

# Sound Recognition Lab

## User Manual

Version 1.3 / March 2023

### Table of Contents

1 Introduction.....	3
1.1 Basic objects, file formats and data types.....	3
1.1.1 Audio files.....	3
1.1.2 Visualizations.....	3
1.1.3 Terminals.....	3
1.1.4 Predicates.....	4
1.1.5 Annotations.....	4
1.2 Workspace.....	4
1.3 Projects.....	7
1.3.1 Structure of projects.....	7
1.3.2 Create and modify SR-Lab projects.....	7
2 Audio files.....	10
2.1 Audio file formats.....	10
2.2 Wizards for audio files.....	10
3 Visualizations.....	11
3.1 Visualizations.....	11
3.2 Layout of the Visualization editor.....	14
3.3 The Visualization manager.....	15
3.4 Properties of Visualizations.....	17
3.5 Toolbar of the Visualization editor.....	19
3.5.1 Sonification controls.....	19
3.5.2 General controls.....	20
3.6 Shortcuts.....	20
3.7 Wizards for Visualizations.....	21
4 Terminals.....	22
4.1 Terminals.....	22
4.1.1 Signatures of Terminals.....	22
4.1.2 The classification algorithm.....	23
4.1.3 The search-area.....	24
4.2 Layout of the Terminal editor.....	24
4.3 The Terminal manager.....	26
4.4 Properties of Terminals.....	28
4.5 Toolbar of the Terminal editor.....	30

4.6 Wizards for Terminals.....	31
4.6.1 Normalize Terminals wizard.....	32
4.6.2 Clone Terminals wizard.....	34
4.6.3 Merge Terminals wizard.....	35
4.6.4 Extract Terminals wizard.....	36
4.6.5 Extract Merged Terminals wizard.....	38
5 Predicates.....	39
5.1 Predicates.....	39
5.1.1 Signatures of Predicates.....	40
5.1.2 The logical classification algorithm.....	40
5.2 Layout of the Predicate editor.....	41
5.3 The Predicate manager.....	43
5.4 Properties of Predicates.....	44
5.5 Toolbar of the Predicate editor.....	46
5.6 Wizards for Predicates.....	47
5.6.1 Interpret Current Visualization wizard.....	48
5.6.2 Find Sequences wizard.....	50
5.6.3 Merge Predicates wizard.....	52
5.6.4 Clean up Predicates wizard.....	53
5.6.5 Extract Predicates wizard.....	54
5.6.6 Extract Merged Predicates wizard.....	55
6 Annotations.....	55
6.1 Types of Annotations.....	55
6.2 The Annotation database interface.....	56
6.3 The Annotation manager.....	57
6.4 Visualization of Annotations.....	58
6.5 Toolbar of the Annotation database interface.....	60
6.6 Manual annotation of audio files.....	60
6.7 Automatic annotation of audio files.....	62
6.8 Wizards for Annotations.....	63
7 Audio Analytics.....	64
7.1.1 Analyze recording times wizard.....	65
7.1.2 Analyze sound pressure levels (dB) wizard.....	65
7.1.3 Analyze Terminals wizard.....	66
7.1.4 Analyze Predicates wizard.....	72
7.1.5 Analyze Terminal / Predicate Annotation wizard.....	74
8 Monitoring acoustic processes.....	78
9 System requirements and known issues.....	80
9.1 System requirements.....	80
9.2 Known issues.....	80

# 1 Introduction

SR-Lab provides interactive visual tools to solve a large variety of everyday audio pattern recognition problems.

With SR-Lab you can model efficient detectors (classifiers) for simple and complex acoustic patterns without special programming knowledge.

Classifiers can be used for automatic classification and analysis of audio data and for monitoring acoustic processes.

SR-Lab can also be used to quickly generate training and test data sets for neural network or support vector based classification hard- and software.

## 1.1 Basic objects, file formats and data types

SR-Lab works with only *five* different data types:

1. Audio files
2. Visualizations (graphs)
3. Terminals (simple classifiers)
4. Predicates (complex classifiers)
5. Annotations (markings in audio data)

In this section these five objects are just briefly listed. In later chapters they will be described in more detail.

### 1.1.1 Audio files

Audio files are representations of acoustic processes that have been digitized in a recording process. SR-Lab can process audio data in the common uncompressed PCM (pulse code modulation) format. How to work with audio files is explained in more detail in Chapter 2.

### 1.1.2 Visualizations

*Visualizations* are graphs, i.e. graphical representations of audio data. They can be created and modified in the Visualization editor. Visualization files (\*.vis) are saved under the project path in the "Visualizations" subfolder. How to work with Visualizations is explained in more detail in Chapter 3.

### 1.1.3 Terminals

*Terminals* are classifiers (detectors) for basic acoustic patterns. They can be created, modified and tested in the Terminal editor. Terminal files (\*.trm) are saved under the project path in the "Terminals" subfolder. How to work with Terminals is explained in more detail in Chapter 4.

### 1.1.4 Predicates

*Predicates* are classifiers (detectors) for complex and abstract acoustic patterns. They can be created, modified and tested in the Predicate editor. Predicate files (\*.prd) are saved under the project path in the "Predicates" subfolder. How to work with Predicates is explained in more detail in Chapter 5.

### 1.1.5 Annotations

Annotations are markings of specific patterns in audio data. They are the result of a search process in which audio data is automatically classified by Terminals or Predicates. Annotations can also be manually created. How to work with Annotations is described in Chapter 6.

#### **Note**

All data types can be used for *audio data analysis*. How this is done is described in Chapter 7.

Both Terminals and Predicates can be used for *real time audio monitoring*. How this is done is described in Chapter 8.

## 1.2 Workspace

The workspace of SR-Lab is divided into the following areas:

1. The *main menu*: This is the central launching point for performing all kinds of tasks.
2. The *Visualization editor*: This is an editor to create and modify Visualizations (see Figure 1) and to visualize Annotations.
3. The *Terminal editor*: This is an editor to create, modify and test Terminals (see Figure 2).
4. The *Predicate editor*: This is an editor to create, modify and test Predicates (see Figure 3).
5. The *Annotation database*: This is a database to handle sets of Annotations (see Figure 4).

After launching SR-Lab, the Visualization editor is shown. Terminal and Predicate editors can be accessed by clicking the tabs "Terminal editor" respectively "Predicate editor" in the top left corner.



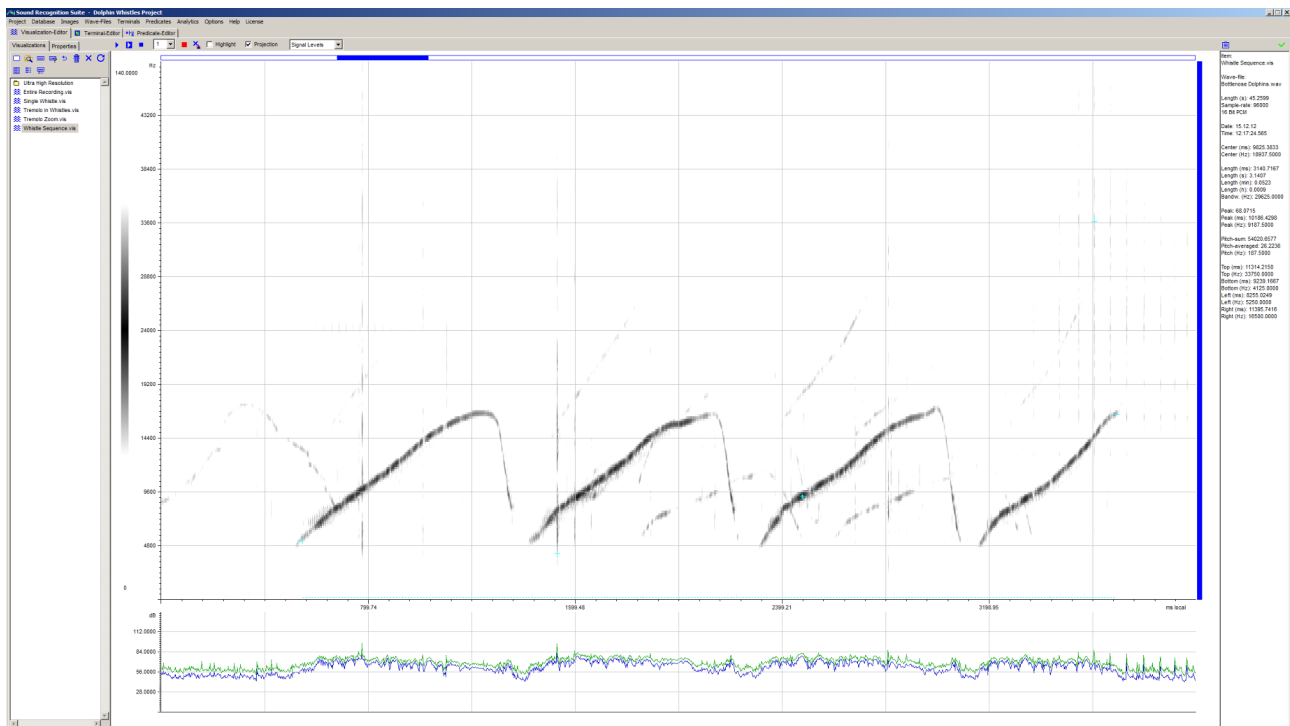


Figure 1: Spectrogram of dolphin whistles in the Visualization editor. The spectrogram shows time (x-axis), frequency (y-axis) and energy (color coded) of the audio signal. Below the spectrogram a graph of the signal level is shown.

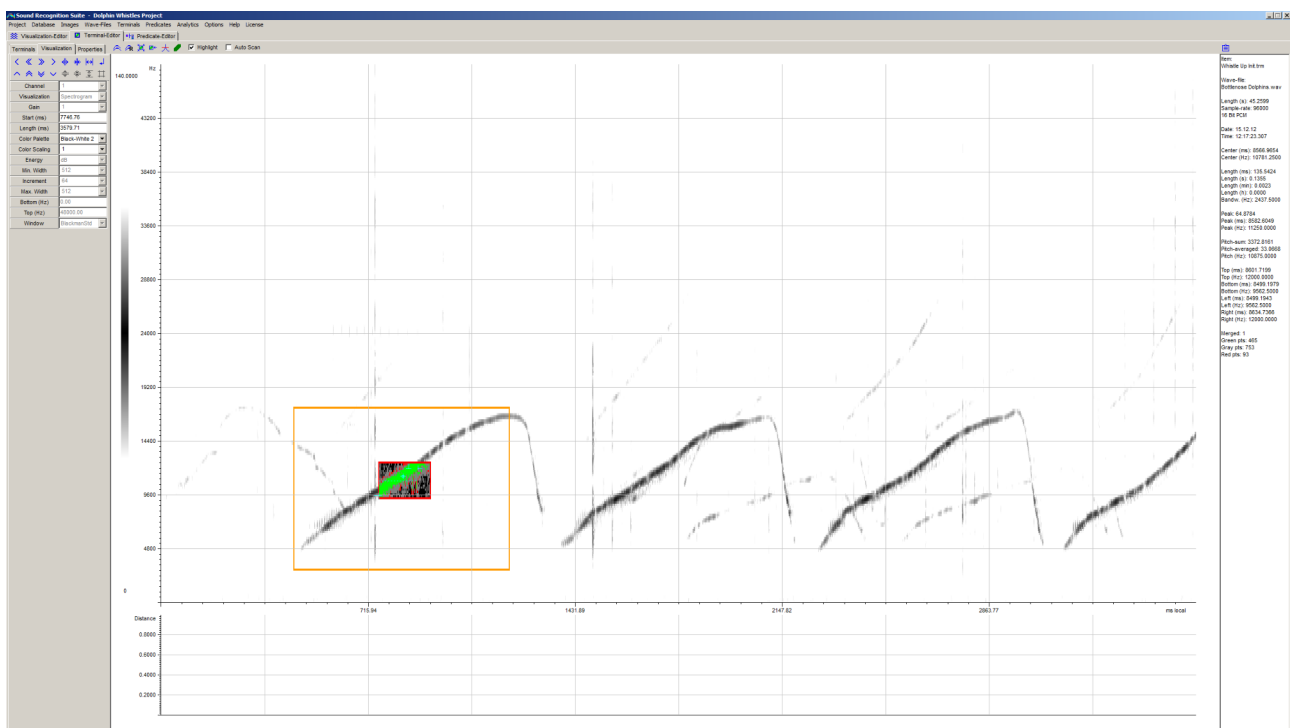


Figure 2: Terminal editor showing a Terminal for a part of a dolphin whistle.

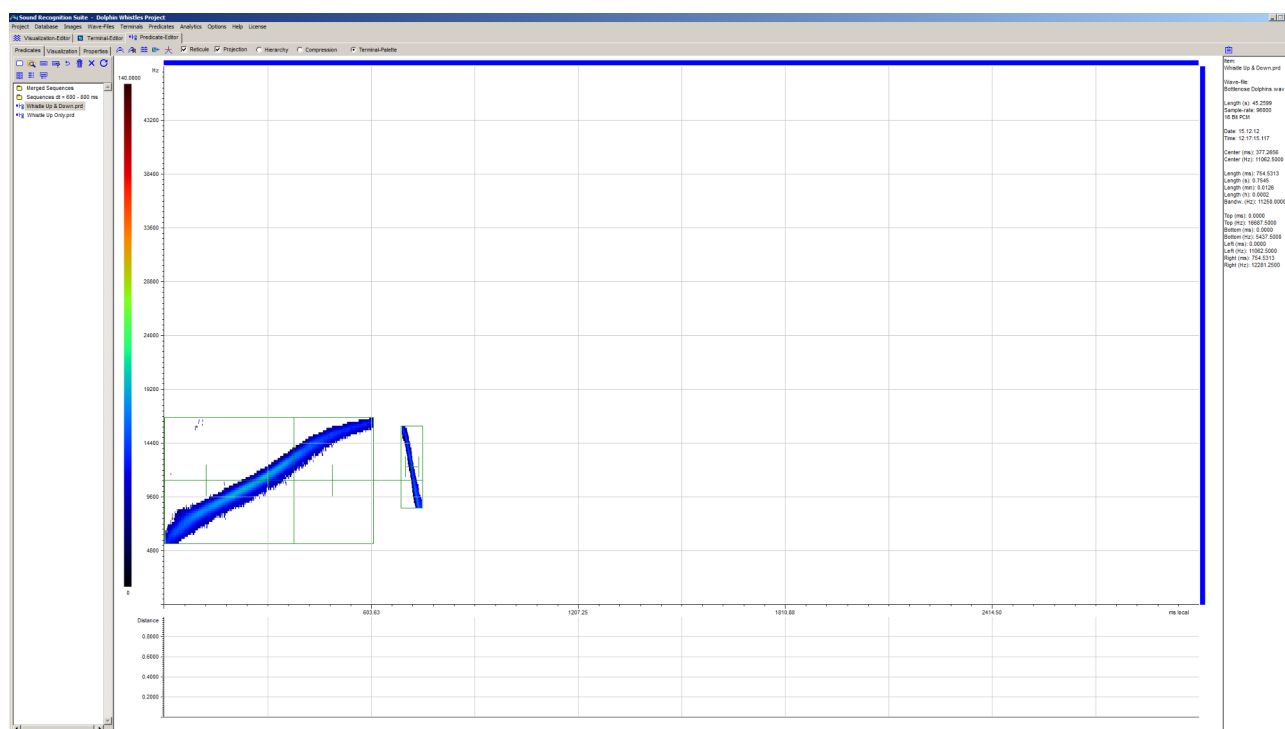


Figure 3: Predicate editor showing a Predicate for a certain whistle type. The Predicate consists of two Terminals connected by the AND operator.

Annotations Database

Count = 1395

Wavefile like: \* First n Items: \*  
Classifier like: \* Distance above: 0

Nr.	Soun...	Chan...	Class...	Simil...	Left S...	Widt...	Top Hz	Heig...
1	ricadi...	1	Crac...	0.208...	10994	745	2411...	1722...
2	ricadi...	1	Crac...	0.087...	28510	745	2411...	1722...
3	ricadi...	1	Crac...	0.233...	28976	745	2411...	1722...
4	ricadi...	1	Crac...	0.207...	32795	745	2411...	1722...
5	ricadi...	1	Crac...	0.140...	36678	745	2411...	1722...
6	ricadi...	1	Crac...	0.122...	44209	745	2411...	1722...
7	ricadi...	1	Crac...	0.202...	48991	745	2411...	1722...
8	ricadi...	1	Crac...	0.084...	51585	745	2411...	1722...
9	ricadi...	1	Crac...	0.203...	89986	745	2411...	1722...
10	ricadi...	1	Crac...	0.103...	97067	745	2411...	1722...
11	ricadi...	1	Crac...	0.118...	107797	745	2411...	1722...
12	ricadi...	1	Crac...	0.244...	108449	745	2411...	1722...
13	ricadi...	1	Crac...	0.083...	109256	745	2411...	1722...
14	ricadi...	1	Crac...	0.149...	109970	745	2411...	1722...
15	ricadi...	1	Crac...	0.191...	148527	745	2411...	1722...
16	ricadi...	1	Crac...	0.185...	150049	745	2411...	1722...
17	ricadi...	1	Crac...	0.170...	154552	745	2411...	1722...
18	ricadi...	1	Crac...	0.163...	157844	745	2411...	1722...
19	ricadi...	1	Crac...	0.147...	169381	745	2411...	1722...

Figure 4: Interface of the Annotation database.

## 1.3 Projects

### 1.3.1 Structure of projects

A SR-Lab project consists of a project folder with the project file (\*.srp) and the subfolders “Visualizations”, “Terminals”, “Predicates”, “Annotations/Terminal Annotations”, “Annotations/Predicate Annotations” and “Settings”. It is not permitted to change this folder structure.

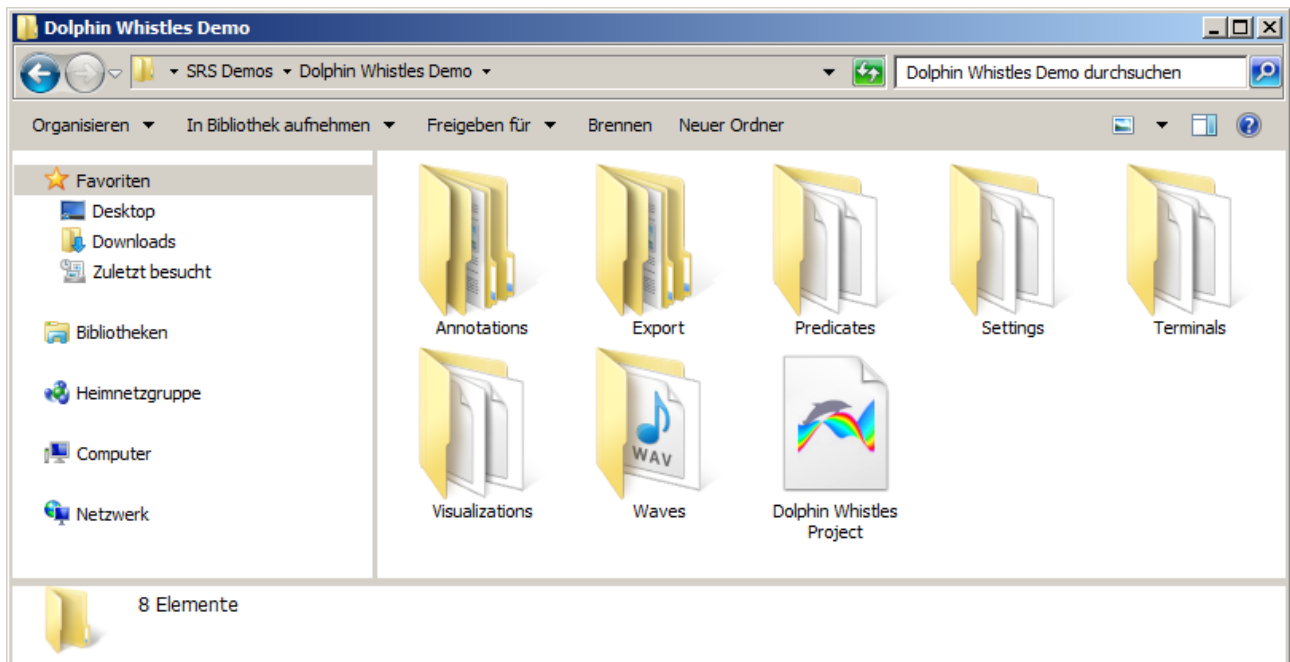


Figure 5: Project folder with subfolders

In addition, there is the "Waves" subfolder for audio files and the "Export" subfolder for all files generated by SR-Lab. These two folders can be moved or replaced at will.

Initially, only the “Settings” folder contains files. All other folders are empty. When working with SR-Lab, all objects are saved in their specific subfolders by default.

Within the subfolders assigned to them, all objects can be organized into further subfolders as desired. Moving, renaming and deleting projects can be done with a file manager.

### 1.3.2 Create and modify SR-Lab projects

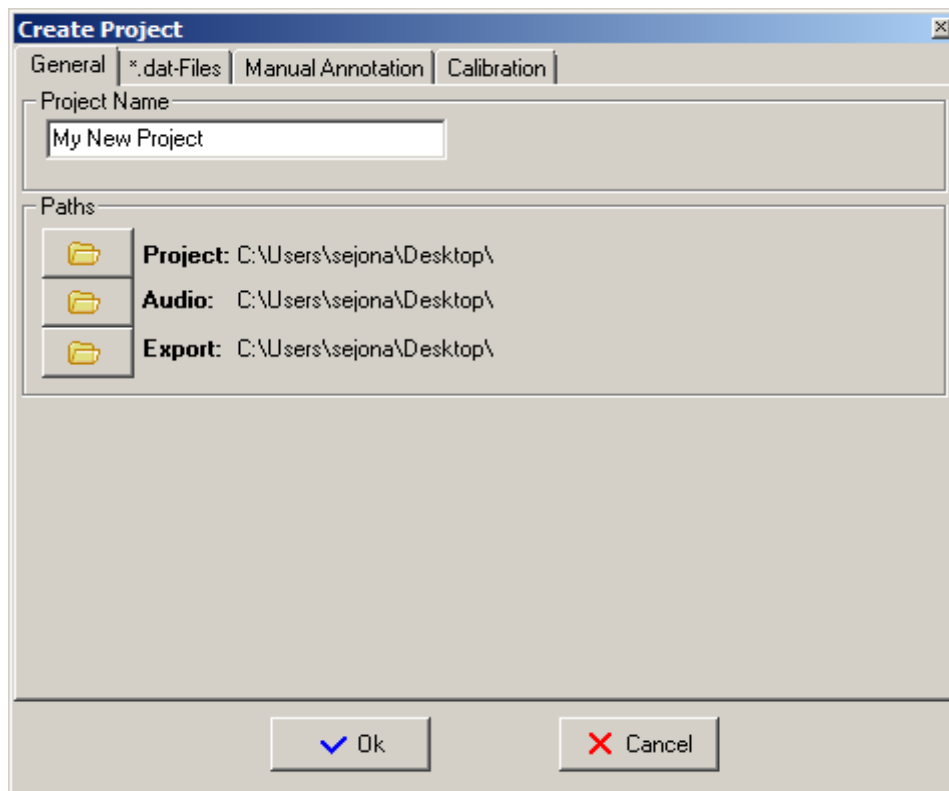


Figure 6: Create project window

1. Click the “New Project” button in the main menu. The “Create Project” window shows up (see Figure 6).
2. Enter the name of the project. Select paths for the project, the default audio directory and the default export directory (see Figure 6).
3. Settings for PCM files without header (.dat files) have to be specified under the ".dat-Files" tab. Specify sample rate, bit depth, the number of channels and offset (see Figure 7). Note that sample rate and encoding depth of uncompressed audio files in \*.wav format don't have to be specified!
4. The "Manual Annotation" tab contains the Key-Terminal table used for manual annotation of audio files. For details see Section 6.6.
5. To calibrate measurements (e.g. in noise control projects) open the calibration index card and specify one or two files that contain a calibrated recording (typical length is 10 seconds). The name of these files has to be of the form XXdB.wav where XX specifies a value in dB (e.g. 94dB.wav – see Figure 8).
6. Click the OK button. The project is created and written into the specified project folder.

After project creation, any of the settings can be changed. To do so, click the "Project Settings" menu item in the main menu and the "Project Settings" window shows up.

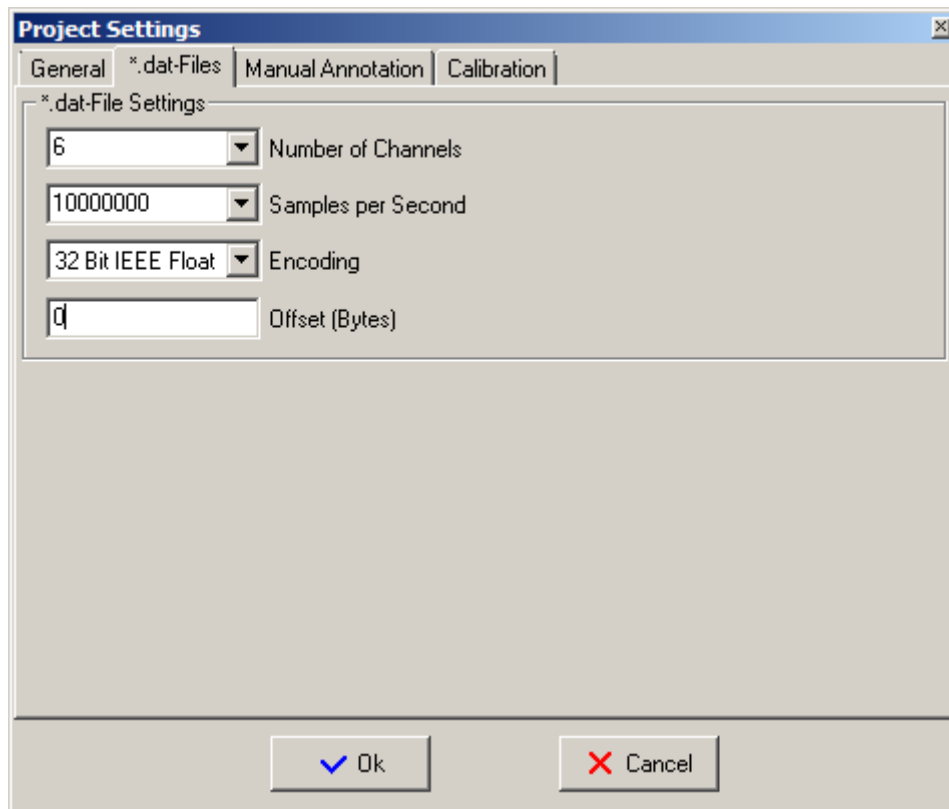


Figure 7: Settings for \*.dat files without header

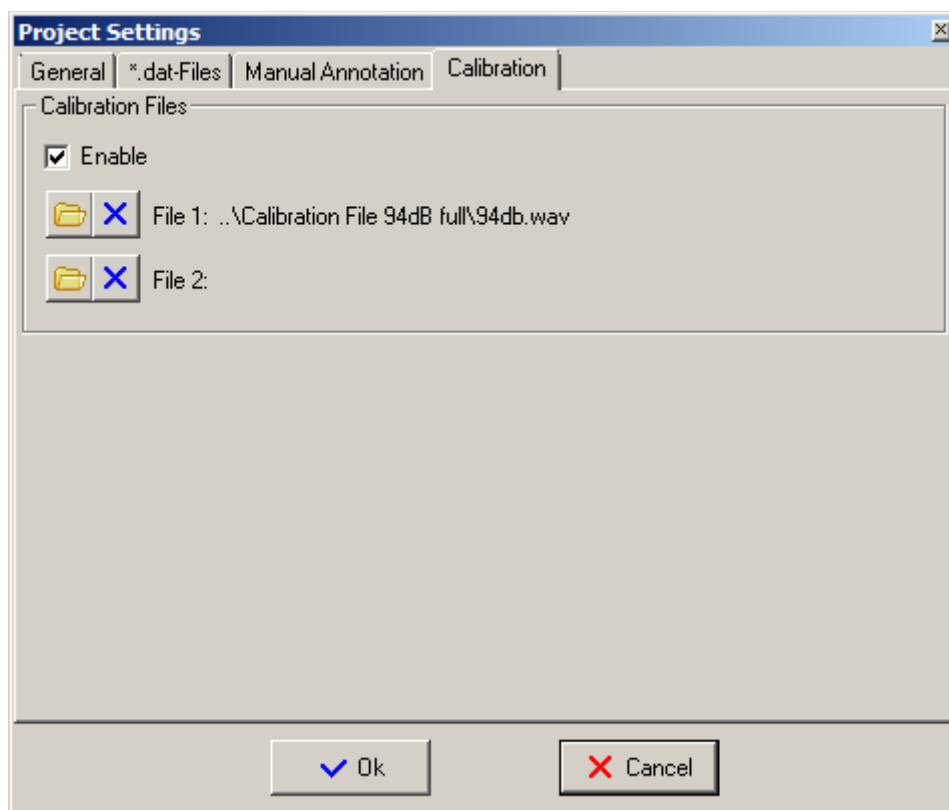


Figure 8: Calibration settings

## 2 Audio files

### 2.1 Audio file formats

Audio files are representations of acoustic processes that have been digitized in a recording process. SR-Lab can process audio data in the common uncompressed PCM (pulse code modulation) format.

Sound-Recognition-Lab can handle the following audio file formats:





1. Uncompressed PCM with RIFF header (\*.wav): Both mono and multi-channel audio files are accepted. Bit depth can be 16, 24, 32 or 32 Bit IEEE float. Any sample rate is possible.
2. Uncompressed PCM without header (\*.dat): Both mono and multi-channel audio files are accepted. Bit depth can be 16, 24, 32 or 32 Bit IEEE float. Any sample rate is possible. Bit depth, sample rate and offset for \*.dat files have to be specified in the "Project Settings" window. Note that *all kinds of PCM signals* can be processed by SR-Lab not only audio signals.





**SR-Lab uses 32-Bit pointers to access audio files. Beware not to cross the 2.1 GB border when accessing audio data. This can result in a crash of the system! Best is to use audio files smaller than 250 MB.**

### 2.2 Wizards for audio files

SR-Lab was created to work with very large amounts of audio data. In many cases it is useful to pre-process the audio files. SR-Lab provides a number of wizards to do so. All tools can be started via the main menu under the menu item "Audio Files".

Table 1: Wizards for audio files

Icon	Menu item	Task
	Extract Time Intervals from Audio Files	Wizard to extract audio data that is within a certain interval in daytime (e.g. from 12:00 a.m. to 3:00 p.m.). This wizard is self-explanatory.
	Apply dBA-Filter to Audio Files	Wizard to filter audio files with a dB(A)-filter. This wizard is self-explanatory.
	Apply FFT-Bandpass Filter to Audio Files	Wizard to filter audio files with a sharp FFT bandpass filter. This wizard is self-explanatory.
	Apply FIR-Bandpass Filter to Audio Files	Wizard to filter audio files with a smooth FIR bandpass filter. Filter settings can be specified in the Options window under the Audio tab. This wizard is self-explanatory.

Icon	Menu item	Task
	Change Bit Depth of Audio Files	Wizard to change the bit depth of audio files. This wizard is self-explanatory.
	Silence Intervals	Wizard to silence predefined intervals in audio files (e.g. an interval starting at 0ms and ending at 500ms). This wizard is self-explanatory.
	Fade In	Wizard to smoothly fade-in audio files. This wizard is self-explanatory.
	Split Audio Tracks	Wizard to generate single audio files from multichannel audio files. From each multichannel file one new file per channel is generated. This wizard is self-explanatory.

## 3 Visualizations

### 3.1 Visualizations

*Visualizations* are graphs, i.e. graphical representations of audio data. They can be created and modified in the Visualization editor. Visualization files (\*.vis) are saved under the project path in the "Visualizations" subfolder.

Visualizations are vital to understand the nature of the acoustic phenomena and the patterns you are dealing with. SR-Lab supports the following types of Visualizations:

1. color coded signal level graphs and oscillograms
2. Standard DFT<sup>1</sup>-based spectrograms (sonograms)
3. Multi-scale high resolution DFT-based spectrograms
4. DFT based phase spectrograms

---

1 DFT = Discrete Fourier Transform

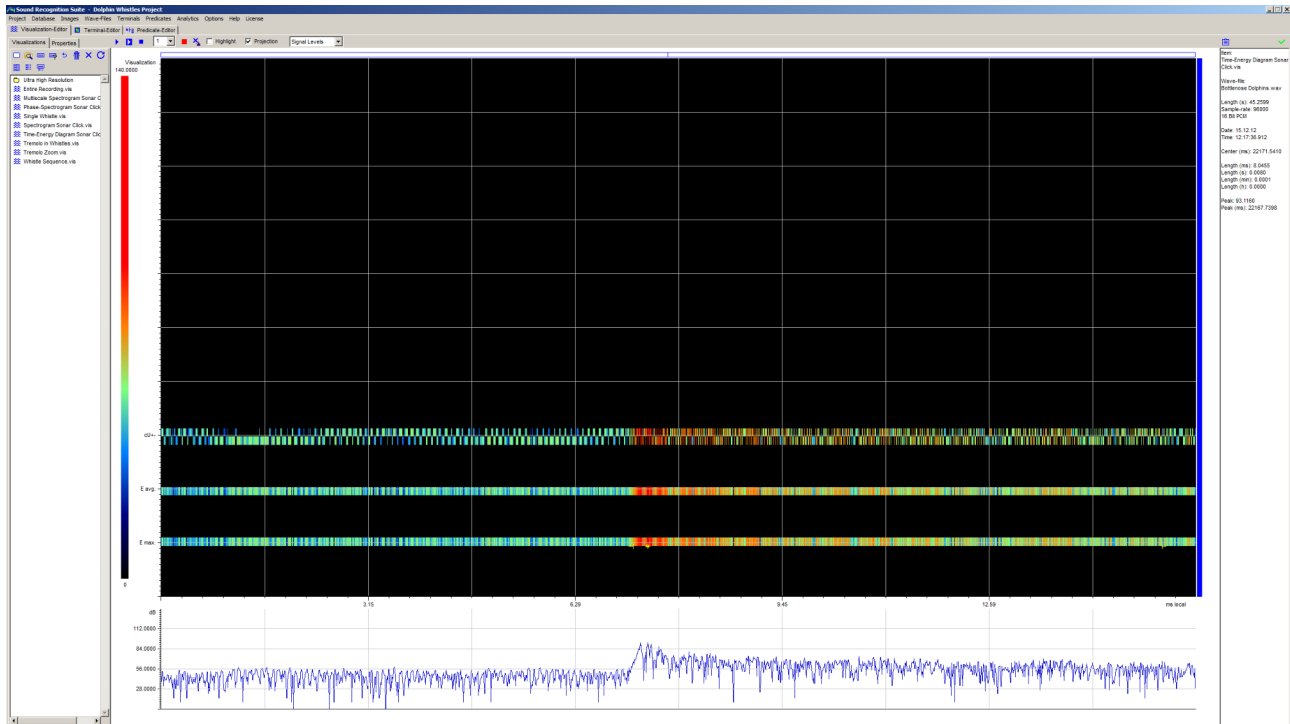


Figure 9: Color coded signal level graphs of a dolphin sonar click. (1)  $cU+-$  = Oscillogram: Positive and negative peak values of the signal; (2)  $E_{avg}$  = Smoothed sound pressure level diagram; (3)  $E_{max}$  = Peak sound pressure level diagram.

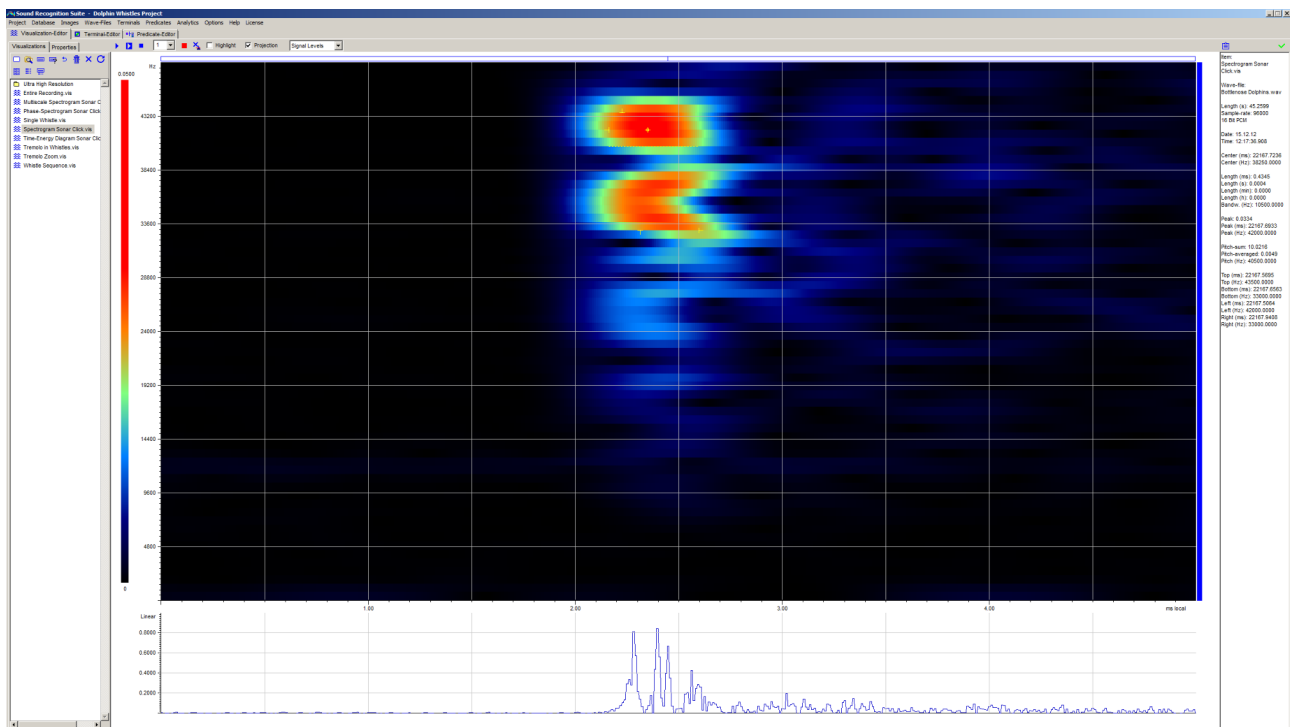


Figure 10: Spectrogram of a dolphin sonar click-sound ( $dt = 5$  ms, DFT window width = 128)



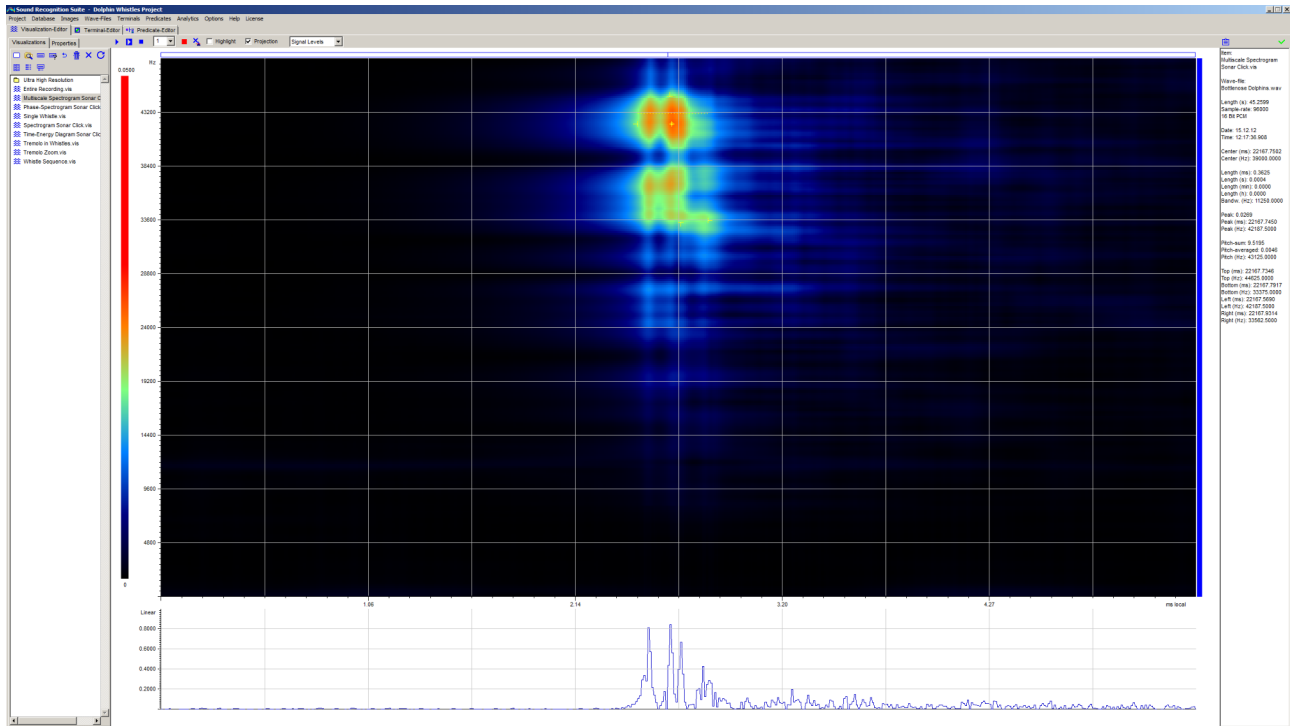


Figure 11: Multi-scale spectrogram of a dolphin sonar click-sound. Variable DFT parameters provide a better combined time-frequency resolution. ( $dt = 5$  ms, DFT window widths = 8, 12, 16, ... 512)

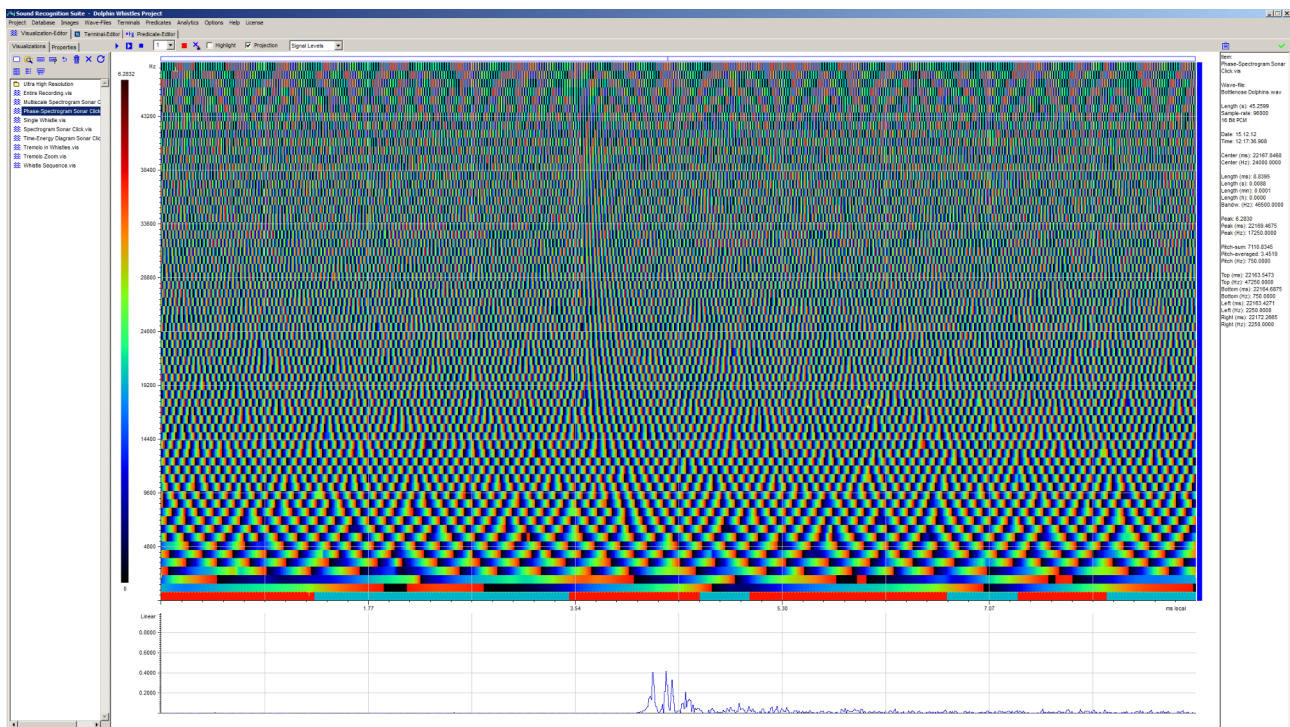


Figure 12: Phase spectrogram of a dolphin sonar click-sound. The the phase information of the DFT is displayed. ( $dt = 5$  ms, DFT window width = 128)

## 3.2 Layout of the Visualization editor

With the help of the Visualization editor you can:

1. Interactively browse audio files
2. Create visual representations of acoustic phenomena
3. Visualize Annotations in audio files
4. Select and sonify (play) acoustic-patterns
5. Manually annotate audio files
6. Perform basic acoustic measurements

The layout of the Visualization editor is shown in Figure 13. The workspace is subdivided into five different areas:

1. On top is a toolbar with sonification and general controls.
2. In the centre the current Visualization is shown.
3. At the left side two tabs are visible. One contains a manager for Visualization files (the Visualization manager) and the other one a property editor for the currently visible Visualization.
4. On the right side is a column that shows general information about the Visualization as well as basic acoustic measurements. All measured values refer to the  $-x$  dB bandwidth. Measured values outside this range are not measured. The  $-x$  dB parameter can be specified in the Options window.
5. Additional graphs can be displayed below the Visualization. You can choose between oscillogram, sound pressure level diagram and distance graphs. The latter relate to currently open Terminals and Predicates (see Section 4.1.2 and 5.1.2).

Note that:

1. Any area in a Visualization can be selected with the mouse buttons: left mouse button  $\rightarrow$  left time, right mouse button  $\rightarrow$  right time, ctrl & left mouse button  $\rightarrow$  bottom frequency, ctrl & right mouse button  $\rightarrow$  top frequency. The selection is indicated by red lines. To zoom into the selection press the "S"-key or click the Zoom-to-Selection button in the property editor.
2. In a Visualization the point with the highest energy value is marked with a cross, the frequency band with the highest energy density (in spectrogram-mode only) is marked with a dotted line. Top, bottom (in spectrogram-mode only), left and right borders within the  $-x$  dB bandwidth are marked with small corners. The color of the markers can be chosen in the Options window.

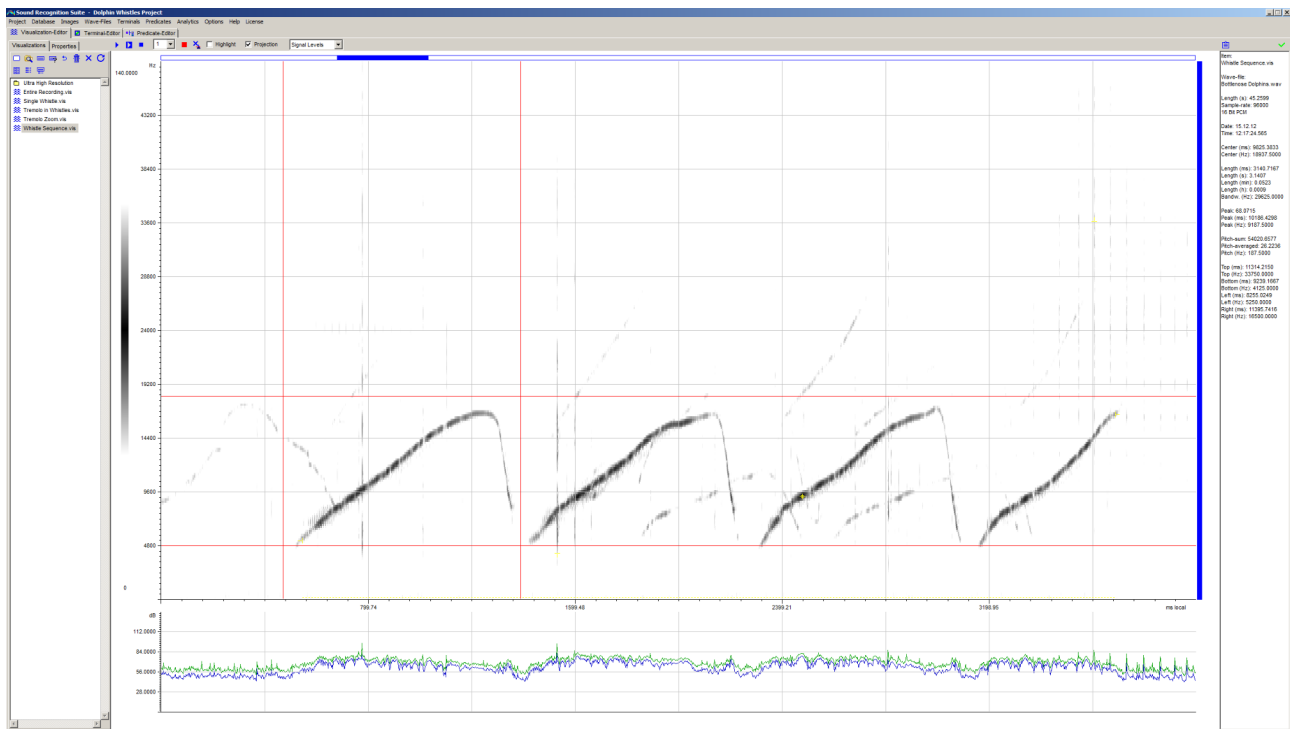


Figure 13: Spectrogram of a sequence of dolphin whistles in the Visualization editor. The leftmost whistle is selected manually. Under the Visualization a sound-pressure level graph is shown. ( $dt = 4000$  ms, DFT window width = 512)

### 3.3 The Visualization manager

With the help of the Visualization manager (see Figure 15) you can manage all Visualization files (\*.vis) of your project. A tree view of the files is displayed in the Visualization manager. Visualizations can be opened by double-clicking the entries of the tree view. Press the caps-lock key to browse the Visualizations with the up and down arrow keys of your keyboard.

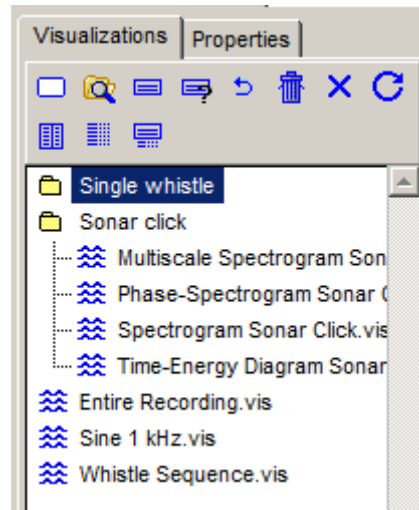


Figure 14: The Visualization manager

On the toolbar of the Visualization manager are several buttons which are explained in Table 2.

Table 2: Buttons of the Visualization manager toolbar

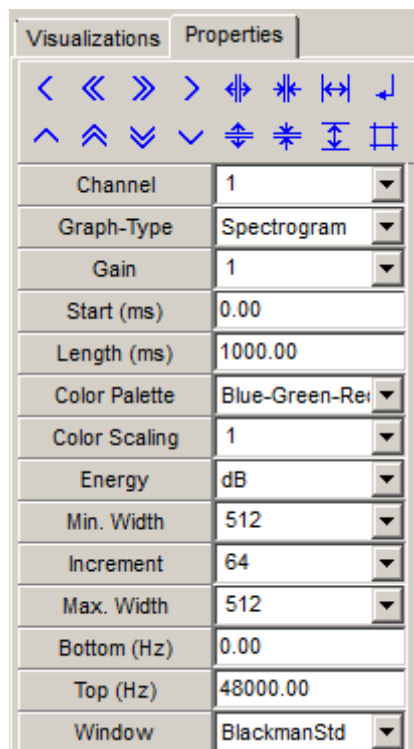
Icon	Function
	Create (initialize) a new Visualization by opening an audio file. The new Visualization is created with the settings-profile "Init.visp". The profile can be changed according to your needs.
	Open Windows-Explorer in the Visualizations folder of the current project. You can create new subfolders and rename or move all Visualization files as you want.
	Save the current Visualization.
	Rename and save the current Visualization.
	Undo changes (revert to saved).
	Delete the selected Visualization file or selected subfolder.
	Close current Visualization.
	Open a wizard to choose a Visualization profile and apply it to selected Visualizations.
	Save general properties of the current Visualization in a profile file (*.visp).
	Open a profile file and apply it to the current Visualization.
	Refresh tree view after file operations.

### 3.4 Properties of Visualizations

A Visualization is a graph that is generated from a chunk of audio data. Neither the audio data nor the graph itself is stored in the Visualization file. Only the path of the audio file and the settings needed to create the graph are stored in the file.

Properties of Visualizations can be changed in a property editor (see Figure 1) which is divided into two sections:

1. A toolbar for navigation in time and frequency domains. By clicking the buttons in the navigation toolbar you can "navigate" in audio data, i.e. change time and/or frequency properties of Visualizations. These properties may also be changed by clicking the scrollbars of the Visualization editor or by using shortcuts (see Section 3.6). Navigation buttons are explained in Table 3.
2. A property-value editor for all properties of Visualizations. Some changes in the editor are executed instantly others require to click the small Enter button top right in the toolbar. Only those properties are shown that apply to the selected graph type. All properties of Visualizations are explained in Table 4.



Visualizations		Properties
<div> <span>&lt;</span> <span>&lt;&lt;</span> <span>&gt;&gt;</span> <span>&gt;</span> <span>⏮</span> <span>⏭</span> <span>⏪</span> <span>⏩</span> <span>↶</span> <span>↷</span> </div>		
Channel	1	
Graph-Type	Spectrogram	
Gain	1	
Start (ms)	0.00	
Length (ms)	1000.00	
Color Palette	Blue-Green-Red	
Color Scaling	1	
Energy	dB	
Min. Width	512	
Increment	64	
Max. Width	512	
Bottom (Hz)	0.00	
Top (Hz)	48000.00	
Window	BlackmanStd	

Figure 15: Property editor for Visualizations

Table 3: Navigation buttons of the Property editor

Icon	Function
	Move one step left along the x-axis (time-axis)
	Move entire screen left along the x-axis
	Move entire screen right along the x-axis
	Move one step right along the x-axis
	Zoom in along the x-axis
	Zoom out along the x-axis
	Zoom out to full along the x-axis
	Execute changes (Enter and refresh)
	Move one step up along the y-axis (frequency-axis)
	Move entire screen up along the y-axis
	Move entire screen down along the y-axis
	Move one step down along the y-axis
	Zoom in along the y-axis
	Zoom out along the y-axis
	Zoom out to full along the y-axis
	Zoom to selection

Table 4: Properties of Visualizations

Property	Value
Channel	Currently visible channel (only one channel is visible at once)
Graph Type	Basic graph type of the Visualization
Gain	Gain multiplier for signal amplitude
Start (ms)	Start time in audio file to compute the graph
Length (ms)	Length of audio data chunk to compute the graph
color Palette	Currently selected color palette

Property	Value
color scaling	color scaling multiplier (amplitude on z-axis)
Energy	Weighting function of the signal energy
Min. Width	Minimum size of the DFT window (a multi-scale spectrogram is computed automatically if Min. Width is smaller than Max. Width)
Increment	Increment for DFT window size in multi-scale spectrogram mode
Max. Width	Maximum size of the DFT window
Bottom (Hz)	Bottom frequency of the Visualization
Top (Hz)	Top frequency of the Visualization
Window	DFT-Windows: Hann, Hamming, Bartlett, Blackman, BlackmanStd, BlackmanOPT

## 3.5 Toolbar of the Visualization editor

### 3.5.1 Sonification controls

Acoustic data underlying a Visualization can be *sonfied* (made audible). You can play acoustic data unfiltered or bandpass-filtered. It is possible to slow-down or speed-up audio output. Sonification controls are located on the toolbar above the Visualization (see Figure 14). The controls are explained in Table 5.



Figure 16:  
Sonification toolbar


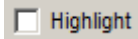
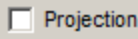
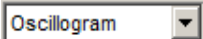
Table 5: Controls of the sonification toolbar

Icon	Function
	Play the current Visualization unfiltered
	Play the current Visualization or the current selection filtered. Bandpass filtering is done by using either the entire visible frequency band or the frequency band marked by the selection.
	Stop playing
	Values below 1 will slow down and values above 1 will speed up audio output.
	Start the Audio monitoring process (see Section 8)

### 3.5.2 General controls

On the toolbar above the Visualization a few more controls are located. They are explained in Table 6.

Table 6: General controls on the toolbar

Icon	Function
	Delete active search record
	Highlight Annotations of in projection mode (see Section 6.4)
	Enable/disable projection mode to map signatures of Terminals or Predicates onto the current Visualization (see Section 6.4)
	Graph shown below the Visualization. You can choose from Oscillogram, Signal Levels and Distances.

## 3.6 Shortcuts

To speed up work in SR-Lab use the shortcuts listed below. Most of these shortcuts can also be used in the Terminal editor and in the Predicate editor.

Table 7: Shortcuts in SR-Lab

Shortcut	Function
Ctrl+N	New project
Ctrl+O	Open project
Ctrl+S	Save object
Ctrl+W	Close object or project
Ctrl+left arrow	Move screen left
Ctrl+right arrow	Move screen right
Ctrl+up arrow	Move screen up
Ctrl+down arrow	Move screen down
Left arrow	Move one step left
Right arrow	Move one step right
Up arrow	Move one step up
Down arrow	Move one step down









Shortcut	Function
S	Zoom to Selection (in Visualization editor)
Esc	Redraw object and erase selection (in Visualization editor)
Blank	Play/Stop current Visualization (in Visualization editor)
Delete	Delete selected Annotation (in Visualization editor)
Caps lock + A-Z keys	Create manual Annotation from selection (in Visualization editor)

### 3.7 Wizards for Visualizations

SR-Lab has several wizards to generate and modify Visualizations. All wizards can be reached via the main menu under the menu item “Visualizations”. Table 8 shows an overview.

*Table 8: Wizards for Visualizations*

Icon	Menu item	Task
	Export Image of current Visualization	Dialogue to save an image (in *.bmp format) showing the current Visualization. This dialogue is self-explanatory.
	Export Images of Visualizations	Wizard for automatic generation of images (in *.bmp format) from Visualizations. Files are saved in the default export directory. This wizard is self-explanatory.
	Extract Audio File from current Visualization	Wizard for automatic extraction of an audio file from the current Visualization. This wizard is self-explanatory.
	Extract Audio File from Selection	Dialogue for automatic extraction of a bandpass filtered audio file from the selected part of the current Visualization.
	Extract Audio Files from Visualizations	Wizard for automatic extraction of audio files from selected Visualizations. Files are saved in the default audio directory. This wizard is self-explanatory.
	Create Visualizations	Wizard for creation of Visualizations from selected audio files. The profile “Init.visp” in the Settings directory is used to initialize the Visualizations. The profile can be overwritten with a new profile if desired. This wizard is self-explanatory.

## 4 Terminals

### 4.1 Terminals

*Terminals* are classifiers (detectors) for basic acoustic patterns. They can be created, modified and tested in the Terminal editor. Terminal files (\*.trm) are saved under the project path in the "Terminals" subfolder.

Terminals are derived from Visualizations from which they inherit all properties. In addition, Terminals have three more components:

1. A *signature*: A vector of data points characteristic for a certain class of acoustic patterns (see Section 4.1.1).
2. A *classification algorithm*: This is an algorithm that systematically compares the signature with audio data. It creates a Terminal Annotation whenever a computed distance measure between signature and data is below a certain threshold (see Section 4.1.2).
3. A *search area*: This is an area in the time-frequency domain surrounding the signature. The area is used for information retrieval purposes (see Section 4.1.3).

These components can be modified in many ways in SR-Lab. Interactive modelling can be done in the Terminal editor, machine programming can be done with a number of tools provided by SR-Lab (see Section 4.6).

In the following subsections the three parts of Terminals are described in more detail.

#### 4.1.1 Signatures of Terminals

A *Terminal signature* is a vector of data points derived from one or more Visualizations. The Terminal editor lets you modify the signature and find out which data points are indicative for a certain class of acoustic patterns. A signature has a three parts:

1. *The positive sub-signature*: A sub-set of data points representing the acoustic activity pattern itself. In manual editing mode, positive data points can be selected from a Visualization matrix using a movable selection frame. The frame acts as a filter, with those data points being filtered out of the matrix that are above a certain threshold value. The threshold can be set in the property editor (see Section 4.4).
2. *The negative sub-signature*: A sub-signature representing the contour of the noise floor surrounding the acoustic activity pattern. This part is called *negative* sub-signature because it does not represent the acoustic activity. In manual editing mode, data points of the negative sub-signature are automatically selected from a Visualization matrix when modifying the positive sub-signature. Negative data points are those that are below a certain threshold and adjacent to positive data points. Both threshold and adjacency parameters can be set in the property editor (see Section 4.4).

3. *The neutral sub-signature*: A sub-signature representing the zone in between positive and negative sub-signatures. This part is called *neutral* sub-signature. The neutral sub-signature is important for the merging algorithm (see Section 4.6.3). It has no influence on the behaviour of the classification algorithm.

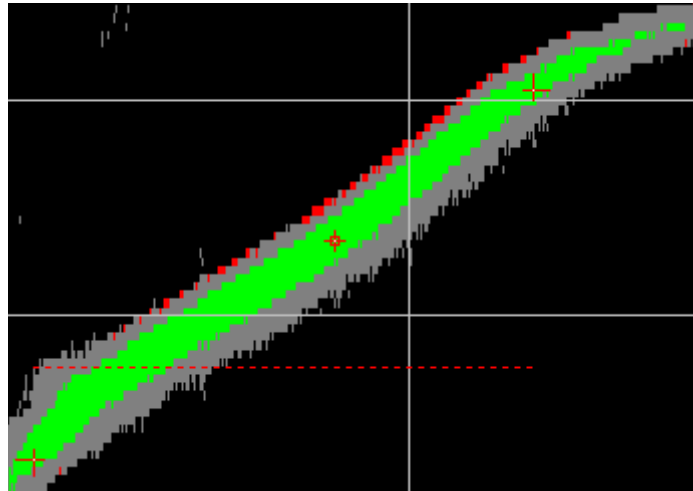


Figure 17: Three parts of a signature for a certain type of dolphin-whistles in highlight mode. The positive sub-signature is green, the negative sub-signature is red and the neutral sub-signature is grey.

#### 4.1.2 The classification algorithm

SR-Lab has a built in classification algorithm to find patterns in audio data that are *similar* to Terminal signatures. In order to calculate the similarity between a piece of audio data and a Terminal signature the algorithm proceeds as follows:

1. Create a Visualization matrix  $M$  from the entire Audio data using the same parameters as were used to generate the signature.
2. Calculate the distance curve  $D^P$  between  $M$  and the positive sub-signature.
3. Calculate the distance curve  $D^N$  between  $M$  and the negative sub-signature.
4. Calculate a combined distance curve  $D^C$  from  $D^P$  and  $D^N$
5. Select all intervals  $I$  in  $D^C$  that are below a certain threshold
6. Create for each interval in  $I$  at the point with the least distance an Annotation

As distance measure the Euclidean distance or the pruned Euclidean distance (only positive values) is used. Not in all cases  $D^N$  must be computed. It is also possible to compute  $D^P$  also, so that  $D^P = D^C$ .

All parameters relevant to the behaviour of the algorithm can be manually changed in the Terminal property editor (see Section 4.4). The behaviour of the algorithm, i.e. how it responds audio data, can be made visible in both Visualization and Terminal editors (see Figure 18).

### 4.1.3 The search-area

The search area surrounds the signature in the time or time-frequency domain. The search area is used for information retrieval purposes, in particular for automatic signature and audio data extraction. This is important for retrieving extended instances of the class defined by the signature (see Section 4.6).

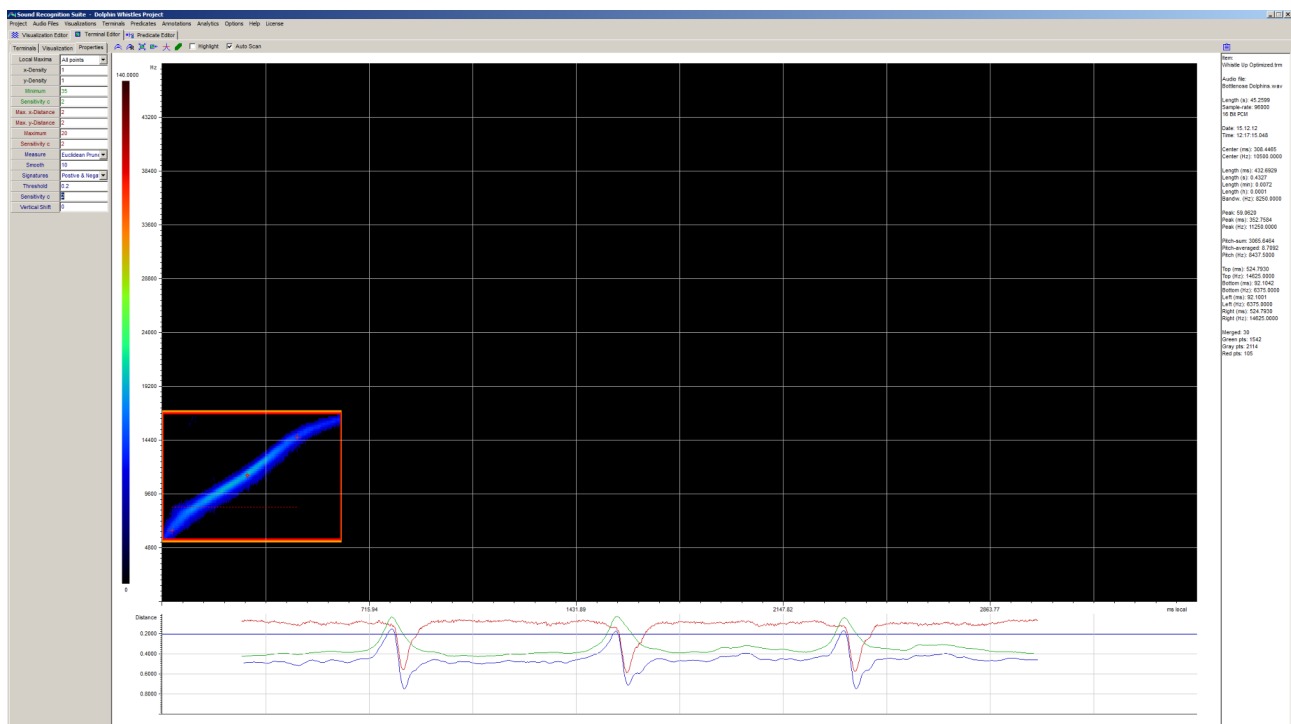


Figure 18: Terminal for a class of Dolphin whistles. Under the Terminal three graphs are shown. They represent the distance of the positive sub-signature (green graph), the negative sub-signature (red graph) and the overall distance (blue graph) to the audio data underlying the Visualization in 13. The horizontal blue line represents the threshold.

## 4.2 Layout of the Terminal editor

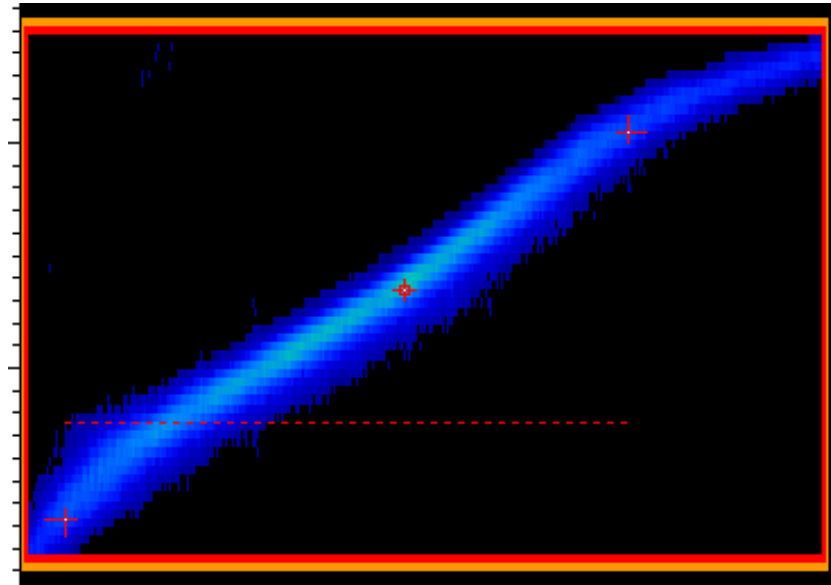
With the help of the Terminal editor you can interactively model Terminals. The editor covers design, implementation and test of Terminals. In particular, you can:

1. Initialize Terminals
2. Create, modify and optimize signatures
3. Modify and optimize the classification algorithm
4. Test the classification algorithm in real world tasks

5. Adjust the search area
6. Perform basic measurements of signatures of acoustic processes

The layout of the Terminal editor is shown in Figure 18. The workspace is subdivided into five different areas:

1. On top (below the main menu) is a toolbar with several general purpose controls (see Section 4.5).
2. In the centre the signature of the current Terminal is shown. The signature is framed by the *red selection frame*. The search area is framed by the *orange search frame*. The red rectangle defines the area for automatic extraction of the signature from the underlying Visualization-matrix. Size and location of the frame can be adjusted by drag & drop. Press the left mouse button and drag the edges or the entire frame to the desired location. (Note that only the red frame can be moved as a whole.)
3. On the left are three index cards. In the first there is a manager for Terminals files (\*.trm). In the second there is a property editor for the visual properties of the current Terminal. The third is a property editor for all other properties of the current Terminal (see Section 4.4).
4. On the right side general information about the Terminal as well as some basic measurements are shown. Note that all measured values refer to the  $-x$  dB range of the displayed matrix. The  $-x$  dB parameter can be specified in the Options window.
5. Three graphs are displayed below the Terminal. These relate the currently open Terminal to the currently open Visualization. The graphs can also be displayed in the Visualization editor below the current Visualization. They show how the classification algorithm behaves in relation to the audio data displayed by the Visualization. The green graph shows the computed distance between the positive sub-signature and the audio data, the red graph shows the computed distance between the negative sub-signature and the audio data. The blue graph shows the computed overall distance of the entire signature to the audio data.



*Figure 19: Signature representing a class of dolphin whistles. In the centre is the point with the highest energy. Limits within -3 dB bandwidth are marked by little corners. The dotted line indicates the frequency band with the highest average energy density.*

Note that:

1. It is possible to navigate through the underlying audio data without switching back to the Visualization editor. Use the arrow keys and the Ctrl-key to do so. Changes in the frequency domain are *not* possible in the Terminal editor.
2. The measuring point with the highest energy within a signature is marked with a cross. The frequency band with the highest average energy density (only in spectrogram mode) is marked with a dotted line. The upper, lower, right and left limits within -x dB bandwidth are marked with small corners (see Figure 19). The color of the markings and the -x value can be chosen in the Options window.

## 4.3 The Terminal manager

With the help of the Terminal manager you can manage all Terminal files (\*.trm) of your project. A tree view of the files is displayed by the manager (see Figure 20). Terminals can be opened by clicking the entries of the tree view. Press the caps-lock key to browse the Terminals with the arrow keys of your keyboard. The buttons of the Terminal manager toolbar are explained in Table 9.

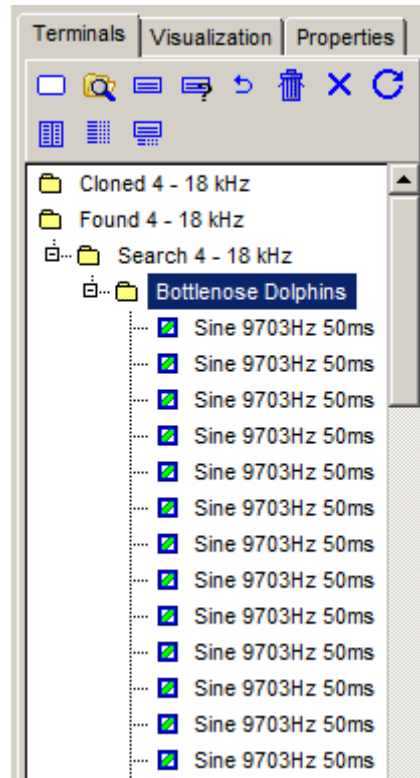




Figure 20: Terminal manager with tree view

Table 9: Buttons of the Terminal manager toolbar

Icon	Function
	Create a new Terminal. A new Terminal is initialized by deriving it from the current Visualization.
	Open Windows-Explorer in the Terminals subfolder of the current project. Within the Terminals subfolder you can create new subfolders and rename or move all Terminal files.
	Save current Terminal.
	Rename and save current Terminal.
	Undo changes (revert to saved).
	Delete selected Terminal file or entire selected subfolder
	Close current Terminal.
	Open a wizard to choose profiles and apply them to selected Terminals.
	Save general properties of current Terminal in a profile file.

Icon	Function
	Open a profile file and apply it to the current Terminal.
	Refresh tree view after file operations.

## 4.4 Properties of Terminals

A Terminal is an object that is derived from a Visualization. It inherits all of its properties and receives a number of new ones. Some of the Visualization properties can still be changed in a property editor under the Visualization tab (Figure 21). Terminal properties can be adjusted in a separate property editor under the Properties tab (Figure 22). All Terminal properties are explained in Table 10.

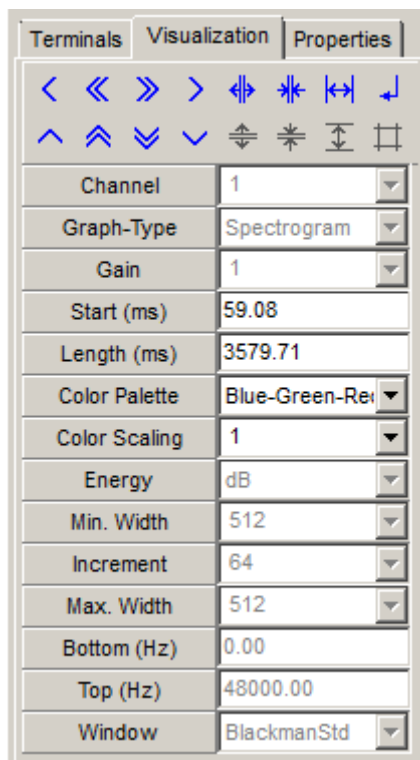


Figure 21: Editor for Visualization properties of Terminals.



Terminals	Visualization	Properties
Local Maxima	All points	
x-Density	1	
y-Density	1	
Minimum	35	
Sensitivity c	2	
Max. x-Distance	2	
Max. y-Distance	2	
Maximum	20	
Sensitivity c	2	
Measure	Euclidean	
Smooth	10	
Compute	Positive & Negative	
Threshold	0.2	
Sensitivity c	1	
Vertical Shift	0	

Figure 22: Editor for Terminal properties

Table 10: Properties of Terminals







Property	Value
Local Maxima	Filter for data points of a Terminal-signature: (1) “All points” means that all data points are used in the signature. (2) “x-maxima” means that only maxima in rows are used. (3) “y-maxima” means that only maxima in columns are used. (4) “xy-maxima” means that only maxima of both rows and columns are used.
x-Density	Integer value n to specify the density of the signature on the x-axis. Only each n-th data point on the x-axis is used for the signature.
y-Density	Integer value m to specify the density of the signature on the y-axis. Only each m-th data point on the y-axis is used for the signature.
Minimum	Minimum threshold for data points so that they are used for the positive sub-signature.
Sensitivity	Sensitivity of the algorithm that computes the distance of the positive sub-signature.
Max. x-Distance	Maximum distance between data points of the positive sub-signature and data points on the x-axis so that they are used for the negative sub-signature.
Max. y-Distance	Maximum distance between data points of the positive sub-signature and data points on the y-axis so that they are used for the negative sub-signature.

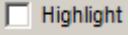
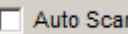
Property	Value
Maximum	Maximum threshold for data points so that they are used for the negative sub-signature.
Sensitivity	Sensitivity of the algorithm that computes the distance of the negative sub-signature.
Measure	Basic similarity measure: (1) Euclidean Distance, (2) Pruned Euclidean Distance (only distances > 0 are considered)
Smooth	Integer value: Smooth the output of the combined distance function.
Signature	(1) Positive: Compute distance only of the positive sub-signature; (2) Positive & Negative: Compute distance of both the positive and negative sub-signatures;
Threshold	Maximum threshold for the combined distance to trigger an annotation.
Sensitivity c	Sensitivity of the algorithm that computes the combined distance.
Vertical shift	Parameter to shift the combined distance along the y-axis (can have a positive or negative value).

## 4.5 Toolbar of the Terminal editor

On top (below the main menu) is a toolbar with several general purpose controls. These controls are explained in Table 11.

Table 11: Controls on the toolbar








Icon	Function
	Apply Visualization properties of the current Terminal to the current Visualization.
	Open the Visualization which the current Terminal was derived from initially.
	Minimize size of both selection frame and search frame of the current Terminal.
	Normalize (set to zero) the start-time of the current Terminal.
	Scan current Visualization with current Terminal. Result is a search record containing Terminal Annotations. This search record is active in memory unless it is deleted by clicking the 'Delete Current Search Records' button in the Visualization editor. Active search records can be added to the Annotation database (see Section 6.3). Items in search records can be mapped graphically (projected) onto Visualizations (see Section 6.4).
	Eraser: Open the eraser thickness choice box. With the right mouse button



Icon	Function
	pressed you can erase positive or negative data points of a signature by moving the mouse cursor. Positive points are erased, if one of the green buttons is pressed. Negative points are erased, if one of the red buttons is pressed. You can choose from thin, middle and thick erasers.
	Signatures can be displayed in highlight mode. In highlight mode, the positive part is green, the negative part is red and the neutral part is grey. Select in the 'Options' window under 'Editors > Highlight Mode' the parts of the signature that are displayed.
	Auto Scan: Check to automatically scan the current Visualization in the Visualization editor. A graph of the distances between the current Terminal signature and the audio data underlying the current Visualization is shown.

## 4.6 Wizards for Terminals

SR-Lab has several wizards to generate and modify Terminals. All wizards can be reached via the main menu under the menu item “Terminals”. Table 12 shows an overview.

Table 12: Wizards for Terminals

Icon		Task
	Export Image of current Terminal	Dialogue to save an image (in *.bmp format) showing the current Terminal. This dialogue is self-explanatory.
	Export Images of Terminals	Wizard for automatic generation of images (in *.bmp format) from Terminals. Files are saved in the default export directory. This wizard is self-explanatory.
	Normalize Terminals	Wizard for normalization of start-time, signature-coefficients and frames of Terminals. This wizard is described in Section 4.6.1.
	Clone Terminals	Wizard for cloning of Terminals along the y-axis. This wizard is described in Section 4.6.2.
	Merge Terminals	Wizard for automatic similarity based merging of Terminals. This wizard is described in Section 4.6.3.
	Extract Terminals	Wizard for extraction of Terminals from Terminal Annotations. This wizard is described in Section 4.6.4.
	Extract Merged Terminals	Wizard for extraction of merged Terminals from Terminal

Icon		Task
		Annotations. No similarity criterion is applied. This wizard is described in Section 4.6.5.
	Scan Current Visualization	Wizard for automatic search of instances of Terminals in the audio data that is underlying the current Visualization. Output is a search record active in memory that contains Terminal Annotations. The search record can be added manually to the Annotation database. This wizard is self-explanatory.
	Scan Audio Files	Wizard for automatic search for instances of Terminals in audio files. Output are data tables in *.csv format containing Terminal Annotations. The tables are automatically added to the Annotation database. This wizard is self-explanatory.

#### 4.6.1 Normalize Terminals wizard

Terminals need to be normalized for the following reasons: (1) Standardization of energy values is useful before merging Terminals, as it compensates for differences in the volume of otherwise similar acoustic processes. (2) Normalization of the time basis and minimization of search and selection frames make standardized neighbourhood relations for the sequencing algorithm possible (see Section 5.6.2); (3) Normalized Terminals use less memory.

Figure 23 shows a Terminal before normalization, Figure 24 shows the same Terminal after normalization.

Table 13: *Normalize Terminals algorithm*

<b>Input</b>	A set T1 of Terminals
<b>Options</b>	<ul style="list-style-type: none"> <li>(1) Normalize Energy [TRUE, FALSE]: Amplification of signatures so that they get standardized maximum energy values</li> <li>(2) Set Time Basis to Zero [TRUE, FALSE]: Setting the start time to zero</li> <li>(3) Minimize Terminals [TRUE, FALSE]: Minimization of search and selection frames</li> </ul>
<b>Output</b>	A set T2 of normalized Terminals

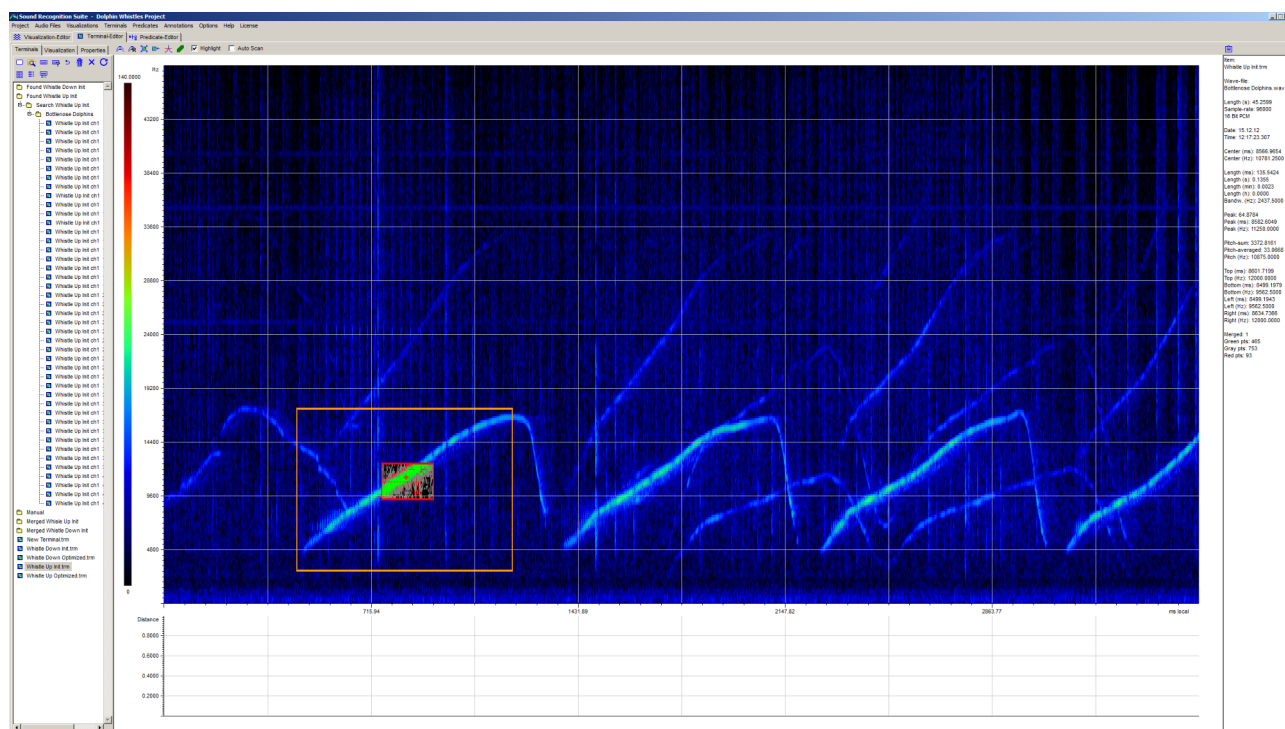


Figure 23: Terminal before normalization (in the background the Visualization matrix is visible)

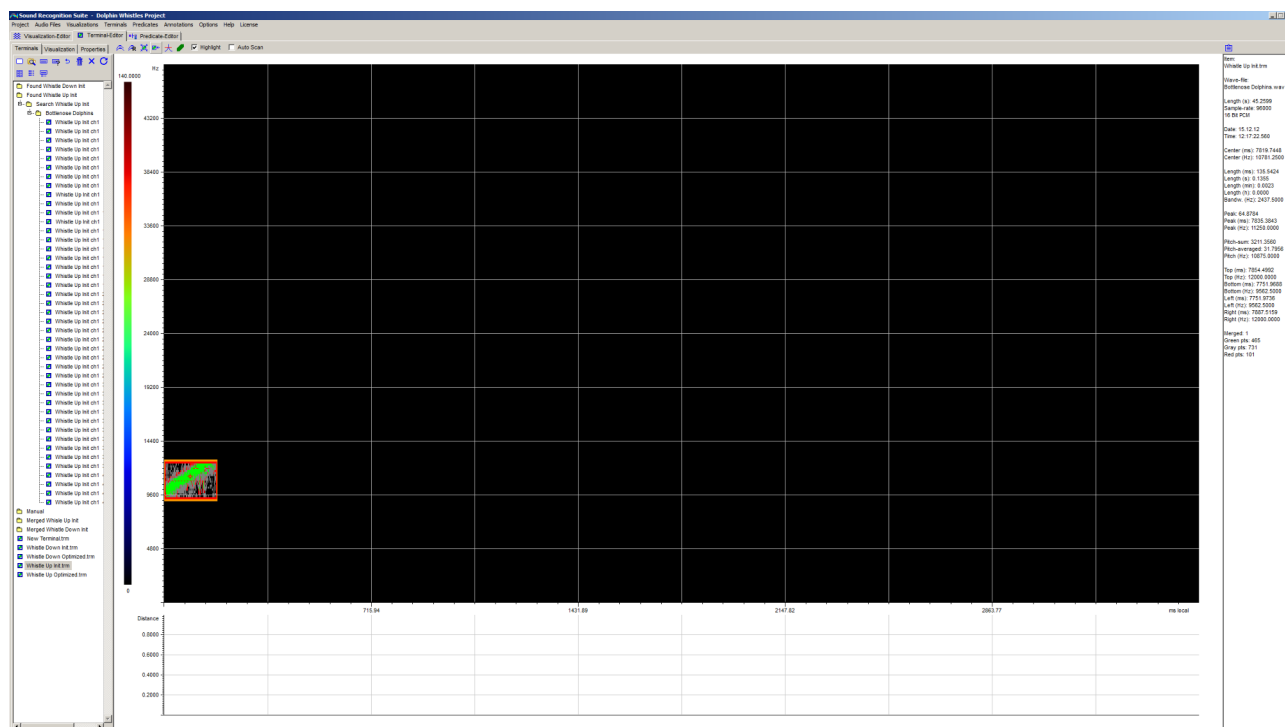


Figure 24: Terminal after normalization. Unnecessary information has been stripped away and the time basis is set to zero.

## 4.6.2 Clone Terminals wizard

Cloning of Terminals along the y-axis is used to generate “swarms” of Terminals that are able to search for acoustic patterns independent from their pitch frequency.

Table 14: Clone Terminals algorithm

<b>Input</b>	A set of Terminals
<b>Options</b>	(1) % Overlap [[Value between 0 and 99] (2) Bottom Frequency [Value in Hz] (3) Top Frequency [Value in Hz]
<b>Output</b>	A set of cloned Terminals

A Terminal is cloned by stepwise shifting its signature along the y-axis within a frequency band. The band is defined by its bottom and top frequencies. The degree of overlap is specified in the overlap parameter in percent. After each shift a new Terminal is generated and saved. All other parameters of the original Terminal remain unchanged.

Figure 25 shows one clone of the Terminal in Figure 24.

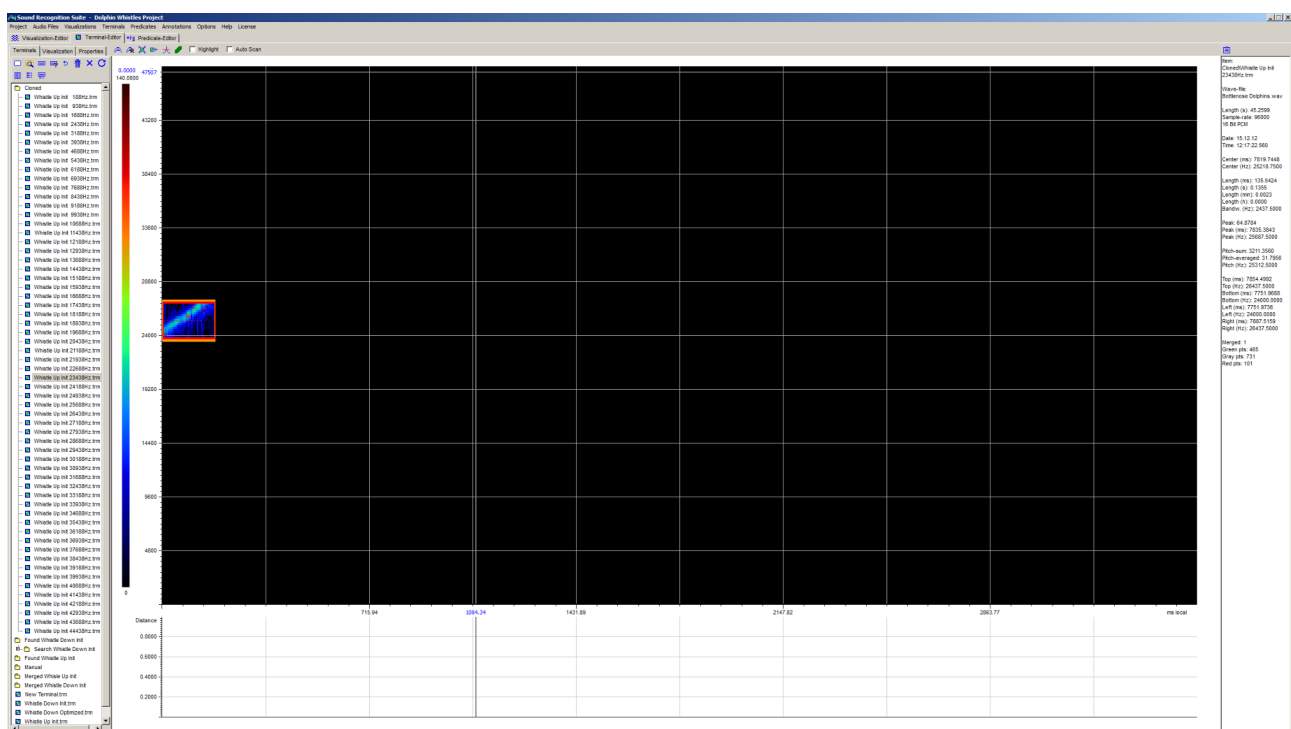


Figure 25: Clone of the Terminal in Figure 24. The Terminal differs from the original one only by the up-shifted signature. At the left, in the tree view a swarm of cloned Terminals is listed.

### 4.6.3 Merge Terminals wizard

Similarity-based merging is an essential technique for automatic optimization of Terminals. It is a way to average out unimportant information. Only important information remains in the signatures of merged Terminals. With the help of this wizard you can merge Terminals with one another by stepwise applying a similarity distance criterion.

*Table 15: Merge Terminals algorithm*

<b>Input</b>	A set T1 of Terminals
<b>Options</b>	<ul style="list-style-type: none"> <li>(1) Save Class Members [TRUE, FALSE]: If TRUE Terminals from T1 used to generate merged Terminals in T2 are saved in separate subfolders. They are called “class members” because the corresponding merged Terminal is a class descriptor for all of them.</li> <li>(2) Min. Similarity and Max Similarity [Value between 0 and 1]: Before merging two Terminals a distance (i.e. similarity) criterion needs to be met by them. The merge algorithm applies this criterion starting at “Min. Distance”, stepwise incrementing (see below) and stopping at “Max. Distance”.</li> <li>(3) Increment [Value between 0 and 1]: Value used for stepwise incrementing the applied distance criterion. For each step a new set of merged Terminals is generated and saved in a separate subfolder.</li> <li>(4) Max. Batch Count [Integer value]: Limit for the number of Terminals in each input batch. If T1 has more elements it is split into batches.</li> </ul>
<b>Output</b>	A set T2 of merged Terminals

Only Terminals that meet a distance criterion can be merged with one another. Each data point in the signature of the merged Terminal is the mean value of all corresponding data points in the signatures of the underlying Terminals. Figure 26 shows a Terminal that was obtained by merging a set of 32 Terminals.

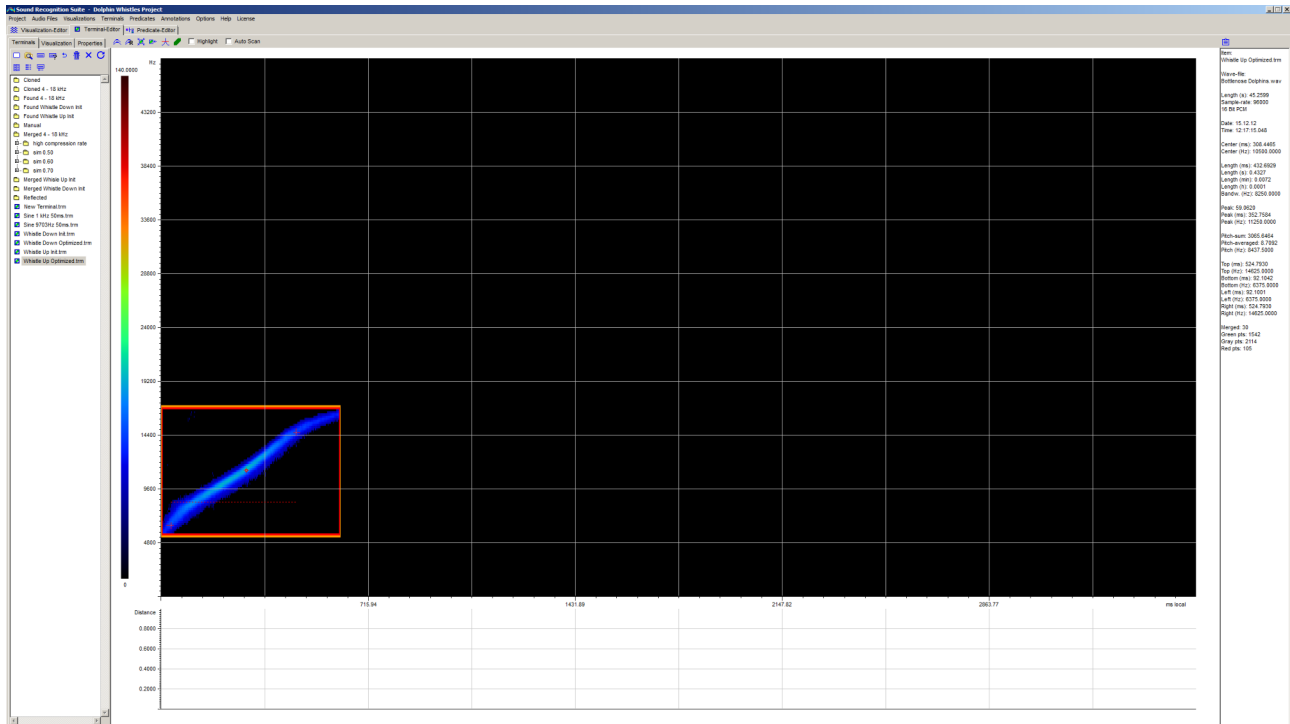


Figure 26: Terminal for a certain whistle type obtained by merging 32 similar Terminals.

#### 4.6.4 Extract Terminals wizard

Automatic extraction of Terminals is an essential technique upon which many further steps are based, e.g. it is used to obtain sets of Terminals for the similarity based merge algorithm (see Section 4.6.3).

A Terminal extracted from an Annotation inherits all properties of the Terminal that generated the Annotation except its signature. The new signature is generated from the audio data marked by the Annotation. Either the selection frame or the search frame of the underlying Terminal is decisive for the audio information to be extracted.

Table 16: Extract Terminals algorithm

<b>Input</b>	A set TA of Terminal Annotations
<b>Options</b>	(1) Expanded [TRUE, FALSE]: If TRUE the orange search frames of the Annotation-generating Terminals are used for audio data extraction. If FALSE the red selection frames of the Terminals are used.
<b>Output</b>	A set T of Terminals extracted from audio data that was marked by the Annotations in TA



Important note: By default the extraction algorithm uses the Terminals that generated the Annotations to extract new Terminals. However, it is possible to replace the original Terminals with other Terminals that have the same names. In this way, Terminals are extracted that are different from those that originally created the Annotations. This can be very useful, because it can reduce computing time in many audio analysis tasks.

Example: Figure 27 shows a Terminal that annotates a certain type of whistle. Figure 28 shows three whistles that were annotated by this Terminal. Figure 29 shows a new Terminal automatically extracted from the central whistle with the “Expanded” option set to TRUE.

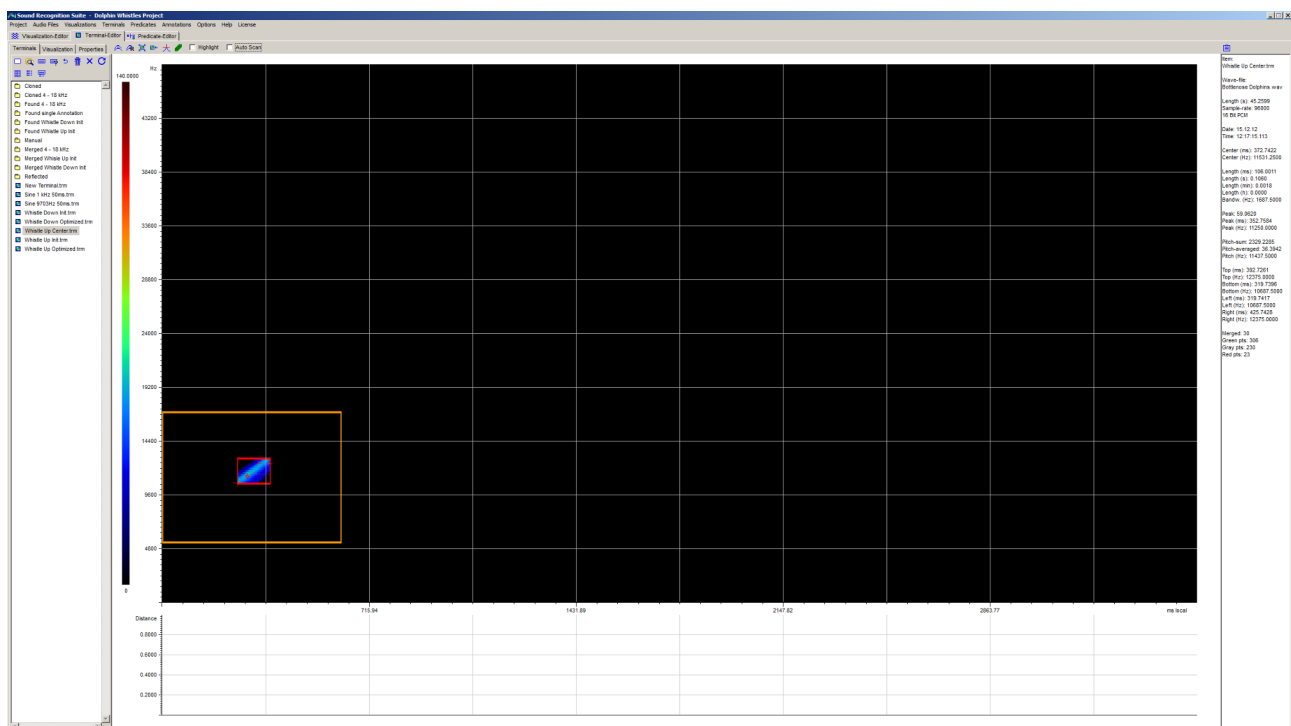


Figure 27: Terminal that annotates whistles with rising frequency. The orange search frame is used for extraction of new Terminals.

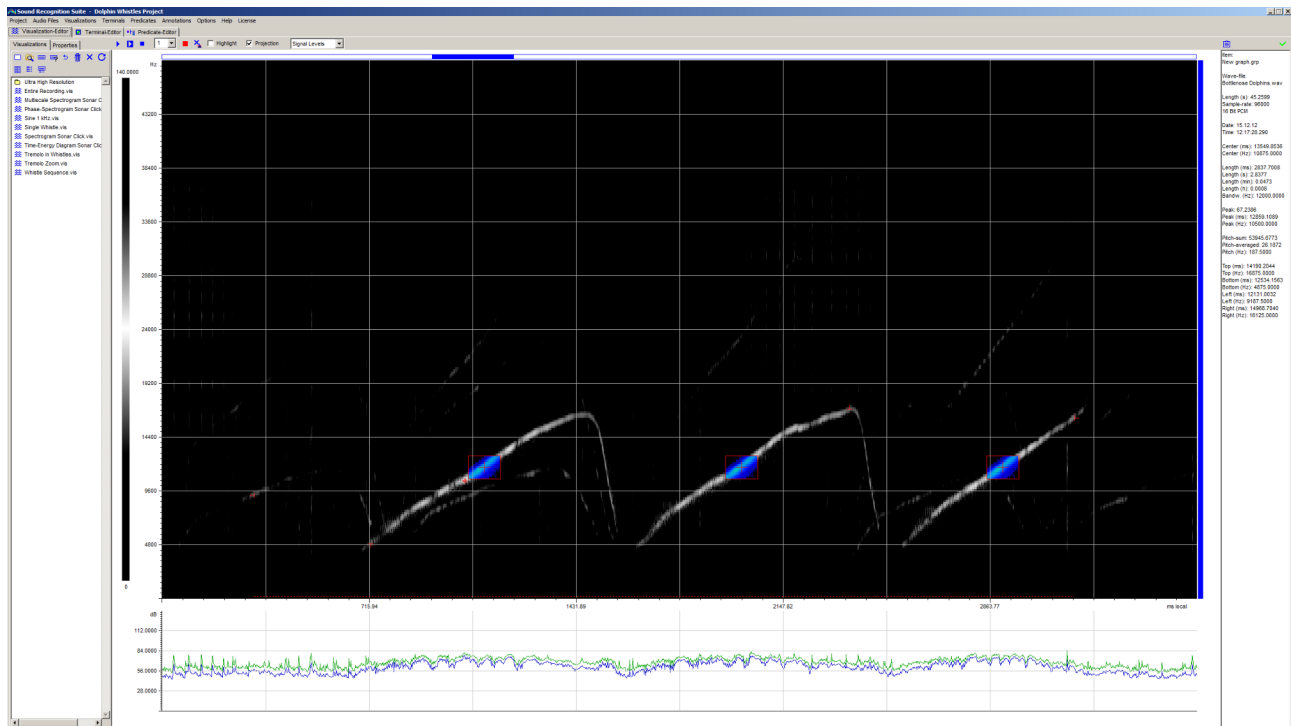


Figure 28: Annotations (blue) generated by the Terminal in Figure 27

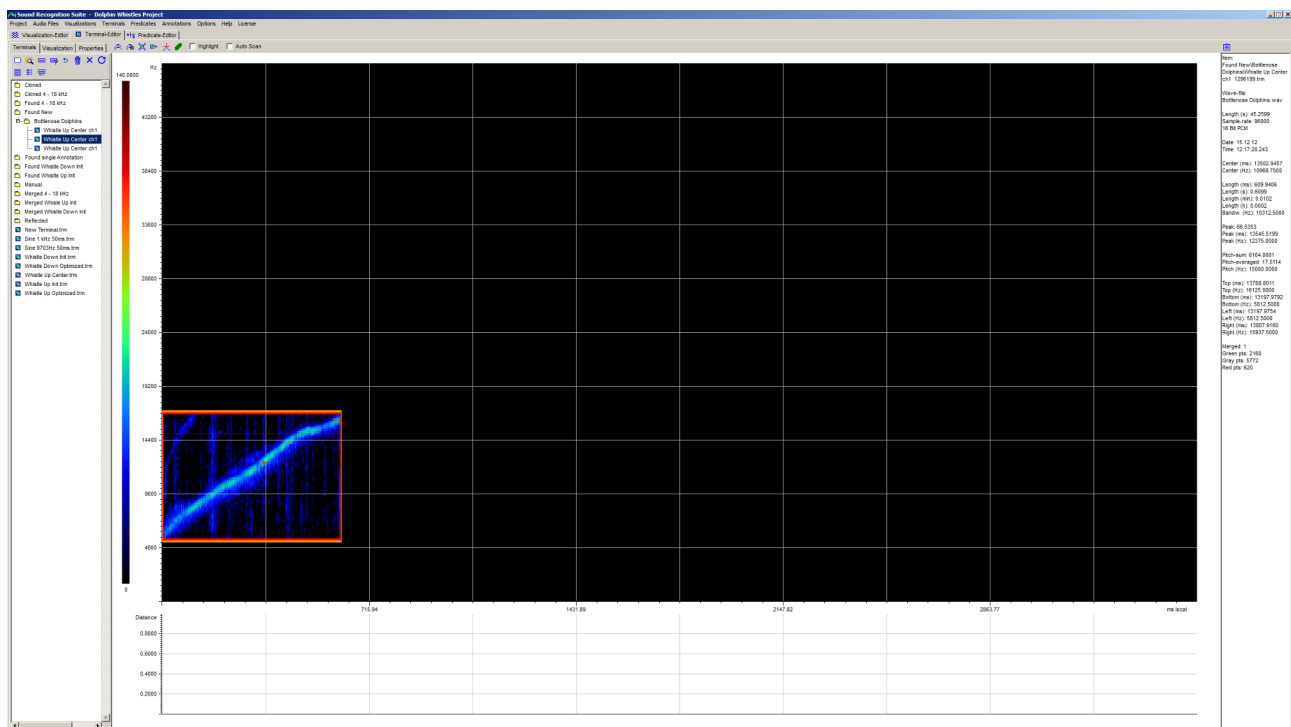


Figure 29: New Terminal extracted from the central Annotation in Figure 28. The selection frame used to extract spectrographic information is expanded and fits the search frame.

#### 4.6.5 Extract Merged Terminals wizard

Merged extraction of Terminals is a method to speed up the process of Terminal optimization. It also helps to save memory on your HDD as no Terminals in intermediate steps need to be saved.

With the help of this wizard you can automatically generate merged Terminals from Terminal Annotations.

Table 17: *Extract Merged Terminals algorithm*

<b>Input</b>	A set TA of Terminal Annotations
<b>Options</b>	(1) The wizard lets you select both the Terminals that are used for merged extraction and the datasets with Annotations.
<b>Output</b>	A set T of merged Terminals extracted from the audio data marked by the Annotations in TA

This wizard unifies the “Extract Terminals” algorithm and the “Merge Terminals” Algorithm described in previous sections. A difference is that no similarity criterion is checked before merging the Terminals. Annotations of one kind (i.e. generated by one Terminal) are extracted and directly merged into one single Terminal.

## 5 Predicates

### 5.1 Predicates

*Predicates* are classifiers (detectors) for complex and abstract acoustic patterns. They can be created, modified and tested in the Predicate editor. Predicate files (\*.prd) are saved under the project path in the "Predicates" subfolder.

A Predicate is derived from a Visualization and inherits all of its properties. Visualization properties however are used only for graphical display. They have no effect on the pattern recognition behaviour of the Predicate. If a Predicate is initialized using the current Visualization, then the Predicate inherits the properties of this Visualization. If it is generated automatically with the help of the sequencing algorithm (see Section 5.6), it inherits the Visualization properties of the first Terminal in the sequence.

A Predicate has two main components:

1. *A symbolic signature*: This is a sequence of Terminals (in symbolic form) that are linked together with logical operators and are arranged in a resolution hierarchy (see Section 5.1.1).
2. *A logical classification algorithm*: This receives a symbolic signature and a piece of audio data as input. In the data, it recognizes acoustic patterns that are similar to the symbolic signature. Output of the algorithm are Predicate Annotations (see Section 5.1.2).

Both components of Predicates can be modified in many ways in SR-Lab. Interactive modelling can be done in the Predicate editor, machine programming can be done with a number of tools provided by SR-Lab (see Section 5.6).

### 5.1.1 Signatures of Predicates

A signature of a Predicate is a sequence of Terminals that are linked together with logical operators (AND and NAND) and can be put into a hierarchical order. The hierarchical order determines in which order Terminals are searched for by the classification algorithm.

Predicate signatures contain only the names and positions of Terminals not the Terminals themselves. For this reason Predicate signatures are called “symbolic signatures”. Its elements are called “symbolic elements”.

After manual initialization the signature is empty and needs to be filled with symbolic elements. This is done with the help of the “Interpret current Visualization” wizard (see Section 5.6.).

In the Predicate editor, symbolic elements are visualized with the help of the associated Terminals. Three different color palettes are available for this purpose: (1) a palette identical with the palette of the Terminals, (2) a palette which indicates the level of each element in the resolution hierarchy and (3) a palette which indicates the compression rate of each symbolic element (applies only to signatures obtained by merging Predicates).

In the Predicate editor it is possible to interactively model symbolic signatures. For example, it is possible to manually select elements, connect them with logical operators and to set up a resolution hierarchy (see Section 5.4).

### 5.1.2 The logical classification algorithm

SR-Lab has a built in classification algorithm to find patterns in audio data that are similar to Predicate signatures. It is a logical hierarchical resolution algorithm, which instantiates symbolic elements during search on demand: First, the algorithm searches the audio data for the first element in the hierarchy. If it is found, it searches for the second element. If this is also found, it searches for the third element, and so on.

Actually, the algorithm searches for patterns of Terminal Annotations in audio data that match the logical structure of the symbolic signature. For each point in time of the audio data it computes a Boolean result (TRUE or FALSE). A Predicate Annotation is generated whenever the result is TRUE.

A Predicate has two tolerance parameters: time tolerance and a discrete item tolerance. Time tolerance relates to deviations in the temporal distance between symbolic elements. The discrete item tolerance refers to the number of symbolic elements that cannot be instantiated in the search but is tolerated by the algorithm.

All parameters relevant to the behaviour of the algorithm can be manually changed in the Predicate property editor (see Section 5.4). The behaviour of the algorithm, i.e. how it responds to audio data, can be made visible in both Visualization and Predicate editors (see Figure 30).

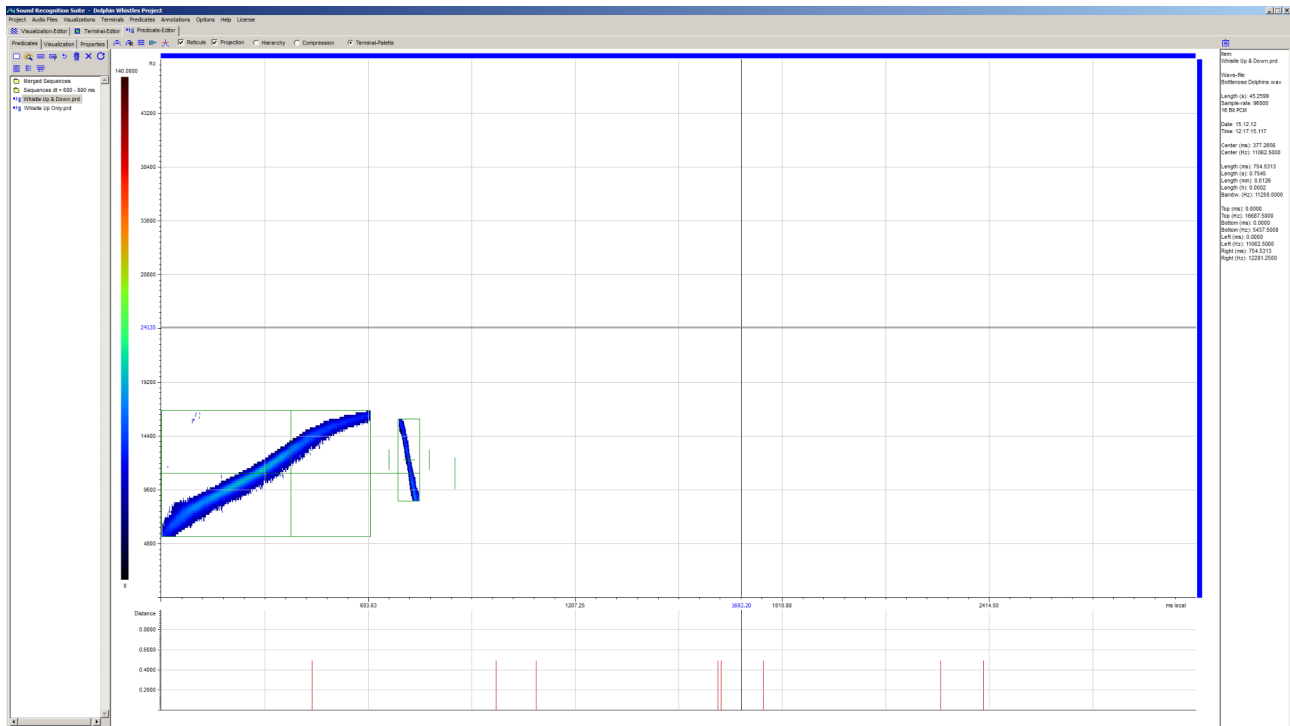


Figure 30: Predicate for a certain type of whistles. The symbolic signature has two elements: (1) A Terminal for the rising frequency and (2) a Terminal for the dropping frequency. Below the Predicate a graph is shown. It displays the points in time at which symbolic elements were instantiated (found) in the audio data underlying the current Visualization (see 13). Each symbolic element is surrounded by a green rectangle and a cross. Vertical lines mark the time tolerance of each element.

## 5.2 Layout of the Predicate editor

With the help of the Predicate editor you can interactively model Predicates. The editor covers design, implementation and test of Predicates. In particular, you can:

1. Initialize Predicates
2. Create, modify and optimize symbolic signatures
3. Modify and optimize the logical classification algorithm
4. Test Predicates in real world tasks
5. Perform basic measurements with regard to Predicate signatures

The layout of the Predicate editor is shown in Figure 30. The workspace is subdivided into five different areas:

- 

Figure 31: Color coded representation of compression rates of symbolic elements. Green indicates that  $> 15$  symbolic elements were merged at this point in time and within precisely this distance to the neighbouring element.

## 5.3 The Predicate manager

With the help of the Predicate manager you can manage all Predicate files (\*.prd) of your project. A tree view of the files is displayed by the manager (see Figure 32). Predicates can be opened by clicking the entries of the tree view. Press the caps-lock key to browse the Predicates with the arrow keys of your keyboard. The buttons of the toolbar are explained in Table 18.

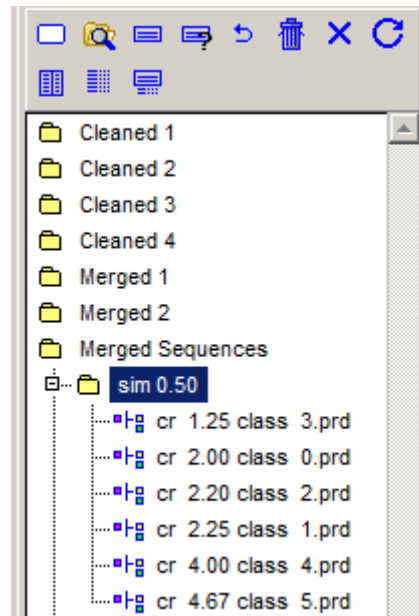













Figure 32: Predicate manager with tree view

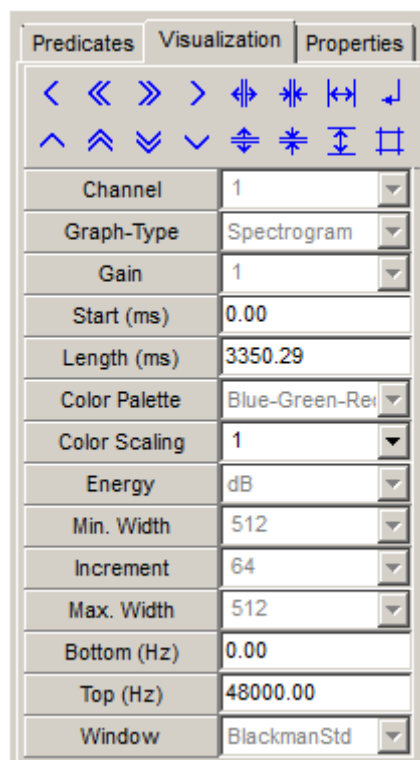
Table 18: Buttons of the Predicate manager toolbar

Icon	Function
	Create a new Predicate. The new Predicate is initialized by deriving it from the current Visualization. After initialization the signature is empty.
	Open Windows-Explorer in the Predicates subfolder of the current project. Within the Predicates subfolder you can create new subfolders and rename or move all files.
	Save current Predicate.
	Rename and save current Predicate.
	Undo changes (revert to saved).
	Delete selected Predicate file or selected subfolder.
	Close current Predicate.
	Open a wizard to choose profiles and apply them to selected Predicates.

Icon	Function
	Save general properties of the current Predicate in a profile file.
	Open a profile file and apply it to the current Predicate.
	Refresh tree view after file operations.

## 5.4 Properties of Predicates

A Predicate is an object that is derived from a Visualization. It inherits all properties of the Visualization and receives a number of new properties. Some of the Visualization properties can still be changed in a property editor under the Visualization tab (Figure 33). The new properties can be adjusted in a separate editor under the Properties tab (Figure 34). All Predicate properties are explained in Table 19.



*Figure 33: Visualization properties of a Predicate*

Note that it is possible to change time and frequency properties of Predicates without switching back to the Visualization editor. Use the Visualization properties editor of the Predicate or the arrow keys and the Ctrl-key (see Section 3.6) to do so.

The Predicate property editor is divided into four sections:



- (1) Tolerance properties that are valid for all elements of the Predicate
- (2) A toolbar for selecting symbolic elements and for editing the hierarchy of the signature
- (3) Properties of single selected symbolic elements
- (4) A tree view in which the resolution hierarchy of the current Predicate is shown and single symbolic elements can be selected as child or parent (Figure 34).

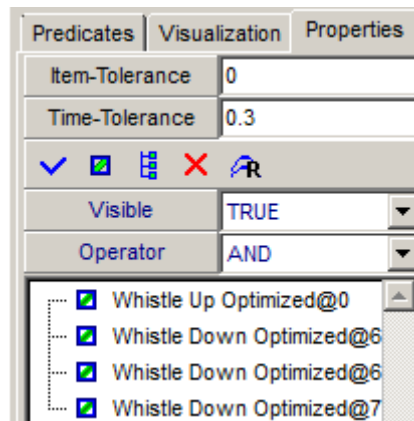


Figure 34: Predicate property editor

Table 19: Predicate property editor

Property	Value
Item-Tolerance	Maximum number of non instantiable elements of the signature that will be tolerated by the classification algorithm.
Time-Tolerance	Maximum temporal deviation of elements that is tolerated by the classification algorithm. The deviation parameter applies to all elements in the signature individually. It is computed by multiplying the length of symbolic elements with the value in the edit-field.
Icon	Function
✓	Accept the current selection and a create parent-child relationship between selected elements. The resulting hierarchy is shown in the tree view below and can be highlighted in the editor.
■	Unselect all symbolic elements.
☐	Break up all parent-child relationships in the signature to flatten the hierarchy.
✗	Delete selected symbolic elements.
⌘	Open the Terminal (in the Terminal editor) linked to the selected symbolic element.

Property	Value
Visible	Visibility of selected symbolic element (TRUE or FALSE).
Operator	Operator for connecting the selected symbolic element with the previous element in the signature (AND or NAND).

The resolution hierarchy can be displayed color coded in the editor (Figure 35). It has to be set up manually. In order to do so, select symbolic elements in the tree view to create parent-child relationships between them: Left click selects an element as parent, right click selects an element as child. It is possible to select one parent and a number of child-elements simultaneously. To save the relationship click the 'Accept' button in the property editor (see above).

Note that you have to finish all changes by clicking the “Save” button in the Predicate manager.








Figure 35: Color coded representation of the resolution hierarchy of a two-part symbolic signature. Black is hierarchy level 0 and red is level 1. The black part is searched for first. The red part is only searched for when the first part was found. In this way, search processes can be accelerated considerably.

## 5.5 Toolbar of the Predicate editor

On top (below the main menu) is a toolbar with several general purpose controls. These controls are explained in Table 20.


Table 20: Controls on the toolbar of the Predicate editor





Icon	Function
	Apply Predicate properties to Visualization: Apply settings (color palette, time- and bandwidth-related parameters etc.) of the current Predicate to the current Visualization.
	Goto Root: Open the Visualization which the current Predicate was derived from (not always applicable).
	Apply settings of the current Visualization to the current Predicate: Done automatically if 'Apply Current Visualization to Predicates' is checked in the Options window.
	Normalize: Set the start time of the current Predicate to zero.
	Scan Screen: Scan the current Visualization with the current Predicate. Result is a search record containing Predicate Annotations. This record is active in memory unless it is deleted by clicking the 'Delete Current Search Records' button in the Visualization editor. Active records can be saved in data tables (see Section 6).
<input checked="" type="checkbox"/> Reticule	Reticule: Show/hide reticules indicating the position of symbolic elements and of the entire signature.
<input checked="" type="checkbox"/> Projection	Projection: Show/hide Terminal signatures of symbolic elements.
<input checked="" type="checkbox"/> Hierarchy	Hierarchy: Color coded highlighting of the resolution hierarchy.
<input type="checkbox"/> Compression	Compression: Color coded highlighting of the compression rate of symbolic elements.
<input type="checkbox"/> Terminal-Palette	Terminal-Palette: Use color palette of Terminals to visualize symbolic elements.

## 5.6 Wizards for Predicates

SR-Lab has several wizards to generate and modify Predicates. All wizards can be reached via the main menu under the menu item “Predicates”. Table 21 shows an overview.

Table 21: Wizards for Predicates

Icon	Menu item	Task
	Export Image of current Predicate	Dialogue to save an image (in *.bmp format) showing the current Predicate. This dialogue is self-explanatory.

	Export Images of Predicates	Wizard for automatic generation of images (in *.bmp format) from Predicates. Files are saved in the default export directory. This wizard is self-explanatory.
	Interpret Current Visualization	Wizard for automatic creation of a Predicate signature by means of interpretation of the current Visualization with selected Terminals. This wizard is described in Section 5.6.1.
	Find Sequences	Wizard for automatic sequencing of Terminal Annotations to generate new Predicates. This wizard is described in Section 5.6.2
	Merge Predicates	Wizard for automatic similarity based merging of Predicates. This wizard is described in Section 5.6.3.
	Clean up Predicates	Wizard for automatic cleaning Predicates from superfluous symbolic elements. This wizard is described in Section 5.6.4.
	Extract Predicates	Wizard for automatic extraction of new Predicates from Predicate Annotations. This wizard is described in Section 5.6.5.
	Extract Merged Predicates	Wizard for automatic generation of merged Predicates from Predicate Annotations. This wizard is described in Section 5.6.6.
	Scan Current Visualization	Wizard to search for instances of Predicates in the audio data underlying the current Visualization. Output is a search record active in memory that contains Predicate Annotations. The search record can be added manually to the Annotation database. This wizard is self-explanatory.
	Scan Audio Files	Wizard to search for instances of Predicates in audio files. Output are data tables in *.csv format containing Predicate Annotations. The tables are automatically added to the Annotation database. This wizard is self-explanatory.

### 5.6.1 Interpret Current Visualization wizard

Interpreting the current Visualization with the help of selected Terminals is a basic method to create symbolic signatures. Interpreting means that a piece of audio data is interpreted as a sequence of acoustic events each of which is classified by a Terminal and represented by a Terminal Annotation. The sequence of Terminal Annotations is then used to form the signature of the Predicate.

By means of the interpretation algorithm a fully functional Predicate is generated. If necessary it can be optimized in further steps.

Table 22: Interpret current Visualization algorithm

<b>Input</b>	Current Visualization, current Predicate P, a set T of Terminals
<b>Options</b>	- none -
<b>Output</b>	A Predicate P" with symbolic signature formed by the Annotations of the Terminals in T

The wizard requires: (1) An open Visualization in the Visualization editor, (2) an open Predicate in the Predicate editor (a new Predicate can be derived from the current Visualization by clicking the “Create New Object” button in the toolbar of the Predicate manager) and (3) selected Terminals that are able to respond properly to the acoustic patterns displayed by the Visualization. By default Predicate signatures are created by interpreting the entire Visualization. If an area inside the Visualization is selected the interpretation is created by interpreting only this selection.

Figure 36 shows a Predicate obtained by interpreting a sequence of whistles (shown in Figure 13) with a swarm of simple Terminals that respond to short sine wave patterns.

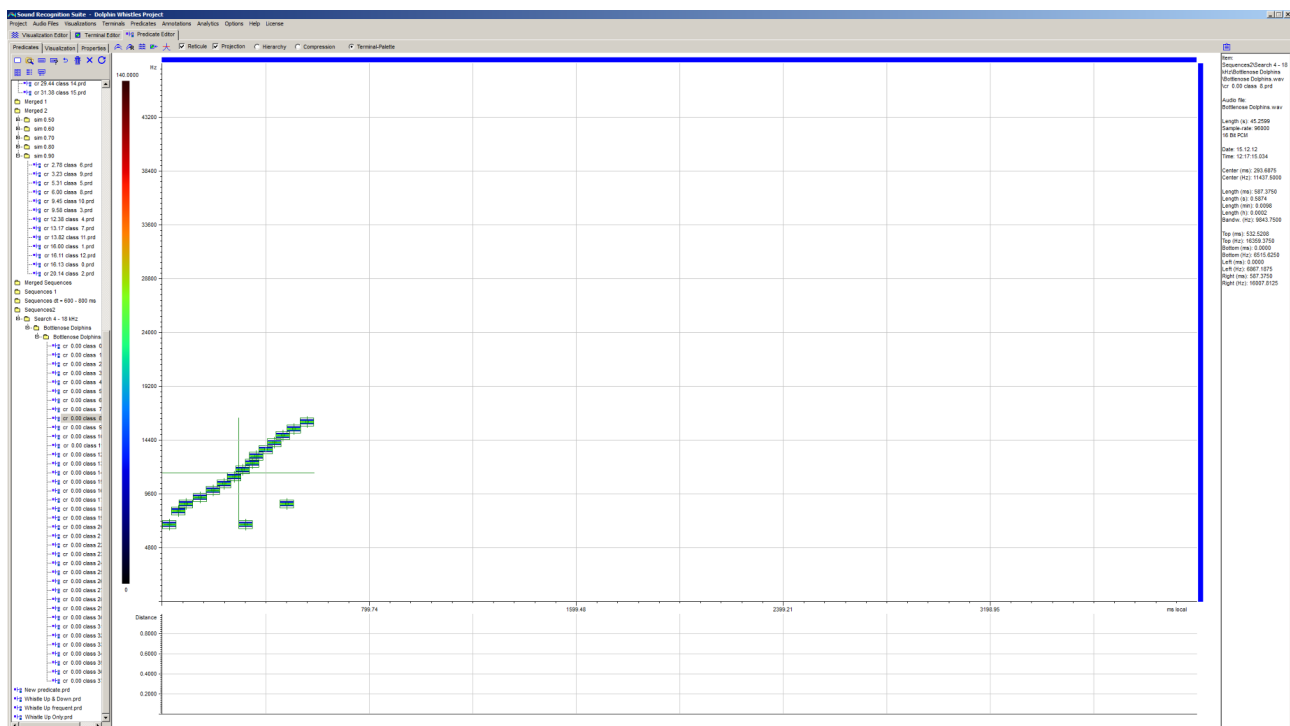


Figure 36: Predicate obtained by interpreting a whistle with a swarm of Terminals that respond to short sine wave patterns.

### 5.6.2 Find Sequences wizard

The sequencing algorithm automates the interpret routine from Section 5.6.1. In sets of Terminal Annotations it searches for sequences or tuples of Annotations that match a predefined adjacency relation.

Figure 39 shows a series whistles annotated by a swarm of Terminals. Figure 38 shows one of the Predicates obtained by sequencing these Annotations.

*Table 23: Sequencing algorithm*

<b>Input</b>	A set TA of Terminal Annotations
<b>Options</b>	<ul style="list-style-type: none"> <li>(1) Create Predicate Annotations [TRUE, FALSE]: If TRUE for each found sequence a Predicate Annotation is generated and added to the database.</li> <li>(2) Create Predicates [TRUE, FALSE]: If TRUE for each found sequence a Predicate is generated and added to the Predicates subfolder.</li> <li>(3) Sequencer [Tuple, Chain]: Two different methods to find sequences. If Tuple is selected simultaneous Annotations are incorporated into the sequences. If Chain is selected only successive Annotations are incorporated into the sequences.</li> <li>(4) df max (Hz) [Integer value]: Maximal allowed distance on the y-axis between two Annotations</li> <li>(5) df min(Hz) [Integer value]: Minimal allowed distance on the y-axis between two Annotations</li> <li>(6) dt max (ms) [Integer value]: Maximal allowed distance on the time axis between two Annotations</li> <li>(7) dt min (ms) [Integer value]: Minimal allowed distance on the time axis between two Annotations</li> <li>(8) Length min [Integer value]: Minimal number of elements required to form a valid sequence</li> </ul>
<b>Output</b>	A set P of Predicates with signatures representing sequences or tuples of Annotations in TA that match the adjacency criterion

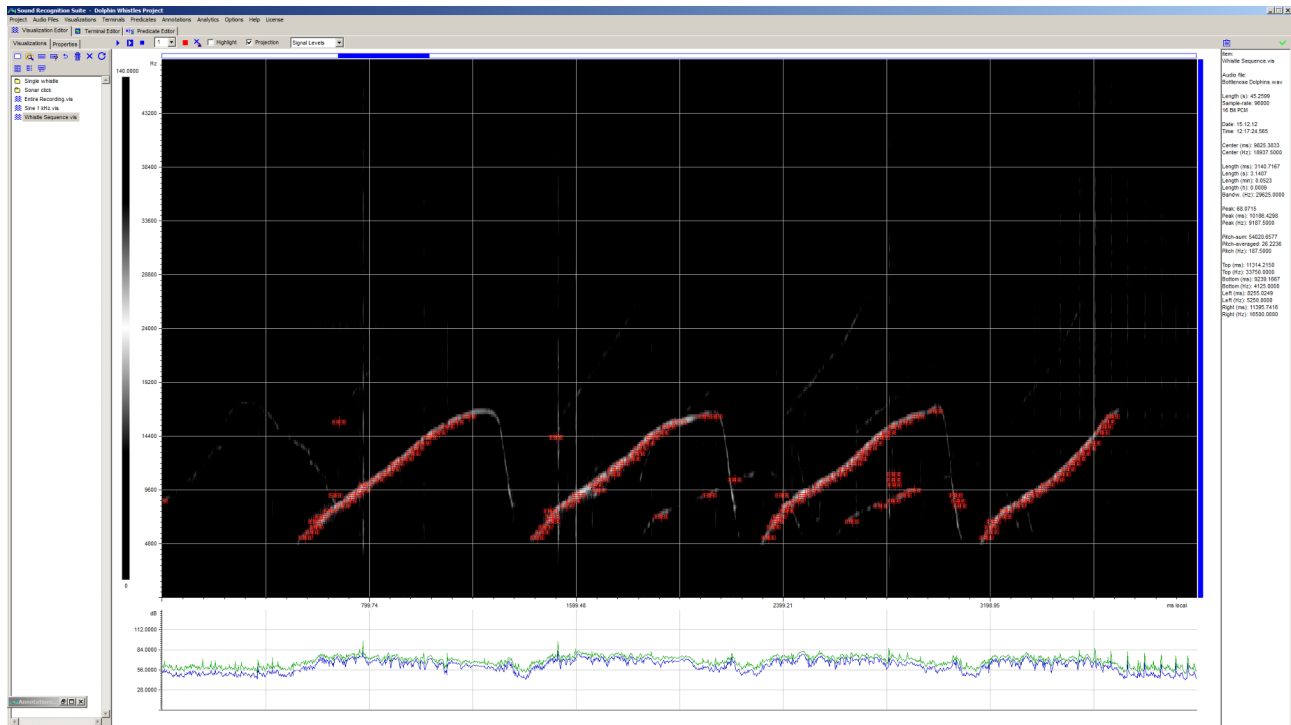


Figure 37: Whistles annotated by Terminals responding to short sine wave patterns. Each Annotation is marked by a small rectangle and a picture of the Terminal signature.

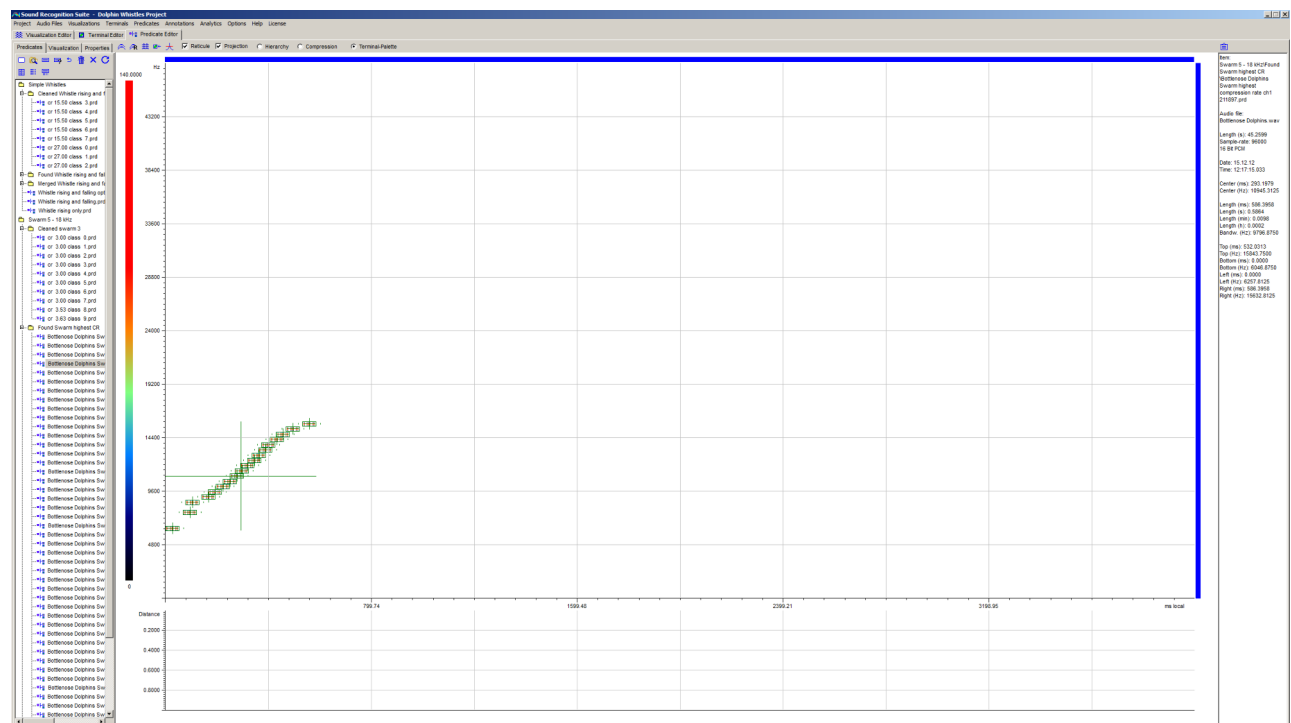


Figure 38: Predicate generated by the sequencing algorithm. The signature represents one valid sequence in the set of Annotations shown in Figure 37.

### 5.6.3 Merge Predicates wizard

Similarity based merging is an essential technique for automatic optimization of Predicates. It is a way to distinguish between important and unimportant symbolic elements in the signature. Important are elements with a high compression rate, i.e. common to many Predicates. Unimportant are elements that have a low compression rate. The compression rate (or “number of merged” rate) of a symbolic element is the number of symbolic elements from other Predicates that were merged together to form the element.

With the help of the merge Predicates wizard you can merge Predicates that match a similarity criterion with one another.

*Table 24: Merge Predicates algorithm*

<b>Input</b>	A set P1 of Predicates
<b>Options</b>	<ol style="list-style-type: none"> <li>(1) Save Class Members [TRUE, FALSE]: If TRUE Predicates from P1 used to generate merged Predicates in P2 are saved in a separate subfolders. They are called “Class Members” because the corresponding merged Predicates are class descriptors for them.</li> <li>(2) Min. Similarity and Max Similarity [Value between 0 and 1]: Before merging two Predicates a similarity or distance criterion needs to be met by them. The merge algorithm applies this criterion starting at “Min. Distance”, stepwise incrementing and stopping at “Max. Distance”.</li> <li>(3) Increment [Value between 0 and 1]: Value used for stepwise incrementing the applied distance criterion. For each step a new set of merged Predicates is generated and saved in a separate subfolder.</li> <li>(4) Max. Batch Count [Integer value]: Limit for the number of Predicates in each input batch. If P1 has more elements than this value it is split into batches.</li> <li>(5) Time Tolerance [Value between 0 and 1]: Value used for calculating the maximum tolerated time deviation of symbolic elements before merging them. The tolerance is calculated by multiplying the length of the elements by this value. For example, if an element has a length of 1 s and the value is 0.5 the tolerated time deviation is <math>1 \text{ s} * 0.5 = 0.5 \text{ s}</math>.</li> </ol>
<b>Output</b>	A set P2 of merged Predicates

Predicates in P1 need to meet a similarity criterion in order to be merged with one another. If this is the case, a new Predicate is created by merging their signatures. Each element in the signature of the new Predicate gets the mean temporal position of all corresponding elements in the signatures of the underlying Predicates.



Figure 39 shows a Predicate that was obtained by merging 20 Predicates. These were generated by the sequencing algorithm (see Section 5.6.2).

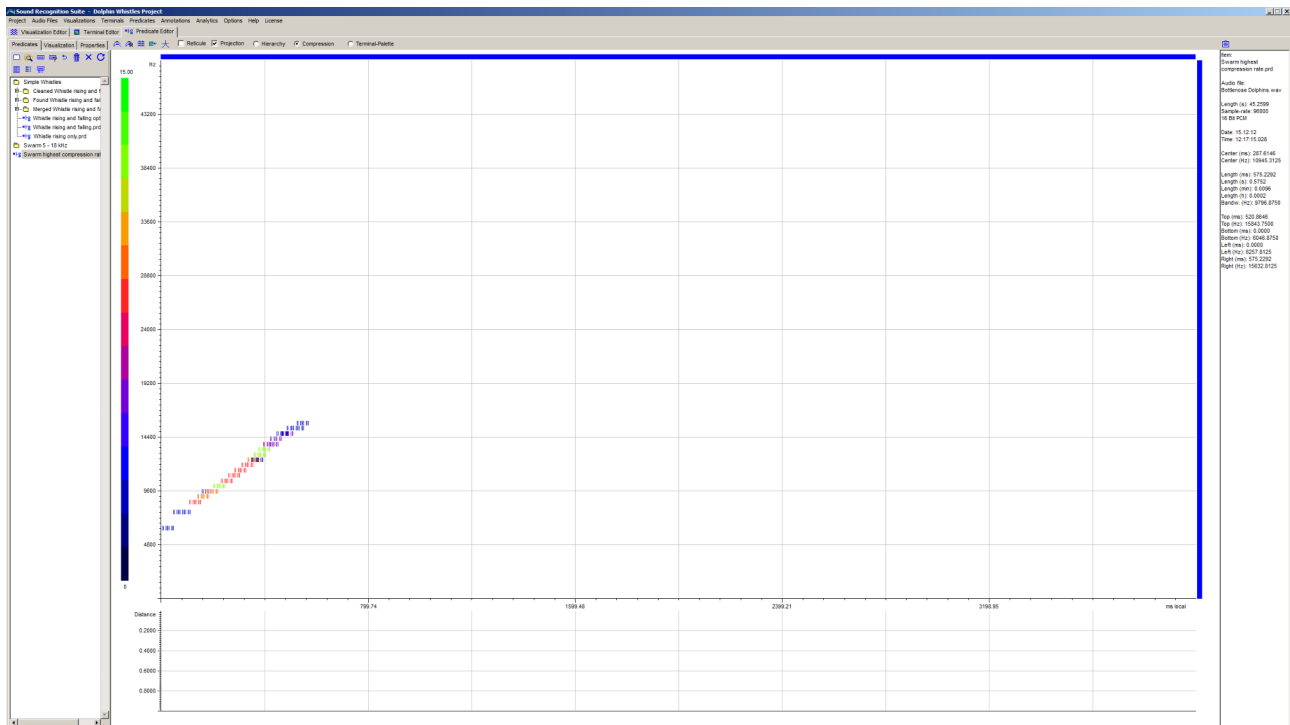


Figure 39: Predicate obtained by merging 20 Predicates generated by the sequencing algorithm. The compression rate of symbolic elements is color coded.

#### 5.6.4 Clean up Predicates wizard

Cleaning up symbolic signature of Predicates is an essential technique to optimize previously merged Predicates (see Section 5.6.3). Symbolic elements with low compression rate are automatically removed from the signature thus getting rid of superfluous pieces of information.

By means of the clean up algorithm more accurate and faster Predicates are obtained. If necessary, the clean up process can be repeated several times.

Table 25: Clean up Predicates algorithm

<b>Input</b>	A set P of Predicates
<b>Options</b>	(1) Min. Number of Merged [Integer]: The minimum compression rate (in terms of the number of merged elements) that an element of the signature must have in order to remain in the signature.
<b>Output</b>	A set P" of Predicates with cleaned up symbolic signatures

Figure 40 shows the Predicate from Figure 39 after cleaning up its signature. The cleaned Predicate has fewer symbolic elements and a better overall compression rate. It works faster and more accurate.

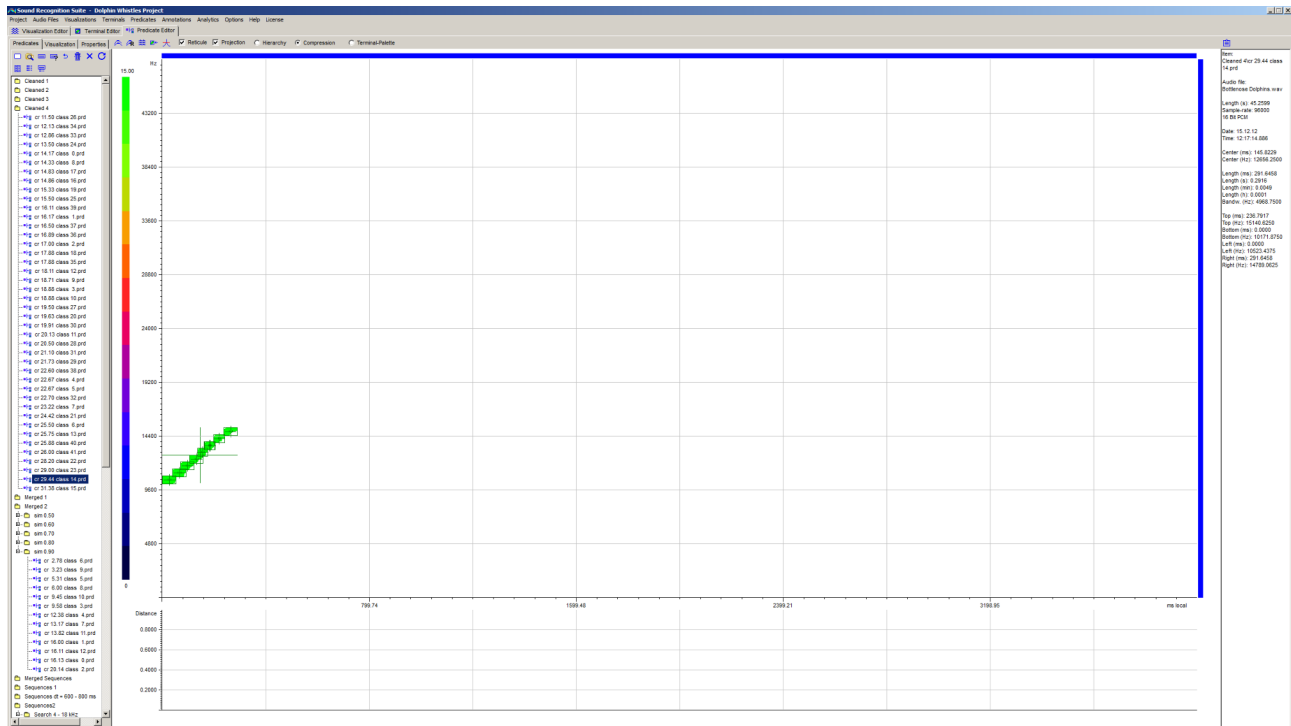


Figure 40: Predicate after cleaning its signature from elements with low compression rate.

### 5.6.5 Extract Predicates wizard

Automatic extraction of Predicates from Predicate Annotations is an essential technique upon which many further steps are based. Extracted Predicates can be used as input for the merge algorithm as well as for audio analytics.

Table 26: Extract Predicates algorithm

<b>Input</b>	A set PA of Predicate Annotations
<b>Options</b>	- none -
<b>Output</b>	A set P of Predicates extracted from the audio data marked by the Annotations in PA

A Predicate extracted from a Predicate Annotation inherits all properties of the Predicate that generated the Annotation except its signature. The new signature is generated from the audio data marked by the Annotation. It differs from the original one in time related properties of symbolic elements.

**Important note:** By default the algorithm uses the Predicates that generated the Annotations in order to extract new Predicates. Before starting the algorithm the original Predicates can be *replaced* by different Predicates with the same names. In this way, Predicates are extracted that are different from those that originally created the Annotations. This can significantly reduce computing time in many audio analysis tasks.

### 5.6.6 Extract Merged Predicates wizard

Merged extraction of Predicates is a method to speed up the process of Predicate optimization. With the help of this wizard you can automatically generate merged Predicates from Predicate Annotations. No similarity criterion is applied before merging Predicates.

*Table 27: Extract merged Predicates algorithm*

<b>Input</b>	A set PA of Predicate Annotations
<b>Options</b>	(1) The wizard lets you select both the Predicates that are used for merged extraction and the datasets with the Annotations.
<b>Output</b>	A set P of merged Predicates extracted from the audio data marked by the Annotations in PA

This wizard unifies the “Extract Predicates” algorithm and the “Merge Predicates” Algorithm described in previous sections. A difference is that no similarity criterion is applied before merging Predicates. Annotations of one kind (i.e. generated by one and the same Predicate) are extracted and directly merged into one new Predicate.

## 6 Annotations

### 6.1 Types of Annotations

Annotations are markings of specific patterns in audio data. They are the result of a search process in which audio data is automatically classified by Terminals or Predicates. Annotations can also be manually created.

SR-Lab distinguishes between two types of Annotations:

1. **Terminal Annotations:** These are Annotations generated by Terminals. Terminal Annotation data tables (in \*.csv format) are saved under the project path in the "Annotations/Terminal Annotations/" subfolder. Terminal Annotations have the following form: [Soundfile, Classifier, Distance, Left Sample, Width in Samples, Top Hz, Height in Hz]. Note that time information is stored as samples from beginning.

2. Predicate Annotations: These are Annotations generated by Predicates. Predicate Annotation data tables (in \*.csv format) are saved under the project path in the "Annotations/ Predicate Annotations/" subfolder. Predicate Annotations have the following form: [Soundfile, Classifier, Distance, Left Sample, Width in Samples, Top Hz, Height in Hz, Constituents]. Note that "Constituents" is a list of Terminal Annotations which is for internal use only.

## 6.2 The Annotation database interface

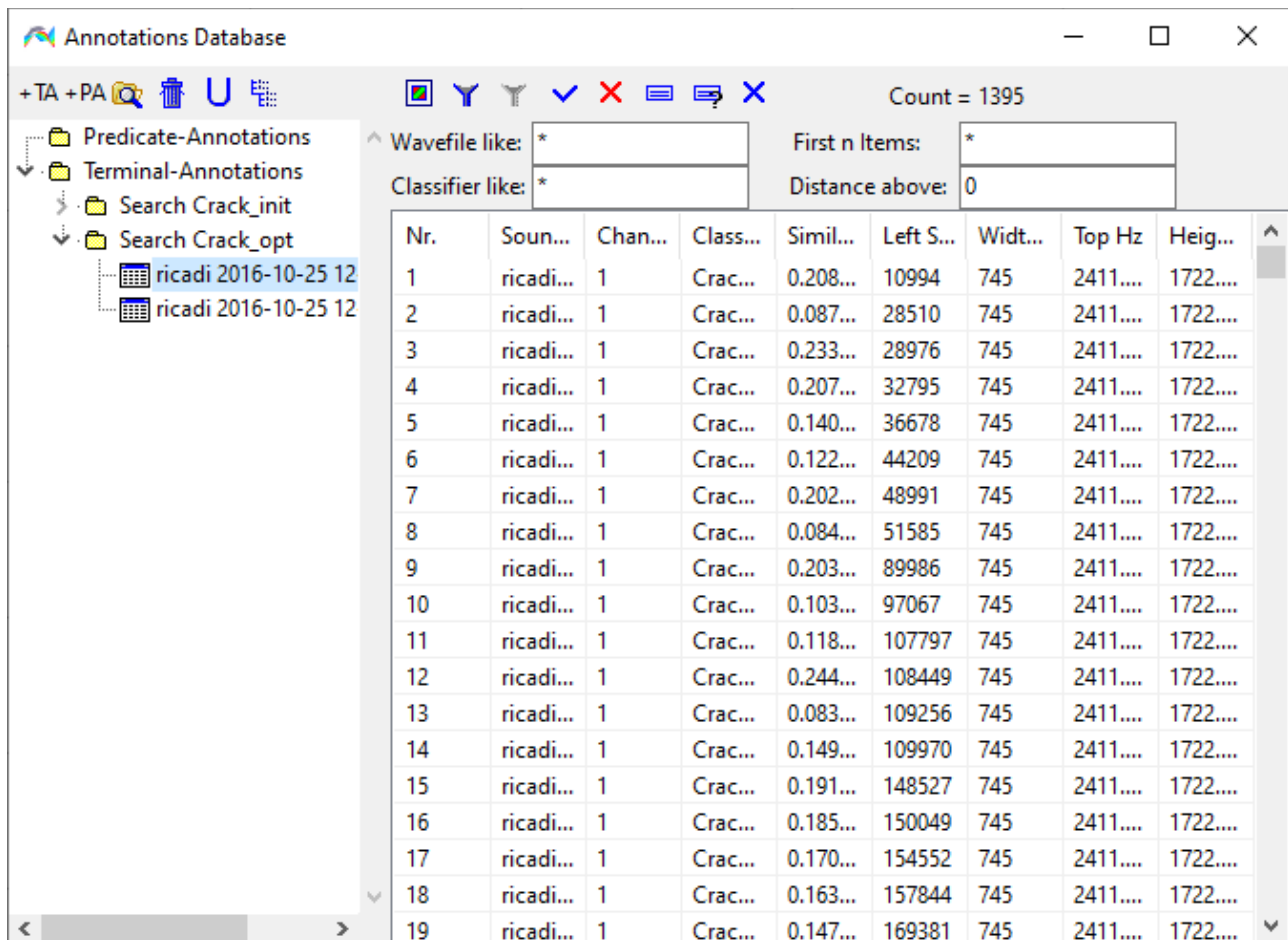


Figure 41: Annotation database interface

With the help of the Annotation database interface you can interactively work with sets of Annotations. The interface covers file management, filtering, sorting and browsing of Annotations. You can:

1. Manually add tables with Annotations
2. Access all tables with Annotations in your project
3. Manage files in the database
4. Sort and filter tables containing Annotations
5. Browse Annotations individually

The layout of the Annotation database interface is shown in 41. The workspace is subdivided into two different areas:

1. On the left is the Annotation manager. It includes a tree view with the data tables of the project and a small toolbar.
2. On the right the currently active Annotation table is shown. Above this table there are edit fields that allow you to configure a filter. On top is a toolbar with several general purpose controls (see Section 6.5).

## 6.3 The Annotation manager

With the help of the Annotation manager you can manage all Annotation files of your project. A tree view of the files is displayed by the manager. Tables can be opened by double-clicking the entries of the tree view. On top is a toolbar with several general purpose controls. These controls are explained in Table 28.

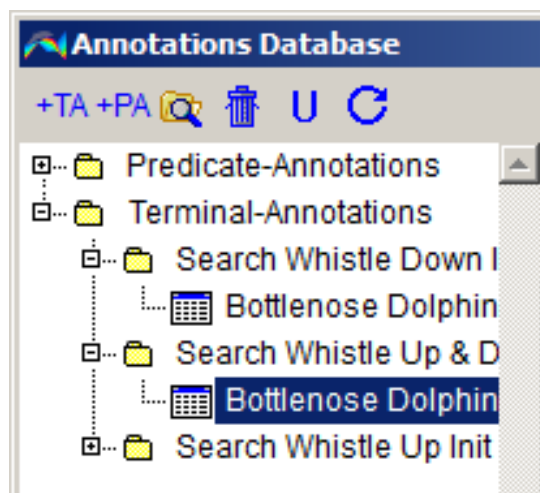


Figure 42: The Annotation manager

Table 28: Buttons of the Annotation manager toolbar

Icon	Function
+TA	Add currently active Terminal search record to database.
+PA	Add currently active Predicate search record to database.
	Open Windows-Explorer to rearrange files in the database.
	Delete selected table or folder (including subfolders).
U	Starts a wizard to select and unite of data tables.

## 6.4 Visualization of Annotations

In order to visualize Annotations they have to be *activated*. This can be done in two different ways:


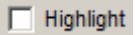
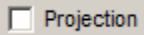
1. Single Annotations can be activated by clicking rows in the table or by using the up- and down arrow keys of the keyboard. An active Annotation can be shown in the Visualization editor (see 43)
2. Entire tables with Annotations can be activated by clicking the “Activate Table” button in the Annotation database interface. This activates the current table (see 44). Annotated audio data then can be browsed without having to activate each Annotation individually.

In the Visualization editor, active Annotations can be displayed in two different ways:

1. The annotated area is marked with a crosshair and a rectangle.
2. The signature of the classifiers that generated the Annotation is projected onto the annotated area. This can be done in simple or in highlight mode.

Furthermore on the toolbar of the Visualization editor there are three controls related to Annotations (see Table 29).

Table 29: Controls related to Annotations on the toolbar of the Visualization editor

Icon	Function
	Delete all active search records
	Highlight projections of Terminals or Predicates in projection mode
	Enable/disable projection mode to map projections of Terminals or Predicates onto the current Visualization

Note that:

- (1) Scanning the current Visualization with selected classifiers (see previous Chapters) or manually adding Terminal Annotations (see Section 6.6) results in active Annotations.
- (2) Negative sub-signatures of Terminals are generally not mapped onto the Visualization in highlight mode.
- (3) Non-instantiated symbolic elements of Predicate Annotations are displayed in dark-blue.

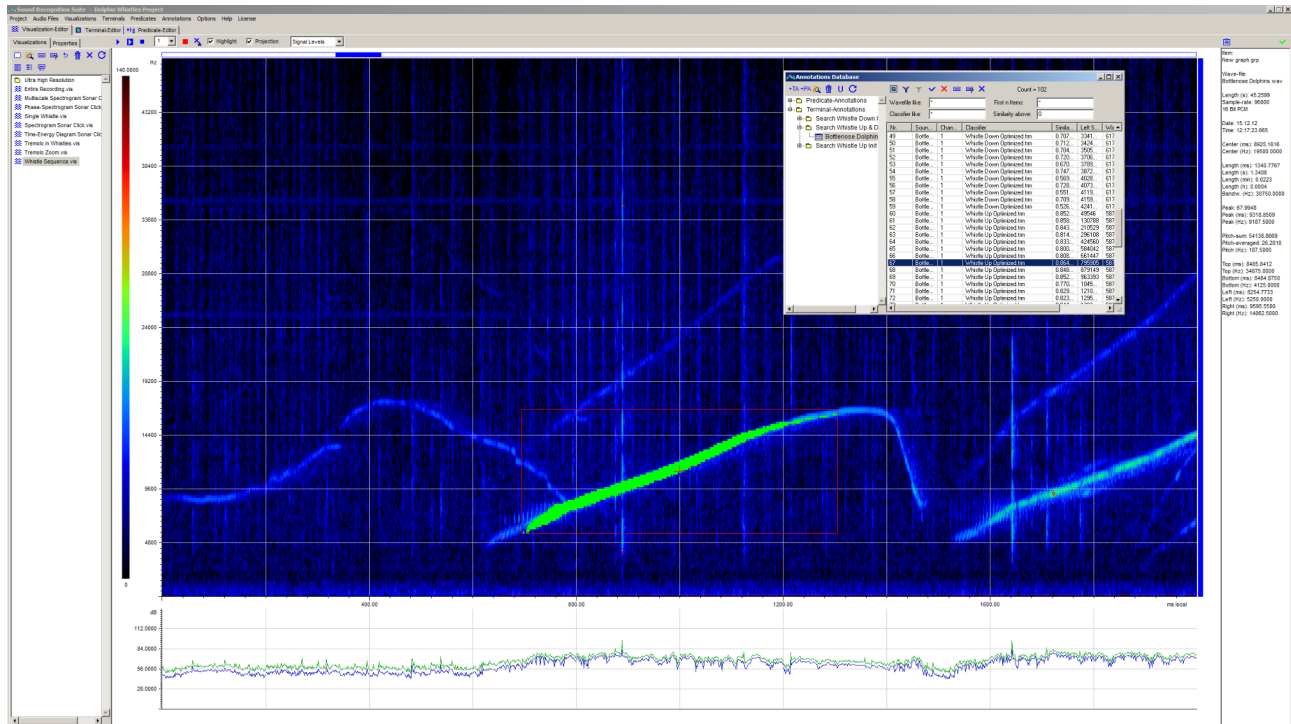


Figure 43: Single Annotation shown in highlight mode in the Visualization editor. The Annotation marks a particular part of a whistle. Only the positive sub-signature (bright green) is mapped onto the graph.

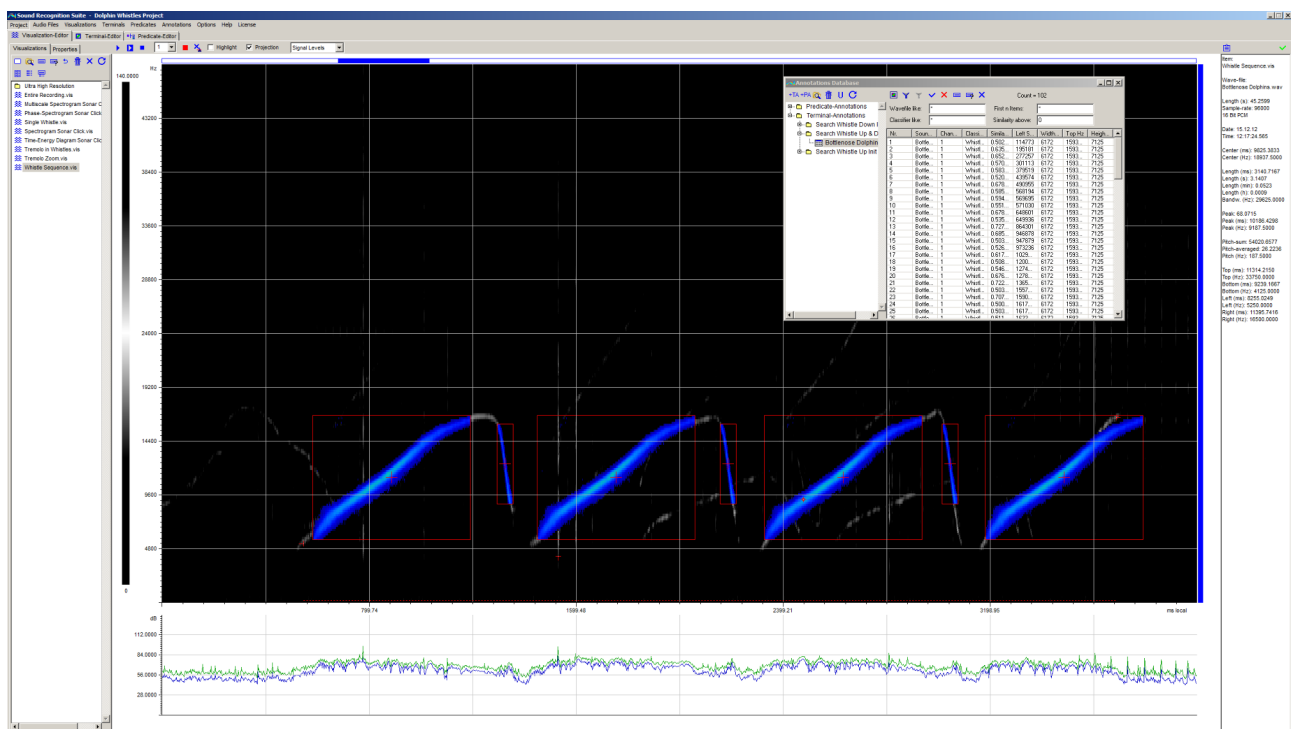










Figure 44: Annotated audio data. Multiple Annotations are displayed simultaneously after an entire table with Annotations had been activated.

## 6.5 Toolbar of the Annotation database interface

On top of the database interface is a toolbar with several general purpose controls. These controls are explained in Table 30.

Table 30: Toolbar of the Annotation database

Icon	Function
	Open classifier that is linked to the selected annotation.
	Apply filter.
	Clear filter.
	Activate the current table.
	Delete selected Annotation.
	Save changes to current data table.
	Rename and save current table.
	Close the current table.

## 6.6 Manual annotation of audio files

Acoustic patterns can be annotated manually in two simple steps:

- (1) Open an audio file in the Visualization editor. Go to the acoustic pattern or event you want to annotate. Use the mouse buttons to mark the pattern in the following way: left mouse button → select left time, right mouse button → select right time, ctrl & left mouse button → select bottom frequency, ctrl & right mouse button → select top frequency (see 45).
- (2) Press the Caps-lock key on your keyboard. Then press a character key (A-Z). An Annotation is added to the current search record. In addition a Terminal of the form "Wavefile\_Sample\_Key.trm" is generated in the subfolder "Terminals/Manual/". In the "Key-Terminal" table in the project settings window (see Figure 46) you can associate character keys with strings, e.g. the character W can be associated with the string "Whistle". All Annotations added by pressing the W-key will then generate a Terminal of the form 'Wavefile\_Sample\_Whistle.trm'.



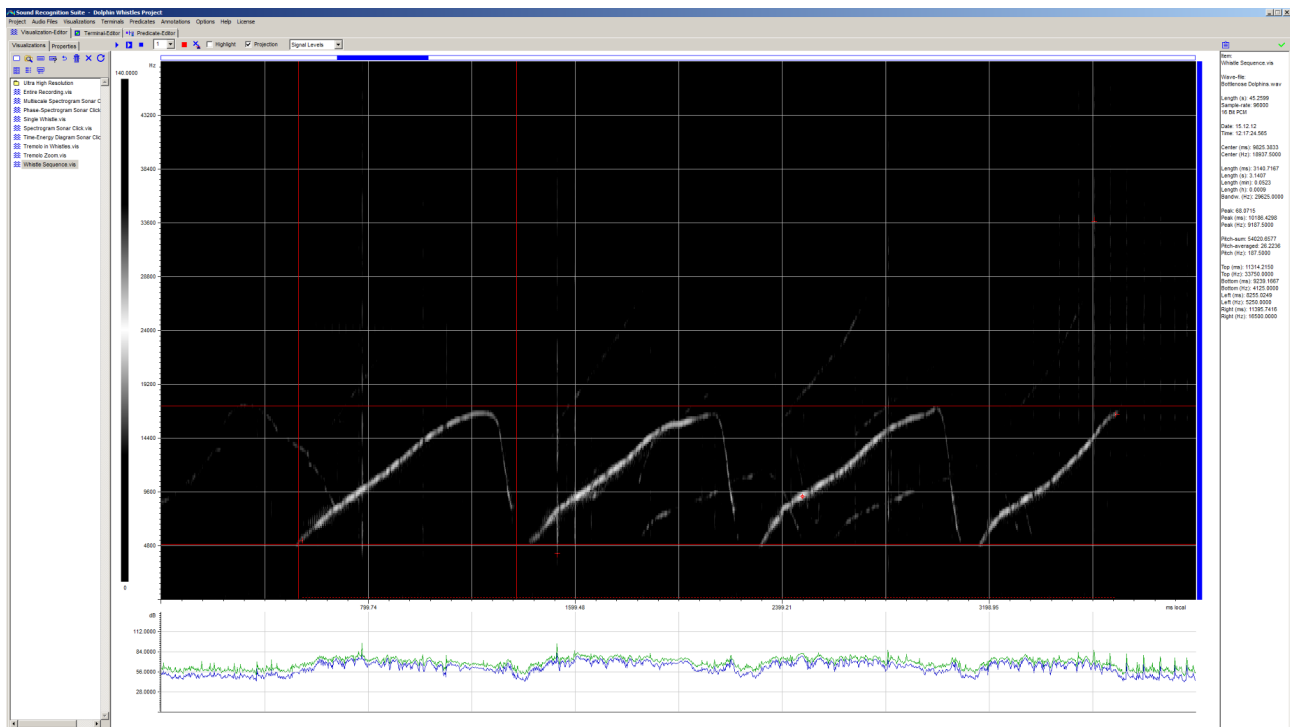


Figure 45: Whistle in Visualization editor selected with mouse buttons

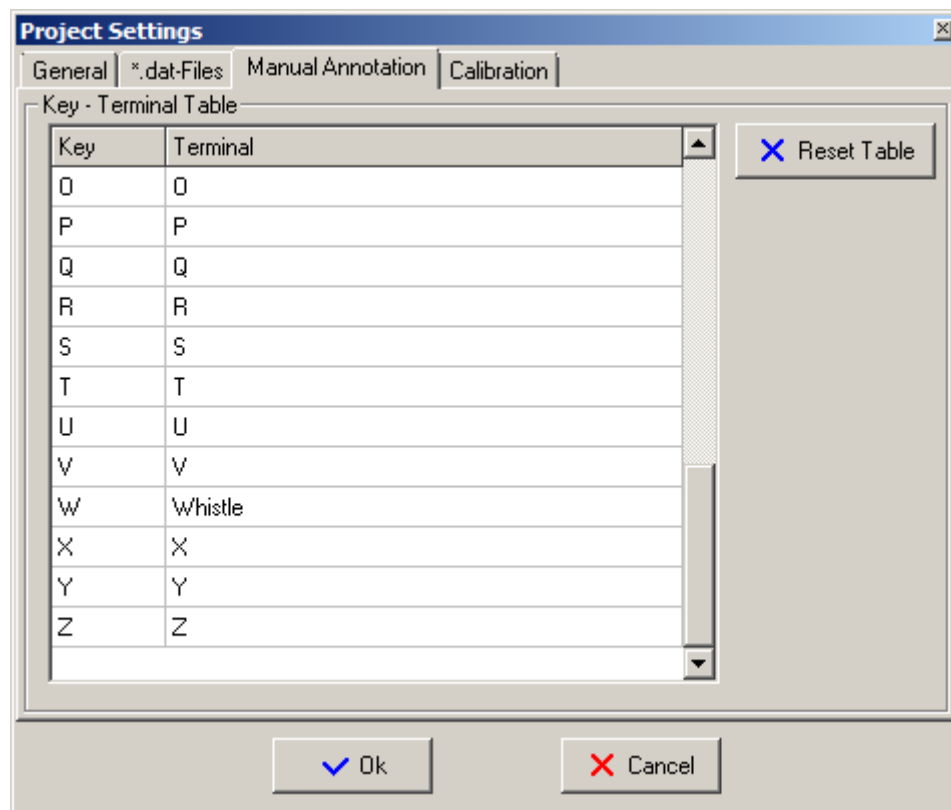


Figure 46: The 'Key-Terminal' table in the project settings window. The character W will be replaced by the string "Whistle" when manually annotating audio data.

Note that:

- (1) Individual Annotations can manually be selected. To do this, press the Shift key and click with the mouse in the frame that surrounds the Annotation. Selected Annotations are highlighted (see 47). They can be deleted by pressing the “Delete” key.
- (2) After having manually annotated audio data, add the active search record containing the Annotations to the database by clicking the '+TA' button in the database interface.
- (3) Terminals generated when manually annotating audio data have to be fine tuned before they can be used for automatic classification of audio data.

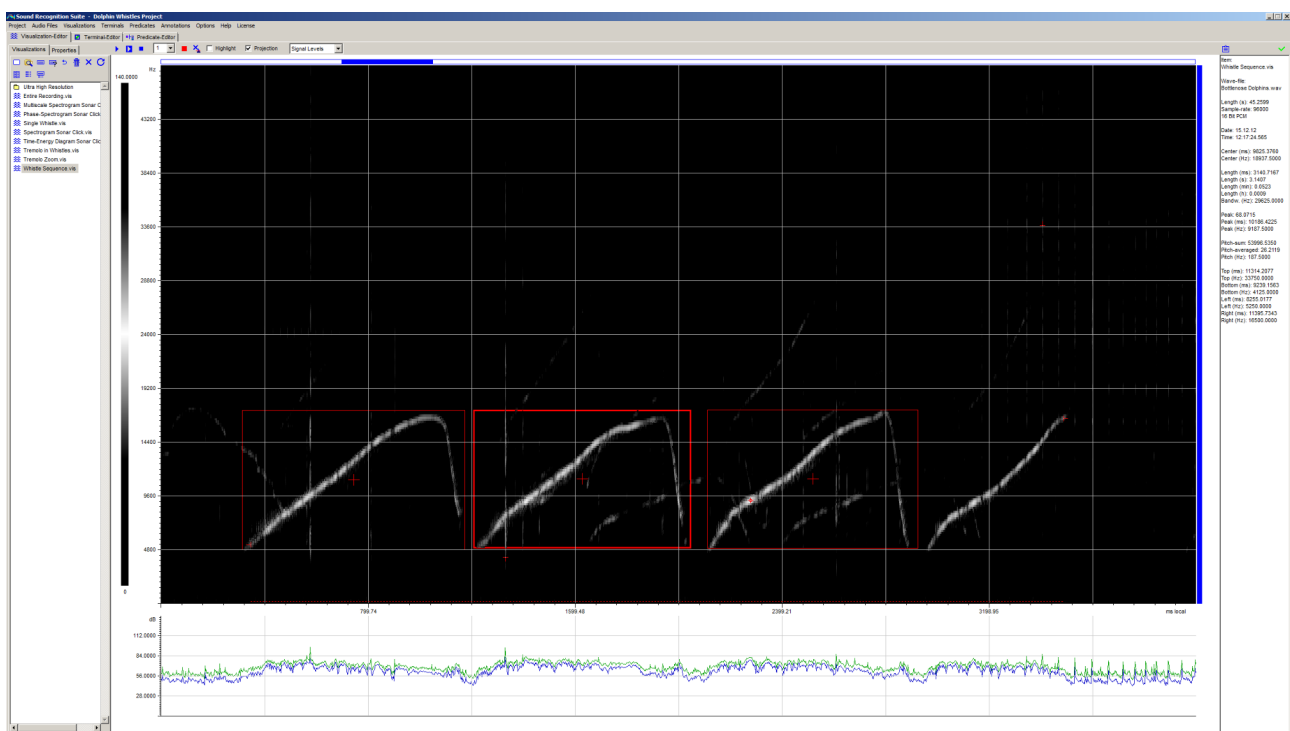





Figure 47: Three manually generated Annotations of dolphin whistles. The one in the middle is selected.

## 6.7 Automatic annotation of audio files

SR-Lab provides several tools for automatic annotation of audio data. The following table shows an overview.

Table 31: Automatic Annotation of Audio-Files with SR-Lab



Icon	Location	Function
	Toolbar of Terminal editor &	<b>Scan Visualization:</b> Scan current visualization with current classifier (visible in editor). Result is a search record containing Annotations. The record is active in memory.

Icon	Location	Function
	Toolbar of Predicate editor	Distance graphs are shown. Useful for interactive testing of single classifiers on short chunks of audio data. See previous chapters for details.
	Main menu / Terminals & Main menu / Predicates	<b>Scan Current Visualization:</b> Wizard to scan the current Visualization with selected classifiers. Result is a search record containing Annotations. The record is active in memory. Distance graphs are not shown. Useful for testing small sets of classifiers on short chunks of audio data. See previous chapters for details.
	Main menu / Terminals & Main menu / Predicates	<b>Scan Audio Files:</b> Wizard to scan audio files with selected classifiers. Result are automatically named data tables containing Annotations. Important tool to search audio file collections of arbitrary size. See previous chapters for details.
	Toolbar of Visualization editor	<b>Monitor Audio Input:</b> Wizard to monitor audio input with the help of Terminals and/or Predicates. Result are annotated recordings (see Chapter 8).

## 6.8 Wizards for Annotations

SR-Lab has two wizards to extract audio data from Annotations. The wizards can be reached via the main menu under the menu item “Annotations”. Table 32 shows an overview.

Table 32: Wizards for Annotations

Icon	Menu item	Task
	Extract Audio Files from Terminal Annotations	<p>Wizard for automatic extraction of Audio files from Terminal Annotations:</p> <ul style="list-style-type: none"> <li>If the option "Filtered" is selected, extracted audio data will be filtered with a bandpass filter. The Terminals" search frame determines the upper and lower cut-off frequencies of the filter.</li> <li>If the option "Expanded" is selected, the search area of the Terminals determines the start and end times for copying the audio data. Otherwise, the signature determines the start and end times.</li> </ul>
	Extract Audio Files from Predicate Annotations	<p>Wizard for automatic extraction of Audio files from Predicate Annotations:</p>







Icon	Menu item	Task
		<ul style="list-style-type: none"> <li>If the option “Filtered” is selected, all audio files are bandpass filtered according to the bandwidth defined by the upper and lower frequency borders of the Predicate that generated the Annotations.</li> </ul>

Note that not only audio data but also new classifiers can be extracted from Annotations. Please see Section 4.6 for automatic extraction of Terminals and Section 5.6 for extraction of Predicates. Section 7.1.5 describes how Annotations can be used to analyze acoustic data.

## 7 Audio Analytics

In many cases the goal of a pattern recognition based analysis of a large quantity of audio data is descriptive statistics about acoustic events in the data. SR-Lab provides several wizards to compile such statistics from audio files, Terminals, Predicates and Annotations. Table 33 shows an overview of the wizards. Each wizard is described in one of the following subsections.

Table 33: Wizards for audio analytics

Icon	Menu item	Task
	Analyze Recording Times	Wizard to analyze recording times of audio files. For details see Section 7.1.1.
	Analyze Sound Pressure Levels (dB)	Wizard to analyze sound pressure levels of audio files. For details see Section 7.1.2.
	Analyze Terminals	Wizard to compile general descriptive statistics from a set of Terminals. For details see Section 7.1.3.
	Analyze Predicates	Wizard to compile general descriptive statistics from a set of Predicates. For details see Section 7.1.4.
	Analyze Terminal Annotations	Wizard to compile general descriptive statistics from Terminal Annotations. For details see Section 7.1.5.
	Analyze Predicate Annotations	Wizard to compile general descriptive statistics from Predicate Annotations. For details see Section 7.1.5.

### 7.1.1 Analyze recording times wizard

This is a wizard to analyze recording times of audio files (see Table 34). Result are data tables which contain useful information about recording times. Table 35 shows which files are generated by the wizard. Figure 48 shows a typical recording times plot.

Table 34: Analyze recording times algorithm

<b>Input</b>	A set A of audio files
<b>Options</b>	(1) Interval length in ms [Integer value]
<b>Output</b>	CSV tables with useful information related to recording times of the files in A

Table 35: Files generated by the Analyze Recording Times wizard

File name	Content
Wave files overview.csv	Short overview about the analyzed audio files.
Recording times.csv	Table that lists all audio files with start time, end time and length of the recording
Recording duration vs time.csv	Table showing how long audio data was recorded in time intervals.
Recording duration vs date time.csv	Table showing how long audio data was recorded in date and time intervals.

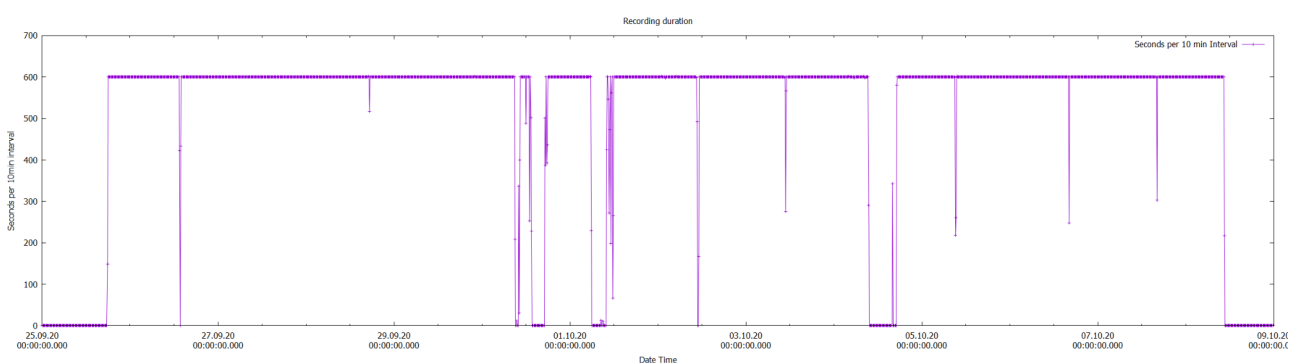


Figure 48: Plot showing recording durations in 10 minute intervals over the course of several days. (The plot was generated by Gnuplot from a “Recording duration vs date time.csv” table.)

### 7.1.2 Analyze sound pressure levels (dB) wizard

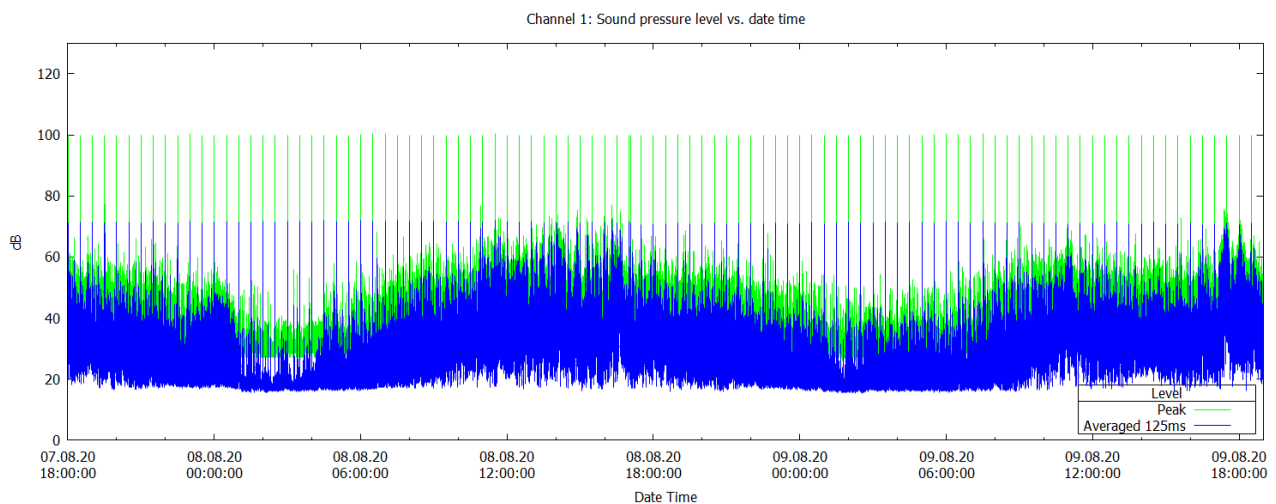
This is a wizard to analyze sound pressure levels of audio files (see Table 36). Result are data tables which contain useful information about sound pressure levels. Table 37 shows which files are generated by the wizard. Figure 49 shows a typical sound pressure level diagram.

Table 36: Analyze Sound Pressure Levels (dB) algorithm

<b>Input</b>	A set A of audio files
<b>Options</b>	(1) Interval length in ms [Integer value]
<b>Output</b>	CSV tables with useful information related to sound pressure levels in dB of the audio files in A.

Table 37: Files generated by the Sound Pressure Levels (dB) wizard. For each channel of the input files a set of output files is generated.

File name	Content
Channel n summary.txt	Short overview about analyzed audio files
SPL (dB) Distribution in Files.csv	Distribution of maximum peak and average SPL values in all analyzed files between 0 and 200 dB.
Channel n SPL vs time.csv	Table showing sound pressure levels according to time.
Channel n SPL vs date time.csv	Table showing sound pressure levels according to date time.

Figure 49: Sound pressure level diagram. Blue are averaged values ( $dt = 125ms$ ) and green are peak values. Spikes indicate an error in the recording hardware.

### 7.1.3 Analyze Terminals wizard

This is a wizard to compile general descriptive statistics from a set of Terminals (see Table 38). Result are data tables containing basic acoustic measurements (see Table 39), date and time

related properties (see Tables 40 and 41), distributions of physical properties (see Table 42) and scatter plots that show how certain features relate to one other (see Table 43).

*Table 38: Analyze Terminals algorithm*

<b>Input</b>	A set T1 of Terminals
<b>Options</b>	(1) Basic Measurements [Always TRUE] (2) Time Distributions [TRUE, FALSE] (3) Standard Distributions [TRUE, FALSE] (4) Scatter Plots [TRUE, FALSE]
<b>Output</b>	Descriptive statistics in CSV data tables and scatter plots in JPG format. All files are saved in the default export directory in the "Terminal Analytics" subfolder.

Option (1) “Basic Measurements” is always TRUE. A table named “Basic properties.csv” is generated. For each Terminal in T1 it contains the information shown in Table 39. Energy level values refer to the selected “Energy” property of the Terminals (log2, log10, dB or linear). Time related information refers to the audio data from which the signature originally was derived from.

*Table 39: Basic measurements of Terminals*

Key	Meaning
Classifier	File name of Terminal
Audio file	Audio file from which the Terminal was originally derived
Audio file length (sec)	Length of audio file from which the Terminal was originally derived
Date	Date stamp of audio file
Time	Time stamp of audio data chunk from which the Terminal was originally derived
Center time (ms)	Center time of signature from beginning of file
Center frequency	Center frequency of signature
Length (ms)	Length of signature in milliseconds
Bandwidth (Hz)	Bandwidth of signature in Hz (- x dB, x to be specified in the Options window)
Peak time (ms)	Time of data point in the signature that has the highest energy value. Measured from beginning of the audio file
Peak frequency (Hz)	Frequency of data point in the signature that has the highest

	energy value
Peak level	Energy level of data point in the signature that has the highest energy value
Pitch frequency (Hz)	Frequency of the band in the signature that has the highest energy density
Pitch energy sum	Sum of energy values in pitch frequency band
Pitch energy averaged	Average of energy values in pitch frequency band
Top time (ms)	Time of data point in the signature that has the highest frequency. Measured from the beginning of the audio file
Top frequency (Hz)	Frequency of data point in the signature that has the highest frequency
Bottom time (ms)	Time of data point in the signature that has the lowest frequency. Measured from the beginning of the audio file
Bottom frequency (Hz)	Frequency of data point in the signature that has the lowest frequency
Left time (ms)	Time of leftmost data point in the signature. Measured from the beginning of the audio file
Left frequency (Hz)	Frequency of leftmost data point in the signature
Right time (ms)	Time of rightmost data point in the signature. Measured from the beginning of the audio file
Right frequency (Hz)	Frequency of rightmost data point in the signature
Number of merged	Number of Terminals merged to obtain the analyzed Terminal
Positive points	Number of positive data points in the signature
Neutral points	Number of neutral data points in the signature
Negative points	Number of negative data points in the signature

If option (2) “Time Distributions” is TRUE a table named “Date time distribution.csv” and a table named “Time distribution.csv” is generated. To compile the distributions a fixed interval length of 60 seconds is used. The table “Date time distribution.csv” contains the information shown in Table 40. The table “Time distribution.csv” contains the information shown in Table 41.

*Table 40: Date time measurements of Terminals*

Key	Meaning
Date Time	Date and time of analyzed interval
Recorded time in interval (ms)	Recorded time in analyzed interval. Intervals have a fixed length of 60 seconds



Items total	Total number of all analyzed Terminals
Items per interval normalized	Number of Terminals in interval. Normalized with respect to recording time in interval
Items per minute normalized	Number of Terminals per minute. Normalized with respect to recording time in interval
Items per second normalized	Number of Terminals per second. Normalized with respect to recording time in interval

*Table 41: Daytime measurements of Terminals*

Key	Meaning
Time	Daytime (without date information) of interval
Recorded time in interval (ms)	Recorded time in interval. All evaluated intervals are 60 s long. More than 60s are evaluated if recordings last for more than one day
Items total	Total number of all evaluated Terminals
Items per interval normalized	Number of Terminals in interval. Normalized with respect to recording time in interval
Items per minute normalized	Number of Terminals per minute. Normalized with respect to recording time in interval
Items per second normalized	Number of Terminals per second. Normalized with respect to recording time in interval

If option (3) “Standard Distributions” is TRUE the distributions of standard physical properties of the signatures of the Terminals are compiled. Each distribution has a fixed size of 500 intervals. Energy level values refer to the selected “Energy” property of the Terminals (log2, log10, dB or linear). The following tables are generated. Figure 50 shows an example.

*Table 42: Tables with distributions*

File	Content
Bandwidth (Hz) distribution.csv	Distribution of bandwidth of signatures
Center frequency (Hz) distribution.csv	Distribution of center frequencies of signatures
Length (ms) distribution.csv	Distribution of length of signatures
Peak frequency (Hz) distribution.csv	Distribution of peak frequencies of signatures
Peak level distribution.csv	Distribution of peak levels of signatures

Pitch frequency (Hz) distribution.csv	Distribution of pitch frequencies of signatures
Top frequency (Hz) distribution.csv	Distribution of top frequencies of signatures

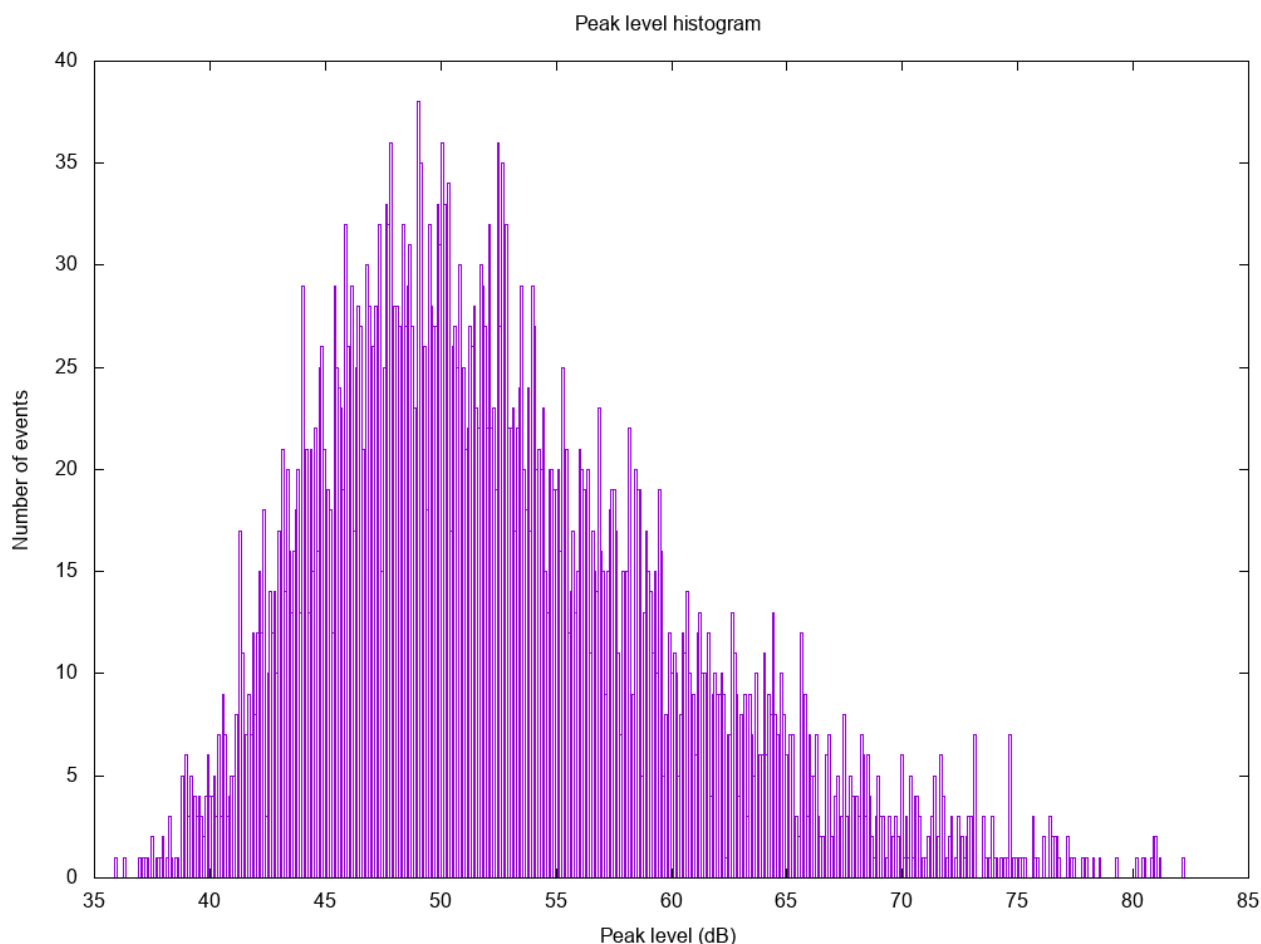


Figure 50: Distribution of peak levels (dB) in a set of Terminals

If option (4) “Scatter plots” is TRUE a number of scatter plots are generated displaying how physical characteristics of the Terminal signatures relate to each other (see Table 43). Each plot is on a 500 x 500 pixel bitmap. The number of occurrences of items per pixel is color coded and numbered (see Figure 51).

Table 43: Types of scatter plots

File	Content
Peak Level vs Top Frequency	Peak level on x-axis, top frequency on y-axis
Peak Level vs Pitch Frequency	Peak level on x-axis, pitch frequency on y-axis
Peak Level vs Peak Frequency	Peak level on x-axis, peak frequency on y-axis

Peak Level vs Center Frequency	Peak level on x-axis, center frequency on y-axis
Length vs Top Frequency	Length on x-axis, top frequency on y-axis
Length vs Pitch Frequency	Length on x-axis, pitch frequency on y-axis
Length vs Peak Frequency	Length on x-axis, peak frequency on y-axis
Length vs Center Frequency	Length on x-axis, center frequency on y-axis
Bandwidth vs Top Frequency	Bandwidth on x-axis, top frequency on y-axis
Bandwidth vs Pitch Frequency	Bandwidth on x-axis, pitch frequency on y-axis
Bandwidth vs Peak Frequency	Bandwidth on x-axis, peak frequency on y-axis
Bandwidth vs Center Frequency	Bandwidth on x-axis, center frequency on y-axis

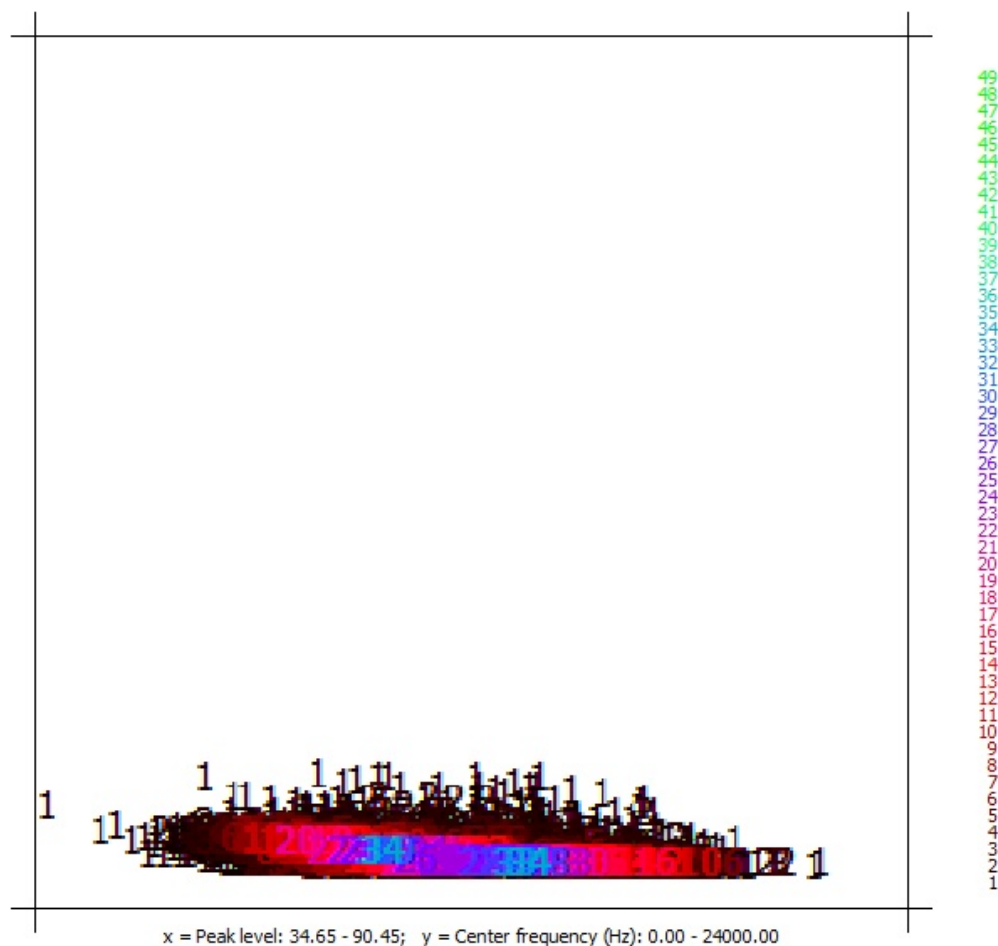


Figure 51: Scatter plot Peak Level vs. Center Frequency. The plot was generated from 5207 Terminals each representing the passing of a motor vehicle. It shows two clusters, one representing motor cycles (left), the other one representing cars (right).

### 7.1.4 Analyze Predicates wizard

This is a wizard to compile general descriptive statistics from a set of Predicates (see Table 44). Result are data tables containing basic measurements (see Table 45), distributions of properties (see Table 46) and scatter plots that show how basic physical characteristics relate to each other (see Table 47).

*Table 44: Analyze Predicates algorithm*

<b>Input</b>	A set T1 of Predicates
<b>Options</b>	(1) Basic Measurements [Always TRUE] (2) Predicate Distributions [TRUE, FALSE] (3) Scatter Plots [TRUE, FALSE]
<b>Output</b>	Descriptive statistics in CSV data tables and scatter plots in JPG format. All files are saved in the default export directory in the "Predicate Analytics" subfolder.

Option (1) "Basic Measurements" is always TRUE. A table named "Basic properties.csv" is generated that contains for each Predicate in T1 the information shown in Table 45.

*Table 45: Basic measurements of Predicates*

Key	Meaning
Classifier	File name of Predicate
Audio file	Audio file from which the Predicate was derived
Audio file length (sec)	Length of audio file from which the Predicate was derived
Date	Date stamp of audio file
Time	Time stamp of audio data chunk from which the Predicate was originally derived
Center time (ms)	Center time of signature starting from zero
Center frequency	Center frequency of signature
Length (ms)	Length of signature in milliseconds
Bandwidth (Hz)	Bandwidth of signature in Hz (- x dB)
Top time (ms)	Time starting from zero of element in the signature that has the highest frequency.
Top frequency (Hz)	Top frequency of element in the signature that has the highest frequency
Bottom time (ms)	Time starting from zero of element in the signature that has the lowest frequency.

Bottom frequency (Hz)	Frequency of element in the signature that has the lowest frequency
Left time (ms)	Time starting from zero of leftmost element.
Left frequency (Hz)	Frequency of leftmost element
Right time (ms)	Time starting from zero of rightmost element in the signature
Right frequency (Hz)	Frequency of rightmost element in the signature
Number of merged	Number of Predicates merged to obtain the analyzed Predicate
Terminal items total	Number of elements in signature
Terminal items per second	Number of elements per second in signature
Compression rate	Overall compression rate in signature (i.e. the sum of compression rates of each element divided by the number of elements)

If option (2) “Predicate Distributions” is TRUE distributions of general properties of Predicate signatures are compiled. Each distribution has fixed size of 500 intervals. Energy level values refer to the “Energy” parameter of the Predicates (log2, log10, dB or linear). Table 46 shows the files that are generated.

*Table 46: Files with Predicate distributions*

File	Content
Bandwidth (Hz) distribution.csv	Distribution of bandwidth of signatures
Center frequency (Hz) distribution.csv	Distribution of center frequencies of signatures
Length (ms) distribution.csv	Distribution of length of signatures
Top frequency (Hz) distribution.csv	Distribution of top frequencies of signatures
Terminal items per second distribution.csv	Distribution of the number of Terminal items per second of signatures
Terminal items total distribution.csv	Distribution of the total number of Terminal items in the signatures
Compression rate distribution.csv	Distribution of compression rates of signatures

If option (3) “Scatter plots” is TRUE a number of scatter plots are generated displaying how the above properties of the signatures relate to one other (see Table 47). Each plot is on a 500 x 500 pixel bitmap.

Table 47: Types of scatter plots

File	Content
Terminal Items per Sec vs Compression	Terminal items per sec on x-axis, Compression rate on y-axis
Terminal Items per Sec vs Length	Terminal items per sec on x-axis, Length on y-axis
Terminal Items Total vs Compression	Terminal items total on x-axis, Compression rate on y-axis
Terminal Items Total vs Length	Terminal items total on x-axis, Length on y-axis

### 7.1.5 Analyze Terminal / Predicate Annotation wizard

These are wizards to compile general descriptive statistics from either Terminal or Predicate Annotations (see Table 48). Result are data tables containing basic measurements (see Table 49), date and time related properties (see Tables 50 and 51), distributions of acoustic properties (see Table 52) and scatter plots that show how certain acoustic characteristics relate to one other (see Table 53).

Table 48: Analyze Terminal / Predicate Annotations algorithm

<b>Input</b>	A set A of Terminal or Predicate Annotations
<b>Options</b>	(1) Basic Measurements [Always TRUE] (2) Time Distributions [TRUE, FALSE] (3) Standard Distributions [TRUE, FALSE] (4) Scatter Plots [TRUE, FALSE]
<b>Output</b>	Descriptive statistics in CSV data tables and scatter plots in JPG format. All files are saved in the default export directory under the "Terminal Annotations Analytics" or "Predicate Annotations Analytics" subfolder.

Option (1) “Basic Measurements” is always TRUE. A table named “Basic properties.csv” is generated. It contains for each Annotation in A the information shown in Table 49. Energy level values refer to the selected “Energy” property of the Terminals or Predicates (log2, log10, dB or linear).

Table 49: Basic measurements of Annotations

Key	Meaning
Classifier	File name of Terminal or Predicate that generated the Annotation

Audio file	Name of annotated audio file
Audio file length (sec)	Length of annotated audio file
Date	Date stamp of Annotation
Time	Time stamp of left border of Annotation
Center time (ms)	Center time of Annotation from beginning of file
Center frequency	Center frequency of Annotation
Length (ms)	Length of Annotation in milliseconds
Bandwidth (Hz)	Bandwidth of Annotation in Hz (- x dB, x to be specified in the Options window)
Peak time (ms)	Time of data point in Annotation with highest energy value. Measured from beginning of audio file
Peak frequency (Hz)	Frequency of data point in Annotation with highest energy value
Peak level	Energy level of data point in Annotation with highest energy value
Pitch frequency (Hz)	Frequency of band in Annotation with the highest energy density
Pitch energy sum	Sum of energy values in pitch frequency band
Pitch energy averaged	Average of energy values in pitch frequency band
Top time (ms)	Time of data point in Annotation with the highest frequency. Measured from the beginning of the audio file
Top frequency (Hz)	Frequency of data point in Annotation with the highest frequency
Bottom time (ms)	Time of data point in Annotation with the lowest frequency. Measured from the beginning of the audio file
Bottom frequency (Hz)	Frequency of data point in Annotation with the lowest frequency
Left time (ms)	Time of leftmost data point in Annotation. Measured from the beginning of the audio file
Left frequency (Hz)	Frequency of leftmost data point in Annotation
Right time (ms)	Time of rightmost data point in Annotation. Measured from the beginning of the audio file
Right frequency (Hz)	Frequency of rightmost data point Annotation

If option (2) “Time Distributions” is TRUE a table named “Date time distribution.csv” and a table named “Time distribution.csv” is generated. To compile the distributions a fixed interval length of 60 seconds is used. The table “Date time distribution.csv” contains the information shown in Table 50. The table “Time distribution.csv” contains the information shown in Table 51.

*Table 50: Date time measurements of Annotations*

Key	Meaning
Date Time	Date and time of interval
Recorded time in interval (ms)	Recorded time in interval. Intervals have a fixed length of 60 seconds
Items total	Total number of all evaluated Annotations
Items per interval normalized	Number of Annotations in interval. Normalized with respect to recording time in interval
Items per minute normalized	Number of Annotations per minute. Normalized with respect to recording time in interval
Items per second normalized	Number of Annotations per second. Normalized with respect to recording time in interval

*Table 51: Daytime measurements of Annotations*

Key	Meaning
Time	Daytime (without date information) of interval
Recorded time in interval (ms)	Recorded time in interval. All evaluated intervals have a fixed length of 60 seconds. More than 60 seconds are evaluated if recordings last for more than one day
Items total	Total number of all evaluated Annotations
Items per interval normalized	Number of Annotations in interval. Normalized with respect to recording time in interval
Items per minute normalized	Number of Annotations per minute. Normalized with respect to recording time in interval
Items per second normalized	Number of Annotations per second. Normalized with respect to recording time in interval

If option (3) “Standard Distributions” is TRUE the distributions of standard physical properties of the signatures of the Annotations are compiled. Each distribution has a fixed size of 500 intervals. Energy level values refer to the selected “Energy” property of the Terminals or Predicates (log2, log10, dB or linear). The files shown in Table 52 are generated.

*Table 52: Files with distributions of Annotations*

File	Content
Bandwidth (Hz) distribution.csv	Distribution of bandwidth of Annotations
Center frequency (Hz) distribution.csv	Distribution of center frequencies of Annotations



Length (ms) distribution.csv	Distribution of length of Annotations
Peak frequency (Hz) distribution.csv	Distribution of peak frequencies of Annotations
Peak level distribution.csv	Distribution of peak levels of Annotations
Pitch frequency (Hz) distribution.csv	Distribution of pitch frequencies of Annotations
Top frequency (Hz) distribution.csv	Distribution of top frequencies of Annotations

If option (4) “Scatter plots” is TRUE a number of scatter plots are generated displaying how physical properties of the Annotations relate to one other (see Table 53). Each plot is on a 500 x 500 pixel bitmap. The number of occurrences of items per pixel is color coded and numbered (see Figure 51).

*Table 53: Types of scatter plots generated from Annotations*

File	Content
Peak Level vs Top Frequency	Peak level on x-axis, top frequency on y-axis
Peak Level vs Pitch Frequency	Peak level on x-axis, pitch frequency on y-axis
Peak Level vs Peak Frequency	Peak level on x-axis, peak frequency on y-axis
Peak Level vs Center Frequency	Peak level on x-axis, center frequency on y-axis
Length vs Top Frequency	Length on x-axis, top frequency on y-axis
Length vs Pitch Frequency	Length on x-axis, pitch frequency on y-axis
Length vs Peak Frequency	Length on x-axis, peak frequency on y-axis
Length vs Center Frequency	Length on x-axis, center frequency on y-axis
Bandwidth vs Top Frequency	Bandwidth on x-axis, top frequency on y-axis
Bandwidth vs Pitch Frequency	Bandwidth on x-axis, pitch frequency on y-axis
Bandwidth vs Peak Frequency	Bandwidth on x-axis, peak frequency on y-axis
Bandwidth vs Center Frequency	Bandwidth on x-axis, center frequency on y-axis

## 8 Monitoring acoustic processes

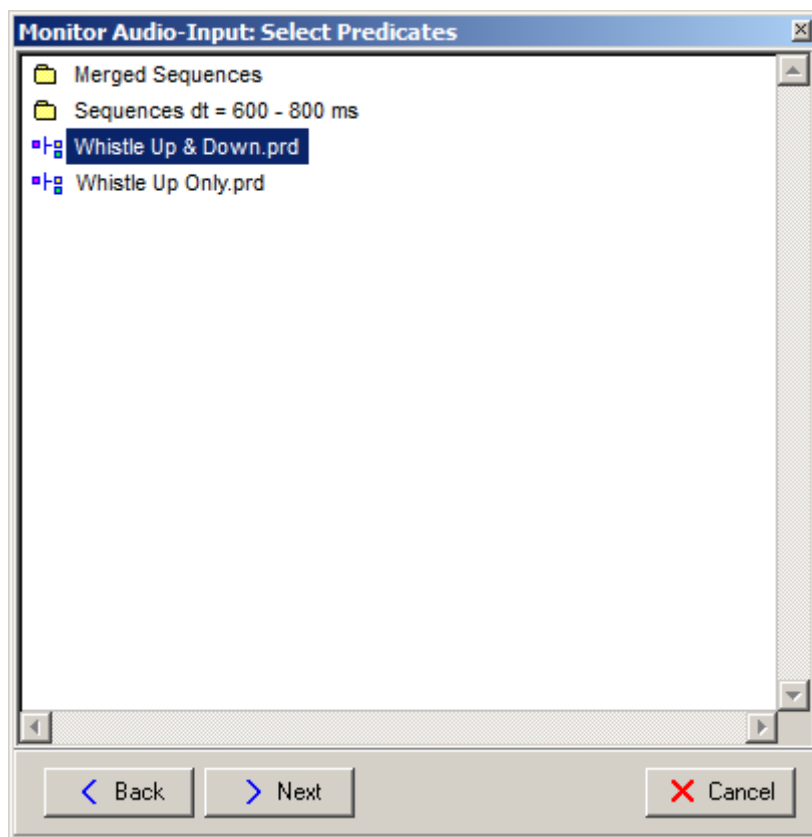


Figure 52: Audio monitoring wizard.

With the help of proper Terminals and Predicates it is possible to quickly annotate large amounts of previously recorded audio data. It is also possible to monitor audio input. In order to do so, start the “Monitor Audio Input” wizard in the Visualization editor (see Figure 52).

Table 54: Audio monitoring algorithm

<b>Input</b>	A set $C$ of Terminal or Predicate Classifiers, audio input stream
<b>Options</b>	<ol style="list-style-type: none"> <li>1. A set of Terminals and/or a set of Predicates</li> <li>2. Save mode [All Recordings, Annotated Recordings only]</li> <li>3. General audio input settings (see Table 55)</li> </ol>
<b>Output</b>	Audio files and Annotations

Before you start the audio monitoring process, it is necessary to configure the audio monitoring settings properly. To do so, open the Options window and click the “Monitoring” tab (see Figure 53). All options are explained in Table 55.

Table 55: Audio monitoring options

Option	Meaning
Number of channels	Number of input channels (currently only 2 channels are supported)
Sampling rate	Sampling rate used for recording
Chunk length	SR-Lab records chunks of audio data. Recorded chunks are analyzed while new audio data is recorded. Determine the length of chunks to be recorded.
Activation level (dB)	Minimum level in dB to trigger audio recording
Maximum level (dB)	Stop recording chunk when this level is surpassed.
Bandpass	Enable a bandpass filter for audio input. Set bottom and top cut-off frequencies. The filter is applied in first stage before measuring the activation level..
Show Visual Feedback	If checked, each analyzed chunk of audio data is shown in the Visualization editor with Annotations. Note that this may slow down audio classification in some cases.

The screenshot shows the 'Options' dialog box with the 'Monitoring' tab selected. The 'Wave Input Format' section contains two dropdown menus: 'Number of channels' set to '1' and 'Sampling rate' set to '48000'. The 'Parameters' section contains three dropdown menus: 'Chunk length (ms)' set to '2000', 'Activation level (dB)' set to '65', and 'Maximum level (dB)' set to '95'. The 'Bandpass' section has two radio buttons: 'Disable' (selected) and 'Enable'. Below the radio buttons are two input fields: 'Bottom (Hz)' set to '150' and 'Top (Hz)' set to '96000'. At the bottom of the dialog, there is a checked checkbox for 'Show Visual Feedback' and an 'Ok' button with a blue checkmark icon.

Figure 53: Options for monitoring acoustic processes.

Result of the audio monitoring process is a set of analyzed audio recordings and sets of Annotations referring to these recordings. By default audio recordings are saved in the folder “Waves/Monitoring/Analyzed” and Annotations are saved in the folder Annotations under Terminal- or Predicate Annotations.

#### Notes:

- (1) The more classifiers are used at once, the slower audio data classification gets. If incoming audio data cannot be analyzed just in time, it is stacked in the subfolder “Waves/Monitoring/”.
- (2) Sampling rate and bit depth of audio data and classifiers have to be identical. Default bit depth for audio monitoring is 32 Bit.
- (3) Incoming audio data is saved in a file under the path “Waves/Input/current.dat”. This file is for internal use only.
- (4) In *SR-Lab Embedded* all described audio monitoring functions can also be used in silent mode for process automation. Inter-process communication is realized via a named pipe server and memory-mapped files.

## 9 System requirements and known issues

### 9.1 System requirements

- Operating System: Microsoft® Windows® 7, 8, 10 or higher.
- Intel® or AMD® processors with x86 / x64 instruction set are required. SR-Lab uses assembler libraries for these processors and currently does not run on other CPUs.

### 9.2 Known issues

- Best results may be achieved with high-resolution graphics. Objects created on a PC with high resolution can be used on PCs with a lower resolution. However, this might result in performance differences.
- SR-Lab requires full reading and writing access to the folders you choose.
- **SR-Lab is 64-Bit software. It does not run 32-Bit systems!**

Sejona-Software  
© Copyright Dr. Sebastian Huebner  
Parkstrasse 3  
D-34346 Hann. Münden / Germany

[support@sejona.de](mailto:support@sejona.de)  
[www.sejona.de](http://www.sejona.de)  
[www.sound-recognition.com](http://www.sound-recognition.com)