

# Text-to-Speech Synthesis

***Prof. Dr. Tanja Schultz***  
***(Folien und Synthesebeispiele von Alan W Black)***

**Universität Karlsruhe, Fakultät für Informatik**

# Speech Synthesis Process

Definition: **Speech synthesis** is the artificial production of human speech

Definition: A **Text-to-speech (TTS)** system converts written text (language) into speech

## Typically 3 Steps:

- **Text Analysis**
  - From strings of characters to words
- Linguistic Analysis
  - From words to pronunciations and prosody
- Waveform Synthesis
  - from pronunciations to waveforms

# Text Analysis

- Character Encodings
- Word Segmentation
- Numbers, Symbols, Non-Standard Words
  - Anything not directly in the lexicon
  - OOV or novel forms
  - Abbreviations, Letter sequences, Acronyms
- May require special rules
  - Train rules for homographs
  - Numbers, roman numerals

Text processing errors are the most common complaints in TTS

# Text Analysis

- This is a pen.
- My cat who **lives** dangerously has nine **lives**.
- He stole \$100 from the bank.
- He stole **1996** cattle on 25 Nov **1996**.
- He stole \$100 million from the bank.
- It's 13 **St.** Andrew **St.** near the bank.
- Its a PIII 650Mhz, 128MB RAM, 13.6Gb SCSI, 24x cdrom and 19" monitor.
- My home page is <http://www.geocities.com/awb/>.

# Text Analysis

*“bunch of hacky rules”*

But here, based on NSW Project from JHU Workshop '99  
(NSW = Normalization of Non-Standard Words)

- Splitter:
  - domain independent tokenizer
- token type classifier:
  - number (ordinal/cardinal/year) – 1996 cattle vs year
  - homograph disambiguator – lives, St.
  - abbrev/letter sequence identifier – Mb, RAM, SCSI
- token type expander:
  - mostly deterministic
  - but abbreviation expander interesting
- language model:

(optionally) choose best expansion

# NSW models for specific domains

- Models for specific domains
- Standard text analyzers fail
- Can build models from labeled data

57 ST E/1st & 2nd Ave Huge  
drmn 1 BR 750+ sf, lots of sun  
clsts. Sundeck & Indry facils. Askg  
\$187K, maint \$868, utils  
incl. Call Bkr Peter 914-428-9054.

- Standard models
- Domain specific models



# Synthesis Process

- Text Analysis:
  - from strings of characters to words
- **Linguistic Analysis**
  - **from words to pronunciations and prosody**
- Waveform synthesis
  - from pronunciations to waveforms

# Lexicons and letter to sound rules

- Need big list of words:
  - often hardest requirement
- Sharing between dialects:
  - CSTR English dialect lexicon  
(Center for Speech Technology Research, Edinburgh)
- But the lexicon is *never* complete:
  - need out of vocabulary pronouncer



# Bootstrapping Lexicons

- Find 250 most frequent words
  - Build lexical entries from them
  - Ensure letter coverage in base set
  - Build letter-to-sound (LTS) rules from this base set
- Select articles of text
- Synthesize each unknown word
  - Add correct words to base list
  - Correct incorrect words and add to base list
  - Rebuild LTS rules with larger list
  - Repeat process

# Letter to sound rules

- Writing rules by hand is difficult
- We provide automatic process:
  - built from lexicon
  - provides phone string plus stress
  - tested on multiple languages

Lexicon	Correct	
	Letters	Words
OALD	95.80%	74.56%
CMUDICT	91.99%	57.80%
BRULEX	99.00%	93.03%
DE-CELEX	98.79%	89.38%
Thai	95.60%	68.76%

# Letter to sound rules: method

- Find alignments:

c	h	e	c	k	e	d
ch	-	eh	-	k	-	t

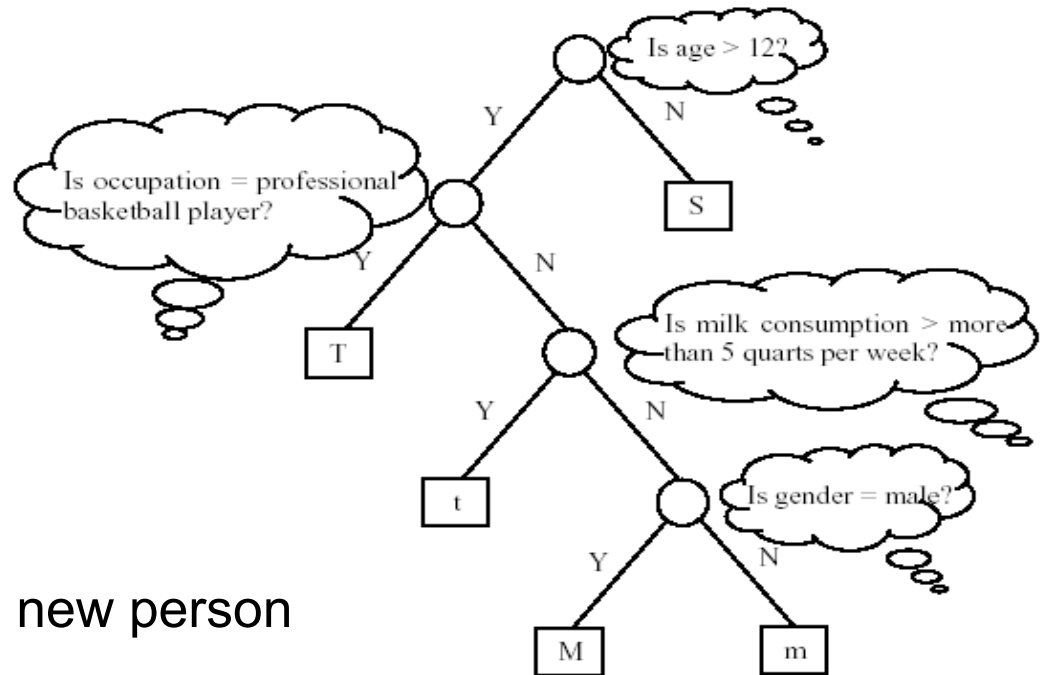
- Using epsilon scattering and EM algorithm
  - or hand specify possible letter-phone combinations
- Build CART models:
  - predict phone (or epsilon)
  - letter context (plus POS ...)

# Classification Trees

Simple binary decision tree  
for height classification:

T=tall,  
t=medium-tall,  
M=medium,  
m=medium-short,  
S=short

Goal: Predict the height of a new person



- Using decision tree to predict someones height class by traversing the tree and answering the yes/no questions
- Choice and order of questions is designed knowledge-based
- Classification and Regression Trees (CART) provide an automatic, data-driven framework to construct the decision process

# An example tree

For letter V: (Example: revving vs Romatov)

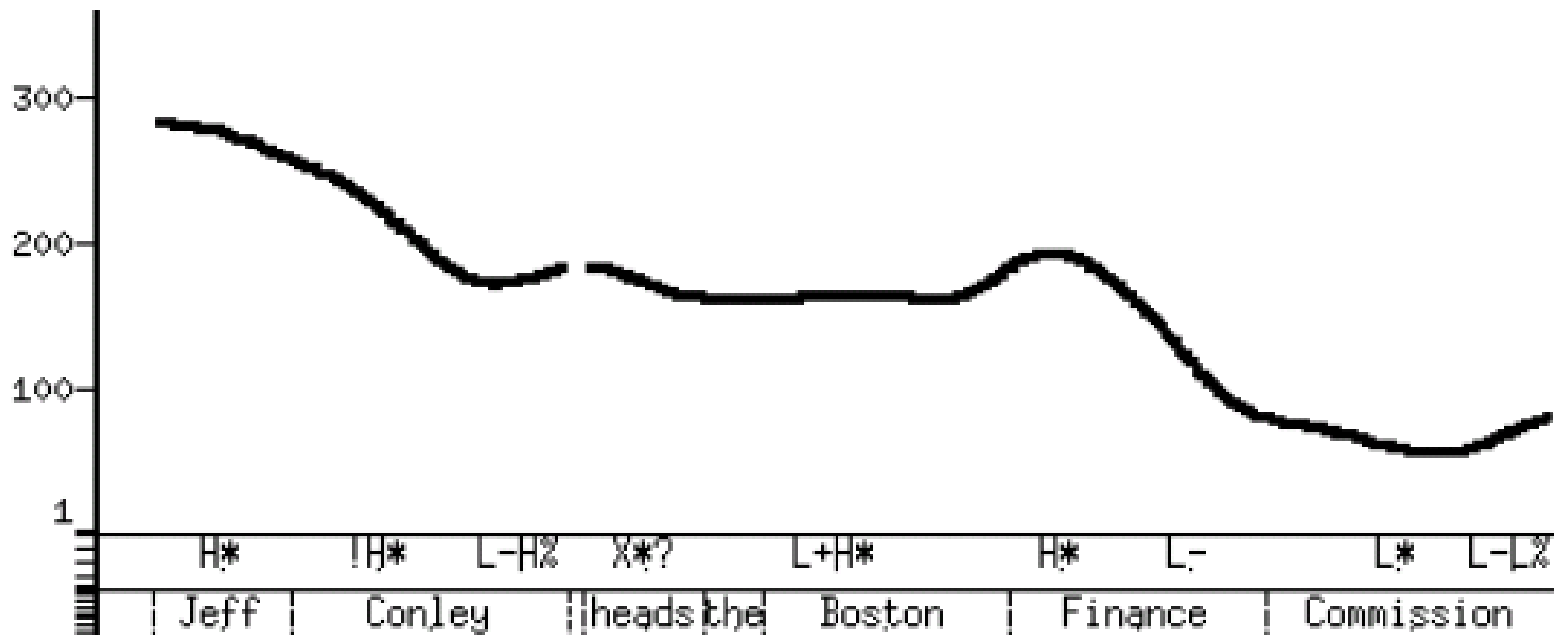
```
if (n.name is v) /* case vv */  
    return _  
    if (n.name is #) /* v at end of word */  
        if (p.p.name is t) /* case t_v like in Romatov */  
            return f  
            return v  
        if (n.name is s)  
            if (p.p.p.name is n)  
                return f  
                return v  
            return v
```

# Prosody

- Phrasing
  - chunking in speech
- Intonation
  - Accents and F0 generation
- Duration:
  - length of segments
- Power:
  - Loudness

Each has underlying “default”, and marked forms

# Intonation marking








# Intonation Marking

- Large pitch range (female)
- Authoritive since goes down at the end
  - News reader
- Emphasis for Finance H\*
- Final has a raise – more information to come
- Female American newsreader from WBUR
- (Boston University Radio)



# Intonation Examples






- Fixed durations, flat F0. 
- Decline F0 
- “hat” accents on stressed syllables 
- accents and end tones 
- statistically trained 

# Synthesis Processes

- Text Analysis:
  - from strings of characters to words
- Linguistic Analysis
  - from words to pronunciations and prosody
- **Waveform synthesis**
  - **from pronunciations to waveforms**

# Waveform Synthesis

Concatenative synthesis:

- Random word/phrase concatenation 
- Phone concatenation 
- Diphone concatenation 
- Sub-word unit selection 
- Cluster based unit selection 

# Waveform Synthesis “diphone”

- All phone-phone transitions in language:
  - **one** occurrence of each
  - assume two-phone context is sufficient
  - may also include common consonant clusters
- Mid-phone is more stable so easier to join
- Inventory size: phones<sup>2</sup>
- A “safe” way to build a synthesizer

# Diphone example: (US English)

- Diphone schema 1348 words (1618 diphones):
  - includes consonant clusters and open vowels
- Prompts are:
  - kal\_0001 ("b-aa" "aa-b") (t aa b aa b aa)
  - kal\_0002 ("p-aa" "aa-p") (t aa p aa p aa)
- Automatic aligning alone:
  - around 15 words *wrong*
  - better than human labeler (first pass)
  - takes 30 minutes (vs 2 weeks)
- KAL voice created in 2 days



# Unit selection synthesis

- Large representative database
- **Multiple** examples of each unit type
- Select “appropriate” units:
  - phonetically and prosodically
- Various unit selection algorithms:
  - target, concatenation cost
  - Cluster
  - phonological structure matching
- famed for being excellent
- famed for being incomprehensible

# Cluster unit selection

- Cluster unit selection (Black&Taylor, Eurosp. 97)
- Decision tree partitioning on acoustic distance:
  - phone index (but diphone selection)
  - pitch synchronous melcep plus power and F0
  - indexed by phone and prosodic context
  - cf. Donovan 95
- Viterbi selection through candidates
  - cf. Hunt and Black 96, Iwahashi and Sagisaka 95
- Does it work?
  - good in parts

# Making synthesis better

- Basic reading clean text in neutral form works
  - few applications require that though
- We need synthesis to be:
- more flexible:
  - not just **text** to speech
- more natural
  - so its doesn't sound like a synthesizer
- more efficient
  - easy to build new voices
  - synthesizes quickly on small machines



# SABLE – Bell Labs <http://www.bell-labs.com/project/tts/sable.html>

<?xml version="1.0"?>

<!DOCTYPE SABLE PUBLIC "-//SABLE//DTD SABLE speech mark up//EN"  
"Sable.v0\_2.dtd"

[]> <SABLE> <SPEAKER NAME="male1">

The boy saw the girl in the park <BREAK/> with the telescope.

The boy saw the girl <BREAK/> in the park with the telescope.

Some English first and then some Spanish.

<LANGUAGE ID="SPANISH">Hola amigos.</LANGUAGE>

<LANGUAGE ID="NEPALI">Namaste</LANGUAGE>

Good morning <BREAK /> My name is Stuart, which is spelled

<RATE SPEED="-40%">

<SAYAS MODE="literal">stuart</SAYAS> </RATE>

though some people pronounce it

<PRON SUB="stoo art">stuart</PRON>. My telephone number is

<SAYAS MODE="literal">2787</SAYAS>.

I used to work in <PRON SUB="Buckloo">Buccleuch</PRON> Place, but no one can  
pronounce that.

By the way, my telephone number is actually

<AUDIO SRC="http://att.com/sounds/touchtone.2.au"/>

<AUDIO SRC="http://att.com/sounds/touchtone.7.au"/>

<AUDIO SRC="http://att.com/sounds/touchtone.8.au"/>

<AUDIO SRC="http://att.com/sounds/touchtone.7.au"/>



# SABLE: for marking emphasis

What will the weather be like today in Boston?  
It will be <emph>rainy</emph> today in Boston.



When will it rain in Boston?  
It will be rainy <emph>today</emph> in Boston.



Where will it rain today?  
It will be rainy today in <emph>Boston</emph>.



# Making it efficient

- Adding new voices:
  - Need new voices and languages
  - tools to record label segment etc
- Synthesis must be fast
  - Dialog systems must respond in time
  - Recognition, dialog plus synthesis under 1 sec.
  - ... and run on a handheld device or mobile phone

# Summary – Essence of Synthesis

- Synthesis can be broken down into three parts:
- 1. Text analysis:
  - expanding homographs, symbols, numbers etc
- 2. Linguistic analysis:
  - pronunciation, letter to sound rules
  - prosody: phrasing, intonation and duration
- 3. Waveform synthesis:
  - diphone voices:
  - unit selection voices
  - limited domain
- Not just **text**-to-speech:
  - concept to speech
  - markup

# Build Your Own Synthetic Voices

The standard voice requires ...

- A phone set
- Text analysis
- Pronunciations
- Prosodic modeling
- Waveform Generation



Plus something else that is difficult

- ... and often language dependent
  - E.g. word segmentation in Chinese
  - Vowels in Arabic
  - Number declensions in Slavic languages

# Multilingual Speech Synthesis

- Common technologies
  - (Diphone: too hard to record and label, coverage)
  - Unit selection is the standard method: *select appropriate sub-word units from large database of natural speech*
  - CONS: too much to record and label
  - Requires 200M per voice (single model across languages)
- New technology: Parametric Synthesis “clustergen (CG)”
  - HMM-generation based synthesis
  - Cluster units to form models, Generate from models
  - PRO: can work with little speech (10 minutes)
  - CONS: speech sounds buzzy, lacks natural prosody
  - Requires 2M per voice (single model across languages)

# Unit Selection vs Parametric Synthesis

- Unit Selection: 
  - large carefully labeled database (often 5000+ utts)
  - hard to speak 5000 good utterances > professionals
  - quality good when good examples available but rapid degradation when out-of-domain
  - little or no prosodic modifications
  - natural delivery; multiple speakers desired to model variability!!!
- Parametric Synthesis: 
  - smaller less carefully labeled database
  - quality consistent
  - resynthesis requires vocoder, (buzzy)
  - can (must) control prosody
  - model size much smaller than Unit DB

**Vocoder** (voc (*voice*) and *encoder*) beruht auf der Zerlegung eines Eingangssignals in seine Frequenzbestandteile, der Übertragung dieser Bestandteile als Parametersatz, sowie der darauf folgenden Resynthese des Signals am Ziel auf der Basis der Parameter aus einem Rauschsignal.

# HMM-Generation Synthesis

- NiTech's HTS (Tokuda et al.)
  - Built as modification of HTK
  - FestVox build process exists
  - Hard coded feature values
- HMM-Generation Synthesis
  - Sub-phonetic features are modelled not as set of instances of units but parametric models
  - View these clusters as averages of instances
- High quality understandable speech (Bennett, 2005)
- Language Independence (Tokuda, 2002)
- Multilingual databases (GlobalPhone) within HMM-generation synthesis (Latorre & Furui, 2005)



# ClusterGen

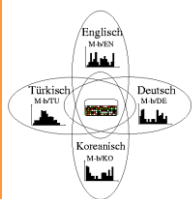
- New synthesis technique added to FestVox
  - Clustering technique for HMM-state sized segments
- Training data is HMM-based labeled speech
  - Labeling system included in FestVox
  - Janus RTk labels are used created by forced alignment
- CLUSTERGEN
  - Reversible (analysis/synthesis) parameterization of speech
  - MCEP analysis and MLSA filter for resynthesis (as in HTS)
  - 24-dim MCEP feature vectors
  - Clustered using wagon CART tree builder
  - Features for tree building are the articulatory features derived from GP IPA-based global phone inventory
  - Cluster optimization:
    - minimize sum of SD of each MCEP feature
    - weight by the number of samples in the cluster

# Universal Sound Inventory & Data

Speech Production is independent from Language  $\Rightarrow$  IPA

1) IPA-based Universal Sound Inventory

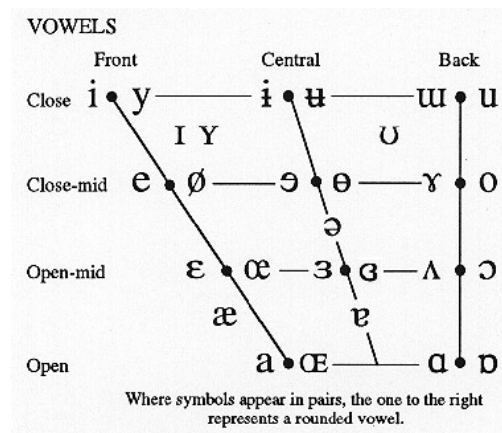
2) Each sound class is trained by data sharing



| Reduction from 485 to 162 sound classes

| *m, n, s, l* appear in all 12 languages

| *p, b, t, d, k, g, f* and *i, u, e, a, o* in almost all



## GlobalPhone

- Use 10 languages  
Ch-Mandarin, Croatian, German, Japanese, Portuguese, Russian, Spanish, Swedish, Turkish + English (WSJ0)
- Use ASR global sound inventory
- Use IPA acoustic features



# Clustering by CART

- Update to Wagon (Edinburgh Speech Tools)
- Clustering
  - F0 and MCEP, tested jointly and separately
  - Features for clustering (51): IPA articulatory features + other phonetic, syllable, phrasal context
- Training Output - Three models:
  - Spectral (MCEP) CART tree
  - F0 CART tree
  - Duration CART tree
- F0 model:
  - Smooth extracted F0 through all speech (i.e. unvoiced regions get F0 values)
  - Chose voicing at runtime phonetically

# FestVox CLUSTERGEN Synthesizer

- Prompt transcriptions, Waveform files (well recorded)
- Labeling
  - Used CI models and forced alignment (JRTk – monolingual ASR)
- Parameter extraction:
  - (HTS's) MCEP/MLSA filter for resynthesis
  - F0 extraction
- Clustering
  - Wagon vector clustering for each HMM-state name
- ClusterGen Synthesis:
  - Generate phoneme strings (as before)
  - For each phone:
    - Find HMM-state names: ah\_1, ah\_2, ah\_3
  - Predict duration of each
    - Create empty MCEP vector to fill duration
    - Predict MCEP and F0 values from corresponding cluster trees
    - Use MLSA filter to regenerate speech

# Measuring Quality

Mean Mel Cepstral Distortion (MCD) over test set  
(smaller is better)

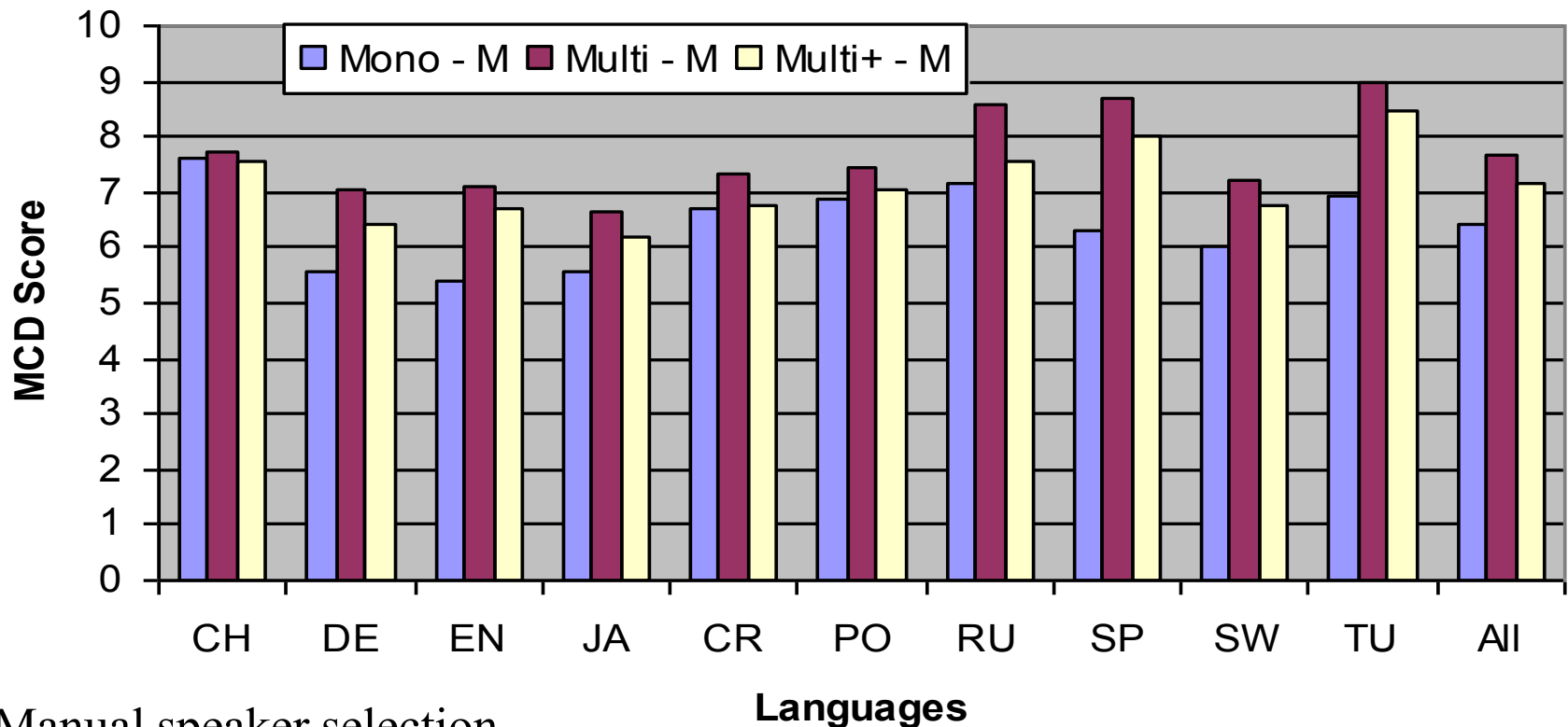
$$10 / \ln 10 \sqrt{2 \sum_{d=1}^{24} \left( mc_d^{(t)} - mc_d^{(e)} \right)^2}$$

Measured on a Cross evaluation set

MCD: Voice Conversion ranges 4.5-6.0

MCD: ClusterGen scores 5.0-8.0

# Mono vs Multilingual Models



Manual speaker selection

⇒ For all languages monolingual TTS performs best

⇒ Multilingual Models perform well ...

... only if knowledge about language is preserved (Multi+)

(only small amount of sharing actually happens)

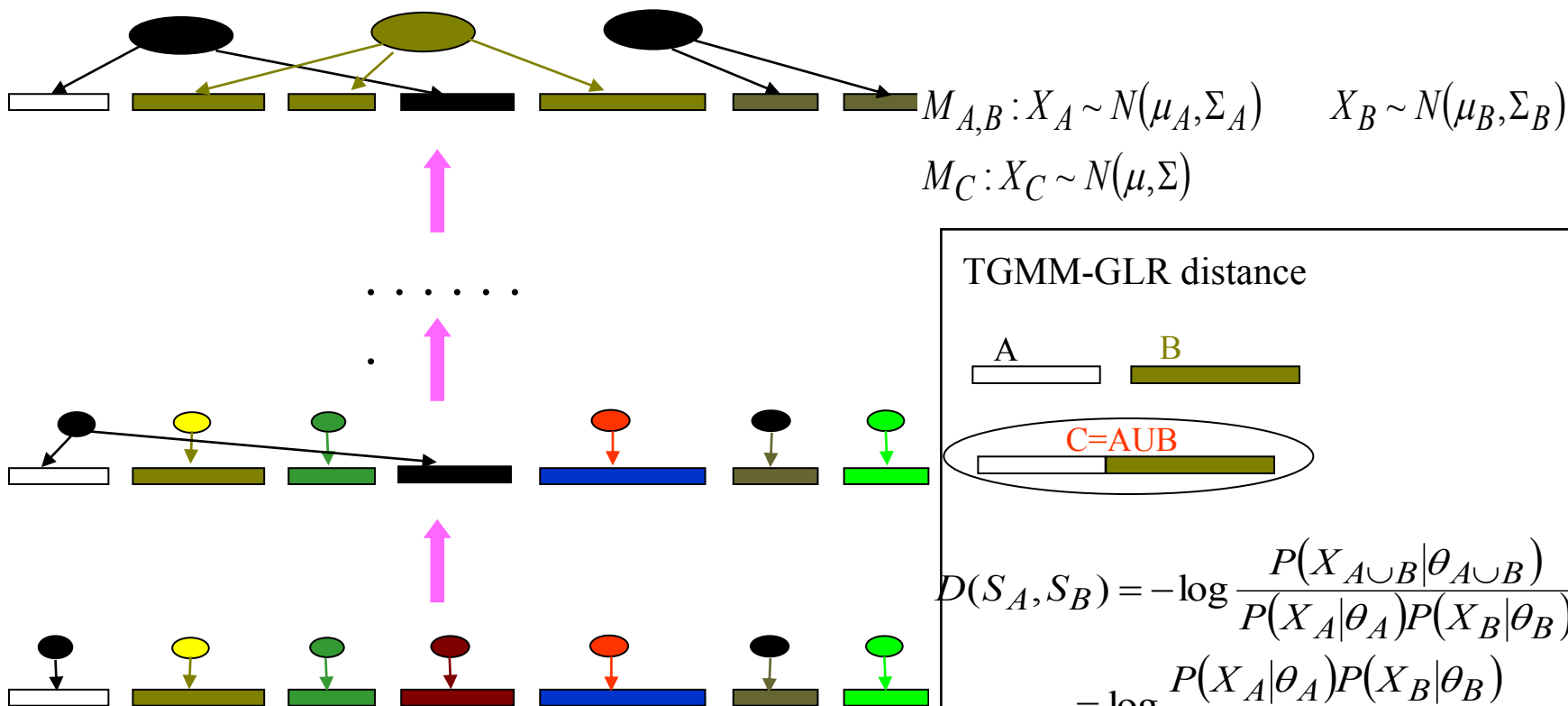
# Speaker Clustering

## Hierarchical Bottom-up Clustering

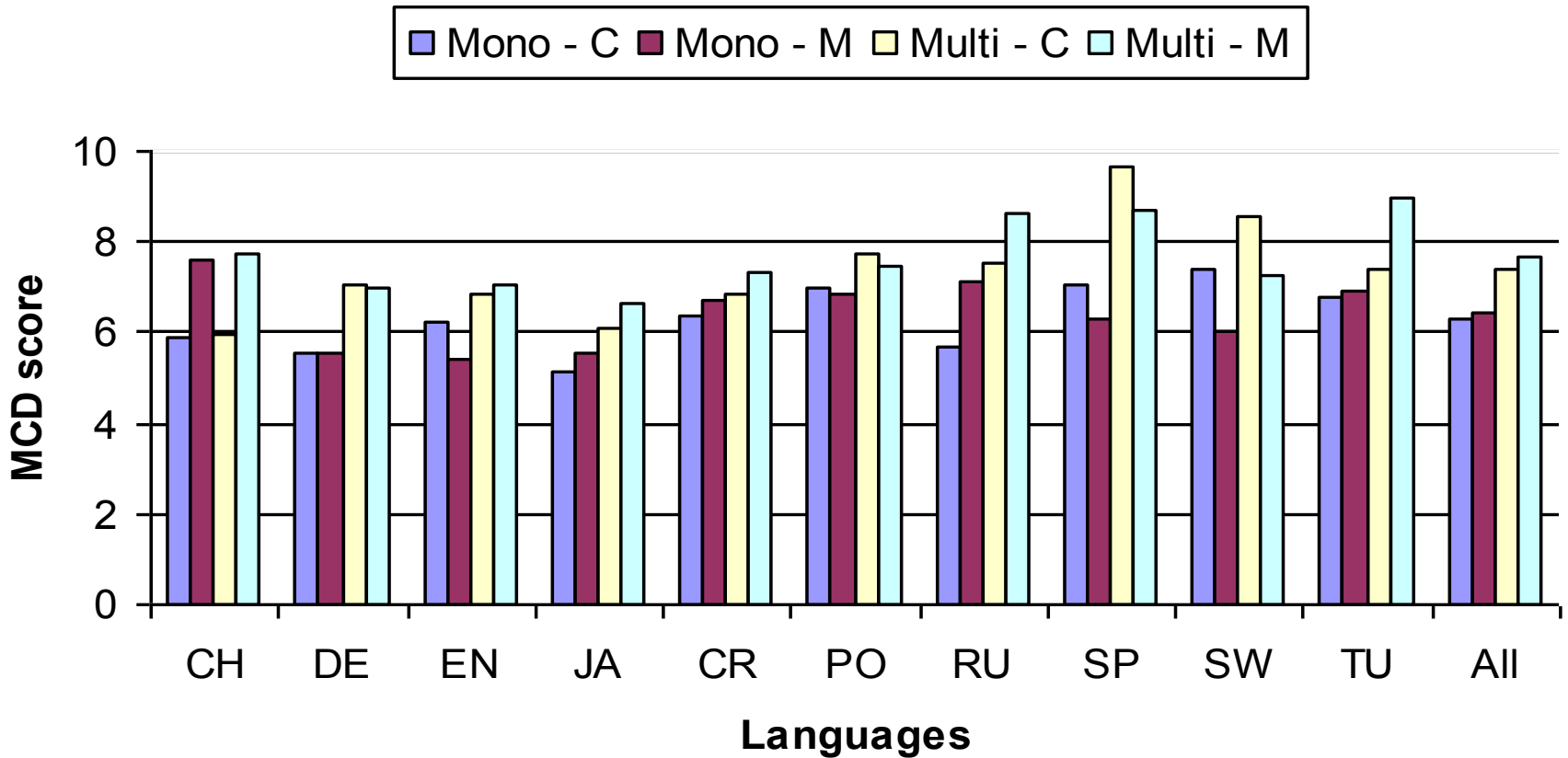
BIC stopping criterion

$$\Delta BIC = BIC(M_C) - BIC(M_{A,B})$$

$$BIC(M) = \log L(X | \mu, \Sigma) - \frac{\lambda}{2} V(M) \log N$$



# Manual vs Clustered Speaker Selection



- ⇒ Selecting similar speakers helps for both Mono and Multi
- ⇒ Multi benefits more as expected: similarity more important
- ⇒ Large variation across languages (label quality? Data size?)



# Conclusion & Future Work on ML-TTS

- ClusterGen allows for much smaller voices
  - Multilingual voice for unit selection: 200Mb
  - Multilingual voice for ClusterGen: 2Mb
- Preserving language information (Multi+) helps
- Selecting similar speakers helps
- All voices are understandable (no formal tests!)

## Future Work:

- ClusterGen is \*very\* young
  - No treatment of dynamics yet
  - Multilingual, Multispeaker DBS
  - Grapheme Based models
  - Voice/Style conversion: model interpolation
  - Signal reconstruction: proper residual modeling

# END

# Synthesizer Voice Quality of New Languages Calibrated with Mean Mel Cepstral Distortion

**John Kominek**  
**Tanja Schultz**  
**Alan W Black**

Language Technologies Institute  
Carnegie Mellon University  
Pittsburgh PA USA

**SLTU Hanoi Vietnam – May 6 2008**

# SPICE project goals

- Rapid development of ASR + TTS for new languages
- SPICE – Speech Processing Interactive Creation and Evaluation
  - a web-based tool suite that streamlines creation of core language technology components
  - Janus multi-lingual ASR
  - Festival multi-lingual TTS
  - text and speech collection
  - phoneme and character set definitions
  - LTS rule builder

## Build Your System

● Text and prompt selection [\(help\)](#)

● Audio collection [\(help\)](#)

● Phoneme selection [\(help\)](#)

● Grapheme-to-phoneme rules [\(help\)](#)

● Lexicon pronunciation creation [\(help\)](#)

● Build acoustic model [\(help\)](#)

● Build language model [\(help\)](#)

● Test ASR system

● Create speech synthesis voice

User: **john** Language: **eng** Project: **recipe\_1000** [\[Logout\]](#)

## Building synthesis voice

### Tasks

Voice Name: `cmu_spice_eng_recipe_1000`

Voice Directory: `cmu_spice_eng_recipe_1000`

Tasks:

- ●  voice (and delete current one)
- `cmu_spice_eng_recipe_1000`
- ●  `waves/`
- ●  `txt.done.data`
- ●  `lexicon lexrules`
- ●  `lab/`
- ●  `ccoefs/`
- ●  `trees/`
- ●  `festvox/`
- ●
- ●  `festvox_cmu_spice_eng_recipe_1000_cg.tar.gz`

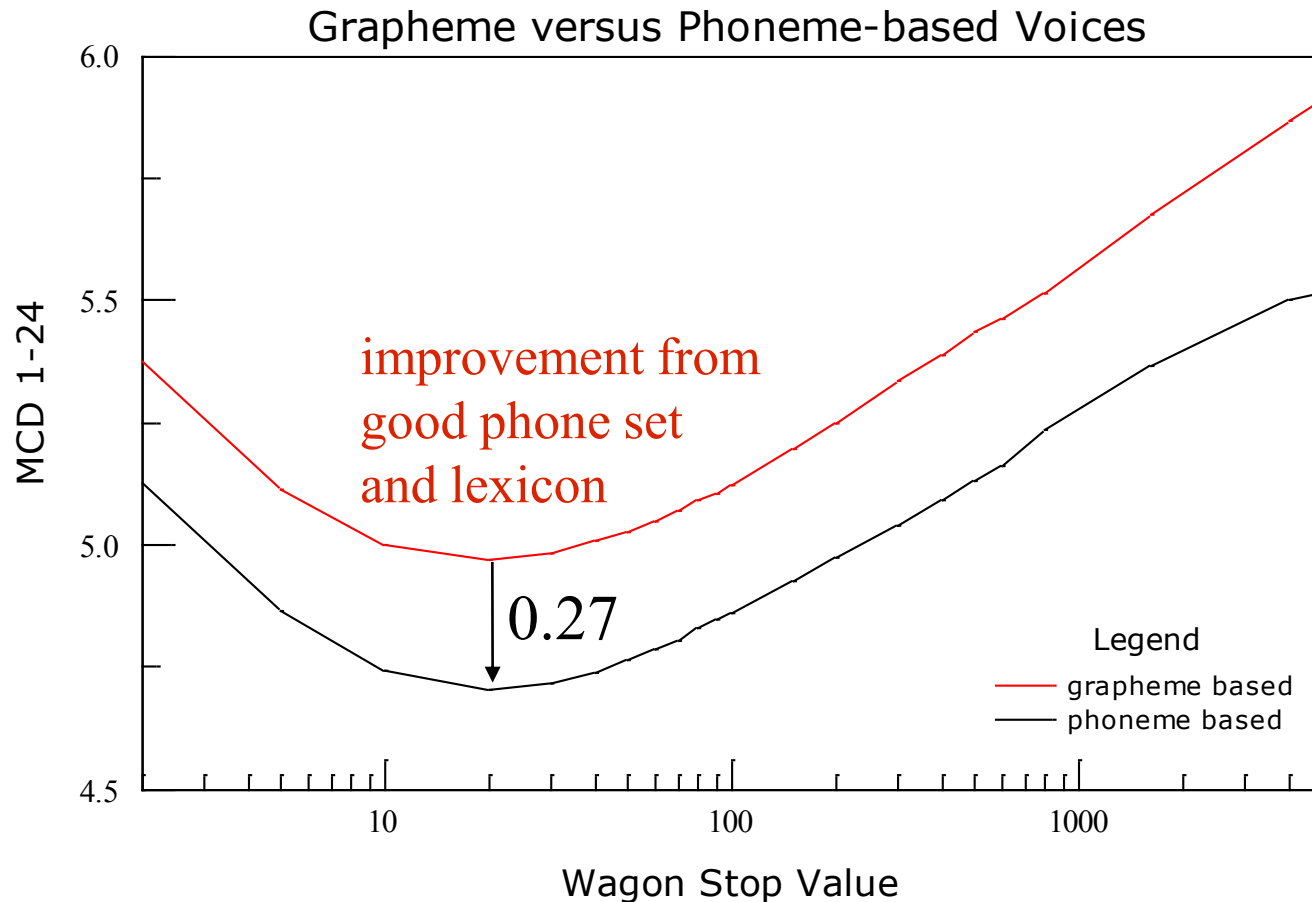
# Initial evaluations

- Conducted 2 semester-long lab courses
  - students use SPICE to create working ASR and TTS in a language of their choice
  - bonus for the ambitious
    - train statistical MT system between two languages
- to create a speech-to-speech translation system
- Evaluation includes
  - user feedback on difficulties
  - time to complete
  - ASR word error rate
  - TTS voice quality (this paper)

# Effect of a good Lexicon

- Want to simulate what you get with a sub-optimal phone set and a poor lexicon
- Idea: use a grapheme-based voice
  - 26 letters a-z are a substitute 'phone' set
  - no IPA and linguistics features
  - English has highly irregular spelling
    - the acoustic classes are impure
    - caveat: measuring global voice quality not mispronounced words
- Results
  - MCD improves by 0.27
  - consistent across CART stop value

# Grapheme vs Phoneme English voices



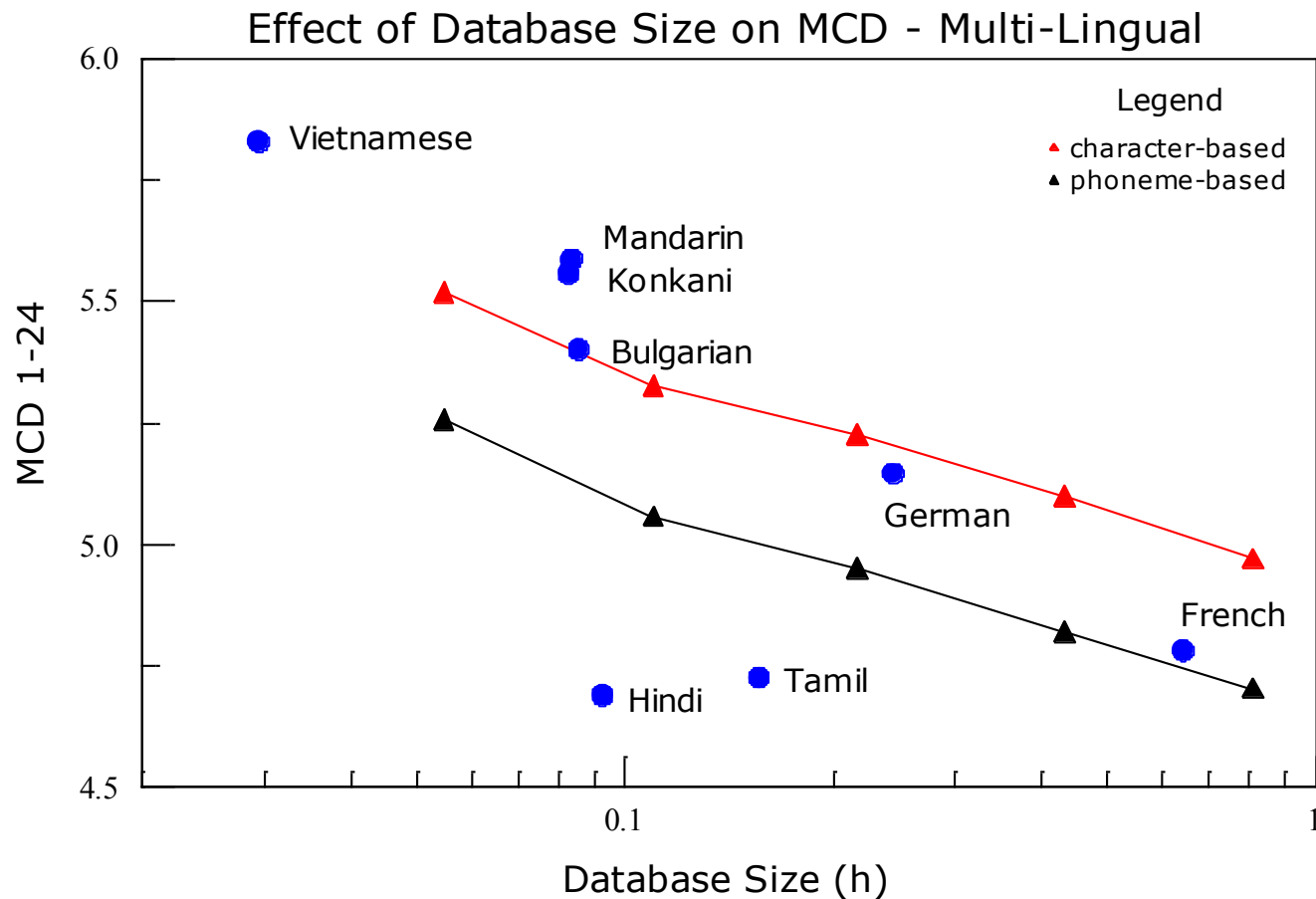


# 10 non-English test languages

- European
  - Bulgarian, French, German, Turkish
- Indian
  - Hindi, Konkani, Tamil, Telugu
- East Asian
  - Mandarin, Vietnamese

# Evaluating non-English voices

- For a frame of reference, we need a *good* and a *bad* voice
  - Phoneme-based English is “*good*”
  - Grapheme-based English is “*bad*”
- Data covers 3m to 1h of speech
  - may be extrapolated to about 4h
- Non-English voices are from student lab projects

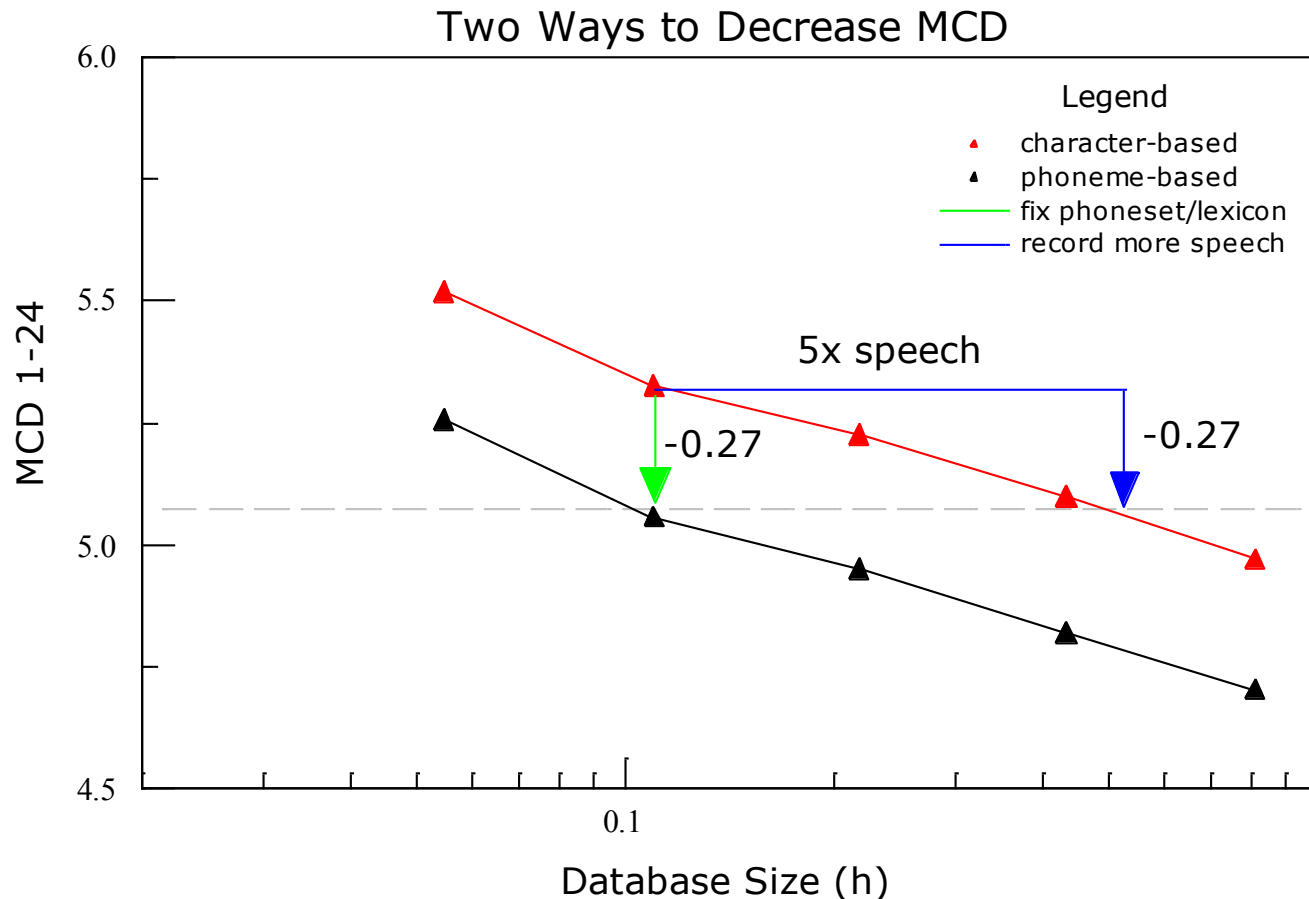


# Characterizing voice quality

- Reference frame permits a quick assessment
  - French is in good shape
  - German could use lexicon improvements
  - Hindi and Tamil are good for their size
    - recommend: collect more speech
  - Bulgarian, Konkani and Mandarin need more speech and a better lexicon
  - Vietnamese voice had character set issues
    - resulted in only  $\frac{1}{4}$  of the speech being used

# More speech or a better lexicon?

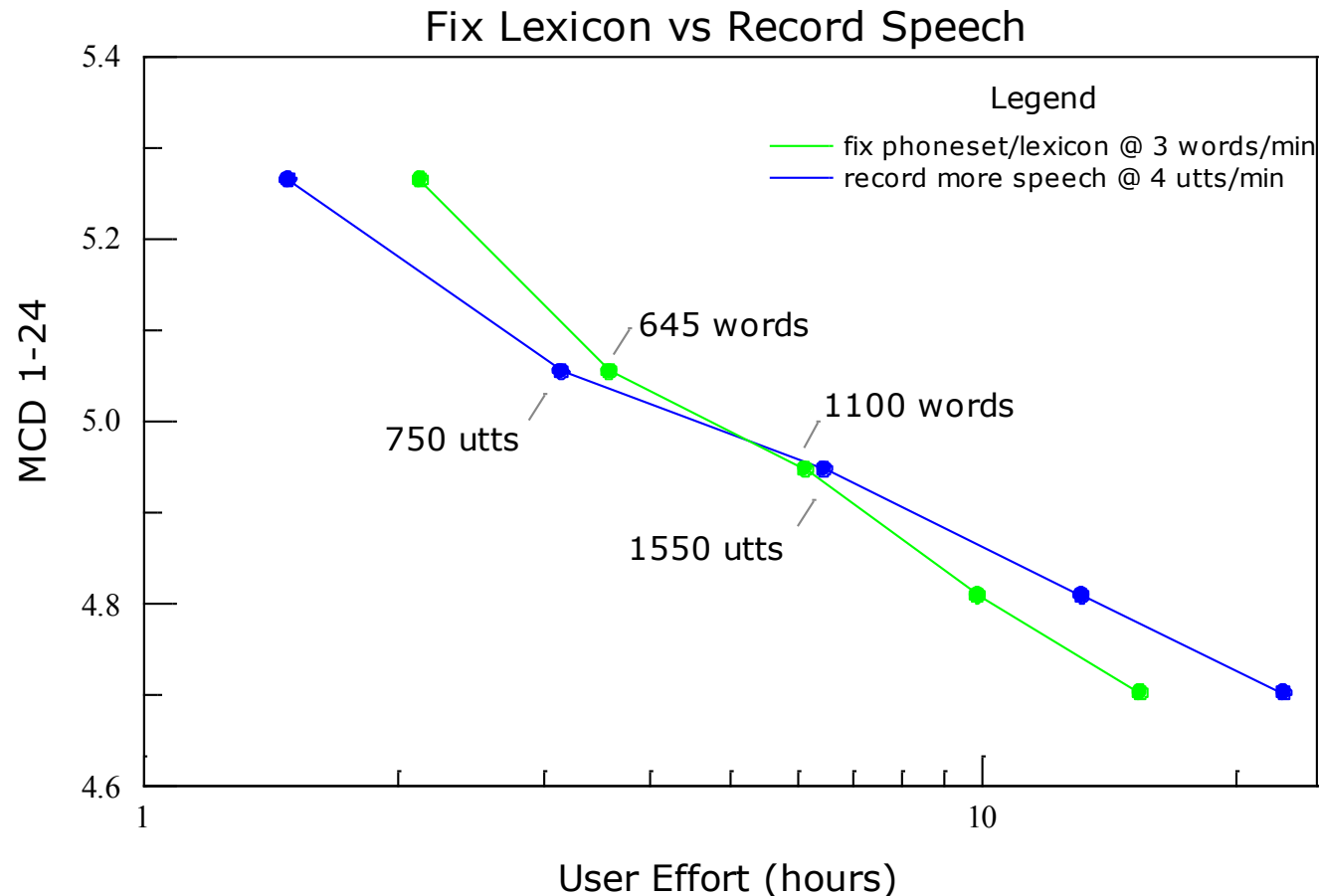
- From the English MCD error curves
  - 5x the speech = fixing the phoneset + lexicon



# More speech or a better lexicon?

- Which is more time effective?
  - assume 3-4 sentence-length recordings per minute
  - assume 2-3 lexicon verifications per minute
- Answer
  - small database — record more speech
  - large database — work on the lexicon
  - the transition point is language-dependent
  - it also depends on the relative speed of recording and lexicon verification

# More speech early, fix words later



# Research Conclusions

- Language dependence
  1. Our language-dependent features are not critical
  2. Best stop value lies in 20-50 range, and is stable
- Measurement
  1. Cepstral distortion is useful quality measure
  2. Two “parallel lines” provide a frame of reference
- Efficiency
  1. Doubling speech reduces MCD by 0.12
  2. Adding lexicon to English reduces MCD by 0.27



# Research Recommendations

- Human factors
  1. Interleave recording and lexicon work  
(too long on one task is mind-numbing)
  2. Emphasize recording early, lexical work later
- Future work
  1. Correlate MCD with listening tests
  2. Field testing with more users
- <http://cmuspice.org>

# Focus on TTS

- Main research questions
  1. To what extent is language-dependent expertise required of the user?
  2. To improve the synthesizer, what is the most efficient use of the user's time?
  3. How can we measure the user's progress?

# Research question in detail

- Language dependence
  1. Which features matter the most in CART tree training?  
Are language-dependent features critical?
  2. What is the best 'stop value' for training?
- Measurement
  1. Can an objective measure be used to estimate the quality of a voice, in any language?
  2. Can this information motivate and inform the user?
- Efficiency
  1. Rate of improvement as more speech is recorded?
  2. Rate of improvement as the lexicon is expanded and corrected?

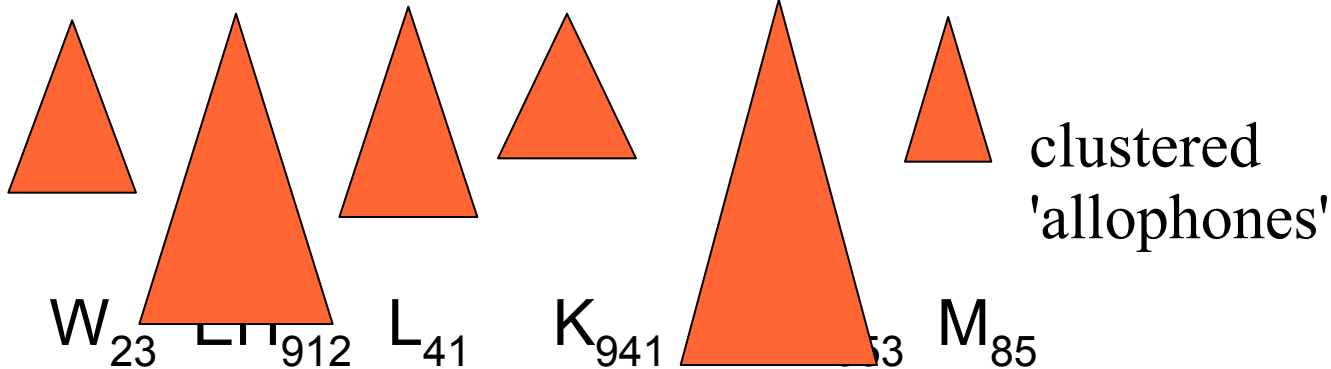
# TTS overview

- “welcome”

lexicon or LTS

symbolics

- W EH L K AH M



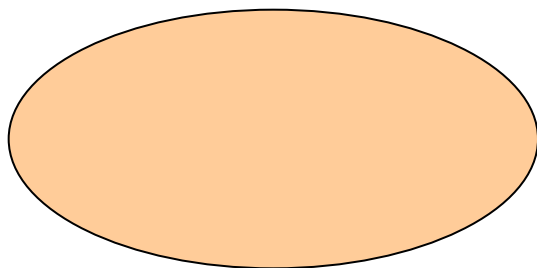
acoustics

units or trajectories

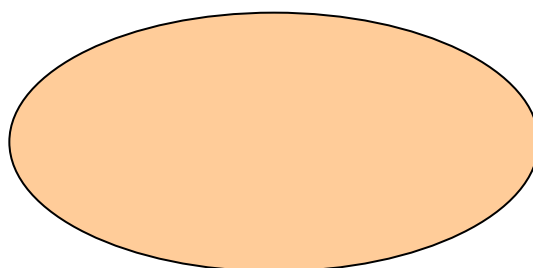
Key point - quality of CART trees depends on:  
training features, amount of speech, label accuracy

# Context-dependent CART training

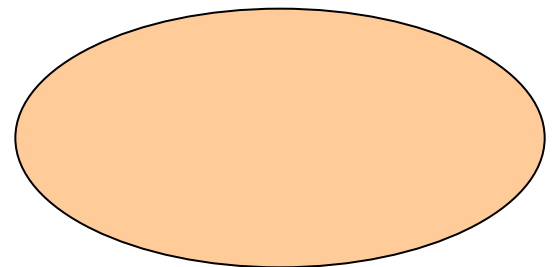
- Suppose text is “hi welcome to”
  - when training the  $EH_1$  state we use name feats
  - prev states: ...  $AY_3 W_1 W_2 W_3$
  - next states:  $EH_2 EH_3 L_1 L_2 \dots$
  - prev phones: # HH AY W
  - next phones: L K AH M



**W**



**EH**



**L**

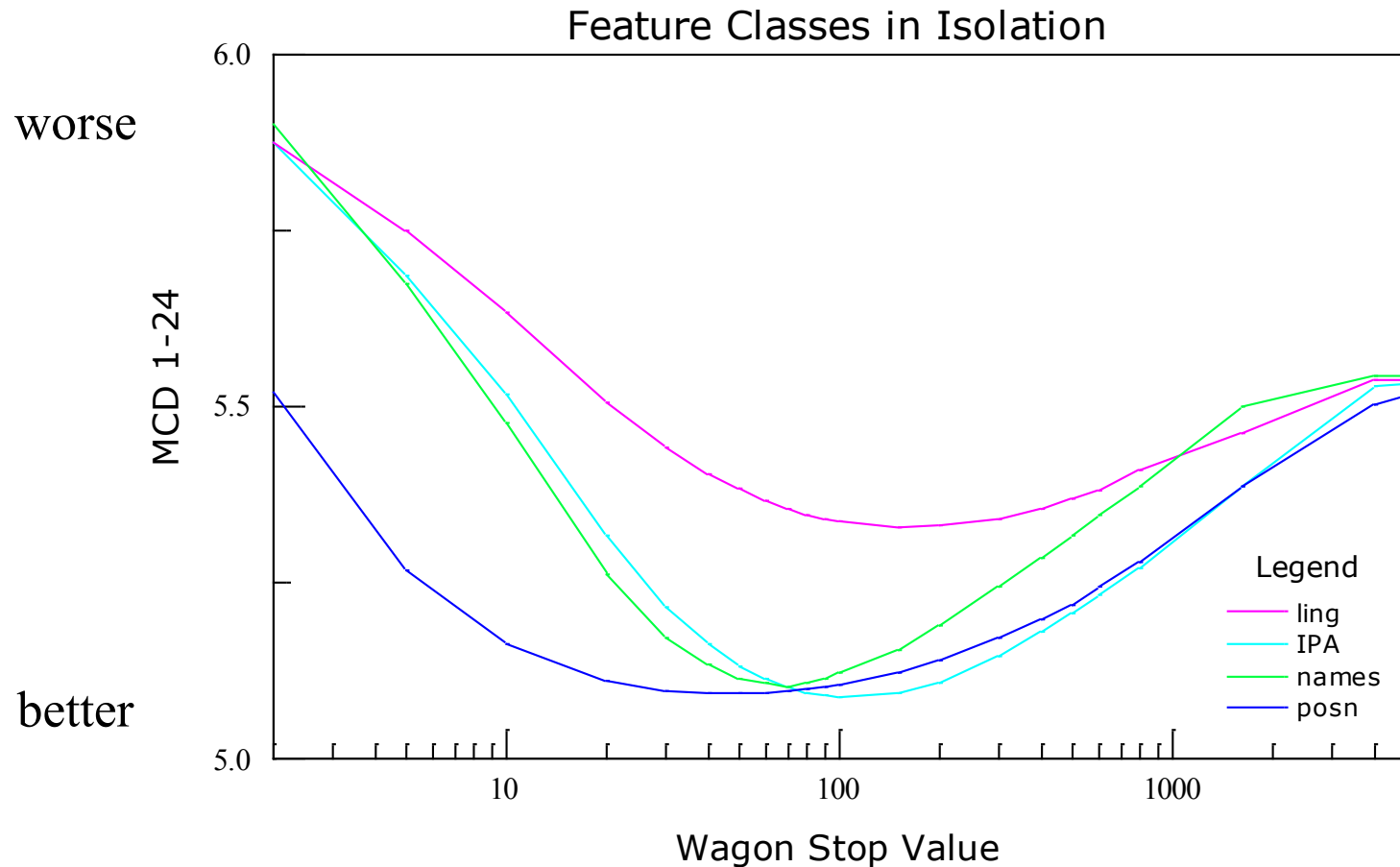
# More CART tree features

- Four categories of training features
  1. names: phoneme and HMM state context
  2. position: e.g. number of frames from beginning of state, percentage in from beginning
  3. IPA: International Phonetic Association features, based on phoneme set
  4. linguistic: e.g. parts of speech, syllable structure
- level of language expertise required
  - 1. and 2. are language-independent
  - 3. requires an IPA-based phoneset
  - 4. requires a computational linguist

# Calibration experiments in English

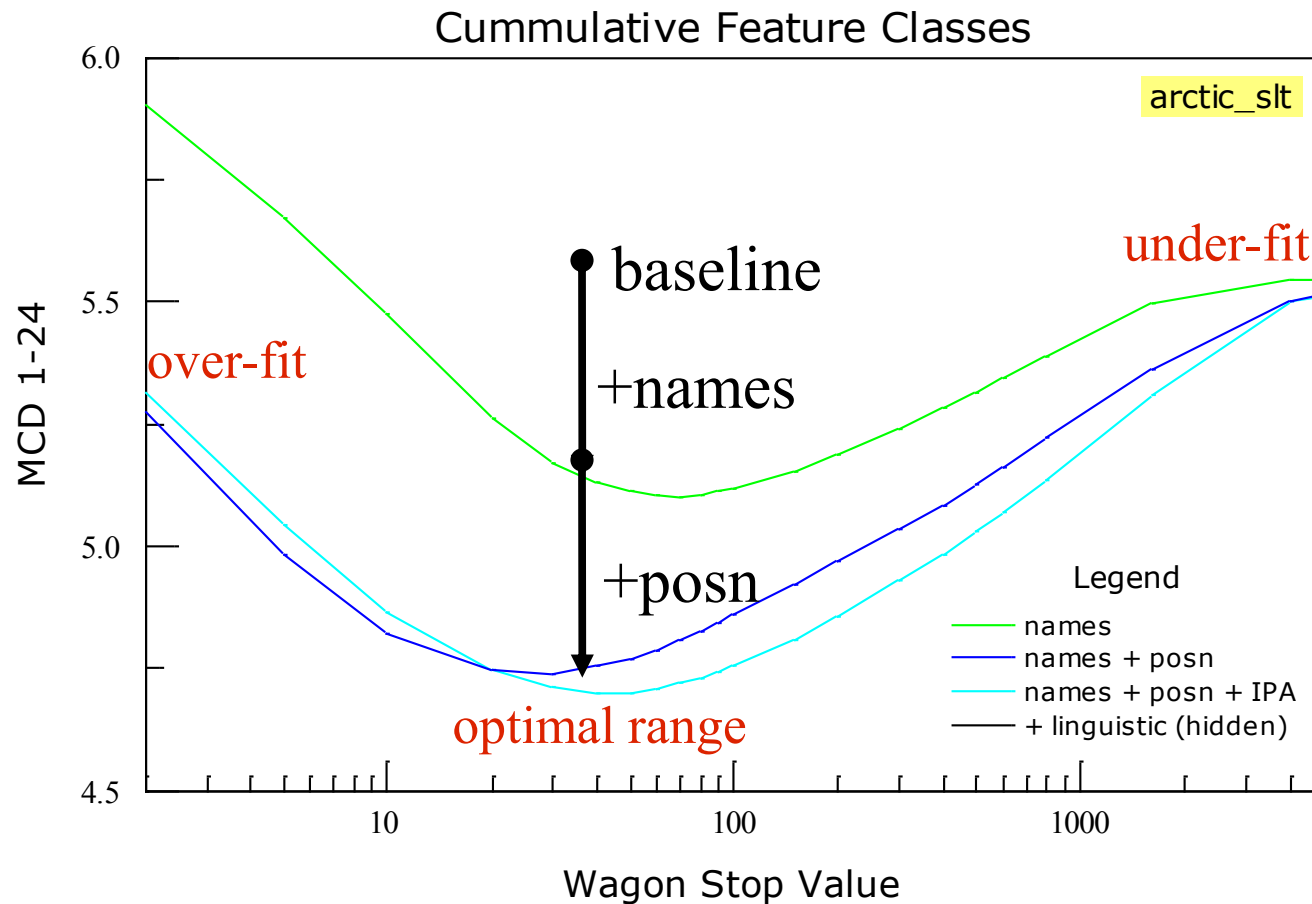
- Use a studio-recorded database (arctic\_slt)
  - 1 hour of clean speech
    - 90% training / 10% test – partitioned 10 times into separate testing sets
    - vary the amount of speech used to train
    - vary the CART training features
    - vary the CART stop value
- Compute mean mel cepstral distortion (MCD)
  - average frame-to-frame Euclidean distance between synthesized and original wavefile
  - let  $v$  = sequence of 25-D cepstral frames, 5 ms step

# Effect of isolated feature classes





# Effect of combined feature classes



# Effect of feature classes

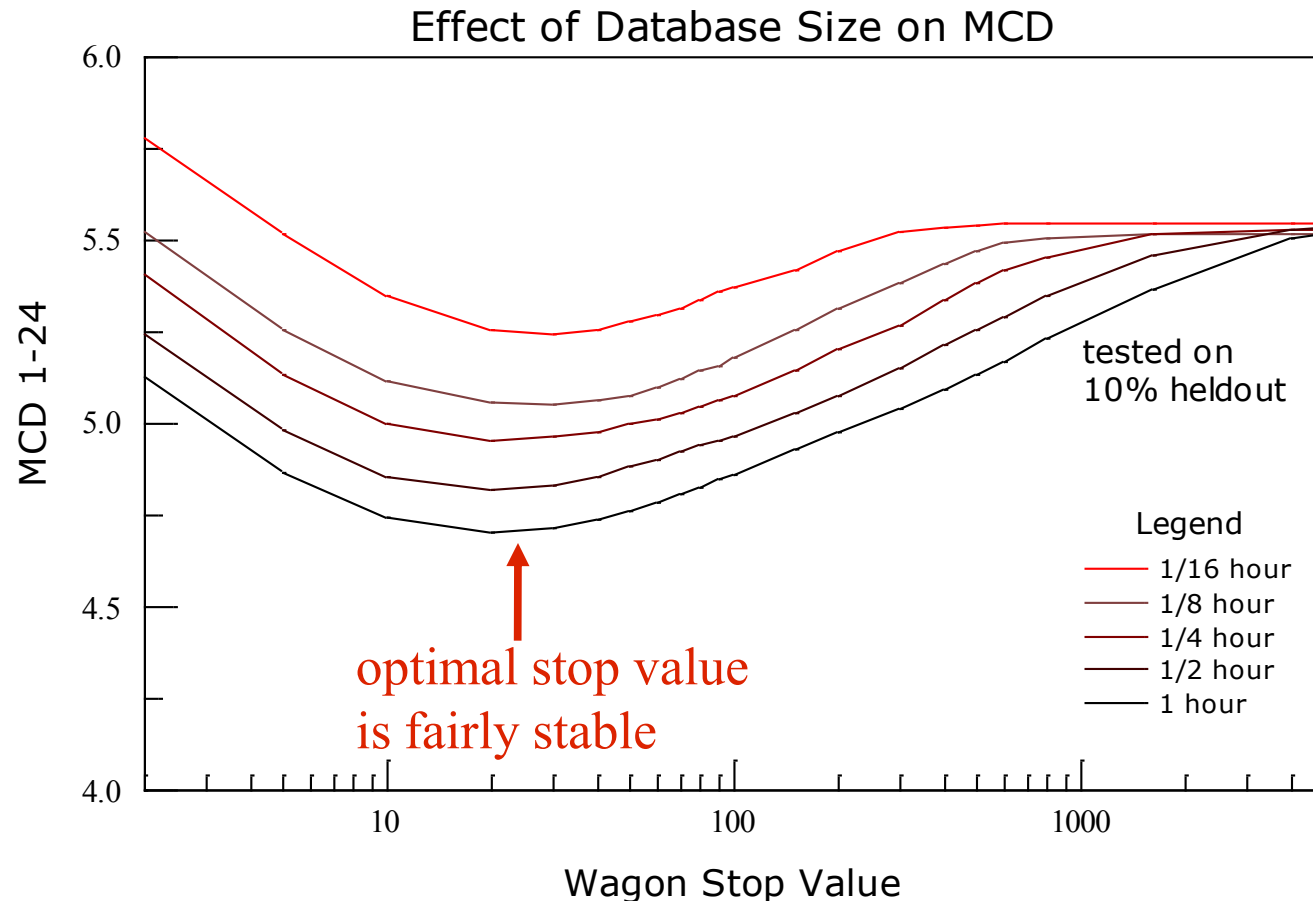
- lower numbers are better
  - $\sim 0.2$  is perceptually noticeable
  - $\sim 0.08$  is statistically significant
- the first two feature classes matter
  - from the minimum values of each feature class...

<i>Feature class</i>	<i>Features</i>	<i>Lang dep.</i>	<i><math>\Delta</math> MCD</i>
no CART trees	0	no	baseline
name symbolics	16	no	- 0.452
position values	7	no	- 0.402
IPA symbolics	72	yes	- 0.001
linguistic sym.	14	yes	+ 0.004

# Effect of database size

- Doubling speech reduces MCD by  $0.12 \pm 0.02$ 
  - a consistent result over many data points
  - thus 4x the speech is needed for a definite perceptual improvement
    - i.e. play two voices side-by-side and the larger voice is clearly better
- Exception at small end
  - from 3.75-→7.5 minutes MCD drops by 0.2
  - 10 min of speech can be considered the bare- minimum starting point

# Effect of database size on MCD curves



# Plenty of room at the high end

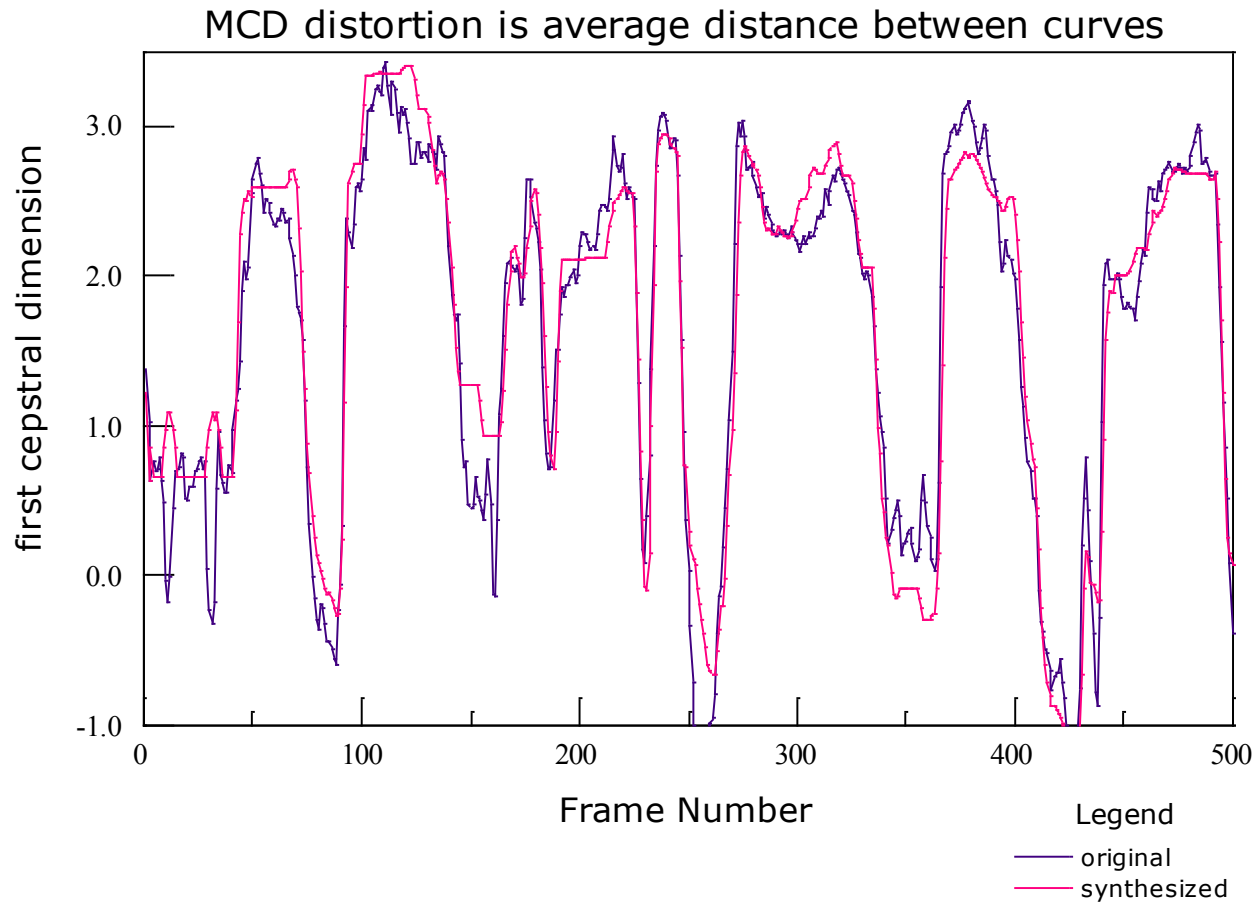
- Point of diminishing returns not evident in these experiments
- Where is the asymptote?
  - don't know yet
  - maybe 20 hours of consistently recorded speech
  - however, large databases recorded over multiple days are plagued by inconsistent recordings

# Backup Material

# Trial experiences

- Jan-May 2007 – 11733 lab course
  - hands on laboratory in building speech-to-speech translation system using SPICE tools
  - 11 students covering Bulgarian, Chinese, German, Hindi, Kankani, Thai, Turkish
- Lessons learned
  - phoneme selection too hard
  - making corrections was awkward
    - “don't bother me when the prediction is correct”
  - want synthesized pronunciation in their voice
  - need guidance on when to attempt voice build
  - need integrated evaluation and testing

# 1-D Illustration of MCD





# More speech early, fix words later

<i>data size (h)</i>	<i>number utts</i>	<i>unique words</i>	$\Delta MCD$	<i>equiv. utts</i>	<i>recording time</i>		<i>lexicon time</i>		<i>better action</i>
					<i>3/min</i>	<i>4/min</i>	<i>2/min</i>	<i>3/min</i>	
1/16	77	384	-.261	356	119	89	192	128	<b>record</b>
1/8	154	645	-.471	752	215	188	323	215	~ record
1/4	311	1103	-.579	1546	515	<b>387</b>	552	<b>368</b>	<b>similar</b>
1/2	607	1770	-.718	3144	1048	786	885	590	~ lexicon
1	1132	2766	-.824	5464	1821	1366	1380	922	<b>lexicon</b>

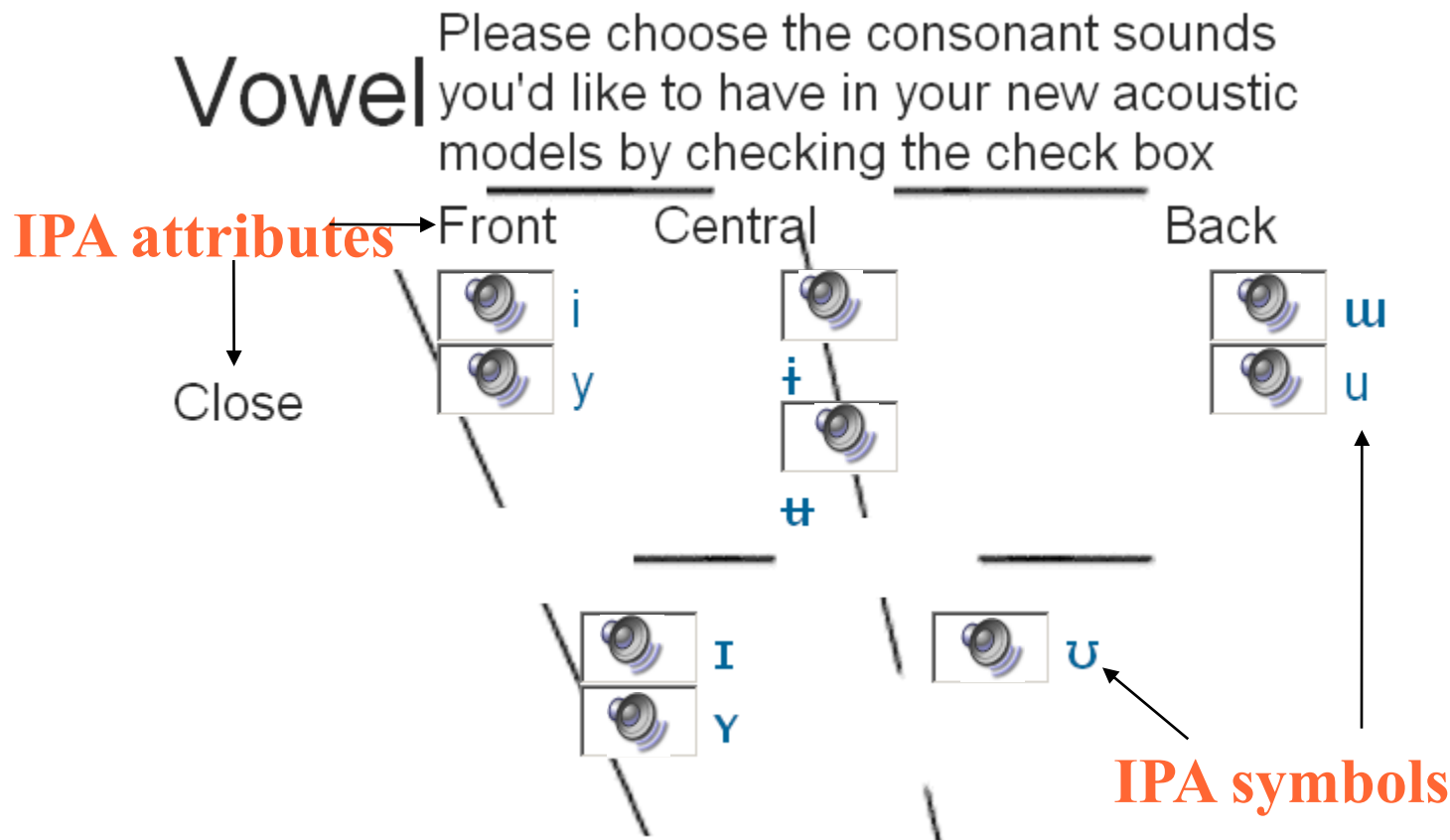
– e.g. third line

311 utts + fixed lexicon = 1546 utts, grapheme-based

387 min to record 1546 utts only (fast case)

368 min to record ¼ hour and fix lexicon (fast case)

# Phonemes are hard to identify



- linguists spend years in careful analysis
- non-linguists are baffled

# Phonemes are hard to use

- Spelling usually different from pronunciation
  - getting a word right is hard
  - keeping it consistent is even harder

CMU SPICE

LexLearner

The screenshot shows the LexLearner web interface. On the left is a sidebar titled 'Build Your System' with a list of steps: 'Text and prompt selection (help)', 'Audio collection (help)', 'Phoneme selection', 'Grapheme-to-phoneme rules (help)', 'Lexicon pronunciation creation (help)', 'Build acoustic model (help)', and 'Build language model (help)'. Below this is a box for 'Phoneme labels for your language:' containing the letters P, B, T, D, K, G, M, N, R, F, W, S, Z, SH, ZH, H, L, I, Y, U, E, O, A, YA, YU, TZ, DZ, TCH. The main area shows a progress bar at 66.7399% and a 'Rule editor' section. Three orange speech bubbles with black text are overlaid: 'system selects new word' points to a word in the list; 'or correct word by hand' points to a 'Skip this word' button; 'verify and submit word' points to a 'Remove this word' button. Three yellow ovals with red borders are also present: one under the first speech bubble, one under the second, and one under the third. The text 'TOVA' is written in red inside the middle yellow oval. At the bottom of the main area, there is a button labeled 'Pause and Build Lexicon'.

# Good G2P rules ease lexicon task

