

rVoice Studio and ActivePrompts

Peter Rutten and David Talkin

Rhetorical Systems Ltd, Edinburgh

`peter.rutten@rhetorical.com`

1. ActivePrompts

ActivePrompts are a new technology from Rhetorical, designed to offer a quicker and cheaper alternative to using voice talents and recording studios for the creation of an application-specific prompt library.

Most speech user interfaces optimize the quality of their speech output by using pre-recorded prompts. In these applications, TTS is often considered to be a fall-back technology to generate speech for the variable content of messages. Although pre-recorded prompts optimize the quality of the static part of messages, they are expensive to create, depend on the availability of a speaker and recording studio, and can be difficult to combine with TTS.

As an alternative to pre-recorded prompts, Rhetorical Systems has developed the technology to create prompts by using the rVoice TTS engine. This technology exploits the fact that rVoice contains a large unit selection database, that can be used to synthesize a large number of alternative versions of a same sentence. Given an appropriate interface to rVoice, it is generally possible to find the sequence of speech units that result in synthetic speech that can pass as a pre-recorded prompt.

We coined the name 'ActivePrompt' for prompts that are created from the rVoice speech unit database. They are different from recorded prompts in the way they can be used in the application. Instead of being a speech file, that is concatenated with speech that is synthesized, they can be stored as a sequence of speech unit identifiers that should be used to synthesize a particular text in a particular context.

2. Pros and cons of ActivePrompts

ActivePrompts have many advantages over normal pre-recorded prompts, including:

- ActivePrompts are quick and easy to produce, and don't require any use of voice talents or recording studios.
- ActivePrompts can be joined seamless with other ActivePrompts and TTS, eliminating the jarring concate-

nation effects often found when joining pre-recorded prompts.

- Using ActivePrompts and TTS ensures a single, consistent voice is used throughout the application. New ActivePrompts can be added to an existing deployment with the assurance that the voice quality will perfectly match the existing ActivePrompts and TTS.

A disadvantage of ActivePrompts is that they have to be created within the limitations of the speech database that is used by the TTS engine. In practice, this means that the speaking style of the prompts must be similar to the speaking style used in the TTS database - which is probably sufficient for most applications.

3. Creating ActivePrompts

ActivePrompts are created in rVoice Studio, which is a graphical user interface that supports the management of ActivePrompts, domain-specific abbreviation tables and pronunciation lexicons.

The acoustic editor of rVoice Studio, depicted in Fig. 1, is the interface to an interactive unit selection algorithm [1] that is running on the TTS server. This user interface allows a developer to describe what is wrong about the synthesized utterance, by selecting words/phones that sound bad, and by setting bias values for the $dur/f0$ of these selected phones. This information is translated to low level parameters that drive two extensions to the basic unit selection algorithm.

A first extension is a bias pruning step, that is able to remove rejected speech units, or fix accepted units, for each target speech unit in the utterance. A second extension is formed by a set of bias cost functions, that essentially allow the unit selection algorithm to take a bad example as a reference, penalize units that are "worse" than the reference, and reward units that are "better" than the reference.

Fig. 2 shows an example, where the word 'not' in the sentence 'somebody is not available' is being modified by interactive unit selection. Only the speech units that make up the word 'not' will change.

When the developer is satisfied with the result, any continuous sequence of words in the sentence can be selected

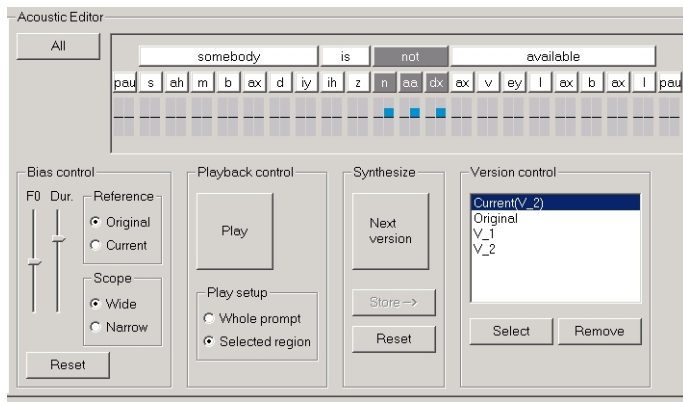


Fig. 1. Acoustic editor.

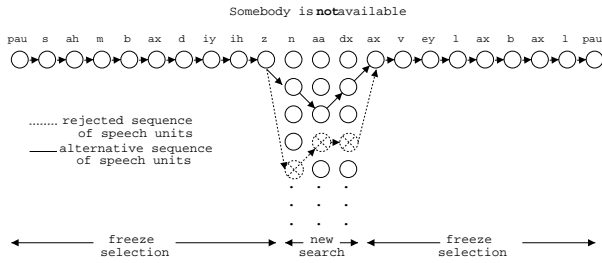


Fig. 2. Creating ActivePrompts - sculpt the word 'not'.

to create the ActivePrompt. A number of prompt properties have to be specified before the prompt is saved to file. These properties control the insertion of prompts in TTS.

4. Using ActivePrompts

The ActivePrompts are stored as special instructions to the text-to-speech engine in a project file. The project file is loaded into the server, and, when activated, tells the rVoice server how to recreate that prompt so it sounds exactly like it sounded when first created within rVoice Studio.

There are essentially 2 ways of invoking ActivePrompts in synthesis. The first way is by specifying the name of the prompt in an XML tag. In this case, rVoice will replace the XML tag with the text stored in the ActivePrompt, and use the sequence of units specified by the ActivePrompt to synthesize this part of the sentence.

A second way to use ActivePrompts is by giving them the 'automatic insertion' property. For this type of ActivePrompts, rVoice will scan the input text for matches in the ActivePrompt library. If a match is found, the ActivePrompt is automatically used instead of default synthesis.

Any text that surrounds a prompt will be synthesized by pure TTS, as depicted in Fig. 3. This example shows how TTS ('John Smith') is combined with an ActivePrompt ('is not available') during unit selection, to create the sentence

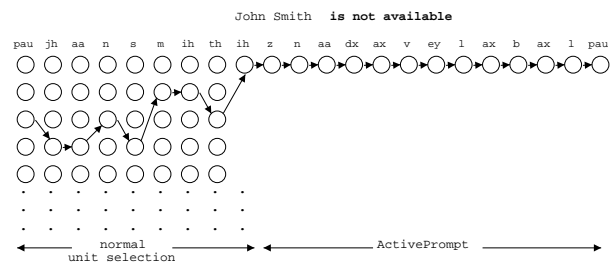


Fig. 3. Integrating ActivePrompts in TTS.

'John Smith is not available'.

5. Current work

We are focusing our current research work on improving the acoustic editor interface, and the interactive unit selection algorithm that underlies the creation of ActivePrompts. Subjects of investigation are:

- How can we provide better feedback to the developer? Although the number of possible versions we can create for an utterance is astronomically large, we don't have an unlimited freedom to manipulate the prosody of a sentence. It should be made clear to the developer where the limitations of the system lie.
- How can we add more intuitive, high-level control parameters to the acoustic editor? Instead of specifying dur/f0 biases on a phone-by-phone basis we want to do things like: put the focus on a certain word in the sentence, add/remove accent from a syllable, add intonation rise, add fall.
- Can we do example driven sculpting? The idea is to derive sculpting parameters from aligning rVoice synthesis with speech recorded from the developer. An acoustic analysis of the target speech signal provides the bias settings for the algorithm.
- Is it useful to integrate DSP to manipulate the speech signal, and use this in an ActivePrompt? For this purpose the interface can be extended with a display of the speech signal, the f0 track and the phonetic segmentation.

6. References

[1] P. Rutten and J. Fackrell, "The application of interactive speech unit selection in TTS systems," in *Eurospeech*, Geneva, Switzerland, 2003, pp. 285–288.