New language

- New language:
  - currently supported (or not)
- Diphone scheme and collected data
- Prosody by rule or from data
- Lexicon:
  - (don't collect your own)
  - letter to sound rules
- Orthographic processing:
  - maybe from roman characters

Doing all parts is probably too much work
Do one part well and rest with simple rules

Limited domain synthesizer

- Existing dialog system:
  - check its language output distribution
  - is it limited
- Possible domains:
  - Roomline
  - Bus schedules.

New prosodic style

- Record data in one (or more) style:
  - build prosodic models for an existing voice
  - build prosodic models for an supported language
- Show it sounds "better" than default

Information presentation

- Get data from resource and say it
- Three stages:
  - get data (from web)
  - massage it into usable form
  - modify synthesizer to say it well
- Possible areas:
  - mapquest.com
  - stock quotes
  - Dictionary lookup (m-w.com)
- Techniques:
  - text analysis
  - multiple voices, other noises
  - speed, style, etc
  - as Sable markup ?

New voice for new language

- Phone set
- Token processing rules (numbers etc)
- Prosodic phrasing method
- Word pronunciation (lexicon and/or letter to sound rules)
- Intonation (accents and F0 contour)
- Durations
- Waveform synthesizer
- plus something hard you forgot about

http://festvox.org

Phoneset

- Can't do anything without this
- Choose an existing one:
  - note dialect of your speaker
- May be defined by your lexicon/diphone db
- may be other factors:
  - lexical stress (accents)
  - allophones (flaps, voicing etc)

Tokenization

- Start from native character sets:
  - its not a TTS engine if you don't
  - (maybe a romanized-only form is useful)
  - maybe ASCII-ized version is required too
- Word boundaries:
  - Chinese, Japanese
  - complex morphology
- Numbers, symbols
- Homographs:
  - no vowels (?)
  - Kanji

Prosodic phrasing

- Part of speech tagging:
  - statistically trained phrasing model
- Hand written rule:
  - but test this
- Just use punctuation
- New techniques (Parikar)
  - Induced POS tags and Phrase Parsing

Word pronunciation

- Find an existing lexicon:
  - note its copyright
- Write letter to sound rules:
  - often easy (though may need native speaker)
  - may be something missing (stress)
- Only have small tailored lexicon
- Use lexicon bootstrapping technique

Intonation

- Get data and train from it:
  - requires some framework
  - require labeling
- Simple rules do work well on simple sentences
- Use an existing model from other language:
  - often works better than writing rules

Durations

- Fixed or average
- Train from data
- Borrow models from other languages:
  - has some justification
  - often works quite well

Waveform Synthesizers

- Collect your own diphones
- Use existing MBROLA database:
  - Faculte Polytechnique de Mons
  - many diphone synthesizers
- Borrow diphones from another language:
  - often works, esp for minority languages
  - givens you something quickly
- Build a clustergen voice from data

Other considerations

- Testing and evaluation:
  - synthesis in other languages always sounds better
  - get *native* speakers to evaluate it
- Ensure you have copyright:
  - if you want to use the voice, make sure
  - you have permission to use *all* parts
- What you think sound good:
  - still sounds awful to others

using festvox.org

- detailed documentation
- mailing list for similar projects
- example databases
- Scripts:
  - diphone_setup cmu nl jan
  - creates directory structure
  - diphone lists
  - basic .scm files

but diphones are boring ...

Building unit selection synthesizers

- Select text with rich phonetic coverage:
  - optimize for diphone coverage
  - or use acoustic modeling techniques
- record *very* carefully
- label *very* carefully
- tune and build clunit synthesizer

Building General SPS Voice

- `SPPPDIR/src/festvox/src/clustergen/setup_cg`
  - `setup_cg INST LANG VOX`
  - `setup_cg cmu de hans`
- Instatiates language files:
  - Need to fill in some things by hand
  - `festvox/*_phoneset.scm`
  - `festvox/*_durdata.scm`
  - `festvox/*_lexicon.scm`

Phonetic based data selection

- From a large set text:
  - select "nice" utterances
  - 5 to 15 words long
  - all in lexicon, no homographs
  - text2utts -dbname txt_ text.txt -o text.data
- Convert text to phonemes:
  - text2utts -level Segment -itype data text.data -o text.seg.data
- Select utterances with maximal (di)phone coverage:
  - dataset_select text.seg.data
- Extract the selected utts from text.data:
  - dataset_subset text.data text.seg.data.selected
- use make_nice_prompts

Selection Example

- alice30.txt (152k)
- `text2utts -dbname alice_ alice30.txt -o alice.data`
  - 1920 total utterances
  - 668 "nice" utterances

  (alice_00003 "THE MILLENNIUM FULCRUM EDITION 3.0")

  (alice_00011 "I shall be late!"")

  (alice_00025 "Would the fall NEVER come to an end!")
- `text2utts -level Segment -itype data alice.data -o alice.seg.data`

  (alice_00003 "pau dh ax m ax l eh n iy ax m f uh l k

  r ax m ax d ih sh ax n th r iy p oy n t z ih r ow pau")
- `data_select alice.seg.data` – 189 utterances
- `dataset_subset alice.data alice.seg.data.selected >alice1.data`

Prompt Selection: new languages

- Get large amount of text
- Build word list:
  - find word frequencies
- Use "nice" sentences:
  - contain only frequency words
  - 5 to 15 words
- To select "phone" coverage:
  - select based on letter context

Synthesis without a Phoneme Set

Use the *letters* as phones

- 26 "phonemes":
  - ("alan" n (a l a n))
  - ("black" n (b l a c k))
- Spanish example (Castillain and Columbian)
  - 419 utterances selected from newspapers
  - SphinxTrain HMM-based acoustic modeling (cf ISL/ASR)
  - Simple pronunciation lexicon:
  - *policìa* → `p o l i c i1 a`
  - *cuatro* → `c u a t r o`

Spanish "letter" synthesizer

Word	Castillian	gloss
casa	/**k** a s a/	house
cesa	/**th** e s a/	stop
cine	/**th** i n e/	cinema
cosa	/**k** o s a/	thing
cuna	/**k** u n a/	cradle
he**c**hizo	/e **ch** i th o/	charm, spell

In Spanish the letter "c" may be pronounced /k/, /ch/ and /th/ or /s/ (depending on dialect). The choice of phone is determined by the letter context.

English "letter" synthesizer?

Use the *letters* as phones

- 26 "phonemes":
  - ("alan" n (a l a n))
  - ("black" n (b l a c k))
- Build SphinxTrain models
-
  - "This is a pen."
  - "We went to the church at Christmas."
  - Festival intro – "do eight meat"

CMU ARCTIC Databases

- Use Gutenberg out-of-copyright texts:
  - mostly "northern" stories (hence "ARCTIC")
- 53996 Nice utterances
- Selection based on phones:
  - set A 593 utterances
  - set B 539 utterances
- Studio recording:
  - 32KHz, EGG, automatically labeled
- Four DBs released:
  - slt US female
  - bdl US male
  - jmk Canadian male
  - awb Scottish male
  - plus rms US male, clb US female
  - plus non-natives: french, japanese, indian

CMU SPICE Project

- Build ASR and TTS models for new languages
- Web-based
  - No speech expertise required (sort of)
- http://plan.is.cs.cmu.edu/Spice
- used in 11-733