FASTLab, Inc: Music Software
Santa Barbara, California
Tel: (805) 895-6252
Email: info@FASTLabInc.com

FASTLab: A Tool for Music Analysis

The FASTLab software is an analysis tool for use in music and sound databases. It uses a number of different analysis techniques to extract both low-level and high-level features from musical selections. In effect, FASTLab "listens" to music and derives over 100 attributes related to the musical structure, instrumentation, style, and other properties. The analysis data can be used for a variety of applications, including user preference matching, music fingerprinting, and database queries. This white paper presents the FASTLab analysis technology and offers several application examples.

Problem Statement

There is already a wealth of music available in databases and on the Internet. The biggest problem in using this resource is the difficulty in finding music that fits a given description or a specific listener's preference profile. Several kinds of applications and web services exist for querying music databases, but they generally rely on text annotation of musical genres, or on classifications generated manually. The first method is problematic because music providers generally supply only very vague or general genre classifications, and the second method fails because it relies on an "army of ants" that listen to short excerpts of music selections and classify them manually. The FASTLab system was written to solve these problems.

FASTLab and its Applications

The FASTLab analysis tool is intended for use in music and sound databases, listener preference-matching applications, and other cases where a multidimensional characterization of a musical selection is needed. It uses a number of different analysis techniques to extract features from musical selections. The present version is oriented towards the analysis of popular music songs stored as MP3 files, and the generation of an XML database. The analysis engine is, however, style- and genre-independent, and in fact, many styles were used to test it during its development, including popular music, contemporary art music, non-western music, and sound effects. In this paper, we describe music as "songs," but FASTLab can just as easily process other styles and genres.

As the world's entire musical inventory, both past and present, is now (or soon will be) digitized and available online in one form or another, there are more and more applications areas for the kind of perceptual feature extraction technology provided by FASTLab. A few of the most promising application domains will be introduced next.

- Preference matching: How does a casual listener navigate the ocean of music that has become available for perusal via the Internet, satellite radio, or other new streaming media? Given that thousands of new selections come online almost every day, it is not practical to manually listen to and annotate every single piece. An automatic procedure that generates meaningful musical parameters (i.e., features of music that have been shown to correlate to users' listening preferences) would greatly enhance any user interface for a music delivery service. The Figure below shows how the FASTLab analysis and statistical package would serve as a back end for a "smart" user-customized music search engine.

- Database access: This problem is perhaps typified most succinctly by the marketing managers who has to make a recommendation to a client "on the spot." Again, the size of the available library of jingles, backgrounds etc. can be very large and the process of manual annotation will be too slow and costly.

• Musical fingerprinting: With the recent court battles over file sharing and online copyright protection, song fingerprinting has become a critical technology. FASTLab's collection of analysis data would be ideally suited to make robust identification software.

• Automatic mastering: Submitting a raw mix to a mastering/encoding process is often beyond either the means or the knowledge of many low-budget artists (i.e., almost everyone). Since we obtain high-level perceptual information from the musical waveform, we can discover enough information to make a good guess at what would be a proper mastering solution. Another useful area would be for Internet radio DJs who would like to "normalize" a mixed set of MP3 files into a coherent-sounding audio stream.

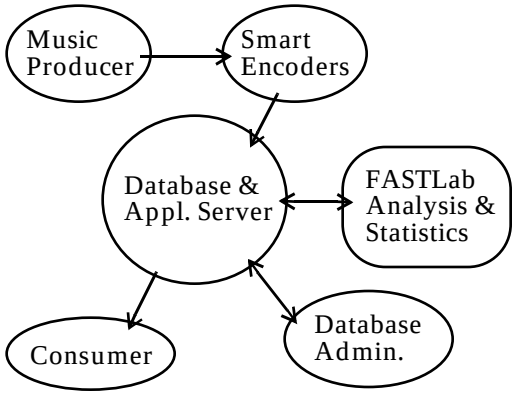

Figure: A FASTLab Application in a Music Database

The above examples are a short sampling of the potential applications of the kind of high-level automatic music analysis that FASTLab provides.

The FASTLab Technology

FASTLab is a flexible system that uses a range of sound analysis techniques in the time and frequency domains. It also uses several second-level heuristic programming techniques such as rule-based matching, fuzzy logic, and classification and regression trees (CART).

The framework is extensible both in terms of the low-level analysis and feature extraction techniques it uses, and in terms of the second-level pattern recognition, matching, and classification tools that are available.

Lastly, the output format is also changeable, the default action being to dump all analysis data to XML files for loading into a meta-data database.

The Figure below shows the interaction of the basic components of FASTLab.

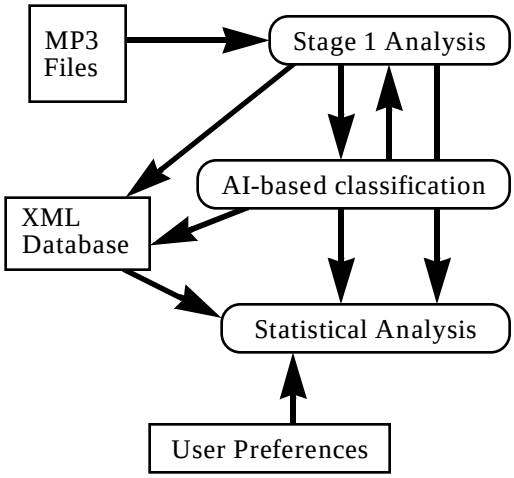

Figure: Overview of FASTLab

The software components of FASTLab are:

Input decoder
Rhythm/Tempo Analyzer
SpectrumAnalyzer
LPC Analyzer
Partial and Formant Tracking
Stereo Correlation and Phase derivation
Filter Module
Bass Pitch Tracker
Instrument Signature ID
Style Matcher
CART Tree Analysis/Matching
XML output
Debugging GUI

Each of these subsystems is implemented by one or more C++ classes; we will describe their operation and give examples of the system's use in the following sections.

The Figure on the next page shows a more detailed overview that relates the various implementation-level analysis and statistics packages to one another.

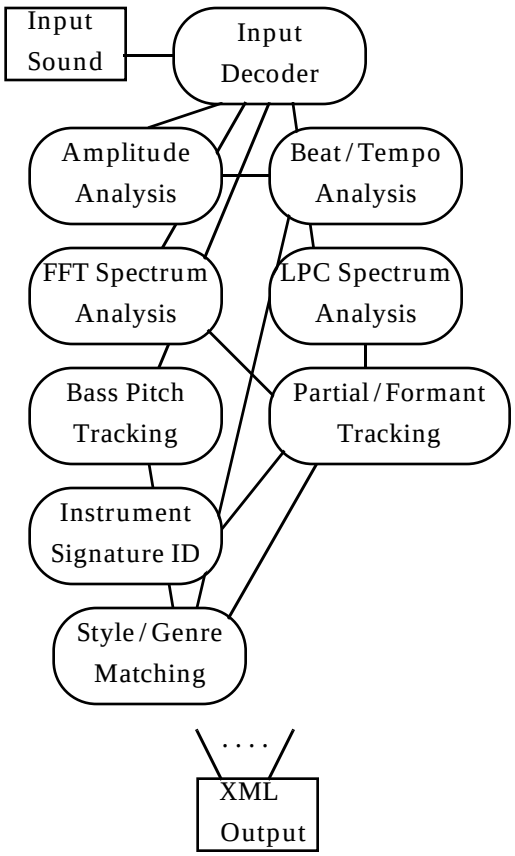

Figure: The FASTLab Architecture

Input Decoder

FASTLab can load files in a variety of formats from a variety of sources. The standard application searches for MP3 files on a local hard disk, but the system can read several other file formats, and can use external protocols over the Internet to access its input data.

Beat Analyzer

The beat analyzer is a beat and tempo analysis package based on several simple insights into the time-domain characteristics of music; it can extract several useful features from the rhythmic information of a song. We start by doing a sliding window root-mean-square (RMS) derivation to get the time-domain "envelope" of the sound signal. The characteristics of this process are configurable. The beat analyzer looks for repeated peaks in this envelope, and can track regular beats to find the tempo and metrical pulse of a sound.

In the second stage, the rhythm template analyzer looks through filtered signals RMS data trying to identify most widely occurring rhythmic patterns in various frequency components. Those can then be used for further classification (see below).

These two features are already very meaningful, and simple extrapolation functions can look for higher-level tempo structure and changes in the basic beat (the time signature) to identify the major sections of a piece of music (verses, sections, or movements). It can also be useful to know whether the strong beats correspond between several frequency bands (without resorting to a full-blown spectrum analysis). There are also several kinds of filters that we can apply to the sound, for example to look for beats in just the low- or high-frequency components.

The system uses a simple rule-based framework for higher-level metrical analysis that allows us to classify musical pieces according to their macro-level beat structure. We can also use this rhythmic information to determine how and where to apply frequency-domain analysis such as voice-finding or instrument signature extraction.

Spectrum Analysis and LPC

The FASTLab spectrum analyzer derives a time sequence of frequency spectra. In the current system, we typically represent the spectral data as amplitude spectra, but we can also use alternative outputs like power spectrum, log of power, log of amp etc.

The LPC (linear predictive coding) analyzer uses a similar but subtly different analysis technique to derive spectral analysis surfaces from selections of a sound.

Component tracking can be applied to both of these spectra; this is a process in which we try to locate spectral features that are constant (or change slowly) over time, as in tracking the overtones of a musical note. this allows us to identify spectral features of the sound such as the average number of notes playing, the "harmonicity" of the notes, and also to identify specific instruments based on

their spectral features (see below).

Stereo Phase Analysis

It is possible to take advantage of certain standard practices in stereo mixing as found in most popular recordings. Usually, the most prominent elements like vocals and solo instrumentals, and low frequency elements like bass and kick drums are mixed center with the remainder panned to left / right in various degrees.

We have instrumented our various analysis objects with operations to make it easy to "reverse-mix" some of this information. The following equations should make this clearer.

From the above, we can express the signal as:

Left = A + B
Right = A + C

It is easy to see that we can remove A from the combined signal, and this is often how vocals are eliminated in Karaoke machines. However, we cannot easily isolate A from the time-domain signal because of the linear dependence. However, in the frequency domain, we can assume the following, A = -A, if we discard phase information. Then we have the following:

F(B-C) = F(B) + F(-C) = F(B+C)

From the sum and difference of the left and right channel and the above identity, we get:

F(2A+B+C) – F(B+C) = F(2A).

From this output, analysis of vocals and bass is greatly facilitated. As mentioned above, all the relevant objects have operations defined on them, that makes this kind of manipulation easy to do.

Spectral Measures

A number of unique algorithms for extracting spectral features have been developed. These can work on the outputs of both the FFT and LPC analysis or any other properly formatted 2D data set for that matter.

Spectral saturation measures how "rich" a signal is, in terms of activity at all possible frequency bins. Thus, a solo instrument recording, with for example a flute, would get a low score, whereas a full blown pop production, like those of the Britney Spears variety, would get a proportionately high score.

Spectral contrast measures transitions in the frequency bins. The algorithm gives high scores if it detects simultaneous large amplitude differences in the time domain, for a large number of bins. Such contrast would indicate that a new section in the song has been reached and is useful not only to identify selections where such contrast is common (e.g. between verse and chorus), but also to segment the song for higher-level structural analysis.

Spectral variety is similar, but does not take into account the simultaneousness of the detected changes. This measure is indicative of general musical "busyness," how much various elements come and go during a piece.

Repeat sections is a highly optimized version of an autocorrelation function performed on all bins for the entire song. The full mathematical version would be prohibitively expensive, but by making a few choice assumptions, it was possible to bring the execution time down to reasonable levels. As the name implies, this measure can be used to detect repeating sections within a song. This works particularly well with a lot of Electronica / Dance styles, where much of the repeating material is generated by computers.

The table below is an example of some scores generated by the Contrast and Variety algorithms for various rock / pop / jazz artists.

	Contrast	**Variety**
FrodoCPU	221.5	323.14
Britney Spears	113.5	326.13
Prodigy	85.88	276.0
GooGoo Dolls	65.51	282.39
Steely Dan	39.88	227.24
DMX	30.92	176.4
Dr Dre	10.48	95.13
Fisher	5.63	129.23
Lyle Mays	1.61	1.8

Instrument Signature ID

An important part of any musical analysis is to identify the makeup of the instrumentation. This is important, not only to get an idea of the timbral character, but also to help subsequent 2nd order processes, like style identification (see below). FASTLab is currently identifying several distinct instrumental sounds, including grunge guitar, snare drum, kick drum and bass.

Stylistic Matching

Several different processes attempt to find patterns in the music that correspond to key features of various musical styles, like e.g. country, rock, funk and so on. This analysis is done in both the time and frequency domains.

The rhythm analyzer places the previously identified rhythmic "hits" on a meter grid with a resolution up to a 16th note. Then each bar is compared to a library of patterns that have been carefully selected for orthogonality and stylistic uniqueness. This procedure is reasonably robust with respect to various noise components, like misplaced downbeat and falsely identified hits.

In a like manner, the bass is isolated through various filters, including the stereo phase filter, and subjected to a similar kind of pattern matching. In this case the identified bass notes are converted to relative semitone intervals to make them key independent and placed on a rhythmic grid. This is then compared to a pattern library of short stylistically typical note sequences.

The final output is various statistics of the recorded pattern occurrences and from this it is possible to make a reasonably good determination of what style the song is in or if it's not present in the current library.

The system derives a range of first-order features from songs, and then uses these to compute about 100 second-order feature. As an example of the raw signal processing functionality of FASTLab, a few of the first-order features are:

Busy lo-range
Busy mid-range
Loudness Contrast
Quiet
Fadeout
Spike average
Bass loudness
Grunge max
Grunge cover
Spectral saturation
Spectral variety
Spectral contrast
Snare prob
Kick prob
Glide count short
Glide count long
Glide chords long
Repeat sections percent
Repeat sections count

CART Classification/Matching

For high-level grouping of the first-order features, and also possibly for detailed genre classification, we use an artificial intelligence technique called classification and regression trees (CART). These trees allow developers using FASTLab to "train" the system for a specific application, musical style, or analysis domain.

FASTLab includes a CART (Classification and Regression Trees) package called Mulcher, which is a second processing stage and reads the main FASTLab application's XML output data. Mulcher predicts song style information using CART trees that are built before-hand based on training data. Both the training and evaluation stages are supported in Mulcher.

In training mode, a special set of pre-labeled data items is used to construct a number of trees for different genre classifications, and a database of training labels is maintained so that valuable training data is not lost as the main FASTLab application evolves.

In runtime mode, Mulcher can either evaluate data stored on disk, or it can accept XML from a TCP socket. Most of the intermediate data and training information can be exported to or imported from XML format.

Mulcher uses a unique CART algorithm to

operate with multidimensional style vectors, which is a step beyond the usual discrete categories. The data model can handle songs with mixed style elements, and it can be expanded as needed without the need to reenter all the training data. The prediction results can be used either by themselves (for example, for preference matching) or together with FASTLab main application output to derive other features.

The screen shot below shows the Mulcher training and run-time GUI. In the example shown here, a song from the group Ace of Base is classified as eurodance (62.5% or disco (37.5%), and the GUI allows the trainer to manually change this in order to refine the system's knowledge.

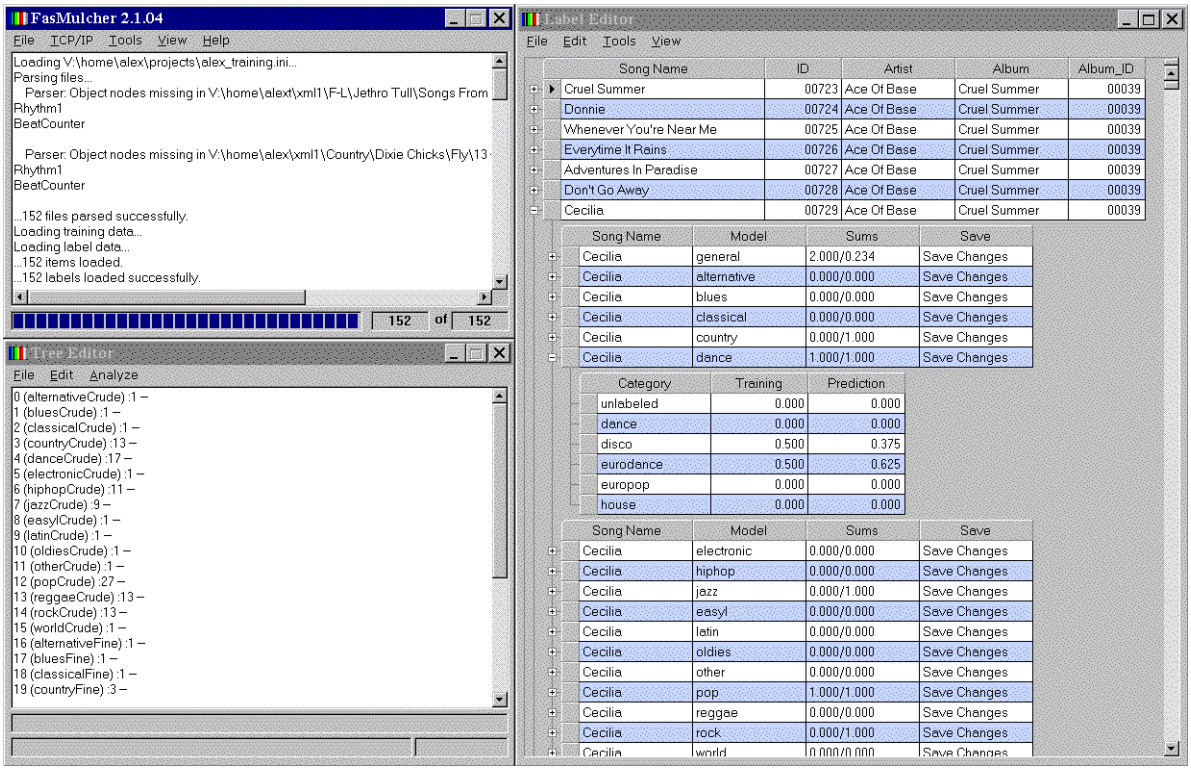

Figure: CART Training View

The FASTLab Debugging GUI

The FASTLab GUI was originally developed for use in debugging the analysis system; its display is divided into several regions, and the GUI controls allow one to show and hide any of several views. The screen shots on the next two pages show the various features of the debugging GUI, and give the reader a feel for the sophistication of the FASTLab meta-data derivation facilities.

The top-most view is always the time-domain function (the signal waveform). The user can zoom in and out and scroll through the selected song, playing the audio data from any point with a single mouse click. The panes below are always locked to the zoom / scroll settings of the top pan. This is very useful when adding new analysis routines or instrument or style "discriminators."

Below the signal waveform pane, one can select to see the spectrum derived through FFT or LPC analysis, the tempo and beat tracking measures, or other data.

The first screen shot below is the basic FASTLab GUI display showing a section of a song and its frequency spectrum. The various controls under the spectrum display allow a developer to select and apply various filters and windows to the signal, to change the parameters of the formant trackers, and to re-run the analysis.

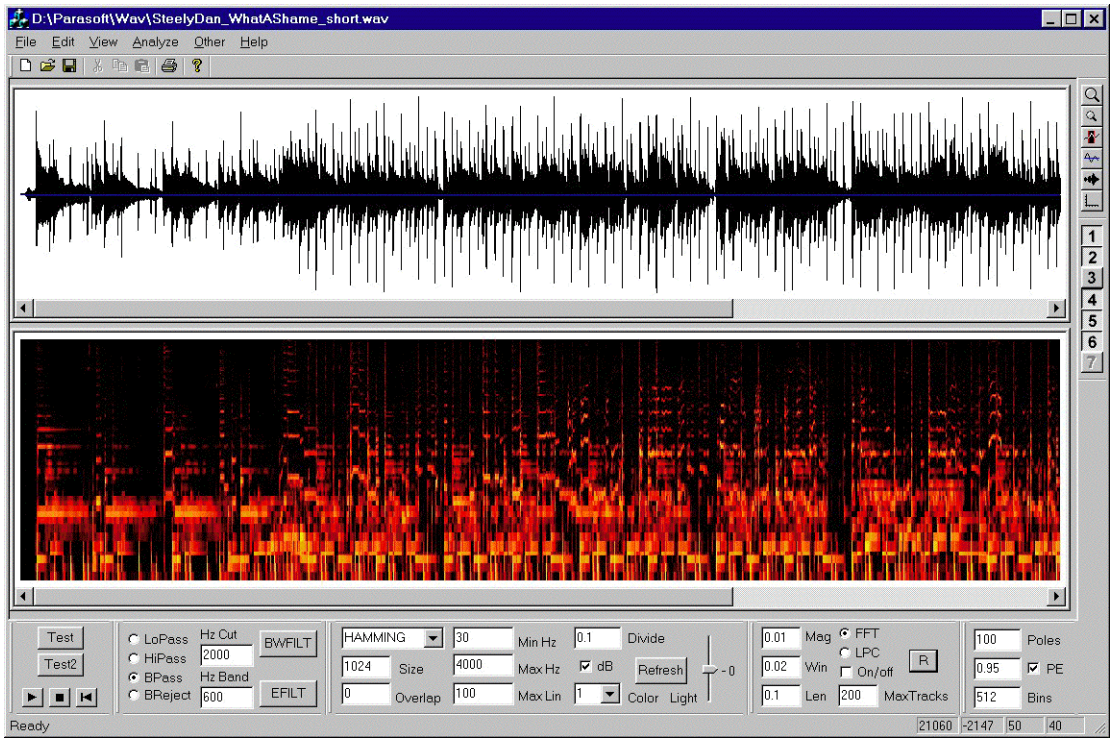

Figure: Basic Debugging GUI

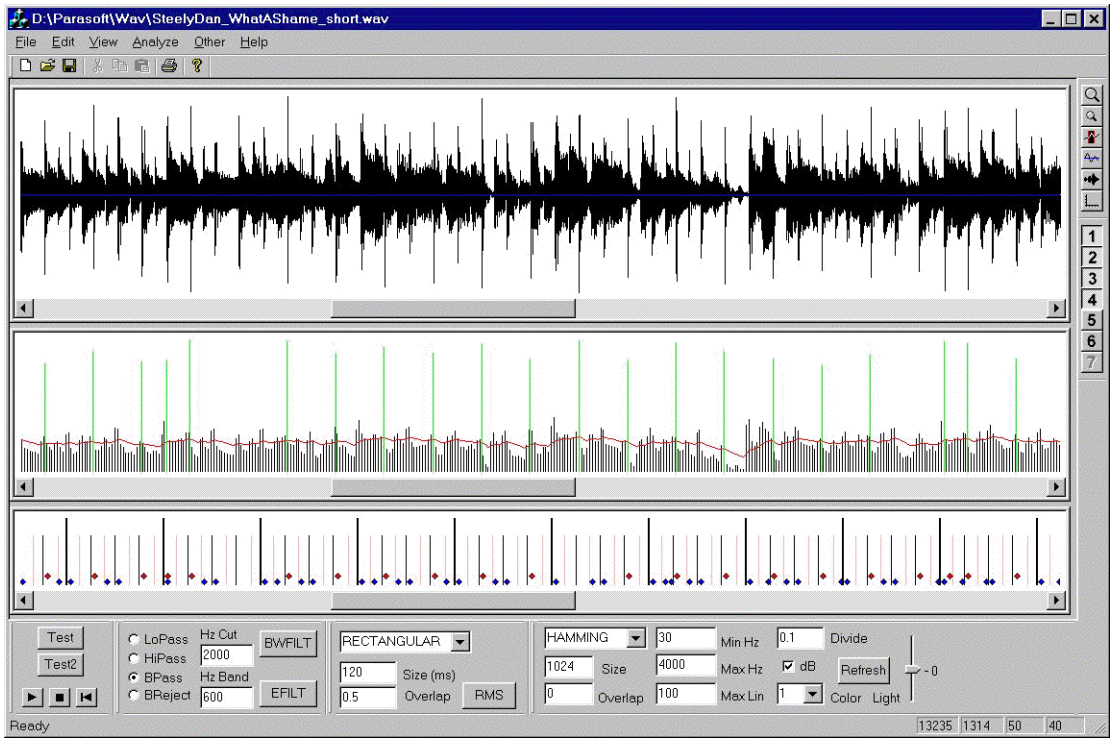

Figure: Beat Analyzer GUI

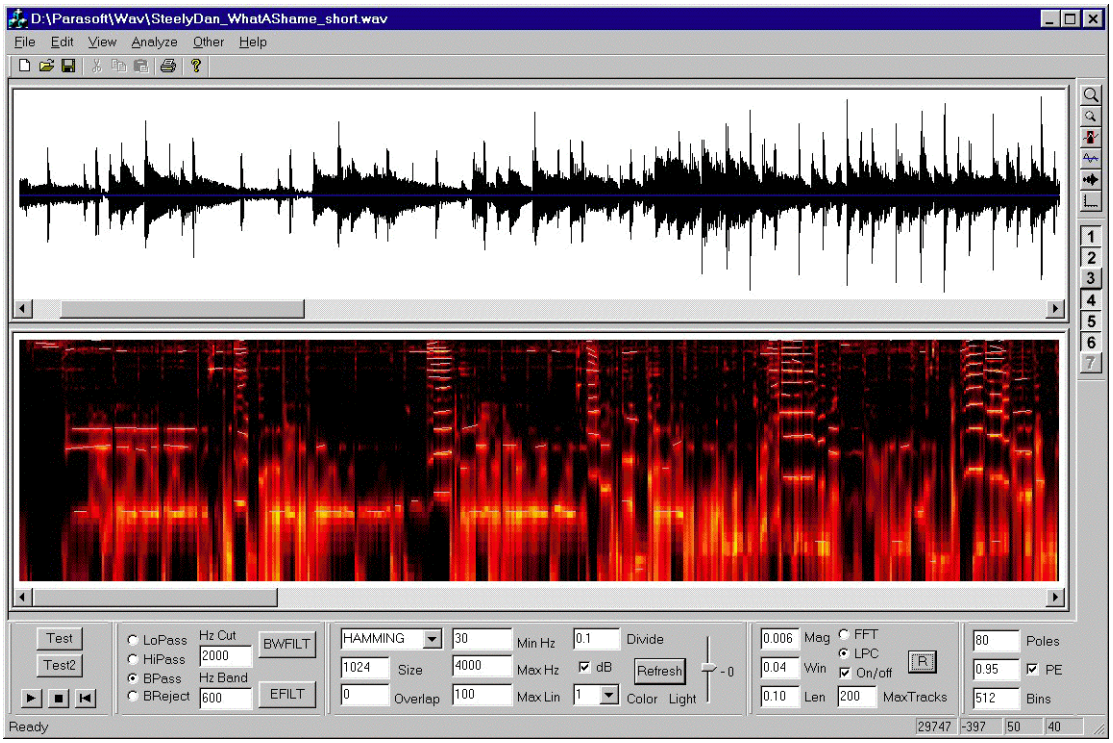

Figure: LPC Tracking GUI

The second screen shot (on the previous page) shows the signal view and the two parts of the beat analyzer: the basic pulse detector (middle pane), and the intelligent beat/tempo analysis results (in the bottom pane). In the example shown here, the system has identified that the given song by Steely Dan is in strict 4/4 time.

The final screen shot (above) illustrates the linear prediction (LPC) analysis and tracking results; in this view, one can see the LPC-derived spectrum and the tracked spectral components (light horizontal lines in the spectrum) that were found by the analyzer.

Implementation Notes

The bulk of FASTLab is written in C++ and is largely cross-platform (i.e, to MacOS or Linux).

The system comprises approximately 40,000 lines of C++ and 6,000 lines of Visual-Basic (for the CART training framework, not needed to run the system once it is trained).

Note: The name FASTLab comes from the first names of its three developers: Frode Holm, Alex Kouznetsov, and Stephen Travis Pope.

Conclusion

The FASTLab analysis engine is able to listen to musical material and perform both fine-grained analysis and detailed stylistic classification. We have introduced several potential applications above, and believe there are many more that will arise in the future.

We have surveyed the competing technologies in detail, and believe that FASTLab is unique in its sophistication and comprehensive approach, so that it represents “best of breed” technology for music classification, preference matching, computer-assisted mastering, and music finger-printing.

Contact Information

Stephen Travis Pope
FASTLab, Inc.
Tel: (805) 895-6252
Email: info@FASTLabInc.com