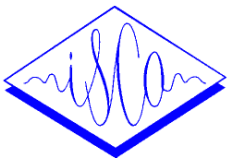

6th International Conference on Spoken Language Processing (ICSLP 2000)
Beijing, China
October 16-20, 2000

WAVESURFER - AN OPEN SOURCE SPEECH TOOL

Kåre Sjölander and Jonas Beskow

{kare|beskow}@speech.kth.se

Centre for Speech Technology, KTH, Drottning Kristinas väg 31, SE-100 44, Sweden

ABSTRACT

In the speech technology research community there is an increasing trend to use open source solutions. We present a new tool in that spirit, WaveSurfer, which has been developed at the Centre for Speech Technology at KTH. It has been designed for tasks such as viewing, editing, and labeling of audio data. WaveSurfer is built around a small core to which most functionality is added in the form of plug-ins. The tool has been designed to work on most common platforms and with the aims that it should be easy to configure and extend. WaveSurfer is provided as open source, under the GPL license with the explicit goal that the speech community jointly will improve and expand its scope and capabilities.

1. INTRODUCTION

The speech research field is dependent on software tools to handle digital recordings of speech. There is a wide array of such tools available, ranging from commercial products to software for download over the Internet with varying kinds of licensing. The capabilities of these software tools also show a wide variation and exhibit a continuous adaptation to the ever-changing needs and interests of the speech community. The open source model [1] is increasingly popular, as exemplified by toolkits such as the CSLU Toolkit [2], the Festival Speech Synthesis System [3], and the ISIP Automatic Speech Recognition Toolkit [4].

The advantages of using open source software for research work are numerous. Frequently, a researcher is faced with a tool that almost does the task at hand, but needs some tweaking to do the right thing. Having access to the source code will allow the researcher, at least in theory, to make the needed modifications. But mere openness is not a guarantee for flexibility. In order for a tool to be truly extensible, it must have well-defined programming interfaces, otherwise extensions and modifications will become hard to develop and maintain. Another important issue is interoperability - a tool must co-exist in harmony with other tools.

WaveSurfer grew out of our needs for a unifying, flexible and extensible tool for research and educational purposes at the Centre for Speech Technology at KTH. In particular for the 1999 summer school on multi-modality in language and speech systems (MiLASS), we had planned various hands-on exercises all dealing with time-based signals in some form, most notably audio signals, but also pitch curves, transcriptions, control parameters to speech synthesizers/talking heads etc. During previous courses we had been using a variety of tools for the different exercises, some of which had rather complex user interfaces. We realized that the students would be more productive if they only had to learn one tool, which could be used for all of the exercises.

We wanted a framework that could accommodate many kinds of functions for speech and other time-based signals in one shell, with a user interface that would be flexible enough to show only the information relevant to the task at hand. Other requirements were support for the platforms, Windows, Linux, Solaris, and HP-UX, easy integration of existing modules, handling of long signals, and quick navigation. The tool should also be user-friendly and customizable and last but not least be available as open source, preferably under the GNU general public license, with active support and development.

We did an initial survey of available tools such as Entropic's *ESPS/waves+*, the Speech viewer in the CSLU toolkit [2], Transcriber from DGA/LDC [5], Syntrillium's CoolEdit, SFS from University College London [6], EMU from Macquarie University [7], and Segmenter/Transcriber from ISIP, Mississippi State University [8]. None of these met all of our requirements, though each of them certainly surpassed them in at least one. It was decided to start development on a new tool. In July 1999, after a month of hard work, an early version of WaveSurfer was ready for use by the participants of the summer school. The tool was a success and was also presented at the Education Arena during Eurospeech ´99 and included on the demonstration CD-ROM. A more developed version of WaveSurfer was made available in January 2000 for public download at http://www.speech.kth.se/wavesurfer/. WaveSurfer has seen multiple revisions since those early days and matured in many ways.

2. WaveSurfer OVERVIEW

WaveSurfer' s user interface is simple yet powerful and has been designed in agreement with common user interface standards. It uses context sensitive popup-menus and direct manipulation to avoid cluttering of the user interface. When a sound file is opened in WaveSurfer a new work area is created. Each sound file and any associated data can be viewed using multiple time-aligned sub-windows, or panes. Each pane may contain waveforms, spectrograms, time-axes, transcription labels, parameter curves and the like. Panes can be dynamically added, configured and removed by the user, and a collection of panes can be saved as a configuration, that later can be applied to any other sound. This allows users to easily customize the interface to fit specific tasks. WaveSurfer handles multiple simultaneous sounds, one per work area. Work areas can also be configured in a master-slave fashion. There is a waveform of the complete sound at the bottom of each work area used for navigation and zooming, which greatly simplifies working with large sound files.

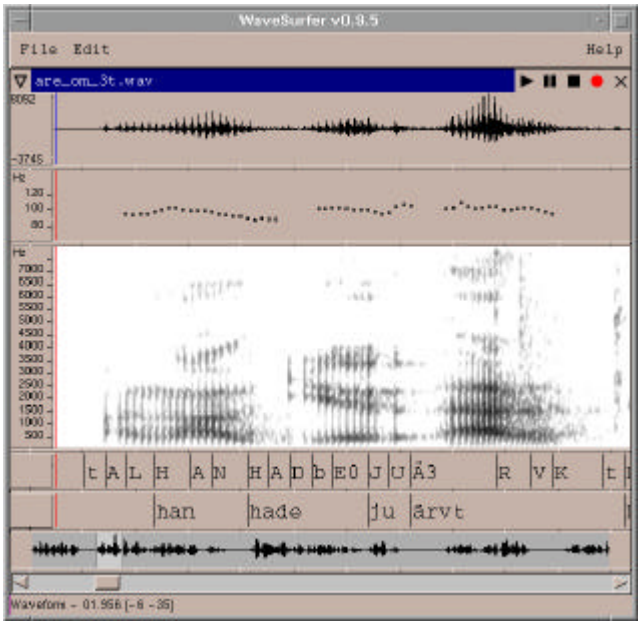

Figure 1: A WaveSurfer screen shot showing from top to bottom waveform, pitch contour, spectrogram, phonetic labels, and word labels. At the bottom is the navigation scrollbar used for navigation and zooming.

Pre-computed waveform data is cached on disk in order to speed up display, thus providing instant file load and quick access anywhere in even the largest of sound files. During playback a moving cursor shows the current play position, optionally, the whole display scrolls. The user interface is shown in Figure 1, with a selection of panes. A time axis could be added anywhere in the display with a selection in a context sensitive popup menu.

WaveSurfer handles standard audio file formats (Windows WAV, Sun AU, Apple AIFF, MPEG layer 3) as well as formats specific to the speech research field, such as NIST/Sphere and proprietary formats from Entropic and Kay Elemetrics. For raw files, or files with unknown format, WaveSurfer has an algorithm which tries to guess how to interpret the data. File data can be kept on disk or loaded into memory if the file is relatively short.

WaveSurfer's labeling facilities currently support the TIMIT, *ESPS/waves+*, and HTK formats. The label format is specified per transcription pane and it is fully possible to have several transcriptions in different formats loaded at the same time for a given sound file.

Printer output is supported through postscript generation of pane contents.

The WaveSurfer tool is available in source form for all supported platforms and in binary format for Windows and Linux RedHat 6.1.

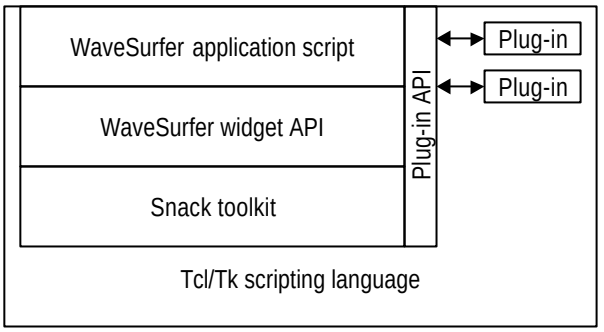

Figure 2: The WaveSurfer architecture, with its three layers, which all can interact with functionality-providing plug-ins. Tcl/Tk is used as a "glue" language.

2.1 Configurations

A prominent feature of WaveSurfer is its flexible configuration facilities. A WaveSurfer configuration includes information about which panes should be drawn and how and were they should appear. Also, associated information such as what kind of data files to load for a certain sound file is included. It is easy to save the current set of panes as a configuration and to apply it later on any loaded sound clip.

Configurations are stored in the users home directory in a succinct text format and can easily be edited, sent by e-mail, or made available on a web-page to other users.

3. IMPLEMENTATION

The WaveSurfer tool is built using the Tcl/Tk [9] scripting language, with scripts and dynamic link libraries wrapped into a single executable. The tool consists of a simple core, combined with a novel plug-in architecture for all task-specific functionality.

WaveSurfer is structured in several distinct layers. At the top-most level there is a relatively brief script defining the basic user interface with its menus. This script constitutes the actual application. Beneath this layer, there is scripting language library, the WaveSurfer widget library, which encapsulates a high level sound visualization widget. The application layer is used to create and destroy such widgets and to connect menu selections to widget methods. The widget library sits on top of the third layer, the Snack toolkit, described in section 3.1, which is implemented as a C-language dynamic link library. These layers make up the core of the WaveSurfer tool as seen in Figure 2. The core simply has the functionality needed to allow a user to load a sound file and create a set of empty panes. In order to create for example a new speech visualization tool a developer would add the necessary functionality using dedicated plug-ins.

WaveSurfer plug-ins can extend the tool at all three levels. For example, support for a new sound file formats can be added by creating an extension to Snack in a separate link library using its C-APIs. Plug-ins can also reconfigure the top-level user

interface, and add functionality here using Tcl's ability to function as a glue language. Most plug-ins, however, extend the widget library with new pane types.

WaveSurfer plug-ins are implemented as Tcl/Tk scripts. During startup, the application will search certain system and user directories for plug-in scripts, and read them into the interpreter. Each plug-in will supply a list of procedures that will be invoked as callbacks as certain events occur in the application. The plug-in API consists of 28 such callbacks, for actions such as opening of a sound file, widget creation and destruction, cursor movement, sound modification notification and many others, of which the plug-in will implement the ones it has interest in.

Tcl/Tk has several attractive characteristics, which made it the natural choice for development language. It is an open source programming system, which is available for most common computer platforms, several UNIX varieties, Windows, and Macintosh. No compilation step is involved during development. Also, you get a fully featured scripting language, including powerful facilities for building user interfaces. Applications can take advantage of the script interpreter, for tasks such as generating configuration scripts, for execution of batch jobs, and to provide a console for power users. This compares with the limited batch processing facilities of many other tools. In our approach there is no need for users to install or even to now about Tcl/Tk at all. Also, there is no conflict with any existing installations. The Tcl interpreter is compact, which makes WaveSurfer fit on an ordinary floppy disk.

3.1 The Snack Toolkit

Designers of sound processing tools face issues like different types of audio hardware interfaces, signal encoding schemes, methods for signal visualization, as well as a multitude of sound file formats. There are also issues related to the presentation of the audio signal, the selection of deployment platform, and the design of a responsive user interface for handling big, sometimes vast, amounts of data. In order to address most of these problems, an audio programming toolkit, Snack [10], specifically designed for the handling of audio data and its presentation, was developed at KTH. Snack offers developers a high level of abstraction in combination with a uniform programming interface on a wide range of computer platforms. The toolkit has components, which a developer combines into an application, leaving most of the details to be handled by the component implementations.

At the core of Snack there is a mixing engine which connects to the computers audio hardware. This engine, which runs in a thread of its own, provides an interface that allows multiple simultaneous connections, with on-the-fly audio format conversions. Internally, Snack handles all sound data as floating point, in order to speed up signal processing computations and to maintain data integrity during a sequence of transformations.

Snack's principal component is the sound object. A sound object has methods for operations such as file I/O, in common audio file formats, playback, recording, editing, conversions, and signal processing. In order to record a sound clip into a sound object all that is needed is an invocation of the objects record method. Sound data will immediately start to accumulate in the background until a subsequent stop method is invoked. The actual sound data can reside in memory, on disk, or be associated with a stream.

Snack also has dedicated visualization objects such as waveforms and spectrograms. These are highly configurable and can be linked to Snack's sound objects. They will automatically reflect all changes to the sound object they are linked to, even in real time, if sound data is currently recorded into an object. The visualization objects know how to render themselves in postscript for hard copies.

Snack is currently implemented as an extension to the Tcl/Tk scripting language, though, for performance reasons, almost all of the code is implemented in C. The scripting layer is used to configure objects and link them to each other. This combination gives applications based on Snack the same degree of performance as dedicated C/C++ application as well as drawing on the benefits from Tcl's rapid development nature and its powerful graphical toolkit. Also, the applications will run all platforms currently supported by Snack, i.e. Windows, Linux, Macintosh, Solaris, HP-UX, FreeBSD, NetBSD, and IRIX.

It is possible to extend Snack both at the script level and at the C-level. There is an internal API to add new commands, file formats, and filters. This API also hides platform differences, which makes it very easy to port code between various platforms. In most cases a simply re-compilation is all that is needed.

The Snack toolkit has been available to the speech community since 1998, as open source, and also seen considerable contribution from its many users. Snack can be obtained both in source and binary-format from http://www.speech.kth.se/snack/, and has had more than 9000 downloads, 5000 alone during the present year. Several speech tools have already been built using Snack at different sites [11,12,13], some of which have also been made freely available, e. g. Transcriber [5]. Snack has also seen other uses, for example, it has successfully been used as an audio platform in conversational dialog systems [10].

4. APPLICATIONS

WaveSurfer has already found a variety of applications. It is currently the standard tool for viewing and editing sound files and transcriptions in our group as well as at other sites.

During the 1999 MiLaSS summer school, as well as during exercises in the under-graduate courses in speech technology given at the department, WaveSurfer has been used as a platform for experimentation with multimodal text-to-speech synthesis and basic acoustic phonetics.

Building on the WaveSurfer plug-in API, we have developed a control interface to the KTH text-to-speech synthesis and animated talking head technology [14]. This TTS-interface allows for interactive control of the speech synthesis process, from text input, via phonetic transcription, down to detailed editing of segment durations and trajectories for synthesizer control parameters. The control parameter tracks are seamlessly

integrated into the WaveSurfer environment in a way that allows parameters to be displayed jointly with other information in the panes. For example, it is possible to display formant parameters controlling a formant synthesizer layered on top of a spectrogram.

The TTS-interface can be used pedagogically, to let students explore the effects of modifying the rule-generated acoustic or visual cues to change the identity of a segment, for example. It can also be used creatively to create dynamic facial animation. In one exercise, each lab group was given a sentence to synthesize and then add convincing animated facial expression to. The resulting sentences from all groups were finally played back as a dialogue between a travel agent and a customer. This interface has also been used for various experiments with perception of multimodal speech and facial cues.

It is also possible to embed WaveSurfer as a speech display widget in applications, a feature used for diagnostic purposes in the speech recognition module of a conversational system [10].

5. CONCLUSIONS

WaveSurfer's dissemination in the speech research community is facilitated by its flexible design, which features a plug-in architecture and a built-in script interpreter. WaveSurfer is open source itself and built using other freely available open source systems. Since WaveSurfer was first released to the community in January 2000, it has been downloaded from approximately 3500 sites, and its user base continues to grow. It is our hope that WaveSurfer will be prove to be a useful addition to the software libraries of researchers all over the world and that the full ramifications of its creation will continue to surprise its authors.

6. ACKNOWLEDGEMENTS

We would like to thank all users of WaveSurfer who have contributed improvement suggestions, bug reports, actual code, or encouragement.

7. REFERENCES

1. Free Software Foundation, "Philosophy of the GNU Project," [http://www.fsf.org/philosophy/], 2000.

2. Schalkwyk, J., de Villiers, J., van Vuuren, S., Vermeulen, P., "CSLUsh: an extensible research environment," *Proceedings of Eurospeech ´97*, pp. 689-692, Rhodes, Greece, 1997.
[http://cslu.cse.ogi.edu/toolkit/]

3. Taylor, P., Black, A., and Caley, R., "The Architecture of the Festival Speech Synthesis System," *Proceedings of the Third ESCA Workshop in Speech Synthesis*, pp. 147-151, 1998.
[http://www.cstr.ed.ac.uk/projects/festival/]

4. Ordowski, M., Deshmukh, N., Ganapathiraju, A., Hamaker, J., and Picone, J., "A public domain speech-to-text system," *Proceedings of Eurospeech ´99*, pp. 2127-2130, Budapest, Hungary, 1999.
[http://www.isip.msstate.edu/projects/speech/software/asr/]

5. Barras, C., Geoffrois, E., Wu, Z., and Liberman, M., "Transcriber: development and use of a tool for assisting speech corpora production," *Speech Communication, special issue on Speech Annotation and Corpus Tools, preprint* , 2000.
[http://www.etca.fr/CTA/Projets/Transcriber/]

6. Huckvale, M., 1987-1998. Speech Filing System. [http://www.phon.ucl.ac.uk/resource/sfs.html].

7. Cassidy, S., "Compiling multi-tiered speech databases into the relational model: Experiments with the Emu system," *Proceedings of Eurospeech ´99*, pp. 2127-2130, Budapest, Hungary, 1999.

8. Deshmukh, N., Ganapathiraju, A., Gleeson, A., Hamaker, J., Picone, J., "Resegmentation of Switchboard," *Proceedings 5th International Conference on Spoken Language Processing (ICSLP ´98)*, pp. 1543-1546, Sydney, Australia, 1998
[http://www.isip.msstate.edu/projects/speech/software/]

9. Ousterhout, J.K., *Tcl and the Tk Toolkit*. Addison Wesley, 1994. [http://dev.scriptics.com/]

10. Sjölander, K., Beskow, J., Gustafson J., Lewin, E., Carlson, R., and Granström, B., "Web-based Educational Tools for Speech Technology," *Proceedings 5th International Conference on Spoken Language Processing (ICSLP ´98)*, pp. 3217-3220, Sydney Australia, 1998.

11. Kim, B., Lee, J., Cha, J., Lee, G., "POSCAT: A Morpheme-based Speech Corpus Annotation Tool," *Proceedings of the Second International Conference on Language Resources and Evaluation*," pp. 567-572, Athens, Greece, 2000.

12. Charfuelán, M., Gil, J., Rogrí guez Gancedo, C., Tapias Merino, D., Hernández Gómez, L., "Dialogue Annotation for Language Systems Evaluation," *Proceedings of the Second International Conference on Language Resources and Evaluation*," pp. 1551-1556, Athens, Greece, 2000.

13. Frid, J., "An Environment for Testing Prosodic and Phonetic Transcriptions," *Proceedings of the 14th International Congress of Phonetic Sciences*, pp. 2319-2322, San Francisco, USA, 1999.

14. Beskow, J., "Animation of Talking Agents," *Proceedings of AVSP'97, ESCA Workshop on Audio-Visual Speech Processing*, Rhodes, Greece, 1997.