UC Merced
UC Merced Electronic Theses and Dissertations

Title
Data-driven approaches to articulatory speech processing

Permalink
https://escholarship.org/uc/item/2x16w7bp

Author
Qin, Chao

Publication Date
2011

Peer reviewed|Thesis/dissertation

eScholarship.org Powered by the California Digital Library
University of California

UNIVERSITY OF CALIFORNIA, MERCED

Data-driven approaches to articulatory speech processing

A dissertation submitted in partial satisfaction of the
requirements for the degree
Doctor of Philosophy

in

Electrical Engineering and Computer Science

by

Chao QIN

Committee in charge:

Professor Miguel Á. Carreira-Perpiñán, Chair
Professor David Noelle
Professor Marcelo Kallmann
Professor Bhiksha Raj (Carnegie Mellon University)

May 2011

Copyright
Chao QIN, May 2011
All rights reserved.

The dissertation of Chao QIN is approved, and it is acceptable in quality and form for publication on microfilm and electronically:

Prof. David Noelle

Prof. Marcelo Kallmann

Prof. Bhiksha Raj

Prof. Miguel Á. Carreira-Perpiñán

Chair

University of California, Merced

May 2011

DEDICATION

To my parents, Jianming Qin and Lifen Rong.

TABLE OF CONTENTS

Signature Page . . . iii

Dedication . . . iv

Table of Contents . . . v

List of Figures . . . viii

List of Tables . . . xi

Acknowledgements . . . xii

Vita and Publications . . . xiii

Abstract of the Dissertation . . . xiv

Chapter 1 Introduction . . . 1
- 1.1 Why articulatory speech processing? . . . 1
- 1.2 Why data-driven approaches? . . . 4
- 1.3 Articulatory data acquisition . . . 4
- 1.4 Challenges and our efforts . . . 6
- 1.5 Structure of thesis . . . 8

Chapter 2 Empirical study of nonuniqueness in articulatory inversion . . . 10
- 2.1 Introduction . . . 10
- 2.2 Nonuniqueness in the full 16D articulatory space . . . 13
  - 2.2.1 Methodological setup . . . 13
  - 2.2.2 Experimental results . . . 14
- 2.3 Nonuniqueness in individual 2D articulatory spaces . . . 16
  - 2.3.1 Methodological setup . . . 20
  - 2.3.2 Experimental results . . . 21
- 2.4 Discussions . . . 23
- 2.5 Summary . . . 24

Chapter 3 Comparison of acoustic features for articulatory inversion . . . 26
- 3.1 Introduction . . . 26
- 3.2 Methodological setup . . . 27
  - 3.2.1 Acoustic features . . . 27
  - 3.2.2 Feature parameterization and post-processing . . . 29
  - 3.2.3 Inversion method . . . 29
- 3.3 Experiments . . . 30
  - 3.3.1 Experimental setup . . . 30
  - 3.3.2 Comparison of acoustic features and parameterizations . . . 32
  - 3.3.3 Effect of time delay . . . 33
- 3.4 Summary . . . 33

Chapter 4 Articulatory inversion of American English /ɹ/ by conditional density modes . . . 35
4.1 Introduction . . . 35
4.2 Computational approaches to articulatory inversion . . . 36
4.3 Inversion by conditional density modes . . . 39
4.4 Experimental results . . . 42
4.5 Summary . . . 46

Chapter 5 Predictive modeling of tongue shapes . . . 48
5.1 Can we predict the entire tongue shape from a few landmarks? . . . 48
5.2 Previous work . . . 50
5.3 Tongue data acquisition . . . 51
5.4 Predictive modeling of tongue shapes . . . 54
5.5 Experimental results . . . 56
5.5.1 Experimental setup . . . 56
5.5.2 Predicting tongue contours from a few landmarks . . . 57
5.5.3 Predictive performance as a function of choice of landmarks . . . 59
5.6 Summary . . . 63

Chapter 6 Adaptation of a predictive model of tongue shapes . . . 65
6.1 Why adapt the predictive model? . . . 65
6.2 Related work . . . 67
6.3 The predictive model of tongue shapes . . . 70
6.4 Feature adaptation of predictive models with full contours . . . 70
6.4.1 Global feature adaptation . . . 71
6.4.2 Local feature adaptation . . . 74
6.5 Feature adaptation of predictive models with partial contours . . . 77
6.6 Experimental results . . . 79
6.6.1 Speaker adaptation of predictive models of tongue shapes . . . 79
6.6.2 Reconstruction of the full tongue contour from EMA/X-Ray microbeam 83
6.7 Discussion . . . 86
6.8 Summary . . . 87

Chapter 7 Estimating missing data sequences in X-ray microbeam recordings . . . 88
7.1 Missing data in X-ray microbeam recordings . . . 88
7.2 Deriving mappings with varying sets of inputs and outputs from a density model 90
7.3 Experimental results . . . 92
7.3.1 Reconstruction of artificially blacked-out data . . . 92
7.3.2 Reconstruction of truly missing data . . . 94
7.4 Summary . . . 96

Chapter 8 Conclusions . . . 98
8.1 Summary . . . 98
8.2 Contribution of the thesis . . . 101
8.3 Public data resources . . . 101
8.4 Directions for future work . . . 102
8.4.1 Articulatory inversion . . . 102
8.4.2 Tongue reconstruction . . . 103

8.4.3 Missing data reconstruction 103
8.4.4 Trajectory inverse kinematics 103

Appendix A Articulatory speech databases 104
A.1 The Wisconsin X-ray microbeam database 104
A.2 The MOCHA-TIMIT database 105
A.3 Tongue contour dataset 109

Appendix B Acoustic feature extraction 113
B.1 Pre-processing 113
B.2 Linear predictive analysis 114
B.3 Filterbank and cepstral analysis 116
B.4 Dynamic features 117

Appendix C Trajectory inverse kinematics 118
C.1 Introduction 118
C.2 Computational approaches to inverse kinematics 120
C.3 Trajectory inverse kinematics by conditional density modes 121
C.3.1 Experimental results 122
C.3.2 Discussions 132
C.4 Trajectory inverse kinematics by nonlinear and nongaussian tracking 134
C.4.1 Conditional density by tracking 135
C.4.2 Experiments with a PUMA 560 robot arm 136
C.5 Practical considerations 141
C.5.1 Fine positioning 141
C.5.2 Real-time manipulation 143
C.5.3 Handling kinematic singularity 144
C.5.4 Handling joint limits and obstacles 144
C.6 Summary 144

Bibliography 147

LIST OF FIGURES

Figure 1.1: Human speech production system 2
Figure 1.2: An X-ray image of midsagittal vocal tract shapes 5
Figure 1.3: State-of-the-art articulatory acquisition techniques 7

Figure 2.1: The acoustic-to-articulatory mapping problem 11
Figure 2.2: Wisconsin X-ray microbeam articulatory database 15
Figure 2.3: Example of nonuniqueness for a reference acoustic vector $\mathbf{y} = /\theta/$ (`tp004` "<u>th</u>ings") 17
Figure 2.4: Examples of sounds showing multimodality 18
Figure 2.5: Examples of sounds showing unimodality 19
Figure 2.6: Histogram of the cloud width (standard deviation per dimension) for the inverse mapping 20
Figure 2.7: Scatterplots of the eigenvalues of the covariance matrix of the inverse set for some articulators 22
Figure 2.8: Histograms of the eigenvalues for individual articulators and the entire vocal tract . 23
Figure 2.9: Sample plots of the inverse sets for a given sound $\mathbf{y}_n$ in the XRMB (speaker `jw11`) in the space of `T1` to `T3`, density contours, modes and palate) 24

Figure 3.1: Smooth articulatory trajectories vs. jagged acoustic trajectories 27
Figure 3.2: Illustration of smoothing of acoustic temporal trajectories 30
Figure 3.3: MOCHA-TIMIT articulatory database 31
Figure 3.4: Performance comparison of different acoustic features and parameterizations 32
Figure 3.5: Effect of time delay between an acoustic frame and an articulatory frame 33

Figure 4.1: Modeling multi-valued mapping by multimodal conditional distribution 40
Figure 4.2: Articulatory dataset for American English /ɹ/ 43
Figure 4.3: Sequence of reconstructions for the retroflex /ɹ/ in utterance `tp096` "<u>r</u>oll" 45
Figure 4.4: Sequence of reconstructions for the bunched /ɹ/ in utterance `tp040` "<u>r</u>ag" 46
Figure 4.5: Temporal true trajectories and reconstructions for the bunched /ɹ/ in utterance `tp050` "<u>r</u>ow" 47
Figure 4.6: Inversion error and correlation per articulator channel 47

Figure 5.1: Locations of tongue pellets in XRMB and MOCHA 49
Figure 5.2: Typical ultrasound tongue images in our contour dataset 53
Figure 5.3: The prediction problem: given the 2D locations of K landmarks located on the tongue midsagittal contour ($\mathbf{x}$), reconstruct the entire contour ($\mathbf{y}$), represented by P 2D points. 54
Figure 5.4: Tongue contour dataset for speaker `maaw0` 56
Figure 5.5: Generalization performance of predictive mapping from $K = 24$ landmarks to the $P = 24$ tongue contour 58
Figure 5.6: Fast training vs. conventional training for RBF 58
Figure 5.7: Selected frames comparing the true contour and the contours estimated by the spline interpolation and our RBF prediction, for $K = 3$ landmarks 60
Figure 5.8: Predictive errors (RMSE) as a function of number of training contours 61
Figure 5.9: Error (RMSE) incurred by the RBF prediction, linf prediction, and spline interpolation 62
Figure 5.10: Optimal placement of K landmarks for spline interpolation, linf, and RBF 64

Figure 6.1: Examples of difficult contour tracking 66
Figure 6.2: Data-driven predictive model of the tongue contour (P1) 70
Figure 6.3: Adaptation of the predictive model given full contours (P2) 71
Figure 6.4: Five sample contours from **S1** aligned with a 2D-wise mapping **g** 74
Figure 6.5: Adaptation of the predictive model without full contours (P3) 77
Figure 6.6: Contour datasets for two speakers 80
Figure 6.7: Aligned contour datasets for two speakers 81
Figure 6.8: Local vs. global adaptation as a function of the number of adaptation contours N . 82
Figure 6.9: Local vs. global adaptation as a function of the number of landmarks K 83
Figure 6.10: Effect of regularization and data selection in adaptation in MOCHA for a given frame 84
Figure 6.11: Realistic tongue reconstruction for MOCHA 85
Figure 6.12: Typical tongue shapes during normal speech production in the contour dataset . . . 86
Figure 6.13: Snapshots of realistic tongue reconstructions for XRMB 86

Figure 7.1: Example of typical mistracks for one pellet in the XRMB (pellet schematic at right) 89
Figure 7.2: Histogram of the number of missing articulators for speakers `jw11` and `jw45` 92
Figure 7.3: Effect of the bandwidth σ on reconstructing missing articulators 93
Figure 7.4: Reconstruction error for each missing articulator 94
Figure 7.5: Reconstruction of artificially blacked-out articulators `T1,T1,T3,T4` for the utterance `tp011` 95
Figure 7.6: Reconstruction of truly missing data (i.e., one articulator is missing at a time) with a GM 97

Figure A.1: Traces of 5 second recordings of 8 articulators for utterance `tp105` from speaker `jw11` 105
Figure A.2: Traces of 2 second recordings of 7 articulators for utterance `001` from speaker `fsew0`. 108
Figure A.3: Outline of hard palate in tongue contour dataset for speaker `maaw0` 110
Figure A.4: Ultrasound recording mechanics 111
Figure A.5: Tongue contour recording with two mental balls attached to the tongue surface . . . 111

Figure B.1: Linear prediction model of speech 115
Figure B.2: Modular representation of a filterbank based cepstral parameterizations 116

Figure C.1: Serial link robot manipulators 119
Figure C.2: Geometry of the planar 2-link arm 122
Figure C.3: Marginal densities $p(\boldsymbol{\theta})$ and $p(\mathbf{x})$ for the fine ($M = 2500$ components) GTM model $p(\boldsymbol{\theta}, \mathbf{x})$ (4–dimensional), as a contour plot 123
Figure C.4: Sample conditional densities $p(\boldsymbol{\theta}|\mathbf{x} = (0.78, 0.48))$ for GTMs and MDN 125
Figure C.5: True and reconstructed trajectories with the fine GTM model and the pseudoinverse 126
Figure C.6: True and reconstructed trajectories with the MDN model and the pseudoinverse . . 126
Figure C.7: Four trajectories in $\boldsymbol{\theta}$-space that produce the same $\mathbf{x}$-trajectory for the planar 2-link robot arm 127
Figure C.8: A schematic plot of PUMA560 (reproduced with Matlab Robotics toolbox [29]) . . 129
Figure C.9: Training set for the PUMA 560 robot arm 129
Figure C.10: Conditional density modes for the MDN model and the PUMA 560 robot arm . . . 130
Figure C.11: Reconstruction of an elliptical trajectory for the PUMA 560 robot arm by MDN . . 131
Figure C.12: Reconstruction of a figure–8 trajectory for the PUMA 560 robot arm by MDN . . . 131
Figure C.13: Reconstruction of the loopy trajectory for the redundant arm by MDN 132

Figure C.14: Reconstruction of the figure–8 trajectory for the redundant arm by MDN 133
Figure C.15: The tracking framework for trajectory inverse kinematics 136
Figure C.16: Illustration for a planar 2-link robot arm of using conditional modes to represent multiple inverses 137
Figure C.17: Reconstruction of a figure-8 trajectory for the PUMA 560 robot arm by nonlinear and nongaussian tracking 138
Figure C.18: Reconstruction of an elliptical trajectory for the PUMA 560 robot arm by nonlinear and nongaussian tracking 139
Figure C.19: Pointwise reconstruction errors in the workspace for the planar two-link robot arm . 142
Figure C.20: Pointwise reconstruction errors of the test set in the workspace for the PUMA 560 robot arm 143
Figure C.21: Example of handling the joint limit by conditional density modes using the planar 2-link robot arm 145

LIST OF TABLES

Table 2.1: Nonuniqueness percentages for speaker `jw11` in XRMB 22

Table 4.1: The average RMSE and correlation for tongue and all coils 45

Table A.1: XRMB sentences by speaker `jw11` used in the experiments 106
Table A.2: XRMB sentences by speaker `jw11` used in the experiments (continued) 107
Table A.3: XRMB speakers used in the experiments 108
Table A.4: Testing sentences selected from the MOCHA-TIMIT database used in the experiments 109
Table A.5: Speakers available in the MOCHA-TIMIT database 109
Table A.6: Tongue contour dataset used in the experiments 112
Table A.7: Speakers recorded for the English tongue contour dataset 112

Table C.1: Reconstruction errors for the 2D robot arm by conditional modes 128
Table C.2: Reconstruction errors for the 3D robot arm (PUMA 560) by MDN 131
Table C.3: Reconstruction errors for the redundant manipulator by MDN 132
Table C.4: Reconstruction errors for PUMA 560 robot arm by nonlinear, nongaussian tracking . 140
Table C.5: Average run time per trajectory point by nonlinear, nongaussian tracking (sec.), $M =$ 1 000 particles on a Quad Core 2.0 Ghz machine with 2GB of RAM. 140
Table C.6: Pointwise reconstruction errors for the 2D robot arm 141
Table C.7: Pointwise reconstruction errors for PUMA560 on the regular grids 142

ACKNOWLEDGEMENTS

There are too many to thank for all the help through the long journey of my PhD. First of all, I am very grateful to my supervisor, Miguel Á. Carreira-Perpiñán, for having guided me with a practical, meticulous and highly dedicated research attitude, for introducing interesting problems, for full-range research skill training including analyzing, presenting, writing, and defending research work as well as programming and teaching, and for infusing me with his intellectual honesty.

I'd like to thank my thesis committee, Prof. David Noelle, Prof. Marcelo Kallmann, Prof. Bhiksha Raj for reviewing my thesis and insightful comments. I'd like to thank Prof. Shawn Newsam and Prof. Stefano Carpin, Prof. Ming-Hsuan Yang, Prof. Florin Rusu, Prof. Alberto Cerpa, all EECS labmates, Tomiko Hale, and Sheryl Wyan for encouragement and help during my UCM period.

I'd like to thank Prof. Steve Renals, Korin Richmond, Prof. Simon King, Dr. Junichi Jamagishi, Zhen-Hua Lin, Songfang Huang, Le Zhang, and many other CSTR labmates for all the help during my stay at Edinburgh University. Special thanks to Dr. Alwan Wrench for recording ultrasound tongue data, which constitutes an essential part of this thesis.

I'd like to thank Prof. Todd Leen, Prof. Brian Roark for serving as my thesis committee as well as other faculty and labmates from AdSyl and CSLU labs for the training in machine learning and speech technology during my stay at OGI .

I'd like to thank specially for my MPhil supervisors, Prof. Tan Lee, Prof. P. C. Ching, and Prof. Helen Meng for opening the door for me to enter the academia world and leading me to the gateway to speech research, and many labmates from CUHK.

I'd like to thank Prof. Karen Livescu for warm response, Prof. John Westbury for Wisconsin X-ray Microbeam database, Shrikanth Narayanan and John Hogden for help on articulatory work, Dr. Frank Soong and Dr. Yao Qian for continuous inspirations.

I am very grateful for the NSF career award and NSF award IIS-0711186 for funding support.

Last but not least, I thank my parents for their selfless and irreplaceable support during every moment of my PhD.

VITA

2005	M. Phil. in Electronic Engineering, The Chinese University of Hong Kong

PUBLICATIONS

Qin, C., Carreira-Perpiñán, M. Á. and Farhadloo, M. (2010): "Adaptation of a tongue shape model by local feature transformations". *Interspeech 2010*, pp. 1596-1599.

Qin, C. and Carreira-Perpiñán, M. Á. (2010): "Estimating missing data sequences in X-ray microbeam recordings". *Interspeech 2010*, pp. 1592-1595.

Qin, C. and Carreira-Perpiñán, M. Á. (2010): "Articulatory inversion of American English /ɹ/ by conditional density modes". *Interspeech 2010*, pp. 1998-2001.

Qin, C. and Carreira-Perpiñán, M. Á. (2010): "Reconstructing the full tongue contour from EMA/X-Ray microbeam". *IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP 2010)*, pp. 4190-4193.

Qin, C. and Carreira-Perpiñán, M. Á. (2009): "The geometry of the articulatory region that produces a speech sound". Invited paper, *43rd Annual Asilomar Conf. Signals, Systems, and Computers*, pp. 1742-1746.

Qin, C. and Carreira-Perpiñán, M. Á. (2009): "Adaptation of a predictive model of tongue shapes". *Interspeech 2009*, pp. 772-775.

Qin, C., Carreira-Perpiñán, M. Á., Richmond, K., Wrench, A. and Renals, S. (2008): "Predicting tongue shapes from a few landmark locations". *Interspeech 2008*, pp. 2306-2309.

Qin, C. and Carreira-Perpiñán, M. Á. (2008): "Trajectory inverse kinematics by conditional density modes". *IEEE Int. Conf. on Robotics and Automation (ICRA 2008)*, pp. 1979-1986.

Qin, C. and Carreira-Perpiñán, M. Á. (2008): "Trajectory inverse kinematics by nonlinear, nongaussian tracking". *IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP 2008)*, pp. 2057-2060.

Qin, C. and Carreira-Perpiñán, M. Á. (2007): "An empirical investigation of the nonuniqueness in the acoustic-to-articulatory mapping". *Interspeech 2007*, p. 74-77. Best student paper award.

Qin, C. and Carreira-Perpiñán, M. Á. (2007): "A comparison of acoustic features for articulatory inversion". *Interspeech 2007*, p. 2469-2472.

ABSTRACT OF THE DISSERTATION

Data-driven approaches to articulatory speech processing

by

Chao QIN

Doctor of Philosophy in Electrical Engineering and Computer Science

University of California, Merced, May 2011

Professor Miguel Á. Carreira-Perpiñán, Chair

Can you hear the shape of your mouth? Can this be done without any intrusive or invasive recording device? Imagine you have collected a set of simultaneous audio and mouth movement recordings for a human subject. Is it possible to recover movements of the vocal tract shape of the subject from arbitrary but normal speech acoustics? This problem is formally known as acoustic-to-articulatory mapping or articulatory inversion. A successful solution to this long standing and unsolved problem can facilitate useful applications in vocal tract visualization, language learning, speech therapy, and improve state-of-art recognition, speech synthesis, speech coding. It is known as the signature problem in articulatory speech processing, which aims to integrate explicitly human speech production knowledge into acoustics based speech technologies.

The main challenges in articulatory speech processing are attributed to (1) ambiguity or nonuniqueness in articulatory inversion, i.e., multiple vocal tract shapes could produce a very similar sound, which makes the inverse mapping one-to-many; (2) speaker variability, e.g., the size and shape of the vocal tract vary among subjects; (3) missing data in articulatory data measurements.

In this thesis, I address these challenges in a data-driven approach by using real speech production data. Firstly, I will present an empirical study of nonuniqueness in articulatory inversion using normal speech production data and show the nonuniqueness may not be very critical. I will show that a machine learning algorithm that addresses the nonuniqueness directly is useful for articulatory inversion for American English rhotic consonant /ɹ/. Secondly, I will discuss reconstruction of the entire tongue shape by predictive models and several strategies of adapting such subject-dependent models automatically. I will show it is possible to reconstruct the entire midsagittal tongue contour realistically from the locations of just 3 or 4 points on it with submillimetric accuracy. Furthermore, given only a few full tongue contour or even just portions of them from the new subject, the adapted model can maintain the submillimetric accuracy and is nearly as good as retraining from scratch using abundant data collected for the new subject. These successful adaptations using few data avoids time-consuming recording, segmenting and labeling thousands of ultrasound or MRI images. Thirdly, I will extend a machine learning algorithm to reconstruct missing articulatory channels from the present ones. Finally, I will apply our algorithms developed for articulatory inversion to the related problem, inverse kinematics, in robotics and graphics and show some promising results.

Chapter 1

Introduction

1.1 Why articulatory speech processing?

Speech is the most natural, convenient, and ubiquitous way of communications. This motivates the study of automatic speech recognition (ASR) that decodes spoken words into text by computer to facilitate interaction between human and computer. In mid-70s, Itakura, Rabiner and Levinson proposed the basic ideas of applying statistical pattern recognition techniques to ASR. Frederick Jelinek and his colleagues in IBM were the first to apply Hidden Markov Model (HMM) to ASR in early-80s, which quickly became popular. By 90s, HMM-based ASR started to dominate the field (See [68] for a historical review). Today, most ASR systems are based on the statistical framework and results developed in the 80s though significant improvements (e.g. discriminative training, pronunciation model, decoding algorithms, etc.) were achieved over the past two decades thanks to emerging computing resources and data. However, the state-of-the-art ASR is still at least an order of magnitude worse than human listeners, especially when speech is corrupted by noises or free style conversion is allowed, i.e., spontaneous speech.

Mainstream speech technology based on HMM have solely used features derived from the acoustic signal. This is because acoustics encode all linguistic information and is very easy to collect. However, the acoustic signal has important limitations, e.g. sensitivity to noises, high variability, etc. In fact, it is not the only possible representation for speech. Articulatory representation offers an alternative. Loosely speaking, articulatory features refer to states of articulators, e.g. tongue, lips, jaw, velum, in the human vocal tract system. There are many different ways to represent articulatory features. In this thesis, we focus on real, dynamic measurements of articulators that characterize the shape of the mouth (or the vocal tract). Other representations include vocal tract variables, articulatory gestures, hand-crafted articulatory features, each has its own strength and limitations (cf. [72] for detailed comparisons).

Articulatory features are inherently related to acoustics. Informally speaking, they offer alternative

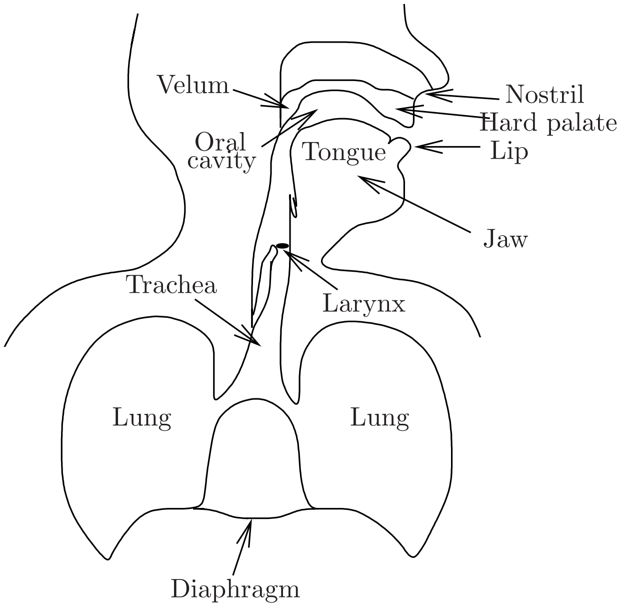


Figure 1.1: Human speech production system.

perspective to view speech. To understand why, we need to describe human speech production, i.e., the process to produce speech using various articulatory organs (see fig. 1.1). Basically, speech production is a physiological process that generates speech by the motion of loosely synchronized articulators. Take speech production of a vowel as an example. First, air is expelled from the lung and its pressure triggers the vibrating mode of the elastic vocal folds. As a result, the air is chopped up into a sequence of puffs, which are further modulated by the transfer function determined by the shape of vocal tract and result in a quasi-periodic sound wave. Finally, the acoustic wave radiates from the lips and are picked up by human ears. Therefore, acoustics are causally related to articulatory features. More crucially, the physical nature of human speech production endows articulatory representation with several attractive properties:

1. **Articulatory features are useful to classify sounds.** First, in articulatory phonetics [78], vowels can be classified by height, backness of the tongue and degree of lip rounding and consonants by state of vocal cords, place and manner of articulation, and velum height. Second, according to acoustic theory of speech production [44], vocal tract shape determines directly the spectral shape of a sound while the resulting acoustics is often blended with source information encoding speaker characteristics. In principle, articulatory features provide more discriminative power for phoneme recognition. [1]

---

[1] Note one of the biggest challenges in acoustics feature extraction is how to separate the spectral shape from the source

2. **Articulatory features are less variable and more robust.** They reflect directly the physical states of the vocal tract (VT). Acoustics can be viewed as the outcome of simulating a highly sophisticated physical process consisting of several stages. Modification in any stage could significantly alter the original signal, e.g. noise. In contrast, articulatory features do not suffer from any external interference and behave stably especially for critical articulation [104].

3. **Articulatory movements are slower and smoother due to physical constraints.** This is an appealing property for statistical modeling and low bit-rate speech coding. For example, they are quite suitable for modeling with the hidden Markov model, which assumes a quasi-stationary stochastic process.

4. **Articulatory features are more interpretable for speech characteristics.** For example, movement of the second formant from high to low is easily interpreted in terms of articulatory domain (i.e., tongue moving from the front of the mouth to the back) but more complicated to explain in acoustic domain, e.g. changes in line spectral frequencies [89].

Despite its appealing property, a fundamental question is how to obtain articulatory information in the first place. Unlike acoustics which can be easily collected by a high-quality microphone, acquiring articulatory states is much more expensive because most of articulators are not visible, i.e., located inside the mouth. Today, various articulatory imaging techniques enable us to record the movement of these articulators, e.g. X-ray microbeam (XRMB), electromagnetic articulography (EMA), ultrasound, and magnetic resonance imaging (MRI) (see detailed comparisons of these techniques in section 1.3). However, in practical scenarios, these acquisition equipments can not be assumed available. A pragmatic solution is to estimate articulatory states from audio. This is the well-known problem of acoustic-to-articulatory mapping, a.k.a. articulatory inversion. It is one of the most important problems in articulatory speech processing, i.e., analyzing, modeling, processing speech in the articulatory domain. There are also some active researches that use visual modality to constraint the mapping, i.e., audiovisual-to-articulatory inversion [92]. In this thesis, we assume input from audio only.

Another important question is that can we replace acoustics entirely with articulatory features? The answer is generally no, at least with articulatory features currently in use. It is true that articulatory gestures are very important for speech perception. Human can recognize articulatory gestures through ”inverting” acoustics in some unknown way. However, the practical bottleneck is how to obtain the accurate and reliable articulatory gestures from measured acoustics. Reliable estimation of articulatory gesture from acoustics is still an open question and the state-of-the-art articulatory inversion system still can not deliver the satisfactory inversion results. On the other hand, acoustics based speech technology,

---

information, which is assumed to contain little linguistic information but speaker characteristics.

e.g. speech recognizers and synthesizers, perform far better than articulatory counterparts as of current technology, even using the ground-truth articulatory measurements. Besides, acquisition of acoustic data is far easier and cheaper than that of articulatory data. However, articulatory information has been proven to be useful to complement acoustics, e.g. integrating articulatory features into existing acoustic HMM based systems has been proven to reduce recognition errors [166, 73, 72] and increase flexibility of text-to-speech synthesis [89]. Beyond improving conventional speech technology (e.g. coding, recognition, synthesis), articulatory features can find very useful applications in speech animation (e.g. visualizing vocal tract, animated talking head), language learning (e.g. pronunciation training) and speech therapy (e.g. diagnosis of dysarthria through estimation of the velum height without invasive and obtrusive articulatory measurements).

1.2 Why data-driven approaches?

As shown in the previous section, speech production is a highly sophisticated physical process. It is very difficult to develop a comprehensive and accurate mathematical model for such a process that is characterized by nonlinear dynamics and noise effects. Traditionally, there were many attempts to modeling speech production mathematically, e.g. *Webster's horn equation*, a second-order linear partial differential equation describing sound propagation in the human vocal tract. However, such model is often built upon many assumptions, which make it very coarse and difficult to simulate complex speech production events, e.g. turbulence effects in producing fricative sounds.

Today, it is possible to collect large-scale, simultaneous recordings of articulatory and audio data with many advanced articulatory imaging techniques. With these real speech production data, data-driven techniques become convenient and inexpensive tools to **explore statistical relationship among articulators and between acoustics and articulation**.

On the other hand, difficult problems in articulatory speech processing can serve as test benches to examine existing algorithms (e.g. for sequential data learning, missing data reconstruction, dimensionality reduction) and inspire new ones.

1.3 Articulatory data acquisition

Data-driven articulatory speech processing requires measured articulatory and acoustic data. Most of articulatory acquisition techniques available today can be used along with audio acquisition except MRI (see fig. 1.2–1.3). Representative techniques are briefly reviewed here (see [143, 126, 21] for detailed reviews).

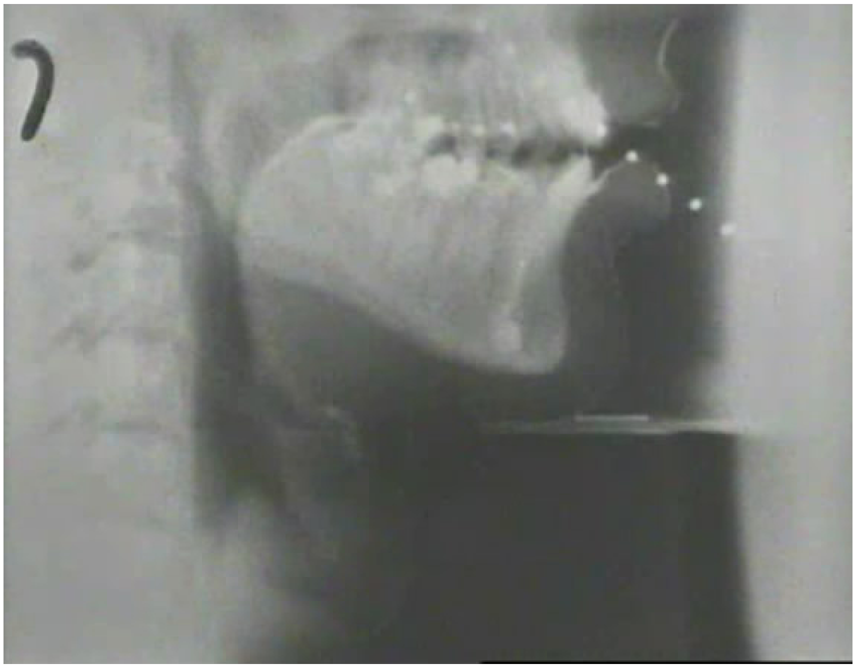


Figure 1.2: An X-ray image of midsagittal vocal tract shapes. Note that X-ray does not image soft tissues very well. X-ray computed tomography is a better alternative because it consists of multiple X-ray scans.

X-ray. It images dynamic movements of the entire 2D (and possibly 3D) vocal tract shape with high spatial and temporal resolutions (see fig. 1.2). Thus, it seems to be an “ideal” technique to capture vocal tract shaping. However, it is rarely used in speech research because long-term exposure to X-ray radiation is highly dangerous.

X-ray microbeam. It radiates the subject with a narrow, high-energy X-ray beam to track 2D positions of gold receiver coils attached to key articulators, e.g. lips, tongue, jaw. It can offer high temporal resolution up to 100 Hz that is sufficient to capture rapid movements of articulators during sound production. However, it only provides spatially sparse information — a few fleshpoint measurements about shape of upper vocal tract. Also, XRMB equipment is expensive, which limits its popularity.

Electromagnetic articulography. It tracks locations of EMA sensors attached to articulators with electromagnetic field and thus has no risk of radiation poison. It is often used in speech production studies and is a principal alternative to XRMB. 3D EMA systems become available recently to measure lateral information of the VT.

Ultrasound. It can image the entire 2D tongue shape very well except the anterior tongue tip with high temporal resolution (up to 100 Hz). It is noninvasive and safe. But it does not image any other important articulator, e.g. lips. jaw, velum, and parts of the tongue can be hard to see. Besides, ultrasound images typically contain speckle noises, making it difficult to extract tongue contours from them.

Magnetic resonance imaging. It is also a safe and noninvasive technique. It can provide high spatial solution to capture the entire VT including pharyngeal structures by a powerful magnetic field. However, its temporal resolution is very poor and thus it is unable to record dynamics of speech articulation though real-time MRI is under development [21] and might be a promising technique. The biggest problem of MRI is the difficulty to record with audio simultaneously due to scanner noises. Also, MRI recording is very expensive.

Unfortunately, no state-of-the-art articulatory imaging technique provides “ideal” articulatory measurements (high temporal and spatial resolution, entire 3D VT shape, cheap, noninvasive, unobtrusive, and easy to use). This poses significant challenges in data-driven articulatory modeling and processing.

1.4 Challenges and our efforts

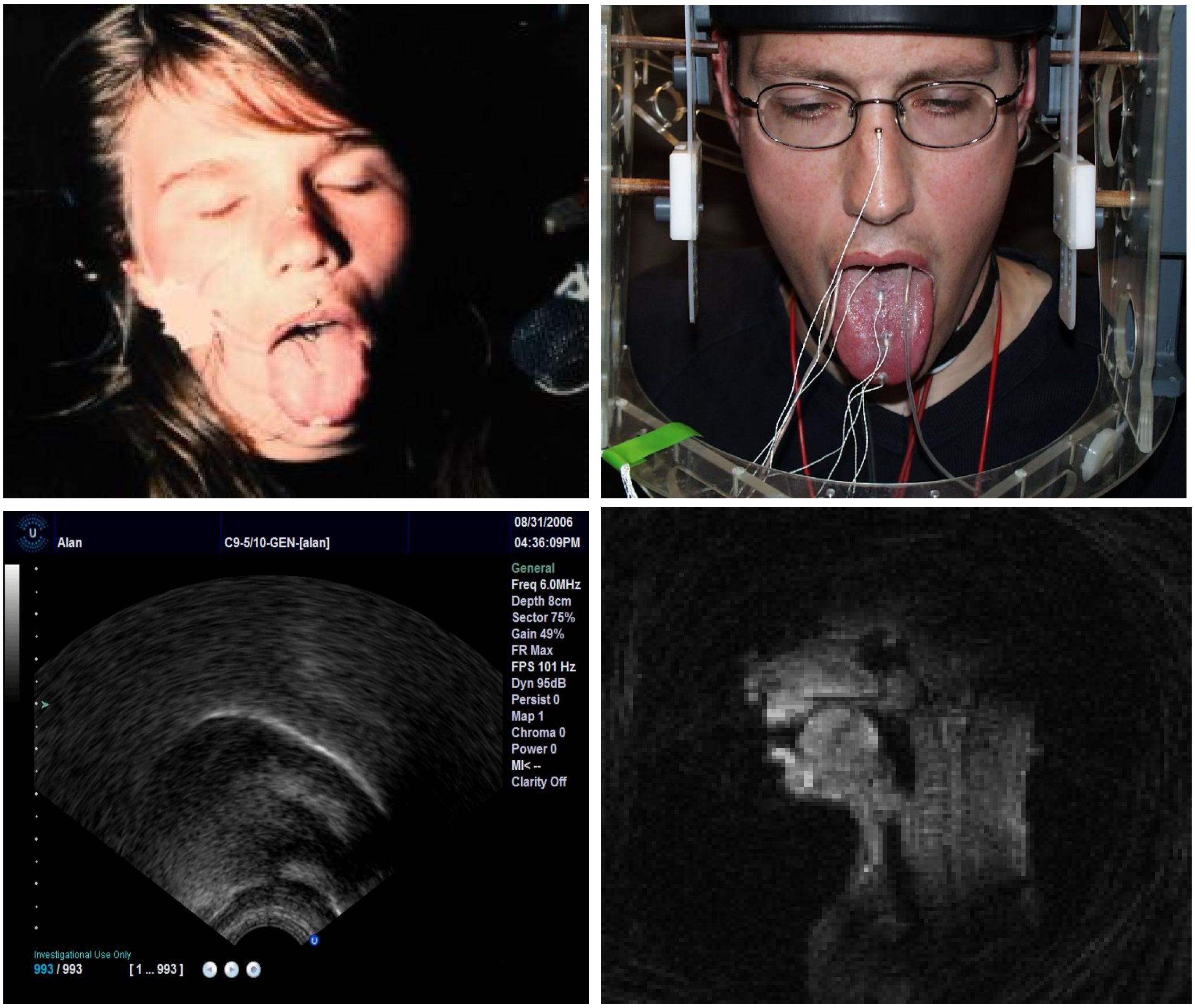
In this section, we list several major challenges in articulatory speech processing in particular for articulatory inversion and our efforts to address them.

Challenges.

- **Nonuniqueness** Conventionally, this refers to nonuniqueness of the instantaneous acoustic-to-articulatory mapping. It means certain sounds may be produced in very different ways in the articulatory space. This is the signature problem of articulatory inversion that plagues many inversion algorithms.

- **Variability** It is a general challenge to all speech research. It consists of intra-speaker and inter-speaker variability. Sources of variability include age, accent, health, social status, etc. Even the same utterance repeated by the same speaker in the same recording session could result in different acoustics and possibly different vocal tract shapes. In articulatory speech processing, many works carry out experiments on data from a single speaker and thus disregard inter-speaker variability. Therefore, an important question is how to generalize the mapping estimated from one speaker to another one. This is the problem of speaker adaptation.

- **Missing data** There are two types of missing data in articulatory measurements. The first missing data refers to incomplete measurements of the VT shape by many articulatory acquisition techniques (e.g. EMA/XRMB only provides sparse measurements of upper vocal tract shapes;



General
Freq 6.0MHz
Depth 8cm
Sector 75%
Gain 49%
FR Max
FPS 101 Hz
Dyn 95dB
Persist 0
Map 1
Chroma 0
Power 0
MI< --
Clarity Off

Figure 1.3: State-of-the-art articulatory acquisition technique. *Top left*: X-Ray microbeam (picture from University of Wisconsin). *Top right*: EMA (picture from University of Edinburgh); *Bottom left*: Ultrasound. *Bottom right*: MRI (picture from SAIL, University of Southern California).

ultrasound only images the outline of the tongue but not any other articulator). The second missing data is attributed to articulatory tracking errors (e.g. in EMA/XRMB, some pellets could fall off or displace, or could be mistracked over long-term recordings; in ultrasound, apex of tongue are often poorly imaged and so are other parts of tongue occasionally).

- **Coarticulation** It refers to the assimilation of the place of articulation of one speech sound to that of an adjacent speech sound. This increases significantly the variability of speech articulation because physical realizations of vocal tract shapes and acoustics for one sound depend on its phonetic context whose combination could be enormous. On the other hand, this may be a blessing to the algorithm design because this introduces another constraint — temporal smoothness for us to leverage, which hopefully can resolve or alleviate the instantaneous nonuniqueness.

- **High-dimensionality** Measurements of the vocal tract shape in various key articulators result in high-dimensional articulatory vectors in particular when incorporating 3D information. However, normal speech articulation is highly constrained and many articulators (e.g. lower lip and jaw) are physically coupled. Consider producing the syllable /sa/. The jaw typically rotates around some axis and traces a 1D curve despite the 2D Cartesian measurements of the jaw. Therefore, speech articulation may be characterized by many fewer variables and hence ambient articulatory features can be assumed to lie in a low-dimensional speech manifold.

In summary, to date articulatory speech processing is still an open area. A lot of open research questions are yet to be answered. Research in this direction is promising because human seems to be able to solve inversion effortless (at least for sounds relevant to speech). For example, infants are able to learn their mother tongue through babbling. Or, in voice mimicry, a foreign speaker is able to mimic the voice of a reference speaker without visualizing the tongue movement of the reference speaker. Another similar example is the *Million Dollar Recovery Machine* [127]. These evidence implies good "code breaking" algorithm exists but yet to be discovered.

Our efforts. We address several challenges in this thesis: nonuniqueness, variability, missing data.

1.5 Structure of thesis

The thesis consists of four parts.

Part I (chapters 2–4) is dedicated to articulatory inversion. Chapter 2 presents empirical study on nonuniqueness. Chapter 3 compares acoustic features and parameterization for articulatory inversion. Chapter 4 presents a machine learning algorithm to articulatory inversion.

Part II (chapters 5–6) is dedicated to tongue reconstruction. Chapter 5 describes predictive modeling of tongue shapes. Chapter 6 presents algorithms to adapt the predictive models of tongue shapes.

Part III (chapter 7) is dedicated to missing data reconstruction. Chapter 7 describes our probabilistic approaches to fill in XRMB mistracks for articulatory databases

Appendix C is dedicated to trajectory inverse kinematics in robot arms, a problem related to articulatory inversion.

Chapter 2

Empirical study of nonuniqueness in articulatory inversion

2.1 Introduction

The acoustic-to-articulatory mapping problem, or articulatory inversion, consists of recovering the sequence of vocal tract shapes that produce a given acoustic speech signal [132]. The forward articulatory-to-acoustic mapping transforms a vocal tract (VT) shape or the articulatory configuration into an acoustic signal. It can be implemented computationally in various ways based on the acoustic theory of speech production [44]. For example, the VT can be modeled as a concatenation of several straight acoustic tubes with time-varying cross-sectional area function $A(x)$ extending from the glottis ($x = 0$) to the lips ($x = L$), where L is the length of the VT. Sound wave propagation in the VT can be described by *Webster's horn equation*, a second-order linear partial differential equation governing the relationship between the pressure (and volume velocity) as a function of x and a given $A(x)$. One can use this equation to compute the acoustic signal resulting from a given $A(x)$ for some glottal excitations. It can be shown that for lossless vocal tracts with fixed boundary conditions, the area functions $A(x)$ and $1/A(L - x)$ (where L is the length of the vocal tract) produce the same acoustics. Thus, in theory the forward mapping is not only highly nonlinear but also many-to-one, i.e., different VTs can produce the same acoustics (as evidenced by ventriloquists). This makes its inverse not only highly nonlinear but also one-to-many or multi-valued (See fig. 2.1).

A solution (even partial) of this problem would have important applications in speech recognition, synthesis and coding, and language learning and speech therapy. For example, in speech coding, one

---

This chapter is mainly based on references [108, 112]

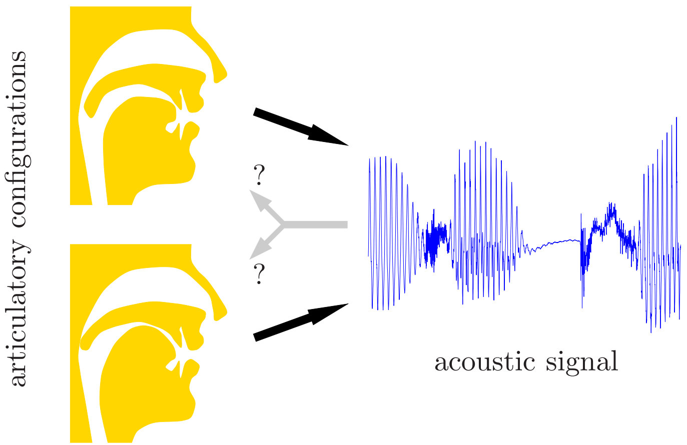


Figure 2.1: The acoustic-to-articulatory mapping problem: one VT configuration produces a unique acoustic signal, but certain acoustic signals may be produced by multiple VT configurations. (After [24])

can replace spectral parameters with slow-varying articulatory parameters [131]. In automatic speech recognition (ASR), articulatory features could be used to augment the existing acoustic features to further improve recognition performance [166, 46]. In articulatory speech synthesis [129], the articulatory model converts a set of VT shape controlling parameters to a VT area function, which is then used to generate a voicing signal by filtering the source signal as often seen in a voice mimic system [45]. Finally, knowledge of speech sound articulations can provide visual aids for language learning and therapy [13].

Articulatory inversion is a long-standing problem and remains unsolved, in particular for unvoiced sounds. The problem is hard because the inverse mapping is strongly nonlinear and multi-valued, i.e., multiple vocal tract (VT) shapes can produce the same acoustics—unlike the forward, or articulatory-to-acoustic, mapping, which though nonlinear is uni-valued. Many techniques have been proposed over the years to solve it (see [132, 24, 122] and also chapter 4 for reviews), some using mapping approximators (e.g. neural nets), while others try to address the multi-valued nature of the mapping directly (e.g. codebooks or density models).

In this chapter, we perform a systematic study of the nonuniqueness of the inverse mapping using real human speech production data (for a single speaker). The traditional evidence for the nonuniqueness of the inverse mapping can be summarized as follows.

Articulatory compensation. One example is the bite-block experiment, where a speaker is capable of producing acoustic signals perceptually close to the intended sounds even when the jaw is fixed in an unnatural position by a bite-block [88]. Another example are ventriloquists, who can produce intelligible speech without moving their lips.

Theoretical or modeling studies. Use of *Webster's horn equation* shows nonuniqueness. Computational studies based on articulatory models also indicate nonuniqueness, e.g. Atal et al. [9, 132] manipulated an articulatory model to demonstrate how very different VT shapes could produce acoustics with nearly identical values of the first three formants. Note that tongue shapes produced by the articulatory model Atal used in [9] could be unrealistic.

The American English /ɹ/. Speakers of rhotic dialects of American English use many different articulatory configurations for the approximant consonant /ɹ/ (the 'r' as in 'perk' or 'rod'), which are all acoustically characterized by an extremely low frequency of the third formant (often close to that of the second formant). These configurations differ most in the palatal constriction and have traditionally been divided into contrasting categories of retroflex (tongue tip raised, tongue dorsum lowered) and bunched (tongue dorsum raised, tongue tip lowered), though there really seems to exist a continuum between them [155, 43]. These different configurations occur both within and across speakers: some speakers may use one type of configuration exclusively while others switch between two or three different types in different phonetic contexts and according to prosodic variations.

However, it is known that human articulation is highly constrained, particularly for the tongue. For example, as shown by Stone and others [144, 145, 146], the shapes that the tongue typically adopt during speech production are sufficiently constrained.

There also exist some **arguments for the uniqueness of the inverse mapping**. For example, Ladeforged et al. [79] stated *"Whenever a speaker produced the vowel /iː/ as in 'heed', the body of the tongue always raised up towards the hard palate. Whenever anyone produces the vowel /ɑː/ as in 'father' the tongue is always low and somewhat retracted. In fact, with a few exceptions, the articulatory positions used by different speakers producing a given speech sound are always very similar to one another."* Another indirect evidence is that many different computational approaches perform similarly. For example, most techniques for articulatory inversion yields similar reconstruction errors (around 2 mm), whether they use methods for uni-valued mappings (e.g. neural networks) or for multi-valued mappings (e.g. codebooks or density models).

Therefore, the question of to what extent does nonuniqueness occur in normal human speech remains unclear, because (1) articulatory models are only crude approximations to the geometry of the human VT, (2) the vocal acrobatics of ventriloquists and bite-block experiments are not representative of the typical behavior of the human VT, and (3) the /ɹ/ remains an isolated, exceptional case. The only way to answer the question is by studying large amounts of human articulatory data, which have become available in recent years (Wisconsin X-ray microbeam database [154], Multichannel Articulatory database [161]— though none of these two databases provide information about the lower VT).

We approach the problem by analyzing simultaneously recorded articulatory and acoustic data using statistical machine learning techniques. The goal of this chapter is to report such a study. It is worth noting that our work on nonuniqueness has been extended by others [98, 3].

2.2 Nonuniqueness in the full 16D articulatory space

2.2.1 Methodological setup

Wisconsin X-ray microbeam database (XRMB) provides pairs $(\mathbf{x}, \mathbf{y})$ for articulatory vectors $\mathbf{x}$ and acoustic vectors $\mathbf{y}$ (see appendix A for details about this database). Our basic idea is to fix one acoustic vector $\mathbf{y}_n$ and search the database for articulatory vectors $\{\mathbf{x}_m\}$ that approximately map to $\mathbf{y}_n$ (*inversion*); we call this the inverse set for $\mathbf{y}_n$. Then, a *clustering* algorithm determines whether the point cloud $\{\mathbf{x}_m\}$ is unimodal or not. Repeating this for every acoustic vector $\mathbf{y}$ allows exploration of the nonuniqueness of the inverse mapping for a full range of sounds. Let us consider each step in detail.

Inversion. This requires a distance between acoustic vectors $\mathbf{y}$. We use linear predictive coding (LPC) coefficients because they are closely related to the vocal tract spectral envelope, which allows direct visualization of spectral differences and formant structures; and because they have been shown to perform well with articulatory inversion techniques (see details about acoustic features in chapter 3 and appendix B). As for acoustic distance $d(\mathbf{y}, \mathbf{y}')$ we use the Itakura distance [118] [1], which emphasizes the role of the formants and is a reasonable approximation to a perceptual distance (see appendix B for more details about Itakura distance). In practice, we found that use of Itakura distance reliably returns a set of ambiguity frames that sound very similar to the target frame. One alternative to Itakura distance is the Itakura-Saito distance [65], which is a measure of the perceptual difference between an original spectrum and an approximation of that spectrum. This distance is empirically found to be highly correlated with human perception [14]. In our experiments, we found the inverse sets returned by using these two distances differed insignificantly. Therefore, we fixed the Itakura distance in this chapter. The VT shape representation is simpler: each component of the articulatory vector $\mathbf{x}$ is the horizontal or vertical coordinate (in mm) of a pellet. Next, we fix a reference distance r for which we consider two acoustic vectors to be roughly the same sound. $r = 0.1$ guarantees that the inverse set returned belong to the same sound category. This is validated by observing that feature vectors for consecutive frames in an utterance are a distance 0.06–0.1 apart. On the other hand, very big r would return the entire dataset as its inverses and cluster structures would be obscured. In practice, we found $r = 0.1$ would lead to an excessively small inverse set for majority of acoustic frames and hence

---

[1] We used the routine `distitar.m` in the Matlab package `VOICEBOX` to calculate the Itakura spectral distances between two sets of LPC coefficients.

make the subsequent clustering unreliable. In practice, we found $0.2 \leq r \geq 0.4$ is valid by trial and error. In this section, we chose $r = 0.4$. While the specific set $\{\mathbf{x}_m\}$ of articulatory vectors returned for an acoustic vector $\mathbf{y}$ depends on r, the output of the clustering algorithm does not as long as r is neither too small (too few $\mathbf{x}$ are returned, erasing clusters) nor too large (too many $\mathbf{x}$ are returned, obscuring clusters). An approximate inversion of this type is unavoidable given the discrete nature of the data. The search strategy goes back to the use of articulatory codebooks. Roweis [127] proposed a similar search strategy using a Mahalanobis distance but returning a fixed number K of neighbors, which is much harder to estimate since K depends on the particular acoustic vector $\mathbf{y}$. Fixing K nearest neighbors for each acoustic frame is very likely to introduce some outliers (frames with different phone identity to the reference one) and biases in clustering analysis. The assumption of our method is to collect all frames produced under various contexts and in the different utterances but correspond to the same or similar sound to the reference frame. Also, it is hard to set K, i.e., K that is appropriate for vowels may be inappropriate for consonants. On the other hand, ϵ-ball is more suitable since it uses the direct acoustic distance measure (Itakura distance) to differentiate frames. Frames with the same or similar phone identity will causally have small distance and otherwise large distance (for different phone). In summary, the inversion for an acoustic vector $\mathbf{y}$ returns a set $\{\mathbf{x}_m : d(\mathbf{y}_m, \mathbf{y}) \leq r\}$.

Clustering. The large size of the database requires an automatic procedure to determine whether the resulting point cloud $\{\mathbf{x}_m\}$ is multimodal. The complex shape of the cloud and the fact that we do not know how many clusters it contains prevents us from using parametric models such as a Gaussian mixture. Instead, we use a nonparametric kernel density estimate with Gaussian kernel and bandwidth σ, i.e., we define a density $p(\mathbf{x}) \propto \sum_m G\left(\frac{\mathbf{x}-\mathbf{x}_m}{\sigma}\right)$. Inspection of many point clouds suggested using $\sigma = 6$ mm (again, this number is chosen by trail and error to ensure the approximated density is neither too spiky or too smooth. We found empirically that the results were not sensitive to small variations of σ). Finally, we find the modes of $p(\mathbf{x})$ using a *mean-shift algorithm* [23], which iterates a hill-climbing algorithm initialized at every $\mathbf{x}_m$ and collects all the resulting, distinct modes (see section 4.3 for details on derivation and iteration). Strongly unimodal clouds $\{\mathbf{x}_m\}$ yield a single mode while clustered or elongated clouds yield several modes.

2.2.2 Experimental results

Dataset. We use the XRMB [154], which records, simultaneously with the acoustic wave, the positions of 8 pellets in the midsagittal plane of the VT (see fig. 2.2), sampled at 146 Hz, for various types of speech (isolated words, prose, etc.). The XRMB measurement error for the pellets is 0.7 mm (see appendix A for more details about this database). We use LPC of order 20 to obtain an accurate formant structure (for order 12, F3 is smoothed out in e.g. /ɹ/). The acoustic feature vectors use a window and step size to yield 146 Hz as well; we removed silent frames by endpoint detection, a procedure to detect and remove silence from an utterance. There are many algorithms developed for endpoint detection, e.g. zero-crossing rate [118]. In this chapter, we adopt a simple but effective strategy that discriminates silent and nonsilent speech frames by vector quantization on frame energy (see appendix B for more details). More accurate decisions can be obtained through the time-aligned phone level transcription for an utterance if available. We use a single speaker (`jw11`, male with Midland dialect of American English, 90 utterances including isolated words, prose passages, etc.), resulting in a dataset of 45 000+ vectors $(\mathbf{x}, \mathbf{y})$ with $\mathbf{x} \in \mathbb{R}^{16}$ and $\mathbf{y} \in \mathbb{R}^{20}$.

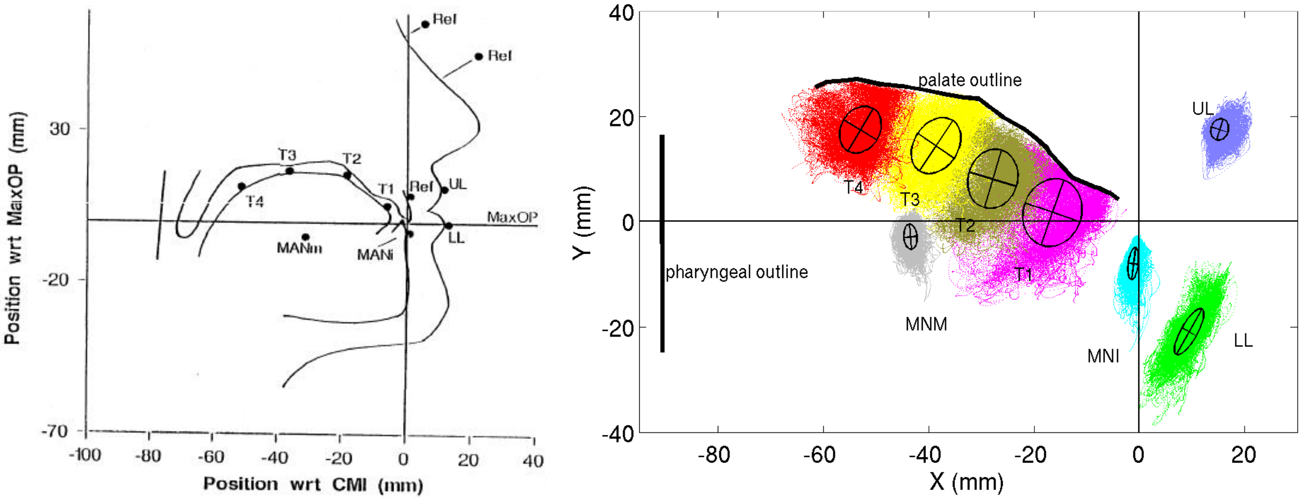


Figure 2.2: Wisconsin X-ray microbeam database. *Left*: pellet locations in the XRMB. *Right*: plot of the entire dataset for speaker `jw11`; each pellet's data uses a different color and shows a contour line of one standard deviation centered at its mean.

Fig. 2.3 shows the result of inverting an acoustic vector $\mathbf{y}$ corresponding to the sound /θ/. The cloud $\{\mathbf{y}_m\}$ of acoustic vectors at distance $\leq r = 0.4$ is strongly unimodal but the corresponding cloud of articulatory vectors $\{\mathbf{x}_m\}$ is clearly multimodal, and our mode-finding algorithm detects this (modes in green). The LPC spectral envelope confirms the acoustic similarity of the cloud and the VT representation shows the corresponding VT shapes to be significantly different. Thus, widely different VT shapes produce approximately the same acoustic sound /θ/, indicating nonuniqueness of the instantaneous inverse mapping in this particular case. It is known that use of neighboring acoustic frames may help to reduce nonuniqueness. We verifie this by running same experiments with dynamic features (Δ, Δ^2) [2].

---

[2] Use of dynamic features is a common and useful practice in speech processing. Basically, the dynamic features can be computed by a linear regression on the static features. More details can be found in appendix B and [118].

These dynamic features did not separate the acoustic cloud (if they did, this might have disambiguated the VT shape). We did not find significant difference to using static features.

Fig. 2.4 shows several other sounds for which we find multimodality (/ɹ/, /l/, /w/), again clearly showing similar spectral envelopes (with Itakura distances < 0.1) but qualitatively different VT shapes, in particular the tongue. In contrast, fig. 2.5 shows several sounds for which the mapping is unimodal (/æ/, /uː/, /y/): the VT shapes are essentially the same for a given acoustic vector (with Itakura distances up to 0.5). In each sound, the acoustic vectors shown come from different utterances and different contexts, i.e., they are not just consecutive frames in the same utterance.

While this demonstrates the use of multiple VT shapes to produce the same acoustic sound for certain sounds, how frequently does nonuniqueness occur overall? We found that around 20% of the acoustic vectors yield a multimodal cloud in articulatory space. This suggests that, while nonuniqueness does happen, by and large it is an infrequent situation.

Fig. 2.6 gives further indirect evidence for nonuniqueness. For each acoustic vector we computed the standard deviation per dimension of articulatory space (specifically, $\frac{1}{16}\sqrt{\lambda}$ where λ is the top eigenvalue of the 16×16 covariance matrix of the articulatory cloud, i.e., the principal variance) and plotted their distribution. If the articulatory clouds were strongly unimodal, we would expect a symmetric distribution, but instead the distribution is skewed with a long right tail. This indicates that, while most articulatory clouds have a standard deviation ≈ 7 mm, a small proportion has a larger std deviation (up to 25 mm).

2.3 Nonuniqueness in individual 2D articulatory spaces

In this section, we use the similar approach as in section 2.2.1 to study the nonuniqueness of individual articulators. As noted earlier, nonuniqueness of the entire VT shape does not imply nonuniqueness of each articulator. Likewise, a Gaussian mixture in XY space with modes at $(\pm 1, 0)$ only has one mode in Y space. We already noted from section 2.2 that the same sound could be produced by very different tongue shapes but with almost the same upper lip position. As before, the basic idea is to fix one acoustic vector $\mathbf{y}_n$ and search the database for its *inverse set*, i.e., all articulatory vectors $\{\mathbf{x}_m\}$ that approximately map to $\mathbf{y}_n$ (*inversion*). Then, we apply a *clustering* algorithm to determine whether the inverse set $\{\mathbf{x}_m\}$ in each articulator's 2D space (e.g. `ULx` and `ULy`) is unimodal or not, and compute shape statistics from the inverse set for each articulator. Repeating this for every acoustic vector $\mathbf{y}_n$ in the database allows an exploration of the nonuniqueness of the inverse mapping for a wide range of sounds, and a characterization of the geometry of the inverse set. Let us consider each step in detail.

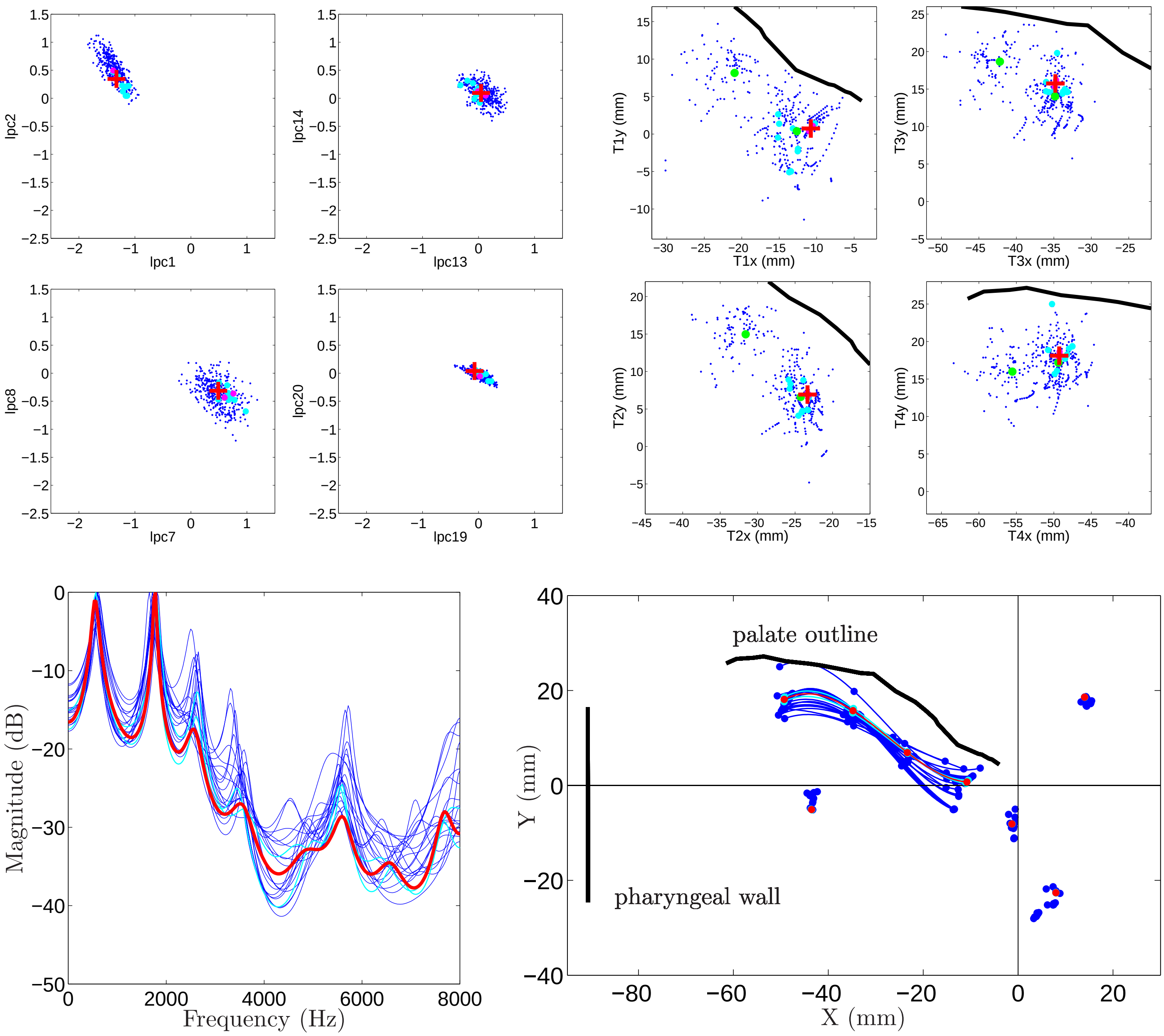


Figure 2.3: An example of nonuniqueness in articulatory space for a reference acoustic vector $\mathbf{y} = /\theta/$ (red, utterance `tp004` “things”). *Top left*: point cloud $\{\mathbf{y}_m\}$ in acoustic space with $d(\mathbf{y}_m, \mathbf{y}) \leq r$ (containing $\sim$ 400 points), clearly unimodal. For further resolution, we show points at distances r, $r/2$, $r/4$ in different colors (blue, cyan, magenta, respectively). *Top right*: corresponding point cloud $\{\mathbf{x}_m\}$ in articulatory space, clearly multimodal (modes in green). *Bottom left*: $\{\mathbf{y}_m\}$ as spectral envelopes. *Bottom right*: $\{\mathbf{x}_m\}$ as VT shapes (we fit a spline to the 4 tongue pellets for visualization). The bottom plots show a subset of the curves to avoid clutter.

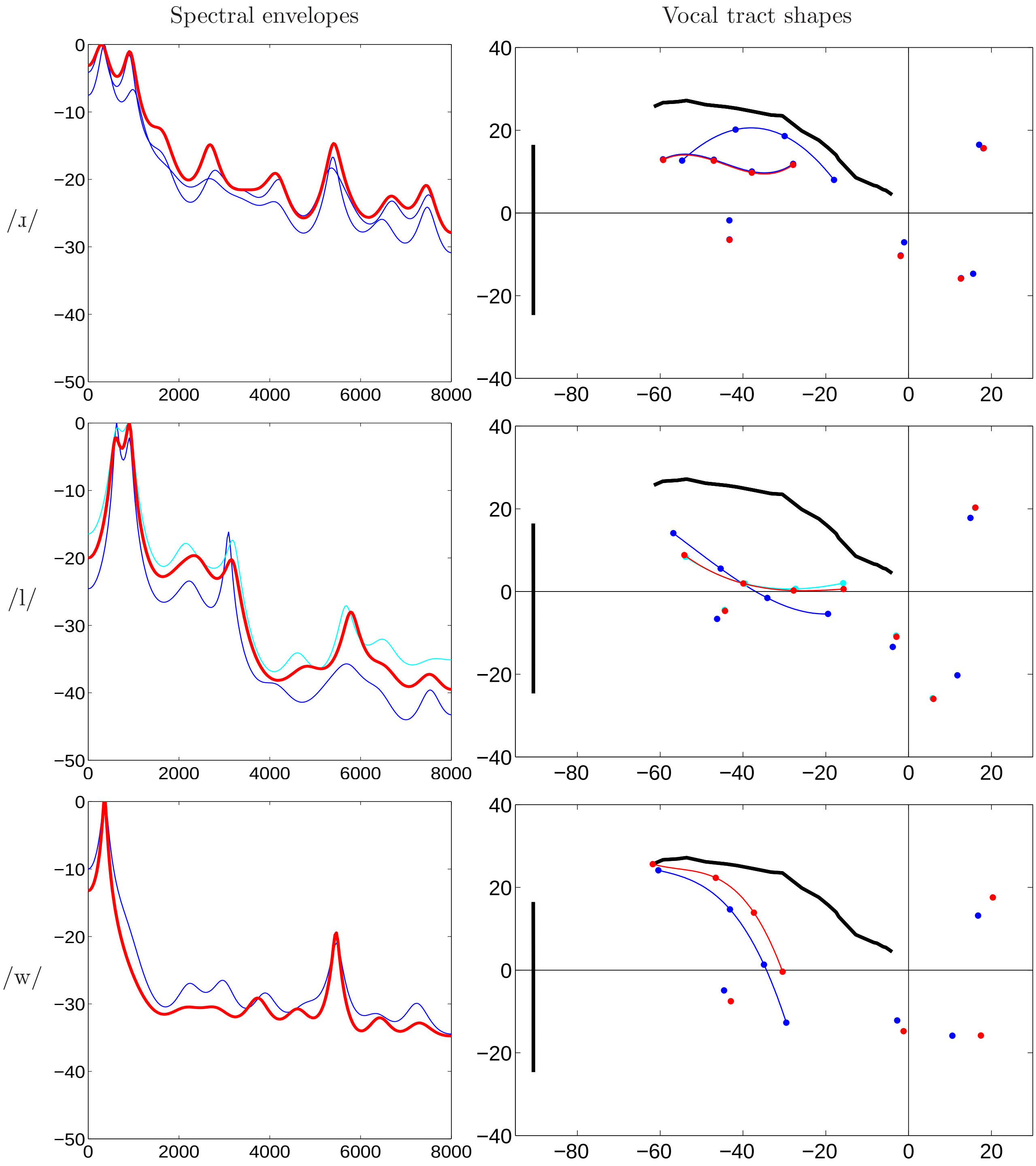


Figure 2.4: Examples of sounds showing multimodality. Spectral envelopes: magnitude (dB) vs frequency (Hz); VT shapes: X (mm) vs Y (mm). Utterances: /ɹ/, `tp009` “row”; /l/, `tp037` “long”; /w/, `tp044` “work”. For /ɹ/, the well-known retroflex and bunched tongue shapes [155, 43] are evident. We show only a very small subset of the curves and points in the cloud to avoid clutter; color scheme as in fig. 2.3.

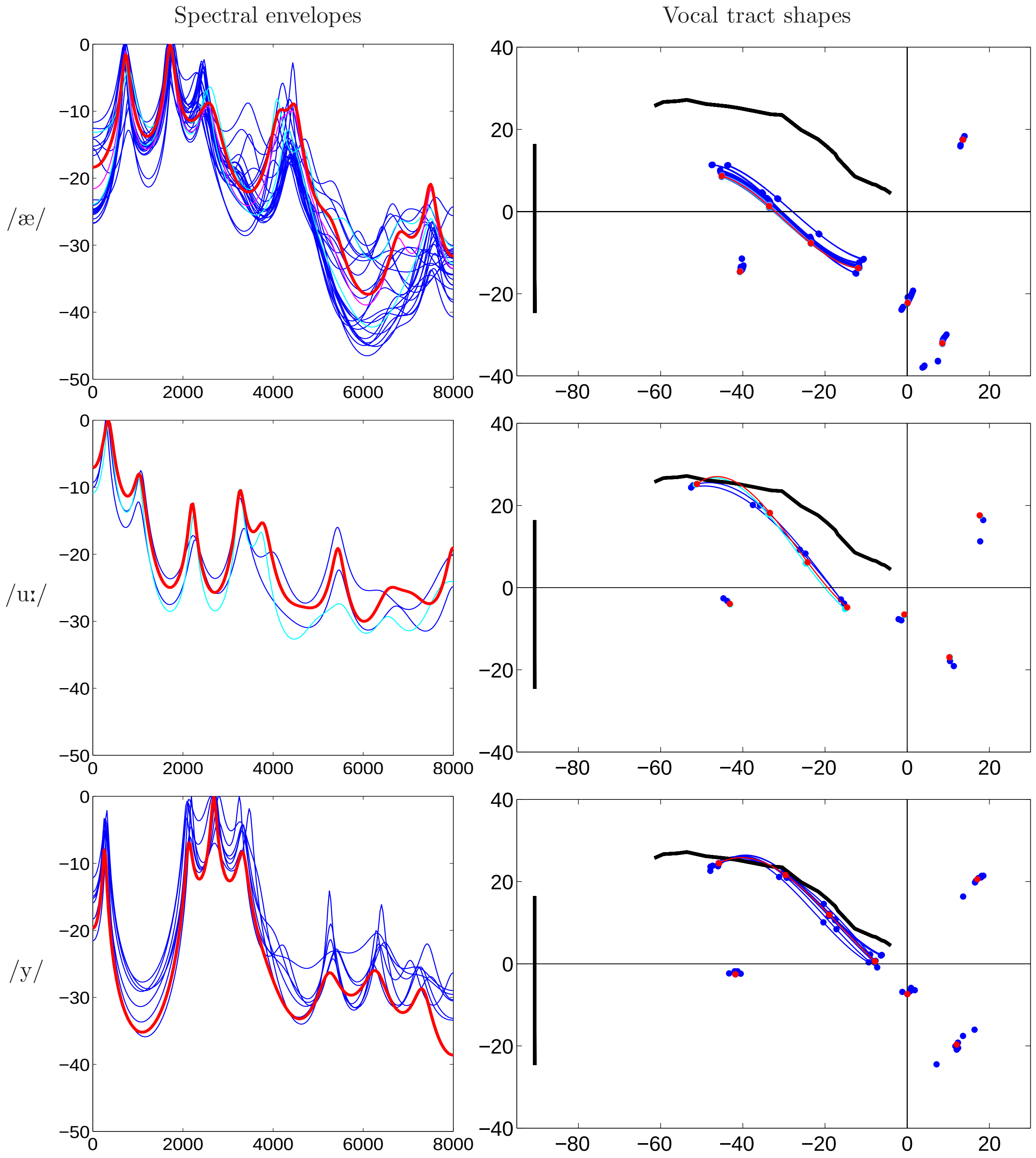


Figure 2.5: Examples of sounds showing unimodality. Utterances: /æ/, `tp001` “has”; /uː/, `tp001` “school”; /y/, `tp040` “you”.

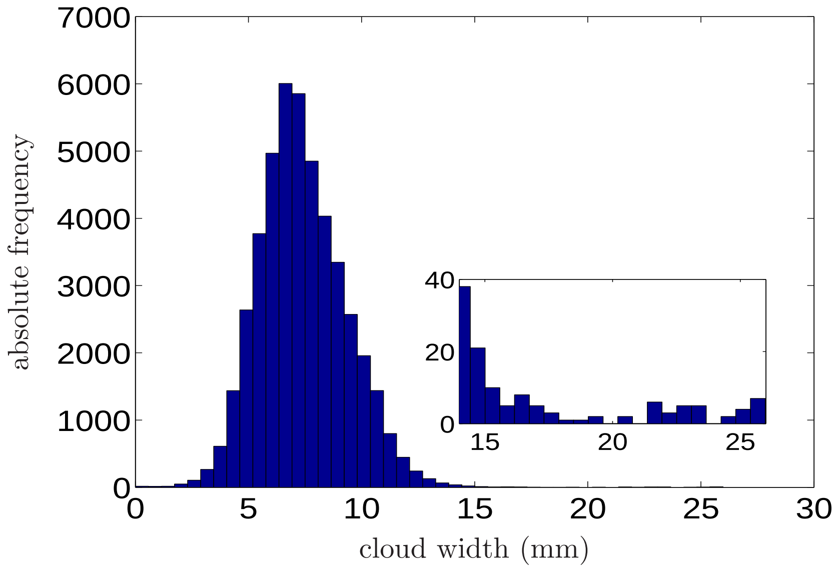


Figure 2.6: Histogram of the cloud width (standard deviation per dimension) for the inverse mapping. The inset blows up the right tail.

2.3.1 Methodological setup

Searching for multimodality in the inverse set in each articulator's space. We use same acoustic distance metric and vocal tract shape representation as in section 2.2.2. Here, we fix an acoustic reference distance $r = 0.2$ for which we consider two acoustic vectors to be roughly the same sound. Recall that in section 2.2.2 we used $r = 0.4$, but further analyses indicate that this may be a bit large and include some frames that have different phonetic identities in the inverse set, hence affecting the mode search. We find that the size of the inverse set varies considerably depending on the acoustic vector, so that searching for the K nearest vectors as in [127], instead of for those vectors with $d(\mathbf{y}, \mathbf{y}') \leq r$, results in missing true inverses or including false inverses depending on the value of K. We also discard those acoustic vectors whose inverse set contains less than 10 vectors [3] so as to obtain meaningful statistics. An approximate inversion of this type is unavoidable given the discrete nature of the data. As before, we use $\sigma = 6$ mm; we found this to be a reasonable value based on visual inspection of the 2D inverse sets.

Shape statistics of the inverse set. The number of modes gives only partial information about the geometry of the inverse set. For example, counting number of modes alone can not easily differentiate multiple separated modes and a quantized representation of a ridge of density because both cases could return comparable number of modes (see fig. 2.9). If the geometry of the inverse set is a manifold

---

[3] This number is determined empirically.

of dimension zero, it can consist of one or more tight clusters (and so one or more modes). If it has dimension one and is thus elongated, it may also consist of one or more modes along it. Therefore, we report additional shape statistics for each inverse set $\{\mathbf{x}_m\}$ (for a given acoustic vector) based on its covariance matrix. By doing this, it is possible to further classify inverse sets into three cases bases on the intrinsic dimensionality of inverse sets. Its eigenvalues $\lambda_1 \geq \lambda_2$ of the covariance matrix measure the spread of the inverse set along its principal axes. If $\lambda_2 \approx \lambda_1$, the inverse set is usually distributed as a round cloud. If both λ_1 and λ_2 are quite small, the inverse set is tightly concentrated and may be considered a zero-dimensional manifold. If $\lambda_2 \ll \lambda_1$, the inverse set has an elongated shape, perhaps corresponding to a 1D manifold. These shape statistics only depend on the acoustic reference distance r, but on no other parameters (e.g. σ, since they are not obtained from the kernel density estimate). We also explore visually the inverse sets for many acoustic vectors to try to characterize their shape.

2.3.2 Experimental results

Dataset. We use the same dataset as in section 2.2.2. Due to the fact that LPC is not effective at modeling unvoiced sounds, e.g. fricative and plosive, we eliminated those unvoiced frames (roughly 5%) from the original dataset of 45 760 vectors, making the final dataset 43 260 vectors.

Exploratory analysis of the geometry and dimensionality of the inverse set. Fig. 2.7 shows the distributions of the square-root of λ_2 vs λ_1 (in mm) for selected articulators and for the entire VT. Each point corresponds to one acoustic vector and is colored according to the number of modes of its inverse set. Points can lie roughly on the diagonal or below and to the right of it, corresponding to circular and elongated shapes, resp. Table 2.1 lists the percentage of frames with 1, 2 or more modes in each articulator space and in the entire VT space:

This shows that multimodality occurs in all articulators, i.e., for each articulator there are acoustic vectors for which multiple VT shapes exist that differ in that articulator (and possibly others). As noted in section 2.2.2, the percentage of multimodal frames in the entire VT shape is small (here, 21.9%). Multimodality is very infrequent for `UL`, `MNI`, and `MNM` (upper lip, teeth), which mostly show circular, tight inverse sets, that may be considered as 0D manifolds. Multimodality is more frequent for the tongue (`T1`–`T4`, in particular the tip, `T1`) and the lower lip (`LL`).

Fig. 2.8 shows the histogram of each square-root eigenvalue for individual articulators and for the entire VT. `T1` to `T4` and `LL` have higher variability than other articulators (`UL`, `MNI`, `MNM`). Many frames satisfy $\sqrt{\lambda_3} \leq \sqrt{\lambda_2} \leq \sqrt{\lambda_1} \leq 4$ mm and can be considered as tight inverse sets. The full-VT histogram shows that λ_1 is typically quite larger than λ_2 and λ_3, and that the latter are more comparable. Thus, many inverse sets in the 14D space are somewhat or considerably elongated in 1D; this can also be seen

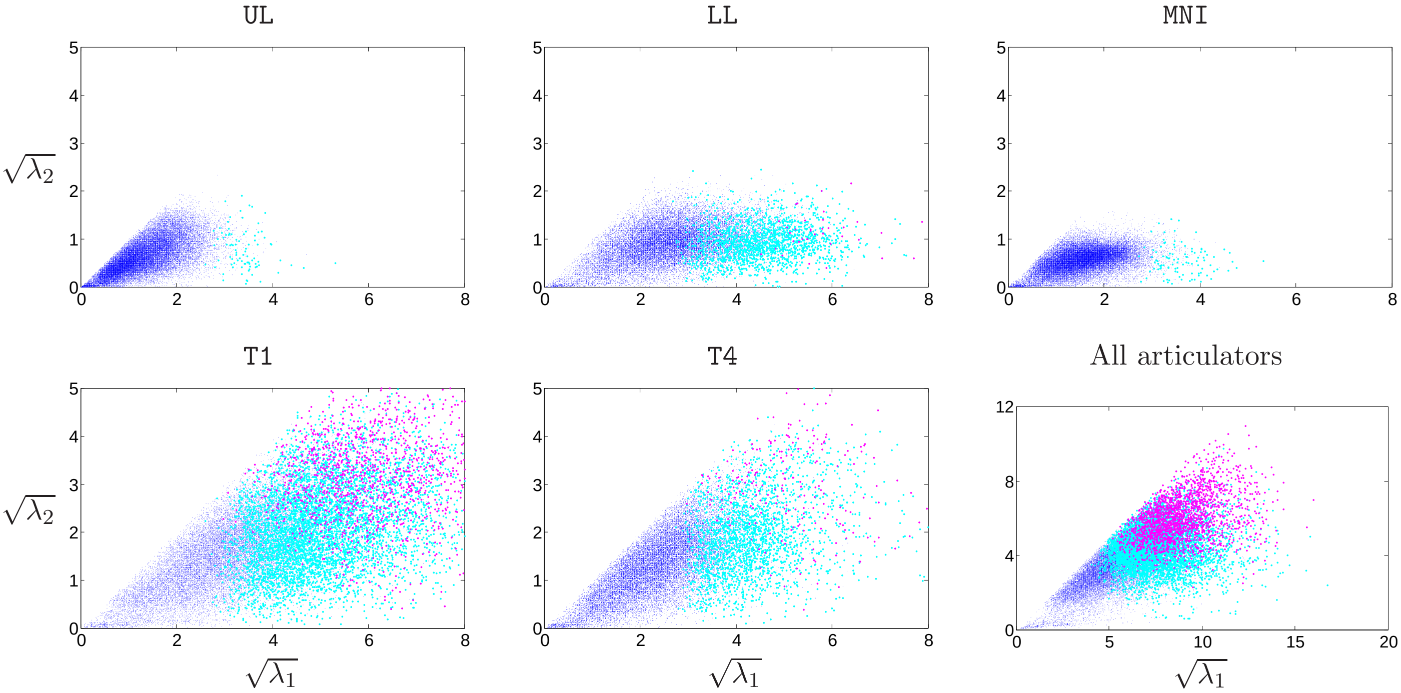


Figure 2.7: Scatterplots of the eigenvalues ($\sqrt{\lambda_2}$ vs $\sqrt{\lambda_1}$) of the covariance matrix of the inverse set for some articulators. The colors (blue, cyan, magenta) indicate the number of modes (1, 2, 3+, resp.) for each inverse set.

modes	UL	LL	MNI	MNM	T1	T2	T3	T4	all
1	99.6	93.1	99.5	99.8	78.3	88.3	89.1	91.4	78.1
2	0.4	6.7	0.5	0.2	17.6	10.2	9.6	7.9	16.7
3+	0	0.2	0	0	4.1	1.5	1.3	0.7	5.2

Table 2.1: Nonuniqueness percentages for speaker `jw11` in XRMB.

in the 2D projections in fig. 2.9. They are particularly common with the lips and teeth but also with the tongue. We suspect this may be the result of rigid 1D motion (for example, the jaw can mostly rotate around its axis, so the lower teeth track a circle) of an articulator that has little effect on the acoustics, or more generally a coordinated motion of several articulators. Finally, as reported in section 2.2, we also find clearly multimodal sets with two or more tight clusters (0D manifolds).

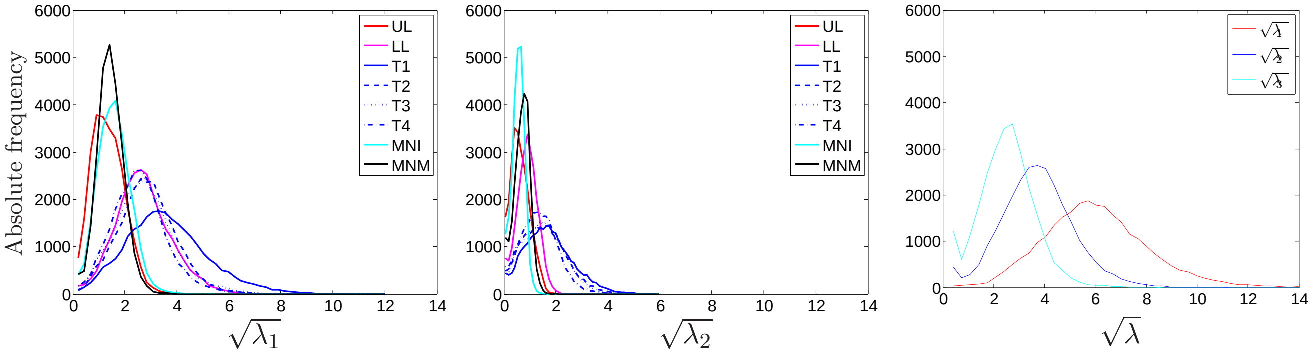


Figure 2.8: *Left, middle*: histograms of $\sqrt{\lambda_1}$ and $\sqrt{\lambda_2}$ for each articulator. *Right*: histogram of $\sqrt{\lambda_1}$, $\sqrt{\lambda_2}$, $\sqrt{\lambda_3}$ for the entire vocal tract.

Fig. 2.9 shows inverse sets (in the tongue 2D spaces) representative of the variety of shapes we find: compact unimodal (e.g. vowels), compact multimodal (e.g. "<u>th</u>e" or /ɹ/ in "<u>r</u>ow" or "<u>r</u>eal"), or elongated 1D shapes (e.g. glides /l/, /w/) corresponding to quantized representation of a ridge of density. Other sounds (e.g. /m/) seem to show very complex tongue shapes.

In summary, we find most inverse sets are compact unimodal, but among the remaining ones, we find many that are elongated in a 1D shape (possibly indicating rigid motion of a non-critical articulator) or that consist of two compact but separated clusters (distinct 0D manifolds). Beyond this, we find sets with more complex shapes too.

2.4 Discussions

Relation with critical articulators. The issue of nonuniqueness of the vocal tract shape is related but not identical to that of critical articulators [104]. The latter refers to the sensitivity of the acoustics as a function of small changes in different articulators. For a given phoneme, a critical articulator is one such that motions of it can strongly alter the sound, while motions of a non-critical articulator have a small effect on the sound. For example, the lower lip is critical for producing /b/ (since slightly opening the lips alters the acoustics strongly), but the tongue dorsum is not; this is reflected in a low variance of the lower lip's position over different realizations of /b/ sounds. In contrast, nonuniqueness (strictly

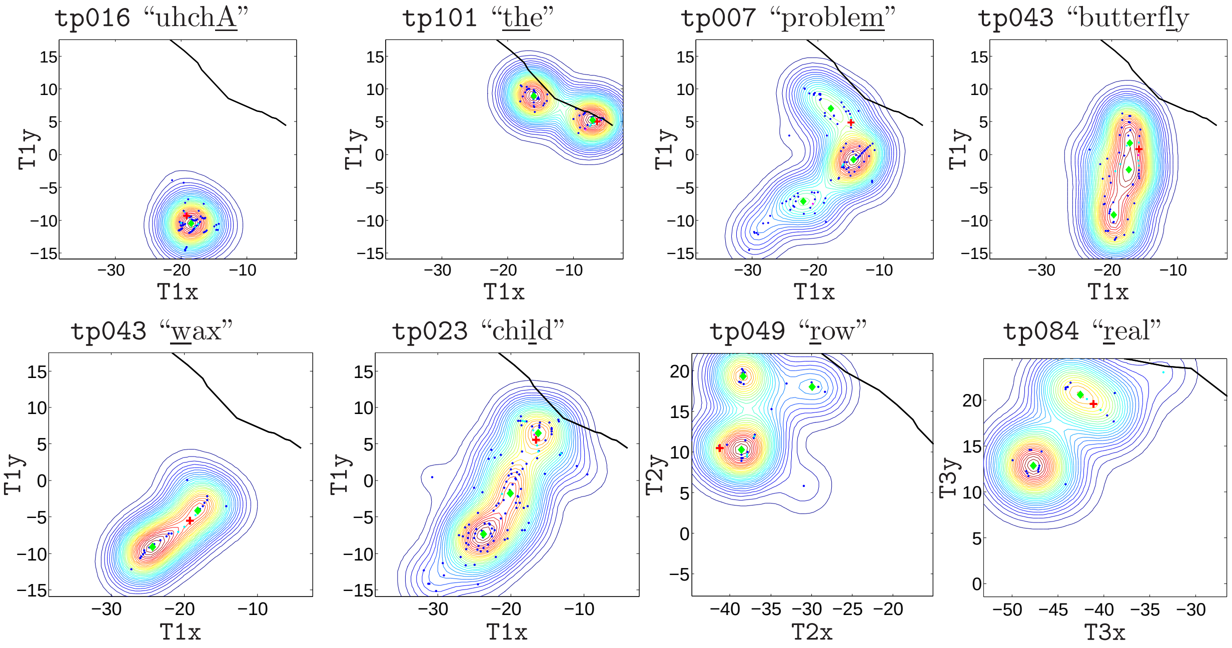


Figure 2.9: Sample plots of the inverse sets (blue dots) for a given sound $\mathbf{y}_n$ in the XRMB (speaker `jw11`) in the space of `T1` to `T3`, density contours, modes (green dot) and palate (black line). The red mark is the articulatory vector $\mathbf{x}_n$ corresponding to the sound.

defined) means entirely different vocal tract shapes produce exactly the same acoustics. Depending on how loosely we define nonuniqueness (i.e., how much acoustic variation we tolerate), a non-critical articulator may or may not result in nonuniqueness. More importantly, a critical articulator need not be uniquely determined. For example, the tongue body in /ɹ/ has a bimodal distribution of two tight clusters; thus, while small variations of the tongue can change the acoustics significantly, entirely different tongue shapes result in almost the same acoustics.

2.5 Summary

We have presented the first systematic, large-scale studies of nonuniqueness using articulatory data for normal speech from the XRMB. Empirical searching for nonuniqueness in the full articulatory space gives direct, quantitative evidence of the presence of nonuniqueness of the inverse mapping in normal human speech; but also suggest that, while some sounds are indeed produced in multiple ways, often a unique VT shape is used. This is also consistent with the fact that most techniques for articulatory inversion yield similar reconstruction errors (around 2 mm), whether they use methods for uni-valued mappings (e.g. neural nets) or for multi-valued mappings (e.g. codebooks or density models). In addition, our results of refined searching for nonuniqueness in individual articulatory spaces suggests that nonuniqueness affects all the vocal tract articulators that we considered (in particular the tongue). However, for any given acoustic sound some or even all articulators may be strongly constrained. The set of articulatory shapes that correspond to a given sound (within a small Itakura distance in acoustic space) is usually tightly concentrated around a roughly spherical region in articulator space (dimension 0). We do find many sounds that show more complex shapes: multimodality (dimension 0), very elongated in a straight or curved path (dimension 1), or even more complex.

While we think our work is the first systematic, large-scale study of nonuniqueness, it is important to note its limitations, which future studies may improve upon: (1) We considered a single speaker from the XRMB, and did not study the relevance of the acoustic context for the cases where nonuniqueness occurred. (2) The VT representation provided by the XRMB is incomplete, lacking data about the lower VT. It is thus possible that sounds that are produced with the same upper VT shape do differ in the lower VT, thus increasing the frequency of nonuniqueness. Techniques such as dynamic MRI may offer a full representation of the VT in the future. (3) In this study, we consider the nonuniqueness in instantaneous inverse mapping. It is interesting to investigate to what degree does nonuniqueness go away when conditioned on immediately previous and following acoustic vectors.

Chapter 3

Comparison of acoustic features for articulatory inversion

3.1 Introduction

Front-end parameterizations such as representations of articulatory and acoustic features are important to the success of all computational approaches. Thanks to the technologies such as XRMB and EMA, we can use as articulatory representations the measured locations of coils on different articulators such as the tongue and the lips. For acoustic representations, it is well known that acoustic features and their parameterizations such as the short-time window length are essential to ASR performance. The same stands for the inverse mapping. To our knowledge, however, there was no work about the best acoustic parameterizations for articulatory inversion. Most previous works simply chose one of the popular acoustic features used in ASR such as Mel-frequency cepstral coefficients (MFCC) [37, 53], filterbanks (FBANK) [123], linear predictive coding (LPC) coefficients in chapter 2, line spectral frequencies (LSF) [127], and perceptual linear prediction (PLP) [22]. None of them compare laterally all the performance of these acoustic parameterizations for the inversion task.

One major problem of existing acoustic features is the jaggedness of their temporal trajectories. While this issue may not matter for ASR, it is a significant problem for articulatory inversion. This is because, while the acoustic trajectory is very jagged, the articulatory one is very smooth. Thus, when applying a mapping (e.g. a neural net) to the acoustic vectors, their output will also be jagged and cause a large error with respect to the articulatory target because the mapping is typically smooth (Some articulatory inversion work post-process the inverted trajectories with a low-pass filter). Fig. 3.1 shows

---

This chapter is mainly based on the reference [107]

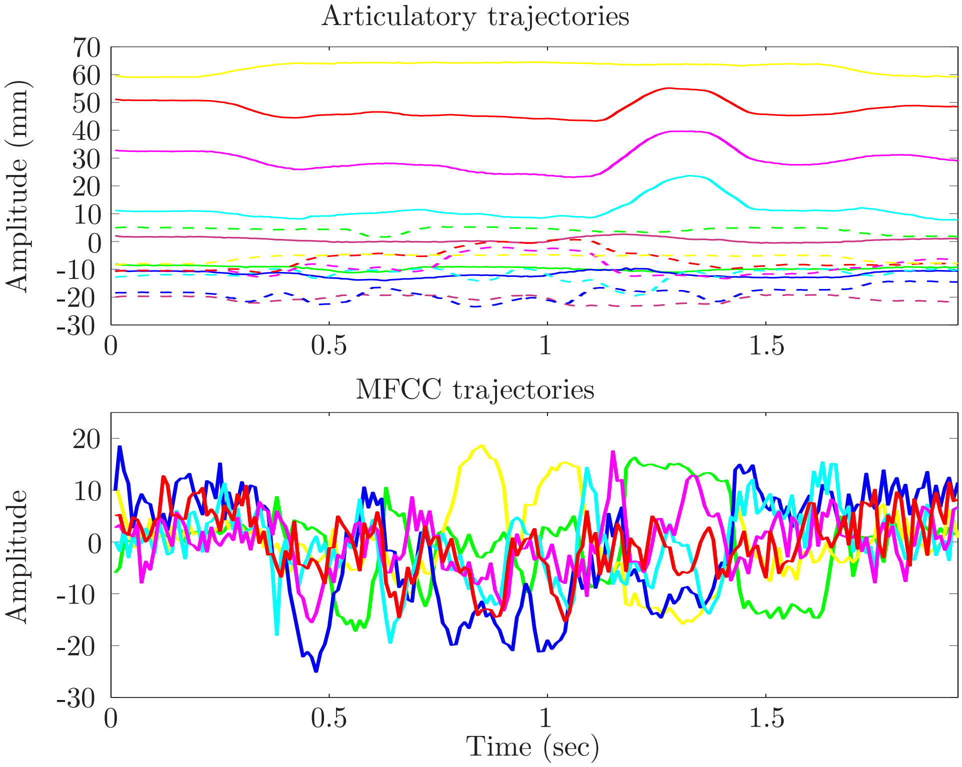


Figure 3.1: Smooth articulatory trajectories vs. jagged acoustic trajectories. All articulators move smoothly over time (*solid line*: horizontal locations of articulators; *dashed line*: vertical locations of articulators), while the acoustic trajectories are highly jagged over time (high-order acoustic features are more jagged than low-order ones). This means that a mapping applied to the acoustic sequence will yield a jagged articulatory sequence and thus a large inversion error. Note that for clarity, we only show odd order MFCC trajectories.

such an example of corresponding smooth articulatory and jagged acoustic temporal trajectories.

The goals of this study are to find out the acoustic features and parameterizations that work best for the inverse mapping, and to explore ways to alleviate the jaggedness of acoustic trajectories. Besides, we also study the effect on inversion performance of a time delay between articulatory and acoustic frames.

3.2 Methodological setup

3.2.1 Acoustic features

In this chapter, we compare most popular acoustic features that are widely used in speech/speaker recognition, speech synthesis, and speech perception. Although the acoustic waveform is a one-dimensional signal in the time domain, it is generally not a good representation for statistical pattern recognition due to its high variability. Essentially, one seeks a reliable, invariant and robust feature representation for acoustics. Conventionally, such a representation can be found in the frequency domain of the speech signal. In fact, most of acoustic features described below can be viewed as a smooth representation or an estimate of spectral envelop of the spectrogram [119, 118], a time-varying spectral representation that shows how the spectral density of a speech signal varies with time. Since the speech signal is a nonstationary process, spectrogram is often computed by the short-time Fourier transform on the windowed signal in the time domain. Given the spectrogram, the most salient acoustic features are formants, i.e., spectral peaks of the spectrogram. Formants are often used in early works for synthetic articulatory and acoustic data. Their temporal trajectories are very smooth since formants change slowly with time. However, formants only provide good characterizations for vowels and they are often difficult to estimate reliably. Since we address all types of sounds in this study, we do not include formants as one of the acoustic features to be compared, although one of the recent work on articulatory inversion uses formant frequencies and their energies as acoustic features [103]. In the following, we describe briefly several popular acoustic features used in our experiments (see appendix B for details). These acoustic features are usually categorized into [118]:

Linear predictive analysis. LPC performs the short-time spectral analysis on speech frames with an all-pole filter. It provides a good approximation to the vocal tract spectral envelope for voiced speech and achieves a reasonable source-filter separation. It is less effective for unvoiced and transient regions of speech. A variant of the LPC, LSF, is often known to be better behaved than the LPC while containing exactly the same information as the LPC [139]. Another important extension of the LPC is the LPC cepstral (LPCC) coefficients, a short-time cepstral representation. The cepstral analysis is used to decorrelate dependences among variables of the acoustic features to facilitate the HMM modeling.

Filterbank and cepstral analysis. FBANK is essentially an estimation of the envelope of the spectrum, which is obtained by applying Discrete Fourier Transform (DFT) on the windowed signal. It is motivated by the fact that human ears resolve frequencies nonlinearly across the auditory spectrum. It is therefore a popular alternative to the LPC since it provides a much more straightforward way to obtain the desired nonlinear frequency resolution. MFCC is a smoothed short-time cepstrum calculated from the log filterbank amplitude using the discrete cosine transformation. It is a robust feature containing much information about the vocal tract regardless of the source of the glottal excitation and can be used to represent all classes of speech sounds. It is the main choice for many ASR applications.

Auditory-inspired representations. Another robust variant, PLP, was originally proposed as a way of warping the spectra to minimize the differences between speakers while preserving the important speech information [58]. In addition, RASTA is a separate technique that applies a band-pass filter to the energy in each frequency subband in order to smooth over short-term noise variations and to remove any constant offset resulting from static spectral coloration in the speech channel [59].

3.2.2 Feature parameterization and post-processing

Integration of dynamic features. Instantaneous dynamic features, i.e., velocities and accelerations, are known to improve the performance of speech/speaker recognition. Dynamic features are shown to be useful to constraint and improve the inversion [149] (see reviews on articulatory inversion techniques in chapter 4).

Variable window length. The short-time window length affects the smoothness of the acoustic features. A longer window is expected to produce smoother acoustic trajectories because two consecutive frames share more data points.

Smoothing acoustic features. Smoothing the acoustic trajectories is another way to alleviate their jaggedness. It is important to keep discriminative information contained in the acoustic data as much as possible while smoothing. In this study, we perform the smoothing using a double filtering routine (implemented by `filtfilt` in Matlab Signal Processing Toolbox). First, the signal is filtered in the forward direction using an FIR filter. Second, the filtered signal is reversed and run back through the FIR filter. The cut-off frequency θ of this double filter determines the smoothing level. θ varies from 1 (no smoothing) to 0 (infinite smoothing removing all frequency components of acoustic trajectories). We use $\theta = 1, 0.5, 0.25$ respectively to denote different levels of smoothing. Fig. 3.2 shows an example of smoothing on a single MFCC temporal trajectory.

3.2.3 Inversion method

In this study, we use the neural network approach to the inverse mapping. In particular, we choose the multilayer perceptron (MLP) to map from acoustic features to articulatory ones. One appealing advantage of the neural network is that once trained, it requires much less computational efforts compared to other methods. Although this approach ignores the multi-valued nature of the inverse mapping (since a neural net can only learn a uni-valued mapping), it is still a robust method and is useful to make fair comparisons among all sets of acoustic features. We also confirmed some of our experiments with the other inversion methods.

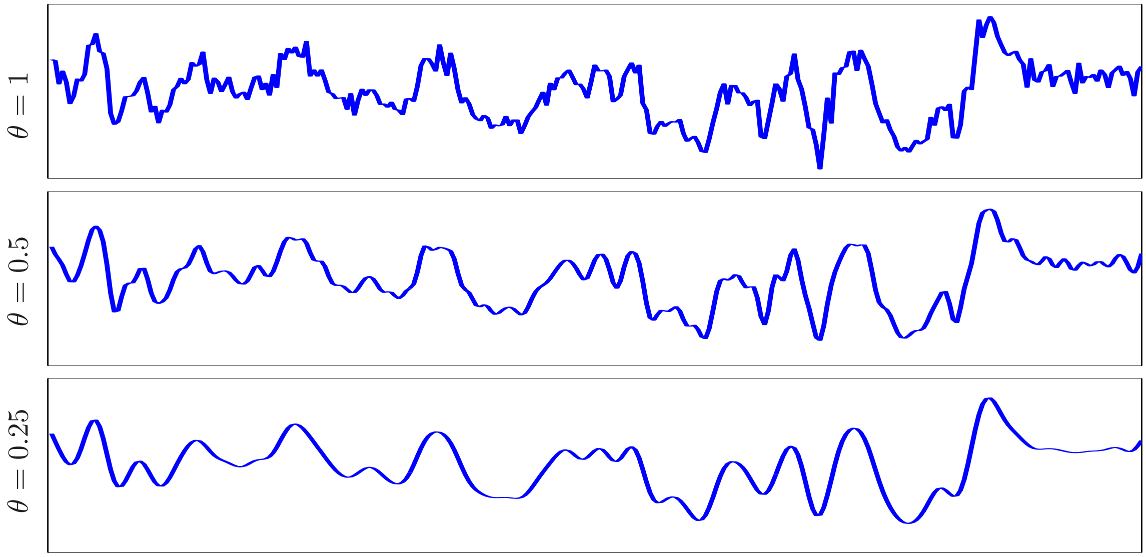


Figure 3.2: Illustration of smoothing of acoustic temporal trajectories. We set the cut-off frequency $\theta = 1, 0.5, 0.25$ respectively in the ascending order of smoothing levels.

3.3 Experiments

3.3.1 Experimental setup

Dataset We used the Multichannel Articulatory (MOCHA) database developed by Queen Margaret University [162]. Three speakers with different regional accents have been made available so far. Four data streams are recorded synchronously for each speaker: the acoustic waves (16KHz sampling rate) together with laryngograph, electropalatography (EPG) and EMA data. The EMA recorded the movements of received coils in the midsagittal plane at 500Hz. Coils (pellets) are attached to the upper and lower lips, lower incisor, tongue tip, tongue body, tongue dorsum, and velum. Each speaker records a set of 460 British TIMIT sentences designed to be phonetically diverse. We use data from the female speaker `fsew0` with a northern English accent. Fig. 3.3 shows the distribution of EMA data from this speaker. We divide the dataset into two parts. One part contains 10 000 frames (randomly selected from 366 utterances) and is used for training; the other part contains 2 000 frames (from 8 unseen utterances) and is used for testing. We fix the frame shift of the short-time Hamming window to 10ms, which is roughly the average duration of speech sounds. Articulatory trajectories are downsampled to match the acoustic data. More details about this database can be found in appendix A.

Silence removal. Exclusion of silent frames from the training set is essential to the neural network training. This is because during the silent periods, the vocal tract can in principle take any configuration.

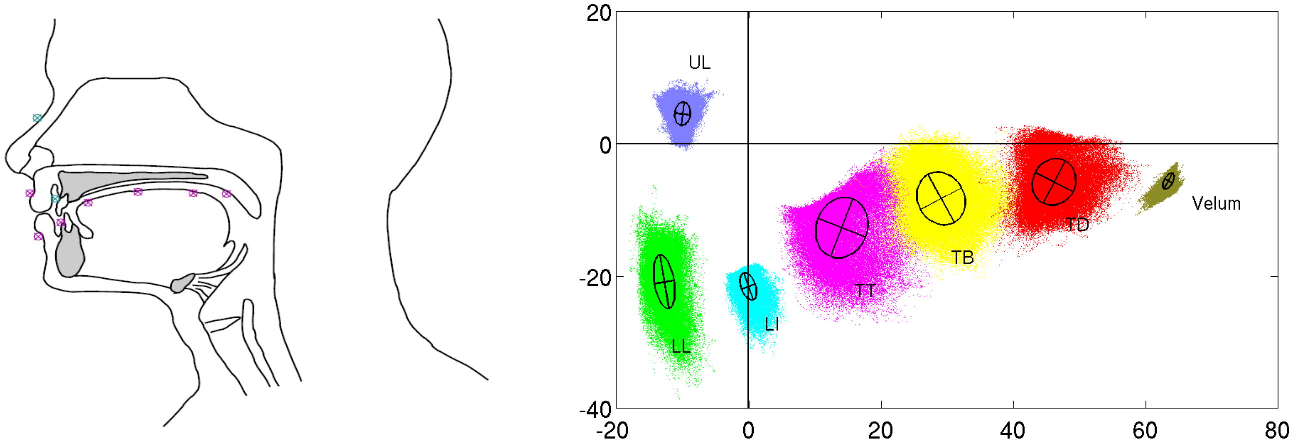


Figure 3.3: MOCHA-TIMIT articulatory database. *Left*: pellet locations in the MOCHA database. *Right*: plot of the entire dataset for speaker `fsew0`; each pellet's data uses a different color and shows a contour line of one standard deviation centered at its mean.

During training, given an acoustic feature vector corresponding to silence, the network would try to map it to a large range of possible articulatory vectors, and result in a poor mapping. We apply the frame-energy based endpoint detection to remove silence, short pauses, and transient regions of speech.

Acoustic feature sets. Combinations of following sets are compared: (1) Acoustic features: LPC, LSF, FBANK, MFCC, LPCC, PLP, RASTA-PLP. (2) Dynamic features: static features only and static plus dynamic features. (3) Hamming window length: 25 ms, 35 ms, 45 ms, 64 ms, 80 ms, and 96 ms. (4) Smoothing level: $\theta = 1, 0.5, 0.25$.

Inversion method. We used a MLP with a single layer of 55 hidden units. The MLP was trained with scaled conjugate gradients using the Netlab Toolbox for Matlab (`http://www.ncrg.aston.ac.uk/netlab`).

Performance metric. We use Root-mean-square error (RMSE) and Pearson correlation as our performance metric. Both are widely used in measuring performance of articulatory inversion. RMSE is defined as: $e_j = \sqrt{\frac{1}{N}\sum_n \|a_j^{(n)} - b_j^{(n)}\|^2}$, where, j is articulator index, N is number of speech frames in the utterance, a and b are true and reconstructed trajectories respectively. Pearson correlation quantifies for a given articulator the similarity in shape between two trajectories regardless of magnitude, i.e., whether they rise and fall in synchrony: $c_j = \frac{\sum_n (a_j^{(n)} - \bar{a}_j)(b_j^{(n)} - \bar{b}_j)}{\sqrt{\sum_n (a_j^{(n)} - \bar{a}_j)^2 \sum_n (b_j^{(n)} - \bar{b}_j)^2}}$ where $\bar{a}$ and $\bar{b}$ are the means of the two sequences.

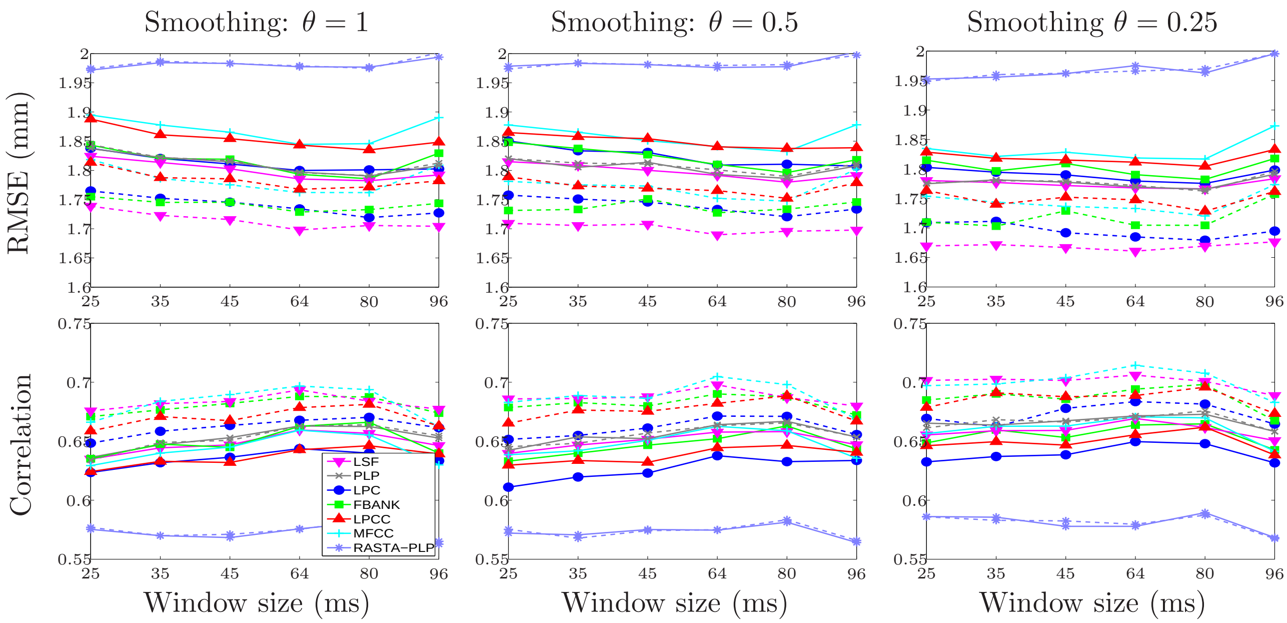


Figure 3.4: Performance comparison of different acoustic features (dashed/solid lines: with/without dynamic features, respectively), window size and smoothing level by MLP for `fsew0`. Errorbars for standard errors of the means are not shown here to avoid clutters. They are roughly 10% of the means.

3.3.2 Comparison of acoustic features and parameterizations

Fig. 3.4 compares the performance of different acoustic feature sets. The results for RMSE and correlation are essentially equivalent. Integrating dynamic features consistently outperforms using only static features, except for PLP and RASTA-PLP. The smoother the acoustic features, the better the performance. Relatively long windows (best at 64 to 80 ms) also improve the inversion accuracy. When acoustic features are smooth enough ($\theta = 0.25$), long windows make little difference. MFCC and LPCC perform similarly. The performance of acoustic features roughly degrades in the decreasing order of closeness to the vocal tract. Among all, LSF and PLP are the best acoustic features for articulatory inversion and RASTA-PLP is (significantly) the worst. However, the overall difference is small, with all methods achieving an RMSE of 1.65 to 2.00 mm and a correlation of 0.56 to 0.71.

It is interesting to note that our acoustic feature ranking in terms of correlation (i.e., MFCC performs the best, which is consistent with the finding in [53] and is followed by LSF, LPCC etc) match well with the finding in [53], although the latter uses mutual information to measure the statistical dependency between the acoustic feature and the articulatory position.

We repeated the experiment with two other methods (results now shown): using a Gaussian-mixture regression, and using a method based on representing multi-valued mappings with conditional density modes, i.e., `cmode`, as described in chapter 4. The results were largely the same, except that adding dynamic features did not improve as much.

3.3.3 Effect of time delay

Up to now, we have attempted to recover the articulatory frame at time t from the acoustic frame obtained by multiplying the acoustic wave by a short-time window centered at time t. However, it is possible that a different alignment of the articulatory and acoustic streams could result in a smaller RMSE; for example, the time spent by a wave traveling along the vocal tract may introduce a delay in the resulting acoustic waveform. While this delay would likely depend on the particular sound produced, for simplicity we consider a global delay. We conduct another empirical study to find out the best time delay, using as acoustic parameterization the best one described above (LSF with dynamic features, 64 ms window, smoothing $\theta = 0.25$). Fig. 3.5 shows that the best performance is achieved when the short-time window is centered 15 ms after the articulatory frame. This finding is consistent with the pilot study by Hodgen et al. [61] in which a 14.4 ms time delay was found to be optimal. But again, while consistent over different experiments, the improvement over the baseline (i.e., no delay) is very small.

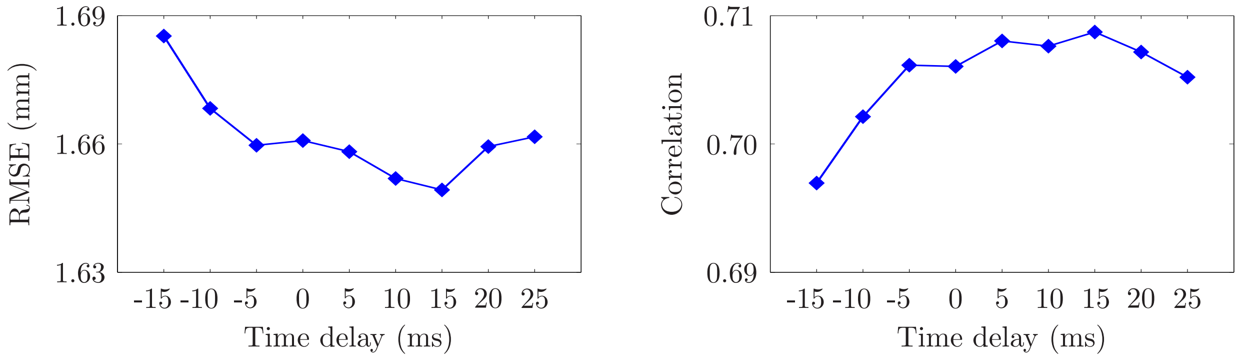


Figure 3.5: Effect of time delay between an acoustic frame and an articulatory frame. Errorbars for standard errors of the means are not shown here to avoid clutters. They are roughly 10% of the means.

3.4 Summary

We have presented the first empirical study on the best acoustic parameterization for articulatory inversion. Using a simple inversion method (a neural network) as a baseline, we have compared different acoustic features, with varying lengths of the short-time Hamming window and different levels of smoothing of the acoustic temporal trajectories. Relatively large windows and smoothing were shown to alleviate the jaggedness of acoustic features. We found that best results are generally obtained with acoustic features that are more closely related to the vocal tract (in particular LSF, although PLP performed just as well), using dynamic features, 64 to 80 ms short-time window, double-filtering smoothing of cut-off frequency $\theta = 0.25$, and a 15 ms time delay between articulatory and acoustic frames. However, the improvement over other combinations of features or smoothing was very small (around 0.3 mm, to yield an RMSE of around 1.65 mm). These results also held when using the other inversion methods (different from the neural net).

It is important to note the limitations of our study. We used only one speaker from one database (MOCHA), and specific choices of e.g. the smoothing method. However, while other choices may alter the RMSE in absolute terms, we do not expect major changes to the relative ranking of the features (which, from fig. 3.4, is quite consistent over different conditions).

Chapter 4

Articulatory inversion of American English /ɹ/ by conditional density modes

4.1 Introduction

Although many different approaches have been proposed for articulatory inversion, the problem is still not generally solved. In this chapter, we focus on two aspects of the problem: we propose an algorithm that directly addresses the multi-valued nature of the inverse mapping, and we demonstrate it with real articulatory data for a well-known case that clearly displays multiple articulations, the American English /ɹ/ (see [43] and references therein). As already noted in chapter 2, the American English /ɹ/ can adopt two canonical but contrasting tongue shapes: (1) retroflex (tongue tip raised, tongue dorsum lowered). (2) bunched (tongue tip lowered, tongue dorsum raised).

Much of the early evidence on multiple articulations comes from synthetic models (e.g. [9]) or unnatural conditions (e.g. bite-block experiments), with less actual evidence from normal, real speech. In chapter 2 our experiments based on the large-scale articulatory database (XRMB), show that only a small portion (around 15%) of all acoustic frames correspond to more than one vocal tract shape. One such case is the American English /ɹ/. The performance of inversion algorithms has been often evaluated with synthetic datasets [9], or with real datasets for which nonuniqueness is of lesser or no importance (for example, with vowels). Other work (e.g. [125]) has used a large, real articulatory dataset (MOCHA) but the performance with the small proportion of it that shows nonuniqueness was

---

This chapter is mainly based on the reference [113]

not quantified. This chapter focuses exclusively on utterance subsequences containing /ɹ/, thus with a large proportion of nonuniqueness.

This chapter is organized as follows. Section 4.2 briefly reviews major computational approaches to articulatory inversion. Section 4.3 describes our inversion based on conditional density modes. Experimental results are presented in section 4.4. In this chapter, we notate articulatory and acoustic variables as $\mathbf{x}$ and $\mathbf{y}$, respectively.

4.2 Computational approaches to articulatory inversion

Articulatory inversion is a long standing problem in speech research. Various approaches to articulatory inversion have been proposed over the past three decades. However, a satisfactory algorithm towards all sounds is still missing. Many computational approaches can be classified in different ways, e.g. whether they address nonuniqueness, whether they consider the temporal correlation among sequential data. Schroeter and Sondhi [132] reviewed much of earlier work. Our review follows [24] and features some new methods.

Methods based on forward modeling. Work by Flanagan et al. [45] and Levinson and Schmidt [84] use *analysis-by-synthesis* (ABS), a close-loop optimization to retrieve the VT shape $\mathbf{x}_t$ such that its synthetic acoustics closely matches with the real one $\mathbf{y}_t$, i.e., $\mathbf{x}_t^* = \arg\min_{\mathbf{x}_t} \|\mathbf{y}_t - \mathbf{f}(\mathbf{x}_t)\|^2$, where the forward or articulatory-to-acoustic mapping $\mathbf{f}$ is often provided by an articulatory synthesizer. ABS is also used in the *voice mimic* system [121, 27] that allows automatic adjustment of parameters in articulatory speech synthesizer to mimic arbitrary human speech. It is worth noting that the success of the solution crucially depends on initial guess conventionally obtained from an articulatory codebook (see below). In this forward modeling framework, the forward mapping $\mathbf{f}$ is the key. Over years, various forward mappings have been proposed, e.g. area-function based acoustic tube approximation of vocal tract [165, 147], data-driven approximation by a feed-forward neural networks [71], a mixture of linear models [127, 60], self-organizing preduction models [19], etc. However, these approximations are either crude (e.g. synthetic speech [165]) or inaccurate (due to the hard nature of direct mapping from a smooth feature to a noisy one) or very specific to particular sound categories ([165]). An accurate, reliable, robust forward mapping or speech production model is yet to be developed for general sounds. Availability of such model could be a break-through in speech technology and be expected to improve the state-of-the-art methods and enable new machine learning approaches. Another limitation of this framework is attributed to its local nature: ABS only provides local optimal solutions, which are likely to be invalid for the multi-valued inverse mapping. The local nature of ABS approaches depends crucially on the initial solution, which is hard to obtain. Even if a good local optimum can be obtained by a good initial guess from a comprehensive articulatory codebook, it can only track one solution branch and miss other valid solutions irreversibly. In these regards, ABS methods are much less preferable than other methods that address the multi-value mapping.

Methods based on learning instantaneous inverse mapping. Work by Shirai and Kobayashi [135] and Papcun et al. [104] use a *neural network* (NN), e.g. multi-layer perceptron (MLP) to approximate the inverse or acoustic-to-articulatory mapping $\mathbf{f}^{-1}$. Papcun et al. [104] also propose to use the acoustic context window rather than a single acoustic frame as input to the MLP, which is found to be effective in resolving mapping ambiguity and improving inversion. However, use of NN ignores the nonuniqueness because NN can not model the multi-value $\mathbf{f}^{-1}$. Moreover, it ignores the temporal correlation in the sequential data since it treats highly correlated neighboring frames in the sequence independently during training and testing phases [1]. Despite these modeling deficits, NN performs robustly and produces competitive inversion. Work by Richmond [125] and Toda et al. [149] derive $\mathbf{x}$ from the mean of a conditional density $p(\mathbf{x}|\mathbf{y})$. However, this *conditional mean* approach should perform similarly to the NN [16] in that, asymptotically on the number of parameters, they estimate the true conditional mean of the data. Later, Richmond [124] and Toda et al. [149] improve the *conditional mean* by applying the speech parameter generation algorithm [150] proposes for the HMM-based speech synthesis such that the correlation among sequential data is enforced during inversion. This is done by explicitly exploiting the constraints between dynamic and static articulatory features and inverting the entire articulatory trajectory instead. Intuitively speaking, this amounts to using a trajectory-wide acoustic context window to recover the instantaneous inverse. Another interpretation comes from the dual view of the speech parameter generation algorithm as a smooth kernel that post-processes the trajectory [96]. Therefore, by construction the inverted articulatory trajectory often appears smoother and results in smaller RMSE and higher correlation.

Methods based on codebook (obtained from a *synthetic* model) lookup and dynamic programming. This method deals with the multi-value $\mathbf{f}^{-1}$ directly. *Codebook lookup* is another popular framework for inversion. A typical articulatory codebook consists of $100,000$ entries of paired $(\mathbf{x}, \mathbf{y})$. Designing a good codebook often trades off spatial resolution and time complexity. The codebook is typically constructed by root-shape interpolation [102]. As a result, the codebook often contains significant number of unrealistic VT shapes even after post-pruning. Work by Atal et al. [9] and Larar et al. [81] search for the single VT shape $\mathbf{x}_t$ that matches the closest to the given acoustic key $\mathbf{y}_t$ in the articulatory codebook. However, the resulting sequence of the VT shapes is often highly irregular because only the constraint of acoustic match is considered. Schroeter and Sondhi [130, 132] improve this by

---

[1] In contrast, sequential data learning methods [17] such as hidden Markov model and linear dynamic system explicitly explore these temporal correlation.

relaxing the acoustic match to allow multiple candidate VT shapes and enforcing temporal continuity in articulatory space. They applied dynamic programming (DP) to find an optimal path through the candidate space. The time complexity of DP is quadratic on the averaged number of candidates. In the *hypercube codebook* approach proposed by Ouni and Laprie [102], a codebook is constructed as a hierarchy of hypercubes by densely sampling in articulatory space such that within each hypercube the articulatory-to-acoustic mapping $\mathbf{f}$ can be approximately by a linear mapping. This method assumes $N = 7$ articulatory parameters, $M = 3$ acoustic parameters (i.e., F1–F3) and a $\mathbf{f}$ given by Maeda's articulatory model [91]. Starting with a bounding-box hypercube for the whole articulatory domain, they recursively subdivide each hypercube into 2^N hypercubes until certain criterion holds. Within each hypercube, $\mathbf{f}$ is approximated linearly by computing $\mathbf{f}$ exactly at its center and approximating the Jacobian of $\mathbf{f}$ with finite differences. They also apply DP to retrieve the articulatory trajectory in the candidate VT space given by a particular inverse solution and a nullspace at each time. To improve the rough trajectory estimation due to the coarse nature of the codebook and the discontinuities across hypercubes, they apply the a secondary smoothing step as [80], i.e., variationally minimizing an objective that penalizes deviations from neutral positions. However, this study did not experiment on real articulatory data neither compare with other codebook methods. It remains interested in these quantitative comparisons. In *ensemble neural networks* [120], the articulatory codebook is split into many articulatory-acoustic clusters such that $\mathbf{f}^{-1}$ can be well fitted by a MLP within each cluster. DP is used to select the MLPs that ensure smoothly varying VT shapes while maintaining a good acoustic match. However, splitting clusters can be very difficult and requires lots of heuristics. In *conditional modes*, originally proposed for a more general problem of missing data reconstruction [22], DP search is performed in the mode space where the conditional density modes derived from $p(\mathbf{x}|\mathbf{y}_t)$ represent multiple inverses. $p(\mathbf{x}|\mathbf{y}_t)$ can be computed from a joint density $p(\mathbf{x}, \mathbf{y}_t)$ by e.g. a Gaussian mixture or estimated directly by e.g. Mixture of Experts [66] using the real data. One may view the joint density $p(\mathbf{x}, \mathbf{y}_t)$ is an articulatory codebook with infinite number of codebook entries. The advantages of *conditional modes* over *Codebook* are: faster, less storage, no quantization error.

This is the approach we are going to apply to articulatory inversion in this chapter.

Methods based on sequential data learning. These methods are based on sequential data learning, e.g. hidden Markov models, linear dynamic system [17]. They aim to leverage temporal correlation among the sequential data. Work by Hiroya and Honda [60] proposed an HMM-based speech inversion method. During training, an articulatory HMM is built for each phoneme [2] and within each state of the

---

[2] A phoneme is linguistically distinct speech sound. Formally, a phoneme is the smallest segmental unit of sound employed to form meaningful contrasts between utterances [8]. For American English, there are about 42 phonemes including vowels, diphthongs, semivowels and consonants [119].

HMM a linear mapping is estimated to approximate $\mathbf{f}$. During inversion, an HMM state sequence is derived from acoustic signal via Viterbi decoding and then a smoothed articulatory trajectory is generated through the speech parameter generation algorithm [150]. Zhang and Renals [164] propose a similar method where the parameter generation algorithm was an integral part of trajectory HMM framework to facilitate deriving smoothed output mean trajectories. There exist some methods considering the inversion from a *tracking* perspective where $\mathbf{x}$ and $\mathbf{y}$ are treated as the hidden state and the observations respectively. In this framework, two components are necessary: a dynamical model that characterizes the relationship between $\mathbf{x}_t$ and $\mathbf{x}_{t-1}$; a measurement model that depicts the nonlinear $\mathbf{f}$. Work by Shirai and Honda [134] apply extended Kalman filter (EKF) to estimate the articulatory trajectories from speech. Their dynamical model is a second-order linear function that captures the elasticity and damping of the articulators and their observation model is the third-order polynomial. All models are phone-independent [3]. The estimated articulatory trajectories are smooth but only for vowel sequences. Dusan and Deng [37, 36] improve this by considering phone-dependent models. Dusan and Deng [38] further generalize the above method by imposing phonetic and phonological constraints on the dynamic and observation models. They define phonological units as speech segments between two consecutive phonemes in a hope to account for coarticulation. At the inference stage, EKF is repetitively run over the model parameters of all phonological units to identify the best phonological unit that gives the highest likelihood. This repeated for all frames. Therefore, one disadvantage of this approach is the excessive computational complexity in practice because of the enormous number of phonological units.

4.3 Inversion by conditional density modes

Our inversion algorithm specializes a more general approach proposed for reconstructing sequences with missing data [22, 24]. It works as follows. *Offline*, assuming we are given a training set of articulatory-acoustic vector pairs $\{(\mathbf{x}_n, \mathbf{y}_n)\}$ (here obtained from the XRMB), estimate a joint density model $p(\mathbf{x}, \mathbf{y})$, or directly a conditional density model $p(\mathbf{x}|\mathbf{y})$. At *run time*, to invert a given acoustic sequence $\mathbf{y}_1, \dots, \mathbf{y}_T$, we obtain the conditional density $p(\mathbf{x}|\mathbf{y}_t)$ and its modes for each frame t; the latter explicitly represent the multiple inverse solutions at each $\mathbf{y}_t$. Then, we obtain the $\mathbf{x}$-trajectory by minimizing a constraint over the entire set of modes. Let us describe each step in detail.

Density model. The key property of the conditional distribution $p(\mathbf{x}|\mathbf{y})$ is that it will be peaked around the inverse solutions for $\mathbf{y}$ (see fig. 4.1). Thus, its modes are representatives of the inverses. If we represent the density with a Gaussian mixture, then we can approximate the true data density to

---

[3] A phone-independent model refers to a global model for all sounds and a phone-dependent model refer to a specific model for each sound.

arbitrary accuracy by using a sufficiently large number of components [16]. Thus, the modes can approximate the inverses as closely as desired (at a corresponding computational cost). A more interesting issue is that the topology of the manifold of inverse branches can be very complex, where the number of inverse branches may depend on the value of $\mathbf{y}$; for example, at a singularity two different branches intersect (fig. 4.1). The density can deal with these topology changes naturally in that e.g. two different modes can merge into a single one, and vice versa. This is a useful property that frees us from having to guess the number of inverses and fix it beforehand.

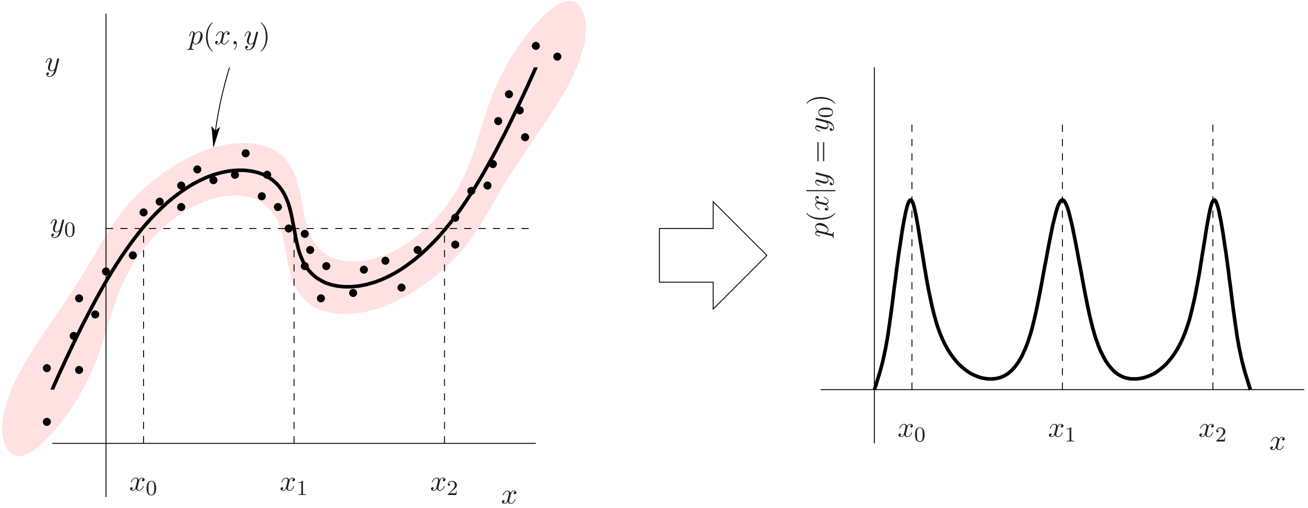


Figure 4.1: *Left*: joint probability density $p(x, y)$ (shaded area) and forward mapping f (thick black line). *Right*: multi-valued mapping $x = f^{-1}(y_0) = \{x_0, x_1, x_2\}$ from the multimodal conditional distribution $p(x|y)$. Depending on the value of y_0, there may be 3 modes (as shown), 2 or 1, and correspondingly 3, 2 or 1 inverses for y_0. (After [24])

We can estimate the conditional density with statistical machine learning techniques given the dataset of pairs $(\mathbf{x}, \mathbf{y})$. One option is to learn a *joint* density model $p(\mathbf{x}, \mathbf{y})$ and then obtain from it the conditional distribution

$$p(\mathbf{x}|\mathbf{y}) = p(\mathbf{x}, \mathbf{y})/p(\mathbf{y}) = p(\mathbf{x}, \mathbf{y})/\int p(\mathbf{x}, \mathbf{y})\, d\mathbf{x}. \tag{4.1}$$

A convenient density model for which computing conditional distributions is straightforward is a homoscedastic, isotropic Gaussian mixture, $p(\mathbf{z}) = \sum_{m=1}^{M} \pi_m p(\mathbf{z}|m)$ where $p(\mathbf{z}|m) \sim \mathcal{N}(\mathbf{z}; \boldsymbol{\mu}_m, \sigma^2 \mathbf{I})$. The Gaussian mixture parameters π_m, $\boldsymbol{\mu}_m$ and σ^2 (proportion, mean and covariance) are estimated from the training set by maximum likelihood with the EM algorithm [16]. Alternatively, one can use a nonparametric Gaussian kernel density estimate [17] if the dataset is not large (e.g. our case for the American English /ɹ/). Another option is to learn directly a *conditional* density model $p(\mathbf{x}|\mathbf{y})$; this is more efficient and economical because it needs to model a density in fewer dimensions (only $\mathbf{x}$, not $\mathbf{x}$

and $\mathbf{y}$) and be estimated by Mixture of Experts [66], Mixture Density Networks [16]. We study the joint density in section 4.4.

Mode finding. Assume we have a conditional density $p(\mathbf{x}|\mathbf{y})$ in the form of a Gaussian mixture, i.e., $p(\mathbf{x}|\mathbf{y}) = \sum_{m=1}^{M} \pi_m(\mathbf{y}) p(\mathbf{x}|m,\mathbf{y})$, where the mixture component $p(\mathbf{x}|m,\mathbf{y}) = \mathcal{N}(\mathbf{x}; \boldsymbol{\mu}_m(\mathbf{y}), \Sigma_m(\mathbf{y}))$ is a Gaussian kernel. Efficient algorithms for finding all the modes of a Gaussian mixture exist [23] that iterate a hill-climbing algorithm from every centroid of the Gaussian mixture. We provide a brief derivation as follows (see the detailed treatment in [23]). Let $\mathbf{g}$ be the gradient of $p(\mathbf{x}|\mathbf{y})$

$$\mathbf{g} = \nabla p(\mathbf{x}|\mathbf{y}) = \sum_{m=1}^{M} \pi_m(\mathbf{y}) \nabla p(\mathbf{x}|m,\mathbf{y}) = \sum_{m=1}^{M} p(\mathbf{x}, m|\mathbf{y}) \, \Sigma_m^{-1}(\mathbf{y}) \, (\boldsymbol{\mu}_m(\mathbf{y}) - \mathbf{x}) \tag{4.2}$$

where $\nabla p(\mathbf{x}|m,\mathbf{y}) = p(\mathbf{x}|m,\mathbf{y}) \, \Sigma_m^{-1}(\mathbf{y}) \, (\boldsymbol{\mu}_m(\mathbf{y}) - \mathbf{x})$. Equating (4.2) to zero, we can obtain a fixed-point iteration

$$\mathbf{x}^{(\tau+1)} = \mathbf{f}(\mathbf{x}^{(\tau)}) \text{ with } \mathbf{f}(\mathbf{x}) = \left(\sum_{m=1}^{M} p(m|\mathbf{x};\mathbf{y}) \Sigma_m^{-1}(\mathbf{y}) \right)^{-1} \sum_{m=1}^{M} p(m|\mathbf{x};\mathbf{y}) \, \Sigma_m^{-1}(\mathbf{y}) \, \boldsymbol{\mu}_m(\mathbf{y}) \tag{4.3}$$

In our case where we use the homoscedastic, isotropic Gaussian kernel ($\Sigma_m = \sigma^2 \, \mathbf{I}$ for $\forall m$), the fixed-point iteration reduces to a simplified form

$$\mathbf{x}^{(\tau+1)} = \sum_{m=1}^{M} p(m|\mathbf{x}^{(\tau)};\mathbf{y}) \boldsymbol{\mu}_m(\mathbf{y}) \tag{4.4}$$

where the posterior probability $p(m|\mathbf{x}^{(\tau)};\mathbf{y})$ is the normalized version of

$$\pi_m(\mathbf{y}) \, \exp\big(-\frac{1}{2} \, \|(\mathbf{x}^{(\tau)} - \boldsymbol{\mu}_m(\mathbf{y}))/\sigma\|^2 \, \big) \tag{4.5}$$

This fixed-point iteration is called the *Gaussian mean-shift algorithm*, where $\mathbf{f}(\mathbf{x}) - \mathbf{x}$ is the mean shift (which is proportional to the gradient of $p(\mathbf{x}|\mathbf{y})$ for isotropic kernels). Our Matlab implementation of *Gaussian mean-shift algorithm* is available at `http://faculty.ucmerced.edu/mcarreira-perpinan/software.html`.

With isotropic components, *Gaussian mean-shift* does not require inverting matrices and takes $\mathcal{O}(kM^2)$ where M is the number of components in the Gaussian mixture and k the average number of iterations per component. The computational time can be drastically reduced (at a small approximation error) by thresholding out mixture components far from the acoustic vector $\mathbf{y}_t$, and by accelerated variations of mean-shift [25]. By the nature of Gaussian mixtures, the number of modes returned at each frame is finite [23]. With redundant systems, in which excess degrees of freedom result in a continuous set of inverses, the modes will be a quantized version of this (see the redundant manipulator in appendix C for example).

Global optimization with dynamic programming. Assume we have collected for each time t in the trajectory all the modes (candidate vocal tract shapes for the acoustic frame $\mathbf{y}_t$). In principle, each of these modes represents a correct solution for step t (following a certain solution branch), but a given branch that is valid for part of the trajectory may be invalid for another part because of forbidden regions. To determine the solution, we minimize a global, trajectory-wide constraint over the set of modes. Generally, we consider a constraint of the form $\mathcal{C} + \lambda\mathcal{F}$ (for $\lambda \geq 0$), where:

- $\mathcal{C}(\mathbf{x}_1, \dots, \mathbf{x}_T) = \sum_{t=1}^{T-1} \|\mathbf{x}_{t+1} - \mathbf{x}_t\|$ represents a *continuity constraint* (integrated 1st derivative) [4]. This constraint is inspired by the fact that articulators move continuously and smoothly due to physical constraints. It penalizes discontinuous jumps in $\mathbf{x}$-space and encourages short trajectories. Alternatively, we could use $\mathcal{S}(\mathbf{x}_1, \dots, \mathbf{x}_T) = \sum_{t=1}^{T-2} \|\mathbf{x}_{t+2} - 2\mathbf{x}_{t+1} + \mathbf{x}_t\|$, which represents *smoothness* (integrated 2nd derivative).
- $\mathcal{F}(\mathbf{x}_1, \dots, \mathbf{x}_T) = \sum_{t=1}^{T} \|\mathbf{y}_t - \mathbf{f}(\mathbf{x}_t)\|$ represents a *forward constraint* (integrated acoustic error), and penalizes invalid inverses, i.e., modes $\mathbf{x}_t$ that do not map near the desired $\mathbf{y}_t$. This helps to eliminate spurious modes produced by ripple in the density model [5]. Note that in this chapter, since a reliable $\mathbf{f}$ is not available, the *forward constraint* is not imposed in the experiments.

Effectively, this is a form of planning in articulatory space. *Global* minimization of the constraint can be obtained by dynamic programming in $\mathcal{O}(N\nu^2)$ where ν is the average number of modes per step (usually very small), thus in linear time on the trajectory length N. Computationally, this is generally negligible compared to the mode-finding step. The articulatory trajectory returned represents a minimum-energy sequence of motions and represents the fact that physical articulators move slowly.

4.4 Experimental results

Dataset. We use data from the American English /ɹ/ from speaker `jw11` in XRMB. /ɹ/ is a sound with well-documented nonuniqueness both within and across speakers, where the tongue shapes have traditionally been divided into contrasting categories of retroflex (tongue tip raised, tongue dorsum lowered) and bunched (tongue dorsum raised, tongue tip lowered), though there really seem to exist a continuum between them [43, 155] (see fig. 4.2). So, it is useful as a test bench for algorithms that can deal with multi-valued mapping. Since there are no phonetic labels available in the XRMB,

---

[4] We did not empirically validate the effectiveness of the continuity constraint. Interested readers may refer to [132] for details.

[5] This refers to the nonsmooth density estimate in low-probability areas. This phenomenon is well-known in kernel density estimation with fixed-kernels [137] and is particularly noticeable in the tails of the distribution being approximated [24].

we manually choose frames corresponding to /ɹ/ from the entire database containing 45 760 paired articulatory-acoustic frames (as in chapter 2, we use 20-order LPC as acoustic features). Specifically, we manually identify frames of /ɹ/ that have typical retroflex and bunched tongue shapes. For each of these frames, we search in the original database its nearest neighbors in articulatory and acoustic spaces respectively. Then, the frames in the intersection of the two nearest neighbor sets are considered as frames of /ɹ/. We validate them by listening to the acoustics. Noisy frames are removed manually. Most frames in this dataset correspond to initial /ɹ/, e.g. "right" and "row"; some correspond to middle /ɹ/, e.g. "program" and "problem", and some to final /ɹ/, e.g. "never" and "order", all of which last shortly. The final dataset of /ɹ/ (fig. 4.2 shows the various tongue shapes it can adopt) contains a training set of 402 frames and 71 frames from 6 test trajectories from different utterances. Among the latter, 3 are retroflex (e.g. "right" in `tp099`, "roll" in `tp096`) and the other 3 are bunched (e.g. "rag" in `tp017`, "row" in `tp050`). Each trajectory contains a small stretch of /ɹ/ and possibly its neighboring sounds. Given the low relative frequency of /ɹ/ among all sounds, our dataset must necessarily be small. The dataset is available from the authors. Note that (see fig. 4.2) there seem to be more than 2 canonical tongue shapes for /ɹ/, i.e., there are big variability within groups of the bunched (e.g. the other bunched-like shapes belong to "row" in `tp050`) and retroflex tongue shapes. Such variability could be caused by coarticulation, which makes inversion harder. Also, the tongue coils `T1,T2,T3,T4` and `LL` show more variability than other articulators.

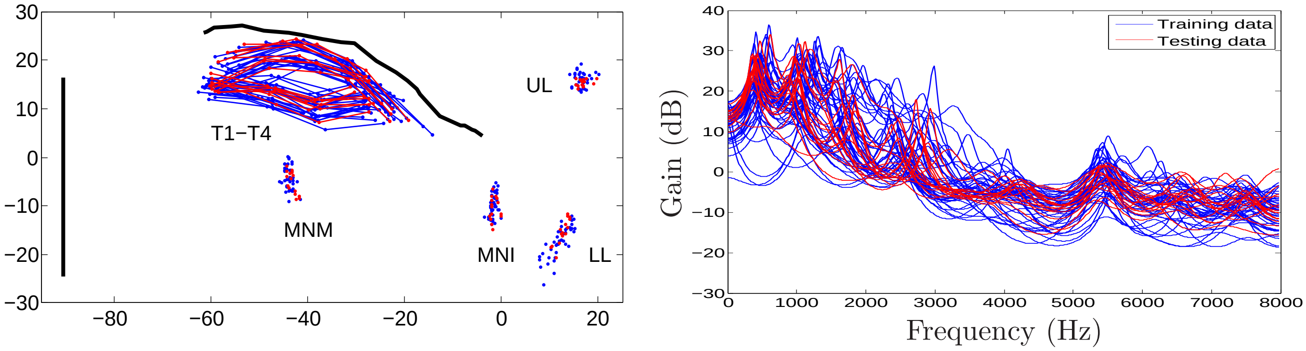


Figure 4.2: *Left*: datasets in the articulatory space (2D position in mm of each coil, with tongue coils `T1` to `T4` connected by line segments). *Right*: datasets in the acoustic space (spectral envelopes). Only a subset of frames shown to avoid clutter. The variablity of articulatory frames are due to different acoustic context, coarticulation, etc.

Methods. We refer to our algorithm as `dpmode`, and compare it to: (1) conditional mean `cmean` of $p(\mathbf{x}|\mathbf{y}_t)$ (which reconstructs $\mathbf{x}_t$ with $\mathrm{E}\left\{\mathbf{x}|\mathbf{y}_t\right\}$ independently for each frame); (2) a radial basis function network (`rbf`), which is asymptotically equivalent to `cmean`; (3) an oracle method `cmode` which picks the optimal mode at each frame (i.e., the one closest to the true $\mathbf{x}_t$); this provides a lower bound in the error achievable by `dpmode`. However, notes that `cmode` is sometimes slightly worse than `dpmode` for certain trajectories because of different performance metric, i.e., `cmode` is chosen based on L2 norm in the entire articulatory space.

Conditional density. We derive the conditional density $p(\mathbf{x}|\mathbf{y})$ from the joint density $p(\mathbf{x}, \mathbf{y})$, which is estimated by a Gaussian kernel density estimate (KDE) on the training set. So, in our case $p(\mathbf{x}|\mathbf{y})$ contains 402 Gaussian components. The bandwidth σ was set to $\sigma = 11$ by validation.

Feature normalization. The dynamic ranges of acoustic $\mathbf{x}$ and articulatory $\mathbf{y}$ are very different, i.e., $\sigma_{\mathbf{x}} = 5 - 20$, $\sigma_{\mathbf{y}} = 0.1 - 0.3$. We found empirically rescaling the articulatory data by $s = 8$ helped to smooth $p(\mathbf{x}|\mathbf{y}_t)$ and reduce spurious modes.

Performance metric. We use RMSE and Pearson correlation as defined in chapter 3.

Modes of the conditional density. Figs. 4.3–4.4 show the conditional modes over time for all testing trajectories. The middle frames in each test trajectory clearly show a multimodal conditional density, and the modes reliably identify both retroflex and bunched shapes, though occasionally additional, spurious modes exist (when number of modes is more than 3). In these middle frames, `cmean` is the average of the two canonical shapes, which is generally an invalid shape. The start and end frames tend to show a unimodal distribution, which implies the neighboring sounds have little nonuniqueness. In these cases, the conditional modes still identify the correct shape, but `cmean` performs well. This demonstrates that the density method is effective at identifying the multiple articulations that correspond to a given acoustics. On average, there are 3 or 4 conditional modes per testing frame depending on the scale s. In some cases (not shown here), `dpmode` would favor the alternative global trajectory instead of the true one, very similar to the case in robot arm inverse kinematics (see details in appendix C).

Inversion results. Figs. 4.3–4.4 show the reconstructions at each frame. `dpmode` significantly outperforms `cmean` and `rbf` and picks the correct shapes (either retroflex or bunched but not in between). The best inversion by `dpmode` is attained in the utterance `tp050` "$\underline{\text{r}}$ow", i.e., 0.65mm compared with 1.48mm by `cmean` (see fig. 4.5 for the trajectory reconstruction). The worst inversion by `dpmode` occurs in the utterance `tp096` "$\underline{\text{r}}$oll", i.e., 1.59mm compared with 1.98mm by `cmean`.

Fig. 4.6 plots the aggregated inversion errors on all test trajectories with each methods. On average, `dpmode` achieves an RMSE of less than 1.3 mm while `cmean` or `rbf` have an RMSE of over 1.9 mm. The advantage of `dpmode` is strongest in reconstructing the tongue coils. The RMSE values by `dpmode` in the presence of pervasive nonuniqueness are comparable with other methods that achieved RMSE of 1.5 to 1.9 mm in tasks with little nonuniqueness (see chapter 3 and [125]). However, the correlation

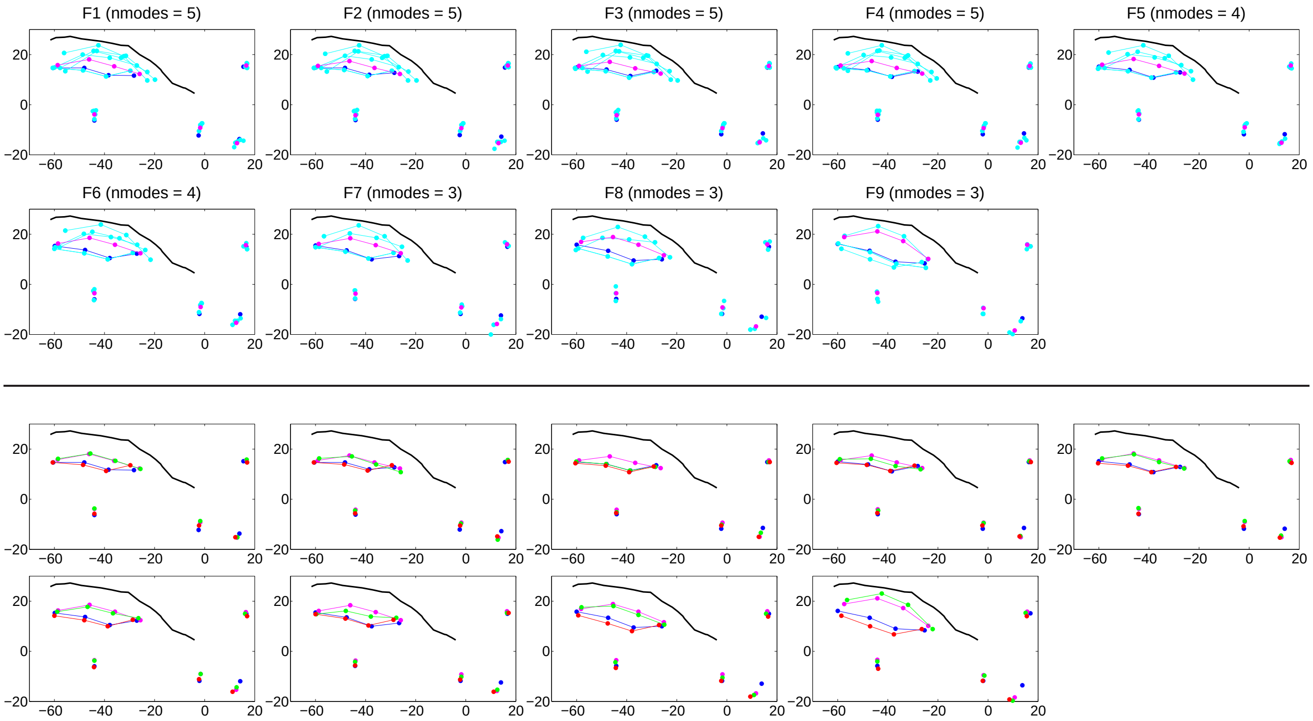


Figure 4.3: Sequence of reconstructions for the retroflex /ɹ/ in utterance `tp096` “roll”. *Top panels*: plots of the conditional modes (cyan; number of modes above each plot), `cmean` (magenta) and true value (blue) of all articulators. Tongue coils `T1`–`T4` are connected by segments. *Bottom panels*: plots of the reconstructions by all methods at each time frame: `dpmode` (red), `cmean` (magenta), `rbf` (green), and true (blue).

values for some articulators (`UL`, `MNI`, `MNM`) are misleadingly low for all methods, i.e., `dpmode` typically achieves less than 0.5 and likewise for `cmean` and `rbf` (see fig. 4.6 for details); the reason is that in our short utterances those articulators barely move (e.g. `MNMx` in fig. 4.4). The average RMSE (correlation) for the tongue and all coils are listed in table 4.1.

	`rbf`	`cmean`	`cmode`	`dpmode`
Tongue	2.66 (0.67)	3.08 (0.55)	1.19 (0.94)	1.20 (0.94)
All	2.07 (0.52)	2.26 (0.48)	1.27 (0.72)	1.30 (0.71)

Table 4.1: The average RMSE and correlation for tongue and all coils of articulatory inversion of American English /ɹ/ by various methods.

Although our dataset is small, the results are consistent and show `dpmode` is very good for dealing with nonuniqueness. We are yet to test the methods on a a comprehensive and accurate dataset of /ɹ/ segments, which could be obtained, e.g. by adding data from more speakers and possibly by use of

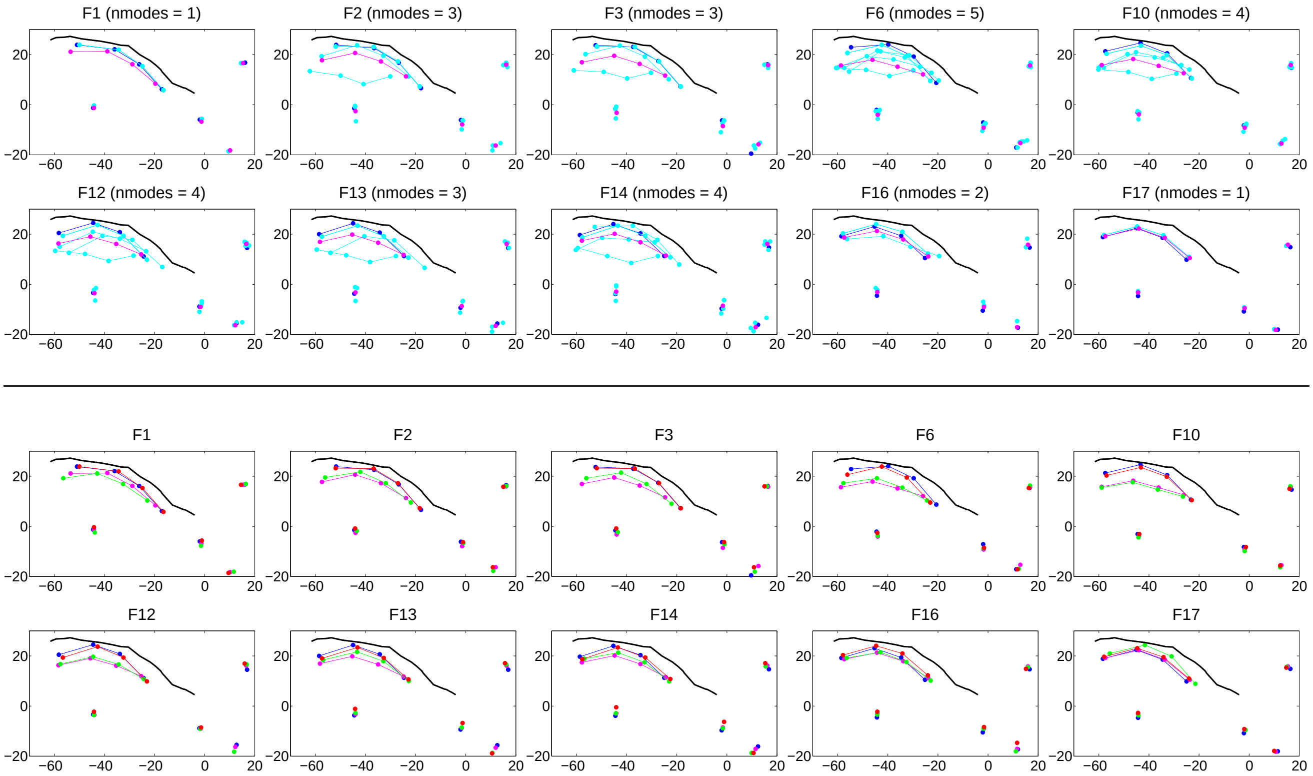


Figure 4.4: Like fig. 4.3 but for the bunched /ɹ/ in the utterance `tp040` "r̲ag".

time-aligned phonetic labels if available, which contains starting and ending time of sounds and hence can be used to segment acoustic frames.

4.5 Summary

We have proposed an articulatory inversion algorithm that uses a density model learned on real measured articulatory data to predict (possibly multiple) feasible, typical vocal tract shapes for a given acoustics, and disambiguates a sequence by choosing the smoothest path among these shapes. The algorithm correctly recovers either a retroflex or a bunched shape for the American English /ɹ/, while a neural network recovers an incorrect average of both. Since the algorithm is computationally more costly than a neural network, but nonunique articulations are overall infrequent in speech, a practical strategy may be to apply the algorithm selectively by detecting first the presence of nonuniqueness from the acoustics. For example, given one frame (or a short sequence of frames), one could perform nonuniqueness analysis. If the analysis returns positive (i.e., the target frame is determined empirically to be associated with multimodality), one may consider applying our conditional modes to resolve ambiguity.

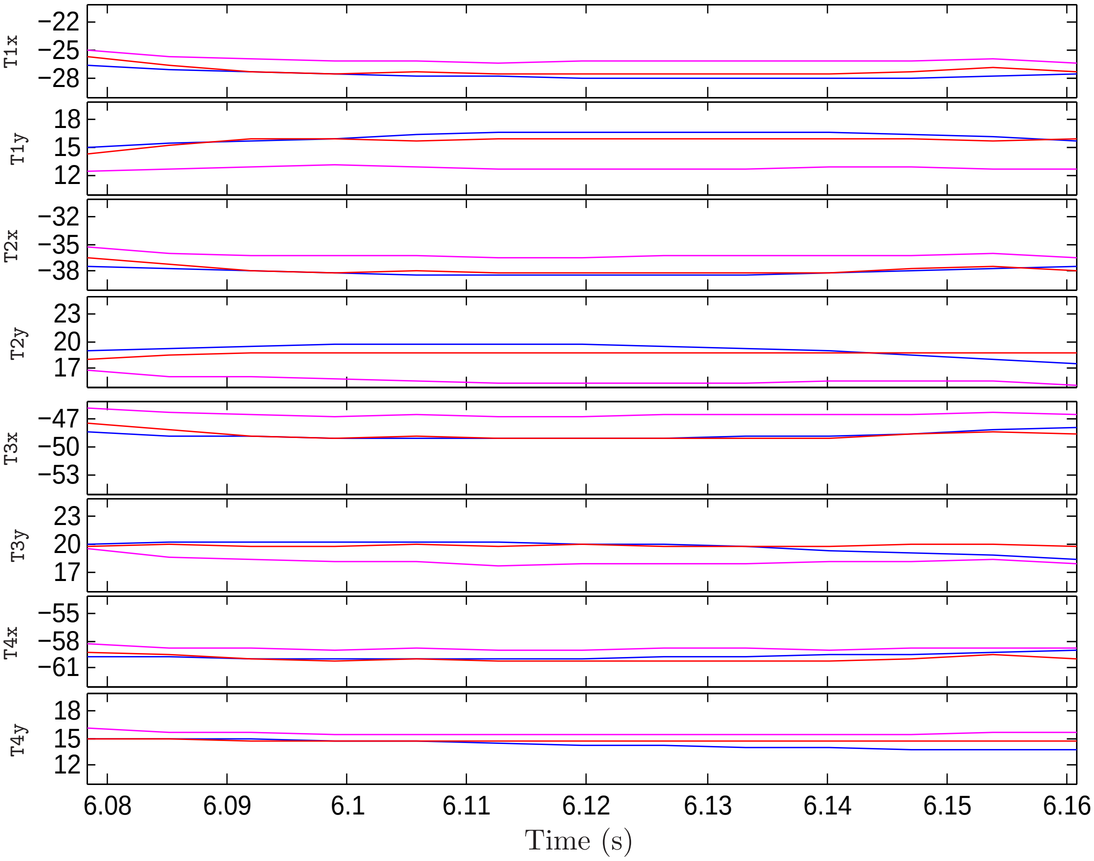


Figure 4.5: Temporal true trajectories (blue) and their reconstructions with `dpmode` (red) and `cmean` (magenta) for the bunched /ɹ/ in utterance `tp050` "r̲ow" (for coils `T1` to `T4`). `rbf` (not shown to avoid clutter) performs similarly to `cmean`.

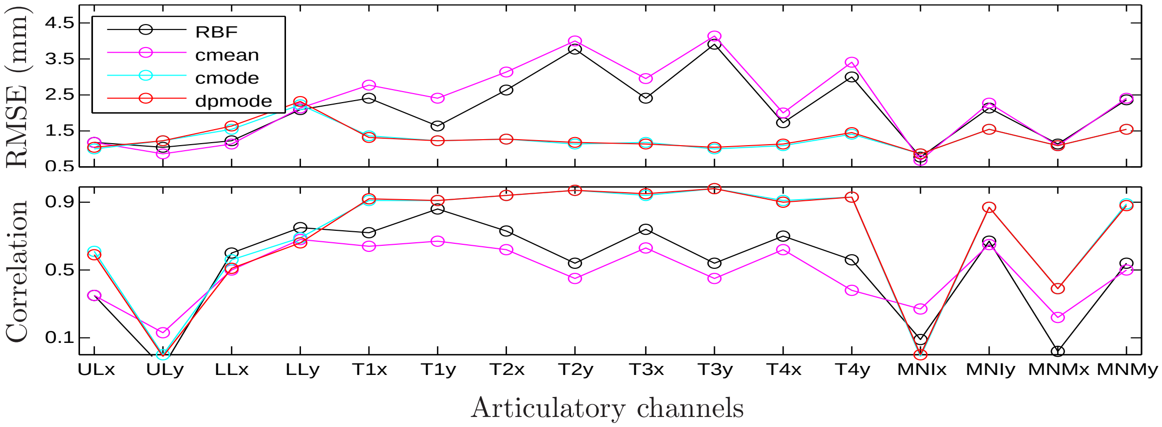


Figure 4.6: Inversion error (RMSE in mm) and correlation (in $[-1, 1]$) per articulator channel for each method.

Chapter 5

Predictive modeling of tongue shapes

5.1 Can we predict the entire tongue shape from a few landmarks?

Visualizing the entire tongue shape is the key to animated talking head and vocal tract visualization and also helps speech perception [41, 159, 158, 42]. However, the two large-scale articulatory databases (Fig. 5.1) using fleshpoint measurements, the Wisconsin XRMB (using XRMB) [154] and MOCHA (using EMA) [161] give only the 2D locations of 3 or 4 metal pellets attached to the tongue tip and dorsum (as well as the locations of the lips and other articulators, and the acoustic wave). While being real and dynamic measurements, these sparse representations are insufficient and could miss crucial speech production information, e.g. back of tongue is missing, and in principle could introduce extra ambiguity while inverting speech. Given the location of these pellets at a given time, what does the entire tongue shape look like? Do 3 or 4 pellets encode crucial information from the tongue for speech production? Are they enough to characterize the tongue shape accurately at all? Where do we place them optimally along the tongue?

To answer these questions, we consider the problem of reconstructing the shape of the tongue given the location of a few landmarks on its surface from XRMB/MOCHA. The ability to derive the full tongue shape from a few pellets would allow to animate the tongue shape for visualization purposes, and could be used as an input to methods for articulatory speech synthesis and inversion. This study would also help to determine the optimal number and placement of pellets during EMA or XRMB recording. The challenge is that XRMB/MOCHA only provide landmarks but not even one full contour as ground truth, which makes predictive modeling infeasible. Fortunately, as shown in chapter 1, it is possible to obtain more complete representations of the tongue shape through other articulatory imaging techniques, e.g. ultrasound or MRI, though they have their limitations . Therefore, the question becomes

---

This chapter is mainly based on the reference [117]

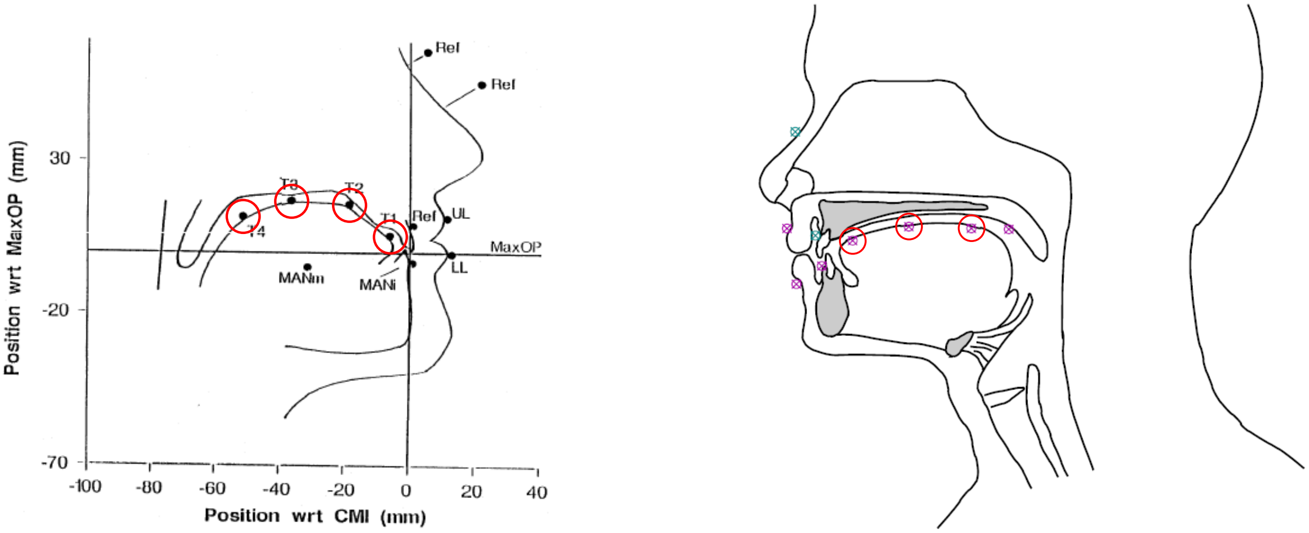


Figure 5.1: Locations of pellets in articulatory databases: XRMB (left, 4 tongue pellets located from the extreme end of the tongue tip), MOCHA (right, 3 tongue pellets located from the extreme end of the tongue tip).

whether one could leverage both articulatory imaging modalities (data fusion perspective). We propose our two-step approach: (1) predictive modeling of tongue shapes, i.e., derive a functional relationship between landmarks and full tongue shapes using ultrasound data from one speaker; (2) adapting the speaker-dependent predictive model, i.e., reconstruct the tongue for XRMB/MOCHA using the adapted model.

In this chapter, we address the problem of predictive modeling of tongue shapes. Here, we focus on reconstructing the midsagittal contour of the tongue rather than its full 3D shape, because our ultrasound data is limited to 2D images. Our approach can also be extended to the 3D case in a straightforward way. A simple reconstruction approach (that we and others have used) is to fit a smooth contour (e.g. a cubic or even piecewise linear spline) to the landmarks, justified by the observation that the tongue body is continuous and reasonably smooth during speech. However, smoothness is not enough to characterize the real behavior of the tongue, which can display very complex shapes during normal speech. For example, its midsagittal contour can show humps or valleys between landmarks or bend significantly in the tip or root (Figs. 5.2 and 5.7); and the tongue cannot go through the palate or teeth. It is possible to try to model the tongue shape by having a function with many control parameters and to model compression against the palate or teeth by assuming constant volume, as done in the Baldi talking head [28]. However, setting these parameters is difficult and time-consuming even for an expert, and even under the best settings the predicted shape may not look realistic enough. A similar problem arises in computer animation of the human body, where a combination of motion-capture and machine learning are able to reproduce realistic motion.

We follow a machine learning approach, where we estimate a nonlinear mapping from the landmark locations to the tongue contour using ultrasound data recorded for a subject during normal continuous speech. With this ground truth, estimating the optimal parameters can be done by numerical minimization of the reconstruction error, and we find that the predicted tongue contours look very realistic. In addition, we can also estimate the optimal location of the landmarks on the tongue, and the number of landmarks we need to achieve a given error. Section 5.2 discusses some previous work, section 5.3 describes the data collection, section 5.4 the predictive model and section 5.5 the experimental results.

5.2 Previous work

In this section, we review representative computational approaches to tongue reconstruction.

Spline interpolation. It reconstructs the tongue shapes by cubic spline interpolation of sparse representations of a tongue, e.g. a few tongue pellets (or landmarks) in 2D tongue shape reconstruction or a few 2D coronal contours or landmarks in 3D tongue reconstruction. Its advantage is that spline is scale and rotational invariant to placement of landmarks. By construction, it offers smooth reconstruction. However, it is unsupervised – reconstructing a given tongue contour independently of all other contours based on only a smoothness assumption. As a result, reliable and realistic interpolation by spline crucially depends on sufficient and proper placement of landmarks. Unfortunately, in practice, e.g. EMA recordings, placing too many tongue pellets (e.g. > 5) is restrictive because they might induce electromagnetic interference and affect normal speech production as well. On the other hand, sufficient but improperly placed landmarks would trigger erratic behaviors of spline, e.g. oscillating between landmarks and beyond end landmarks. From our own experience, spline interpolation would provide often shapes that are unrealistic and penetrate the palate, velum or teeth, in particular when extrapolating beyond landmarks. Indeed, note in fig. 2.3–2.5 (which were constructed using a spline interpolation) how the predicted tongue contour penetrates the palate, and gives a contour between the first and last landmark only. Stone and Lundberg [145] show that sixty ultrasound slices were needed to reconstruct 3D tongue shapes well for various English vowels and consonant by cubic B-spline interpolation. Lundberg and Stone [90] show the possibility to reconstruct the midsagittal contour from 6 landmarks with submillimetric accuracy only if they were placed optimally.

Linear regression. It formulates tongue reconstruction as a supervised regression problem: fit a linear predictive mapping (from landmarks to full contours) using full contours extracted from ultrasound recordings. The ground truth tongue contours are often acquired through ultrasound recordings (See

[142] for comprehensive reviews on ultrasound recordings of tongue shapes). It can reconstruct the tongue shape from locations of 3 or 4 pellets attached to the tongue in the midsagittal plane with submillimetric accuracy. This significantly improve spline interpolation, because variations of tongue shapes are "memorized" during the training phase. Lindau-Webb and Ladefoged [87] investigated the optimal placement of 2 tongue pellets used in XRMB recordings via linear regression analysis. But the optimal locations they found seem not reliable because the dataset they use was too small. Kaburagi and Honda [69] reported the average RMSE error of 0.48 mm in reconstructing the 2D ultrasound tongue contours from 4 optimally placed XRMB coils. Whalen et al. [156] considered inferring midsagittal pharynx shapes from the tongue shape via linear regression using MRI data of only 11 vowels. They reported the reconstruction error of less than 2 mm.

Articulatory model. It reconstructs the entire VT shape through sampling an articulatory model, which is a geometrical model that describes VT shapes in terms of a small set of articulatory parameters. The articulatory model is often estimated through factor analysis on measurements of the entire VT [57]. Therefore, articulatory models are built in an unsupervised fashion. For example, Maeda's articulatory model [91] is derived from a factor analysis on 1000 X-ray images of two female French speakers. This model describes the 2D midsagittal VT shape using 7 articulatory parameters (1 for jaw, 3 for tongue, 2 for lips, 1 for larynx) and 3 speaker-specific grid parameters (width of grid d, the increment β, polar grids θ) for the semi-polar grid coordinate system. Basically, each VT profile is computed as a linear combination of basis vectors spanning the VT space. The model also computes cross-sectional areas of acoustic tubes, which are often used as input to articulatory speech synthesizer to generate speech. Popular articulatory models include Maeda's [91], Mermelstein's [95], ASY [129] and some 3D tongue models [39, 10, 12]. Articulatory models are often restricted to certain speakers because of expensive collection of articulatory data. Also, they are often very coarse and sampling articulation models could produce unrealistic VT shapes.

5.3 Tongue data acquisition

In order to map landmarks to a full tongue contour, we need ground-truth data for tongue contours. Specifically, we consider a dataset consisting (for a given speaker) of N contours $\{\mathbf{y}_n\}_{n=1}^{N}$, where each contour $\mathbf{y} \in \mathbb{R}^{2P}$ is a vector giving the 2D coordinates of each of P points along the tongue. We collected such a dataset from ultrasound recordings.

Tongue contour recording. Unlike EMA and XRMB, ultrasound technology can image real-time, dynamic movement of the entire midsagittal tongue contour during speech in a noninvasive and unobtrusive way. Other advantages, such as high temporal resolution, portability and low cost make it very appealing in speech research. Ultrasound has disadvantages as well: the images contain speckle noise and unrelated edges; it only images the tongue but not any other articulator; the only visible imaged area is between the thyroid cartilage and the front of the mandible because of shadows; see [142] for a guide on using ultrasound to analyze tongue motion. One well known issue is data loss at tongue tip and root. For example, the extreme end of tongue tip is often invisible on ultrasound images for production of front vowels and dental consonants because the air in the sublingual cavity block the ultrasound propagation. It is possible to localizing the tongue tip by attaching a sensor on it [6]. But our recorded data did not use any reference sensor. Fig. 5.2 shows two typical midsagittal ultrasound images of our tongue in a 120-degree wedge-shaped scan. To create such images, the transducer is placed below the chin and the beam passes upwards through a thick section of the tongue. Once the sound wave reaches the air at the surfaces of the tongue, it reflects back and creates a highlighted line. The black region immediately below is the tongue body. The tongue surface is the lower edge of the highlighted line. One may notice that the tongue tip is not imaged but some highlighted lines in the ultrasound images that can not be part of the tongue (e.g. those edges next to the teeth shadow in the left image of fig. 5.2). These so called unrelated edges are actually the floor of the mouth which reflects the ultrasound beam and prevent it from entering the tongue tip (hence no image for tongue tip). Occasionally, unrelated edges also happen on tongue dorsum because the reflection there is typically not perpendicular to the beam. Then, sound waves are refracted in multiple directions and thus either never received by the transducer or spending more time traveling backing. Consequently, the tongue dorsum is either invisible or imaged as an unrelated edge somewhere on top of the tongue.

Tongue contour tracking. Given a set of 2D ultrasound images of a tongue (e.g. fig. 5.2), our goal is to extract the tongue contour (the lower edge of the highlighted strip) from each image. Manual tracking of the tongue contours suffers from several drawbacks well known in biomedical image analysis, including user bias, user fatigue, and not being able to achieve reproducible results. Also, it becomes infeasible for a large number of ultrasound images. Therefore, it is crucial to have an automated system for tongue movement analysis. However, the noisy nature of ultrasound images makes it very difficult at present to track tongue shapes reliably and automatically (see [100] for a comprehensive survey on ultrasound image segmentation), and this is compounded when dealing with multiple utterances and speakers. There are many algorithms developed specifically for tongue contour tracking [138, 85, 7, 148] (see [63] for a survey in tongue contour tracking in ultrasound images). We used EdgeTrak [85] for which a software implementation is available. Its algorithm is essentially based on the active contour [70], which iteratively minimizes an energy function designed to detect contours of the object in the

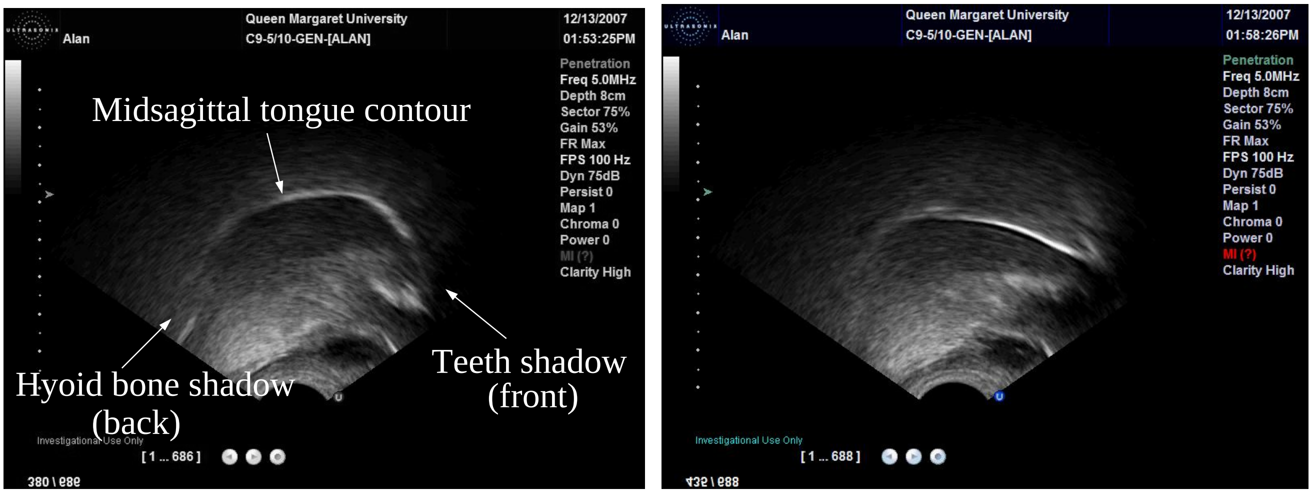


Figure 5.2: Typical ultrasound tongue images in our contour dataset. Artifacts such as noise, invisibility of tongue parts, bone shadows, sound reflection and interlacing video coding present difficulties for automatic tongue contour tracking.

image. However, none of these techniques can provide reliable tracking over time. For example, we observed in practice that EdgeTrak could get stuck at a local minimum and lose track, hence the need for manual corrections. Since in our study it was important to obtain high-quality ground-truth contours, we adopted a semi-automatic approach: we used EdgeTrak to provide a reasonable tongue contour at each frame, and then we adjusted the contours manually if necessary. It is worth mentioning one issue in contour tracking: loss of point correspondence over time. This makes it difficult to specify the location of landmarks. The fundamental reason traces back to volume preservation of tongue [105]: tongue stretches or compresses to preserve it volume. One possible way to enforce correspondence is to recording ultrasound and XRMB/EMA simultaneously as in [141, 69, 6], although it is difficult due to interference between the two channels. As shown in chapter 6, this problem does affect our reconstruction but it is mitigated effectively by use of regularization.

Tongue contour dataset. Following the procedure described above, we have created an ultrasound database at Queen Margaret University. It contains two speakers (one male, `maaw0`, and one female, `feal0`) with different Scottish accents. Two data streams were recorded synchronously for each speaker: acoustic waves (which we did not use in this study) and ultrasound videos. The ultrasound recorded the movements of the tongue in the midsagittal plane at 100 Hz. Each speaker recorded a set of British TIMIT sentences designed to be phonetically balanced. We also recorded two utterances containing two repetitions of three isolated words, “heed had whod”, with ascending and descending pitches, respectively. After discarding unreliably tracked contours, we obtained the following contour datasets.

- `maaw0`: dataset of 8671 contours, which were recorded in two sessions. Specifically, the first session **S1** consists of 3727 contours from 10 utterances and the second session **S2** consists of 4944 contours from 12 utterances.
- `feal0`: dataset of 7272 contours from 19 utterances.

More details on the tongue contour dataset can be found in Appendix A.

5.4 Predictive modeling of tongue shapes

We define the tongue reconstruction problem as follows. Of the P points along the contour, we choose K (say, 3) to represent the landmarks, or pellets affixed to the tongue (call this vector $\mathbf{x} \in \mathbb{R}^{2K}$). We then want to predict all P points (or rather, the remaining $P-K$) using a mapping $\mathbf{f}(\mathbf{x}) = \mathbf{y}$ that we estimate from a training set $\{\mathbf{x}_n, \mathbf{y}_n\}_{n=1}^N$ (see fig. 5.3). The predictive mapping $\mathbf{f}$ is fitted by least squares:

$$\min_{\mathbf{f}} E(\mathbf{f}) = \sum_{n=1}^{N} \|\mathbf{y}_n - \mathbf{f}(\mathbf{x}_n)\|^2 + \lambda \mathcal{R}(\mathbf{f}) \tag{5.1}$$

where E is the predictive square error and $\mathcal{R}(\mathbf{f})$ denotes the quadratic regularization with parameter $\lambda \geq 0$. We consider two predictive mappings for $\mathbf{f}$:

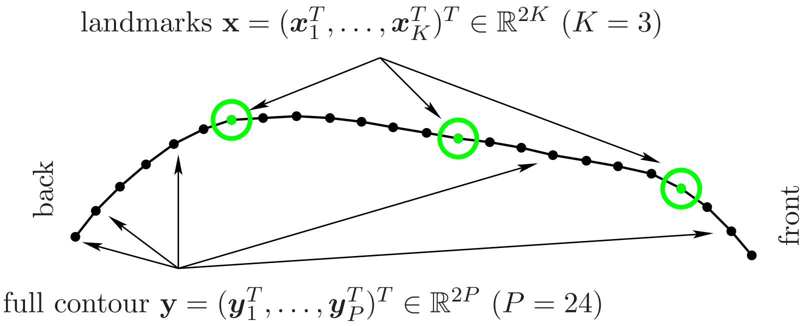

landmarks $\mathbf{x} = (\boldsymbol{x}_1^T, \dots, \boldsymbol{x}_K^T)^T \in \mathbb{R}^{2K}$ $(K = 3)$
back
front
full contour $\mathbf{y} = (\boldsymbol{y}_1^T, \dots, \boldsymbol{y}_P^T)^T \in \mathbb{R}^{2P}$ $(P = 24)$

Figure 5.3: The prediction problem: given the 2D locations of K landmarks located on the tongue midsagittal contour ($\mathbf{x}$), reconstruct the entire contour ($\mathbf{y}$), represented by P 2D points.

RBF predictive mapping. We represent $\mathbf{f}$ using a radial basis function (RBF) network [16]. Specifically, we consider M Gaussian basis functions of width σ and a bias term $\mathbf{w}$:

$$\mathbf{f}(\mathbf{x}) = \mathbf{W}\Phi(\mathbf{x}) + \mathbf{w} = \textstyle\sum_{m=1}^{M} \mathbf{w}_m \phi_m(\mathbf{x}) + \mathbf{w} \tag{5.2}$$

where $\phi_m(\mathbf{x}) = \exp\left(-\frac{1}{2}\|(\mathbf{x} - \boldsymbol{\mu}_m)/\sigma\|^2\right)$, $m = 1, \dots, M$, with centers $\boldsymbol{\mu}_m$, and $\mathbf{W}$ is a $2P \times M$ weight matrix. The reason for choosing a RBF network is that, besides being able to approximate many mappings accurately given sufficient number of basis centers [16], it also simplifies considerably our computations. In this experiment, where we need to fit a large number of mappings. However, in a practical situation where one simply needs to fit a single predictive mapping, one would get about equal good results with other nonlinear mappings, such as neural nets or Gaussian processes, provided a good model selection is achieved. We use $\mathcal{R}(f) = \operatorname{tr}\left(\mathbf{W}\mathbf{G}_{\mathbf{xx}}\mathbf{W}^T\right)$ for regularization (on $\mathbf{W}$ only, not $\mathbf{w}$) where $\mathbf{G}_{\mathbf{xx}}$ of $M \times M$ with elements $\phi_m(\boldsymbol{\mu}_m)$. The solution for the weights is unique and given by:

$$\min_{\mathbf{W},\mathbf{w}} \left\|\mathbf{Y} - \mathbf{W}\mathbf{G}_{\mathbf{xy}} - \mathbf{w}\mathbf{1}^T\right\|_F^2 + \lambda \operatorname{tr}\left(\mathbf{W}\mathbf{G}_{\mathbf{xx}}\mathbf{W}^T\right) \Rightarrow \tag{5.3}$$

$$\mathbf{W}(\mathbf{G}_{\mathbf{xy}}\mathbf{G}_{\mathbf{xy}}^T + \lambda\mathbf{G}_{\mathbf{xx}}) = (\mathbf{Y} - \mathbf{w}\mathbf{1}^T)\mathbf{G}_{\mathbf{xy}}^T \tag{5.4}$$

$$\mathbf{w} = \tfrac{1}{N}(\mathbf{Y} - \mathbf{W}\mathbf{G}_{\mathbf{xy}})\mathbf{1} \tag{5.5}$$

where we use a regularization term (on $\mathbf{W}$ only, not $\mathbf{w}$ since $\mathbf{w}$ does not operate on $\mathbf{X}$) with user parameter $\lambda \geq 0$, $\mathbf{1}$ is a column vector of N ones, and with the matrices $\mathbf{G}_{\mathbf{xy}}$ of $M \times N$ with elements $\phi_m(\mathbf{x}_n)$; $\mathbf{G}_{\mathbf{xx}}$ of $M \times M$ with elements $\phi_m(\boldsymbol{\mu}_m)$; $\mathbf{Y}$ of $D \times N$ (outputs) and $\mathbf{X}$ of $L \times N$ (inputs); $\mathbf{W}$ of $D \times M$ (weights) and $\mathbf{w}$ of $D \times 1$ (biases). The equation for $\mathbf{w}$ shows it captures the average error not accounted for by $\mathbf{W}$. Substituting it in the equation for $\mathbf{W}$, the explicit solution for $\mathbf{W}$ is given by the $M \times M$ positive definite linear system (positive semidefinite in non-generic cases):

$$\mathbf{W}\left(\mathbf{G}_{\mathbf{xy}}\mathbf{G}_{\mathbf{xy}}^T + \lambda\mathbf{G}_{\mathbf{xx}} - \tfrac{1}{N}(\mathbf{G}_{\mathbf{xy}}\mathbf{1})(\mathbf{G}_{\mathbf{xy}}\mathbf{1})^T\right) = \mathbf{Y}\left(\mathbf{I} - \tfrac{1}{N}\mathbf{1}\mathbf{1}^T\right)\mathbf{G}_{\mathbf{xy}}^T. \tag{5.6}$$

Linear predictive mapping. We can also represent $\mathbf{f}$ using a linear function (linf) as in [69]. For $\mathbf{f}(\mathbf{x}) = \mathbf{W}\mathbf{x} + \mathbf{w}$ (where $\mathbf{W}$ is of $D \times L$ and $\mathbf{w}$ is of $D \times 1$ dimension), minimizing E over $\mathbf{W}$ and $\mathbf{w}$ yields the normal equations:

$$\mathbf{W}\left(\mathbf{X}\mathbf{X}^T - \tfrac{1}{N}(\mathbf{X}\mathbf{1})(\mathbf{X}\mathbf{1})^T\right) = \mathbf{Y}\left(\mathbf{I} - \tfrac{1}{N}\mathbf{1}\mathbf{1}^T\right)\mathbf{X}^T \tag{5.7}$$

$$\mathbf{w} = \frac{1}{N}(\mathbf{Y} - \mathbf{W}\mathbf{X})\mathbf{1} \tag{5.8}$$

where $\mathbf{1}$ is a column vector of of N ones. We do not regularize $\mathbf{f}$ as it is not necessarily to obtain a good mapping.

Both RBF and linear predictive mapping are implemented by our Matlab package, available at `http://faculty.ucmerced.edu/mcarreira-perpinan/software.html`. In addition, we use spline interpolation as a baseline since it is a popular tongue reconstruction method [145, 90, 163]. We choose the cubic spline interpolation (implemented by the Matlab function `interp1(X,Y,‘spline’)`).

5.5 Experimental results

In this section, we study performance of our predictive models of tongue shapes. In particular, we investigate following open questions: (1) How well can the RBF model predictive the entire tongue shape and how does it compare with linear model and spline interpolation? (2) Where to place K landmarks optimally?

5.5.1 Experimental setup

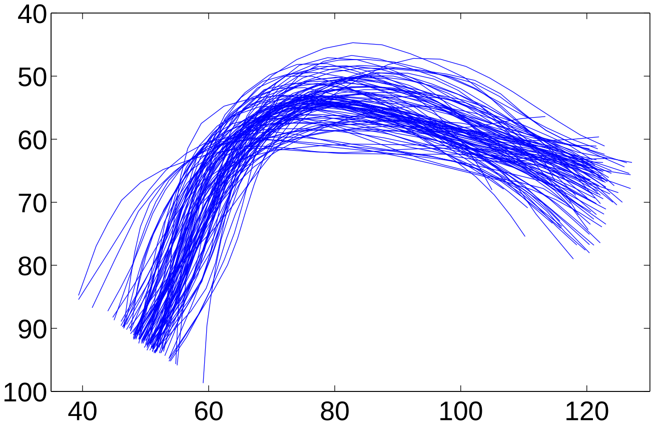


Figure 5.4: Contour dataset for speaker `maaw0`. Only a subset of contours plotted to avoid clutter.

Dataset. We use the contour dataset from `maaw0` and group datasets **S1** and**S2** together in the following experiments since we found the session mismatch affect predictive modeling very little (see details in chapter 6). Thus, the dataset consists of 8671 tongue contours with $P = 24$ points. For cross-validation, we select 30% (2601) frames randomly from the dataset for testing and the rest 70% for training set, among which 80% (4856) for learning and 20% (1214) for validation. Fig. 5.4 shows a subset of contour datasets for `maaw0`.

Comparison methods. We compare our RBF predictive mapping with (1) linear predictive mapping and (2) spline interpolation, for which we predict the contour $\mathbf{y}$ by considering a uniform grid of P locations along the X axis (with known Y values for K points) and applying to it the `spline` function.

Predictive performance measure. We report the root-mean-square error (RMSE) in mm for each contour point $i = 1, \ldots, P$: $(\frac{1}{N}\sum_{n=1}^{N}(y_i^{(n)} - \hat{y}_i^{(n)})^2)^{1/2}$, where n is the index of the contour in the dataset, and $\mathbf{y}_n$ and $\hat{\mathbf{y}}_n$ are the true and reconstructed tongue contours, respectively.

5.5.2 Predicting tongue contours from a few landmarks

RBF model estimations. In practice, training an RBF network is usually done in a suboptimal but efficient way. Rather than using a full-blown nonlinear optimization over all the parameters (centers, widths, weights), prone to local optima, one first estimates the centers of the BFs based only on the inputs $\mathbf{x}$ (using k-means, trying a few random initializations), and then estimates the weights based on the inputs $\mathbf{x}$ and outputs $\mathbf{y}$ (a linear least-squares problem with a unique solution given by a linear system). The number of BFs, the BF width and the regularization parameter l are obtained by cross-validation or perhaps fixed to a single value by a rule of thumb. The assumption, which works well in practice, is that if the BFs span well the input data then the weights will be able to achieve a good mapping no matter the outputs. Running k-means and cross-validating is much more costly than solving a single linear system, and in our case has to be repeated for every single combination of landmarks. We can limit the computation significantly, again in a suboptimal way, by separating the training into two steps. The goal of the first step is to fix once and for all the BF centers and widths and the regularization parameter l. We do this by solving the problem of estimating a RBF network that maps a contour using $K = P$ landmarks (i.e., the full contour) to itself. Thus, we force the RBF network to approximate the identity mapping and so the BFs span as well as possible the full input space (of $2.K$ dimensions). By cross-validating, this gives us the BF centers ($M = 600$) and width ($\sigma = 30$), the regularization parameter ($l = 10^{-3}$), and the weights (which we discard). Note that appropriate choices of σ and λ are important. Fig. 5.5 illustrates their effects on predictive and generalization performance of our RBF model with $M = 600$ basis functions as measured by RMSE predictive error and Frobenius norm of the weight matrix $\|\mathbf{W}\|_F$ [1], respectively. In the second step, given a specific choice of $K < P$ landmarks, we simply solve a linear system to obtain the corresponding weights in a unique way (no local optima). We repeat this for each desired combination of landmarks.

Our approximation is based on the same assumption used in regular RBF training, that the BF parameters are essentially determined by the inputs. Besides, we checked that it is close to optimal by estimating separately a few of the desired mappings; the resulting accuracy was very similar as shown in fig. 5.6.

Prediction comparison. Fig. 5.7 compares in selected frames the true tongue contour and the contours estimated by spline interpolation, linf and RBF prediction (trained by the optimal parameters), given $K = 3$ fixed landmarks (representing 3 EMA pellets). Fig. 5.7 also illustrates the rather complex shapes that the tongue can adopt, with significant changes in curvature, in particular when retracting the tip.

---

[1] A RBF model with large $\|\mathbf{W}\|_F$ would likely fail the prediction and adaptation because it will be very sensitive to variation on locations of input landmarks (landmark specification).

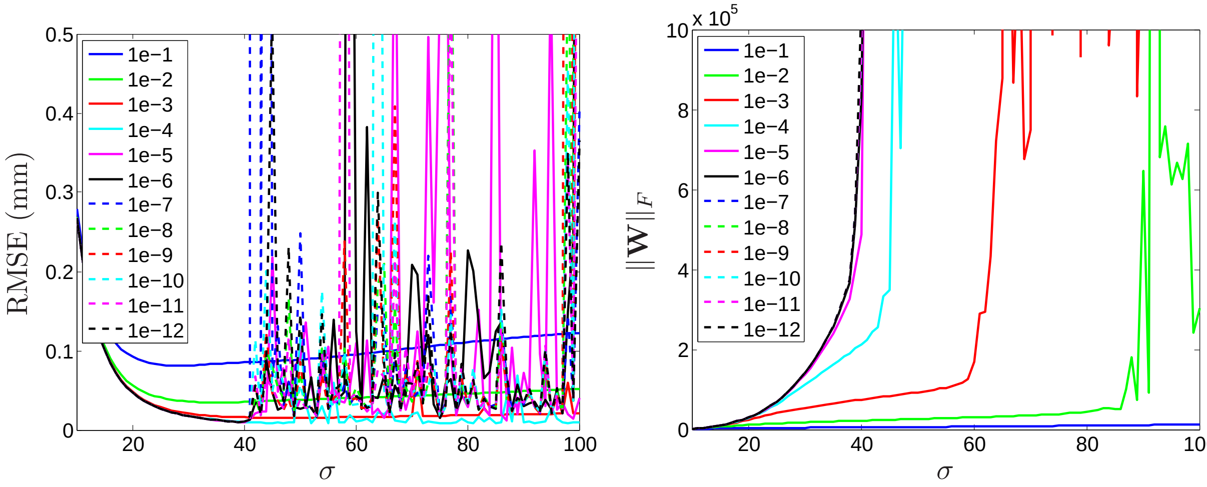


Figure 5.5: Generalization performance of RBF predictive mapping **f** (estimated by the RMSE on the test set and $\|\mathbf{W}\|_F$) from $K = 24$ landmarks to the $P = 24$ tongue contour (for $M = 600$). Each color denotes a different λ. The jagged graphs for high values of σ are due to the unstable linear system characterized by enormous $\|\mathbf{W}\|_F$.

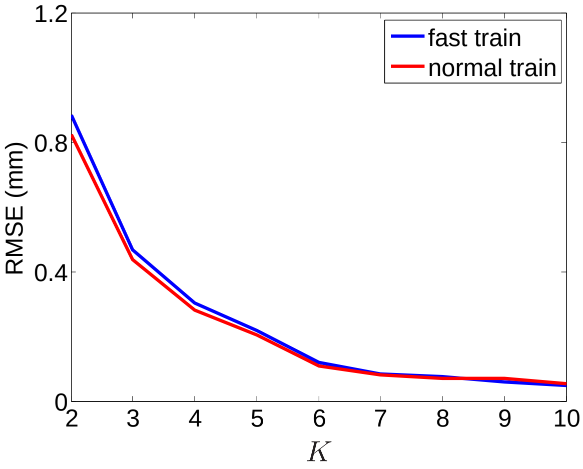


Figure 5.6: Comparison of fast and conventional training for RBF as a function of number of landmarks K (equidistantly placed along the contours). Normal training here refers to the suboptimal way of RBF training. The legend “normal train” refers to the conventional RBF training.

The contour predicted by the RBF overlaps almost perfectly with the true contour, so the latter is barely visible. Linear predictive mapping performs similarly though it is consistently outperformed by RBF. The spline contour often deviates significantly from the true one in particular. For example, since the spline behaves like an elastic bar, it is impossible for it to predict a sharp valley or hump between two adjacent landmarks (e.g. frames 2,4,9,12). When the landmarks are aligned (e.g. frames 2,6) the spline naturally adopts almost a straight line shape, which is physically infeasible for the tongue, and different indeed from the true contour. Spine performs extremely poorly when extrapolating tongue tip and root (e.g. frames 7,9,12). Though not shown here, spline interpolation is also very sensitive to the locations of landmarks. For example, when any two landmarks are aligned horizontally, it would produce big oscillations among and beyond these landmarks. In all these situations the RBF prediction works very well. The advantage of the prediction based on a training set is largest when extrapolating beyond the end landmarks, near the root or the tip of the tongue. Thus, in the following experiments on determining optimal locations of landmarks, such landmark configurations are not included in computing RMSE error.

Sufficient contours for training good predictive models. It is interesting to estimate how many contours are sufficient for training good predictive models. To investigate this, we fix number of landmarks and their placements, and vary choice of training contours. Fig. 5.8 plots the predictive errors as a function of the number of training contours N (using $K = 3, 4$ landmarks) for both RBF and linear predictive mapping. In general, the predictive error decreases with N and eventually almost reaches the optimal value (training with abundant data). For $K = 3$, training RBF using $N < 10$ contours has an enormous (several mm) error. Using 50 contours yields an optimal error less than 1 mm, while using 400 yields less than 0.4 mm. Using more contours can yields diminishing returns. Results are similar for $K = 4$ and linear predictive mapping except the latter requires less training contours and reaches the diminishing return much earlier than RBF. Both RBF and linear predictive mapping are robust to choice of training contours (note the tight error bars). These findings imply that tongue may actually adopt limited shape repository for various sounds in normal speech production and agree with [145].

5.5.3 Predictive performance as a function of choice of landmarks

In this section, we investigate optimal number of landmarks and their optimal placements. This study will be useful as a guide on how to place EMA tongue pellets optimally in EMA recordings.

Coincidental placement of landmarks. One interest question would be what if we place two landmarks identically on the tongue surface. Intuitively, this would reduce effective number of landmarks

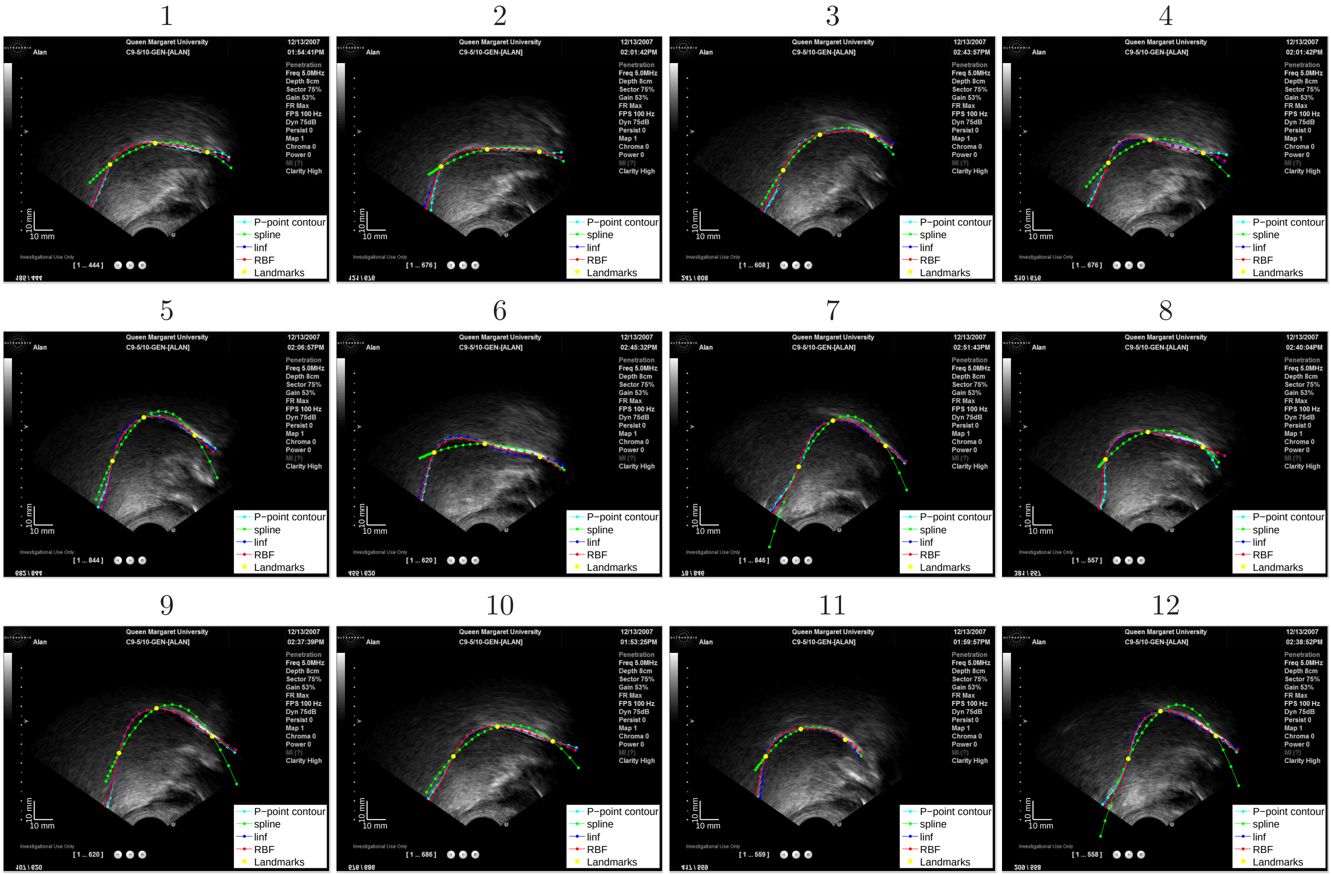


Figure 5.7: Selected frames comparing the true contour (cyan) and the contours estimated by the spline interpolation (green) and our RBF prediction (red), for $K = 3$ landmarks (yellow dots).

by one. For example, landmarks [6 6 18 18] [2] chosen out of the 24-point contour is equivalent to [6 18]. This intuition is validated through experiments on RBF prediction: [6 6 18 18] produces RMSE of 0.87 mm while [6 18] produces 0.88 mm (as representatives for many cases we tested). However, such choices of landmarks are impractical and hence we did not experiment them. We also tested the quasi-extreme case, i.e., [6 7 18 19], which produces 0.49 mm. The significant reduction in RMSE is attributed to the increased number of landmarks and their relatively separated placement. This indicates that it is useful to place more landmarks on the tongue even if two of them are placed closely. It is also evident in Fig. 5.9 that the worst placement for $K = 3$ is still better than the best placement for $K = 2$ for RBF and linf predictions.

Optimal number and locations of the landmarks. In order to determine the optimal location of

---

[2] Each number inside the bracket represents the local index of contour points. For example, [6 6 18 18] means we choose 6th, 6th, 18th, 18th points out of $P = 24$ contour points as our $K = 4$ landmarks.

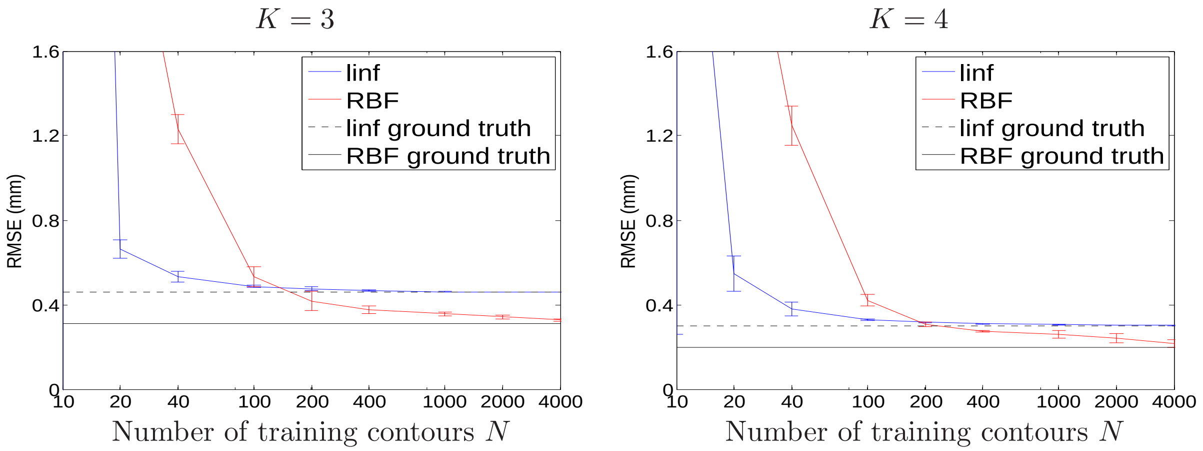


Figure 5.8: Predictive errors (RMSE) as a function of number of training contours for $K = 3$ (left) and $K = 4$ (right). Ground truths are computed with abundant training data ($N = 4000$). Landmarks for RBF and linf are placed at their respective optimal locations found in section 5.5.3. Errorbars are over 10 random choices of the N training contours (among which 30% are used for validation)

.

K landmarks, we would need to test each of the $\binom{P}{K}$ combinations. We limit the computational cost involved as follows: (1) We use the fast RBF training as described earlier, which makes it feasible to test large K. (2) We ignore unreasonable arrangements of landmarks by dividing the contour into K consecutive segments and constraining each landmark to select points from one; for example, for $K = 3$, landmarks 1, 2 and 3 can only select points 1–8, 9–16 and 17–24, respectively. This prevents landmarks from being all very close, or very far from each other, which illustrated in the previous paragraph would lead to a much worse prediction. This resulted in 145, 513, 1297, 2501, 4097, 5187, 6562, 5833, 5185 combinations for $K = 2, 3, 4, 5, 6, 7, 8, 9, 10,$ respectively.

Fig. 5.9 (bottom row, right) reports the RMSE averaged over the P contour points using RBF. For optimal placements, we also plot the error standard deviations over the test set. Fig. 5.9 (bottom row, left) shows that the prediction errors at each contour point are roughly quasi-symmetric around the fixed landmarks, with the lowest error on the landmarks themselves and the highest errors at midpoints between landmarks, or at the ends of the contour. With only $K = 2$ landmarks, RBF yields an optimal error of 0.80mm, while using 3 yields less than 0.4 mm and 4 yields 0.3mm. Using more landmarks yields diminishing returns; it is also practically harder to attach that many pellets ($K > 5$) to the tongue as it will introduce interference and affect normal speech production. The line labeled "worst" produces roughly twice an error than the "optimal". However, even those "worst" placement can produce reasonable predictions. In fact, they are the worse not among all placements, because we have ruled out impractical pellet arrangement that would indeed yield a far larger error (e.g. having all

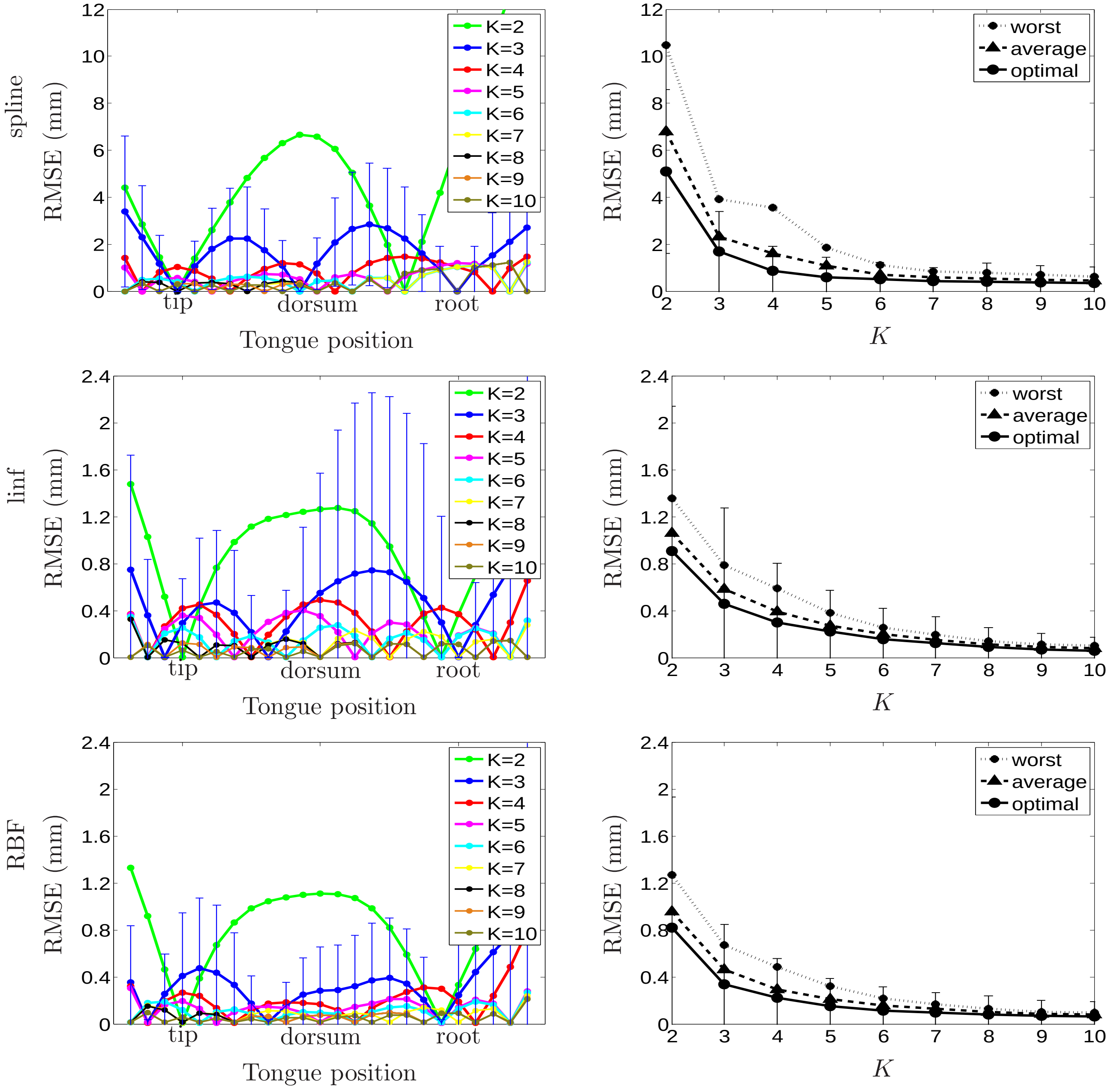


Figure 5.9: Error (RMSE) incurred by the RBF prediction (bottom row) and linf prediction (middle row) and spline interpolation (top row) of the tongue contour w.r.t. the ground-truth contour. Error bars plotted on optimal placement for $K = 3$ (to avoid clutter) are over the entire testing set. *Left*: RMSE (mm) for each contour point (averaged over all contours in the dataset) for different numbers K of landmarks, for the optimal landmark placement. *Right*: RMSE (mm) for each contour (averaged over all contours in the dataset and over all points in the contour), as a function of the number of landmarks K, for: the worst placement of the landmarks over the combinations we considered (solid line), the average over all combinations (dashed), and the optimal placement (dotted, corresponding to the left panel).

pellets next to each other).

Overall, linf produces similar predictions as RBF with some performance degradation. As shown in fig. 5.9 (middle row, right), using 2 landmarks yields an optimal error of 0.91 mm, while using 3 yields 0.5mm and 4 yields less than 0.4mm. This result is consistent with [69] where error of 0.48 mm was reported. Using RBF can achieve $25\% - 50\%$ reduction in RMSE except for $K = 2$ and $30\% - 50\%$ reduction on error standard deviation on top of linf prediction. The improvement is very consistent. linf produces symmetric predictive errors w.r.t. tip/dorsum/root, while RBF produces better predictions on tip and dorsum than root. On both ends of tongue, the two predictors perform closely.

Consistent with the previous section, the spline interpolation shown in fig. 5.9 (top row) is away much worse (by 5 times) than RBF prediction. The significant errors occur mostly around two end landmarks. Also, note that the error does not decrease consistently as K increases. This is because spline interpolation is quite sensitive to the placement of landmarks. In case any two landmarks are aligned roughly horizontally, the spline interpolation yields significant oscillation between them.

Fig. 5.10 shows the optimal location of the landmarks for $K = 2$ to 10 for three prediction methods. For all, the landmarks are roughly equidistant along the tongue contour except for small K, but somewhat closer to each other near the tongue tip, consistent with the fact that the tongue tip shows more complex movements than the rest of the tongue. For RBF, the end landmarks are close to the contour ends (tip and back). The scale bar allows to determine the positions in mm, and (after rescaling by the total tongue length) one can determine the approximately optimal placement for a different speaker. The approximate locations of the 3 pellets that were used in the MOCHA database are quite close to the optimal ones. From Fig. 5.9 we then estimate that the tongue contours may be reconstructed from the 3 MOCHA pellets with an error of around 0.4 mm at each point on the tongue contour. The fact that the "worst" and "average" lines in Fig. 5.9 (right) increase the error by only about 0.3 mm means that, if we cannot place the landmarks optimally as given by Fig. 5.10, the following recipe will yield reasonable results: place two pellets 2 to 4 mm from the tongue ends (tip and root, i.e., as far forward and backward as possible), and place the remaining $K - 2$ pellets so all K pellets are regularly spaced. Results are similar for linf and spline interpolation except they prefer to have end landmarks right at ends of contours. Optimal landmarks placement for $K = 4$ by RBF and linf are slightly different. linf's is roughly equidistant while RBF's has landmarks shifted towards the tongue tip.

5.6 Summary

We have shown that realistic tongue contours (with errors below 0.4 mm) may be predicted from as few as 3 or 4 landmarks (optimally located on the tongue) using a nonlinear mapping learned from ultrasound

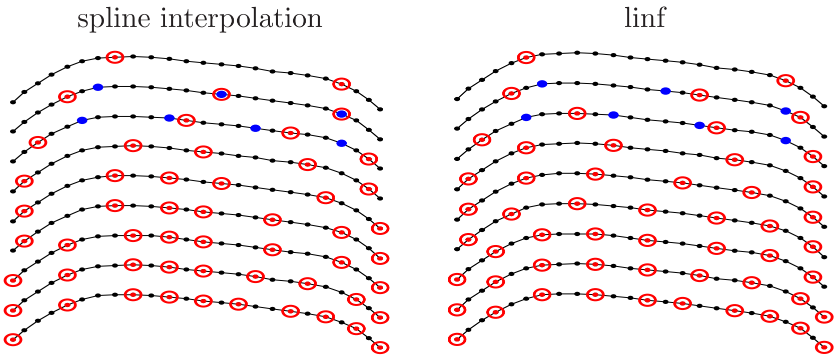


Figure 5.10: Optimal placement of K landmarks for spline interpolation (left), linf (middle), and RBF (right) depicted on a sample tongue contour (the tip is to the right and the root to the left). The blue dots in $K = 3$ and $K = 4$ show the approximate locations of tongue pellets in the MOCHA and XRMB databases respectively.

data. This information may be used to determine the optimal number and locations of pellets for EMA and X-ray microbeam technology. Our experiments demonstrate this nonlinear approach is much more successful than spline interpolation and also outperform the linear mapping. This quantifies the extent to which the EMA/X-ray data is a good representation of the tongue. The method is also applicable to predicting the 3D shape from landmarks if 3D ground truth is available.

Chapter 6

Adaptation of a predictive model of tongue shapes

6.1 Why adapt the predictive model?

In chapter 5, we have shown that it is possible to recover the full midsagittal contour of the tongue with submillimetric accuracy from the location of just 3–4 landmarks on it. This involves fitting a predictive mapping from the landmarks to the contour using a training set consisting of contours extracted from ultrasound recordings. However, extracting sufficient contours is a slow and costly process. In this chapter, we consider adapting a predictive mapping obtained for one condition (such as a given recording session, recording modality, speaker or speaking style) to a new condition, given only a few new contours and no phonetic correspondences between new and old conditions. There are several reasons why we may want to adapt an existing predictive model.

One obvious reason lies in recording and extracting ultrasound tongue contours. Firstly, recording with ultrasound several utterances from a given speaker often involves separate sessions because of subject fatigue, and removing and wearing again the stabilizing helmet and reattaching the ultrasound probe introduces misalignments. Secondly, although the imaging process itself is relatively straightforward given the ultrasound equipment and stabilizing helmet (as in chapter 5), extracting the contours from the ultrasound images is problematic. This is because the large amount of noise, false edges, shadows and other disturbances make automatic tracking systems unreliable, and their results must be corrected by hand. At an ultrasound recording rate of 100 Hz, a few utterances result in thousands of contours and require a considerable manual effort. Another practical problem is that ultrasound does

---

This chapter is mainly based on references [111, 115, 116] not image well tongues of many speakers in the first place, with the tongue surface boundary being incomplete and unclear in most frames (see fig. 6.1), leaving few contours usable to train the model.

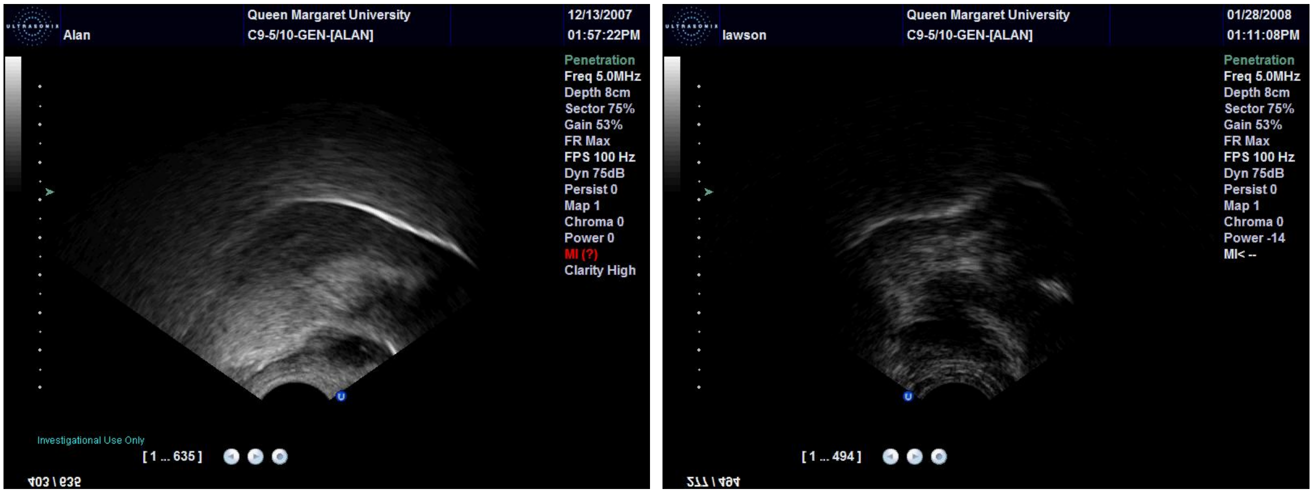


Figure 6.1: Examples of incomplete and unclear ultrasound images of tongue surfaces

Other reasons for adaptation comes from the need of data fusions — leveraging different articulatory acquisition techniques. Firstly, it is known that many techniques are complementary to each other, e.g. EMA and ultrasound, though none of them is perfect. It becomes increasingly interesting to couple these complementary techniques to obtain a more complete picture of dynamic movements of articulators (e.g. [6, 5]) [1]. Hence, registering different imaging modalities (or adapting one imaging modality from another) becomes necessary, because they are often measured in different coordinate systems. Secondly, in practice we may obtain contours for a new speaker from a different imaging modality (e.g. MRI) recorded separately. Finally, these new contours could even be partial measurements (e.g. XRMB, EMA). For example, a practical scenario is to reconstruct through the adapted predictive models of tongue shapes the full contours from locations of 3 or 4 tongue pellets in MOCHA/X-ray microbeam databases where not even a single full contour is available [2].

Successful adaptation can mitigate recording burdens and enable useful applications in articulatory models, articulatory inversion, visualization, speech production studies. Quick, automatic adaptation of an existing well-trained model for a reference speaker given a small number of segmented contours from the new speaker becomes attractive.

Formally, we formulate the **adaptation problem** as follows: given a predictive mapping **f** for

---

[1] Recording different imaging modalities simultaneously could be difficult due to interference between two channels.

[2] As demonstrated in chapter 5, reconstruction by predictive models far outperforms interpolation methods based on splines in particular for extrapolating beyond the end landmarks.

speaker 1 and a new, small training set $\mathcal{S}_2 = \{(\mathbf{x}_n, \mathbf{y}_n)\}_{n=1}^{N}$ with N contours from speaker 2, estimate a new predictive mapping for speaker 2. In this chapter, we focus on speaker adaptation using full or partial contours from the new speaker although the adaptation framework could be extended to other tasks such as session alignment and modality registration.

With only a small dataset for the new speaker, it becomes difficult to estimate an accurate, complex predictive mapping. More critically, these limited new data makes it infeasible to perform generic model adaptation (as reviewed in section 6.2) explicitly because adaptation could easily lead to overfitting. Therefore, we propose adaptation by *feature normalization* — estimating feature transformation $\mathbf{g}$ that maps new data to old data. The new predictive mapping will be obtained as the composition $\mathbf{g}^{-1} \circ \mathbf{f} \circ \mathbf{g}$. We propose two types of transformations: (1) *global feature adaptation.* (2) *local feature adaptation.* The fundamental difference is that the latter considers a separate *linear, invertible feature transformation* at each landmark and each contour point. More crucially, this generalization eliminates the bias and mitigates the stagnation issues. Note both adaptations assume no correspondence, i.e., adaptation data does not necessarily to contain the same phonetic transcription as the training data.

In summary, the need to obtain models for new conditions (such as a new recording session, recording modality, speaker or speaking style) and the difficulty of extracting quality contours make it a necessity to adapt an existing predictive model given only few full or even partial contours from ultrasound recordings or other imaging modality. We propose to adapt the predictive model automatically by feature normalization that does not require correspondences. The resulting methods are very robust and achieve high accuracy with just a few contours. We review some related work in section 6.2, describe the predictive model in section 6.3 and the adaptation methods in sections 6.4–6.5 respectively, and show experimental results in section 6.6.

6.2 Related work

Speaker adaptation is an important topic in speech science and has been researched for years. In the following, I organize representative work in different contexts.

Adapting acoustics HMMs for speech recognition. These methods (reviewed in [160]) can be further classified in the following types. *Maximum-a-posteriori* methods (e.g. [51]) apply Bayes' rule using as prior the trained model. This converges to the true maximum-likelihood (ML) estimate with infinite data, but performs poorly with little data because only a few parameters are updated. *Parameter tying* methods (e.g. *MLLR, maximum likelihood linear regression* [83, 50, 48]) aim to avoid overfitting problems (as there are enormous number of parameters in HMMs with respect to the limited data for adaptation) by applying shared and usually linear transformations to the HMM means and covariances.

They do not converge to the ML estimate and so do worse than MAP with abundant data, but with little data they do update all parameters and reduce the error much more. Hiroya and Honda [60] presented a similar method but in the context of acoustic-to-articulatory mapping. It indirectly adapts the acoustic parameters to compensate geometrical difference of vocal tract shapes by adjusting the linear acoustic-to-articulatory model in each HMM state. *Feature normalization methods* transform the features instead of the model parameters so the new speech better matches the one used to train the original HMM. They are closest in spirit to our adaptation work. This is done by arbitrary linear transformations (constrained MLLR for HMMs [33] or HMM-ANNs [99]) or special cases of them, such as cepstral mean normalization or vocal tract length normalization by frequency warping (e.g. [82] implicitly results in linearly tying the HMM parameters). *Speaker space methods* define a space where each point represents one speaker model (e.g. by concatenating all means in a supervector) and interpolate among several speaker models by clustering or PCA (e.g. [77, 49]). All above methods have been successfully applied in HMM-based speech recognition and require a word-level transcription (i.e., correspondences) such that each phone HMM can be adapted by data corresponding to the phone. However, it is difficult to apply them methods to our problem. This is mainly because our adaptation data is very limited with respect to number of model parameters and they do not assume correspondences.

Adapting articulatory models for articulatory speech processing. Recall from chapter 5, an articulatory model, e.g. Maeda's model, is a mathematical model that characterizes the geometrical shape of VT as a function of a few controllable parameters, which are often chosen by hand. Some articulatory models are learned by applying principal component analysis on VT shapes segmented from MRI or X-ray images. Adapting an existing articulatory model has the advantages of reusing an existing model to reconstruct the VT for a new speaker without recording large amount of speaker-specific articulatory data, which is time-consuming and expensive. It also enables synthesizing speech through articulatory synthesizer with the reconstructed VT as input. As a result, the adaptation can be evaluated in terms of both visual and acoustic match. On the other hand, adaptation would inherit the problems of articulatory models, e.g. producing unrealistic VT shapes and unrealistic synthetic voice. Furthermore, human intervention or extra articulatory modality are often needed for adaptation tasks. It is worth noting that it is possible to adapt an articulatory model with an optimization procedure as our adaptation framework. Recall that in a typical articulatory model, e.g. Maeda's model, VT outline can be completely determined by scale factors and a set of articulatory parameters. Therefore, these methods can be further classified based on which parameters to adapt:

- *Methods that adapt scale factors* (e.g. [94, 54, 20, 40]). They adapt scale factors (i.e., the semi-polar grid) that controls VT dimensions of oral and pharynx cavity. This is often achieved by superimposing the articulatory model on sagittal MRI image of some vowels and then estimating the factors manually that achieve best overall fit between the exterior contours produced by adapted model and those extracted measurements. Note that the goal is to adapt the articulatory model such that it can generate sounds that specific to new speakers.

- *Methods that adapt articulatory parameters* (e.g. [15, 1]). They adapt articulatory parameters, e.g. jaw height, tongue location and shape, lip aperture and protrusion, larynx height, from other articulatory measurement, e.g. XRMB or EMA. Birkholz and Kroger [15] adjusts manually parameters of the 3D vocal tract model of a Russian speaker such that model-derived outline match closely with the measured MRI outline of vocal tract of a German speaker. Bawab et al. [1] derive articulatory parameters of Maeda's model manually from EMA data in MOCHA through a geometrical mapping. In addition, the method can return the entire VT shape w.r.t EMA data and hence is able to synthesize the corresponding speech sound through an articulatory synthesizer. These adaptations are essentially feature normalization. However, they could be insufficient because they ignore speaker-specific parameters, e.g. length and shape of VT, encoded in scale factors. Also, the method has to use heuristics to adapt some parameters, e.g. jaw height and lip protrusion, that don't have correspondence to measured EMA data.

- *Methods that adapt both scale factors and articulatory parameters* (e.g. [11, 151, 2]) They adjust speaker-specific parameters and derive articulatory parameters by inverting the linear articulatory model. Toutios et al. [151] estimates articulatory parameters by minimizing distance between model output and measurements while respecting geometrical constraints on articulatory parameters. In addition, they introduce regularization terms to encourage the temporal smoothness of estimated articulatory trajectory and solve the new optimization problem by variational calculus with initialization from the original unregularized solution. This method can be viewed as "on-line" adaptation — deriving separate articulatory parameters for each EMA measurement. Since estimating articulatory parameters at each frame involves solving a complicated optimization problem that is prone to local optima, this adaptation is very expensive. Bawab et al. [2] improve [1] by adapting 3 grid parameters in Maeda's model and finding the best geometric-fit VT shape through a codebook search. This idea is very close to [151] except the latter searches the closest VT shape in model space (in contrast, [2] search in data space through sampling the model). Note that searching for the best-fit VT shape in a huge codebook independently could be expensive and this does not respect temporal smoothness. This method is very related to ours in that it can reconstruct the entire tongue shape by using only EMA data in MOCHA.

6.3 The predictive model of tongue shapes

We briefly review predictive modeling of tongue shapes. We want to predict the full tongue contour $\mathbf{y} = (\boldsymbol{y}_1^T, \dots, \boldsymbol{y}_P^T)^T \in \mathbb{R}^{2P}$ consisting of P points $\boldsymbol{y}_i \in \mathbb{R}^2$ given only the positions $\mathbf{x} = (\boldsymbol{x}_1^T, \dots, \boldsymbol{x}_K^T)^T \in \mathbb{R}^{2K}$ of K landmarks $\boldsymbol{x}_i \in \mathbb{R}^2$ (fig. 6.2). Our approach fits a predictive mapping $\mathbf{f}$ by minimizing the predictive square error $E(\mathbf{f}) = \sum_{n=1}^{N} \left\|\mathbf{y}_n - \mathbf{f}(\mathbf{x}_n)\right\|^2$ (plus a regularization term for RBFs) given a sufficiently large training set, and $\mathbf{f}(\mathbf{x}) = \mathbf{W}\mathbf{x} + \mathbf{w}$ (linear) or $\mathbf{f}(\mathbf{x}) = \mathbf{W}\Phi(\mathbf{x}) + \mathbf{w}$ (RBF) with M basis functions $\phi_m(\mathbf{x}) = \exp\left(-\frac{1}{2}\left\|(\mathbf{x} - \boldsymbol{\mu}_m)/\sigma\right\|^2\right)$. The RBF is trained in an efficient but slightly suboptimal way (as commonly done) by fixing the centers $\boldsymbol{\mu}_m$ by k–means and cross-validating the width σ and the regularization parameter λ.

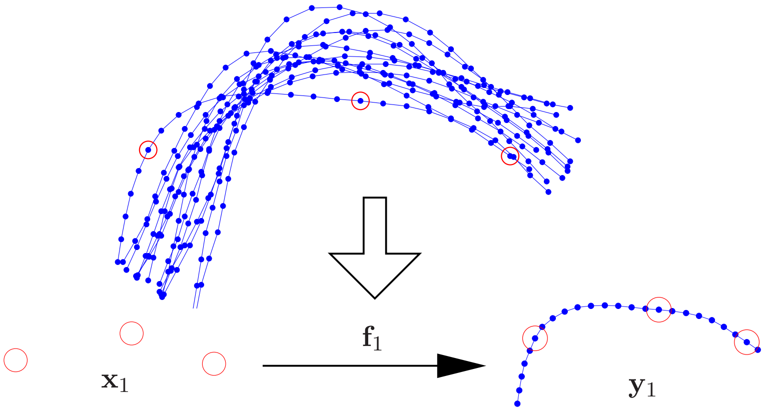


Figure 6.2: Training a predictive model $\mathbf{f}_1$ for speaker 1 with sufficient numbers of full contours (here $K = 3$ landmarks and $P = 24$ points).

6.4 Feature adaptation of predictive models with full contours

In this section, we consider the following adaptation task (see fig. 6.3): given a predictive mapping $\mathbf{f}_1$ for speaker 1 and a new, small training set $\mathcal{S}_2 = \{(\mathbf{x}_n, \mathbf{y}_n)\}_{n=1}^{N}$ with N full contours from speaker 2 (i.e., each adaptation data item is a pair $(\mathbf{x}_n, \mathbf{y}_n)$ where $\mathbf{y}_n$ is the P–point contour and $\mathbf{x}_n$ the K–point input (a subset of $\mathbf{y}_n$)), estimate a new predictive mapping $\mathbf{f}_2$ for speaker 2.

Given the limited training set $\mathcal{S}_2$, it is difficult to estimate an accurate, complex predictive mapping from scratch [3]. The data sparsity also poses significant challenges on performing model adaptation, because model parameters in our predictive mappings (e.g. weights, centers, widths in RBF) significantly outnumber the adaptation data. One way to circumvent this is to tie model parameters and adapt them

---

[3] In chapter 5, we show that it is necessary to have sufficient contours, e.g. $N > 200$, to estimate such a mapping.

together using same data. However, this explicit parameter tying is very difficult to implement since it requires information that are not available in our problem setting. Rather, we take the *feature normalization* approach related to that used in acoustic HMM adaptation [83, 160]. In summary, adaptation of an existing predictive mapping $\mathbf{f}$ is done as follows: (1) estimate an linear, invertible transformation through which map new data to old data (see fig. 6.4 for example); (2) apply the old predictive mapping and maps back (with the inverse transformation) the predictive full contour to the new speaker space.

We propose two algorithms for feature normalization. The first algorithm enforces all the landmarks and contour points to share a global linear transformation. It turns out to be simple but effective and achieves reasonable accuracy in just seconds of CPU because it involves estimating very few parameters. It can also be extended to adapt a full-contour model given only partial contours (see section 6.5). The second algorithm extends the first one by allowing each landmark and contour point to have separate linear transformations. This extension turns out to be very useful because it eliminates the bias and mitigates the error stagnation problem (i.e., increasing K or adaptation data does not reduce the predictive error) by relaxing the transformations at the expense of increased complexity. These two versions of adaptation makes our algorithms very flexible and convenient for users to trade off runtime and accuracy. Let us describe two algorithms in details in following sections.

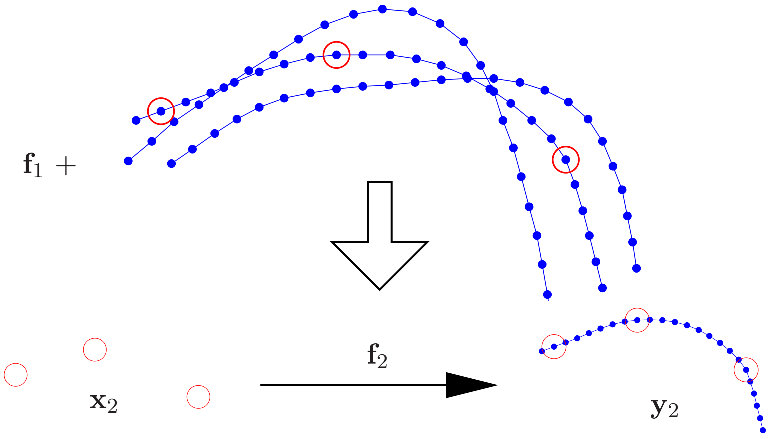


Figure 6.3: Adapting $\mathbf{f}_1$ for speaker 1 to speaker 2 given a few full contours.

6.4.1 Global feature adaptation

The key aspect of this *global feature normalization* is to apply the same transformation to each 2D point of an $\mathbf{x}$– or $\mathbf{y}$–contour. We take this *2D-wise alignment transformation* $\mathbf{g}(\boldsymbol{x}) = \mathbf{A}\boldsymbol{x} + \mathbf{b}$ to be linear, in order to ensure it is invertible and has few parameters. Consequently, the inputs $\mathbf{x}$ and outputs $\mathbf{y}$ also undergo *invertible linear transformations* $\mathbf{g}_\mathbf{x}$, $\mathbf{g}_\mathbf{y}$:

$$\tilde{\mathbf{x}} = \mathbf{g}_\mathbf{x}(\mathbf{x}) = \left(\begin{smallmatrix} \mathbf{A}\boldsymbol{x}_1+\mathbf{b} \\ \cdots \\ \mathbf{A}\boldsymbol{x}_K+\mathbf{b} \end{smallmatrix}\right) = (\mathbf{I}_K \otimes \mathbf{A})\mathbf{x} + \mathbf{1}_K \otimes \mathbf{b} \tag{6.1}$$

$$\tilde{\mathbf{y}} = \mathbf{g}_\mathbf{y}(\mathbf{y}) = \left(\begin{smallmatrix} \mathbf{A}\boldsymbol{y}_1+\mathbf{b} \\ \cdots \\ \mathbf{A}\boldsymbol{y}_P+\mathbf{b} \end{smallmatrix}\right) = (\mathbf{I}_P \otimes \mathbf{A})\mathbf{y} + \mathbf{1}_P \otimes \mathbf{b} \tag{6.2}$$

where $\otimes$ is the Kronecker product. The adapted predictive mapping is given by $\mathbf{g}_\mathbf{y}^{-1} \circ \mathbf{f} \circ \mathbf{g}_\mathbf{x}$. Then, adaptation requires estimating only the 6 parameters $\mathbf{A}_{2\times2}$ and $\mathbf{b}_{2\times1}$, which are shared among all 2D points in a contour (so one contour is enough if $P > 2$ points). To estimate $\{\mathbf{A}, \mathbf{b}\}$ we need to define a suitable error function. The obvious candidate is the same one that was used to estimate the predictive mapping $\mathbf{f}$, namely the predictive squared error $E(\mathbf{A}, \mathbf{b})$:

$$\min_{\mathbf{A},\mathbf{b}} E(\mathbf{A}, \mathbf{b}) = \sum_{n=1}^{N} \left\| \mathbf{y}_n - \mathbf{g}_\mathbf{y}^{-1}(\mathbf{f}(\mathbf{g}_\mathbf{x}(\mathbf{x}_n))) \right\|^2. \tag{6.3}$$

Even when $\mathbf{f}$ is linear, E is very nonlinear, involving both $\mathbf{A}$ and its inverse, and has a complicated gradient. Instead, we use a proxy error function $F(\mathbf{A}, \mathbf{b})$:

$$\min_{\mathbf{A},\mathbf{b}} F(\mathbf{A}, \mathbf{b}) = \sum_{n=1}^{N} \left\| \mathbf{g}_\mathbf{y}(\mathbf{y}_n) - \mathbf{f}(\mathbf{g}_\mathbf{x}(\mathbf{x}_n)) \right\|^2 \tag{6.4}$$

which is easier to handle because it involves only $\mathbf{A}$ (not its inverse). This has the disadvantage that if a ground-truth $\mathbf{A}$, $\mathbf{b}$ is used to transform the data, then in general this ground-truth is not a minimizer of F. However, in practice it is very close to a minimizer, and the small bias incurred is a fair price to pay for the simplicity and efficiency of the algorithm. This is further discussed in the following. Also, we do not constrain $\mathbf{A}$ to be invertible in (6.4), as we find this unnecessary in practice.

To minimize F, we compute the gradients ($\operatorname{vec}(\cdot)$ concatenates the columns of its argument into a single column vector)

$$\frac{\partial F}{\partial \operatorname{vec}(\mathbf{A})} = 2 \sum_{n=1}^{N} \mathbf{r}_n^T \mathbf{P}_n \qquad \mathbf{P}_n = \frac{\partial \mathbf{r}_n}{\partial \operatorname{vec}(\mathbf{A})} \tag{6.5}$$

$$\frac{\partial F}{\partial \operatorname{vec}(\mathbf{b})} = 2 \sum_{n=1}^{N} \mathbf{r}_n^T \mathbf{Q}_n \qquad \mathbf{Q}_n = \frac{\partial \mathbf{r}_n}{\partial \operatorname{vec}(\mathbf{b})} \tag{6.6}$$

where $\mathbf{r}_n(\mathbf{A}, \mathbf{b}) = \mathbf{g}_\mathbf{y}(\mathbf{y}_n) - \mathbf{f}(\mathbf{g}_\mathbf{x}(\mathbf{x}_n))$. Using the formula $(\mathbf{C}^T \otimes \mathbf{A}) \operatorname{vec}(\mathbf{B}) = \operatorname{vec}(\mathbf{ABC})$, we obtain

$$\begin{aligned} \mathbf{r}_n &= \mathbf{P}_n \operatorname{vec}(\mathbf{A}) + \mathbf{Q} \operatorname{vec}(\mathbf{b}) - \mathbf{w} \\ \mathbf{P}_n &= \mathbf{y}_n^{\square} \otimes \mathbf{I}_2 - \mathbf{W}(\mathbf{x}_n^{\square} \otimes \mathbf{I}_2) \\ \mathbf{Q}_n &= \mathbf{1}_P \otimes \mathbf{I}_2 - \mathbf{W}(\mathbf{1}_K \otimes \mathbf{I}_2) = \mathbf{Q} \end{aligned}$$

for the linear predictive mapping, and

$$\begin{aligned}
\mathbf{r}_n &= \tilde{\mathbf{y}}_n - \mathbf{W}\Phi(\tilde{\mathbf{x}}_n) - \mathbf{w}\\
\mathbf{P}_n &= \mathbf{y}_n^{\square} \otimes \mathbf{I}_2 + \frac{1}{\sigma^2}\mathbf{W}\,\mathrm{diag}\left(\Phi(\tilde{\mathbf{x}}_n)\right)(\tilde{\mathbf{x}}_n\mathbf{1}_M^T - \mathbf{M})^T(\mathbf{x}_n^{\square} \otimes \mathbf{I}_2)\\
\mathbf{Q}_n &= \mathbf{1}_P \otimes \mathbf{I}_2 + \frac{1}{\sigma^2}\mathbf{W}\,\mathrm{diag}\left(\Phi(\tilde{\mathbf{x}}_n)\right)(\tilde{\mathbf{x}}_n\mathbf{1}_M^T - \mathbf{M})^T(\mathbf{1}_K \otimes \mathbf{I}_2)
\end{aligned}$$

for the RBF predictive mapping, where $\tilde{\mathbf{x}}_n = \mathbf{g}_{\mathbf{x}}(\mathbf{x}_n)$, $\tilde{\mathbf{y}}_n = \mathbf{g}_{\mathbf{y}}(\mathbf{y}_n)$, $\mathbf{M}_{2K\times M} = (\boldsymbol{\mu}_1, \dots, \boldsymbol{\mu}_M)$, $\mathbf{y}^{\square} = (\boldsymbol{y}_1 \cdots \boldsymbol{y}_P)^T$ of $P \times 2$, and $\mathbf{x}^{\square} = (\boldsymbol{x}_1 \cdots \boldsymbol{x}_K)^T$ of $K \times 2$.

The solution for the linear case is unique and given by the following positive definite 6×6 linear system:

$$\begin{pmatrix} \overline{\mathbf{P}^T\mathbf{P}} & \overline{\mathbf{P}}^T\mathbf{Q} \\ \mathbf{Q}^T\overline{\mathbf{P}} & \mathbf{Q}^T\mathbf{Q} \end{pmatrix} \begin{pmatrix} \mathrm{vec}(\mathbf{A}) \\ \mathrm{vec}(\mathbf{b}) \end{pmatrix} = \left(\overline{\mathbf{P}}\ \mathbf{Q}\right)^T \mathbf{w} \tag{6.7}$$

where $\overline{\mathbf{P}} = \frac{1}{N}\sum_{n=1}^N \mathbf{P}_n$ and $\overline{\mathbf{P}^T\mathbf{P}} = \frac{1}{N}\sum_{n=1}^N \mathbf{P}_n^T\mathbf{P}_n$.

The solution for the RBF case requires nonlinear optimization of F using the gradient equations given above. We have found BFGS (a quasi-Newton method with superlinear convergence [101]) to be very efficient and reliable [4]; when initializing it from the $\{\mathbf{A}, \mathbf{b}\}$ obtained for the linear predictive mapping and the Gauss-Newton approximation for the initial Hessian, it converges to machine precision in 5–15 iterations. Random initializations have failed to find local optima in F with our datasets.

The computational complexity of the adaptation algorithms in number of multiplications is $\mathcal{O}(32NP)$ for the linear case and $\mathcal{O}(14NM(P+K))$ per BFGS iteration for the RBF case, where the constants given are approximate and assume 2D-wise vectors. For $N = 10$ adaptation contours, $M = 500$ basis functions, $P = 24$ points and $K = 3$ landmarks, RBF adaptation takes 0.1 s and linear adaptation takes 4 ms in a 2.66 GHz PC.

Bias incurred by the proxy objective function. In general, if the data is transformed with a ground-truth $\{\mathbf{A}_0, \mathbf{b}_0\}$, then the latter is a minimizer of E (up to local optima) but not of F. This is due to the fact that E and F are minimizing errors in different variables (just like the regression lines of y on x and x on y are different). However, we can prove that the bias $(\mathbf{A}, \mathbf{b}) - (\mathbf{A}_0, \mathbf{b}_0)$ is proportional to the predictive error on the original training set (and zero if the model was perfect, i.e., $\mathbf{y}_n = \mathbf{f}(\mathbf{x}_n)$). Our predictive errors are very low, particularly for the RBF case, and in synthetic examples we observe a relative bias of less than 1%. This is a fair price to pay for the simplicity and efficiency of the algorithm. There are two further differences in the RBF case with respect to the linear one that cause the adapted model to be slightly better than expected. (1) The original $\mathbf{f}$ may be slightly suboptimal because of the heuristic RBF training; minimizing F will not only adapt but also indirectly optimize over the centers

---

[4] Gradient descent does not work here because the objective function, near the ground truth minimizer is extremely ill-conditioned. With proper re-scaling datasets, its convergence may be improved.

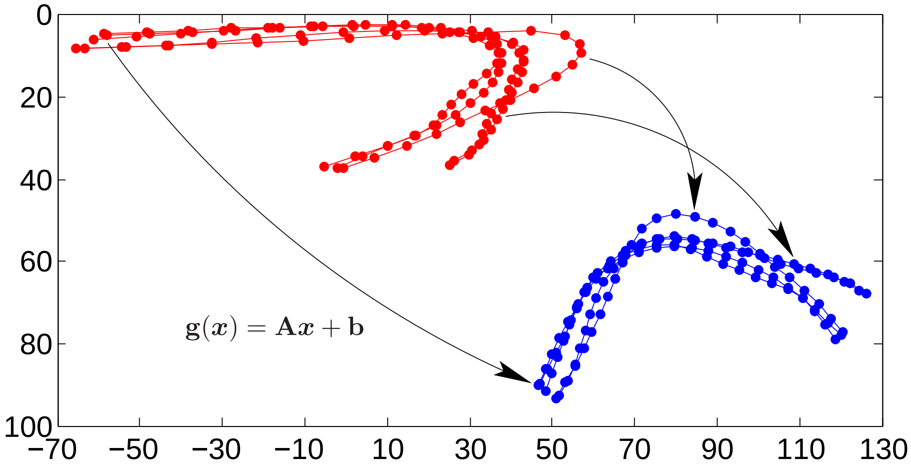


Figure 6.4: Five sample contours from **S1** aligned with a 2D-wise mapping **g** with $\mathbf{A} = \left(\begin{smallmatrix} 0.26 & 1.82 \\ -0.36 & 0.26 \end{smallmatrix}\right)$, $\mathbf{b} = \left(\begin{smallmatrix} 52.8 \\ 67.4 \end{smallmatrix}\right)$. *Red*: adaptation contours; *Blue*: original contours, recovered by **g**.

and width. (2) The adapted RBF model results in non-radial basis functions (with inverse covariance $(\mathbf{I}_K \otimes \mathbf{A}^T\mathbf{A})/\sigma^2$), and is thus more flexible than the original one which has radial basis functions with inverse covariance $\sigma^{-2}\,\mathbf{I}$.

6.4.2 Local feature adaptation

Coupling 2D wise transformation make the *global feature adaptation* algorithm simple, fast and effective, but also restrictive. As shown later, the accuracy of this algorithm deteriorates somewhat when adapting to a completely different speaker; specifically, the prediction error stagnates quickly (with just 5 to 10 adaptation contours) and relatively far from the optimal one that we would achieve if training with abundant data (0.3 to 0.7 mm more, which is over twice that error). There are two basic reasons for this. Firstly, every 2D point in the contour (including the landmarks) undergoes exactly the same linear transformation, resulting in only 6 adaptation parameters. While this works perfectly under translation, rotation, scaling and global shearing, it does not allow for different transformations in different points of the tongue. This is very restrictive, because we should expect more complex variations arising from anatomical factors, sex and age (for example, the new speaker might have a longer tip but a shorter dorsum than the old one) as well as other factors such as speaking style, language, etc. Secondly, in order to simplify the optimization, a proxy objective function is used that introduces a (small) bias in the transformation.

To address above problems, we propose the *local feature adaptation*. The key idea is to use *local, linear feature transformations* at each landmark and at each contour point. This increases the number of parameters and hence the flexibility of the adaptation, which can take advantage of a larger number of adaptation contours and achieve an error much close to the optimal one (0.1 to 0.3 mm more); besides, we optimize the real reconstruction error without bias. In addition, we apply regularization that reduces variance if using very few contours.

Specifically, we adapt an existing predictive mapping $\mathbf{f}$ by estimating two invertible linear mappings $\mathbf{g}_{\mathbf{x}}$ and $\mathbf{g}_{\mathbf{y}}$ (with few parameters) that map new data to old data in the landmark ($\mathbf{x}$) and contour ($\mathbf{y}$) spaces, respectively. Each mapping $\mathbf{g}$ is defined as a concatenation of separate, local linear mappings that map a 2D point to another 2D point:

$$\tilde{\mathbf{x}} = \mathbf{g}_{\mathbf{x}}(\mathbf{x}) = \begin{pmatrix} \mathbf{A}^{\mathbf{x}}_1 \boldsymbol{x}_1 + \mathbf{b}^{\mathbf{x}}_1 \\ \cdots \\ \mathbf{A}^{\mathbf{x}}_K \boldsymbol{x}_K + \mathbf{b}^{\mathbf{x}}_K \end{pmatrix}, \ \tilde{\mathbf{y}} = \mathbf{g}_{\mathbf{y}}(\mathbf{y}) = \begin{pmatrix} \mathbf{A}^{\mathbf{y}}_1 \boldsymbol{y}_1 + \mathbf{b}^{\mathbf{y}}_1 \\ \cdots \\ \mathbf{A}^{\mathbf{y}}_P \boldsymbol{y}_P + \mathbf{b}^{\mathbf{y}}_P \end{pmatrix}. \tag{6.8}$$

The adapted predictive mapping is given by $\mathbf{g}_{\mathbf{y}}^{-1} \circ \mathbf{f} \circ \mathbf{g}_{\mathbf{x}}$ and requires estimating $6(K+P)$ parameters that we write collectively as $(\mathbf{A}^{\mathbf{x}}, \mathbf{b}^{\mathbf{x}}, \mathbf{A}^{\mathbf{y}}, \mathbf{b}^{\mathbf{y}})$. The adapted model is linear if $\mathbf{f}$ was linear, and a basis function network where the basis functions are non-radial if $\mathbf{f}$ was a radial basis function network. Recall that in the global adaptation in section 6.4.1, $\mathbf{A}^{\mathbf{x}}_i = \mathbf{A}^{\mathbf{y}}_j = \mathbf{A}$ and $\mathbf{b}^{\mathbf{x}}_i = \mathbf{b}^{\mathbf{y}}_j = \mathbf{b}$, so there were only 6 parameters.

Objective function. To estimate $(\mathbf{A}^{\mathbf{x}}, \mathbf{b}^{\mathbf{x}}, \mathbf{A}^{\mathbf{y}}, \mathbf{b}^{\mathbf{y}})$, we minimize the predictive squared error $E(\mathbf{A}^{\mathbf{x}}, \mathbf{b}^{\mathbf{x}}, \mathbf{C}^{\mathbf{y}}, \mathbf{d}^{\mathbf{y}})$:

$$\min E(\mathbf{A}^{\mathbf{x}}, \mathbf{b}^{\mathbf{x}}, \mathbf{C}^{\mathbf{y}}, \mathbf{d}^{\mathbf{y}}) = \sum_{n=1}^{N} \left\| \mathbf{y}_n - \mathbf{g}_{\mathbf{y}}^{-1} \mathbf{f}(\mathbf{g}_{\mathbf{x}}(\mathbf{x}_n)) \right\|^2 \tag{6.9}$$

where we introduce new parameters $\mathbf{C}^{\mathbf{y}}_j$, $\mathbf{d}^{\mathbf{y}}_j$, so we work with

$$\mathbf{y} = \mathbf{g}_{\mathbf{y}}^{-1}(\tilde{\mathbf{y}}) = \begin{pmatrix} \mathbf{C}^{\mathbf{y}}_1 \tilde{\mathbf{y}}_1 + \mathbf{d}^{\mathbf{y}}_1 \\ \cdots \\ \mathbf{C}^{\mathbf{y}}_P \tilde{\mathbf{y}}_P + \mathbf{d}^{\mathbf{y}}_P \end{pmatrix} \qquad \begin{aligned} \mathbf{C}^{\mathbf{y}}_j &= (\mathbf{A}^{\mathbf{y}}_j)^{-1} \\ \mathbf{d}^{\mathbf{y}}_j &= -(\mathbf{A}^{\mathbf{y}}_j)^{-1} \mathbf{b}^{\mathbf{y}}_j \end{aligned} \tag{6.10}$$

instead of $\mathbf{g}_{\mathbf{y}}$, simplifying the optimization (no matrix appears as an inverse). In section 6.4.1, we optimized a proxy function $F(\mathbf{A}, \mathbf{b})$:

$$\min_{\mathbf{A},\mathbf{b}} F(\mathbf{A}, \mathbf{b}) = \sum_{n=1}^{N} \left\| \mathbf{g}_{\mathbf{y}}(\mathbf{y}_n) - \mathbf{f}(\mathbf{g}_{\mathbf{x}}(\mathbf{x}_n)) \right\|^2 \tag{6.11}$$

because $E(\mathbf{A}, \mathbf{b})$ must contain both $\mathbf{A}$ and $\mathbf{A}^{-1}$ and its gradient and optimization are more complicated. Our new approach has several advantages over this (apart from being more flexible): (1) As mentioned in section 6.4.1, the $(\mathbf{A}, \mathbf{b})$ that minimize F differ somewhat from those optimizing E and are thus suboptimal. (2) Optimizing E in the new approach is quite simpler because the parameters of $\mathbf{g}_{\mathbf{x}}$ and $\mathbf{g}_{\mathbf{y}}$ are decoupled (see gradients below), in fact E separates over each $(\mathbf{C}^{\mathbf{y}}_j, \mathbf{d}^{\mathbf{y}}_j)$ for fixed $(\mathbf{A}^{\mathbf{x}}, \mathbf{b}^{\mathbf{x}})$. Besides, the function F is not useful with the new parameters because it has a trivial solution: setting

$\mathbf{A}^{\mathbf{x}}$, $\mathbf{b}^{\mathbf{x}}$ and $\mathbf{C}^{\mathbf{y}}$ to zero then each term in F is a constant that $\mathbf{d}^{\mathbf{y}}$ can pick up, so $F = 0$. However, the local transformation approach does not carry over to the case where the adaptation data contains only partial contours (see section 6.5) because then we have no data to fit $(\mathbf{C}^{\mathbf{y}}_j, \mathbf{d}^{\mathbf{y}}_j)$, for $j = 1, \dots, P$ and they play no role in (6.9).

Optimizing E**.** The gradients of E are ($\operatorname{vec}(\cdot)$ concatenates the columns of its argument into a single column vector)

$$\begin{aligned}
\frac{\partial E}{\partial \operatorname{vec}(\mathbf{A}^{\mathbf{x}})} &= 2\sum_{n=1}^{N} \mathbf{r}_n^T \mathbf{P}^{\mathbf{x}}_n & \mathbf{P}^{\mathbf{x}}_n &= \frac{\partial \mathbf{r}_n}{\partial \operatorname{vec}(\mathbf{A}^{\mathbf{x}})} \\
\frac{\partial E}{\partial \operatorname{vec}(\mathbf{b}^{\mathbf{x}})} &= 2\sum_{n=1}^{N} \mathbf{r}_n^T \mathbf{Q}^{\mathbf{x}}_n & \mathbf{Q}^{\mathbf{x}}_n &= \frac{\partial \mathbf{r}_n}{\partial \operatorname{vec}(\mathbf{b}^{\mathbf{x}})} \\
\frac{\partial E}{\partial \operatorname{vec}(\mathbf{C}^{\mathbf{y}})} &= 2\sum_{n=1}^{N} \mathbf{r}_n^T \mathbf{P}^{\mathbf{y}}_n & \mathbf{P}^{\mathbf{y}}_n &= \frac{\partial \mathbf{r}_n}{\partial \operatorname{vec}(\mathbf{C}^{\mathbf{y}})} \\
\frac{\partial E}{\partial \operatorname{vec}(\mathbf{d}^{\mathbf{y}})} &= 2\sum_{n=1}^{N} \mathbf{r}_n^T \mathbf{Q}^{\mathbf{y}}_n & \mathbf{Q}^{\mathbf{y}}_n &= \frac{\partial \mathbf{r}_n}{\partial \operatorname{vec}(\mathbf{d}^{\mathbf{y}})}
\end{aligned}$$

where $\mathbf{r}_n(\mathbf{A}^{\mathbf{x}}, \mathbf{b}^{\mathbf{x}}, \mathbf{C}^{\mathbf{y}}, \mathbf{d}^{\mathbf{y}}) = \mathbf{y}_n - \operatorname{diag}\left(\mathbf{C}^{\mathbf{y}}_1, \dots, \mathbf{C}^{\mathbf{y}}_P\right) \mathbf{z}_n - \operatorname{vec}(\mathbf{d}^{\mathbf{y}})$ and $\mathbf{z}_n = (\boldsymbol{z}_{n1}^T, \dots, \boldsymbol{z}_{nP}^T)^T = \mathbf{f}(\mathbf{g}_{\mathbf{x}}(\mathbf{x}_n))$. For the linear mapping function we obtain ($\otimes$ is the Kronecker product)

$$\frac{\partial \mathbf{r}_n}{\partial \operatorname{vec}(\mathbf{A}^{\mathbf{x}})} = -\operatorname{diag}\left(\mathbf{C}^{\mathbf{y}}_j\right) \mathbf{W} \operatorname{diag}(\boldsymbol{x}_{n1}^T \otimes \mathbf{I}_2, \dots, \boldsymbol{x}_{nK}^T \otimes \mathbf{I}_2) \tag{6.12}$$

$$\frac{\partial \mathbf{r}_n}{\partial \operatorname{vec}(\mathbf{b}^{\mathbf{x}})} = -\operatorname{diag}\left(\mathbf{C}^{\mathbf{y}}_j\right) \mathbf{W} \tag{6.13}$$

$$\frac{\partial \mathbf{r}_n}{\partial \operatorname{vec}(\mathbf{C}^{\mathbf{y}})} = -\operatorname{diag}(\boldsymbol{z}_{n1}^T \otimes \mathbf{I}_2, \dots, \boldsymbol{z}_{nP}^T \otimes \mathbf{I}_2) \tag{6.14}$$

$$\frac{\partial \mathbf{r}_n}{\partial \operatorname{vec}(\mathbf{d}^{\mathbf{y}})} = -\mathbf{I}_{2P} \tag{6.15}$$

and for the RBF mapping we obtain (notation as in section 6.4.1)

$$\frac{\partial \mathbf{r}_n}{\partial \operatorname{vec}(\mathbf{A}^{\mathbf{x}})} = \frac{1}{\sigma^2} \operatorname{diag}\left(\mathbf{C}^{\mathbf{y}}_j\right) \mathbf{W} \operatorname{diag}\left(\Phi'_n\right) (\tilde{\mathbf{x}}_n \mathbf{1}_M^T - \mathbf{M})^T \operatorname{diag}\left(\boldsymbol{x}_{ni}^T \otimes \mathbf{I}_2\right) \tag{6.16}$$

$$\frac{\partial \mathbf{r}_n}{\partial \operatorname{vec}(\mathbf{b}^{\mathbf{x}})} = \frac{1}{\sigma^2} \operatorname{diag}\left(\mathbf{C}^{\mathbf{y}}_j\right) \mathbf{W} \operatorname{diag}\left(\Phi'_n\right) (\tilde{\mathbf{x}}_n \mathbf{1}_M^T - \mathbf{M})^T \tag{6.17}$$

and the same formulas for $(\mathbf{C}^{\mathbf{y}}, \mathbf{d}^{\mathbf{y}})$. The solution for both linear and RBF cases requires nonlinear optimization of E using these gradient equations. As in section 6.4.1, we found BFGS to be effective and reliable. E has local optima and we initialize BFGS from the solution obtained by the global adaptation method.

BFGS constructs approximate inverse Hessian matrices of order $6(P + K)$, so it will not work if P is large (consider a detailed 2D tongue shape representation of $P = 100 \times 100 = 10^4$ points in 3D). The fact that E decouples over each $(\mathbf{C}^{\mathbf{y}}_j, \mathbf{d}^{\mathbf{y}}_j)$ for fixed $(\mathbf{A}^{\mathbf{x}}, \mathbf{b}^{\mathbf{x}})$ suggests alternating minimization of E:

1. Fix $(\mathbf{A}^{\mathbf{x}}, \mathbf{b}^{\mathbf{x}})$ (thus fixing $\mathbf{f}(\mathbf{g}_{\mathbf{x}}(\mathbf{x}_n))$) and minimize E over each $(\mathbf{C}^{\mathbf{y}}_j, \mathbf{d}^{\mathbf{y}}_j)$. Since E is linear over the latter, the unique solution is given by P linear systems of 6×6.
2. Fix $(\mathbf{C}^{\mathbf{y}}, \mathbf{d}^{\mathbf{y}})$ and minimize E over $(\mathbf{A}^{\mathbf{x}}, \mathbf{b}^{\mathbf{x}})$ with BFGS. This requires matrices of order $6K$ only.

A disadvantage of the alternating optimization is that it converges very slowly. We use the full BFGS in our experiments.

Regularizing E**.** As in section 6.5, we can penalize $\mathbf{A}^{\mathbf{x}}$ and $\mathbf{C}^{\mathbf{y}}$ with large condition numbers by adding the following term to E:

$$\lambda \mathcal{C}(\mathbf{A}^{\mathbf{x}}, \mathbf{C}^{\mathbf{y}}) = \lambda \left(\sum_{i=1}^{K} C(\mathbf{A}^{\mathbf{x}}_i) + \sum_{i=1}^{P} C(\mathbf{C}^{\mathbf{y}}_i) \right) \tag{6.18}$$

$$\lambda > 0, \quad C(\mathbf{A}) = \operatorname{tr}(\mathbf{A}^T\mathbf{A}) - D(\det(\mathbf{A}^T\mathbf{A}))^{1/D}. \tag{6.19}$$

With very little adaptation data ($N = 10$), this increases robustness to misspecification of landmarks and reduces overfitting (see below the reason for this regularization term).

6.5 Feature adaptation of predictive models with partial contours

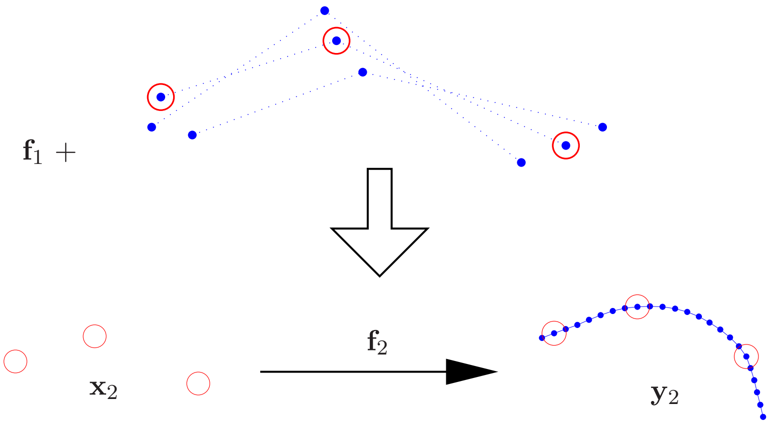


Figure 6.5: Adapting $\mathbf{f}_1$ for speaker 1 to speaker 2 given partial contours containing only the landmark positions.

Now, we consider a more challenging adaptation task (see fig. 6.5): given as adaptation data for a new speaker not the full contours with P points but only the much sparser K–landmark contours (N of them), estimate the predictive mapping $\mathbf{f}_2$ as before. Thus, we have no training data or ground truth for the remaining $P - K$ points at all. This is the problem with e.g. the MOCHA database, which contain the 2D locations of $K = 3$ pellets over time during speech for several unknown speakers, but not a single full contour. Given this information alone, how can we reconstruct the full tongue contour from the K points for unseen speakers? We propose an extension of our *global feature adaptation* by considering as input $\mathbf{x}$ and also as output $\mathbf{y} = \mathbf{x}$ the pellet coordinates in these databases. Note that the *local feature adaptation* does not carry over in this case because we have no data to fit $(\mathbf{C}_j^{\mathbf{y}}, \mathbf{d}_j^{\mathbf{y}})$. We define the new problem (minimized with BFGS):

$$\min_{\mathbf{A},\mathbf{b}} F_{\mathbf{x}}(\mathbf{A},\mathbf{b}) = \sum_{n=1}^{N} \|\mathbf{g}_{\mathbf{x}}(\mathbf{x}_n) - \mathbf{f}_{\mathbf{x}}(\mathbf{g}_{\mathbf{x}}(\mathbf{x}_n))\|^2 \tag{6.20}$$

where $\mathbf{f}_{\mathbf{x}}$ is the components extracted from $\mathbf{f}$ corresponding to the K landmarks. This is equivalent to seeking $\{\mathbf{A}, \mathbf{b}\}$ such that the adapted model $\mathbf{g}_{\mathbf{x}}^{-1} \circ \mathbf{f}_{\mathbf{x}} \circ \mathbf{g}_{\mathbf{x}}$ best approximates the identity mapping and interpolates the landmarks. We then apply $\{\mathbf{A}, \mathbf{b}\}$ to reconstruct the entire contour as $\mathbf{g}_{\mathbf{y}}^{-1} \circ \mathbf{f} \circ \mathbf{g}_{\mathbf{x}}$.

Note this approach does not work if $\mathbf{f}$ is linear, because then $\mathbf{f}_{\mathbf{x}}$ became the identity when minimizing $E(\mathbf{f}) = \sum_{n=1}^{N'} \|\mathbf{y}_n - \mathbf{f}(\mathbf{x}_n)\|^2$ during training, and $F_{\mathbf{x}}(\mathbf{A},\mathbf{b}) = 0$ in (6.20) for any $\{\mathbf{A}, \mathbf{b}\}$. In contrast, a Gaussian RBF with a finite number of basis functions approximates the identity to high but not perfect accuracy, and only within a finite domain of the input $\mathbf{x}$, thus (6.20) has a well-defined minimum that implicitly aligns the new speaker's input with the domain of the old speaker's one. Also note that *we do not need correspondences*, i.e., pairs of inputs of the old and new speakers corresponding to the same sound; achieving such correspondences is not only time-consuming but also ill defined, as it is not clear what sounds from both speakers should be considered the "same".

We have found an additional problem when applying the method proposed to our ultrasound data. Essentially, our models (trained and adapted) are as good as the data used to train the predictive mapping $\mathbf{f}$. When tracking the tongue contour in ultrasound images, it is very difficult to detect compression or stretching of the tongue because the air-tongue interface is featureless (and the tip or back of the tongue can partially disappear)—a situation similar to the aperture problem in computer vision. Thus, our training contours show mostly equidistant contour points, and we observe that a (small) proportion of the MOCHA frames show distances between pellets differing by up to 30%. If the adaptation data contains such frames, the adapted model can be far from the best one (fig. 6.10**a**). Note this problem is caused not by our adaptation algorithm but by our contour data, and the ultimate solution would be to collect tongue contours that show compression and stretching as naturally occurring during speech (perhaps attaching metal pellets to the subject's tongue with ultrasound imaging). It is possible to use only MOCHA frames with roughly equidistant pellets (fig. 6.10**b**), but this discards useful adaptation data and is unreliable. However, we have found one way of achieving very good overall adaptation with our existing data: to regularize problem (6.20) to encourage $\mathbf{A}$ to have a low condition number. This works because much of the misalignment between speakers can be explained by a scaling and rigid motion (which has $\operatorname{cond}(\mathbf{A}) = 1$), and we do observe that poorly adapted models obtained when using all sorts of MOCHA frames indeed yield a poorly conditioned $\mathbf{A}$. We then solve

$$\min_{\mathbf{A},\mathbf{b}} F_{\mathbf{x}}(\mathbf{A},\mathbf{b}) + \lambda C(\mathbf{A}), \quad \lambda \geq 0. \tag{6.21}$$

Directly minimizing $C(\mathbf{A}) = \mathrm{cond}\,(\mathbf{A}) = \|\mathbf{A}\|_2 \, \|\mathbf{A}^{-1}\|_2$ is difficult, so we use instead the much simpler

$$C(\mathbf{A}) = \mathrm{tr}\,(\mathbf{A}^T\mathbf{A}) - D \det{(\mathbf{A}^T\mathbf{A})}^{1/D} \quad \text{for } \mathbf{A}_{D\times D}, \tag{6.22}$$

which satisfies $C(\mathbf{A}) \geq 0$ and $C(\mathbf{A}) = 0$ iff $\mathrm{cond}\,(\mathbf{A}) = 1$ (so it is minimal when $\mathrm{cond}\,(\mathbf{A})$ is minimal), and is piecewise quadratic for $D = 2$. In our experiments we find that, for a wide range of λ, this method reliably obtains the best results of all options and realistically reconstructs the full tongue contour (within and beyond the MOCHA pellets) for most frames. Frames with significantly non-equidistant pellets do show distortions, but this is unavoidable with our data.

Before training the predictive model and running the adaptation algorithm, we need to determine which of the P contour points are the K landmarks so that this matches as closely as possible the landmark locations in the new dataset. MOCHA and XRMB give approximate information as to how the pellets were attached to the tongue (e.g. “2 mm from the tongue tip”) that can be used for this purpose. The location of the reconstructed tongue relative to the velum/teeth/palate can also be used to refine this estimate.

The computational complexity of the adaptation algorithm per BFGS iteration is $\mathcal{O}(NMK)$ with N adaptation contours, M radial basis functions and K landmarks. Convergence occurs in around 10 iterations.

6.6 Experimental results

In this section, we study performance of our adaptation algorithms on practical speaker adaptation tasks. In particular, we try to answer following questions: (1) How are they compared with retraining? That is, minimize the predictive error $E(\mathbf{f})$ of (5.1) over all parameters of $\mathbf{f}$. (2) How robust is our methods to choice of adaptation data?. (3) How realistic is the reconstructed VT from a few landmarks?

6.6.1 Speaker adaptation of predictive models of tongue shapes

In this experiment, we aim to quantify how well our global and local adaptation methods perform on a real speaker adaptation task.

Dataset. We use data from `maaw0` and `feal0` in the ultrasound database as in chapter 5. Fig. 6.6 shows the contour datasets from both speakers, which display significant differences in shape. (Some differences in the tongue root of `feal0` are due to its being poorly visible in the ultrasound; this poses an additional challenge for the adaptation algorithms.)

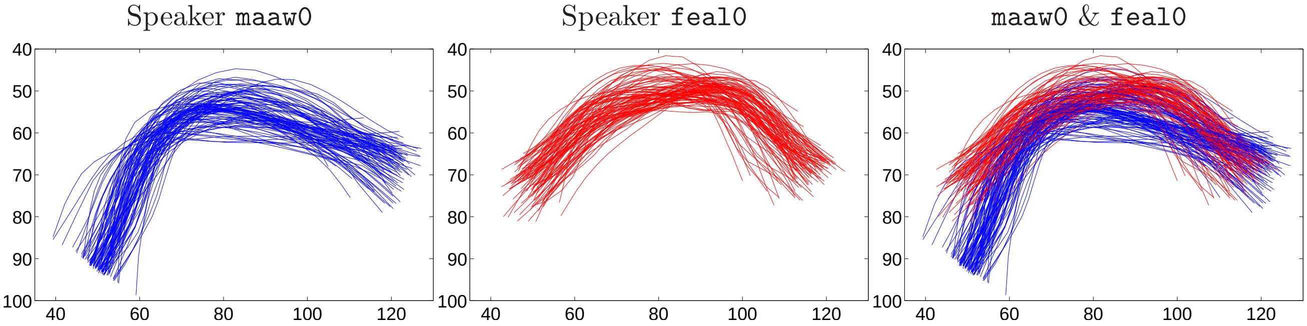


Figure 6.6: Contour datasets for two speakers. Only a subset of contours plotted to avoid clutter.

Predictive models. We use (1) an RBF mapping with $M = 500$ basis functions, width $\sigma = 55$ and regularization parameter $\lambda = 10^{-4}$, trained by cross-validation on the 2 236 contours of **S1**; and (2) a linear mapping, given as a baseline (as it is consistently outperformed by the RBF mapping); we also use it to obtain initial $\{\mathbf{A}, \mathbf{b}\}$ for the RBF. The K landmarks were chosen optimally from the P contour points as in section 5.5.3.

Adaptation task. We adapt a predictive model for `maaw0` (learned on 2 236 tongue contours from its first session) to `feal0`. We use up to 500 contours from `feal0` for adaptation/retraining and the remaining 2 409 contours for testing.

Comparison methods. We compare our local and global feature adaptation with (1) *retraining* the predictive mapping from scratch on the adaptation data and (2) *PCA alignment*, a simple adaptation method that uses neither correspondences nor the predictive mapping $\mathbf{f}$; it finds $\{\mathbf{A}, \mathbf{b}\}$ by matching the mean and covariance (principal axes' angle and variance) of the original and the adaptation datasets, each considered as a collection of 2D points (i.e., all the points from all the contours). Note care is needed to set the sign of the principal axes' angle to ensure correct alignment. We initialize the local method with the parameters of the global one. The optimal baseline is achieved by retraining the predictive model with abundant data. All the error values we quote are RMSE predictive errors E per contour point in mm on the test data.

Results. Figs. 6.8–6.9 plot the errors after adaptation/retraining as a function of the number of adaptation contours N, and the number of landmarks K. Using the predictive model of `maaw0` directly on `feal0` without adaptation would incur an error > 2 mm. With global adaptation, the RBF beats the linear $\mathbf{f}$ consistently by over 0.4 mm as in section 6.4.1. With just one contour, the RBF achieves an error of 0.9 mm robustly (note the tight errorbars). However, while the error decreases with N, it stagnates when $N = 10$ far from reaching the optimal value (retraining with abundant data). The performance gap is around 0.3 mm. With local adaptation, both the linear and RBF $\mathbf{f}$ work very well with $N \geq 20$ contours, consistently and significantly outperforming the global adaptation (see fig. 6.7 for visual comparisons). Surprisingly, the adaptation error of the linear and RBF cases are now comparable. With $N < 7$ to 20 contours, the local adaptation is less stable and has an average error larger than the global one. This is likely an overfitting effect, since the local method has now more parameters. The error decreases with N, stagnating around $N = 50$ but very close to the ground truth (less than 0.1 mm worse). PCA alignment (not shown in the figures) is consistently worse than both global and local adaptation and with a larger variance even for larger N. Retraining catches up local adaptation for $N \approx 50$ to 90 contours and is essentially useless for $N < 20$.

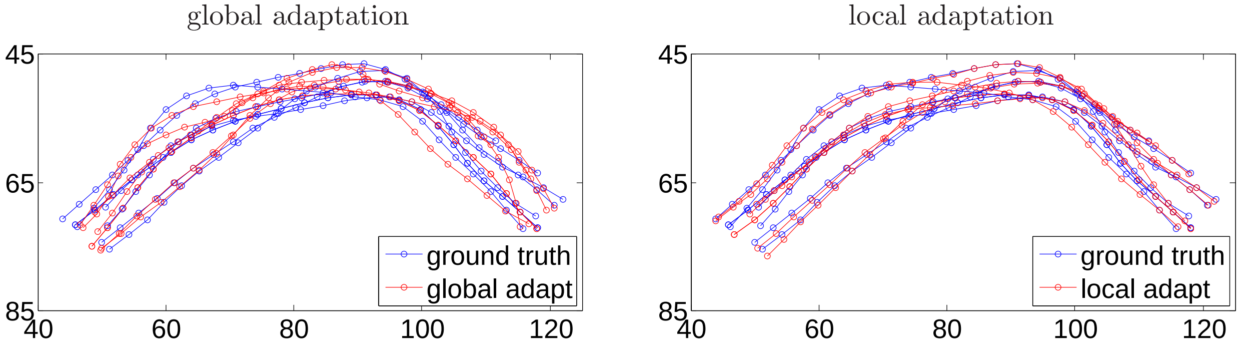


Figure 6.7: Contour dataset for `maaw0` aligned to that for `feal0` ($K = 3$). Only a subset of contours plotted to avoid clutter.

From fig. 6.9, as K increases, the predictive error decreases (it is easier to reconstruct the contour given more landmarks), and the adaptation error closely tracks it. The local adaptation consistently beats the global one by 40% across K if using $N > 20$ contours. The articulatory databases use $K = 3$ (MOCHA) and $K = 4$ (XRMB), and the advantage of the local adaptation over the global one is strongest in this region.

Suitable amounts of regularization (linear: $\lambda = 10$, RBF: $\lambda = 10^4$) reduce the error for the local adaptation (RBF in particular) if using very few contours. Global adaptation benefits marginally from regularization.

In summary, using data from two speakers (`maaw0` and `feal0`) with significantly different shapes, our experiments show the global method gives a reasonable adaptation but stagnates with as few as 5 to 10 contours. The local method keeps reducing the error with more contours and stagnation happens only with many more contours, producing reconstruction results close to retraining the predictive model

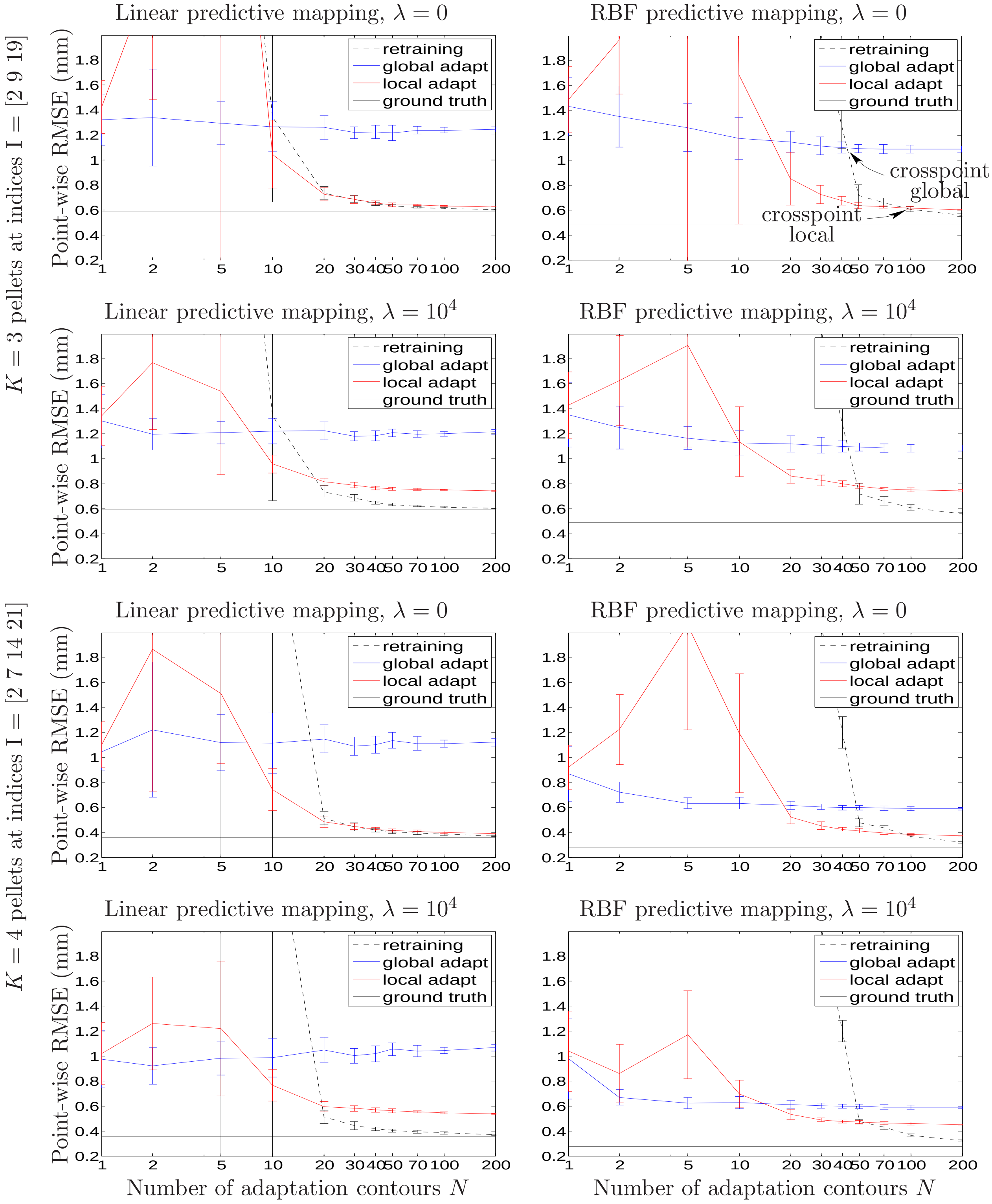


Figure 6.8: Predictive error E (as RMSE per contour point in mm) after adaptation as a function of the number of adaptation contours N (for given (K, λ)). Errorbars over 10 random choices of the N adaptation contours. Note the crosspoints with the retraining curve.

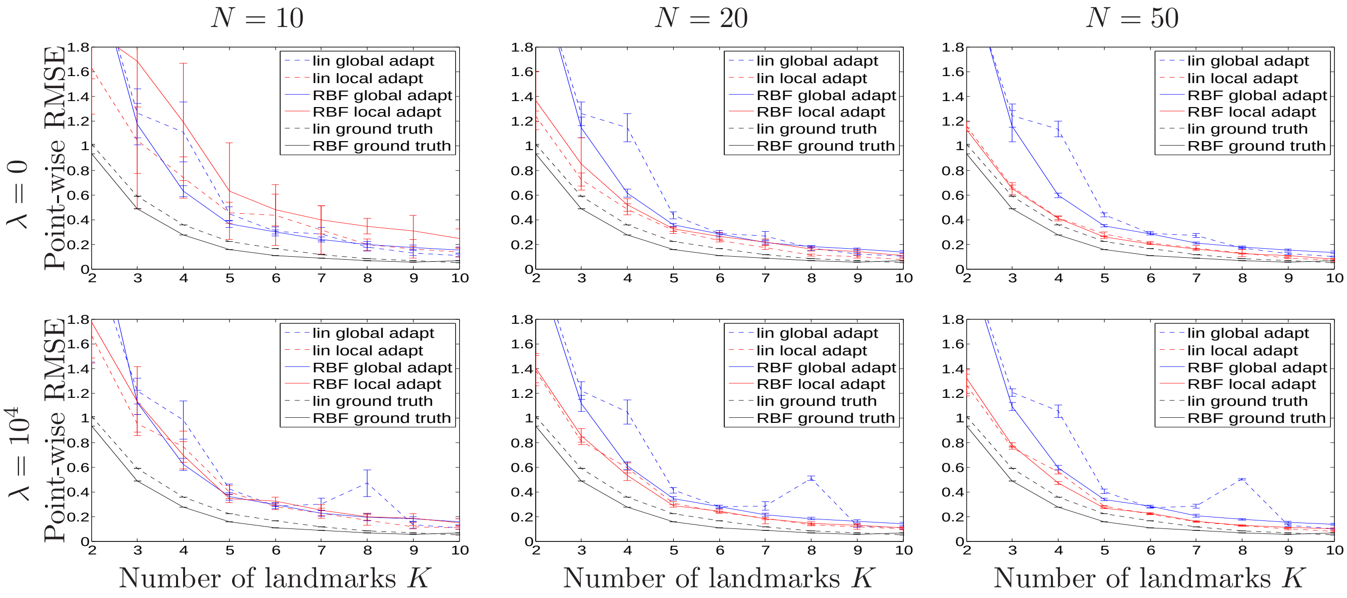


Figure 6.9: As fig. 6.8 but as a function of the number of landmarks K (for given (N, λ)).

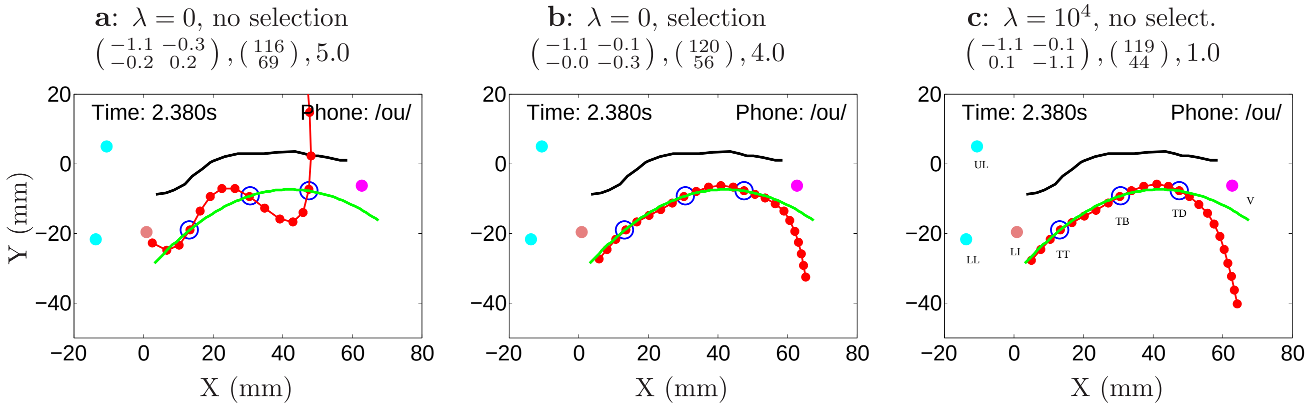
for the new speaker on abundant data. With very few contours ($N < 10$), the local method needs regularization to reduce its variance, and performs worse than the global one. With more than 50 contours, retraining is the better option. Thus, the user has options to guide data collection and achieve the best result in each application. All these statements hold for various numbers of landmarks K. The local adaptation method takes 3 (linear) and 10 (RBF) minutes of CPU time in a workstation for $N = 50$ and $K = 3$.

6.6.2 Reconstruction of the full tongue contour from EMA/X-Ray microbeam

This experiment aims to quantify how well our adaptation method could reconstruct the full tongue contour from only a few tongue pellets for MOCHA and XRMB. The roles of adaptation in these tasks include: (1) compensating session misalignment since data for target and reference speakers were usually recorded in different experimental settings, e.g. face orientation, coordinate systems. Even for the same speaker, adaptation is still necessary to compensate those misalignment caused by different setups. Our adaptation algorithm is found to be very effective to correct session misalignment [111]; (2) compensating geometrical differences between different speakers. We use the same RBF predictive model as in previous experiments (trained with abundant data).

Figures 6.10–6.11 show results for MOCHA (speaker `fsew0`), which has $K = 3$ tongue pellets. We estimated their locations in our predictive model as [4 9 14], as this gave visually the best results.

Effect of regularization and data selection. Fig. 6.10 shows results using $N = 3\,600$ partial contours for adaptation. In fig. 6.10**a** the N contours were randomly selected from the MOCHA database and no regularization was used ($\lambda = 0$). The reconstructed contour oscillates wildly, its ends are too long and it can even appear upside-down; note the diagonal of $\mathbf{A}$ has very different values of opposite sign. In fig. 6.10**b** we first eliminated all MOCHA contours having inter-pellet distance below a certain threshold (see section 6.5) and randomly selected N partial contours from these for adaptation without regularization ($\lambda = 0$). The reconstructed contours are now better, but the result is sensitive to the threshold used and we lose useful adaptation data. In fig. 6.10**c** we used contours without selecting them (as in 6.10**a**), and regularization with $\lambda = 10^4$ (for this dataset, $\lambda \in [10^2, 10^4]$ gave similar results). The reconstructed contours are the best. The resulting $\{\mathbf{A}, \mathbf{b}\}$ are essentially a translation and uniform scaling, as one might expect. These conclusions hold over different choices of the N contours and the value of N, and demonstrate the need for regularization. The experiments below use a predictive model adapted with $N = 10^4$ contours and regularization.


a: $\lambda = 0$, no selection $\begin{pmatrix} -1.1 & -0.3 \\ -0.2 & 0.2 \end{pmatrix}, \begin{pmatrix} 116 \\ 69 \end{pmatrix}, 5.0$

b: $\lambda = 0$, selection $\begin{pmatrix} -1.1 & -0.1 \\ -0.0 & -0.3 \end{pmatrix}, \begin{pmatrix} 120 \\ 56 \end{pmatrix}, 4.0$

c: $\lambda = 10^4$, no select. $\begin{pmatrix} -1.1 & -0.1 \\ 0.1 & -1.1 \end{pmatrix}, \begin{pmatrix} 119 \\ 44 \end{pmatrix}, 1.0$

Figure 6.10: Effect of regularization and data selection in adaptation in MOCHA for a given frame (the same in all 3 plots). **a**: no regularization, randomly selected adaptation set. **b**: no regularization, carefully selected adaptation set. **c**: adaptation with regularization $\lambda = 10^4$, randomly selected adaptation set. Color scheme as fig. 6.11; $\mathbf{A}$, $\mathbf{b}$ and $\operatorname{cond}(\mathbf{A})$ over each plot.

Realistic tongue contour reconstruction. Fig. 6.11 shows representative reconstructed tongue contours for MOCHA. Although, by the very nature of our goal, we do not have ground truth full contours from the MOCHA speaker to compare with, we do have strong indirect evidence that our reconstructions are quite realistic: (1) The reconstructed contours interpolate well the 3 input pellets, and they respect physical constraints (even though we did not impose this in any way when estimating the model): the tongue very rarely goes through the palate, velum or lower incisor (6% of all 10 000 frames of 460 utterances we tested, and then by less than 1 mm); see also fig. 6.13. (2) Comparing visually our contours with those from the ultrasound database (fig. 6.12) shows similar shapes, in particular in the back of the tongue (beyond the innermost pellet): “pick” (fig. 6.11) and frame 177/`maaw0_177` (fig. 6.12); “overall” (fig. 6.11) and frame 300/`maaw0_054` (fig. 6.12). (3) The contours correlate well with the phoneme articulation. Note how precisely reconstructed is the posterior tongue-palate contact in “pick” and the narrow alveolar constriction in “overall” and “thieves”; see also fig. 6.13. This information, which is crucial for speech production and possibly for articulatory synthesis and inversion, is not readily visible from the pellet locations alone (cf. [155]).

We do obtain less realistic reconstructions for a small proportion of frames, usually those having a small inter-pellet distance, or that are not well represented in our ultrasound dataset. We think this could be improved by collecting a more comprehensive contour dataset, without the need for changes in the algorithm.

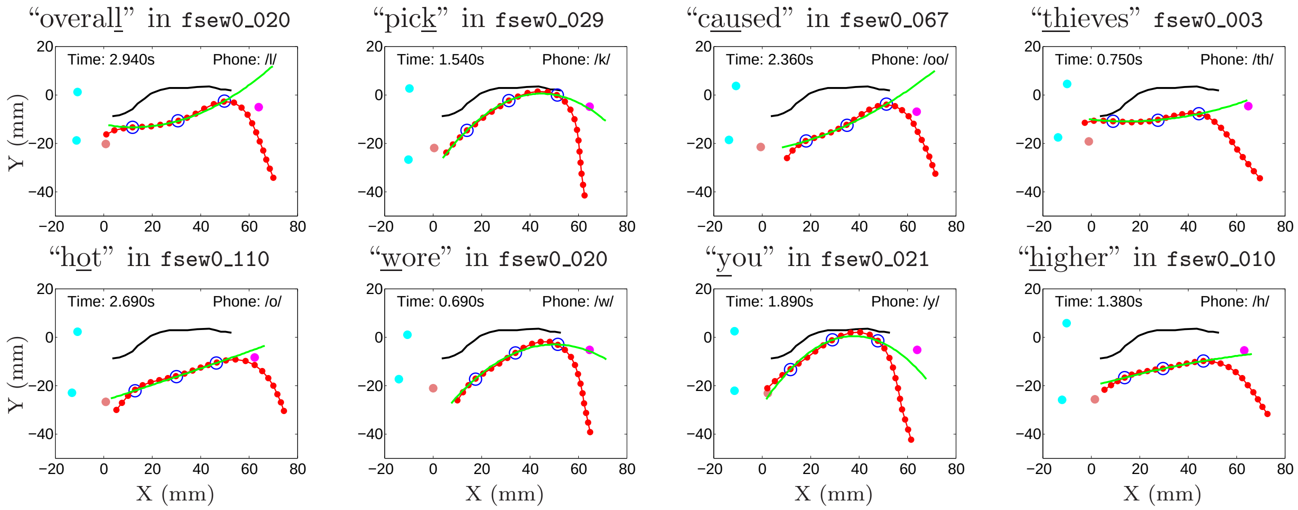


Figure 6.11: Realistic tongue reconstruction for MOCHA; $\lambda = 10^4$, $\mathbf{A} = \left(\begin{smallmatrix} -1.13 & -0.07 \\ 0.06 & -1.05 \end{smallmatrix}\right)$, $\mathbf{b} = \left(\begin{smallmatrix} 121 \\ 48 \end{smallmatrix}\right)$, $\operatorname{cond}(\mathbf{A}) = 1.07$. Black curve: estimate of the palate computed as the convex hull of all the tongue pellets in the entire MOCHA data for speaker `fsew0`. Red curve: reconstructed tongue contour. Green curve: contour reconstructed by a cubic spline. The markers show the EMA pellets (tongue: open blue; lips: cyan; lower incisor: brown; velum: magenta). Lips to the left. See utterance animations, and Matlab packages MOCHAtools/XRMBtools that implement the tongue reconstruction algorithm for the MOCHA/XRMB databases, at `http://faculty.ucmerced.edu/mcarreira-perpinan`.

Comparison with an interpolating spline. Fig. 6.10–6.11 (MOCHA) and 6.13 (XRMB) also show the reconstruction using a cubic interpolating spline (green curve). (For XRMB speaker `jw11`, which uses $K = 4$ tongue pellets, we chose landmarks [3 7 11 15]; other parameters as for MOCHA.) Although the spline can often give a reasonable contour between the pellets, beyond them it generally looks completely unrealistic (e.g. see “overall” or “caused” in fig. 6.11). The spline can also oscillate wildly

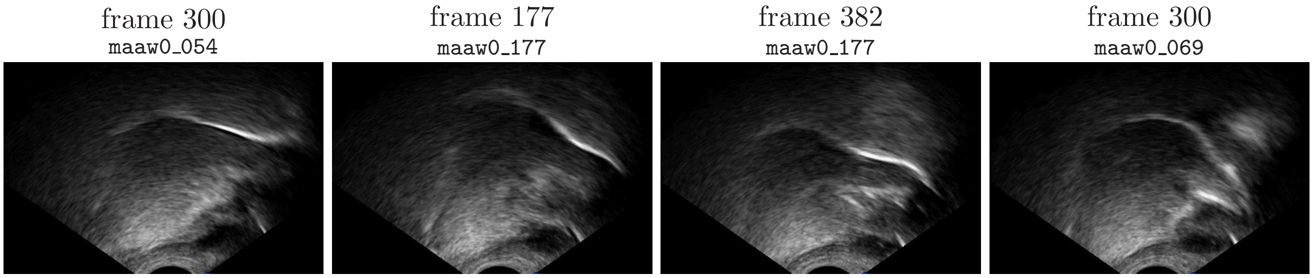


Figure 6.12: Typical tongue shapes during normal speech production in the ultrasound database (lips to the right).

between the pellets as in fig 6.13. Constraining the spline a priori not to go through the palate, velum or teeth seems very difficult, while such constraints are implicitly learned in our data-driven approach.

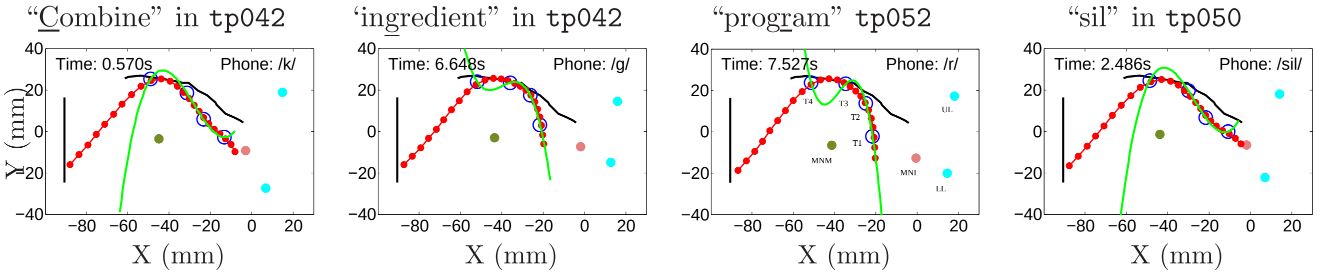


Figure 6.13: Snapshots of realistic tongue reconstructions for XRMB, $\lambda = 10^4$, $\mathbf{A} = \begin{pmatrix} 1.07 & -0.46 \\ -0.15 & -0.67 \end{pmatrix}$, $\mathbf{b} = \begin{pmatrix} 130 \\ 62 \end{pmatrix}$, $\operatorname{cond}(\mathbf{A}) = 1.75$. Color scheme as in fig. 6.11, but the palate was actually traced from a mold from the speaker. Lips to the right.

6.7 Discussion

Our adaptation methods do not need correspondences, that is, contour pairs $(\mathbf{y}_1, \mathbf{y}_2)$ from recording sessions 1 and 2 (or speakers 1 and 2) that correspond to the same sound or phonetic gesture. Accurate correspondences are hard to obtain; although one might ask speaker 2 to utter a few prototype sounds and match these to equivalent ones in speaker 1, inaccuracies will arise, and besides it may be hard for speaker 2 to understand and produce exactly the sound requested. If we did have correspondences, one might simply fit a 2D-wise alignment mapping $\mathbf{g}$ to the corresponding pairs directly. However, this ignores the existing predictive model and would likely perform less well than our method; our real objective is to reduce the predictive error rather than find the best alignment of the contours.

Our global adaptation can be seen as speaker normalization using linear transformations $\mathbf{g}_\mathbf{x}$, $\mathbf{g}_\mathbf{y}$ of a specific structure, namely replicating an unknown 2×2 block $\mathbf{A}$ along their diagonal, and having extremely few free parameters altogether (6, independently of the contour size). This indirectly ties all the RBF parameters (weights, centers, widths), but it is more intuitive to do this in the feature space because of the 2D-wise structure of the tongue contours. Also, our problem is not density estimation by maximum likelihood, but least-squares regression with linear or RBF models, and this requires using both $\mathbf{g}$ and $\mathbf{g}^{-1}$.

Although we illustrate our adaptation algorithms with 2D tongue contours, all the equations carry over to data structured in 3D blocks, and can be extended to adapt predictive models of the 3D tongue shape. Also, our adaptation framework is very flexible and can be extended naturally to handle adaptation with incomplete contours that contain variable missing patterns from a new speaker.

6.8 Summary

We have presented two novel feature adaptation based methods for adapting a tongue model to a new speaker. The **global adaptation**, estimates a global 2D-wise linear transformation for all contour points. It runs extremely fast (e.g. <1 second for adapting a RBF model) and can be used to correct misalignments between different ultrasound recording sessions. Furthermore, it can be extended to handle important applications where only partial adaptation contours are available. We have demonstrated that such extension can recover realistic tongue contours for articulatory databases, based only on the 2D coordinates for the tongue pellets provided in the latter (without the need for correspondences, full contours or any other information like MRI images of VT, audio, geometric articulatory models). The reconstructed tongue satisfies physical constraints (e.g. not going through the palate, teeth or velum) without having to apply the latter explicitly, and provides detailed information not readily available in the database such as the precise location of tongue-palate constrictions. This could be very useful for speech production, articulatory synthesis and inversion. The **local adaptation**, improves the first one by considering local transformations that align each contour point separately. This makes the method more flexible and eliminates its estimation bias. It asymptotes close to the one retrained with abundant data, and distinctly outperforms retraining and the global method when the number of adaptation contours is not very small (10 to 50). Thus the user should use the global, local, or retraining methods with less than 10, 10 to 50, and more than 50 contours, respectively. Finally, all adaptation methods are applicable to any 2D or 3D shapes and thus open the door for reconstructing the entire vocal tract shape of an unknown speaker from a few landmarks on it, provided one can train a predictive model for a reference speaker using data for the full vocal tract of the latter (recorded with e.g. MRI or X-ray).

Chapter 7

Estimating missing data sequences in X-ray microbeam recordings

7.1 Missing data in X-ray microbeam recordings

In this chapter, we consider one particular problem, mistracked or missing pellets, that affects techniques (such as X-ray microbeam or EMA) that are based on tracking over time the 2D or 3D positions of pellets attached to the tongue, lips and other articulators (see fig. 7.1). These techniques, often through publicly available databases such as MOCHA or XRMB, are widely used in work in articulatory speech processing, speech production, and this thesis. Mistracks in these techniques happen for various reasons, from pellets unattaching to sensor malfunction [154, 122]. Some of the reasons depend on the recording technology. With X-ray microbeam [154], on which we focus in this chapter, mistracks can happen because the microbeam looks for a pellet but is not able to find it, or because it follows the wrong pellet, e.g. the T1 raster follows the T2 pellet. This may be caused by the pellet accelerating too quickly; by shadowing from tissue, bone, teeth and fillings; or by pellets coming into close proximity. Mistracks can often be detected by the recording technology (e.g. when losing a pellet) or a posteriori (e.g. if following the wrong pellet, the values for two pellets will be nearly identical over time); but sometimes they are not detected at all. We will focus here on detected mistracks, and assume that the recording system provides a binary label indicating whether each component of the vector containing the coordinates of all pellets is present or not.

With X-ray microbeam, mistracks of a given pellet occur more commonly in subsequences of 50 to 500 ms, often near the beginning of a record, after which the pellet is recaptured (a record is defined as a

---

This chapter is mainly based on the reference [114]

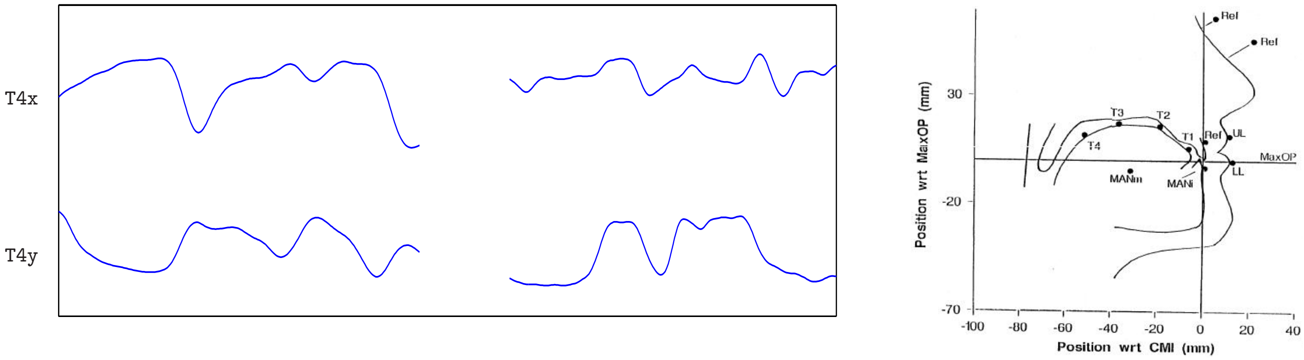


Figure 7.1: Example of typical mistracks for one pellet in the XRMB (pellet schematic at right). The mistrack duration is around 0.5 sec. The pellet can move drastically over this period, so one cannot simply interpolate it linearly.

single continuous task interval, e.g. an utterance or an isolated word recording). Mistrack proportions in the XRMB are small (around 1.9%), but the proportion of records containing mistracks is at least 35% [154, p. 65]. Since recording is expensive, cumbersome and (with X-ray microbeam) risky, this means that one cannot just discard records and re-record them again until they are perfectly tracked. Also, discarding the mistracks is wasteful and would break the continuity of articulation, which in turn makes it more difficult to recover articulatory movements from acoustics [132]. Therefore, reconstructing the mistracks becomes a necessity. At present, the XRMB indicates which frames are missing in each record, but provides no reconstruction.

The fundamental approach we follow is based on the following question: given that say only the tongue dorsum pellet is missing, can we reconstruct it from the location of the rest of the pellets? More generally, is there enough information in the present components of the data to predict the missing components? If the answer is yes—which it largely is in our problem considering human articulation are quite constrained, at least if not too much data is missing—then we can apply machine learning algorithms to estimate the missing data quite accurately. In addition, the method must handle time-varying missing data patterns in a transparent way. We describe such an algorithm in section 7.2, and report very successful experimental results with the XRMB in section 7.3.

Related work. If the mistrack sequence are very short, e.g. < 20 ms, we can recover the mistracks by the spline interpolation assuming the articulators move continuously. However, this method does not use information from those present components. Besides, these short-term mistracks are not the typical cases in the XRMB. From chapters 5–6, we know that it is possible to reconstruct the entire midsagittal tongue contour with submillimetric accuracy from the positions of just 3–4 points on it. However, it was assumed that some components were always present (the 3–4 pellets) and the rest always missing, so the problem reduces to fitting a single mapping from the present to the missing components. This is unsatisfactory in our case because which components are present and which are missing varies from frame to frame; thus, the number of combinations of (present,missing) variables (such as missing = {tongue tip, lower lip} and present = {rest}) grows exponentially, and there is neither enough data nor enough computation available to fit all those mappings. Roweis [127] proposed to learn a low-dimensional manifold to represent the data and then intersect this with the constraints provided by the present values. This geometric approach is only efficient with (locally) linear manifolds. Furthermore, the reconstructed trajectories can be quite jagged since the method does not consider the continuity constraint explicitly. Another approach to missing data reconstruction is via the expectation-maximization (EM) algorithm [52].

7.2 Deriving mappings with varying sets of inputs and outputs from a density model

Our goal is to obtain a flexible, efficient way to construct mappings "on demand" between an arbitrary set of input and output variables. Our approach is based on [22, 24]. Call the articulatory variables $\mathbf{x} = (x_1, \dots, x_D)$ (in our problem, $D = 16$ for the 2D positions of 8 pellets). At frame $\mathbf{x}_t$ in the utterance, let $\mathcal{P}_t$ and $\mathcal{M}_t$ be two sets of indices with $\mathcal{P}_t \cap \mathcal{M}_t = \varnothing$ and $\mathcal{P}_t \cup \mathcal{M}_t = \{1, \dots, D\}$, indicating the present and missing variables at that frame, respectively. The idea is to encode all possible input-output relations in a master joint density $p(x_1, \dots, x_D)$, and derive from it a mapping $\mathbf{x}_{\mathcal{P}} \rightarrow \mathbf{x}_{\mathcal{M}}$ as the mean of the conditional distribution $p(\mathbf{x}_{\mathcal{M}}|\mathbf{x}_{\mathcal{P}})$. The conditional distribution answers the question "what can we say about the values of $\mathbf{x}_{\mathcal{M}}$ given I know the values of $\mathbf{x}_{\mathcal{P}}$?".

For this to be useful, computing the conditional distributions must be done efficiently, yet they must be able to represent arbitrary nonlinear mappings. We can satisfy both needs by defining the joint density to be a Gaussian mixture with M components $p(\mathbf{x}) = \sum_{m=1}^{M} \pi_m \mathcal{N}(\mathbf{x}; \boldsymbol{\mu}_m, \boldsymbol{\Sigma}_m)$. The conditional distribution can be obtained as $p(\mathbf{x}_{\mathcal{M}}|\mathbf{x}_{\mathcal{P}}) = p(\mathbf{x})/p(\mathbf{x}_{\mathcal{P}})$ in terms of the joint and marginal distributions, all of which are Gaussian mixtures, and they equal (the indices assume we extract the corresponding block matrices, e.g. the marginalized variables are simply removed):

$$p(\mathbf{x}_{\mathcal{P}}) = \sum_{m=1}^{M} \pi_m \mathcal{N}(\mathbf{x}_{\mathcal{P}}; \boldsymbol{\mu}_{m,\mathcal{P}}, \boldsymbol{\Sigma}_{m,\mathcal{P}\mathcal{P}}) \tag{7.1}$$

$$p(\mathbf{x}_{\mathcal{M}}|\mathbf{x}_{\mathcal{P}}) = \sum_{m=1}^{M} \pi_{m,\mathcal{M}|\mathcal{P}} \mathcal{N}(\mathbf{x}_{\mathcal{M}}; \boldsymbol{\mu}_{m,\mathcal{M}|\mathcal{P}}, \boldsymbol{\Sigma}_{m,\mathcal{M}|\mathcal{P}}) \tag{7.2}$$

where the GM parameters $\{\pi_{m,\mathcal{M}|\mathcal{P}},\ \boldsymbol{\mu}_{m,\mathcal{M}|\mathcal{P}},\ \boldsymbol{\Sigma}_{m,\mathcal{M}|\mathcal{P}},\ \text{for } \forall m\}$ can be computed as

$$\pi_{m,\mathcal{M}|\mathcal{P}} = \pi_m \mathcal{N}(\mathbf{x}_\mathcal{P}; \boldsymbol{\mu}_{m,\mathcal{P}}, \boldsymbol{\Sigma}_{m,\mathcal{PP}})/p(\mathbf{x}_\mathcal{P}) \tag{7.3}$$

$$\boldsymbol{\mu}_{m,\mathcal{M}|\mathcal{P}} = \boldsymbol{\mu}_{m,\mathcal{M}} + \boldsymbol{\Sigma}^T_{m,\mathcal{PM}}\boldsymbol{\Sigma}^{-1}_{m,\mathcal{PP}}(\mathbf{x}_\mathcal{P} - \boldsymbol{\mu}_{m,\mathcal{P}}) \tag{7.4}$$

$$\boldsymbol{\Sigma}_{m,\mathcal{M}|\mathcal{P}} = \boldsymbol{\Sigma}_{m,\mathcal{MM}} - \boldsymbol{\Sigma}^T_{m,\mathcal{PM}}\boldsymbol{\Sigma}^{-1}_{m,\mathcal{PP}}\boldsymbol{\Sigma}_{m,\mathcal{PM}} \tag{7.5}$$

Conditional mean. We can derive the desirable mapping $\mathbf{f}(\mathbf{x}_\mathcal{P})$ by summarizing $p(\mathbf{x}_\mathcal{M}|\mathbf{x}_\mathcal{P})$. One effective way is to derive it through conditional mean $\mathrm{E}\{\mathbf{x}_\mathcal{M}|\mathbf{x}_\mathcal{P}\}$.

$$\mathbf{f}(\mathbf{x}_\mathcal{P}) = \mathrm{E}\{\mathbf{x}_\mathcal{M}|\mathbf{x}_\mathcal{P}\} = \boldsymbol{\mu}_{\mathcal{M}|\mathcal{P}}(\mathbf{x}_\mathcal{P}) = \sum_{m=1}^{M} \pi_{m,\mathcal{M}|\mathcal{P}}(\mathbf{x}_\mathcal{P})\, \boldsymbol{\mu}_{m,\mathcal{M}|\mathcal{P}}(\mathbf{x}_\mathcal{P}) \tag{7.6}$$

To indicate uncertainty of missing data reconstruction, one can compute the errorbars for prediction. We can estimate the errorbar from Gaussian mixture covariance $\boldsymbol{\Sigma}_{\mathcal{M}|\mathcal{P}}$, defined as:

$$\boldsymbol{\Sigma}_{\mathcal{M}|\mathcal{P}} \equiv \mathrm{E}\left\{(\mathbf{x}_\mathcal{M} - \boldsymbol{\mu}_{\mathcal{M}|\mathcal{P}})(\mathbf{x}_\mathcal{M} - \boldsymbol{\mu}_{\mathcal{M}|\mathcal{P}})^T|\mathbf{x}_\mathcal{P}\right\} \tag{7.7}$$

$$= \sum_{m=1}^{M} \pi_{m,\mathcal{M}|\mathcal{P}}\left(\boldsymbol{\Sigma}_{m,\mathcal{M}|\mathcal{P}} + (\boldsymbol{\mu}_{m,\mathcal{M}|\mathcal{P}} - \boldsymbol{\mu}_{\mathcal{M}|\mathcal{P}})(\boldsymbol{\mu}_{m,\mathcal{M}|\mathcal{P}} - \boldsymbol{\mu}_{\mathcal{M}|\mathcal{P}})^T\right) \tag{7.8}$$

That is, we compute the error bar for the missing articulatory channel as the square root of the corresponding eigenvalue of $\boldsymbol{\Sigma}_{\mathcal{M}|\mathcal{P}}$.

Conditional modes. Alternatively, we can apply the conditional modes, developed in chapter 4, which results in a multivalued mapping.

If the conditional distributions are unimodal for each frame, using the conditional mean gives a good reconstruction for the missing values. If at some frame there were many missing values, the latter would be poorly constrained and their distribution would likely be multimodal, but empirically we observe that this is not the case in our data. Therefore, in this chapter, we focus on reconstruction using the conditional mean.

In summary, the method is as follows. The joint density model is learned offline using a complete data set $\{\mathbf{x}_n\}$ using the EM algorithm (note that it is also possible to estimate the joint density via EM from incomplete datasets [52]. But this is not the case in our problem since we have enough complete data.). At each frame $\mathbf{x}_t$ we determine which components $\mathcal{M}_t$ are missing, and reconstruct them as $\mathrm{E}\{\mathbf{x}_{\mathcal{M}_t}|\mathbf{x}_{\mathcal{P}_t}\}$. Note each frame in the utterance is reconstructed independently of the others, i.e., we apply no temporal smoothing.

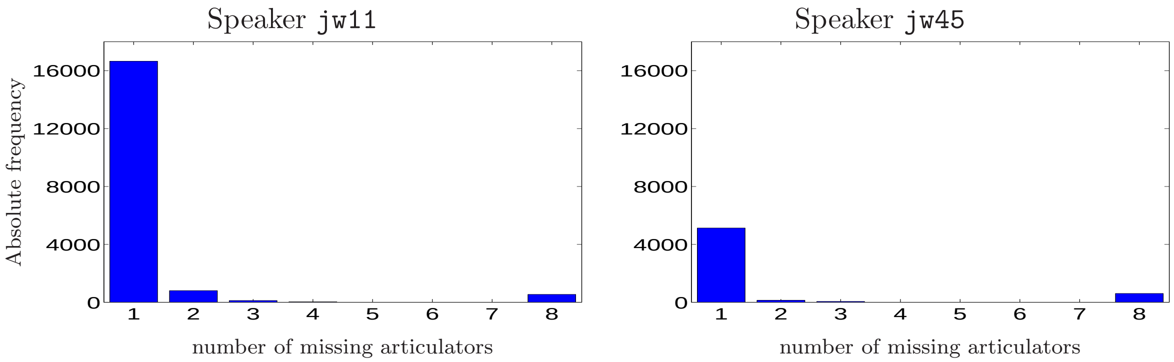


Figure 7.2: Histogram of the number of missing articulators for speakers `jw11` and `jw45`.

7.3 Experimental results

Dataset. We use articulatory data from two speakers, `jw11` and `jw45` from XRMB, with mistrack percentages of 11.32% and 3.55%, respectively. Mistracks occur most often on one articulator at a time and very rarely on multiple articulators (fig. 7.2). We focus on the reconstruction of a single missing articulator, but our method is generally applicable to cases of multiple missing articulators. We partition the data for each speaker into training and testing sets. They contain 50 000 frames randomly sampled from 49 utterances and 10 000+ frames from 14 utterances, respectively.

Joint density. To estimate the joint density $p(\mathbf{x})$, we explore two types of models.

- *Nonparametric Gaussian kernel density estimate* (KDE). We try isotropic (KDEi, $\boldsymbol{\Sigma}_m = \sigma^2\mathbf{I}\ \ \forall m$) and full-covariance matrices (KDEF, $\boldsymbol{\Sigma}_m = \sigma_F^2\hat{\boldsymbol{\Sigma}}_m$). In KDEi, the user supplies a bandwidth σ so each covariance is $\sigma^2\mathbf{I}$. In KDEF, we estimate a full covariance matrix $\hat{\boldsymbol{\Sigma}}_m$ for each mixture component m from its 100 nearest neighbors in the training set of 10000 frames, and multiply this by a user bandwidth σ_F so each covariance is $\sigma_F^2\boldsymbol{\Sigma}_m$.

- *Parametric density estimate by Gaussian mixtures* (GM). We try GM with $M = 32$, 64 and 128 components and each with a full-covariance matrix (this gave better results than using isotropic or diagonal covariances). Each GM is trained with EM from 10 random initializations.

7.3.1 Reconstruction of artificially blacked-out data

This experiment aims to quantify the reconstruction accuracy with ground truth. Given an utterance with complete articulatory measurements, we black out two channels of one articulator over the entire utterance, and then infer their positions given the remaining 7 articulators, and compare with the ground truth. We repeat this for each articulator and for several utterances.

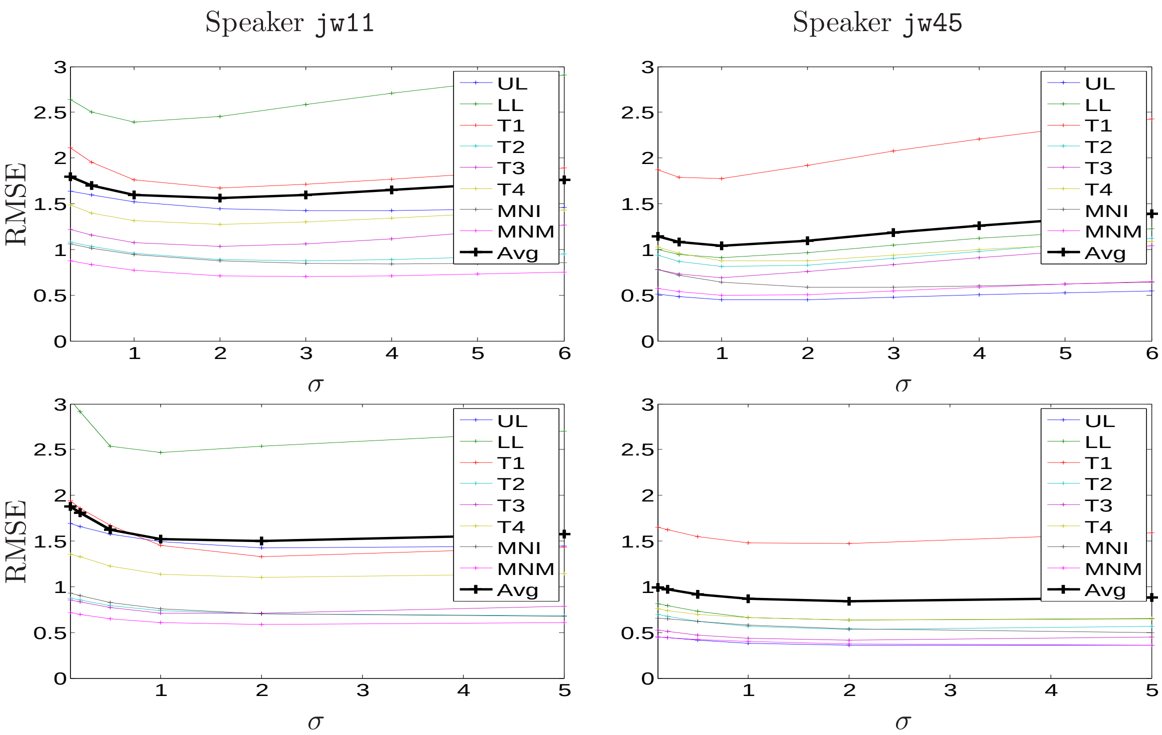
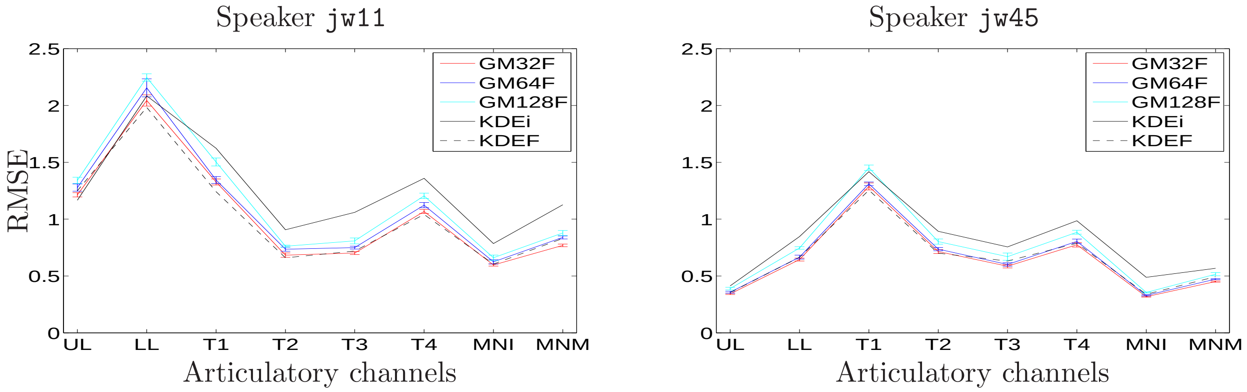


Figure 7.3: Effect of the bandwidth σ on reconstructing missing articulators. *Top*: KDEi. *Bottom*: KDEF. The "Avg" curve is a weighted sum of the reconstruction error of each articulator, with weights inversely proportional to the respective error.

First, we study the choice of model parameters. Although many rules exist to set the bandwidth of a KDE in an unsupervised setting, here we can set it to minimize our reconstruction error. Fig. 7.3 plots the effect of the KDE bandwidth. For each blacked-out articulator, we compute the RMSE by averaging over a subset of the testing set for the given σ or σ_F. Each articulator favors a slightly different σ. On average, we found $\sigma = 1.75$ (`jw11`) and 1 (`jw45`), and $\sigma_F = 1$ (`jw11`, `jw45`) to be optimal, and use these values for the rest of the experiments. The reconstruction error also varies among different articulators. In general, it is easier to reconstruct the dorsum tongue pellets `T2,T3,T4` (around 1 mm RMSE) than `T1` (the tongue tip) and the lips (1.5+ mm RMSE). This agrees with the observation that the latter tend to be more variable and harder to predict (`T1`) or less coupled from other articulators (lips). However, the relative difficulty in the reconstruction does differ among the speakers, as does the absolute reconstruction error (the average RMSE differs by 0.5 mm among both speakers).

Next, we quantify the reconstruction error for individual missing articulators with each density model. Fig. 7.4 plots the averaged RMSE over the test set for individual missing articulators. Although by little, the GMs consistently outperform the KDEs by an average 0.2 mm. KDEF beats KDEi but requires considerably more computation. Among all GMs, the one with $M = 32$ components beats all others. All these results hold for both speakers and they are consistent with fig. 7.3. For example, they


Speaker `jw11`

Speaker `jw45`

Figure 7.4: Reconstruction error for each missing articulator. For the Gaussian mixtures, the (tiny) errorbars are over 10 random initializations of EM.

confirm that the lower lip for `jw11` and `T1` for `jw45` are the hardest to be reconstructed. On average, the reconstruction for all articulators is 1 to 1.5 mm for `jw11` (except for the lower lip, with a RMSE of 2.0 mm) and 0.5 to 1 mm for `jw45` (except for `T1`, with a RMSE of 1.4 mm). Thus, we conclude that a highly parsimonious Gaussian mixture (just 32 full-covariance components) can achieve a very accurate reconstruction. Recall that the measurement error in the XRMB is around 0.5 mm [154].

Fig. 7.5 shows typical reconstructions of tongue pellets' trajectories for missing periods of 5 sec. The reconstructed trajectories are very close and correlated with the true ones even though the latter heavily oscillate over the missing period. This holds for all density models, although as mentioned the GM provides the best reconstruction (lowest reconstruction error and also smoothest reconstructions). KDEF is again better than KDEi and occasionally beats GM (e.g. the reconstruction of `T1` between 3.2 and 3.4 seconds). Even though we use no temporal information, discontinuous or jagged reconstructions happen only very rarely. `T1` is often more difficult to reconstruct than other tongue pellets since its motion is less coupled with them. On the other hand, it is very easy to reconstruct `T2` from `T3` or vice versa, which is due to the fact that `T2` is highly correlated with `T3`. The results are consistent among both speakers. Reconstruction errors seem to be correlated with those from articulatory inversion. For example, in both cases it is more difficult to recover the movement of lips than the tongue pellets although RMSE on lips appear deceptively low.

7.3.2 Reconstruction of truly missing data

Fig. 7.6 shows the reconstructions of truly missing articulators (for which we have no ground truth) using the GM with $M = 32$ components; we show the phonetic labels (which are available from the synchronized speech and potentially could be used to constrain the reconstruction) to help validate the reconstruction. Overall, the reconstructed articulatory trajectories are quite smooth, and the endpoints

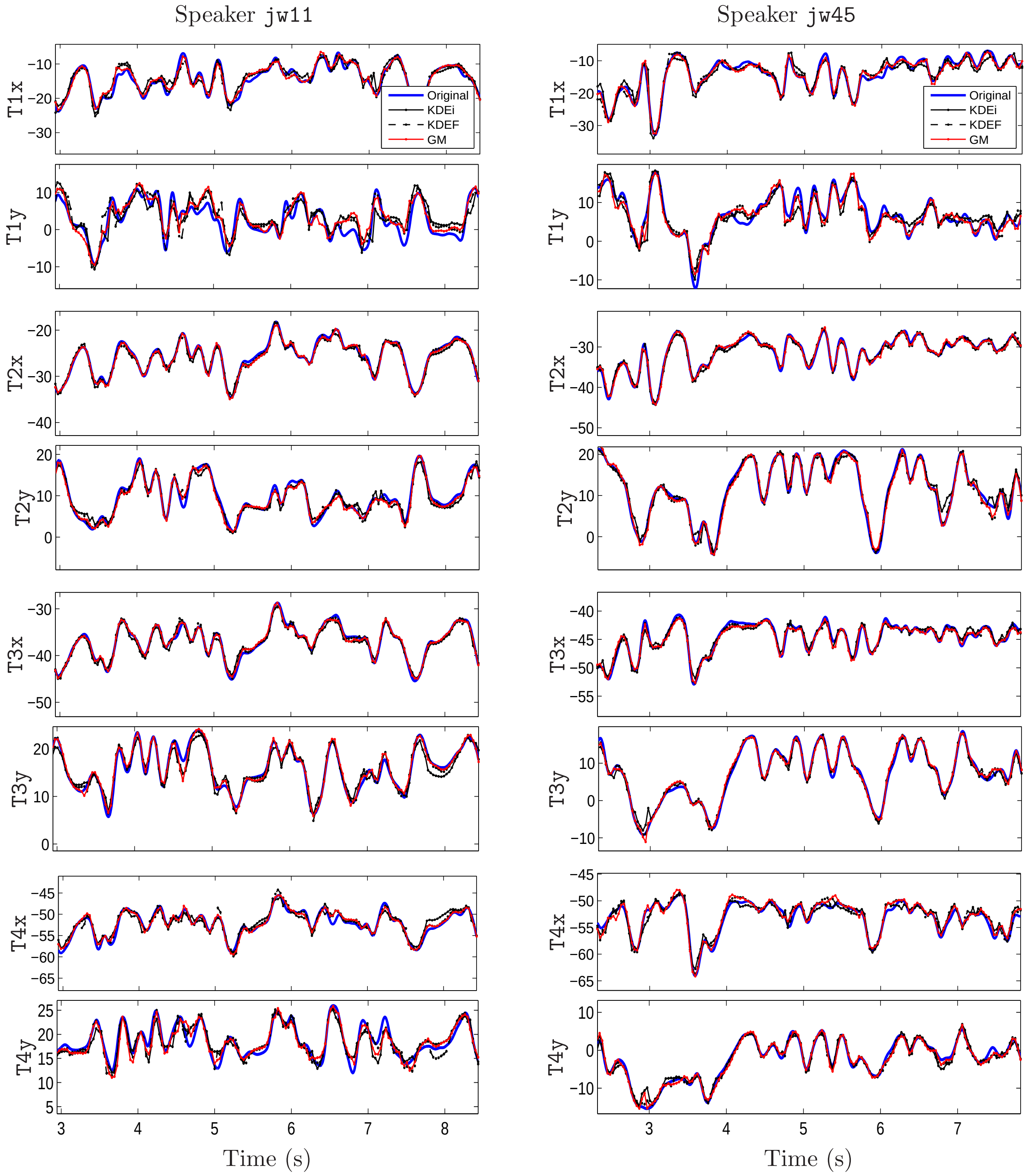


Figure 7.5: Reconstruction of artificially blacked-out articulators `T1,T1,T3,T4` (top to bottom) for the utterance `tp011`.

of the reconstructed data typically match very well with the present data, even though this was not enforced (each frame is reconstructed independently). Small discontinuities do occur, likely caused by the transition from one mixture component to another. This may be improved by a better density model, or by using temporal information. Visually, the trajectories look realistic, particularly if comparing with the corresponding phonetic label of the missing data, and if we compare the same phonetic context in a case where it is missing with a case where it is present. This happens in the reconstruction of mistracks of T4 for `tp12_jw11`: note the context grandfather, especially for `T4y`. In addition, we show the errorbars of one standard deviation along with the reconstructions. They are quite small relatively, i.e., within 5% of dynamic ranges of articulatory channels. They move smoothly across time generally except for some segments, e.g. T4 for `tp012_jw11`, where rapid changes occur during short periods. This indicates our density model is not smooth and could be improved in future. In addition, we also show the reconstructions by spline interpolation in fig. 7.6. The results clearly confirmed that the unsupervised spline interpolation fails to reconstruct the mistracks of long periods because the articulators typically undergo drastic changes during these periods.

7.4 Summary

We have extended an algorithm for missing data reconstruction and applied it to recovering missing pellet tracks in X-ray microbeam recordings, where the pellets are missing over extended periods, and the subset of missing pellets changes over time. A surprisingly parsimonious density model was sufficient to produce very accurate reconstructions for most pellets, even when the trajectory oscillates drastically over the period where it is missing. One limitation of the approach is that it relies on estimating a density model of the data ahead of time using a complete dataset (with no missing values). While this is not a problem with existing, large articulatory databases, future work should address reconstruction in more challenging situations, such as (near) real-time, or where little or no complete data are available for training.

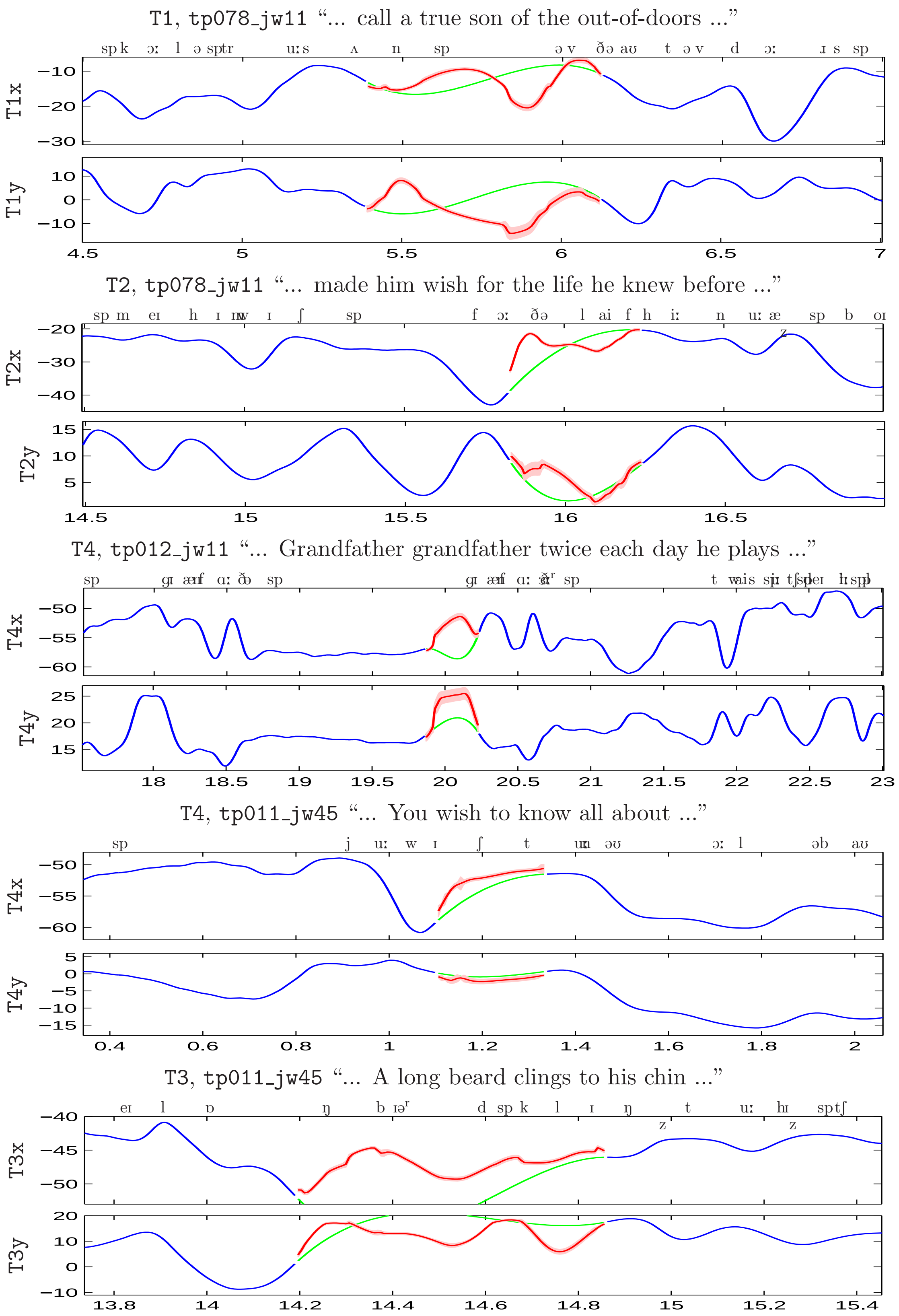


Figure 7.6: Reconstruction of truly missing data (i.e., one articulator is missing at a time) with a GM (red); rest of the articulatory trajectory in blue, and phonetic labels. The shaded, pink colored error regions plotted to indicate uncertainty of missing data reconstruction. Interpolation by spline is plotted in green.

Chapter 8

Conclusions

8.1 Summary

Articulatory speech processing involves processing and modeling in articulatory domain. It regains its popularity recently because (1) acoustics based speech technologies seem to hit their performance bottleneck and very difficult to be improved upon. (2) speech production databases containing real articulatory measurements become widely available in many languages, which enables data-driven studies. Although it is a long-standing problem in speech research, there are still many hard problems to be solved and open questions to be answered. In this thesis, we address several key problems in this field, i.e., *articulatory inversion*, *tongue reconstruction*, *missing data reconstruction*, *trajectory inverse kinematics*.

In chapter 2, we address nonuniqueness in acoustic-to-articulatory mapping, a crucial problem that plagues many computational algorithms. Due to lack of reliable speech production model, analytical analysis of the problem is infeasible. Rather, we study nonuniqueness by a systematic, large-scale empirical investigation using articulatory data for normal speech from the Wisconsin XRMB. Searching for nonuniqueness in the full articulatory spaces does give direct, quantitative evidence of the presence of nonuniqueness of the inverse mapping in normal human speech; but also suggest that, while some sounds are indeed produced in multiple ways, often a unique VT shape is used. In the refined searching for nonuniqueness in individual articulatory spaces suggests that nonuniqueness affects all articulators we considered especially the tongue. However, for any given acoustic sound some or even all articulators may be strongly constrained. The set of articulatory shapes that correspond to a given sound is usually tightly concentrated around a roughly spherical region in articulator space (dimension 0). We do find many sounds that show more complex shapes: multimodality (dimension 0), very elongated in a straight or curved path (dimension 1), or even more complex. The limitation of our work includes:

(1) We considered a single speaker from the XRMB, and did not study the relevance of the acoustic context for the cases where nonuniqueness occurred. (2) The VT representation provided by the XRMB is incomplete, lacking data about the lower VT. It is thus possible that sounds that are produced with the same upper VT shape do differ in the lower VT, thus increasing the frequency of nonuniqueness. As a promising articulatory acquisition technique, dynamic MRI may offer a full representation of the VT in the future. In chapter 3, we aim to identify the best acoustic features and their parametrization for articulatory inversion. We found relatively large windows and smoothing help to alleviate the jaggedness of acoustic features and improve inversion accuracy. The best acoustic feature is found to be LSF (although PLP performed just as well). The best parametrization is obtained by using dynamic features, 64 to 80 ms short-time window, double-filtering smoothing of cut-off frequency $\theta = 0.25$, and a 15 ms time delay between articulatory and acoustic frames. However, the improvement over other combinations of features or smoothing was very small (around 0.3 mm, to yield an RMSE of around 1.65 mm). The results may be specific to the single speaker. However, while other choices may alter the RMSE in absolute terms, we do not expect major changes to the relative ranking of the features. In chapter 4, we have presented an articulatory inversion algorithm, conditional modes, that employs a density model to predict (possibly multiple) feasible, typical vocal tract shapes for a given acoustics, and disambiguates a sequence by choosing the smoothest path among these shapes. The algorithm correctly recovers either a retroflex or a bunched shape for the American English /ɹ/, while a neural network recovers an incorrect average of both. While being computationally more costly than neural networks, we can adopt a practical strategy that apply the algorithm selectively on acoustic frames that exhibit nonuniqueness.

The goal of chapter 5 is to study predictive modeling of tongue shapes by a nonlinear mapping. We found the nonlinear predictive model significantly outperforms the conventional spline interpolation by an order of magnitude. It also consistently improve over the linear mapping. In addition, we identify the best landmarks placements empirically, which is very instructive on how to place tongue pellets during EMA/XRMB recordings. In chapter 6, we aim to adapt the predictive model previously learned on one speaker to another. We propose two feature adaptation based methods. **Global adaptation** estimates a global linear transformation. It run extremely fast (e.g. <1 second for adapting a RBF model) and can be used to correct misalignments between different ultrasound recording sessions. Furthermore, it can be extended to handle important applications where only partial adaptation contours are available. We have demonstrated that such extension can recover realistic tongue contours for articulatory databases, based only on the 2D coordinates for the tongue pellets provided in the latter (without the need for correspondences, full contours or any other information like MRI images of VT, audio, geometric articulatory models). The reconstructed tongue satisfies physical constraints (e.g. not going through the palate, teeth or velum) without having to apply the latter explicitly, and provides detailed information not readily available in the database such as the precise location of tongue-palate constrictions. This could be very useful for research in speech production and articulatory synthesis and inversion. **Local adaptation**, improves the first one by considering local transformations that align each contour point separately. This makes the method more flexible and eliminates its estimation bias. It asymptotes close to the one retrained with abundant data, and distinctly outperforms retraining and the global method when the number of adaptation contours is not very small (10 to 50). Thus the user should use the global, local, or retraining methods with less than 10, 10 to 50, and more than 50 contours, respectively.

In chapter 7, we address a practical missing data problem in articulatory data acquisition. That is, we aim to recover the some mistracked articulatory channels from other available ones. We have extended an algorithm for missing data reconstruction and applied it to recovering missing pellet tracks in X-ray microbeam recordings. We found a surprisingly parsimonious density model was sufficient to produce very accurate reconstructions for most pellets, even when the trajectory oscillates drastically over the period where it is missing. One limitation of the approach is that the quality of reconstruction relies on the quality of a density model that must be estimated offline using a complete dataset. While this is not a problem with existing, large articulatory databases, future work should address reconstruction in more challenging situations, such as (near) real-time, or where little or no complete data are available for training.

In appendix C, we apply our conditional modes approach to trajectory IK, recovering the sequence of joint angles of a robot arm such that its hand reaches a sequence of targets in the work space. It is related to articulatory inversion for the nonunique inverse mapping. Compared with the conventional approaches (e.g. Jacobian pseudoinverse), our method can effectively deal with trajectories containing singularities, where the inverse mapping changes topology, and with complicated angle domains caused by mechanical constraints (e.g. to prevent self-intersection of body limbs in a humanoid robot). We have demonstrated the method with trajectory IK for the industrial robot arms, i.e., PUMA 560 with known forward and inverse mappings. We also confirm by experiments that our method is able to deal with other practical issues: fine positioning, real-time manipulation, joint limits and obstacles. Alternatively, we have demonstrated that the conditional density may be estimated online by nonlinear, nongaussian trackers, e.g. particle filter. In addition, we formulate the trajectory IK as a tracking problem and show various Gaussian or nongaussian trackers can be used independently to produce competitive results. One limitation of our methods is the scalability to robot arms with high DOF because estimating $p(\boldsymbol{\theta}|\mathbf{x})$ in high-dim space would become very challenging and still an open research area.

8.2 Contribution of the thesis

The main contributions of this thesis are:

- Articulatory inversion (chapter 2–4)
    1. Presented the first empirical study on nonuniqueness of instantaneous acoustic-to-articulatory mapping using large-scale speech production data.
    2. Presented the first study on comparisons of acoustic representation for articulatory inversion.
    3. Applied the framework of conditional density modes to articulatory inversion successfully.
- Tongue reconstruction (chapter 5–6)
    1. Proposed predictive modeling tongue shapes by nonlinear regression.
    2. Proposed adaptation framework based on feature normalization.
    3. Reconstructed tongue shapes from a few landmarks in articulatory databases
- Missing data reconstruction (chapter 7)
    1. Extended a missing data reconstruction method to reconstruct mistracks in an articulatory database
- Trajectory inverse kinematics (appendix C)
    1. Applied the condition density modes to trajectory IK successfully. Designed and implemented the smoothness constraint and its optimization by dynamic programming.
    2. Proposed the tracking framework for trajectory IK. To the best of our knowledge, our work may be the first work on formulating trajectory IK as a tracking problem.

8.3 Public data resources

This thesis produces a dataset and several code packages of potential use for future research. They will be publicly and freely available from my home page in a hope to be useful for the other researchers. These resources include:

- A distribution of contour datasets of ultrasound tongue images and extracted 2D tongue contours.
- Corrected orthographies for several speakers in XRMB.

- A Matlab package for manipulating Gaussian mixture and mode-finding.
- Two Matlab packages containing tools to read raw data from articulatory databases — XRMBtools for XRMB and MOCHAtools for MOCHA, into Matlab format, visualize the databases, and plot animations of the vocal tract and acoustics.
- A Matlab package for recovering mistracks for arbitrary missing data patterns in XRMB.
- A Matlab package for acoustic feature extraction from speech signals.
- A Matlab package tongue prediction and adaptation.

8.4 Directions for future work

8.4.1 Articulatory inversion

Particle filtering. EKF still assumes the posterior distribution $p(\mathbf{x}_t|\mathbf{y}_{1:t})$ to be a Gaussian. Thus, it can fail in certain nonlinear and non-Gaussian problems with multimodal posterior distributions, such as articulatory inversion due to the multi-valued inverse mapping. On the other hand, particle filtering (PF) is a new and promising method since it is able to approximate the nonlinear and non-Gaussian posterior distribution $p(\mathbf{x}_t|\mathbf{y}_{1:t})$ by a set of weighted samples. As in the **Kalman filtering**, the two components are necessary here: a measurement model (given by an estimated forward mapping) and a dynamic model (e.g. given by a random walk model). To our knowledge, no work has applied PF to articulatory inversion before. Note that PF only provides a set of particles and weights at each time step instead of effective inverse solutions. One conventional way to use these particles and weights is to directly use the mean of the posterior distribution as an instantaneous inverse solution; we will also use the modes derived from the posterior distribution by smoothing the particles and then using the mode-finding algorithm.

Combining sequential data and manifold learning. It is well-known that articulators move smoothly and articulatory data can be assumed to lie on a low-dim speech manifold. These temporal and spatial constraints in principle can be used to constraint the inverse mapping. Previous work have applied sequential data learning algorithms, e.g. HMM [60, 164] and LDS [38], to explore the constraint of temporal smoothness. Work by Roweis [127] is the first successful attempt to combine HMM and LDS where manifold learning is integral part of his constrained HMM framework [128]. This inversion framework could be a promising direction to pursue. Alternatively, one can first apply dimensionality reduction to articulatory data and then do inversion in low-dim space.

8.4.2 Tongue reconstruction

Adaptation on missing data. Our local adaptation framework can be extended to handle the missing adaptation data simply by dropping terms involving missing components — missing data deleted.

3D tongue shapes modeling and adaptation. All equations and formulae in our predictive modeling and adaptation could carry over to 3D tongue reconstruction.

Application of tongue modeling and adaptation techniques to motion retargetting in graphics. The idea of predictive modeling and adaptation of tongue shapes from a few landmarks may find useful application in motion retargetting, a key problem in computer graphics research.

8.4.3 Missing data reconstruction

Our method essentially leverages spatial constraints in geometrical XRMB data. Thus, it is possible to improve reconstructions by exploiting other constraints given by the dataset. The following are some possible constraints to pursue in future.

Temporal constraint. Our method reconstructs each missing frame independently without considering the temporal order of the trajectory data. Therefore, Incorporating temporal constraints would be useful. Classical algorithms for sequential data, e.g. HMM or LDS [17] should help.

Acoustic constraint. In Wisconsin X-ray microbeam database, there are simultaneous audio recordings. Therefore, one could infer missing articulators data from acoustics through articulatory inversion.

Phonetic constraint. Apart from audio, phonetic information are also available in the dataset. One way to leveraging phonetic information would be training GMMs separately for each sound and apply GMMs according to the phonetic label for the missing data.

8.4.4 Trajectory inverse kinematics

We will apply our approach to trajectory IK in other domains, e.g. animation in computer graphics, articulated pose tracking in computer vision, protein folding in computational biology, where neither the inverse nor possibly the forward mappings are known, and having complex mechanical constraints that are best captured by data-driven approaches.

Appendix A

Articulatory speech databases

A.1 The Wisconsin X-ray microbeam database

The Wisconsin X-ray microbeam database (XRMB) database [154] was designed for research in speech production, phonetics, and linguistics by John Westbury and his team at University of Wisconsin [154, 153]. This database is well-known for its richness of phonetics, large-scale, high-quality, simultaneous recordings of speech production data and audio. The database contains different kinds of measurements for isolated words, number sequences, read sentences, paragraphs and other tasks spoken by 57 native speakers (25 men, 32 women) with varieties of regional dialects of American English and background. Each recorded 20 minutes of simultaneously recordings of articulatory measurements and audio. The articulatory measurements consist of traces of the positions of 8 receiver coils in the midsagittal plane on the speaker's articulators (4 on tongue, 2 on lips, 2 on the jaw, sampled at about 147Hz) [1], a palate, a neck wall vibration. Three reference coils (2 placed on the bridge of nose and 1 on upper incisor) were recorded to compensate head translation and pitch. Articulatory measurement error is about 0.7 mm [154]. The audio, i.e., acoustic waveform, was recorded in presence of significant room and machine noise with a sampling rate of 21739 Hz. More details about data collections and recording specifications (e.g. bead placement, coordinate systems, and XRMB tracking mechanism) can be found in [154].

The database does provides word-level transcriptions of each task, identical for each speaker. However, what was actually said often differs from these ideal transcriptions. We provide the "actually-said" version of word-level transcriptions for several speakers at `https://eng.ucmerced.edu/people/cqin`, which are useful for the task such as forced alignment, i.e., time-aligned speech segmentation of acoustic

---

[1] Although the database contains 2D measurement of articulatory in midsagittal plane. However, as shown by Stone and others [144, 145, 146] and our work described in Chapters 5,6,7, shape that tongue assumed during normal speech are sufficiently constrained and hence it is possible to infer the full tongue shape from knowledge of sparse pellet midsagittal measurement.

waveforms.

Fig. A.1 shows the trace of 8 articulators over the utterance `tp105` for speaker `jw11` plotted with the routine `XRMBtrace.m` in our Matlab package `XRMBtools`. Table A.1–A.2 show the sentences used in our experiments and table A.3 lists the subjects used from XRMB.

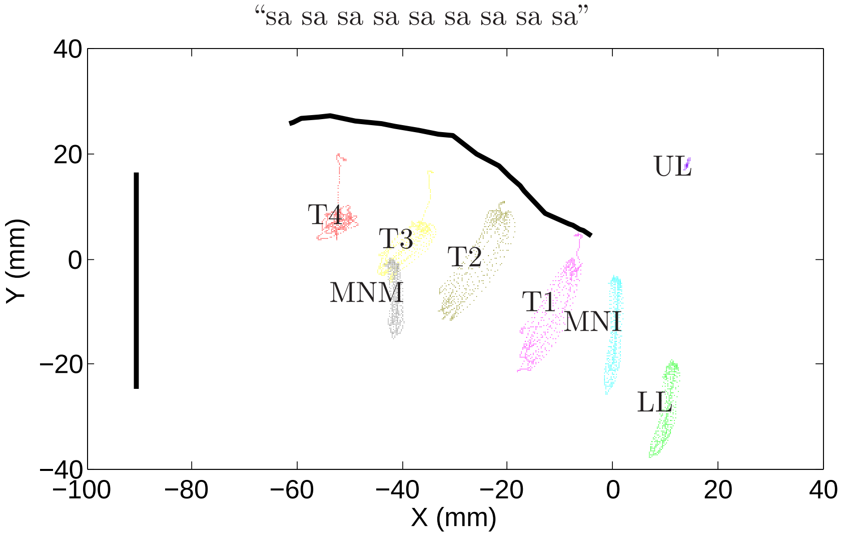


Figure A.1: Traces of 5 second recordings of 8 articulators for utterance `tp105` from speaker `jw11`.

I am very grateful to Dr. John Westbury for provide the XRMB and for the courtesy of allowing us to include the sentences from XRMB in our Matlab package `XRMBtools`.

A.2 The MOCHA-TIMIT database

The Multichannel Articulatory (MOCHA) database developed by Alan Wrench in Queen Margaret University [162] is another very popular resource for articulatory speech processing and linguistics research. This database is also known for its rich phonetic coverage, large-scale, simultaneous recordings of speech production data and audio. The database contains different kinds of measurements for 460 TIMIT read sentences spoken by 40 native speakers with northern and southern British English. Each recorded 20 minutes of simultaneously recordings of articulatory measurements and audio. The articulatory measurements consist of traces of the positions of 7 pellets in the midsagittal plane on the speaker's articulators (3 on tongue, 2 on lips, 1 on the jaw, 1 on the velum, recorded at 500Hz sampling rate by Electromagnetic Articulograph), a laryngograph (sampled at 16000 Hz), EPG (sampled at 200 Hz), a video of front view of mouth area. Two reference coils (1 placed on the bridge of nose and 1 on upper incisor) were recorded to compensate head translation and pitch. Articulatory measurement error is

ID	Word-level transcription	Duration (s)
tp001	problem children dormer never dormitory school has	7.50
tp002	nothing this street even special children ship	7.50
tp003	nine seven three nine two eight six eight five eight four nine five five ...	10.50
tp004	row special but special things although glowing	7.50
tp005	people told look moment programmer moment quite	7.50
tp006	hail this right dormer told already blend	7.50
tp007	she is about two or three when can we go home Hispanic costumes are ...	10.50
tp008	form ship back almost things school program	7.50
tp009	order row shoot used right nothing been	7.50
tp010	the other one is too big don't do Charlie's dirty dishes she had your ...	10.50
tp011	you wish to know all about my grandfather well he is nearly ninety three ...	25.00
tp012	twice each day he plays skillfully and with zest upon a small organ except ...	22.00
tp013	side sewed seed sod sued sawed sid sad surd said sud soid sowd sood sayed	20.00
tp017	don't ask me to carry an oily rag like that you can shoot at the ship or do ...	10.50
tp018	dormitory words about light first about	7.50
tp019	the coat has a blend of both light and dark fibers across the street stands ...	10.50
tp020	i assume moisture will damage this ship's hull the coat has a blend of both ...	10.50
tp021	country understand silk sense hail both	7.50
tp022	school dormer children seemed house had but	7.50
tp023	coat blend street child dormer had	7.50
tp024	things in a row provide a sense of order put these two back second children ...	10.50
tp026	the other one is too big combine all the ingredients in a large bowl the ...	14.00
tp027	seemed yourself across right sense second could	7.50
tp028	program told this across between children	7.50
tp029	when all else fails use force you can shoot at the ship or do nothing put ...	10.50
tp030	the point of the pro of the program will be told before long shaving cream ...	10.50
tp031	we are open every Monday evening I'll make sense of the problem in a ...	10.50
tp032	dorm before programmer blend sense told	7.50
tp033	beautiful row than dorm sense second	7.50
tp034	combine all the ingredients in a large bowl the other one is too big combine ...	10.50
tp035	much order smooth people have wou	7.50
tp036	they remained lifelong friends and companions they all know what I said the ...	10.50
tp037	sense long house across programmer problem	7.50
tp038	the other one is too big combine all the ingredients in a large bowl the ...	14.00
tp039	when all else fails use force combine all the ingredients in a large bowl ...	10.50
tp040	you must blend certain things to make a special wax the dormitory is between ...	10.50
tp041	but point about ship house early	7.50
tp042	combine all the ingredients in a large bowl the other one is too big combine ...	10.50
tp043	you can shoot at the ship or do nothing the gorgeous butterfly ate a lot of ...	10.50
tp044	seemed problem moment become were seemed wo	7.50
tp045	the other one is too big she always jokes about too much garlic in his food ...	10.50
tp046	the point of the program will be told before long across the street stands ...	10.50
tp047	dormitory among second street across find	7.50
tp048	the point of the program will be told before long second children are often ...	10.50
tp049	row order problem country told dorm coa	7.50

Table A.1: XRMB sentences by speaker `jw11` used in the experiments. For some sentences, we only show the truncated word-level transcriptions for clarity. The full transcription for `jw11` and several other speakers in XRMB can be found at `https://eng.ucmerced.edu/people/cqin/`.

ID	Word-level transcription	Duration (s)
tp051	seven seven eight nine three eight eight eight seven six one three three ...	10.50
tp052	program dorm dormer order didn't house program	7.50
tp053	she had your dark suit in greasy wash water all year you can shoot at ...	10.50
tp054	long light programmer information above sigh	7.50
tp055	put these two back the point of the program will be told before long the ...	10.50
tp056	the sermon emphasized the need for affirmative action second children are ...	10.50
tp057	combine all the ingredients in a large bowl the other one is too big ...	14.00
tp058	around both country had ship yet both	7.50
tp059	the coat has a blend of both light and dark fibers they all know what I ...	14.50
tp061	point man enjoy long much shoot had	7.50
tp062	against people first long from people weigh	7.50
tp063	six five eight two two six nine seven two one seven four two four two three ...	10.50
tp064	I'll make sense of the problem in a moment the point of the program will be ...	10.50
tp065	coat began cash blend this pushed flip	7.50
tp066	shoot country both shoot cash program second	7.50
tp068	when all else fails use force don't ask me to carry an oily rag like that ...	10.50
tp070	so much that light there house special	7.50
tp071	do they go up and down you must blend certain things to make a special wax ...	10.50
tp072	five six eight one nine nine eight six seven four four one six six	7.00
tp073	cash nothing point what school that becau	7.50
tp074	the dormitory is between the house and the school combine all the ...	10.50
tp075	put those two back no put these two back put those two back no put ...	17.50
tp076	major first about back this nothing wax	7.50
tp077	it's just a little thing grandmother outgrew her upbringing in petticoats ...	10.50
tp078	in late fall and early spring the short rays of the sun call a true son of ...	25.00
tp079	the desk that had tied him down was gone and his one thought was for quail ...	25.00
tp080	tom stopped near stream to rest soon after he had laid down his gun he ...	22.00
tp083	seemed himself point hail wax dormitory	7.50
tp084	you must blend certain things to make a special wax I think that's real ...	10.50
tp085	porcupines resemble sea urchins across the street stands a country school ...	10.50
tp086	put this one right here cheap stockings run the first time they're worn ...	10.50
tp087	first second head long dormitory that right	7.50
tp088	one two three four five six seven eight nine ten eleven twelve thirteen ...	20.00
tp089	himself cash point conversation that shoot had	7.50
tp091	hail light wax light ship blend school	7.50
tp092	four three seven five one two five three six four seven nine six two one ...	10.50
tp093	the other one is too big combine all the ingredients in a large bowl the ...	10.50
tp094	put these two down no put these two back put these two down no put these ...	17.50
tp095	himself blink but about measure coat wax	7.50
tp096	does creole cooking use curry a roll of wire lay near the wall don't ask ...	10.50
tp097	the other one is too big if I had that much cash i'd buy the house you can ...	10.50
tp098	if I had that much cash i'd buy the house she had your dark suit in greasy ...	10.50
tp099	wax order coat right problem first much	7.50
tp100	himself back street nothing back things street	7.50
tp101	elderly people are often excluded when all else fails use force the ...	10.50

Table A.2: XRMB sentences by speaker `jw11` used in the experiments (continued). For some sentences, we only show the truncated word-level transcriptions for clarity. The full transcriptions for `jw11` and several other speakers in XRMB can be found at `https://eng.ucmerced.edu/people/cqin/`.

Speaker	Gender	Accent
jw11	male	Dialect of Wisconsin
jw45	male	Dialect of Indiana

Table A.3: XRMB speakers used in the experiments.

about 0.45 mm [122]. The audio, i.e., acoustic waveform, was recorded with a sampling rate of 16000 Hz. More details about data collections and recording specifications (e.g. pellet placement, coordinate systems, and Electromagnetic Articulograph tracking mechanism) can be found in [162].

The database provides word-level transcriptions of each task and time-aligned phone level transcription computed by forced-alignment for each speaker.

Fig. A.2 shows the trace of 7 articulators over the utterance `001` for speaker `fsew0` plotted with the routine `MOCHAtrace.m` in our Matlab package `MOCHAtools`. Table A.4 show the sentences used in our experiments and table A.5 lists 3 subjects available in MOCHA database.

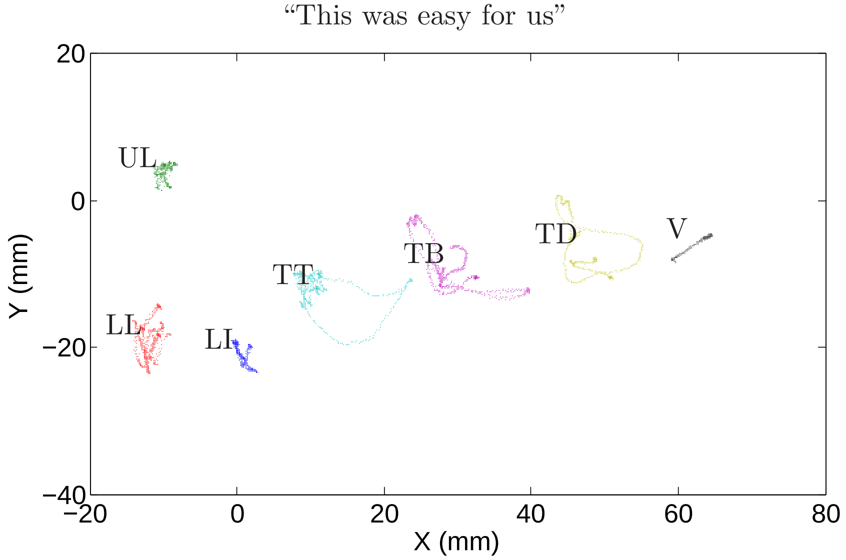


Figure A.2: Traces of 2 second recordings of 7 articulators for utterance `001` from speaker `fsew0`.

Table A.4 shows the TIMIT sentences used for testing in our experiments [2] and table A.5 the 1 subject used from the MOCHA-TIMIT database.

I am very grateful to Dr. Alan Wrench and CSTR, Edinburgh for provide the MOCHA-TIMIT database for our research and and Dr. Korin Richmond for clarifying technical questions about it.

---

[2] The TIMIT sentences used for training are not shown here but can be found at `https://eng.ucmerced.edu/people/cqin/`

ID	Word-level transcription	Duration (s)
006	Bright sunshine shimmers on the ocean	3.22
016	A roll of wire lay near the wall	3.22
026	Most young rabbits rise early every morning	3.97
036	Only the most accomplished artists obtain popularity	4.47
196	Straw hats are out of fashion this year	3.47
216	I gave them several choices and let them set the priorities	4.47
406	They assume no burglar will ever enter here	3.72
456	Ralph prepared red snapper with fresh lemon sauce for dinner	4.97

Table A.4: Testing sentences selected from the MOCHA-TIMIT database used in the experiments.

Speaker	Gender	Accent
fsew0	female	Southern English
msak0	male	Northern English
maps0	male	Northern English

Table A.5: Speakers available in the MOCHA-TIMIT database.

A.3 Tongue contour dataset

The ultrasound tongue contour dataset used in chapter 5–6 was created at Queen Margaret University. It contains two speakers (one male, `maaw0`, and one female, `feal0`) with different Scottish accents. Two data streams were recorded synchronously for each speaker: acoustic waves and ultrasound videos. The acoustic waves recorded audio signals in Microsoft PCM format at 16000 Hz. The ultrasound videos recorded the movements of the tongue in the midsagittal plane of the vocal tract at 100 Hz. To achieve phonetic balance, each speaker recorded a set of British TIMIT sentences designed for this purpose and each ultrasound recording lasted 5 to 6 seconds approximately. We also recorded two utterances containing two repetitions of three isolated words, “heed had whod”, with ascending and descending pitches, respectively. Total durations for ultrasound recordings are 145 and 90 seconds for `maaw0` and `feal0`, respectively. We extracted tongue contours from ultrasound images (i.e., the lower edge of the highlighted strip as shown in fig. A.5) in a semi-automatic approach: (1) track contours by EdgeTrak [85], an open contour tracker based on energy minimization; (2) correct tracking errors by hand. We discarded those unreliably tracked contours and obtained the datasets of 8671 contours for `maaw0` and 7272 contours for `feal0`. In addition to tongue contour recordings, we also recorded the outline of hard palate by tracing the tongue tip along the hard palate, illustrated in fig. A.3.

Although the ultrasound probe is held against the chin while recording, it is possible in principle

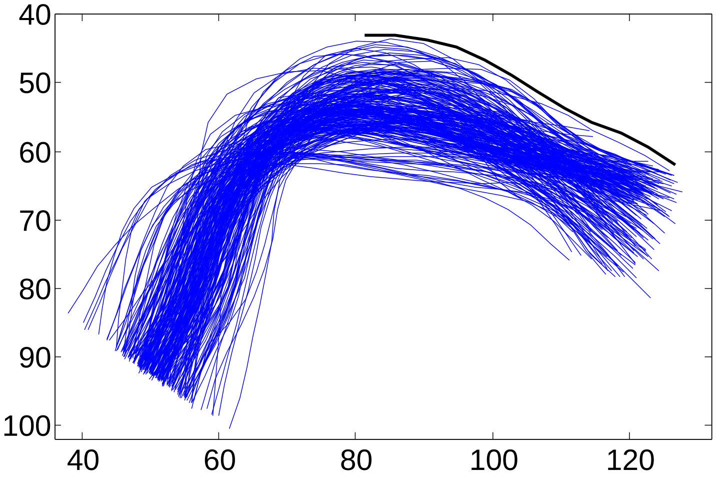


Figure A.3: Outline of hard palate in tongue contour dataset for speaker `maaw0`. *Blue*: tongue contours (a subset plotted to avoid clutter). `Black`: outline of hard palate. Note that the tongue contours and the hard palate were recorded in different sessions and the alignment is done by hand.

that the chin and the probe shift w.r.t. each other during recording. This would require normalizing the contours w.r.t. a fixed reference. However, we found this unnecessary for two reasons: in a pilot experiment, we compared the prediction results with normalization (by shifting the data to zero mean and a given orientation) and without normalization, and found little difference; in addition, we used a device (Fig. A.4) to stabilize the probe w.r.t. the head. However, there was a break when the speaker `maaw0` removed the helmet after recording 10 utterances and wore it again for the remaining 12. This introduced a slight but noticeable mismatch. Thus, `maaw0` has datasets **S1** (3727 contours from 10 utterances) and **S2** (4944 contours from 12 utterances) for sessions 1 and 2, respectively.

It is worth noting that no EMA pellet or receiver coil was glued on the tongue as reference points, which are useful for identifying the extremity of the tongue tip (often invisible in ultrasound images) and aligning temporal contour points (i.e., alleviate landmark misspecification problems described in chapter 6). This is mainly because such recording setup was not available at the time of recording. Instead, we attempted an alternative, i.e., glue two mental balls at locations of EMA tongue pellets along the tongue during recording, as shown in fig. A.5. However, in the end we did not implement this because of following concerns on the metal balls: (1) they have the risk of falling off the tongue and being swallowed during speech productions. (2) they must be big enough (i.e., its diameter ≥ 6.5mm) to be imaged by ultrasound. However, the resulting balls would be heavy and hence affect naturalness of speech production. (3) they will introduce the comet tail in ultrasound images and aggravate difficulty of contour tracking.

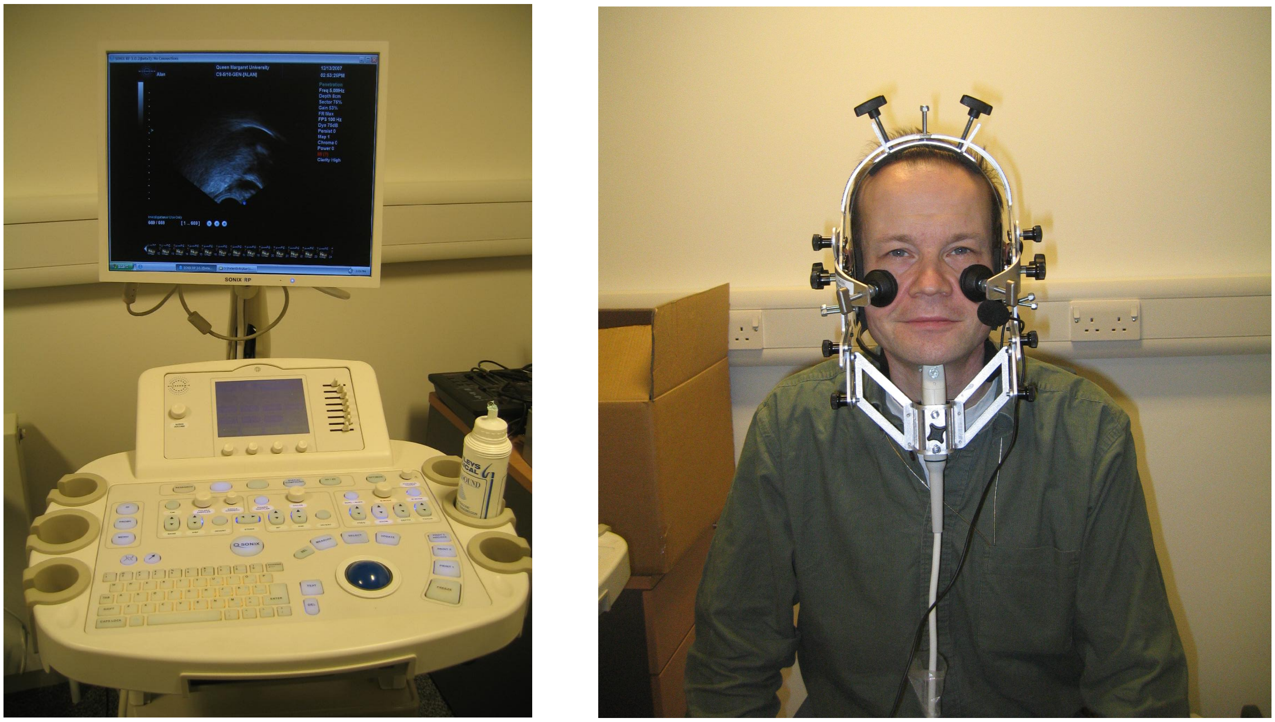


Figure A.4: *Left*: 2D ultrasound machine used. *Right*: device to stabilize the head (to reduce motion w.r.t. the ultrasound probe) [133].

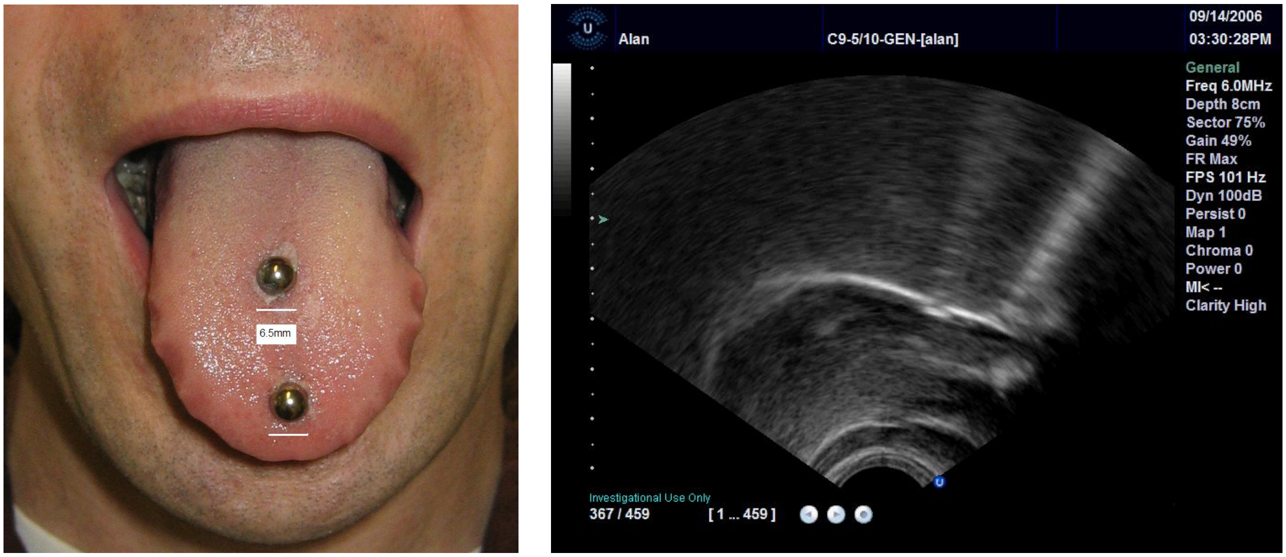


Figure A.5: Tongue contour recording with two mental balls attached to the tongue surface. *Left*: Two mental balls glued to the tongue surface. *Right*: Cometic tail effect induced by the two mental balls.

Table A.6 shows the TIMIT sentences used in our experiments and table A.7 the 2 subjects used from the English tongue contour dataset (all native English speakers but with different accents).

I am very grateful to Dr. Alan Wrench and Queen Margaret University for recording ultrasound data of tongue movement for our research and for clarifying technical questions about it.

ID	Word-level transcription	Duration (s)
054	The eastern coast is a place for pure pleasure and excitement	6.82
069	Eat your raisins outdoors on the porch steps	4.44
088	Flying standby can be practical if you want to save money	6.33
161	The government sought authorization of his citizenship	6.35
162	As co-authors, we presented our new book to the haughty audience	6.81
169	Employee layoffs coincided with the company's reorganization	5.58
177	The haunted house was a hit due to outstanding audiovisual effects	6.70
216	I gave them several choices and let them set the priorities	5.04
252	The cigarettes in the clay ashtray overflowed onto the oak table	6.21
283	To further his prestige, he occasionally reads the Wall Street Journal	8.37
317	Those answers will be straightforward if you think them through carefully first	3.17
319	If people were more generous, there would be no need for welfare	5.20
326	The groundhog clearly saw his shadow, but stayed out only a moment	6.08
332	Al received a joint appointment in the biology and the engineering	5.54
338	An adult male baboon's teeth are not suitable for eating shellfish	5.46
340	Gus saw pine trees and redwoods on his walk through Sequoia National Forest	6.01
341	Rob made Hungarian goulash for dinner and gooseberry pie for desert	6.07
352	The patient and the surgeon are both recuperating from the lengthy operation	5.99
355	Many wealthy tycoons splurged and bought both a yacht and a schooner	6.16
369	Each untimely income loss coincided with the breakdown of a heating system part	7.67
461	Heed Had Whod (high-to-low pitch)	10.37
462	Heed Had Whod (low-to-high pitch)	8.46
463	A (/ɑː/) I (/iː/) U (/uː/) E (/e/) O (/ɒ/)	6.66

Table A.6: Tongue contour dataset used in the experiments.

Speaker	Gender	Accent
maaw0	male	Scotish English
feal0	female	Scotish English

Table A.7: Speakers recorded for the English tongue contour dataset.

Appendix B

Acoustic feature extraction

B.1 Pre-processing

The front-end signal processing transforms the speech signal to a set of feature vectors. The aim is to obtain a new representation that is more compact and less redundant than the raw signal. Such representation is more suitable for statistical modeling and calculation of distance measures. The popular features used in speech technology are based on (1) linear predictive analysis, e.g. linear predictive coding (LPC) coefficients, line spectrum frequencies (LSF), linear predictive cepstral coefficient (LPCC)). (2) filterbank analysis, e.g. filterbank spectra (FBANK), Mel-frequency cepstral coefficient (MFCC). (3) auditory models of speech, e.g. perceptual linear predictive (PLP). One convention in acoustic feature extraction is to pre-process the speech signal and we describe the step briefly as follows.

The speech signal is first pre-emphasized, that is, a filter is applied to it. The filter's frequency response emphasizes on the high frequency part of the spectrum, which are generally decreased by the speech production process. The pre-emphasized signal is obtained by applying the following filter:

$$\tilde{s}(n) = s(n) - \alpha \cdot s(n-1) \tag{B.1}$$

where α is the pre-emphasis parameters (a most common value for α is about 0.95). By doing this, the spectrum magnitude of the outgoing pre-emphasized speech will have a 20 dB boost in the upper frequencies and 32 dB increase at the *Nyquist* frequency. The pre-emphasized speech signal is then segmented into frames, which are spaced 20-30 ms apart, with 10-15 ms overlaps for short-time spectral analysis. Each frame is multiplied by a fixed length window. The Hamming window are the most widely used since they taper the original signal on the sides and thus reduce the side effects.

Endpoint detection. In speech application, it is sometimes useful to classify acoustic frames into silent or nonsilent category. For example, in articulatory inversion, silent frames are often discarded during training phrase because those frames contain little information about locations of the articulators. This is also known as endpoint detection in speech processing. Formally, it is a process of clamping the interval that contains only the desired speech, e.g. voiced speech. There exist many algorithms for performing endpoint detection, e.g. zero-crossing rate [118]. In this thesis, we adopt a simple but effective procedure based on frame-energy. The algorithm is outline as follows:

1. Calculate the energy for each frame within an utterance.
2. Build an energy codebook by applying vector quantization.
3. Label each frame into silent or nonsilent according to the codebook the frame assigned to.

As described earlier, the popular acoustic features used in speech technology can be broadly classified into three categories. In the following sections, we provide an brief overview of feature extractions based on linear predictive analysis and filterbank analysis. More extensive treatment can be found in [118].

B.2 Linear predictive analysis

Linear predictive coding (LPC) [119, 118] is widely used in speech applications (e.g. recognition, coding, modeling, etc.) because the speech production process can be well modeled by linear prediction. It is essentially an *autoregressive* signal model of a discrete-time speech signal $x(n)$ where each sample is modeled by a linear combination of previous p speech samples:

$$x(n) = \sum_{i=1}^{p} a_i x(n-i) + Gu(n) \tag{B.2}$$

where p is the order of linear prediction, $\{a_i, i = 1, \ldots, p\}$, assumed constant over the short-time speech analysis frame, are LPC coefficients, G is the gain of the excitation, $u(n)$ is a excitation signal, which can either be a quasi-periodic train of impulses (for modeling voiced sounds) or a random noise source (for modeling unvoiced or fricative sounds). LPC coefficients $\{a_i\}$ determines the spectral shape of a sound and are closely related to the shape of the vocal tract. By taking Z-transform, (B.2) can be rewritten in the frequency domain as

$$S(z) = \sum_{i=1}^{p} a_i z^{-i} S(z) + GU(z) \tag{B.3}$$

Consequently, the transfer function from source $u(n)$ to $x(n)$

$$H(z) = \frac{S(z)}{GU(z)} = \frac{1}{1 - \sum_{i=1}^{p} a_i z^{-i}} = \frac{1}{A(z)} \tag{B.4}$$

, which is also known as an *all-pole filter* whose impulse response are given by a_i. Fig. B.1 shows the corresponding block diagram of this model.

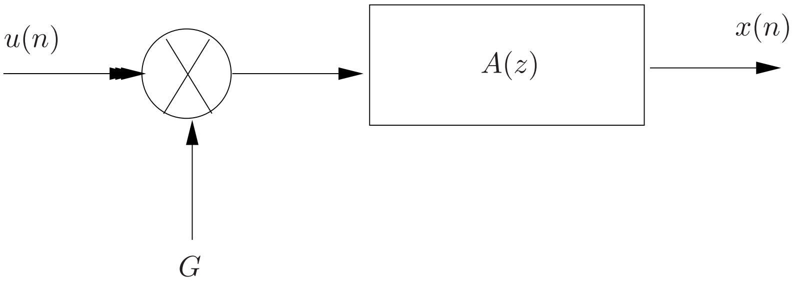


Figure B.1: Linear prediction model of speech (after [118]).

Our primary interest is to determine the set of LPC coefficients $\{a_i\}$ from the short-time windowed signal $x(n)$. One approach is through the autocorrelation method. This problem reduces to solving a linear system of a $p \times p$ matrix whose entries are computed from the autocorrelation function. Because this matrix has special structure, i.e., it is a Toeplitz matrix, it can be efficiently solved by several well-known procedures, e.g. the Levinson-Durbin algorithm [118]. A detailed analysis of LPC model and properties is beyond the scope of this thesis can be found in [119, 118].

Spectral distortion measure. Given two vectors of LPC coefficients, it is often necessary to compute the "distance" between two LPC vectors in pattern recognition application such as speech recognition. The most useful distance measures for LPC coefficients are Itakura distance [64] and Itakura-Saito distance [65]. They have been extensively analyzed in [55, 118] and thus in the following we only outline the practical procedures to compute these two distortions. The LPC coefficients aim to minimize the residual energy between the true magnitude spectrum of the speech frame and the LPC model spectrum. This suggests that one may compute the "distance" between two LPC vectors by comparing their residual energies between each of their reconstructed spectra and "true" spectrum. Let $\mathbf{a}$ and $\hat{\mathbf{a}}$ be the pth-order LPC coefficients computed from two (windowed) speech frames $x(n)$ and $x\hat{(n)}$ respectively. It is known (from Exercise 3.6 in [118]) that the prediction error (residual energy) by linear predictive analysis can be written in the form

$$E^{(p)} = \mathbf{a}^T \mathbf{R}_x \mathbf{a} \tag{B.5}$$

where $\mathbf{R}_x$ is the Toeplitz matrix calculated from the autocorrelation sequences of the signal $x(n)$. Thus, a reasonable measure of spectral distance between two frames of speech represented by $\mathbf{a}$ and $\hat{\mathbf{a}}$, and augmented matrices $\mathbf{R}$ and $\hat{\mathbf{R}}$ is

$$D(\mathbf{a}, \hat{\mathbf{a}}) = \frac{\hat{\mathbf{a}}^T \mathbf{R}_x \hat{\mathbf{a}}}{\mathbf{a}^T \mathbf{R}_x \mathbf{a}} \tag{B.6}$$

Based on (B.6), Itakura distance is defined as

$$D_{\mathrm{I}}(\mathbf{a}, \hat{\mathbf{a}}) = \log \frac{\mathbf{a}^T \mathbf{R}_x \mathbf{a}}{\hat{\mathbf{a}}^T \mathbf{R}_{\hat{x}} \hat{\mathbf{a}}} \tag{B.7}$$

Similarly, Itakura-Saito distance is defined as

$$D_{\mathrm{IS}}(\mathbf{a}, \hat{\mathbf{a}}) = \frac{\hat{\mathbf{a}}^T \mathbf{R}_{\hat{x}} \hat{\mathbf{a}}}{\mathbf{a}^T \mathbf{R}_x \mathbf{a}} - \log \frac{\hat{\mathbf{a}}^T \mathbf{R}_{\hat{x}} \hat{\mathbf{a}}}{\mathbf{a}^T \mathbf{R}_x \mathbf{a}} - 1 \tag{B.8}$$

It is evidence that both distance measures are asymmetric, i.e., $D_{\mathrm{I}}(\mathbf{a}, \hat{\mathbf{a}}) \neq D_{\mathrm{I}}(\hat{\mathbf{a}}, \mathbf{a})$ and $D_{\mathrm{IS}}(\mathbf{a}, \hat{\mathbf{a}}) \neq D_{\mathrm{IS}}(\hat{\mathbf{a}}, \mathbf{a})$. However, asymmetry is usually not a problem for applications such as speech quality evaluation [14]. In practice, when the distortion is small, the Itakura distance measure is not very different from the Itakura-Saito distance measure. Both distance measures are very popular in speech recognition. In particular, the Itakura-Saito distance has many interesting properties. In particular, it has been shown to be highly correlated with subjective quality evaluation. For example, when the Itakura-Saito distance between $x(n)$ and $x(\hat{n})$ is less than 0.1, they would be perceived nearly identically by human ears [14].

B.3 Filterbank and cepstral analysis

Fig. B.2 shows a modular representation of a filterbank based cepstral representation. Once the speech signal has been windowed, Discrete Fourier Transform (DFT) is used to transfer these time-domain samples into frequency-domain ones. Usually, Fast Fourier Transform (FFT) is used to compute the DFT, and thus a power spectrum is obtained.

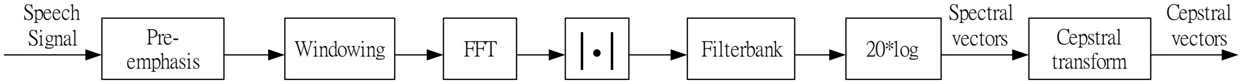


Figure B.2: Modular representation of a filterbank based cepstral parameterizations.

Filterbank spectrum. This spectrum presents a lot of fluctuations, and we are usually not interested in all the details of the them. Only the envelope of the spectrum is of interest. Another reason for the smoothing of the spectrum is the reduction of the size of spectral vectors. To realize the smoothing and get the envelope of the spectrum, we multiply the spectrum by a filterbank. A filterbank is a series of bandpass filters that are multiplied one by one with spectrum in order to get an average value in individual frequency bands. The filterbank is defined by the shape of the filters and by their frequency location (left frequency, central frequency, and right frequency). Triangular filter are often used and they can be located differently over the frequency. Mel scale for frequency localization of the filters is usually applied in most front-end feature extractions. This scale is an auditory scale which is similar to the frequency scale of the human ear. The localization of the central frequencies of the filters is given by

$$f_{\mathrm{MEL}} = 2595 * \log_{10}(1 + f_{\mathrm{LIN}}/700) \tag{B.9}$$

Finally, we take the log of this spectral envelope and multiply each coefficient by 20 in order to obtain the envelope in dB. At this stage of the processing, spectral vectors can be obtained.

Cepstral analysis. Discrete Cosine Transform (DCT), is usually applied to the spectral vectors in speech processing and yields cepstral coefficients [118]

$$C_n = \sum_{k=1}^{K} S_k \cdot \cos[n(k - \frac{1}{2})\frac{\pi}{K}],\ n = 1, 2, \ldots, L \tag{B.10}$$

where K is the number of log-spectral coefficients calculated previously, S_k are the log-spectral coefficients, and L is the number of cepstral coefficients that we want to calculate ($L \leq K$). This transformation decorrelates features, which leads to using diagonal covariance matrices instead of full covariance matrices. Finally, a cepstral vector is obtained for each analysis window.

There is an additional operation called *cepstral mean subtraction* (CMS) which is used to remove from the cepstrum contributions of slowly varying convolution noises.

B.4 Dynamic features

In addition to the static features computed either as LPC or cepstral coefficients, the time derivative approximations are incorporated in feature vectors to represent the dynamic characteristic of speech signal. To combine the dynamic properties of speech, the first and second order differences of these cepstral coefficients may be used. And these dynamic features have been shown to be beneficial to speech and speaker recognition performance [47, 140]. For MFCC, the first-order delta MFCC (ΔC_m) and second-order delta-delta MFCC ($\Delta\Delta C_m$) are computed as [47]:

$$\Delta C_m = \frac{\sum_{k=-l}^{l} k \cdot C_{m+k}}{\sum_{k=-l}^{l} |k|} \tag{B.11}$$

$$\Delta\Delta C_m = \frac{\sum_{k=-l}^{l} k^2 \cdot C_{m+k}}{\sum_{k=-l}^{l} k^2} \tag{B.12}$$

Appendix C

Trajectory inverse kinematics

C.1 Introduction

From previous chapters on articulatory inversion, we know that an acoustic frame may be produced by different vocal tract shapes. This characteristic nonuniqueness also appears in other computational fields, e.g. realistic animation of articulated characters [56] in computer graphics, the protein loop closure in computational biology [76], robot arm inverse kinematics (IK) in robotics [30, 136]. IK is formally defined as inferring joint angles of a robot manipulator (see fig. C.1 for various robot manipulators) given the desired position of the robot's hand or end-effector. This is in contrast to forward kinematics (FK), which maps the joint angles to the position of the end-effector and essentially characterizes the geometrical relationship between joint angles of articulators and the end-effector's position. Furthermore, FK typically maps very different joint angles to the same end-effector's position and this makes its inverse one-to-many. As an example, the two-link planar robot arm can reach a target in (Cartesian) the workspace by either "elbow-up" or "elbow-down" arm configurations. Therefore, one could easily imagine the following analogs: acoustic frame vs. Cartesian position; vocal tract shape vs. joint angle. Likewise, FK resembles articulatory-to-acoustic mapping and so as IK to acoustic-to-articulatory mapping. What is different is that in IK we have the well-defined forward kinematics, available for each robot manipulator [30, 136]. We want to study IK in this chapter because the algorithms we develop for IK can be useful for articulatory inversion as well.

In this chapter, we consider a variation of the IK problem, *trajectory inverse kinematics* of a (say) robot arm, where given a sequence of positions $\mathbf{x}_1, \dots, \mathbf{x}_T$ in (Cartesian) workspace of the end-effector, we want to obtain a feasible sequence of joint angles $\boldsymbol{\theta}_1, \dots, \boldsymbol{\theta}_T$ that produce the $\mathbf{x}$-sequence (we do not consider dynamics in this thesis). Given the joint angles, the end-effector position is given by the

---

This chapter is mainly based on references [109, 110]

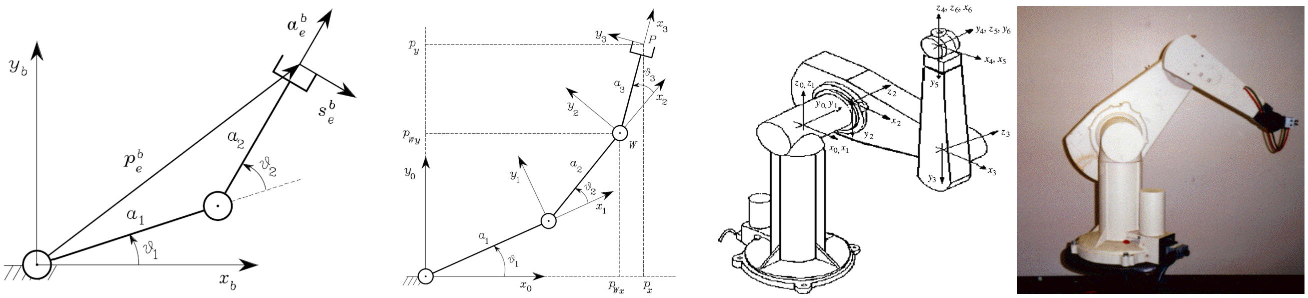


Figure C.1: Serial link robot manipulators. *Left 3*: illustrations of planar 2-link arm, planar 3-link arm, and PUMA 560. *Right*: real world PUMA 560 robot arm.

forward kinematics mapping, $\mathbf{f} : \boldsymbol{\theta} \to \mathbf{x}$, which is usually known. For example, $\mathbf{f}$ can be obtained in closed form for a kinematic chain as a product of homogeneous transformation matrices, one per link (however, we remark that this is not always the case, as in articulatory inversion). However, the inverse $\mathbf{f}^{-1}(\mathbf{x})$ can take multiple values, or for redundant manipulators (where $\dim \boldsymbol{\theta} > \dim \mathbf{x}$), an infinite number of them; this makes it difficult to represent and compute $\mathbf{f}^{-1}$. At the same time, we want the recovered sequence of joint angles to trace a continuous, realizable trajectory. Importantly, we aim to not only solve IK at each trajectory point, but also to obtain an angle trajectory that is globally feasible (e.g. avoiding discontinuities or forbidden regions). It is worth noting that this problem is different from trajectory planning (a.k.a. trajectory generation), which basically aims to generate a trajectory from the initial to the desired final positions while avoiding obstacles in the workspace. In *trajectory inverse kinematics*, we assume that the planning problem in workspace has been solved and we are given a trajectory in it.

We apply the method of conditional density modes of [22, 24] discussed in chapter 4 to trajectory IK. The goal of this study is twofold: (1) propose an new algorithm for IK; (2) quantify the performance of conditional density modes in a practical and challenging problem. Note [24] applies this algorithm to trajectory IK but tries only a planar 2-link robot arm without forbidden regions. As a recap, this method directly represents multivalued mappings using density models. It is a machine learning method that learns trajectory IK given a training set of input-output pairs $(\mathbf{x}, \boldsymbol{\theta})$ and possibly but not necessarily given the forward mapping $\mathbf{f}$. It is a global method in that it disambiguates multiple branches by minimizing a trajectory-wide constraint. The goal of the method is to obtain a $\boldsymbol{\theta}$-trajectory that, while not perfectly accurate, is sufficiently close at each point to the correct trajectory that it can be refined (if desired) using a local method.

In addition, we propose another algorithm that approaches IK by tracking in section C.4. In this tracking framework, we construct a conditional distribution of $\boldsymbol{\theta}$ (= unobserved states) given past and current $\mathbf{x}$ (= measurements) by nonlinear, nongaussian trackers, which are capable of modeling multimodal distributions. We review previous approaches in section C.2, describe and demonstrate our methods in section C.3.1 and section C.4.2.

C.2 Computational approaches to inverse kinematics

Inverse kinematics is a fundamental problem in robotics and graphics. It has been researched for years and there exist many practical and well-established approaches (see [136]). Desirable features of ideal IK solvers include: (1) run fast to allow real-time manipulation. (2) highly accurate for fine-positioning. (3) capable of finding all local solutions. (4) capable of handling singularity, obstacles, collision, and joint limits. (5) widely applicable to various manipulators. (6) stable and repeatable solutions. State-of-the-art approaches generally do not bear all above features. Let us briefly review some of these approaches here.

Analytic methods One tries to obtain the IK mapping in closed form (e.g. [106]); this is only possible for certain types of manipulators, and even then it can be complicated.

Local methods *Local* methods [157, 86, 35] are based on linearizing $\mathbf{f}$ to obtain $\dot{\mathbf{x}} = \mathbf{J}(\boldsymbol{\theta})\dot{\boldsymbol{\theta}}$, where $\mathbf{J}$ is the Jacobian of $\mathbf{f}$ (Resolved Motion Rate Control [157]). This equation can then be integrated numerically in order to obtain the global trajectory for $\boldsymbol{\theta}$. For redundant manipulators, where $\mathbf{J}$ has more columns than rows, a unique value of $\dot{\boldsymbol{\theta}}$ may be obtained by optimizing a suitable objective (such as energy) over the null space of the Jacobian; in particular, one can obtain the pseudoinverse method by minimizing $\|\dot{\boldsymbol{\theta}}\|^2$, yielding $\dot{\boldsymbol{\theta}} = \mathbf{J}^+(\boldsymbol{\theta})\dot{\mathbf{x}}$, at a computational cost $\mathcal{O}(m^2n)$ where $m = \dim \mathbf{x} < n = \dim \boldsymbol{\theta}$. However, the idea breaks down at singularities $\boldsymbol{\theta}^*$, where $\mathbf{J}(\boldsymbol{\theta}^*)$ becomes singular; this is caused by the existence of multiple inverse branches intersecting at $\boldsymbol{\theta}^*$. Also, the numerical error can accumulate over time, and the computational cost is high since many pseudoinverses of non-sparse Jacobians must be computed. Other local methods [35] use an augmented set of variables $(\dot{\mathbf{x}}, \boldsymbol{\theta})$ rather than just $\dot{\mathbf{x}}$. Another local method (well-known in articulatory inversion) is *analysis-by-synthesis*, which directly finds an inverse value $\boldsymbol{\theta}$ of $\mathbf{f}$ by iteratively minimizing the squared error $E(\boldsymbol{\theta}) = \|\mathbf{x} - \mathbf{f}(\boldsymbol{\theta})\|^2$ with a numerical optimization method, e.g. gradient descent, where $\nabla E = 2\,\mathbf{J}(\boldsymbol{\theta})^T(\mathbf{f}(\boldsymbol{\theta}) - \mathbf{x})$. Unfortunately, which inverse value is found depends on the initial value for $\boldsymbol{\theta}$, and the iteration may also get stuck at non-inverse values where $\mathbf{J}(\boldsymbol{\theta})^T(\mathbf{f}(\boldsymbol{\theta}) - \mathbf{x}) = \mathbf{0}$ but $\mathbf{f}(\boldsymbol{\theta}) \neq \mathbf{x}$. However, the method is useful if the initial $\boldsymbol{\theta}$ is sufficiently close to the inverse sought.

Global methods *Global* methods [97, 93] propose a variational approach where the trajectory of $\boldsymbol{\theta}$ minimizes a functional $\int_{t_0}^{t_1} G(\boldsymbol{\theta}, \dot{\boldsymbol{\theta}}, t)\, dt$ (such as energy and manipulability) subject to the forward kinematic constraint $\mathbf{x}(t) = \mathbf{f}(\boldsymbol{\theta}(t))$ and appropriate boundary conditions. The trajectory is obtained by numerical integration of the corresponding Euler-Lagrange equation. However, the method still suffers from singularities [93] and needs the user to provide boundary conditions that are often unknown. Thus, an important problem of many of these methods are the singularities of the Jacobian. These correspond to the intersection of multiple inverse branches (violating the inverse function theorem), and while locally any of these branches is valid a priori, globally perhaps only one is valid.

Data-driven methods A different type of methods is based on machine learning (data-driven) techniques. These methods estimate the inverse mapping using a training set of input-output pairs $(\mathbf{x}, \boldsymbol{\theta})$. While $\mathbf{f}$ may be learned by fitting a (say) neural net directly to pairs $(\boldsymbol{\theta}, \mathbf{x})$, learning the inverse mapping by fitting a neural net to pairs $(\mathbf{x}, \boldsymbol{\theta})$ would average the different inverse branches, yielding invalid solutions. For example, consider the function $x = f(\theta) = \theta^2$; the least-squares error $\|\theta - g(x)\|_2^2$ is minimized by $g(x) = 0$, the average of the two branches $\pm\sqrt{x}$, and this is what a neural net would yield. The distal learning approach [67] first trains a neural net to model the forward kinematics $\mathbf{f}$; then another net is prepended to this, and the resulting, cascaded network is retrained to learn the identity but keeping unchanged the weights of the forward model. This results in the prepended portion of the network learning one of the possible inverses (with the other branches being irreversibly lost). DeMers and Kreutz-Delgado [31, 32] try to identify (by clustering) subsets of the data corresponding to different branches, i.e., representing one-to-one mappings, and then they fit to each of them a neural net. However, in practice it is hard to identify such subsets. D'Souza et al. [35] fit a locally weighted projection regression to map $(\dot{\mathbf{x}}, \boldsymbol{\theta})$ to $\dot{\boldsymbol{\theta}}$ as in the local methods discussed above.

C.3 Trajectory inverse kinematics by conditional density modes

We will demonstrate our method on several robot arms including one industrial arm. It is important to point out that our method is not restricted to these arms but generally applicable to other arms. We apply our method, conditional density modes, discussed in chapter 4 to trajectory IK. A recap of the method is given below. Assume we have a training set of pairs $(\boldsymbol{\theta}, \mathbf{x})$ with $\mathbf{x} = \mathbf{f}(\boldsymbol{\theta})$, where $\mathbf{f}$ is the forward kinematics mapping (if we do not know $\mathbf{f}$, we could estimate it by fitting e.g. a neural net). At run time, given a trajectory $\mathbf{x}_1, \dots, \mathbf{x}_T$ in workspace, we want to obtain a trajectory $\boldsymbol{\theta}_1, \dots, \boldsymbol{\theta}_T$ of joint angles that yields the $\mathbf{x}$-trajectory while avoiding discontinuous jumps in $\boldsymbol{\theta}$-space. Our method works as follows. *Offline*, it estimates a density model $p(\boldsymbol{\theta}, \mathbf{x})$ for both variables, or just a conditional density $p(\boldsymbol{\theta}|\mathbf{x})$, using the training set. At *run time*, for each $n = 1, \dots, T$ we obtain the conditional density $p(\boldsymbol{\theta}|\mathbf{x}_t)$ and its modes; the latter explicitly represent the multiple inverse solutions at each $\mathbf{x}_t$. Then, we obtain the $\boldsymbol{\theta}$-trajectory by minimizing a constraint over the entire set of modes.

C.3.1 Experimental results

We show proof-of-concept experiments for several simple robot arms. Our goal is to illustrate the methods' performance with known ground truth for different settings. We consider the following methods: the Jacobian pseudoinverse (local method baseline); a conditional mean method, which estimates a univalued inverse mapping (as a neural net would do); and our conditional modes method, which estimates multivalued mappings and disambiguates the solution by minimizing a global constraint. We study different choices of the density model (full and conditional) and of the global constraint (continuity constraint $\mathcal{C}$, smoothness constraint $\mathcal{S}$, forward constraint $\mathcal{F}$).

Planar 2-link robot arm

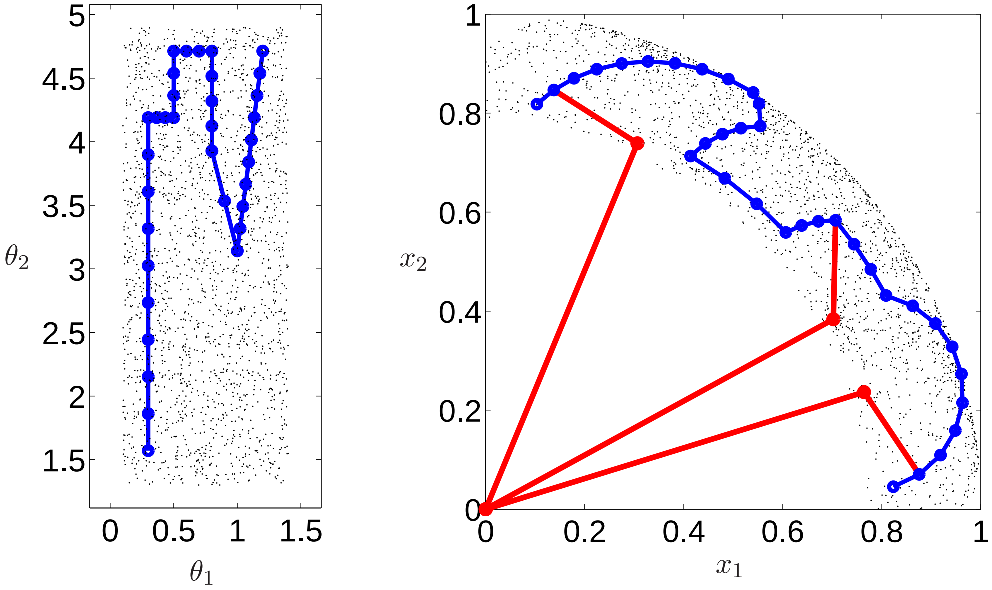


Figure C.2: Geometry of the planar 2-link arm of sec. C.3.1 (left: $\boldsymbol{\theta}$-space, right: $\mathbf{x}$-space). The black dots are the training set of pairs $(\boldsymbol{\theta}, \mathbf{x}) \in \mathbb{R}^4$, which indicate the reachable region of workspace and which are used to estimate the joint density model offline. The blue curve is a sample trajectory to be reconstructed at run time, and the red lines schematically represent the robot arm in 3 different configurations. Points near the two ends of the workspace can only be reached by one configuration because of limits on θ_1.

First, we consider a planar 2-link robot arm (fig. C.2) for which it is possible to visualize the conditional density and study the method. The forward mapping is

$$x_1 = l_1 \cos\theta_1 + l_2 \cos(\theta_1 + \theta_2)$$

$$x_2 = l_1 \sin\theta_1 + l_2 \sin(\theta_1 + \theta_2)$$

where $l_1 = 0.8$ and $l_2 = 0.2$. The inverse mapping can be computed analytically and has 2 solutions (elbow up/down). Singularities occur when $|\mathbf{J}(\boldsymbol{\theta})| = |l_1 l_2 \sin\theta_2| = 0 \Leftrightarrow \theta_2 = 0,\ \pm\pi$, i.e., when the arm is fully stretched or folded. To make the problem more complex, we limit the $\boldsymbol{\theta}$-domain to $[0.3, 1.2] \times [1.5, 4.7]$ rad so that certain branches are invalid in certain regions of the workspace. For example, the region at the right end of the workspace is only reachable as elbow-up, and the region at the left end as elbow-down. More generally, the feasible $\boldsymbol{\theta}$-domain could have a very complicated shape, where the range of allowed values for a single angle θ_i depends on the values of other angles θ_j, e.g. to avoid self-intersections in a humanoid robot. Respecting these constraints is simple in our method, since the training set will only contain feasible configurations by construction, and the density modes will always lie on high-density regions (not so the mean!). The trajectory in fig. C.2 goes through singularities (when the arm is fully stretched); a local method may choose a branch that later on is unable to reach the trajectory, but our method can choose the correct branch by keeping track of all local solutions and then disambiguating them with the global constraint.

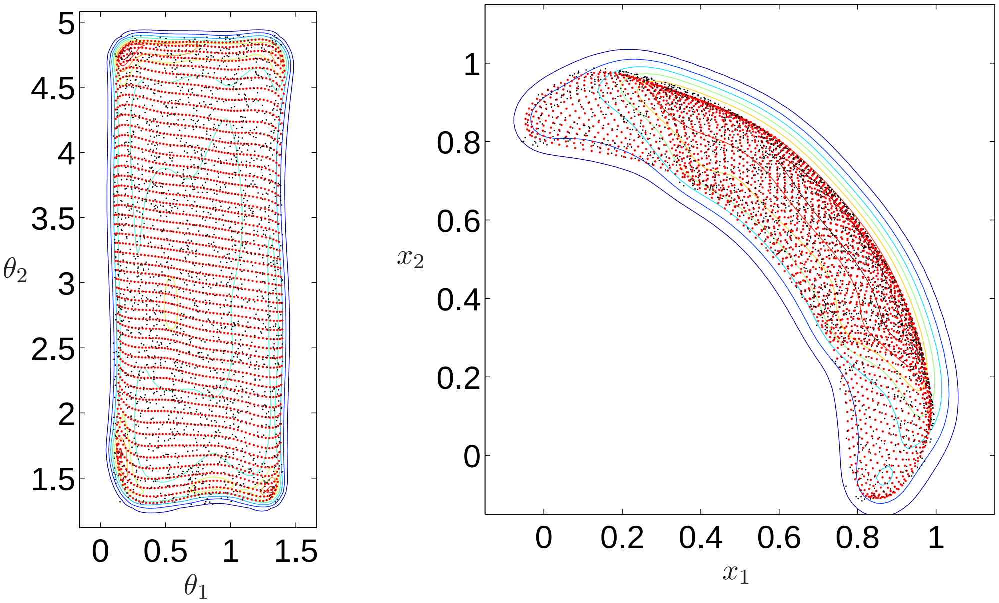


Figure C.3: Marginal densities $p(\boldsymbol{\theta})$ (left) and $p(\mathbf{x})$ (right) for the fine GTM model $p(\boldsymbol{\theta}, \mathbf{x})$ (4–dimensional), as a contour plot. The component centers of the Gaussian mixture are indicated by red dots.

We generated a training set of 2 000 pairs $(\boldsymbol{\theta}, \mathbf{x})$ by uniformly sampling the $\boldsymbol{\theta}$-space[1] and mapping with $\mathbf{f}$ (black dots in fig. C.2). We trained density models by maximum likelihood:

---

[1] We included samples in a slightly larger domain $[0.1, 1.4] \times [1.3, 4.9]$ to avoid boundary effects in the density model. For GTM, we also added a bit of noise (with standard deviation of 0.05) to improve the smoothness of the resulting density.

- Full density $p(\mathbf{x}, \boldsymbol{\theta})$: we could have trained a GM directly, but instead we trained a generative topographic mapping (GTM) model [18], since the intrinsic dimensionality of $(\mathbf{x}, \boldsymbol{\theta})$ is 2 (not 4, because of the forward mapping). GTM is a latent variable model that yields a GM constrained to lie in a low-dimensional manifold. We tried 2 GTM models, one coarse (with $M = 225$ components in the GM) and one fine (with $M = 2\,500$). Fig. C.3 shows the resulting density, or rather the marginals $p(\boldsymbol{\theta})$ and $p(\mathbf{x})$ for visualization purposes.

- Conditional density $p(\boldsymbol{\theta}|\mathbf{x})$: we used a mixture density network (MDN) [16]. This is a particular case of mixtures of experts [66] that yields a GM

$$p(\boldsymbol{\theta}|\mathbf{x}) = \sum_{m=1}^{M} \pi_m(\mathbf{x}) \mathcal{N}(\boldsymbol{\theta}; \boldsymbol{\mu}_m(\mathbf{x}), \sigma_m(\mathbf{x})) \tag{C.1}$$

  where the functions $\pi_m(\mathbf{x})$, $\boldsymbol{\mu}_m(\mathbf{x})$ and $\sigma_m(\mathbf{x})$ are neural nets. We used $M = 2$ components and 2-layer neural nets with 10 hidden units. Note a MDN is different from a neural net; the latter is a uni-valued function, while the MDN represents multimodal densities, whose number of modes depends on $\mathbf{x}$.

Fig. C.4 shows, for each model, the conditional density for a particular $\mathbf{x}$ value. The conditional density model (MDN) gives a sharply peaked density with 2 modes near the true inverses. The fine GTM model gives also a bimodal density but less sharp, and the coarse GTM model gives a multimodal density where spurious modes arise along the line connecting the true inverses. The reason for this is the interference from the additional dimensions (for $\mathbf{x}$) that GTM is modeling, so that more components are necessary to achieve an accurate conditional density. However, as seen below, all 3 models succeed in recovering the true trajectory thanks to the forward constraint $\mathcal{F}$ (which filters out the spurious modes).

Figs. C.5–C.6 show the reconstructed trajectories for each density model (we obtained similar results with other trajectories). We also show the trajectory that results from using the mean of the conditional density. This yields the GM regression mapping and is essentially equivalent to fitting a neural net directly to pairs $(\mathbf{x}, \boldsymbol{\theta})$. Since it can only represent a uni-valued mapping, it averages the two inverse branches, resulting in the fully stretched configuration, which is invalid (i.e., it does not equal the desired $\mathbf{x}$) for most $\mathbf{x}$; it is valid where the inverse is uni-valued, namely at the ends of the workspace. When using conditional modes, all 3 density models (MDN, coarse GTM, fine GTM) succeed in reconstructing the true trajectory with good accuracy, but more importantly, yielding a globally correct trajectory that chooses the appropriate branch at all steps.

It is very interesting to note that the $\mathbf{x}$-trajectory of fig. C.2 can actually be produced by different $\boldsymbol{\theta}$-trajectories (fig. C.7). In theory, they all have exactly the same value for the global constraint, but

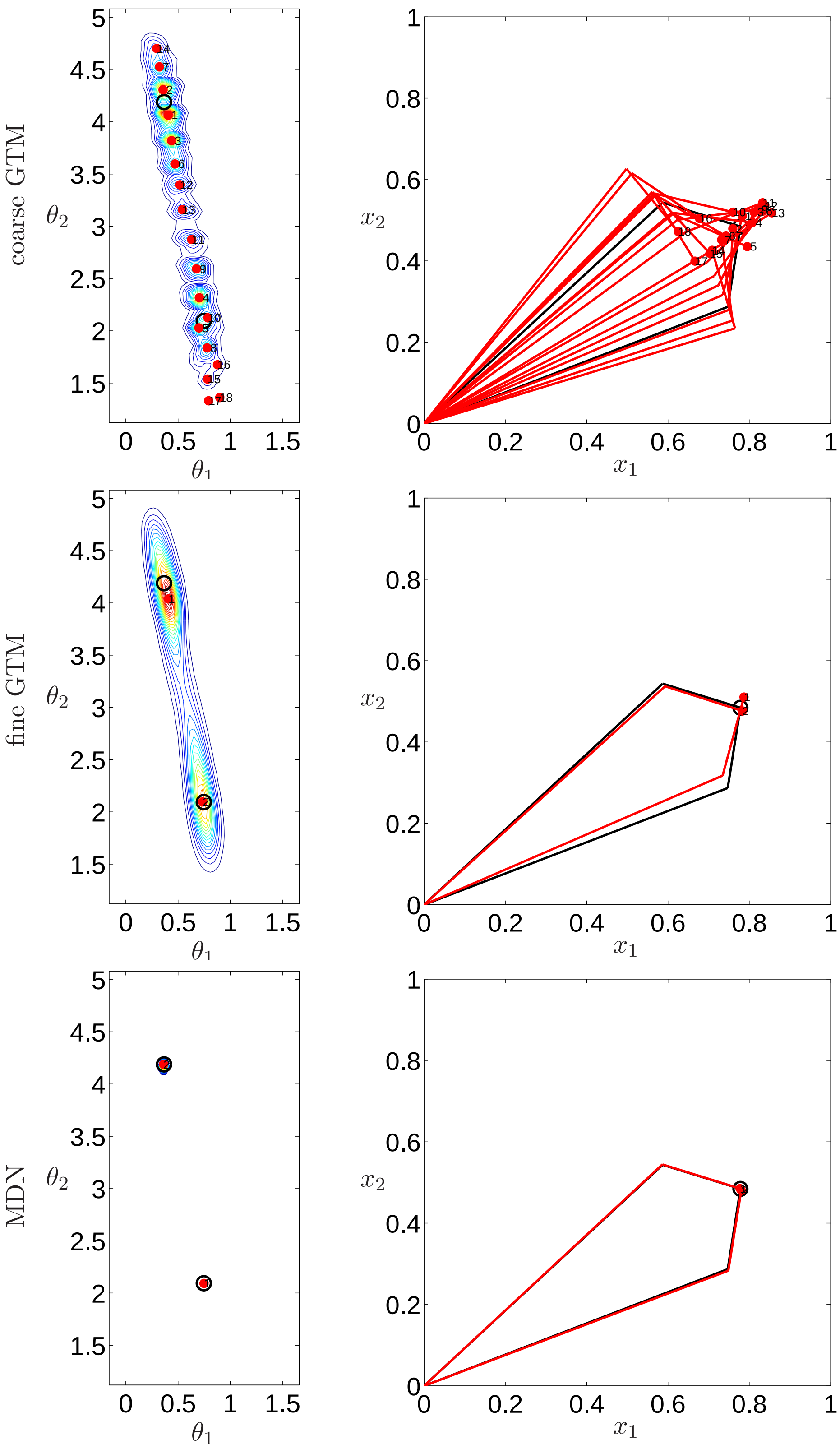


Figure C.4: Sample conditional densities $p(\boldsymbol{\theta}|\mathbf{x} = (0.78, 0.48))$ for 3 models: coarse GTM (top, 18 modes), fine GTM (middle, 2 modes) and MDN (bottom, 2 modes). *Left*: contours of the conditional density in $\boldsymbol{\theta}$-space, its modes (red dots) and the true inverses (black circles). *Right*: robot arm configurations for the modes (red) and true inverses (black).

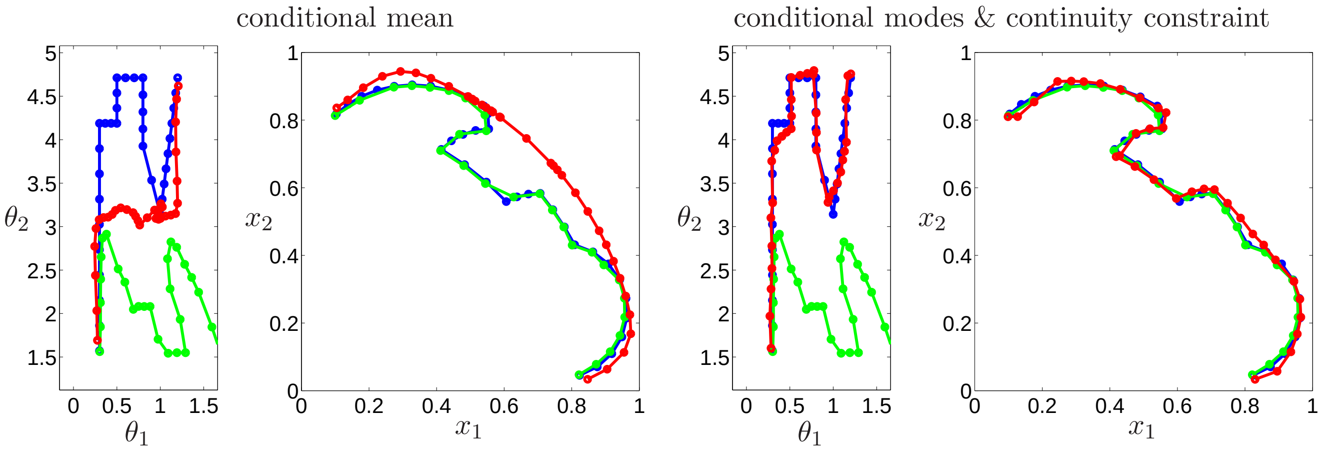


Figure C.5: True (blue) and reconstructed trajectories with the fine GTM model (red) and the pseudoinverse (green). *Left 2*: using the conditional mean, *Right 2*: using the modes and the continuity constraint $\mathcal{C}$. The pseudoinverse solution is one of the trajectories of fig. C.7.

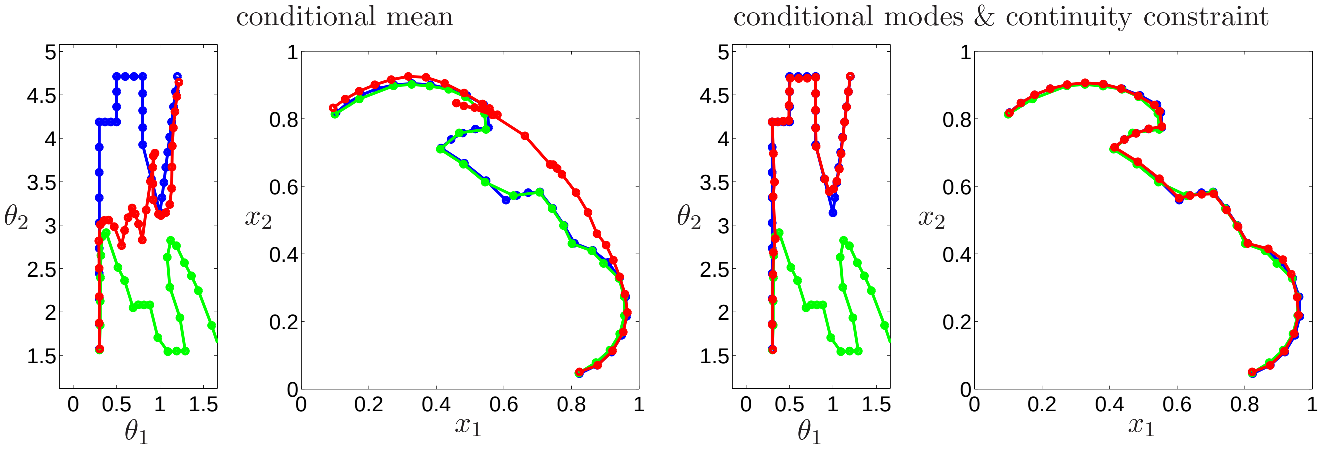


Figure C.6: As fig. C.5 but for the MDN model (red).

in practice they differ slightly due to the particular training set and model used. The pseudoinverse method, being local, can only find one of these trajectories (fig. C.5–C.6, green). In our method, the dynamic programming search considers all these trajectories and selects the one with globally minimal constraint value. However, if (say) the region $[1, 1.5] \times [1.5, 2]$ of $\boldsymbol{\theta}$-space were not allowed (e.g. because of mechanical constraints) then the trajectory found by the pseudoinverse method would be invalid; a local method has to decide which inverse branch to take at the singularity near $\boldsymbol{\theta} = (0.3, 3)$ and does not benefit from the information about the forbidden $\boldsymbol{\theta}$–rectangle that lies in the future (assuming the trajectory starts near $\boldsymbol{\theta} = (0.3, 1.6)$). Our method does benefit from it by learning (through the training set) only those regions and branches that are actually feasible and succeeds in reconstructing the correct trajectory.

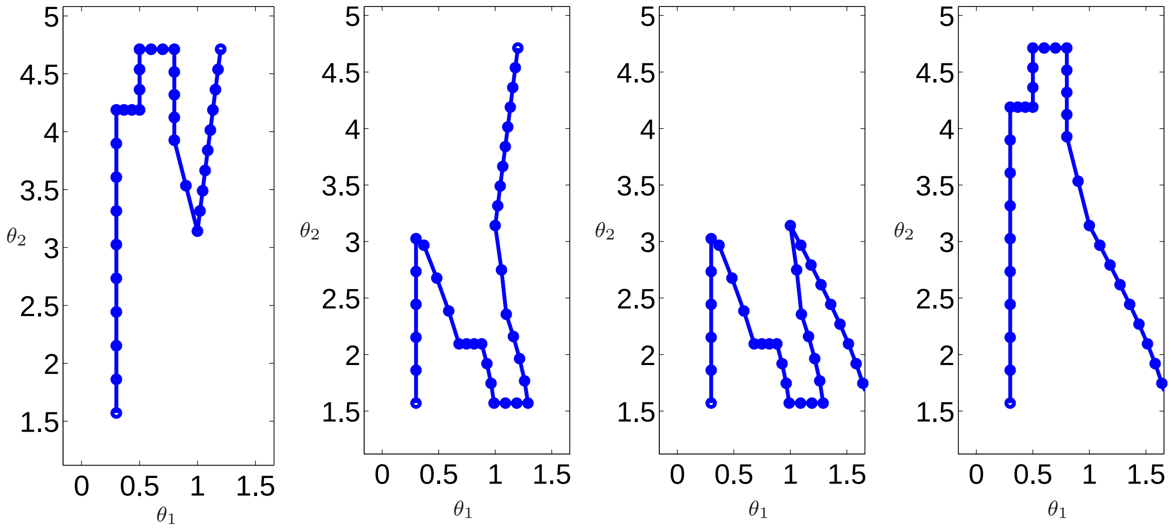


Figure C.7: Four trajectories in $\boldsymbol{\theta}$-space that produce the same $\mathbf{x}$-trajectory of fig. C.2 (blue) for the planar 2-link robot arm.

Table C.1 gives the errors in $\boldsymbol{\theta}$ and $\mathbf{x}$ w.r.t. the true trajectory (true = any of fig. C.7). For $\mathbf{x}$ they are of around 2% of the length of the fully stretched arm ($l_1 + l_2 = 1$) for the fine and coarse GTM models, and of 0.5% for the MDN model. These errors are very close to the "oracle" column, which gives the error achieved if the closest modes to the true solution were selected. We could refine the trajectory and reduce the error as much as desired in a post-processing step by initializing an analysis-by-synthesis search at each point in the trajectory. We find that the continuity constraint alone is enough to find the correct trajectory with the MDN and the fine GTM model, but not with the coarse GTM model, because of the spurious modes it has (which provide shortcuts that the continuity constraint favors). However, adding the forward constraint $\mathcal{F}$ as $\mathcal{C} + \lambda\mathcal{F}$ (over a wide range of $\lambda > 0$) yields the correct trajectory for all methods. The smoothness constraint $\mathcal{S}$ performs as well as the continuity constraint $\mathcal{C}$. The errors when using the mean of the density are considerably larger, but only the figures show how truly bad its solutions are.

Angle reconstruction error $\frac{1}{T}\sum_{t=1}^{T} \|\boldsymbol{\theta}_t - \hat{\boldsymbol{\theta}}_t\|$ (rad)

Model	mean	oracle	$\mathcal{C}$	$\mathcal{S}$	$\mathcal{C}+\lambda\mathcal{F}$	$\mathcal{S}+\lambda\mathcal{F}$
coarse GTM	0.783	0.083	0.628	0.704	0.118	0.122
fine GTM	0.798	0.114	0.114	0.127	0.114	0.127
MDN	0.668	0.037	0.037	0.037	0.037	0.037
pseudoinv	0.06					

Workspace reconstruction error $\frac{1}{T}\sum_{t=1}^{T} \|\mathbf{x}_t - \mathbf{f}(\hat{\boldsymbol{\theta}}_t)\|$

Model	mean	oracle	$\mathcal{C}$	$\mathcal{S}$	$\mathcal{C}+\lambda\mathcal{F}$	$\mathcal{S}+\lambda\mathcal{F}$
coarse GTM	0.084	0.024	0.094	0.097	0.022	0.028
fine GTM	0.084	0.021	0.021	0.021	0.021	0.021
MDN	0.072	0.005	0.005	0.005	0.005	0.005
pseudoinv	0.016					

Table C.1: Reconstruction errors for the 2D robot arm by conditional modes

Both the pseudoinverse and our method can achieve low reconstruction error, depending on the chosen number of iterations and of GM components. Besides its ability to ensure globally feasible trajectories, our method has the advantage of being less sensitive to singularities. Near singularities, our mean-shift algorithm converges sublinearly while the pseudoinverse method is numerically unstable and could take far more iterations to converge.

PUMA 560 robot arm with 6 DOF

Figs. C.11–C.12 and table C.2 show similar experiments for a PUMA 560 robot arm (see fig. C.1 and fig. C.8) with 3 DOF for position $\boldsymbol{\theta} = (\theta_1,\ \theta_2,\ \theta_3)$, 3 DOF for orientation (which we ignore), and a 3D workspace $\mathbf{x} \in \mathbb{R}^3$. The (point) IK can be solved analytically for this robot [106] and yields 4 solution branches (two combinations of elbow up/down; fig. C.10); we use the implementation of the Matlab Robotics Toolbox [29]. As before, we limit the angle domain in order to complicate the topology of the inverse mapping, and generate a training set of 5 000 pairs $(\boldsymbol{\theta}, \mathbf{x})$ (shown in fig. C.9). The GTM (full density model) that we trained (results not shown) failed to produce a good reconstruction because of the existence of multiple spurious modes. The reasons for this are the higher dimensionality of the space, but also the fact that GTM is practically limited to an intrinsic dimensionality of at most 2, while in this case the intrinsic dimensionality is 3. We also trained a MDN (conditional density model) with

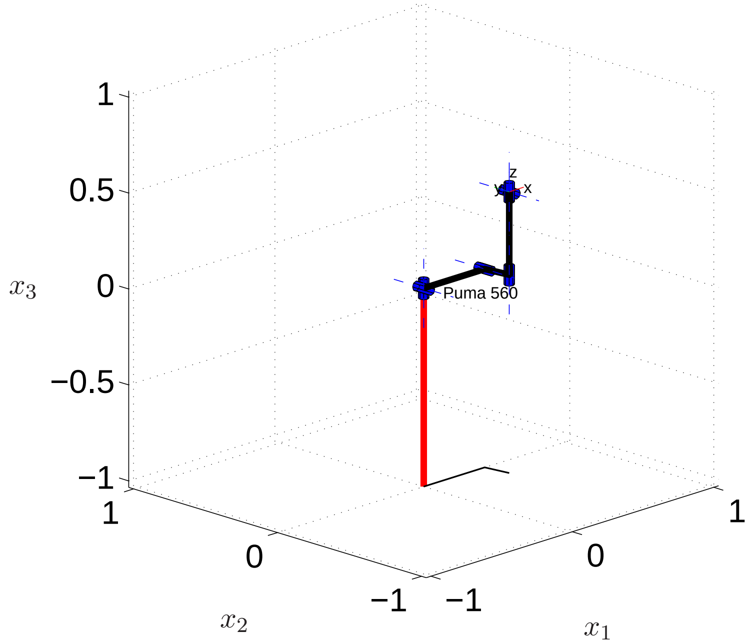


Figure C.8: A schematic plot of PUMA560 (reproduced with Matlab Robotics toolbox [29])

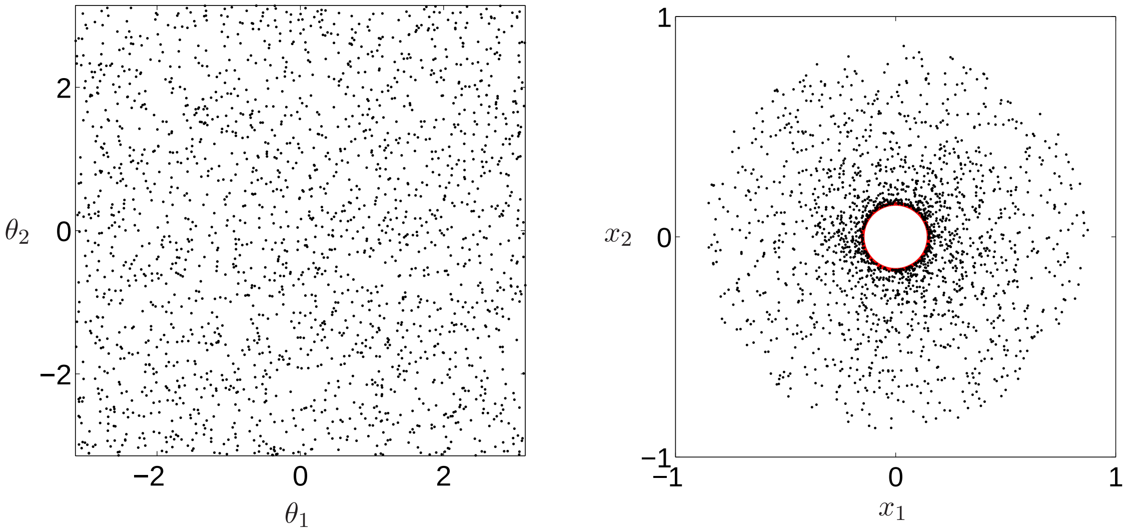


Figure C.9: Training set for the PUMA 560 robot arm of sec. C.3.1 (top views, left: $\boldsymbol{\theta}$-space, right: $\mathbf{x}$-space). The workspace contains an unreachable region shaped like a vertical cylinder passing through the robot foot.

$M = 12$ components (and neural nets with 300 hidden units), which did succeed in reconstructing various trajectories, with errors of similar magnitude as with the planar arm of sec. C.3.1; we show a sample of results, for 3 trajectories (an elliptical closed loop, a figure–8 closed loop, and an open trajectory; figs. C.11–C.12, table C.2). Again, the symmetry of the problem results in several equivalent global solutions; the pseudoinverse and our method choose different ones. The larger errors occur for points near a cylindrical hole at the center of the workspace which is not reachable by the robot, because of boundary effects of the density model. They could be reduced by increasing the number of components in the GM, or more efficiently by refining the trajectory with a local method. The "oracle" (best achievable) error (not shown) was very similar to that of $\mathcal{C} + \lambda\mathcal{F}$.

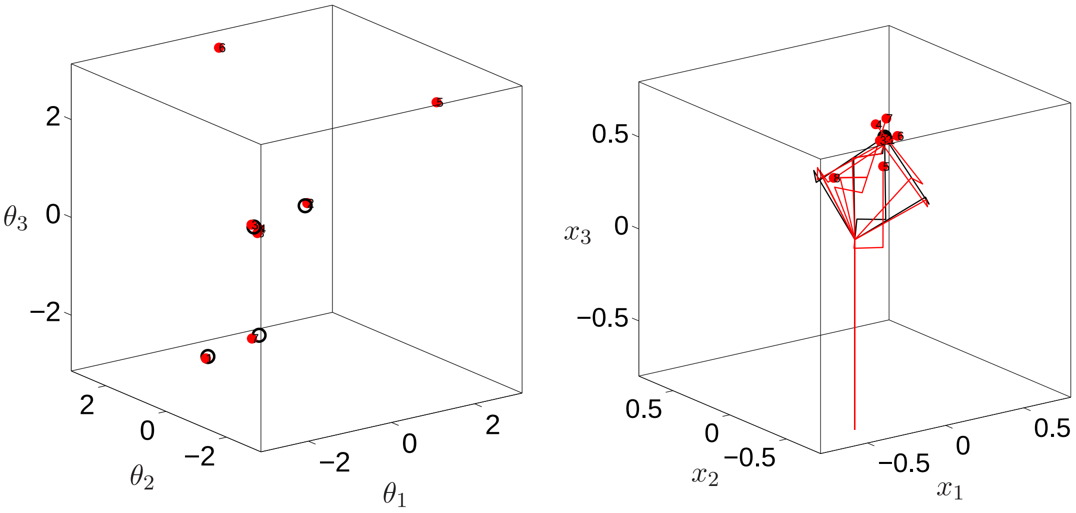


Figure C.10: *Left*: modes (red dots) for the conditional density $p(\boldsymbol{\theta}|\mathbf{x})$ for the MDN model and the PUMA 560 robot arm. There are 4 true inverses (black circles), which are well represented by the modes, but there are also two spurious modes (which are removed by the forward constraint $\mathcal{F}$, since they map far from the desired $\mathbf{x}$). *Right*: modes and true inverses in workspace, represented as schematic arms.

Redundant planar 3-link robot arm

When $\dim \boldsymbol{\theta} > \dim \mathbf{x}$ (redundant manipulator), an infinite number of inverses $\boldsymbol{\theta}$ exist for a given $\mathbf{x}$. The corresponding density $p(\boldsymbol{\theta}|\mathbf{x})$ would ideally be uniform over this set of inverses. Instead, because we use a Gaussian mixture, this uniform density becomes approximate and has multiple modes distributed over the set of inverses. Thus, these modes act as a quantized representation of the inverse set, and are available for use by the global constraint (which could also incorporate terms suggested by arguments of movement economy, such as integrated jerk or torque). We show this with a planar 3-link robot arm with 3 dof for $\boldsymbol{\theta}$ (link lengths: 3, 2.5, 2; foot at $\mathbf{x} = \mathbf{0}$) and a 2D workspace $\mathbf{x} \in \mathbb{R}^2$ (see fig. C.1 for illustration).

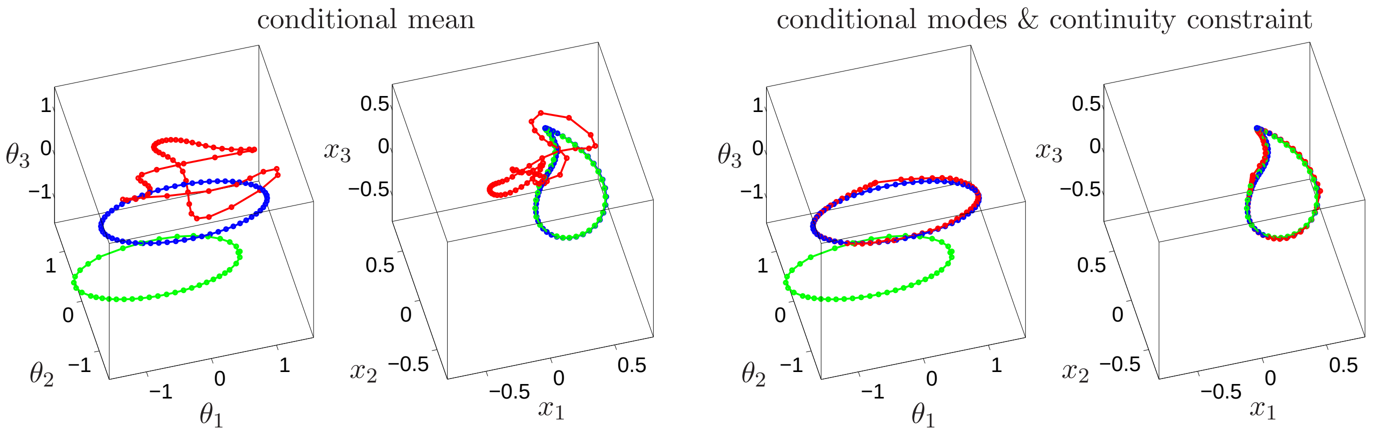


Figure C.11: Reconstruction of an elliptical trajectory (blue) for the PUMA 560 robot arm: MDN (red); pseudoinverse (green). *Left 2*: mean of the density, *Right 2*: modes and continuity constraint $\mathcal{C}$.

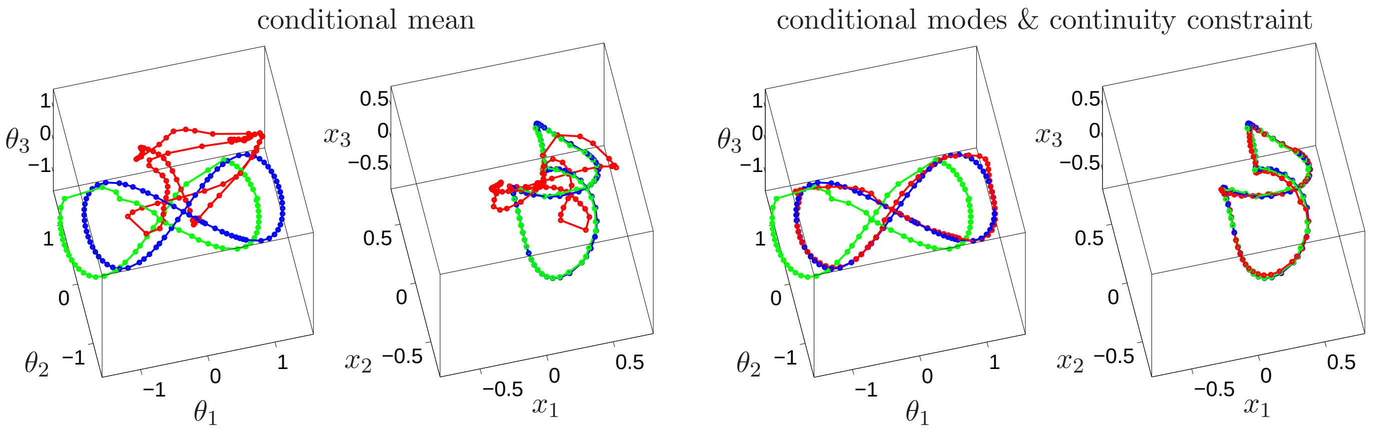


Figure C.12: As fig. C.11 but for a figure–8 trajectory.

Angle reconstruction error $\frac{1}{T}\sum_{t=1}^{T} \|\boldsymbol{\theta}_t - \hat{\boldsymbol{\theta}}_t\|$ (rad)

Traj.	pseudoinv	mean	$\mathcal{C}$	$\mathcal{S}$	$\mathcal{C} + \lambda\mathcal{F}$	$\mathcal{S} + \lambda\mathcal{F}$
Ellipse	0.072	2.110	0.076	0.069	0.071	0.069
Figure–8	0.076	1.990	0.082	0.081	0.081	0.080
Open	0.042	2.140	0.173	0.778	0.173	0.176

Workspace reconstruction error $\frac{1}{T}\sum_{t=1}^{T} \|\mathbf{x}_t - \mathbf{f}(\hat{\boldsymbol{\theta}}_t)\|$

Traj.	pseudoinv	mean	$\mathcal{C}$	$\mathcal{S}$	$\mathcal{C} + \lambda\mathcal{F}$	$\mathcal{S} + \lambda\mathcal{F}$
Ellipse	0.025	0.819	0.030	0.029	0.029	0.029
Figure–8	0.019	0.750	0.028	0.027	0.027	0.027
Open	0.007	0.665	0.055	0.080	0.055	0.055

Table C.2: Reconstruction errors for the 3D robot arm (PUMA 560) by MDN

We generate a training set in a subset of $[0, 2\pi]^3$ and train a MDN ($M = 36$ components, neural nets with 300 hidden units). Figs. C.13–C.14 and table C.3 show experiments for three trajectories in $\mathbf{x}$–space (a circle, a loopy trajectory with self-intersections and a figure–8). The larger errors occur when the robot arm is close to fully-stretched configurations (corresponding to singularities). Both the pseudoinverse and our method are able to retrieve continuous (but different) trajectories in $\boldsymbol{\theta}$–space. As before, near singularities the pseudoinverse method is unstable and takes many iterations to converge.

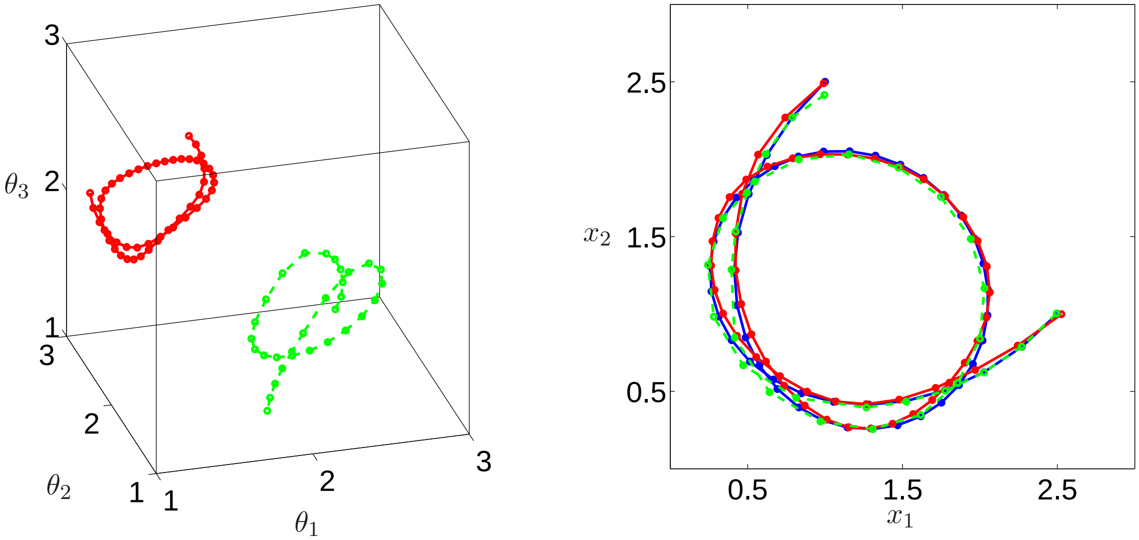


Figure C.13: Reconstruction of the loopy trajectory for the redundant arm: MDN with constraint $\mathcal{C} + \lambda\mathcal{F}$ (red); pseudoinverse (green).

Workspace reconstruction error $\frac{1}{T}\sum_{t=1}^{T} \|\mathbf{x}_t - \mathbf{f}(\hat{\boldsymbol{\theta}}_t)\|$

Model	pseudoinv	mean	$\mathcal{C}$	$\mathcal{S}$	$\mathcal{C} + \lambda\mathcal{F}$	$\mathcal{S} + \lambda\mathcal{F}$
Circle	0.060	4.610	0.173	0.185	0.140	0.161
Loopy	0.106	3.970	0.231	0.040	0.040	0.040
Figure–8	0.135	3.930	0.069	0.069	0.069	0.069

Table C.3: Reconstruction errors for the redundant manipulator by MDN

C.3.2 Discussions

Our method, by directly representing multivalued mappings and using a global constraint, is able to achieve feasible, globally correct solutions to trajectory IK even in the presence of (1) singularities of the Jacobian, where the forward mapping has multiple local inverses, and (2) complicated angle domains, which are captured through the training set. The power of the density model is its flexibility:

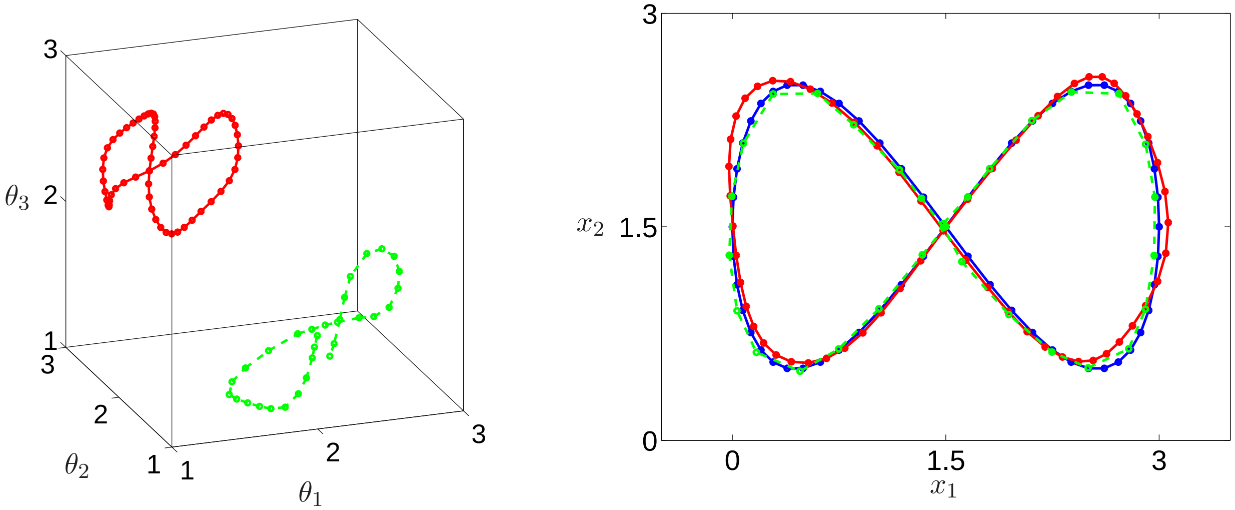


Figure C.14: As fig. C.13 but for a figure–8 trajectory.

in principle, it represents implicitly (through its modes) all the feasible solution branches once and for all, even when their topology can be very complex (e.g. with a number of branches that depends on $\mathbf{x}$) because of the nonlinearity of the forward mapping, or because of mechanical constraints. The disadvantage is that (1) the mappings are implicit, and must be made explicit at run time by mode finding, which causes a computational cost; and (2) the inverse values $\mathbf{x}_t$ returned are approximations, i.e., $\mathbf{f}(\mathbf{x}_t)$ is not exactly $\mathbf{y}_t$. We discuss several aspects of the method next.

Data collection In common with other machine learning methods, we need a training set of pairs $(\boldsymbol{\theta}, \mathbf{x})$. These can be collected by sampling the $\boldsymbol{\theta}$-space and computing $\mathbf{x} = \mathbf{f}(\boldsymbol{\theta})$, if the forward mapping $\mathbf{f}$ is known, or by recording $(\boldsymbol{\theta}, \mathbf{x})$ while the robot is performing a task (perhaps imitating a human). This has the advantage of yielding valid pairs (by definition) and sampling only those areas of $\boldsymbol{\theta}$-space that correspond to *typical* motion, rather than feasible but atypical motions. This advantage already appeared in articulatory inversion. Besides, typical behavior may result in correlations between joints that reduce the intrinsic dimensionality of the $\boldsymbol{\theta}$-space. This idea is being exploited in motion-capture systems and has wide applicability in IK in graphics [56] and articulated pose tracking in computer vision [152, 26].

Accuracy of the solution The density model need not be overly accurate; it suffices to yield modes near the true solution, and spurious modes (if there are not too many of them) may be filtered out by the forward constraint. Being data-driven, a limitation of our method is that the estimated global trajectories are not perfectly accurate. This is because the modes (even if computed exactly) do not necessarily coincide with the inverses, depending on the number of components in the GM. However, they can be made very close to the true trajectory and may be refined a posteriori if desired by a local method (e.g. resolved motion rate control or analysis-by-synthesis), that, on its own, might not find a globally correct solution.

Run time In practice, the run time is dominated by the mode-finding step, which takes $\mathcal{O}(kM^2)$ [23] where M is the number of components in the GM and k the average number of iterations per component (≈ 50). When using a full density model $p(\mathbf{x}, \boldsymbol{\theta})$, M is very large, which prevents use in real time. But with a conditional density model $p(\boldsymbol{\theta}|\mathbf{x})$ (e.g. a MDN), which besides is more accurate, we can limit M to a number slightly larger than the (estimated) maximum number of solution branches for all $\mathbf{x}$, which is far smaller. The mode-finding algorithms, e.g. Gaussian mean-shift, can also be significantly accelerated [25], again noting that there is no need to converge with large accuracy. Our method does not use the Jacobian and needs no matrix inversions. In our unoptimized Matlab implementation for the PUMA arm, our method took 50/10/4 ms per point (worst/average/best), while the pseudoinverse method took 200/30/10 ms.

In summary, our method can obtain very accurate solutions if a GM with a large enough number of components is used. However, its main strength is in being able to find a globally feasible solution without suffering from singularities of the Jacobian (since it does not use the Jacobian or possibly even a closed-form forward mapping), and dealing in a natural way (through the training set) with complex angle domains that are very difficult to express in analytical form.

C.4 Trajectory inverse kinematics by nonlinear and nongaussian tracking

In this section, we propose a tracking approach to IK. That is, we consider IK as a tracking problem. Under the tracking framework, we aim to estimate angles $\boldsymbol{\theta}$ (= unobserved states) of the kinematic system (= a dynamic system or a state-space model) from past and current coordinates $\mathbf{x}$ (= observed measurements) recursively. This framework is particularly suitable to trajectory IK because the input trajectory in workspace is a typical representative of sequential data, e.g. time series [17]. As the most well-known linear and Gaussian tracker, Kalman filter assumes (1) the unobserved states, as well as the observed measurements, are multivariate Gaussian distributions. (2) the state-space model is linear. Therefore, the posterior distribution of unobserved states $\boldsymbol{\theta}_t$ given observations $\mathbf{x}_1, \dots, \mathbf{x}_t$ is also Gaussian and the state inference (i.e., computing the posterior and deriving meaningful statistics from it) can be done exactly. However, such assumption become inappropriate in IK because the posterior is typically multimodal (i.e., each mode corresponds to a different arm configuration). We address this issue by considering nonlinear, nongaussian tracker, e.g. particle filters (PFs). In PFs, the restrictions on Gaussian distributions are relaxed and the posterior can adopt arbitrary distribution. The price to pay is that the exact inference becomes intractable. To our best knowledge, this is the first approach to IK based on (nonlinear, nongaussian) tracking.

Consider a given $\mathbf{x}$–trajectory $\mathbf{x}_1, \dots, \mathbf{x}_T$ in workspace. Our overall algorithm works as follows: (1) We run a particle filter (or smoother) to obtain at each $n = 1, \dots, T$ a conditional distribution of $\boldsymbol{\theta}$ given positions of the end-effector up to time t, $p(\boldsymbol{\theta}_t|\mathbf{x}_{1:t})$ (or the entire $\mathbf{x}$–trajectory, $p(\boldsymbol{\theta}_t|\mathbf{x}_{1:T})$). (2) We run a mode-finding algorithm on each distribution to find all its modes, which represent the multiple inverse solutions at each $\mathbf{x}_t$. (3) We obtain a unique $\boldsymbol{\theta}$–trajectory by minimizing a constraint $\mathcal{C} + \lambda\mathcal{F}$ over the entire set of modes with dynamic programming. Essentially, the basic framework (steps 2 and 3) in the conditional modes of section 4.3 remains and what changes is now we learn online the conditional distribution $p(\boldsymbol{\theta}|\mathbf{x}_{1:t})$ or $p(\boldsymbol{\theta}|\mathbf{x}_{1:T})$ that is estimated by a particle filter or smoother at each time. In addition, we also apply various linear and nonlinear trackers independently to infer the angles at time t given positions of the end-effector up to time t or T.

C.4.1 Conditional density by tracking

Our eventual objective is to obtain the multiple inverses of each $\mathbf{x}_t$ (i.e., values $\boldsymbol{\theta}$ s.t. $\mathbf{f}(\boldsymbol{\theta}) = \mathbf{x}_t$) from the modes of the conditional distribution of angles $\boldsymbol{\theta}$ given coordinates $\mathbf{x}$, $p(\boldsymbol{\theta}_t|\mathbf{x}_{1:t})$. We propose to construct the latter with a nonlinear, nongaussian tracker, where we consider the coordinates $\mathbf{x}$ as observed measurements and the angles $\boldsymbol{\theta}$ as unobserved states. Under the tracking framework [4, 34], as shown in fig. C.15, the dynamic state-space model is given by $\boldsymbol{\theta}_t = \boldsymbol{\theta}_{t-1} + \omega_{t-1}$, $\mathbf{x}_t = \mathbf{f}(\boldsymbol{\theta}_t) + \upsilon_t$. We model the dynamics $p(\boldsymbol{\theta}_t|\boldsymbol{\theta}_{t-1})$ as a random walk with Gaussian noise ω_t, and the measurement model $p(\mathbf{x}_t|\boldsymbol{\theta}_t)$ is readily given by $\mathbf{f}$ with Gaussian noise υ_t. If the posterior distribution $p(\boldsymbol{\theta}_t|\mathbf{x}_{1:t})$ can be assumed unimodal, *extended Kalman filters* (EKFs) [4] can succeed. But, crucially, the posterior $p(\boldsymbol{\theta}_t|\mathbf{x}_{1:t})$ for IK is multimodal due to the many-to-one $\mathbf{f}$. We then consider *particle filters* (PFs), which can approximate multimodal distributions by a set of weighted samples (the particles), and have demonstrated superior performance over EKFs in many nonlinear, nongaussian problems. Various versions of PFs have been developed, including *sequential importance resampling PF* (SIR-PF), sigma-point PF, unscented PF and others [34]. They mostly differ in the choice of the proposal distribution, whose support should cover the support of the true posterior. Here, we focus on the SIR-PF, which uses the transition prior as the proposal distribution, but other PFs would work as well as long as they can approximate multimodal posteriors. Furthermore, one can refine the posterior distribution by using all the measurements: $p(\boldsymbol{\theta}_t|\mathbf{x}_{1:T})$. This leads to *extended Kalman smoothers* (EKSs) and *particle* *smoothers* (PSs). We consider two versions of PSs: *forward-backward smoother* (FBS), which maintains the original particle locations from the PF but reweights them; and *two-filter smoother* (TFS) [74, 62], which combines a forward and a backward PF. The computational cost for M particles is $\mathcal{O}(M)$ for PFs and $\mathcal{O}(M^2)$ for PSs, which may be reduced to $\mathcal{O}(M \log M)$ by approximate, fast algorithms [75].

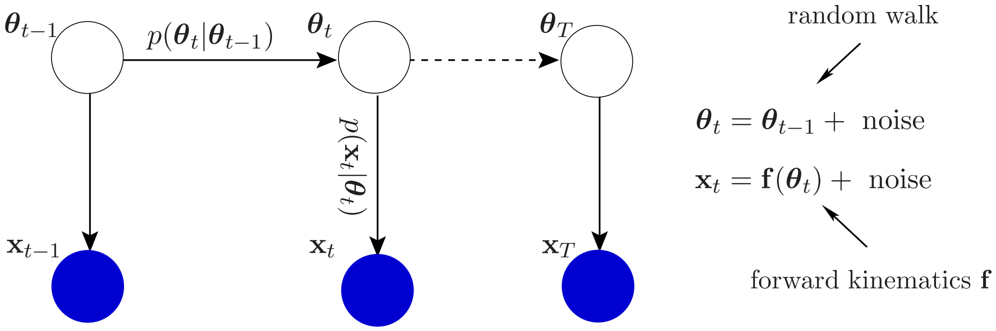

$\boldsymbol{\theta}_{t-1}$
$p(\boldsymbol{\theta}_t|\boldsymbol{\theta}_{t-1})$
$\boldsymbol{\theta}_t$
$\boldsymbol{\theta}_T$
$p(\mathbf{x}_t|\boldsymbol{\theta}_t)$
$\mathbf{x}_{t-1}$
$\mathbf{x}_t$
$\mathbf{x}_T$
random walk
$\boldsymbol{\theta}_t = \boldsymbol{\theta}_{t-1} +$ noise
$\mathbf{x}_t = \mathbf{f}(\boldsymbol{\theta}_t) +$ noise
forward kinematics $\mathbf{f}$

Figure C.15: The tracking framework for trajectory inverse kinematics. Empty cirles stand for hidden states $\boldsymbol{\theta}$. Cirles filled in blue stand for observable measurements $\mathbf{x}$.

PFs (or PSs) only provide a set $\{\boldsymbol{\theta}_t^m, w_t^m\}_{m=1}^M$ of weighted particles to approximate the posterior $p(\boldsymbol{\theta}_t|\mathbf{x}_{1:t})$ at time n. Here, we use a Gaussian kernel density estimate to construct the conditional density, $p(\boldsymbol{\theta}_t|\mathbf{x}_{1:t}) = \sum_{m=1}^M w_t^m \exp(-\left\|\boldsymbol{\theta}_t - \boldsymbol{\theta}_t^m\right\|^2 /2\sigma^2)$, where σ is the kernel width. Unlike most tracking work, we use all the modes instead of just the mean as the statistical estimate from the conditional density, as we must track multiple inverse branches.

C.4.2 Experiments with a PUMA 560 robot arm

We illustrate the performance of our tracking methods with known ground truth. All experiments were repeated 20 times with random initializations for each run in order to calculate variance estimates of performance.

We consider the PUMA 560 robot arm as in section C.3.1. We run the traditional pseudoinverse method and our approach with the following trackers: EKF/EKS [2], SIR-PF/FBS/TFS [3]. The process and measurement noises are isotropic with variances $\sigma_\omega^2 = 0.125$ and $\sigma_v^2 = 0.005$, respectively. Note that these parameters can also be estimated from the training set, if available, consisting of data for both states $\boldsymbol{\theta}$ and measurements $\mathbf{x}$. For SIR-PF/FBS/TFS we used $M = 1\,000$ particles and a kernel density estimate width $\sigma = 0.04$. Fig. C.16 illustrates (for the two-link robot arm, which is easier to

---

[2] We used the Kalman filter toolbox for Matlab by Kevin Murphy, available at `http://www.cs.ubc.ca/~murphyk/Software/Kalman/kalman.html`

[3] We used our own Matlab implementation with reference to ReBEL, available at `http://choosh.csee.ogi.edu/rebel/`.

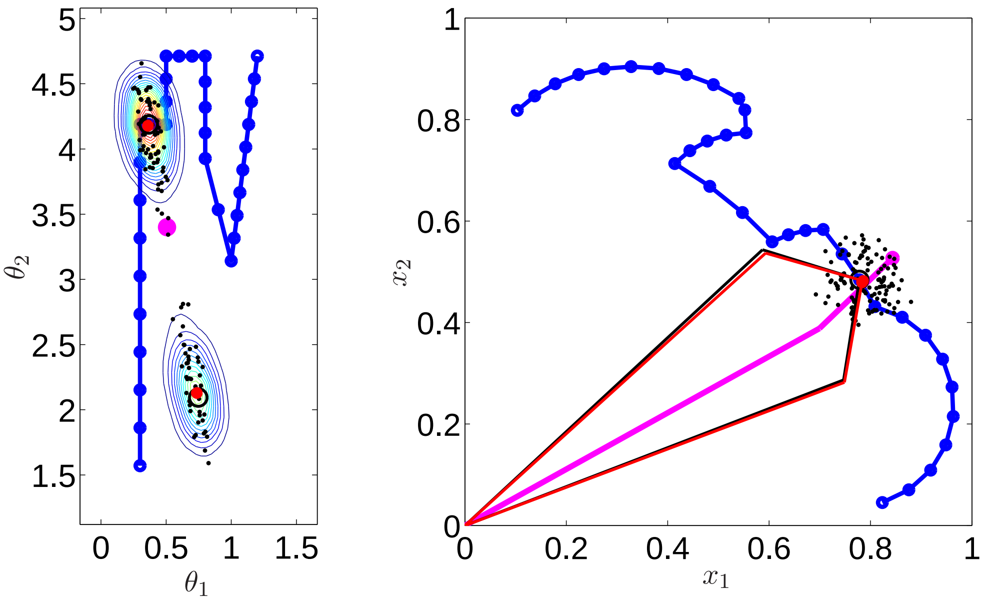


Figure C.16: Illustration for a planar 2-link robot arm of using conditional modes to represent multiple inverses. *Right*: a point $\mathbf{x} \in [0,1]^2$ in workspace can be reached by two joint angle configurations, elbow up/down. *Left*: the two configurations are captured as the two modes of a conditional density $p(\boldsymbol{\theta}|\mathbf{x})$ obtained from a PF (contours). Its mean (magenta dot) is not a valid inverse, mapping to a fully stretched arm. The plots show a trajectory in $\mathbf{x}$ and $\boldsymbol{\theta}$ space (blue) and the particles (black dots) using a SIR-PF.

visualize) how the modes of the conditional posterior on $\boldsymbol{\theta}$ represent the inverses of $\mathbf{f}$. Figs. C.17–C.18 show reconstructions for a figure–8 and elliptical trajectories.

As noted in section C.3.1, there are several $\boldsymbol{\theta}$–trajectories that produce the given $\mathbf{x}$–trajectory due to the symmetry of $\mathbf{f}$. Gaussian trackers (EKF/EKS) behaves similarly to pseudoinverse, i.e., being local, stick to one of these trajectories from the beginning (depending on its initialization) and can never recover the others later on. Likewise, at singularities of $\mathbf{f}$ (fully stretched arm), where multiple inverse branches merge or diverge, these methods (pseudoinverse, EKF/EKS) choose one of the branches and the rest are irreversibly lost.

The problem with this local choice is that it may turn out to be wrong later on in the trajectory, e.g. only an elbow-up configuration may be valid in certain workspace regions because of mechanical limits on some angles. In contrast, nongaussian trackers (SIR-PF, FBS, TFS) are capable of tracking all branches (thus all solutions) at every time; it is only at the end that a globally valid trajectory (avoiding discontinuities) is obtained by minimizing the constraint. Fig. C.17 also shows how if we simply use the mean of the PF distribution (instead of all its modes), a wrong, "average" trajectory is obtained. Table C.4 gives the errors in $\boldsymbol{\theta}$ and $\mathbf{x}$ w.r.t. ground truth for several trajectories. Generally, we find little

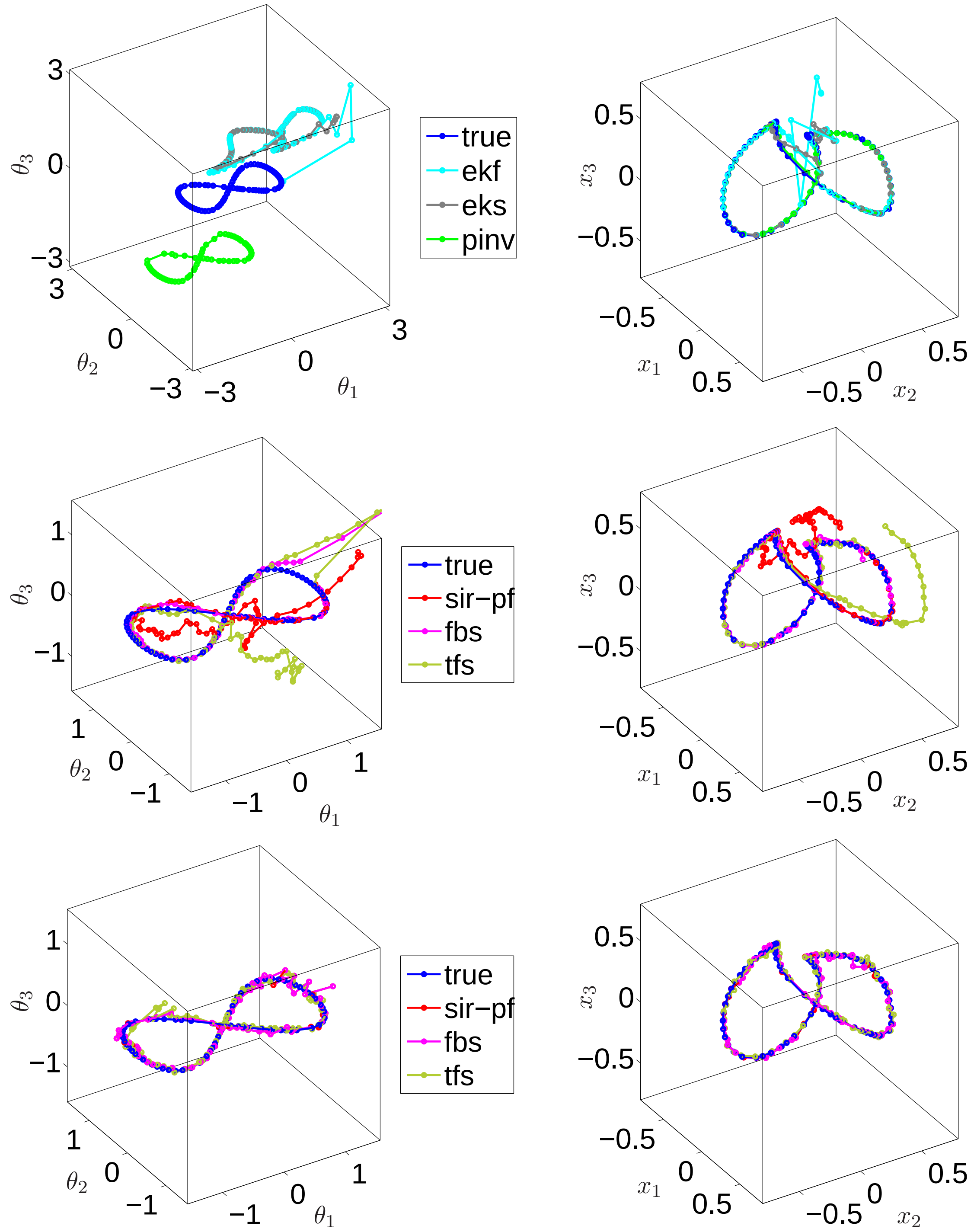


Figure C.17: Reconstruction of a figure-8 trajectory for the PUMA 560 robot arm. *Left*: $\boldsymbol{\theta}$-space, *right*: $\mathbf{x}$-space. *Top*: pseudoinverse, EKF, EKS. *Middle*: conditional means from SIR-PF, FBS, TFS. *Bottom*: conditional modes from SIR-PF, FBS, TFS and constraint $\mathcal{C} + \lambda\mathcal{F}$.

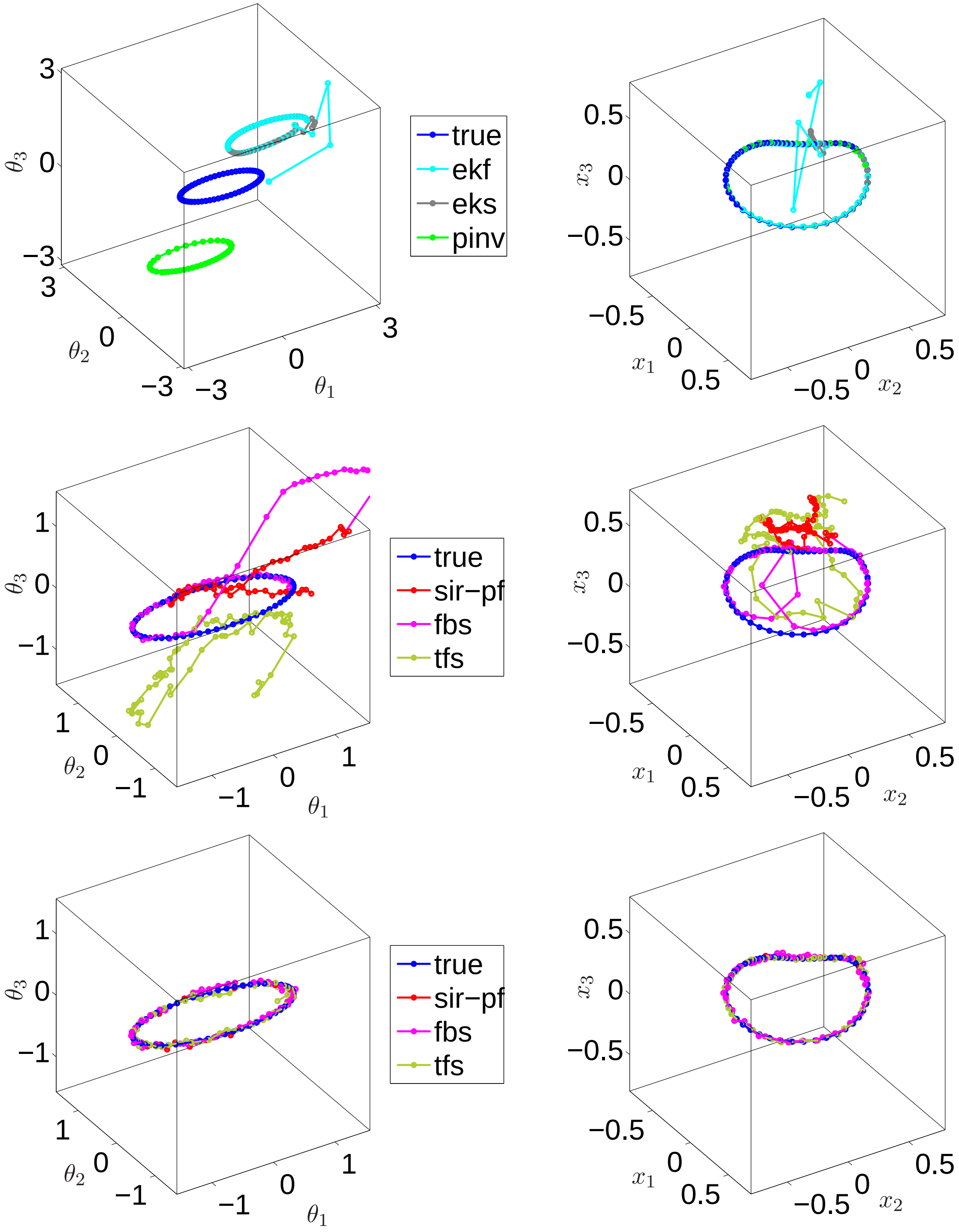


Figure C.18: As fig. C.17 but for an elliptical trajectory.

difference among the 3 multimodal trackers tested (SIR-PF, FBS, TFS), which all succeed in recovering the ground truth with good accuracy. We do occasionally find (results not shown) that, depending on the initialization, the SIR-PF may fail to track all the modes and thus miss possible inverse branches (this could be corrected using more particles). The smoothers (FBS, TFS) are more robust than the filter (SIR-PF) in that the backward filter helps to recover such modes. We also obtained very similar results when using the smoothness constraint; and using a continuity constraint only (i.e., $\lambda = 0$), which indicates that the modes from the conditional density are accurate representatives of the true inverses.

Angle reconstruction error $\frac{1}{T}\sum_{t=1}^{T}\|\boldsymbol{\theta}_t - \hat{\boldsymbol{\theta}}_t\|$ (rad)

Trajectory	pseudoinv	EKF	EKS	SIR-PF mean	FBS mean	TFS mean	SIR-PF $\mathcal{C}+\lambda\mathcal{F}$	FBS $\mathcal{C}+\lambda\mathcal{F}$	TFS $\mathcal{C}+\lambda\mathcal{F}$
Elliptical	0.072	0.274	0.149	1.954 ± 0.654	0.925 ± 0.655	1.304 ± 0.898	0.116 ± 0.025	0.116 ± 0.025	0.100 ± 0.028
Figure–8	0.076	0.324	0.207	1.791 ± 0.520	1.239 ± 0.969	1.505 ± 0.807	0.141 ± 0.014	0.141 ± 0.014	0.144 ± 0.010
Open	0.042	1.230	0.706	2.543 ± 0.925	1.157 ± 0.996	0.826 ± 0.160	0.150 ± 0.028	0.150 ± 0.000	0.150 ± 0.028

Workspace reconstruction error $\frac{1}{T}\sum_{t=1}^{T}\|\mathbf{x}_t - \mathbf{f}(\hat{\boldsymbol{\theta}}_t)\|$

Trajectory	pseudoinv	EKF	EKS	SIR-PF mean	FBS mean	TFS mean	SIR-PF $\mathcal{C}+\lambda\mathcal{F}$	FBS $\mathcal{C}+\lambda\mathcal{F}$	TFS $\mathcal{C}+\lambda\mathcal{F}$
Elliptical	0.025	0.084	0.045	0.523 ± 0.182	0.052 ± 0.037	0.320 ± 0.236	0.033 ± 0.005	0.033 ± 0.005	0.029 ± 0.003
Figure–8	0.019	0.083	0.059	0.409 ± 0.117	0.039 ± 0.022	0.252 ± 0.111	0.031 ± 0.002	0.031 ± 0.003	0.033 ± 0.005
Open	0.007	0.252	0.261	0.940 ± 0.405	0.119 ± 0.095	0.079 ± 0.037	0.042 ± 0.006	0.041 ± 0.006	0.035 ± 0.006

Table C.4: Reconstruction errors for PUMA 560 robot arm by nonlinear, nongaussian tracking ($\mathbf{x}_t$: given $\mathbf{x}$–trajectory, $\hat{\boldsymbol{\theta}}_t$: reconstructed $\boldsymbol{\theta}$–trajectory, $\boldsymbol{\theta}_t$: true $\boldsymbol{\theta}$–trajectory).

Finally, we compared deriving the conditional density from a particle filter with learning offline a conditional density $p(\boldsymbol{\theta}|\mathbf{x})$ (e.g. with a Gaussian mixture) given a training set of pairs $(\boldsymbol{\theta}_t, \mathbf{x}_t)$, as done in section C.3.1. The latter gave slightly lower reconstruction errors ($\boldsymbol{\theta}/\mathbf{x}$ spaces: elliptical, 0.071/0.029; figure–8, 0.081/0.027; open, 0.173/0.055). However, offline collection of such training data may be a problem, as it is difficult to sample a high-dimensional space.

Table C.5 lists average running times in seconds (per trajectory point) with different trackers in our Matlab implementation.

pseudoinv	EKF	EKS	SIR-PF	FBS	TFS
0.0096	0.0044	0.0046	0.069	0.49	0.44

Table C.5: Average run time per trajectory point by nonlinear, nongaussian tracking (sec.), $M = 1\,000$ particles on a Quad Core 2.0 Ghz machine with 2GB of RAM.

C.5 Practical considerations

In this section, we address several practical issues in traditional IK, i.e., fine positioning, real-time manipulation, handling kinematic singularity, and handling joint limits and obstacles.

C.5.1 Fine positioning

Fine positioning refers to accuracy of inverse solutions in $\mathbf{x}$-space. By construction, it can be achieved in nonlinear, nongaussian tracking by setting the coefficients of the covariance of measurement noises sufficiently small. Therefore, we focus on whether fine positioning can be achieved using conditional density modes. To measure it, we report the average pointwise reconstruction error, which is calculated as follows: (1) Sample uniformly in $\mathbf{x}$-space and removing grid points outside the feasible region. (2) Compute all true inverses and conditional density modes for each sample point in $\mathbf{x}-$space and prune the infeasible true inverses (e.g. prune any true solution located outside the feasible region). (3) Associate each true solution with a closest conditional density mode and compute the error between them in both $\boldsymbol{\theta}$ and $\mathbf{x}$ spaces. [4].

Angle pointwise error $\|\boldsymbol{\theta}_t - \hat{\boldsymbol{\theta}}_t\|$ (rad) averaged over number of true inverses

	mode			mixture component mean		
Model	avg	max	min	avg	max	min
coarse GTM	0.078	0.24	0	0.077	0.22	0
fine GTM	0.096	0.625	0	0.024	0.144	0
MDN	0.034	0.4	0	0.034	0.4	0

Workspace pointwise error $\|\mathbf{x}_t - \mathbf{f}(\hat{\boldsymbol{\theta}}_t)\|$ averaged over number of true inverses

	mode			mixture component mean		
Model	avg	max	min	avg	max	min
coarse GTM	0.021	0.104	0.0002	0.023	0.103	0.0002
fine GTM	0.018	0.090	0.0001	0.007	0.072	0.00001
MDN	0.0045	0.058	$6 \cdot 10^{-5}$	0.0045	0.058	$6 \cdot 10^{-5}$

Table C.6: Pointwise reconstruction errors for the 2D robot arm

Two-link robot arm. Table C.6 gives the pointwise errors (i.e., the average (avg), minimum (min), maximum (max)) in $\boldsymbol{\theta}$ and $\mathbf{x}$-space for the planar two-link robot arm. MDN achieves the lowest errors of 0.034 and 0.0045 on average in $\boldsymbol{\theta}$ and $\mathbf{x}$-space respectively. This result indicates that conditional density modes can satisfy fine positioning requirement. GTMs perform sightly worse but still give very

---

[4] Note that occasionally in PUMA560, we may perform phrase unwrap in data association because $\mathbf{f}$ is invariant to the periodic variables

appealing accuracy. In addition, we also report the reconstruction errors using mixture component means, i.e., $\{\mu_m(\mathbf{x})\}_{m=1}^{M}$, of our density models as inverse solutions. Their results are deceptively comparable to modes. However, the mixture component means are not practical and reliable solution for following reasons: (1) mixture components could significantly outnumber the modes; However, the majority of them represent spurious solutions, which could cause the dynamic programming (recall that its complexity is quadratic on number of candidates) very slow and possibly return an invalid solution trajectory. (2) modes are much better summarization of multimodal distribution than mixture component means, which behaves reliably only when they are well separately as in our cases. For MDN, reconstruction errors seem to be uniformly distributed, though slightly larger at the upper-left and lower-right corner of the workspace as shown in fig. C.19.

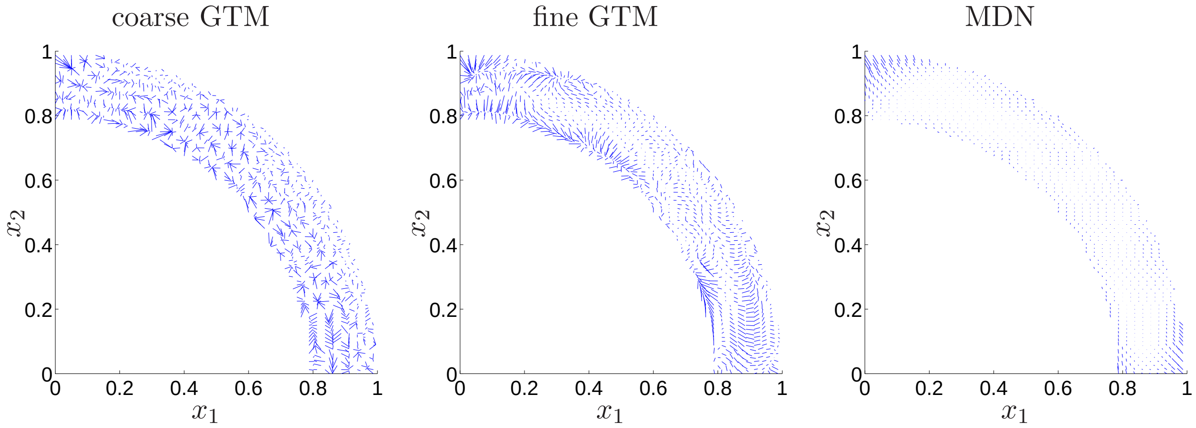


Figure C.19: Pointwise reconstruction errors of the test set in the workspace for the planar two-link robot arm. *Left*: Coarse GTM. *Middle*: fine GTM. *Right*: MDN. For each test point, a line is drawn between the true and predicted positions and its length is proportional to the predictive error.

Angle pointwise error $\|\boldsymbol{\theta}_t - \hat{\boldsymbol{\theta}}_t\|$ (rad) averaged over number of true inverses

	mode			mixture component mean		
Model	avg	max	min	avg	max	min
MDN	0.14	1.1	0.0068	0.14	1.1	0.0068

Workspace pointwise error $\|\mathbf{x}_t - \mathbf{f}(\hat{\boldsymbol{\theta}}_t)\|$ averaged over number of true inverses

	mode			mixture component mean		
Model	avg	max	min	avg	max	min
MDN	0.02	0.15	0.00042	0.02	0.15	0.00042

Table C.7: Pointwise reconstruction errors for PUMA560 on the regular grids

PUMA560. Table C.7 reports the similar pointwise reconstruction errors for PUMA 560 robot arm. MDN gives the errors of 0.014 and 0.02 on average in $\boldsymbol{\theta}$ and $\mathbf{x}$-space respectively. 0.141 in $\boldsymbol{\theta}$ appears to be high. This is because occasionally, one true local solution may be missed by modes and this contribute a lot to the error. As in fig. C.20, the big inverse errors all occur around the unreachable cylinder because of boundary effects of density model. This can be improved by obtaining a better conditional density model. Again, using mixture component means yields almost identical results to using modes since the modes coincide more or less to the means.

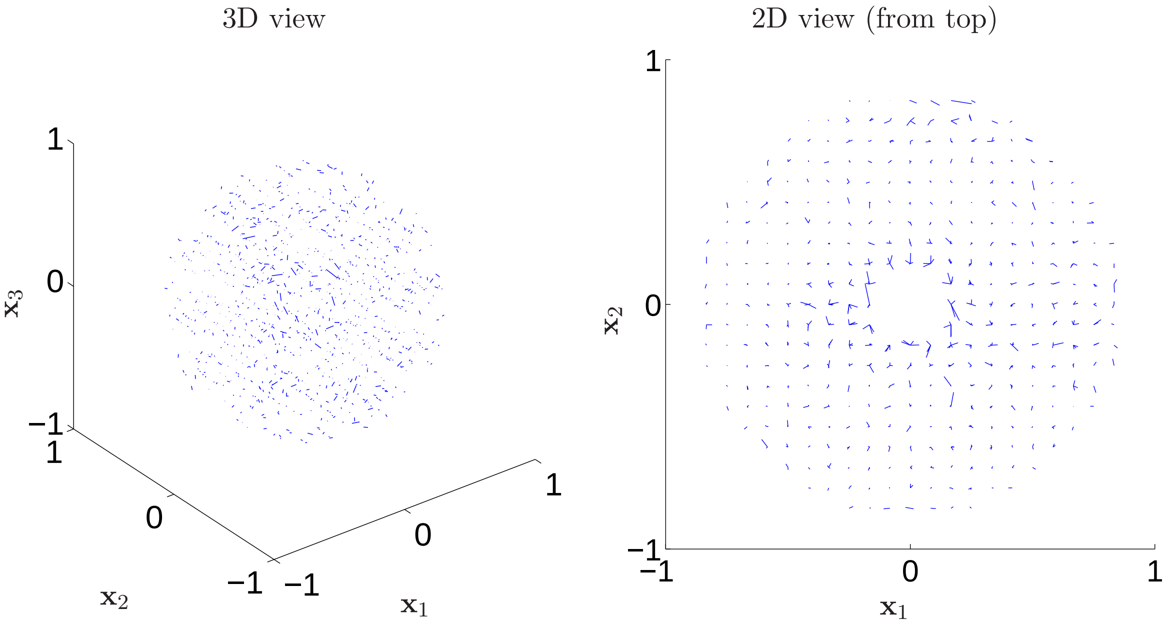


Figure C.20: Pointwise reconstruction errors of the test set in the workspace for the PUMA 560 robot arm. For each test point we have drawn a line between the true and predicted positions. The length of a line indicates the magnitude of the corresponding predictive error.

C.5.2 Real-time manipulation

Runtime is a critical issue for real manipulation. Pseudoinverse generally runs very fast, i.e., using less than 3ms to move the end-effector from one position to another in the workspace. It is therefore suitable for real-time application. However, pseudoinverse converges very slowly near the neighborhood of singularity. For conditional modes, most of runtime is spent on mode-finding, which can be significantly accelerated. In practice, we found the mode-finding only iterates once on average using MDN because it tends to directly jumps to the mode from the mean. For tracking, as show in table C.5, EKF takes less than 5 ms to move from one trajectory point to another while PF needs 10 ms because it needs sufficient particles to the conditional density. One can reduce runtime by using fewer particles at the expense of sacrificing accuracy. If too few particles are used, the risk of mistracking one local solution would increase. However, if one is interested in tracking just one such local trajectory, it is possible to use many fewer particles while maintaining tracking accuracy.

C.5.3 Handling kinematic singularity

Kinematic singularity occurs when the Jacobian at certain joint configuration becomes rank-deficient. At singularity, the manipulator behaves very differently, e.g. reduced mobility, increased number of inverse solutions. In the singular neighborhood, small velocity in the workspace may cause significant velocities in the joint space. In such cases, pseudoinverse becomes problematic since the Jacobian is no longer invertible. Unlike pseudoinverse, our conditional modes does not involve computation of the Jacobian but derive the modes from $p(\boldsymbol{\theta}|\mathbf{x})$, which deals with the change in the topology of the manifold of inverses naturally. For example, in two-link robot arm, near singularity (a.k.a, boundary singularity), e.g. the arm is fully-stretched, the "elbow-up" and "elbow-down" modes start to merge into a single one, i.e., the "full-stretched" mode while our conditional density changes from bimodal to unimodal. Likewise, in our nonlinear, nongaussian tracking framework, different modes derived from $p(\boldsymbol{\theta}_t|\mathbf{x}_{1:t})$ given by PFs merges into a single one when approaching the singular region and split afterwards.

C.5.4 Handling joint limits and obstacles

Joint limits can be handled conveniently in our framework. For example, in conditional modes, one can either prune conditional modes that disregard joint limits before dynamic programming, or limit the training set according to $\boldsymbol{\theta}$-domain so that certain branches are no longer valid (as in fig. C.21). In particle filtering, one can simply set the weights of particles that fall outside the joint limits to zero before the resampling step. By doing this, only those particles in the feasible regions will propagate over time. Obstacle avoidance in $\mathbf{x}$–space or $\boldsymbol{\theta}$-space can be handled similarly. In particle filtering, one can also set the weights of particles whose mappings to the workspace, e.g. links and effectors, are within the neighborhood of the obstacles to zero.

C.6 Summary

We have our conditional modes approach to trajectory IK that can deal with trajectories containing singularities, where the inverse mapping changes topology, and with complicated angle domains caused by mechanical constraints (e.g. to prevent self-intersection of body limbs in a humanoid robot)—a hard problem for local methods (e.g. Jacobian pseudoinverse). Given a training set $(\boldsymbol{\theta}, \mathbf{x})$, the method learns

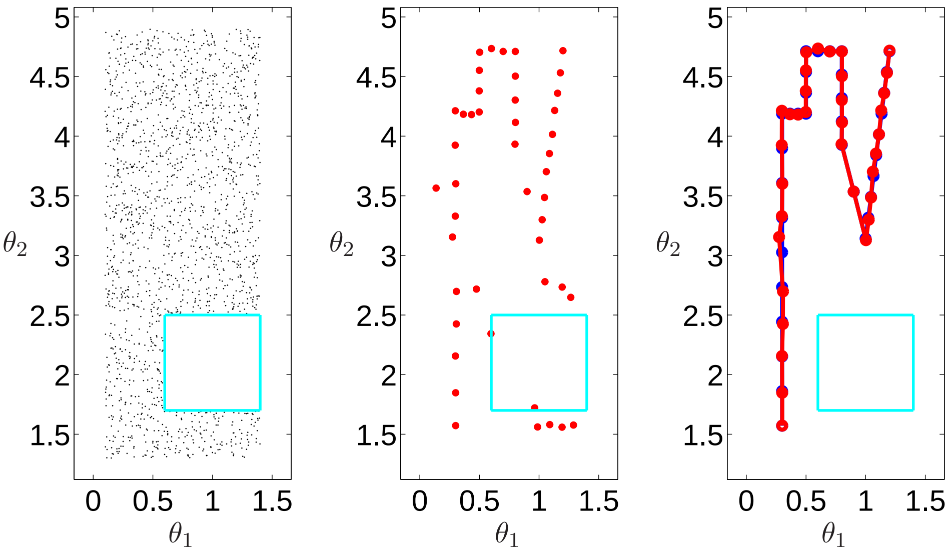


Figure C.21: Example of handling the joint limit by conditional density modes using the planar 2-link robot arm. *Left*: The dataset with a hole in contrast to fig. C.2, where the training set has no hole. *Middle*: Plots of all conditional modes. *Right*: Reconstruction by MDN. *Cyan box*: the forbidden region in the joint space. The conditional density is derived from a Gaussian mixture with 64 components and each with a full covariance. Inside the forbidden region, the conditional density is negligible and thus no condition modes.

a conditional density $p(\boldsymbol{\theta}|\mathbf{x})$ (using a mixture density network, MDN) that implicitly represents the branches of the inverse mapping $\boldsymbol{\theta} = \mathbf{f}^{-1}(\mathbf{x})$; the mappings are obtained by finding the modes of the conditional density using a Gaussian mean-shift algorithm, and the final $\boldsymbol{\theta}$-trajectory is obtained by minimizing a global, trajectory-wide constraint over the set of modes. We have demonstrated the method with trajectory IK for simple robot arms (e.g. PUMA 560) with known forward and inverse mappings. Additionally, we have formulated IK as a tracking problem and proposed the first tracking framework for it. We demonstrate that the conditional density estimated online by nonlinear, nongaussian trackers, e.g. particle filter or smoother, can be integrated seamlessly into the conditional modes framework. Alternatively, we show that various Gaussian or nongaussian trackers can be used independently to produce competitive results. We also confirm by experiments that our method is able to deal with other practical issues in IK: fine positioning, real-time manipulation, joint limits and obstacle avoidance. One limitation of our methods is the scalability to robot arms with high DOF because estimating $p(\boldsymbol{\theta}|\mathbf{x})$ in high-dim space would become very challenging and still an open research area.

Future work could apply this approach to trajectory IK in other domains (such as animation in computer graphics, articulated pose tracking in computer vision or articulatory inversion in speech), where neither the inverse nor possibly the forward mappings are known, and having complex mechanical constraints that are best captured by data-driven approaches. Another advantage of the method is its probabilistic nature: it can model noise in the measured $\boldsymbol{\theta}$, $\mathbf{x}$ and estimate the uncertainty in the reconstructed trajectory (error bars); it is also applicable when some of the $\mathbf{x}$ variables are missing or unspecified (e.g. for a humanoid robot we might not care about the hand position when walking). For example, one can use the techniques, developed in chapter 7, to reconstruct the missing data.

Bibliography

[1] Z. Al Bawab, B. Raj, and R. M. Stern. Analysis-by-synthesis features for speech recognition. In *Proc. ICASSP*, pages 4185–4188, 2008.

[2] Z. Al Bawab, L. Turicchia, R. M. Stern, and B. Raj. Deriving vocal tract shapes from electromagnetic articulograph data via geometric adaptation and matching. In *Proc. Interspeech*, pages 2051–2054, 2009.

[3] G. Ananthakrishnan, D. Neiberg, and O. Engwall. In search of non-uniqueness in the acoustic-to-articulatory mapping. In *Proc. Interspeech*, pages 2799–2802, 2009.

[4] B. D. Anderson and J. B. Moore. *Optimal filtering*. Dover Publications, INC, Mineola, New York, 1979.

[5] M. Aron, N. Ferveur, E. Kerrien, M. O. Berger, and Y. Laprie. Acquisition and synchronization of multimodal articulatory data. In *Proc. Interspeech*, 2007.

[6] M. Aron, E. Kerrien, M. O. Berger, and Y. Laprie. Coupling electromagnetic sensors and ultrasound images for tongue tracking: Acquisition set up and preliminary results. In *Proceedings of the 7th Speech Production Workshop: Models and Data*, pages 435–442, 2006.

[7] M. Aron, A. Roussos, M. O. Berger, E. Kerrien, and P. Maragos. Multimodality acquisition of articulatory data and processing. In *Proc. EUSIPCO*, 2008.

[8] International Phonetic Association. *Handbook of the International Phonetic Association: a guide to the use of the international phonetic alphabet*. Cambridge University Press, 1999.

[9] B. S. Atal, J. J. Chang, M. V. Mathews, and J. W. Tukey. Inversion of articulatory-to-acoustic transformation in the vocal tract by a computer-sorting technique. *J. Acoustic Soc. Amer.*, 63(5):1535–1555, 1978.

[10] P. Badin, G. Bailly, L. Reveret, M. Baciu, C. Segebarth, and C. Savariaux. Three-dimensional linear articulatory modeling of tongue, lips and face, based on MRI and video images. *J. Phonetics*, 30(3):533–553, 2002.

[11] P. Badin, E. Baricchi, and A. Vilain. Determining tongue articulation: from discrete fleshpoints to continuous shadow. In *Proc. Eurospeech*, pages 47–50, 1997.

[12] P. Badin and A. Serrurier. Three-dimensional linear modeling of tongue: Articulatory data and models. In *Proceedings of the 7th Speech Production Workshop: Models and Data*, pages 395–402, 2006.

[13] O. Balter, O. Engwall, A.M. Oster, and H. Kjellstrom. Wizard-of-oz test of ARTUR: a computer-based speech training system with articulation correction. In *7th International ACM SIGACCESS Conference on Computers and Accessibility*, pages 36–43, 2005.

[14] J. Benesty. *Springer Handbook Of Speech Processing*. Springer Verlag, 2008.

[15] P. Birkholz and B. J. Kroger. Vocal tract model adaptation using magnetic resonance imaging. In *Proceedings of the 7th Speech Production Workshop: Models and Data*, pages 493–500, 2006.

[16] C. M. Bishop. *Neural Networks for Pattern Recognition*. Oxford University Press, New York, Oxford, 1995.

[17] C. M. Bishop. *Pattern recognition and machine learning*. Springer New York, 2006.

[18] C. M. Bishop, M. Svensén, and C. K. I. Williams. GTM: The generative topographic mapping. *Neural Computation*, 10(1):215–234, 1998.

[19] C. S. Blackburn and Steve Young. A self-learning predictive model of articulator movements during speech production. *J. Acoustic Soc. Amer.*, 107(3):1659–1670, March 2000.

[20] L.-J. Boë, L. Menard, and S. Maeda. Adaptation of control strategies during the vocal tract growth inferred from simulation studies with an articulatory model. In *Proc. 5th Seminar on Speech Production*, pages 277–280, 2000.

[21] E. Bresch, Y. C. Kim, K. Nayak, D. Byrd, and S. Narayanan. Seeing speech: Capturing vocal tract shaping using real-time magnetic resonance imaging. *IEEE Signal Processing Magazine*, 2008.

[22] M. Á. Carreira-Perpiñán. Reconstruction of sequential data with probablistic models and continuity constraints. In *NIPS*, pages 414–420, 1999.

[23] M. Á. Carreira-Perpiñán. Mode-finding for mixtures of gaussian distributions. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 22(11):1318–1323, 2000.

[24] M. Á. Carreira-Perpiñán. *Continuous latent variable models for dimensionality reduction and sequential data reconstruction*. PhD thesis, University of Sheffield, UK, 2001.

[25] M. Á. Carreira-Perpiñán. Acceleration strategies for gaussian mean-shift image segmentation. In *Proc. CVPR*, pages 1160–1167, 2006.

[26] M. Á. Carreira-Perpiñán and Z. Lu. The laplacian eigenmaps latent variable model. In *Proc. AISTATS*, pages 59–66, 2007.

[27] S. Chennoukh, D. Sinder, G. Richard, and J. L. Flanagan. Improved techniques for voice mimic system using an articulatory codebooks. In *Proc. Eurospeech*, pages 429–432, 1997.

[28] M. M. Cohen, J. Beskow, and D. W. Massaro. Recent developments in facial animation: An inside view. In *Proc. of the Third Auditory-Visual Speech Processing Conference*, pages 201–206, 1998.

[29] P. I. Corke. A robotics toolbox for MATLAB. *IEEE Robotics and Automation Magazine*, 3(1):24–32, 1996.

[30] J. J. Craig. *Introduction to Robotics. Mechanics and Control.* Addison-Wesley, 1989.

[31] D. DeMers and K. Kreutz-Delgado. Canonical parameterization of excess motor degrees of freedom with self-organizing maps. *IEEE Transactions on Neural Networks*, 7(1):43–55, 1996.

[32] D. DeMers and K. Kreutz-Delgado. Learning global properties of nonredundant kinematic mappings. *Int. J. of Robotics Research*, 17(5):547–560, 1998.

[33] V. V. Digalakis, D. Rtischev, and L. G. Neumeyer. Speaker adaptation using constrained estimation of Gaussian mixtures. *IEEE Trans. Speech and Audio Process.*, 3(5):357–366, 1995.

[34] A. Doucet, N. de Freitas, and N. Gordon. *An introduction to sequential Monte Carlo Methods*, chapter Chapter 1, pages 3–14. Springer-Verlag, 2001.

[35] A. D'Souza, S. Vijayakumar, and S. Schaal. Learning inverse kinematics. In *Proc. IROS*, pages 298–303, 2001.

[36] S. Dusan and L. Deng. Estimation of articulatory parameters from speech acoustics by Kalman filtering. In *Proc. CITO Researcher Retreat-Hamilton Canada*, 1998.

[37] S. Dusan and L. Deng. Recovering vocal tract shapes from MFCC parameters. In *Proc. ICSLP*, pages 3087–3090, 1998.

[38] S. Dusan and L. Deng. Acoustic-to-articulatory inversion using dynamical and phonological constraints. In *Proceedings of the 5th Speech Production Workshop: Models and Data*, pages 237–240, 2000.

[39] O. Engwall. A 3D tongue model based on MRI data. In *Proc. ICSLP*, pages 901–904, 2000.

[40] O. Engwall. From real-time MRI to 3D tongue movements. In *Proc. ICSLP*, pages 1109–1112, 2004.

[41] O. Engwall and P. Wik. Are real tongue movements easier to speech read than synthesized? In *Proc. Interspeech*, 2009.

[42] O. Engwall and P. Wik. Can you tell if tongue movements are real or synthesized? In *Proc. AVSP*, 2009.

[43] C. Y. Espy-Wilson, S. E. Boyce, M. Jackson, S. Narayanan, and A. Alwan. Acoustic modeling of american english /ɹ/. *J. Acoustic Soc. Amer.*, 108(1):343–356, 2000.

[44] G. Fant. *Acoustic Theory of Speech Production.* The Hague: Mouton, second edition, 1960.

[45] J. L. Flanagan, K. Ishizaka, and K. L. Shipley. Signal models for low bit-rate coding of speech. *J. Acoustic Soc. Amer.*, 68(3):780–791, 1980.

[46] Joe Frankel and S. King. ASR – articulatory speech recognition. In *Proc. Eurospeech*, pages 599–602, 2001.

[47] S. Furui. Speaker-independent isolated word recognition using dynamic features of speech spectrum. *j-ieee-assp*, 34(1):52–59, 1986.

[48] M. J. F. Gales. Maximum likelihood linear transformations for HMM–based speech recognition. *Computer Speech and Language*, 12(2):75–98, 1998.

[49] M. J. F. Gales. Cluster adaptive training of hidden Markov models. *IEEE Trans. Speech and Audio Process.*, 8(4):417–428, 2000.

[50] M. J. F. Gales and P. C. Woodland. Mean and variance adaptation within the MLLR framework. *Computer Speech and Language*, 10(4):249–264, 1996.

[51] J.-L. Gauvain and C.-H. Lee. Maximum a-posteriori estimation for multivariate Gaussian mixture observations of Markov chains. *IEEE Trans. Speech and Audio Process.*, 2:1291–1298, 1994.

[52] Z. Ghahramani and M. I. Jordan. Supervised learning from incomplete data via an em approach. In *NIPS*, pages 120–127, 1994.

[53] P. K. Ghosh and S. Narayanan. A generalized smoothness criterion for acoustic-to-articulatory inversion. *J. Acoustic Soc. Amer.*, 128:2162, 2010.

[54] U. Goldstein. *An articulatory model for the vocal tracts of growing children.* PhD thesis, MIT, Cambridge, MA, 1980.

[55] A. Gray and J. D. Markel. Distance measures for speech processing. *IEEE Trans. Acoust., Speech, and Signal Process.*, 24(5):380–391, 1976.

[56] K. Grochow, S. L. Martin, A. Hertzmann, and Z. Popović. Style-based inverse kinematics. *ACM Transactions on Graphics (TOG)*, 23(3):522–531, 2004.

[57] R. Harshman, P. Ladefoged, and L. Goldstein. Factor analysis of tongue shapes. *J. Acoustic Soc. Amer.*, 62(3):693–707, 1997.

[58] H. Hermansky. Perceptual linear predictive (PLP) analysis of speech. *J. Acoustic Soc. Amer.*, 87(4):1738–1752, 1990.

[59] H. Hermansky and N. Morgan. RASTA processing of speech. *IEEE Trans. Acoust., Speech, and Signal Process.*, 2(4):578–589, 1994.

[60] S. Hiroya and M. Honda. Estimation of articulatory movements from speech acoustics using an HMM-based speech production model. *IEEE Trans. Speech and Audio Process.*, 12(2):175–185, 2004.

[61] J. Hogden, A. Löfqvist, V. Gracco, I. Zlokarnik, P. E. Rubin, and E. Saltzman. Accurate recovery of articulator positions from acoustics: New conclusions based on human data. *J. Acoustic Soc. Amer.*, 100(3):1819–1834, 1996.

[62] M. Isard and A. Blake. A smoothing filter for condensation. In *Proc. ECCV*, pages 767–781, 1998.

[63] K. Iskarous. Detecting the edge of the tongue. *J. Clinical Linguistics and Phonetics*, 19(6-7):555–565, 2005.

[64] F. Itakura. Minimum prediction residual principle applied to speech recognition. *IEEE Trans. Acoust., Speech, and Signal Process.*, 23(1):62–72, 1975.

[65] F. Itakura and S. Saito. A statistical method for estimation of speech spectral density and formant frequencies. *Electronics & Communications in Japan*, 53-A(1):36–43, 1970.

[66] R. A. Jacobs, M. I. Jordan, S. J. Nowlan, and G. E. Hinton. Adaptive mixtures of local experts. *Neural Computation*, 3(1):79–87, 1991.

[67] M. I. Jordan and D. E. Rumelhart. Forward models: Supervised learning with a distal teacher. *Cognitive Science*, 16:307–354, 1992.

[68] B.-H. Juang and L. R. Rabiner. Automatic speech recognition–a brief history of the technology development. *Encyclopedia of Language and Linguistics, Elsevier*, 2005.

[69] T. Kaburagi and M. Honda. Determination of sagittal tongue shape from the positions of points on the tongue surface. *J. Acoustic Soc. Amer.*, 96(3):1356–1366, 1994.

[70] M. Kass, A. Witkin, and D. Terzopoulos. Snakes: Active contour models. *Int. J. Computer Vision*, 1(4):321–331, 1988.

[71] Christopher T. Kello and D. C. Plaut. A neural network model of the articulatory-acoustic forward mapping trained on recordings of articulatory parameters. *J. Acoustic Soc. Amer.*, 116(4):2354–2364, October 2004.

[72] S. King, J. Frankel, K. Livescu, E. McDermott, K. Richmond, and M. Wester. Speech production knowledge in automatic speech recognition. *J. Acoustic Soc. Amer.*, 121(2):723–742, 2007.

[73] S. King and A. Wrench. Dynamical system modelling of articulator movement. In *Int. Conf. Phonetic Science*, pages 2259–2262, 1999.

[74] G. Kitagawa. Monte Carlo filter and smoother for non-Gaussian nonlinear state space models. *Journal Of Computational And Graphical Statistics*, 5(1):1–25, 1996.

[75] M. Klaas, M. Briers, N. de Freitas, A. Doucet, S. Maskell, and D. Lang. Fast particle smoothing: If I had a million particles. In *Proc. ICML*, pages 481–488, 2006.

[76] R. Kolodny, L. Guibas, M. Levitt, and P. Koehl. Inverse kinematics in biology: The protein loop closure problem. *Int. J. of Robotics Research*, 24(2-3):151–163, 2005.

[77] R. Kuhn, J. C. Junqua, P. Nguyen, and N. Niedzielski. Rapid speaker adaptation in eigenvoice space. *IEEE Trans. Acoust., Speech, and Signal Process.*, 8(6):695–707, 2000.

[78] P. Ladefoged. *A Course in Phonetics, Third Edition.* Fort Worth: Harcourt Brace College Publishers, 1993.

[79] P. Ladefoged, R. Harshman, L. Goldstein, and L. Rice. Generating vocal tract shapes from formant frequencies. *J. Acoustic Soc. Amer.*, 1978.

[80] Y. Laprie and B. Mathieu. A variational approach for estimating vocal tract shapes from speech signal. In *Proc. ICASSP*, pages 929–932, 1998.

[81] J. N. Larar, J. Schroeter, and M. M. Sondhi. Vector quantisation of the articulatory space. *IEEE Trans. Acoust., Speech, and Signal Process.*, 36(12):1812–1818, 1988.

[82] L. Lee and R. Rose. A frequency warping approach to speaker normalization. *IEEE Trans. Speech and Audio Process.*, 6(1):49–60, 1998.

[83] C. J. Leggetter and P. C. Woodland. Maximum likelihood linear regression for speaker adaptation of continuous density hidden Markov models. *Computer Speech and Language*, 9(2):171–185, 1995.

[84] S. E. Levinson and C. E. Schmidt. Adaptive computation of articulatory parameters from the speech signal. *J. Acoustic Soc. Amer.*, 74(4):1145–1154, 1983.

[85] M. Li, C. Kambhamettu, and M. Stone. Automatic contour tracking in ultrasound images. *J. Clinical Linguistics and Phonetics*, 19(6-7):545–554, 2005.

[86] A. Liegeois. Automatic supervisory control of configuration and behavior of multibody mechanisms. *IEEE Trans. Systems, Man, and Cybernetics*, 7(12):868–871, 1977.

[87] M. Lindau-Webb and P. Ladefoged. Methodological studies using an x-ray microbeam system. *UCLA Working Papers in Phonetics*, 72:82–90, 1989.

[88] B. Lindblom, J. Lubker, and T. Gay. Formant frequencies of some fixed-mandible vowels and a model of speech motor programming by predictive simulation. *J. of Phonetics.*, 7(2):147–161, 1979.

[89] Z. H. Ling, K. Richmond, J. Yamagishi, and R. H. Wang. Integrating articulatory features into HMM-based parametric speech synthesis. *IEEE Transactions on Audio, Speech, and Language Processing*, 17(6):1171–1185, 2009.

[90] A. J. Lundberg and M. Stone. Three-dimensional tongue surface reconstruction: Practical considerations for ultrasound data. *J. Acoustic Soc. Amer.*, 106(5):2858, 1999.

[91] S. Maeda. Compensatory articulation during speech: Evidence from the analysis and synthesis of vocal tract shapes using an articulatory model. *Speech Production And Speech Modelling*, pages 131–149, 1990.

[92] S. Maeda, M.-O. Berger, O. Engwall, Y. Laprie, P. Maragos, B. Potard, and J. Schoentgen. Acoustic-to-articulatory inversion: Methods and acquisition of articulatory data. Technical report, LORIA, 2006.

[93] D. P. Martin, J. Baillieul, and J. M. Hollerbach. Resolution of kinematic redundancy using optimization techniques. *IEEE Transactions on Robotics and Automation*, 5(4):529–533, 1989.

[94] B. Mathieu and Y. Laprie. Adaptation of Maeda's model for acoustic to articulatory inversion. In *Proc. Eurospeech*, pages 2015–2018, 1997.

[95] P. Mermelstein. Articulatory model for the study of speech production. *J. Acoustic Soc. Amer.*, 53(4):1070–1082, 1973.

[96] Y. Minami, E. McDermott, A. Nakamura, and S. Katagiri. A theoretical analysis of speech recognition based on feature trajectory models. In *Proc. ICSLP*, pages 549–552, 2004.

[97] Y. Nakamura and H. Hanafusa. Optimal redundancy control of robot manipulators. *Int. J. of Robotics Research*, 6(1):32–42, 1987.

[98] D. Neiberg, G. Ananthakrishnan, and O. Engwall. The acoustic to articulation mapping: Non-linear or non-unique? In *Proc. Interspeech*, pages 1485–1488, 2008.

[99] J. Neto, L. Almeida, M. Hochberg, C. Martins, L. Nunes, S. Renals, and T. Robinson. Speaker-adaptation for hybrid HMM-ANN continuous speech recognition system. In *Proc. Eurospeech*, pages 2171–2174, 1995.

[100] J. A. Noble and D. Boukerroui. Ultrasound image segmentation: A survey. *IEEE Trans. Medical Imaging*, 25(8):987–1010, 2006.

[101] J. Nocedal and S. J. Wright. *Numerical Optimization.* Springer, New York, 2006.

[102] S. Ouni and Y. Laprie. Modeling the articulatory space using a hypercube codebook for acoustic-to-articulatory inversion. *J. Acoustic Soc. Amer.*, 118(1):444–460, July 2005.

[103] I. Y. Ozbek, M. Hasegawa-Johnson, and M. Demirekler. Formant trajectories for acoustic-to-articulatory inversion. In *Proc. Interspeech*, pages 4267–4270, 2009.

[104] G. Papcun, J. Hochberg, T. R. Thomas, F. Laroche, J. Zacks, and S. Levy. Inferring articulation and recognizing gestures from acoustics with a neural network trained on x-ray microbeam data. *J. Acoustic Soc. Amer.*, 92(2):688–700, 1992.

[105] V. Parthasarathy, M. Stone, and J. L. Prince. Spatiotemporal visualization of the tongue surface using ultrasound and kriging SURFACES. *J. Clinical Linguistics and Phonetics*, 19(6-7):529–544, 2005.

[106] R. P. Paul and H. Zhang. Computationally efficient kinematics for manipulators with spherical wrists based on the homogeneous transformation representation. *Int. J. of Robotics Research*, 5(2):32, 1986.

[107] C. Qin and M. Á. Carreira-Perpiñán. A comparison of acoustic features for articulatory inversion. In *Proc. Interspeech*, pages 2469–2472, 2007.

[108] C. Qin and M. Á. Carreira-Perpiñán. An empirical investigation of the nonuniqueness in the acoustic-to-articulatory mapping. In *Proc. Interspeech*, pages 74–77, 2007.

[109] C. Qin and M. Á. Carreira-Perpiñán. Trajectory inverse kinematics by conditional density modes. In *Proc. ICRA*, pages 1979–1986, 2008.

[110] C. Qin and M. Á. Carreira-Perpiñán. Trajectory inverse kinematics by nonlinear, nongaussian tracking. In *Proc. ICASSP*, pages 2057–2060, 2008.

[111] C. Qin and M. Á. Carreira-Perpinán. Adaptation of a predictive model of tongue shapes. In *Proc. Interspeech*, pages 772–775, 2009.

[112] C. Qin and M. Á. Carreira-Perpiñán. The geometry of the articulator region that produces a speech sound. In *43th Annual Asilomar Conference on Signals, Systems, and Computers (ASILOMARSSC 2009)*, pages 1742–1746, 2009.

[113] C. Qin and M. Á. Carreira-Perpiñán. Articulatory inversion of american english /ɹ/ by conditional density modes. In *Proc. Interspeech*, pages –, 2010.

[114] C. Qin and M. Á. Carreira-Perpiñán. Estimating missing data sequences in x-ray microbeam recordings. In *Proc. Interspeech*, pages –, 2010.

[115] C. Qin and M. Á. Carreira-Perpinán. Reconstructing the full tongue contour from EMA/X-Ray microbeam. In *Proc. ICASSP*, pages 4190–4193, 2010.

[116] C. Qin, M. Á. Carreira-Perpinán, and Mohsen Farhadloo. Adaptation of a tongue shape model by local feature transformations. In *Proc. Interspeech*, 2010.

[117] C. Qin, M. Á. Carreira-Perpinán, K. Richmond, A. Wrench, and S. Renals. Predicting tongue shapes from a few landmark locations. In *Proc. Interspeech*, pages 2306–2309, 2008.

[118] L. Rabiner and B.-H Juang. *Fundamentals of Speech Recognition*. Prentice Hall, Englewood Cliffs, New Jersey, 1993.

[119] L. R. Rabiner and R. W. Shafer. *Digital Processing of Speech Signals*. Prentice-Hall, 1978.

[120] M. G. Rahim, C. C. Goodyear, W. B. Kleijn, J. Schroeter, and M. M. Sondhi. On the use of neural networks in articulatory speech synthesis. *J. Acoustic Soc. Amer.*, 93(2):1109–1121, 1993.

[121] G. Richard, M. Goirand, D. Sinder, and J. L. Flanagan. Simulation and visualization of articulatory trajectories estimated from speech signals. In *Proc. Int. Symposium on Simulation, Visualization and Auralization for Acoustic Research and Education (ASVA'97)*, 1997.

[122] K. Richmond. *Estimating Articulatory Parameters from the Acoustic Speech Signal*. PhD thesis, University of Edinburgh, 2001.

[123] K. Richmond. Mixture density networks, human articulatory data and acoustic-to-articulatory inversion of continuous speech. In *Proc. Workshop on Innovation in Speech Processing*, 2001.

[124] K. Richmond. Trajectory mixture density network for the acoustic-articulatory inversion mapping. In *Proc. ICSLP*, pages 577–580, 2006.

[125] K. Richmond, S. King, and P. Taylor. Modelling the uncertainty in recovering articulation from acoustics. *Computer Speech and Language*, 17(2–3):153–172, 2003.

[126] R. Ridouane. Investigating speech production: A review of some techniques, 2006.

[127] S. Roweis. *Data Driven Production Models for Speech Processing*. PhD thesis, California Institute of Technology, 1999.

[128] S. Roweis. Constrained hidden Markov models. In *NIPS*, pages 782–788, 2000.

[129] P. Rubin, T. Baer, and P. Mermelstein. An articulatory synthesizer for perceptual research. *J. Acoustic Soc. Amer.*, 70(2):321–328, 1981.

[130] J. Schroeter and M. M. Sondhi. Dynamic programming search of articulatory codebooks. In *Proc. ICASSP*, pages 588–591, 1989.

[131] J. Schroeter and M. M. Sondhi. Speech coding based on physiological models of speech production. In S. Furui and M. M. Sondhi, editors, *Advances in Speech Signal Processing*, pages 231–267. New York: Dekker, 1992.

[132] J. Schroeter and M. M. Sondhi. Techniques for estimating vocal tract shapes from the speech signal. *IEEE Trans. Acoust., Speech, and Signal Process.*, 2(1):133–150, 1994.

[133] J. M. Scobbie, A. A. Wrench, and M. van der Linden. Head-probe stabilisation in ultrasound tongue imaging using a headset to permit natural head movement. In *Proceedings of the 8th Speech Production Workshop: Models and Data*, pages 373–376, 2008.

[134] K. Shirai and M. Honda. Estimation of articulatory motion. In *Dynamic Aspects of Speech Production*, pages 279–302. Tokyo University Press, 1976.

[135] K. Shirai and T. Kobayashi. Estimation of articulatory motion using neural networks. *J. of Phonetics.*, 19(3–4):379–385, 1991.

[136] B. Siciliano, L. Sciavicco, L. Villani, and G. Oriolo. *Robotics: modelling, planning and control.* Springer Verlag, 2008.

[137] B.W. Silverman. *Density Estimation For Statistics And Data Analysis.* London: Chapman and Hall, 1986.

[138] Y. Sinan, A. Chandra, C. Kambhamettu, and M. Stone. Automatic extraction and tracking of the tongue contours. *IEEE Trans. Medical Imaging*, 18(10):1035–1045, 1999.

[139] F. Soong and B.-H Juang. Line spectrum pair (LSP) and speech data compression. In *Proc. ICASSP*, pages 37–40, March 1984.

[140] F. K. Soong and A. E. Rosenberg. On the use of instantaneous and transitional spectral information in speaker recognition. *IEEE Trans. Acoust., Speech, and Signal Process.*, 36(6):871–879, 1988.

[141] M. Stone. A three-dimensional model of tongue movement based on ultrasound and x-ray microbeam data. *J. Acoustic Soc. Amer.*, 87:2207–2217, 1990.

[142] M. Stone. A guide to analysing tongue motion from ultrasound images. *J. Clinical Linguistics and Phonetics*, 19(6-7):455–501, 2005.

[143] M. Stone. *Laboratory Techniques for Investigating Speech Articulation*, pages 9–38. Wiley-Blackwell, 2010.

[144] M. Stone, M.H. Goldstein, and Yongqing Zhang. Principal component analysis of cross sections of tongue shapes in vowel production. *Speech Communication*, 22(2-3):173–184, 1997.

[145] M. Stone and A. Lundberg. Three-dimensional tongue surface shapes of English consonants and vowels. *J. Acoustic Soc. Amer.*, 99(6):3728–3737, 1996.

[146] M. Stone, T.H. Shawker, T.L. Talbot, and A.H. Rich. Cross-sectional tongue shape during the production of vowels. *J. Acoustic Soc. Amer.*, 83(4):1586–1596, 1988.

[147] Brad H. Story. A parametric model of the vocal tract area function for vowel and consonant simulation. *J. Acoustic Soc. Amer.*, 117(5):3231–3254, May 2005.

[148] L. Tang and G. Hamarneh. Graph-based tracking of the tongue contour in ultrasound sequences with adaptive temporal regularization. In *IEEE Computer Society Workshop on Mathematical Methods in Biomedical Image Analysis (MMBIA10)*, 2010.

[149] T. Toda, A. W. Black, and K. Tokuda. Statistical mapping between articulatory movements and acoustic spectrum using a gaussian mixture model. *Speech Communication.*, 50(3):215–227, 2008.

[150] K. Tokuda, T. Yoshimura, T. Kobayashi, and T. Kitamura. Speech parameter generation algorithms for HMM-based speech synthesis. In *Proc. ICASSP*, pages 660–663, 1995.

[151] A. Toutios, S. Ouni, and Y. Laprie. Protocol for a model-based evaluation of a dynamic acoustic-to-articulatory inversion method using electromagnetic articulography. In *Proceedings of the 8th Speech Production Workshop: Models and Data*, pages 317–320, 2008.

[152] R. Urtasun, D. J. Fleet, A. Hertzmann, and P. Fua. Priors for people tracking from small training sets. In *Proc. ICCV*, pages 403–410, 2005.

[153] J. R. Westbury. The significance and measurement of head position during speech production experiments using the X-Ray microbeam system. *J. Acoustic Soc. Amer.*, 89(4):1782–1791, 1991.

[154] J. R. Westbury. *X-Ray Microbeam Speech Production Database User's Handbook Version 1.0*, 1994. With the assistance of Greg Turner and Jim Dembowski.

[155] J. R. Westbury, M. Hashi, and M. J. Lindstrom. Differences among speakers in lingual articulation for American English /ɹ/. *Speech Communication.*, 26(3):203–226, 1998.

[156] D. H. Whalen, A. M. Kang, H. S. Magen, R. K. Fulbright, and J. C. Gore. Predicting midsagittal pharynx shape from tongue position during vowel production. *Journal of Speech, Language and Hearing Research*, 42(3):592–603, 1999.

[157] D. E. Whitney. Resolved motion rate control of manipulators and human prostheses. *IEEE Transactions on Man-Machine Systems*, 10(2):47–53, 1969.

[158] P. Wik and O. Engwall. Can visualization of internal articulators support speech perception? In *Proc. Interspeech*, pages 2627–2630, 2008.

[159] P. Wik and O. Engwall. Looking at tongues–can it help in speech perception? In *Proc. FONETIK*, 2008.

[160] P. C. Woodland. Speaker adaptation for continuous density HMMs: A review. In *ISCA Tutorial and Research Workshop (ITRW) on Adaptation Methods for Speech Recognition*, pages 11–19, 2001.

[161] A. A. Wrench. A multi-channel/multi-speaker articulatory database for continuous speech recognition research. In *Phonus 5, Institute of Phonetics, University of the Saarland*, pages 1–13, 2000.

[162] A. A. Wrench and W. J. Hardcastle. A multichannel articulatory speech database and its application for automatic speech recognition. In *Proceedings of the 5th Speech Production Workshop: Models and Data*, 2000.

[163] C. Yang and M. Stone. Dynamic programming method for temporal registration of three-dimensional tongue surface motion from multiple utterances. *Speech Communication*, 38(1-2):201–209, 2002.

[164] L. Zhang and S. Renals. Acoustic-articulatory modelling with the trajectory HMM. *IEEE Letters on Signal Processing*, 15:245–248, 2007.

[165] X. Zhou, C. Y. Espy-Wilson, S. Boyce, M. Tiede, C. Holland, and A. Choe. A magnetic resonance imaging-based articulatory and acoustic study of retroflex and bunched american english /r/. *J. Acoustic Soc. Amer.*, 123(6):4466, 2008.

[166] I. Zlokarnik. Adding articulatory features to acoustic features for automatic speech recognition. *J. Acoustic Soc. Amer.*, 97(5):3246, 1995.