Speech Processing 15-492/18-492

Speech Synthesis
Waveform generation 2
Speech Synthesis

- **Text Analysis**
  - Chunking, tokenization, token expansion

- **Linguistic Analysis**
  - Pronunciations
  - Prosody

- **Waveform generation**
  - From phones and prosody to waveforms
Unit Selection
The “standard” method
“Select appropriate sub-word units from large databases of natural speech”

Parametric Synthesis: [NITECH: Tokuda et al]
HMM-generation based synthesis
Cluster units to form models
Generate from the models
“Take ‘average’ of units”
Old vs New

Unit Selection:  
large carefully labelled database  
quality good when good examples available  
quality will sometimes be bad  
no control of prosody  

Parametric Synthesis:  
smaller less carefully labelled database  
quality consistent  
resynthesis requires vocoder, (buzzy)  
can (must) control prosody  
model size much smaller than Unit DB
Example CG Voices

7 Arctic databases:

1200 utterances, 43K segs, 1hr speech

awb  bdl
clb  jmk
ksp  rms
slt
## Data size vs Quality

### slt_arctic data size

<table>
<thead>
<tr>
<th>Utts</th>
<th>Clusters</th>
<th>RMS F0</th>
<th>MCD</th>
</tr>
</thead>
<tbody>
<tr>
<td>50</td>
<td>230</td>
<td>24.29</td>
<td>6.761</td>
</tr>
<tr>
<td>100</td>
<td>435</td>
<td>19.47</td>
<td>6.278</td>
</tr>
<tr>
<td>200</td>
<td>824</td>
<td>17.41</td>
<td>6.047</td>
</tr>
<tr>
<td>500</td>
<td>2227</td>
<td>15.02</td>
<td>5.755</td>
</tr>
<tr>
<td>1100</td>
<td>4597</td>
<td>14.55</td>
<td>5.685</td>
</tr>
</tbody>
</table>
Databases size vs Quality

- **SPS**
  - $rms_{100}$
  - $rms_{1132}$

- **Unit selection**
  - $rms_{100}$
  - $rms_{1132}$
Advantages of SPS

- **Statistical Parameter Synthesis**
  - More robust to errors in data
  - Requires less data
  - Models are smaller (< 2MB vs > 1GB)
  - Parametric models allows further processing
Disadvantages of SPS

- **Statistical Parametric Synthesis**
  - “buzziness” of resynthesized speech
  - *Doesn’t sound as good as the best unit selection*
  - *Still experimental*
Parametric Speech Models

- **Emotional Speech Synthesis**
  - Can collect small amounts of emotional speech
  - Build models that transform base model

- **Cross Lingual Speech Synthesis**
  - From language independent models
  - Transform with small amount of target language

- **Use various ASR techniques**
  - Adaptation
  - Discriminative training
  - Use as much CPU as the ASR people
Corpus-based Synthesis

- *Doesn’t really “just work”*
  - Need to consider database content
  - Speaker style
  - What you send to the synthesizer
The right type of database

- **Recording style defines synthesis style**
  - News stories will give news style-synthesizer
  - News style not appropriate for dialog system

- **Natural vs controlled prompts**
  - Natural utterances good for general synthesizer
  - Domain targeted better for domain synthesizer
The right type of speaker

- Professional speakers are better
  - Consistent style and articulation
  - Lecturers, teachers are often better
  - You can learn to do it well
- Ideal selection process (AT&T: Syrdal 99)
  - Record 20 professional speakers
  - Build limit synthesizers from them
  - Collect many peoples preferences (> 200)
  - Record the “best” speaker(s)
- Find correlates in human speech
  - High power in unvoiced speech
  - High power in higher frequencies
  - Larger pitch range
- Different people prefer different voices
  - Provide a choice
  - Errors are sometimes diminished by novelty
The right type of things to synthesize

- **Instead of making the db appropriate**
  - Restrict the text input

- **Domain synthesis**
  - “The temperature is X degrees and the outlook is Y”.

- **Make the database directly match text**
  - Fill templates with values
Limited Domain Synthesis

- **General Unit Selection Synthesis**
  - Can be high quality
  - Sometimes bad quality
  - Expensive to tune

- **Limited Domain Synthesis**
  - Design database to match exactly what you want to synthesize
  - Only reasonable if building voice per application is easy
Building a Voice

- Designing the Prompts
- Recording the Prompts
- Labeling the Utterances
- Finding parameters (F0, MCEP)
- Building the synthesis voice
- Tuning and Testing
Designing the Prompts

- From a grammar
  - System says: The temperature is X degrees
- From example data
  - Using example output from the existing system
- From thinking about it
  - But you *will* make mistakes
- Ideally:
  - Word coverage
  - Bi-gram coverage
  - Prosody position coverage
- Design prompts to limit prosodic variance
  - Boston, is that where you want to go?
  - Do you want to go to Boston?
Domains

- **Fixed template filling**
  - Talking clocks, 24 utterances
  - Weather 100 utterances (don’t say place name)
- **Larger domains (spoken dialog systems)**
- **Let’s Go bus information (Hybrid)**
  - Standard prompts
  - Times and bus numbers
  - 15,000 bus stop names (not fully covered)
  - Backup general synthesis prompts
A talking clock

- **Design the prompts:**
  - The time is now, about five past one, in the morning
  - The time is now, just after ten past two, in the morning
  - The time is now, exactly quarter to three, in the morning
  - The time is now, almost twenty past four, in the morning

- **Get full word coverage**
  - *really* test you have word coverage
  - No, *really* test you have word coverage
Record the prompts

- **Get highest quality recordings**
  - Recording studio
  - Head mounted mike
  - Repeatable conditions
- **Get signed permission**
  - Explain what you are doing
Label the data

- Using HMM-based or DTW-based system
  - Find the phoneme segments
- Simple cases (< 50 utterances)
  - Use DTW
  - Synthesize the prompts
  - Align synthesized prompts with actual prompts
Automatic Labeling
Automatic Labeling (bad)
Parameterization

- **Extract pitch marks from data**
  - Find voices/unvoiced regions
  - Add “fake” pitch marks during unvoiced regions

- **Extract MFCC pitch synchronously**
  - Instead of a fixed frame advance (e.g. 5ms)
  - Extract it at each pitch mark
  - Try to capture the spectrum at the pitch period
Pitchmarks
Building a LDOM synthesizer

- Build cluster tree on each unit type
  - Not just on phones
  - Tag phones with word they come from
  - $d_{\text{limited}}$ and $d_{\text{domain}}$ are treated as different
Tuning and Testing

- **Test it on some real data**
  - Ensure number/symbol expansions are correct
- **Prompts should probably be word expanded**
  - Flight US187 -> flight u s one eight seven
- **Remove bad prompts**
  - Or fix labels
- **Remember to keep access to the speaker**
  - If you have to update the system, you need the same speaker available
Summary

- **Unit selection vs Statistical Parametric Synthesis**
  - US: can be excellent (but not always)
  - SPS: more robust

- **Building a voice**
  - Databases design, recording, labeling
  - Parameter extraction and model building

- **Limited domain synthesis**