

# PERFORMATIVE STATISTICAL PARAMETRIC SPEECH SYNTHESIS APPLIED TO INTERACTIVE DESIGNS

**MARIA ASTRINAKI**

September 2010 - September 2014



A dissertation submitted to the Faculty of Engineering of the University of Mons,  
for the degree of Doctor of Philosophy in Engineering Science

Supervisor: Thierry Dutoit

Co-Supervisor: Nicolas d'Alessandro

This work was partly supported by a public-private partnership between  
University of Mons and Acapela Group SA, Belgium.

Maria Astrinaki

*Performative Statistical Parametric Speech Synthesis Applied to Interactive Designs*

Change speech one sample at a time...

© September 2010 - September 2014

## **Jury members**

Prof. **Pierre Manneback** – Université de Mons

Prof. **Thierry Dutoit** – Université de Mons

Prof. **Junichi Yamagishi** – National Institute of Informatics, Tokyo

Dr. **Nicolas d’Alessandro** – Université de Mons

Dr. **Olivier Delgrange** – Université de Mons

Dr. **Vincent Pagel** – Acapela Group SA

numediart - Institute for New Media Art Technology  
Circuit Theory and Signal Processing Lab  
University of Mons



*Omaha* means family.  
Family means nobody gets left behind, or forgotten.  
— Lilo & Stitch

Dedicated to my family, here and there.

Στην εδώ και εκεί οικογένεια μου.



## ABSTRACT

---

This dissertation introduces interactive designs in the context of statistical parametric synthesis. The objective is to develop methods and designs that enrich the Human-Computer Interaction by enabling computers (or other devices) to have more expressive and adjustable voices. First, we tackle the problem of interactive controls and present a novel method for performative HMM-based synthesis (pHTS). Second, we apply interpolation methods, initially developed for the traditional HMM-based speech synthesis system, in the interactive framework of pHTS. Third, we integrate articulatory control in our interactive approach. Fourth, we present a collection of interactive applications based on our work. Finally, we unify our research into an open source library, *MAGE*. To our current knowledge *MAGE* is the first system for interactive programming of HMM-based synthesis that allows realtime manipulation of all speech production levels. It has been used also in cases that are not related to speech, such as audio-visual laughter and stylistic gait synthesis and reconstruction.

We realise that performative HMM-based synthesis can find applications in several domains, such as entertainment and gaming, performing arts, assistive applications, culture, education, linguistics, speech pedagogy and therapy. Nevertheless, artificial speech, laughter or motion, still lacks of naturalness. However, an important contribution of this dissertation is the engagement of the user in the synthesis and generation process. This sets a first basis for the application of crowd sourcing and gamification approaches in the HMM-based synthesis domain, which will help us not only tackle several existing problems and improve the existing technology, but also form new questions to pursuit.

**Keywords:** speech synthesis, statistical parametric synthesis, hidden Markov models, HMM-based speech synthesis, HMMs, HTS, *MAGE*.



*We have seen that computer programming is an art,  
because it applies accumulated knowledge to the world,  
because it requires skill and ingenuity, and especially  
because it produces objects of beauty.*

— Donald E. Knuth [1]

## ACKNOWLEDGEMENTS

---

First and foremost, I have to admit that I have been extremely lucky throughout my life to work with people who are far more talented than I am and to learn and evolve from their wisdom. This is the reason why you are reading this book and why I have so many people to thank.

I owe my deepest gratitude to Prof. Thierry Dutoit, my supervisor, for his guidance, advice and support. This book is as much his as it is mine and I am thankful that I had the chance to know, work with and learn from him. Equally, I would like to express my special appreciation and thanks to Dr. Nicolas d’Alessandro, my co-supervisor, for encouraging every step of my research and for allowing me to grow as a research scientist.

Likewise, I owe an enormous debt of gratitude to Prof. Junichi Yamagishi for his patience, motivation, and immense knowledge. A great teacher who patiently provided vision, encouragement and invaluable advice. I could not thank him enough for not only being such a great scientist but also an amazing host in both Edinburgh and Tokyo. Similarly, I am deeply grateful to Prof. Simon King, as his trust and support gave me the opportunity to collaborate with so many other talented researchers and scientists. No words could express my gratitude to Dr. Alexis Moinet for his collaboration, involvement and unlimited support. It has been a privilege, indeed, and I am internally grateful for his contribution, which is as invaluable as it is generous.

I acknowledge the industrial partnership of Acapela Group, for the support and funding sources that made my research work possible, and especially Dr. Vincent Pagel. I am also thankful to Mr. Geoffrey Wilfart, who actively supported me during the most challenging parts.

I would like to extend my appreciation to Prof. Zhen-Hua Ling, Dr. Christophe Veaux, Dr. Korin Richmond, Dr. Rob Clark and Dr. Oliver Watts for their invaluable insights and help. I am indebted to all my colleagues in NII and CSTR, for welcoming me so warmly. It was a pleasure to share so many things with them, from both a scientific and human point of view. I am forever thankful to all my colleagues

in TCTS, with whom I share my days, numerous breaks, discussions and fun moments. They made the lab a wonderful workplace and home for the past few years. Special thanks should go to Alexis, Onur, Joëlle & Jérôme, Loïc, François, Benjamin, Hüseyin, Thomas, Anderson and of course Nathalie. My PhD journey was made so enjoyable due to the many friends I made and who became an important part of my life, thank you!

Lastly, I would like to thank my loving and patient Benoît, whose faithful support and friendship fuels every step I take. He and his family gave me a place I will always call home. I am also thankful to my family for all their unconditional love and encouragement. For my parents, Γιάννης Ασρινάκης and Ρίτσα Υπερηφάνου who raised and educated me with a love for both the arts and science and supported me in all my pursuits, even when they went beyond the boundaries of language and geography. My sister Νίνα and my brother Νίκος, my best friends and companions, for their unconditional faith in me. Your big, bright smiles ease my way though life!

# CONTENTS

---

<b>INTRODUCTION</b>	<b>1</b>
<b>1 INTRODUCTION</b>	<b>3</b>
1.1 Motivations . . . . .	4
1.2 Applications . . . . .	6
1.3 Outline of the manuscript . . . . .	7
1.3.1 Organisation of part I . . . . .	7
1.3.2 Organisation of part II . . . . .	8
1.3.3 Organisation of part III . . . . .	9
1.3.4 Original content . . . . .	9
<b>I STATE OF THE ART</b>	<b>11</b>
<b>2 SPEECH SYNTHESIS</b>	<b>13</b>
2.1 Synthesis based on vocal tract models . . . . .	14
2.1.1 Articulatory synthesis . . . . .	14
2.1.2 Formant synthesis . . . . .	16
2.2 Synthesis by concatenation . . . . .	18
2.2.1 Diphone synthesis . . . . .	18
2.2.2 Unit selection synthesis . . . . .	20
2.3 Statistical parametric synthesis . . . . .	21
2.4 Discussion . . . . .	22
<b>3 STATISTICAL PARAMETRIC SYNTHESIS</b>	<b>27</b>
3.1 Hidden Markov Models . . . . .	28
3.2 HMM-based speech synthesis . . . . .	30
3.2.1 Speech parameter generation . . . . .	31
3.2.2 Incorporating dynamic features . . . . .	32
3.2.3 Training . . . . .	35
3.2.4 Synthesis . . . . .	36
3.2.5 Advantages . . . . .	38
3.2.6 Drawbacks . . . . .	44
3.3 HMM-based synthesis in other fields . . . . .	44
3.3.1 HMM-based singing synthesis . . . . .	45
3.3.2 Audio-visual laughter synthesis . . . . .	45

3.3.3	Stylistic gait synthesis . . . . .	46
3.4	Discussion . . . . .	47
<b>II</b>	<b>PERFORMATIVE HMM-BASED SYNTHESIS</b>	<b>49</b>
4	REDUCED-CONTEXT HMM-BASED SPEECH SYNTHESIS	51
4.1	Short-term parameter generation . . . . .	52
4.2	Models trained with reduced phonetic context . . . . .	57
4.3	Evaluation . . . . .	59
4.3.1	Experimental protocol . . . . .	60
4.3.2	Objective evaluation . . . . .	61
4.3.3	Subjective evaluation . . . . .	70
4.4	Discussion . . . . .	74
5	REACTIVE AND CONTINUOUS MODEL INTERPOLATION	77
5.1	Reactive synthesis using model interpolation . . . . .	79
5.1.1	Defining reactive model interpolation . . . . .	79
5.1.2	Interpolation strategies . . . . .	80
5.1.3	Reactive large-scale interpolation . . . . .	82
5.2	Interactive modification of the degree of articulation . . . . .	83
5.2.1	Evaluation . . . . .	84
5.3	Interactive accent interpolation . . . . .	87
5.4	Discussion . . . . .	92
6	REACTIVE ARTICULATORY FEATURE GENERATION	95
6.1	Reactive synthesis using articulatory features . . . . .	96
6.2	Evaluation . . . . .	99
6.2.1	Experimental protocol . . . . .	99
6.2.2	Objective evaluation . . . . .	101
6.2.3	Subjective evaluation . . . . .	105
6.3	Reactive articulatory control . . . . .	107
6.3.1	Reactive articulatory control application . . . . .	107
6.3.2	Challenges . . . . .	108
6.4	Discussion . . . . .	109
<b>III</b>	<b>PERFORMATIVE DESIGNS</b>	<b>113</b>
7	PERFORMATIVE APPLICATIONS	115
7.1	Speech related prototypes . . . . .	116
7.1.1	Reactive speech synthesis controlled by hand gestures .	116
7.1.2	Reactive speech synthesis controlled by facial gestures .	118

7.1.3	Talking guitar . . . . .	118
7.1.4	Accent interpolation through an interactive map . . . . .	120
7.1.5	Realtime articulatory control through a reactive vocal tract . . . . .	122
7.1.6	Reactive audiobooks . . . . .	124
7.1.7	Speaking & singing puppet . . . . .	128
7.1.8	Tanukis . . . . .	131
7.1.9	Other cases . . . . .	131
7.2	Beyond speech . . . . .	132
7.2.1	Reactive laughter synthesis . . . . .	132
7.2.2	Reactive stylistic walk synthesis . . . . .	133
7.2.3	Reactive singing synthesis . . . . .	134
7.2.4	Incremental speech synthesis . . . . .	136
7.3	Potential applications . . . . .	137
7.4	Discussion . . . . .	137
8	MAGE/PHTS: PERFORMATIVE HMM-BASED SYNTHESIS LIBRARY	139
8.1	Realtime architecture of MAGE . . . . .	140
8.2	Reactive controls . . . . .	142
8.2.1	Available controls . . . . .	142
8.2.2	Regular expressions . . . . .	143
8.2.3	Logging user actions . . . . .	144
8.2.4	Introducing a <i>state queue</i> . . . . .	144
8.3	C++ API . . . . .	146
8.4	Discussion . . . . .	148
	<b>CONCLUSIONS</b>	149
9	CONCLUSIONS AND FUTURE WORK	151
9.1	Conclusions . . . . .	151
9.2	Future work . . . . .	155
	<b>APPENDIX</b>	157
A	HTS	159
A.1	Label format . . . . .	159
A.2	Contextual factors . . . . .	161
A.3	Singing contextual factors . . . . .	162
A.4	Decision tree example . . . . .	162
A.5	Question file example . . . . .	169

B	MAGE	175
B.1	Integrations of MAGE . . . . .	175
B.1.1	Pure data & Max MSP object ( <b>mage~</b> ) . . . . .	175
B.1.2	openFrameworks, iOS . . . . .	177
B.2	MAGE demo links . . . . .	178
B.3	License . . . . .	178
C	RELATED PROJECTS	181
C.1	Speech synthesis projects . . . . .	181
C.2	Speech corpuses . . . . .	182
C.3	Other speech related links . . . . .	182
C.4	Beyond speech projects . . . . .	183
C.5	Fun links . . . . .	183
	BIBLIOGRAPHY	185
	PUBLICATIONS	195

## LIST OF FIGURES

---

Figure 1.1	The interdisciplinary nature of speech research . . . . .	5
Figure 2.1	Typical formant synthesiser layout . . . . .	16
Figure 2.2	Simple example of diphone synthesis . . . . .	19
Figure 2.3	Graphical illustration of the unit selection synthesis . . .	21
Figure 2.4	Block diagram of the HMM-based speech synthesis system (HTS) . . . . .	23
Figure 2.5	Examples and techniques in speech synthesis history . .	24
Figure 3.1	Typical 3-state left-to-right HMM . . . . .	29
Figure 3.2	Example of an observation vector of each frame . . . . .	33
Figure 3.3	Overview of HMM-based speech synthesis scheme . . .	37
Figure 3.4	Stylistic model interpolation . . . . .	40
Figure 3.5	Overview of multiple-regression HMM-based emotion-controlling technique . . . . .	42
Figure 3.6	Example postures taken from the Mockey database . . .	46
Figure 4.1	Standard & performative HMM-based speech synthesis	54
Figure 4.2	Short-term parameter generation algorithm . . . . .	57
Figure 4.3	Full & reduced-context training . . . . .	59
Figure 4.4	Simple diagram of the examined <b>cases</b> of reduced and full-context regarding training and synthesis . . . . .	60
Figure 4.5	Mean <i>Mel-Cepstral Distortion</i> introduced when comparing HTS to pHTS along with the two training schemes .	64
Figure 4.6	<i>Root-Mean-Square Error</i> of the fundamental frequency introduced when comparing HTS to pHTS along with the two training schemes . . . . .	64
Figure 4.7	Mean <i>Mel-Cepstral Distortion</i> introduced when comparing full to reduced-context models along with the two synthesis schemes . . . . .	66
Figure 4.8	<i>Root-Mean-Square Error</i> of the fundamental frequency introduced when comparing full to reduced-context models along with the two synthesis schemes . . . . .	67

Figure 4.9	Mean <i>Mel-Cepstral Distortion</i> introduced when comparing HTS to pHTS combined with varying-size shifting window . . . . .	70
Figure 4.10	<i>Root-Mean-Square Error</i> of the fundamental frequency introduced when comparing HTS to pHTS combined with varying-size shifting window . . . . .	71
Figure 4.11	CMOS scores when comparing HTS to pHTS along with the two training schemes . . . . .	72
Figure 4.12	CMOS scores when comparing full to reduced-context model training along with the two synthesis schemes . . .	73
Figure 5.1	Reactive model interpolation . . . . .	81
Figure 5.2	Interpolation strategies . . . . .	82
Figure 5.3	Fluidity scores of the transition between the two degrees of articulation . . . . .	87
Figure 5.4	Interactive accent control by means of a stylistic mixer . .	90
Figure 5.5	Stylistic mixers for interactive model interpolation . . . .	91
Figure 6.1	Reactive generation of acoustic features with articulatory control . . . . .	99
Figure 6.2	Placement of the EMA receivers in the database used for the experiments . . . . .	100
Figure 6.3	Mean <i>Mel-Cepstral Distortion</i> and <i>Root-Mean-Square Error</i> of the fundamental frequency introduced when comparing HTS to pHTS, combined with articulatory features, computed over the <i>phoneme</i> and <i>vowel</i> set . . . . .	102
Figure 6.4	<i>Euclidean Distance</i> between EMA positions generated by HTS and pHTS, computed over the <i>phoneme</i> and <i>vowel</i> set . . . . .	104
Figure 6.5	CMOS scores when comparing HTS to pHTS when using articulatory features . . . . .	106
Figure 6.6	Stylistic mixer for interactive articulatory control . . . .	108
Figure 7.1	Reactive speech synthesis controlled by hand gestures . .	117
Figure 7.2	Reactive speech synthesis controlled by facial gestures . .	119
Figure 7.3	“Talking guitar” performance setup . . . . .	120
Figure 7.4	Realtime accent interpolation through an interactive map .	121
Figure 7.5	Realtime articulatory control through a reactive vocal tract . . . . .	123
Figure 7.6	Reactive audiobooks - voice casting . . . . .	125

Figure 7.7	Reactive audiobooks - voice customisation . . . . .	126
Figure 7.8	Reactive audiobooks - sentence customisation . . . . .	128
Figure 7.9	Kinect as prosodic controller of speech synthesis . . . . .	129
Figure 7.10	“Puppet” avatar as speech & singing synthesis controller.	130
Figure 7.11	Kinect as prosodic controller of singing synthesis . . . . .	130
Figure 7.12	Tanukis . . . . .	131
Figure 7.13	Reactive laughter synthesis (visual) . . . . .	133
Figure 7.14	Reactive stylistic gait synthesis . . . . .	135
Figure 8.1	The MAGE project . . . . .	140
Figure 8.2	MAGE architecture . . . . .	141
Figure 8.3	MAGE reactive controls . . . . .	143
Figure 9.1	Short summary of research achievements . . . . .	155
Figure B.1	The <b>mage~</b> object . . . . .	177

## LIST OF TABLES

---

Table 1.1	Examined systems . . . . .	8
Table 4.1	Mean <i>Mel-Cepstral Distortion</i> introduced when comparing HTS to pHTS along with the two training schemes, computed over the <i>phoneme</i> set . . . . .	63
Table 4.2	Mean <i>Mel-Cepstral Distortion</i> introduced when comparing HTS to pHTS along with the two training schemes, computed over the <i>vowel</i> set . . . . .	63
Table 4.3	<i>Root-Mean-Square Error</i> of the fundamental frequency introduced when comparing HTS to pHTS along with the two training schemes, computed over the <i>phoneme</i> set . . . . .	63
Table 4.4	<i>Root-Mean-Square Error</i> of the fundamental frequency introduced when comparing HTS to pHTS along with the two training schemes, computed over the <i>vowel</i> set . . . . .	63
Table 4.5	Mean <i>Mel-Cepstral Distortion</i> introduced when comparing full to reduced-context models along with the two synthesis schemes, computed over the <i>phoneme</i> set . . . . .	65
Table 4.6	Mean <i>Mel-Cepstral Distortion</i> introduced when comparing full to reduced-context models along with the two synthesis schemes, computed over the <i>vowel</i> set . . . . .	65
Table 4.7	<i>Root-Mean-Square Error</i> of the fundamental frequency introduced when comparing full to reduced-context models along with the two synthesis schemes, computed over the <i>phoneme</i> set . . . . .	65
Table 4.8	<i>Root-Mean-Square Error</i> of the fundamental frequency introduced when comparing full to reduced-context models along with the two synthesis schemes, computed over the <i>vowel</i> set . . . . .	66
Table 4.9	Mean <i>Mel-Cepstral Distortion</i> introduced when comparing HTS to pHTS combined with shifting window including the current and varying number of preceding labels, computed over the <i>phoneme</i> set . . . . .	68

Table 4.10	Mean <i>Mel-Cepstral Distortion</i> introduced when comparing HTS to pHTS combined with shifting window including one succeeding, the current and varying number of preceding labels, computed over the <i>phoneme</i> set . . . . .	68
Table 4.11	<i>Root-Mean-Square Error</i> of the fundamental frequency introduced when comparing HTS to pHTS combined with shifting window including the current and varying number of preceding labels, computed over the <i>phoneme</i> set . . . . .	69
Table 4.12	<i>Root-Mean-Square Error</i> of the fundamental frequency introduced when comparing HTS to pHTS combined with shifting window including one succeeding, the current and varying number of preceding labels, computed over the <i>phoneme</i> set . . . . .	69
Table 4.13	CMOS scores when comparing HTS to pHTS along with the two training schemes . . . . .	72
Table 4.14	CMOS scores when comparing full to reduced-context model training along with the two synthesis schemes . . . . .	73
Table 5.1	Fluidity scores of the transition between the two degrees of articulation . . . . .	86
Table 6.1	Mean <i>Mel-Cepstral Distortion</i> and <i>Root-Mean-Square Error</i> of the fundamental frequency introduced when comparing HTS to pHTS, combined with articulatory features, computed over the <i>phoneme</i> and <i>vowel</i> set . . . . .	102
Table 6.2	<i>Euclidean Distance</i> between EMA positions generated by HTS and pHTS, computed over the <i>phoneme</i> set . . . . .	103
Table 6.3	<i>Euclidean Distance</i> between EMA positions generated by HTS and pHTS, computed over the <i>vowel</i> set . . . . .	103
Table 6.4	CMOS scores when comparing HTS to pHTS when introducing articulatory features . . . . .	106

## ACRONYMS

---

API	Application Programming Interface
AVM	Average Voice Model
CMLLR	Constrained Maximum Likelihood Linear Regression
CMOS	Comparative Mean Opinion Score
dB	Decibel
DRY	Don't Repeat Yourself
EM	Expectation Maximisation
FSR	Force Sensing Resistors
FSS-MRHMM	Feature-Space-Switched Multiple Regression HMM
HMM	Hidden Markov Model
HTS	HMM-based speech synthesis system
Hz	Herz
LSP	Linear Spectral Pair
MAP	Maximum A Posteriori
Mel-CD	Mel-Cepstral Distortion
MGC	Mel Generalized Cepstral
ML	Maximum Likelihood
MLLR	Maximum Likelihood Linear Regression
MLPG	Maximum Likelihood Parameter Generation
MLSA	Mel Log Spectrum Approximation

MND	Motor Neurone Disease
MRHMM	Multiple Regression HMM
NLP	Natural Language Processor
OVE	Orator Verbis Electricis
pHTS	performative HMM-based speech synthesis system
RMSE	Root Mean Square Error
RNLP	Reactive Natural Language Processor
ST-MLPG	Short-Term Maximum Likelihood Parameter Generation
TTS	Text-To-Speech
VTL	Vocal Tract Length
XML	Extensible Markup Language

Author's publication list uses the *alpha* bibliography style.  
Bibliography list uses the *plain* bibliography style.



# INTRODUCTION

INTRODUCTION	1
1 INTRODUCTION	3
1.1 Motivations . . . . .	4
1.2 Applications . . . . .	6
1.3 Outline of the manuscript . . . . .	7



## INTRODUCTION

---

### Contents

---

1.1	Motivations . . . . .	4
1.2	Applications . . . . .	6
1.3	Outline of the manuscript . . . . .	7
1.3.1	Organisation of part I . . . . .	7
1.3.2	Organisation of part II . . . . .	8
1.3.3	Organisation of part III . . . . .	9
1.3.4	Original content . . . . .	9

---

The more one studies the human voice the more fascinating this phenomenon becomes. Of course many people around the world have been similarly charmed about someone else's or their own sound-producing organ. Speech is a very powerful usage of the vocal apparatus, which is a difficult instrument. Yet everyone seems to be an expert player. Humans are able to voluntarily adjust their voices, from very smooth to really rough, and express a great variety of emotions. Even though the generation of speech can be observed simply from its mechanical point of view, i. e. a coordinated movement of the articulators, there is certainly much more to it<sup>1</sup>. How could we possibly detach emotion from speech?

Actually, emotions, breathing, yawns, cries, moans, laughter, squeaks and shouts are an essential part of human communication and in many cases are part of the speech itself. One might argue that crying, laughing or yawning<sup>2</sup> while speaking results in poor speech quality. But is it also poor vocal communication? We see that speech is not just the sound of a language targeting to convey a message but also the mean of transmitting our feelings, identities and personalities. It is the mean to connect with other beings.

---

<sup>1</sup> Overtone singing with X-ray (visited June 2014): <http://vimeo.com/20237275>

<sup>2</sup> Not polite, I agree but we shall use it for the sake of argument.

We could not exclude of course the influence of the environment when it comes to voice and vocal identity. Voices are coloured by accents, dialects, social and regional characteristics and certainly psychological factors. A person's pitch and loudness can be unstable in emotional distress, as for example fear or grief. Gender and age put also their stamp on voice. What is indeed interesting is that although an average person is able to recognise familiar or famous voices, the same person has strong difficulties to verbally describe them<sup>3</sup>.

Even though voice is a profound characteristic of a person, one may control or distort ones vocal identity in order to create vocal caricatures, sound cartoons or mimic others. Ventriloquists or impersonators are bright examples of the human ability to control one's own vocal identity<sup>4</sup>.

Voice is an extraordinary instrument that has been fascinating people for centuries. It is a mean of communication and expression. It is an essential social signal and dynamic gestural phenomenon influenced constantly from environmental factors, either fast-changing such as mood or slowly-changing such as age. It is reflecting ones personality and emotions. And this very nature of speech and voice is what brings us to the motivation of this work.

## 1.1 MOTIVATIONS

Everyone more or less has experienced artificially generated speech in everyday life situations, e. g. call centres, transportation announcements, GPS directions. The field of study that aims in making computers (or machines/devices) speak is known both as *speech synthesis*, i. e. artificially generation of speech waveforms and as *Text-To-Speech* (TTS), i. e. the process of converting written text into speech. Speech synthesis has been long engaging artists and researchers, and it is no longer the case that the state-of-the-art systems sound mechanical and robotic. Besides, the speech research field has an interdisciplinary nature, as shown in [Figure 1.1](#).

We see that the two dominant approaches presented in the [state-of-the-art - unit selection synthesis](#) and [statistical parametric synthesis](#) - have achieved to

---

<sup>3</sup> It is a problem similar to taste description.

<sup>4</sup> 29 Celebrity Impressions, 1 Original Song - Rob Cantor (visited June 2014):

<https://www.youtube.com/watch?v=k6PxMRUgmbA>

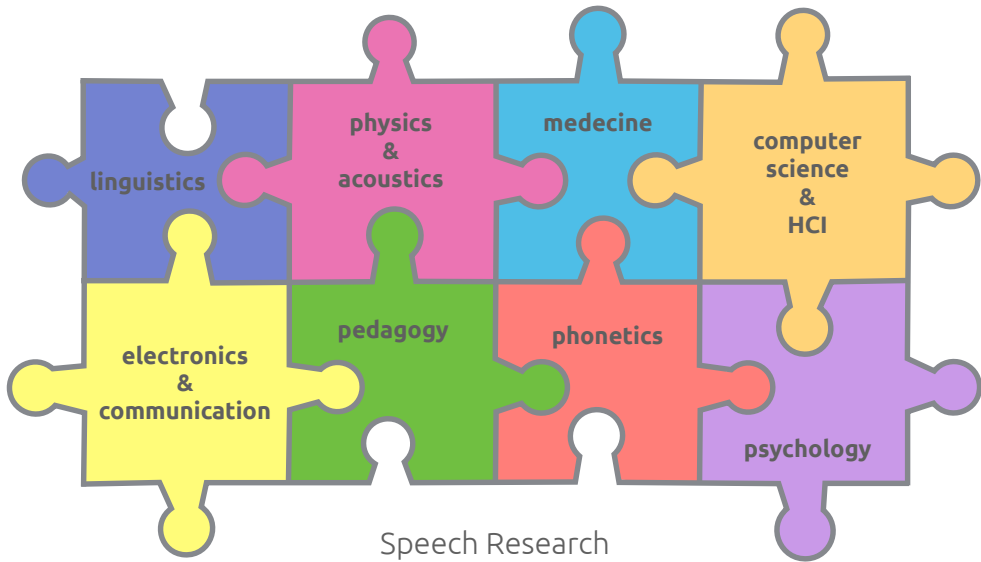


Figure 1.1: The interdisciplinary nature of speech research.

synthesise natural and intelligible artificial speech. However, we see that in order to achieve that high quality output the resulting systems need to take into account complex linguistic context. Additionally, the speech parameters or samples need to be refined over all this context and, in some cases, post-processing is required. Hence, the resulting speech output may not be altered while it is being generated. However, such an approach is not in accordance with the dynamic nature of speech itself.

We realise that in the field of voice synthesis there is need for *performance* and *interaction*, and plenty of room for creativity and expression<sup>5</sup>. We want to bring together fields such as Computer Science, Human-Computer Interaction (HCI) and Engineering and contribute to the interdisciplinary speech research domain. We want to move towards systems that are able to “listen”, “express” and “adjust” and to investigate the role of a *performer* in the speech synthesis process. An important axis of this work is to come up also with interface designs, and embed this *performative* feature in new prototypes.

In this work we focus on the *statistical parametric synthesis* method and more specifically on the *HMM-based speech synthesis system (HTS)* [2], [3], [4].

<sup>5</sup> Problems, barriers and limits boost the creativity.

HTS combines good quality output, flexibility and controllability on different aspects of the generated output. As it will be described in details in the following chapters, HTS consists of two main parts, the training and the synthesis. In both parts complex context-dependent linguistic factors are taken into account and moreover during synthesis, all the possible available linguistic context (i. e. a complete utterance) is used in order to generate the best possible output quality. In other words, the smallest accessible time scale in HTS is the utterance itself, which is indeed sufficient in the context of standard TTS.

Having a sentence-by-sentence system does not really reflect the dynamic nature of human voice and speech: its realtime and reactive/interactive aspect. The terms *realtime* and *reactive/interactive* may have many interpretations depending on which discipline they are being viewed from. For example, HTS itself is actually *computationally realtime*, which means that the duration of the computation is shorter than the duration of the generated speech. However, when it comes to system interaction, these terms are used to describe a system that is able to adequately and instantaneously react to user input. This property is sometimes called *performative*. In this work we propose a novel method for *performative HMM-based synthesis* that we call *pHTS*. It is actually the *realtime* and *reactive/interactive* version of the standard *HMM-based speech synthesis*.

The methods developed in this thesis aim at enriching Human-Computer Interaction by enabling computers (or other devices) to have more expressive and adjustable voices. On the one hand, having a “listening” system that reflects any environmental influence may trigger better and more appropriate responses. However the timing of such decisions is not studied here. On the other hand, involving users in the speech production process may help not only to improve the speech quality but also learn and better understand which properties of a speech influence its perception.

## 1.2 APPLICATIONS

Performative HMM-based speech synthesis can find applications in several domains. Some of them are briefly described hereafter. However note that this list is not exhaustive.

*Entertainment* and *gaming* applications can benefit from the performative aspect of HMM-based synthesis. The speech, audio-visual laughter or motion of an animated character or avatar can be interactively controlled or fine tuned. This same principle can be applied in automatic movie dubbing. Reactive audiobooks, augmented comic books, geolocalized reminders and voice messaging applications or vocal treasure hunting games could profit from such a performative system. The list continues with *performing arts* and of course *new interfaces for musical expression*.

*Assistive applications* for speech-impaired people could certainly use such technology. Linguistic context and emotion of the artificial voice can be controlled accurately in realtime. In order to build and/or reconstruct personalised voices, the fine tuning and customisation can be achieved in a reactive fashion. Voices can be built in order to match the targeted person<sup>6</sup>, in terms of style, accent, etc. The creation of unique interactive personalised voices may benefit to speech-to-speech translation systems, silent speech communication, in-car navigation systems, call centres or even *culture* and *education*. *Linguistics, speech pedagogy and therapy* may also use performative HMM-based speech synthesis in a great variety of applications.

### 1.3 OUTLINE OF THE MANUSCRIPT

Previous sections have advocated the interest of performative speech synthesis and given a general introduction of the field and our motivations. Chapters of this thesis are organised into three parts, that are followed by general [conclusions](#) and completed with the [appendixes](#).

#### 1.3.1 Organisation of part I

The [first Part](#) of this dissertation is dedicated to the state-of-the-art of speech synthesis. [Chapter 2](#) introduces the various existing methods as well as different systems developed on each approach. We briefly describe the articulatory and formant synthesis, the diphone and unit selection synthesis and lastly the statistical parametric synthesis. Several examples of each method are also given. The experienced reader in speech synthesis can safely skip

---

<sup>6</sup> Everyone actually has minimum three voices: the one you hear when you talk, the one that everyone else hears instead and at least one in your head. Imaginary friends are also to be included.

this chapter. [Chapter 3](#) gives a detailed overview of the statistical parametric synthesis approach, on which our work is based. We detail the statistical model - hidden Markov model (HMM) - widely used in this method and then an extended description of the HMM-based speech synthesis system is given. Advantages, drawbacks and applications of this method in fields beyond speech are also discussed. Again, a reader experienced in HMM-based speech synthesis can safely skip this chapter.

### 1.3.2 Organisation of part II

The [second Part](#) of this thesis addresses the specific problem of performative designs in the context of HMM-based speech synthesis. [Chapter 4](#) is actually the core of this work, that leads to the development of a novel method for *performative HMM-based synthesis - pHTS*. *pHTS* is actually the *realtime* and *reactive/interactive* version of the standard *HMM-based speech synthesis*. Here, we describe all the essential modifications needed to achieve the short-term parameter generation. We examine the influence of reduced-context synthesis, either during training or synthesis, as detailed in [Table 1.1](#).

	Full-context training	Reduced-context training
Full-context synthesis (HTS)	<a href="#">System I</a>	<a href="#">System III</a>
Reduced-context synthesis (pHTS)	<a href="#">System II</a>	<a href="#">System IV</a>

Table 1.1: Systems examined in this thesis.

System I. HTS using full linguistic context models

System II. pHTS using full linguistic context models

System III. HTS using reduced linguistic context models

System IV. pHTS using reduced linguistic context models

In order to cross-compare the final output quality of all possible [systems](#), different test [cases](#) are objectively and subjectively examined. [Chapter 5](#) presents the integration of model interpolation in the performative context. It explains how reactive and continuous model interpolation is integrated in pHTS, along with two example cases and its potential applications. Although the simplest interpolation approach is adopted here, very interesting results are obtained from both test cases. Similarly in [Chapter 6](#) the movement of the human articulators is controlled interactively in pHTS, influencing the generated speech. Here the goal is to alter the generated speech samples at the articulatory level rather than directly at the acoustic level. Lastly, the system evaluation is presented, while challenges and applications are discussed.

### 1.3.3 *Organisation of part III*

The [third Part](#) of this thesis completes and concludes our work by presenting actual applications using our library developed for performative HMM-based synthesis. [Chapter 7](#) presents several proof of concept examples related to speech synthesis, focusing mainly on gestural control of prosody. Moreover, real-world application prototypes and examples beyond speech are detailed. [Chapter 8](#) details the essential realtime architecture and multi-threaded controls used to unify the different aspects of our work presented in this manuscript. This led to the creation of an open source library, *MAGE*, so that ideas can be easily implemented and shared.

### 1.3.4 *Original content*

The original work presented in this thesis contains five novelties:

- a novel method for performative HMM-based synthesis [[Chapter 4](#)];
- performative interpolation methods [[Chapter 5](#)];
- interactive control of the articulators [[Chapter 6](#)];
- a collection of interactive applications based on our work [[Chapter 7](#)];
- an open source library that unifies our work [[Chapter 8](#)];



## Part I

### STATE OF THE ART

I	STATE OF THE ART	11
2	SPEECH SYNTHESIS	13
2.1	Synthesis based on vocal tract models . . . . .	14
2.2	Synthesis by concatenation . . . . .	18
2.3	Statistical parametric synthesis . . . . .	21
2.4	Discussion . . . . .	22
3	STATISTICAL PARAMETRIC SYNTHESIS	27
3.1	Hidden Markov Models . . . . .	28
3.2	HMM-based speech synthesis . . . . .	30
3.3	HMM-based synthesis in other fields . . . . .	44
3.4	Discussion . . . . .	47



# 2

## SPEECH SYNTHESIS

---

### Contents

---

2.1	Synthesis based on vocal tract models . . . . .	14
2.1.1	Articulatory synthesis . . . . .	14
2.1.2	Formant synthesis . . . . .	16
2.2	Synthesis by concatenation . . . . .	18
2.2.1	Diphone synthesis . . . . .	18
2.2.2	Unit selection synthesis . . . . .	20
2.3	Statistical parametric synthesis . . . . .	21
2.4	Discussion . . . . .	22

---

Producing speech by means other than the natural vocal apparatus has long fascinated people. Examples of “modern” artificial speech production mechanisms date from the XVIII<sup>th</sup> century with the well-known von Kempelen machine [5]. Prior to that we know of prototypes for various experimental organs that used non-cylindrical pipes to create vowel-like sounds. These attempts to artificially mimic speech (or even singing) were dependent on using the best available technology of the periods, a practice that has continued and can be seen in the historical use of mechanical, electrical and then digital devices. Significant advances in voice research such as the first electrical or computer-based speech synthesisers - respectively known as the Voder [6] and the Vocoder [7] - emphasise this aspect of speech synthesis development. Of course, the development of speech synthesis is not isolated from other developments in speech technology. For instance the development of speech recognition has also benefited from the reduction in cost of computational power and increased availability of general computing (i. e. processing and memory capacities). The availability of synthesis systems, such as the Festival Speech Synthesis System [8], the MBROLA project [9] and the HMM-based speech synthesis system [2], makes the cost of entering

the field of speech synthesis much lower, allowing many research groups to join the development.

In this Chapter we give an overview of various systems used in the speech synthesis field. In [Section 2.1](#) we detail the first synthesis techniques, articulatory and formant synthesis respectively, along with several examples. In [Section 2.2](#) we present the diphone and unit selection synthesis techniques with several application examples. In [Section 2.3](#) we present the statistical parametric synthesis approach, which is the latest-developed technique, and will be further detailed later on, as it is the method upon which our PhD research has been based. Finally in [Section 2.4](#) we discuss the presented approaches, their advantages and disadvantages and the motivations that initiated this research.

## 2.1 SYNTHESIS BASED ON VOCAL TRACT MODELS

Here we will describe briefly the first speech synthesis techniques, also referred to as *first generation techniques*, that are based on models of the human vocal tract. These approaches are less used in nowadays synthesis systems, due to their poor output quality. These systems create the final speech waveform “from scratch” rather than using actual speech samples or statistics as it will be described further on. Actually, the advantage of the format synthesis is that they use small footprints and low computational power, an essential feature for that time and still competitive nowadays. Besides, the articulatory approach is used mainly in speech simulation, to understand the production, and not yet in synthesis, where the pure sound quality matters more.

### 2.1.1 *Articulatory synthesis*

The articulatory approach in speech synthesis refers to techniques that are based on modelling the human vocal tract and the articulation processes occurring there. By modifying the position of the speech articulators, i.e. tongue, lips, lower jaw, the shape of the vocal tract is controlled. The synthesised speech is then generated by simulating the air flow through the modelled vocal tract. In articulatory models we can build complex tube shapes by using smaller tubes, and based on the sound propagation properties to

approximate the human articulatory system. With small movements in these tubes, we can approximate complex speech patterns.

There are both mechanical and software articulatory synthesisers. Actually, the first examples of speech synthesis were developed using a mechanical approach of the articulatory synthesis. In 1779 Kratzenstein [10] built models of the human vocal tract that could produce five long vowel sounds while in 1791 von Kempelen [5] presented the talking machine, a mechanical articulatory synthesiser, that consisted of tubes, bellows and pipes and was able to produce recognisable speech. Based on von Kempelen's design, Wheatstone in 1838 produced a "speaking machine" [11] capable to produce vowels and most of the consonant sounds, even full words. The same year Willis discovered the connection between a specific vowel sound and the geometry of the vocal tract. He also proved that the vowel quality depends only on the length of the tube used and not on its diameter. In 1857 and 1909, Faber and Bell respectively, presented the first "interfaced" articulatory synthesis approaches. Faber builds the "Euphonia"<sup>1</sup> [12], which is actually a complex device controlled by seventeen levers, bellows, and a telegraphic line. This machine was able to create consonant and vowel sounds and was paired with a movable replica of a human face<sup>2</sup>. Similarly, Bell [13] created a rubber-moulded skull with actuators for manipulating tongue and jaw positions. By carefully adjusting the lips, when a bellows forced air through the windpipe a small set of words could be created. In the 50's we have the first electrical, hardware-based vocal tract analogs, that were static [14], [15], and the first articulatory synthesiser with dynamic model was "DAVO" (Dynamic Analog of the VOcal tract) [16], which was controlled by tape recordings of handcrafted control signals. The first computer simulation was presented by Kelly [7] in 1962 and Umeda [16] in 1968 presented the first full TTS system for English. Based on a syntactic analysis module with complex heuristics and an articulatory model, a given sequence of phonemes was transformed into vocal tract shapes, duration, and pitch signals. Its output was rather intelligible but monotonous. The first software articulatory synthesiser was "ASY" [17] developed at Haskins Laboratories in 1981. Modern examples are "CASY" [18] and the use of "ArtiSynth" [19]. "CASY" matched midsagittal vocal tracts to actual magnetic resonance imaging (MRI) data, and used these

---

1 pleasant sounding pronunciation of words - in Greek

2 kinda creepy though

data to construct a 3D model, whereas “ArtiSynth” was a 3D biomechanical toolkit, initially designed for modelling the human vocal tract and to which upper airway simulations was added later for sound production.

Although articulatory synthesis provides a “physical” understanding of the artificial speech production, there are two considerably difficult problems in the articulatory speech synthesis approach. First is the generation of the control parameters, and second is the balancing between accuracy and easy design. More accurate the model is, the closer it follows the human physiology but at the same time it becomes more difficult to control, requiring great computational power. Similarly, the simpler the model, the easier it is to compute and control but it results in poor speech quality compared to other synthesis methods. Therefore it is not used in high-quality speech generation, and is not considered to be the best solution for TTS systems.

### 2.1.2 Formant synthesis

Formant synthesis, also known as *rule-based synthesis*, was the first electrical or computer-based synthesis technique [20]. In this approach, the creation of synthesised speech is based on subtractive synthesis and acoustic-phonetic model. No human speech samples are used at runtime, the parameters used for the synthesis of the output are fundamental frequency, voicing, and noise levels that vary over time [21]. The synthetic waveform produced is intelligible but it sounds robotic and artificial, therefore it is easily distinguishable from human speech [22]. A typical formant synthesiser structure is shown in Figure 2.1.

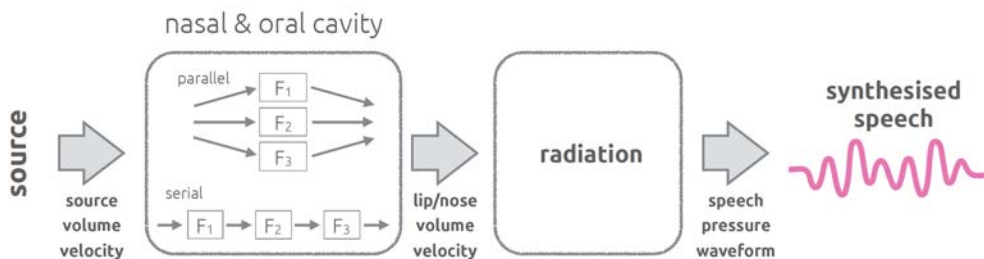


Figure 2.1: A block diagram of a basic formant synthesiser, either parallel or serial.

Formant synthesis was the dominant technique until the early 80’s and we have examples since the beginning of the 20’s. In 1922 Stewart [16] pre-

sented the first full electrical synthesis device able to generate single static vowel sounds with two lowest formants, although neither consonants nor connected utterances were feasible. In the mid 30's Bell Labs developed "The Vocoder" [7], which was then interfaced by Dudley creating "The Voder" [6]. Indeed, a keyboard and foot pedal were employed to control a set of linearly spaced bandpass frequency generators. It was first demonstrated at the World's Fair in 1939 by operators who had trained continuously for one year in order to drive the system and learn to speak with hand gestures. In addition to the obvious contribution in the speech synthesis community, "The Voder" shows also that using hand gestures to drive a speech synthesis system is a difficult task, which often requires lengthy training period. In 1951 Cooper developed the "Pattern Playback" [16] synthesiser at the Haskins Laboratories, which converted recorded spectrogram patterns, either the original or stylised hand-painted formant patterns, into sounds. One year later, Fant presented "OVE I" (Orator Verbis Electris) [23], which was a hand-controlled system enabling the manipulation of fundamental frequency as well as the first and second formant. It was able to produce rather natural vowels and simple sentences. Then in 1953 Lawrence presented the "PAT" (Parametric Artificial Talker) [16], the first formant synthesiser (three electronic formant resonators connected in parallel), where painted patterns were converted into six time functions to control the three formant frequencies, voicing amplitude, fundamental frequency, and noise amplitude. In 1961 we have the vocoder-based "Daisy Bell" song<sup>3</sup> [7]. In 1962 "OVE II" [10] made it possible to synthesise unrestricted connected speech and in 1975 it was interfaced by a language and phonetics program [24] bringing a major breakthrough in the speech synthesis domain. In the late 70's we have "Speak & Spell" from Texas Instruments [10] and "MITalk" system [16] as applications of the formant synthesiser, while in the 80's the "Klattalk" [25] and the "MITalk" system formed the bases for a complete software application: the "DECTalk" - Digital Equipment Corporation (DEC) [25].

The main two advantages of formant synthesisers are that they are memory efficient without the need of big computational power and they are small programs, since there is no need of a pre-recorded speech database. Consequently, formant synthesisers are easily embedded in systems where memory and processor power are especially limited. Although formant synthesis

---

3 First computer to sing - Daisy Bell (visited June 2014): <https://www.youtube.com/watch?v=41U78QP8nBk>

is intelligible and provides high controllability, the final speech output does not sound natural. This is due to the fact that source and filter models are rather simplistic and do not use real speech samples during synthesis, hence missing the subtleties of human speech.

## 2.2 SYNTHESIS BY CONCATENATION

In the case of articulatory synthesis it is evident that the limitations come from the complexity of the models for the speech production. The simple models with fair controllability give poor output quality while more complex models increase the output quality in exchange of more complicated or demanding controls. Similarly, in the case of formant synthesis we are limited because the user input specification corresponding to the parametric representation of the system is highly complex and overwhelming for a human user, even if we refer to an “expert” user. We realise that building a waveform “from scratch” is a very demanding task that leads to intelligible but not so natural-sounding output. A technique attempting to get around these limitations is a data-driven approach; concatenative speech synthesis. By concatenating small segments of pre-recorded human speech the final waveform is produced. Generally, this method gives the most natural results, but sometimes due to unhandled variations in human speech it can result in audible defects.

### 2.2.1 *Diphone synthesis*

A diphone is defined as the smallest segment of sound that contains the coarticulation information for human voice production. Indeed, a diphone is the segment that goes from the stable part of a phoneme to the stable part of the next one. In diphone synthesis a minimal human speech database is used that contains all the diphones that can occur in a given language. In the diphone database, only one occurrence of every diphone is present. Besides, in most advanced approaches is labeled with pitch-marks indicating the pitch-periods. In fact, such databases are speaker dependent, with sizes around a few megabytes of recorded data [26]. At runtime, the targeted output is created by concatenating the diphones from the database. For the diphone concatenation different signal processing techniques are employed,

such as TD-PSOLA [27] or MBROLA [9]. In Figure 2.2 we see how we move from phones, to diphones, and then how these diphones are concatenated to generate the target, in this case “hello”.

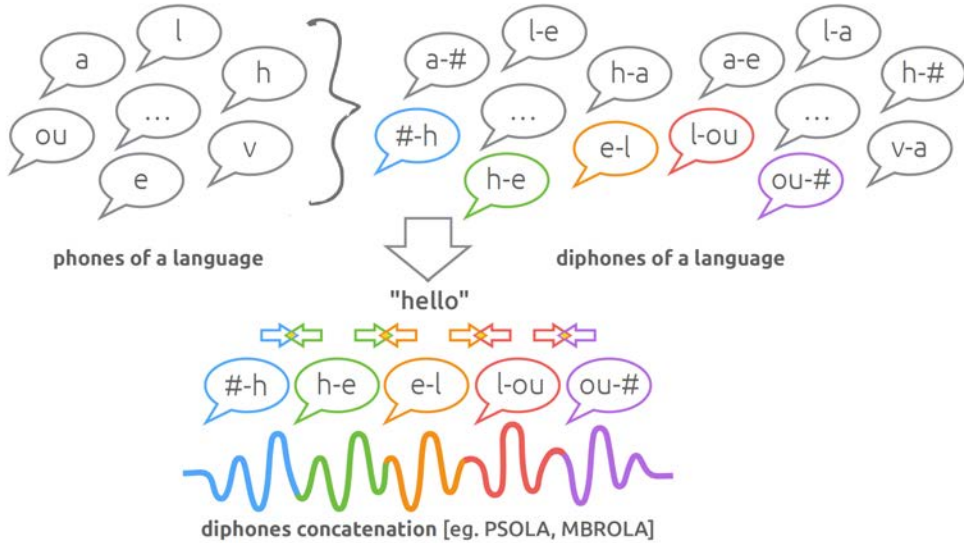


Figure 2.2: How diphones are concatenated to create the target word “hello” [33].

In the 90’s we have application examples such as the “CNET PSOLA” synthesiser, one of the most promising methods for concatenation synthesis at the time, which resulted in a commercial product called “ProVerbe TTS system”. MBR-PSOLA is a method like PSOLA, aiming at improving the quality of the concatenation with spectral methods. MBROLA is the software resulting from this research that its target was “to obtain a set of high-quality speech synthesisers for as many languages as possible, free for use in non-commercial applications as well as to boost up academic research on speech synthesis, and particularly on prosody generation” [28]. Also early version of “Festival Speech Synthesis System” [8] and later versions of “Infovox 330” were based on diphone concatenation of pre-recorded samples of speech. Singing synthesisers, such as “Lyricos” [29] and early versions of “Vocaloid” [30] were based on this speech synthesis approach. However, the disadvantage of diphone-based synthesis is that the quality of the final output is degraded compared to other concatenative synthesis approaches. Additionally, the synthetic speech sounds robotic and without expression.

### 2.2.2 Unit selection synthesis

Unit selection synthesis uses very large databases of pre-recorded human speech. The bigger the database is, the better the coarticulation is captured, which improves the naturalness of the synthetic voice produced. Additionally, the quality of output derives directly from the quality of recordings. Often the database can contain dozens of hours of recorded speech [31], [32]. Actually unit selection synthesis is very similar to diphone synthesis, but instead of using a speech database containing one occurrence of every diphone in a targeted language, this database is enriched with other phonetic units, e. g. triphones, syllables, etc. In brief, during the creation of the database, every utterance is segmented into various levels, such as half-phones, individual phones, diphones, syllables, morphemes, words, phrases, even full sentences. After the segmentation of the database, the units are indexed based on acoustic parameters like the fundamental frequency, duration, position in the syllable, and neighbouring phones. In order to produce the targeted phonetic output at run time, the best candidate chain of units is found by a compromise between a target cost (distance between target and unit) and a concatenation cost (distance between consecutive units) [33]. Figure 2.3 gives a graphical illustration of how diphones, syllables, words or any other linguistic unit are concatenated to create the target phrase.

In the 90's "ATR v-talk" [34], "CHATR" [35] and later versions of "Festive Speech Synthesis System" were the first systems synthesising speech based on the unit selection approach. Since then, unit selection techniques have evolved to become the dominant approach to speech synthesis. From the aspect of singing synthesis, a high-quality singing system, based on unit selection, was created by Meron [36]. It used a MIDI keyboard and supported subtle musical gestures such as vibrato or consonant/vowel articulation. In 2000, followed "Vocaloid" [30], a concatenative singing synthesiser. "Vocaloid" is actually able to produce realistic singing voices incorporating also expressions like vibrato. "CataRT" (Real-Time Corpus-Based Concatenative Synthesis) [37] used a large database of source sounds and is applied for high level instrument synthesis, artistic speech synthesis, as well as interactive exploration of synthesis in general.

The unit selection method provides the greatest intelligibility and naturalness, and sometimes if the output is tuned, it is even indistinguishable from real human speech. On the other hand, to achieve high naturalness, there is

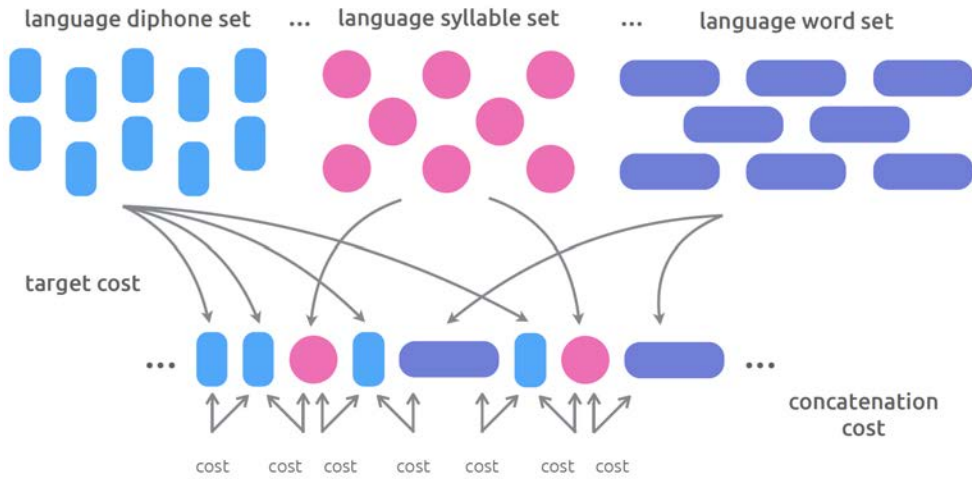


Figure 2.3: Graphical illustration of how diphones, syllables, words or any other linguistic unit are concatenated to create the target phrase.

need for very large databases with high and constant quality of the recordings. However, recording large high-quality databases with sufficient phone variations is very difficult and costly [38]. Also, unit selection algorithms function as black boxes, where any parameter manipulation degrades the quality of the output. This results in non-parametrisable and totally content-oriented and speaker-dependent speech production without any possible on-the-fly influence and low-level controls from the user.

### 2.3 STATISTICAL PARAMETRIC SYNTHESIS

For the past decade, another data-driven approach, called statistical parametric speech synthesis, has grown in popularity in the speech synthesis domain. Statistical parametric speech synthesis does not use real speech samples at synthesis time in direct contrast to concatenative synthesis. Instead it uses the knowledge contained in the pre-recorded speech database. The database is analysed and various production features are extracted. Then the extracted features are used to train a statistical model, and the targeted speech waveforms are generated from the models themselves based on the maximum likelihood criterion [3]. This results in a system with small synthesiser footprints, it gives more flexible control of synthesis methods combined with

good speech quality, and possibilities for portable applications. Moreover, since it is a statistical method, mathematically well defined, it introduces more flexible control separately over the spectrum, fundamental frequency and duration of the artificial speech. For example, model adaptation (mimicking voices) [39], model interpolation (mixing voices) [40] or eigenvoices (producing voices) [41] techniques allow the manipulation of voice characteristics, speaking styles and emotions [3]. On the other hand, the fact that there are no real speech samples to be used during the synthesis time introduces degradations in the final speech quality compared to the unit selection approach, but not as much as in diphone or formant-based synthesis methods.

During the last years, the HMM-based speech synthesis system (HTS) [4] has become a dominant tool in the speech synthesis domain, and many research teams have proposed various methods in order to improve the output quality and increase the controllability on different aspects of the generated output [2], [3], [4]. Actually, HTS is the first complete system for statistical parametric speech synthesis using hidden Markov models (HMMs) as statistical model. Extended description of the HTS system is given in [Chapter 3](#). Briefly, in HTS the pre-recorded speech database is analysed and production features, such as fundamental frequency, spectrum, duration of the phonemes, as well as first and second time derivatives of these features are extracted. These features are modelled simultaneously by HMMs for each phoneme. At synthesis time, the HMM models of each phoneme in the target sentence are concatenated, and synthetic speech spectral, pitch, and duration trajectories are generated from HMMs themselves based on a maximum likelihood criterion, regarding a given target. [Figure 2.4](#) illustrates the basic architecture of HMM-based speech synthesis system [2], which consists of two main parts, training and synthesis.

## 2.4 DISCUSSION

In this Chapter we present briefly the history of the speech synthesis domain. We detailed the main synthesis approaches, i. e. techniques based on vocal tract models, unit concatenation or statistical representation of speech, along with several application examples, dating from the XVIII<sup>th</sup> century until today. [Figure 2.5](#) illustrates the most representative examples and techniques in speech synthesis. We present the advantages and drawbacks of ev-

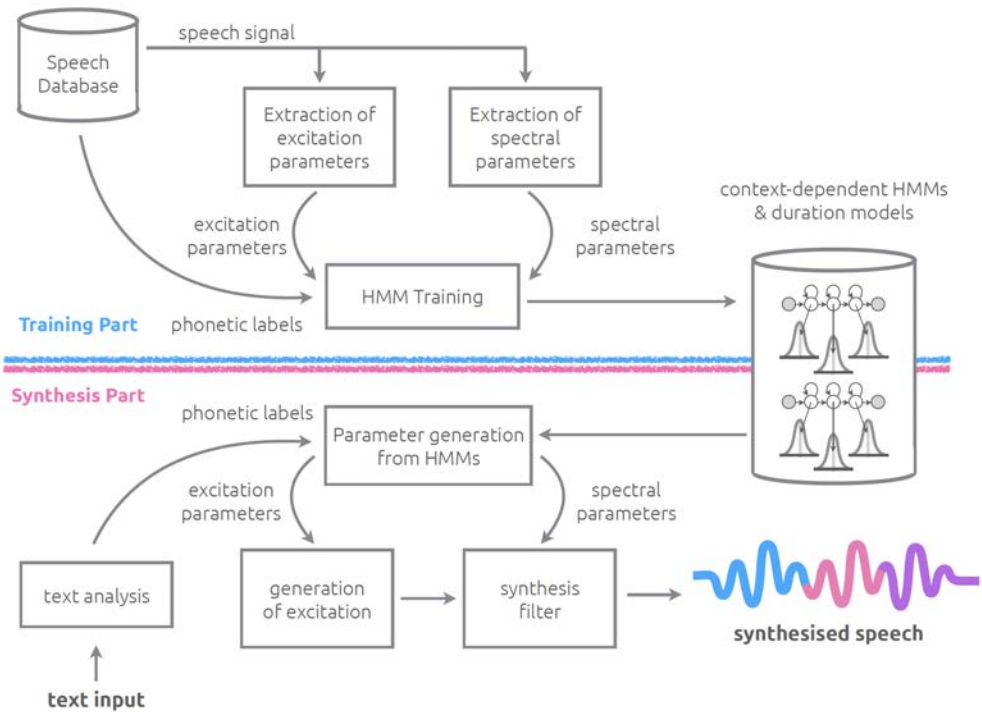


Figure 2.4: Block diagram of the HMM-based speech synthesis system [2]. Top (training part): features are extracted from speech database and used for training of models. Bottom (synthesis part): models are used to generate synthesis parameters corresponding to input phonetic labels. These parameters are then used to synthesise speech samples.

ery approach in order to demonstrate that the lack of interactivity in speech synthesis can be better tackled based on the statistical parametric techniques.

Actually, we are interested in comparing two characteristics in every approach, the final speech quality and the interactive controls. We want a system that is able to provide reactive controls over every production level of the artificial speech combined with high speech quality. However, one may notice the lack of such a system. On the one hand we have systems that are or can be interactive (e. g. “The Voder” or “OVE”), but the output quality is not very natural (i. e. vocal tract model synthesis). Although, the control parameters provide a “physical” and meaningful interpretation to the user, building a waveform “from scratch” is a demanding task, requiring in most cases long

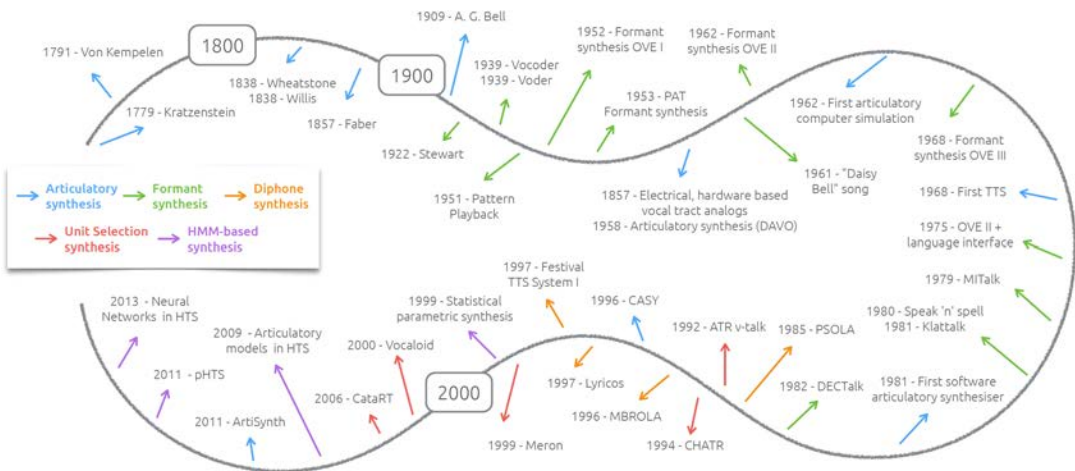


Figure 2.5: Some of the most representative examples and techniques in speech synthesis domain.

training periods. On the other hand, we have systems, mainly based on concatenative approaches, that have good output quality but lack in flexibility and controllability (e. g. "Vocaloid"). Such systems have "natural" sounding output but they demand the creation and use of large databases (i. e. expensive and large memory capacities are required), they are not so flexible and in most cases require offline processing. In-between these two cases, there is the statistical parametric synthesis, where the output quality is good. It is a flexible system requiring small computation and memory capacity. Nonetheless it is also based on an offline approach, not enabling realtime controls.

Ideally, we would like to have a system as reactive as the formant-based system but preserving the quality of the unit selection approach. Compared to other systems, the main advantage of HTS is its output quality combined with its flexibility, even if it is not initially built to be embedded into reactive designs. Interpolation techniques enable us to synthesise speech with various voice characteristics, speaking styles, and emotions that were not included in the natural speech database used for training our system and by applying multiple regression we can further control these voice characteristics. Additionally, HTS has a small number of tuning parameters; it is based on statistical principles and it uses the source filter representation of speech, providing the flexibility to control and modify the spectrum, funda-

mental frequency and duration of speech separately. Furthermore, a small amount of training data is enough to create statistical parametric speech synthesis systems, which leads to a very small footprint. HTS has also a memory-efficient, low-delay speech parameter generation algorithm and a computationally efficient speech synthesis filter.

All these characteristics, make HTS suitable not only for static embedded applications and mobile devices, such as Text-To-Speech applications but also potentially for realtime performative speech and singing synthesis, vocal expressivity and interaction. We want to bring HTS into reactive designs and explore all the above features in a reactive framework. We aim at building interactive application that involve and engage users and can be used for gaming, performances and of course further research.



# 3

## STATISTICAL PARAMETRIC SYNTHESIS

---

### Contents

---

3.1	Hidden Markov Models . . . . .	28
3.2	HMM-based speech synthesis . . . . .	30
3.2.1	Speech parameter generation . . . . .	31
3.2.2	Incorporating dynamic features . . . . .	32
3.2.3	Training . . . . .	35
3.2.4	Synthesis . . . . .	36
3.2.5	Advantages . . . . .	38
	Adaptation . . . . .	38
	Interpolation . . . . .	39
	Eigenvoices . . . . .	41
	Multiple regression . . . . .	41
	Multilingual support . . . . .	43
	Other . . . . .	43
3.2.6	Drawbacks . . . . .	44
3.3	HMM-based synthesis in other fields . . . . .	44
3.3.1	HMM-based singing synthesis . . . . .	45
3.3.2	Audio-visual laughter synthesis . . . . .	45
3.3.3	Stylistic gait synthesis . . . . .	46
3.4	Discussion . . . . .	47

---

One of the most important means of human communication is speech. We see that people have been interested in producing speech in ways other than the human vocal apparatus. A great number of efforts to artificially produce speech can be found in the literature, taking advantage of the best available technology at the time. Historically there are mechanical, electrical and of course computer-based approaches, all aiming at generating natural, intelligible and expressive speech. Techniques such as unit selection and waveform concatenation, have been shown to be able to generate intelligible and natural sounding speech. This is mainly due to the increasing computational

and memory capacities of the computers as well as the increasing availability of large speech databases. However, since these systems are based on large, speaker- and style-dependent databases, it is not easy to control voice characteristics, prosodic features, speaking styles and emotional expressions. Also, building a database containing all the desired features for a speaker is costly and time-demanding. It would be more realistic that the required speech data is as little as possible and easy to obtain, in order to create a new voice, emotion or style.

The need for higher control over speech variations and smaller database footprint led in the late 90's to another data-driven approach called *statistical parametric speech synthesis* [42]. In statistical parametric speech synthesis, a time-series stochastic generative model is used in order to model several speech parameters. When a *hidden Markov model (HMM)* is used as a generative model, then this approach is called *HMM-based speech synthesis*. HMM-based speech synthesis is one of the major approaches used in speech synthesis research and development in the past decade. Briefly, in this method, linguistic specifications and acoustic parameters are used to train the HMMs, which are then used to generate spectral and excitation parameters based on given linguistic context and drive a vocoder in order to generate a speech waveform. Since no real speech samples are used during synthesis, the output quality is lower than the one of unit selection. However, the quality of HMM-based synthetic speech has been improving [43], [44] and many control techniques have been proposed [39], [45].

Since our research is based on the HMM-based speech synthesis, this chapter gives a more detailed overview of the method. In [Section 3.1](#) we detail the statistical model - HMM - used in HMM-based speech synthesis approach. In [Section 3.2](#) an extended description of the system used in this study is presented, while [Section 3.3](#) presents briefly the application of the HMM-based synthesis framework in fields beyond speech. Finally, [Section 3.4](#) discusses the presented method.

### 3.1 HIDDEN MARKOV MODELS

In HMM-based speech synthesis, a HMM is used as a statistical time series model. A hidden Markov model is a finite state machine which generates a sequence of discrete time observations. It changes states at each time

unit (i. e. here frame), according to state transition probability distributions, and then according to the current state's output probability distribution an observation  $\mathbf{o}_t$  is generated at time  $t$ . By definition, an  $N$ -state HMM is characterised by the state transition probability matrix  $\mathbf{A} = \{a_{ij}\}_{i,j=1}^N$ , the output probability distributions  $\mathbf{B} = \{b_i\}_{i=1}^N$ , and initial state probabilities  $\mathbf{\Pi} = \{\pi_i\}_{i=1}^N$ . Hence, the parameters of an HMM  $\lambda$  are denoted as

$$\lambda = (\mathbf{A}, \mathbf{B}, \mathbf{\Pi}) \quad (3.1)$$

$$\lambda = (\{a_{ij}\}_{i,j=1}^N, \{b_i\}_{i=1}^N, \{\pi_i\}_{i=1}^N) \quad (3.2)$$

Figure 3.1 illustrates a typical structural example of a 3-state left-to-right HMM. In this HMM topology, the state index increases or remains the same for every time unit. Generally, such HMMs (i. e. typically 5-state left-to-right) are used to model speech parameter sequences since they are suitable to model time changing signals, such as speech.

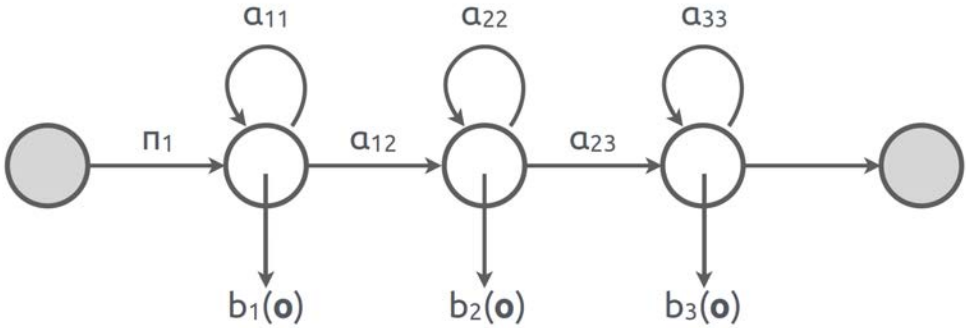


Figure 3.1: A typical structural example of a 3-state left-to-right HMM.

For simplicity the output probability distribution  $b_i(\mathbf{o})$  of the observational data  $\mathbf{o}$  of state  $i$  is assumed to be a single multivariate Gaussian distribution and thus defined as

$$b_i(\mathbf{o}_t) = \mathcal{N}(\mathbf{o}_t; \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i) \quad (3.3)$$

where  $\boldsymbol{\mu}_i$  and  $\boldsymbol{\Sigma}_i$  are the mean vector and covariance matrix respectively.  $\mathbf{o}_t$  is an observation vector of the speech parameters of frame  $t$ .

Let  $\mathbf{q} = (q_1, q_2, \dots, q_T)$  be a state sequence and  $\mathbf{O} = (\mathbf{o}_1, \mathbf{o}_2, \dots, \mathbf{o}_T)$  be an observation sequence, both of length  $T$ . Then, given the HMM  $\lambda$ ,  $P(\mathbf{O}|\mathbf{q}, \lambda)$  can be calculated by multiplying the output probabilities of each state, as

$$P(\mathbf{O}|\mathbf{q}, \lambda) = \prod_{t=1}^T P(\mathbf{o}_t|q_t, \lambda) = \prod_{t=1}^T b_{q_t}(\mathbf{o}_t) \quad (3.4)$$

Given an observation sequence  $\mathbf{O} = (\mathbf{o}_1, \mathbf{o}_2, \dots, \mathbf{o}_T)$  it is interesting<sup>1</sup> to find the best state sequence  $\hat{\mathbf{q}} = (\hat{q}_1, \hat{q}_2, \dots, \hat{q}_T)$ . For instance, by using the joint probability of the observation sequence and the most likely state sequence  $P(\mathbf{O}, \hat{\mathbf{q}}|\lambda)$  the real probability  $P(\mathbf{O}|\lambda)$  can be approximated.

$$P(\mathbf{O}|\lambda) = \sum_{\mathbf{q}} P(\mathbf{O}, \mathbf{q}|\lambda) \simeq \max_{\mathbf{q}} P(\mathbf{O}, \mathbf{q}|\lambda) \quad (3.5)$$

It is difficult to determine  $\hat{\lambda}$  which globally maximises likelihood  $P(\mathbf{O}|\lambda)$  for a given observation sequence  $\mathbf{O}$  in a closed form. Actually, this problem is an optimisation problem from incomplete data including the hidden variable  $\mathbf{q}$  and there is no known way to analytically solve it. The model parameter set can be found by satisfying a certain optimisation criterion such as maximum likelihood (ML) as follows:

$$\hat{\lambda} = \operatorname{argmax}_{\lambda} P(\mathbf{O}|\lambda) = \operatorname{argmax}_{\lambda} \sum_{\mathbf{q}} P(\mathbf{O}, \mathbf{q}|\lambda) \quad (3.6)$$

However, by using the expectation-maximisation (EM) algorithm, which conducts optimisation of the complete dataset, it is possible to obtain a model parameter set  $\lambda$  which locally maximises  $P(\mathbf{O}|\lambda)$ . This optimisation algorithm is often referred as the Baum-Welch algorithm.

### 3.2 HMM-BASED SPEECH SYNTHESIS

The basic concept of a typical statistical parametric speech synthesis system is straightforward. First given a speech database, we extract parametric representations of speech, such as spectral and excitation parameters  $\mathbf{O}$  corresponding to several linguistic specifications  $W$  (such as phoneme labels)

<sup>1</sup> For applications such as decoding, HMM parameter initialisation, etc.

[Section A.1], [Section A.2]. Then the extracted features are modelled by using a set of generative models (e. g. HMMs  $\lambda$ ). A maximum likelihood (ML) criterion is usually used to estimate the model parameters  $\hat{\lambda}$ , as in Equation 3.7. We then generate speech parameters  $\hat{\mathbf{o}}$ , as in Equation 3.8, for given linguistic specifications  $w$  to be synthesised, from the set of estimated models  $\hat{\lambda}$ , in order to maximise their output probabilities. Finally, a speech waveform is reconstructed based on the generated parametric representations of speech<sup>2</sup>.

Training:

$$\hat{\lambda} = \underset{\lambda}{\operatorname{argmax}} P(\mathbf{O}|\mathbf{W}, \lambda) \quad (3.7)$$

Synthesis:

$$\hat{\mathbf{o}} = \underset{\mathbf{o}}{\operatorname{argmax}} P(\mathbf{o}|w, \hat{\lambda}) \quad (3.8)$$

### 3.2.1 Speech parameter generation

In order to directly generate the optimal speech parameter sequence from the HMMs by means of maximum likelihood, we need to obtain a speech parameter vector sequence  $\mathbf{o}$  of length  $T$

$$\mathbf{o} = [\mathbf{o}_1^T, \mathbf{o}_2^T, \dots, \mathbf{o}_T^T] \quad (3.9)$$

in such a way that

$$P(\mathbf{o}|\lambda) = \sum_{\mathbf{q}} P(\mathbf{o}, \mathbf{q}|\lambda) \quad (3.10)$$

is maximised with respect to  $\mathbf{o}$ , where

$$\mathbf{q} = \{q_1, q_2, \dots, q_T\} \quad (3.11)$$

This maximisation can be approximated by solving the following problems, as detailed in [46]:

<sup>2</sup> In statistical parametric speech synthesis any generative model can be used, however when using HMMs the method is well known as HMM-based speech synthesis

Condition 1. For given  $\lambda$  and  $\mathbf{q}$ , maximise  $P(\mathbf{o}|\mathbf{q}, \lambda)$  with respect to  $\mathbf{o}$

Condition 2. For given  $\lambda$ , maximise  $P(\mathbf{o}, \mathbf{q}|\lambda)$  with respect to  $\mathbf{o}$  and  $\mathbf{q}$

Condition 3. For given  $\lambda$ , maximise  $P(\mathbf{o}|\lambda)$  with respect to  $\mathbf{o}$

The maximisation of [Condition 1](#) can be approximated

$$\hat{\mathbf{o}} = \underset{\mathbf{o}}{\operatorname{argmax}} P(\mathbf{o}|\mathbf{w}, \hat{\lambda}) \quad (3.12)$$

$$= \underset{\mathbf{o}}{\operatorname{argmax}} \sum_{\mathbf{q}} P(\mathbf{o}, \mathbf{q}|\mathbf{w}, \hat{\lambda}) \quad (3.13)$$

$$\approx \underset{\mathbf{o}}{\operatorname{argmax}} \max_{\mathbf{q}} P(\mathbf{o}, \mathbf{q}|\mathbf{w}, \hat{\lambda}) \quad (3.14)$$

$$= \underset{\mathbf{o}}{\operatorname{argmax}} \max_{\mathbf{q}} P(\mathbf{q}|\mathbf{w}, \hat{\lambda}) \cdot P(\mathbf{o}|\mathbf{q}, \hat{\lambda}) \quad (3.15)$$

$$\approx \underset{\mathbf{o}}{\operatorname{argmax}} P(\mathbf{o}|\hat{\mathbf{q}}, \hat{\lambda}) \quad (3.16)$$

$$= \underset{\mathbf{o}}{\operatorname{argmax}} N(\mathbf{o}; \mathbf{M}, \mathbf{U}) \quad (3.17)$$

where  $\mathbf{M} = [\mu_{q_1}^T, \mu_{q_2}^T, \dots, \mu_{q_T}^T]$  and  $\mathbf{U} = \operatorname{diag}[U_{q_1}, U_{q_2}, \dots, U_{q_T}]$  are respectively the  $3M \times 1$  mean vector and  $3M \times 3M$  covariance matrix of  $\mathbf{q}$ , where  $M$  is the length of one observation vector  $\mathbf{o}_i$ . The state sequence  $\hat{\mathbf{q}}$ , is the best state sequence for a given linguistic specification

$$\hat{\mathbf{q}} = \underset{\mathbf{q}}{\operatorname{argmax}} P(\mathbf{q}|\mathbf{w}, \hat{\lambda}) \quad (3.18)$$

Note here that [Condition 2](#) and [Condition 3](#) algorithms in [46] respectively maximise [Equation 3.14](#) and [3.12](#)

### 3.2.2 Incorporating dynamic features

From [Equation 3.10](#) to [Equation 3.18](#), the generated parameter vector at frame  $t$ ,  $\hat{\mathbf{o}}_t$  is obtained independently of preceding and succeeding frames. Consequently the speech parameter sequence  $\mathbf{o}$  that maximises [Equation 3.10](#) can be determined simply as a sequence of mean vectors,  $\mu_i$ , of the optimal state  $\hat{\mathbf{q}}$ . However, the transition from one state to another will be abrupt, presenting a discontinuity. Such an assumption does not fit the properties of human speech, where variations are smooth and not so abrupt from frame to

frame. Actually this may cause discontinuities in the generated sequences of spectral parameters, and therefore degrade the quality of the speech output. In order to generate realistic speech parameter trajectories without voice discontinuities, the parameter generation algorithm incorporates relationships between static and dynamic features as constraints for the maximisation problem.

Let  $\mathbf{o}_t$  be a vector of length  $3M$  that consists of the static feature vector  $\mathbf{c}_t = [c_{t1}, c_{t2}, \dots, c_{tM}]^T$  (i. e. cepstral coefficients), of length  $M$ , and its dynamic feature vectors  $\Delta\mathbf{c}_t$  (i. e. delta or velocity of the cepstral coefficients) and  $\Delta^2\mathbf{c}_t$  (i. e. delta-delta or acceleration of the cepstral coefficients)

$$\mathbf{o}_t = [\mathbf{c}_t, \Delta\mathbf{c}_t, \Delta^2\mathbf{c}_t] \quad (3.19)$$

Similarly, for pitch we have the static feature vector  $\mathbf{p}_t$  (i. e. the value of pitch), and its dynamic feature vectors  $\Delta\mathbf{p}_t$  (e. g. delta or velocity of the pitch) and  $\Delta^2\mathbf{p}_t$  (i. e. delta-delta or acceleration of the pitch). Figure 3.2 illustrates an observation vector of each frame, containing static and dynamic features of both spectral and excitation parameters.

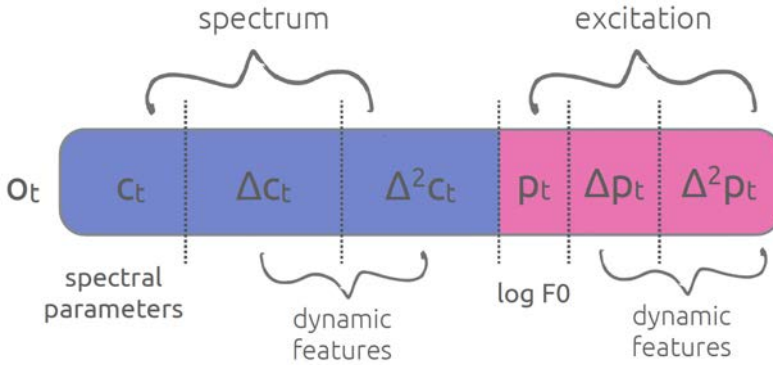


Figure 3.2: Example of an observation vector of each frame [2].

The dynamic feature vectors are calculated by linear combination of the static feature vectors using several frames around the current frame, as

$$\Delta\mathbf{c}_t = \sum_{\tau=-L_-^{(1)}}^{L_+^{(1)}} \mathbf{w}^{(1)}(\tau)\mathbf{c}_{t+\tau} \quad (3.20)$$

$$\Delta^2 \mathbf{c}_t = \sum_{\tau=-L_-^{(2)}}^{L_+^{(2)}} \mathbf{w}^{(2)}(\tau) \mathbf{c}_{t+\tau} \quad (3.21)$$

Then the relationship between  $\mathbf{o}_t$  and  $\mathbf{c}_t$  can be arranged in matrix form as

$$\mathbf{o} = \mathbf{W} \mathbf{c} \quad (3.22)$$

where  $\mathbf{c} = [\mathbf{c}_1^\top, \mathbf{c}_2^\top, \dots, \mathbf{c}_T^\top]^\top$  is a static feature sequence,  $\mathbf{W}$  is a matrix that appends dynamic features to  $\mathbf{c}$ , with

$$\mathbf{W} = [\mathbf{w}_1, \mathbf{w}_2, \mathbf{w}_3]^\top \quad (3.23)$$

$$\mathbf{w}_t = [\mathbf{w}_t^{(0)}, \mathbf{w}_t^{(1)}, \mathbf{w}_t^{(2)}] \quad (3.24)$$

$$\mathbf{w}_t^{(n)} = [\underbrace{0_{M \times M}, \dots, 0_{M \times M}}_{\text{1st}}, \underbrace{\mathbf{w}^{(n)}(-L_-^{(n)}) \mathbf{I}_{M \times M}, \dots, \mathbf{w}^{(n)}(0) \mathbf{I}_{M \times M}}_{(t-L_-^{(n)})\text{-th}}, \underbrace{\mathbf{w}^{(n)}(L_+^{(n)}) \mathbf{I}_{M \times M}}_{(t+L_+^{(n)})\text{-th}}, \underbrace{0_{M \times M}, \dots, 0_{M \times M}}_{\text{T-th}}]^\top \quad (3.25)$$

where  $L_-^{(0)} = L_+^{(0)} = 0$  and  $w_0^{(0)} = 1$ .  $0_{M \times M}$  and  $\mathbf{I}_{M \times M}$  are respectively  $M \times M$  zero and identity matrices.

Under the condition of [Equation 3.22](#) maximising  $P(\mathbf{o}|\mathbf{q}, \lambda)$  with respect to  $\mathbf{o}$  for a fixed state and mixture sequence  $\mathbf{q}$  is equivalent to that with respect to  $\mathbf{c}$ . By setting

$$\frac{\partial \log P(\mathbf{W} \mathbf{c} | \mathbf{q}, \lambda)}{\partial \mathbf{c}} = 0 \quad (3.26)$$

we obtain a set of equations:

$$\mathbf{W}^\top \mathbf{U}^{-1} \mathbf{W} \mathbf{c} = \mathbf{W}^\top \mathbf{U}^{-1} \mathbf{M}^\top \quad (3.27)$$

In order to have a direct solution of [Equation 3.27](#) we need  $O(T^3 \times M^3)$  operations since  $\mathbf{W}^\top \mathbf{U}^{-1} \mathbf{M}^\top$  is a  $TM \times TM$  matrix. However, by taking into account the structure of  $\mathbf{W}^\top \mathbf{U}^{-1} \mathbf{M}^\top$ , [Equation 3.27](#) can be solved by the Cholesky decomposition or the QR decomposition or by an algorithm derived in [46], which can operate in a time-recursive manner [47].

### 3.2.3 Training

As commented earlier and illustrated in [Figure 2.4](#), a basic HMM-based speech synthesis system consists in the training and the synthesis phase. The training part performs the maximum-likelihood estimation of the HMM parameters, as shown in [Equation 3.7](#). Although this process is very similar to that for speech recognition, there are several differences that need to be noted. First, we extract both spectrum, e. g. mel-cepstral coefficients [48] and their dynamic features and excitation parameters, e. g. F0 and its dynamic features simultaneously from a natural speech database. Typically in automatic speech recognition systems (ASR) only spectral parameters are modelled using continuous distributions. The extracted spectrum and excitation parameters are modelled by a set of multi-stream context-dependent HMMs [49]. Since F0 values are not defined in the “unvoiced” regions, we cannot apply the conventional discrete and continuous HMMs to F0 modelling. Many methods have been proposed for the F0 sequences modelling [50]. However HTS uses multi-space probability distributions [51]. A typical multi-space probability distribution for F0 patterns consists of a continuous distribution for voiced frames and a discrete distribution for unvoiced frames. It is important to highlight that in order to keep the synchronisation between spectral and excitation parameters, they are modelled simultaneously by separate streams in a multi-stream HMM [49]. In order to fully describe speech spectrum and excitation are not enough, phoneme durations also must be modelled. Typically in HMM-based speech synthesis, a semi-Markov structure is used in which the distribution of state duration (temporal structure) is approximated by a Gaussian distribution [49].

Another difference is that not only prosodic but also linguistic contexts are taken into account in addition to phonetic ones. HMM-based speech synthesis uses various linguistic contexts such as lexical stress, pitch accent, etc. for the context-dependent modelling of HMMs [52]. Spectral parameters are mainly affected by phoneme information but excitation and duration parameters may be affected by supra-segmental linguistic information. For example, the contexts used in the HTS English recipes [4] are detailed in [Section A.2](#).

By introducing prosodic and linguistic contexts in addition to phonetic ones, the contextual factors become too many in relation to the amount of speech data available. Indeed, as the desired contextual factors increase,

their combinations increase exponentially. Thus, given the limited amount of training data<sup>3</sup>, the context-dependent HMM parameters cannot be estimated with enough accuracy. In order to tackle this problem and be able to estimate model parameters more robustly, we apply state-tying techniques to cluster similar states and to tie model parameters among several context-dependent HMMs. As the spectral, excitation and duration parameters have different context dependencies, they are clustered separately by stream-dependent decision trees [42].

A phonetic decision tree is a binary tree [see Section A.4] in which a *yes/no* phonetic question [see Section A.5] is attached to each node. Initially, all states in a given item list are placed at the root node of a tree. Depending on each answer, the pool of states is successively split and this continues until we reach the leaf nodes. All states in the same leaf node are then tied. The question at each node is chosen, among a long list of all the possible questions, to (locally) maximise the likelihood of the training data given the final set of state tyings. Further references and detailed explanations can be found in [53].

### 3.2.4 *Synthesis*

Once the training phase is complete, the synthesis part is able to generate targeted speech parameters from the HMMs themselves and finally synthesise the corresponding waveforms, as shown in Figure 2.4. The synthesis part performs the maximisation of Equation 3.8, which is also the inverse operation to speech recognition. The text to be synthesised is first converted into a sequence of context-dependent labels. These context-dependent labels have the format presented in Section A.1 and contain the contextual factors described in Section A.2. They are used to navigate the context-dependent decision trees, as illustrated in Section A.4 and retrieve the context-dependent HMMs. Once retrieved, an utterance HMM is constructed by state concatenation. The duration of each state is determined to maximise its probability based on its state duration probability distribution. Then, the speech parameter generation algorithm generates the sequences of spectral and excitation parameters from the utterance HMM. There are several variants of the speech parameter

---

<sup>3</sup> It is practically impossible to prepare a set of speech data including all the possible combinations of these factors.

generation algorithm but typically the Case 1 [46], (here referred as [Condition 1](#) and detailed in [Section 3.2.1](#), [Equation 3.27](#)), is used. Finally, from the generated spectral and excitation parameters the corresponding speech waveform is synthesised by means of a speech synthesis filter, e.g. mel log spectrum approximation (MLSA) filter [54]. [Figure 3.3](#) illustrates a schematic representation of the HMM-based speech synthesis phase, when incorporating both static and dynamic feature constraints. This results in smooth rather than piece-wise parameter trajectories.

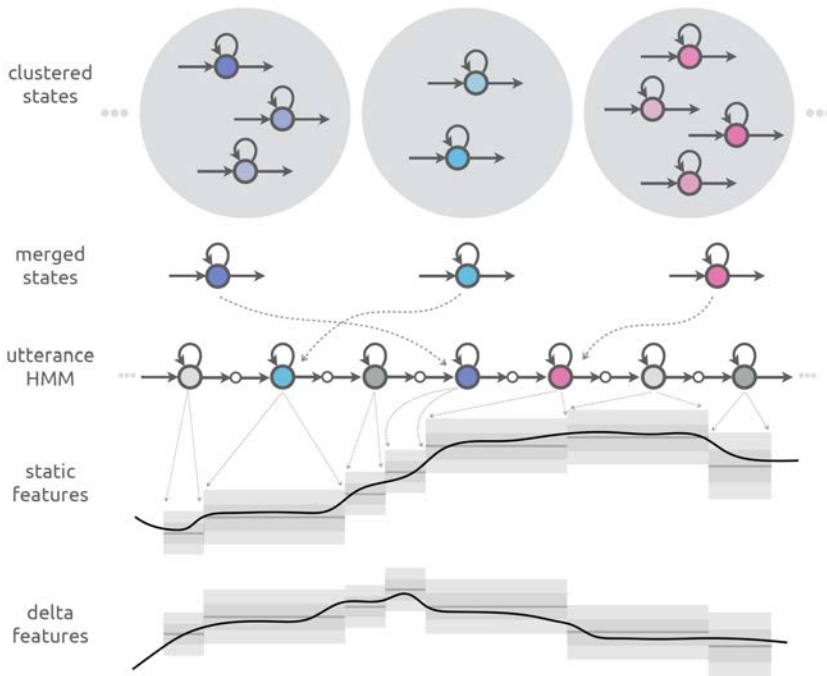


Figure 3.3: Overview of HMM-based speech synthesis scheme [3].

It is important to highlight here that the synthesis part is a realtime process. Indeed the parameter generation and the speech waveform synthesis are completed faster than the duration of the corresponding speech. However, since we need to create an utterance HMM, this means that the full text of the targeted utterance is required and consequently all its context-dependent labels. Therefore here, the smallest accessible unit is the utterance itself. In this work we target to reduce the accessible time to much smaller units, such as phonemes or even frames and provide realtime controls.

### 3.2.5 Advantages

Most of the advantages of statistical parametric synthesis against the other synthesis methods detailed in [Chapter 2](#) are related to its flexibility due to the statistical modelling process combined with intelligibility and naturalness. Due to its statistical nature, techniques such as adaptation, interpolation, multiple regression and multilingual support are mathematically defined. By applying these methods we can achieve control over voice characteristics, speaking styles and emotions. Here we detail these advantages.

#### *Adaptation*

Speaker adaptation is a technique that allows us to *mimic voices*. Actually, it enables us to transform existing speaker-independent acoustic models to match a target speaker using a very small amount of its speech data [39]. Speaker adaptation can be performed by adapting an “Average Voice Model” (AVM) to a specific target speaker or to a new speaking style or emotion. An AVM is a “canonical” speaker-independent HMM where inter-speaker acoustic variation is normalised using speaker-adaptive training (SAT) [39]. The model adaptation techniques, such as Maximum A Posteriori (MAP) [55] and Maximum Likelihood Linear Regression (MLLR) [56], come from the field of speech recognition.

MAP estimation takes advantage of the prior knowledge about the distributions of model parameters. Indeed, if before observing any adaptation data, we know what are the parameters of the model are likely to be we can use limited amount of adaptation data. The MAP estimate of an HMM,  $\lambda$ , is defined as the mode of the posterior distribution of  $\lambda$  as

$$\hat{\lambda} = \underset{\lambda}{\operatorname{argmax}}\{P(\lambda|\mathbf{O}, W)\} \quad (3.28)$$

$$= \underset{\lambda}{\operatorname{argmax}}\{P(\mathbf{O}, \lambda|W)\} \quad (3.29)$$

$$= \underset{\lambda}{\operatorname{argmax}}\{P(\mathbf{O}|W, \lambda) \cdot P(\lambda)\} \quad (3.30)$$

where  $P(\lambda)$  is the prior distribution of  $\lambda$ . However, in case of sparse adaptation data, since every Gaussian distribution is individually updated many model parameters may not be modified. As a result, during synthesis the

adapted characteristics (speaker identity, style, emotion, accent, etc.) alternate between the general and targeted voice within a single utterance.

Adaptation can also be accomplished by using MLLR, which is one of the most important recent developments in speech recognition since it can reduce acoustic mismatch between training and test data. Here, a set of linear transforms is used to map the existing AVM into the new targeted model, so that the adapted model approximates the given adaptation data in the best way. However, since the amount of adaptation data is limited, a regression class tree is normally used to cluster the Gaussian components based on acoustic similarity and to share the same MLLR transform. The state-output distributions (i. e. spectral, excitation and duration parameters) of the adapted model set are obtained as

$$\mathbf{b}_i(\mathbf{o}_t) = \mathcal{N}(\mathbf{o}_t; \hat{\boldsymbol{\mu}}_i, \hat{\boldsymbol{\Sigma}}_i) \quad (3.31)$$

$$\hat{\boldsymbol{\mu}}_i = \mathbf{A}_{r(i)} \boldsymbol{\mu}_i + \mathbf{b}_{r(i)} \quad (3.32)$$

$$\hat{\boldsymbol{\Sigma}}_i = \mathbf{H}_{r(i)}^T \boldsymbol{\Sigma}_i \mathbf{H}_{r(i)} \quad (3.33)$$

where  $\hat{\boldsymbol{\mu}}_i$  and  $\hat{\boldsymbol{\Sigma}}_i$  are the linearly transformed mean vector and covariance matrix of the  $i^{\text{th}}$  state-output distribution.  $\mathbf{A}_{r(i)}$ ,  $\mathbf{H}_{r(i)}$  and  $\mathbf{b}_{r(i)}$  are the mean linear-transformation matrix, the covariance linear-transformation matrix, and the mean bias vector for the  $r(i)^{\text{th}}$  regression class respectively. If the same transform matrices are used for both the mean vectors ( $\mathbf{A}$ ) and the covariance matrices ( $\mathbf{H}$ ) of the state output probability, then the method is called constrained MLLR (CMLLR) or feature-space MLLR.

Model adaptation is a very attractive feature of HMM-based speech synthesis, since with small amounts of training data new voice models can be built. Prior to the development of these techniques a completely new voice model had to be built from scratch. Such a task requires hours of annotated speech recordings from a speaker, while with adaptation methods some minutes of recordings are enough to create a new speaking style, identity, accent, emotion, etc.

### *Interpolation*

The model interpolation technique enables us to *mix voice characteristics* of existing model sets. Indeed we can generate synthetic speech having intermediate voice characteristics among two or more representative pre-trained

stylistic models. The generated speech from model interpolation has a certain style initially not contained in the database. The idea of model interpolation was first proposed to voice conversion and then applied to HMM-based speech synthesis, where HMM parameters are interpolated among some representative HMM sets [40]. Figure 3.4 illustrates the general idea of the interpolation technique. This technique allows us to synthesise speech with various voice characteristics [40], speaking styles [57], dialects [58], and emotions not included in the training data.

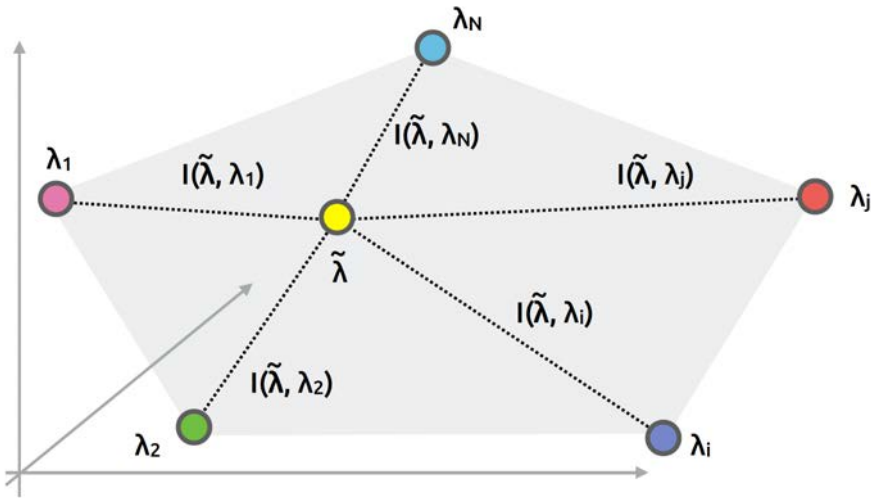


Figure 3.4: Stylistic space (e. g. speaker identity, accent, emotion, identity, etc.) modelled by HMM sets  $\{\lambda_i\}$ . Here,  $\{I(\tilde{\lambda}, \lambda_i)\}$  denotes interpolation ratio [3].

As it has been shown in [40], it is possible to synthesise speech with intermediate voice characteristics of two speakers by simply interpolating the models of these two speakers. It also proposes three interpolation methods:

- interpolation between observations;
- interpolation between output distributions of HMM states taking account of state occupancies;
- interpolation based on Kullback information measure;

However, in this work we adopted the second method, and further details can be found in Chapter 5. It is important to highlight here that this is a

mathematical interpretation of the interpolated combinations. Even though we can create intermediate voice characteristics and styles given pre-trained models and the new model is indeed not included in the initial database, the actual speech might have different real-word interpretations or realisation.

### *Eigenvoices*

We see that we can mimic (*adaptation*) or mix (*interpolation*) voice characteristics, speaking styles, or emotions. However, a problem rises when no adaptation data are available or determining the desired interpolation ratio in order to obtain the required voice is difficult. This problem was addressed by Shichiri et al. by applying the *eigenvoice technique* based on principal component analysis (PCA) to HMM-based speech synthesis [41]. The advantage of the eigenvoice approach is that it reduces the number of parameters to be controlled. By controlling only that reduced set of parameters it becomes easier to manually control the voice characteristics of synthesised speech by setting the weights. Further details can be found in [3], [41].

### *Multiple regression*

Multiple regression technique is used to intuitively *control voice characteristics*. Miyanaga et al. first applied a multiple-regression approach to HMM-based speech synthesis [59]. In [59] mean vectors of state-output distributions are directly controlled with small-dimensional auxiliary features.

$$\mu_i = \mathbf{M}_i \xi, \quad \xi = [1, z^T]^T \quad (3.34)$$

where  $\mathbf{M}_i$  is a multiple-regression matrix and  $z$  is an  $N$ -dimensional control vector,  $z = [z_1, \dots, z_N]^T$ .  $\mathbf{M}_i$  can be estimated to maximise the likelihood of the model for the training data. Each element of  $z$  captures specific voice characteristics, speaking styles, or emotions and in [59] these auxiliary features are manually annotated through subjective listening tests prior to HMM training. By specifying a control vector representing a point in the stylistic space we can create a voice with the required characteristics. The idea of multiple-regression technique is illustrated in Figure 3.5, where each coordinate represents a specific emotion in the emotional space and thus we can browse the emotions in the synthesised speech. Note here, that this is just an example, the stylistic space can be emotions, speaking styles, brightness, etc.

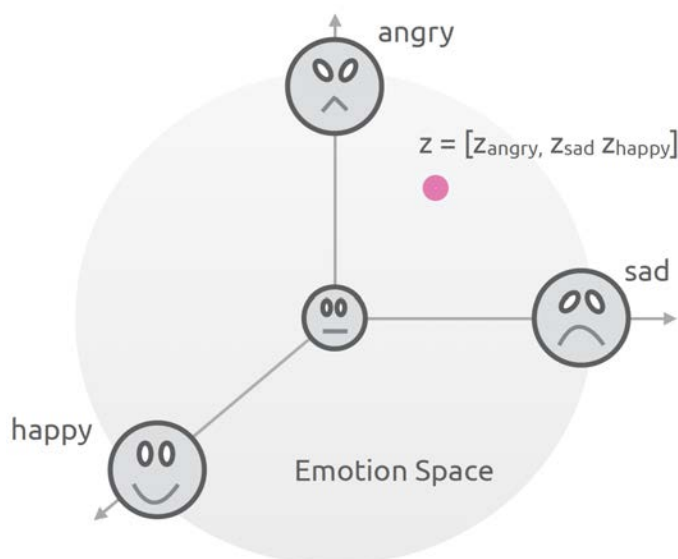


Figure 3.5: Overview of multiple-regression HMM-based emotion-controlling technique. Each coordinate of  $z$  represents a specific emotion, in this case *angry*, *sad*, *happy* [3].

This technique was successfully applied to control articulatory features in the context of HMM-based speech synthesis [60], [61]. In [60], [61] the type of auxiliary feature are the articulatory features, which actually describe the continuous movements of a group of speech articulators, i. e. upper lip, lower lip, jaw, tongue (dorsum, body and tip) as illustrated in Figure 6.2. The movement of the articulators is recorded using human articulography techniques such as electromagnetic articulography. Such an HMM-based speech synthesis system allows articulatory control of the synthesised speech, providing a more meaningful and tangible interpretation of speech. It improves the naturalness of synthesised speech and broadens the flexibility of HMM-based speech synthesis, (e. g. simulate different accents of a language, approximate foreign loan words, etc.). More details about the articulatory control of HMM-based synthesis using multiple regression are given in Chapter 6.

Moreover, it is possible to combine the model adaptation approach with the regression approach, and thus synthesise speech in large stylistic spaces without the need of recording large speech databases [45].

### *Multilingual support*

Since in the framework of HMM-based speech synthesis the only language-dependent element is the set of contextual factors, it is easy to have multilingual support. Indeed, once the contextual factors required for a target language are acquired, then the HMMs can be automatically trained. Such a key property has allowed the development for more than 40 different language systems, by both academic and commercial organisations. A list can be found in [3]. Within the framework of statistical parametric synthesis, it is possible to use a multilingual acoustic model from an existing synthesiser in one language and cross adapt models to the target language based on a very small set of collected sentences. By allowing multiple speakers and multiple languages to be combined into single models, multilingual synthesisers can be easily built [62].

### *Other*

Other advantages of the HMM-based speech synthesis system are its small footprint and its robustness. Since the parameters stored are the HMMs themselves, instead of the actual speech waveforms, its footprint is significantly small when compared to that of a typical unit-selection system. Without using any compression techniques, the typical size of a standard system built is normally less than 2 MB. Thus, such system can easily be used in mobile and embedded devices, where storage capacities can be an issue. Moreover, statistical parametric speech synthesis is more robust than unit-selection synthesis, since it can overcome phonetically unbalanced data, or noise and fluctuations due to the recording conditions as shown in [63].

Another very interesting feature of this method is that it can employ a number of useful technologies developed for HMM-based speech recognition. It provides a unified front-end (text analysis) and back-end (waveform generation) modules for the text-to-speech system, while keeping them independent. Furthermore, HMM-based speech synthesis provides fewer tuning parameters when compared to the unit-selection synthesis, since it is based on mathematically well-defined statistical principles. Finally, since it uses the source-filter representation of speech, the spectrum, excitation, and duration can be modelled, controlled and modified separately, providing high degrees of flexibility.

### 3.2.6 *Drawbacks*

Although HMM-based speech synthesis system is adaptable, controllable, robust, flexible and with small footprint, its major drawback is the quality of synthesised speech, especially when compared against unit-selection. This quality degradation is due to three main reasons: vocoders, acoustic modelling accuracy, and over-smoothing. Since a mel-cepstral vocoder with simple periodic pulse-train or white-noise excitation is used, the final speech sounds “buzzy”. To tackle this problem, many high-quality vocoders have been employed across the literature [2].

The spectral representation of speech has also been contemplated [2]. Since the speech parameters are directly generated from acoustic models, it is important to model accurately both the acoustic and duration properties of speech. The accuracy of the modelling directly affects the quality of synthesised speech. The more precise statistical models we use the better the quality of synthesised speech will be. Another way to improve the model accuracy is by increasing the number of parameters. However, higher-complexity control may result in over-fitting to the training data. Finally, using a different model topology than the current five-state left-to-right structure could improve the speech quality as well [3].

## 3.3 HMM-BASED SYNTHESIS IN OTHER FIELDS

Although hidden Markov models have been known already for decades, nowadays they are still in a state of development. We see that they have a great variety of applications [64], in diverse fields, such as speech recognition and synthesis, medicine, neurosciences, computational biology, bioinformatics, seismology, environment protection and engineering. In the specific case of the HMM-based speech synthesis framework we have seen how HMMs are used to generate speech. However, due to the generative properties of HMMs and the flexibility that the actual HMM-based speech synthesis system provides, many examples beyond speech have been developed. Here we briefly present examples of singing synthesis [65], audio-visual laughter synthesis [66] and stylistic gait synthesis [67] based on HMMs.

### 3.3.1 *HMM-based singing synthesis*

As detailed in [Section 3.2.5](#), statistical parametric speech synthesis can support multiple languages. By assuming that singing is like any other language, we can construct a singing synthesiser. However, in order to define it, it is necessary to introduce linguistic factors such as musical notes and lyrics [65]. For example, the contextual information used for a singing voice synthesis system can be found in [Section A.3](#). More specifically, in order to construct a basic singing system, the initial speech database has to be replaced with a database of singing and the corresponding musical notes. By adding the extra contextual information a singing synthesis system can be constructed similarly to the initial HMM-based text-to-speech synthesis system. An example of an HMM-based singing speech synthesiser is “Sinsy” [65].

### 3.3.2 *Audio-visual laughter synthesis*

Another example of using the the HMM-based speech synthesis framework in a context different than speech is presented in [66] and [68], where Urbain and Cakmak respectively present the audio and visual HMM-based laughter synthesis. As explained before, HMM modelling being known for its flexibility and its ability, it can be used for the synthesis of non-verbal vocalisations, such as laughter.

Traditional HMM-based speech synthesisers use pulse train or white noise, during voiced and unvoiced segments respectively. However, here in order to reduce buzziness in the synthesised vocalised segments of the waveforms, the authors used STRAIGHT [69] for the feature extraction and the Deterministic plus Stochastic Model (DSM) [70] source model for synthesis. As with singing, an audio laughter database was recorded and a set of contextual factors had to be defined. Several other approaches of HMM-based laughter synthesis have been implemented [71], and evaluated, but the best obtained results are when STRAIGHT is used for the feature extraction, DSM for synthesis with the duration predicted by HTS.

Furthermore, the speaker-dependent training of HMMs can be applied not only to audio but also visual laughter synthesis. Audio and visual laughter are synthesised as separate modalities with a forced durations and then

combined together to render audio-visual laughter on a 3D avatar<sup>4</sup>. For that purpose, database designed for laughter synthesis is used: the AV-LASYN database [72]. The AV-LASYN database contains synchronous audio-visual laughter recordings from one male subject. Both audio and visual models were obtained separately, with a process similar to the one used in speech. However it is important to note here that motion data have higher dimensionality (i. e. here 105-dimensional vector are used to represent the overall face motion for a given frame) than speech. Experiments show that the presented visual synthesis appears to be plausible to participants. However the audio defects have an important impact on the perception of quality. More details can be found also in [Section 7.2.1].

### 3.3.3 Stylistic gait synthesis

Tilmanne in [67] presents an expressive gait synthesis system based on hidden Markov models, that actually follows a modified process that was originally developed for speaking style adaptation. In [67] a large database of stylistic walk sequences, the Mockey database [73], was recorded. In this database, one actor performed several expressive walking styles. 11 different arbitrarily chosen styles were acted: proud, decided, sad, top-model, drunk, cool, afraid, tiptoeing, heavy, in a hurry, manly. Figure 3.6 presents four example postures arbitrary chosen from the Mockey database.

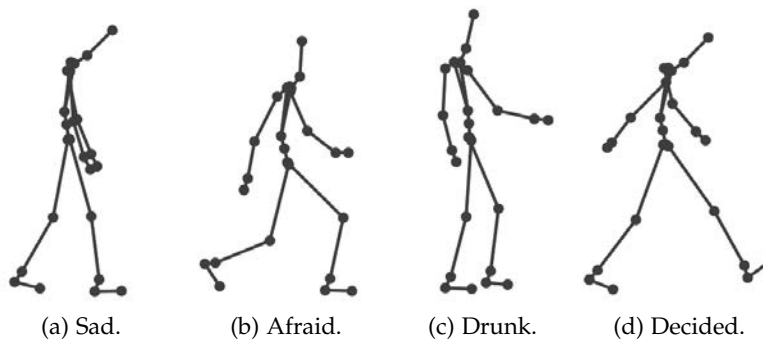


Figure 3.6: Four example postures taken from the motion capture database.

<sup>4</sup> AV-LASYN : AudioVisual Laughter Synthesis Examples (visited June 2014): <http://tcts.fpms.ac.be/~cakmak/personal/>

Additionally to these gait styles a neutral motion walk sequences was also recorded and then used to train an HMM of average walk. Then, this model was used to adapt to a particular stylistic gait sequence by using only a small amount of training data from the target style. Indeed, by starting from a robust neutral walk modelling it was possible to synthesise stylistic walks using only a few data. This method produces very convincing artificial walk sequences where the stylistic gaits can be recognised and look natural to the evaluators<sup>5</sup>. More details can be found also in [Section 7.2.2].

It is important to highlight that, although speech and motion fields present strong similarities, like inter-subject variability, stylistic or temporal variations, they are also very different. For example motion data have a much higher dimensionality (i. e. 54 exponential map parameters describing the skeleton pose at given frame) and cannot be always represented by a finite number of phonemes. Thus, the standard HTS framework was adapted for motion modelling (i. e. gait more specifically), while still taking into account the global variance of the data, in order to avoid excessive smoothing.

### 3.4 DISCUSSION

In this Chapter we present briefly the theory behind the hidden Markov models (HMMs), the generative model used in the context of HMM-based speech synthesis. Then we give an extended description of the system itself. First how the speech parameters are being generated and then why dynamic features are incorporated. We detail the training and synthesis part of the system and emphasise more on its advantages. Especially interpolation and multiple regression are two of the main advantages that we will use further in this work. We also present three applications of the HMM-based synthesis that are not directly linked to speech itself, namely singing, audio-visual laughter synthesis and stylistic gait synthesis.

Due to the adaptability and flexibility of HMM-based speech synthesis approach can have several real-life and scientific application. This may include personalised speech-to-speech translation systems, voice banking and reconstruction, creation of unique personalised voices, movie dubbing, linguistics, cell-phones, smartphones, in-car navigation systems, call centres and so on. Moreover, the gait and audio-visual laughter syntheses broaden the set of

---

<sup>5</sup> Examples of synthesised walks (visited June 2014): <http://tcts.fpms.ac.be/~tilmanne/>

application in other fields such as gaming and avatar control. Further, new TTS (but not only) applications will be seen in the very near future.

However, there is still much to do in statistical parametric synthesis and there are many future directions that should be examined. Although the operation, adaptability and controllability of statistical parametric synthesis has been drastically improved, the output is still not natural. Fortunately, as indicated in [2], there are many ideas that have yet to be fully explored and still many more to be conceived. One idea that aims at improving the statistical parametric synthesis comes with introducing user interaction and reactive designs in the HMM-based synthesis framework, and will be presented in the following chapters.

## Part II

### PERFORMATIVE HMM-BASED SYNTHESIS

II	PERFORMATIVE HMM-BASED SYNTHESIS	49
4	REDUCED-CONTEXT HMM-BASED SPEECH SYNTHESIS	51
4.1	Short-term parameter generation . . . . .	52
4.2	Models trained with reduced phonetic context . . .	57
4.3	Evaluation . . . . .	59
4.4	Discussion . . . . .	74
5	REACTIVE AND CONTINUOUS MODEL INTERPOLATION	77
5.1	Reactive synthesis using model interpolation . . . .	79
5.2	Interactive modification of the degree of articulation	83
5.3	Interactive accent interpolation . . . . .	87
5.4	Discussion . . . . .	92
6	REACTIVE ARTICULATORY FEATURE GENERATION	95
6.1	Reactive synthesis using articulatory features . . .	96
6.2	Evaluation . . . . .	99
6.3	Reactive articulatory control . . . . .	107
6.4	Discussion . . . . .	109



# 4

## REDUCED-CONTEXT HMM-BASED SPEECH SYNTHESIS

---

### Contents

---

4.1	Short-term parameter generation . . . . .	52
4.2	Models trained with reduced phonetic context . . . . .	57
4.3	Evaluation . . . . .	59
4.3.1	Experimental protocol . . . . .	60
4.3.2	Objective evaluation . . . . .	61
	Case A & Case B . . . . .	62
	Case C & Case D . . . . .	65
	Varying-size sliding window . . . . .	67
4.3.3	Subjective evaluation . . . . .	70
	Case A & Case B . . . . .	71
	Case C & Case D . . . . .	73
4.4	Discussion . . . . .	74

---

As new trends in understanding expressivity in speech are being explored, there is a growing need for real-world speech synthesis applications such as encountered in entertainment and gaming, assistance for speech-impaired people or performing arts. However, one might notice from the literature [Chapter 2] that most of the research is based on a large time window (e. g. full sentence) and the overall available context. In our research we try to create a real solid platform for speech synthesis that is able to generate parameter trajectories adequate to react to very short-term user interaction. We call this property “performative”, in other words we aim at building a performative speech synthesis system. Such a performative approach is missing in the context of HMM-based speech synthesis. The challenges of such a platform though are not only the reactive production of expressive speech and the increase of adaptability but also to decrease the latency of the speech synthesis and provide meaningful control mechanisms.

In the typical HMM-based speech synthesis approaches [2], although the actual computation of parameters and waveforms is realtime (i. e. speech is synthesised faster than its actual duration) a certain amount of text is required in advance to be processed and converted into speech as a whole target. Hence, during this text-to-speech conversion process it is hard to modify the output based on any external query (either from the environment or the user). This limitation prevents the system from adapting to any external solicitation within the sentence, as it is being synthesised, resulting in a less interactive system. Thus, we propose the generation of speech parameters within a smaller look-ahead window, with only few contextual information rather than the whole available text. This approach allows to infer speech outputs at various production levels and time scales (e. g. the reactive manipulation speaker identity or prosody). However, the requirements of such a system (e. g. thread policies, memory management and data buffering) are totally different from the original one in order to support user interaction.

In this Chapter we describe the research that led to the development of the actual core of the performative HMM-based speech synthesis system, called pHTS. [Section 4.1](#) details the proposed short-term parameter generation algorithm and [Section 4.2](#) presents our approach in combining this proposed algorithm with models trained on reduced linguistic context. Finally in [Section 4.3](#) we evaluate the proposed synthesis and model training approaches while in [Section 4.4](#) we discuss the obtained results.

#### 4.1 SHORT-TERM PARAMETER GENERATION

In order to build a performative synthesis system that is able to react to very short-term user interaction and enables us to influence the generated parameter trajectories in realtime, it is crucial to ensure a small time window for parameter generation and sound synthesis. In other words, it is essential to generate the parameter trajectories locally and faster than the duration of the corresponding sound (realtime computation) in such a fashion that they can be manipulated with the smallest possible delays (small time window), and thus enable user interaction.

However, as we saw in [Chapter 3](#) the original HTS system uses an “all-at-once” approach. Indeed, during synthesis a provided context-dependent phoneme sequence (label file) is used to retrieve all context-dependent HMMs

(one per phoneme), which are then concatenated, constructing an  $n$ -unit long HMM sequence. This utterance HMM is then used to generate sequences of spectral and excitation parameters, by maximising the probability of the whole speech parameter sequence, using Maximum Likelihood Parameter Generation algorithm (MLPG) [46]. Finally, the targeted speech output is reconstructed by using excitation generation and a speech synthesis filter, usually Mel Log Spectrum Approximation (MLSA) filter [48] with pulse-train or white-noise excitation. Consequently, in HTS even if this “all-at-once” approach guarantees smoothly generated trajectories, computed faster than their actual duration (realtime computation), it prevents these trajectories from being modified according to any external solicitation, and therefore the system cannot reach an acceptable level of interactivity. Actually, the smallest accessible time scale in HTS is the complete targeted utterance, which is sufficient in the context of standard TTS. However, having a sentence-by-sentence system just reduces the number of possible applications to use cases where no user interaction is required when the sentence is being produced, e. g. reading emails, audiobooks, GPS application, phone centres, public announcements, etc. All in all having a system with much smaller acceptable timescales can facilitate user interaction and enrich the application domain.

In direct contrast to HTS synthesis, here we propose a Short-Term Maximum Likelihood Parameter Generation algorithm (ST-MLPG), where instead of the whole available phoneme sequence (complete utterance) we empirically use a reduced observation window containing a maximum of 3 phonetic labels at any given time (i. e.  $windowSize = 3$ ). These labels are the current one and, if available, the two previous ones (i. e.  $P = 2$ ). As discussed in [Section 4.4](#), we avoid using the labels following the current one (i. e.  $F = 0$ ). More specifically, as the phonetic labels are being streamed one by one, and thus they are not all available at a given time, only the current phonetic label is used to concatenate the context-dependent HMMs and construct a small HMM sequence (i. e. maximum 3-HMMs long). Then this small HMM sequence, consisting only of the context-dependent HMMs of one phonetic label and, if available, the two previous ones is used to generate the corresponding sequences of spectral and excitation parameters. Then the targeted speech samples are generated. A consequence of the reduced observation window approach is that the generated speech parameter trajectories do not correspond to the overall maximum of probability (MLPG as in HTS - [System I](#) & [System III](#)), but to a local optimisation over a three pho-

netic label sliding window (ST-MLPG as in pHTS - [System II](#) & [System IV](#)). [Figure 4.1](#) shows a graphical comparison of HTS and pHTS, i.e. comparing parameter trajectory generation based on maximum likelihood solution over the complete phonetic label sequence (MLPG) and parameter trajectory generation based on local optimisation over a three phonetic label sliding window (ST-MLPG).

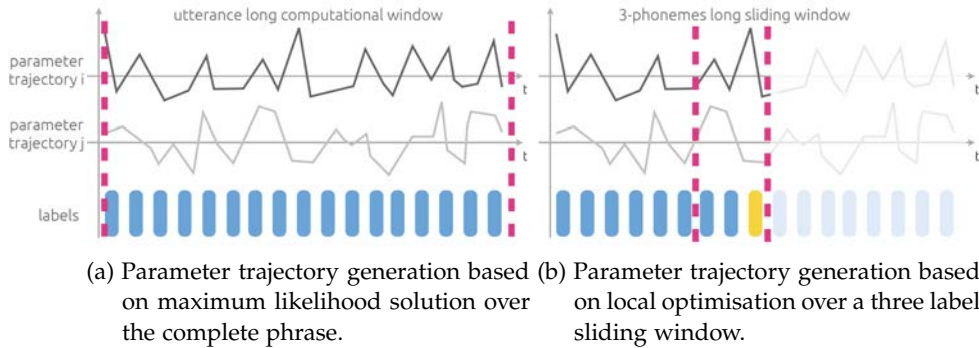


Figure 4.1: Parameter trajectory generation based on (a) MLPG in HTS and on (b) ST-MLPG in pHTS with the empirically defined sliding window. The information on the right of the sliding window is considered to be future information, thus not yet available for the computations.

The size of the sliding window was first empirically defined, so that it preserves the final speech quality with no dependencies on future context ( $F = 0$ ), as illustrated in [Figure 4.1](#). However, in [Section 4.3.2](#) it is shown to be a good compromise between reactivity and speech quality. Not depending on any succeeding phonetic labels allows reactive user influence over the generated speech parameters and samples, as discussed in [Section 4.4](#). Despite the fact that the sliding window can depend on any number of phonetic labels preceding and following the current one that fit best the synthesis purpose, increasing the window size consequently results in an increased computational complexity that may introduce latencies larger than accepted in interactive applications. It is also important to realise here that although the size of the sliding window is constant in terms of number of phonetic labels, the number of frames used for the parameter generation of the current label varies. However, any other arbitrary combination can be used, such as five preceding, one current and four succeeding phonetic la-

bels (i. e.  $\text{windowSize} = 10$  labels, with  $F = 4$  and  $P = 5$ ). Actually, since every label has a different number of frames given its duration, every time that the sliding window advances in order to compute the parameters for the “next” current label, the total number of frames within the window changes accordingly, i. e. varies in every iteration.

More specifically, based on the definitions presented in [Chapter 3](#), for a given continuous HMM  $\lambda$ , we derive an algorithm for determining the speech parameter vector sequence within a sliding window. First we define the number of frames ( $f_l$ ) used by the sliding window when processing the  $l^{\text{th}}$  label and thus the size of the computational matrices.

$$f_l = \sum_{i=l-P}^{l+F} \sum_{s=1}^N d_{i,s} \quad (4.1)$$

$$f_l = \sum_{i=l-P}^{l-1} \sum_{s=1}^N d_{i,s} + \sum_{s=1}^N d_{l,s} + \sum_{i=l+1}^{l+F} \sum_{s=1}^N d_{i,s} \quad (4.2)$$

$$f_l = \alpha_l + \beta_l + \gamma_l \quad (4.3)$$

where  $P$  and  $F$  are respectively the number of phonetic labels preceding and following the current one and  $N$  is the number of states in each phoneme HMM (generally 5 in standard HTS).  $d$  is the number of frames that correspond to a state of a label, i. e.  $d_{i,s}$  indicates the number of frames of  $i^{\text{th}}$  label in the  $s^{\text{th}}$  state.  $f_l$ ,  $\alpha_l$ ,  $\beta_l$  and  $\gamma_l$  are respectively the total number of frames in the sliding window, of the preceding labels, of the current label and of the succeeding labels while  $l$  is the index of the current label.

These restrictions over the number of frames must be incorporated in [Equation 3.9](#), [3.10](#), [3.11](#), [3.22](#) and [3.27](#), in order to determine the speech parameter vector sequence within a sliding window

$$\text{Equation 3.9} \Rightarrow \mathbf{o}_l = [\mathbf{o}_1^T, \mathbf{o}_2^T, \dots, \mathbf{o}_{f_l}^T] \quad (4.4)$$

so that

$$\text{Equation 3.10} \Rightarrow P(\mathbf{o}_l|\lambda) = \sum_{\mathbf{q}_l} P(\mathbf{o}_l, \mathbf{q}_l|\lambda) \quad (4.5)$$

is maximised with respect to  $\mathbf{o}_l$ , where

$$\text{Equation 3.11} \Rightarrow \mathbf{q}_l = \{q_1, q_2, \dots, q_{f_l}\} \quad (4.6)$$

Then, respectively maximising  $P(\mathbf{o}_l | \mathbf{q}_l, \lambda)$  with respect to  $\mathbf{o}_l$  for a certain state sequence  $\mathbf{q}_l$

$$\text{Equation 3.22} \Rightarrow \mathbf{o}_l = \mathbf{W}_l \mathbf{C}_l \quad (4.7)$$

Under this condition (Equation 4.7), maximising  $P(\mathbf{o}_l | \mathbf{q}_l, \lambda)$  with respect to  $\mathbf{o}_l$  we obtain a new set of equations:

$$\text{Equation 3.27} \Rightarrow \mathbf{W}_l^T \mathbf{U}_l^{-1} \mathbf{W}_l \mathbf{C}_l = \mathbf{W}_l^T \mathbf{U}_l^{-1} \mathbf{M}_l^T \quad (4.8)$$

$\mathbf{W}_l^T \mathbf{U}_l^{-1} \mathbf{W}_l$  is an  $f_l M \times f_l M$  matrix with  $M$  the dimension of an observation vector  $\mathbf{o}_i$ . Solving directly this set of equations has complexity of  $O(f_l^3 M^3)$ , that can be optimised to  $O(f_l M^3)$  using various decomposition methods. The obtained parameter sequence  $\mathbf{C}_l$  for this sliding window is then

$$\mathbf{C}_l = [c_1, c_2, \dots, c_{f_l}]^T \quad (4.9)$$

From all the  $\mathbf{C}_l$  obtained over the sliding window, we only keep the parameters that correspond to the frames of the current label

$$\mathbf{C}'_l = [c_{\alpha_l+1}, \dots, c_{\alpha_l+\beta_l}]^T \quad (4.10)$$

Ideally, in order to avoid any possible discontinuities between  $\mathbf{C}'_l$  and  $\mathbf{C}'_{l+1}$ , the  $c_{\alpha_l+\beta_l}$  coefficient must be the same as the  $c_{\alpha_{l+1}}$ . Technically, although these coefficients are computed with some models in common, the surrounding models used are slightly different (i. e. oldest and newest). This can result in either discontinuities (i. e. periodic distortions) and instabilities (i. e. saturation and glitches). In order to tackle this we used stabilisation of the generated coefficients and filter interpolation, respectively, as in standard HTS. Firstly, as shown in [48] the generated coefficients can be stabilised either by keeping the log approximation error not exceeding 0.24 dB or 0.2735 dB depending on  $\text{Pa}$ , the Padé approximation order used (i. e.  $\text{Pa} = 4$  and  $\text{Pa} = 5$  respectively) or by just stabilising the filter although the accuracy of log approximation is lost. Secondly, when using the now-stable coefficients to generate the actual samples we progressively interpolate from one filter

coefficient to the next so that we avoid any abrupt variation of the filter coefficients. This enables us to have smooth output without discontinuities.

Figure 4.2 gives a graphical representation of how the speech parameter vector sequence  $C'_l$  is determined within a sliding window containing in total  $f_l$  frames (Equation 4.3), where  $\alpha_l$ ,  $\beta_l$  and  $\gamma_l$  correspond to the number of frames of the preceding, current and succeeding labels respectively. It is important to highlight here that Figure 4.2 is a schematic representation and does not exactly reflect reality since the actual number of frames varies in every iteration but not in the Figure.

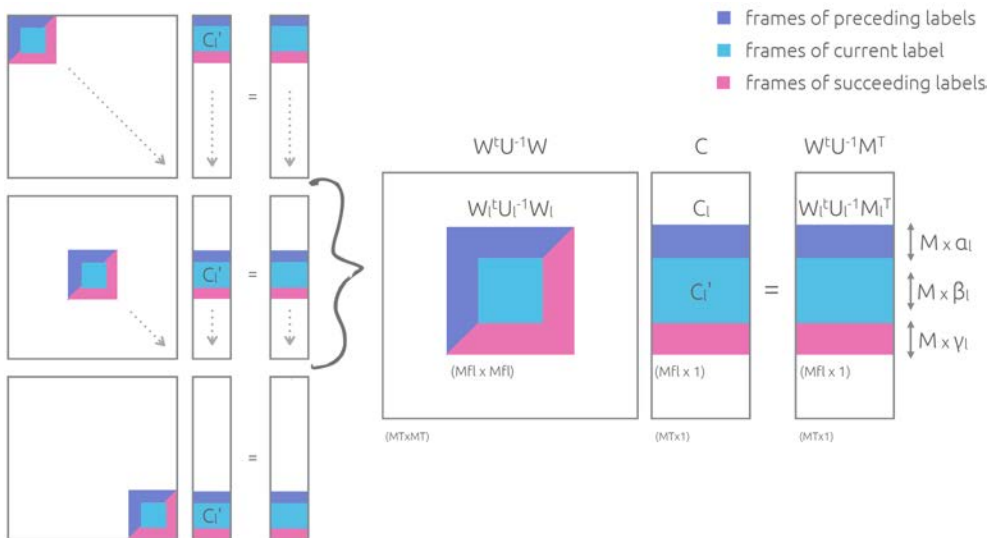


Figure 4.2: Determining the speech parameter vector sequence  $C'_l$  within a sliding window of  $f_l$  frames, where  $\alpha_l$ ,  $\beta_l$  and  $\gamma_l$  correspond to the number of frames of the preceding, current and succeeding labels respectively.

#### 4.2 MODELS TRAINED WITH REDUCED PHONETIC CONTEXT

The short-term parameter generation algorithm can also be considered as a way to reduce the context during synthesis. This approach enables interactive controls over the parameters being generated. Similarly, the same reduced-context principle can be applied while training the models used for synthesis (System III and System IV). Indeed, in HTS, labels are used to de-

scribe phonemes with all their contextual dependencies. The phonetic and acoustic realisation of each phoneme depends on its preceding/succeeding neighbours and on the larger segments in which this phoneme is contained (e. g. syllable, word, phrase, utterance) [74] building a dependency graph. Iteratively, for every larger segment, (i. e. syllable, word, phrase, utterance) a similar dependency graph is built. Along with these dependencies, various pieces of information are available: relative position of the current segment in the larger containing segment (e. g. relative position of the phoneme in the syllable), and the possible presence of an accent or a stress on the current and/or the surrounding segments. The complete dependency list is presented in [Section A.1](#).

However, since at synthesis time we assume that we have no future dependencies it is only fair to use models for which no succeeding dependencies are taken into account while training. In other words, in our attempt to discard any future linguistic dependencies, we reduce the phonetic context used at the training stage. The aim of retraining our models with a reduced set of linguistic factors, depending only on preceding and currently synthesised phonemes is to evaluate to what extent the future linguistic information is influencing the final speech quality, especially when combined with a short-term parameter generation algorithm. Thus, all the contextual information related to segments larger than the syllable, i. e. word and phrase, is discarded during the training phase. The amount of considered linguistic information is now limited only to preceding and current phonemes, as well as previous syllables. The whole database of context-dependent HMMs is then retrained in consequence. Note that for the new training we use the standard implementation of the HTS toolkit available in [4], with a reduced set of questions. [Figure 4.3](#) illustrates a graphical representation of the linguistic factors used during full and reduced-context model training.

It is important to realise that the number of linguistic factors taken into account during training, strongly depends on the targeted application. For example here, we use linguistic factors related only to preceding and current phonemes, as well as previous syllables. Such models are suitable to interactively build and control the phonetic context input [ABd<sup>+</sup>11a], [dTA<sup>+</sup>13], although not limited to. Any other design choice that includes different linguistic factors, i. e. any past related information, can be suitable. Of course this needs further investigation but it is not included here.

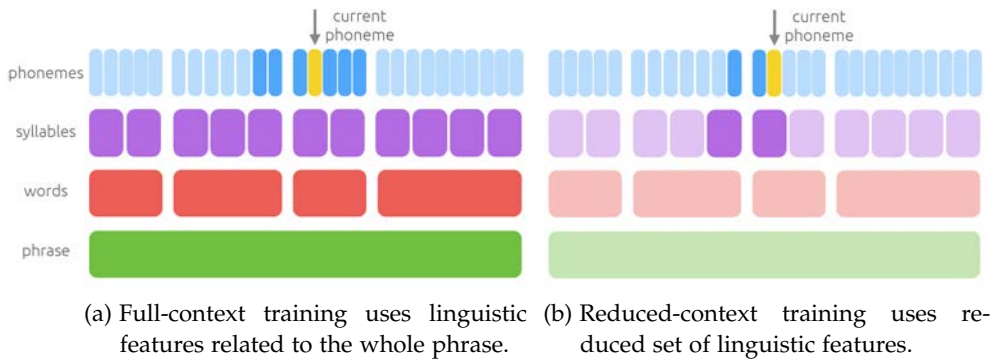


Figure 4.3: Linguistic factors used during full and reduced-context training.

### 4.3 EVALUATION

Both objective and subjective experiments were conducted in order to evaluate the similarities between trajectories generated using HTS (MLPG algorithm) and trajectories generated using pHTS (ST-MLPG algorithm). The two synthesis methods are compared while using models trained on either full ([System I](#) and [System II](#)) or reduced linguistic context ([System III](#) and [System IV](#)). In an attempt to assess the distortions introduced by the model training itself (i. e. using either full or reduced linguistic factors) both methods are employed and tested independently. In other words, trajectories generated by one method (either HTS or pHTS) when using full-context models are compared to those generated when using the same method and reduced-context models. Here, we present an extensive comparison by examining these four possible cases:

Case A. HTS vs. pHTS using full linguistic context models  
[\[System I vs. System II\]](#)

Case B. HTS vs. pHTS using reduced linguistic context models  
[\[System III vs. System IV\]](#)

Case C. HTS using full vs. reduced linguistic context models  
[\[System I vs. System III\]](#)

Case D. pHTS using full vs. reduced linguistic context models  
[\[System II vs. System IV\]](#)

A graphical illustration of all the examined [cases](#) of reduced and full-context regarding both training and synthesis is presented in [Figure 4.4](#)

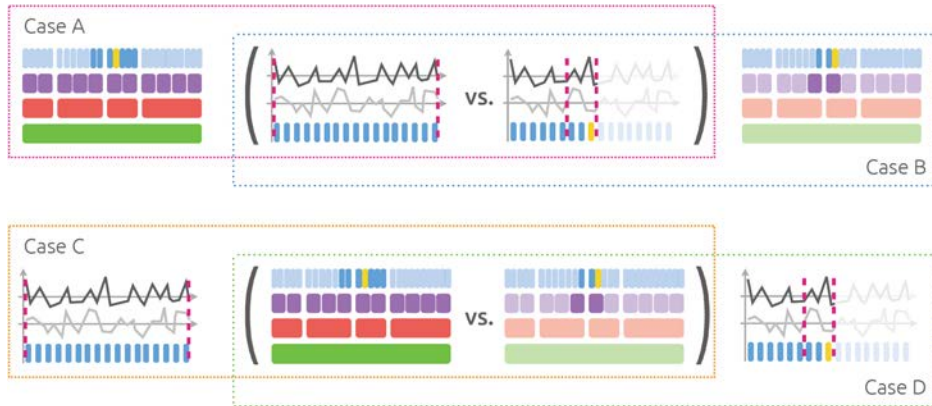


Figure 4.4: A simple diagram of all the examined [cases](#) of reduced and full-context regarding both training and synthesis.

Additionally to these four [cases](#) we conduct further objective tests where pHTS is used with different sliding window sizes. More specifically, instead of using the sliding window with the empirically defined size of two preceding ( $P = 2$ ), one current and no succeeding ( $F = 0$ ) phonetic label, we vary the number of preceding labels from 0 to 2 and succeeding labels from 0 to 1. We take into account all possible combinations of window sizes and compare the generate trajectories with those generated by HTS. Here we use only full-context models, similarly to [Case A](#), even though the same setup can be repeated with reduced-context models.

This experiment aims to find the best size for the sliding window used in pHTS, independently of the training of the models. It is also important to evaluate to what extent the proposed short-term parameter generation method is influenced by the length of the sliding window used, i.e. the number of phonetic labels preceding and following the current one. Note here that no subjective tests have been conducted for this case.

#### 4.3.1 *Experimental protocol*

For our tests, we use the speaker-dependent training demo in English that is provided in [4], with the AWB, BDL, JMK, RMS male speakers and the CLB,

SLT female speakers from the CMU ARCTIC database [31], [32]. All training sentences included in the demo are used to train our system, either with full or reduced linguistic context.

We synthesised 40 sentences based on the phonetic labels provided in the demo, which were not used for training. Speech signals were sampled at 48 kHz and the traditional Mel Generalized Cepstral (MGC) coefficients [75] (with  $\alpha = 0.55$ ,  $\gamma = 0$  and order of MGC analysis = 35) were obtained. Speech waveforms are synthesised with MLSA filtering [48]. Note that state durations vary between full and reduced-context models, since durations are modelled differently in each approach. The duration difference is small, less than 10%, mainly observed on the generated pauses. However, in order to objectively compare our results on a frame-by-frame basis, synthesis was forced to use the full-context phoneme durations, ensuring that all streams are synchronous and comparable. For the subjective evaluation the phoneme durations were modelled.

#### 4.3.2 Objective evaluation

In order to evaluate the distortion introduced by pHTS, using either full [Case A] or reduced-context models [Case B], as well as the distortion introduced by the reduced-context modelling itself in both synthesis systems [Case C and Case D], we make use of objective quality measurements by applying two different metrics, previously used in related research [76]. The first metric used is the *Mel-Cepstral Distortion* (Mel-CD), expressed in decibel (dB), a distance measure calculated between the target and the estimated mel-cepstrum.

*Mel-Cepstral Distortion* for one frame is defined by

$$\text{Mel-CD} = \frac{10}{\ln(10)} \sqrt{2 \sum_{d=1}^D (c_d^{\text{target}} - c_d^{\text{estimate}})^2} \quad (4.11)$$

where  $c_d$  are mel-cepstral coefficients generated by the two different systems, and  $D$  is the mel-cepstral coefficient order. Then we compute the mean *Mel-Cepstral Distortion* over all the frames available in the database.

The second metric used is the *Root-Mean-Square Error* of the fundamental frequency (F0-RMSE) expressed in Hertz (Hz).

*Root-Mean-Square Error* is defined by

$$F0 - RMSE = \sqrt{\frac{\sum_{t=1}^T (F0_t^{\text{target}} - F0_t^{\text{estimate}})^2}{T}} \quad (4.12)$$

where  $F0_t$  are the F0 values generated by the two different systems, and  $T$  is the total number of frames. Note that the F0-RMSE is computed for frames where both voice models are considered to be voiced, since F0 is not observed in unvoiced regions. Both metrics are used further in [Chapter 6](#).

#### *Case A & Case B*

[Table 4.1](#) and [4.2](#) show the *Mel-Cepstral Distortion* averaged over the *phoneme* and the *vowel* set respectively, over the 40 test sentences, for all, male and female speakers, in [Case A](#) and [Case B](#), with 95% confidence intervals. Similarly, [Table 4.3](#) and [4.4](#) show the *Root-Mean-Square Error* of the fundamental frequency. [Figure 4.5](#) and [4.6](#) gather all the obtained results for both *Mel-CD* and *F0-RMSE* respectively, in form of graphs among [Case A](#) and [Case B](#).

For [Case A](#) and [Case B](#) results indicate that the proposed pHTS system (ST-MLPG algorithm) lead to very similar segmental quality to the original one, independently of the linguistic context used for the training of the models. Indeed, less than 1 dB as spectral difference and less than 2 Hz of frequency difference, computed on both *phoneme* and *vowel* sets, is usually accepted as the difference limen for spectral transparency [77]. Although the results are encouraging, it is known that measuring distortion on the F0 trajectory (or any speech parameter trajectory) only gives a partial understanding of how the suprasegmental quality is affected by these training or synthesis modifications. Hence, this objective evaluation is completed with a set of subjective tests that validate [Case A](#) and [Case B](#) and obtain the user preferences as detailed in [Section 4.3.3](#).

Mel-CD [dB] (over <i>phoneme</i> set)	all speakers	male speakers	female speakers
Case A	$0.149 \pm 0.001$	$0.205 \pm 0.001$	$0.107 \pm 0.001$
Case B	$0.184 \pm 0.001$	$0.261 \pm 0.001$	$0.109 \pm 0.001$

Table 4.1: Mean *Mel-CD* (in dB) introduced by Case A and Case B computed over the *phoneme* set, with 95% confidence intervals.

Mel-CD [dB] (over <i>vowel</i> set)	all speakers	male speakers	female speakers
Case A	$0.272 \pm 0.002$	$0.288 \pm 0.002$	$0.240 \pm 0.003$
Case B	$0.576 \pm 0.005$	$0.678 \pm 0.008$	$0.371 \pm 0.005$

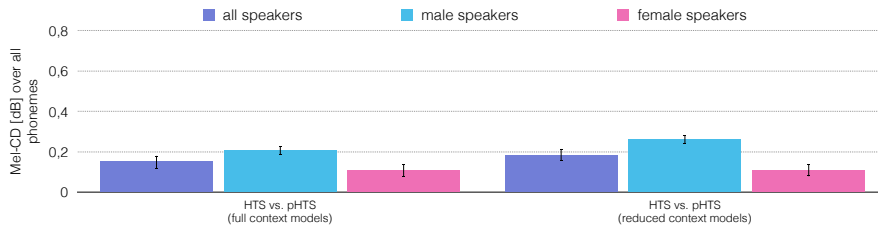
Table 4.2: Mean *Mel-CD* (in dB) introduced by Case A and Case B computed over the *vowel* set, with 95% confidence intervals.

F0-RMSE [Hz] (over <i>phoneme</i> set)	all speakers	male speakers	female speakers
Case A	$1.053 \pm 0.003$	$1.076 \pm 0.004$	$1.009 \pm 0.005$
Case B	$1.399 \pm 0.013$	$1.609 \pm 0.020$	$0.978 \pm 0.005$

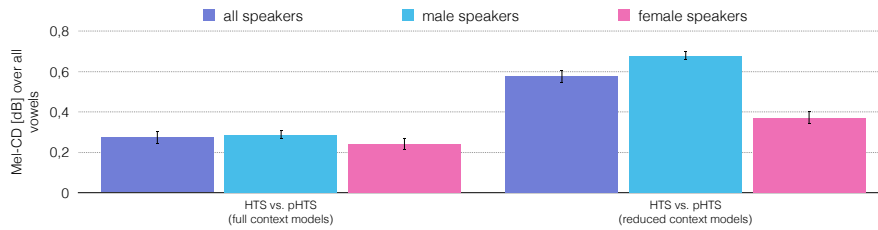
Table 4.3: *F0-RMSE* (in Hz) introduced by Case A and Case B computed over the *phoneme* set, with 95% confidence intervals.

F0-RMSE [Hz] (over <i>vowel</i> set)	all speakers	male speakers	female speakers
Case A	$0.981 \pm 0.007$	$0.993 \pm 0.009$	$0.956 \pm 0.013$
Case B	$0.823 \pm 0.059$	$0.947 \pm 0.087$	$0.676 \pm 0.035$

Table 4.4: *F0-RMSE* (in Hz) introduced by Case A and Case B computed over the *vowel* set, with 95% confidence intervals.

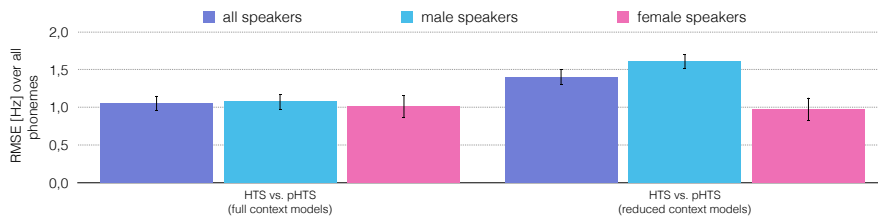


(a) Mean *Mel-CD* (in dB) computed over the *phoneme* set for Case A and Case B.

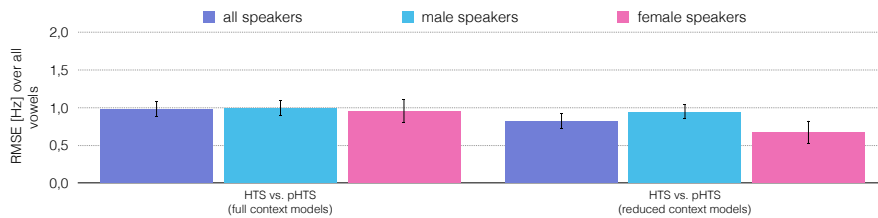


(b) Mean *Mel-CD* (in dB) computed over the *vowel* set for Case A and Case B.

Figure 4.5: Mean *Mel-Cepstral Distortion* introduced when comparing Case A and Case B, with 95% confidence intervals.



(a) *F0-RMSE* (in Hz) computed over the *phoneme* set for Case A and Case B.



(b) *F0-RMSE* (in Hz) computed over the *vowel* set for Case A and Case B.

Figure 4.6: *Root-Mean-Square Error* of the fundamental frequency (in Hz) introduced when comparing Case A and Case B, with 95% confidence interval.

### Case C & Case D

Table 4.5 and 4.6 show the *Mel-Cepstral Distortion* averaged over the *phoneme* and the *vowel* set respectively, over the 40 test sentences, for all, male and female speakers, in Case C and Case D, with 95% confidence intervals. Similarly, Table 4.7 and 4.8 show the *Root-Mean-Square Error* of the fundamental frequency. Figure 4.7 and 4.8 gather all the obtained results for both *Mel-CD* and *F0-RMSE* respectively, in form of graphs for better comparison among Case C and Case D.

<b>Mel-CD [dB] (over <i>phoneme</i> set)</b>	<b>all speakers</b>	<b>male speakers</b>	<b>female speakers</b>
Case C	$1.037 \pm 0.004$	$1.335 \pm 0.005$	$0.440 \pm 0.004$
Case D	$1.098 \pm 0.004$	$1.425 \pm 0.005$	$0.444 \pm 0.004$

Table 4.5: Mean *Mel-CD* (in dB) introduced by Case C and Case D computed over the *phoneme* set, with 95% confidence intervals.

<b>Mel-CD [dB] (over <i>vowel</i> set)</b>	<b>all speakers</b>	<b>male speakers</b>	<b>female speakers</b>
Case C	$2.021 \pm 0.007$	$1.917 \pm 0.009$	$2.230 \pm 0.014$
Case D	$2.363 \pm 0.009$	$2.330 \pm 0.011$	$2.428 \pm 0.015$

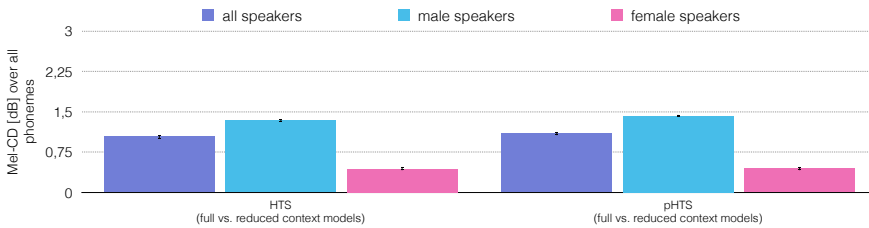
Table 4.6: Mean *Mel-CD* (in dB) introduced by Case C and Case D computed over the *vowel* set, with 95% confidence intervals.

<b>F0-RMSE [Hz] (over <i>phoneme</i> set)</b>	<b>all speakers</b>	<b>male speakers</b>	<b>female speakers</b>
Case C	$6.164 \pm 0.021$	$6.564 \pm 0.030$	$5.363 \pm 0.019$
Case D	$6.653 \pm 0.014$	$7.199 \pm 0.019$	$5.560 \pm 0.019$

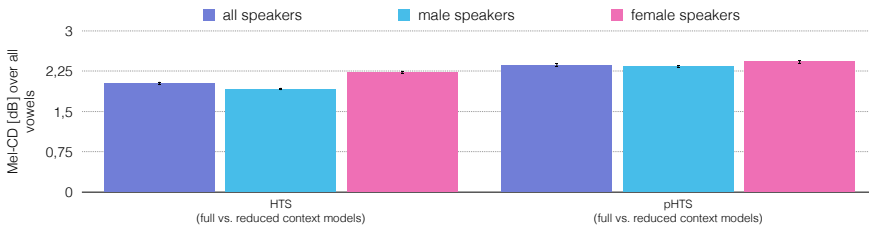
Table 4.7: *F0-RMSE* (in Hz) introduced by Case C and Case D computed over the *phoneme* set, with 95% confidence intervals.

<b>F0-RMSE [Hz] (over <i>vowel</i> set)</b>	<b>all speakers</b>	<b>male speakers</b>	<b>female speakers</b>
Case C	$9.230 \pm 0.005$	$12.637 \pm 0.007$	$7.418 \pm 0.007$
Case D	$9.962 \pm 0.060$	$13.320 \pm 0.090$	$8.245 \pm 0.020$

Table 4.8: *F0-RMSE* (in Hz) introduced by Case C and Case D computed over the *vowel* set, with 95% confidence intervals.



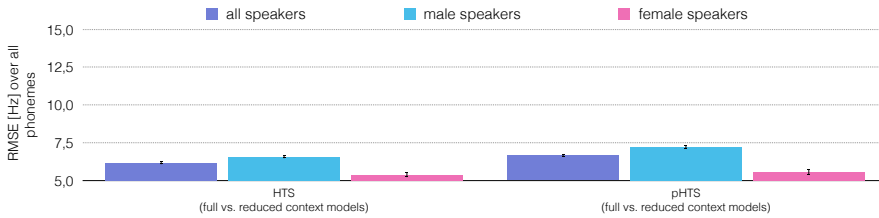
(a) Mean *Mel-CD* (in dB) computed over the *phoneme* set for Case C and Case D.



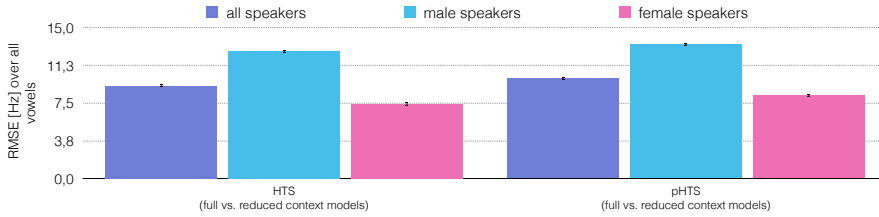
(b) Mean *Mel-CD* (in dB) computed over the *vowel* set for Case C and Case D.

Figure 4.7: Mean *Mel-Cepstral Distortion* introduced when comparing Case C and Case D, with 95% confidence intervals.

As could be expected, we see here that in direct contrast to Case A and Case B the biggest distortions are introduced by the linguistic context contained in the models used, independently of the method used for the speech parameter generation (Case C and Case D). Actually we observe about 2 dB of spectral distortion and about 9 Hz of fundamental frequency difference (i. e. about 6 Hz when computed on the *phoneme* set and about 10 Hz when computed on the *vowel* set). These objective evaluations are completed with a set of subjective tests that validate Case C and Case D and obtain the user preferences as detailed in Section 4.3.3.



(a) F0-RMSE (in Hz) computed over the *phoneme* set for Case C and Case D.



(b) F0-RMSE (in Hz) computed over the *vowel* set Case C and Case D.

Figure 4.8: Root-Mean-Square Error of the fundamental frequency (in Hz) introduced when comparing Case C and Case D, with 95% confidence interval.

### Varying-size sliding window

Furthermore, the same objective quality metrics, i. e. *Mel-Cepstral Distortion* and *Root-Mean-Square Error* of the fundamental frequency, were used in order to evaluate the distortion introduced by pHTS when compared to HTS (i. e. Case A) while using varying-size sliding window. Table 4.9 and 4.10 show the *Mel-CD* averaged over the *phoneme* set, over the 40 test sentences, for all speakers, male and speakers, when using a shifting window that includes the current and varying number of phonetic labels preceding and following the current one (i. e.  $0 \leq P \leq 2$  and  $0 \leq F \leq 1$ ), with 95% confidence intervals. Table 4.11 and 4.12, respectively, show the results obtained for F0-RMSE. In Figure 4.9 and 4.10 all the obtained results for *Mel-CD* and F0-RMSE respectively, are presented in form of graphs for better comparison among the different window sizes for all, male and female speakers.

From the graphical representation of the distortions introduced by the size of the sliding window (i. e. the number of phonetic labels preceding and following the current one taken into account for the parameter generation), we see that the highest distortions are introduced when only the current label

Mel-CD [dB] (over <i>phoneme</i> set)	all speakers	male speakers	female speakers
F = 0 & P = 0	0.364 ± 0.003	0.502 ± 0.004	0.188 ± 0.003
F = 0 & P = 1	0.150 ± 0.001	0.206 ± 0.002	0.137 ± 0.001
F = 0 & P = 2	0.149 ± 0.001	0.205 ± 0.001	0.107 ± 0.001

Table 4.9: Mean *Mel-CD* (in dB) introduced when comparing HTS to pHTS combined with shifting window including the current and varying number of preceding labels, computed over the *phoneme* set, with 95% confidence intervals.

Mel-CD [dB] (over <i>phoneme</i> set)	all speakers	male speakers	female speakers
F = 1 & P = 0	0.261 ± 0.003	0.365 ± 0.004	0.153 ± 0.003
F = 1 & P = 1	0.144 ± 0.001	0.166 ± 0.001	0.120 ± 0.001
F = 1 & P = 2	0.144 ± 0.001	0.165 ± 0.001	0.120 ± 0.001

Table 4.10: Mean *Mel-CD* (in dB) introduced when comparing HTS to pHTS combined with shifting window including one succeeding, the current and varying number of preceding labels, computed over the *phoneme* set, with 95% confidence intervals.

is used (i. e. `windowSize = 1`,  $F = 0$  and  $P = 0$ ). The lowest distortions are observed when two preceding, the current and one succeeding label is used (i. e. `windowSize = 4`,  $F = 1$  and  $P = 2$ ). In the first case we have spectral distortion of about 0.5 dB and less than 2 Hz of fundamental frequency difference. Respectively, for the second case we have almost no difference between the compared parameters trajectories. The empirically defined size of the sliding window, (i. e. `windowSize = 3`,  $F = 0$  and  $P = 2$ ), which is actually the previously described [Case A](#), introduces distortions in-between the best and worst presented cases. Specifically, for [Case A](#) there is about 0.2 dB

<b>F0-RMSE [Hz] (over <i>phoneme</i> set)</b>	<b>all speakers</b>	<b>male speakers</b>	<b>female speakers</b>
F = 0 & P = 0	1.676 ± 0.004	1.628 ± 0.005	1.772 ± 0.007
F = 0 & P = 1	1.065 ± 0.004	1.081 ± 0.005	1.031 ± 0.005
F = 0 & P = 2	1.053 ± 0.003	1.076 ± 0.004	1.009 ± 0.005

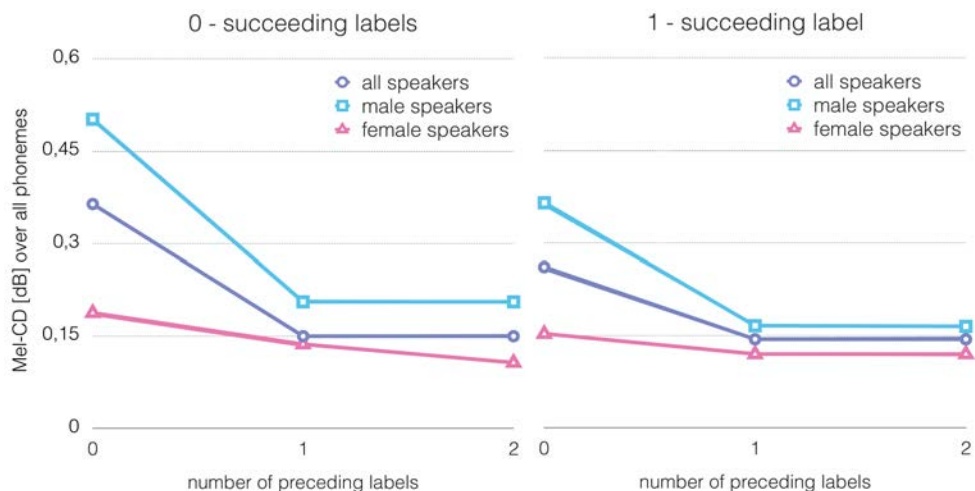
Table 4.11: *F0-RMSE* (in Hz) introduced when comparing HTS to pHTS combined with shifting window including the current and varying number of preceding labels, computed over the *phoneme* set, with 95% confidence intervals.

<b>F0-RMSE [Hz] (over <i>phoneme</i> set)</b>	<b>all speakers</b>	<b>male speakers</b>	<b>female speakers</b>
F = 1 & P = 0	1.264 ± 0.004	1.183 ± 0.004	1.427 ± 0.006
F = 1 & P = 1	0.169 ± 0.004	0.125 ± 0.001	0.256 ± 0.002
F = 1 & P = 2	0.104 ± 0.001	0.078 ± 0.001	0.155 ± 0.002

Table 4.12: *F0-RMSE* (in Hz) introduced when comparing HTS to pHTS combined with shifting window including one succeeding, the current and varying number of preceding labels, computed over the *phoneme* set, with 95% confidence intervals.

as spectral difference and about 1 Hz of frequency difference, which is usually accepted as the difference limen. More details about the interpretation of these results can be found in [Section 4.4](#).

It becomes evident from the acquired results that the distortions introduced by the size of the sliding window used in pHTS are rather small. Results also support the empirical definition of the sliding window size, while presenting marginal improvements when using succeeding contextual dependencies. Above we did not conduct any objective tests for all the possible window sizes, only for [Case A](#), as detailed in [Section 4.3.3](#).



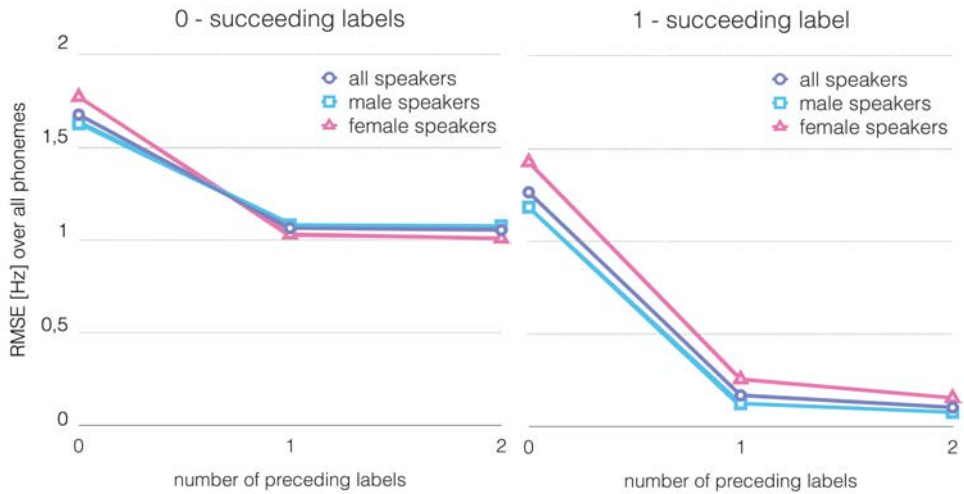
(a) Mean *Mel-CD* when using pHTS with a shifting window including current and preceding labels, computed over the *phoneme* set. (b) Mean *Mel-CD* when using pHTS with a shifting window including one succeeding, one current and preceding labels, computed over the *phoneme* set.

Figure 4.9: Mean *Mel-CD* introduced when comparing HTS to pHTS combined with varying-size shifting window, computed over the *phoneme* set.

### 4.3.3 Subjective evaluation

The subjective evaluation of HTS and pHTS along with the two training schemes is conducted via a Comparative Mean Opinion Score (CMOS) test, so that we can cross-evaluate the four test cases. Here listeners grade the perceived quality of a speech signal in relation to a reference speech signal. The CMOS scale is a 7-point scale ranging from  $-3$  for *much worse* to  $+3$  for *much better* with  $0$  for *about the same*.

A first group of 34 people participated to the test comparing HTS to pHTS using either full (Case A) or reduced (Case B) context model training. Further on, a second group of 59 people participated to the complementary test comparing the full to the reduced-context model training, using either HTS (Case C) or pHTS (Case D). Participants in both groups were speech and non-speech experts, native and non-native English speakers. Each test was composed of 24 randomly-chosen sentences, no longer than 8 seconds. Here



(a)  $F_0$ -RMSE when using pHTS with a shifting window including current and preceding labels, computed over the *phoneme* set. (b)  $F_0$ -RMSE when using pHTS with a shifting window including one succeeding, one current and preceding labels, computed over the *phoneme* set.

Figure 4.10:  $F_0$ -RMSE introduced when comparing HTS to pHTS combined with varying-size shifting window, computed over the *phoneme* set.

the durations were modelled and not forced, since frame-by-frame comparison is not needed. For each sentence, subjects were asked to listen to both versions (randomly shuffled) and to attribute a score according to their overall preference.

#### Case A & Case B

Table 4.13 shows the preference scores when comparing HTS to pHTS when using either full (Case A) or reduced (Case B) context models, as well as the overall preference of the used synthesis method. Here negative values indicate preference for the pHTS approach. Results are presented for all, male and female speakers, with 95% confidence intervals. In Figure 4.11 results are presented as graphs for easier comparison. It is important to keep in mind that we use a 7-point scale, ranging from  $-3$  to  $+3$ .

CMOS [7-scale] HTS vs. pHTS	all speakers	male speakers	female speakers
(overall)	$-0.046 \pm 0.058$	$-0.102 \pm 0.109$	$-0.018 \pm 0.068$
Case A	$-0.105 \pm 0.081$	$-0.151 \pm 0.149$	$-0.083 \pm 0.096$
Case B	$0.012 \pm 0.083$	$-0.053 \pm 0.159$	$0.045 \pm 0.097$

Table 4.13: Preference scores for all, male and female speakers when comparing HTS to pHTS while using either full (Case A) or reduced (Case B) context models, with 95% confidence intervals. Negative values indicate preference for the pHTS approach.

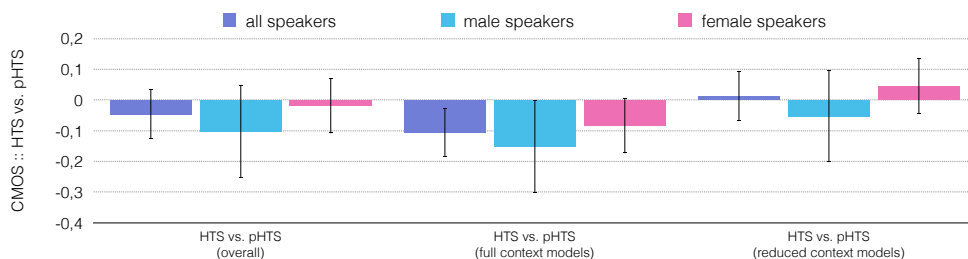


Figure 4.11: Preference scores for all, male and female speakers when comparing HTS to pHTS while using either full (Case A) or reduced (Case B) context models, with 95% confidence intervals. Negative values indicate preference for the pHTS approach.

Interestingly we observe that users tend to prefer the performative approach in HMM-based speech synthesis over the original one, either when using models trained on full or reduced-context. However, with a closer look, it is evident that the obtained scores are very close to 0, i. e. *about the same* and the 0 value often falls within the confidence interval. Probably this effect is due to random choices made by users when the two tested sentences were too similar. In other words, the introduced distortions are not observed by the listeners and thus we can argue that the proposed short-term parameter generation has an unnoticeable impact on the suprasegmental quality. The proposed performative HTS can be used in place of HTS when reactivity is a required feature.

### Case C & Case D

The preference scores when comparing full to reduced-context model training while synthesising using either HTS (Case C) or pHTS (Case D), along with the overall preference of the model training are presented in Table 4.14 and in Figure 4.12 as graphs. Results are shown for all, male and female speakers, with 95% confidence intervals. Here, positive values indicate preference for the HTS approach. It is important to keep in mind that we use a 7-point scale, ranging from  $-3$  to  $+3$ .

CMOS [7-scale] full vs. reduced	all speakers	male speakers	female speakers
(overall)	$0.470 \pm 0.058$	$0.596 \pm 0.068$	$0.216 \pm 0.107$
Case C	$0.425 \pm 0.082$	$0.528 \pm 0.097$	$0.219 \pm 0.149$
Case D	$0.514 \pm 0.083$	$0.665 \pm 0.096$	$0.214 \pm 0.155$

Table 4.14: Preference scores for all, male and female speakers when comparing full to reduced-context models when synthesising with either HTS (Case C) or pHTS (Case D), with 95% confidence intervals. Positive values indicate preference for the HTS approach.

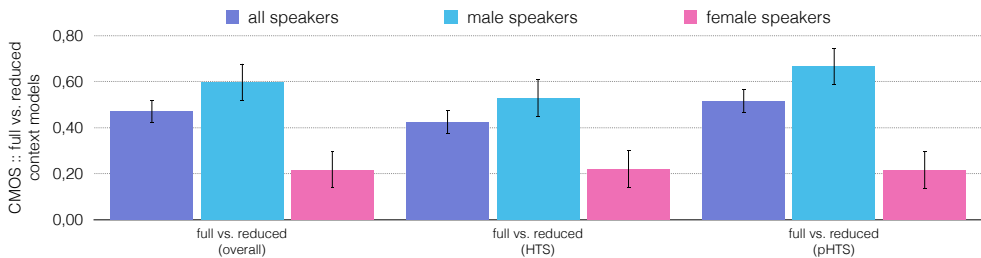


Figure 4.12: Preference scores for all, male and female speakers when comparing full to reduced-context models while synthesising using either HTS (Case C) or pHTS (Case D), with 95% confidence intervals. Positive values indicate preference for the HTS approach.

We see that the reduction of subjective quality introduced by the reduced-context training, is relatively minor and independent from the synthesis

method used. Indeed, the scores obtained with this test show that we stay between 0 and 1, i. e. *about the same* and *slight preference* for full-context models. While being noticeable, it means that the introduced distortions are not stressed by the listeners as a real issue regarding the suprasegmental quality. This indicates that reducing the amount of contextual information to only preceding and current factors at the training stage, as detailed in [Section 4.2](#) has a small impact on the suprasegmental quality, as expected. However, we see that the effect on the segmental quality is also negligible although higher degradation was expected.

#### 4.4 DISCUSSION

In this Chapter we presented our work on the performative HMM-based speech synthesis system which includes reduced-context synthesis as well as reduced-context training. Firstly, reduced-context synthesis is achieved by means of the proposed short-term parameter generation algorithm (ST-MLPG). Secondly, for the reduced-context training we reduce the linguistic context factors taken into account for the model building. The proposed performative HMM-based speech synthesis system opens the enclosed loop of the original system allowing the user to interact on every level of artificial speech production (i. e. phonetic labels, parameter generation, models and samples). Indeed, by generating parameter trajectories within a small sliding window it is possible to apply various controls over the models and the parameters themselves corresponding to the currently synthesised label. A complete description of these controls can be found in [Section 8.2](#). We see that the short-term parameter generation algorithm gives results very close to the original parameter generation algorithm (MLPG) and users tend to be unable to distinguish between the two approaches. Therefore we conclude that when reactivity and user interaction is needed in the HMM-based speech synthesis framework, pHTS can be used instead of HTS. Applications introducing performative HTS are presented in [Chapter 7](#).

The proposed ST-MLPG is a simplification of the original MLPG, that allows the trajectory generation process to start once the first phonetic label is received and not at the end of a full phonetic label sequence. It is the key algorithm we have introduced in this PhD thesis for performative HMM-based speech synthesis, allowing smooth trajectory generation over a small

time window, suitable for short-term user interaction. A similar approach is presented by Koishida et al. in [47], where the parameter trajectories are generated in a time-recursive manner. The principal difference is that the number of frames used is constant at every given time, while in our case it varies depending on  $f_1$ , the total number of frames that correspond to the labels currently included in the observation window. The main reason for choosing ST-MLPG over Koishida's solution is that since the controls are correlated to the context, (i. e. a high level control when compared to the frame), it is easier for the user to interactively manipulate the generated speech at the phoneme level. For example controlling accent or degree of articulation separately on every phoneme [Chapter 5] has a more meaningful interpretation rather than when using frames. Moreover, using regular expressions over the input context enables complex and accurate controls [Section 8.2.2].

In order to have reactive parameter generation we use a reduced observation window, whose size is defined empirically. Although the results suggest that the distortions introduced by the size of the sliding window are small, it is shown that by adding one label of succeeding dependency there is almost no distortion when compared to the original approach. However such an approach introduces a one-phoneme delay in the user controls and thus, we propose as a better compromise to keep the empirically defined window size and have the minimum possible delay. Yet, in cases where small delays in interaction can be tolerated then certainly increasing the window size and including an extra future label is recommended.

In previous research [ABdD11] the distortions introduced by pHTS when using a much larger sliding window, i. e. one succeeding, the current and all the available preceding labels ( $F = 1$  and  $P = +\infty$ ) were higher than those presented here. The main reason is that in the present thesis during synthesis we tested the stability of the MLSA digital filter [Section 4.1] of the generated spectral parameters (i.e. by using *mlsachek*, SPTK-3.6 [78] that implements [48]) and if necessary modify them depending on  $P_a$ , the Padé approximation order used - in our case  $P_a = 5$  so that the log approximation error does not exceed 0.2735 dB. Actually, stabilising the generated spectral parameters is the reason that eliminating any future dependencies is possible ( $F = 0$ ) and at the same time have small spectral distortions. Of course we should also take into account the fact that the training and synthesis techniques were improved by the HTS community since the time of publication.

In [ABdD11] we used HTS version 2.1 and HTS-engine version 1.00 while here we used HTS version 2.2 and HTS-engine version 1.06.

Finally, we observe that the introduced distortions are mainly dependent on the kind of models used for the synthesis and not the synthesis method itself. Indeed, when reduced-context models are used, the highest distortions are obtained. Thus, when using reduced-context models coherently at training and synthesis, users seem to have a slight preference for models trained on full-context. Hence, it is recommended that full-context models are used, in order to preserve maximum speech quality along with interactive user controls. However, the reduced-context models can be used in application that target interactive contextual control and reactive creation of the contextual input. Finally, we should highlight that when it comes to interactive synthesis and controls the assessment should not focus only on the speech quality itself, since sounds becomes yet another factor among many others in the context of interaction.

# 5

## REACTIVE AND CONTINUOUS MODEL INTERPOLATION

---

### Contents

---

5.1	Reactive synthesis using model interpolation . . . . .	79
5.1.1	Defining reactive model interpolation . . . . .	79
5.1.2	Interpolation strategies . . . . .	80
5.1.3	Reactive large-scale interpolation . . . . .	82
5.2	Interactive modification of the degree of articulation . . . . .	83
5.2.1	Evaluation . . . . .	84
	Experimental protocol . . . . .	85
	Subjective evaluation . . . . .	86
5.3	Interactive accent interpolation . . . . .	87
5.4	Discussion . . . . .	92

---

As it has been previously addressed, interactivity in HMM-based speech synthesis has barely been considered. We can only highlight the consideration of computing HMM-based trajectories in realtime [3], i.e. faster than the duration of the corresponding sound. A solution to this comes with the proposed performative HMM-based speech system, called pHTS. pHTS computes the synthetic speech reactively using the short-term parameter generation algorithm. It results into on-the-fly generation of speech parameters, consecutively for each label, therefore enabling the modification of the ongoing sound generation [Chapter 4]. Although pHTS brings performative controls over the generation of artificial speech, these controls are until now limited to prosody and, to some extent, speaking style, i. e. pitch, speed, volume, vocal tract length and state duration. If such controls lead to interesting results while directly applied to pitch curve and durations (application examples are detailed in Chapter 7), the leap forward comes when using it with more complex model transformations.

In this Chapter, we want to evaluate how some existing speaker interpolation techniques can benefit from our new reactive and continuous controls. Model interpolation has been tested for changing the speaking style [79] or speaker emotion [57] in the standard HTS framework. When integrated into pHTS, model interpolation algorithms can be applied separately on every phoneme and impact the resulting speech sound reactively. This means that we can vary the speaking style, speaker identity or emotion on the continuous scale independently on every phonetic label as the speech signal is being synthesised and heard by the user. This property enables the application of variable interpolation strategies under different interaction cases. For example transitioning between hypo- and hyper-articulated speech [79] in quickly-changing use cases, e. g. while an artificial speaker tries to get back the attention of a distracted listener or overcome sudden background noise. Another example can be the interactive voice customisation and fine tuning of an artificial voice for speech-impaired people, e. g. creating the patient's accent by using several other accents. Another important aspect is that the expressivity range of artificial voices can be enriched if emotions are reactively controlled while a sentence is being synthesised. Additionally, this approach enables the realtime browsing of the complete stylistic space of a given database<sup>1</sup> of recordings. The important feature of browsing a database on-the-fly is that we are able to observe the transitions between different interpolation combinations as well as the behaviour of the models in unusual cases. It is important to highlight here that this is a mathematical interpretation of the interpolated combinations. Actual interpolated results might have different real-word interpretations or realisation. Note that realtime control of the accent or degree of articulation is just an illustration of the realtime interpolation concept. This approach can be applied potentially to any model interpolation strategy, regarding speech but also non-speech cases.

The structure of this Chapter is as follows: [Section 5.1](#) details how the reactive and continuous model interpolation is integrated in the pHTS framework. [Section 5.2](#) and [Section 5.3](#) demonstrate two cases where interactive model interpolation is applied. Firstly, a simple case is presented, where we reactively control the degree of articulation and secondly, a more complex case is depicted, where we browse the complete space of English accents in realtime. Finally [Section 5.4](#) discusses results and challenges of the proposed approach of interactive model interpolation.

---

<sup>1</sup> A database can be any collection of signals, e. g. speech, laughter, gait, etc.

## 5.1 REACTIVE SYNTHESIS USING MODEL INTERPOLATION

The flexibility of statistical parametric speech synthesis based on HMMs is well-established, e. g. [57], [3]. All aspects of speech, comprising the spectral envelope, excitation parameters, and segment duration are modelled and generated simultaneously in one framework. A significant advantage of this model-based parametric approach is the ability to manipulate the model parameters in principled ways, such as adaptation to new data or interpolation between models, and thus manipulate the characteristics of the generated speech. These techniques have already been applied to generating or morphing between different speakers, different types of emotional speech, and different speaking styles [57].

Speaker interpolation is performed in [40] by interpolating HMM parameters among HMM sets from some representative speakers, showing that by interpolating between two speakers' models it is possible to synthesise speech with voice characteristics in-between the two speakers. Yamagishi et al. [80] replace the speaker models with models of different speaking styles (i. e. emotions). Likewise we will apply this approach to the degree of articulation (Section 5.2) and accent interpolation (Section 5.3).

### 5.1.1 *Defining reactive model interpolation*

Since techniques such as interpolation were initially developed for the traditional HTS system, we slightly modified and integrated them into pHTS, so that we can expose some interpolation parameters to the interactive control of users. Among the three interpolation methods detailed in [40], we adopted the simplest one, where each HMM state is assumed to have a single Gaussian output distribution and therefore the final interpolation product is the interpolation among  $N$  Gaussian distributions. Let  $S_1, S_2, \dots, S_N$  be the  $N$  styles represented in our database and  $\lambda_1, \lambda_2, \dots, \lambda_N$  be the models corresponding to every style. Respectively, let  $\tilde{S}$  be the style obtained by interpolating  $n$  out of the  $N$  styles and  $\tilde{\lambda}$  be its corresponding model. However, if there is not a tying structure common to all models it is rather difficult to obtain  $\tilde{\lambda}$  by interpolating  $\lambda_n$  models. As suggested in [80] during synthesis  $n$  pdf sequences are generated from  $\lambda_n$  independently, and then a pdf sequence is obtained that corresponds to  $\tilde{\lambda}$  by interpolating these  $n$  pdf

sequences. An additional reason to use this approach is that in the interactive framework of pHTS, it is practically impossible to create in advance a  $\tilde{\lambda}$  model for every possible user control.

Since in pHTS enables control on every phonetic label independently, let  $\tilde{S}_l$  be the style obtained for the current label  $l$  in the proposed sliding window. Therefore, the mean and variance vectors  $\tilde{\mu}_l$  and  $\tilde{\sigma}_l$  of the style  $\tilde{S}_l$  are obtained by the weighted sum (with  $w_{l1}, w_{l2}, \dots, w_{ln}$  the weights) of the mean and variance vectors  $\mu_{l1}, \mu_{l2}, \dots, \mu_{ln}$  and  $\sigma_{l1}^2, \sigma_{l2}^2, \dots, \sigma_{ln}^2$  of the representative styles as follows:

$$\tilde{\mu}_l = \sum_{k=1}^n w_{lk} \mu_{lk} \quad (5.1)$$

$$\tilde{\sigma}_l^2 = \sum_{k=1}^n w_{lk}^2 \sigma_{lk}^2, \quad \text{where} \quad \sum_{k=1}^n w_{lk} = 1 \quad \text{and} \quad 1 \leq n \leq N \quad (5.2)$$

Then as explained in [Section 4.1](#) from [Equation 5.1](#) and [Equation 5.2](#) we can determine the speech parameter vector sequence for the currently synthesised label within the boundaries of the sliding window by means of ST-MLPG. Hence, it is possible to change the style of every single phoneme individually, i. e. change the interpolation weights used for every phoneme independently during synthesis and therefore achieve interactive model interpolation as the targeted sentence is being synthesised. This feature allows reactive and continuous control over the degree of interpolation between various models. Moreover, it is possible to have different interpolation weights separately for every generated feature stream. Finally, any other control, i. e. pitch, duration, speed, vocal tract length can be also added and applied accordingly. [Figure 5.1](#) gives a graphical representation of the short-term parameter generation while interpolating multiple models.

### 5.1.2 Interpolation strategies

By means of interpolation it is possible to synthesise speech with voice characteristics that were not included in the initial training database, as detailed in [\[3\]](#). Here we also use style as a placeholder for speaker identity, speaking style or emotions represented as  $S_1, S_2, \dots, S_N$  and the corresponding models

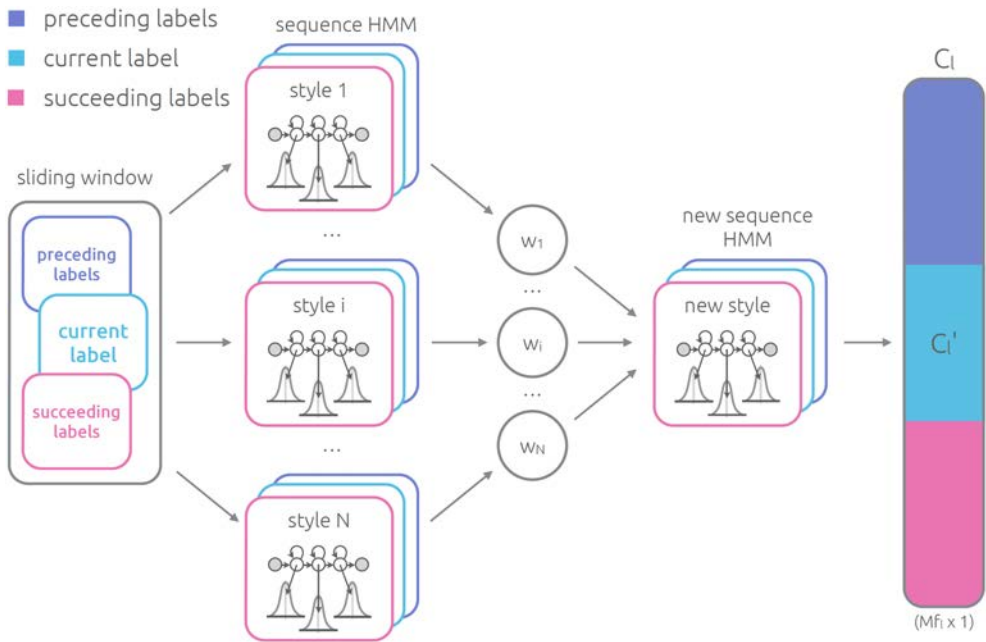


Figure 5.1: Short-term parameter generation while interpolating multiple models. Reactive and continuous model interpolation is achieved by controlling the interpolation weights used for every phoneme independently.

as  $\lambda_1, \lambda_2, \dots, \lambda_N$ . Let  $\lambda_0$  be the average voice model [39], i. e. the model trained on all the available training data.  $\tilde{S}_i$  is obtained by the weighted sum of the mean and variance vectors of certain styles (Equation 5.1 and Equation 5.2).

Depending now on the interpolation strategy, i. e. the manner in which the weights are set and the models are selected, we can:

- *blend* two or more arbitrary styles, by simple interpolation between two or more arbitrary models  $\lambda_n$ ;
- *inhibit* a given style, by interpolation between a model  $\lambda_n$  and the average model  $\lambda_0$ ;
- *exaggerate* a given style, by interpolation between a model  $\lambda_n$  and the average model  $\lambda_0$ , extrapolating beyond the model  $\lambda_n$ ;
- *invert* a given style, by interpolation between a model  $\lambda_n$  and the average model  $\lambda_0$ , extrapolating beyond the average voice model  $\lambda_0$ ;

Figure 5.2 illustrates these strategies. More specifically, *blending* between styles creates an in-between style. *Inhibition* smoothes/softens the style of model  $\lambda_n$  and *exaggeration* controls its intensity. Finally, *inversion* creates the opposite/negative of a style<sup>2</sup>.

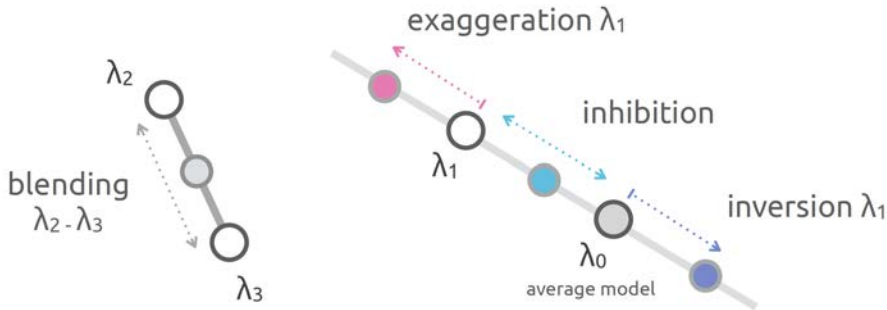


Figure 5.2: Exaggeration, inhibition, inversion and blending of given styles by means of linear interpolation [TdAR14].

All these interpolation strategies are initially developed for the original system, HTS, yet in that framework it is not possible to continuously browse the complete stylistic space of the available models and observe the transitions between different interpolation combinations. On the other hand, pHTS makes the interactive exploration of a database possible, and in many cases it simply makes it fun. Completely new styles can be built by combining any of the existing ones with refined transitional weights for every feature stream, enabling the free browsing of the complete stylistic space.

### 5.1.3 Reactive large-scale interpolation

It becomes obvious from Equation 5.1 and Equation 5.2 that by means of linear interpolation we can use up to  $N$  number of models in order to obtain the style  $\tilde{S}$  and its corresponding  $\tilde{\lambda}$  model. Although, in theory  $N$  can grow up to  $\infty$ , from the literature [80], [81] we see that in most cases it is not larger than five. The question that rises here is what happens if we use simultaneously more than 10 models? What happens when we have to deal with large-scale interpolation, e. g. 20, 100, 200 or even 500 models at a time?

<sup>2</sup> This is a mathematical interpretation of interpolation.

In the usual context of speech it is not so meaningful to interpolate many models at a time since in most cases the results are very averaged. However, this is due to the fact that usually all the feature streams of the models are interpolated in the same way while “traveling” from one style to the other. On the other hand, the creation of more complex styles, not included in the initial recordings, might require more sophisticated interpolation strategies as well. For example, in order to create style  $\tilde{S}$ , we might need to interpolate the spectral features of style C created by  $\lambda_c$  models ( $1 \leq c \leq N$ ), the excitation features of style E created by  $\lambda_e$  models ( $1 \leq e \leq N$ ) and the durations of style D created by  $\lambda_d$  models ( $1 \leq d \leq N$ ), combined also with non uniform weights.

As the number of interpolated models increases so as to generate the parameter trajectories, the demand of memory and computational power increases equally whether it targets a complete utterance (HTS) or just a phoneme (pHTS). The main difference though between HTS and pHTS is that, in the first case, even if the complexity results in computations that are not realtime (i. e. slower than the duration of the corresponding sound) it does not appear in the final output received by the user. However, in the latter case if the computations are not efficient enough it results in delays and glitches that give an unpleasant result and degrade the final speech quality. We apply this reactive large-scale interpolation approach in [Section 5.3](#) and actually create an interactive accent control application in [Section 7.1.4](#).

## 5.2 INTERACTIVE MODIFICATION OF THE DEGREE OF ARTICULATION

In [\[79\]](#), [\[82\]](#) Picart et al. detail how the degree of articulation can be continuously controlled in the HMM-based speech synthesis framework. An average voice model, corresponding to neutral speech is adapted by means of Constrained Maximum Likelihood Linear Regression (CMLLR) transformed with hypo- and hyper-articulated speech data. “Hyper-articulated speech” refers to the situation of a teacher/speaker talking in front of a large audience (i. e. important articulation efforts have to be made to be understood by everybody). “Hypo-articulated speech” refers to the situation of a person talking in a narrow environment or very close to someone (i. e. few articulation efforts have to be made to be understood). Due to the parametric representation of speech production (segmental and suprasegmental) used in

HMM-based synthesis, interpolation between these different speaking styles is possible. Hence speech sentences can be synthesised on a continuous degree of articulation scale. Although the use of articulatory data in HTS can provide a more continuous phonetic space [60], controlling the degree of articulation depending on the user preferences or adjusting the system to its environment, such as sudden increases of noise, while the targeted sentence is synthesised is still not feasible.

However, when these models are used in the performative HMM-based context it is actually possible to control the degree of articulation in realtime, and continuously browse the complete space. In the context of a collaborative work around hypo- and hyper-articulated speech [AdP<sup>+</sup>12], we had the chance to apply our reactive model interpolation idea to a French database that was recorded in [82] by a professional male speaker, aged 25 and native French (Belgium) speaker. The database contains three separate sets, each set corresponding to one degree of articulation (neutral, hypo- and hyper-articulated). For each set, the speaker was asked to pronounce the same 1359 phonetically balanced sentences (around 75, 50 and 100 minutes of neutral, hypo- and hyper-articulated speech respectively), as neutrally as possible from the emotional point of view. A headset was provided to the speaker for both hypo- and hyper-articulated recordings, in order to induce him to speak naturally while modifying his degree of articulation. Further details can be found in [82]. In [79] and this work, our average voice model corresponds to the standard neutral HMM model, using 75-dimensional Mel Generalized Cepstral (MGC) coefficients [75], including their first and second derivatives. For each degree of articulation, this neutral HMM model is adapted using CMLLR, with hypo- and hyper-articulated speech data to produce a hypo- and hyper-articulated HMM models.

### 5.2.1 *Evaluation*

In this Section, our goal is to evaluate how realtime interpolation between various models can be perceived by the user. Here, we present the results of a subjective test that we conducted to evaluate the user's perception of abruptness/fluidity in transitioning between hypo- and hyper-articulated speech, although this approach can be applied to any interpolation strategy. For this test, we address the idea of varying speaking style along the sentence, as

if an external condition had changed it: sudden background noise, listener reactions or gestures, change of emotion, etc. We are interested in how a time-varying continuous control (in this case of the degree of articulation) is perceived by the listener, as a way to evaluate what level of refinement is needed in a context-reactive speech synthesiser.

### *Experimental protocol*

For this subjective evaluation, listeners were asked to listen to sentences synthesised with pHTS combined with one of the three different articulation trajectories:

- step trajectory, where the first half of the sentence is exaggerated-hypo (exaggerated-hyper) articulated and the second half is exaggerated-hyper (exaggerated-hypo) articulated;
- linear trajectory, where the degree of articulation changes linearly from exaggerated-hypo (exaggerated-hyper) articulated to exaggerated-hyper (exaggerated-hypo) articulated over the whole sentence;
- step-linear trajectory, where the first quarter of the sentence is exaggerated-hypo (exaggerated-hyper) articulated, the last quarter is exaggerated-hyper (exaggerated-hypo) articulated and the degree of articulation is linearly transitioning between the two extrema in the middle.

For the cases of linear and step-linear trajectory, the degree of articulation is actually evolving linearly with the phoneme sequence (whose duration is dependent upon the speech rate), and not with time. In other words, for every phoneme we assign a different degree of articulation, but this control is not applied linearly on time, since every phoneme has a different duration, depending on the phoneme itself as well as on the weights and the models themselves used for interpolation. The test consisted of 30 sentences randomly chosen among the held-out set of the database (used neither for training nor for adaptation). Participants were asked to score the fluidity of the transition between the two degrees of articulation. In order to do that, they were given a continuous scale ranging from 1 (very abrupt) to 5 (very fluid). These scales were extended one point further on both sides in order to prevent border effects. We can also note that these sentences are relatively

short, 2 or 3 seconds each. It means that we rather try to assess listener’s ability to discriminate short-term variations in the degree of articulation. Since this control can only be performed with pHTS, and not with the regular version of HTS the listening test on the perception of abruptness/fluidity in transitioning between different degrees of articulation only involves pHTS.

### *Subjective evaluation*

The obtained results are presented in [Table 5.1](#). It is observed that participants could correctly discriminate between various types of changes in the degree of articulation, happening within a sentence, as expected from the reference [79]. Particularly the step and linear variations of the degree of in pHTS led to a clear listener decision, respectively for considering this change very abrupt or very fluid. Furthermore, as expected, the step-linear transition is observed to be perceived in between the step and the linear transitions on the abruptness/fluidity scale.

<b>trajectory strategy</b>	<b>fluidity / abruptness (1-very abrupt to 5-very fluid)</b>
step trajectory	$1.84 \pm 0.31$
step-linear trajectory	$3.12 \pm 0.52$
linear trajectory	$3.99 \pm 0.66$

Table 5.1: Fluidity scores of the transition between the two degrees of articulation, with 95% confidence intervals.

Varying the parameterisation of articulation changes, i. e. from step to linear, and showing that such parametrisation is discriminated by listeners is a way to validate that the suprasegmental quality of the degree of articulation is a meaningful parameter to be included in a reactive speech synthesiser. Indeed, we can conjecture that quality of variation in the speaking style is a consistent cue in verbal communication and provides useful information on the speaker’s state and context. It can then be concluded from this experiment that the proposed performative approach allows both a continuous and reactive interpolation of models in the framework of HMM-based

speech synthesis, which is here shown to enable a perceptually-motivated control of the degree of articulation by the user.

Figure 5.3 gives a more complete image of the obtained results. The x axis represents the estimated abruptness, from very abrupt to very fluid; the y axis represents the normalised frequency of occurrence of this choice during the test; the values for the three types of sentences are displayed. We can observe that there is a correlation between the type of sentence presented to the listeners and the maximum frequency on the abruptness axis. Indeed, step trajectories in the degree of articulation are perceived as abruptly changing, and linear trajectories in the degree of articulation as fluidly changing. We can also highlight some correlation between the step-linear trajectories in the degree of articulation and the middle of the abruptness axis, although the maximum of that frequency curve seems to be slightly on the fluidly changing side of the abruptness axis.

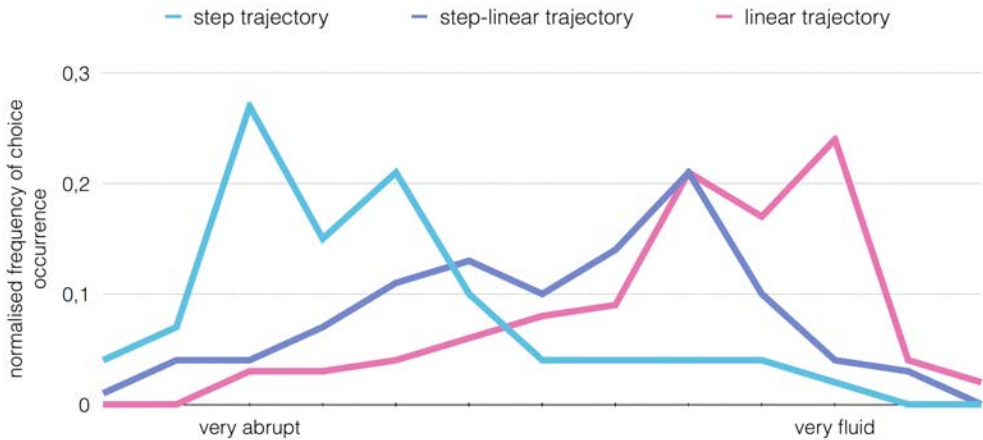


Figure 5.3: Fluidity scores of the transition between the two degrees of articulation, when using step, step-linear or linear control trajectories.

### 5.3 INTERACTIVE ACCENT INTERPOLATION

As detailed in Section 5.1, reactive and continuous model interpolation is possible, while preserving the intelligibility and naturalness achieved by the conventional HMM-based speech synthesis system [Section 4.3]. However,

this reactive control is applied on only three different models. On the one hand this is enough as a proof of concept but on the other hand it is limited by the small number of available models. We think that it is important to have reactive and continuous model interpolation also in large-scale [Section 5.1.3], i. e. using simultaneously several different models in the reactive framework of pHTS. It is important to highlight here that using arbitrary models, such as the models provided in the CMU ARCTIC database [31], [32], for interpolating several speaker models at the same time could show, to some extent, if achieving the large-scale model interpolation is possible but it would not show whether it is meaningful or not.

The availability of the Voicebank project [83] brings a meaningful use for the reactive and continuous large-scale model interpolation. This is a collaborative project for voice banking and voice reconstruction, initiated in 2011, between the Centre for Speech Technology Research (CSTR) based at Edinburgh University, the Euan MacDonald Centre for MND and the Anne Rowling Regenerative Neurology Clinic.

The aim of this project is to create personalised voices, for use in a communication aid, for individuals who have lost the ability to speak due to a neurodegenerative disease (specifically motor neurone disease (MND)). The CSTR voice banking corpus, which contains a total of around 500 speakers with many and varied UK and North American accents was used to test the proposed realtime large-scale model interpolation. Here, the stylistic model space is actually the “accent” space. In this case, we use the term “accent” to mean the pronunciation of individual vowels and consonants (such as the height of a particular vowel). We assume there is, at least to a first approximation, a continuum of accents: that is, accent changes gradually in accordance with other speaker factors such as geographical location. Therefore, since the acoustic realisation of phonemes gradually changes, accent can be manipulated through the interpolation of models. In other words, given two voices with differing accents, it is possible to create an accent “half way” between them. Similarly, given several voices it is reasonable to create an accent “somewhere” in-between them (i. e. *blending*). As described in Section 3.2.5, speaker adaptation can be performed by adapting an average voice model (AVM) to a specific target speaker by means of Maximum A Posteriori estimation (MAP) or Maximum Likelihood Linear Regression (MLLR). In this work from our speaker database, we choose different accents, i. e. Scottish, English, North American, Australian and we build an accent AVM for each.

When these accent AVMs are built, by means of MLLR or MAP, we can adapt them to any speaker with any accent. As shown above, the interpolation between these different speaking styles is possible allowing us to synthesise speech sentences on a continuous accent degree scale in realtime. It is important to note here, that varying the dialect of a TTS synthesis is more complex because this can involve changes beyond accent, into the phonology, vocabulary and grammar of the language; it is no longer reasonable to assume there is a continuum and one must deal with discontinuities, such as abrupt changes in the phonology (e.g. the number of phonemes in the language). An example of this can be found in [58].

For the interactive accent interpolation, it is important to separate our speaker accent from other characteristics somehow so listeners can pay attention only to accent transitions. One choice might be to use a single speaker who can produce all accents correctly. However, this is impractical: it would be hard or impossible to find such a speaker. Instead, we decided to use multiple speakers who have similar voices. By interpolating their acoustic models we obtain an average voice whose main distinguishable characteristic is the accent itself. In this case, the listeners only hear such an interpolated voice (i.e. never that of a single speaker).

Using the pHTS framework, we have prototyped interactive accent control of the HMM-based speech synthesiser where accent is controlled by means of a stylistic (i.e. accent) mixer, as illustrated in Figure 5.4. As users interact with the system, the perceived accent gradually changes depending on the particular speakers being interpolated. For any given model  $\lambda_n$  users are able to “invert”, “exaggerate”, or “inhibit” it. Combining then all the weights  $w_n$  the final accent  $\tilde{S}_l$  and its corresponding model  $\tilde{\lambda}_l$  are obtained. As a result, a completely unique accent is created for every phoneme of the sentence as it is being generated.

For simplicity reasons, when using the accent mixer only ten accent models were randomly selected from the complete corpus based on gender, i.e. either male or female voices. Although it is possible to have a mixer separately for every feature stream (i.e. here duration, spectrum and excitation) of the provided models, again for simplicity reasons in this case we use the same interpolation weight for every feature stream. This was a first test case for the large-scale interpolation, where simple sliders are employed to set the  $w_n$  weight for every  $\lambda_n$  used. A complete application of this work, using the full Voicebank corpus combined with a more meaningful geolocalized

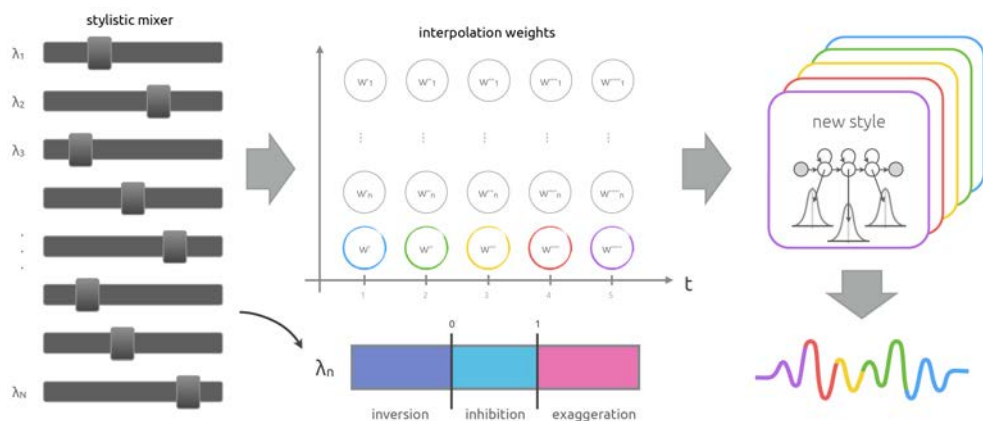


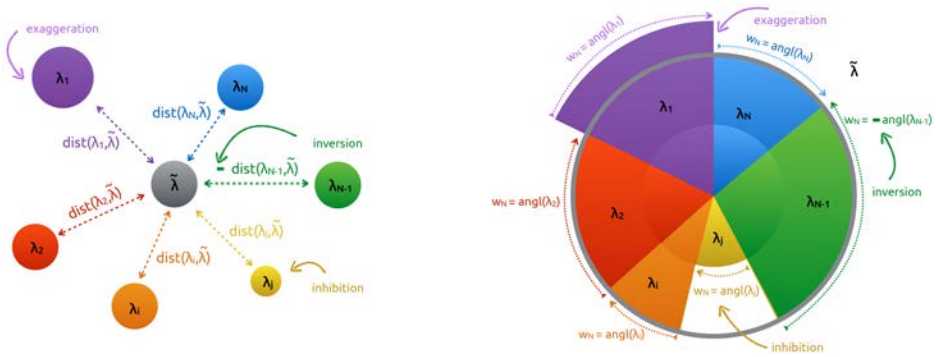
Figure 5.4: Creating a unique accent by means of model interpolation when using an accent mixer to set the interpolation weight vector.

interpolation control, can be found in [Section 7.1.4](#). Note here, that interpolating between several voice models is a computationally heavy process, which can lead to saturation and delays in the final speech output.

Although sliders give a good visual feedback to the user on how the stylistic space is being browsed, it becomes rather complex when many models are involved. Such an approach is more suitable for the previously detailed reactive control of the degree of articulation<sup>3</sup>, but not so much in this more complex case. In order to provide more meaningful controls over the interpolation weights (i. e. over the whole model or separately over every feature stream) the stylistic mixer can have other forms than a collection of simple sliders. [Figure 5.5](#) illustrates two possible (but not limited to) representation forms. In the first case, we use *distance* to set the interpolation weights while in the second case we use a *quantitative* approach.

When using *distance*, the stylistic mixer has the form of a graph. We represent every accent (i. e. model) in a space using circles, distributed either randomly or clustered based on certain characteristics. Let  $\tilde{\lambda}$  be also represented as a circle, but in this case a distance between the existing models is created ([Figure 5.5a](#)). Then by moving the  $\lambda_n$  models closer to or further from the  $\tilde{\lambda}$ , the *distance*  $\text{dist}(\lambda_n, \tilde{\lambda})$  is set and thus the corresponding interpolation weights are set. The closer a model is to  $\tilde{\lambda}$  the higher its weight is and the further it is from the  $\tilde{\lambda}$  the smaller its weight is. In this approach, the

<sup>3</sup> Reactive control of the articulation degree demo (visited June 2014): <https://vimeo.com/53605118>



(a) Setting the interpolation weights by means of *distance* from the  $\tilde{\lambda}$  model.

(b) Setting the interpolation weights by means of *quantity* of the  $\tilde{\lambda}$  model.

Figure 5.5: Using either a graph-formed or a piechart-formed stylistic mixer for interactive model interpolation.

“exaggeration” of a style is possible by increasing the size of the circle representing a model  $\lambda_n$ , and respectively by decreasing the the size of the circle the style is “inhibited”. In order to “invert” a style then *negative* distance must be defined.

When using *quantity*, the stylistic mixer takes the form of a “piechart”. Here, every  $\lambda_n$  is presented as a sector while  $\tilde{\lambda}$  itself is the collection of the sectors, i. e. the complete “piechart” (Figure 5.5b). By increasing or decreasing the angle of a sector  $\text{angle}(\lambda_n, \tilde{\lambda})$ , the interpolation weights of every model are defined accordingly. The bigger the angle, the higher the weight is, and respectively the smaller the angle, the lower the weight. Increasing and decreasing the radius of a sector results in “exaggeration” and “inhibition” of a style respectively. Finally, defining a negative sector angle results in style “inversion”. Again, it is possible to control the interpolation weights for all feature streams of a model either collectively or separately. Note here that the presented *distance* and *quantity* stylistic mixers were not tested in the context of interactive accent interpolation. However they are expected to be evaluated in a different interpolation case with a more simplified form (i. e. supporting only “blending” and not “exaggeration”, “inhibition” or “inversion”) as detailed in Section 7.1.6. It is important to highlight that the actual mapping strategy and computation of the interpolation weights is let to the designer to define.

Some preliminary tests indicate that it was difficult to distinguish between different original accents, or interpolated accents, even when referring to native english speakers, since they have to be exposed to these accents. Although it is not straightforward to formally evaluate the proposed interactive accent interpolation control we can certainly demonstrate that the manipulation of accent in realtime is possible and that it could result in the implementation of various applications, i. e. for speech therapy and pedagogy or even for cultural heritage (i. e. geolocalized evolution or immigration of accents).

#### 5.4 DISCUSSION

Even though model interpolation is possible with HTS, this is done over all the generated speech parameter trajectories, resulting in a full sentence with a constant (and hence not a transitional) speaking style, emotion, etc. Therefore it is important to highlight that this control can only be performed with pHTS. Indeed, it takes the short-term generation of speech parameter trajectories described in [Section 4.1](#) to actually be able to perform continuous and reactive modification over the provided models in realtime, as the speech samples are generated.

In the first application of interactive model interpolation we used models for neutral, hypo- and hyper-articulated speech and synthesised sentences where the degree of articulation varies linearly over the synthesised phonemes. Since every phoneme has a different duration, that is influenced also by the articulation models used, the final articulatory transitions are not linear in time. Results obtained from the subjective evaluation show that users can discriminate between different trajectories transitioning from one degree of articulation to the other. This type of model interpolation control can only be performed with pHTS and not with the regular version of HTS. Hence, it is difficult to obtain any objective or subjective comparison between the two systems. Since pHTS has almost similar segmental quality to HTS [[Section 4.3](#)] we conjecture that, likewise, in the context of reactive model interpolation there is negligible quality degradation. Furthermore, we demonstrate that large-scale interpolation is feasible in realtime, and limited only by the hardware (i. e. tested on a standard personal computer). Preliminary tests of the interactive accent control show that even though the proposed accent control mechanism is meaningful for the user, the expres-

siveness of the output is not clearly perceived since the voice differences are very slight between neighbouring speakers and not clearly perceived by non-native english speakers.

In both demonstrated cases, the model manipulation is phoneme-based and no restriction as to how and when modifications could be made were imposed. This allows the user to generate arbitrary patterns for the resulting speech, which generally sound very similar but may sound rather unnatural in some cases. Setting more restricted patterns to convey the final target, such as the ones used in natural human speech, could probably lead to more distinguishable results. To allow meaningful manipulation at a higher level so as to enable suprasegmental control of expressiveness, we would need to use a better underlying model such as longer-unit model [84], [85] and a statistical model that is based on a physical model [86] to restrict the modifications of accent and degree of articulation to a subset of possibilities that sound more natural. Further evaluation - both objective and subjective as well as user studies - would be beneficial for our research. Additionally, it would be interesting to have the point of view of linguists and native speakers (e.g. English and French in our cases).

Independently of the applied interpolation scheme in the large-scale context, there are some aspects that can benefit from improvement. First, it would be interesting to optimise the existing implementation in order to make it less computationally heavy and memory demanding. Further optimisation could be achieved by pruning the available styles. For example, in the case of accent, to linguistically prune the available voices. Such a pruning would lead to only keeping the most interesting voices in terms of the targeted style (e.g. accent), and therefore use more efficiently the available memory and CPU of a given device.

We think that pHTS is among the first tools that allows us to explore interactive browsing of stylistic databases. Additionally, by controlling the interpolation weights separately on every feature stream, the possible transitions and combinations are infinite. This opens the doors for creating completely new refined styles as well as investigating transitions and combinations among various styles and their feature streams.



---

 REACTIVE ARTICULATORY FEATURE GENERATION
 

---

 Contents
 

---

6.1	Reactive synthesis using articulatory features . . . . .	96
6.2	Evaluation . . . . .	99
6.2.1	Experimental protocol . . . . .	99
6.2.2	Objective evaluation . . . . .	101
6.2.3	Subjective evaluation . . . . .	105
6.3	Reactive articulatory control . . . . .	107
6.3.1	Reactive articulatory control application . . . . .	107
6.3.2	Challenges . . . . .	108
6.4	Discussion . . . . .	109

---

The controls that have been available until now, have been over the acoustic features, as used in both the conventional HMM-based speech synthesis and pHTS. These parameters are the ones required by the vocoder. Such parameters do not necessarily have a “physical” or “intuitive” meaning to the user. However, the physical nature of human speech production means that an articulatory parameterisation of speech has interesting properties. The articulatory features quantitatively describe the positions and the continuous movements of a group of human articulators, such as tongue, jaws, lips, velum. Such features have relatively slow evolution through time, they are not influenced in the same way by acoustic noise and other environmental conditions. They can provide a straightforward and simple explanation for speech characteristics and they provide meaningful interpretation of the speech production to the user.

A method for integrating articulatory features in HMM-based speech synthesis has already been proposed [60],[61], where the articulatory features were recorded using electromagnetic articulography (EMA). In this method, a unified acoustic-articulatory model is trained and a piecewise linear trans-

form is adopted to model the dependency of the acoustic features on the articulatory features. During synthesis, the articulatory features are generated from the previously trained models. Then, these generated articulatory features can be manipulated in arbitrary ways which in turn affect the generation of acoustic features. In this way, the characteristics of the synthetic speech can be controlled via an articulatory representation. The motivation of the work described here is to see whether reactive articulatory control is possible, to evaluate the results and then to explore the potential of different interactive applications that take advantage of the physical nature of the articulators.

Given the unified acoustic-articulatory model and a set of phonetic labels, it is possible to reactively generate the target speech samples. Simultaneously, it is possible to influence the generated articulatory features by replacing them with the user input. In this way, we can achieve the goal of altering the generated speech samples at the articulatory level rather than directly at the acoustic level. In [Section 6.1](#) we detail the modification in pHTS in order to generate and alter articulatory features in realtime. [Section 6.2](#) presents the objective and subjective evaluation of this approach. [Section 6.3](#) describes the reactive controls over the generated articulatory features, while in [Section 6.4](#) we discuss the results and the challenges of manipulating the articulatory data in realtime.

## 6.1 REACTIVE SYNTHESIS USING ARTICULATORY FEATURES

In HMM-based speech synthesis, a sequence of contextual phonetic labels is used to predict an optimal state sequence and the duration, in frames, of each state. In the case of acoustic features (i.e. spectral parameters), a state  $j$  corresponds to a multivariate Gaussian distribution whose parameters are  $\mu_j$ , its mean vector, and  $\Sigma_j$ , its covariance matrix. Given these parameters, the sequence of states and their durations are used to generate an optimal sequence of acoustic features  $\mathbf{X}$  that are then combined with synthetic source parameters to synthesise speech using, for instance, a vocoder. As detailed in [Chapter 3](#) and in [Chapter 4](#) respectively, the state sequence and the computation of parameters are performed by means of either the one-pass or the shifting short-term window approach.

The framework for HMM-based parametric speech synthesis has been expanded in [61] so that the acoustic models become dependent on articulatory features. One of the methods presented is called “feature-space-switched multiple regression HMM” (FSS-MRHMM). MRHMM consists in replacing the mean vector  $\boldsymbol{\mu}_j$  of each state by a linear combination of synthetic articulatory features  $\boldsymbol{\xi}_t$  and  $\boldsymbol{\mu}_j$ , before computing the optimal sequence of acoustic features. Therefore, the Gaussian distribution for each state  $t$  is defined as

$$b_j(\mathbf{x}_t|\mathbf{y}_t) = \mathcal{N}(\mathbf{x}_t; \mathbf{A}_t \boldsymbol{\xi}_t + \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j) \quad (6.1)$$

with  $\mathbf{x}_t$  a vector of static acoustic features and their first and second derivatives.  $\mathbf{A}_t$  is the articulatory-to-acoustic projection matrix and  $\boldsymbol{\xi}_t$  is an expanded articulatory feature vector, which means it contains  $[\mathbf{y}_t^\top, 1]$ , with  $\mathbf{y}_t$  a vector of static articulatory features and their first and second derivatives.

Usually,  $\mathbf{y}_t$  is generated using standard HMM-based synthesis with its own specific models of articulatory features. However, these features can be generated within the sliding window of ST-MLPG as detailed in Chapter 4. Additionally, they can be replaced by other values, based on either phonetic knowledge or user influence and thus modifying the identity of the synthetic phonemes (i. e. consonants, vowels, even full trajectories).

More specifically, in the case of FSS-MRHMM, a finite set of  $M$  matrices  $\{\mathbf{A}_1, \dots, \mathbf{A}_M\}$  is trained along with an  $M$ -mixture Gaussian mixture model (GMM) of the articulatory space. Then, at synthesis time, instead of a single Gaussian the probability density function of each state is written as

$$b_j(\mathbf{x}_t|\mathbf{y}_t) = \sum_{k=1}^M \zeta_k(t) \mathcal{N}(\mathbf{x}_t; \mathbf{A}_k \boldsymbol{\xi}_t + \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j) \quad (6.2)$$

where  $\zeta_k(t)$  is the probability for the mixture component  $k$  given  $\mathbf{y}_t$ . However, with such a model, the parameter generation would require to use an expectation-maximization-based iterative estimation [87]. The expectation-maximization (EM) algorithm cannot be applied in the context of a reactive application such as pHTS and we simplified it by considering only the mixture with maximum  $\zeta_k(t)$ , as proposed in [61].

Therefore, Equation 6.2 is rewritten as

$$k = \operatorname{argmax}_{k=[1,\dots,M]} \zeta_k(t) \quad (6.3)$$

$$b_j(\mathbf{x}_t|\mathbf{y}_t) = \mathcal{N}(\mathbf{x}_t; \mathbf{A}_k \boldsymbol{\xi}_t + \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j) \quad (6.4)$$

Therefore, at synthesis time the parameter generation criterion becomes

$$P(\mathbf{X}|\mathbf{Y}, \lambda, \mathbf{q}^*) = \prod_{t=1}^T b_j(\mathbf{x}_t|\mathbf{y}_t) \quad (6.5)$$

where  $\lambda$  is an HMM,  $\mathbf{q}^*$  is the optimal state sequence predicted using the trained state duration and  $T$  denotes the total number of frames (i.e. total number of states). Finally, Equation 6.5 is solved as detailed in Section 3.2 and in Section 4.1 respectively for the offline and realtime approach. Further details on the training of the different models (acoustic, articulatory, GMM and  $\mathbf{A}_k$ ) can be found in [61].

More specifically, in pHTS during synthesis, a given sequence of context-dependent phonetic labels is used to concatenate the context-dependent HMMs within the sliding window. The articulatory features are then predicted from the sequence HMMs by means of a short-term maximum output probability parameter generation algorithm that incorporates dynamic features. It is possible though during synthesis that  $\boldsymbol{\xi}_t$ , the generated articulatory features, may be modified or replaced either by user input or according to phonetic knowledge (as explained in [61]). Hence, the corresponding acoustic features are then generated, using Equation 6.3 and Equation 6.4, in order to reflect those articulatory changes. The speech waveform is then synthesised from the generated mel-cepstral and F0 parameter sequences using Mel Log Spectrum Approximation (MLSA) filter [48], with pulse-train (voiced frames) or white-noise excitation (unvoiced frames).

As explained in Section 4.1, only the current phonetic label (and if available, the two previous labels) is taken into account for the feature generation and therefore the generated parameter trajectories are locally-maximised. Here, for every phonetic label, the articulatory features are generated, taking into account the control of the user over the articulators (if any). Then, the acoustic features are generated in order to correspond to these articulatory features, as illustrated in Figure 6.1. The aim of this approach is to use

the articulatory features in order to replace to some extent the predefined context and to modify the acoustic features accordingly. In other words, the intention is to reactively alter a given context and its acoustic features by using only modifications over the articulatory features provided by the user. The user controls can be applied to the current phonetic label, without any dependencies on succeeding phonetic labels.

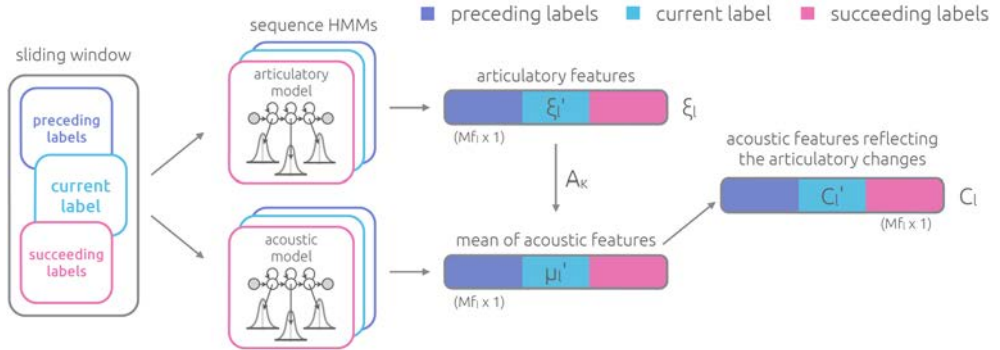


Figure 6.1: Realtime generation of acoustic features with articulatory control using acoustic-articulatory model.

## 6.2 EVALUATION

Both objective and subjective experiments were conducted in order to evaluate the similarities between trajectories generated using HTS (MLPG algorithm - [System I](#) & [System III](#)) and trajectories generated using pHTS (ST-MLPG algorithm - [System II](#) & [System IV](#)). Although the results obtained in [Section 4.3](#) show that objectively HTS and pHTS have very similar quality and subjectively the differences are indistinguishable among users, it is important to see how the speech quality is influenced by adding the articulatory features. The two synthesis methods are compared while using models trained on a multichannel articulatory database as detailed in [\[60\]](#).

### 6.2.1 Experimental protocol

As described in [\[60\]](#), the multichannel articulatory database contains 1263 phonetically balanced sentences, read by a male British English speaker.

Both acoustic waveforms and articulatory data (EMA) were recorded simultaneously. 1200 sentences were used for the training while the remaining 63 were held out as a test set. The six EMA sensors recorded the three dimensional spatial location of the following articulators: *tongue dorsum*, *tongue body*, *tongue tip*, *lower lip*, *upper lip* and *lower incisor*, as illustrated in Figure 6.2. Relative to viewing the speaker's head from the front, the x-axis represents the front-back movements, the y-axis represents the bottom-top movements and the z-axis represents the right-left movements. However, since the movement on the z-axis is negligible, they are not taken into account, and hence only the x- and y- coordinates are included in the database. For every frame we have 12 static articulatory features, i.e. which can also provided 12 degrees of freedom, 6 for the x-axis and 6 for the y-axis, the composed F0 and 40-order frequency-warped LSPs [88] plus the gain dimension, which is then converted into the traditional 40-order Mel Generalized Cepstral (MGC) [75].

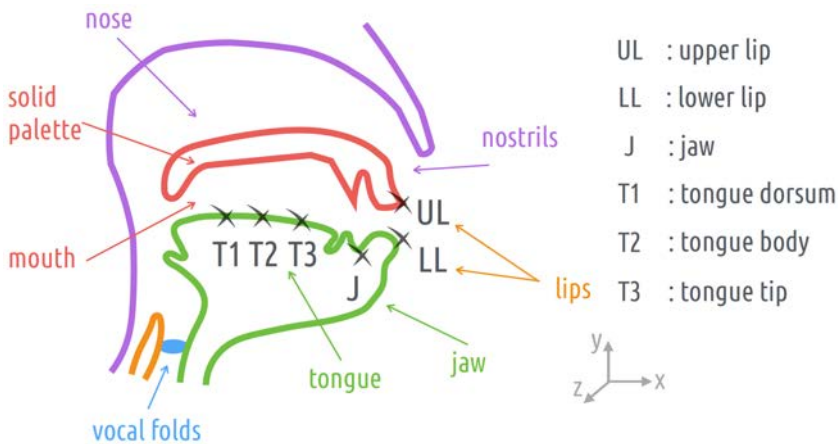


Figure 6.2: Placement of the 6 EMA receivers in the database used for the experiments. Only the x- and y- coordinates are included in the database.

It is important to highlight here that the 40-order LSPs are used with the STRAIGHT vocoder [69] while the 40-order MGCs are used with the standard MLSA vocoder. Actually, the reason that we convert our LSPs coefficients into MGCs, is that we do not have a realtime version of the STRAIGHT vocoder. Thus we need the MGCs so that we can synthesise the targeted output reactively by means of MLSA filtering. More specifically, LSPs and MGCs are both generated in realtime, but in the first case we save the LSPs

in a file and then use the STRAIGHT vocoder to create the audio samples offline, while in the second case we employ the MLSA vocoder and our samples are created in realtime. In order to ensure that state durations do not vary and be able to objectively compare our results on a frame-by-frame basis, synthesis was forced to use the phoneme durations, ensuring that both LSP and MGC-based synthetic speech samples are synchronous and comparable. However for the subjective evaluation the phoneme durations are modelled and not forced.

### 6.2.2 Objective evaluation

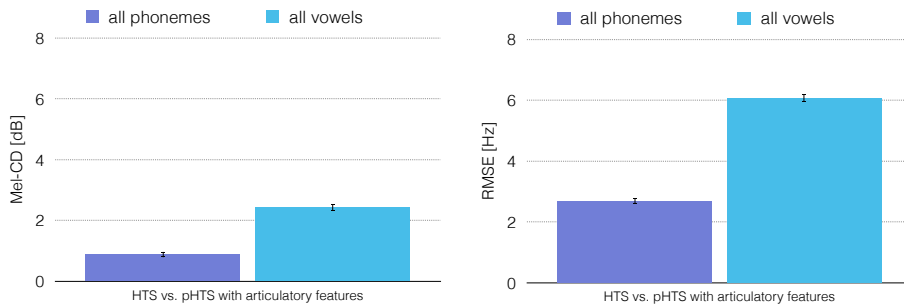
In order to evaluate the distortion introduced by pHTS, when combined with articulatory features, we make use of objective quality measurements by applying two different metrics, previously used in related research [76] and this work [Section 4.3]. The first metric used is the *Mel-Cepstral Distortion* (Mel-CD), expressed in decibel (dB), a distance measure calculated between the target and the estimated mel-cepstrum. The second metric used is the *Root-Mean-Square Error* of the fundamental frequency (F0-RMSE) expressed in Hertz (Hz). Note that the F0-RMSE is computed for frames where both data models are considered to be voiced, since F0 is not observed in unvoiced regions. Additionally, we use as a third metric the *Euclidean Distance* (ED), expressed in millimetres (mm), a distance measure calculated between the target and the estimated position of the EMA features.

Table 6.1 shows both the *Mel-Cepstral Distortion* and the *Root-Mean-Square Error* of the fundamental frequency averaged over the the *phoneme* and the *vowel* set respectively, over the 63 test sentences, with 95% confidence intervals. Figure 6.3 illustrates the obtained results as graphs.

Results indicate that the proposed pHTS system (ST-MLPG algorithm) lead to very similar segmental quality to the original one (MLPG algorithm) when articulatory features are integrated. Indeed, less than 1 dB as spectral difference and less than 3 Hz of frequency difference is observed when computed on phonemes while less than 2.5 dB as spectral difference and less than 6.5 Hz of frequency difference is observed when computed on vowels. However, we see that the distortions are higher when articulatory features are introduced than those obtained in Section 4.3.2. Although the results are encouraging, it is known that measuring distortion on any speech param-

HTS vs. pHTS with EMA features	Mel-CD [dB]	F0-RMSE [Hz]
phoneme set	$0.874 \pm 0.08$	$2.686 \pm 0.10$
vowel set	$2.409 \pm 0.09$	$6.080 \pm 0.12$

Table 6.1: Mean *Mel-CD* (in dB) and *F0-RMSE* (in Hz) introduced when comparing HTS to pHTS, combined with articulatory features, computed over the *phoneme* and *vowel* set, with 95% confidence intervals.



(a) Mean *Mel-CD* (in dB) computed over the *phoneme* set and *vowel* set.

(b) *F0-RMSE* (in Hz) computed over the *phoneme* set and *vowel* set.

Figure 6.3: Mean *Mel-CD* (in dB) and *F0-RMSE* (in Hz) introduced when comparing HTS to pHTS, combined with articulatory features, computed over the *phoneme* and *vowel* set, with 95% confidence intervals.

ter trajectory only gives a partial understanding of how the suprasegmental quality is affected by these synthesis modifications. Hence, this objective evaluation is completed with a set of subjective tests that cross-compare the influence of pHTS when it is combined with articulatory features. The obtained user preferences are detailed in [Section 6.2.3](#).

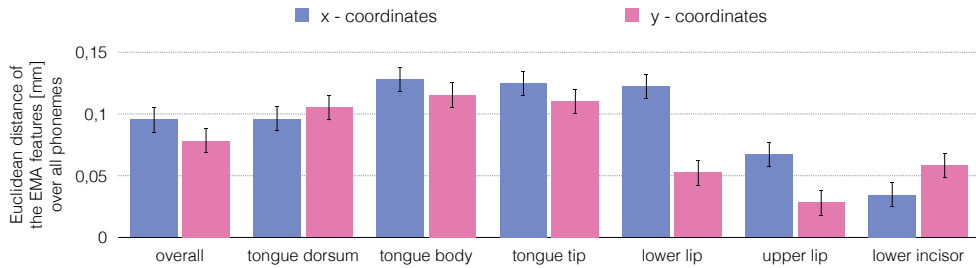
[Table 6.2](#) and [Table 6.3](#) show the *Euclidean Distance* averaged over the *phoneme* and the *vowel* set respectively, over the 63 test sentences for every EMA point for both *x*- and *y*- coordinates, with 95% confidence intervals. Additionally, [Figure 6.4](#) gives a graphical representation of the obtained results for better comparison.

<b>HTS vs. pHTS with EMA features (over phoneme set)</b>	<b>Euclidean Distance x-coordinate [mm]</b>	<b>Euclidean Distance y-coordinate [mm]</b>
overall	$0.095 \pm 0.08$	$0.078 \pm 0.09$
tongue dorsum	$0.095 \pm 0.01$	$0.104 \pm 0.01$
tongue body	$0.127 \pm 0.02$	$0.115 \pm 0.02$
tongue tip	$0.124 \pm 0.01$	$0.110 \pm 0.01$
lower lip	$0.121 \pm 0.03$	$0.052 \pm 0.03$
upper lip	$0.066 \pm 0.03$	$0.027 \pm 0.02$
lower incisor	$0.034 \pm 0.01$	$0.057 \pm 0.01$

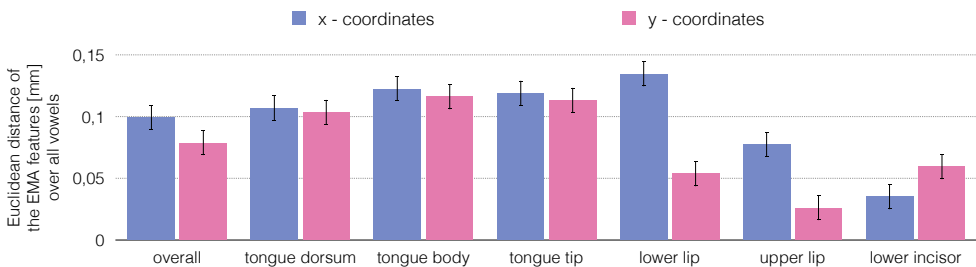
Table 6.2: *Euclidean Distance* in (mm) between EMA positions generated by HTS and pHTS, when combined with articulatory features (6 EMA points) and computed over the *phoneme* set for both x- and y- coordinates, with 95% confidence intervals.

<b>HTS vs. pHTS with EMA features (over vowel set)</b>	<b>Euclidean Distance x-coordinate [mm]</b>	<b>Euclidean Distance y-coordinate [mm]</b>
overall	$0.099 \pm 0.04$	$0.078 \pm 0.05$
tongue dorsum	$0.106 \pm 0.01$	$0.103 \pm 0.01$
tongue body	$0.122 \pm 0.02$	$0.116 \pm 0.03$
tongue tip	$0.119 \pm 0.02$	$0.113 \pm 0.02$
lower lip	$0.134 \pm 0.01$	$0.053 \pm 0.02$
upper lip	$0.077 \pm 0.01$	$0.026 \pm 0.01$
lower incisor	$0.035 \pm 0.02$	$0.059 \pm 0.02$

Table 6.3: *Euclidean Distance* in (mm) between EMA positions generated by HTS and pHTS, when combined with articulatory features (6 EMA points) and computed over the *vowel* set for both x- and y- coordinates, with 95% confidence intervals.



(a) *Euclidean Distance* in (mm) computed over the *phoneme* set.



(b) *Euclidean Distance* in (mm) computed over the *vowel* set.

Figure 6.4: *Euclidean Distance* in (mm) between EMA positions generated by HTS and pHTS, when combined with articulatory features (6 EMA points) and computed over the *phoneme* and *vowel* set for both x- and y- coordinates, with 95% confidence intervals.

As indicated from the obtained results the proposed pHTS system (ST-MLPG algorithm) the generated positions of the EMA points are very close to those generated by the original one (MLPG algorithm). Actually, we see that the overall difference is about 0.09 millimetres when computed on both the *phoneme* and the *vowel* set. Likewise, the results are encouraging, but we can not argue how these distortions are perceived by the users without subjective evaluation, as presented in [Section 6.2.3](#).

It is evident here that the distortions in the segmental quality are higher than those presented in [Section 4.3.2](#). This is not so surprising, since here we introduce an additional feature stream, i.e. the articulatory, that once generated influences the final quality of the output. Hence, probably due to the fact that there is initially some distortion introduced in the generation of the articulatory features themselves, as shown in [Table 6.2](#) and [Ta-](#)

ble 6.3, the generation of spectral and excitation features is influenced in consequence, resulting in higher distortions than expected in the acoustic domain, yet still limited. It is probable, that for better estimation of the articulatory features a larger observation window could be needed than the one used here, containing a larger number of preceding and succeeding phonetic labels. Another factor influencing the quality, could be the conversion of the frequency-warped LSPs into the traditional Mel Generalized Cepstral (MGC). However, this is something that cannot be avoided, without retraining the whole database, since MGCs are essential for the realtime MLSA filtering. Note that both assumptions are not tested in this work.

### 6.2.3 Subjective evaluation

The subjective evaluation of HTS and pHTS along with the articulatory features is conducted via a Comparative Mean Opinion Score (CMOS) test, so that we can evaluate our results using the STRAIGHT or the MLSA vocoder. It is important to clarify here, that there was no cross-comparison between vocoding methods. Here listeners grade the perceived *naturalness* and *intelligibility* of a speech signal in relation to a reference speech signal. The CMOS scale is a 7-point scale ranging from  $-3$  for *much worse* to  $+3$  for *much better* with  $0$  for *about the same*.

A group of 23 people participated to the test comparing HTS to pHTS combined with articulatory features using either the STRAIGHT or the MLSA vocoder. Participants were speech and non-speech experts, native and non-native English speakers. Each test was composed of 24 randomly-chosen sentences. Here the durations were modelled and not forced, since frame-by-frame comparison is not needed. For each sentence, subjects were asked to listen to both versions (randomly shuffled) and to attribute a score according to their over-all preference. It is important to highlight here that there is not cross-comparison between vocoders, since the target of this experiment is not the comparison of the vocoding quality but the comparison of the quality of the generated acoustic features generated by pHTS and HTS.

Table 6.4 shows the preference scores when comparing HTS to pHTS with articulatory features when using either the STRAIGHT or the MLSA vocoder, as well as the overall preference of the used synthesis method. Here positive values indicate preference for the HTS approach. Results are presented for

both *naturalness* and *intelligibility*, with 95% confidence intervals. Figure 6.5 illustrates the results as graphs for easier comparison.

CMOS [7-scale]	overall	STRAIGHT	MLSA
naturalness	$0.224 \pm 0.08$	$0.285 \pm 0.12$	$0.162 \pm 0.10$
intelligibility	$0.214 \pm 0.08$	$0.221 \pm 0.13$	$0.208 \pm 0.12$

Table 6.4: Preference scores when comparing HTS to pHTS along with articulatory features when using either the STRAIGHT or the MLSA vocoder, with 95% confidence intervals. Positive values indicate preference for the HTS approach.

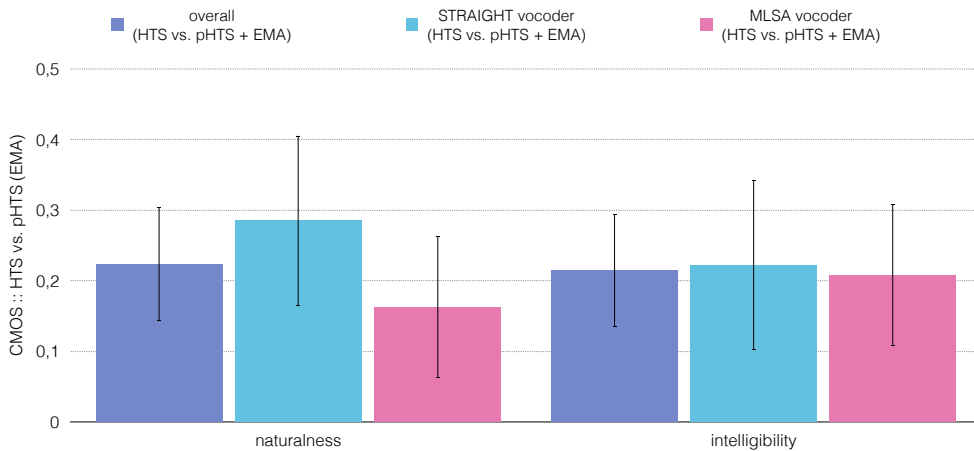


Figure 6.5: Preference scores when comparing HTS to pHTS along with articulatory features when using either the STRAIGHT or the MLSA vocoder, with 95% confidence intervals. Positive values indicate preference for the HTS approach.

It is evident that the scores obtained with this test show that we stay between 0 and 1, i.e. *about the same* and *slight preference* for the standard approach. While being noticeable, it means that the introduced distortions are not stressed by the listeners as a real issue regarding the suprasegmental quality, using either STRAIGHT or MLSA as a vocoding method. This most

likely indicates that introducing the articulatory features slightly degrades the quality when it comes to the reactive approach, however without significant importance for the users. On the other hand incorporating articulatory features introduces a different level of control and flexibility that we think is interesting for its realtime aspect.

### 6.3 REACTIVE ARTICULATORY CONTROL

As detailed in [Section 6.1](#), reactive and continuous control of the articulatory feature stream is feasible, while preserving the intelligibility and naturalness achieved by the conventional HMM-based speech synthesis system [[Section 6.2](#)]. Indeed, until now manipulation of the acoustic features could be achieved only based on prior phonetic knowledge reflected on the articulators. However here, we have shown that the user can influence independently the position of the EMA points for every phonetic label in realtime. This means, that it is possible to manipulate vowels, consonants even full phonetic trajectories simply by changing the values of  $\xi_t$  in [Equation 6.4](#).

#### 6.3.1 *Reactive articulatory control application*

Using the pHTS framework, we present our first attempt to give controllability to the user by means of simple sliders. The aim of this work is to investigate how users interact and control the articulators in order to manipulate the overall characteristics of synthesised speech, as well as the identity of specific vowels. We prototyped an interactive application for the manipulation of all 6 EMA points, where the x- and y- coordinates are controlled by means of an “articulatory” mixer (similar to the one in [Section 5.3](#)), as illustrated in [Figure 6.6](#). As users interact with the system, the influence of the articulatory controls over the acoustic features is gradually perceived. This was a first test case for the reactive control of the EMA points, which is reflected on the final acoustic features, where simple sliders are employed to adjust the initially generated coordinates. Although sliders give a fair visual feedback to the user on how the articulatory space is being browsed, it becomes rather complex since there are 12 degrees of freedom, i. e. the x- and y- coordinates of the 6 EMA points. Moreover these sliders do not really provide a meaningful control or any “physical” feedback to the user. In order to

tackle this, we combine this work with a reactive vocal tract application for better visualisation and more meaningful controls. A complete description of this can be found in [Section 7.1.5](#).

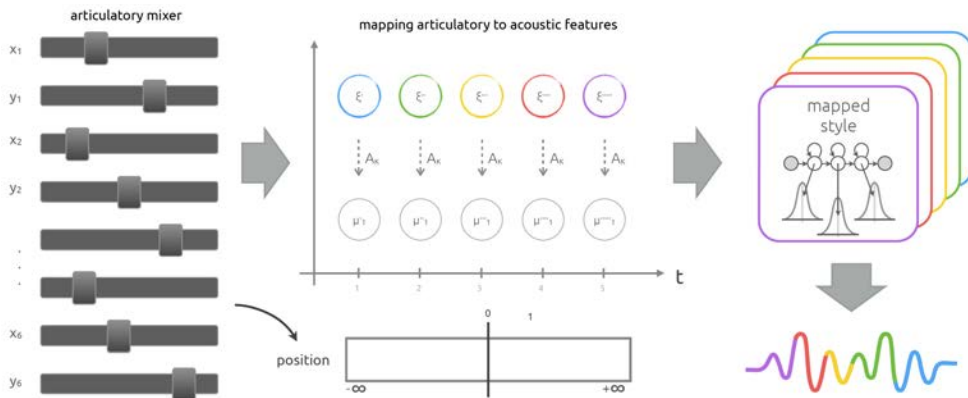


Figure 6.6: Controlling the articulatory features in order to manipulate the acoustic features through a realtime articulatory to acoustic mapping.

### 6.3.2 Challenges

One of the aims of this work is to allow the user to reactively control the speaking style as well as the content by modifying the articulatory features. However, the movement of the articulators is so fast that the user is not able to input the expected movements fast enough through the interface. Therefore, instead of trying to contextually manipulate a full phrase we decided to transform only one vowel into another. This simplifies the problem of the fast changing articulatory position but introduces the problem of the duration of the synthesised vowel. When a single phoneme is synthesised with the standard model duration it is too short to be intelligible. Hence, in this case we synthesise long vowels by increasing the generated duration of every state, using a bigger scale factor for the stable state (e. g. 10, 20 or 50 times longer than the original duration), followed by a long pause. The drawback of this approach is that it increases the latency since we apply controls/computations one phoneme at a time and here the vowel/phoneme duration is scaled up. In other words, setting the duration of the stable state 50 times higher, results in a very long phoneme that need to be processed and gener-

ated before any additional controls may be applied. To tackle this issue we developed a state-by-state generative approach detailed in [Section 8.2.4](#).

As a test case, the user is asked to listen to a vowel synthesised using phonetic labels and by moving the six EMA points reactively from the interface (either sliders or reactive vocal tract) to transform it into another target vowel. However, after some exploratory tests we see that this approach requires from the user to move the six EMA points, in the two dimensional space (12 degrees of freedom) accurately enough so that he will achieve the acoustic target. Such a task is very demanding and rather difficult, and in most cases the user does not reach the target. Moreover, the initial position of the EMA points can be overwritten with any arbitrary position and with any speed, which may result in movements of the articulators that do not always respect the “physiology” and “mechanisms” of the human articulators, and thus the output becomes “unstable”. What makes this task more difficult is that we use context-dependent models. Although the FSS-MRHMM approach can determine the regression matrices without using context information, the  $\mu_j$  and  $\Sigma_j$  as shown in [Equation 6.4](#) are still context-dependent. More specifically, the required modifications over the articulatory features (EMA points) in order to acoustically achieve a target, differ depending on the initial phone synthesised using the provided label. A solution to this would be to use “tailored” context features, where the vowel identities are removed for acoustic model clustering and therefore better articulatory controllability can be achieved [61].

Although it is not straightforward to formally evaluate the proposed interactive control of the articulators and hence no proper tests were performed, we have demonstrated that the manipulation of EMA points in realtime is possible and that it could result in the implementation of several different applications [[Section 7.1.5](#)], i. e. speech therapy and pedagogy or even cross-accent speaker adaptation (e. g. changing a British English accent to an American one) and simulating Lombard effects in synthesised speech as an instant response to environmental noise conditions.

## 6.4 DISCUSSION

In this Chapter we presented a method that enables reactive articulatory control over HMM-based parametric speech synthesis using the pHTS frame-

work. We presented also an application that enables the user to reactively control the position of the articulators through a graphical user interface. We see that reactive articulatory control is feasible, and combined with an interface allows us to explore different aspects of the speech production. Additionally, all the reactive controls described in the previous chapters can be applied here (e. g. prosody controls).

However, we realise that the final segmental quality of the output is more distorted than expected from previous experiments [Section 4.3.2]. We assume that the acoustic features themselves are in principle generated with minimum distortions, however probably due to the additional articulatory feature stream the error increases. In other words, since the realtime prediction of the position of the EMA introduces some distortions, these distortions are reflected in the final speech output through the articulatory-to-acoustic mapping, consequently the output quality degrades. Better prediction of the EMA points could be achieved by using a different sliding window, containing a larger number of preceding and succeeding phonetic labels. However such an assumption needs further investigation. It would be also interesting to examine cases of reduced and full context regarding both training and synthesis while incorporating the articulatory features so that we estimate the influence of the voice model itself. Similarly, such a hypothesis demands further research. Results could also be improved by retraining the whole models in order to use Mel Generalized Cepstral (MGC) and not frequency-warped LSPs, so that any conversion error is avoided. Although users tend to distinguish between standard and performative HTS when incorporating articulatory features, listeners tend not to stress the introduced distortions. Actually, the obtained scores show a slight preference to the standard method. Even though pHTS slightly degrades the final quality of the output it brings a different level of control and flexibility for the user.

Furthermore, we realise that the manipulation of the articulators by the user, even though it seems rather straightforward, is very demanding and difficult. It is very easy to transform a phone into another random phone while experimenting with the interface, but it becomes rather complicated when a specific target vowel modification is asked. Though, every time this work was publicly demonstrated [AMY<sup>+</sup>13a], [Section 7.1.5] users tended to enjoy their funny results and also tried to drive the system to its limits. There are aspects of the system that would benefit from improvement. Currently, there are no restrictions over the user manipulation patterns, but probably

limiting or “guiding” the possible user controls might lead in more distinguishable vowel modifications. It is important to either decrease the degrees of freedom available to the user or to allow the manipulation of only some EMA points while the system provides the correct coordinates for the remaining ones. For example, a case would be where the user is allowed to manipulate only the three EMA points over the tongue, while the remaining three are automatically adjusted. However, such a simplification of the interface must by all means be combined with using “tailored” context feature during synthesis for better articulatory controllability.

Based on such a framework, it would be meaningful to conduct user studies and listening tests. The user studies will show us how users manipulate the acoustic space by means of articulatory control as well as how skilled a user should be. The listening tests will help us measure how other listeners perceive the result of these manipulations. Initially, as explained above, the user would be asked to transform a given vowel into a target vowel only by controlling the articulatory features. The success of the user will be determined by objectively evaluating the acoustic and articulatory features generated by taking into account (or not) the user input for the target vowel. Then, it would be interesting to see how other users perceive these modifications, and in addition to the objective evaluation, we would like to perform also some listening tests to subjectively evaluate performance on the vowel modification task. Note here that this is an ongoing research and more implementations and evaluations will follow.



## Part III

### PERFORMATIVE DESIGNS

III	PERFORMATIVE DESIGNS	113
7	PERFORMATIVE APPLICATIONS	115
7.1	Speech related prototypes . . . . .	116
7.2	Beyond speech . . . . .	132
7.3	Potential applications . . . . .	137
7.4	Discussion . . . . .	137
8	MAGE/PHTS: PERFORMATIVE HMM-BASED SYNTHESIS LIBRARY	139
8.1	Realtime architecture of MAGE . . . . .	140
8.2	Reactive controls . . . . .	142
8.3	C++ API . . . . .	146
8.4	Discussion . . . . .	148



# 7

## PERFORMATIVE APPLICATIONS

---

### Contents

---

7.1	Speech related prototypes . . . . .	116
7.1.1	Reactive speech synthesis controlled by hand gestures . . . . .	116
7.1.2	Reactive speech synthesis controlled by facial gestures . . . . .	118
7.1.3	Talking guitar . . . . .	118
7.1.4	Accent interpolation through an interactive map . . . . .	120
7.1.5	Realtime articulatory control through a reactive vocal tract . . . . .	122
7.1.6	Reactive audiobooks . . . . .	124
7.1.7	Speaking & singing puppet . . . . .	128
7.1.8	Tanukis . . . . .	131
7.1.9	Other cases . . . . .	131
7.2	Beyond speech . . . . .	132
7.2.1	Reactive laughter synthesis . . . . .	132
7.2.2	Reactive stylistic walk synthesis . . . . .	133
7.2.3	Reactive singing synthesis . . . . .	134
7.2.4	Incremental speech synthesis . . . . .	136
7.3	Potential applications . . . . .	137
7.4	Discussion . . . . .	137

---

In [Part II](#) we see that performative control is possible in the HMM-based synthesis framework. However, it is important to test the proposed pHTS framework in real applications. Thus, we built many creative test cases in order to prove the contribution of pHTS. Our work though has also motivated others to build numerous prototypes serving their own purposes. We see that performative synthesis is indeed needed in various domains. It engages the user, and therefore enriches the various synthesised results in ways not possible with the standard framework, making it a useful tool.

The structure of this Chapter is as follows: in [Section 7.1](#) we present several examples related to speech synthesis while in [Section 7.2](#) beyond speech

prototypes are detailed. In [Section 7.3](#) we discuss potential application frameworks of pHTS, and finally in [Section 7.4](#) a summary of the Chapter is presented. Several video demos can be found in [Section B.2](#). It is important to highlight that the applications presented here make use of the MAGE/pHTS library as detailed in [Chapter 8](#).

## 7.1 SPEECH RELATED PROTOTYPES

Most of the prototypes presented in this section are built as a proof of concept attempting to reactively control the synthesised speech. They mainly focus on prosodic and contextual controls of the generated speech through hand gestures. Moreover, we have cases that target real-world applications, such entertainment platforms, and not just proofs of concept.

### 7.1.1 *Reactive speech synthesis controlled by hand gestures*

The first application of our work is to reactively control speech through hand gestures<sup>1</sup>, as published in [[ABd<sup>+</sup>11a](#)],[[ABdD11](#)]. The controller we choose for that purpose is HANDSKETCH [[89](#)], a new musical instrument prototype that uses a graphic tablet. We use the pen-based gesture controls of HANDSKETCH in order to manipulate in realtime the prosody and speech quality of the generated speech. The polar coordinates on the fan (angle, radius), control pitch and speed respectively. Angular movements of the pen control the generated pitch of every sample. It is possible to simply overwrite the pitch or deviate it by a ratio given by the angle. Similarly, moving the pen along the radius can accelerate or decelerate speech. Pen pressure controls speech intensity and pen x tilt, i. e. the inclination of the pen relatively to the tablet along the x axis, modifies the vocal tract length. [Figure 7.1](#) shows the mapping of the HANDSKETCH control space to the speech parameter generation space of pHTS.

Attempting to bring also contextual controls for the user we take advantage of our work in [Section 4.2](#) and we create a very simple reactive natural language processor (RNLP) module. This RNLP module takes as input phonemes passed from the user (e.g. by tapping a touch screen), which are then parsed into alphanumeric phonetic labels, that contain only the re-

---

<sup>1</sup> HandSketch demo (visited June 2014): <https://vimeo.com/39558917>

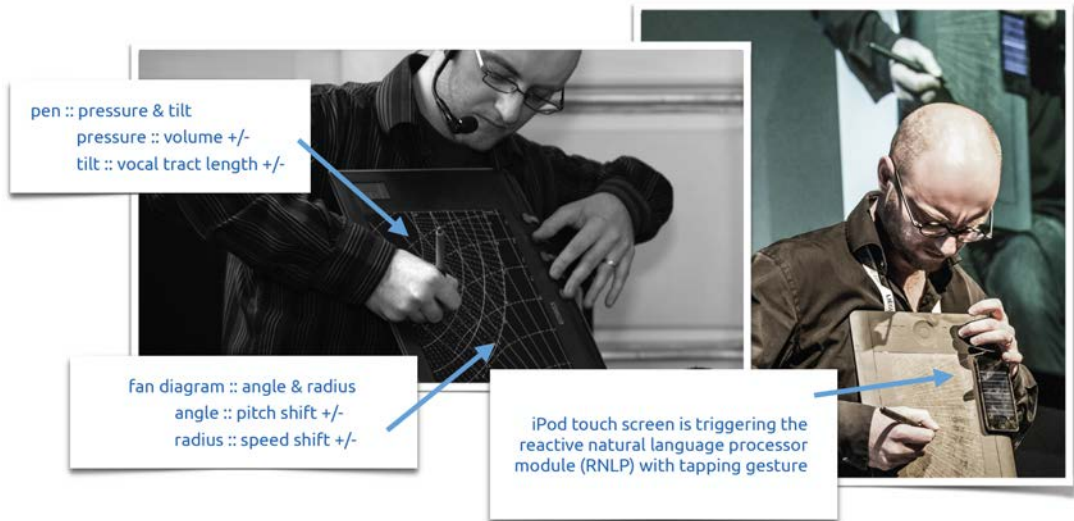


Figure 7.1: Mapping of the HANDSKETCH control space to the speech parameter generation space of pHTS.

duced set of linguistic factors. Additionally, new HMM voice models had to be trained on that reduced set of linguistic factors, as illustrated in [Figure 4.3](#). Finally, these phonetic labels composed on-the-fly are then passed as contextual input in order to produce the targeted speech samples while using the reduced-context models. Force Sensing Resistors (FSRs) on the side of HANDSKETCH were initially used to trigger the the RNLP module. Later, the FSRs were replaced by an iPhone/iPod, so that every time the user tapped on the touch screen the RNLP module was triggered. Combining pHTS, HANDSKETCH and the RNLP, we have a first prototype, that enables on-the-fly control of phonetic context, prosody and voice quality at the same time in the framework of performative HMM-based speech synthesis.

It is important to note here that prosody and voice quality can be controlled rather easily and intuitively by the user, while contextual control is very demanding. Every time this work was publicly presented, users seemed to engage and enjoy manipulating the voice creating really unexpected and amusing outcomes. However, the contextual control was kindly avoided due its complexity.

### 7.1.2 *Reactive speech synthesis controlled by facial gestures*

Along the lines with our work in [Section 7.1.1](#), we create a similar application where the generated speech is reactively controlled by face gestures<sup>2</sup> instead of hand gestures. This work was published in [\[AdD12b\]](#). Based on realtime face tracking (FaceOSC [\[90\]](#)) we convert some realtime-measured facial features into speech synthesis features such as prosody and context. In this case the head tilt controls the speed and pitch shift of the output while the opening of the mouth triggers the RNLP module. By tilting the head right or left the speed of the generated speech is accelerated or decelerated, respectively. When tilting the head up or down the generated pitch trajectory is shifted up or down. If the opening of the mouth is detected, a random phoneme is triggered and processed by the RNLP. [Figure 7.2](#) shows an instance of the realtime face tracking along with the presented controls. Other mappings are possible, such as the one proposed in [\[dAD13\]](#).

Once again it is a very funny application, but we realise that neither on prosody nor on context can we have accurate control of the generated speech due to the nature of the face and head movements, which are indeed not fast enough and strongly interdependent. However, we could argue that if we were able to recognise some facial expressions and map these expressions to control the synthesised speech, e. g. interpolation of emotions or styles from expressive voice databases, then the facial expressions could be transcribed into artificial speech.

### 7.1.3 *Talking guitar*

Recruiting an electric guitar as a controller and using algorithms to detect several different guitar playing techniques [\[91\]](#) is a more artistic application of our work, presented in [\[AdR<sup>+</sup>13\]](#), [\[ARM<sup>+</sup>12\]](#). A variety of musical gestures could be used to create a “talking guitar”<sup>3</sup>, however here we focus on continuous (e. g. bend, slide, plucking point) and triggering gestures (e. g. pitch, note on, note off, hammer-on, pull-off, palm muting, natural harmonic) that we combine with appropriate mapping, in order to control the synthesised speech.

---

<sup>2</sup> FaceOSC demo (visited June 2014): <https://vimeo.com/39567236>

<sup>3</sup> Talking guitar live demo (visited June 2014): <https://vimeo.com/100508133>



Figure 7.2: Reactive speech synthesis controlled by facial gestures. The head tilt controls prosody while the mouth opening triggers the RNLP module.

Any new attack triggers three consecutive precomputed full-context phonetic labels in order to keep a correlation between guitar sound and speech. Empirically we see that triggering one label with every attack is not very appropriate: the sound corresponding to one label is too short. However, series of three labels give the guitarist a reactivity that matches closely its playing. Also, mapping one-to-one both amplitudes and both pitches (i. e. guitar amplitude is mapped to voice amplitude and respectively for pitch) forces the melody and the voice to be two tied discourses dependent on the guitarist's will. Thus, using series of labels gives the guitarist a wider range in the guitar/voice relationship. Longer series of labels are used to enable the guitarist to e. g. play different notes or add certain effects while the text was being pronounced. Moreover, certain notes played on a predefined string are used to trigger labels. It is important to highlight here, that using syllable-by-syllable context control could be more meaningful rather than triggering 3 sequential labels at a time. However, since *regular expressions* [Section 8.2.2] were still not introduced in the framework of pHTS, refined context-based controls were still not possible.

In order to control the prosody and speech quality of the generated speech, we use the vocal tract length and speed parameters as well as continuous

data extracted from guitar detection. Here, “bends” are mapped to vocal tract length and speed. Bending the string can be configured by the guitarist either to control the voice quality or to slow down the generated speech. Other parameters such as duration and pitch were modified directly during the performance by means of simple sliders, but no mapping with the guitar has been tried yet. [Figure 7.3](#) the performance setup of the “talking guitar” project is illustrated.



Figure 7.3: The performance setup of the “talking guitar” project where speech synthesis is interactively controlled by an electrical guitar.

#### 7.1.4 Accent interpolation through an interactive map

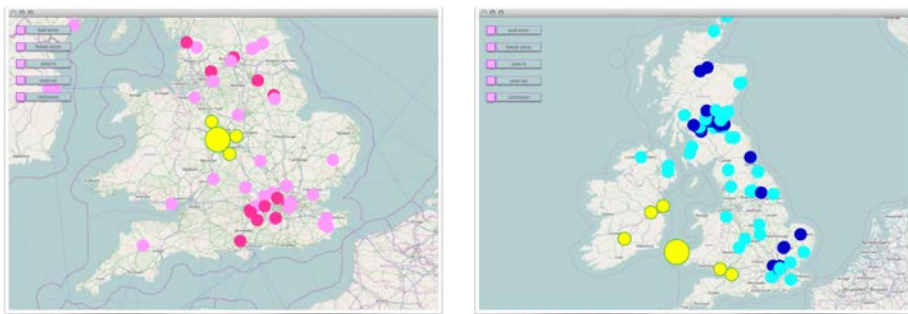
Large scale reactive model interpolation [[Section 5.1.3](#)] motivated us to prototype an interactive map application<sup>4</sup> where several geolocalized accents of English are presented and the user is able to navigate through the map, select various accents and interpolate them, as it was published in [[AYdD13](#)].

More specifically, every accent is represented as a coloured circle on the map (blue for male, pink for female) based on the provided geographic locations. It is possible to select the gender of the voice and alter it during synthesis. The user operates an “active” region, represented as a yellow circle around the cursor, in order to select accents to be interpolated. The map operation is achieved with standard hand gesture controls. There are zoom in and zoom out functions, as well as navigational functions, in order to

<sup>4</sup> Interactive accent map demo (visited June 2014): <https://vimeo.com/51985913>

move across the map. These controls are available with mouse scroll up or down and with clicking and dragging respectively. A touch screen can be employed to provide a more intuitive user experience.

There are two interpolation strategies: *collision* and *continuous* that the user can also alternate in realtime. Both strategies are presented in Figure 7.4. During the *collision* interpolation mode, when the yellow “active” region overlaps with one or more accent circles, the corresponding accents are selected, linearly interpolated and synthesised interactively [Section 5.1]. When the cursor does not overlap with any accent, a default accent is used. During the *continuous* mode, each time the cursor moves, the distance between the cursor and all the available accents is computed and the N-nearest neighbouring accents are selected and interpolated. Note here, that if N is set to high value (i. e. more than 6), then there is no significant differences while changing the selected accents. The final output sounds as if it is an “averaged” accent, [Section 5.3]. In both interpolation strategies we use uniform weights of  $w = 1/N$  for each accent and for each feature stream (i. e. spectrum, pitch and duration), although more complex interpolation schemes may be adopted. It is important to note here, that interpolating between several voice models is a computationally heavy process, which can lead to saturation and delays in the final speech output. Thus it should be avoided to set N to very high values (i. e. more than 15).



(a) Interpolation of female accents using the *collision* strategy with  $N = 3$ . (b) Interpolation of male accents using the *continuous* strategy with  $N = 5$ .

Figure 7.4: Interpolation of female and male accents through an interactive map, using either *collision* or *continuous* interpolation. The larger circle is the “active” region while the smaller yellow circles are the selected accents.

Every time this work is presented to the public we see that people have fun experimenting and listening to accents, trying to identify some of them or even create their own, especially native speakers. We see here that the manipulation of accent in realtime is possible and that it could result in the implementation of several real life and scientific applications. For example it could be useful in anthropological studies, linguistics, cultural patrimony and voice personalisation. However further research is required.

#### 7.1.5 *Realtime articulatory control through a reactive vocal tract*

In an attempt to facilitate more meaningful user controls as well as to enhance our work presented in [Chapter 6](#) and published in [AMY<sup>+</sup>13b], we created a reactive vocal tract application<sup>5</sup>. This application depicts a two dimensional midsagittal view of the human vocal tract. Every articulator is presented with a different colour, along with six electromagnetic articulography (EMA) points represented as white circles placed on the articulators, as described in [60]. The position of these EMA points can be controlled by the user using a mouse or touch screen, while the represented vocal tract reacts and adjusts its shape accordingly.

The user is also provided with a list of predefined vocal tract shapes and EMA sets. These predefined configurations of vocal tract shapes were obtained from speaker-dependent magnetic resonance imaging (MRI) scans and EMA [92]. When the user selects one of these vocal tract and EMA configurations, his previous controls are instantly overwritten and the selected shape is displayed, as illustrated in [Figure 7.5](#). Then, by selecting one EMA point, the user is able to move it in the two dimensional coordinate space. Note here that the user controls take place in the MRI space and not in the EMA space that we use for the performative synthesis. This means that the controls have to be appropriately transformed in order to be correctly synthesised. This is achieved by means of specific transformation matrices, where EMA points are shifted and rotated to MRI coordinate space as explained in [92]. By default, the current shape of the vocal tract is reactively altered so that the user can have not only audio but also visual feedback of the applied controls. However, it is also possible to have a static vocal tract, so that the user can have a reference point to the initial configuration chosen.

---

<sup>5</sup> Reactive generation of articulatory features demo (visited June 2014): <https://vimeo.com/67404386>

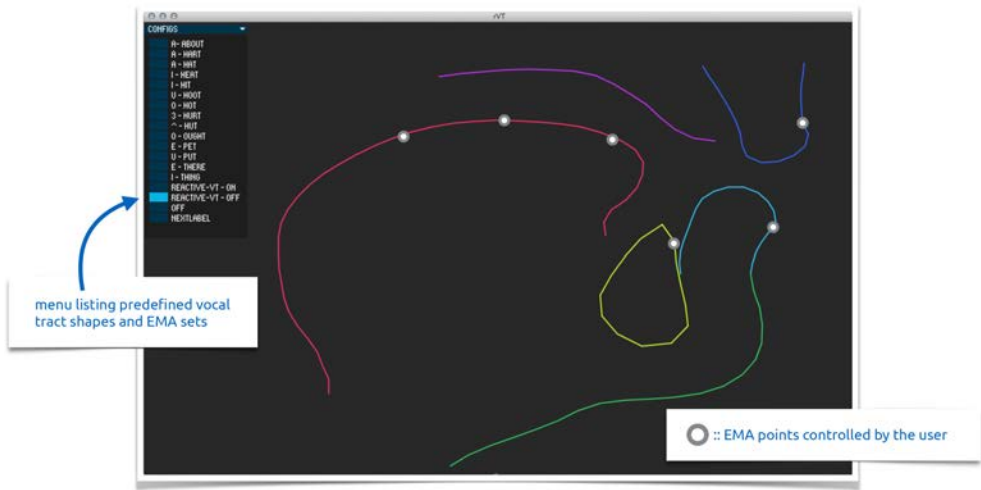


Figure 7.5: Realtime control of the articulators through continuous manipulation of EMA points and use of predefined configurations of vocal tract shapes.

There are no limits to the possible position of the EMA points, allowing the user to move the six EMA points, in the two dimensional MRI space (12 degrees of freedom). This means that the user is free to place these points in coordinates that are “unnatural” either from a physical point of view or as sequence of movements. This results in movements of the articulators that do not always respect the physiology and mechanisms of the human articulators. Indeed, it is probable that the input movements have not been “seen” during the training phase. Hence, the models cannot accurately estimate them and will/may give unnatural/unrealistic results.

The aim of this application is to allow the user to reactively control the speaking style as well as the context by modifying the articulators themselves though a visual representation of the vocal tract. However, the movement of the articulators, is so fast that the user is not able to input the expected movements fast enough. Additionally, the “correct” or “expected” position of the articulators is not always evident. As a test case, the user is asked to listen to a vowel synthesised using phonetic labels and to transform it into another target vowel by moving the six EMA points from the interface. However, after some exploratory tests we saw that this approach requires the user to accurately control the six EMA points in order to achieve the acoustic target. Such a task is very demanding and rather difficult, and

in most cases the user does not reach the target [Section 6.3]. It is important to either decrease the degrees of freedom available to the user or to allow the manipulation of only some EMA points while the system provides the correct coordinates for the remaining ones. For example, a case would be where the user is allowed to manipulate only the three EMA points over the tongue, while the remaining three are automatically adjusted. Providing a “colour-coded map” denoting the “accepted” or “suggested” regions of every EMA point could advice or guide the user to choose suitable coordinate sets. This would help the user to have a better understanding of the required modifications of the articulators in order to achieve the acoustically desired target. As a result of this proof of concept, we see that the control of the articulators in realtime is technically possible<sup>6</sup>. With further adaptation and development we think it could be used for the implementation of real life applications for voice pedagogy and therapy.

#### 7.1.6 *Reactive audiobooks*

The application *reactive audiobooks* is a novel entertainment platform for realtime customisation of an audiobook, proposed by the *National Institute of Informatics (NII)*. *Reactive audiobooks* relies on both the performative and standard HMM-based speech synthesis systems. Here audiobook listeners can not only listen to novels but also can enjoy to create attractive and vivid audiobooks interactively by themselves. The platform allows voice and individual sentence customisation as well as sharing these customisations among users. This is an ongoing project started in 2014, therefore extended details and exact description of the full platform will be given in future publications. However, a first design of the *reactive audiobooks* application, its targets and voice personalisation crowd sourcing possibilities are presented here.

The user is able to access and control six main functions from the top of the application, as presented in Figure 7.6. Starting from left to right we have: *casting*, *customise voice*, *customise sentence*, *add new voice*, *add new book* and *sound*. Additionally, as shown on the right frame of the application, the option *book name* allows the user to select from a variety of audiobooks and have access to its full description (e. g. text, characters, etc.). Finally, the

<sup>6</sup> We also see that when users have the chance to play with the reactive vocal tract they have fun and in several cases try the most “unnatural” combinations “just to see what happens”

main frame of the application is dedicated to the controls for each of the main functions mentioned.

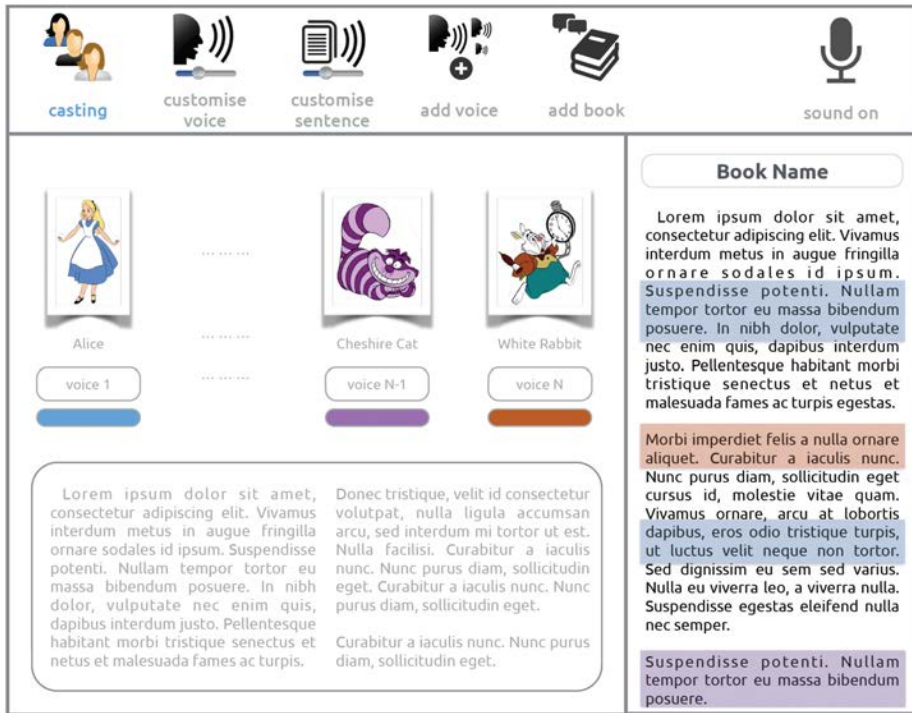
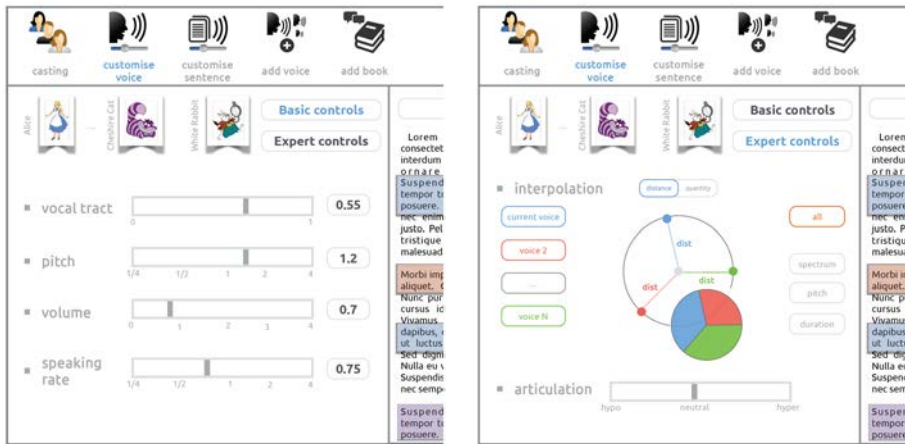


Figure 7.6: Casting the voices of a book's characters.

The first step for customisation comes with the *casting* option that allows the user to assign a different voice to every character of the book as well as the narrator, as illustrated in Figure 7.6. Assigning a voice can be an arbitrary action or based on prior knowledge of the physiology and personality of the character. Once the user has completed the *casting* of the book, all the controls are saved in an XML file that describes the customised audiobook. Here, the user also has access to various information about the assigned voices, such as voice descriptions (e. g. breathy, low, etc.) and various statistics.

The *customise voice* options enables the user to customise a voice currently assigned to a character. There are two ways of achieving such a customisation, either using *basic*, "high-level" controls or using *expert*, more accurate, "low-level" controls. *Basic* controls, presented in Figure 7.7a allow ma-

nipulation via simple sliders over the pitch, volume, vocal tract length and speaking rate of a given voice. On the other hand, *expert* mode, presented in Figure 7.7b provides more complex controls, such as control of the articulation degree and voice interpolation [Section 5.1.1]. For the control of the articulation degree we use also a simple slider [Section 5.2], but for the voice interpolation we use a circle to define the interpolation weights of the feature streams [Section 5.3]. The user selects the voices for interpolation and then the visual strategy, *distance* or *quantity* in order to set the interpolation weights. In the first case the voices selected for interpolation are presented as points on the circumference of a circle. By moving the control point (grey) initially placed in the centre, it is possible to specify the distance between all the voices, and accordingly set the weights, either for all or just specific feature streams. In the latter case, the voices selected for interpolation are presented as sectors of the circle. Similarly, by resizing these sectors the weights are set. Finally, all these controls, both *basic* and *expert*, are saved in an XML file that describes a new voice. This voice can be assigned to any character and be shared among different users or books. Note here that by voice we mean either a standard voice model or an XML that describes certain modifications of a standard voice model.



(a) *Basic* customisation mode allow “high-level” controls over the pitch, volume, vocal tract length and speaking rate.

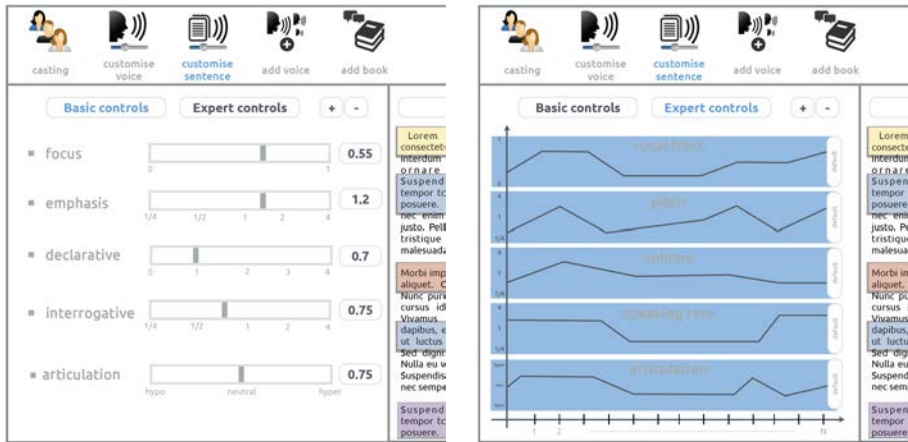
(b) *Expert* customisation mode allow “low-level” controls over the articulation degree and voice interpolation.

Figure 7.7: Customisation of a character’s voice.

The *customise sentence* option enables the user to customise a given sentence of a book. It is very similar to the *customise voice* option but in this case we focus on crafting the pronunciation style of a single sentence and not a voice. Such a customisation can be achieved by using *basic* or *expert* controls over a sentence, as illustrated in [Figure 7.8](#). First, *basic* controls enable the user to manipulate the overall “style” of a sentence via simple sliders. So a typically neutral sentence can be more focused, more emphatic, hypo- or hyper- articulated, etc. *Expert* mode provides more refined, time-varying controls over a selected sentence. It is possible to select the contextual level over which the controls will be applied (e. g. word, syllable, phoneme) and then create a modification trajectory to control the vocal tract, pitch, volume, speaking rate and articulation degree. In other words, all these parameters can be modified within a given sentence and create personalised pronunciations, expressions and styles. Then, all these controls over a given sentence are saved in the XML file of the customised audiobook.

The *add voice* function allows the user to record and therefore create a voice based on his own, share it with other users or share/download other customised voices. The *add books* option enables the sharing and downloading of complete *reactive audiobooks*. Finally, the *sound* option simply turns on and off the audio.

In brief, through this application the user can reactively customise an audiobook in terms of the characters’ voices and pronunciation “style” of individual sentences. The use of pHTS is essential here since it will provide the necessary instant feedback to the user for the best possible fine tuning (e. g. when tuning a voice by interpolation). The *reactive audiobooks* project, apart from its entertainment aspect, aims to bring also valuable insights in terms of speech manipulation and control. Which speech parameters are mainly controlled by the users, how and to which extent? How are sentences customised, at which phonetic level and with which parameters? How do we define “high” level controls such as *focus* or *emphasis* for a sentence? Which voice is more frequently modified, which voice is more frequently assigned to which character? What is the correlation of the assigned voice to the personality or physiology of the character? How do different users customise the same audiobook? What kind of books tend to be customised? Do users collaborate in customising a single audiobook? All these questions may benefit from the crowdsourcing aspect of the *reactive audiobooks* platform.



(a) *Basic* customisation mode allow “high-level” controls of the overall “style” of a sentence, such as emphasis, focus, articulation, etc.

(b) *Expert* customisation mode allow “low-level”, more refined controls of the overall “style” of a sentence by means of modification trajectories.

Figure 7.8: Customisation of a specific sentence.

### 7.1.7 Speaking & singing puppet

Our work has also motivated others to build new applications. In [CKAY13] Clark presents an approach where the Kinect sensor [93] is employed as a controller for reactive speech synthesis. There, the skeleton tracking of a Microsoft Kinect sensor is mapped to control rather simple aspects of the generated speech, such as pitch and duration. The skeleton tracking of the Kinect detects several parameters but here the focus is on the left and right hand movements of the user. In case there is more than one user, the person closest to the Kinect is being tracked. The vertical axis of the right hand shifts the generated pitch of pHTS by bias values and the vertical axis of the left hand controls pre-defined sensible speaking rate values. Figure 7.9 shows the use of the presented application.

As with other applications of pHTS, a formal evaluation is not straightforward. However, both children and adults seem to enjoy interacting with the speech control via the Kinect. This work was displayed at public exhibition spaces in *City Screen York*, *Sheffield Winter Garden* and *Hull - Hull Truck Theatre* in the context of the *Articulate: The Art and Science of Synthetic Speech* exhibi-



Figure 7.9: Kinect skeleton tracking controlling the prosody of pHTS. Left hand controls pitch and right hand the duration of the synthetic speech.

tion organised by the *Creative Speech Technology (CreST) Network* [94]. The media coverage for this *Articulate Roadshow* involved BBC and BBC Radio as well as magazines and newspapers<sup>7</sup>.

Later, this work was enhanced by Veaux [VAO<sup>+</sup>13]. Initially, the raw skeleton used in [CKAY13] was replaced by an interactive avatar, a puppet, as shown in Figure 7.10a. The puppet is rendered by a realtime avatar animation software, Animata [95]. In other words, the tracked skeleton is no longer displayed but it is used to drive the puppet and control the prosody of the generated speech. This prototype was part of the *Edinburgh College of Art - Edinburgh Neuroscience Art/Science competition*, where it won a prize during the local multimedia exhibition. Figure 7.10b shows people playing with the puppet and controlling the prosody of the generated speech.

Further on, in [VAO<sup>+</sup>13] Veaux uses this puppet to control singing synthesis, similarly by mapping prosodic parameters to gestures and body posture of the user tracked by a Microsoft Kinect sensor. In this case, pHTS was slightly modified in order to support vibrato along with singing speed, fundamental frequency (F0) and vocal tract length (VTL) controls. The vertical

<sup>7</sup> Media coverage for the Articulate Roadshow (visited June 2014):

<http://crestnetwork.org.uk/page/articulate-media>



(a) realtime rendered puppet replacing the raw Kinect skeleton. (b) People at the *Edinburgh College of Art - Edinburgh Neuroscience Art/Science competition* controlling the prosody of the generated speech through the puppet.

Figure 7.10: Puppet as speech & singing synthesis controller.

axis of the right hand overwrites the vocal tract length of the generated samples and the vertical axis of the left hand shifts the generated pitch. The movement of the user on the horizontal axis controls the singing speed and the distance from the Kinect sensor controls the vibrato. [Figure 7.11](#) shows the mapping of the skeleton controlling the singing synthesis.

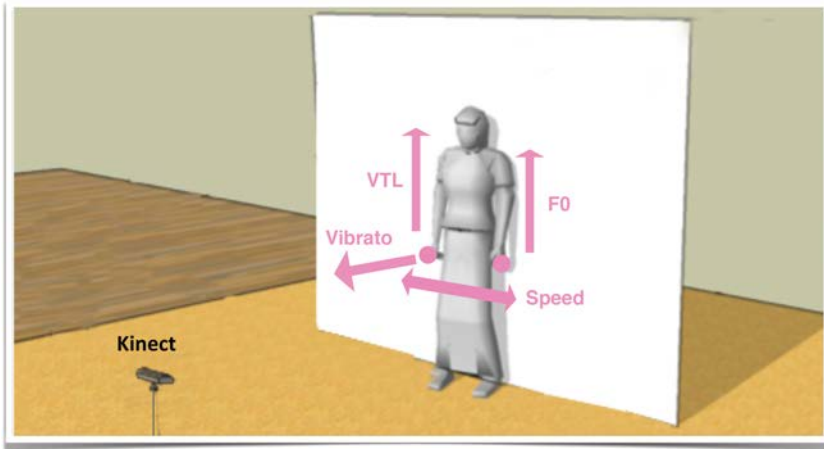


Figure 7.11: Kinect skeleton tracking mapped to control the prosody of pHTS singing synthesis [VAO<sup>+</sup>13].

### 7.1.8 *Tanukis*

Further on, *Tanukis* [96] is a game proposed by F. Zajega and is currently under development. The *Tanukis* project<sup>8</sup> aims at investigating the possibility of an emotive relationship between a human and a virtual avatar. In the Japanese mythology, tanukis or tanukibayashi, are mischievous spirits like badgers that have the ability to change shape at will. In the game, Tanukis are polymorph gentle spirits that have many possible presentation “forms”. Our work on pHTS will be used in the game in order to give voice to the Tanukis, and they will be able to reactively alter their speaking style depending on how the player interacts with them. Figure 7.12 shows instances of the project.



Figure 7.12: Instances of the Tanukis game.

### 7.1.9 *Other cases*

In his master thesis Abelman [97] uses pHTS in order to create contrastive or general emphasis in a sentence as well as to form various question types<sup>9</sup>. Again here, the Microsoft Kinect sensor is employed as a controller. In brief, a hand gesture is used to trigger a combination of pitch and speed shift so that the user can transform an initially neutral sentence into a question or to emphasise certain parts of it.

A SuperCollider [98] plug-in for performative HMM-based speech synthesis and incremental text input was implemented by Puleo in [99]. SuperCollider is an expressive dynamic programming language that can be used

<sup>8</sup> Tanukis demo (visited June 2014): <https://vimeo.com/76850234>

<sup>9</sup> Altering speech synthesis prosody through realtime natural gestural control demo (visited June 2014): <https://vimeo.com/72305725>

for live coding (i. e. the performer modifies and executes code in realtime throughout the performance). Such a work is important since it broadens the audience of our work and opens the doors to more users and therefore more ideas and applications in various domains.

## 7.2 BEYOND SPEECH

Performative HTS was initially developed mainly to generate speech-related features, as in the original HMM-based synthesis system. However pHTS, as the original system, is able to reactively generate any type of parameters when compatible HMM-based models are provided [Section 3.3]. In this Section we describe briefly the applications of pHTS outside of the speech framework, such as HMM-based synthesis of singing, laughter or walk. Also we will examine the possibility of realtime text input. However this is part of future experimentation and validation of the performative feature generation, and therefore no results are presented at this stage.

### 7.2.1 *Reactive laughter synthesis*

Audio-visual laughter synthesis falls out of the scope of the initial application and design of pHTS. However reactive control of simultaneously generated laughter and corresponding facial expressions is a very interesting case. Here, we briefly explain our involvement in combining pHTS with the modelling of the laughter synthesis, both for sound and face motion. In [66], Urbain has recently addressed the HMM-based acoustic laughter synthesis problem. The AV-LASYN database [72] was used to train acoustic and visual models of laughter. These full-context audio-visual models were used in the pHTS framework, using the proposed ST-MLPG without any reduced-context model retraining [Section 4.1].

In the first case the acoustic models were used to generate laughter sound samples, and the generated pitch was shifted by means of a simple slider. By similar means, the visual models are used in the pHTS framework in order to synthesise facial motion. The generated parameter trajectories are projected into the original 3D space so that a rigged 3D face model can be rendered by Blender [100] in realtime. Here, as a proof of concept, a succession of neutral and laughing faces are synthesised in a loop while

the user is able to control the intensity of the visual laughter in realtime. A simple slider is used in order to amplify or attenuate the generated trajectory dynamics. Figure 7.13 shows the reactive visual laughter control in Blender. Such an approach, if very simple, shows that audio-visual laughter synthesis is feasible through pHTS. However, further investigation is required in order to have a better understanding of the performative controls over the laughter synthesis. Such a task will probably be included in the ongoing dissertation of Cakmak<sup>10</sup>. Preliminary tests of the reactive laughter synthesis show that the quality of the short-term generated output is not noticeably different from the one synthesised by the original approach. All the controls available in pHTS can be applied during audio-visual laughter synthesis, allowing the generation of different laughing styles and facial laughing expressions. However, objective or subjective tests need to be conducted.

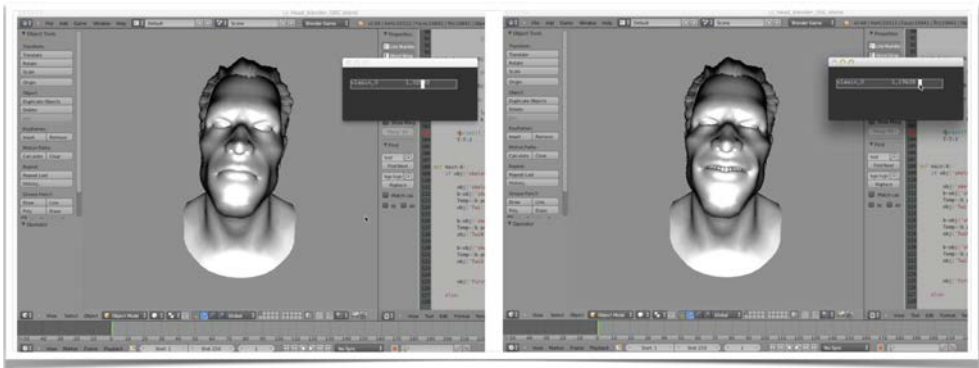


Figure 7.13: Reactive control of laughter intensity of a 3D face model in Blender.

### 7.2.2 *Reactive stylistic walk synthesis*

In her work, Tilmanne shows how expressive gait synthesis was introduced in the HMM-based synthesis framework [67]. The Mockey database [73], a stylistic gait motion-capture database was used to train eleven styles (proud, decided, sad, cat-walk, drunk, cool, afraid, tiptoeing, heavy, in a hurry, manly). Additionally, [101] presents the continuous “style” control of stylistic walk

<sup>10</sup> AV-LASYN : Audio-Visual Laughter Synthesis (visited June 2014):  
<http://tcts.fpms.ac.be/cakmak/personal/>

synthesis through linear interpolation in the HMM-based synthesis framework. With the eleven gait models, it is possible to compute a “neutral” style trained on all the styles. Then walk sequences can be generated that display either one of the styles present in the training database or the neutral one. The model parameters space is considered as a continuous stylistic space, where the values corresponding to each recorded style can be interpolated by means of exaggeration, inhibition and inversion [Chapter 5] and create new walk style models, that were not initially included in the database. In this approach the authors enable to browse the complete stylistic walk space. However, the control of the style and the walk synthesis is still using the standard non performative parameter generation, that rules out interactive user exploration.

In order to tackle this lack of user interactivity, the style and gait synthesis was brought in the pHTS framework. The described full-context gait models were used in the pHTS framework, using the proposed ST-MLPG without any reduced-context model retraining [Section 4.1]. We participated in the development of a reactive gait style exploration application that enables the user to control and browse the style of the synthesised walk by means of interpolation in realtime, as presented in [dTA<sup>+</sup>13]. The product of the user controls over motion parameter sequences is also visualised in realtime in Blender [100]. More specifically, while the application synthesises an infinite walk sequence (a loop of left and right steps) the user browses the stylistic space in realtime, through a set of sliders controlling the influence of each original style by actually setting the interpolation weights of every style, as illustrated in Figure 7.14. The modified walk trajectories are rendered by means of a virtual 3D character in Blender in realtime and thus the user has a direct visual feedback of the input controls. This proof of concept application opens the doors to many possibilities as the size of motion-capture databases nowadays explodes and more and more applications seek new possibilities for exploring motion style or comparing motions.

### 7.2.3 *Reactive singing synthesis*

Many similarities exist between speech and singing voice, though the differences are significant, especially for analysis and modelling. However, HMM-based Singing Voice Synthesis System (Sinsy) [65] has already been pro-

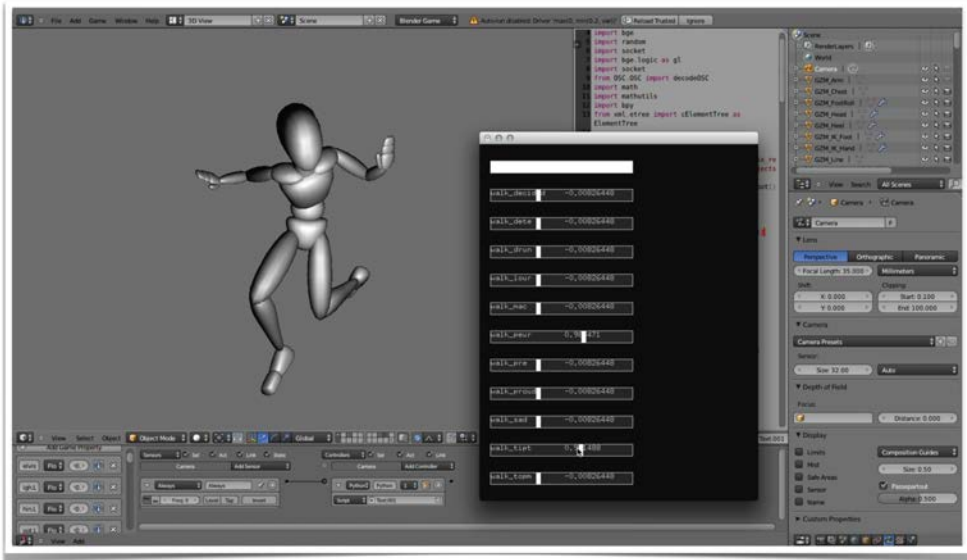


Figure 7.14: Reactive control of the style of the synthesised walk by means of interpolation visualised in realtime on a virtual 3D character in Blender.

posed. HMM-based singing synthesis works in a similar way compared to speech. It still uses models for the generation of spectrum, frequencies and durations. Yet, it is challenging to see the output of models trained on singing data when using short-term parameter trajectories to synthesise the final output. Based on the first approach presented for reactive singing synthesis [VAO<sup>+</sup>13], [Section 7.1.7] we see that the output is intelligible, with the correct rhythm and without any significant quality degradation when compared to the offline synthesis approach. It is possible to reactively manipulate the generated pitch, the vocal tract length, the speed of every frame, as well as the duration of every state of every model.

We think that using pHTS and an accurate controller such as HANDSKETCH combined with appropriately trained singing models it would be possible to artificially synthesise targeted songs in the context of a performance. Such an attempt would require linguistic pre-processing of the lyrics, so that the song would be transcribed into phonetic labels. These labels would be clustered for each sung note of the song. For example take the lyrics of the song and introduce a special character in the text at the place the singer changes note. Then tricking the phonetiser to group the decoded phonemes accordingly

and propagate this grouping to the output of Natural Language Processor (NLP). Once this grouping is achieved then the performer can add the desired controls over the generated trajectories. That way, a performer could learn to mimic the pitch contour with the pen and, at the same time, trigger clusters of labels according to how the lyrics are split into notes in the song. However, currently there are no assessed evaluation measurements to prove this assumption.

#### 7.2.4 *Incremental speech synthesis*

Returning back to speech synthesis, but focusing on the generation of the phonetic context, we see that in general, TTS systems rely on full pre-specified utterances that are available beforehand for the synthesis. Such an approach limits the reactivity and realtime context control of speech synthesis. There are indeed applications such as reactive dialogue systems and speech devices for disabled people that require in several cases that the phonetic context of the output is incrementally created. While pHTS brings performative controls and realtime synthesis, it is evident that it still relies on labels with linguistic context, which are computed offline prior to synthesis. Indeed, the synthesised context is static, not computed in at runtime. The only reactive context control is the order in which these pre-computed phonetic labels are sent to be processed (e. g. randomly shuffling them).

A first attempt to tackle this problem was presented in [Section 7.1.1](#) by building the RNLP module<sup>11</sup>. However, this was an empirical approach that we did not investigate further. In [dTA<sup>+</sup><sub>13</sub>] we worked on a new realtime linguistic front-end that allows the continuous incremental creation of the context-dependent labels. The reason of building a new front-end is that the current front-ends (e. g. “Festival TTS” system or “MARY-tts”) first assume that the full utterance is present at analysis time and second they are simply not fast enough for realtime analysis. Similar to our work in [Section 4.2](#), the full linguistic context taken into account was reduced and the models retrained with the new context. The system uses word-sized chunks, so every time the user completes inputting a word, it is looked up in the CMUDict 0.4 dictionary which provides stress patterns. Then syllables and phones are

---

<sup>11</sup> Realtime incremental speech synthesis prototype demo (visited June 2014):  
<https://www.youtube.com/watch?v=KAUv3kTiwWo>

retrieved and the corresponding labels are built. No letter-to-sound rules are included. The user by typing on a keyboard provides the textual input which is then reactively converted into appropriate contextual input for synthesis. Such a task is time demanding though and the user is not able to input sufficient amount of context to be synthesised in realtime, even while using simple word prediction. Thus, synthesis was slowed by a factor of 2.5 to allow a skilled typer to type fast enough. All the controls available in pHTS can be applied during the use of this new front-end. However, objective or subjective tests need to be conducted for further evaluation of the system.

### 7.3 POTENTIAL APPLICATIONS

We have already described various applications of the proposed pHTS framework. However, this is only a fraction of the possible uses of pHTS. Our work contributes in exploring reactive and expressive controls not only over voice but also over gait and audio-visual laughter. This opens the doors for various entertaining applications, such as gaming, animated characters and interactive avatars, using not only speech but also gait and laughter. Likewise, movie dubbing and *reactive audiobooks* can benefit of our work. Also, assistive applications for speech-impaired people could use pHTS for realtime contextual control, fine tuning and customisation in order to build reactive personalised voices. Silent speech communication as well as speech pedagogy and therapy could benefit from this performative approach. The list goes on with performing art applications and new interfaces for musical expression. Note that this is not an exhaustive application list.

### 7.4 DISCUSSION

Here we have gathered different application approaches for reactive text, speech, singing, audio-visual laughter and gait synthesis. All these prototypes have different backgrounds and motivations, serving different purposes. Yet their common aspect lies in the performative control of the applied statistical modelling.

Most of the prototypes focus mainly on the gestural control of prosodic and contextual speech parameters, demonstrating that prosodic control can be both meaningful and accurate while contextual controls need further re-

search. The interpolation among accents is also investigated through an interactive map. Likewise, an application controlling speech synthesis through the adjustment of the articulators was tested. These latter approaches prove that even more complex aspects of speech synthesis can be reactively controlled. Furthermore, we have worked on real-world entertaining application such as *reactive audiobooks* and *Tanukis*.

Additionally, prototypes beyond the initial speech framework were presented. Singing synthesis is one case but we think that audio-visual laughter and gait synthesis are excellent examples of applying pHTS in different contexts from the one it was designed for. Especially when it comes to the realtime control of an animated character. The incremental speech synthesis system that we implemented as a proof of concept is also an interesting step towards realtime creation of speech from textual input.

However we see that it is not straightforward to formally evaluate most of the proposed prototypes, especially those using gestures or body postures in order to control the generated speech, but we can certainly demonstrate that interactive manipulation of all the levels of artificial speech production in realtime is feasible. There are two aspects here that need further investigation. The first one is how users are able to influence speech production interactively, how accurate the applied controls are and of course how meaningful they are for the user. The second aspect is how the result of these controls is perceived by listeners. Currently, no restriction are imposed as to how and when modifications could be made, giving variable controls that are not always meaningful. Setting more restricted patterns to convey the final target, could probably lead to better controllability. Certainly further research in this specific area of Human Computer Interaction (HCI) is required.

## MAGE/pHTS: PERFORMATIVE HMM-BASED SYNTHESIS LIBRARY

---

### Contents

---

8.1	Realtime architecture of MAGE . . . . .	140
8.2	Reactive controls . . . . .	142
8.2.1	Available controls . . . . .	142
8.2.2	Regular expressions . . . . .	143
8.2.3	Logging user actions . . . . .	144
8.2.4	Introducing a <i>state queue</i> . . . . .	144
8.3	C++ API . . . . .	146
8.4	Discussion . . . . .	148

---

Part II shows that by using the short-term parameter generation approach [Chapter 4] the standard HMM-based speech synthesis framework becomes performative and therefore allows reactive model interpolation [Chapter 5] and reactive parameter mapping between different feature spaces [Chapter 6]. However, in order to support user interaction and application designs [Chapter 7] it is essential to introduce a realtime architecture and multi-threaded controls.

MAGE/pHTS is an open source library [Section B.3] based on the performative HMM-based speech synthesis framework, that unifies our work and implements the required architectural modifications<sup>1</sup> for user interactions. To our current knowledge MAGE/pHTS is the first system for reactive programming of HMM-based speech synthesis that allows reactive prosodic and contextual user control. It has been also in cases that are not related to speech, such are audio-visual laughter and stylistic gait synthesis and reconstruction.

---

<sup>1</sup> Modifications have been applied on the HTS-engine version 1.06



Figure 8.1: MAGE, an open source library for performative HMM-based synthesis.  
<http://mage.numediart.org/>.

In this Chapter we detail the realtime architecture of MAGE/pHTS<sup>2</sup> that allows reactive user control over the available contextual information, speech prosody, speaking style and quality. Moreover we present its simple C++ API that allows performative HMM-based synthesis to be easily integrated into realtime frameworks [102],[103]; to run on various devices and to create different prototypes Chapter 7. In this Chapter we detail the realtime architecture of MAGE (Section 8.1) that allows reactive user control (Section 8.2). Moreover, in Section 8.3, we present the API of the MAGE library and finally in Section 8.4 we discuss its contributions.

## 8.1 REALTIME ARCHITECTURE OF MAGE

MAGE integrates multiple threads and queues in the pHTS framework. Each thread controls a different production level of the artificial speech and therefore allows accurate control over that production level. Each queue, which is actually implemented as a ring buffer [104], is shared between two threads and ensures correct data exchanges. As illustrated in Figure 8.2, MAGE uses three main threads: the *label thread*, the *parameter generation thread* and the *audio generation thread* while three queues are shared between paired-threads: the *label queue*, the *parameter queue* and the *sample queue*. There is also another queue, called *model queue* and is used within the *parameter generation thread*.

<sup>2</sup> For the sake of brevity, further on when using MAGE we imply MAGE/pHTS

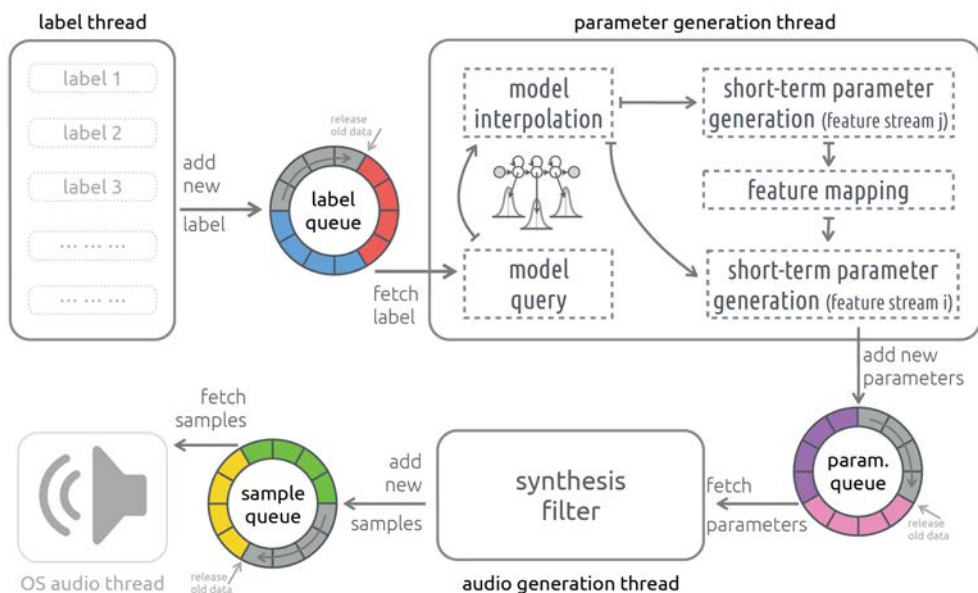


Figure 8.2: Reactive parameter generation using multiple threads and queues in MAGE.

The *label thread* controls the input sequence of the phonetic labels, by pushing the received phonetic labels onto the *label queue*. Then, the *parameter generation thread* reads from the *label queue* one phonetic label at a time. For that single label the corresponding models are retrieved and every state is passed to the *model queue*. From there the corresponding speech parameters are generated (sequences of spectral and excitation parameters including first and second derivatives of the static features), which are locally-maximized using by default only the current phonetic label/model (and if available, the two previous ones), as explained in [Section 4.1](#). However, it is possible to change the size of the sliding window by means of configuration parameters. The generated speech parameters (e. g. envelope, pitch, etc.) are stored in the *parameter queue*. Finally, the *audio generation thread* generates the actual speech samples corresponding to the input phonetic label and stores them in the *sample queue* so that the system's *audio thread* will access them and deliver them to the user's audio system.

## 8.2 REACTIVE CONTROLS

Accessing and controlling every thread has a different impact over the synthesised speech. Also the delay in applying any received user controls may vary from a single sample to a phonetic labels depended on the thread handling the control. For example controlling the interpolation scheme has a different latency than shifting the generated pitch trajectory. Extended description of the controls and respectively the delays are given below.

### 8.2.1 Available controls

The *label thread* is responsible for the contextual control of the system. Indeed, the context of the targeted output can be easily manipulated in realtime by simply controlling the order in which the available phonemes for processing will be input into the system. Note here that reactive contextual control still requires a lot of research in order to find the best set of contextual factors to be used for training as well as for the realtime building of the labels.

The *parameter generation thread* can reactively modify the way the available models are used for the parameter sequence generation [3], [Section 4.1]. In other words, user controls such as reactive model interpolation (with different interpolation weights among the various feature streams), state duration manipulation or mapping between different feature spaces (i. e. how a given stream influences another [Chapter 6]) can be applied on the current phonetic label through this thread. These controls are slightly less reactive than those applied in the *audio generation thread*, since they are dependent on the current label and not the current frame or sample but still this is not perceivable by the user.

Finally, the *audio generation thread* reactively manipulates the vocoding of every sample, resulting in prosody and voice quality controls for manipulating pitch, speed, volume and vocal tract length. The controls are applied to every sample separately and thus, they are much more reactive and accurate.

We see that the delay in applying any received user control can be as long as one phonetic label for the *label thread* and the *parameter generation thread* while it can be as short as a single speech sample for the *audio generation thread*. Figure 8.3 illustrates how the different production levels of the generated speech can be influenced by the user controls.

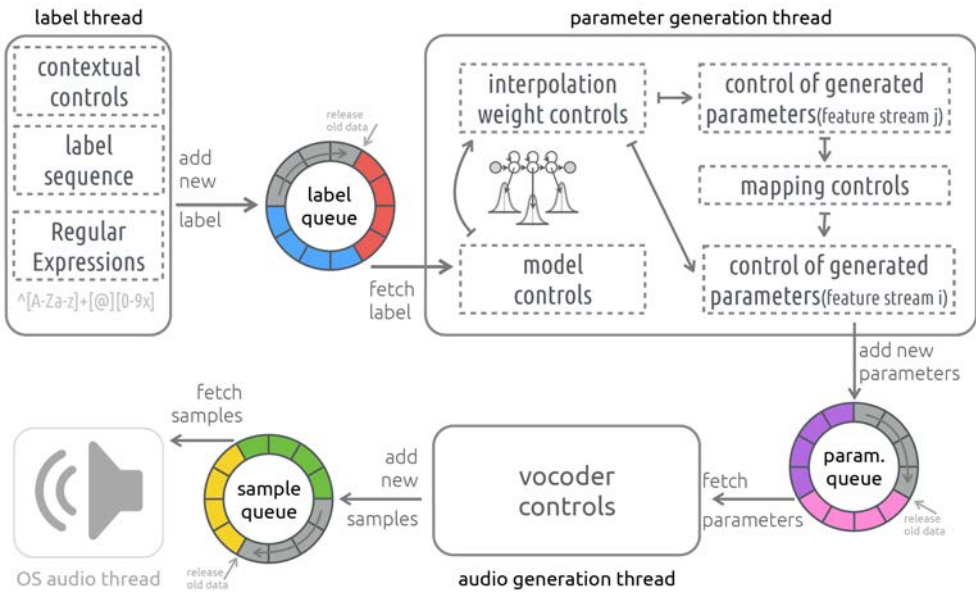


Figure 8.3: User can apply controls on every thread manipulating the generated speech over the different production levels.

### 8.2.2 Regular expressions

Apart from rather simple controls such as control of the sequence of the input labels or straightforward reactive model interpolation or prosody and voice quality on the currently generated output, MAGE supports more complex controls by using *regular expressions* [105]. In HTS and consequently in pHTS, every phonetic label, in addition to phoneme information, contains various linguistic contexts (e.g. stress, pitch accent, tone, etc.) [Section A.2]. Given *regular expressions* describing the format of the input labels [Section A.1] the user is able to “query” and apply certain controls conditionally. This can be done on every production level of the artificial speech<sup>3</sup>. For example, for the contextual control that occurs in the *label thread*, if the current phoneme identity is “pau” then skip the synthesis of this label, which leads to absence of pauses in the output. Similarly, for model control that occurs in the *parameter generation thread*, if the current syllable is stressed then interpolate speaking style  $i$  with speaking style  $j$  using interpolation

<sup>3</sup> This is an easter egg footnote - if you are reading this then you may ask for chocolate

weight vector  $y$ . A last example for the actual sample manipulation that occurs in the *audio generation thread*, if the next phoneme identity is “o” and the previous syllable stressed then increase the generated pitch by 50%.

### 8.2.3 *Logging user actions*

Working with reactive controls on HMM-based synthesis gives very interesting results and it is important that these results can be reproduced, either on-the-fly in a performance but also “offline” as a “playback” of the initially applied controls. In the first case, the reproduction of an effect depends on the performer but in the second case it depends on the system. In order to tackle this, we introduce a simple logging system in MAGE. When enabled, it records the different user controls (e. g. labels, pitch, interpolation weights, etc.) with a timestamp corresponding to the index of the current sample. It also records the generated feature trajectories with the applied modifications. Such a logging system also serves the purpose of detecting and explaining unexpected situations, which are frequently present in reactive applications. Depending on many factors (e. g. thread priority and delay, process scheduling, etc.), the application time of a control might be different than its input time. Thus, when such a timing mismatch occurs it is rather difficult to reproduce it, find where and why it happened and actually fix it. Hence, logging the user controls helps in solving and understanding such situations.

Moreover, logging the applied controls can also be used in the context of “Analysis-by-Performance” [106]. Analysis-by-Performance is an approach studying the gestural behaviour of a performer while imitating a certain sound effect by means of an appropriate digital musical instrument. By observing and studying the gestures of a performer, while imitating a certain task or effect, e. g. mimicking vibrato, converting natural to expressive speech, reacting to sound changes in the environment, etc. can bring new insights and lead to new production models.

### 8.2.4 *Introducing a state queue*

As we have seen until now, the available phonetic labels are streamed, the corresponding context-dependent HMMs (generally 5-states for each model in standard HTS) are retrieved and within a sliding window our parameters

are generated. But what happens if we do not want to retrieve our state sequence from the HMMs themselves? Why should a user be obliged to only push left-to-right HMMs into the model queue and, from there, compute the duration of each state and the sequence of corresponding observations? What if the synthesis target comes not in form of labels but in form of states (e. g. generated from a recognition system)? One example case where such control were needed is presented in [TdR<sup>+</sup>14], where the lower-body dimensions of a walking sequence is “reconstructed” from the upper-body gestures in realtime.

Therefore, we added a *state queue* into MAGE as an alternative to the *model queue*. While the *model queue* is usually filled with sequences of states corresponding to models selected to match the labels in the *label queue*, the *state queue* is fed directly with one state at a time. Each state corresponds to one frame of observations and, as such, has a duration of one. If the system must remain in a state for  $N$  frames, that state is simply pushed in the queue  $N$  times. This enables arbitrary patterns and number of states for the HMMs and thus overcomes the limiting effect of the *model queue*.

As for the short-term computation of observation frames from the *state queue*, it is achieved almost as for the *model queue*. The most significant difference is that one can set  $M$ , the number of frames to be computed whereas in the *model queue* the frames are computed for one complete model at a time, and the number of frames generated,  $\beta_1$  (as defined in Equation 4.3), is equal to the duration of that model. The context for the short-term parameter computation is set in the same fashion as for the *model queue*, except that the user sets a number of states to be considered before and after instead of a number of models. This notably allows to always use a constant amount of contextual information, for instance 17 states in the past and 3 states in the future of the  $M$  states that correspond to the  $M$  frames to be computed. This contrasts with the *model queue* for which the amount of contextual information is the sum of the durations of each model used as context and thus can change at every step. Using the state queue with  $M = 1$ , one can even make the computation for one state at a time. In other words, one can generate one frame at a time, while still using surrounding states as the context for the short-term parameter computation.

Specifically in [TdR<sup>+</sup>14], every time a state is generated by the recognition process and passed to the system, it is then pushed into the internal *state queue*. Each element of the *state queue* corresponds to an analysis frame (i. e.

of the upper part of the body that has been recognised) and for which we are going to use the state models to synthesise a set of features (i. e. for the lower part of the body). The states present in the *state queue* can be divided into three groups:

- past states ( $Q_P$ ): the first  $P$  state models in the queue (i.e. the oldest models) correspond to frames of features that have already been computed in previous iteration(s)
- current states( $Q_C$ ): the next  $C$  state models corresponds to frames of features that will be computed with the current iteration
- future states ( $Q_F$ ): the last  $F$  state models corresponds to frames of features that will be computed in upcoming iteration(s)

$Q_P$  and  $Q_F$  give some contextual information around  $Q_C$  necessary to compute the  $C$  current frames in a smooth continuity with the past and future frames and, as such, large values of  $P$  and  $F$  ensure a better reconstruction. However, increasing  $P$  and  $F$  also increases the cost of computing the current set of features. Besides, each “future” state in the queue actually corresponds to a frame of input features that has already been recorded and recognised. Thus, using  $F$  future states creates a delay of  $F$  frames between the time a set of features (i. e. from the upper part of the body) is input in the system and the time its corresponding set of features is output (i. e. lower body) . Also note that state sequences recognised a few seconds in the past have generally no impact on the result of the computation of the current features, therefore overly large values of  $P$  are unnecessary.

In the case at hand [TdR<sup>+</sup>14], we want to minimise the delay between the input and the output to be as reactive as possible. Therefore, we set  $P = 20$  and  $F = 0$  to set the *state queue* at a zero-frame delay. As for  $C$ , the relatively low frame rate in gesture recognition allows us to make the whole computation one frame at a time and we set  $C = 1$ . Note however that for higher frame rate (or more computationally expensive cases) we may need to compute blocks of several frames per iteration ( $C > 1$ ) instead.

### 8.3 C++ API

MAGE is a C++ library that can be easily integrated to various realtime programming frameworks, such as openFrameworks [102], iOS [107], SuperCol-

lider [98], Pure Data [103] and Max/MSP [108]. It can be used to build standalone performative applications or applications that have an interface and communicates via OSC messages [109] with the synthesis part of a different application implementing MAGE. In this Section we describe how MAGE is integrated in these realtime programming schemes by using just the provided API. This comprehensive API implements high level control functions that are easy to understand. An external C++ application can access it directly by including the `mage.h` and calling its functions appropriately.

The following example illustrates a simple use case of the API. Note that every function and class of MAGE belongs to the namespace MAGE. More sophisticated examples can be found in [AMW<sup>+</sup>].

```
#include "mage.h"
using namespace MAGE;
...
Mage *mage = new Mage();
mage->addEngine( "slt", "slt.conf" );
...
mage->setAlpha(0.42);
mage->setPitch(50, MAGE::shift);
...
float * buffer = new float[bsize];
for( int k = 0; k < bsize; k++ )
    buffer[k] = mage->popSamples();
...
delete[] buffer;
delete mage;
```

In details, an instance of MAGE class is created and a voice model (trained with standard HTS) is loaded using a key name and a configuration file. Then a series of various controls can be applied, for example changing the value of the vocal tract length or shifting the generated pitch. Then, the generated speech samples are delivered to the audio thread of the system. Finally, before exiting the application the allocated memory is deallocated and the synthesis application is terminated.

The applications described in [Chapter 7](#) are actual examples of C++ programs embedding and using MAGE directly through its API. More details of integrating MAGE in different realtime programming schemes can be found in [Appendix B](#).

## 8.4 DISCUSSION

The current version of MAGE only integrates a fraction of the various techniques that have been developed for the standard HMM-based speech synthesis system. It would be interesting to build an “umbrella” that will bring together techniques such as adaptation [39], other interpolation approaches [40], [Chapter 5], articulatory control<sup>4</sup> [60], [61], [Chapter 6] etc., in the performative framework of MAGE. So far, all these developed techniques could be applied sequentially, one at a time, but we hypothesise that being able to reactively apply them and combine them over the acoustic output, will increase the potential of speech synthesis. All the knowledge built around HTS until today will be also usable in a performative way, thereby adding another parameter in the speech production equation: the user. Engaging the user in the speech production process will bring new insights in our perception of artificial speech. It will broaden the set of possible real life and scientific applications [Chapter 7].

Integrating this unified platform and building various application sets, will enable the reactive creation of desired voice identity and speaking style or expressions, that were not initially included in the database. Users can participate and contribute to this unified platform, engage to applications and have a different experience when it comes to artificial speech. Even with predefined phonetic labels, a performer can transform passive listening to an interactive experience. Thus, the creativity and imagination of the performer will be expressed even on a static context and if desired, the reactive participation of the audience can be included.

---

<sup>4</sup> Not available in the public version of MAGE

## CONCLUSIONS

CONCLUSIONS	149
9 CONCLUSIONS AND FUTURE WORK	151
9.1 Conclusions . . . . .	151
9.2 Future work . . . . .	155



## CONCLUSIONS AND FUTURE WORK

---

### Contents

---

9.1	Conclusions . . . . .	151
9.2	Future work . . . . .	155

---

This document has presented several innovative research areas in performative HMM-based synthesis. For each of these areas, a significant contribution to the state-of-the-art has been brought during this PhD thesis. This Chapter concludes this thesis work. The structure of [Section 9.1](#) follows the five main axes presented in the [Introduction](#) and detailed in [Part II](#) and [Part III](#). Finally, [Section 9.2](#) presents possible future works in this area.

### 9.1 CONCLUSIONS

We can see that statistical parametric synthesis offers a wide range of techniques to control and improve generated output. However, there is still much to do in the field, many ideas to be fully explored and still many more to be conceived. Exploring these new ideas, that might even come from diverse research fields, may bring solutions to bridge the gap between natural and synthesised output and its meaningful control.

Performative HMM-based synthesis (pHTS) is an approach that was inspired from the statistical parametric speech synthesis. It enables the creative involvement of the user/performer during the production of artificial speech. Indeed, pHTS led to the development of an open source library, `MAGE/pHTS` that allows interactive control on every level of the speech (or gait, or audio-visual laughter) production, making it easy to create various control interfaces and applications.

The project has focused mainly on creating a performative framework that will make HMM-based synthesis accessible to different fields, such as performing arts, musical instrument design, avatar control, etc., as opposite to speech only. Additionally, by providing a user-friendly API the user base is not limited only to speech experts but it is broadened also to artists, performers, linguists, etc. At the same time, the goal of our work is to provide an output quality that is perceptually identical to the one of the original system, or at least whose degradation cannot be perceived by the subjects.

Besides [Chapter 1](#) and [Part I](#), dedicated to [introduction](#) and to [state-of-the-art](#) of statistical parametric speech synthesis, five novel contributions related to interactive HMM-based synthesis are presented in [Chapter 4](#) to [Chapter 8](#).

- *pHTS*, a novel method for performative HMM-based speech synthesis;
- *Interactive model interpolation*, using interpolation methods in the proposed performative framework;
- *Interactive articulatory control*, controlling interactively the generated speech through the articulators themselves;
- *Real-world applications*, a collection of interactive applications based on our work, for speech and beyond speech fields;
- *MAGE/pHTS: performative HMM-based synthesis library*<sup>1</sup>, an open source library that unifies our work and allows others to access and use it;

The **first** contribution, pHTS, is a new method that extends the original HTS system, so that controls over the various production levels can be almost instantly applied [[Chapter 4](#)]. It combines quality and controllability. It takes advantage of all the existing knowledge built around HTS, while providing controls over the frames, models or phonetic labels. This significantly reduces the accessible time scale, so that users have instant feedback of their controls. After cross-validating the four possible [systems](#) [[Table 1.1](#)], in a set of four test [cases](#) [[Figure 4.4](#)], we have obtained very interesting results, indicating that the reduced-context synthesis has a quality equivalent to the

---

<sup>1</sup> The name [MAGE](#) comes actually from the names of the people that actively contributed in the creation of the library at its birth: *Maria*, *Alexis*, *Geoffrey* - indeed a great inspiring team!

original systems, and only the reduced-context training actually degrades the final quality. In other words, that means that in cases where interactivity is required, pHTS can be used in the place of HTS.

The **second** contribution has been the application of interpolation methods, initially developed for the traditional HMM-based speech synthesis system, in the interactive framework of pHTS [Chapter 5]. By using more complex model transformations, than just pitch, speed, volume, vocal tract length and state duration, we obtain far more interesting results. We are able to interactively combine models and generate new styles not initially included in the recorded database, while the output is being generated. Based on this property, it is possible to change the speaking style, speaker identity, emotion, accent, etc., on the continuous scale independently on every phonetic label as the speech signal is being synthesised and heard by the user. It was also shown that we were able to apply our work on reactive large-scale interpolation scheme as well as to browse the stylistic space of gait and visual laughter synthesis.

The **third** contribution consists in studying the effect of integrating articulatory features in the interactive framework of pHTS [Chapter 6]. This enables us to reactively control the generated speech through the articulators themselves. The interesting property of this approach is that it gives a “physical” or “intuitive” meaning to the user, rather than using “abstract” vocoder parameters. However, this approach exhibits poorer quality than the one encountered with the original system. One of the possible reasons might be that the articulatory features require larger estimation window, thus distortions are introduced which are then reflected in the final speech output through the articulatory-to-acoustic mapping. Moreover, the conversion from LSPs to MGCs, might also be the reason for additional errors. Another issue that rises here is the high complexity of control parameters (i. e. articulators, but not limited to). Nevertheless, we have obtained interesting results, where users do not actually stress the small quality degradation. Even though the controls seem rather straightforward, at the end it is very demanding and difficult for the user to achieve a certain target. However, users tend to enjoy the challenge of controlling the system and its resulting output when driven to extreme states.

With our **fourth** contribution, we have tried to develop not only proofs of concept but also actual applications [Chapter 7]. We developed several prototypes related to speech, most of which focus mainly on the gestural con-

trol of prosodic and contextual speech parameters. Additionally, we demonstrate various other applications that enable more complex controls of the artificial speech, such as accent and articulatory controls. Besides these experimental applications demonstrating the feasibility and principles of performative HMM-based speech synthesis, we present also *Reactive Audiobooks* and *Tanukis*, which are targeted as real-world applications. Moreover, prototypes beyond the initial speech framework were developed. We showed that controlling the audio-visual laughter and stylistic gait of an animated character in realtime is not only possible but also a lot of fun. We see that different disciplines can benefit from our work, since performative HMM-based synthesis can be used for many applications, either academic or commercial.

The **fifth** and last contribution is the creation of *MAGE/pHTS*, an open source library for performative HMM-based synthesis that unifies the work presented in this thesis [Chapter 8]. The design and structure of *MAGE/pHTS* is so flexible, that almost any new advances of the HMM-based speech synthesis framework can be easily integrated in the proposed performative system, while preserving the accurate control on all production levels and the quality of the initial system. The principle of this library is its accessibility, being open source and combined with an easy API, anyone can access it and use it in order to develop or test novel application ideas. Figure 9.1 gives a short summary of research achievements of this thesis.

To conclude this PhD, we would like to recall the importance of interactivity in speech synthesis. It is an essential social signal and dynamic phenomenon, transmitting various information (e. g. social, verbal, emotional, etc.) and is influenced constantly from fast-changing environmental factors (e. g. speaker's mood, feedback from a listener/speaker, etc.). Thus, the main motivation of this work was to advance the state-of-the-art in HMM-based synthesis, and to interactively involve performers in this synthesis process. Moreover, we believe that interactive control can improve the naturalness of generated output by studying the complex user controls in human communication. There are many challenging tasks to be tackled still, but we hope that this dissertation has contributed in promoting new ideas and will help others to bring forward even more interesting questions.



Figure 9.1: Short summary of the research achievements of this thesis.

## 9.2 FUTURE WORK

As with every work, and certainly when a new idea, method or solution is proposed, there is always room for improvement and continuation towards its advancement and completion<sup>2</sup>. It becomes evident that further evaluation of the system is required. More objective and subjective tests must be conducted, using different methods, experimental conditions and protocols. This will enable us to have a wider and more concrete view of the final quality of the system. Besides, user studies are essential for a complete evaluation of the proposed applications. They will show us how users manipulate

<sup>2</sup> While writing this dissertation, we were also able to incorporate plenty new features in MAGE/pHTS, leading to a new version. New examples are being constructed and the documentation is being improved. Additionally, all the source code of the applications presented here will be publicly available in the near future with the release of the new version of MAGE/pHTS

a given feature space, how skilled a user should be and how an application can be improved in order to facilitate better user interaction.

In the near future we would like to improve the current realtime system architecture so that we will be able to integrate more aspects of the HMM-based synthesis state-of-the-art in the proposed performative framework. This will allow us to propose different controls over the speech production levels as well as new realtime application designs. We should highlight here that comprehensive system- and application- wise evaluation is essential.

Apart from a structured roadmap, it is substantial to envision other possibilities, directions and paths. We believe that a leap forward will come by using game thinking and game mechanics in non-game contexts so that users can help us solve synthesis problems. Gamification has been studied and applied in several domains, targeting to engage, teach or entertain users so that the perceived ease of use of information systems can be improved. The gamification of HMM-based synthesis, making it part of the everyday life of the user/performer in an interesting, fun, enjoyable or even competitive fashion, could bring very interesting insights in the speech community. Creating such applications with crowdsourcing aspects will help us not only tackle several existing problems and improve the existing technology, but also form new questions to pursuit.

## APPENDIX

APPENDIX	157
A HTS	159
A.1 Label format . . . . .	159
A.2 Contextual factors . . . . .	161
A.3 Singing contextual factors . . . . .	162
A.4 Decision tree example . . . . .	162
A.5 Question file example . . . . .	169
B MAGE	175
B.1 Integrations of MAGE . . . . .	175
B.2 MAGE demo links . . . . .	178
B.3 License . . . . .	178
C RELATED PROJECTS	181
C.1 Speech synthesis projects . . . . .	181
C.2 Speech corpuses . . . . .	182
C.3 Other speech related links . . . . .	182
C.4 Beyond speech projects . . . . .	183
C.5 Fun links . . . . .	183





# HTS

## Contents

A.1	Label format . . . . .	159
A.2	Contextual factors . . . . .	161
A.3	Singing contextual factors . . . . .	162
A.4	Decision tree example . . . . .	162
A.5	Question file example . . . . .	169

In this Appendix you will find detailed information regarding the HTS system [4].

### A.1 LABEL FORMAT

$p1 \wedge p2 - p3 + p4 = p5 @p6\_p7 / A : a1\_a2\_a3$   
 $/B : b1 - b2 - b3 @b4 - b5 \&b6 - b7 \#b8 - b9 \$b10 - b11 !b12 - b13 ;b14 - b15 |b16$   
 $/C : c1 + c2 + c3 /D : d1\_d2 /E : e1 + e2 @e3 + e4 \&e5 + e6 \#e7 + e8$   
 $/F : f1\_f2 /G : g1\_g2 /H : h1 = h2 \wedge h3 = h4 |h5 /I : i1\_i2 /J : j1 + j2 - j3$

p1	the phoneme identity before the previous phoneme
p2	the previous phoneme identity
p3	the current phoneme identity
p4	the next phoneme identity
p5	the phoneme after the next phoneme identity
p6	position of the current phoneme identity in the current syllable (forward)
p7	position of the current phoneme identity in the current syllable (backward)
a1	whether the previous syllable stressed or not (0: not stressed, 1: stressed)
a2	whether the previous syllable accented or not (0: not accented, 1: accented)
a3	the number of phonemes in the previous syllable

b1	whether the current syllable stressed or not (0: not stressed, 1: stressed)
b2	whether the current syllable accented or not (0: not accented, 1: accented)
b3	the number of phonemes in the current syllable
b4	position of the current syllable in the current word (forward)
b5	position of the current syllable in the current word (backward)
b6	position of the current syllable in the current phrase (forward)
b7	position of the current syllable in the current phrase (backward)
b8	the number of stressed syllables before the current syllable in the current phrase
b9	the number of stressed syllables after the current syllable in the current phrase
b10	the number of accented syllables before the current syllable in the current phrase
b11	the number of accented syllables after the current syllable in the current phrase
b12	the number of syllables from the previous stressed syllable to the current syllable
b13	the number of syllables from the current syllable to the next stressed syllable
b14	the number of syllables from the previous accented syllable to the current syllable
b15	the number of syllables from the current syllable to the next accented syllable
b16	name of the vowel of the current syllable
c1	whether the next syllable stressed or not (0: not stressed, 1: stressed)
c2	whether the next syllable accented or not (0: not accented, 1: accented)
c3	the number of phonemes in the next syllable
d1	gpos (guess part-of-speech) of the previous word
d2	the number of syllables in the previous word
e1	gpos (guess part-of-speech) of the current word
e2	the number of syllables in the current word
e3	position of the current word in the current phrase (forward)
e4	position of the current word in the current phrase (backward)
e5	the number of content words before the current word in the current phrase
e6	the number of content words after the current word in the current phrase
e7	the number of words from the previous content word to the current word
e8	the number of words from the current word to the next content word
f1	gpos of next word
f2	the number of syllables in next word
g1	the number of syllables in the previous phrase
g2	the number of words in the previous phrase
h1	the number of syllables in the current phrase
h2	the number of words in the current phrase
h3	position of the current phrase in this utterance (forward)
h4	position of the current phrase in this utterance (backward)
h5	TOBI endtone of the current phrase
i1	the number of syllables in the next phrase
i2	the number of words in the next phrase

j <sub>1</sub>	the number of syllables in this utterance
j <sub>2</sub>	the number of words in this utterance
j <sub>3</sub>	the number of phrases in this utterance

## A.2 CONTEXTUAL FACTORS

By default in HTS, the contextual factors taken into account for training of the HMMs [4] are listed below :

- phoneme
  - {preceding, current, succeeding} phonemes <sup>1</sup>
  - position of current phoneme in current syllable
- syllable
  - number of phonemes at {preceding, current, succeeding} syllable
  - accent of {preceding, current, succeeding} syllable
  - stress of {preceding, current, succeeding} syllable
  - position of current syllable in current word
  - number of {preceding, succeeding} stressed syllables in current phrase
  - number of {preceding, succeeding} accented syllables in current phrase
  - number of syllables {from previous, to next} stressed syllable
  - number of syllables {from previous, to next} accented syllable
  - vowel within current syllable
- word
  - guess at part of speech of {preceding, current, succeeding} word
  - number of syllables in {preceding, current, succeeding} word
  - position of current word in current phrase
  - number of {preceding, succeeding} content words in current phrase
  - number of words {from previous, to next} content word
- phrase
  - number of syllables in preceding, current, succeeding phrase
  - position in major phrase
  - ToBI<sup>2</sup> endtone of current phrase
- utterance
  - number of syllables in current utterance

<sup>1</sup> The two preceding and the two succeeding phonemes of the current phoneme are taken into account

<sup>2</sup> Tones and Break Indices

### A.3 SINGING CONTEXTUAL FACTORS

Additional contextual factors taken into account for training of the HMMs for singing [2] are listed below :

- syllable
  - the position of the current syllable within the current musical note and phrase
- musical note
  - musical tone, key, beat, tempo, length, and dynamics of preceding, current, and succeeding musical notes
  - the position of the current musical note within the current phrase
  - tied and slurred flags
  - distance between the current musical note and the preceding/succeeding accent and staccato
  - the position of the current musical note within the current crescendo and decrescendo
- song
  - number in the song
  - number of musical notes in the song
  - number of phrases in the song

### A.4 DECISION TREE EXAMPLE

This decision tree corresponds to a standard HTS training. It is the third state tree for the mel-cepstrum coefficients (“mcp\_s3”). It has 212 nodes, indexed from 0 to 211, pointed out with a hyphen (‘-’). In the left-hand column there is the ordered list of nodes with its corresponding questions. For each node, in the right-hand side, there are the instructions to go on down the tree. They will either lead to the following node (i. e. from the node 0, if the current phoneme is in a vowel it then goes to node -1, if not, it goes to node -6) or to an indexed final leaf (i. e. “mgc\_s3\_1”, pointing to the 1st cell of the probability matrix).

```
{*}[3].stream[1]
```

```
{
```

0	C-Vowel	-1	-6
-1	C-Voiced_Consonant	-2	-4
-2	C-Continuent	-3	-7
-3	C-Consonant	-16	-24
-4	C-r	-5	-45
-5	C-Nasal	-9	-23
-6	C-ay	-8	-55
-7	C-Fortis_Consonant	-12	-17
-8	C-Syl_High_Vowel	-10	-15
-9	C-Glide	-11	-19
-10	C-Front_Vowel	-14	-18
-11	C-Front_Stop	-13	-51
-12	C-hh	-84	-74
-13	C-dh	-27	-59
-14	C-Syl_er	-20	-60
-15	C-uw	-25	-127
-16	L-Phrase_Num-Words==0	-30	-43
-17	C-Negative_Strident	-28	-38
-18	C-Syl_ey	-33	-68
-19	C-Central	-39	-40
-20	C-Unrounded_Vowel	-21	-22
-21	C-Syl_ow	-52	-58
-22	C-Syl_ax	-36	-76
-23	C-m	-29	-67
-24	L-Fricative	-26	-89
-25	C-Syl_iy	-48	-34
-26	C-k	-31	-109
-27	C-Fricative	-32	-159
-28	C-Back	-35	-94
-29	C-Central	-135	-54
-30	L-Nasal	-44	-47
-31	C-p	-42	-101

-32	C-jh	-37	"mgc_s3_1"
-33	C-Word_GPOS==cc	-46	-161
-34	L-l	-62	-171
-35	L-Vowel	-65	-50
-36	C-Central_Vowel	-87	-57
-37	C-Central	-112	-41
-38	C-Front	"mgc_s3_2"	-144
-39	C-y	-75	-125
-40	R-Vowel	"mgc_s3_3"	-81
-41	L-Nasal	-61	"mgc_s3_4"
-42	C-Stop	"mgc_s3_5"	-73
-43	L-silences	"mgc_s3_6"	-113
-44	L-l	-49	"mgc_s3_7"
-45	Seg_Fw<=1	-53	-182
-46	L-Central_Fricative	-78	-156
-47	L-m	-90	-191
-48	L-Consonant	-117	-82
-49	L-Fricative	-56	-203
-50	L-Syllabic_Consonant	-91	"mgc_s3_8"
-51	L-silences	-69	"mgc_s3_9"
-52	C-Central	-160	-64
-53	L-Front_Vowel	-80	-179
-54	L-Vowel	-205	-85
-55	L-Nasal	-63	-194
-56	L-Long_Vowel	-71	-66
-57	L-n	-70	"mgc_s3_10"
-58	L-n	-115	"mgc_s3_11"
-59	L-Voiced_Consonant	-88	-142
-60	L-Coronal_Consonant	-102	-79
-61	L-Vowel	-99	-193
-62	L-r	-77	"mgc_s3_12"
-63	L-l	-72	"mgc_s3_13"
-64	R-r	"mgc_s3_14"	-114
-65	L-Nasal	-86	-207

-66	L-Front	"mgc_s3_16"	"mgc_s3_15"
-67	L-Unrounded_Vowel	-148	-153
-68	Seg_Bw==1	-106	-137
-69	L-Central_Fricative	-151	"mgc_s3_17"
-70	R-z	-100	"mgc_s3_18"
-71	L-No_Continent	-105	-133
-72	R-Nasal	-108	"mgc_s3_19"
-73	L-Nasal	-83	-146
-74	L-silences	-104	"mgc_s3_20"
-75	L-Unrounded_Vowel	-118	-208
-76	L-Central_Fricative	-96	-209
-77	L-Non_Anterior	-97	"mgc_s3_21"
-78	Seg_Fw==1	-93	-120
-79	L-Negative_Strident	-196	"mgc_s3_22"
-80	L-Consonant	-157	-201
-81	L-Unrounded_Vowel	-132	-172
-82	L-Neither_F_or_L	-122	-98
-83	Seg_Bw==1	-175	-110
-84	L-Voiced_Consonant	-95	"mgc_s3_23"
-85	R-Central_Fricative	-128	"mgc_s3_24"
-86	L-Back_Stop	-107	"mgc_s3_25"
-87	L-Nasal	"mgc_s3_27"	"mgc_s3_26"
-88	L-Vowel	-180	-123
-89	C-Coronal_Consonant	"mgc_s3_28"	-173
-90	LL-Reduced_Vowel	-187	"mgc_s3_29"
-91	L-Long_Vowel	-92	-138
-92	L-Fronting_Vowel	"mgc_s3_31"	"mgc_s3_30"
-93	L-Unvoiced_Consonant	-163	"mgc_s3_32"
-94	L-pau	-189	"mgc_s3_33"
-95	R-Unvoiced_Consonant	-111	"mgc_s3_34"
-96	L-Fortis_Consonant	-186	"mgc_s3_35"
-97	L-Coronal_Consonant	-116	"mgc_s3_36"
-98	L-Fricative	-103	"mgc_s3_37"
-99	L-l	-152	"mgc_s3_38"

-100	C-Syl_ah	"mgc_s3_39"	-131
-101	L-Nasal	-178	"mgc_s3_40"
-102	L-Anterior_Consonant	-141	"mgc_s3_41"
-103	L-Liquid	"mgc_s3_42"	-164
-104	R-IVowel	"mgc_s3_44"	"mgc_s3_43"
-105	L-Short_Vowel	"mgc_s3_46"	"mgc_s3_45"
-106	L-l	-126	"mgc_s3_47"
-107	L-Central_Stop	-136	"mgc_s3_48"
-108	L-Central_Fricative	-130	-177
-109	L-Nasal	-121	"mgc_s3_49"
-110	R-Central_Fricative	-119	"mgc_s3_50"
-111	L-Phrase_Num-Words<=1	"mgc_s3_51"	-134
-112	L-Fricative	-184	"mgc_s3_52"
-113	R-Phrase_Num-Syls==1	"mgc_s3_54"	"mgc_s3_53"
-114	L-m	"mgc_s3_56"	"mgc_s3_55"
-115	L-Central_Fricative	"mgc_s3_58"	"mgc_s3_57"
-116	L-er	-150	"mgc_s3_59"
-117	L-pau	"mgc_s3_61"	"mgc_s3_60"
-118	L-t	-140	"mgc_s3_62"
-119	Pos_C-Syl_in_C-Phrase(Bw)==1	-124	"mgc_s3_63"
-120	Pos_C-Syl_in_C-Word(Fw)<=1	-169	"mgc_s3_64"
-121	R-Syl_Num-Segs==0	"mgc_s3_66"	"mgc_s3_65"
-122	R-Nasal	-147	"mgc_s3_67"
-123	L-ih	"mgc_s3_69"	"mgc_s3_68"
-124	C-Syl_Stress==0	"mgc_s3_71"	"mgc_s3_70"
-125	L-Syl_Num-Segs==0	-204	"mgc_s3_72"
-126	R-Neigther_F_or_L	-174	-185
-127	L-m	-139	"mgc_s3_73"
-128	L-IVowel	-129	"mgc_s3_74"
-129	L-Fronting_Vowel	-165	"mgc_s3_75"
-130	L-Front_Consonant	-143	"mgc_s3_76"
-131	L-Central_Fricative	-155	"mgc_s3_77"
-132	L-Non_Coronal	"mgc_s3_79"	"mgc_s3_78"
-133	L-Unvoiced_Stop	-145	"mgc_s3_80"

-134	Pos_C-Syl_in_C-Phrase(Fw)<=5	"mgc_s3_81"	-200
-135	C-Syl_Central_Vowel	"mgc_s3_83"	"mgc_s3_82"
-136	L-r	-149	"mgc_s3_84"
-137	L-Neigther_F_or_L	-158	"mgc_s3_85"
-138	L-Front	"mgc_s3_87"	"mgc_s3_86"
-139	L-r	-167	"mgc_s3_88"
-140	L-Voiced_Consonant	-154	"mgc_s3_89"
-141	L-Unvoiced_Consonant	-183	"mgc_s3_90"
-142	L-Voiced_Fricative	"mgc_s3_92"	"mgc_s3_91"
-143	L-r	-162	"mgc_s3_93"
-144	L-Unrounded_Vowel	"mgc_s3_95"	"mgc_s3_94"
-145	LL-l	"mgc_s3_97"	"mgc_s3_96"
-146	Pos_C-Syl_in_C-Phrase(Bw)==1	"mgc_s3_99"	"mgc_s3_98"
-147	L-Coronal_Consonant	"mgc_s3_101"	"mgc_s3_100"
-148	L-Vowel	"mgc_s3_103"	"mgc_s3_102"
-149	L-Liquid	-206	"mgc_s3_104"
-150	L-m	-192	"mgc_s3_105"
-151	L-No_Continuent	-210	"mgc_s3_106"
-152	L-Liquid	"mgc_s3_108"	"mgc_s3_107"
-153	L-Central	"mgc_s3_110"	"mgc_s3_109"
-154	L-Syl_Num-Segs==0	"mgc_s3_112"	"mgc_s3_111"
-155	L-Neigther_F_or_L	-170	"mgc_s3_113"
-156	L-s	"mgc_s3_115"	"mgc_s3_114"
-157	C-Syl_AVowel	"mgc_s3_117"	"mgc_s3_116"
-158	L-dh	"mgc_s3_119"	"mgc_s3_118"
-159	L-eh	-168	"mgc_s3_120"
-160	L-Affricate_Consonant	-190	"mgc_s3_121"
-161	L-pau	"mgc_s3_123"	"mgc_s3_122"
-162	L-Word_Num-Syls==0	"mgc_s3_125"	"mgc_s3_124"
-163	L-Liquid	"mgc_s3_127"	"mgc_s3_126"
-164	L-Anterior_Consonant	"mgc_s3_129"	"mgc_s3_128"
-165	L-er	-166	"mgc_s3_130"
-166	Pos_C-Syl_in_C-Phrase(Bw)==1	"mgc_s3_132"	"mgc_s3_131"
-167	L-n	-176	"mgc_s3_133"

-168	L-Vowel	"mgc_s3_135"	"mgc_s3_134"
-169	L-g	"mgc_s3_137"	"mgc_s3_136"
-170	L-k	-188	"mgc_s3_138"
-171	R-pau	-198	"mgc_s3_139"
-172	L-iy	"mgc_s3_141"	"mgc_s3_140"
-173	L-Positive_Strident	"mgc_s3_143"	"mgc_s3_142"
-174	L-Unvoiced_Stop	-181	"mgc_s3_144"
-175	L-No_Continuent	"mgc_s3_146"	"mgc_s3_145"
-176	L-y	-199	"mgc_s3_147"
-177	R-Syl_Num-Segs==0	"mgc_s3_149"	"mgc_s3_148"
-178	L-Low_Vowel	"mgc_s3_151"	"mgc_s3_150"
-179	LL-dh	"mgc_s3_153"	"mgc_s3_152"
-180	L-Consonant	"mgc_s3_155"	"mgc_s3_154"
-181	L-r	-195	"mgc_s3_156"
-182	L-Unrounded_Vowel	"mgc_s3_158"	"mgc_s3_157"
-183	L-aw	"mgc_s3_160"	"mgc_s3_159"
-184	L-pau	"mgc_s3_162"	"mgc_s3_161"
-185	R-Glide	"mgc_s3_164"	"mgc_s3_163"
-186	L-Nasal	"mgc_s3_166"	"mgc_s3_165"
-187	L-ng	"mgc_s3_168"	"mgc_s3_167"
-188	L-Coronal_Consonant	"mgc_s3_170"	"mgc_s3_169"
-189	L-Anterior_Consonant	"mgc_s3_172"	"mgc_s3_171"
-190	R-Phrase_Num-Words<=2	"mgc_s3_174"	"mgc_s3_173"
-191	LL-Short_Vowel	"mgc_s3_176"	"mgc_s3_175"
-192	L-Rounded_Vowel	"mgc_s3_178"	"mgc_s3_177"
-193	L-Fronting_Vowel	"mgc_s3_180"	"mgc_s3_179"
-194	L-Front	"mgc_s3_182"	"mgc_s3_181"
-195	L-Not_Affricate	"mgc_s3_184"	"mgc_s3_183"
-196	L-Nasal	-197	"mgc_s3_185"
-197	R-Nasal	"mgc_s3_187"	"mgc_s3_186"
-198	LL-Nasal	"mgc_s3_189"	"mgc_s3_188"
-199	L-Central_Stop	"mgc_s3_191"	"mgc_s3_190"
-200	L-aa	"mgc_s3_193"	"mgc_s3_192"
-201	Seg_Bw<=1	-202	"mgc_s3_194"

-202	L-Central_Fricative	"mgc_s3_196"	"mgc_s3_195"
-203	L-s	"mgc_s3_198"	"mgc_s3_197"
-204	L-Non_Coronal	"mgc_s3_200"	"mgc_s3_199"
-205	L-r	"mgc_s3_202"	"mgc_s3_201"
-206	L-pau	"mgc_s3_204"	"mgc_s3_203"
-207	L-n	"mgc_s3_206"	"mgc_s3_205"
-208	Num-Syl_from_prev-StressedSyl<=0	"mgc_s3_208"	"mgc_s3_207"
-209	L-Negative_Strident	"mgc_s3_210"	"mgc_s3_209"
-210	L-UVowel	"mgc_s3_211"	-211
-211	R-Front_Vowel	"mgc_s3_213"	"mgc_s3_212"

#### A.5 QUESTION FILE EXAMPLE

Be aware this is only a little portion of the example questions file. This very same list is repeated 5 times, changing the prefix "LL-" which stands for *left-left* and makes reference to the phoneme before the previous one. There are four other different labels "L-", "C-", "R-" and "RR-", for the previous, current, next, and after the next phoneme respectively. Besides, these are only questions concerning phonetic features. There are many other questions within the file related to all the other phoneme features appearing in the label file and listed in [Section A.1](#).

QS "LL-Vowel"	{aa <sup>^</sup> ,ae <sup>^</sup> ,ah <sup>^</sup> ,ao <sup>^</sup> ,aw <sup>^</sup> ,ax <sup>^</sup> ,axr <sup>^</sup> ,ay <sup>^</sup> ,eh <sup>^</sup> ,el <sup>^</sup> ,em <sup>^</sup> ,en <sup>^</sup> ,er <sup>^</sup> ,ey <sup>^</sup> ,ih <sup>^</sup> ,ix <sup>^</sup> ,iy <sup>^</sup> ,ow <sup>^</sup> ,oy <sup>^</sup> ,uh <sup>^</sup> ,uw <sup>^</sup> }
QS "LL-Consonant"	{b <sup>^</sup> ,ch <sup>^</sup> ,d <sup>^</sup> ,dh <sup>^</sup> ,dx <sup>^</sup> ,f <sup>^</sup> ,g <sup>^</sup> ,hh <sup>^</sup> ,hv <sup>^</sup> ,jh <sup>^</sup> ,k <sup>^</sup> ,l <sup>^</sup> ,m <sup>^</sup> ,n <sup>^</sup> ,nx <sup>^</sup> ,ng <sup>^</sup> ,p <sup>^</sup> ,r <sup>^</sup> ,s <sup>^</sup> ,sh <sup>^</sup> ,t <sup>^</sup> ,th <sup>^</sup> ,v <sup>^</sup> ,w <sup>^</sup> ,y <sup>^</sup> ,z <sup>^</sup> ,zh <sup>^</sup> }
QS "LL-Stop"	{b <sup>^</sup> ,d <sup>^</sup> ,dx <sup>^</sup> ,g <sup>^</sup> ,k <sup>^</sup> ,p <sup>^</sup> ,t <sup>^</sup> }
QS "LL-Nasal"	{m <sup>^</sup> ,n <sup>^</sup> ,en <sup>^</sup> ,ng <sup>^</sup> }
QS "LL-Fricative"	{ch <sup>^</sup> ,dh <sup>^</sup> ,f <sup>^</sup> ,hh <sup>^</sup> ,hv <sup>^</sup> ,s <sup>^</sup> ,sh <sup>^</sup> ,th <sup>^</sup> ,v <sup>^</sup> ,z <sup>^</sup> ,zh <sup>^</sup> }
QS "LL-Liquid"	{el <sup>^</sup> ,hh <sup>^</sup> ,l <sup>^</sup> ,r <sup>^</sup> ,w <sup>^</sup> ,y <sup>^</sup> }
QS "LL-Front"	{ae <sup>^</sup> ,b <sup>^</sup> ,eh <sup>^</sup> ,em <sup>^</sup> ,f <sup>^</sup> ,ih <sup>^</sup> ,ix <sup>^</sup> ,iy <sup>^</sup> ,m <sup>^</sup> ,p <sup>^</sup> ,v <sup>^</sup> ,w <sup>^</sup> }
QS "LL-Central"	{ah <sup>^</sup> ,ao <sup>^</sup> ,axr <sup>^</sup> ,d <sup>^</sup> ,dh <sup>^</sup> ,dx <sup>^</sup> ,el <sup>^</sup> ,en <sup>^</sup> ,er <sup>^</sup> ,l <sup>^</sup> ,n <sup>^</sup> ,r <sup>^</sup> ,s <sup>^</sup> ,t <sup>^</sup> ,th <sup>^</sup> ,z <sup>^</sup> ,zh <sup>^</sup> }
QS "LL-Back"	{aa <sup>^</sup> ,ax <sup>^</sup> ,ch <sup>^</sup> ,g <sup>^</sup> ,hh <sup>^</sup> ,jh <sup>^</sup> ,k <sup>^</sup> ,ng <sup>^</sup> ,ow <sup>^</sup> ,sh <sup>^</sup> ,uh <sup>^</sup> ,uw <sup>^</sup> ,y <sup>^</sup> }

QS "LL-Front_Vowel"	{ae^*,eh^*,ey^*,ih^*,iy^*}
QS "LL-Central_Vowel"	{aa^*,ah^*,ao^*,axr^*,er^*}
QS "LL-Back_Vowel"	{ax^*,ow^*,uh^*,uw^*}
QS "LL-Long_Vowel"	{ao^*,aw^*,el^*,em^*,en^*,en^*,iy^*,ow^*,uw^*}
QS "LL-Short_Vowel"	{aa^*,ah^*,ax^*,ay^*,eh^*,ey^*,ih^*,ix^*,oy^*,uh^*}
QS "LL-Diphthong_Vowel"	{aw^*,axr^*,ay^*,el^*,em^*,en^*,er^*,ey^*,oy^*}
QS "LL-Front_Start_Vowel"	{aw^*,axr^*,er^*,ey^*}
QS "LL-Fronting_Vowel"	{ay^*,ey^*,oy^*}
QS "LL-High_Vowel"	{ih^*,ix^*,iy^*,uh^*,uw^*}
QS "LL-Medium_Vowel"	{ae^*,ah^*,ax^*,axr^*,eh^*,el^*,em^*,en^*,er^*,ey^*,ow^*}
QS "LL-Low_Vowel"	{aa^*,ae^*,ah^*,ao^*,aw^*,ay^*,oy^*}
QS "LL-Rounded_Vowel"	{ao^*,ow^*,oy^*,uh^*,uw^*,w^*}
QS "LL-Unrounded_Vowel"	{aa^*,ae^*,ah^*,aw^*,ax^*,axr^*,ay^*,eh^*,el^*,em^*, en^*,er^*,ey^*,hh^*,ih^*,ix^*,iy^*,l^*,r^*,y^*}
QS "LL-Reduced_Vowel"	{ax^*,axr^*,ix^*}
QS "LL-IVowel"	{ih^*,ix^*,iy^*}
QS "LL-EVowel"	{eh^*,ey^*}
QS "LL-AVowel"	{aa^*,ae^*,aw^*,axr^*,ay^*,er^*}
QS "LL-OVowel"	{ao^*,ow^*,oy^*}
QS "LL-UVowel"	{ah^*,ax^*,el^*,em^*,en^*,uh^*,uw^*}
QS "LL-Unvoiced_Consonant"	{ch^*,f^*,hh^*,k^*,p^*,s^*,sh^*,t^*,th^*}
QS "LL-Voiced_Consonant"	{b^*,d^*,dh^*,dx^*,el^*,em^*,en^*,g^*,jh^*, l^*,m^*,n^*,ng^*,r^*,v^*,w^*,y^*}
QS "LL-Front_Consonant"	{b^*,em^*,f^*,m^*,p^*,v^*,w^*}
QS "LL-Central_Consonant"	{d^*,dh^*,dx^*,el^*,en^*,l^*,n^*,r^*,s^*,t^*,th^*,z^*,zh^*}
QS "LL-Back_Consonant"	{ch^*,g^*,hh^*,jh^*,k^*,ng^*,sh^*,y^*}
QS "LL-Fortis_Consonant"	{ch^*,f^*,k^*,p^*,s^*,sh^*,t^*,th^*}
QS "LL-Lenis_Consonant"	{b^*,d^*,dh^*,g^*,jh^*,v^*,z^*,zh^*}
QS "LL-Neigther_F_or_L"	{el^*,em^*,en^*,hh^*,l^*,m^*,n^*,ng^*,r^*,w^*,y^*}
QS "LL-Coronal_Consonant"	{ch^*,d^*,dh^*,dx^*,el^*,en^*,jh^*,l^*,n^*,r^*, s^*,sh^*,t^*,th^*,z^*,zh^*}
QS "LL-Non_Coronal"	{b^*,em^*,f^*,g^*,hh^*,k^*,m^*,ng^*,p^*,v^*,w^*,y^*}
QS "LL-Anterior_Consonant"	{b^*,d^*,dh^*,dx^*,el^*,em^*,en^*,f^*,l^*,m^*,n^*, p^*,s^*,t^*,th^*,v^*,w^*,z^*}

QS "LL-Non_Anterior"	{ch <sup>^</sup> ,g <sup>^</sup> ,hh <sup>^</sup> ,jh <sup>^</sup> ,k <sup>^</sup> ,ng <sup>^</sup> ,r <sup>^</sup> ,sh <sup>^</sup> ,y <sup>^</sup> ,zh <sup>^</sup> }
QS "LL-Continuent"	{dh <sup>^</sup> ,el <sup>^</sup> ,em <sup>^</sup> ,en <sup>^</sup> ,f <sup>^</sup> ,hh <sup>^</sup> ,l <sup>^</sup> ,m <sup>^</sup> ,n <sup>^</sup> ,ng <sup>^</sup> ,r <sup>^</sup> ,s <sup>^</sup> ,sh <sup>^</sup> ,th <sup>^</sup> ,v <sup>^</sup> ,w <sup>^</sup> ,y <sup>^</sup> ,z <sup>^</sup> ,zh <sup>^</sup> }
QS "LL-No_Continuent"	{b <sup>^</sup> ,ch <sup>^</sup> ,d <sup>^</sup> ,g <sup>^</sup> ,jh <sup>^</sup> ,k <sup>^</sup> ,p <sup>^</sup> ,t <sup>^</sup> }
QS "LL-Positive_Strident"	{ch <sup>^</sup> ,jh <sup>^</sup> ,s <sup>^</sup> ,sh <sup>^</sup> ,z <sup>^</sup> ,zh <sup>^</sup> }
QS "LL-Negative_Strident"	{dh <sup>^</sup> ,f <sup>^</sup> ,hh <sup>^</sup> ,th <sup>^</sup> ,v <sup>^</sup> }
QS "LL-Neutral_Strident"	{b <sup>^</sup> ,d <sup>^</sup> ,el <sup>^</sup> ,em <sup>^</sup> ,en <sup>^</sup> ,g <sup>^</sup> ,k <sup>^</sup> ,l <sup>^</sup> ,m <sup>^</sup> ,n <sup>^</sup> ,ng <sup>^</sup> ,p <sup>^</sup> ,r <sup>^</sup> ,t <sup>^</sup> ,w <sup>^</sup> ,y <sup>^</sup> }
QS "LL-Glide"	{hh <sup>^</sup> ,l <sup>^</sup> ,el <sup>^</sup> ,r <sup>^</sup> ,y <sup>^</sup> ,w <sup>^</sup> }
QS "LL-Syllabic_Consonant"	{axr <sup>^</sup> ,el <sup>^</sup> ,em <sup>^</sup> ,en <sup>^</sup> ,er <sup>^</sup> }
QS "LL-Voiced_Stop"	{b <sup>^</sup> ,d <sup>^</sup> ,g <sup>^</sup> }
QS "LL-Unvoiced_Stop"	{p <sup>^</sup> ,t <sup>^</sup> ,k <sup>^</sup> }
QS "LL-Front_Stop"	{b <sup>^</sup> ,p <sup>^</sup> }
QS "LL-Central_Stop"	{d <sup>^</sup> ,t <sup>^</sup> }
QS "LL-Back_Stop"	{g <sup>^</sup> ,k <sup>^</sup> }
QS "LL-Voiced_Fricative"	{jh <sup>^</sup> ,dh <sup>^</sup> ,v <sup>^</sup> ,z <sup>^</sup> ,zh <sup>^</sup> }
QS "LL-Unvoiced_Fricative"	{ch <sup>^</sup> ,f <sup>^</sup> ,s <sup>^</sup> ,sh <sup>^</sup> ,th <sup>^</sup> }
QS "LL-Front_Fricative"	{f <sup>^</sup> ,v <sup>^</sup> }
QS "LL-Central_Fricative"	{dh <sup>^</sup> ,s <sup>^</sup> ,th <sup>^</sup> ,z <sup>^</sup> }
QS "LL-Back_Fricative"	{ch <sup>^</sup> ,jh <sup>^</sup> ,sh <sup>^</sup> ,zh <sup>^</sup> }
QS "LL-Affricate_Consonant"	{ch <sup>^</sup> ,jh <sup>^</sup> }
QS "LL-Not_Affricate"	{dh <sup>^</sup> ,f <sup>^</sup> ,s <sup>^</sup> ,sh <sup>^</sup> ,th <sup>^</sup> ,v <sup>^</sup> ,z <sup>^</sup> ,zh <sup>^</sup> }
QS "LL-silences"	{pau <sup>^</sup> ,h# <sup>^</sup> ,brth <sup>^</sup> }
QS "LL-aa"	{aa <sup>^</sup> }
QS "LL-ae"	{ae <sup>^</sup> }
QS "LL-ah"	{ah <sup>^</sup> }
QS "LL-ao"	{ao <sup>^</sup> }
QS "LL-aw"	{aw <sup>^</sup> }
QS "LL-ax"	{ax <sup>^</sup> }
QS "LL-axr"	{axr <sup>^</sup> }
QS "LL-ay"	{ay <sup>^</sup> }
QS "LL-b"	{b <sup>^</sup> }
QS "LL-ch"	{ch <sup>^</sup> }
QS "LL-d"	{d <sup>^</sup> }

QS "LL-dh"	{dh <sup>^*</sup> }
QS "LL-dx"	{dx <sup>^*</sup> }
QS "LL-eh"	{eh <sup>^*</sup> }
QS "LL-el"	{el <sup>^*</sup> }
QS "LL-em"	{em <sup>^*</sup> }
QS "LL-en"	{en <sup>^*</sup> }
QS "LL-er"	{er <sup>^*</sup> }
QS "LL-ey"	{ey <sup>^*</sup> }
QS "LL-f"	{f <sup>^*</sup> }
QS "LL-g"	{g <sup>^*</sup> }
QS "LL-hh"	{hh <sup>^*</sup> }
QS "LL-hv"	{hv <sup>^*</sup> }
QS "LL-ih"	{ih <sup>^*</sup> }
QS "LL-iy"	{iy <sup>^*</sup> }
QS "LL-jh"	{jh <sup>^*</sup> }
QS "LL-k"	{k <sup>^*</sup> }
QS "LL-l"	{l <sup>^*</sup> }
QS "LL-m"	{m <sup>^*</sup> }
QS "LL-n"	{n <sup>^*</sup> }
QS "LL-nx"	{nx <sup>^*</sup> }
QS "LL-ng"	{ng <sup>^*</sup> }
QS "LL-ow"	{ow <sup>^*</sup> }
QS "LL-oy"	{oy <sup>^*</sup> }
QS "LL-p"	{p <sup>^*</sup> }
QS "LL-r"	{r <sup>^*</sup> }
QS "LL-s"	{s <sup>^*</sup> }
QS "LL-sh"	{sh <sup>^*</sup> }
QS "LL-t"	{t <sup>^*</sup> }
QS "LL-th"	{th <sup>^*</sup> }
QS "LL-uh"	{uh <sup>^*</sup> }
QS "LL-uw"	{uw <sup>^*</sup> }
QS "LL-v"	{v <sup>^*</sup> }
QS "LL-w"	{w <sup>^*</sup> }
QS "LL-y"	{y <sup>^*</sup> }

QS "LL-z" {z^\*}  
QS "LL-zh" {zh^\*}  
QS "LL-pau" {pau^\*}  
QS "LL-h#" {h#^\*}  
QS "LL-brth" {brth^\*}



# B

## MAGE

---

### Contents

---

B.1	Integrations of MAGE . . . . .	175
B.1.1	Pure data & Max MSP object ( <b>mage~</b> ) . . . . .	175
B.1.2	openFrameworks, iOS . . . . .	177
B.2	MAGE demo links . . . . .	178
B.3	License . . . . .	178

---

This Appendix contains detailed information regarding the integrations of MAGE/pHTS, related links and license.

#### B.1 INTEGRATIONS OF MAGE

In this Section we describe the integration of MAGE as a Pure data & Max MSP object as well as an openFrameworks & iOS application, by using just the provided API.

##### B.1.1 *Pure data & Max MSP object (**mage~**)*

The **mage~** external object is implemented with one input and one output. The input is used to receive control messages from the user, whereas the output gives raw speech samples. It starts and initialises the three MAGE threads, as described in [Section 8.1](#). It also creates and initialises all the functions in order to receive and apply the user inputs. Basically, there are two kinds of controls in **mage~** that can be reactively applied to affect the synthesised output. First, the reactive manipulation of the context, that can be applied on the phoneme level. Second, the reactive control over the voice quality, prosody and speaker identity, that can be applied either on the phoneme

level or on the sample level, depending on the nature of the control and the thread that applies it.

In details, the available controls in the **mage~** object for context manipulation are:

```
label alice.lab
labelfill
labelnext
labelinsert 10
labelreplace 54
labelswitch 7
```

where the message `label alice.lab` loads a list of phonetic labels from the label file `alice.lab` and `labelfill` sends all the loaded labels into the **mage~** object so that the processing will start, using one label at a time allowing the user to focus on other controls and not only on the contextual control of the voice. Similarly, `labelnext` sends a label from that list to MAGE and goes to the next label of the list. `labelinsert N` sends the  $N^{\text{th}}$  label of the list to MAGE, `labelreplace N` sends the  $N^{\text{th}}$  label of the list to MAGE and makes `labelnext` jump to its next label. `labelswitch N` sends the  $N^{\text{th}}$  label of the list and makes `labelnext` point to label  $N + 1$ . If a command reaches the end of the list, it loops back to the start.

Additionally, some examples of the available controls in the **mage~** object for voice quality, prosody and model manipulation are:

```
reset
alpha 0.42
speed 2 scale
pitchshift 128
interpolate awb 0.75
interpolate slt 0.25
```

where the message `reset` when received resets the **mage~** object to its default values. `alpha` sets the value of the vocal tract length parameter to 0.42. `speed 2 scale` multiplies the current speed of the speech output by two. `pitchshift 128` shifts / adds 128 Hz to the currently produced pitch trajectory. `interpolate awb 0.75` sets the interpolation weight of the speaker

“awb” to 75% and respectively interpolate slt 0.25 sets the interpolation weight of the speaker “slt” to 25%. Here we see that, we can reactively change the speaker identity by switching gradually and mixing between two voices (i.e. from 100% awb to 100% slt). The usage of the messages is presented in [Figure B.1](#).

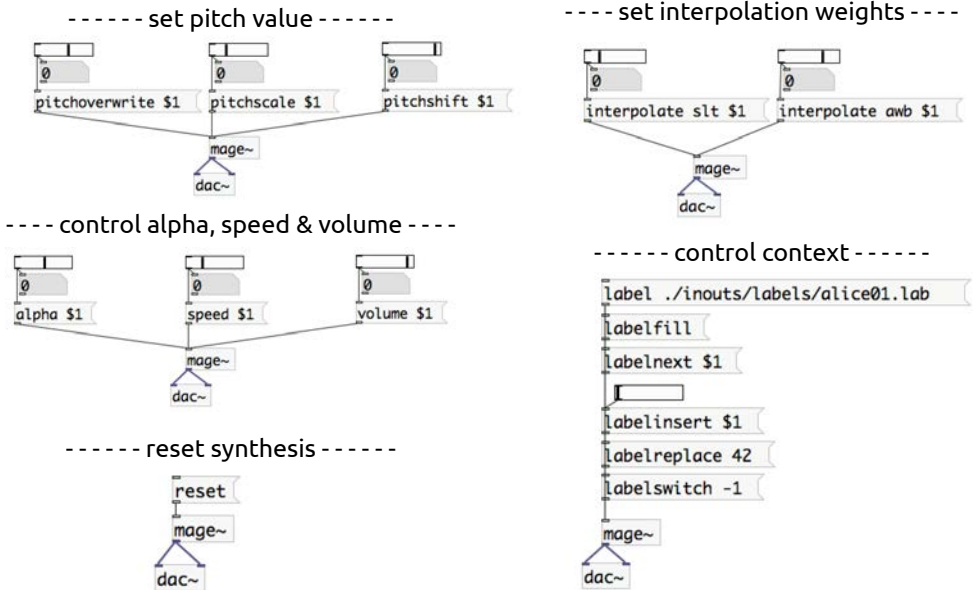


Figure B.1: Context, voice quality, prosody and model control in `mage~`.

### B.1.2 *openFrameworks, iOS*

Another approach to MAGE integration using the above described controls is to create an `openFrameworks` or `iOS` application based on one of the several examples distributed with the source code [AMW<sup>+</sup>]. These small control examples illustrate the various ways through which MAGE parameters can be modified while the speech is synthesised. Besides, a more complete example is available as `of0scControl` which, when started, launches the MAGE threads and permits to control each parameter of the speech synthesis through OSC messages sent from any kind of interface supporting OSC protocol.

## B.2 MAGE DEMO LINKS

- MAGE Platform for Performative Speech Synthesis  
( <http://mage.numediart.org/> )
- HandSketch demo  
( <https://vimeo.com/39558917> )
- FaceOSC demo  
( <https://vimeo.com/39567236> )
- Interactive accent map demo  
( <https://vimeo.com/51985913> )
- Reactive generation of articulatory features demo  
( <https://vimeo.com/67404386> )
- Reactive interpolation of hypo- / hyper- articulated speech demo  
( <https://vimeo.com/53605118> )
- Altering speech synthesis prosody through real time natural gestural control demo  
( <http://vimeo.com/72305725> )
- Talking guitar live demo  
( <https://vimeo.com/100508133> )

## B.3 LICENSE

MAGE / *pHTS* ( *the performative HMM-based speech synthesis system* ) is free software : you can redistribute it and/or modify it under the terms of the *GNU General Public License* as published by the Free Software Foundation, either version 3 of the license, or any later version.

MAGE / *pHTS* is distributed in the hope that it will be useful, but *WITHOUT ANY WARRANTY*; without even the implied warranty of *MERCHANTABILITY* or *FITNESS FOR A PARTICULAR PURPOSE*. See the *GNU General Public License* for more details.

You should have received a copy of the *GNU General Public License* along with MAGE / *pHTS*. If not, see <http://www.gnu.org/licenses>.

Copyright 2011 University of Mons :  
Numediart Institute for New Media Art ( [www.numediart.org](http://www.numediart.org) )  
Acapela Group ( [www.acapela-group.com](http://www.acapela-group.com) )

Developed by :  
[Maria Astrinaki](#), [Alexis Moinet](#), [Geoffrey Wilfart](#),  
[Nicolas d'Alessandro](#) and [Thierry Dutoit](#)



# C

## RELATED PROJECTS

---

### Contents

---

c.1	Speech synthesis projects . . . . .	181
c.2	Speech corpuses . . . . .	182
c.3	Other speech related links . . . . .	182
c.4	Beyond speech projects . . . . .	183
c.5	Fun links . . . . .	183

---

In this Appendix there are some information for several interesting synthesis projects, mainly regarding speech synthesis but not limited to.

### C.1 SPEECH SYNTHESIS PROJECTS

- HMM-based Speech Synthesis System  
( <http://hts.sp.nitech.ac.jp/> )
- HMM-based singing voice synthesis system  
( <http://www.sinsy.jp/> )
- The Voicebank project  
( <http://www.smart-mnd.org/voicebank/about/home.html> )
- Creative Speech Technology (CreST) Network  
( <http://crestnetwork.org.uk/index.php> )
- Ultrax Speech project  
( <http://www.ultrax-speech.org/> )
- Ultrax Speech project : Produce an animated 2D representation of the vocal tract ( <http://homepages.inf.ed.ac.uk/korin/ultraxis2012/> )

## C.2 SPEECH CORPUSES

- International dialects of English archive  
( <http://www.dialectsarchive.com/globalmap> )
- CSTR VCTK Corpus : English Multi-speaker Corpus for the CSTR Voice Cloning Toolkit  
( <http://homepages.inf.ed.ac.uk/jyamagis/page3/page58/page58.html> )
- CSTR NAM TIMIT Plus corpus  
( <http://homepages.inf.ed.ac.uk/jyamagis/page3/page57/page57.html> )
- The speech accent archive  
( <http://accent.gmu.edu/> )

## C.3 OTHER SPEECH RELATED LINKS

- TED Talk “*Roger Ebert: Remaking my voice*”  
( [http://www.ted.com/talks/roger\\_ebert\\_remaking\\_my\\_voice.html](http://www.ted.com/talks/roger_ebert_remaking_my_voice.html) )
- Dialect quiz shows where others talk like you do  
( <http://spark.rstudio.com/jkatz/DialectQuiz/> )
- The International Dialects of English Archive  
( <http://www.dialectsarchive.com/> )
- An average voice is beautiful, say scientists  
( <http://phys.org/news183666976.html> )
- Hearing Voices  
( <http://www.wnyc.org/story/224328-hearing-voices/> )
- Overtone singing with X-ray  
( <http://vimeo.com/20237275> )
- HANDSKETCH  
( <http://handsketch.net/> )

## C.4 BEYOND SPEECH PROJECTS

- Tanukis demo  
( <http://vimeo.com/76850234> )
- Audio-visual Laughter Synthesis  
( <http://tcts.fpms.ac.be/~cakmak/personal/> )
- Examples of synthesized walks  
( <http://tcts.fpms.ac.be/~tilmanne/> )

## C.5 FUN LINKS

- 29 Celebrity Impressions, 1 Original Song - Rob Cantor  
( <https://www.youtube.com/watch?v=k6PxMRUgmbA> )
- Funny French Guy Learning American Accent  
( <https://www.youtube.com/watch?v=zeV0ag5z2-0> )
- The Italian Man Who Went to Malta  
( <https://www.youtube.com/watch?v=m9tyqeIfY00> )
- Burnistoun Voice Recognition Elevator  
( <https://www.youtube.com/watch?v=QGQHzIQ1jV0> )
- Jamie Costa's Vine Impressions of Robin Williams  
( <https://www.youtube.com/watch?v=VKQsI1yQFwE> )



## BIBLIOGRAPHY

---

- [1] D. E. Knuth, "Computer Programming as an Art," *Communications of the ACM*, vol. 17, pp. 667–673, December 1974.
- [2] K. Tokuda, Y. Nankaku, T. Toda, H. Zen, J. Yamagishi, and K. Oura, "Speech Synthesis Based on Hidden Markov Models," *Proceedings of the IEEE*, vol. 101, pp. 1234–1252, May 2013.
- [3] H. Zen, K. Tokuda, and A. W. Black, "Statistical parametric speech synthesis," *Speech Communication*, vol. 51, no. 11, pp. 1039–1064, 2009.
- [4] K. Tokuda, H. Zen, J. Yamagishi, A. W. Black, T. Masuko, S. Sako, T. Toda, T. Nose, and K. Oura, "The HMM-based Speech Synthesis System (HTS)." <http://hts.sp.nitech.ac.jp>. visited: June 2014.
- [5] W. von Kempelen, *Mechanismus der menschlichen Sprache nebst der Beschreibung seiner sprechenden Maschine*. J. B. Degen, 1791.
- [6] H. Dudley, "The Carrier Nature of Speech," *Bell System Tech*, pp. 495–515, 1940.
- [7] J. L. Kelly and C. Lochbaum, "Speech synthesis," in *Proceedings of 4th International Congress on Acoustics*, pp. 1–4, 1962.
- [8] A. W. Black and K. A. Lenzo, "Building Synthetic Voices," tech. rep., Language Technologies Institute, Carnegie Mellon University, 2007.
- [9] T. Dutoit, V. Pagel, N. Pierret, F. Bataille, and O. van der Vrecken, "The MBROLA Project: Towards a set of high quality speech synthesizers of use for non commercial purposes," in *Proceedings of ICSLP*, 1996.
- [10] S. Lemmetty, "Review of speech synthesis technology," Master's thesis, Laboratory of Acoustics and Audio Signal Processing, Helsinki University of Technology, Finland, 1999.
- [11] J. L. Flanagan, "Speech analysis: Synthesis and perception," 1972.
- [12] "Joseph Faber's Talking Euphonia." <http://goo.gl/R3cpX0>. visited: June 2014.
- [13] A. G. Bell, "Making a talking machine," *Beinn Bhreagh Recorder*, pp. 61–72, 1909.
- [14] K. H. Dunn, "Calculation of vowel resonances and an electrical vocal tract," *Acoustical Society of America*, vol. 22, pp. 740–53, 1950.

- [15] N. K. Stevens, S. Kasowski, C. Fant, and M. Gunnar, "An electrical analog of the vocal tract," *Acoustical Society of America*, vol. 25, pp. 734–42, 1953.
- [16] J. Allen, S. Hunnicutt, and D. Klatt, *From Text to Speech: The MITalk System*. Cambridge University Press, 1987.
- [17] P. Rubin, T. Baer, A. Black, and P. Mermelstein, "An articulatory synthesizer for perceptual research," *Journal of the Acoustical Society of America*, pp. 321–328, 1981.
- [18] P. Rubin, E. Saltzman, L. Goldstein, R. McGowan, M. Tiede, and C. Browman, "CASy and extensions to the task-dynamic model," in *Proceedings of the 1st ESCA Tutorial and Research Workshop on Speech Producing Modeling - 4th Speech Production Seminar*, 1996.
- [19] J. Lloyd, I. Stavness, and S. Fels, "The ArtiSynth Toolkit For Rigid-Deformable Biomechanics," in *Proceedings of ISB Technical Group on Computer Simulation Symposium*, 2011.
- [20] R. Carlson and B. Granström, "A text-to-speech system based entirely on rules," in *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP'76)*, 1976.
- [21] T. Styger and E. Keller, *Fundamentals of Speech Synthesis and Speech Recognition: Basic Concepts, State of the Art, and Future Challenges*. Chichester: John Wiley, 1994.
- [22] C. R., S. T., and S. A., "Data-driven formant synthesis," Tech. Rep. TMH-QPSR44: 121–124, Speech, Music and Hearing part of School of Computer Science and Communication (KTH), 2002.
- [23] G. Fant, "Speech communication research," *Royal Swedish Academy of Engineering Sciences*, vol. 2, pp. 331–337, 1953.
- [24] R. Carlson and B. Granstrom, "A Text-to-Speech system based on a phonetically oriented programming language," Tech. Rep. SLT-QPSR: 1–4, Speech, Music and Hearing part of School of Computer Science and Communication (KTH), 1975.
- [25] D. Klatt, "The klattalk text-to-speech conversion system," in *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP'82)*, vol. 7, pp. 1589–1592, 1982.

- [26] F. Charpentier and M. Stella, "Diphone Synthesis Using an Overlap-Add Technique for Speech Waveforms Concatenation," in *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP'86)*, vol. 11, pp. 2015–2018, 1986.
- [27] E. Moulines and W. Verhelst, *Speech Coding and Synthesis*. Eds. Amsterdam : Elsevier Science B. V., 1995.
- [28] "MBROLA Project Development Team, Faculte Polytechnique de Mons, MULTITEL-TCTS Lab." <http://tcts.fpms.ac.be/synthesis/mbrola.html>. visited: June 2014.
- [29] M. W. Macon, L. Jensen-Link, J. Oliverio, M. Clements, and E. B. George, "Concatenation-Based MIDI-to-Singing Voice Synthesis," in *Proceedings of Audio Engineering Society International Conference*, vol. 103, 1997.
- [30] Yamaha, "Vocaloid." <http://www.vocaloid.com>. visited: June 2014.
- [31] A. W. Black, "CMU ARCTIC speech synthesis databases." [http://festvox.org/cmu\\_arctic](http://festvox.org/cmu_arctic). visited: June 2014.
- [32] J. Kominek and A. W. Black, "CMU ARCTIC databases for speech synthesis," Tech. Rep. CMU-LTI-03-177, Language Technologies Institute, School of Computer Science, Carnegie Mellon University, 2003.
- [33] P. Taylor, *Text-to-Speech Synthesis*. Cambridge University Press, 2009.
- [34] Y. Sagisaka, N. Kaiki, N. Iwahashi, and K. Mimura, "ATR v-TALK speech synthesis system," in *Proceedings of ICSLP*, pp. 483–486, 1992.
- [35] A. Hunt and A. Black, "Unit Selection in a Concatenative Speech Synthesis System Using a Large Speech Database," in *Proceedings of IEEE International Conference of Acoustics, Speech, and Signal Processing*, pp. 373–376, 1996.
- [36] Y. Meron, *High Quality Singing Synthesis using the Selection-Based Synthesis Scheme*. PhD thesis, University of Tokyo, 1999.
- [37] D. Schwarz, G. Beller, B. Verbrugghe, and S. Britton, "Real-time corpus-based concatenative synthesis with Catart," in *Proceedings of the 9th International Conference on Digital Audio Effects (DAFx-06)*, pp. 279–282, 2006.
- [38] A. Black, "Unit selection and emotional speech," in *Proceedings of Eurospeech*, pp. 1649–1652, 2003.
- [39] J. Yamagishi, *Average-voice-based speech synthesis*. PhD thesis, Tokyo Institute of Technology, 2006.

- [40] T. Yoshimura, T. Masuko, K. Tokuda, T. Kobayashi, and T. Kitamura, "Speaker interpolation for HMM-based speech synthesis system," *Journal of the Acoustic Society of Japan*, vol. 21, no. 4, pp. 199–206, 2000.
- [41] K. Shichiri, A. Sawabe, K. Tokuda, T. Masuko, T. Kobayashi, and T. Kitamura, "Eigenvoices for HMM-based speech synthesis," in *Proceedings of International Conference on Spoken Language Processing*, pp. 1269–1272, 2002.
- [42] T. Yoshimura, K. Tokuda, T. Masuko, T. Kobayashi, and T. Kitamura, "Simultaneous modeling of spectrum, pitch and duration in HMM-based speech synthesis," in *Proceedings of Eurospeech*, pp. 2347–2350, 1999.
- [43] T. Toda and K. Tokuda, "A speech parameter generation algorithm considering global variance for HMM-based speech synthesis," *IEICE transactions on Information and Systems*, vol. 90, no. 5, pp. 816–824, 2007.
- [44] Y.-J. Wu and K. Tokuda, "Minimum generation error training by using original spectrum as reference for log spectral distortion measure," in *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP'09)*, pp. 4013–4016, IEEE, 2009.
- [45] T. Nose, M. Tachibana, and T. Kobayashi, "HMM-based style control for expressive speech synthesis with arbitrary speaker's voice using model adaptation," *IEICE transactions on Information and Systems*, vol. 92, no. 3, pp. 489–497, 2009.
- [46] K. Tokuda, T. Kobayashi, T. Masuko, T. Kobayashi, and T. Kitamura, "Speech Parameter Generation Algorithms for HMM-Based Speech Synthesis," in *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP'00)*, vol. 3, pp. 1315–1318, IEEE, 2000.
- [47] K. Koishida, K. Tokuda, T. Masuko, and T. Kobayashi, "Vector quantization of speech spectral parameters using statistics of static and dynamic features," *IEICE transactions on Information and Systems*, vol. 84, no. 10, pp. 1427–1434, 2001.
- [48] T. Fukada, K. Tokuda, T. Kobayashi, and S. Imai, "An adaptive algorithm for mel-cepstral analysis of speech," in *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP'92)*, vol. 1, pp. 137–140, IEEE, 1992.
- [49] S. Young, G. Evermann, M. Gales, T. Hain, D. Kershaw, X. Y. Liu, G. Moore, J. Odell, D. Ollason, D. Povey, V. Valtchev, and P. Woodland, "The Hidden Markov Model Toolkit (HTK)." <http://htk.eng.cam.ac.uk/>. visited: June 2014.
- [50] K. Ross and M. Ostendorf, "A dynamical system model for generating Fo for synthesis," in *Proceedings of the Second ESCA/IEEE Workshop on Speech Synthesis*, pp. 131–134, 1994.

- [51] K. Tokuda, T. Masuko, N. Miyazaki, and T. Kobayashi, "Multi-space probability distribution hmm," *IEICE transactions on Information and Systems*, vol. 85, no. 3, pp. 455–464, 2002.
- [52] K. Tokuda, H. Zen, and A. W. Black, "An HMM-based speech synthesis system applied to English," in *Proceedings of IEEE Workshop on Speech Synthesis*, pp. 227–230, IEEE, 2002.
- [53] S. Young, G. Evermann, M. Gales, T. Hain, D. Kershaw, G. Moore, J. Odell, D. Olason, D. Povey, V. Valtchev, and P. Woodland, *The HTK book (for HTK Version 3.4)*, vol. 2. Entropic Cambridge Research Laboratory Cambridge, 2006.
- [54] S. Imai, K. Sumita, and C. Furuichi, "Mel log spectrum approximation (MLSA) filter for speech synthesis," *Electronics and Communications in Japan (Part I: Communications)*, vol. 66, no. 2, pp. 10–18, 1983.
- [55] J. Gauvain and C. Lee, "Maximum a posteriori estimation for multivariate Gaussian mixture observations of Markov chains," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 2, no. 2, pp. 291–298, 1994.
- [56] C. J. Leggetter and P. C. Woodland, "Maximum likelihood linear regression for speaker adaptation of continuous density hidden Markov models," *Computer Speech & Language*, vol. 9, no. 2, pp. 171–185, 1995.
- [57] M. Tachibana, J. Yamagishi, T. Masuko, and T. Kobayashi, "Speech synthesis with various emotional expressions and speaking styles by style interpolation and morphing," *IEICE transactions on Information and Systems*, vol. 88, no. 11, pp. 2484–2491, 2005.
- [58] M. Pucher, D. Schabus, J. Yamagishi, F. Neubarth, and V. Strom, "Modeling and interpolation of Austrian German and Viennese dialect in HMM-based speech synthesis," *Speech Communication*, vol. 52, no. 2, pp. 164–179, 2010.
- [59] T. Nose, J. Yamagishi, T. Masuko, and T. Kobayashi, "A style control technique for HMM-based expressive speech synthesis," *IEICE transactions on Information and Systems*, vol. 90, no. 9, pp. 1406–1413, 2007.
- [60] 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*, vol. 17, no. 6, pp. 1171–1185, 2009.
- [61] Z.-H. Ling, K. Richmond, and J. Yamagishi, "Articulatory control of HMM-based parametric speech synthesis using feature-space-switched multiple regression," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 21, no. 1, pp. 207–219, 2013.

- [62] A. W. Black and T. Schultz, "Speaker clustering for multilingual synthesis," in *Proceedings of Multilingual Speech and Language Processing*, 2006.
- [63] J. Yamagishi, Z. Ling, and S. King, "Robustness of HMM-based speech synthesis," in *Proceedings of Interspeech 2008*, pp. 581–584, 2008.
- [64] F. Lagona, A. Maruotti, and M. Picone, *A Non-Homogeneous Hidden Markov Model for the Analysis of Multi-Pollutant Exceedances Data, Hidden Markov Models, Theory and Applications*. InTech, 2011.
- [65] "Sinsy - HMM-based Singing Voice Synthesis System." <http://sinsy.jp/>. visited: June 2014.
- [66] J. Urbain, H. Cakmak, and T. Dutoit, "Evaluation of HMM-based laughter synthesis," in *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP'13)*, pp. 7835–7839, IEEE, 2013.
- [67] J. Tilmanne, A. Moinet, and T. Dutoit, "Stylistic gait synthesis based on hidden Markov models," *EURASIP Journal on advances in signal processing*, vol. 2012, no. 1, pp. 1–14, 2012.
- [68] H. Cakmak, J. Urbain, J. Tilmanne, and T. Dutoit, "Evaluation of HMM-based visual laughter synthesis," in *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP'14)*, pp. 4578–4582, IEEE, 2014.
- [69] H. Kawahara, "A style control technique for HMM-based expressive speech synthesis," *Acoustical science and technology*, vol. 27, no. 6, pp. 349–353, 2006.
- [70] T. Drugman and T. Dutoit, "The deterministic plus stochastic model of the residual signal and its applications," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 20, no. 3, pp. 968–981, 2012.
- [71] J. Urbain, *Acoustic Laughter Processing*. PhD thesis, University of Mons, 2014.
- [72] H. Cakmak, J. Urbain, and T. Dutoit, "The AV-LASYN Database : A synchronous corpus of audio and 3D facial marker data for audio-visual laughter synthesis," in *Proceedings of the 9th International Conference on Language Resources and Evaluation (LREC'14)*, 2014.
- [73] J. Tilmanne, S. Hidot, and T. Ravet, "MockKey: motion capture as a tool for keyframing animation," Tech. Rep. 4, Numediart Institute, Mons, Belgium, December 2009.
- [74] A. W. Black and K. A. Lenzo, "Building synthetic voices," *Language Technologies Institute, Carnegie Mellon University and Cepstral LLC*, 2003.

- [75] K. Tokuda, T. Kobayashi, and S. Imai, "Recursive calculation of mel-cepstrum from lp coefficients," *IEICE transactions on Information and Systems*, vol. 71, pp. 128–131, 1994.
- [76] J. Yamagishi and T. Kobayashi, "Average-voice-based speech synthesis using HSMM-based speaker adaptation and adaptive training," *IEICE transactions on Information and Systems*, vol. 90, no. 2, pp. 533–543, 2007.
- [77] K. K. Paliwal and B. S. Atal, "Efficient vector quantization of LPC parameters at 24 bits/frame," *IEEE Transactions on Speech and Audio Processing*, vol. 1, no. 1, pp. 3–14, 1993.
- [78] "Speech Signal Processing Toolkit (SPTK)." <http://sp-tk.sourceforge.net/>. visited: June 2014.
- [79] B. Picart, T. Drugman, and T. Dutoit, "Continuous Control of the Degree of Articulation in HMM-Based Speech Synthesis," in *Proceedings of Interspeech 2011*, pp. 1797–1800, 2011.
- [80] J. Yamagishi, T. Masuko, and T. Kobayashi, "HMM-based expressive speech synthesis - Towards TTS with arbitrary speaking styles and emotions," in *Proceedings of Special Workshop in Maui (SWIM)*, 2004.
- [81] M. Tachibana, J. Yamagishi, K. Onishi, T. Masuko, and T. Kobayashi, "HMM-based speech synthesis with various speaking styles using model interpolation," in *Speech Prosody 2004, International Conference*, 2004.
- [82] B. Picart, T. Drugman, and T. Dutoit, "Analysis and Synthesis of Hypo and Hyperarticulated Speech," in *Proceedings of Speech Synthesis Workshop 7 (SSW7)*, 2010.
- [83] "The Voicebank Project." <http://www.smart-mnd.org/voicebank/about/introduction.html>. visited: June 2014.
- [84] Y. Qian, Z. Wu, B. Gao, and F. Soong, "Improved prosody generation by maximizing joint probability of state and longer units," in *Proceedings of Audio, Speech, and Language Processing, IEEE Transactions*, vol. 19, pp. 1702–1710, August 2011.
- [85] Ö. Çetin and M. Ostendorf, "Multi-Rate and Variable-Rate Modeling of Speech At Phone and Syllable Time Scales," in *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP'05)*, vol. 1, pp. 665–668, 2005.
- [86] K. Yoshizato, H. Kameoka, D. Saito, and S. Sagayama, "Statistical approach to Fujisaki-model parameter estimation from speech signals and its quantitative evaluation," in *Proceedings of Speech Prosody*, 2012.

- [87] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum Likelihood from incomplete data via the EM algorithm," *Journal of the Royal Statistical Society. Series B (Methodological)*, vol. 39, no. 1, pp. 1–38, 1977.
- [88] Z.-H. Ling, Y.-J. Wu, Y.-P. Wang, L. Qin, and R.-H. Wang, "USTC system for Blizzard Challenge 2006 an improved HMM-based speech synthesis method," in *Blizzard Challenge Workshop, 2006*.
- [89] N. d’Alessandro and T. Dutoit, "HandSketch Bi-Manual Controller: Investigation on Expressive Control Issues of an Augmented Tablet," in *Proceedings of the 7th International Conference on New Instruments for Musical Expression (NIME’07)*, pp. 78–81, 2007.
- [90] K. McDonald, "FaceOSC." <http://vimeo.com/26098366>. visited: June 2014.
- [91] L. Reboursière, O. Lähdeoja, T. Drugman, S. Dupont, C. Picard, and N. Riche, "Left and right-hand guitar playing techniques detection," in *Proceedings of the International Conference on New Instruments for Musical Expression (NIME’12)*, 2012.
- [92] K. Richmond and S. Renals, "Ultrax: An Animated Midsagittal Vocal Tract Display for Speech Therapy," in *Proceedings of Interspeech 2012*, 2012.
- [93] Z. Zhang, "Microsoft kinect sensor and its effect," *IEEE MultiMedia*, vol. 19, no. 2, pp. 4–10, 2012.
- [94] A. Edwards and C. Newell, "Creative Speech Technology (CreST) Network." <http://crestnetwork.org.uk/index.php>. visited: June 2014.
- [95] P. Nemeth, G. Papp, and B. Samu, "Animata - Realtime Animation Editor." <http://animata.kibu.hu/>. visited: June 2014.
- [96] F. Zajéga, "Tanukis." <http://www.frankiezafe.org/index.php?id=239>. visited: June 2014.
- [97] D. Abelman, "Altering speech synthesis prosody through real time natural gestural control," Master’s thesis, The University of Edinburgh, 2013.
- [98] J. McCartney, "SuperCollider : Real-time audio synthesis and algorithmic composition." <http://supercollider.sourceforge.net/>. visited: June 2014.
- [99] A. Puleo, "MageCollider – Integration of the Real-Time Speech Synthesis Tool MAGE into SuperCollider Musical Programming Language," Master’s thesis, Faculty of Engineering of the University of Mons, Belgium, 2014.
- [100] "Blender." <http://www.blender.org/>. visited: June 2014.

- [101] J. Tilmanne and T. Dutoit, "Continuous control of style through linear interpolation in Hidden Markov Model based stylistic walk synthesis," in *International Conference on Cyberworlds (CW)*, pp. 232–236, IEEE, 2011.
- [102] Z. Lieberman, T. Watson, and e. Arturo Castro, "openFrameworks." <http://www.openframeworks.cc>. visited: June 2014.
- [103] M. Puckette, "Pure data." <http://puredata.info/>. visited: June 2014.
- [104] D. E. Knuth, "The art of computer programming," *Communications of the ACM*, vol. 3 (2nd ed.) : sorting and searching, 1998.
- [105] J. van Leeuwen, *Handbook of Theoretical Computer Science*, vol. A : Algorithms and Complexity. Elsevier and MIT Press, 1990.
- [106] N. d'Alessandro, C. Ooge, T. Dutoit, and S. Fels, "Analysis-by-Performance: Gesturally-Controlled Voice Synthesis as an Input for Modelling of Vibrato in Singing," in *Proceedings of the ICMC 2011-International Computer Music Conference*, 2011.
- [107] "iOS Dev Center." <https://developer.apple.com/devcenter/ios/index.action>. visited: June 2014.
- [108] Cycling 74, "Max msp." <http://cycling74.com/products/max/>. visited: June 2014.
- [109] A. Schmeder, A. Freed, and D. Wessel, "opensoundcontrol.org." <http://opensoundcontrol.org>. visited: June 2014.



## PUBLICATIONS

---

- [ABd<sup>+</sup><sub>11a</sub>] M. Astrinaki, O. Babacan, N. d’Alessandro, T. Dutoit, and S. Fels. “MAGE/pHTS in Collaborative Vocal Puppetry (CoVoP) - Multi-User performative HMM-based Voice Synthesis on Distributed Platforms”. In *Proceedings of the 7th International Summer Workshop on Multimodal Interfaces - eINTERFACE’11*, Pilsen, Czech Republic, August 2011.
- [ABd<sup>+</sup><sub>11b</sub>] M. Astrinaki, O. Babacan, N. d’Alessandro, T. Dutoit, and S. Fels. “MAGE/pHTS in Collaborative Vocal Puppetry (CoVoP) - Multi-User performative HMM-based Voice Synthesis on Distributed Platforms”. Technical Report 3, Numediart Institute, Mons, Belgium, September 2011.
- [ABd<sup>+</sup><sub>11c</sub>] M. Astrinaki, O. Babacan, N. d’Alessandro, B. Picart, and T. Dutoit. “pHTS for Max/MSP: A Streaming Architecture for Statistical Parametric Speech Synthesis”. Technical Report 1, Numediart Institute, Mons, Belgium, March 2011.
- [ABdD<sub>11</sub>] M. Astrinaki, O. Babacan, N. d’Alessandro, and T. Dutoit. “sHTS: A Streaming Architecture for Statistical Parametric Speech Synthesis”. In *First International Workshop on Performative Speech and Singing Synthesis (p3s-2011)*, Vancouver, BC, Canada, March 2011.
- [Add<sub>12a</sub>] M. Astrinaki, N. d’Alessandro, and T. Dutoit. “MAGE - A Platform for Tangible Speech Synthesis”. In *Proceedings of the 12th Conference on New Interfaces for Musical Expression (NIME’12)*, pages 353–356, Ann Arbor, Michigan, USA, May 2012.
- [Add<sub>12b</sub>] M. Astrinaki, N. d’Alessandro, and T. Dutoit. “MAGEFaceOSC: Performative Speech Synthesis Based On Realtime Face Tracking”. Technical Report 1, Numediart Institute, Mons, Belgium, March 2012.
- [AdP<sup>+</sup><sub>12</sub>] M. Astrinaki, N. d’Alessandro, B. Picart, T. Drugman, and T. Dutoit. “Reactive and Continuous Control of HMM-based Speech Synthesis”. In *IEEE Workshop on Spoken Language Technology (SLT 2012)*, Miami, Florida, USA, December 2012.
- [AdR<sup>+</sup><sub>13</sub>] M. Astrinaki, N. d’Alessandro, L. Reboursiere, A. Moinet, and T. Dutoit. “MAGE 2.0: New features and its application in the development

- of a talking guitar". In *Proceedings of the 13th Conference on New Interfaces for Musical Expression (NIME'13)*, pages 547–550, Daejeon, Seoul, Korea Republic, May 2013.
- [AMdD13] M. Astrinaki, A. Moinet, N. d'Alessandro, and T. Dutoit. "Pure Data External for Reactive HMM-based Speech and Singing Synthesis". In *Proceedings of the 16th International Conference on Digital Audio Effects (DAFx-13)*, pages 78–81, Maynooth, Ireland, September 2013.
- [AMW<sup>+</sup>] M. Astrinaki, A. Moinet, G. Wilfart, N. d'Alessandro, and T. Dutoit. "Mage Platform for Performative Speech Synthesis". <http://mage.numediart.org/>. visited: June 2014.
- [AMY<sup>+</sup>13a] M. Astrinaki, A. Moinet, J. Yamagishi, K. Richmond, Z.-H. Ling, S. King, and T. Dutoit. "MAGE - Reactive articulatory feature control of HMM-based parametric speech synthesis". In *Proceedings of the 8th ISCA Speech Synthesis Workshop (SSW8)*, page 243 (poster), Barcelona, Spain, September 2013.
- [AMY<sup>+</sup>13b] M. Astrinaki, A. Moinet, J. Yamagishi, K. Richmond, Z.-H. Ling, S. King, and T. Dutoit. "MAGE - Reactive articulatory feature control of HMM-based parametric speech synthesis". In *Proceedings of the 8th ISCA Speech Synthesis Workshop (SSW8)*, pages 207–212, Barcelona, Spain, September 2013.
- [ARM<sup>+</sup>12] M. Astrinaki, L. Reboursiere, A. Moinet, R. Graham, N. d'Alessandro, and T. Dutoit. "Is This Guitar Talking or What !?". In *Proceedings of the 8th International Summer Workshop on Multimodal Interfaces - eNTerFACE'12*, pages 47–55, Metz, France, July 2012.
- [AYdD13] M. Astrinaki, J. Yamagishi, N. d'Alessandro, and T. Dutoit. "reactive accent interpolation through an interactive map application". In *Proceedings of the 14th Conference of the International Speech Communication Association (Interspeech 2013)*, Lyon, France, August 2013.
- [CKAY13] A. J. R. Clark, M. A. Konkiewicz, M. Astrinaki, and J. Yamagishi. "Reactive Control of Expressive Speech Synthesis Using Kinect Skeleton Tracking". Technical report, 2013.
- [dAD13] N. d'Alessandro, M. Astrinaki, and T. Dutoit. "MAGEFace: Performative Convension of Facial Characteristics into Speech Synthesis Parameters". In *Proceedings of the 5th International Conference on Intelligent Technologies for Interactive Entertainment (INTETAIN 2013)*, pages 179–182, Mons, Belgium, July 2013.

- [dBAW<sub>12</sub>] N. d’Alessandro, O. Babacan, M. Astrinaki, and J. Wang. “PEACH: An Optimized Framework for Performative Pitch-Synchronous Overlap-Add Sound Synthesis Using Point-Cloud Visualization”. Technical Report 4, Numediart Institute, Mons, Belgium, December 2012.
- [dT<sup>+</sup><sub>13</sub>] N. d’Alessandro, J. Tilmanne, M. Astrinaki, T. Hueber, R. Dall, T. Ravet, A. Moinet, H. Cakmak, O. Babacan, A. Barbulescu, V. Parfait, V. Huguenin, E. S. Kalayci, and Q. Hu. “Reactive Statistical Mapping: Towards the Sketching of Performative Control with Data”. *IFIP Advances in Information and Communication Technology (AICT) - eNTerFACE’13*, 2013.
- [TdAR<sub>14</sub>] J. Tilmanne, N. d’Alessandro, M. Astrinaki, and T. Ravet. “Exploration of a Stylistic Motion Space Through Realtime Synthesis”. In *Proceedings of the 9th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications (VISIGRAPP 2014), Special Session on ICT-based Analysis and Modeling of Intangible Cultural Heritage (IAMICH 2014)*, Lisbon, Portugal, January 2014.
- [TdR<sup>+</sup><sub>14</sub>] J. Tilmanne, N. d’Alessandro, T. Ravet, M. Astrinaki, and A. Moinet. “Full-Body Gait Reconstruction Using Covariance-Based”. In *Proceedings of the 50th annual convention of the AISB (AISB-50)*, London, UK, April 2014.
- [VAO<sup>+</sup><sub>13</sub>] C. Veaux, M. Astrinaki, K. Oura, A. J. R. Clark, and J. Yamagishi. “Gesture Control of HMM-Based Singing Voice Synthesis”. In *Proceedings of the 8th ISCA Speech Synthesis Workshop (SSW8)*, pages 247–248, Barcelona, Spain, September 2013.



# REASONS WHY PEOPLE WHO WORK WITH COMPUTERS SEEM TO HAVE A LOT OF SPARE TIME...

eviljagme.com

Web Developer



'Its uploading'

Sysadmin



'Its rebooting'

Programmer

Hacker



'Its scripted'

3D Artist



'Its rendering'

IT Consultant



'Its your problem now'

Speech Synthesis



'Its training'

## COLOPHON

This document was typeset using the typographical look-and-feel classicthesis developed by André Miede. The style was inspired by Robert Bringhurst's seminal book on typography "*The Elements of Typographic Style*". classicthesis is available for both  $\LaTeX$  and  $\text{LyX}$ :

<http://postcards.miede.de/>