

CS 224S / LINGUIST 236 Speech Recognition and Synthesis

Dan Jurafsky

Lecture 3: TTS Overview, History, and Letter-to-Sound

IP Notice: lots of info, text, and diagrams on these slides comes (thanks!) from Alan Black's excellent lecture notes and from Richard Sproat's great new slides.

1/11/05

CS 224S Winter 2005

1

Outline

- History of Speech Synthesis
- State of the Art, including Demos
- Overview of Speech Synthesis
- Overview of Festival
 - Where it lives, its components
 - Its scripting language: Scheme
- Letter-to-Sound Rules
 - (or "Grapheme-to-Phoneme Conversion")

1/11/05

CS 224S Winter 2005

2

Dave Barry on TTS

"And computers are getting smarter all the time; scientists tell us that soon they will be able to talk with us.

(By "they", I mean computers; I doubt scientists will ever be able to talk to us.)

1/11/05

CS 224S Winter 2005

3

History of TTS

- Pictures and some text from Hartmut Traunmüller's web site:
 - <http://www.ling.su.se/staff/hartmut/kempline.htm>
- Von Kempeln 1780 b. Bratislava 1734 d. Vienna 1804
- Leather resonator manipulated by the operator to try and copy vocal tract configuration during sonorants (vowels, glides, nasals)
- Bellows provided air stream, counterweight provided inhalation
- Vibrating reed produced periodic pressure wave

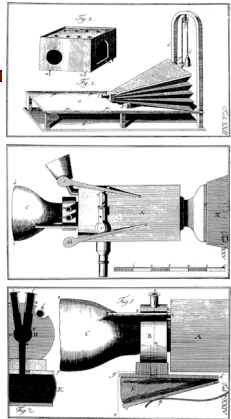
1/11/05

CS 224S Winter 2005

4

Von Kempelen:

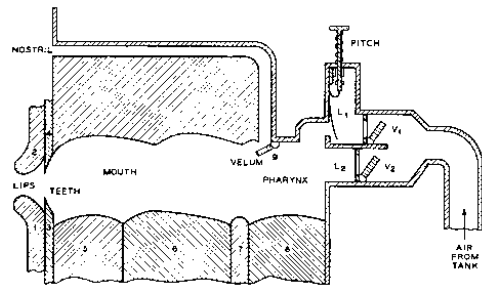
- Small whistles controlled consonants
- Rubber mouth and nose; nose had to be covered with two fingers for non-nasals
- Unvoiced sounds: mouth covered, auxiliary bellows driven by string provides puff of air



1/11/05
From Traunmüller's web site

CS 224S Winter 2005

Closer to a natural vocal tract: Riesz 1937



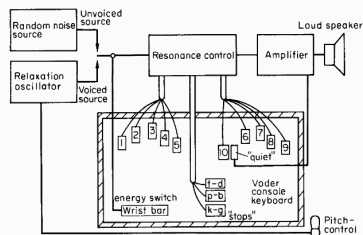
1/11/05

CS 224S Winter 2005

6

Homer Dudley 1939 VODER

- Synthesizing speech by electrical means
- 1939 World's Fair



1/11/05

7

Homer Dudley's VODER

- Manually controlled through complex keyboard
- Operator training was a problem



1/11/05

CS 224S Winter 2005

8

An aside on demos

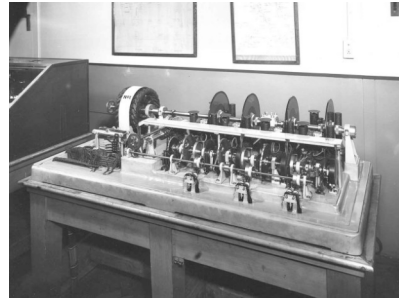
- That last slide
- Exhibited Rule 1 of playing a speech synthesis demo:
- Always have a human say what the words are **right before** you have the system say them

1/11/05

CS 224S Winter 2005

9

The 1936 UK Speaking Clock



1/11/05

CS 224S Winter 2005

10

From <http://www.ukonline.co.uk/freshwater/clocks/spkaclock.htm>

The UK Speaking Clock

- July 24, 1936
- Photographic storage on 4 glass disks
- 2 disks for minutes, 1 for hour, one for seconds.
- Other words in sentence distributed across 4 disks, so all 4 used at once.
- Voice of "Miss J. Cain"

1/11/05

CS 224S Winter 2005

11

A technician adjusts the amplifiers of the first speaking clock



1/11/05

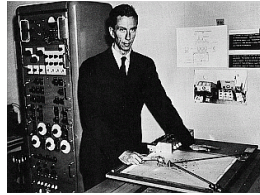
CS 224S Winter 2005

12

From <http://www.ukonline.co.uk/freshwater/clocks/spkaclock.htm>

Gunnar Fant's OVE synthesizer

- Of the Royal Institute of Technology, Stockholm
- **Formant Synthesizer for vowels**
- **F1 and F2 could be controlled**



1/11/05

CS 224S Winter 2005

13

From Traumnüller's web site

Cooper's Pattern Playback

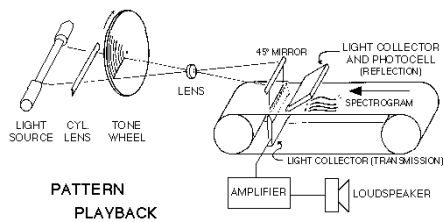
- Haskins Labs for investigating speech perception
- Works like an inverse of a spectrograph
- Light from a lamp goes through a rotating disk then through spectrogram into photovoltaic cells
- Thus amount of light that gets transmitted at each frequency band corresponds to amount of acoustic energy at that band

1/11/05

CS 224S Winter 2005

14

Cooper's Pattern Playback



1/11/05

CS 224S Winter 2005

15

Modern TTS systems

- 1960's first full TTS: Umeda et al (1968)
- 1970's
 - Joe Olive 1977 concatenation of linear-prediction diphones
 - Speak and Spell
- 1980's
 - 1979 MIT MITalk (Allen, Hunnicut, Klatt)
- 1990's-present
 - Diphone synthesis
 - Unit selection synthesis



1/11/05

CS 224S Winter 2005

16

Types of Modern Synthesis


- **Articulatory Synthesis:**
 - Model movements of articulators and acoustics of vocal tract
- **Formant Synthesis:**
 - Start with acoustics, create rules/filters to create each formant
- **Concatenative Synthesis:**
 - Use databases of stored speech to assemble new utterances.

1/11/05

CS 224S Winter 2005

Text from Richard Sproat slides 17

Formant Synthesis

- Were the most common commercial systems while (as Sproat says) computers were relatively underpowered.
- 1979 MIT MITalk (Allen, Hunnicut, Klatt)
- 1983 DECTalk system 
- The voice of Stephen Hawking

1/11/05

CS 224S Winter 2005

18

Concatenative Synthesis

- All current commercial systems.
- **Diphone Synthesis**
 - Units are diphones; middle of one phone to middle of next.
 - Why? Middle of phone is steady state.
 - Record 1 speaker saying each diphone
- **Unit Selection Synthesis**
 - Larger units
 - Record 10 hours or more, so have multiple copies of each unit
 - Use search to find best sequence of units

1/11/05

CS 224S Winter 2005

19

TTS Demos (all are Unit-Selection)

- **ATT:**
 - <http://www.naturalvoices.att.com/demos/>
- **Rhetorical (= Scansoft)**
 - <http://www.rhetorical.com/cgi-bin/demo.cgi>
- **Festival**
 - http://www-2.cs.cmu.edu/~awb/festival_demos/index.html
- **Cepstral**
 - <http://www.cepstral.com/cgi-bin/demos/general>
- **IBM**
 - <http://www-306.ibm.com/software/pervasive/tech/demos/tts.shtml>

1/11/05

CS 224S Winter 2005

20

Architecture

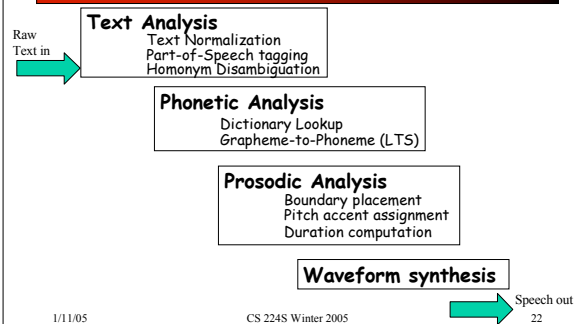
- The three types of TTS
 - Concatenative
 - Formant
 - Articulatory
- Only cover the segments+f0+duration to waveform part.
- A full system needs to go all the way from random text to sound.

1/11/05

CS 224S Winter 2005

21

TTS Architecture



1/11/05

CS 224S Winter 2005

22

Text Normalization

- Analysis of raw text into pronounceable words
- Sample problems:
 - He stole \$100 million from the bank
 - It's 13 St. Andrews St.
 - The home page is <http://www.stanford.edu>
 - yes, see you the following tues, that's 11/12/01
- Steps
 - Identify tokens in text
 - Chunk tokens into reasonably sized sections
 - Map tokens to words
 - Identify types for words

1/11/05

CS 224S Winter 2005

23

Grapheme to Phoneme

- How to pronounce a word? Look in dictionary! But:
 - Unknown words and names will be missing
 - Turkish, German, and other hard languages
 - uyarlaStiramadklarlmzdanmlSshzcaslna
 - "(behaving) as if you are among those whom we could not civilize"
 - uyyar +laS +tIr +ama +dlk +lar +lmz +dan +mlS +slnz +caslna
civilized +bec +caus +NegAble +ppart +pl +p1pl +abl +past +2pl +Asif
- So need Letter to Sound Rules
- Also homograph disambiguation (wind, live, read)

1/11/05

CS 224S Winter 2005

24

Prosody:

from words+phones to boundaries, accent, F0, duration

- **Prosodic phrasing**
 - Need to break utterances into phrases
 - Punctuation is useful, not sufficient
- **Accents:**
 - Predictions of accents: which syllables should be accented
 - Realization of F0 contour: given accents/tones, generate F0 contour
- **Duration:**
 - Predicting duration of each phone

1/11/05

CS 224S Winter 2005

25

Waveform synthesis:

from segments, f0, duration to waveform

- **Collecting diphones:**
 - need to record diphones in correct contexts
 - I sounds different in onset than coda, t is flapped sometimes, etc.
 - need quiet recording room, maybe EEG, etc.
 - then need to label them very very exactly
- **Unit selection: how to pick the right unit? Search**
- **Joining the units**
 - dumb (just stick'em together)
 - PSOLA (Pitch-Synchronous Overlap and Add)
 - MBROLA (Multi-band overlap and add)

1/11/05

CS 224S Winter 2005

26

Festival

- **Open source speech synthesis system**
- **Designed for development and runtime use**
 - Use in many commercial and academic systems
 - Distributed with RedHat 9.x
 - Hundreds of thousands of users
 - **Multilingual**
 - No built-in language
 - Designed to allow addition of new languages
 - **Additional tools for rapid voice development**
 - Statistical learning tools
 - Scripts for building models

1/11/05

CS 224S Winter 2005

Text from Richard Sproat 27

Festival as software

- <http://festvox.org/festival/>
- **General system for multi-lingual TTS**
- **C/C++ code with Scheme scripting language**
- **General replaceable modules:**
 - Lexicons, LTS, duration, intonation, phrasing, POS tagging, tokenizing, diphone/unit selection, signal processing
- **General tools**
 - Intonation analysis (f0, Tilt), signal processing, CART building, N-gram, SCFG, WFST

1/11/05

CS 224S Winter 2005

Text from Richard Sproat 28

Festival as software

- <http://festvox.org/festival/>
- No fixed theories
- New languages without new C++ code
- Multiplatform (Unix/Windows)
- Full sources in distribution
- Free software

1/11/05

CS 224S Winter 2005

29

Text from Richard Sproat

CMU FestVox project

- Festival is an engine, how do you make voices?
- Festvox: building synthetic voices:
 - Tools, scripts, documentation
 - Discussion and examples for building voices
 - Example voice databases
 - Step by step walkthroughs of processes
- Support for English and other languages
- Support for different waveform synthesis methods
 - Diphone
 - Unit selection
 - Limited domain

1/11/05

CS 224S Winter 2005

30

Text from Richard Sproat

Synthesis tools

- I want my computer to talk
 - Festival Speech Synthesis
- I want my computer to talk in my voice
 - FestVox Project
- I want it to be fast and efficient
 - Flite

1/11/05

CS 224S Winter 2005

31

Text from Richard Sproat

Using Festival

- How to get Festival to talk
- Scheme (Festival's scripting language)
- Basic Festival commands

1/11/05

CS 224S Winter 2005

32

Text from Richard Sproat

Getting it to talk

- **Say a file**
 - festival --tts file.txt
- **From Emacs**
 - say region, say buffer
- **Command line interpreter**
 - festival> (SayText "hello")

1/11/05

CS 224S Winter 2005

Text from Richard Sproat

Scheme: the scripting lg

- **Advantages of a scripting lg**
 - Convenient, easy to add functionality
- **Why Scheme?**
 - Holdover from the LISP days of AI.
 - Many people like it.
 - It's very simple

1/11/05

CS 224S Winter 2005

Text adapted from Richard Sproat

Quick Intro to Scheme

- **Scheme is a dialect of LISP**
- **expressions are**
 - atoms or
 - lists
 - a bed "hello world" 12.3
 - (a b c)
 - (a (1 2) seven)
- **Interpreter evaluates expressions**
 - Atoms evaluate as variables
 - Lists evaluate as functional calls
 - bxx
 - 3.14
 - (+ 2 3)

1/11/05

CS 224S Winter 2005

Text from Richard Sproat

Quick Intro to Scheme

- **Setting variables**
 - (set! a 3.14)
- **Defining functions**
 - (define (timestwo n) (* 2 n))

 - (timestwo a)
 - 6.28

1/11/05

CS 224S Winter 2005

Text from Richard Sproat

Lists in Scheme

```
• festival> (set! alist '(apples pears bananas))
• (apples pears bananas)
• festival> (car alist)
• apples
• festival> (cdr alist)
• (pears bananas)
• festival> (set! blist (cons 'oranges alist))
• (oranges apples pears bananas)
• festival> append alist blist
• #<SUBR(6) append>
• (apples pears bananas)
• (oranges apples pears bananas)
• festival> (append alist blist)
• (apples pears bananas oranges apples pears bananas)
• festival> (length alist)
• 3
• festival> (length (append alist blist))
• 7
```

1/11/05

CS 224S Winter 2005

Text from Richard Sproat 37

Scheme: speech

- **Make an utterance of type text**
festival> (set! uttl1 (Utterance Text "hello"))
#<Utterance 0xf6855718>
- **Synthesize an utterance**
festival> (utt.synth uttl1)
#<Utterance 0xf6855718>
- **Play waveform**
festival> (utt.play uttl1)
#<Utterance 0xf6855718>
- **Do all together**
festival> (SayText "This is an example")
#<Utterance 0xf6961618>

1/11/05

CS 224S Winter 2005

Text from Richard Sproat 38

Scheme: speech

- **In a file**
(define (SpeechPlus a b)
 (SayText
 (format nil
 "%d plus %d equals %d"
 a b (+ a b))))
- **Loading files**
festival> (load "file.scm")
t
- **Do all together**
festival> (SpeechPlus 2 4)
#<Utterance 0xf6961618>

1/11/05

CS 224S Winter 2005

Text from Richard Sproat 39

Scheme: speech

```
(define (sp_time hour minute)
  (cond
    (( < hour 12)
     (SayText
      (format nil
        "It is %d %d in the morning"
        hour minute )))
    (( < hour 18)
     (SayText
      (format nil
        "It is %d %d in the afternoon"
        (- hour 12) minute )))
    (t
     (SayText
      (format nil
        "It is %d %d in the evening"
        (- hour 12) minute )))))
```

1/11/05

CS 224S Winter 2005

Text from Richard Sproat 40

Getting help

- Online manual
 - <http://festvox.org/docs/manual-1.4.3>
- Alt-h (or esc-h) on current symbol short help
- Alt-s (or esc-s) to speak help
- Alt-m goto man page
- Use TAB key for completion

1/11/05

CS 224S Winter 2005

41

Word pronunciations

- Now that you've tried doing this by hand!

1/11/05

CS 224S Winter 2005

42

Lexicons and Lexical Entries

- You can explicitly give pronunciations for words
 - Each lg/dialect has its own lexicon
 - You can lookup words with
 - (lex.lookup WORD)
 - You can add entries to the current lexicon
 - (lex.add.entry NEWENTRY)
 - Entry: (WORD POS (SYL0 SYL1...))
 - Syllable: ((PHONE0 PHONE1 ...) STRESS)
 - Example:
'("cepstra" n ((k eh p) l) ((s t r aa) 0)))

1/11/05

CS 224S Winter 2005

43

Converting from words to phones

- Two methods:
 - Dictionary-based
 - Rule-based (Letter-to-sound=LTS)
- Early systems, all LTS
- MITalk was radical in having huge 10K word dictionary
- Now systems use a combination
- CMU dictionary: 127K words
 - <http://www.speech.cs.cmu.edu/cgi-bin/cmudict>

1/11/05

CS 224S Winter 2005

44

Dictionaries aren't always sufficient

- **Unknown words**
 - Seem to be linear with number of words in unseen text
 - Mostly person, company, product names
 - But also foreign words, etc.
- **So commercial systems have 3-part system:**
 - Big dictionary
 - Special code for handling names
 - Machine learned LTS system for other unknown words

1/11/05

CS 224S Winter 2005

45

Letter-to-Sound Rules

- **Festival LTS rules:**
- (LEFTCONTEXT [ITEMS] RIGHTCONTEXT = NEWITEMS)
- **Example:**
 - (# [c h] C = k)
 - (# [c h] = ch)
- **# denotes beginning of word**
- **C means all consonants**
- **Rules apply in order**
 - "christmas" pronounced with [k]
 - But word with ch followed by non-consonant pronounced [ch]
- E.g., "choice"

1/11/05

CS 224S Winter 2005

46

What about stress: practice

- **Generally**
- **Pronounced**
- **Exception**
- **Dictionary**
- **Significant**
- **Prefix**
- **Exhale**
- **Exhalation**
- **Sally**

1/11/05

CS 224S Winter 2005

47

Stress rules in LTS

- **English famously evil: one from Allen et al 1987**
- $V \rightarrow [1\text{-stress}] / X_C^* \{V_{\text{short}} C C^* | V\} \{[V_{\text{short}} C^* | V]\}$
- **Where X must contain all prefixes:**
- **Assign 1-stress to the vowel in a syllable preceding a weak syllable followed by a morpheme-final syllable containing a short vowel and 0 or more consonants (e.g. difficult)**
- **Assign 1-stress to the vowel in a syllable preceding a weak syllable followed by a morpheme-final vowel (e.g. oregano)**

1/11/05

CS 224S Winter 2005

48

Modern method: Learning LTS rules automatically

- Induce LTS from a dictionary of the language
- Black et al. 1998
- Applied to English, German, French
- Two steps: alignment and (CART-based) rule-induction

1/11/05

CS 224S Winter 2005

49

Alignment

- Letters: c h e c k e d
- Phones: ch _ eh _ k _ t
- Black et al Method 1:
 - First scatter epsilons in all possible ways to cause letters and phones to align
 - Then collect stats for $P(\text{letter}|\text{phone})$ and select best to generate new stats
 - This iterated a number of times until settles (5-6)
 - This is EM (expectation maximization) alg

1/11/05

CS 224S Winter 2005

50

Alignment

- Black et al method 2
- Hand specify which letters can be rendered as which phones
 - C goes to k/ch/s/sh
 - W goes to w/v/f, etc
- Once mapping table is created, find all valid alignments, find $p(\text{letter}|\text{phone})$, score all alignments, take best

1/11/05

CS 224S Winter 2005

51

Alignment

- Some alignments will turn out to be really bad.
- These are just the cases where pronunciation doesn't match letters:
 - Dept d ih p aa r t m ah n t
 - CMU s iy eh m y uw
 - Lieutenant l eh f t eh n ax n t (British)
- Also foreign words
- These can just be removed from alignment training

1/11/05

CS 224S Winter 2005

52

Building CART trees

- Build a CART tree for each letter in alphabet (26 plus accented) using context of +-3 letters
- # # # c h e c -> ch
- c h e c k e d -> _
- This produces 92-96% correct LETTER accuracy (58-75 word acc) for English

1/11/05

CS 224S Winter 2005

53

Improvements

- Take names out of the training data
- And acronyms
- Detect both of these separately
- And build special-purpose tools to do LTS for names and acronyms
- Names:
 - Can do morphology (Walters -> Walter, Lucasville)
 - Can write stress-shifting rules (Jordan -> Jordanian)
 - Rhyme analogy: Plotsky by analogy with Trotsky (replace tr with pl)
 - Liberman and Church: for 250K most common names, got 212K (85%) from these modified-dictionary methods, used LTS for rest.

1/11/05

CS 224S Winter 2005

54