OpenSource SpeechSynth

Final project for class *Multilingual Computing*, Spring 2000.
Original idea by Deb Roy, implemented by Stefan Marti
http://hyperdrive.media.mit.edu/cgi-bin/stefanm/synth

Main idea

The main idea of this project was to create a text-to-speech synthesizer for minority languages, for which no commercial speech synthesizer software is available yet, e.g., Swiss German. Other applications include societies with a high analphabetism rate. Written text from the Web can be synthesized and made audible after a few people have trained the system through the Web.

Open Source

This project is based on the principle of open source software: many people contribute a little, so that everybody can profit from the accumulated resources. For the *OpenSource SpeechSynth*, the most likely scenario is that some people train the system, and everybody can use it. To encourage people to participate, open source projects ideally have some built-in "social credit" mechanisms. (In the case of the OpenSource SpeechSynth, such mechanisms are not in place yet. An obvious option would be to credit the people who have contributed the most.)

Word based synthesis

The speech synthesizing method used in this project is word based (e.g., Campbell 1998, Olive 1980), which means that the smallest sound units are words. Sentences are obtained by waveform concatenation of word sounds, bypassing computationally intensive digital sound processing. Obviously, word based synthesizer need a large vocabulary of sound files to generate arbitrary text, or otherwise can cover only specific well defined domains with a somewhat limited vocabulary, e.g., daily news. However, due to the combination with the open source idea where a lot of people contribute a few words and generate a large corpus, even general speech synthesis is thinkable.

Web based approach

Open source projects will succeed most likely if they are based on ubiquitous platforms like the Web. Since a lot of non-early-adopters should be able to profit from such a project, it has to be based on today's standard Web technology. This excludes the use of rare plug-ins like Java Media and the latest Shockwave.

Scenario

Here is a typical scenario: Since no rare plug-ins are possible, the user records sounds of words, e.g., with Microsoft Soundrecorder, and stores them locally in a file. On the *OpenSource SpeechSynth* Web interface, she can upload these prerecorded sound files. The more sound files are uploaded, the more universal the synthesizer gets. The user then can submit a text and gets back an audio file containing a synthesized voice reading that text.

Performance

Because ideally a lot of people with different computing infrastructure (especially different microphones) are uploading sound files, the synthesized text will probably not sound smooth, but still understandable. An option would be to accept several sound files from different users per word, and then let the system choose the ones that match best in pitch and voice character. (Such a feature is not implemented yet.)

Portability

An important aspect of the OpenSource SpeechSynth is its portability: the CGI script is small and can run on any simple Web server. The only resource needed is hard drive space to store the sound files. The required CPU resources are currently relatively low since a special concatenation script was written that does not require to load the sound files that are concatenated into memory completely.

Scalability I: multilingual

Another important aspect of the current system is its scalability: Although the current prototype (version 0.2) can handle only English, the next software update will be able to handle an unlimited amount of languages. About 80 can be selected already, and new languages can be added with a mouse click.

Scalability II: adding more digital sound processing

The project is also scalable concerning the synthesizing algorithm. Currently, the sound files are concatenated, with the option to truncate every word sound at the beginning and the end for a fixed amount of samples (four options currently). This improves the intelligibility of the synthesized text because it takes in account that most of the uploaded samples have leading and trailing silent parts. Like that, the system minimizes the pauses between the words. However, it would be easy to implement a more sophisticated silence detection mechanism (and it will be implemented soon.) Other DSP could be added later easily.

Wish list

Future improvements include (partly mentioned above):

- Improve the security of the system by locking all file read and write operations (currently, only part of the file operations are locked). Furthermore, log all user activity.
- Implement a silence detection algorithm to truncate poorly edited sound files.
- Build in filtering mechanisms to prevent from "unfriendly" uploads: filtering the uploaded vocabulary (taking out certain four-letter words), checking the quality of uploaded sound files, restricting the length of uploaded sound files, etc.
- Implement software versions with different character font sets.
- Implement a "social credit" mechanism.

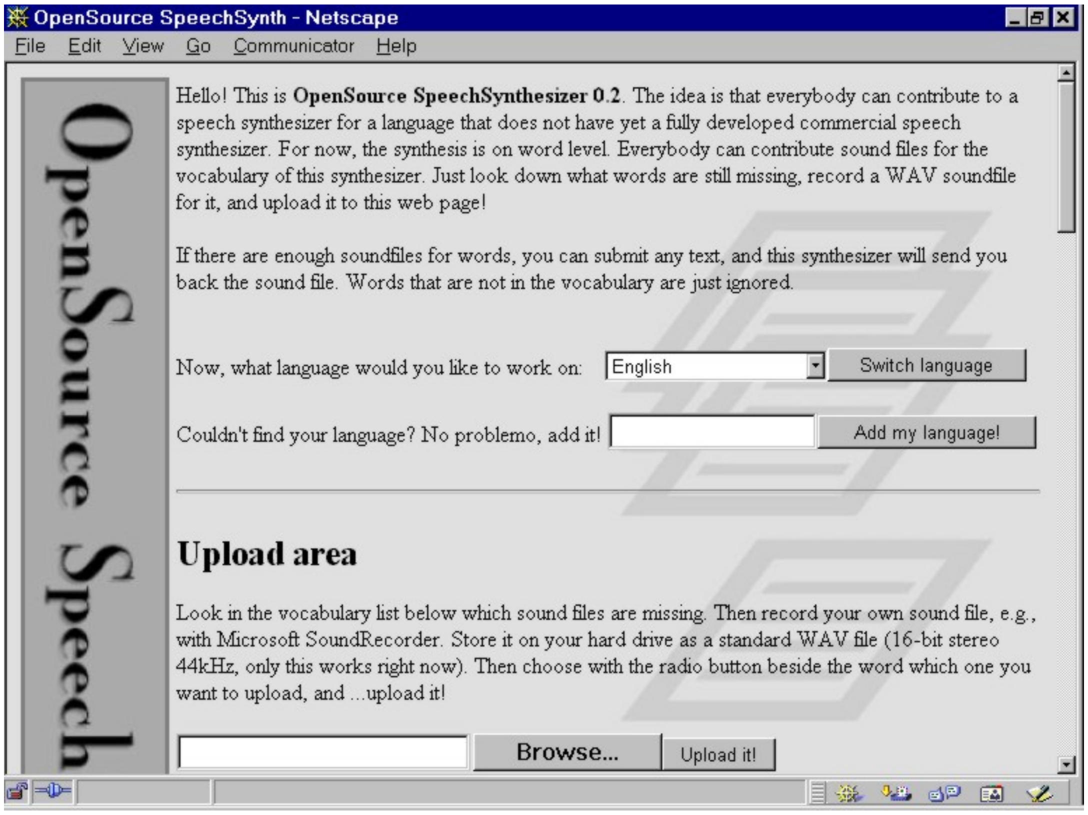
- Enable multiple sound files per word. Matching the possible words for a synthesized sentence so that the most similar sound files are used, e.g., by distinguishing male from female sound samples.
- Support more file formats, and enable streaming delivery of audio back to the user, e.g., with RealAudio.
- Add the option of a Web based recording of sound with Java Media or Shockwave for the power user. This would simplify the recording process, bypassing file format problems. However, it is not common in today's Web browsers since it requires access to local computer resources (microphone, loudspeaker, file reading and writing) which today is regarded as a security problem.
- Allow the use of other speech sound sources like annotated voice chats to increase the vocabulary (speculative).
- Infer unknown words from existing vocabulary by splitting words into phoneme like parts (highly speculative).

Very Basic References

Campbell, N. (1998). Stages of processing n CHATR speech synthesis. *IEICE Technical Report, SP98-84*, pp.47-54, Nov. 1998.

Olive, J.P. (1980). A scheme for concatenating units for speech synthesis. *Proceedings of IEEE-ICASSP80*, 568-571.

Screenshots



Hello! This is **OpenSource SpeechSynthesizer 0.2**. The idea is that everybody can contribute to a speech synthesizer for a language that does not have yet a fully developed commercial speech synthesizer. For now, the synthesis is on word level. Everybody can contribute sound files for the vocabulary of this synthesizer. Just look down what words are still missing, record a WAV soundfile for it, and upload it to this web page!

If there are enough soundfiles for words, you can submit any text, and this synthesizer will send you back the sound file. Words that are not in the vocabulary are just ignored.

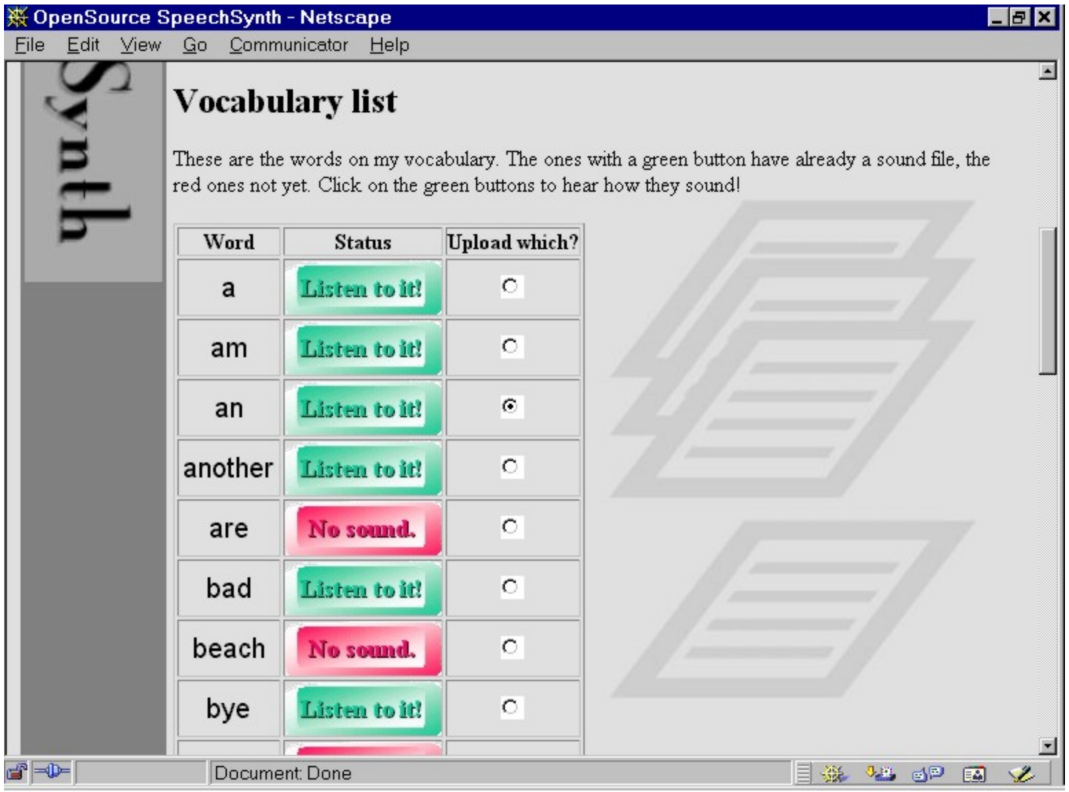
Now, what language would you like to work on: English

Couldn't find your language? No problemo, add it!

Upload area

Look in the vocabulary list below which sound files are missing. Then record your own sound file, e.g., with Microsoft SoundRecorder. Store it on your hard drive as a standard WAV file (16-bit stereo 44kHz, only this works right now). Then choose with the radio button beside the word which one you want to upload, and ...upload it!

Screenshot 1: Upper part of Web page: language selection and upload area.

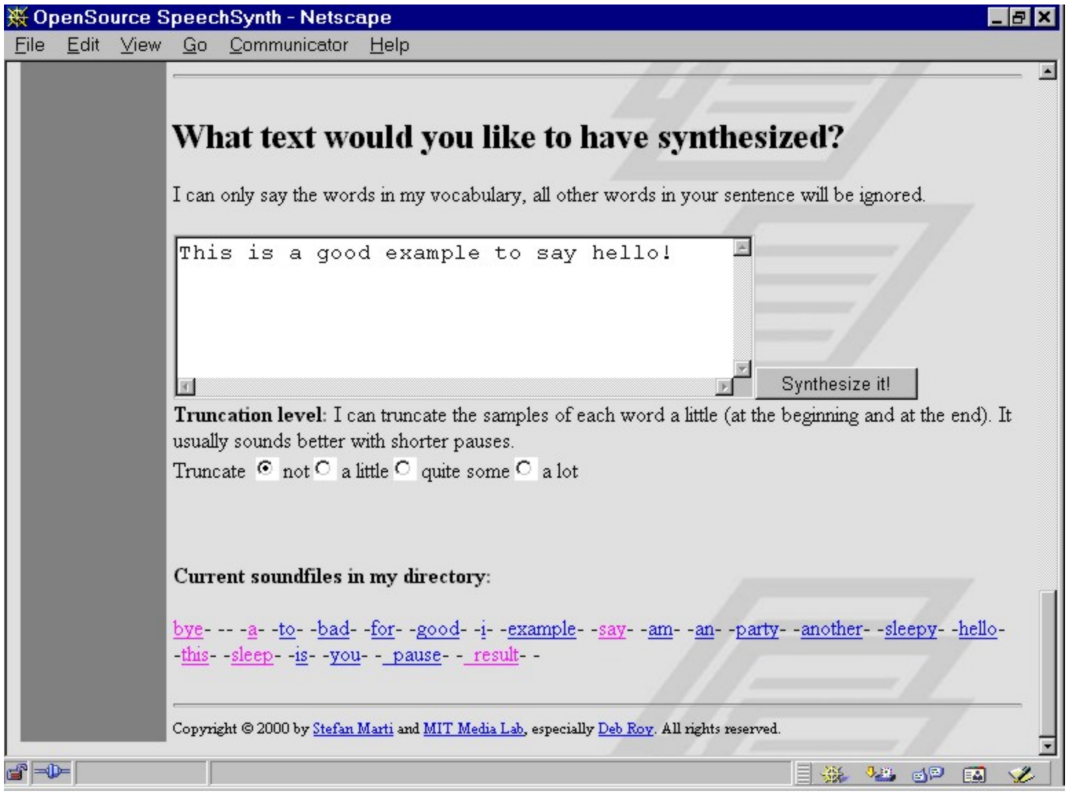

OpenSource SpeechSynth - Netscape
File Edit View Go Communicator Help
Synth
Document: Done

Vocabulary list

These are the words on my vocabulary. The ones with a green button have already a sound file, the red ones not yet. Click on the green buttons to hear how they sound!

Word	Status	Upload which?
a	Listen to it!	○
am	Listen to it!	○
an	Listen to it!	●
another	Listen to it!	○
are	No sound.	○
bad	Listen to it!	○
beach	No sound.	○
bye	Listen to it!	○

Screenshot 2: Middle part: beginning of vocabulary list.


OpenSource SpeechSynth - Netscape
File Edit View Go Communicator Help
Synthesize it!

What text would you like to have synthesized?

I can only say the words in my vocabulary, all other words in your sentence will be ignored.

```
This is a good example to say hello!
```

Truncation level: I can truncate the samples of each word a little (at the beginning and at the end). It usually sounds better with shorter pauses.

Truncate [✓ not] a little quite some a lot

Current soundfiles in my directory:

bye- -- -a- -to- -bad- -for- -good- -i- -example- -say- -am- -an- -party- -another- -sleepy- -hello- -this- -sleep- -is- -you- - pause- - result- -

Copyright © 2000 by Stefan Marti and MIT Media Lab, especially Deb Roy. All rights reserved.

Screenshot 3: Lower part: synthesizer interface with truncation selection.