Speech synthesis

Igor Sz oke ÚPGM FIT BUT Brno, szoke@fit.vutbr.cz

FIT BUT Brno

Speech synthesis - content

- Overview: Some terms, history, presence, ...
- Structure: Modules, scheme, brief description, ...
- Text normalization: Numbers, abbreviation, phonetic transcription
- Prosody: Melody, accents, timing, ...
- Units: Manual, automatic, corpus-based
- Signal synthesis: PSOLA, DSM, ...
- TTS applications: EPOS, Festival, ...

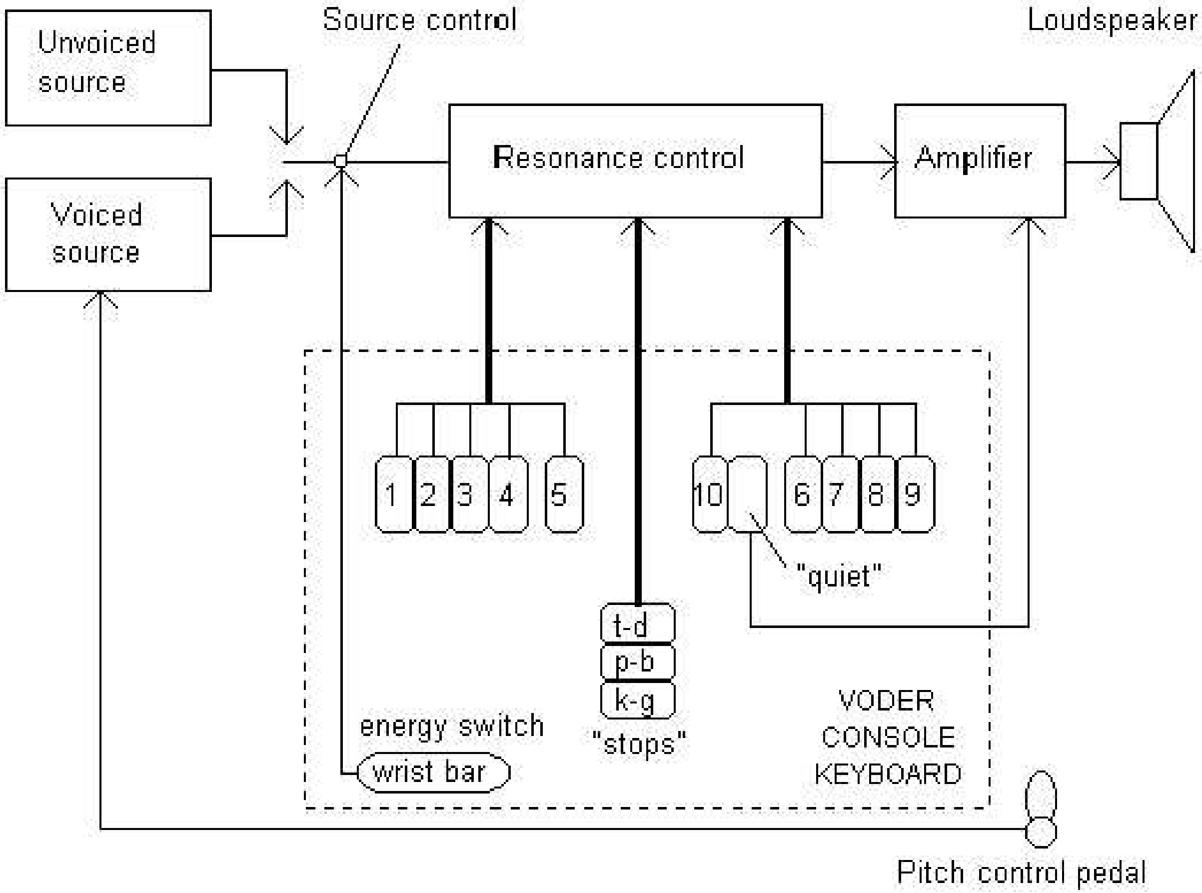
Speech synthesis - overview

- Recognition (User⇒PC), Synthesis (PC⇒User)
- Usage: when no other possibility to receive information (only by voice): blind people, phone applications (call centers), some experimental applications
- Possible usage: Anywhere you need to be focused on another else than the display such as car, intelligent house (kitchen, . . .), office
- TTS system is less computational expensive than ASR, but universal TTS is very huge
  .
- TTS system is interdisciplinar: Signal processing, theoretical informatics, natural language processing, phonetics, database systems, . . .

Speech synthesis - overview

- 1846 - The first mechanical synthesizer called *speech organ* - it could sing *God Save the Queen*
- 1922 - The first electrical synthesis device (buzzer and 2 resonant circuit for the first two formants), 1923 added the third format
- 1939 - The first electrical speech synthesizer *Voder*, human controlled (by keyboard and pedal)
- 1961 - The first phonemic-synthesis-by-rule program for digital computer
- 1968 - The first full text-to-speech system
- 1980's - The beginning of commercial TTS
- 1985 - PSOLA developed - prosodic modifications
- 1990's - Deterministic/Stochastic models, large databases, automatically labelled databases, . . .

Speech synthesis - overview



Block scheme of the Voder

Speech synthesis - overview

- Telephone applications such as: help lines, call centers, . . .
- Integrated into dialog systems - banks (not in CR) - Jean Hennebert lecture last year (see http://...cernocky/oldspeech)
- Navigation systems in cars
- Office and home (vision - who really use it now?). Bill Gates ⇒ We will handle PC via voice till 2014. So we will see ;o)
- Future? Read some Sci-Fi book :-)

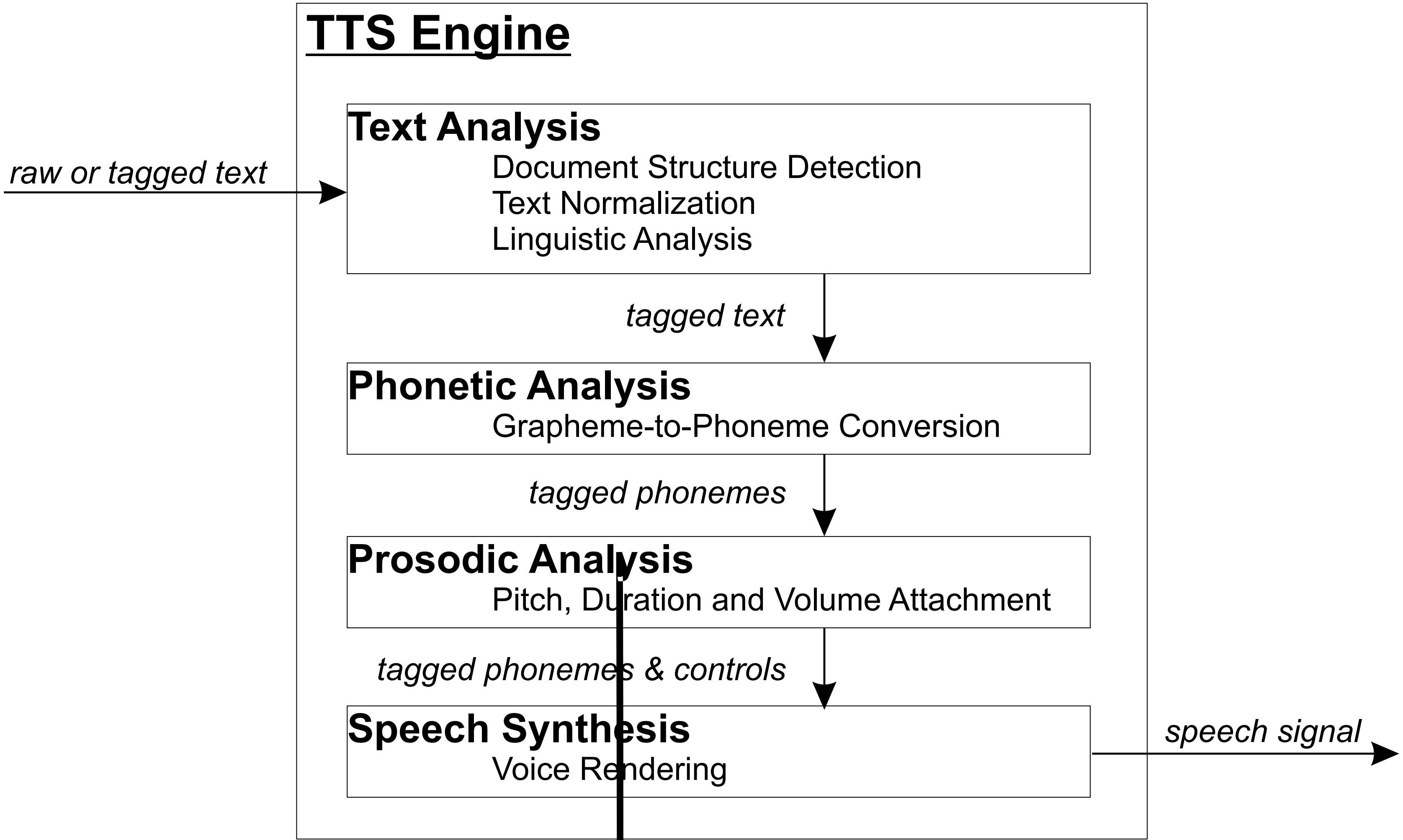
Speech synthesis - overview

- General TTS system is complicated and development is hard. Read a text and try to realize what you must do (prosody, lexical structure analysing, . . .). You can easily listen any incorrectness (listen strange Czech). Bad prosody can change meanings. If intelligibility is worse, you must concentrate on listening TTS ⇒ you can't do anything else ⇒ so is better to **read** the information. . .

Speech synthesis - TTS system complexity

- Part of the command and control application. TTS system with low vocabulary, units can be words or sentences. Usually no prosodic modifications in synthesis. "Low quality" concatenation of the units.
- Telephone application. TTS system with medium vocabulary (limited), word or sub-word units. Optional prosodic modifications in synthesis.
- TTS system in office application (input is arbitrary text). TTS system with large vocabulary, phoneme like units. Prosodic modifications and "high quality" concatenation expected.

Speech synthesis - TTS system structure

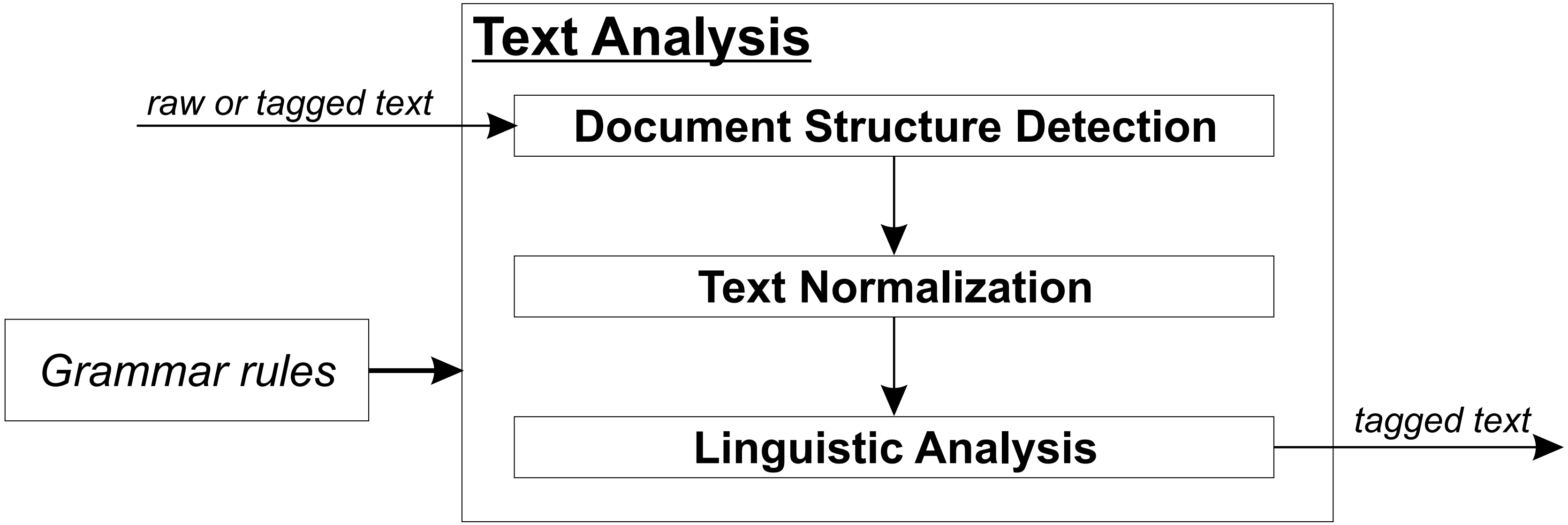


General TTS system scheme

Speech synthesis - TTS system structure

- Input is plain text (email, article, command, . . .) or tagged text (VoiceXML, HTML, . . .) with control commands for TTS system
- Text analysis tries to understand input text and puts sematic tags into the text
- Phonetic analysis parses text into phoneme string
- Prosodic analysis adds prosodic controls to the phoneme string (melody, accent, rate, pauses)
- Speech synthesis generates speech signal from given phoneme string (or other units) and prosodic controls
- VoiceXML ⇒ Possible topic for BP, RP, 'hack it' and write report . . . to be discussed with Igor.

Speech synthesis - Text analysis



Text Analysis block scheme

Speech synthesis - Text Analysis

- Text Analysis (*TeA*) is the first block in TTS system
- Tries to understand input text
- Puts tags with information about semantic of the text
- Can be used in other applications (Speech Recognition - Dialog systems)
- Language dependent. Different problems in different languages (English, French, Czech, Chines, . . .).
- Works with rules, knowledge base and huge corpus (dictionary)

Speech synthesis - TeA - Document Structure Detection

- Needed for large documents, document parts understanding (Chapter, Paragraph, Table, . . .)
- DSD is not needed for TTS system working with words or short simple sentence
- Puts to the text new sentences like "*Next chapter*" or "*Table with three columns*"
- Can generate information for prosodic analysis (little delay before next paragraph, . . .)
- Sentence breaking
- Output should be text compound of tagged sentences
- Works with rules

Speech synthesis - TeA - Text Normalization

- Important, every TTS should have it
- Substitution of "non-text" tokens by their text representation
- Numbers, dates, times (1, 2.4., 13:30, $5, . . .), abbreviations (BUT, . . .), symbols ($, @, ;-), . . .), other formats (chemical or math formulas, . . .)
- Uses spelling for special cases
- Declination rules in Czech language!
- Output should be text without special symbols
- Works with rules (eg. regular expressions for number conversion)

Speech synthesis - TeA - Text Normalization

Abbreviations:

'f.p.s.'→'frames per second', 'BUT'→'Brno University of Technolgy'

Symbols:

':'→'divided' or 'colon', '.'→'point' or 'dot' or nothing (part of sentence syntax)

Examples of reg. expr. for number normalization (for Czech language):

1([0-9][0-9]) → sto \1

2([0-9][0-9]) → dvě stě \1

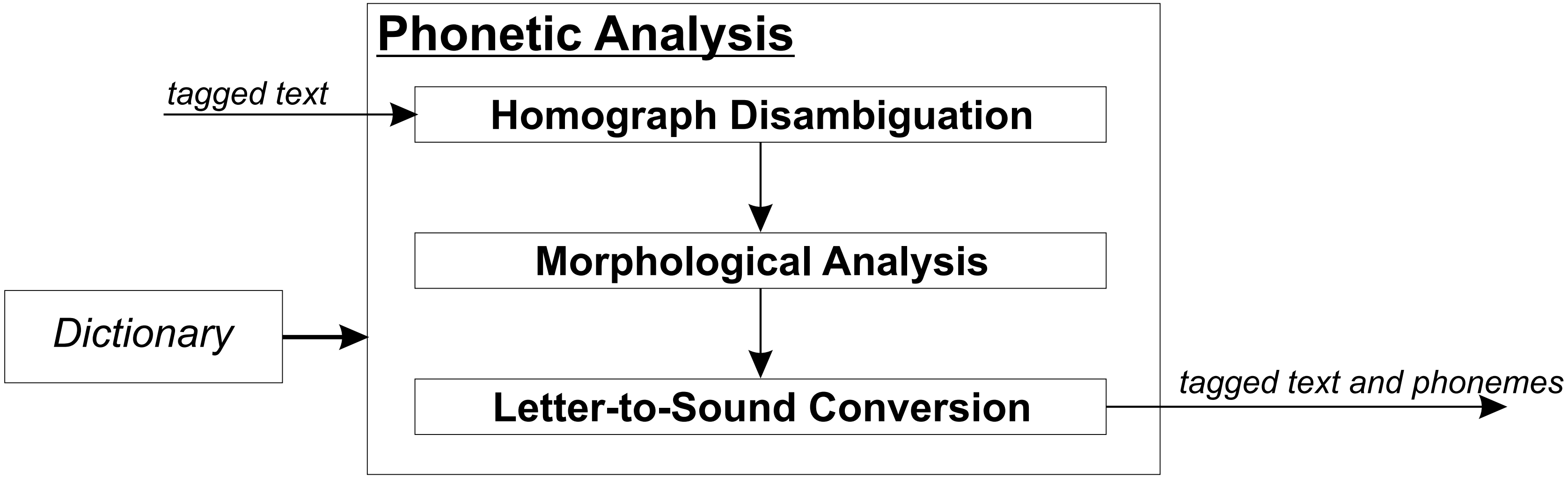
([34])([0-9][0-9]) → \1 sta \2

([5-9])([0-9][0-9]) → \1 set \2

Speech synthesis - TeA - Linguistic Analysis

- Syntactic and semantic parsing of the text
- Sentence breaking (clauses, sub-sentences)
- Word type, case, gender, . . .
- Word sense (*bank*), emphasis, direct speech (*he said: "Hello!"*, ...), sentence type identification, ...
- Generates information for prosodic analysis, badly understand text ⇒ incorrect prosody ⇒ it will not sound naturally or incorrect prosody can change meaning of the text!
- Output should be pure text with tagged semantic information
- Works with rules (grammars, decision trees, . . .)

Speech synthesis - Phonetic Analysis



Phonetic Analysis block scheme

Speech synthesis - Phonetic Analysis

- Phonetic Analysis (*PhA*) is the second block in TTS system
- Tries to split text into phonemes
- Defines what synthesizer will say
- Generates string of phonemes
- Closely tied to Text Analysis
- Main part is Grapheme-to-Phoneme conversion (letter-to-sound)
- Works with rules and database of units

Speech synthesis - PhA - Homograph Disambiguation

- Homograph - two words with different pronunciation (phonemes, stress) and same text presentation
- Example: *read* ([*ri:d*] × [*red*]), *bass* ([*baes*]fish × [*beys*]instrument)
- Decision of pronunciation is based on Text Analysis
- Some words with different sense and same pronunciation and text presentation (*bat*) could have impact to other words prosody
- Works with information from TA, rules and probability

Speech synthesis - PhA - Morphological Analysis

- Decomposition of the word into prefix, root and suffix
- Example: prefixes *in-, un-, pre-, sub-*, suffixes *-s, -'s, -ed, -ment*
- Prefixes and suffixes can be similar to syllables - better for unit selection (discussed later)
- Works with information from TA, rules and dictionary

Speech synthesis - PhA - Letter-to-Sound

- Convert letters(graphemes) to sounds(phonemes) - phonetic transcription
- Phonemes are units for spoken language representation (can be converted to other units in Speech Synthesizer part)
- Works with transcription rules and dictionary (pronunciation)

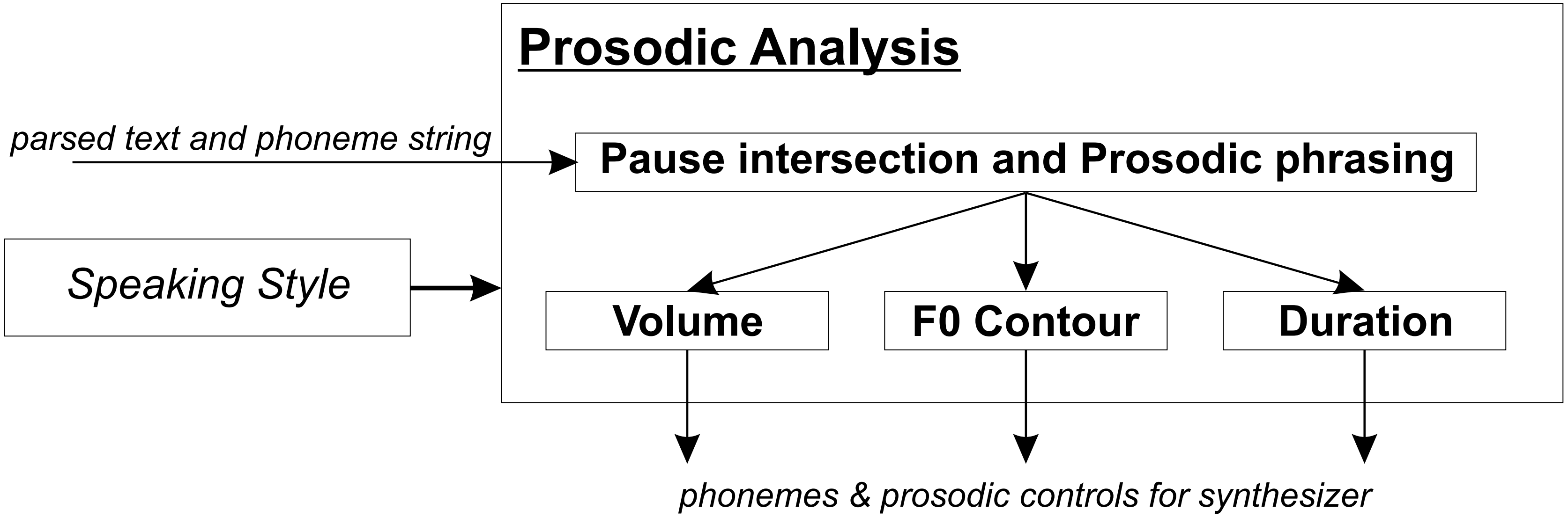
Speech synthesis - PhA - Letter-to-Sound

Example of few transcription rules for Czech language (A$\rightarrow$B/C?D means *A* rewrite to *B* if left context matches with *C* and right with *D*)

Basic rules	ch $\rightarrow \chi$ / ?	q $\rightarrow$ kv / ?	ě $\rightarrow$ je / $\langle$b,p,f,v$\rangle$?
d $\rightarrow$ ď / ?$\langle$i,í$\rangle$	x $\rightarrow$ gz / ?$\langle$ZPS,JS$\rangle$	ex $\rightarrow$ egz / $\sim$? SA	x $\rightarrow$ ks / ? $\langle$NPS,$\sim\rangle$
Spodoba znělosti	NPS1 $\rightarrow$ $\neg$NPS1 / ? ZPS	v $\rightarrow$ f / ? NPS	h $\rightarrow \chi$ / s ?
Spodoba koartikulační	ts $\rightarrow$ c / ?	dš $\rightarrow$ č / ?	n $\rightarrow \eta$ / ? $\langle$k,g$\rangle$

$\sim$ is space, NPS is non-voiced, see Psutka's book !

Speech synthesis - Prosodic analysis



Prosodic Analysis block scheme

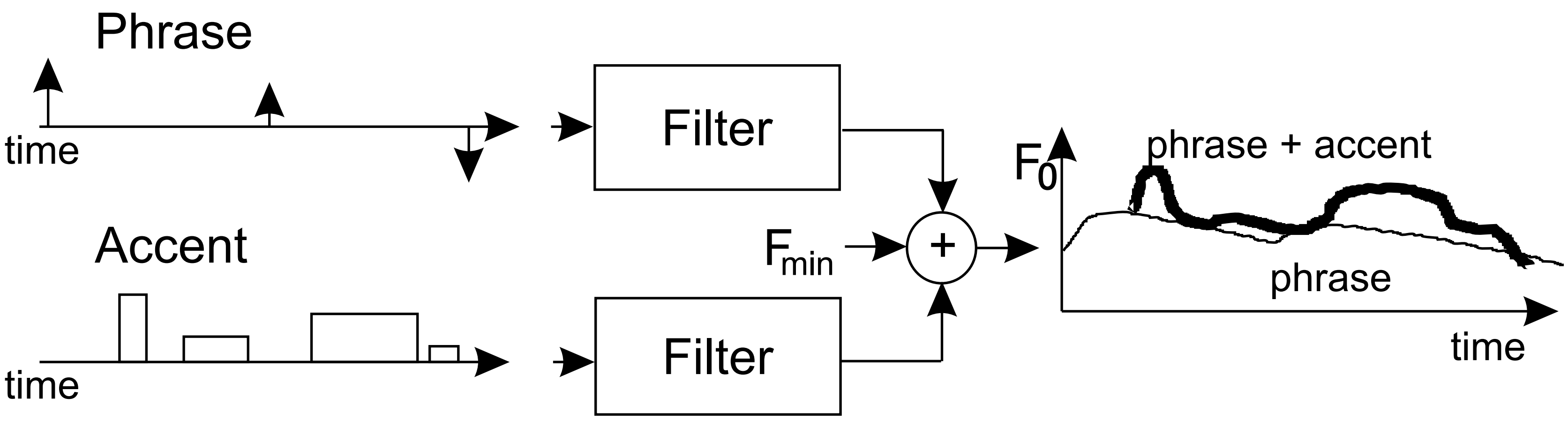
Speech synthesis - Prosodic Analysis

- Prosodic Analysis (*PrA*) is the third block in TTS system
- For simple TTS systems can be omitted
- Adds to the input phoneme string commands for Speech Synthesizer for prosodic modifications
  - F_0: melody
  - Volume: stress (emphasis)
  - Duration: pauses, speech rate
- Can work with database of prosodic samples, different speaking styles

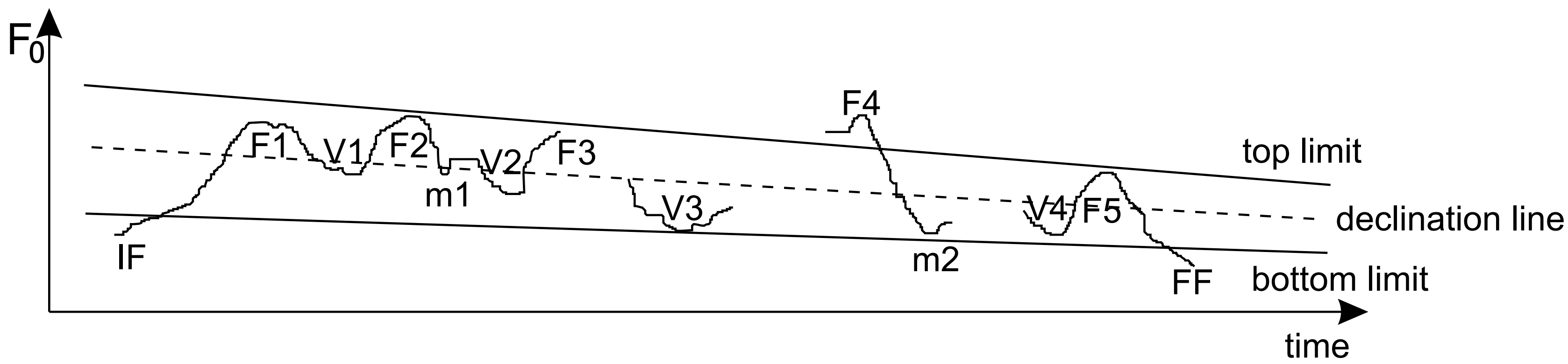
Speech synthesis - PrA - Methods

- Usually generates important points of prosody (from TeA and PhA), then F_0 contour is generated from it.
- Many different methods, no uniform approach
- Acoustic methods
  - Fujisaki method: uses filtering of discrete events
  - Acoustical stylization method: uses interpolation of important parts in pitch contour (hill, valleys)
- Perception methods - theory of human perception of prosody
  - finite set of prosodic segments, concatenation of the best segment
- Some methods are corpus based $\Rightarrow$ database of pitch contours and their concatenation
- Others are parametric, generate discrete events or string of symbols

Speech synthesis - PrA - Methods

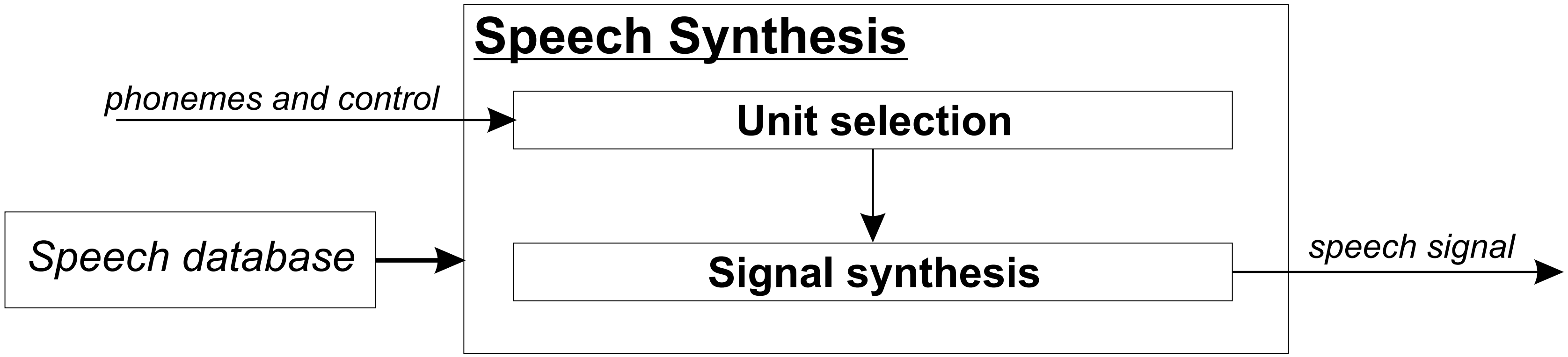


Fujisaki's pitch contour generator block scheme



Acoustical stylization method example

Speech synthesis - Speech Synthesis



Speech Synthesis block scheme

Speech synthesis - Speech Synthesis

- Speech Synthesis (*SpS*) is the last block in TTS system
- This block generates speech signal from given string of phonemes and control commands

Speech synthesis - SpS - Unit Selection

- Selection of the best units (phonemes, diphones, variable length units, . . .)
- Great influence to generated speech quality
- Tries to minimize number of concatenations (if it is possible ⇒ variable length units)
- Tries to select best unit (minimal transition costs)
- Works with index of the speech database (fast searching)

Speech synthesis - SpS - Unit Selection

- Quality of produced speech depends on number of concatenations (in concatenative TTS system). Problem with coarticulation between phonemes.
- Larger vocabulary ⇒ more units, smaller units ⇒ less units but more concatenations ⇒ compromise needed (each concatenation can cause discontinuity in signal and spectrum).

Unit length	Unit type	#Units	Quality
Short	Alophone	$\sim$ 300	Low
	Phoneme	30-40	
	Diphone	$\sim$ 1500	
	Triphone	$\sim$ 30K	
	Demisyllable	$\sim$ 2000	
	Syllable	$\sim$ 11K	
	Word	100K-1.5M	
	Phrases	∞	
Long	Sentences	∞	High

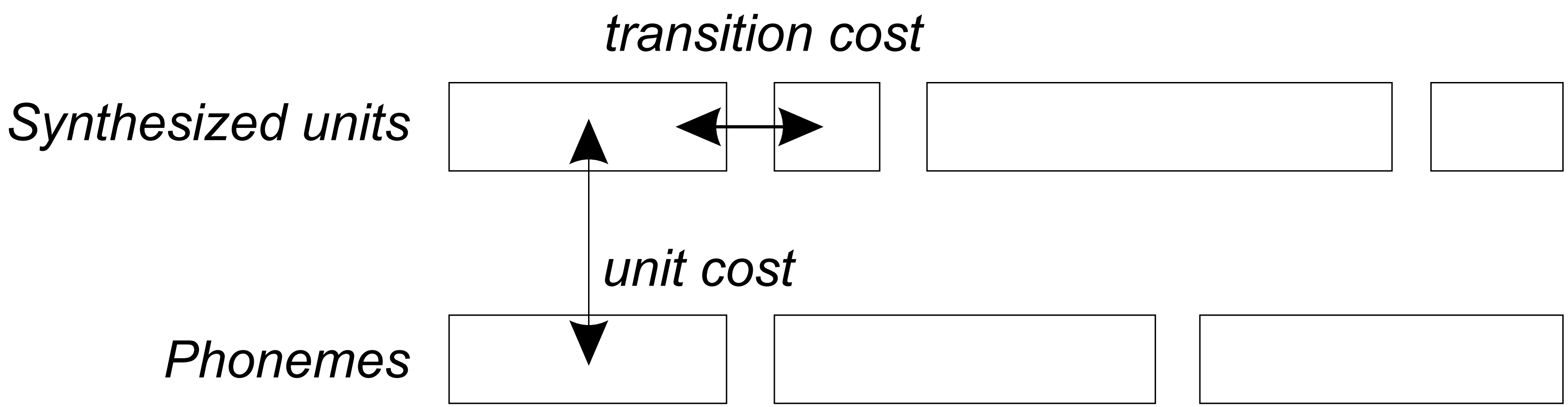
Speech synthesis - SpS - Unit Selection

- Phoneme is context independent letter like unit
- Alophone is context dependent phoneme
- Diphone is unit from center of one phoneme to center of the other phoneme
- Triphone like diphone but we jump over one phoneme
- Syllable is the smallest compact unit in the speech
- Demisyllable like diphone

Speech synthesis - SpS - Unit Selection

- Units of the same kind were used in history (inventory of phonemes)
- Actually TTS systems with variable units length. The best way:
  - Large database of speech which is completely parameterized (PSOLA, DSM)
  - Database is phonetically and prosodically labelled ⇒ index file created
  - The best units (pieces) selected in synthesis
  - Criteria for selection: long units, less concatenations, matching units (low unit and transition cost in concatenation), if searched unit does not appear in database find the closest one
- Compare this approach with database systems

Speech synthesis - SpS - Unit Selection



Example of speech units and unit and transition cost

Speech synthesis - SpS - Signal synthesis

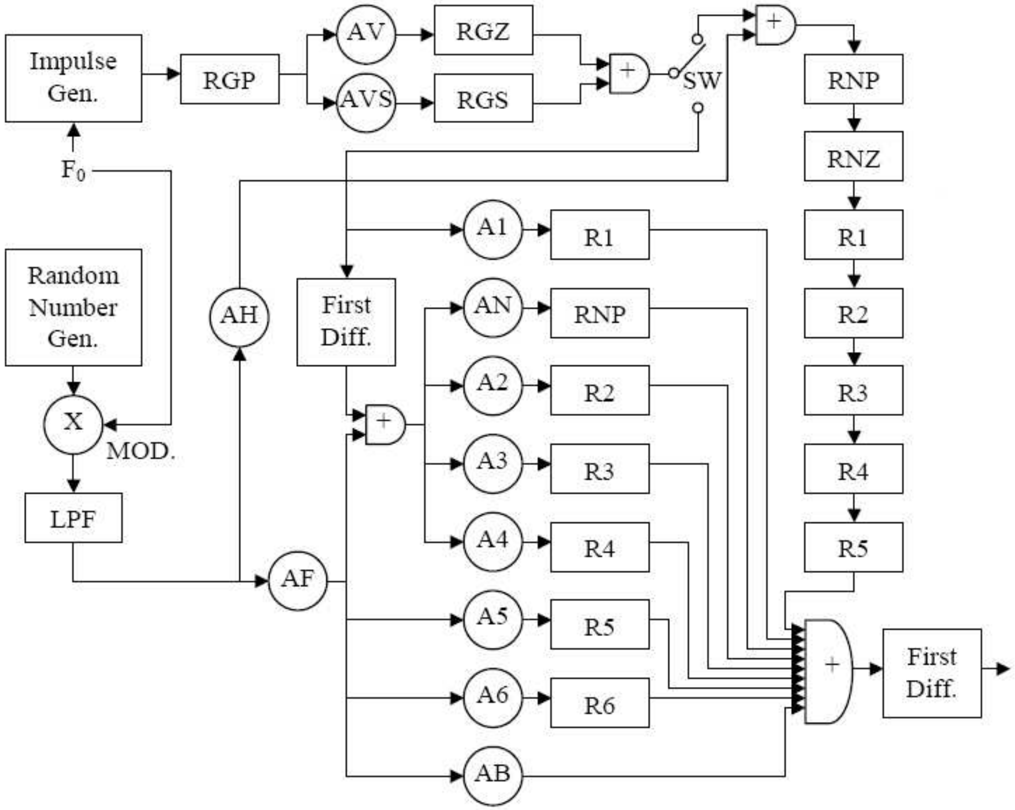
- Synthesizes the speech signal
- Gets units (usually parameterized) and concatenates them together
- Modifies prosody of parameterized speech given by control commands
- Synthesize speech
  - Formant synthesis (older)
  - Concatenative synthesis (newer)

Speech synthesis - SpS - Signal synthesis

Formant signal synthesis

- A complicated filter with impulse and random number generator as excitation.
- Signal from these generators passes through filter (about 40 parameters) and output is speech
- Parameters for the filter are stored in database
- Advantage: constant speech quality, low memory and computation cost (developed in 80's)
- Disadvantages: low speech quality

Speech synthesis - Signal Synthesis



Formant speech synthesizer by Klatt

Speech synthesis - SpS - Signal synthesis

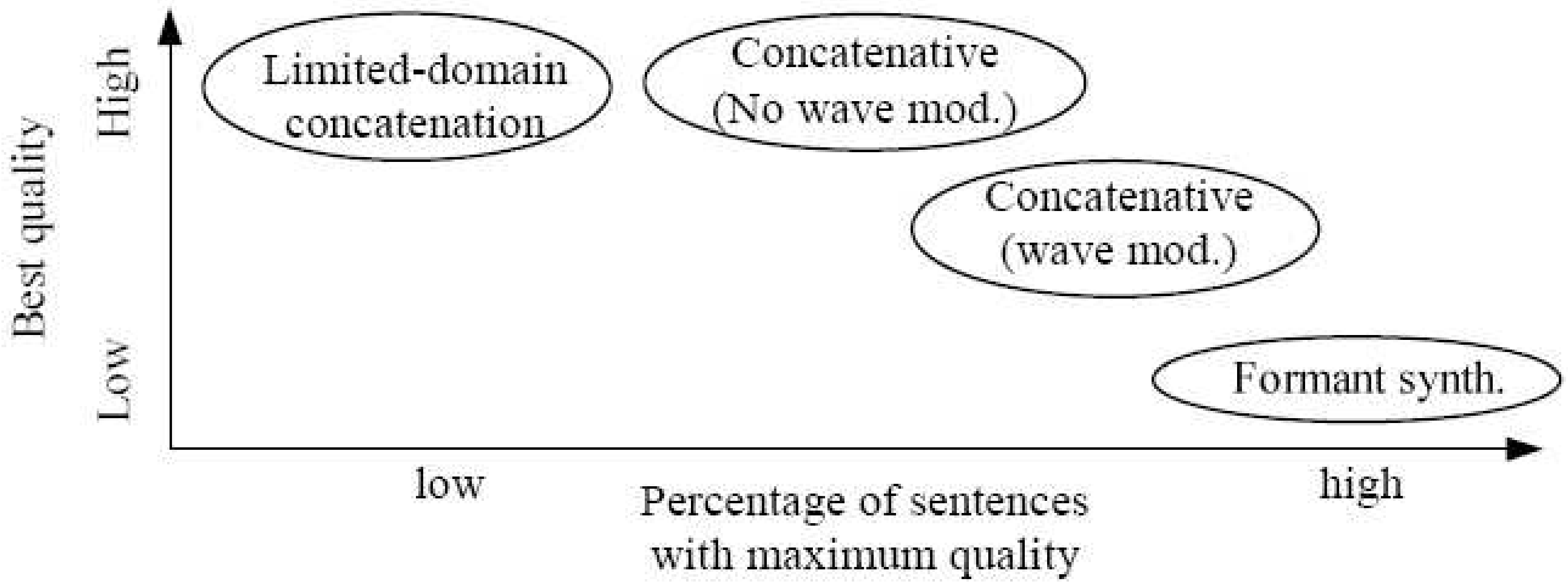
Concatenative signal synthesis

- Original speech units (said by human)
- From "simple" concatenation (earlier) to "sophisticated" concatenation with many modifications (nowadays)
- Units are stored in database
- Advantages: high quality speech, low computation consumption
- Disadvantages: not constant speech quality, large memory consumption
- note: Advantages and and Disadvantages depend on used method

Quality of signal synthesis methods

- Little domain concatenation: no waveform modifications, cannot synthesize arbitrary text (high quality (varying) for very small vocabulary)
- Concatenative with no waveform modification: can synthesize arbitrary text, prosody can be synthesized by selecting suitable units (quite high quality (varying) for arbitrary text)
- Concatenative with waveform modification: parametric prosodic modifications, (medium quality (constant) for arbitrary text)
- Formant synthesis: can synthesize arbitrary text with prosody (low quality (constant) for arbitrary text)

Speech synthesis - Signal Synthesis



Quality of different speech synthesis methods

Speech synthesis - SpS - Signal synthesis

Methods of unit concatenation

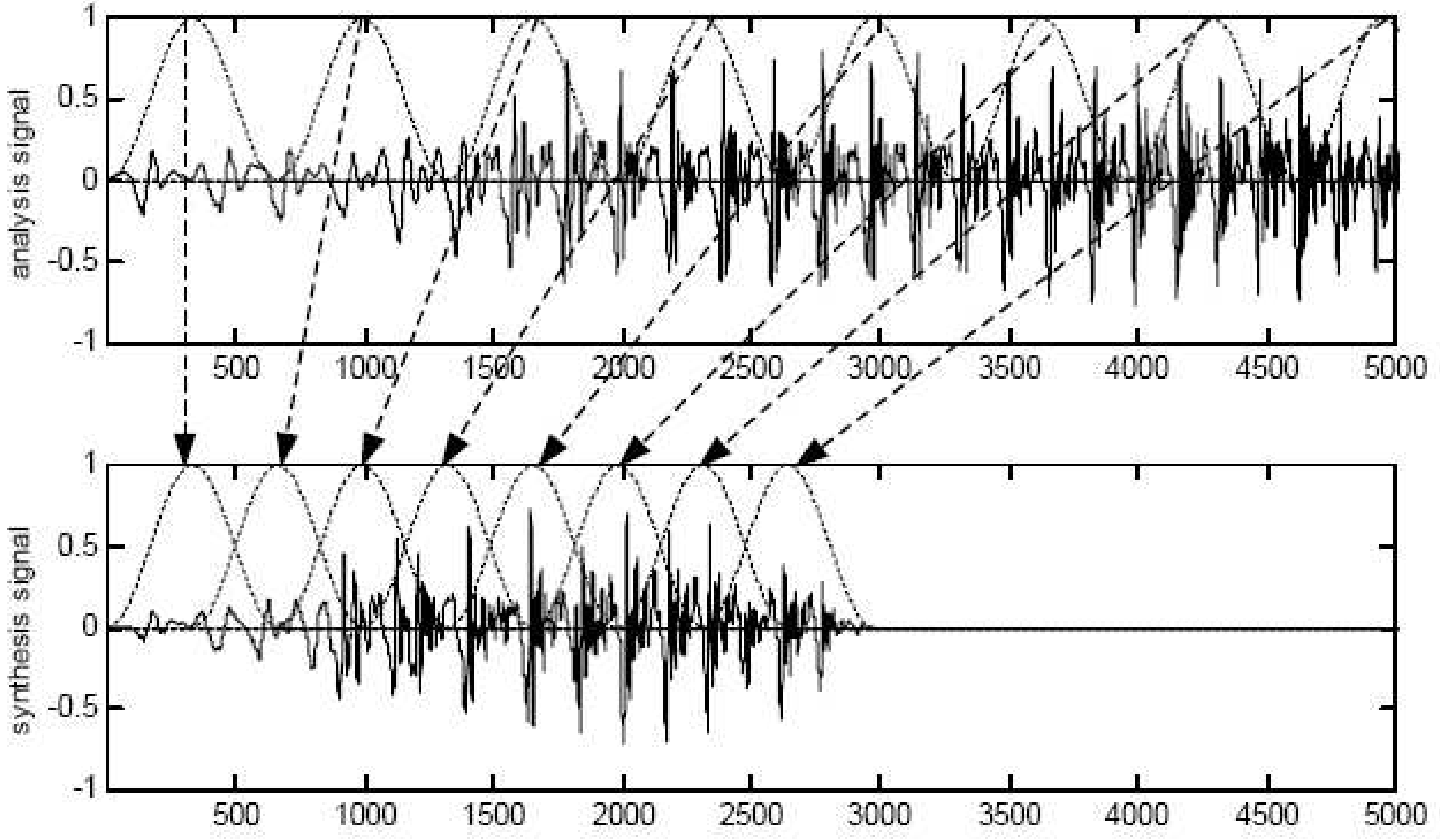
- Pure concatenation: used with phrases and word like units, simple and the worst quality (concatenation of isolated words ⇒ good quality), no prosodic modifications
- Overlapping: used with syllable and longer units, simple and the worst quality (little bit better then pure concat.), problem with pitch synchronization
- Pitch synchronous: usually based on more sophisticated methods for unit parametrization (PSOLA, DSM), uses vector interpolation etc., used with all types of units
- Others: you can develop some ⇒ BP, RP, DP

Speech synthesis - SpS - Signal synthesis

Methods of prosodic modifications

- OLA (OverLap and Add): nonparametric, simple and the worst quality, can be used only for time-scale modifications (faster/slower speech), no pitch synchronization ⇒ spectral discontinuities
- PSOLA (Pitch Synchronous OLA): nonparametric, more complicated and medium quality, can be use to time-scale and pitch-scale modifications, problems with voiced fricatives ('z'), in concatenation amplitude, phase, pitch mishmashes
- DSM (Deterministic and Stochastic Model): parametric, very complicated and the highest quality, speech is parameterized (harmonic and noise components) and then resynthesized with given speech rate and fundamental frequency

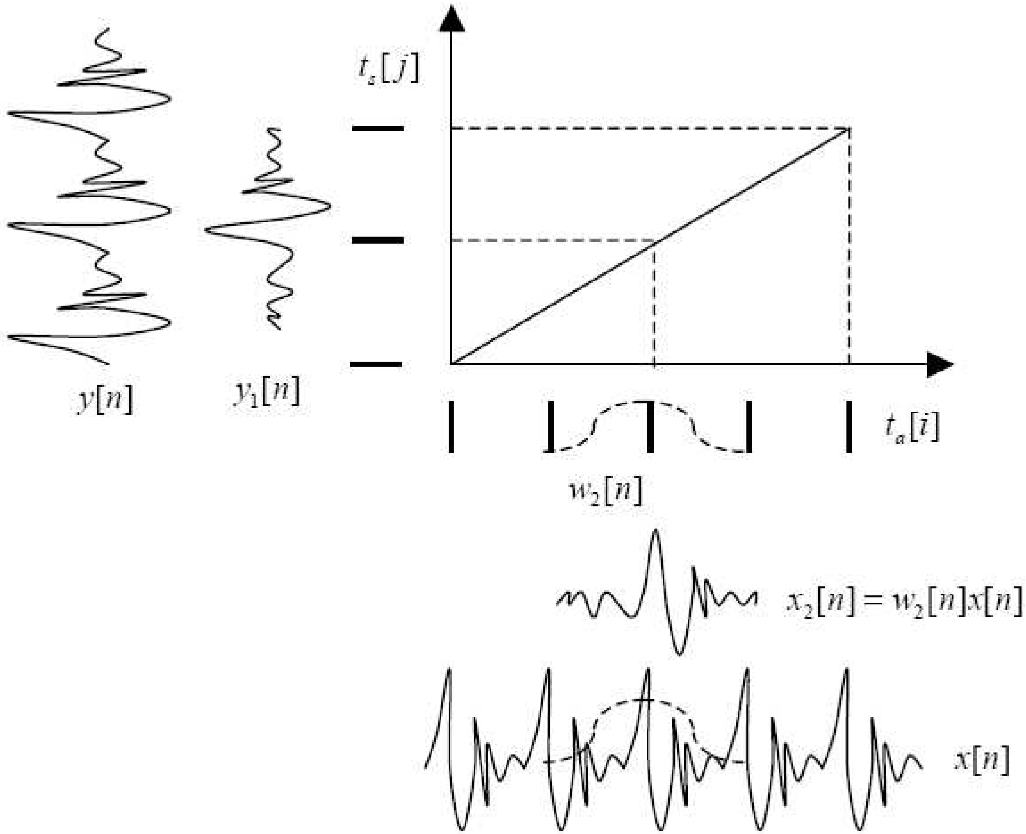
Speech synthesis - SpS - Signal Synthesis



Example of OLA technique

Speech synthesis - SpS - Signal synthesis

Core of prosodic modifications (pitch and rate) is on creation of synthesized "pitch marks" and then on mapping original pitch periods onto synthesized pitch marks.

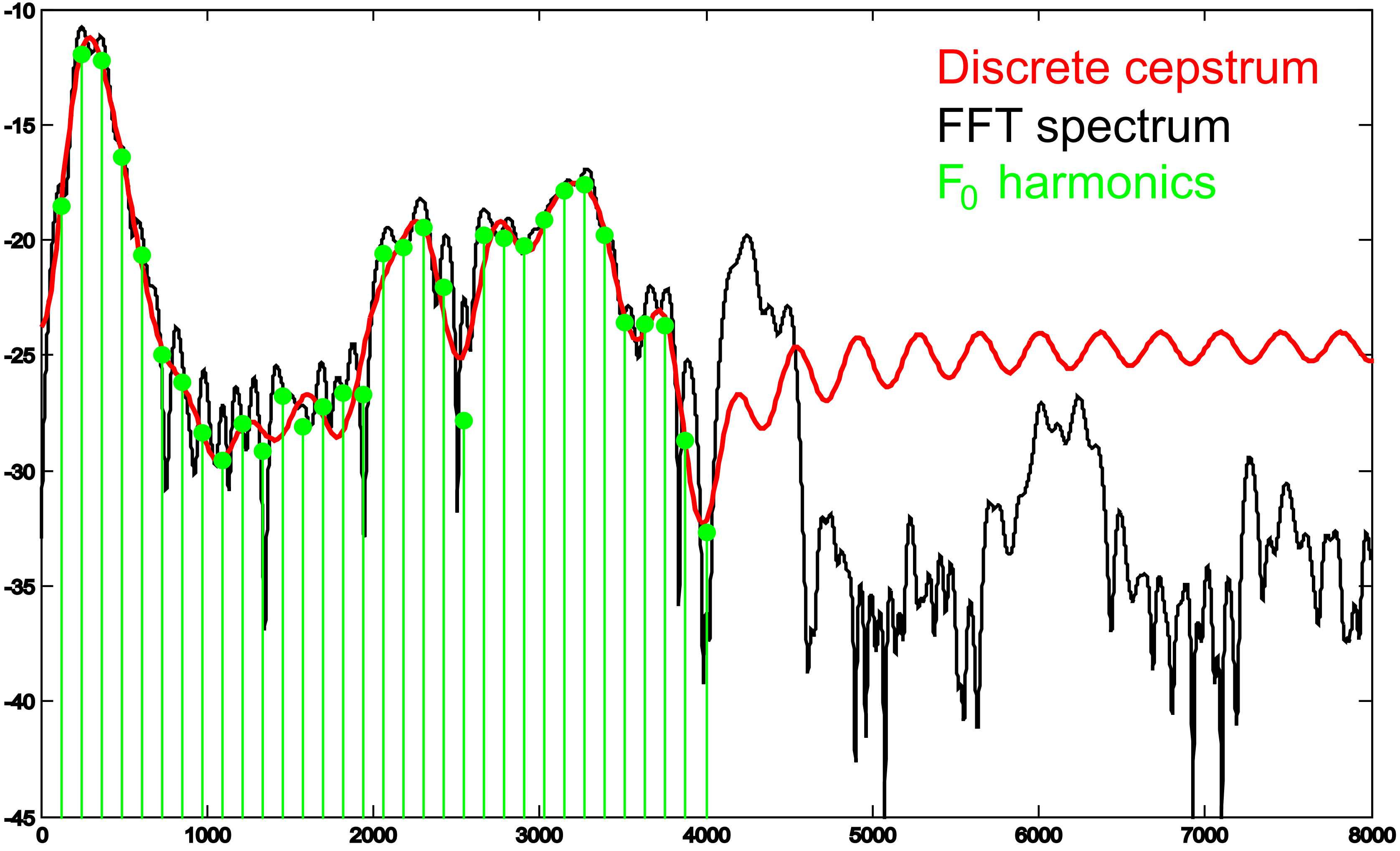


Example of PSOLA technique, duration shortened by 40% and pitch period increased by 60%

Speech synthesis - SpS - Signal synthesis

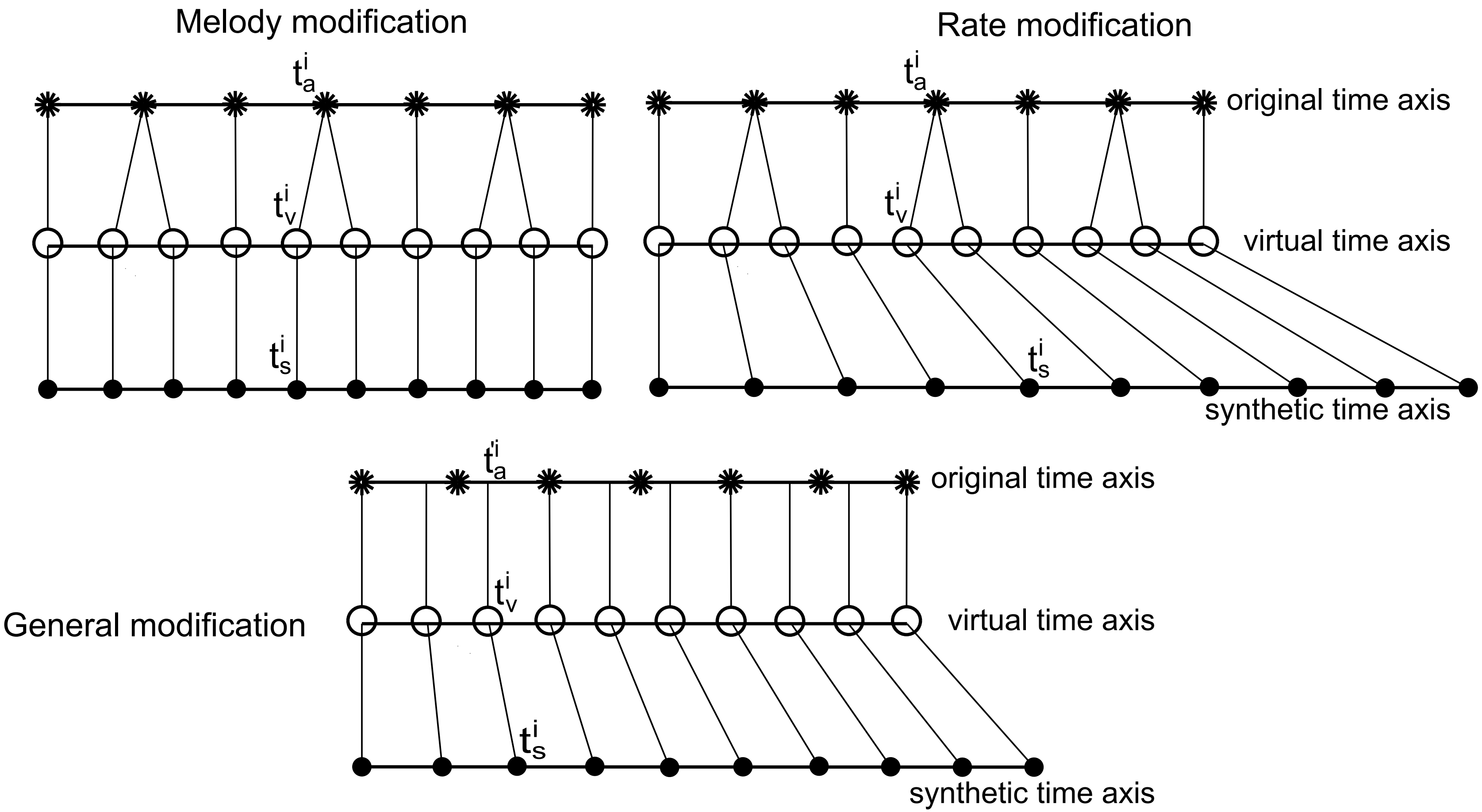
Harmonic and Noise Model

- One type of DSM is Harmonic and Noise Model, deterministic component is modelled by harmonic model and stochastic component is modelled by noise generator
- Harmonic model parameterizes only fundamental frequency harmonics in range $\langle F_0, \sim 4kHz \rangle$, it takes parameters of envelope of the harmonics magnitude
- Noise part: substraction of original signal and harmonic component, parameterized by spectral envelope
- Synthesis of the harmonic component is done by sampling of the magnitude envelope on given F_0 and synthesis of the harmonics.
- Synthesis of the noise component is done by filtering white noise
- Harmonic and noise components are finally summed together



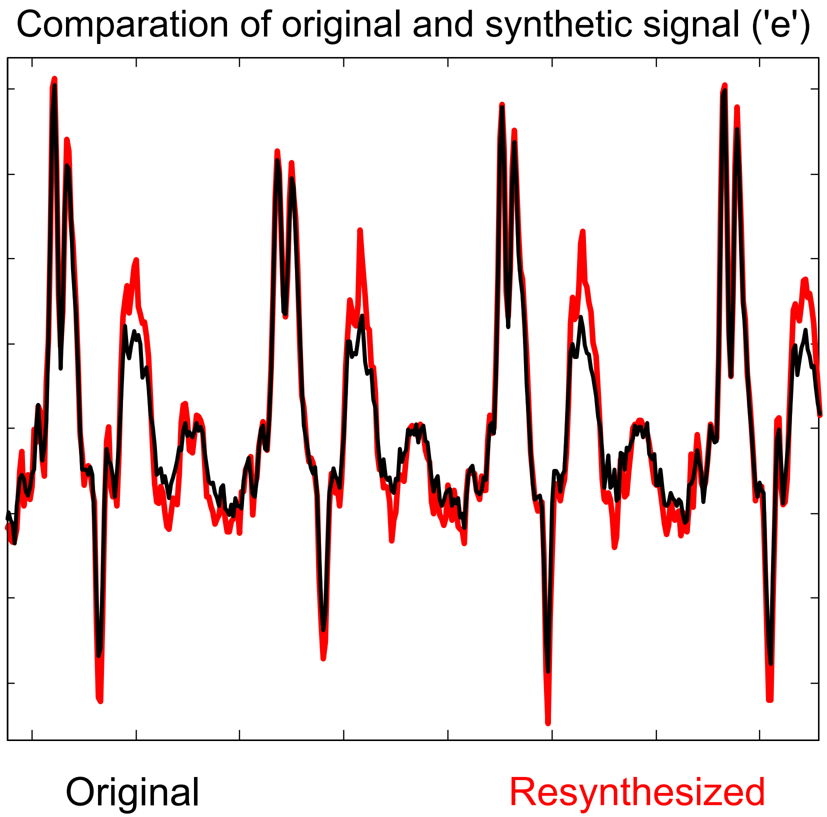
Example of the one frame of HNM harmonic component parametrization.

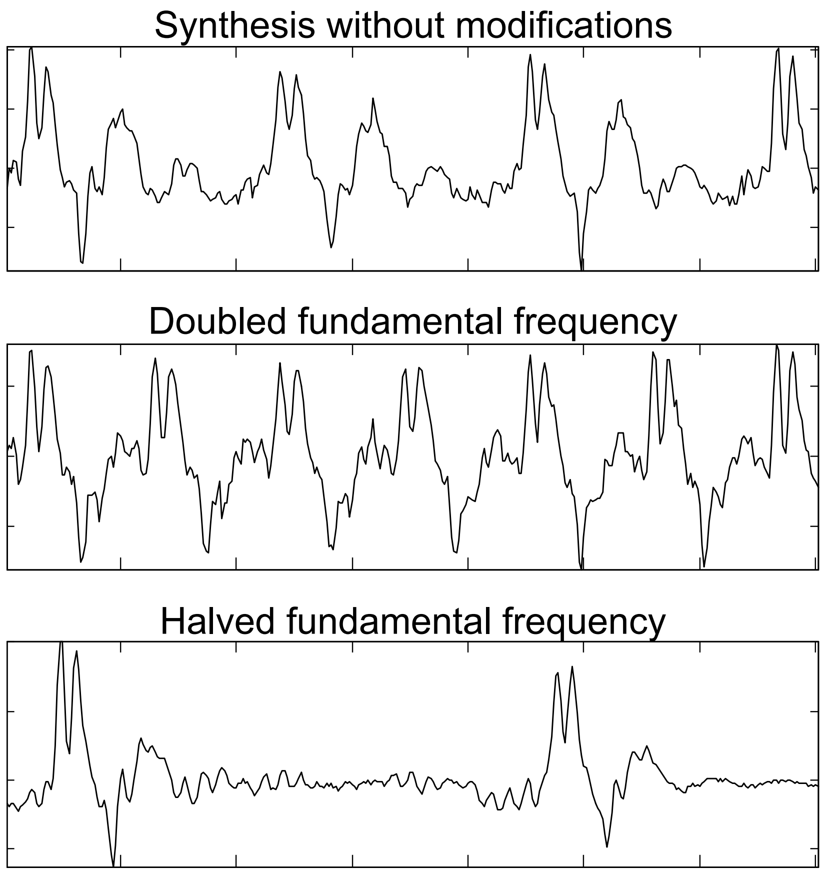
Speech synthesis - SpS - Signal synthesis



Example of DSM mapping technique, advantage against PSOLA is, that we can compute parameters of new pitch period by interpolation from the neighbouring ones

Speech synthesis - SpS - Signal synthesis





Example of HNM (Harmonic and Noise Model - version of DSM) synthesized signal. Original signal, resynthesized (no changes), doubled and halved fundamental frequency

Speech synthesis - TTS - Evaluation

- Tests of functionality (structure), speech quality (intelligibility)
- Checking functionality of the system components (TeA, PhA, PrA, SpS)
- Intelligibility test with group of listeners, triples of words which differs only in one phoneme (letter) and meaningless sentences are generated, listeners must write what they listened (or check one possibility)
- Phonetically balanced sentences and normal text are generated, listeners answers how much effort does the understanding "cost".

Speech synthesis - TTS systemes

- EPOS (epos.ure.cas.cz/): One of the best Czech (and Slovak) TTS system, source for research is free, contains prosody, LPC synhesis
- Speechtech (www.speechtech.cz/): Czech commercial TTS, contains prosody, HNM synthesis
- Igor's TTS (www.fit.vutbr.cz/∼szoke/speech/TTS/): TTS system for Czech developed as RP+SP+DP, contains prosody, HNM synthesis, units with variable length, are you interested? You can continue in developing (BP, RP, DP).
- Festival (www.cstr.ed.ac.uk/projects/festival/): Multi-language TTS system, source for research is free, contains prosody, diphone units (general units are in developing progress), and much more (it is huge universal system), are you interested? You can try to 'hack it' and develop support for Czech (BP, RP, DP).

Play the examples!