

Speech recognition and synthesis

1 TTS and ASR: A synthesis

- Introduction
- ASR
- HMM based TTS
- Bibliography

Copyright ©2007-2008 R.J.J.H. van Son, GNU General Public License [FSF(1991)]



Introduction

Classical *New and Improved*

TTS: Text \Rightarrow Accents \Rightarrow Phonemes \Rightarrow Prosody \Rightarrow Sound ASR: Text \Leftarrow Accents \Leftarrow Phonemes \Leftarrow Prosody \Leftarrow Sound ?

ASR: Sound \Rightarrow MFCC \Rightarrow HMM \Rightarrow Language Model \Rightarrow Text TTS: Sound \Leftarrow MFCC \Leftarrow HMM \Leftarrow Language Model \Leftarrow Text ?

TTS and ASR: Recognition by synthesis, or synthesize by recognition

- Both synthesis and recognition work by comparing speech to a stored model
- Recognition works by synthesizing speech
- Synthesis works by reproducing stored speech
- Can a synthesizer really be used to recognize?
- Can a recognizer really be used to synthesize?



Introduction: The computational challenge

There is no data like more data, but how to use it?

- Computers become exponentially faster over time
- Speech corpora become exponentially larger over time
- Current HMM speech recognition only marginally better than 10 years ago
- Current synthesis idem
- How can computer speed and corpus size be harnessed?



ASR: Standard HMM

Problems in HMM

- Conditional Independent and Identical Distribution (IID)
- Speech can be described as a sequence of discrete units (phonemes)
- Strip all non-verbal (indexial) information
- Cannot use indexial information (eg, coarticulation)
- Adapt to rate, hypo/hyperarticulation
- \Rightarrow standard HMM models cannot store enough information



ASR: Structure-based approach

Model the parameters of speech production

- Establish mathematical models for stochastic trajectories or segments
- Eg, piecewise polynomials, linear dynamic systems, nonlinear dynamic systems
- Model speech dynamics i.o. acoustics (hypo/hyperarticulation, rate)
- Hidden dynamic models look at articulation
⇒ Articulatory Synthesis
- Combine hidden dynamic vectors with observed acoustic feature vectors
- ⇒ Explicitly model and train other factors



ASR: Template based recognition

Speech contains two types of information

- Verbal information
- Indexial (non-verbal) information
- Words versus Form
- HMM handles the words, but not the Form
- Form includes F_0 and speaking rate
- Form also includes speaker specific information
- Fine phonetic detail can influence recognition
- Eg, 1st syllable of *ham* versus *hamster*
- ASR model needed that can handle indexial information



ASR: Speech recognition by unit synthesis

HMM derives abstract model from examples

- Don't abstract, use examples directly
- Store speech of many speakers
- For each speaker, store lots of speech (words)
- Store different styles of speech, and label them
- \Rightarrow Template-, exemplar-, instance- based ASR



ASR: Template based

Store as much speech as possible

- Add transcriptions and labels
- Store all indexial and textual information
- What was said, by whom, and how
- Words are stored as many example feature vectors “trajectories”
- Preserving as many details as possible
- Incoming signal is compared to sequences of trajectories



ASR: Cookbook

Train knowledge sources of a template based recognizer

- 1 Segment and transcribe the training database.
The result is a segmentation file with phoneme transcriptions and phonetic boundary timings
- 2 Merge consecutive segments at will to produce supra-phonemic templates
- 3 Determine meta-information for each segment
- 4 Determine suitable transition costs to each pair of possible meta-tags
- 5 Assign suitable acoustic scaling matrices for each frame in the database
- 6 Calculate an indexing structure for fast k-nearest neighbours selection
- 7 Compute weights to combine the different knowledge sources



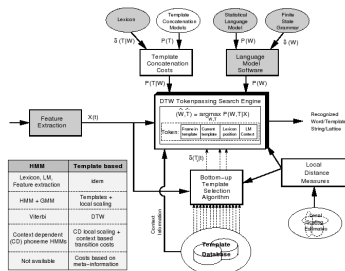
ASR: Template definition

A template is the representation of an actual segment of speech. It consists of

- a sequence of consecutive acoustic feature vectors (or frames)
- a transcription of the sounds or words it represents (typically one or more phonetic symbols)
- knowledge of neighbouring templates (a template number if no templates overlap)
- a tag with meta-information



ASR: Architectural overview



Correspondence between HMM and Template based ASR

- Distances are calculated to template cluster centroids
- Clustered acoustic vectors are a kind of degenerated HMM
- Train the costs of going from one template (fragment) to another

[De Wachter et al.(2007)De Wachter, Matton, Demuyne, Wambacq, Cools, and Van Compernelle]



HMM based TTS: The challenge of TTS

There is never enough speech

- Every utterance and situation are different
- “Expressive” speech needs even more different utterances
- Recording a new speaker for every new application is not acceptable
- Speaker time becomes limiting factor
- ... and it is never enough



HMM based TTS: Speech synthesis by HMM recognition

Unit selection is inflexible

- Speech units cannot be adapted to needs
- Abstract from specific speaker and example
- Model speech stochastically and select most likely utterance



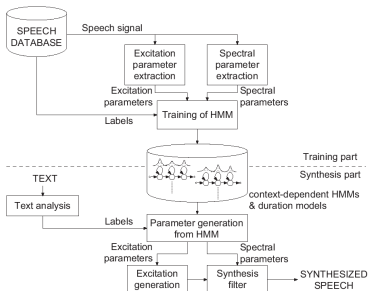
HMM based TTS: HMM models

Add a lot of flexibility

- HMM states can average over a cluster of contexts
- Store the dynamics of spectral change etc.
- HMM models for a new speaker can be learned
- New speaker or language *only* needs the difference
- The difference can be determined on just a little speech
- HMM TTS is *adaptable*
- (and you can indeed synthesize *MFCC* vectors)



HMM based TTS

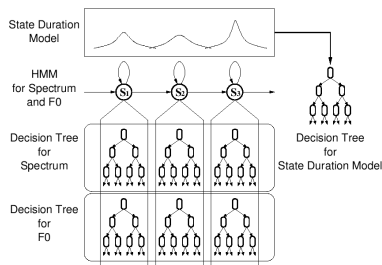


Store abstract, generalized units

- HMM states summarize spectrum words
- Model intonation (excitation) separately
- Label HMMs with contextual information

[Tokuda et al.(2002)Tokuda, Zen, and Black]

HMM based TTS:



Decision-tree based context clustering

- Many contextual factors (e.g., phone identity factors, stress-related factors, locational factors)
- Context-dependent HMMs
- Not enough speech and time \Rightarrow Cluster

[Tokuda et al.(2002)Tokuda, Zen, and Black]



HMM based TTS:

Table 1. Binary file size of HTS run-time engine.

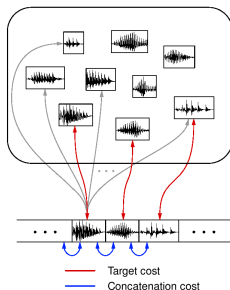
module		size
decision tree	spectrum	102 kbyte
	F_0	156 kbyte
	duration	116 kbyte
distribution	spectrum	457 kbyte
	F_0	81 kbyte
	duration	39 kbyte
converter		3 kbyte
synthesizer		34 kbyte
total		988 kbyte

Reduce footprint

- Small enough for PDAs
- Ten times Real-Time (on P4)
- HTS example using Alan's voice

[Tokuda et al.(2002)Tokuda, Zen, and Black]

HMM based TTS:

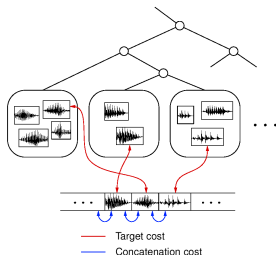


Classical Unit Selection scheme

- Concatenate using Target and Concatenation costs
- Use whole speech database
- Concatenate in real time

[Tokuda et al.(2002)Tokuda, Zen, and Black]

HMM based TTS:



HTS scheme

- Cluster all units on context in advance
- But use only statistics of cluster, not original templates
- Concatenation cost corresponds to dynamic feature parameter

[Tokuda et al.(2002)Tokuda, Zen, and Black]



HMM based TTS:

Table 2. Relation between unit selection and generation approaches.

Unit selection	HTS
Clustering (possible use of HMM)	Clustering (use of HMM)
Multi-template	Statistics
Single tree	Multiple tree (Spectrum, F0, duration)
Advantage: • High quality at waveform level Disadvantage: • Discontinuity • Hit or miss • Large run-time data • Fixed voice	Disadvantage: • Vocoded speech (buzzy) Advantage: • Smooth • Stable • Small run-time data • Various voices

Comparison

- Unit selection often very good, sometimes really bad
- HMM often bad (vocoder)
- HMM is much smaller and adaptable (retraining)

[Tokuda et al.(2002)Tokuda, Zen, and Black]



Further Reading I



James F. Allen, Donna K. Byron, Myroslava Dzikovska, George Ferguson, Lucian Galescu, and Amanda Stent.

Toward conversational human-computer interaction.

AI Magazine, Winter, 2001.

URL <http://www.cs.rochester.edu/research/cisd/pubs/2001/allen-et-al-aimag2001.pdf>.



P. Boersma.

Praat, a system for doing phonetics by computer.

Glott International, 5:341–345, 2001.

URL <http://www.Praat.org/>.



P. Boersma and D. Weenink.

Praat 4.2: doing phonetics by computer.

Computer program: <http://www.Praat.org/>, 2004.

URL <http://www.Praat.org/>.



Daniel Jurafsky and James H. Martin.

Speech and Language Processing.

Prentice-Hall, 2000.

ISBN 0-13-095069-6.

URL <http://www.cs.colorado.edu/~martin/slp.html>.

Updates at <http://www.cs.colorado.edu/>



Helmer Strik, Albert Russel, Henk van den Heuvel, Catia Cucchiari, and Lou Boves.

A spoken dialog system for the dutch public transport information service.

Int. Journal of Speech Technology, 2:121–131, 1997.

URL <http://lands.let.ru.nl/literature/strik.1996.4.ps>.

Link is to an older version.



Further Reading II



W. Wesseling and R. J. J. H. van Son.

Timing of experimentally elicited minimal responses as quantitative evidence for the use of intonation in projecting TRPs.
In *Proceedings of Interspeech2005*, Lisbon, 2005.



Wieneke Wesseling and R.J.J.H. Van Son.

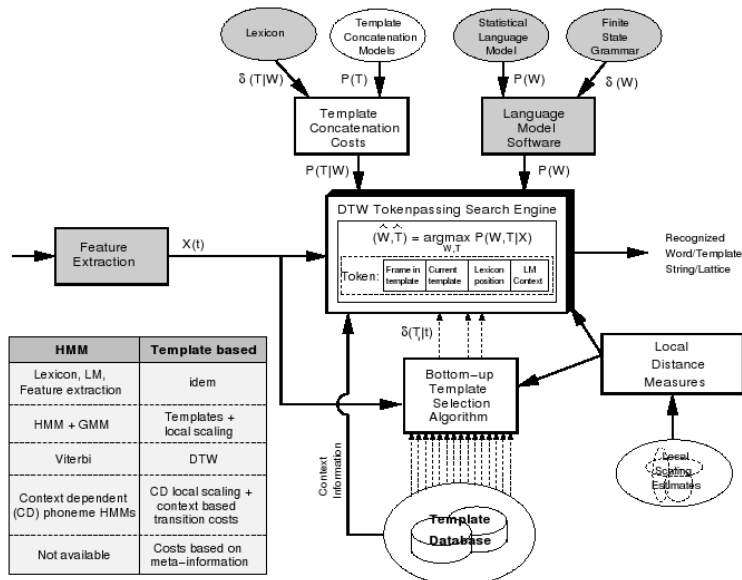
Early Preparation of Experimentally Elicited Minimal Responses.

In *Proceedings of SIGdial 2005*, September 2005.

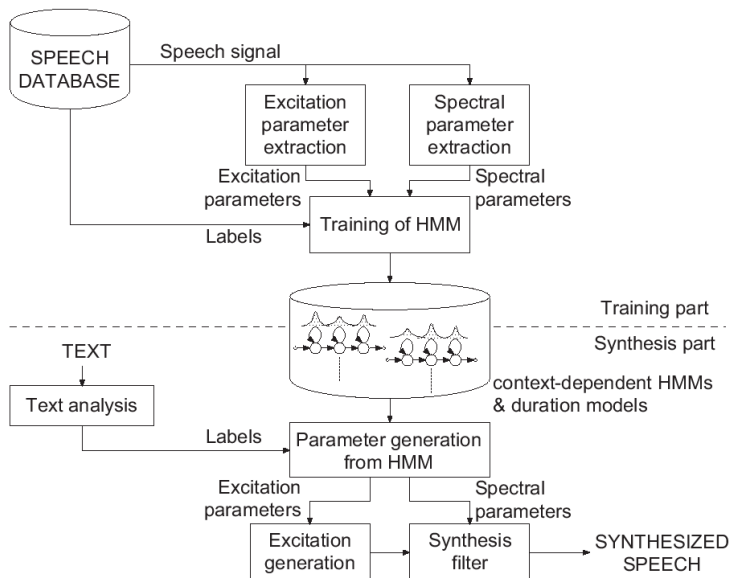
URL <http://www.fon.hum.uva.nl/rob/Publications/ArtikelSIGdial2005.pdf>.



Architectural overview template based ASR



Architectural overview HMM based synthesis



Copyright License

Copyright ©2007-2008 R.J.J.H. van Son, GNU General Public License [FSF(1991)]

This program 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 2 of the License, or (at your option) any later version. This program 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 this program; if not, write to the Free Software Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA 02110-1301, USA.

