\pi}_{t-1}}, \mathbf{\Phi}_{Y\hat{\pi}_t}, \mathbf{\Phi}_{Z_{\hat{\pi}_{t-1}}}) \\ =&p(\hat{\mathbf{y}}_t|\hat{\mathbf{x}}_t, \mathbf{Y}_{\hat{\pi}_t}, \mathbf{X}_{\hat{\pi}_t}, \mathbf{\Phi}_{Y\hat{\pi}_t})p(\hat{\mathbf{z}}_{t-1}|\mathbf{x}_{t-1}^*, \mathbf{Z}_{\hat{\pi}_{t-1}}, \mathbf{X}_{\hat{\pi}_{t-1}}, \mathbf{\Phi}_{Z_{\hat{\pi}_{t-1}}}) \end{aligned} \tag{5.29}$$

Optimising this joint likelihood constrains the visual features of the first frame synthesised from a given SGPDM model of VLMM state $\hat{\pi}_t$ to be similar to the last visual features belonging to the SGPDM model of the previous VLMM state, $\hat{\pi}_{t-1}$, thus ensuring continuity across states.

It is to be noted that switching state-space models like the SLDS and DPDS do not need a smoothness constraint in synthesis because the state-space is continuous with the propagation of the continuous states at the discrete state boundaries. On the other hand, the SSGPDM is a segmented model with each new discrete state requiring sampling of the continuous states from the corresponding SGPDM. As a result, the smoothness constraint is necessary to ensure continuous visual trajectories in synthesis.

5.4.4 Leading and Trailing Pauses

The synthesis techniques described cater only for speech-related facial gestures, without consideration for non-verbal cues such as pauses before and after the sentence. In order to generate videorealistic output, we need to cater for the leading and trailing pauses.

The video sequences in the corpora have either silences or breaths before and after the utterance. The methods described so far have only considered the utterance in between, excluding the silence and breath frames. In order to generate a realistic animation, we need to synthesise visual features for these frames as well. We do this by learning separate SGPDMs between the audio and visual features for the leading and trailing silences and breaths. Then in the synthesis, we try to find intermediate latent points that maximise the joint likelihood of the audio data and latent points given the parameters of the corresponding trained SGPDM. From the latent points, the visual features can be synthesised, as described previously. Eqn.5.30 illustrates the optimisation of latent points for silence frames.

$$\mathbf{X}^*_{sil} = \arg\max_{\hat{\mathbf{X}}} p(\hat{\mathbf{Y}}_{sil}, \hat{\mathbf{X}}_{sil} | \mathbf{Y}_{sil}, \mathbf{X}_{sil}, \mathbf{\Phi}_{Y_{sil}}, \mathbf{\Phi}_{dyn_{sil}}) \tag{5.30}$$

where $\mathbf{Y}_{sil}$ are the training audio data and $\mathbf{X}_{sil}$ the training latent points, $\mathbf{\Phi}_{Y_{sil}}$ are the hyperparameters of the audio GP and $\mathbf{\Phi}_{dyn_{sil}}$ are the hyperparameters of the dynamical GP for the silence SGPDM.

This approach works because generally the frames that are labelled as silence are not really silence but contain lower amplitude sounds preceeding or following the utterance being made. The SGPDM is thus able to capture the correlations between the corresponding audio and visual features. For breath, the audio will consist of a hissing sound and the visual data will correspond to the opening of the mouth for inspiration.

A low-pass filter [131] is applied to the final synthesised visual features in order to smooth the animation and minimise jumps, thus creating a seamless animation that blends well at the boundaries. The Matlab function `interp` is used for that purpose.

5.4.5 Modelling Coarticulation

Using VLMM states as switching states for the SSGPDM and in conjunction with the sequential optimisation method presented earlier, we explicitly model both preservatory and anticipatory coarticulation (refer to Chapter 2 Section 2.1.5 for more details on coarticulation). The dynamics of phonetic contexts are captured by modelling each VLMM state using a SGPDM model. When synthesising the visual parameters, previous phonetic context is taken into account by the smoothness constraint mentioned in the previous section. This accounts for *preservatory* coarticulation. In addition, each

VLMM state encapsulates the context of phonemes. The sequential optimisation algorithm takes into account future phonemes that occur within a particular VLMM state, thus accounting for *anticipatory* coarticulation. The sequential prediction method also has a model of anticipatory coarticulation, because the dynamical GP should predict the coarticulatory dynamics within a given VLMM state. However, the model of anticipatory coarticulation is weaker than when using sequential optimisation because information from the audio is not used to infer the state-space, which can lead to higher ambiguity in the synthesis.

Our hypothesis is that explicitly modelling forward and backward coarticulation would lead to better results in visual speech synthesis. In order to test this, we need to compare our results against a model that does not model anticipatory coarticulation, which is the case for the SSGPDM with phonemes as switching states. If phonemes are used as the switching states for the SSGPDM, then there is no model of forward coarticulation because the next phonetic states are unknown and therefore no consideration is made for these using both sequential optimisation and prediction, which tends to yield short-term bursts of mouth opening with no anticipation of future behaviour. However, using the smoothness constraint, anticipatory coarticulation is still taken into account, which can lead to acceptable animations for the English language, given that English is predominantly anticipatory (refer to Chapter 2 Section 2.1.5). In Chapter 6, we present experiments to compare the two switching state representations for the SSGPDM as well as the two synthesis algorithms proposed.

5.5 Discussion

The application of the SSGPDM to visual speech synthesis addresses the limitations of the SGPDM. The first limitation of the SGPDM is the limit to the number of frames that can be modelled using a single SGPDM. With the SSGPDM, we can segment the data such that each switching state models a given substructure within the data. The substructures can either be phonemes or phonetic contexts obtained using a VLMM trained on phonemes. As a result, we can use a lot more training data with the SSGPDM as compared to the SGPDM, because each switching state from the SSGPDM would be modelled using a SGPDM, leading to a partitioning of the training data across SGPDMs. The second limitation of the SGPDM is that a single dynamical and observation model is used to account for the whole data, which is not a valid assumption. With the SSGPDM, different observation and dynamical models are used to represent the multiple dynamics involved in speech. The dynamics of phonetic contexts can be highly non-linear and are thus suitable for modelling using non-linear Gaussian processes (GPs).

By having a joint probabilistic model of audio and visual data, audio-visual speech

synthesis can also be achieved from a sequence of phonemes. This requires that we use source-filter speech features such as LPC or LSF. Given a stream of phonemes, we can then use the SSGPDM in prediction mode to generate a sequence of latent points, from which we can generate both the audio and visual features.

One limitation of the SSGPDM, however, is that training takes a long time and the saved models occupy significant space. For the LIPS SSGPDM, 1.2Gb of storage space is required whilst for the DEMNOW SSGPDM, the requirement is about 1.8Gb of storage space, due to size of the kernel matrices of the GPs.

5.6 Chapter Summary

The main aim of this chapter was to address the limitations of the SGPDM by augmenting it with switching states, yielding the switching SGPDM (SSGPDM). We presented an introduction to the SSGPDM before delving into an efficient way of finding switching states corresponding to commonly occuring phonetic contexts using the variable length Markov model (VLMM). Training and inference algorithms for the VLMM were also described. Experiments were presented to determine the optimal memory length of the VLMM as well as whether the VLMM should be trained on repeating or non-repeating phonemes. We then presented training algorithms for the SSGPDM when phonetic contexts are used as switching states as well as when phonemes are used as switching states. We also showed visualisations of latent spaces in 3D for various states of SSGPDMs with both phoneme and phoneme VLMM switching states, using volume rendering Matlab code. Synthesis algorithms for audio-visual mapping using the SSGPDM were derived both for the batch and online processing scenarios. Finally, post-processing steps to cater for leading and trailing poses were described.

Chapter 6

Evaluation

> Extraordinary claims require extraordinary evidence.
>
> Carl Sagan

This chapter presents an evaluation of our visual speech synthesis methods. Only objective evaluation is done for the SGPDM because it is not a full-fledged visual speech synthesiser. It forms the basis for our more powerful synthesiser, namely the SSGPDM. The results from the SSGPDM are therefore evaluated both objectively and subjectively. We also compare the objective results of the SGPDM with those of other state-space models and objective results of the SSGPDM with those of other state-space and switching state-space models that have been previously applied to visual speech synthesis. In this chapter, we shall refer to ground truth as the features or videos corresponding to real visual speech sequences parameterised with the active appearance model (AAM).

6.1 Evaluation Methods for Visual Speech Synthesis

The quality of visual speech synthesis can be measured using both objective and subjective methods. Objective approaches involve measuring the error or correlation between real and synthetic visual features [103, 278, 63, 81, 188, 101, 270] as well as comparing the evolution of their trajectories over time. Error measures such as L_1-norm , L_2-norm, L_∞-norm and average mean squared error (AMSE), as well as correlation measures such as average correlation coefficient (ACC), provide a measure of the static comparison between frames of real and synthetic visual features. Comparison of real and synthetic visual feature trajectories gives an indication of the dynamic correlation between the two. Recently, Xie and Liu [313] and Englebienne [101] also proposed automated lip-reading tests as a measure of the intelligibility of synthetic visual speech.

Whilst both the error and correlation are good indicators of how the real and synthetic features compare statistically, they do not provide a measure of the realism, naturalness and intelligibility of the artificial talking head. Specifically, the synthetic animation might have jitter or be asynchronous with the audio and this is hard to measure statistically. As a result, subjective evaluation was found to be essential to evaluate synthetic talking heads [53, 120, 278, 63, 129, 196, 312, 313, 282, 101, 208, 158]. Realism can be measured using Turing tests, where viewers are shown real and synthetic videos either in pairs [120, 101] or one at a time [120, 313] and asked to choose which sequences are real and synthetic. Naturalness can be measured by using scoring tests, which involve asking viewers to rate the level of naturalness on an ordinal scale, e.g. from 1 to 5 [278, 129, 196, 312, 313]. Intelligibility can be measured by asking viewers to lip-read videos with sound turned off [137, 120].

Unfortunately, there are no benchmarks to compare the performances of different visual speech synthesisers because different researchers use different datasets as well as different objective and subjective evaluations. Recently, Theobald et al. [281] attempted to induce interest in the research community to come up with standardised benchmarks by having the LIPS corpus as well as a visual speech synthesis challenge. However, researchers use different parameterisations for the face, which makes benchmarking for objective evaluation difficult. Instead, subjective evaluation was used to compare the results of participants in the visual speech synthesis challenge [281].

Theobald [278] also described a way of correlating objective and subjective results, such that for future tests, objective evaluation alone can be used to predict the subjective evaluation results. However, Englebienne [101] showed that objective results do not necessarily correlate with subjective results because a synthesiser that gave less favourable objective results was found to be perceptually better based on subjective tests. We thus present objective and subjective results independently.

6.2 Objective Evaluation

Objective evaluation techniques for synthetic visual speech consist of: error measures, correlation measures, visual features trajectory comparison and visual or audio-visual speech recognition using ground truth and synthetic visual features. As mentioned in Chapter 5 Section 5.4.4, we also perform synthesis for leading and trailing pauses for the SSGPDM in order to generate a more natural visual output. However, Theobald [278] pointed out that objective evaluation should only be done for the speech-related visual frames, at the exclusion of leading and trailing pauses. This is because the synthesiser's prediction for leading and trailing pauses may not necessarily correlate well with ground truth due to the variations of facial gestures possible, although the synthesised output is plausible. This would lead to significant errors which could

mask errors or correlation associated with speech. As a result, we perform objective evaluation for speech-related visual frames only.

The following presents a review of objective tests commonly carried out for visual speech synthesis.

6.2.1 Error Measures

If we have two univariate vectors $\mathbf{z} = \{z_t\}_{t=1}^T$ and $\hat{\mathbf{z}} = \{\hat{z}_t\}_{t=1}^T$, the difference between them can be measured using the L_1, L_2 or L_∞-norms.

The L_1-norm, also known as the sum of absolute differences (SAD), is given by:

$$L_1 = \frac{1}{T}\sum_{t=1}^{T} |(z_t - \hat{z}_t)| \tag{6.1}$$

The L_2-norm, also known as the root mean squared error (RMSE), is given by:

$$L_2 = \sqrt{\frac{1}{T}\sum_{t=1}^{T} (z_t - \hat{z}_t)^2} \tag{6.2}$$

The mean squared error (MSE) is the square of the RMSE:

$$\text{MSE} = \frac{1}{T}\sum_{t=1}^{T} (z_t - \hat{z}_t)^2 \tag{6.3}$$

The L_∞-norm, also known as the maximum absolute error (MAE), is given by:

$$L_\infty = \max_{1\leq t\leq T} (z_t - \hat{z}_t) \tag{6.4}$$

If we have multivariate data, then the average mean squared error is more commonly used. Let $\mathbf{Z} = \{\mathbf{z}_t\}_{t=1}^T$ and $\hat{\mathbf{Z}} = \{\hat{\mathbf{z}}_t\}_{t=1}^T$ be two sequences of aligned D-dimensional vectors. The average mean squared error (AMSE) can be computed according to:

$$\text{AMSE} = \frac{1}{TD}\sum_{t=1}^{T}\sum_{d=1}^{D} (z_{t,d} - \hat{z}_{t,d})^2 \tag{6.5}$$

The L_1, L_2 and L_∞-norms for multivariate data are given by:

$$L_1 = \frac{1}{TD}\sum_{t=1}^{T}\sum_{d=1}^{D} |(z_{t,d} - \hat{z}_{t,d})| \tag{6.6}$$

$$L_2 = \sqrt{\frac{1}{TD}\sum_{t=1}^{T}\sum_{d=1}^{D} (z_{t,d} - \hat{z}_{t,d})^2} \tag{6.7}$$

$$L_\infty = \frac{1}{T}\sum_{t=1}^{T}(\max_{1\leq d\leq D} (z_{t,d} - \hat{z}_{t,d})) \tag{6.8}$$

Englebienne [101] presented quantitative results in terms of L_1, L_2 and L_∞-norms. The L_2 norm gives a more average-case statistical comparison whilst the L_1 and L_∞ are better at spotting extreme deviations from ground truth. In our case, we use

subjective tests to potentially idenfity extreme variations such as lack of smoothness in visual speech synthesis. We therefore use the AMSE as the error measure between ground truth and synthetic visual features as has been done by researchers such as Gutierrez-Osuna et al. [129] and Xie and Liu [312].

6.2.2 Correlation Measures

Given two aligned D-dimensional vectors $\mathbf{Z} = \{\mathbf{z}_t\}_{t=1}^{T}$ and $\hat{\mathbf{Z}} = \{\hat{\mathbf{z}}_t\}_{t=1}^{T}$, the average correlation coefficient (ACC) is computed according to:

$$\text{ACC} = \frac{1}{TD} \sum_{t=1}^{T} \sum_{d=1}^{D} \frac{(z_{t,d} - \mu_d)(\hat{z}_{t,d} - \hat{\mu}_d)}{\sigma_d \hat{\sigma}_d} \tag{6.9}$$

where μ_d is the mean of the dth dimension of $\mathbf{y}$ across the frames from 1 to T and σ_d is the corresponding variance and $\hat{\mu}_d$ is the mean of the dth dimension of $\hat{\mathbf{z}}$ across all frames and $\hat{\sigma}_d$ is the corresponding variance.

The ACC was used by Xie and Liu [312], Gutierrez-Osuna et al. [129] and Tao et al. [270] in evaluating their visual speech synthesis methods. We found from our experiments, that high ACC is more closely related to high visual quality of synthesised animations as compared to low AMSE. This is because under-articulated animations might give low AMSE whilst not looking visually plausible. This is the reason why we used the ACC in the model selection and optimal speech parameterisation experiments in Chapter 4. We therefore also use ACC in our objective evaluation on test data.

6.2.3 Visual Feature Trajectories Comparison

Several researchers show the plots of individual modes of visual trajectories for ground truth and the corresponding synthetic sequences, in order to show how they compare through time frames [129, 117, 313, 192].

Other researchers have tried to average across all modes of variations [312] or alternatively, across some of the modes of variation relevant to the evaluation between carried out [158]. In our case, because we use normalised AAM parameters with modes of variations corresponding to expressive cues and pose variations removed, we plot the trajectories of visual parameters averaged across all dimensions. This is because the average would correspond only to speech-related content, which is what we are willing to evaluate. However, this approach reflects tends to depict the variation in AAM modes of higher scale whilst masking the trend in those of lower scale. Quantitative results in terms of error and correlation measures give a more accurate view of the overall performance of the visual speech synthesiser.

Given two aligned D-dimensional vectors $\mathbf{Z} = \{\mathbf{z}_t\}_{t=1}^{T}$ and $\hat{\mathbf{Z}} = \{\hat{\mathbf{z}}_t\}_{t=1}^{T}$, corresponding to real and synthesised visual features, we can compute the mean across all

dimensions for any given frame t as follows:

$$u_t = \frac{1}{D}\sum_{d=1}^{D} z_{t,d} \tag{6.10}$$

$$\hat{u}_t = \frac{1}{D}\sum_{d=1}^{D} \hat{z}_{t,d} \tag{6.11}$$

$$\tag{6.12}$$

We can then plot the time-series evolution of u_t and $\hat{u}_t$ as trajectories on a single plot to compare the evolution of the visual features, which can give an indication of how well the dynamics of the synthesised features match those of ground truth features.

6.2.4 Automated Lip-reading

More recently, some researchers have proposed to use either automated lip-reading tests using either visual features only or combined audio and visual features in audio-visual speech recognition (AVSR) to evaluate the information content of synthetic visual features as compared to ground truth. Englebienne [101] used a constrained lip-reading method to evaluate synthetic visual speech. Single word utterances were synthesised using the DPDS and the multiple phonetic transcriptions possible for that word were unrolled using a phoneme HMM. The DPDS was then used to compute the likelihood of the visual features given each transcription and the transcription with the highest likelihood was chosen as the optimal, which was then compared against the ground truth transcription. It was found that the DPDS gave the highest recognition performance as compared to other methods [101]. Xie and Liu [313] applied audio-visual speech recognition using a multistream HMM [195] with synthetic visual features and audio features corrupted with noise at different signal-to-noise (SNR) ratios, in order to evaluate the different variants of the proposed visual speech synthesis technique.

Hilder et al. [137] compared human vs. automated lip-reading by training word-level HMMs with phonemes as states. The visual features used comprised of shape-only as well as combined shape and texture AAM features. The recognition experiment was to both recognise individual words being uttered as well as the phonemes and visemes within these words. It was found that humans are better at spotting words as compared to computers but the recognition at phoneme and viseme level was much higher using computers.

The use of audio-visual speech recognition to evaluate the different visual speech synthesis methods would provide some useful feedback about the merits and demerits of the different techniques. However, due to time constraints, this was not carried out and is a worthy line of research in future work. In this work, we have restricted ourselves to human lip-reading as described in Section 6.3.3.

6.2.5 Experiments

We perform quantitative evaluation for both SGPDM and SSGPDM by comparing ground truth against synthetic visual data using AMSE and ACC. We also plot ground truth and synthetic AAM trajectories to compare the dynamics of the synthetic AAM features against ground truth.

SGPDM Results

A training set of 50 sequences have been used for training SGPDM models for both LIPS and DEMNOW, using the optimal model parameters found in Chapter 4 and using both a KBR back-constraint with respect to audio and an autoregressive dynamics on the latent space. The speech features used for LIPS are RASTA-PLP processed at 25Hz, whilst for DEMNOW, we use MFCC features processed at 100Hz and downsampled to 29.97Hz using polyphase quadrature filtering. These speech features were found to be the best predictors of visual AAM features for LIPS and DEMNOW in experiments presented in Chapter 4 Section 4.5. We used 20 sequences, different from the validation set used in Chapter 4, for testing. We have also trained Brand's model [32], the shared LDS [186] and the coupled HMM [312] on the 50 training sequences and performed synthesis on the 20 test sequences. The results are summarised in Table 6.1. The visual trajectories of ground truth and synthetic AAM features for four test sequences of LIPS and four test sequences of DEMNOW are shown in Figures 6.1 and 6.2, respectively.

The results show that the SGPDM performs better than the Voice Puppetry of Brand [32], the coupled HMM of Xie and Liu [312] and the shared LDS of Lehn-Schiøler et al. [186]. This supports our hypothesis that having a joint non-parametric and non-linear state-space model of audio and video performs better than parametric models that are linear [186] or locally linear [32, 312]. The shared LDS uses linear dynamical and observation mappings as opposed to the non-linear mappings in the SGPDM, which accounts for the better quantitative results of the SGPDM compared to the shared LDS, thus supporting our claim that using a non-linear state-space model is a better generative model of speech as compared to a linear state-space model. The methods of Brand [32] and Xie and Liu [312] use piecewise linear models that cluster speech behaviour into locally linear subspaces represented by Gaussian distributions. This approach allows the non-linearities in audio-visual mapping to be better modelled than the shared LDS [186], thus resulting in better quantitative results as compared to the shared LDS.

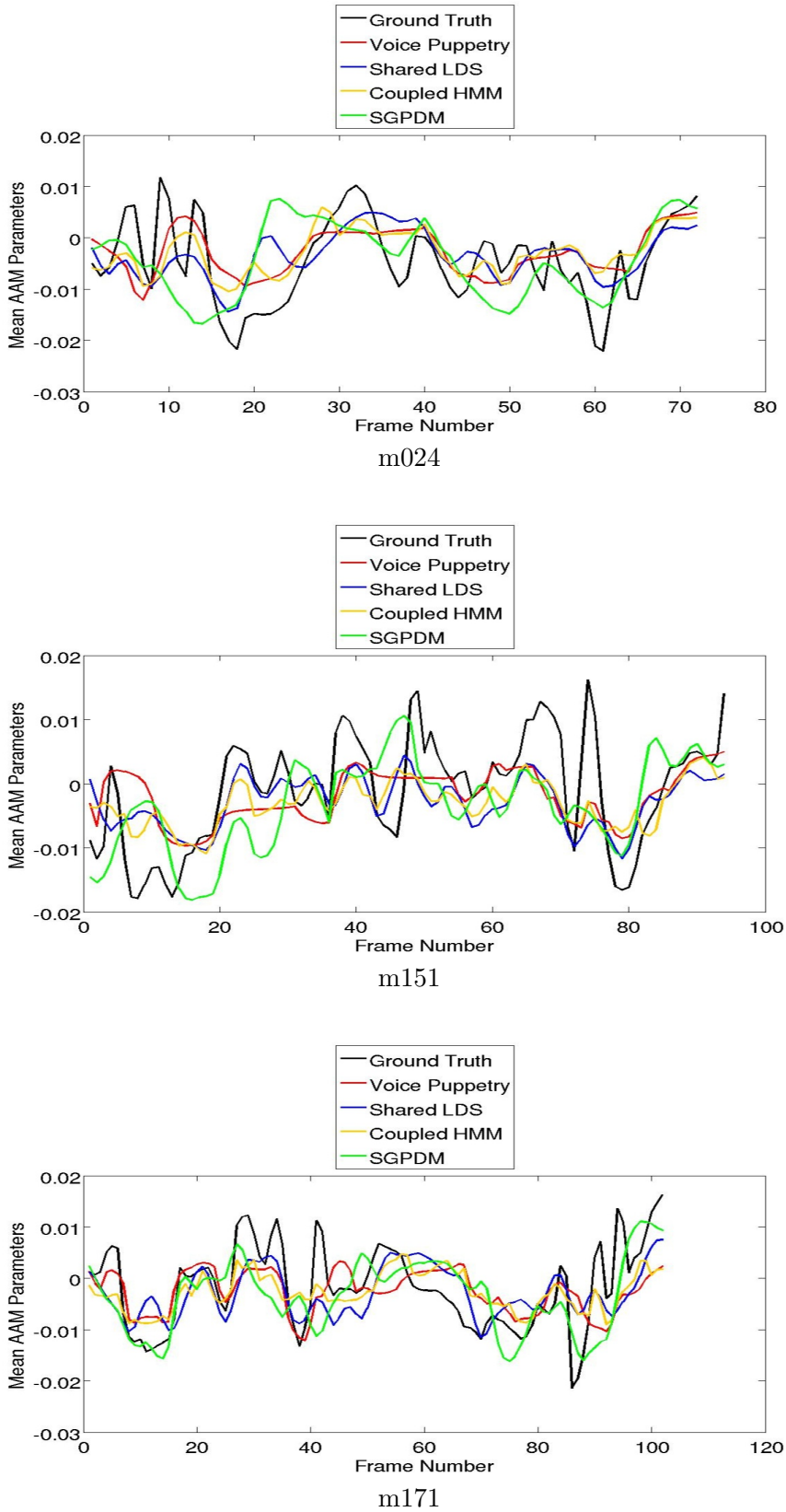


Figure 6.1: Comparing ground truth and synthethic trajectories for: Voice Puppetry [32], shared LDS [186], Coupled HMM [312] and SGPDM for three test LIPS sequences.

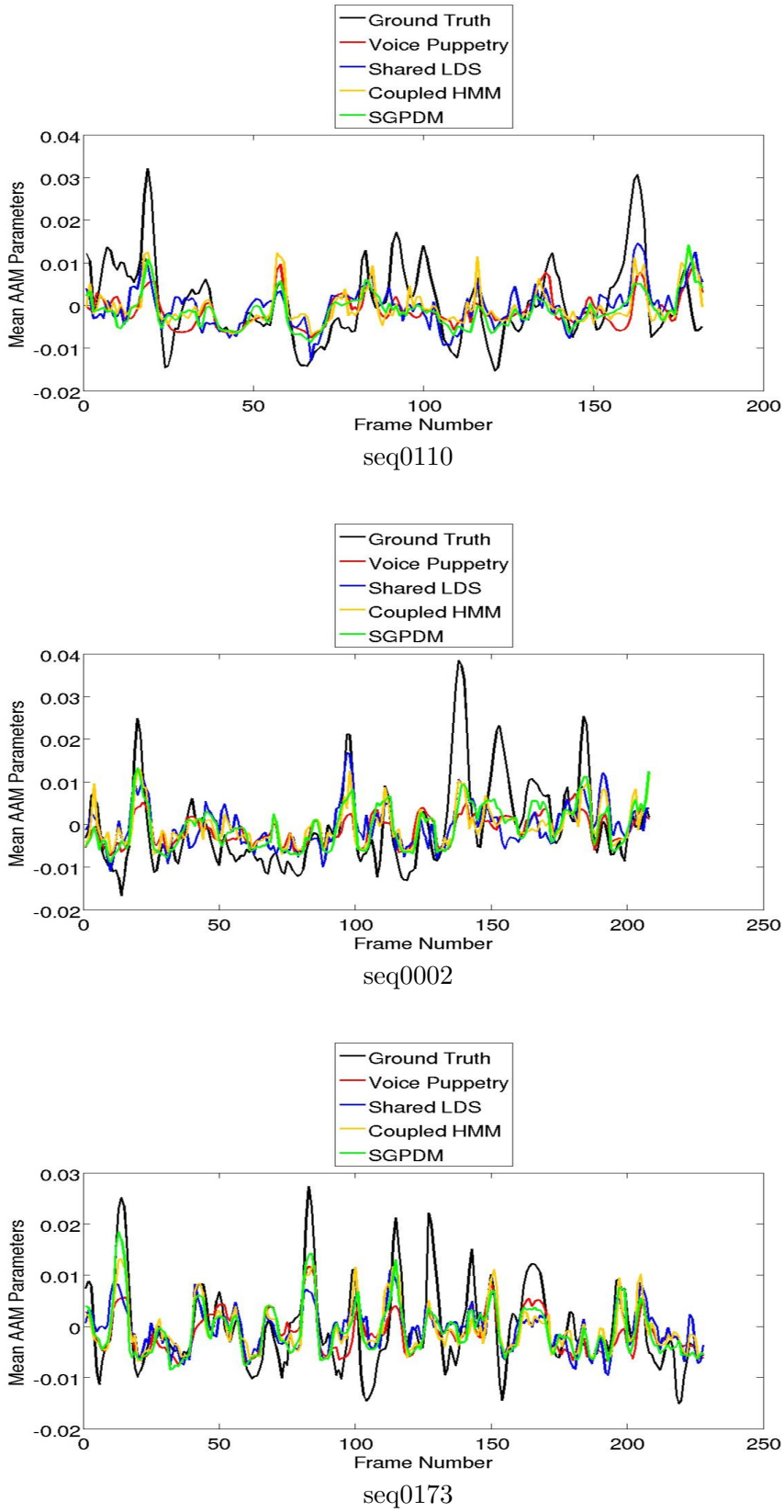


Figure 6.2: Comparing ground truth and synthethic trajectories for: Voice Puppetry [32], shared LDS [186], Coupled HMM [312] and SGPDM for three test DEMNOW sequences.

		LIPS		DEMNOW	
Method	Synthesis Method	AMSE	ACC	AMSE	ACC
SGPDM	Sequence Optimisation	0.0535 ± 0.0090	0.4595 ± 0.0930	0.0210 ± 0.0087	0.5594 ± 0.1836
Shared LDS	Kalman Filtering	0.0546 ± 0.0121	0.4394 ± 0.0872	0.0242 ± 0.0089	0.4699 ± 0.0878
Brand	Geodesic Interpolation	0.0544 ± 0.0095	0.4557 ± 0.0659	0.0221 ± 0.0095	0.5548 ± 0.1235
Coupled HMM	Baum-Welch Inversion	0.0537 ± 0.0094	0.4565 ± 0.0689	0.0219 ± 0.0092	0.5541 ± 0.1100

Table 6.1: SGPDM quantitative evaluation results for LIPS and DEMNOW datasets.

			LIPS		DEMNOW	
Method	Switching State	Synthesis method	AMSE	ACC	AMSE	ACC
SSGPDM	Phoneme VLMM	Sequential Optimisation	0.0413 ± 0.0063	0.6681 ± 0.0603	0.0123 ± 0.0036	0.6739 ± 0.0786
SSGPDM	Phoneme VLMM	Sequential Prediction	0.0436 ± 0.0085	0.6431 ± 0.0740	0.0137 ± 0.0046	0.6533 ± 0.0649
SSGPDM	Phoneme	Sequential Optimisation	0.0560 ± 0.0105	0.5305 ± 0.0691	0.0174 ± 0.0050	0.5471 ± 0.1236
SSGPDM	Phoneme	Sequential Prediction	0.0599 ± 0.0110	0.4947 ± 0.0763	0.0206 ± 0.0051	0.5011 ± 0.1145
DPDS	Phoneme	ML Prediction	0.0470 ± 0.0115	0.6061 ± 0.0778	0.0160 ± 0.0043	0.5600 ± 0.1040
Brand	N/A	Geodesic Interpolation	0.0514 ± 0.0086	0.5341 ± 0.0945	0.0146 ± 0.0044	0.6195 ± 0.0917
Coupled HMM	N/A	Baum-Welch Inversion	0.0465 ± 0.0091	0.5967 ± 0.0722	0.0161 ± 0.0046	0.5559 ± 0.0872
Shared LDS	N/A	Kalman Filtering	0.0568 ± 0.0099	0.4641 ± 0.0890	0.0188 ± 0.0049	0.4365 ± 0.0919

Table 6.2: SSGPDM quantitative evaluation results for LIPS and DEMNOW datasets.

SSGPDM Results

For the SSGPDM experiments, we again use, for each SGPDM model comprising the SSGPDM, the optimal parameters found using the model selection experiments in Chapter 4. The visual features used are the normalised AAM parameters and the audio features are the same as for the SGPDM, i.e. RASTA-PLP processed at 25Hz for LIPS and MFCC downsampled from 100Hz to 29.97Hz using polyphase quadrature filtering for DEMNOW.

For the LIPS dataset, we used 250 sequences for training and 28 sequences for testing whilst for DEMNOW, we used 550 sequences for training and 100 sequences for testing. The training and test sets are non-overlapping. We train SSGPDM models with phoneme VLMM switching states as well as with phonemes as switching states. We compare synthesis results obtained using both sequential optimisation and sequential prediction. Our results are compared against the Voice Puppetry of Brand [32], the DPDS of Englebienne et al. [102], the coupled HMM of Xie and Liu [312] and the shared LDS of Lehn-Schiøler [185] using the same training sequences as for the SSGPDM. The corresponding quantitative results are presented in Table 6.2.

The results show that the SSGPDM using phoneme VLMM as the switching states and sequential optimisation as the synthesis method gives the best results for both LIPS and DEMNOW. We attribute these results to three factors. First, both continuous and discrete audio are used to make predictions of visual speech in our method, thus integrating acoustic information with language structure. Second, we take into account both forward and backward context in our synthesis framework, thus accounting for preservatory and anticipatory coarticulation. Third, we use non-parametric GPs in our switching state-space model, which generates results that best match ground truth as compared to parametric models such as the HMM [32], coupled HMM [312], shared LDS [185] and DPDS [102]. Our results are a big improvement over the DPDS because

we model both audio and video with non-linear dynamics and observation functions at the level of phonetic contexts, as opposed to the DPDS which models only video using linear observation and dynamical mappings at the level of phonemes.

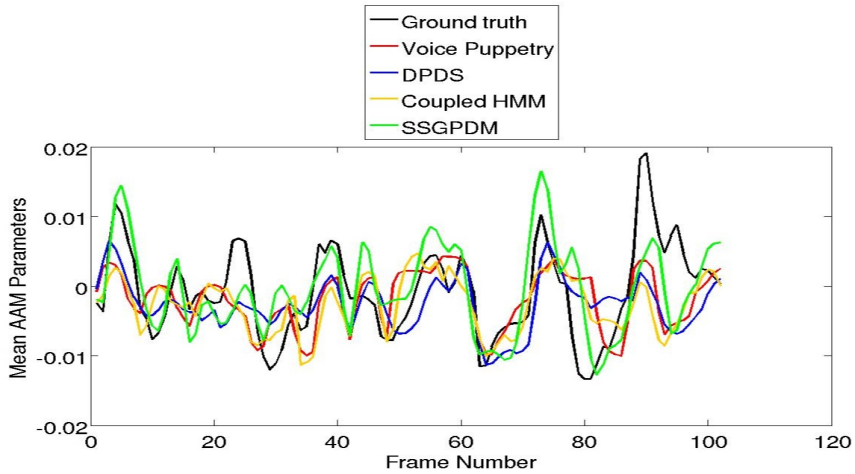
We also compare the trajectories of AAM parameters for ground truth against synthetic AAM parameters obtained using: the SSGPDM that gave best quantitative results (SSGPDM with phoneme VLMM as switching states and sequential optimisation as synthesis method), Voice Puppetry [32], DPDS [102] and the coupled HMM [312]. The plots for four test sequences of LIPS are shown in Figure 6.3 and four test sequences of DEMNOW are shown in Figure 6.4. The plots for the shared LDS have been omitted in order not to clutter the diagrams. The figures clearly show the high correlation between AAM trajectories obtained from SSGPDM and ground truth as compared to other methods, which supports the quantitative results.

6.2.6 Discussion

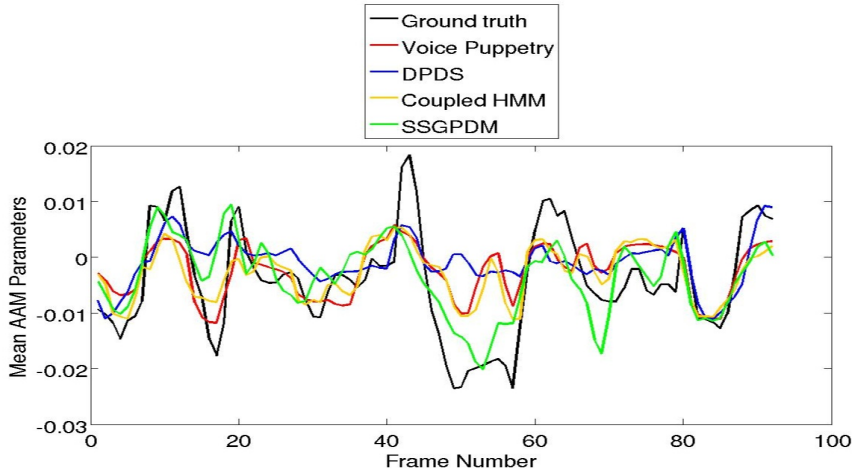
We have compared the results of the SGPDM against the state-space models of Brand [32], Lehn-Schiøler et al. [186] and Xie and Liu [312]. The results of the SSGPDM have also been compared against the methods of Brand [32] and Xie and Liu [312] in addition to the closely-related switching state-space model of Englebienne et al. [102]. It is to be noted that the works of Brand [32] and Englebienne et al. [102] do not require normalisation of the AAM data in order to have frontal pose because the synthesis is driven from a sequence of discrete labels that uses either interpolation [32] or a dynamical mapping [186] to predict a smooth path through the AAM parameters. This cannot be enforced to the same extent when we have a joint probabilistic model of audio and video because the correlations between the pose and the audio would be captured and this would result in spurious pose variations in synthesis. Thus, we had to adopt the normalisation procedures mentioned in Chapter 3 Section 3.4.1. The normalisation procedure also gives us a standardised frame of reference to measure the errors and correlation between ground truth and synthesised AAM parameters.

In Englebienne [101] it was reported the quantitative results were not reliable because the DPDS gave worse quantitative results than the Voice Puppetry of Brand [32] whilst subjective evaluation found that participants confused animations generated by the DPDS with ground truth more than was the case for Voice Puppetry animations. One possible reason could be that pose differences between ground truth and synthetic visual features led to quantitative results that were not very meaningful. In our work, the pose normalisation procedure makes the quantitative results more meaningful because they correspond to visual features related to speech movements in a normalised coordinate space.

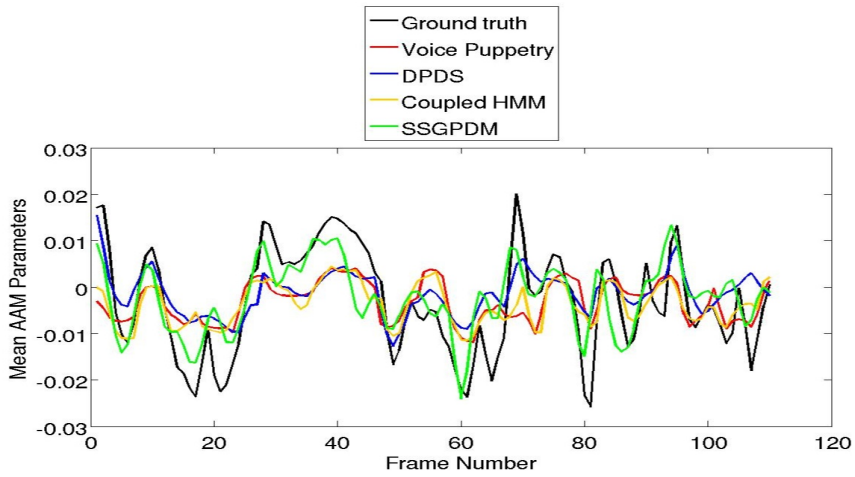
Our experiments demonstrate that the Voice Puppetry method [32], shared LDS



m203



m099



m108

Figure 6.3: Comparing ground truth and synthethic trajectories for the: Voice Puppetry [32], DPDS [102], Coupled HMM [312] and SSGPDM for three test LIPS sequences.

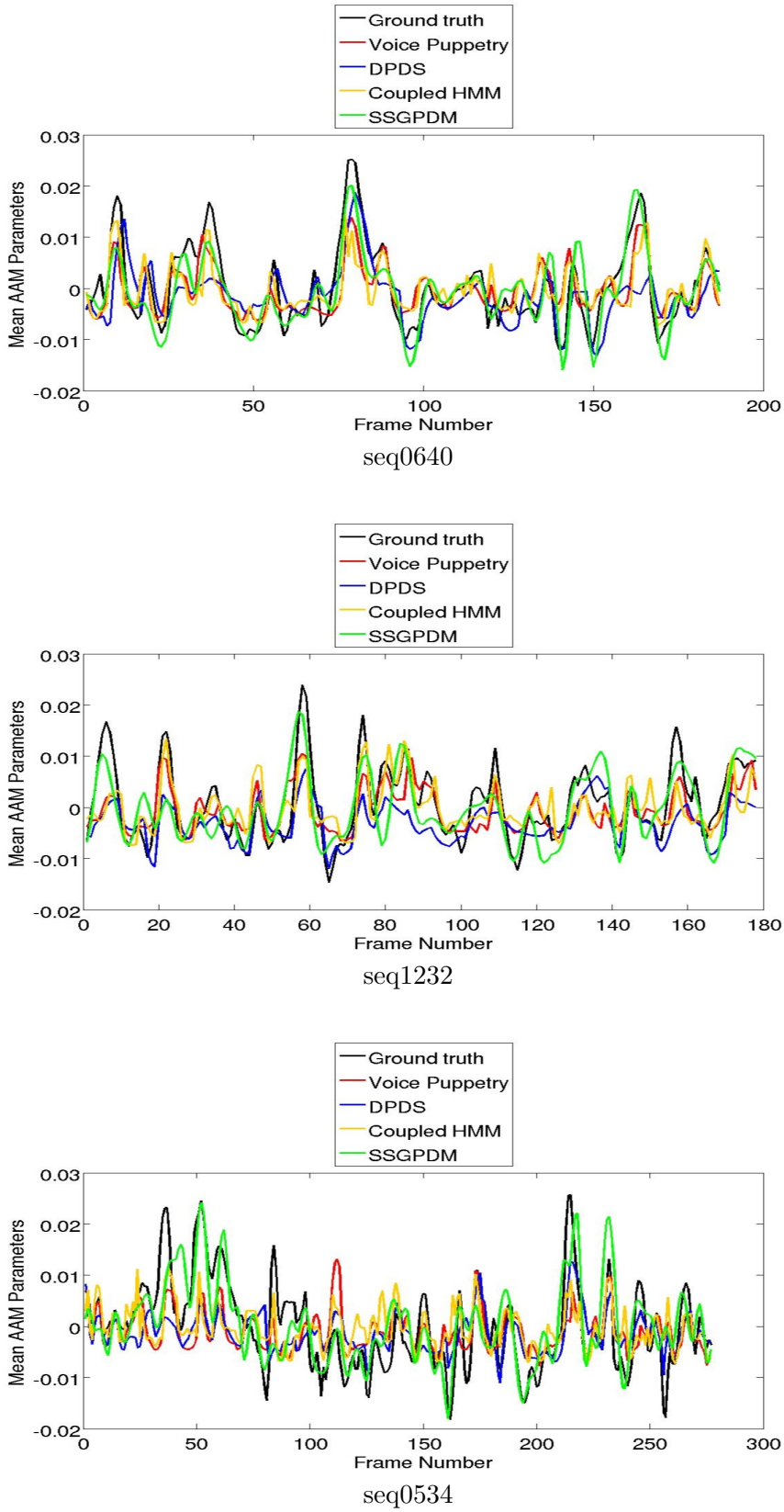


Figure 6.4: Comparing ground truth and synthethic trajectories for the: Voice Puppetry [32], DPDS [102], Coupled HMM [312] and SSGPDM for three test DEMNOW sequences.

Model	Audio Representation	Dynamics	Coarticulation Model
HMM	Discretised in synthesis	Implicit in trajectory synthesis	Forward & Backward
Coupled HMM	Continuous	Implicit in speech features	Backward
Shared LDS	Continuous	Single linear	Backward
SGPDM	Continuous	Single non-linear	Forward & Backward
DPDS	Discrete	Multiple linear	Backward
SSDPDM	Discrete & Continuous	Multiple non-linear	Forward & Backward

Table 6.3: Summary of probabilistic models for visual speech synthesis.

[186], DPDS [102] and coupled HMM [312] result in synthetic animations that are under-articulated, which is also depicted from the AAM trajectories in Figures 6.1, 6.2, 6.3 and 6.4. The SSGPDM method results in more articulated animations that are closer to ground truth. The reasons for this are two-fold. First, parametric models like the HMM, coupled HMM, shared LDS and even DPDS are described by a compact set of parameters and thus tend to underestimate the predictive variance, as evidenced by under-articulated facial movements. They tend to predict an average-case facial behaviour, at the expense of extremities such as emphasis. On the other hand, the SGPDM and SSGPDM are non-parametric models that use the training data to make predictions and can thus better predict the variance in synthesis. This leads to facial animation that looks more natural and is more in line with the audio speech. Moreover, the SSGPDM uses both the phonetic and continuous audio representation of speech, which are complementary and provide more information to the synthesis of visual speech from audio speech. This is different to the Voice Puppetry, shared LDS and DPDS methods, which use only one representation of speech.

Table 6.3 summarises the characteristics of the different probabilistic models applied to visual speech synthesis. The SSGPDM with phoneme VLMM switching states represents audio using both continuous and discrete features. The SSGPDM also uses multiple non-linear dynamical models to represent the different types of dynamics involved in speech. Finally, the sequential optimisation method used with the SSGPDM accounts for both forward and backward coarticulation. These characteristics account for the better performance of the SSGPDM compared to other probabilistic models applied to visual speech synthesis.

Previous researchers have demonstrated that text-driven (including phoneme-driven) synthesisers perform better than continuous speech-driven synthesisers [277] in terms of quantitative results. This is because continuous speech-driven synthesisers are more susceptible to noise than text-driven synthesisers which present less ambiguity. However, we have shown in this work that an approach that combines continuous speech-driven synthesis with the underlying phonetic information gives better results than phoneme-driven synthesisers such as the DPDS of Englebienne et al. [102]. The improved results also arise as a result of modelling phonetic context using non-linear

dynamical systems as opposed to using linear dynamical systems to model phonemes. Our work thus has similarities to the works of Cao et al. [42] and Edge et al. [92], where both the discrete speech units or phonemes and the continuous audio parameters were used to predict visual parameters. However, their works adopted a sample-based approach whereas we adopt a learning-based approach (refer to Chapter 2 Section 2.2.2).

6.3 Subjective Evaluation

Humans are very sensitive to slight imperfections in facial animation and, as a result, objective evaluation is not sufficient to accurately measure the effectiveness of a visual speech synthesiser. It is thus imperative to perform subjective tests with humans to evaluate the naturalness, realism and intelligibility of the facial animations. From the synthesised AAM parameters, we can generate image frames as described in Chapter 3 Section 3.1.3. The frames can then be encoded to video at the appropriate frame rate before being mixed with the test audio file to generate a speech-synchronised facial animation video.

We now present a review of the three categories of subjective tests commonly used for visual speech synthesis in order to motivate the choice of our experiments.

6.3.1 Scoring Tests

Scoring tests consist of asking viewers to rate the quality of a given aspect of the facial animation on a scale from poor to excellent. The International Telecommunication Union Telecommunication Standardization Sector (ITU-T) [157] recommends the mean opinion score (MOS), which is a scale from 1 (lowest) to 5 (highest), to rate the perceived quality of media being transmitted over a channel. Various researchers have adopted the MOS for evaluating the perceptual aspects of facial animation [278, 196, 312, 313, 282, 208, 158]. The aspects commonly evaluated include naturalness, acceptability and realism. According to Theobald [278], naturalness is a general measure of performance that indicates the smoothness and realism of the dynamics of the facial features whilst acceptability is a measure of how suitable a system is for a given application, for e.g. as a natural interface for a computer. Realism encapsulates both static realism (photorealism) and dynamic realism (videorealism), which can be rated using a scoring test. For example, Devin [82] did an experiment where participants were showed both real and synthetic videos of interactive facial behaviour one at a time, and they were asked to pick on a scale of 1 to 7 the level of realism.

In our experiments, we measure: *quality of mouth articulation*, *naturalness* and *agreeableness*. We define *quality of mouth articulation* as a measure of how well the lips sync with the audio and *naturalness* as a measure of the plausibility of the overall

face of the talking head. We define *agreeableness* as a measure of how comfortable the user would be to interact with the avatar if it were used as a natural interface for the computer. This allows us to measure the *acceptability* of the facial animations as a natural interface for the computer as well as to determine if our animations fall within the Uncanny Valley (refer to Chapter 1 Section 2.3). The term *uncanny* comes from the German word *unheimleich* which means the opposite of familiar or agreeable [284]. Thus, rating the level of *agreeableness* of our facial animations also allows us to test for the Uncanny Valley effect.

In addition, we investigate the effect of upper face expressions such as eye blinks to the perception of naturalness and agreeableness. We thus generate four categories of videos: ground truth with static eyes, synthetic with static eyes, ground truth with original eyes and synthetic with original eyes, with the same number of videos in each category. The techniques used to paste static eyes to the videos as well as to paste the eyes from ground truth to synthetic videos are described in Section 6.3.5. The static eyes image is taken from the facial image reconstructed from AAM parameters with all modes of variation set to zero, i.e. the mean AAM parameters.

6.3.2 Realism Tests

In order to assess the realism of synthetic facial animations, scoring tests can be used, as mentioned in the previous section. However, the disadvantage of scoring tests is that they do not give a clear cut picture of which videos are perceived as real and synthetic. An alternative is to use the Turing test [287] proposed by Alan Turing in 1950 as the ultimate test for Artificial Intelligence. The original test was applied to a natural language conversation between a human participant on one end and either a machine or a human on the other end. If the participant is able to distinguish between the machine and human conversations, then the Turing test is deemed to have failed. The same test can be applied to facial animation, by subjecting viewers to both real and synthetic videos and testing whether they can distinguish between these two categories. There are two variants of the Turing test that have been used by researchers in facial animation. The first one involves the "single-view" approach, i.e. showing the videos one at a time and asking viewers to decide whether each video is real or synthetic [120, 313]. The second type involves "paired-view" approach, i.e. showing the videos in pairs and asking the viewers to choose which one is real [120, 101].

The Turing test will almost always fail if a natural video of a person, with eye and head gestures as well as expressive cues such as smiles, is compared against a synthetic video. For that purpose, both real and synthetic videos need to be normalised to remove these artefacts. In our case, videos reconstructed from normalised AAM parameters are appropriate because they do not contain pose variations or expressive cues. In

our Turing test, we show only the part of the face below the eyes in order to remove biases that might occur as a result of eye movements. Thus, the evaluation is pertinent only to speech-related facial movements on the bottom half of the face. Furthermore, we adopt the single-view test in order to limit the ability of participants to learn to distinguish the subtle differences between real and synthetic videos, which might bias their response to the other tests.

6.3.3 Human Lip-reading Tests

Human lip-reading can be a good measure of the intelligibility of a talking face. However, intelligibility tests have not been very effective because of the low rate of lip-reading. Geiger et al. [120] showed that the number of words identified correctly by viewers, when whole sentences were shown, to be 14.5%, whilst for synthetic videos, it was 7.5%. When single words were shown, the recognition rate was 14.7% for real videos and 6.1% for synthetic videos. In a more recent study by Hilder et al. [137], it was demonstrated that when single words were shown to participants, the percentage of correctly identified words was 14.5% on real videos prior to training and 18.8% after undergoing training in lip-reading. Another approach used by Hilder et al. [137] was to ask viewers to write down the word being uttered and then a decomposition of the word into phonemes and visemes was done, before comparing the percentage of correctly identified phonemes and visemes. It was found that the viseme recognition rate was higher than phoneme recognition rate, which was in turn higher than the word recognition rate. It has also been shown that an intelligibility test that involves showing utterances of individual words as opposed to whole sentences leads to higher rates of human lip-reading [83].

Ouni et al. [228] did extensive experiments to investigate the visual contribution to speech perception. In the experiments, 27 consonant-vowel (CV) syllables were used and the test data was factored into: auditory-only, visual-only natural talker, visual-only synthetic talker, bimodal natural talker and bimodal synthetic talker. In addition, the audio was corrupted with noise at different signal-to-noise ratios (SNRs). The participants were asked to recognise the syllable in terms of vowels and consonants. At a SNR of -11 dB, the recognition rates for natural videos were: unimodal auditory at 50%, unimodal visual at 66% and bimodal at 87%. For synthetic videos the recognition rates were: unimodal visual at 52% and bimodal at 74%. The audio-visual lip-reading rate is higher than visual-only lip-reading tests. We believe that integrating audio in a lip-reading test biases the results because poor lip-syncing can be compensated by audio provided that it minimally synchronises with the lips movements, as is commonly encountered in animated cartoons. Moreover, using both audio and visual channels might lead to the McGurk effect [207], discussed in Chapter 2 Section 2.1.2. Cosker

[63] proposed a McGurk test to test the intelligibility of synthetic visual speech. The test was aimed at evaluating whether or not viewers are able to spot the McGurk effect in real and synthetic videos. For example, an audio containing "Bat" and a video containing "Vet" would produce a McGurk response of "Vat". 60 videos (30 real and 30 synthetic) were presented in a random order to viewers and the participants were asked to write down the response they perceived whilst listening to and viewing the video simultaneously. The results showed that the McGurk effect was stronger in real videos than in synthetic videos. Taking into account the McGurk effect in human lip-reading tests with corrupted audio would require a more elaborate experimental setting with potential loopholes, which we have tried to avoid.

Because hearing-impaired people are better trained at lip-reading, it might be worth performing the intelligibility test on this category of people. However, studies have shown that the overall performance of the hearing-impaired in viseme recognition is not significantly different as compared to their healthy counterparts [229]. We thus restrict our tests to non hearing-impaired individuals and, given the low lip-reading rate observed thereof, we propose an alternative to unrestricted lip-reading by giving multiple choices of the sentence being uttered, one of which is correct. We give four choices, thus making the chance rate of picking the right answer to be 25%. The sentences are chosen from the transcriptions in the data corpora. We group sentences of about the same length by giving them the same label, so that for each video, we select a random index from 1 to 4 to show the correct answer whilst for the remaining three choices, we present a sentence with the same label as the correct one, thus ensuring that the sentences are all of about the same length. We reached the choice of 4 sentences after initial prototyping. It was found that 5 sentences was too much and significantly increased the amount of time it took for participants to be able to lip-read the videos, whilst 3 sentences was found to make the lip-reading too easy. One limitation of this approach is that the sentences have different phonetic contents. Ideally, the false sentences should be similar to the true sentence, which can be done by grouping sentences of similar content together. However, due to time constraints, this was not achieved and the only measure of similarity was the length of the sentences.

6.3.4 Statistical Hypothesis Testing

Statistical hypothesis testing can be used to answer yes or no questions about a population from a limited sample by using probability theory [55]. In our case, we want to test the hypothesis of whether or not humans perceive real and synthetic videos as comparable. So, the approach is to first *assume that they are comparable*, which is the *null hypothesis*. Based on data gathered from a limited number of participants, statistical inference is performed to compute the probability (or p-value) of the hypothesis

being true. If the probability is less than a threshold, then we reject the null hypothesis and conclude that the real and synthetic videos are not comparable. The threshold commonly used is $p = 0.05$ [55]. If we reject the null hypothesis with $p < 0.05$, then this means that we have less than 5% chance of being wrong. In that case, it would be said that at the 5% level of significance, there is sufficient evidence to conclude that the real videos are better than the synthetic videos.

In general, hypothesis testing is used to make inferences about a population given a limited sample. Most evaluations are limited to a small sample and we are interested in how these results can generalise to the whole population. There are various tests that can be used depending on the parameters being compared. For example, to test the hypothesis that the mean of a population is equal to a given value, the z-test [55] can be used if we have a big enough sample. If the sample size is small, then the Student's t-test [55] is used. In other cases, we might need to perform a two-sample t-test if we want to compare the means of two samples and determine whether they could have been drawn from populations with equal means. The two-sample t-test assumes that the two samples are independent. If the assumption is that the two samples are dependent, then the paired-sample t-test can be used. If we want to compare two or more populations in terms of their variance, the analysis-of-variance (ANOVA) test can be used [89]. The above tests assume that the underlying data follows a Gaussian distribution. For non-Gaussian data, such as ordinal and nominal data, a family of tests called non-parametric tests have been developed [143]. The non-parametric equivalent of the paired-sample t-test is the Wilcoxon signed-rank [143] test and that of the two-sample t-test is the Wilcoxon rank-sum test [143]. The Krusal-Wallis test [143] is the non-parametric equivalent of ANOVA when the data is non-Gaussian.

In this thesis, we use hypothesis tests on our qualitative results to compare real vs. synthetic animations. In particular, we compare the MOS obtained from scoring tests for real and synthetic videos. In our scoring tests, different videos are shown in the ground truth and synthetic categories and as a result, samples of MOS scores are independent. Because the data is on the ordinal scale, the Wilcoxon rank-sum test [143] is the appropriate test to be used. In our human lip-reading tests, we also show different videos drawn randomly from a pool of videos to each participant and, as a result, the intelligibility scores computed for each participant are independent. Because we show three videos in each category (LIPS real, LIPS synthetic, DEMNOW real, DEMNOW synthetic) to every participant, the intelligibility scores would be either: 0, 0.33, 0.67 or 1, which is on an equal-interval scale but non-Gaussian because there are only four possibilities. An ordinal scale is therefore the best representation for the intelligibility scores, which makes the Wilcoxon rank-sum test [143] appropriate to compare the scores between ground truth and synthetic categories.

6.3.5 Eye Blinks

In our subjective experiments, we aim to investigate how our proposed method predicts mouth articulation from test audio. Thus, the main objective of the evaluation is to investigate the quality of lower face movements in synthetic videos compared to ground truth videos. However, we also want to investigate the effect that upper face gestures such as eye blinks have on the perception of visual speech and in particular whether animations that lack upper face gestures fall into the Uncanny Valley (refer to Chapter 2 Section 2.3). We thus need to have a way of reintroducing eye blinks in the synthetic videos. Previous researchers like Cave et al. [45] have investigated the correlation between the fundamental frequency of speech with eye movements. Lee et al. [182] presented a statistical model for prediction of eye blinks, which takes into account head rotation as well as whether the character is in listening or talking mode. Deng et al. [78] adopted a non-parametric sampling method [97] to generate eye gaze and eye blinks for synthetic facial animation. Weissenfeld et al. [310] used both phonetic and prosodic information of speech to predict eye movements of a character in either listening or talking modes. Dziemianko et al. [90] used the trajectory HMM [325] to learn a mapping from a combination MFCC features and the fundamental frequency to eye movements. Whilst the prediction of visual prosody such as eye and head movements would be an interesting future line of research to pursue, we restrict ourselves to copy the eye movements from ground truth into the synthetic videos. The pasting of texture around the eye region to a target image needs to be done such that it is unnoticeable for viewers. The approach that we adopt makes use of both the shape and texture information of the AAM features and is described below.

Consider that we have both the shape and texture for a target image (Figure 6.5a) that has no eye blinks and a source image (Figure 6.5b) that contains a blink and that we want to paste the eye blinks from the source image to the target image. Since the AAM works with a shape-free texture representation that consists of the face image warped to the mean shape, we can copy a section of the texture from the top to below the eyes from the source to the target shape-free image. However, when the image is reconstructed from AAM parameters, the texture will be warped to the target shape and thus we need to copy the shape of the eyes representing a blink to the target shape. Because the shapes might have variations in scale, translation and rotation, we extract the shape components corresponding to the eye and nose from the source shape and align them to the corresponding shape components of the target shape using generalised Procrustes analysis [126]. Figure 6.5c illustrates this shape alignment procedure. The aligned shape components for the eyes and nose are then copied from the source shape to the target shape. In our experiments, it was found that we need to copy both the eyes and nose components because the proportions of the nose points with respect to

the eye points need to be maintained, otherwise some undesirable artefacts appear in the reconstructed image. Figure 6.5d shows the segmented face image without blinks whilst Figure 6.5e shows the face image with blinks pasted. The illustration has been shown for an actual annotated image for convenience but the same approach can be adopted with the reconstructed shape and texture from AAM parameters. In this case, we would have two sets of AAM parameters, one of which encodes a blink and the other does not. From the AAM parameters, we can reconstruct both the shape and the shape-free image representing the texture and then use the pasting procedure described above.

6.3.6 Experiments

We conducted subjective tests with human participants to evaluate the quality of our synthetic facial animation that gives the best quantitative results, i.e. SSGPDM with phoneme VLMM as switching states and with sequential optimisation as the synthesis method. The tests were carried out over a span of two months with volunteers spending about 30 minutes in front of a computer to participate in the experiments. A total of 50 participants, mostly research students and staff from the School of Computer Science, The University of Manchester, were asked to take the tests independently. The experiments were done after obtaining approval of an ethics committee within the school[1]. The videos used in the experiments were from the AAM reconstructions and involved only the face region with the background segmented. The videos for all the subjective tests were shown at a resolution of 300×300.

Three subjective tests were conducted: a Turing test, a perceptual test and an intelligibility test. For each test, an equal number of real and synthetic videos were shown. The order in which the videos were shown was randomised.

We conducted statistical hypothesis testing for the perceptual and intelligibility tests in order to determine whether there is a significant difference between the ground truth and synthetic videos. We used the Wilcoxon rank-sum test [143] for the perceptual mean opinion scores (MOS) and for the intelligibility scores obtained from the 50 participants. In our case, the null hypothesis is that ground truth and synthetic video scores are comparable and we use a 5% level of significance.

The main screen for the tests is shown in Figure 6.6 with links to each test appearing on the right. The three tests were done in a sequential manner with only the link to one test enabled at a time and the links to all tests disabled after the three tests were done. This was done in order to prevent participants from running the same test more than once. The order in which the links to the tests appeared was randomised so as to avoid any bias. The results of each test were stored in text files on the experimental

[1] https://ethics.cs.manchester.ac.uk/

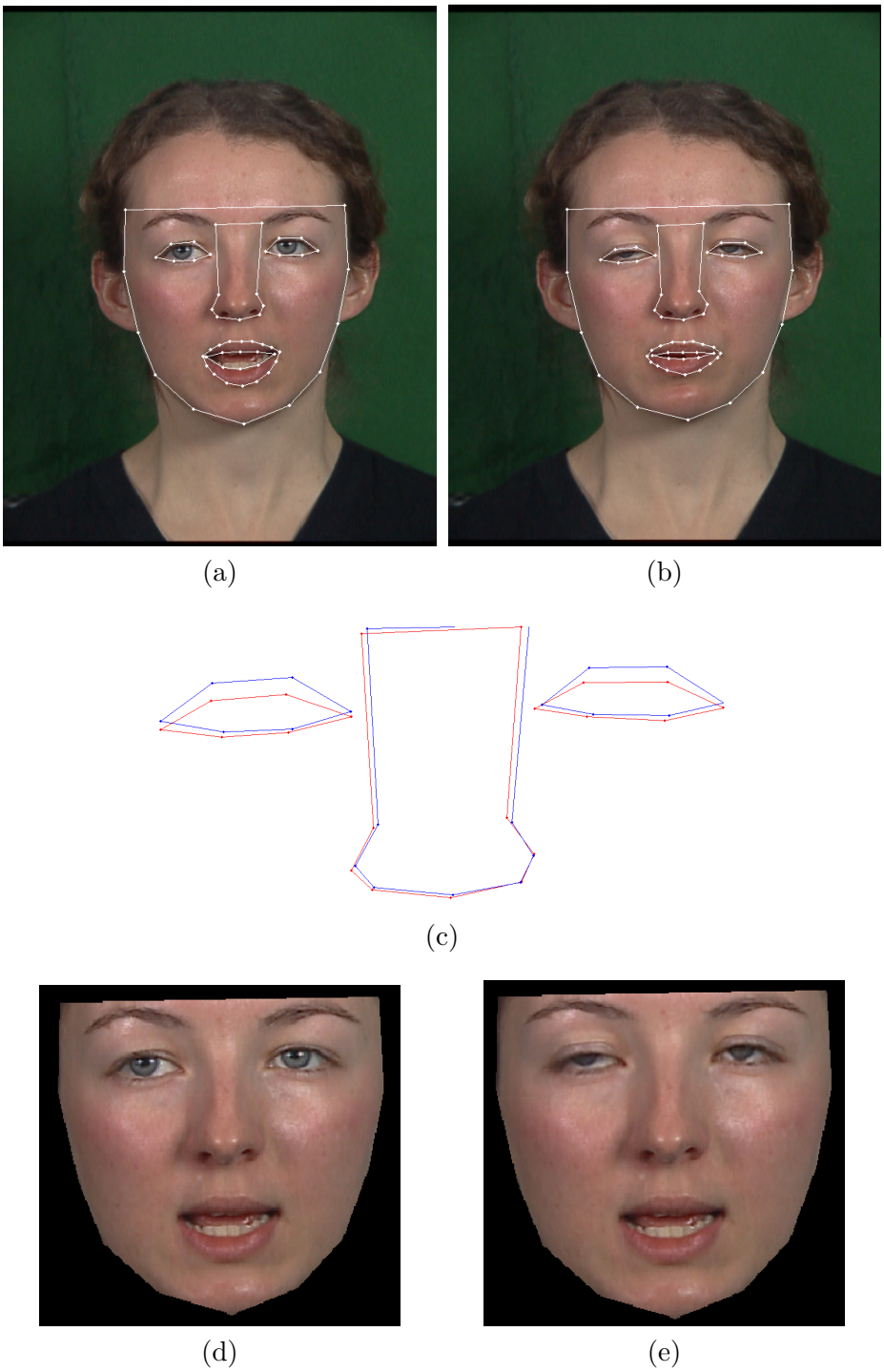


Figure 6.5: Process of pasting eye blinks: (a) Original image with no eye blinks. (b) Image from which eye blinks are to be pasted from. (c) Shape alignment using generalised Procrustes analysis. (d) Segmented image without eye blinks. (e) Segmented image with eye blinks pasted.

computer.

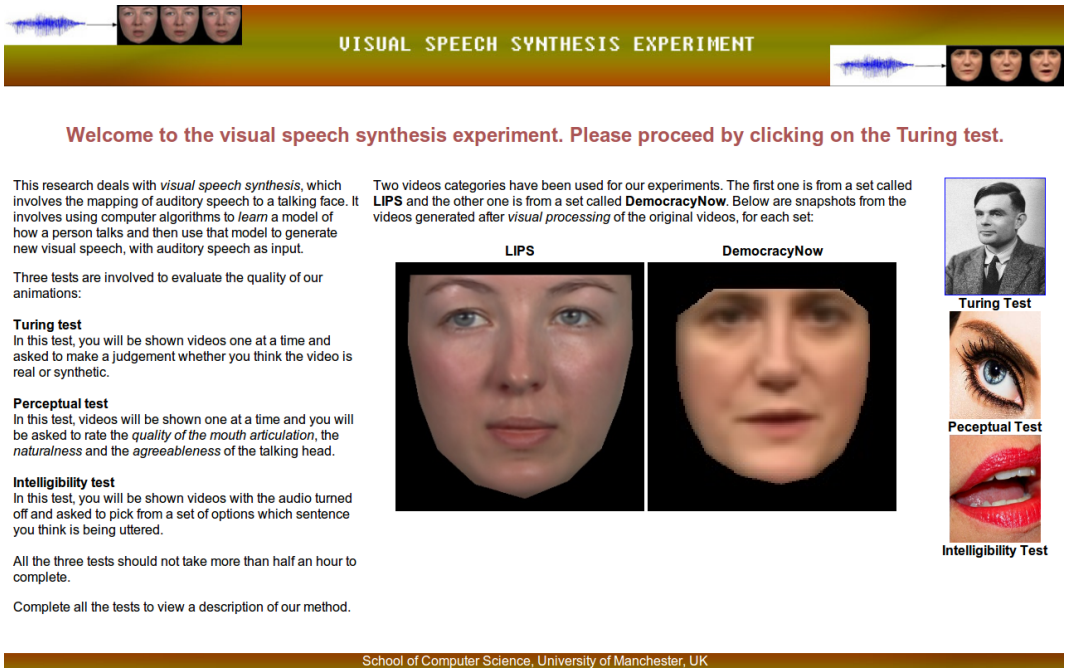


Figure 6.6: Visual speech synthesis experiment - Main page.

Demographics of the Subjective Tests Participants

In order to obtain details of the participants, the pop-up window in Figure 6.7 was shown when the link to the first test was clicked. This allowed the collection of details of the participants in view of obtaining the demographics for the tests.

The distribution of the participants in terms of age and sex is shown in Table 6.4 and the distribution in terms of hours of computer usage and native vs. non-native speaker is shown in Table 6.5. None of the participants were colour blind and no one chose the "prefer not to say" option in the popup questions. From Table 6.5, it can be seen that 86% of the participants were frequent computer users and 70% of the participants were non-native speakers. The frequent computer usage is an important criterion because it allows us to determine how viewers judge the quality of the facial animations as a potential natural interface for the computer, given that frequent computer users are accustomed to more conventional Human-Computer Interfaces (HCI). We expect native speakers to be better lip-readers than non-native speakers. The fact that the majority of the participants were non-native speakers means that we need to compare the intelligibility scores between ground truth and synthetic videos for both native and non-native speakers separately in order to check that they are consistent with the overall results.

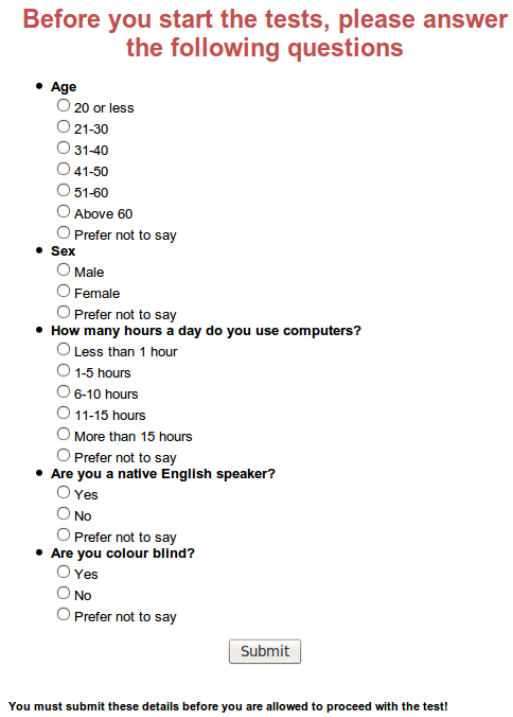


Figure 6.7: Visual speech synthesis experiment - Popup page for user profile.

Age group	Sex	Percentage	Percentage
20 or less	Male Female	2% 2%	4%
21-30	Male Female	40% 16%	56%
31-40	Male Female	25% 9%	34%
41-50	Male Female	2% 0%	2%
51-60	Male Female	2% 0%	2%
Above 60	Male Female	2% 0%	2%

Table 6.4: Distribution of the participants of the subjective tests in terms of age and sex.

Hours of computer usage	Native/non-native speaker	Percentage	Percentage
Less than 1 hour	Native Non-native	0% 0%	0%
1-5 hours	Native Non-native	6% 8%	14%
6-10 hours	Native Non-native	12% 34%	46%
11-15 hours	Native Non-native	10% 24%	34%
More than 15 hours	Native Non-native	2% 4%	6%

Table 6.5: Distribution of the participants of the subjective tests in terms of hours of computer usage and native vs. non-native speaker.

Turing Test

The Turing test involved showing videos one at a time and asking participants to choose whether they are real or synthetic. In order to limit the evaluation to lower face movements, the Turing test involved showing only the lower half of the face. The test involved showing 16 videos: 4 in each of the following categories: LIPS real, LIPS synthetic, DEMNOW real, DEMNOW synthetic. The videos were chosen from a pool of 60 sequences of both real and synthetic videos. The screenshot for the Turing test is shown in Figure 6.8.

The Turing test results are illustrated in Figure 6.9 and show that 38% and 45% of synthetic videos were confused as ground truth for LIPS and DEMNOW respectively against 30% and 27% of ground truth videos being confused as synthetic for the same categories. The less favourable results for LIPS can be explained by the fact that the LIPS corpus is at a higher resolution with the inner details of the mouth visible, which makes viewers much more sensitive to slight imperfections. The results obtained in the Turing test are comparable or even improve over state-of-the-art results reported in the literature. For example, in the work of Xie and Liu [313], a single-view Turing test was carried out where 38% of synthetic videos were mistaken as real, which is similar to the results we obtain for the LIPS dataset. However, for the DEMNOW dataset, the percentage of synthetic videos confused as real is close to chance level, which is very encouraging. Visual speech synthesis that can pass the Turing test still remains a major challenge and additional factors such as expressiveness and prosody have to be taken into account to make a synthetic avatar more realistic.

Perceptual Test

The perceptual test involved showing videos one at a time and asking participants to give a score from 1 (poor) to 5 (excellent), on the *quality of mouth articulation*, the *naturalness* and the *agreeableness*. The video sequences were selected randomly but subsequently fixed for all participants in order to be able to compute mean opinion scores (MOS) on the same sequences. The perceptual test involved showing 24 videos with 6 in each of the following categories: LIPS real, LIPS synthetic, DEMNOW real, DEMNOW synthetic. In addition, for each of the 6 videos in each category, we show 3 videos with eyes pasted from ground truth and 3 videos with static eyes image pasted according to the technique described in Section 6.3.5. The aim of this is to investigate the effect of upper face movements such as eye blinks on the naturalness and agreeableness of the talking head. Figure 6.10 shows the screenshot for the Perceptual test.

A mean opinion score (MOS) is then computed for each video category per participant. Figures 6.11, 6.12 and 6.13 show the results of the perceptual tests for: *quality*

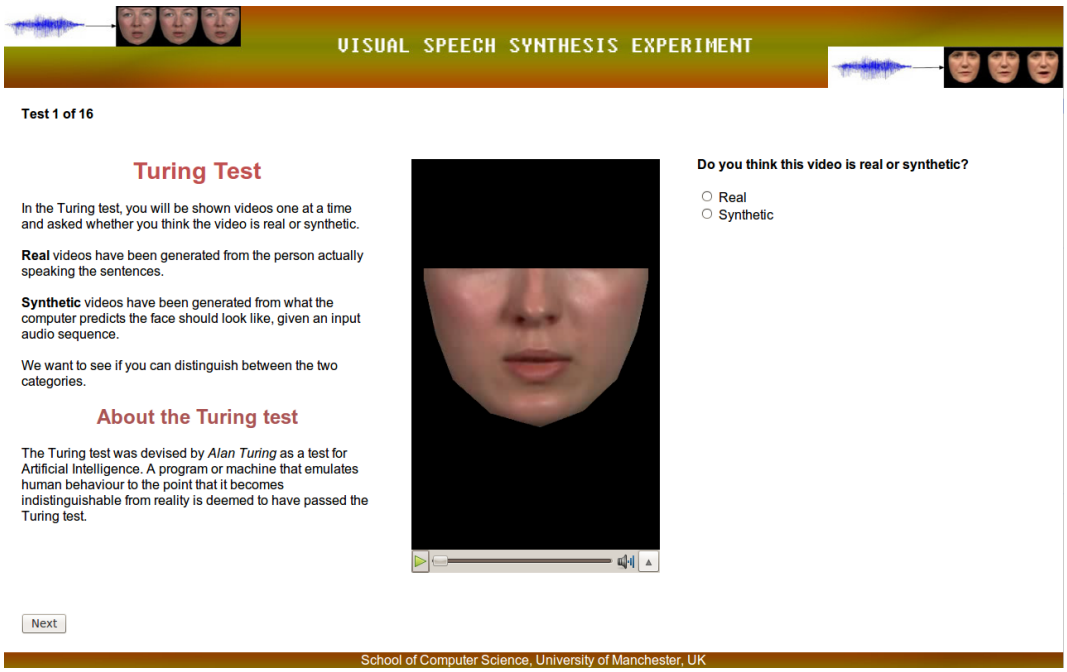


Figure 6.8: Visual speech synthesis experiment - Turing test.

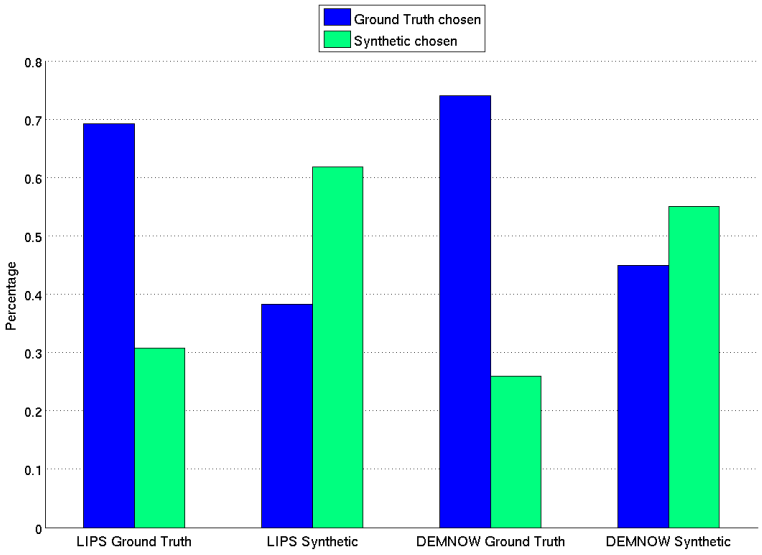


Figure 6.9: Turing test results.

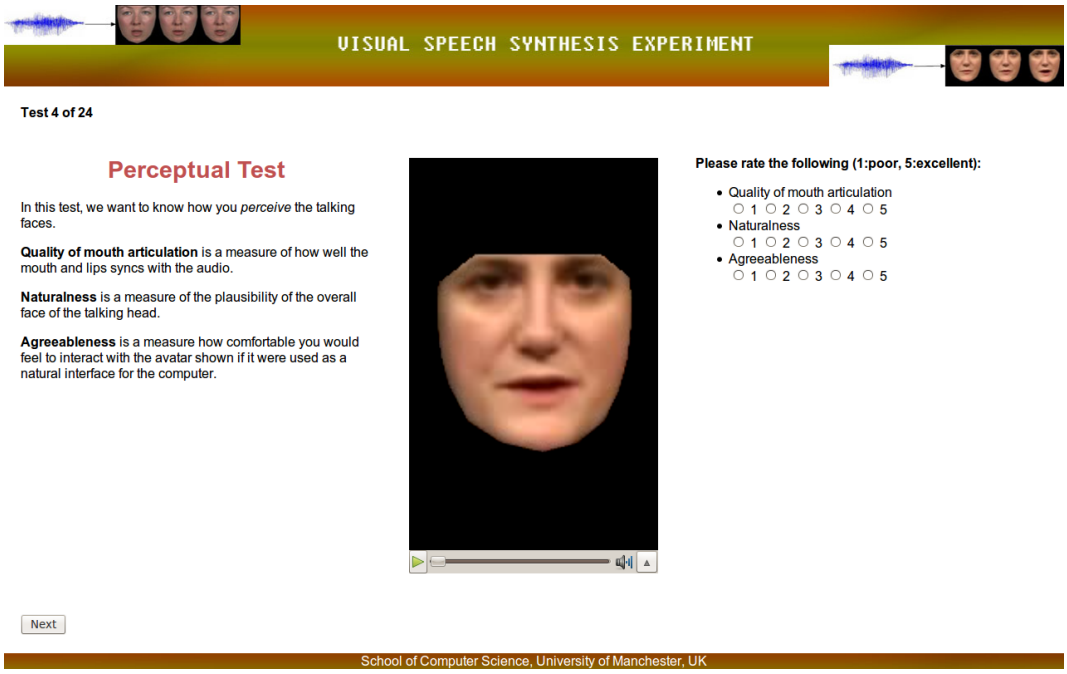


Figure 6.10: Visual speech synthesis experiment - Perceptual test.

	Mouth Articulation		Naturalness		Agreeableness	
Category	Ground Truth	Synthetic	Ground Truth	Synthetic	Ground Truth	Synthetic
LIPS - eyes from ground truth	4.13 ± 0.63	3.70 ± 0.82	3.63 ± 0.84	3.10 ± 0.97	3.55 ± 0.90	3.10 ± 1.04
LIPS - static eyes	3.99 ± 0.77	3.78 ± 0.77	3.10 ± 0.96	3.19 ± 0.92	3.11 ± 0.90	3.12 ± 0.92
DEMNOW - eyes from ground truth	4.08 ± 0.76	3.73 ± 0.90	3.87 ± 0.85	3.56 ± 0.87	3.76 ± 0.91	3.50 ± 0.92
DEMNOW - static eyes	4.15 ± 0.64	3.92 ± 0.80	3.60 ± 0.88	3.50 ± 0.83	3.54 ± 0.91	3.44 ± 0.91

Table 6.6: MOS scores for perceptual test.

of mouth articulation, *naturalness* and *agreeableness* respectively, with standard deviations computed over the MOS of the 50 participants. Table 6.6 also gives the exact MOS scores and standard deviations computed for each category.

For the animations with static eyes, the difference between the MOS of *quality of mouth articulation* for real and synthetic videos is non-significant for both LIPS ($p > 0.13$) and DEMNOW ($p > 0.19$). For *naturalness*, the MOS difference between real and synthetic videos is found to be non-statistically significant for both LIPS ($p > 0.7$) and DEMNOW ($p > 0.5$). Similarly for *agreeableness*, the MOS difference between real and synthetic animations is non-statistically significant both for LIPS ($p > 0.95$) and DEMNOW ($p > 0.59$). For synthetic animations where eyes have been pasted from ground truth, the difference between the MOS of *quality of mouth articulation* for real and synthetic videos is significant for LIPS ($p < 0.01$) but not DEMNOW ($p > 0.05$). For *naturalness*, the MOS difference between real and synthetic videos is also found to be significant for LIPS ($p < 0.01$) but not DEMNOW ($p > 0.1$). The same observation applies to *agreeableness*, where the MOS difference between real

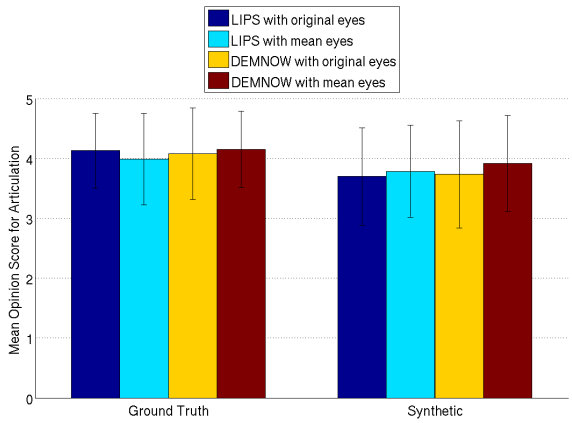


Figure 6.11: Perceptual results for *quality of mouth articulation*.

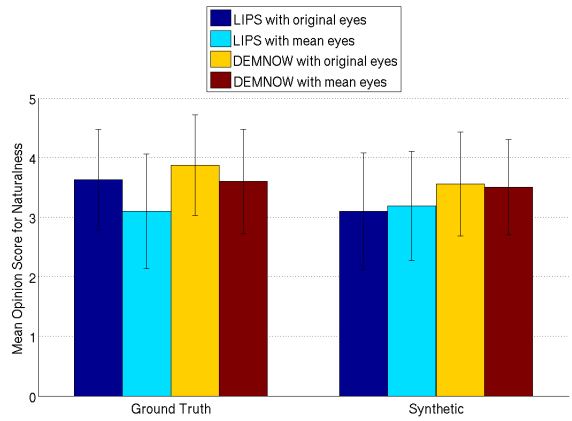


Figure 6.12: Perceptual results for *naturalness*.

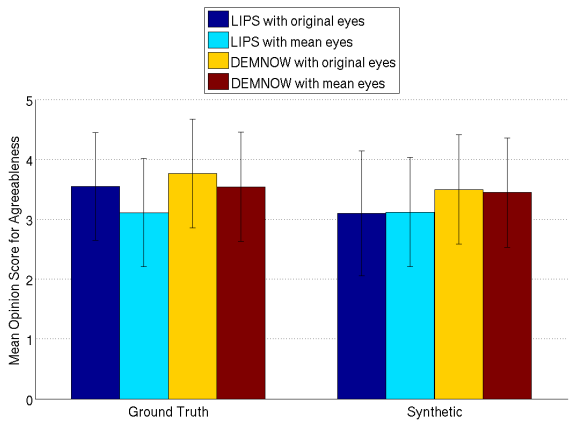


Figure 6.13: Perceptual results for *agreeableness*.

and synthetic animations is found to be statistically significant for LIPS ($p < 0.01$) but not DEMNOW ($p > 0.17$). These results can be explained by the fact that pasting the eyes from ground truth for the LIPS dataset results in slight imperfections in the overall animation, which also surprisingly affects the perception of *quality of mouth articulation* adversely. The effect of eyes pasting is less noticeable in the DEMNOW dataset, mostly due to the lower fidelity of the videos. With static eyes, the difference of the different qualitative factors are mostly insignificant between ground truth and synthetic videos. In this case, the ground truth and synthetic videos are on a level playing field with the only variable being the lip and lower face movements. It can thus be concluded that pasting the eyes from ground truth to synthetic videos degrades the quality of synthetic videos. It is interesting to note that this unnatural effect when pasting eyes in the LIPS dataset also seems to make viewers rate down the *quality of mouth articulation* with the difference between the real and synthetic MOS scores being statistically significant. This leads us to conclude that the dynamics of the whole face are important in the perception of speech and if the upper face do not have the right movements, then the perception of the *quality of mouth articulation* is also affected.

The effect of eye movements on the perception of facial animation can also be examined by comparing MOS scores of *naturalness* and *agreeableness* within the ground truth and synthetic video categories. In ground truth videos, there is a sharp preference for animations with original eyes. However, in synthetic videos, the same MOS scores do not show this sharp preference. For LIPS, viewers actually slightly prefer animations with static eyes, whilst for DEMNOW, there is a slight preference for videos with eyes pasted from ground truth. This reinforces the conclusion that the unnatural effects that arise from pasting eyes from ground truth are more prominent in the LIPS corpus than in the DEMNOW corpus, because of the higher fidelity of the LIPS corpus.

The MOS scores of *naturalness* are very similar to those of *agreebleness* across the different categories. Our aim for including *agreeableness* was to test for the Uncanny Valley effect [215]. For LIPS videos, the sharp drop in *agreeableness* scores in ground truth videos when static eyes are used and also the low scores of *agreeableness* in synthetic videos, both when static eyes are used and when the eyes are pasted from ground truth, can be an indication of these animations falling into the Uncanny Valley. Indeed, these videos exhibit a certain level of "eeriness" that can lead to strong repulsion in participant viewers. The DEMNOW videos do not exhibit this sharp drop in *agreeableness* when eyes are pasted from the ground truth videos, possibly due to the lower fidelity of the DEMNOW videos with effects of eye pasting being less noticeable.

Intelligibility Test

The intelligibility test involved showing videos one at a time, with the sound turned off, and asking participants to pick from a set of four sentence choices, one of which is correct. The intelligibility test involved showing 12 videos: 3 in each category (LIPS real, LIPS synthetic, DEMNOW real, DEMNOW synthetic), chosen randomly from the pool of videos. The video sequences were chosen from a pool of 60 videos of paired real and synthetic videos, just like for the Turing test. The screenshot for the intelligibility test is shown in Figure 6.14.

We then compute the intelligibility score of each video category, for each of the 50 participants, by taking the number of correctly chosen sentences divided by the total number of sequences for the corresponding category. The mean intelligibility score for each category is then computed over all participant scores. The scores are as follows: LIPS real: 74%, LIPS synthetic: 62%, DEMNOW real: 68%, DEMNOW synthetic: 57%. The results reveal that the difference between the intelligibility scores for real and synthetic videos is not statistically significant: $p > 0.09$ for LIPS and $p > 0.12$ for DEMNOW. The intelligibility scores for the different categories are illustrated in Figure 6.15. It can thus be inferred that the intelligibility of real and synthetic videos are comparable.

We also compared the intelligibility scores of native vs. non-native speakers with the results shown in Table 6.7. We used the Wilcoxon rank-sum test [143] to compare the intelligibility scores between ground truth and synthetic videos within native and non-native speakers. For native speakers, the difference is non-significant for LIPS ($p > 0.5$) and DEMNOW ($p > 0.4$). For non-native speakers, the difference is again non-significant for both LIPS ($p > 0.05$) and DEMNOW ($p > 0.1$). However, the difference of the intelligibility scores between real and synthetic videos is more pronounced amongst non-native speakers, which is what we expect.

It is to be noted that the proposed intelligibility test with multiple choices is more reliable than unrestricted lip-reading as used by Geiger et al. [120] and Hilder et al. [137], which showed the lip-reading rate to be less than 20% for real videos and less than 10% for synthetic videos (refer to Section 6.3.3). To the best of our knowledge, we are the first to conduct intelligibility tests with multiple choices and, based on the results obtained, we find it to provide a robust measure of the intelligibility of synthetic visual speech. However, it should be pointed that that the intelligibility scores that we obtained cannot be interpreted in an absolute way. Rather, they provide a relative measure of the intelligibility of ground truth and synthetic visual speech.

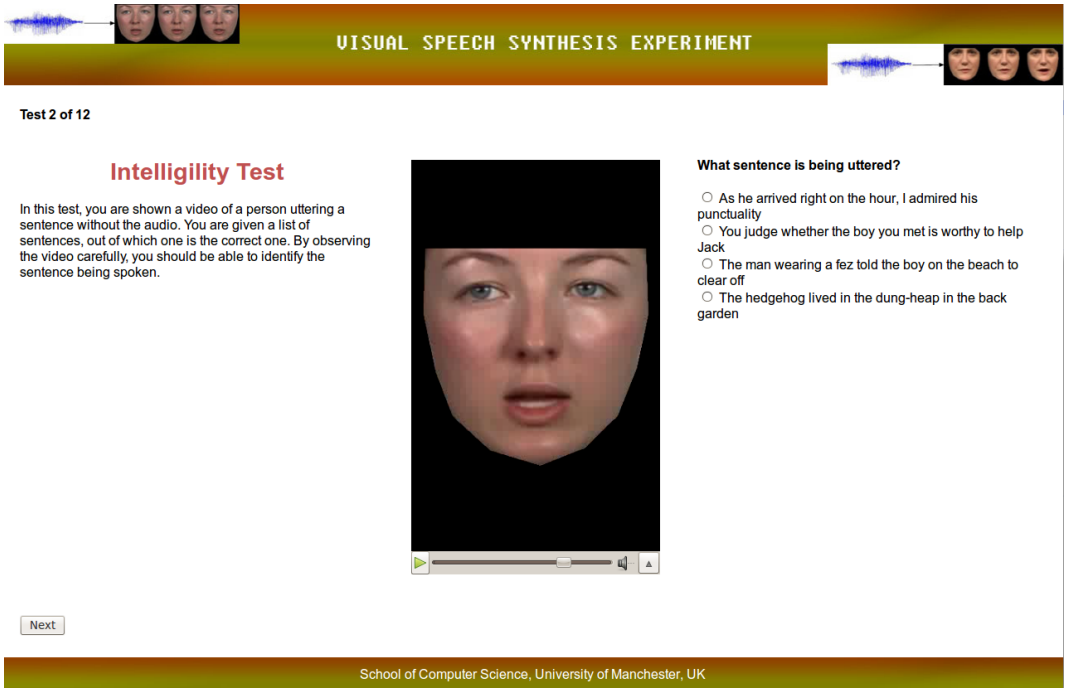


Figure 6.14: Visual speech synthesis experiment - Intelligibility test.

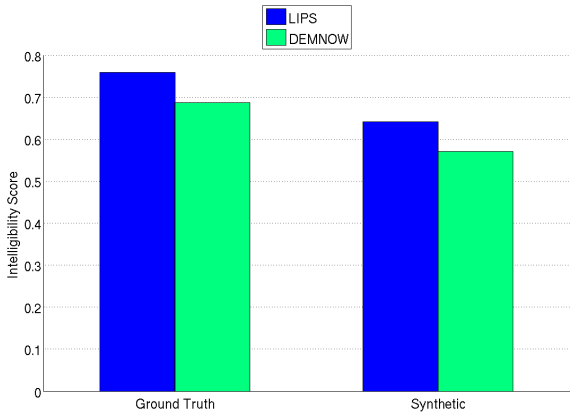


Figure 6.15: Intelligibility scores.

	Native		Non-native	
Video Category	Ground Truth	Synthetic	Ground Truth	Synthetic
LIPS	87.8%	82.8%	68.5%	54.9%
DEMNOW	75.6%	65.6%	64.1%	54.6%

Table 6.7: Comparison of intelligibility scores between native and non-native speakers.

6.3.7 Discussion

In the subjective tests, the aim was to investigate the way humans perceive the synthetic talking head generated using our method that gives the best quantitative results, i.e. SSGPDM with phoneme VLMM switching states and with sequential optimisation for synthesis. The tests carried out were: perceptual, Turing and intelligibility tests. The results of the Turing test are comparable to the state-of-the-art. The perceptual and intelligibility tests revealed that the synthetic talking heads are mostly comparable to ground truth. Our perceptual tests also demonstrate that our talking head can be used as a natural interface for the computer or mobile devices, provided the avatar exhibits proper eye movements. Facial animations that exhibit poor eye movements, or no eye movements at all, elicit strong repulsion by human participants, possibly indicative of the Uncanny Valley effect [215].

One limitation of our experimental setup was that we did not log the number of times the participants played the videos to arrive at an answer. We later realised that this could give an additional categorisation of the results because some people played the videos only once whilst others played the videos multiple times to make their decisions. Because we did not impose any limits to the number of times that participants replayed the videos, we expect participants who replayed more to be more sensitive to slight imperfections for the Turing and perceptual test but also to be better able to perform lip-reading for the intelligibility test.

One of the aims of this work was to achieve photorealism and videorealism in speech-driven facial animation. The photorealism criterion is met as evidenced by image frames from the resulting animations shown in Appendix D, which look like photographs. The Turing test also revealed that a high percentage of viewers confuse the synthetic videos for real videos, which is strong evidence for videorealism. However, full videorealism can only be achieved if the synthetic animations are integrated into a real environment with background and with the avatar exhibiting non-verbal and expressive cues. This remains a very challenging problem in visual speech synthesis.

6.4 Chapter Summary

This chapter presented an evaluation of our visual speech synthesis methods. Objective results were presented for the SGPDM as well as both objective and subjective results for the SSGPDM. Our objective results reveal that our joint models of audio and visual perform better than comparable methods of visual speech synthesis. Moreover, the subjective results show that the realism, perceptual characteristics and intelligibility of our synthetic videos are mostly comparable to ground truth videos processed using the AAM.

Chapter 7

Conclusion and Future Directions

> The ability to perceive or think differently is more important than the knowledge gained.
>
> David Bohm

This thesis presents two joint probabilistic models of audio and video as generative models of speech, which can then be used to predict visual from audio parameters in view of synthesising speech-driven facial animation. The first one is the shared Gaussian process dynamical model (SGPDM), which learns a shared latent space between audio and visual parameters. The second one augments the SGPDM with switching states, to yield a switching state-space model called the switching shared Gaussian process dynamical model (SSGPDM). We used two audio-visual corpora to train our models, namely the LIPS corpus [281], featuring a female British speaker reading sentences from the Messiah corpus [278], and the DEMNOW corpus [102], featuring a female American speaker giving news presentations. Different synthesis techniques were presented and, finally, we presented both objective and subjective evaluations of the proposed methods.

Our hypothesis set in Chapter 1 was that the use of both a discrete and continuous audio representation to predict visual speech and explicitly modelling the non-linearities in audio-visual mapping using non-parametric Gaussian processes [251] would help address the under-articulation problem in parametric learning-based methods [32, 185, 312, 102] and match ground truth animations more closely. This has been confirmed from the objective evaluation done in Chapter 6, where we have compared our method with previous methods that use either a discrete or continuous audio representation and either adopt a linear or piecewise linear appproach to audio-visual mapping. We have also shown that a switching state-space model that explicitly models phonetic context achieves better quantitative results than ones which do not, for example the DPDS of Englebienne et al. [102].

We aimed to achieve both photorealism and videorealism, which we believe has been realised based on the subjective evaluation presented in the previous chapter. Moreover, our subjective experiments also show that the intelligibility achieved in synthetic videos generated with our method is comparable to ground truth. We have shown that the perceptual aspects of our synthesised facial animation compare favourably with ground truth. However, it has been found that the lack of upper facial movements, such as eye blinks, leads to strong repulsion in humans, which might be indicative of the Uncanny Valley [215]. Moreover, the realism achieved is only relative to that of ground truth videos parameterised with the active appearance model (AAM) [60]. AAM parameterisation leads to some blurring of the reconstructed facial images, which compromises the quality of the synthetic visual speech.

It should be noted that the focus of this thesis has been the proposal of a novel behaviour model for synthesising visual speech. We have limited ourselves to using a 2D representation of the face because of the availability of public 2D audio-visual datasets. Nonetheless, the proposed method should also be extensible to 3D models of the face represented in a compact parametric form, for e.g. the 3D Morphable Model [25].

The following outlines the summary of this thesis.

7.1 Thesis Summary

- **Visual processing** - The active appearance model (AAM) [60] was used to extract visual parameters from images. We presented a method to train the AAM by automatically selecting image frames belonging to each of the phonemes, in order to cater for all possible speech-related facial expressions. In addition, Chapter 3 proposed visual normalisation techniques to obtain frontal pose, which is an important requirement for our visual speech synthesis method.
- **Audio processing** - Different speech parameterisations were used to process audio, namely LPC, LSF, MFCC and RASTA-PLP [151], as well as different audio-visual synchronisation methods, namely: 1) processing the speech at the same frequency as the visual frame rate, and downsampling speech parameters from 100Hz to the visual frame rate using 2) polyphase quadrature filtering [256] and 3) median filtering [6]. These were discussed in Chapter 3.
- **Non-linear state space model of audio and video** - The SGPDM is a non-parametric and non-linear state-space model that can be used to couple the audio and visual streams of speech using a shared latent space. The model involves having Gaussian process (GP) [251] mappings from the latent space to the audio and visual spaces, as well as an autoregressive GP mapping on the latent space, to map the previous latent point to the next. We presented model selection

experiments to determine the optimal parameters of the SGPDM to be used for audio-visual mapping. In addition, experiments were presented using the SGPDM to determine the audio speech features and audio-visual synchronisation method that best predict the visual speech features for both the LIPS and DEMNOW corpora. The SGPDM was presented in Chapter 4.

- **Augment the state-space model with switching states** - The SGPDM was found to have some major limitations as a generative model of speech. First, it uses single observation and dynamical models for the whole data, which is not a valid assumption given that speech involves multiple dynamics. Second, being a non-parametric model, the SGPDM can only handle a limited amount of data because the size of the model grows with the data, thus making training intractable for data exceeding a few thousand frames. In order to address this limitation, we augmented the SGPDM with switching states that represent the multiple dynamics, yielding the switching SGPDM (SSGPDM). The switching states were found automatically and explicitly model phonetic context. This was achieved by training a variable length Markov model (VLMM) [130, 253] on phonetic data to find commonly occuring fragments of speech. The audio and visual data within these fragments were then extracted and modelled using a SGPDM as multiple sequences. Chapter 5 presented the SSGPDM with VLMM switching states.
- **Use full audio information in training and synthesis** - Techniques that use only the phonetic representation discard prosodic aspects of speech whilst techniques that use only continuous speech features do not take into account the structure of language. A main contribution of this work is the use of both phonetic information and continuous speech features for visual speech synthesis. We achieve this by learning a variable-order Markov model on the stream of phonemes to obtain a segmentation of commonly occuring phonetic contexts and learning generative models of both continuous audio and visual speech for each segment. For synthesis, the test phonetic information is first used to infer the phonetic contexts, which are then used to predict visual from audio parameters using the corresponding joint models of audio and video. The SSGPDM presented in 5 uses both discrete and continuous speech data to predict visual data whilst exploiting the structure of language through the VLMM.
- **Synthesis techniques for both batch and online applications** - Two techniques were presented for synthesis. The first one is called *sequential optimisation* and is suited for batch processing, assuming that the whole audio data is available at the beginning. Because it takes into account future audio information, both forward and backward coarticulation are modelled. The second technique is

called *sequential prediction* and assumes that the audio data is arriving in an online fashion and is able to predict the next frame from the previous. This method can be used for real-time applications, although a real-time inference algorithm for VLMM states is needed. We discuss how this can be achieved in the *Future Directions* Section 7.3. The synthesis algorithms for the SSGPDM were presented in Chapter 5.

- **Post-processing** - We present a post-processing method to reintroduce eyeblinks to synthetic videos by using ground truth AAM features to reconstruct the shape and texture and then aligning the shape of the eyes of the ground truth with that of the synthetic. Then, the aligned eyes shape are copied to the synthetic shape. Moreover, the normalised texture corresponding to the upper part of the face with the eyes is copied to the synthetic texture. The post-processing technique was described in Chapter 6.
- **Evaluation** - Quantitative evaluation results of our proposed method were compared against other related methods such as the Voice Puppetry of Brand [32], the shared LDS of Lehn-Schiøler et al. [186], the coupled HMM of Xie and Liu [312] and the DPDS of Englebienne et al. [102]. The SSGPDM with phoneme VLMM as switching states and with sequential optimisation for synthesis was found to give the best quantitative results. We also conducted three subjective tests, namely: a Turing test, a perceptual scoring test and an intelligibility test. Our contributions were: test real and synthetic videos with and without eye blinks and test for the Uncanny Valley effect [215] by having participants score the level of agreeableness of the animations. Moreover, we also presented four sentence choices in the intelligibility test, one of which was the correct one, in order to address the limitations of unrestricted intelligibility tests as described by Geiger et al. [120] and Theobald et al. [282]. Chapter 6 dealt with objective and subjective evaluation of our work.

7.2 Limitations of the Proposed Method

- **Need for phonetic labels in synthesis** - Our current method assumes that we have phonetic labels for the test audio sequence, which can be obtained from speech recognition tools such as HTK [317]. We thus treated recognition and synthesis as two different problems. In principle, we could devise a Viterbi algorithm [293] to infer phonetic states or phonetic contexts from continuous audio by associating each VLMM state to a distribution over audio parameters. However, we expect the recognition results to be lower than using HTK, which trains richer models based on context-dependent HMMs. As observed by other researchers such as Ostendorf and Bulyko [225], recognition and synthesis are two separate

problems with the former being discriminative and the second being generative. Therefore, the models used for synthesis are not necessarily ideally suited for recognition.

- **Non real-time synthesis** - The SSGPDM synthesis technique that gives the best quantitative results, namely sequential optimisation, has no real-time performance. The *sequential prediction* technique can be used for real-time applications but a real-time inference algorithm for the VLMM states is needed, as mentioned previously.
- **Dependence on speech parameterisation** - Our experiments have shown that different speech parameterisation and audio-visual synchronisation techniques give optimal results for the two data corpora used in this work, namely the LIPS and DEMNOW corpora. These different results were found to arise as a result of the setting in which the original data was recorded. The LIPS corpus was recorded in a controlled setting with much slower speaking rate than the DEMNOW corpus, which involves an anchor in a fast-paced newscast environment. If our method is to be used practically, a decision needs to be made by a human operator on the audio parameters to use based on knowledge of the nature of the data. A direction of future work could be to infer the speech parameters automatically from the data, using classification methods in Machine Learning.
- **Need for visual normalisation** - As opposed to techniques that generate a smooth trajectory of visual parameters from discrete state indices [32, 105, 102], our method requires visual normalisation in order that the correlations between pose and continuous audio parameters are not captured, which would otherwise lead to spurious pose variations in the synthetic animation.
- **Lower fidelity animations** - By adopting a 2D appearance model for facial representation and a learning-based method for audio-visual mapping, we obtain lower fidelity animations as compared to sample-based methods that reorder original images to achieve facial animation. In particular, the use of the active appearance model (AAM) [60] results in some blurring particularly around the mouth regions. Sample-based approaches have become the state-of-the-art in recent years, winning the LIPS visual speech synthesis challenges in 2008 [191] and 2009 [303]. However, the disadvantage of sample-based approaches is that they cannot be adapted to novel facial identities in transferable speech animation. Transferability of identity in AAMs has been demonstrated by Theobald et al. [280] and could be a possible extension of our work.

7.3 Future Directions

The following gives a list of ways in which the current work could be furthered in the future as well as different paths that could have been explored if more time were available.

- **Real-time synthesis** - The first step in achieving real-time synthesis would be to infer VLMM states in real-time, which could make use of particle filtering methods as demonstrated by Hou et al. [146], Stefanov et al. [268] and Caillette et al. [40]. However, real-time synthesis does not take future context into account, which means anticipatory coarticulation was not modelled. Possible ways to address this would be to have a short lag in order to be able to capture some future context or, alternatively, using the backward context to try to anticipate what is coming next. We believe that the VLMM or alternative higher order Markov models can be applied to this problem in order to anticipate future visual speech behaviour from past context.
- **Learn language model from continuous audio** - At present we learn a language model using the VLMM on phonetic data. In theory, it should be possible to infer the language model directly from continuous audio data. The variable length hidden Markov model (VLHMM) [306] could be used for simultaneously clustering the audio parameters and learning the variable length Markov model.
- **Sequential filtering** - The sequential prediction algorithm for synthesis does not use the full audio information for the whole test sequence. Instead, only the first frame belonging to a given VLMM state is used to optimise the latent point and the subsequent latent points are predicted from the previous using the dynamical GP. A more powerful method would be to use filtering to infer the latent states from both the audio and the dynamics GP mappings. A *sequential filtering* algorithm could be devised, making use of GP Bayes filters proposed by Ko and Fox [170] and Deisenroth et al. [72].
- **Combine sample-based and learning-based approaches** - In this thesis, we adopt a learning-based approach to visual speech synthesis, which results in some blurring in the final animation due to parameterisation of the face with appearance models. One way to improve the fidelity of the animation would be to combine the learning-based approach with a sample-based approach, where the final animation is generated from image frames belonging to the original corpus, using a similarity measure between images reconstructed from the AAM parameters and original images in the training set. This is similar to the approach adopted by Wang et al. [302].
- **Visual prosody** - Synthesis of visual prosody such as eye blinks and head movements is the next step in creating videorealistic speech animation that is as close

to natural as possible. Synthesis of eye movements has been demonstrated by Deng et al. [80] and Dziemianko et al. [90]. Head movements synthesis has also been demonstrated by Hofer and Shimodaira [139] and Sargin et al. [260]. It has been found that the fundamental frequency and pitch of speech are correlated with eye and head movements [139, 260, 90]. A possible direction of future work is to add additional modes to the joint probabilistic models in order to incorporate prosodic information. Prosodic information in both audio and visual modalities would be correlated with both the speech content and the underlying phonemes. As a result, a decomposition of the latent spaces into shared and private spaces as demonstrated by Ek et al. [99] and Salzmann et al. [259] can be adopted. The BIWI corpus [106] illustrated in Figure 7.1, can be used for prosody-driven facial animation.

- **Transferable speech animation** - Ideally, it would be desirable to have to learn a model of speech production on only one subject and then adapt the models using the limited data of a new corpus. This involves two components. The first is adaptation of the facial models and has been demonstrated by several researchers [294, 65, 280]. The second problem is the adaptation of the speech production models to correct the mismatch in audio recordings, transmission channel, environment noise, speaker, speaking style, as well as application contexts, and has been achieved using techniques such as maximum-a-posteriori (MAP), maximum likelihood linear regression (MLLR) and clustering adaptation [47]. With transferable speech animation, it should be possible to learn a speech model on LIPS data and then adapt it to DEMNOW data or vice-versa.
- **Expressive speech animation** - As humans, we are very sensitive to facial expressions and any synthetic facial animation that lacks expressiveness would quickly look unnatural to us. The ultimate aim of facial animation is to have fully expressive animation with gestures such as smiles, frowns and other facial cues to convey emotion. Expressive facial animation has been demonstrated by various researchers [43, 52, 81, 297] and involves two stages: expression analysis and expression synthesis. In expression analysis, the goal is to extract emotive cues from the speech data. Expression synthesis then maps these expressive cues to the face by having a model that factors style from content. Multilinear facial models have been widely used in expression synthesis [292, 198, 294]. The BIWI corpus [106] can be used for expressive facial animation because each utterance in the corpus is provided in different affective states such as: negative, anger, sadness, stress, contempt, fear, surprise, excitement, confidence, happiness and positive [106].

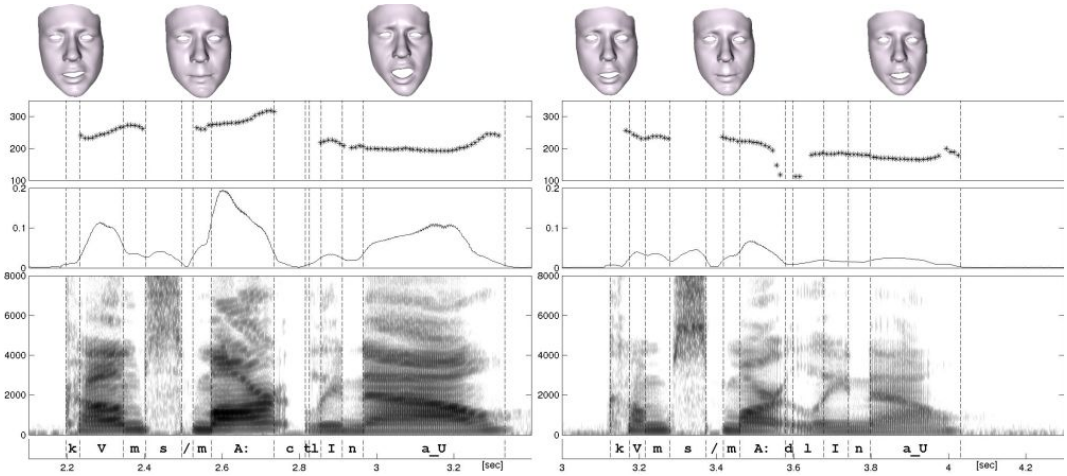


Figure 7.1: The BIWI 3D Audiovisual Corpus of Affective Communication [106].

Bibliography

[1] N. Ahmed, T. Natarajan, and K. R. Rao. Discrete cosine transform. *IEEE Trans. Computers*, pages 90–93, 1974.

[2] Oleg Alexander, Mike Rogers, William Lambeth, Matt Chiang, and Paul Debevec. Creating a photoreal digital actor: The digital emily project. *Conference for Visual Media Production*, 0:69–80, 2009.

[3] Oleg Alexander, Mike Rogers, William Lambeth, Jen-Yuan Chiang, Wan-Chun Ma, Chuan-Chang Wang, and Paul Debevec. The digital emily project: Achieving a photorealistic digital actor. *IEEE Computer Graphics and Applications*, 30:20–31, 2010.

[4] Brian Amberg, Andrew Blake, and Thomas Vetter. On compositional image alignment, with an application to active appearance models. In *CVPR'09: Proc. of the IEEE International Conference on Computer Vision and Pattern Recognition*, pages 1714–1721, 2009.

[5] Helen A. Cowan André-Pierre Benguerel. Coarticulation of upper lip protrusion in french. *Phonetica*, 30:41–55, 1974.

[6] Gonzalo R. Arce. *Nonlinear Signal Processing: A Statistical Approach.* Wiley Interscience, 2005.

[7] B. S. Atal and M. R. Schroeder. Predictive coding of speech signals. In *Proc. of IEEE Conf. on Communication and Processing*, pages 360–361, 1967.

[8] Francis R. Bach and Michael I. Jordan. A probabilistic interpretation of canonical correlation analysis. Technical report, University of California, Berkeley, 2005.

[9] Jörg-Hendrik Bach, Jörn Anemüller, and Birger Kollmeier. Robust speech detection in real acoustic backgrounds with perceptually motivated features. *Speech Communication*, 53:690–706, 2011.

[10] Gérard Bailly, Oxana Govokhina, Frédéric Elisei, and Gaspard Breton. Lip-synching using speaker-specific articulation, shape and appearance models. *EURASIP Journal on Audio, Speech and Music Processing*, 2009.

[11] Simon Baker and Iain Matthews. Equivalence and efficiency of image alignment algorithms. In *CVPR'01: Proc. of the IEEE International Conference on Computer Vision and Pattern Recognition*, volume 1, pages 1090–1097, 2001.

[12] M. Banbrook, S. McLaughlin, and I. Mann. Speech characterization and synthesis by nonlinear methods. *IEEE Transactions on Speech and Audio Processing*, 7(1): 1–17, 1999.

[13] D. Barber. Expectation correction for smoothed inference in switching linear dynamical systems. *Journal of Machine Learning Research*, 7:2515–2540, 2006.

[14] BEEP. The British English Pronunciation (BEEP) Dictionary. Online, December 2011. URL http://mi.eng.cam.ac.uk/comp.speech/Section1/Lexical/beep.html.

[15] F. Bell-Berti and K. Harris. Temporal patterns of coarticulation: Lip rounding. *Journal of the Acoustical Society of America*, 71(2):449–454, 1982.

[16] Serge Belongie, Jitendra Malik, and Jan Puzicha. Shape matching and object recognition using shape contexts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24:509–522, 2002.

[17] Jonas Beskow. Rule-based visual speech synthesis. In *Proc. of Eurospeech*, 1995.

[18] Jonas Beskow and Mikael Nordenberg. Data-driven synthesis of expressive visual speech using an mpeg-4 talking head. In *Proc. Interspeech 2005*, pages 793–796, 2005.

[19] Franck Bettinger, Timothy F. Cootes, and Christopher J. Taylor. Modelling facial behaviours. In *BMVC'02: Proc of British Machine Vision Conference*, 2002.

[20] Franck Bettinger, Timothy F. Cootes, and Christopher J. Taylor. A model of facial behaviour. In *FGR'04: Proc. of IEEE International Conference on Automatic Face and Gesture Recognition*, pages 797–806, 2004.

[21] Christopher M. Bishop. *Neural Networks for Pattern Recognition*. Oxford University Press, 1995.

[22] Christopher. M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag New York, Inc., 2006.

[23] R. A. Bladon and A. Al-Bamerni. One stage and two-stage temporal pattems of velar coarticulation. *The Journal of the Acoustical Society of America*, 72(S1): S104, 1982.

[24] R. A. W. Bladon and A. Al-Bamerni. Coarticulation resistance in english /l/. *Journal of Phonetics*, 4:137150, 1976.

[25] Volker Blanz and Thomas Vetter. A morphable model for the synthesis of 3D faces. In *SIGGRAPH '99: Proceedings of the 26th annual conference on Computer graphics and interactive techniques*, pages 187–194, 1999.

[26] Volker Blanz, Curzio Basso, Thomas Vetter, and Tomaso Poggio. Reanimating faces in images and video. In *EUROGRAPHICS'03: 24th Annual Conference of the European Association for Computer Graphics*, volume 22, pages 641–650, 2003.

[27] I. Borg and P.J.F. Groenen. *Modern Multidimensional Scaling: Theory and Applications*. Springer, 2005.

[28] George Borshukov, Dan Piponi, Oystein Larsen, J.P.Lewis, and Christina Tempelaar-Lietz. Universal capture - image-based facial animation for "the matrix reloaded". In *ACM SIGGRAPH 2003 Sketches and Applications Program*, 2003.

[29] Stephen Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004. ISBN 0521833787.

[30] M. Brand, N. Oliver, and A. Pentland. Coupled hidden Markov models for complex action recognition. In *CVPR'97: Proc. of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 994–999, 1997.

[31] Matthew Brand. Coupled hidden Markov models for modeling interacting processes. Technical Report TR-405, MIT Media lab Perceptual Computing/Learning and Common Sense, November 1996.

[32] Matthew Brand. Voice puppetry. In *SIGGRAPH '99: Proc. of the ACM Conference on Computer Graphics and Interactive Techniques*, 1999.

[33] Matthew Brand. Structure learning in conditional probability models via an entropic prior and parameter extinction. *Neural Computation*, 11(5):1155–1182, 1999.

[34] Christoph Bregler, Michele Covell, and Malcolm Slaney. Video rewrite: driving visual speech with audio. In *SIGGRAPH '97: Proc. of the ACM Conference on Computer Graphics and Interactive Techniques*, 1997.

[35] Martin Breidt. Facial animation in the industry. Technical report, MPI for Biological Cybernetics, Tbingen, 2006.

[36] E. Oran Brigham. *The Fast Fourier Transform: An Introduction to Its Theory and Application.* Prentice Hall, 1973.

[37] Richard L. Burden and J. Douglas Faires. *Numerical Analysis.* Thompson, 2005.

[38] J. Burg. Maximum entropy spectral analysis. In *Proc. of the 37th Meeting of the Society of Exploration Geophysicists*, 1967.

[39] Fabrice Caillette, Aphrodite Galata, and Toby Howard. Real-time 3-D human body tracking using variable length Markov models. In *BMVC'05: Proceedings of British Machine Vision Conference*, pages 469–478, 2005.

[40] Fabrice Caillette, Aphrodite Galata, and Toby Howard. Real-time 3-D human body tracking using learnt models of behaviour. *Computer Vision and Image Understanding*, 109(2):112–125, 2008.

[41] Yong Cao, Petros Faloutsos, and Frédéric Pighin. Unsupervised learning for speech motion editing. In *SCA'03: Proc. of the ACM SIGGRAPH/Eurographics symposium on Computer animation*, pages 225–231, 2003.

[42] Yong Cao, Petros Faloutsos, Eddie Kohler, and Frédéric Pighin. Real-time speech motion synthesis from recorded motions. In *SCA '04: Proc. of the ACM SIGGRAPH/Eurographics symposium on Computer animation*, 2004.

[43] Yong Cao, Wen C. Tien, Petros Faloutsos, and Frédéric Pighin. Expressive speech-driven facial animations. *ACM Transactions on Graphics*, 24(4):1283–1302, 2005.

[44] Maria. D. P. Carretero, David Oyarzun, Amalia Ortiz, Iker Aizpurua, and Jorge Posada. Virtual characters facial and body animation through the edition and interpretation of mark-up languages. *Computers & Graphics*, 29(2):189–194, 2005.

[45] C. Cave, I. Guaitella, R. Bertrand, S. Santi, F. Harlay, and R. Espesser. About the relationship between eyebrow movements and F0 variations. In *ICSLP'96: Proc. of International Conference on Spoken Language*, volume 4, pages 2175–2178, 1996.

[46] Chandramouli Chandrasekaran and Asif A. Ghazanfar. When what you see is not what you hear. *Nature Neuroscience*, 14(6):675–676, June 2011.

[47] Yao-Jen Chang and Tony Ezzat. Transferable videorealistic speech animation. In *SCA '05: Proc. of the 2005 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, pages 143–151. ACM, 2005.

[48] Jixu Chen, Minyoung Kim, Yu Wang, and Qiang Ji. Switching Gaussian process dynamic models for simultaneous composite motion tracking and recognition. In *CVPR'09: Proc. of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2655–2662, 2009.

[49] Tsuhan Chen. Audiovisual speech processing. *IEEE Signal Processing Magazine*, 18(1):9–21, 01 2001.

[50] Kyoungho Choi, Ying Luo, and Jenq-Neng Hwang. Hidden Markov model inversion for audio-to-visual conversion in an MPEG-4 facial animation system. *Journal of VLSI Signal Processing Systems*, 29:51–61, 2001.

[51] Erika Chuang and Christoph Bregler. Performance driven facial animation using blendshape interpolation. Technical report, Standford University, 2002.

[52] Erika Chuang and Christoph Bregler. Mood swings: expressive speech animation. *ACM Transactions on Graphics*, 24(2):331–347, 2005.

[53] M. M. Cohen, R. L. Walker, and D. W. Massaro. Perception of synthetic visual speech,. *Speechreading by Man and Machine: Models, Systems and Applications*, pages 153–158, 1996.

[54] M.M. Cohen and D.W. Massaro. Modeling coarticulation in synthetic visual speech. *Models and Techniques in Computer Animation*, pages 139–156, 1993.

[55] Paul R. Cohen. *Empirical methods for artificial intelligence*. MIT Press, 1995. ISBN 0-262-03225-2.

[56] Pierre Comon. Independent component analysis, a new concept? *Signal Processing*, 36:287–314, April 1994.

[57] T. F. Cootes, D. H. Cooper, C. J. Taylor, and J. Graham. Trainable method of parametric shape description. *Image and Vision Computing*, 10:289–294, 1992.

[58] T.F. Cootes and C.J. Taylor. Active shape models. In *BMVC'92: Proc. of the British Machine Vision Conference*, 1992.

[59] Timothy F. Cootes, Gareth J. Edwards, and Christopher J. Taylor. Active Appearance Models. In *ECCV '98: Proceedings of the 5th European Conference on Computer Vision-Volume II*, pages 484–498. Springer-Verlag, 1998.

[60] Timothy F. Cootes, Gareth J. Edwards, and Christopher J. Taylor. Active Appearance Models. *IEEE PAMI*, 23(6):681–685, 2001.

[61] Eric Cosatto. *Sample-Based Talking Head Synthesis*. PhD thesis, Ecole Polytechnique Federal de Lausanne (EPFL), 2002.

[62] Eric Cosatto, Gerasimos Potamianos, and Hans Peter Graf. Audio-visual unit selection for the synthesis of photo-realistic talking-heads. In *ICME'00: Proc. of IEEE International Conference on Multimedia & Expo*, pages 619–622, 2000.

[63] Darren Cosker. *Animation of a Hierarchical Appearance Based Facial Model and Perceptual Analysis of Visual Speech*. PhD thesis, Cardiff University, 2005.

[64] Darren Cosker, Dave Marshall, Paul. L. Rosin, and Yulia Hicks. Speech driven facial animation using a hidden Markov coarticulation model. In *ICPR '04: Proceedings of the 17th International Conference on Pattern Recognition*, volume 1, pages 128–131. IEEE Computer Society, 2004a.

[65] Darren Cosker, Steven Roy, Paul L. Rosin, and David Marshall. Re-mapping animation parameters between multiple types of facial model. In *MIRAGE'07*, volume 4418 of *Lecture Notes in Computer Science*, pages 365–376. Springer, 2007.

[66] Michele Covell. Eigen-points: Control-point location using principle component analyses. In *FG '96: Proceedings of the IEEE International Conference on Automatic Face and Gesture Recognition*, 1996.

[67] I. Craw and P. Cameron. Parameterising images for recognition and reconstruction. In *BMVC'91: Proc. of British Machine Vision Conference*, 1991.

[68] R. Daniloff and R. Hammarberg. On defining coarticulation. *Journal of Phonetics*, 1:239–248, 1973.

[69] Steven B. Davis and Paul Mermelstein. Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 28(4), 1980.

[70] Salil Deena and Aphrodite Galata. Speech-driven facial animation using a shared Gaussian process latent variable model. In *ISVC'09: Proc. of the International Symposium on Visual Computing*. Springer, 2009.

[71] Salil Deena, Shaobo Hou, and Aphrodite Galata. Visual speech synthesis by modelling coarticulation dynamics using a non-parametric switching state-space model. In *ICMI-MLMI'10: Proc. of the International Conference on Multimodal*

Interfaces and Workshop on Machine Learning for Multimodal Interaction. ACM, 2010.

[72] Marc P. Deisenroth, Marco F. Huber, and Uwe D. Hanebeck. Analytic moment-based Gaussian process filtering. In *ICML'09: Proc. of the ACM International Conference on Machine Learning*, pages 225–232, 2009.

[73] D.J. Dekle, C. A. Fowler, and M.G. Funnell. Audiovisual integration in perception of real words. *Perceptual Psychophysiology*, 51(4):355–62, 1992.

[74] B. Delaunay. Sur la sphére vide. *Akademii Nauk SSSR, Otdelenie Matematicheskikh i Estestvennykh Nauk*, 7:793–800, 1934.

[75] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society, series B*, 39(1):1–38, 1977.

[76] Zhigang Deng. *Data-driven facial animation synthesis by learning from facial motion capture data*. PhD thesis, University of Southern California, Los Angeles, CA, USA, 2006. Adviser-Neumann, Ulrich.

[77] Zhigang Deng and Ulrich Neumann. *Data-Driven 3D Facial Animation*. Springer, 2007.

[78] Zhigang Deng, J. P. Lewis, and Ulrich Neumann. Practical eye movement model using texture synthesis. In *ACM SIGGRAPH 2003 Sketches & Applications*, pages 1–1, 2003.

[79] Zhigang Deng, J. P. Lewis, and Ulrich Neumann. Synthesizing speech animation by learning compact speech co-articulation models. In *CGI'05: Proc. of the IEEE Computer Graphics International*, pages 19–25, 2005.

[80] Zhigang Deng, John P. Lewis, and Ulrich Neumann. Automated eye motion using texture synthesis. *IEEE Computer Graphics and Applications*, 25:24–30, 2005.

[81] Zhigang Deng, Ulrich Neumann, J. P. Lewis, Tae-Yong Kim, Murtaza Bulut, and Shrikanth Narayanan. Expressive facial animation synthesis by learning speech coarticulation and expression spaces. *IEEE Trans. on Visualization and Computer Graphics*, 12:1523–1534, 2006.

[82] Vincent Devin. *An interactive talking head*. PhD thesis, School of Computing, University of Leeds, 2002.

[83] Priya Dey, Steve C. Maddock, and Rod Nicolson. Evaluation of a viseme-driven talking head. In *TPCG'10: Proc. of Eurographics Theory and Practice of Computer Graphics*, pages 139–142, 2010.

[84] Ostendorf Digalakis, M. Ostendorf, and V. Digalakis. The stochastic segment model for continuous speech recognition. In *Proc. of the Asilomar Conference on Signals, Systems and Computers*, pages 964–968, 1991.

[85] Edsger Wybe Dijkstra. A Note on Two Problems in Connection with Graphs. *Numerical Mathematics*, 1:269–271, 1959.

[86] Barbara Dodd. The role of vision in the perception of speech. *Perception*, 6(1): 31–40, 1977.

[87] Arnaud Doucet and Christophe Andrieu. Iterative algorithms for optimal state estimation of jump Markov linear systems. *IEEE Transactions of Signal Processing*, 5:2487 – 2490, 2000.

[88] Arnaud Doucet, Neil J. Gordon, and Vikram Krishnamurthy. Particle filters for state estimation of jump Markov linear systems. *IEEE Transactions on Signal Processing*, 49:613–624, 2001.

[89] Shirley Dowdy, Stanley Weardon, and Daniel Chilko. *Statistics for Research*. Wiley Series on Probability and Statistics, 3rd edition, 2004.

[90] Michal Dziemianko, Gregor Hofer, and Hiroshi Shimodaira. HMM-based automatic eye-blink synthesis from speech. In *Proc. of Interspeech*, pages 1799–1802, 2009.

[91] J. Edge and A. Hilton. Visual speech synthesis from 3D video. In *IET European Conference on Visual Media Production*, 2006.

[92] J. Edge, A. Hilton, and P. Jackson. Model-based synthesis of visual speech movements from 3D video. *EURASIP Journal of Audio, Speech and Music Processing*, 2009.

[93] James D. Edge and Steve Maddock. Image-based talking heads using radial basis functions. In *Proceedings of the Theory and Practice of Computer Graphics*, 2003. ISBN 0-7695-1942-3.

[94] James D. Edge and Steve Maddock. Constraint-based synthesis of visual speech. In *ACM SIGGRAPH 2004 Sketches*. ACM, 2004.

[95] James D. Edge, Manuel A. S. Lorenzo, and Steve Maddock. Reusing motion data to animate visual speech. In *Symposium on Language, Speech and Gesture for Expressive Characters, AISB Convention: Motion, Emotion and Cognition*, pages 66–74, 2004.

[96] G.J. Edwards, C.J. Taylor, and T.F. Cootes. Learning to identify and track faces in image sequences. In *FGR'98: Proc. of the IEEE International Conference on Face and Gesture Recognition*, pages 260–265, 1998.

[97] A.A. Efros and T.K. Leung. Texture synthesis by non-parametric sampling. In *ICCV'99: Proc. of the IEEE International Conference on Computer Vision*, volume 2, pages 1033 –1038, 1999.

[98] Carl Henrik Ek, Philip H. S. Torr, and Neil D. Lawrence. Gaussian process latent variable models for human pose estimation. In *MLMI'07: Proc of the 4th International Workshop on Machine Learning for Multimodal Interaction*, 2007.

[99] Carl Henrik Ek, Jon Rihan, Philip H. S. Torr, Gregory Rogez, and Neil D. Lawrence. Ambiguity modeling in latent spaces. In *MLMI'08: Proc. 5th International Workshop on Machine Learning for Multimodal Interaction*, 2008.

[100] Carl Henrik Ek, Peter Jaeckel, Neil Campbell, Neil D. Lawrence, and Chris Melhuish. Shared Gaussian process latent variable models for handling ambiguous facial expressions. In *American Institute of Physics Conference Series*, 2009.

[101] Gwenn Englebienne. *Animating faces from speech.* PhD thesis, School of Computer Science, University of Manchester, 2009.

[102] Gwenn Englebienne, Tim F. Cootes, and Magnus Rattray. A probabilistic model for generating realistic lip movements from speech. In *NIPS'07: Advances in Neural Information Processing Systems*, 2007.

[103] Olov Engwall. Evaluation of a system for concatenative articulatory visual speech synthesis. In *Proc. of Interspeech*, 2002.

[104] Tony Ezzat. *Trainable Videorealistic Speech Animation.* PhD thesis, MIT, 2002.

[105] Tony Ezzat, Gadi Geiger, and Tomaso Poggio. Trainable videorealistic speech animation. In *SIGGRAPH '02: Proc. of the ACM conference on Computer Graphics and Interactive Techniques*, 2002.

[106] G. Fanelli, J. Gall, H. Romsdorfer, T. Weise, and L. Van Gool. A 3-D audio-visual corpus of affective communication. *IEEE Transactions on Multimedia*, 12(6):591 – 598, October 2010.

[107] E. Farnetani. *Speech Production and Speech Modeling*, chapter V-C-V lingual coarticulation and its spatiotemporal domain, pages 93–130. Kluwer Academic, Dordrecht, 1990.

[108] T.A. Faruquie, C. Neti, N.Rajput, L.V. Subramaniam, and A. Verma. Translingual visual speech synthesis. In *ICME 2000: Proc. of IEEE International Conference on Multimedia and Expo*, volume 2, pages 1089 –1092, 2000.

[109] David A. Ferrucci, Eric W. Brown, Jennifer Chu-Carroll, James Fan, David Gondek, Aditya Kalyanpur, Adam Lally, J. William Murdock, Eric Nyberg, John M. Prager, Nico Schlaefer, and Christopher A. Welty. Building Watson: An Overview of the DeepQA Project. *AI Magazine*, 31(3):59–79, 2010.

[110] Cletus G. Fisher. Confusions among visually perceived consonants. *Journal of Speech and Hearing Research*, 11:764–804, 1968.

[111] Terrence W. Fong, Illah Nourbakhsh, and Kerstin Dautenhahn. A survey of socially interactive robots. *Robotics and Autonomous Systems*, 2003.

[112] J. B. J. Fourier. *Théorie analytique de la chaleur*. Didot, 1822.

[113] C. A. Fowler, P. Rubin, R. Remez, and M. Turvey. Implications for speech production of a general theory of action. In B. Butterworth, editor, *Language Production I*, page 371420. London: Academic Press, 1980.

[114] Carol. A. Fowler and Elliot Saltzman. Coordination and coarticulation in speech production. *Language and Speech*, 36:171–195, 1993.

[115] Emily B. Fox, Erik B. Sudderth, and Michael I. Jordan. Nonparametric bayesian learning of switching linear dynamical systems. In *NIPS'08: Advances in Neural Information Processing Systems*, 2008.

[116] Jeroen Fransen, Dave Pye, Tony Robinson, Phil Woodland, and Steve Young. WSJCAM0 corpus and recording description. Technical report, Engineering Department, Cambridge University, 1994.

[117] Shengli Fu, Ricardo Gutierrez-Osuna, Anna Esposito, Praveen K. Kakumanu, and Oscar N. Garcia. Audio/visual mapping with cross-modal hidden Markov models. *IEEE Transactions on Multimedia*, 7(2):243–549, 2005.

[118] Aphrodite Galata, Neil Johnson, and David Hogg. Learning variable length Markov models of behaviour. *Computer Vision and Image Understanding*, 81 (3):398–413, 2001.

[119] Aphrodite Galata, Anthony G. Cohn, Derek Magee, and David Hogg. Modeling interaction using learnt qualitative spatio-temporal relations and variable length Markov models. In *ECAI'02: Proc. of European Conference on AI*, 2002.

[120] Gadi Geiger, Tony Ezzat, and Tomaso Poggio. Perceptual evaluation of video-realistic speech. In *CBCL Paper 224/AI Memo 2003003, MIT*, pages 2003–003, 2003.

[121] Jennifer George and Paul Gnanayutham. An experiment using personalised multimedia interfaces for speech therapy. In *Computers Helping People with Special Needs*, volume 5105, pages 1236–1243. 2008.

[122] Zoubin Ghahramani and Geoffrey E. Hinton. Variational learning for switching state-space models. *Neural Computation*, 12(4):831–864, 2000.

[123] E. Gladilin, S. Zachow, P. Deuflhard, and H. Hege. Anatomy- and physics-based facial animation for craniofacial surgery simulations. *Medical and Biological Engineering and Computing*, 42:167–170, 2004.

[124] Gene H. Golub and Charles F. Van Loan. *Matrix computations (3rd ed.)*. Johns Hopkins University Press, 1996.

[125] Oxana Govokhina, Gérard Bailly, Gaspard Breton, and Paul C. Bagshaw. Tda: a new trainable trajectory formation system for facial animation. In *Proc. of INTERSPEECH*, 2006.

[126] J. Gower. Generalized procrustes analysis. *Psychometrika*, 40:33–51, 1975.

[127] Keith Grochow, Steven L. Martin, Aaron Hertzmann, and Zoran Popović. Style-based inverse kinematics. *ACM Transactions on Graphics*, 23(3):522–531, 2004.

[128] Camilla Gustavsson, Linda Strindlund, and Emma Wiknertz. Verification, validation and evaluation of the virtual human markup language (VHML). Master's thesis, Department of Electrical Engineering, Linköping University, 2002.

[129] R. Gutierrez-Osuna, P.K. Kakumanu, A. Esposito, O.N. Garcia, A. Bojorquez, J.L. Castillo, and I. Rudomin. Speech-driven facial animation with realistic dynamics. *IEEE Transactions on Multimedia*, 7(1):33 – 42, 2005.

[130] Isabelle Guyon and Fernando Pereira. Design of a linguistic postprocessor using variable memory length Markov models. In *ICDAR'95: Proc. of IEEE International Conference on Document Analysis and Recognition*, pages 454–457, 1995.

[131] Richard Wesley Hamming. *Digital Filters*. Signal Processing Series. Prentice–Hall, Englewood Cliffs, 1977.

[132] David Hanson, Andrew Olney, Steve Prilliman, Eric Mathews, Marge Zielke, Derek Hammons, Raul Fernandez, and Harry Stephanou. Upending the Uncanny Valley. In *AAAI'05: Proc. of the Americanl Association for Artificial intelligence Conference*, pages 1728–1729, 2005.

[133] William J. Hardcastle and Nigel Hewlett. *Coarticulation: Theory, data and techniques*. Cambridge University Press, 2000.

[134] H. Hermansky and N. Morgan. RASTA processing of speech. *IEEE Transactions on Speech and Audio Processing*, 2(4):578 –589, 1994.

[135] Hynek Hermansky. Perceptual linear predictive (PLP) analysis of speech. *The Journal of the Acoustical Society of America*, 87(4):1738–1752, 1990.

[136] Hynek Hermansky, Nelson Morgan, Aruna Bayya, and Phil Kohn. RASTA-PLP speech analysis technique. In *ICASSP'92: IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 1, 1992.

[137] Sarah Hilder, Richard Harvey, and Barry-John Theobald. Comparison of human and machine-based lip-reading. In *AVSP'09: Proceedings of the International Conference on Auditory-Visual Speech Processing*, 2009.

[138] Jessica Hodgins, Sophie Jörg, Carol O'Sullivan, Sang Il Park, and Moshe Mahler. The saliency of anomalies in animated human characters. *ACM Transactions on Applied Perception*, 7(4):1–14, 2010.

[139] Gregor Hofer and Hiroshi Shimodaira. Automatic head motion prediction from speech data. In *Proc. of Interspeech*, pages 722–725, 2007.

[140] Gregor Hofer, Junichi Yamagishi, and Hiroshi Shimodaira. Speech-driven lip motion generation with a trajectory HMM. In *Interpeech'08: Proceedings of Interspeech*, 2008.

[141] Gregor Hofer, Korin Richmond, and Michael Berger. Lip synchronization by acoustic inversion. In *ACM SIGGRAPH 2010 Posters*, 2010.

[142] Gregor Otto Hofer. *Speech-driven animation using multi-modal Hidden Markov Models*. PhD thesis, School of Informatics, University of Edingburg, 2009.

[143] Myles Hollander and Douglas A. Wolfe. *Nonparametric Statistical Methods*. Wiley-Interscience, 2nd edition, 1999.

[144] John Holmes and Wendy Holmes. *Speech Synthesis and Recognition*. Taylor & Francis, Inc., 2nd edition, 2002.

[145] Harold Hotelling. Relations between two sets of variates. *Biometrika*, 28(3/4): 321–377, 1936.

[146] Shaobo Hou, Aphrodite Galata, Fabrice Caillette, Neil Thacker, and Paul Bromiley. Real-time body tracking using a Gaussian process latent variable model. In *ICCV'07: Proc. of the IEEE 11th International Conference on Computer Vision*, pages 1–8, 2007.

[147] J. Hu, W. Turin, and M. K. Brown. Language modeling using stochastic automata with variable length contexts. *Computer Speech & Language*, 11:1–16, 1997.

[148] Fu Jie Huang and Tsuhan Chen. Real-time lip-synch face animation driven by human voice. In *IEEE Second Workshop on Multimedia Signal Processing*, pages 352 –357, 1998.

[149] Haoda Huang, Jinxiang Chai, Xin Tong, and Hsiang-Tao Wu. Leveraging motion capture and 3d scanning for high-fidelity facial performance acquisition. In *SIGGRAPH'11: Proc. of the ACM International Conference on Computer Graphics and Interactive Techniques*, 2011.

[150] Xuedong Huang, Fileno Alleva, Hsiao wuen Hon, Mei yuh Hwang, and Ronald Rosenfeld. The SPHINX-II speech recognition system: An overview. *Computer, Speech and Language*, 7:137–148, 1992.

[151] Xuedong Huang, Alex Acero, and Hsiao-Wuen Hon. *Spoken Language Processing: A Guide to Theory, Algorithm and System Development.* Prentice Hall PTR, 2001.

[152] Aapo Hyvärinen. Fast and robust fixed-point algorithms for independent component analysis. *IEEE Trans. on Neural Networks*, 10(3):626–634, 1999.

[153] Aapo Hyvärinen and Errki Oja. Independent component analysis: Algorithms and applications. *Neural Networks*, 13(4-5):411–430, 2000.

[154] F. Itakura. Line spectrum representation of linear predictive coefficients of speech signals. *Journal of Acoustical Society of America*, 57:S35(A), 1975.

[155] F. Itakura and S. Saito. A statistical method for estimation of speech spectral density and formant frequencies. *Electrical and Communication in Japan*, 53-A (1):36–43, 1970.

[156] ITU-R. ITU-R Recommendation BT.470-6 Conventional Television Systems, 1998.

[157] ITU-T. Methods for subjective determination of transmission quality - series p: Telephone transmission quality; methods for objective and subjective assessment of quality, 1996.

[158] Jia Jia, Shen Zhang, Fanbo Meng, Yongxin Wang, and Lianhong Cai. Emotional audio-visual speech synthesis based on PAD. *IEEE Transactions on Audio, Speech, and Language Processing*, 19(3):570 –582, 2011.

[159] Chnstian Martyn Jones and Satnam Singh Dlay. Human-computer interaction and animation system for simple interfacing to virtual environments. In *IEEE International Conference on Systems, Man and Cybernetics*, 1997.

[160] Michael J. Jones and Tomaso Poggio. Multidimensional morphable models. In *ICCV '98: Proc. of IEEE the International Conference on Computer Vision*, 1998.

[161] Kolja Kähler, Jörg Haber, and Hans-Peter Seidel. Geometry-based muscle modeling for facial animation. In *GRIN'01: No description on Graphics interface*, pages 37–46. Canadian Information Processing Society, 2001.

[162] Praveen K. Kakumanu, Anna Esposito, Oscar N. Garcia, and Ricardo Gutierrez-Osuna. A comparison of acoustic coding models for speech-driven facial animation. *Speech Communication*, 48(6):598–615, 2006.

[163] Rudolph Emil Kalman. A new approach to linear filtering and prediction problems. *Transactions of the ASME–Journal of Basic Engineering*, 82(Series D): 35–45, 1960.

[164] Michael Kass, Andrew Witkin, and Demetri Terzopoulos. Snakes: Active contour models. *International Journal of Computer Vision*, 1:321–331, 1988.

[165] Patricia A. Keating. The window model of coarticulation: articulatory evidence. In J. Kingston and M. Beckman, editors, *Papers in Laboratory Phonology*, pages 451–470. Cambridge University Press, 1990.

[166] Erwin Keeve, Sabine Girod, Ron Kikinis, and Bernd Girod. Deformable modeling of facial tissue for craniofacial surgery simulation. *Computer Aided Surgery*, 3: 3–228, 1998.

[167] Scott A. King. Animating speech in games. In *MIG 2008: First International Workshop on Motion in Games*, pages 234–245. Springer-Verlag, 2008.

[168] M. Kirby and L. Sirovich. Application of the Karhunen-Loeve procedure for the characterization of human faces. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 12:103–108, 1990. ISSN 0162-8828.

[169] Dennis H. Klatt. Review of the ARPA speech understanding project. *Journal of the Acoustical Society of America*, 62(6):1345–1366, 1977.

[170] Jonathan Ko and Dieter Fox. GP-BayesFilters: Bayesian filtering using Gaussian process prediction and observation models. *Autonomous Robots*, 27(1):75–90, 2009.

[171] D. Koller and N. Friedman. *Probabilistic Graphical Models: Principles and Techniques*. MIT Press, 2009.

[172] V. A. Kozhevnikov and L. S. Chistovich. Speech: articulation and perception. *U. S. Dept. of Commerce, Clearinghouse for Federal Scientific and Technical Information, Joint Publications Research Service*, pages 256–270, 1965.

[173] Frank R. Kschischang, Brendan J. Frey, and Hans andrea Loeliger. Factor graphs and the sum-product algorithm. *IEEE Transactions on Information Theory*, 47: 498–519, 1998.

[174] Sumedha Kshirsagar and Nadia Magnenat-Thalmann. Visyllable based speech animation. In *Eurographics Computer Graphics Forum*, volume 22, 2003.

[175] Peter Ladeforged. *A course in phonetics*. Thomson Learning, 2001.

[176] P. L. Lai and C. Fyfe. Kernel and nonlinear canonical correlation analysis. *International Journal of Neural Systems*, 10(5):365–377, 2000.

[177] Neil D. Lawrence. Gaussian process latent variable models for visualisation of high dimensional data. In Sebastian Thrun, Lawrence Saul, and Bernhard Schölkopf, editors, *NIPS'03: Advances in Neural Information Processing Systems*. MIT Press, 2003.

[178] Neil D. Lawrence. Probabilistic non-linear principal component analysis with Gaussian process latent variable models. *Journal of Machine Learning Research*, 6:1783–1816, 2005.

[179] Neil D. Lawrence. Learning for larger datasets with the Gaussian process latent variable model. In M. Meila and X. Shen, editors, *AISTATS'07: Proceedings of of the Eleventh International Workshop on Artificial Intelligence and Statistics*, San Juan, Puerto Rico, 2007. Omnipress.

[180] Neil D. Lawrence and Joaquin Quiñonero-Candela. Local distance preservation in the GP-LVM through back constraints. In *ICML'06: Proc. of the International Conference on Machine Learning*, pages 513–520, 2006.

[181] Oscar Martinez Lazalde, Steve Maddock, and Michael Meredith. A constraint-based approach to visual speech for a mexican-spanish talking head. *International Journal of Computer Games Technology*, pages 9:1–9:7, 2008.

[182] Sooha Park Lee, Jeremy B. Badler, and Norman I. Badler. Eyes alive. *ACM Transactions on Graphics*, 21:637–644, 2002.

[183] Soonkyu Lee and Dongsuk Yook. Audio-to-visual conversion using hidden Markov models. In *PRICAI '02: Proc. of the 7th Pacific Rim International Conference on Artificial Intelligence*, pages 563–570. Springer-Verlag, 2002.

[184] Yuencheng Lee, Demetri Terzopoulos, and Keith Walters. Realistic modeling for facial animation. In *SIGGRAPH '95: Proc. of the 22nd Annual Conference on Computer Graphics and Interactive Techniques*, pages 55–62, 1995.

[185] Tue Lehn-Schiøler. *Making Faces State-Space Models Applied to Multi-Modal Signal Processing*. PhD thesis, Technical University of Denmark, 2005.

[186] Tue Lehn-Schiøler, Lars Kai Hansen, and Jan Larsen. Mapping from speech to images using continuous state space models. In *MLMI'04: Proc. of 1st International Workshop on Machine Learning for Multimodal Interaction*, 2005.

[187] Joe Letteri, Stephen Rosenbaum, and Richard Baneham. Making "Avatar". In *ACM SIGGRAPH 2010 Production Sessions*, 2010.

[188] Yan Li and Heung-Yeung Shum. Learning dynamic audio-visual mapping with input-output hidden Markov models. *IEEE Transactions on Multimedia*, 8(3): 542–549, 2006.

[189] Yu-Ming Liang, Sheng-Wen Shih, Arthur Chun-Chieh Shih, and Hong-Yuan Mark Liao. Understanding human behavior using a language modeling approach. In *ICIIHM'08: Proc. of the 2006 International Conference on Intelligent Information Hiding and Multimedia*, pages 331–334, 2006.

[190] B. Lindblom. *Speech Production and Speech Modeling*, chapter Explaining phonetic variation: A sketch of the H&H theory, pages 403–439. Kluwer Academic, Dordrecht, 1990.

[191] Kang Liu and Joern Ostermann. Realistic facial animation system for interactive services. In *LIPS 2008: Visual Speech Synthesis Challenge, Special Session in Interspeech 2008*, 2008.

[192] Kang Liu and Joern Ostermann. Optimization of an image-based talking head system. *EURASIP Journal on Audio, Speech, and Music Processing*, 2009.

[193] Anders Löfqvist. *Speech Production and Speech Modelling*, chapter Speech as audible gestures, pages 289–322. Kluwer Academic, Dordrecht, 1990.

[194] J. Lubker and T. Gay. Anticipatory labial coarticulation: Experimental, biological, and linguistic variables. *Journal of the Acoustical Society of America*, 71(2): 437–448, 1982.

[195] J. Luettin, G. Potamianos, and C. Neti. Asynchronous stream modeling for large vocabulary audio-visual speech recognition. In *ICASSP '01: Proc of IEEE International Conference onAcoustics, Speech, and Signal Processing*, volume 1, pages 169 –172, 2001.

[196] Jiyong Ma, Ron Cole, Bryan Pellom, Wayne Ward, and Barbara Wise. Accurate visible speech synthesis based on concatenating variable length motion capture data. *IEEE Transactions on Visualization and Computer Graphics*, 12:266–276, 2006.

[197] J. Macdonald, S. Andersen, and T. Bachmann. Hearing by eye: how much spatial degradation can be tolerated? *Perception*, 29:1155–1168, 2000.

[198] Ives Macedo, Emilio Vital Brazil, and Luiz Velho. Expression transfer between photographs through multilinear AAM's. In *SIBGRAPI'06: Proceedings of the 19th Brazilian Symposium on Computer Graphics and Image Processing*, pages 239–246. IEEE Computer Society, 2006.

[199] Akinobu Maejima, Hiroyuki Kubo, and Shigeo Morishima. Realistic facial animation by automatic individual head modeling and facial muscle adjustment. In *International Conference on Virtual and Mixed Reality*, pages 260–269, 2011.

[200] J. Makhoul. Spectral analysis of speech by linear prediction. *IEEE Trans. on Audio and Electroacoustics*, 21(3):140–148, 1973.

[201] J. D. Markel and A. H. Gray. On autocorrelation equations as applied to speech analysis. *IEEE trans. on Audio and Electroacoustics*, pages 69–79, 1973.

[202] John D. Markel and Augustine H.Gray. *Linear prediction of speech.* Springer-Verlag, 1976.

[203] Andrew Marriott. *A Facial Animation case study for HCI: the VHML-based Mentor System*, chapter MPEG-4 Facial Animation - The standard, implementations and applications. John Wiley, 2002.

[204] Iain Matthews and Simon Baker. Active appearance models revisited. *International Journal of Computer Vision*, 60(1):135 – 164, November 2004.

[205] Wesley Mattheyses, Lukas Latacz, and Werner Verhelst. Optimized photorealistic audiovisual speech synthesis using active appearance modeling. In *AVSP'10: Proc. of the International Conference on Auditory-visual Speech Processing*, pages 148–153, 2010.

[206] P. S. Maybeck. *Stochastic models, estimation and control.* Academic Press, 1982.

[207] H. McGurk and MacDonald. Hearing lips and seeing voices. *Nature*, 264:746–748, 1976.

[208] Javier Melenchón, Elisa Martínez, Fernando De La Torre, and José A. Montero. Emphatic visual speech synthesis. *IEEE Trans. on audio, speech and language processing*, 17(3):459–468, 2009.

[209] P. Menzerath and A. de LacerdaIn. Koartikulation, steuerung und lautabgrenzung. *Phonetische Studien*, 1, 1933.

[210] James Mercer. Functions of Positive and Negative Type, and their Connection with the Theory of Integral Equations. *Philosophical Transactions of the Royal Society of London Series A*, 209:415–446, 1909.

[211] B. Mesot and D. Barber. Switching linear dynamical systems for noise robust speech recognition. *IEEE Transactions on Audio, Speech, and Language Processing*, 15(6):1850–1858, 2007.

[212] E. M. Mikhail, J. S. Bethel, and J. C. McGlone. *Introduction to Modern Photogrammetry.* John Wiley & Sons, Inc., New York, 2001.

[213] Bart Mills. In Forest Gump, historical figures speak for themselves. *Chicago Tribune*, 1994.

[214] Kenneth L. Moll and Raymond G. Daniloff. Investigation of the timing of velar movements during speech. *The Journal of the Acoustical Society of America*, 50 (2B):678–684, 1971.

[215] Masahiro Mori. The Uncanny Valley. *Energy*, 7(4):33–35, 1970.

[216] MPEG-4. MPEG-4 international standard, 1998. URL http://mpeg.chiariglione.org/standards/mpeg-4/mpeg-4.htm.

[217] Florian Müller and Alfred Mertins. Noise robust speaker-independent speech recognition with invariant-integration features using power-bias subtraction. In *Proc. of Interspeech*, 2011.

[218] N. Nadtoka, J.R. Tena, A. Hilton, and J.D. Edge. High-resolution animation of facial dynamics. In *CVMP'04: Proc of the European Conference On Visual Media Production*, 2004.

[219] NTSC. National Television Systems Committee (NTSC) standard, 1953.

[220] Harry Nyquist. Certain topics in telegraph transmission theory. *Transactions of the American Institute of Electrical Engineers (AIEE)*, 47:617–644, 1928.

[221] J. J. Odell. *The use of context in large vocabulary speech recognition.* PhD thesis, Cambridge University, 1995.

[222] Sang Min Oh, James M. Rehg, Tucker Balch, and Frank Dellaert. Learning and inference in parametric switching linear dynamical systems. In *ICCV '05: Proceedings of the Tenth IEEE International Conference on Computer Vision*, pages 1161–1168, Beijing, China, 2005. IEEE Computer Society.

[223] Sang Min Oh, James M. Rehg, Tucker Balch, and Frank Dellaert. Data-driven MCMC for learning and inference in switching linear dynamic systems. In *AAAI'05: Proc. of the American Association for Artificial Intelligence Conference*, volume 2, pages 944–949, 2005.

[224] S. E. G. Öhman. Coarticulation in VCV utterances: Spectrographic measurements. *The Journal of the Acoustical Society of America*, 39(1):151–168, 1966.

[225] M. Ostendorf and I. Bulyko. The impact of speech recognition on speech synthesis. In *Proceedings of IEEE Workshop on Speech Synthesis*, pages 99 – 106, 2002.

[226] M. Ostendorf and S. Roukos. A stochastic segment model for phoneme-based continuous speech recognition. volume 37, pages 1857 –1869, 1989.

[227] M. Ostendorf, I. Bechwati, and O. Kimball. Context modeling with the stochastic segment model. In *ICASSP'92: Proc. of IEEE International conference on Acoustics, Speech and Signal Processing*, pages 389–392, 1992.

[228] Slim Ouni, Michael M. Cohen, Hope Ishak, and Dominic W. Massaro. Visual contribution to speech perception: measuring the intelligibility of animated talking heads. *EURASIP Journal on Audio, Speech and Music Processing*, 2007.

[229] Elmer Owens and Barbara Blazek. Visemes observed by hearing-impaired and normal-hearing adult viewers. *Journal of Speech and Hearing Research*, 28(3): 381–393, 1985.

[230] Frederic I. Parke and Keith Waters. *Computer facial animation.* A. K. Peters, Ltd., second edition, 2008. ISBN 9781568814483.

[231] Frederick I. Parke. *A parametric model of human faces.* PhD thesis, University of Utah, 1974.

[232] V. Pavlovic, J.M. Rehg, Tat-Jen Cham, and K.P. Murphy. A dynamic Bayesian network approach to figure tracking using learned dynamic models. In *ICCV'99: Proc. of the Seventh IEEE International Conference on Computer Vision*, volume 1, pages 94–101, 1999.

[233] Vladimir Pavlović, James M. Rehg, and Tat-Jen Cham. A dynamic Bayesian network approach to tracking using learned switching dynamic models. In *Proc. of the Third International Workshop on Hybrid Systems: Computation and Control*, pages 366–380, 2000.

[234] Vladimir Pavlović, James M. Rehg, and John MacCormick. Learning switching linear models of human motion. In *NIPS'00: Advances in Neural Information Processing Systems*, pages 981–987, 2000.

[235] P. Paysan, R. Knothe, B. Amberg, S. Romdhani, and T. Vetter. A 3D face model for pose and illumination invariant face recognition. In *AVSS'09: Proc. of the IEEE International Conference on Advanced Video and Signal Based Surveillance*, 2009.

[236] K. Pearson. On lines and planes of closest fit to systems of points in space. *Philosophical Magazine*, 2:559–572, 1901.

[237] Yuru Pei and Hongbin Zha. Stylized synthesis of facial speech motions. *Computer Animation and Virtual Worlds*, 18(4-5):517–526, 2007.

[238] Yuru Pei and Hongbin Zha. Visyllable-specific facial transition motion embedding and extraction. In *ICIP'09: Proceedings of the IEEE International Conference on Image Processing*, pages 1789–1792, 2009.

[239] Catherine Pelachaud, Norman I. Badler, and Mark Steedman. Generating facial expressions for speech. *Cognitive Science*, 20(1):1–46, 1996.

[240] J. Perkell and M. Cohen. Preliminary support for a "hybrid model" of anticipatory coarticulation. In *Proceedings of the 12th International Congress on Acoustics*, page A36. Canadian Acoustical Association, 1986.

[241] J. W. Picone. Signal modeling techniques in speech recognition. *Proceedings of the IEEE*, 81(9):1215–1247, 1993.

[242] S. Platt and N. Badler. Animating facial expressions. *ACM SIGGRAPH Computer Graphics*, 15(3):245–252, 1981.

[243] John G. Proakis and Dimitris G. Manolakis. *Digital signal processing (3rd ed.): principles, algorithms, and applications*. Prentice-Hall, Inc., 1996. ISBN 0-13-373762-4.

[244] Stephen J. Pueblo. Videorealistic facial animation for speech-based interfaces. Master's thesis, Department of Electrical Engineering and Computer Science, Massachussets Institute of Technology, 2009.

[245] Joaquin Quiñonero-Candela and Carl Edward Rasmussen. A unifying view of sparse approximate Gaussian process regression. *Journal of Machine Learning Research*, 6:1939–1959, 2005.

[246] Lawrence Rabiner and B. Gold. *Theory and application of digital signal processing*. Prentice-Hall, 1975.

[247] Lawrence Rabiner and Biing-Hwang Juang. *Fundamentals of speech recognition*. Prentice-Hall, Inc., 1993.

[248] Lawrence R. Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. In *Readings in speech recognition*, volume 77, pages 257–286, 1989.

[249] R.R. Rao, Tsuhan Chen, and R.M. Mersereau. Audio-to-visual conversion for multimedia communication. *IEEE Transactions on Industrial Electronics*, 45(1): 15–22, 1998.

[250] Carl Edward Rasmussen and Hannes Nickisch. Gaussian Processes for Machine Learning (GPML) Toolbox. *Journal of Machine Learning Research*, 11:3011–3015, 2010.

[251] Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian Processes for Machine Learning (Adaptive Computation and Machine Learning)*. The MIT Press, 2006.

[252] Jonathan Richards. Lifelike animation heralds new era for computer games. *Times Online*, August 2008. URL http://technology.timesonline.co.uk/tol/news/tech_and_web/article4557935.ece.

[253] Dana Ron, Yoram Singer, and Naftali Tishby. The power of amnesia: Learning probabilistic automata with variable memory length. *Machine Learning*, 25:117–149, 1996.

[254] Antti-Veikko Rosti and Mark Gales. Rao-Blackwellised Gibbs Sampling for Switching Linear Dynamical Systems. In *ICASSP'04: Proc. of IEEE International Conference on Acoustics, Speech, and Signal Processing*, pages 809–812, 2004.

[255] Antti-Veikko Rosti and M.J.F. Gales. Switching linear dynamical systems for speech recognition. Technical report, University of Cambridge, 2003.

[256] J. Rothweiler. Polyphase quadrature filters–a new subband coding technique. In *ICASSP '83: IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 8, pages 1280–1283, 1983.

[257] Sam T. Roweis and Lawrence K. Saul. Nonlinear dimensionality reduction by locally linear embedding. *SCIENCE*, 290:2323–2326, 2000.

[258] Payam Saisan, Alessandro Bissacco, Ro Chiuso, and Stefano Soatto. Modeling and synthesis of facial motion driven by speech. In *ECCV'04: European Conference on Computer Vision*, pages 456–467, 2004.

[259] Mathieu Salzmann, Carl Henrik Ek, Raquel Urtasun, and Trevor Darrell. Factorized orthogonal latent spaces. *Journal of Machine Learning Research - Proceedings Track*, 9:701–708, 2010.

[260] Mehmet E. Sargin, Yucel Yemez, Engin Erzin, and Ahmet M. Tekalp. Analysis of head gesture and prosody patterns for prosody-driven head-gesture animation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30:1330–1345, August 2008.

[261] Claude. E. Shannon. Communication in the Presence of Noise. *Proceedings of the IRE*, 37(1):10–21, 1949.

[262] Matt Shannon, Heiga Zen, and William J. Byrne. The effect of using normalized models in statistical speech synthesis. In *Proc. of Interspeech*, pages 121–124, 2011.

[263] Aaron Shon, Keith Grochow, Aaron Hertzmann, and Rajesh Rao. Learning shared latent structure for image synthesis and robotic imitation. In *NIPS'05: Advances in Neural Information Processing Systems 18*, pages 1233–1240. MIT Press, 2005.

[264] Eftychios Sifakis, Andrew Selle, Avram Robinson-mosher, and Ronald Fedkiw. Simulating speech with a physics-based facial muscle model. In *SCA'06: Proc. of Symposium on Computer Animation*, pages 261–270, 2006.

[265] L. Sirovich and M. Kirby. Low-dimensional procedure for the characterization of human faces. *Journal of the Optical Society of America*, 4(3):519–524, 1987.

[266] Edward Snelson and Zoubin Ghahramani. Sparse Gaussian processes using pseudo-inputs. In *NIP'06: Advances in Neural Information Processing Systems*, pages 1257–1264. MIT press, 2006.

[267] Nikolay Stefanov, Aphrodite Galata, and Roger Hubbold. Real-time hand tracking with variable-length Markov models of behaviour. In *V4HCI'05: IEEE International Workshop on Vision for Human-Computer Interaction in conjunction with CVPR*, 2005.

[268] Nikolay Stefanov, Aphrodite Galata, and Roger Hubbold. Real-time hand tracker using variable-length Markov models of behaviour. *Computer Vision and Image Understanding*, 108(2):98–115, 2007.

[269] Barnabás Takács and Bernadette Kiss. The virtual human interface: A photorealistic digital human. *IEEE Computer Graphics and Applications*, 23(5):38–45, 2003.

[270] Jianhua Tao, Le Xin, and Panrong Yin. Realistic visual speech synthesis based on hybrid concatenation method. *IEEE Transactions on Audio, Speech & Language Processing*, 17(3):469–477, 2009.

[271] C. J. Taylor, D. H. Cooper, and J. Graham. Training models of shape from sets of examples. In *BMVC'92: Proc. of the British Machine Vision Conference*, pages 9–18, 1992.

[272] Paul Taylor. *Text-to-Speech Speech Synthesis*. Cambridge University Press, 2009.

[273] A.Murat Tekalp and Jörn Ostermann. Face and 2-D mesh animation in MPEG-4. *Signal Processing: Image Communication*, 15(4-5):387 – 421, 2000.

[274] Joshua B. Tenenbaum, Vin Silva, and John C. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500):2319–2323, 2000.

[275] D. Terzopoulos and K. Waters. Analysis and synthesis of facial image sequences using physical and anatomical models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(6):569–579, 1993.

[276] Demetri Terzopoulos, Barbara Mones, Michael M. Cohen, Frederic Parke, Doug Sweetland, and Keith Waters. SIGGRAPH 97 panel on facial animation: Past, present and future, 1997. URL http://www.cs.ucla.edu/~dt/siggraph97-panel/.

[277] B. Theobald, G. Cawley, A. Bangham, I.Matthews, and N. Wilkinson. Comparing text-driven and speech-driven visual speech synthesisers. In *Proc. of Interspeech*, 2008.

[278] Barry-John Theobald. *Visual speech synthesis using shape and appearance models.* PhD thesis, School of Information Systems, University of East Anglia, 2003.

[279] Barry-John Theobald and Nicholas Wilkinson. A real-time speech-driven talking head using Active Appearance Models. In *AVSP'07: Proc. of the International Conference on Auditory-Visual Speech Processing*, 2007.

[280] Barry-John Theobald, Iain A. Matthews, Jeffrey F. Cohn, and Steven M. Boker. Real-time expression cloning using appearance models. In *ICMI '07: Proceedings of the 9th International Conference on Multimodal Interfaces*, pages 134–139. ACM, 2007.

[281] Barry-John Theobald, Sascha Fagel, Gérard Bailly, and Frédéric Elisei. LIPS2008: Visual speech synthesis challenge. In *Proc. of Interspeech*, 2008.

[282] Barry-John Theobald, Nicholas Wilkinson, and Iain Matthews. On evaluating synthesised visual speech. In *AVSP'08: Proc. of International Conference on Auditory-visual Speech Processing*, pages 7–12, 2008.

[283] Angela Tinwell, Mark Grimshaw, Debbie Abdel Nabi, and Andrew Williams. Facial expression of emotion and perception of the Uncanny Valley in virtual characters. *Computers in Human Behavior*, 2010.

[284] Angela Tinwell, Mark Grimshaw, and Andrew Williams. *Games Computing and Creative Technologies*, chapter Uncanny Speech, pages 213–234. 2011.

[285] Michael E. Tipping and Chris M. Bishop. Probabilistic principal component analysis. *Journal of the Royal Statistical Society, Series B*, 61:611–622, 1999.

[286] M. K. Titsias and N. D. Lawrence. Bayesian Gaussian process latent variable model. In *AISTATS'10: Proc of the International Conference on Artificial Intelligence and Statistics*, pages 844–851, 2010.

[287] Alan M. Turing. Computing machinery and intelligence. *Mind*, 59:433–460, 1950.

[288] Matthew Turk and Alex Pentland. Eigenfaces for recognition. *Journal of Cognitive Neuroscience*, 3:71–86, January 1991.

[289] Ryan Turner, Marc Peter Deisenroth, and Carl Edward Rasmussen. State-space inference and learning with Gaussian processes. *Journal of Machine Learning Research - Proceedings Track*, 9:868–875, 2010.

[290] Raquel Urtasun, David J. Fleet, and Pascal Fua. 3D people tracking with Gaussian process dynamical models. In *CVPR'06: Proc. of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 238–245, 2006.

[291] Saeed Vaseghi. Multimedia signal processing lecture notes. Technical report, Department of Electronics and Computer Engineering ,Brunel University, 2008.

[292] M. A. O. Vasilescu and Demetri Terzopoulos. Multilinear analysis of image ensembles: Tensorfaces. In *ECCV '02: Proceedings of the 7th European Conference on Computer Vision-Part I*, pages 447–460. Springer-Verlag, 2002.

[293] Andrew J. Viterbi. Error bounds for convolutional codes and an asymptotically optimal decoding algorithm. *IEEE Transactions on Information Theory*, 13:260–269, 1967.

[294] Daniel Vlasic, Matthew Brand, Hanspeter Pfister, and Jovan Popovic. Face transfer with multilinear models. In *SIGGRAPH '06: ACM SIGGRAPH 2006 Courses*, page 24, Boston, Massachusetts, 2006.

[295] John Volkman, Stanley Smith Stevens, and Edwin Newman. A scale for the measuremnt of the psychological magnitude of pitch. *Journal of the acoustical society of america*, 8:185–190, 1937.

[296] J. Volkmann, S. S. Stevens, and E. B. Newman. A scale for the measurement of the psychological magnitude pitch. *The Journal of the Acoustical Society of America*, 8(3):208–208, 1937.

[297] Kevin Wampler, Daichi Sasaki, Li Zhang, and Zoran Popović. Dynamic, expressive speech animation from a single mesh. In *SCA '07: Proceedings of the 2007 ACM SIGGRAPH/Eurographics symposium on Computer animation*, pages 53–62. Eurographics Association, 2007.

[298] Guang-Yi Wang, Mau-Tsuen Yang, Cheng-Chin Chiang, and Wei-Kai Tai. A talking face driven by voice using hidden Markov model. *Journal of Information Science Engineering*, 22(5):1059–1075, 2006.

[299] Jack M. Wang, David J. Fleet, and Aaron Hertzmann. Gaussian process dynamical models. In *NIPS'05: Advances in Neural Information Processing Systems 18*. MIT Press, 2005.

[300] Jack M. Wang, David J. Fleet, and Aaron Hertzmann. Gaussian process dynamical models for human motion. *IEEE trans. on Pattern Analysis and Machine Intelligence*, 30(2):283–298, 2008.

[301] Lijuan Wang, Xiaojun Qian, Lei Ma, Yao Qian, Yining Chen, and Frank K. Soong. A real-time text to audio-visual speech synthesis system. In *Proc. of Interspeech*, 2008.

[302] Lijuan Wang, Wei Han, Xiaojun Qian, and Frank Soong. Photo-real lips synthesis with trajectory-guided sample selection. In *SSW'10: Proc. of the Speech Synthesis Workshop*, 2010.

[303] Lijuan Wang, Xiaojun Qian, Wei Han, and Frank K. Soong. Synthesizing photo-real talking head via trajectory-guided sample selection. In *INTERSPEECH'10:Proc. of the Annual Conference of the International Speech Communication Association*, pages 446–449, 2010.

[304] Lijuan Wang, Yi-Jian Wu, Xiaodan Zhuang, and Frank K. Soong. Synthesizing visual speech trajectory with minimum generation error. In *ICASSP'11: Proc. of the IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 4580–4583, 2011.

[305] Shengzheng Wang, James C.Gee, and Jie Yang. A framework for craniofacial surgery simulation based on prespecified target face configurations. In *ISBI'09: Proceedings of the Sixth IEEE international Symposium on Biomedical Imaging*, pages 1055–1058, 2009.

[306] Yi Wang, Lizhu Zhou, Jianhua Feng, Jianyong Wang, and Zhi-Qiang Liu. Mining complex time-series data by learning Markovian models. In *ICDM '06: Proc. of the IEEE International Conference on Data Mining*, pages 1136–1140, 2006.

[307] Keith Waters. A muscle model for animation three-dimensional facial expression. *ACM SIGGRAPH Computer Graphics*, 21(4):17–24, 1987.

[308] H. Weid. The CMU Pronunciation Dictionary, release 0.6. Carnegie Mellon University, 1998. URL http://www.speech.cs.cmu.edu/cgi-bin/cmudict.

[309] Thibaut Weise, Sofien Bouaziz, Hao Li, and Mark Pauly. Real-time performance-based facial animation. In *SIGGRAPH'11: Proc. of the ACM International Conference on Computer Graphics and Interactive Techniques*, 2011.

[310] Axel Weissenfeld, Kang Liu, and Jörn Ostermann. Video-realistic image-based eye animation via statistically driven state machines. *The Visual Computer: International Journal of Computer Graphics*, 0(0), 2009.

[311] S. Weitz. *Nonverbal communication: readings with commentary*. Oxford University Press, 1979. ISBN 9780195024470.

[312] Lei Xie and Zhi-Qiang Liu. A coupled HMM approach to video-realistic speech animation. *Pattern Recognition*, 40(8):2325–2340, 2007.

[313] Lei Xie and Zhi-Qiang Liu. Realistic mouth-synching for speech-driven talking face using articulatory modelling. *IEEE Transactions on Multimedia*, 9(3):500–510, 2007.

[314] E. Yamamoto, S. Nakamura, and K. Shikano. Lip movement synthesis from speech based on hidden Markov models. *Speech Communication*, 26(1–2):105–115, 1998.

[315] Jonathan S. Yedidia, William T. Freeman, and Yair Weiss. *Understanding belief propagation and its generalizations*, pages 239–269. Morgan Kaufmann Publishers Inc., 2003. ISBN 1-55860-811-7.

[316] Lijun Yin, Xiaochen Chen, Yi Sun, Tony Worm, and Michael Reale. A high-resolution 3D dynamic facial expression database. In *FGR'08: Proc. of IEEE International Conference on Automatic Face and Gesture Recognition*, 2008.

[317] S.J. Young. The HTK Hidden Markov Model Toolkit: Design and philosophy. Technical report, University of Cambridge, Department of Engineering, 1993.

[318] Steve J. Young, D. Kershaw, J. Odell, D. Ollason, V. Valtchev, and P. Woodland. *The HTK Book Version 3.4*. Cambridge University Press, 2006.

[319] I.A. Ypsilos. *Capture and Modelling of 3D Face Dynamics*. PhD thesis, School of Electronics and Physical Sciences, University of Surrey, 2004.

[320] Ioannis A. Ypsilos and Adrian Hilton. Video-rate capture of dynamic face shape and appearance. In *Journal of New Generation Computing*, pages 117–122, 2004.

[321] Ioannis A. Ypsilos, Adrian Hilton, Aseel Turkmani, and Philip J. B. Jackson. Speech-driven face synthesis from 3D video. In *3DPVT'04:Second International Symposium on 3D Data Processing, Visualization and Transmission*, 2004.

[322] Heiga Zen and Mark J. F. Gales. Decision tree-based context clustering based on cross validation and hierarchical priors. In *ICASSP'11: Proc. of the IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 4560–4563, 2011.

[323] Heiga Zen, Keiichi Tokuda, and Tadashi Kitamura. Simultaneous clustering of phonetic context, dimension, and state position for acoustic modeling using decision trees. *Systems and Computers in Japan*, 36:44–55, December 2005.

[324] Heiga Zen, Takashi Nose, Junichi Yamagishi, Shinji Sako, Takashi Masuko, Alan W. Black, and Keiichi Tokuda. The HMM-based Speech Synthesis System (HTS) Version 2.0. In *Proc. of ISCA Workshop on Speech Synthesis*, 2007.

[325] Heiga Zen, Keiichi Tokuda, and Tadashi Kitamura. Reformulating the HMM as a trajectory model by imposing explicit relationships between static and dynamic feature vector sequences. *Computer Speech & Language*, 21(1):153–173, 2007.

[326] Xiaodan Zhuang, Lijuan Wang, Frank K. Soong, and Mark Hasegawa-Johnson. A minimum converted trajectory error (MCTE) approach to high quality speech-to-lips conversion. In *Proc. of Interspeech*, pages 1736–1739, 2010.

[327] Eckart Zitzler, Marco Laumanns, and Stefan Bleuler. A tutorial on evolutionary multiobjective optimization. In *In Metaheuristics for Multiobjective Optimisation*, pages 3–38. Springer-Verlag, 2003.

[328] Onno Zoeter and Tom Heskes. Deterministic approximate inference techniques for conditionally gaussian state space models. *Statistics and Computing*, 16(3): 279–292, 2006.

[329] E. Zwicker. Subdivision of the audible frequency range into critical bands. *Acoustical Society of America Journal*, 33:248, 1961.

Appendix A

Fast ICA Algorithm

A description of the FastICA algorithm is given here. Much of the material has been adapted from [153].

Given a vector $\mathbf{x} = \{x_n\}_{n=1}^N$, the aim of ICA is to find a linear projection $\mathbf{A}$ that projects a vector of independent components, $\mathbf{s} = \{s_n\}_{n=1}^N$ to $\mathbf{x}$.

$$\mathbf{x} = \mathbf{A}\mathbf{s} \tag{A.1}$$

By independent components, it is meant that each component is statistically independent from the other. Two variables y_1 and y_2 are statistically independent if their joint probability distribution can be be factorised as follows:

$$p(y_1, y_2) = p(y_1)p(y_2) \tag{A.2}$$

Statistical independence does not follow from uncorrelation. Statistically independent variables are uncorrelated but the converse is not true. For two variables, y_1 and y_2 to be uncorrelated, their covariance should be zero:

$$\mathbb{E}[y_1 y_2] - \mathbb{E}[y_1]\mathbb{E}[y_2] = 0 \tag{A.3}$$

The independent components can be computed from $\mathbf{x}$ using the inverse matrix of $\mathbf{A}$, $\mathbf{W} = \mathbf{A}^{-1}$ as follows:

$$\mathbf{s} = \mathbf{W}\mathbf{x} \tag{A.4}$$

For the components to be independent, they have to be non-Gaussian. Thus, independent components can be estimated by maximising their non-Gaussianity. There are several measures of non-Gaussianity, namely: kurtosis, which is the fourth-order moment of the distribution; negentropy, which is based on the information-theoretic measure of differential entropy; and mutual information, which is a measure of the dependence of two random variables. The FastICA algorithm uses negentropy as the measure of non-Gaussianity.

A.1 Negentropy

Entropy is a measure of the randomness of a random variable. The more random or unpredictable or unstructured a variable is, the larger is its entropy. The entropy of a discrete random variable Y is given by Eqn A.5, where a_i are the possible values of Y.

$$H(Y) = -\sum_i P(Y = a_i) \log P(Y = a_i) \tag{A.5}$$

For continuous-valued random variables and vectors, the differential entropy is used, which, for a random vector $\mathbf{y}$ is given by:

$$H(\mathbf{y}) = \int f(\mathbf{y}) \log f(\mathbf{y}) d\mathbf{y} \tag{A.6}$$

A Gaussian variable has the largest entropy among all random variables of equal variance. Thus, entropy is a good measure of non-Gaussianity. However, to obtain a measure that is zero for a Gaussian variable and is always non-negative, a modified version of differential entropy, called negentropy is used. The negentropy J of a random vector y is defined by:

$$J(\mathbf{y}) = H(\mathbf{y}_{gauss}) - H(\mathbf{y}) \tag{A.7}$$

where $\mathbf{y}_{gauss}$ is a Gaussian random variable that has the same covariance matrix as $\mathbf{y}$.

In practice, computing the negentropy of a random variable or vector can be quite difficult, so approximations are used instead. A classicial method of approximating negentropy is through the use of higher-order moments such as:

$$J(y) \approx \frac{1}{12}\mathbb{E}[y^3]^2 + \frac{1}{48} kurt(y)^2 \tag{A.8}$$

where $kurt(y)$ is the kurtosis of y.

However, more robust measures of negentropy have been developed [153]. The approximation used in the FastICA algorithm is:

$$J(y) \propto \big(\mathbb{E}[G(y)] - \mathbb{E}[G(v)]\big)^2 \tag{A.9}$$

where v is a Gaussian variable of zero mean and unit variance and G is any non-quadratic function. If $G(y) = y^4$, then we obtain a kurtosis-based approximation, as in Eqn A.8. The following choices for the function G have been proven useful:

$$G_1(u) = \frac{1}{a_1} \cosh a_1 u, \quad G_2(u) = -\exp\left(-u^2/2\right) \tag{A.10}$$

where $1 \leq a_1 \leq 2$ is some suitable constant.

A.2 Pre-processing

Prior to applying any ICA algorithm, some pre-processing is usually carried out in order to make the ICA estimation simpler and better conditioned.

A.2.1 Centering

The first step is mean-centering $\mathbf{x}$:

$$\mathbf{x} = \mathbf{x} - \mathbb{E}[\mathbf{x}] \tag{A.11}$$

This is mostly done to simplify the ICA algorithm.

A.2.2 Whitening

After mean-centering, the variable $\mathbf{x}$ is whitened. Whitening is a linear transforming of $\mathbf{x}$ so that the new vector $\tilde{\mathbf{x}}$ consists of uncorrelated components with variances equal to one. In other words, the covariance of $\tilde{\mathbf{x}}$ equals the identity matrix:

$$\mathbb{E}[\tilde{\mathbf{x}}\tilde{\mathbf{x}}^T] = \mathbf{I} \tag{A.12}$$

Whitening is done by first performing the eigendecomposition of the covariance matrix $\mathbf{C} = \mathbb{E}[\tilde{\mathbf{x}}\tilde{\mathbf{x}}^T]$

$$\mathbf{C} = \mathbf{P}\mathbf{\Lambda}\mathbf{P}^T \tag{A.13}$$

where $\mathbf{P}$ is an orthogonal matrix of eigenvectors of $\mathbf{C}$ and $\mathbf{\Lambda} = diag(\lambda_1, \ldots \lambda_n)$ is the diagonal matrix of its eigenvalues. The whitened matrix $\tilde{\mathbf{x}}$ is then given by:

$$\tilde{\mathbf{x}} = \mathbf{P}\mathbf{\Lambda}^{-\frac{1}{2}}\mathbf{P}^T\mathbf{x} \tag{A.14}$$

where $\mathbf{\Lambda}^{-\frac{1}{2}}$ is computed as $\mathbf{\Lambda}^{-\frac{1}{2}} = diag(\lambda_1^{-\frac{1}{2}}, \ldots, \lambda_n^{-\frac{1}{2}})$.

A useful procedure is to reduce the dimensionality of the data by discarding components whose eigenvalues are too small, as done during PCA. This reduces the noise in the data. In our method, we apply ICA to principal components, which can be whitened by normalising the variance of each principal component to one.

A.3 FastICA for one unit

In this section, the FastICA algorithm for finding one independent component is presented, followed by a generalisation of the algorithm to multiple components in the next section.

The FastICA algorithm is based on a fixed-point iteration scheme for estimating the weight vector $\mathbf{w}$, such as to maximise the non-Gaussianity of $\mathbf{w}^T\mathbf{x}$, as measured by Eqn A.9. An approximative Newton iteration is used for the optimisation. Taking g

to be the derivative of the function G, the derivatives of the functions in Eqn A.10 are given by:

$$g_1(u) = \tanh(a_1 u), \quad g_2(u) = u \exp\left(-u^2/2\right) \tag{A.15}$$

The general form of the FastICA algorithm for one unit is given by the following steps:

1. Choose an initial (e.g. random) weight vector $\mathbf{w}$.
2. Set $\mathbf{w}^+ = \mathbb{E}[\mathbf{x}G(\mathbf{w}^T\mathbf{x})] - \mathbb{E}[g(\mathbf{w}^T\mathbf{x})]\mathbf{w}$.
3. Set $\mathbf{w} = \mathbf{w}^+/\|\mathbf{w}^+\|$.
4. If not converged, go back to 2.

A.4 FastICA for several units

To run the FastICA algorithm for several independent components, the algorithm for one component needs to be run for each component, followed by a decorrelation of the outputs $\mathbf{w}_1^T\mathbf{x}, \ldots \mathbf{w}_n^T\mathbf{x}$ using a decorrelation method similar to Gram-Schmidt orthogonalisation method [124]. Thus, after p independent components, $\mathbf{w}_1 \ldots \mathbf{w}_p$ have been estimated, the one-unit fixed-point algorithm for $\mathbf{w}_{p+1}$ is run and after each iteration, $\mathbf{w}_{p+1}$ is substracted from the projections $\mathbf{w}_{p+1}^T\mathbf{w}_j\mathbf{w}_j$ for $j = 1 \ldots p$ of the previously estimated p vectors. This is followed by a renormalisation of $\mathbf{w}_{p+1}$. These two steps are summarised as follows:

1. Set $\mathbf{w}_{p+1} = \mathbf{w}_{p+1} - \sum_{j=1}^{p} \mathbf{w}_{p+1}^T\mathbf{w}_j\mathbf{w}_j$
2. Set $\mathbf{w}_{p+1} = \mathbf{w}_{p+1}/\sqrt{\mathbf{w}_{p+1}^T\mathbf{w}_{p+1}}$

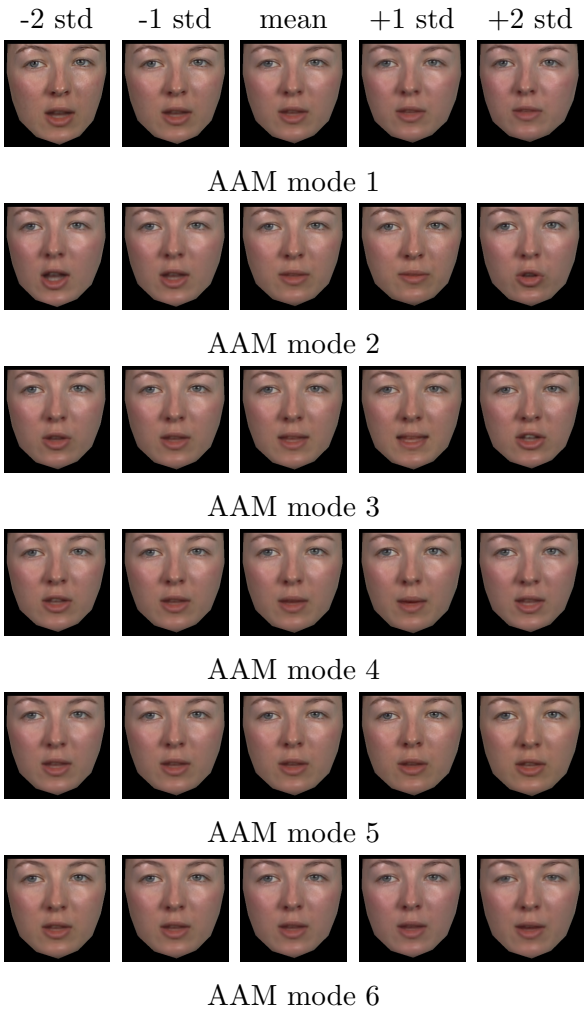
An alternative to the above algorithm is sometimes used when no vectors are priviledged over others. Taking $\mathbf{W} = (\mathbf{w}_1, \ldots, \mathbf{w}_n)^T$, which have been estimated using the FastICA algorithm for one unit, the following two steps can be used to find the independent components:

1. Set $\mathbf{W} = \mathbf{W}/\sqrt{\|\mathbf{W}\mathbf{W}^T\|}$
   Repeat step 2 until convergence (i.e. change in negentropy measure less than threshold):
2. Set $\mathbf{W} = \frac{3}{2}\mathbf{W} - \frac{1}{2}\mathbf{W}\mathbf{W}^T\mathbf{W}$

Appendix B

AAM Modes of Variation

LIPS AAM Modes of Variation



LIPS AAM Modes of Variation

-2 std -1 std mean +1 std +2 std

AAM mode 7

AAM mode 8

AAM mode 9

AAM mode 10

AAM mode 11

AAM mode 12

AAM mode 13

AAM mode 14

AAM mode 15

AAM mode 16

LIPS AAM Modes of Variation

-2 std -1 std mean +1 std +2 std

AAM mode 17

AAM mode 18

AAM mode 19

AAM mode 20

AAM mode 21

AAM mode 22

AAM mode 23

AAM mode 24

AAM mode 25

AAM mode 26

LIPS AAM Modes of Variation

-2 std -1 std mean +1 std +2 std

AAM mode 27

AAM mode 28

AAM mode 29

AAM mode 30

AAM mode 31

AAM mode 32

AAM mode 33

DEMNOW AAM Modes of Variation

-2 std -1 std mean +1 std +2 std

AAM mode 1

AAM mode 2

AAM mode 3

AAM mode 4

AAM mode 5

AAM mode 6

AAM mode 7

AAM mode 8

AAM mode 9

AAM mode 10

DEMNOW AAM Modes of Variation

-2 std -1 std mean +1 std +2 std

AAM mode 11

AAM mode 12

AAM mode 13

AAM mode 14

AAM mode 15

AAM mode 16

AAM mode 17

AAM mode 18

AAM mode 19

AAM mode 20

DEMNOW AAM Modes of Variation

-2 std -1 std mean +1 std +2 std

AAM mode 21

AAM mode 22

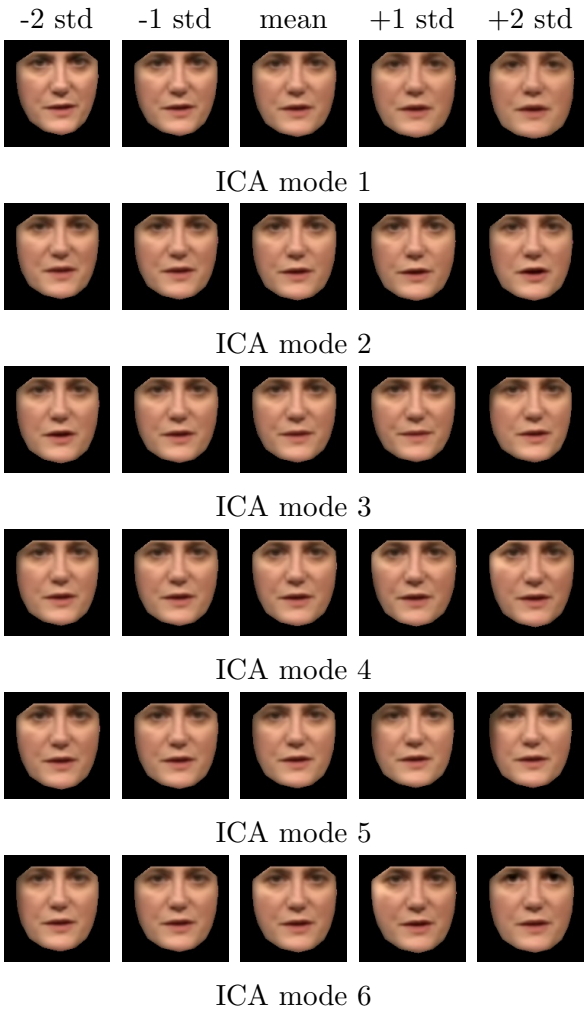
AAM mode 23

AAM mode 24

Appendix C

ICA Modes of Variation

DEMNOW ICA Modes of Variation



DEMNOW ICA Modes of Variation

-2 std -1 std mean +1 std +2 std

ICA mode 7

ICA mode 8

ICA mode 9

ICA mode 10

ICA mode 11

ICA mode 12

ICA mode 13

ICA mode 14

ICA mode 15

ICA mode 16

DEMNOW ICA Modes of Variation

-2 std -1 std mean +1 std +2 std

ICA mode 17

ICA mode 18

ICA mode 19

ICA mode 20

ICA mode 21

ICA mode 22

ICA mode 23

ICA mode 24

Appendix D

Animation Frames

Synthesis frames using phoneme VLMM SSGPDM with sequential optimisation for a test sequence of the LIPS corpus is given in Figure D.1.

Synthesis frames using phoneme VLMM SSGPDM with sequential optimisation for a test sequence of the DEMNOW corpus is given in Figure D.2.

Examples of ground truth and synthetic videos can also be found on: http://aig.cs.man.ac.uk/people/salil/visual_speech_synthesis_videos/.

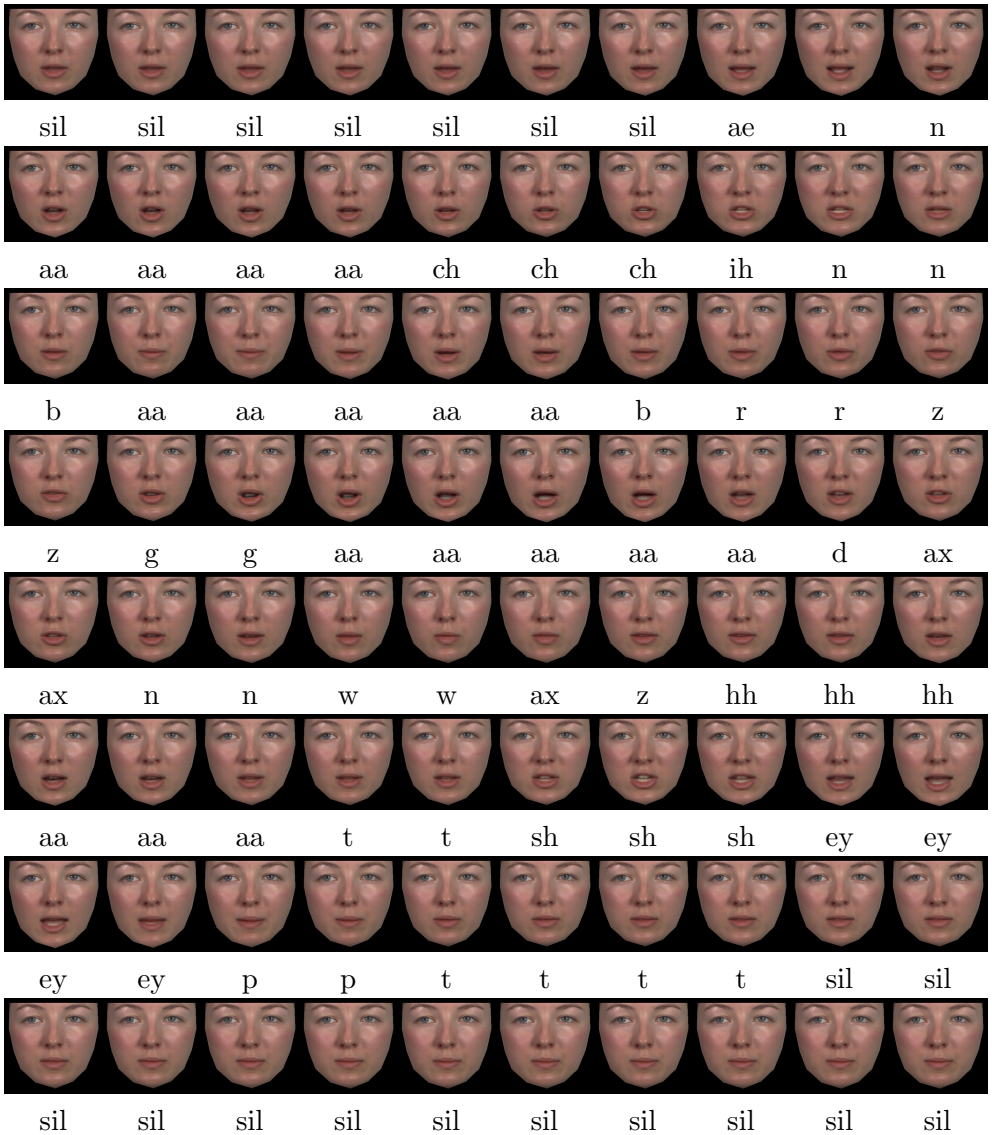


Figure D.1: Synthesis frames a sequence of the LIPS dataset with the utterance: "An arch in Barbara's garden was heart shaped" with BEEP phonetic labels underneath each frame.

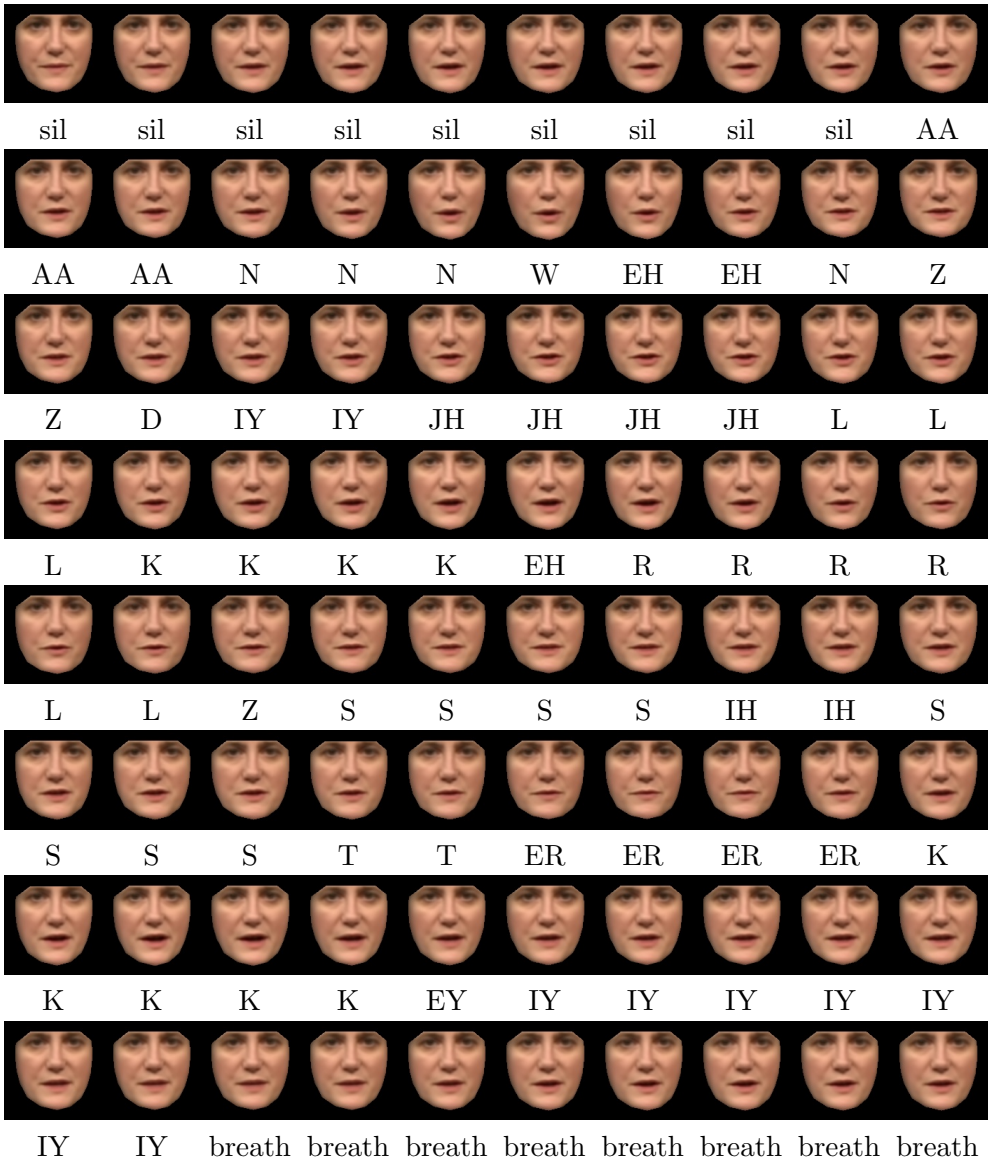


Figure D.2: Synthesis frames for a sequence of the DEMNOW dataset with the utterance: "On Wednesday Jill Carroll's sister Katie" with CMU phonetic labels underneath each frame.