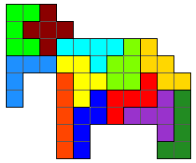
TELIDA: A Package for Manipulation and Visualization of Timed Linguistic Data

Titus von der Malsburg, Timo Baumann, David Schlangen

Department for Linguistics, University of Potsdam, Germany



Introduction

TELIDA consists of
- a light-weight graphical viewer,
- an accompanying library and
- command-line tools

to **view**, **process** and **manage** timed linguistic data.

Unlike other tools, TELIDA not only represents *interval* and *point events*, in several *tracks*, but also *hypothesis events*, which are the points of creation of hypotheses about events (and may contain even arbitrary, user-defined data).

TEDview, the viewing component, has some **unique features** such as
- a **film view** that proceeds smoothly during play,
- ability to process and **display incremental data**, visualizing the dynamics of speech processing and interpretation as it happens,
- ability of **online-visualization** for speech processing applications (e.g. spoken dialogue systems), helping to debug and understand temporal aspects of system behaviour,
- fine-grained control of visualization for intuitive use.

The accompanying library features
- easy import from and export to a **variety of formats** (Praat (Boersma, 2001), Wavesurfer (Sjölander and Beskow, 2000), XML)
- **object-oriented interface** to handling timed linguistic data, with a broad API suited to process language data
- handling of **incremental** and **n-best** speech processing results with a focus on evaluation measurements
- fully **documented** and thus (fairly) easy to extend
- written in Perl.

The library-based tools contain
- simple scripts that capture the most common tasks like adding or removing alignments, searching in transcription files, etc.
- scriptable command-line applications that give fast and easy access to most of the library functions (removing the need to actually program a script yourself).

The Viewer

Unlike other tools (Wavesurfer, Praat, and others), a track's display changes with time (represented by a red line) if there are **hypothesis events**, always displaying the content of the most recent hypothesis. This makes TEDview **ideal to display** and understand the **dynamic aspects** of language processing, e.g. in dialogue systems or the ACT-R framework, as processing unfolds.

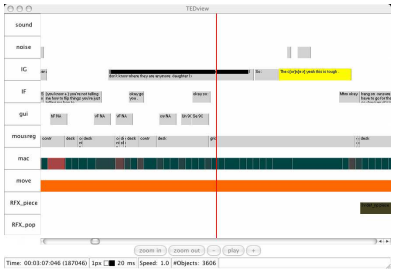
TEDview is not limited to displaying alignments in tracks, but can be extended to **display all sorts of data** in separate inspector widgets through the use of plug-ins.

Additionally, TEDview can be fed data incrementally via TCP, allowing multiple components to write to it simultaneously and acting as the viewing/debugging component of multiple agents in a distributed language processing system.

Analysis of Dialogue Data

In the DEAWU project (Schlangen and Fernández, 2007), we combined Praat transcriptions and MMAX2 (Müller and Strube, 2006) annotations in a time-aligned view.

We also encoded numerical values through color of labels, (e.g. mouse motion in the *mac* track above).

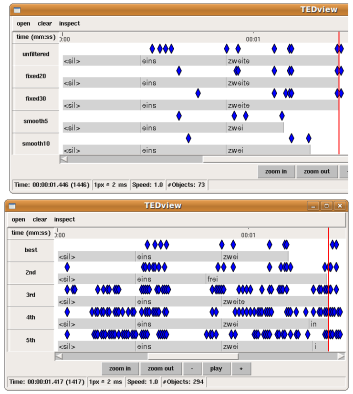


Analysis of SDS Performance

In the INPRO project we use TEDview to analyze and visualize the **incremental hypotheses** of several modules of a spoken dialogue system (Baumann et al., 2009).

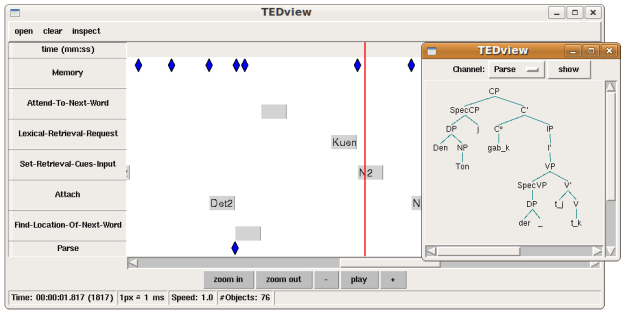
This application makes heavy use of TEDview's incrementality features.

Incremental changes to a track are shown as diamonds. When playing, the track display changes as time passes the diamonds.



Analysis of Cognitive Models

In another project, we use TEDview to visualize the output of an ACT-R (Anderson et al., 2004) simulation of human sentence parsing (Patil et al., 2009). Different tracks show different aspects of the model over (simulated) time. Notice the plug-in that incrementally displays generated parse trees.



Extending TEDview

- generic plug-in architecture to display data of almost any kind.
- plug-ins are written in Python
  → many pre-written widgets can be used in plug-in writing.
- plug-ins for parse-trees and bar-charts are already included
- As an example, the code for the parse-tree widget, which makes use of NLTK's (Bird et al., 2009) parse drawing ability is shown to the right.

```
class SyntaxInspector(InspectorWidget):
    def __init__(self, master, channel, **opts):
        InspectorWidget.__init__(self, master, channel, **opts)
        self.channel = channel
        self.cf = CanvasFrame(parent=self)
        self.cf.pack()
        self.trees_on_display = []
        self.display(channel.latest_events())

    def display(self, events):
        while self.trees_on_display:
            tw = self.trees_on_display.pop()
            self.cf.destroy_widget(tw)
        for event in events:
            tree = Tree.parse(unicode(event))
            tw = TreeWidget(self.cf.canvas(), tree, draggable=1)
            self.cf.add_widget(tw, 10, 10)
            tw.bind_click_trees(tw.toggle_collapsed)
            self.trees_on_display.append(tw)
```

Bibliography

Timo Baumann, Michaela Atterer, and David Schlangen. 2009. Assessing and Improving the Performance of Speech Recognition for Incremental Systems. In *Proceedings of NAACL-HLT 2009*, Boulder, USA.

Steven Bird, Ewan Klein, and Edward Loper. 2009. *Natural Language Processing with Python. Analyzing Text with the Natural Language Toolkit*. Oreilly & Associates Inc.

Paul Boersma. 2001. Praat, a system for doing phonetics by computer. *Glot International*, 5(9-10):341-345.

Christoph Müller and Michael strube. 2006. Multi-level annotation of linguistic data with MMAX. In *Corpus Technology and Language Pedagogy: New Resources, New tools, New Methods*, pages 197-214. Peter Lang.

Umesh Patil, Marisa Boston, John Hale, Shravan Vasishth, and Reinhold Kliegl. 2009. The interaction of surprisal and working memory cost during reading. In *Proc. of the CUNY sentence processing conference*, Davis, USA.

David Schlangen and Raquel Fernández. 2007. Speaking through a noisy channel – experiments on inducing clarification behaviour in human-human dialogue. In *Proceedings of interspeech 2007*, Antwerp, Belgium.

Kare Sjölander and Jonas Beskow. 2000. Wavesurfer – an open source speech tool. In *Proc. of the Sixth ICSLP*, Beijing, China.

Download

TELIDA is open source software and available for download from `http://www.ling.uni-potsdam.de/~timo/code/telida/` .

Further Information

Please contact the authors via e-mail at { **malsburg | timo | das** } @ling.uni-potsdam.de . More information on this and related research is available on the authors' websites where you can also find a PDF version of this poster.

This work was funded by a DFG grant in the Emmy Noether programme.