

NaniTrans: a Speech Labelling Tool

David Portabella, Albert Febrer, Asunción Moreno

Universitat Politècnica de Catalunya
Jordi Girona 1-3 08034 Barcelona, SPAIN
<http://gps-tsc.upc.es/veu>
asuncion@tsc.upc.es

Abstract

This paper deals with a description of NaniTrans, a tool for segmentation and labeling of speech. The tool is programmed to work on the MATLAB application interface, in any of the supported platforms (Unix, Windows, Macintosh). The tool has been designed to annotate large speech databases, which can be also partially preprocessed (but require manual supervision). It supports the definition of an environment of annotation: set of annotation levels (orthographic, phonetic, etc.), display mode (how to show information), graphic representation (waveform, spectrogram), keyboard short-cuts, etc. This configuration is then used on a speech database. A safe file locking system allows many annotators to work concurrently on the same speech database.

The tool is very friendly and easy to use by non experienced annotators, and it is designed to optimize speed using both keyboard and mouse. New options or speech processing tools can be easily added by using any MATLAB or user defined function.

1. Introduction

Transcription of large corpora is a hard task and usually requires experienced annotators. Sometimes the use of automatic signal processing tools can do this work. Nevertheless in most applications manual supervision can be avoided.

Most of time the process of annotation of a speech database consists on two basic steps. First, a list of speech files is automatically processed to derive a set of label files predicting the required annotation (being phonetic annotation, prosodic attributes estimation, etc.). Then human annotators supervise this set of labels (most of time is faster to supervise –only move- than to introduce the labels from the beginning), by observing the speech file (its waveform, spectrogram, pitch contour, etc.) besides the annotation.

In some of the most popular transcription tools it is difficult to modify or add new functions, mainly because the source code is not always available. And it is not always easy to personalize the environment, finding problems when trying to make compatible manual annotation to preprocessing steps, or adjusting the adequate representation. A large number of different transcription tools can be found on the LDC web, but with some different functionality.

In this paper we present a tool based on MATLAB. The main advantage of this software is that many signal processing tools are already implemented in the MATLAB library, and it is easy to implement most of speech processing techniques by combining a few of its supported functions. Adding new programs and functions is very easy, even C/C++ compiled routines. Moreover the tool can work in different platforms and operating systems. Additionally, a safe file locking system allows many annotators to work concurrently on the same speech database.

The tool has been designed to support segmentation and labeling of speech signals. A configuration menu let the user to set up working directories, file formats, annotation types, graphic windows and keyboard short-cuts. This configuration is then applied to the set of files to annotate.

The system has been used by non-experienced annotators. A synthesis database has been segmented and annotated (pitch epochs and phonetic annotation). The experience shows that the system is friendly, fast and easy to use.

2. File management

NaniTrans has been designed with three main objectives in mind: robustness, flexibility, easy of use and speed.

The revision system must be robust in the sense that it must work by itself. That means that, once configured and started the procedure there must be no doubt about each signal file will be revised once and only once. No matter how long it takes to revise the whole database, either on hour or one year; how many annotators work concurrently; or any other thing. In order to achieve a system as robust, in the above sense, as possible, NaniTrans performs *File Selection* and *File Locking* which are explained below. These techniques and structure have been previously implemented and used in the tool NaniBD (Nogueiras, 1998), making directory compatibility.

It must be flexible in the sense that NaniTrans can be adapted to a variety of signal databases. Any type of annotation can be used: orthographic, phonetic, prosodic, etc. One or many annotation types can be used simultaneously. Any type of graphic can be easily added using MATLAB. By default it supports waveform, spectrogram, energy and pitch contour representations.

In order to make the system as fast and easy of use as possible, both the mouse and keyboard should be used. The mouse is only used to move, add or delete marks and to choose what label to edit. The keyboard is used to perform the commands and edit labels. The keyboard has to be configured to implement quick commands.

2.1. File Selection

Many speech databases present a tree directory structure that reflects its contents in terms of speakers and/or sessions, as in the standard structure of Speechdat family (<http://www.speechdat.org>), i.e. the SpeechdatCar database.

All speech files share a common *base directory* (for example `/speech/signal/`). The name of the speech files (*filename*) are specified from the base directory (for example `block20/block201/v201A`). The program uses then an *index* file listing the filenames conforming the database.

The base directory for the label files can be the same as for the signal files, or another different (for example `/speech/label/`). NaniTrans would automatically create the path of the label files from label base directory if it does not exist. Label files will be defined by the label base directory, their corresponding speech filenames, and the specified extension.

And example of this structure can be found in table 1.

Base directory	Filename	Ext.
/speech/signal/	block20/block201/v201A	.wav
	block20/block201/v201B	.wav
	block20/block202/v202A	.wav
	...	
/speech/label/	block20/block201/v201A	.lab1
	block20/block201/v201A	.lab2
	block20/block201/v201B	.lab1
	block20/block201/v201B	.lab2
	block20/block202/v201A	.lab1
	block20/block202/v201A	.lab2
	...	

Table 1. Directory structure example.

So in the configuration dialog of NaniTrans the following specifications must be detailed:

- an index file: list of filenames.
- for the speech files: the base directory, format and extension.
- for the label files: the base directory, format and extension.

2.2. File Locking

A safe file locking system allows many annotators to work concurrently on the same speech database. NaniTrans ensures that each speech file is revised once and just once. In order to accomplish this, NaniTrans implements a locking strategy, whose objectives are:

- not revising or processing any speech file whose revision has already started;
- and keeping track of the speech files successfully revised

This is done using two different locks:

ExtStart: when an annotator starts revising a speech file a zero file with this extension is created. Another annotator cannot start revising a speech file if this locking file already exist.

ExtEnd: when the annotator finishes revising a speech file, the *ExtStart* locking file is deleted and a zero file with this extension is created. Another annotator cannot start revising a speech file if this locking file already exist.

The *ExtStart* and *ExtEnd* extensions can be defined by the user. For example `.start` and `.end`, as in the example of table 2.

Using both the *ExtStart* and *ExtEnd* locking extensions ensures that all speech files are fully processed

only once in no matter what conditions. In this way, for example, we can add speech files at any time.

Base directory	Filename	Ext.
/speech/control/	block20/block201/v201A	.end
	block20/block201/v201B	.start
	...	

Table 2. File locking example.

3. Annotation

NaniTrans accepts different types of annotation that must be configured. Many annotation types can be used simultaneously. It must be specified whether the annotation type require labels. For example, pitch epoch marks do not need any labels (they just indicate the time instant), but phonetic labels need to specify the annotated allophone. If the annotation has labels, it must be specified if they are free format (for example orthographic annotation) or there is one fixed vocabulary (as a set of allophones, as a SAMPA vocabulary).

Type	Labels	Ext
Pitch epoch	No labels	.pe
Phone segmentation	SAMPA vocabulary	.ph
Orthographic annotation	Free format	.ort

Table 3. Annotation examples.

The format used to load and to save the annotation file also has to be configured for each mark type. It has some basic standard formats, but it is very easy to add new formats. Formats have been designed according to other observed labeling tools and output of some speech processing programs.

4. Visualization

The visualization on the NaniTrans tool can be explained by defining three concepts: a) the *scene*, which is the definition of one representation mode (speech signal and associated annotation); b) the *window*, which is an instance of visualization of a particular scene and c) the *screen*, which consists of the menu, windows and status bar.

Many windows (from the same or different scenes) can be visualized at the same time, which can optionally be synchronized.

These concepts are detailed below.

4.1. Scenes

A scene is a definition of a representation mode. It specifies how the speech signal will be visualized (that is, waveform, spectrogram, energy, pitch contour or any other implemented), and what annotation types will be used in that scene as well as the position of the labels respect of the speech representation (allowing above, below or inside graphic position). Other configuration parameters allow to choose the colors of annotation marks, labels and background as well as the position of the label respect to the mark.

For example, in table 4 two different scenes are defined. Scene *A* visualizes the spectrogram and allows annotation of allophones and noise. Both annotation levels are visualized above the speech graphic and allophone annotation is also showed inside the graphic. Scene *B* visualizes the speech waveform. The annotation of allophones is the same than in scene *A*, and pitch epoch marks are visualized inside and below.

Figures 1 and 2 show the aspect of scenes *A* and *B* respectively.

Scene	Visualization	Annotation	Position
A	spectrogram	allophones	above, inside
		noise	above
B	waveform	allophones	above, inside
		pitch epochs	below, inside

Table 4. Scene definition examples.

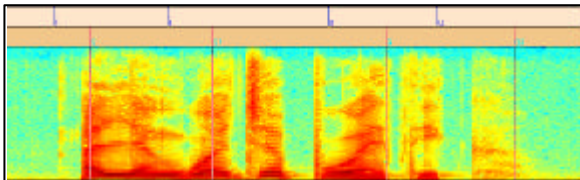


Figure 1. Visualization of scene *A* of table 4.

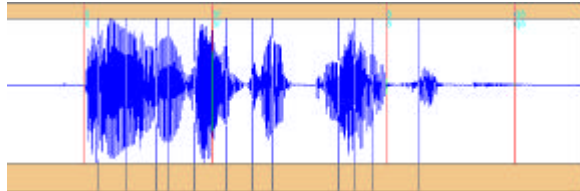


Figure 2. Visualization of scene *B* of table 4.

4.2. Windows

Some confusion may appear here. A window is a positioned area in the NaniTrans screen. It is a particular visualization of a defined scene.

Many windows may be defined. Each window is linked to one or more scenes, but only one scene is visible at a time in that window. This enables to define just one scene, and use it in two windows (while one could show the entire scene, the other could be zoomed). When a window has more than one scene linked, the user can select another scene of the window maintaining the zoomed state.

NaniTrans allows to synchronize one window with one or more other windows. When synchronized, zooming in one, zooms in all of them. For example, one window should be linked to scene *A* of the last subsection (table 4),

it would be the visualization in figure 1. Another window should be linked to scene *B*, as in figure 2. If both windows are synchronized the a zoom in one of the windows applies to the other.

4.3. The Screen

The screen is defined here as the whole visualization of the program. It consists of:

- menu*: it should not be used (at least not frequently) as keyboard short-cuts can be personalized and allow faster interaction of the user. It can be easily modified by accessing to the program code
- windows*: All the defined windows which are currently visible. There is the option to hide a window at any time.
- Status bar*: provides information about the current situation.

The main objective here is to use the maximum of the screen to show just what is really needed. Visible windows are always resized to make use of the maximum space in the screen..

In figure 3 a screen of NaniTrans is shown. In this particular example with three windows corresponding to three different scenes.

5. Keyboard short-cuts.

The keyboard must be configured to use short-cuts. A key can be configured to execute an ordered sequence of commands such as add label, play all, play selection, zoom, hide window, change scene, next signal, etc. Most common actions in a particular annotation scheme must be corresponded with a keyboard definition in order to save time and efforts of the annotators.

An example of a currently used keyboard short-cuts definition is shown in table 5.

Short-cut	commands
Z	Zoom In (an area must be selected)
X	Zoom Out (last window clicked)
C	Moves to the left (last window clicked)
V	Moves to the right (last window clicked)
A	Play All Speech Signal
S	Play Selection (an area must be selected)
D	Play Window (last window clicked)
Q	Show/Hide third window
W	Change the scene of third window
Enter	Confirm revision of this speech file Go to next Signal
Esc	Cancel revision of this speech file Quit NaniTrans

Table 5. Keyboard short-cuts example.

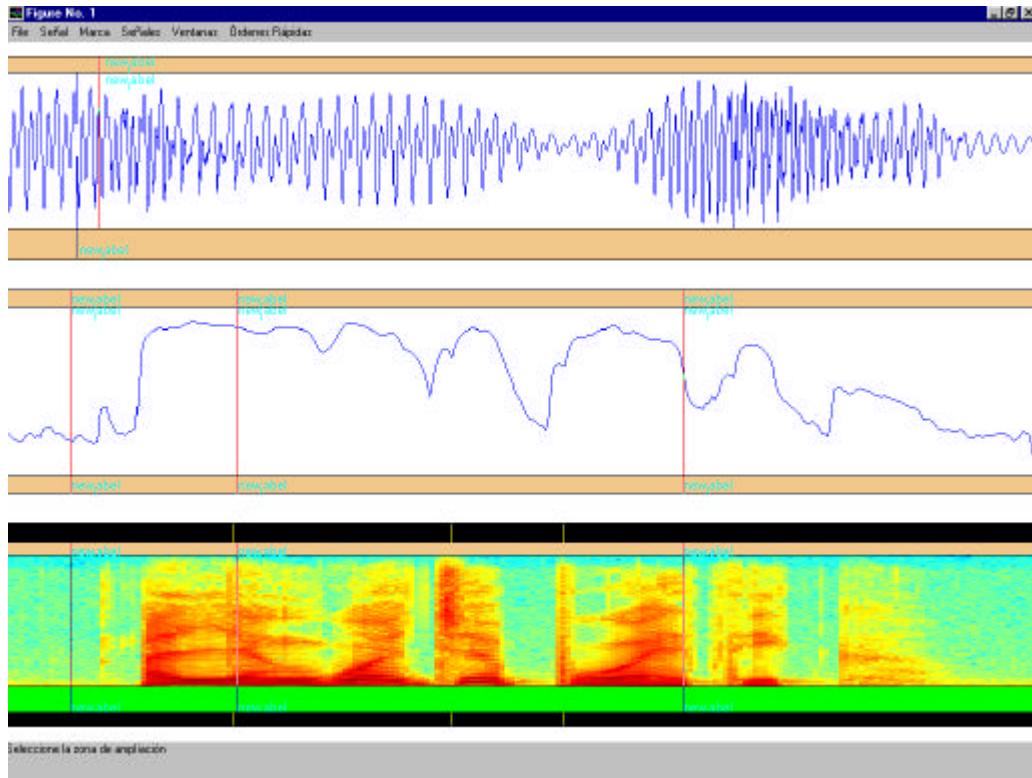


Figure 3. Screen sample of NaniTrans.

Analysis of Speech, in Proceedings of First COST-G6 Workshop on Digital Audio Effects (DAFX98), Barcelona, Spain.

6. Conclusions

In this paper we have presented a tool to annotate speech databases. In comparison to other tools, it has to be observed that it is fully configurable and easy to incorporate new features. And this can be done easily by using the set of functions in the MATLAB library. Any configuration and modification is directly usable in any of the environments supported by the platform.

It is a tool fully designed to automate the process of working on a speech database consisting on a large set of speech and multiple label files, allowing its organization in directory trees. This tool presents an advantage over other tools designed to work on a single file (Febrer, 1998).

A file locking system is implemented in the tool to allow multiple annotators working on the same speech database, ensuring that each file is labeled once and only once.

As a summary, we think that it is a robust tool which can be of interest by researchers involved in speech database segmentation and annotation of any kind. This software is for free distribution, and can be downloaded from <http://gps-tsc.upc.es/veu>.

7. References

- Albino Nogueiras and Asunción Moreno (1998), NaniBD: a set of tools for Transcribing and Validating Speech Databases, Proceedings of the First International Conference on Language Resources and Evaluation, LREC'98, Granada, Spain.
- Miquel Febrer, Albert Febrer, Antonio Bonafonte and Ignasi Esquerra (1998), Aneto: a Tool for Prosody