Proc. of NCC 2010, Chennai, India

A Melody Detection User Interface for Polyphonic Music

Sachin Pant, Vishweshwara Rao, and Preeti Rao
Department of Electrical Engineering
Indian Institute of Technology Bombay, Mumbai 400076, India
Email: {sachinp, vishu, prao}@ee.iitb.ac.in

Abstract— **The automatic extraction of the melody of the music from polyphonic recordings is a challenging problem for which no general solutions currently exist. We present a novel interface for semi-automatic melody extraction with the goal to provide highly accurate pitch tracks of the lead voice with minimal user intervention. Audio-visual feedback facilitates the validation of the obtained melodic contour and user control of the analysis parameters enables direct and effective control over the voice pitch detection module by the intelligent user. This paper describes the interface and discusses the features in which it differs from more conventional audio analyses interfaces.**

I. Introduction

Melody extraction from polyphony finds application as a front end in several music information retrieval (MIR) based applications, such as query-by-humming, cover song identification and audio-to-score alignment, as well as musicology and pedagogy. A rough definition of the melody of a song is the monophonic pitch sequence that a listener might reproduce if asked to hum a segment of polyphonic music [1]. Polyphony indicates that more than one musical sound source may be simultaneously present. The melodic pitch sequence is usually manifested as the fundamental frequency (F0) contour of the lead musical instrument in the polyphonic mixture (here considered to be the singing voice). Although there exists a considerable body of work in pitch extraction from monophonic (single-source) audio, advances in research that enable melodic pitch extraction from polyphonic audio (a harder problem because of increased signal complexity due to polyphony) have only recently been made (in the last decade).

The melody extraction problem comprises of two sub-problems viz. melodic pitch detection and sung vocal segment detection. Both these sub-problems continue to be far from solved for use in practical automatic music transcription systems for operation on large datasets of polyphonic music across various genres and styles. However the applicability of available algorithms can be extended considerably by employing semi-automatic approaches tailored for specific applications [2].

This paper describes a graphical user interface (GUI) for semi-automatic melody extraction from polyphonic music. There were several motivating factors behind the design of the proposed GUI. Melody based reference templates required for the searchable database in query-by-humming systems must be extracted from polyphonic soundtracks. High accuracy voice-pitch tracks from available commercial recordings of classical music performances can be useful for musicological analyses as well as serve in music tutoring applications for Indian classical singing. The final objective in the design of the interface is to facilitate the extraction and validation of the voice pitch from polyphonic music with minimal human intervention. Since the manual marking of vocal segment (sung phrase) boundaries is much easier than automatic detection of the frame-by-frame voice pitch, the focus on the design of the back-end melody extraction program has been on automatic and high accuracy vocal pitch extraction [3].

Publicly available pitch detection interfaces, such as PRAAT [4] and the Aubio pitch detector VAMP plugin [5] for the Sonic Visualizer program [6], have been designed for monophonic audio and cannot be expected to perform acceptably well on polyphonic audio. However, a few polyphonic transcription tools also exist. The polyphonic transcription VAMP plugin [7] has been designed exclusively for guitar and piano music and outputs MIDI notes. Melodyne's Direct Note Access (DNA) [8] attempts to isolate simultaneously played musical notes but is not freely available for evaluation. Neither of these programs attempts to extract a single, high-resolution melodic contour (i.e. pitch track of the lead voice) from polyphonic audio.

To best of our knowledge, the only interface specifically designed to facilitate melodic pitch extraction from polyphonic audio is MiruSinger [9]. MiruSinger was primarily developed for improving singing skills by comparison between the user's sung pitch contour and the melody extracted from original CD polyphonic recordings. It uses another contemporary melody extraction algorithm as a back-end [10]. The MiruSinger melody extraction interface offers several helpful features such as re-synthesis of the extracted melodic contour for validation, user correction of detected vocal segments and user correction of the extracted melodic pitch by choosing from different local salient F0 candidates. However there is no flexibility in terms of parametric control of the back-end executable.

In our interface we have included some features from the MiruSinger interface and have also added some novel features, which we feel will further ease the melody extraction process. The next section describes the design layout and operation of our melody extraction interface. Section 3 brings out the salient features of interface with respect to facilitating melody extraction and refinement. Section 4 lists the development details. Section 5 presents a summary and some future enhancements to be made to the interface.

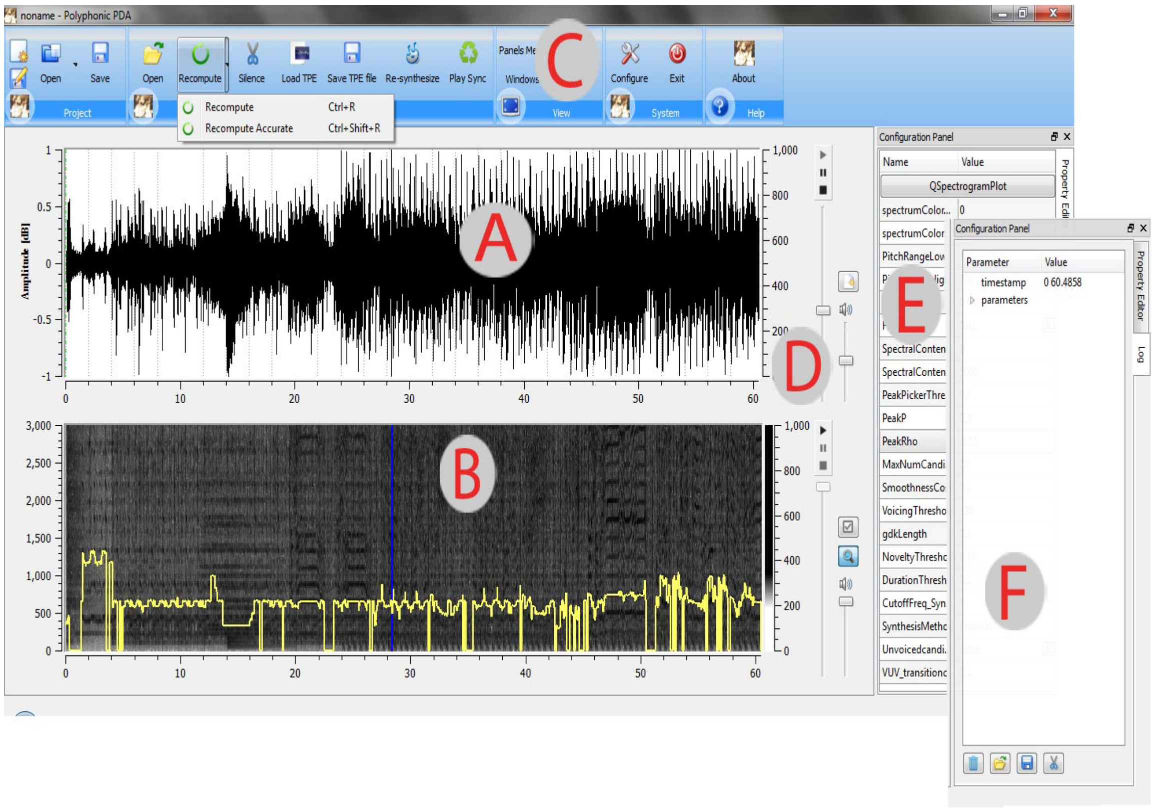


Fig. 1. Snapshot of melody extraction interface

II. Interface: Description And Operation

The basic features that are expected from any interface intended for audio analysis/editing comprise of waveform and spectrogram displays, selection and zooming features, and audio playback. In this section we describe the layout of the interface that, in addition to these basic features, also has features designed for the melody extraction task. The operation of the interface is also described.

A. Description

A snapshot of the interface is provided in Fig. 1. It consists of a waveform viewer (A), a spectrogram viewer (B), a menu bar (C), controls for audio viewing, scrolling and playback (D), a parameter window (E), a log viewer (F). The waveform and spectrogram of the audio are displayed in A & B respectively. The horizontal axis corresponds to the timeline, with data moving from left to right. The right vertical axis in B corresponds to F0 frequency. The vertical bar at the center is the “present time”. Controls (D) are provided for playing back the audio, controlling the volume, and the progress of the audio. Moving markers provide timing information. The menu bar (C) and the parameter window (E) control the use of the melody extractor. A log viewer (F) is provided to save and load the analysis parameters for the different segments.

B. Audio Analysis

The audio example to be analyzed, in .wav format, is loaded into the interface. The waveform and spectrogram of the music are automatically displayed. The melody extractor is invoked by pressing the ↻ button in the menu. By default, the melody extractor is called in single-F0 tracking mode, which is found to perform quite accurately on most audio examples. Alternatively the user may also select to use a more accurate, but slower, melody extraction algorithm (See Section III.F.) by checking the dual-F0 option in the drop-down menu under the ↻ button. This function is especially useful when an accompanying pitched instrument is of comparable, or greater, loudness than the voice. The resulting pitch contour is displayed as a yellow curve in B. The estimated F0 contour is plotted on top of the spectrogram, which helps visually validate the estimated melody by observing the shape and/or the extent of overlap between the pitch contour and any of the voice harmonics. Voice harmonics are typically characterized by their jittery/unsteady nature. Audio feedback is also provided by pressing the ▶ button on the right of the spectrogram. This plays back a natural re-synthesis of the estimated F0 contour. The extracted pitch contour of the entire audio clip can be synthesized using the ♺ button from the menu (C).

C. Saving and Loading Sessions

The interface provides an option of saving the final complete melody as well as the parameters used for computing melody for different selected regions by using the 💾 button. A user can save the pitch extracted in a specific file format (TPE), which has three columns containing the Time stamp (in sec), the Pitch (in Hz), and the frame-level signal Energy respectively. This amounts to saving a session. This TPE file can be loaded later for further analysis and processing. Also, the parameters of the melody extractor used during the analysis can be saved in an XML file.

III. Interface: Salient Features

As mentioned before we have attempted to incorporate some features from the MiruSinger melody extraction interface, with further enhancements, and also have incorporated some new features that increase functionality of the interface. The salient features of our melody extraction interface are described below.

A. Novel Melody Extractor

The melody extraction back-end system used by our interface has been extensively evaluated on polyphonic vocal music and has demonstrated very accurate voice pitch extraction performance. We found that the design considerations we made also resulted in performance on par with state-of-the-art systems when evaluated at the Audio Melody Extraction Task at MIREX 2008[1] [3]. The system utilizes a spectral harmonic-matching pitch detection algorithm (PDA) followed by a computationally-efficient, optimal-path finding technique that tracks the melody within musically-related melodic smoothness constraints. An independent vocal segment detection system then identifies audio segments in which the melodic line is active/silent by the use of a melodic pitch-based energy feature.

Further our melody extraction system uses non-training-based algorithmic modules i.e. is completely parametric, unlike those that incorporate pattern classification or machine learning techniques [11], [12]. The performance of such systems is highly dependent on the diversity and characteristics of the training data available. In polyphonic music the range of accompanying instruments and playing (particularly singing) styles across genres are far too varied for such techniques to be generally applicable. When using our interface users, with a little experience and training, can easily develop an intuitive feel for parameter selections that result in accurate voice-pitch contours.

B. Validation

The user can validate the extracted melodic contour by a combination of audio (re-synthesis of extracted pitch) and visual (spectrogram) feedback. We have found that by-and-large the audio feedback is sufficient for melody validation except in the case of rapid pitch modulations, where matching the extracted pitch trajectory with that of a clearly visible harmonic in the spectrogram serves as a more reliable validation mechanism.

Currently there are two options for re-synthesis of the extracted voice-pitch contour. The default option is for a natural synthesis of the pitch contour. This utilizes the harmonic amplitudes as detected from the polyphonic audio resulting in an almost monophonic playback of the captured pitch source. This type of re-synthesis captures the phonemic content of the underlying singing voice that serves as an additional cue for validation of the extracted pitch. However, in the case of low-energy voiced utterances especially in the presence of rich polyphonic orchestration it was found that harmonics from other instruments also get synthesized, which may confuse the user.

In such cases, an alternate option of complex-tone synthesis with equal amplitude harmonics also exists. Here the user will have to use only the pitch of the audio feedback for validation since the nature of the complex tone is nothing like the singing voice. In complex tone synthesis the frame-level signal energy may be used but we have found that this leads to audible bursts especially if the audio has a lot of percussion. Alternatively we have also provided a constant-energy synthesis option which allows the user to focus on purely the pitch content of the synthesis and not be distracted by sudden changes in energy. This option can be selected from the parameter list (E). An additional feature that comes in handy during melodic contour validation is the simultaneous, time-synchronized playback of the original recording and the synthesized output. This can be initiated by clicking the ♻ button on the menu (C). A separate volume control is provided for the original audio and synthesized playback. By controlling these volumes separately, we found that users were able to make better judgments on the accuracy of the extracted voice-pitch.

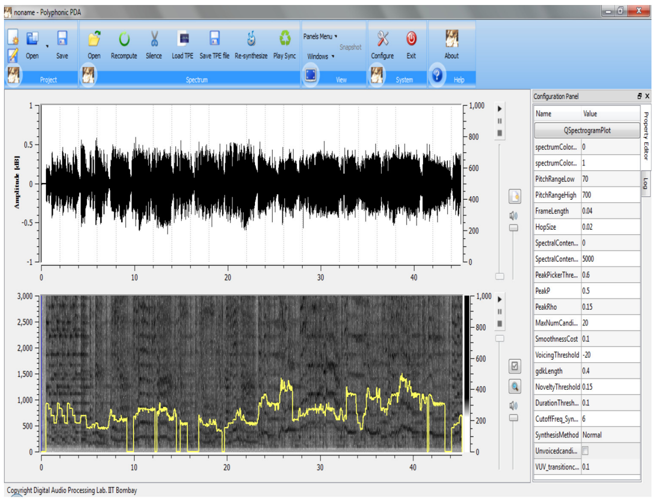
C. Inter-segment Parameter Variation

Typical parameters that affect the performance of our melody extraction system are the F0 search range, frame-length, lower-octave bias and melodic smoothness tolerance. An intelligent user will be able to tune these parameters, by observing certain signal characteristics, to obtain a correct output melody. For example, in the case of male singers, who usually have lower pitch than females, lowering the F0 search range and increasing the frame-length and lower-octave bias results in an accurate output. In the case of large and rapid pitch modulations, increasing the melodic smoothness tolerance is advisable.

It may sometimes be possible to get accurate voice-pitch contours by using a fixed-set of analysis parameters for the whole audio file. But many cases were observed, especially of male-female duet songs and excerpts containing variations in rates of pitch modulation, where the same parameter settings did not result in an accurate pitch contour for the whole file. In order to alleviate such a problem the interface allows different parameters to be used for different segments of audio. This allows for easy manipulation of parameters to obtain a more accurate F0 contour. The parameter window (E) provides a facility to vary the parameters used during analysis.

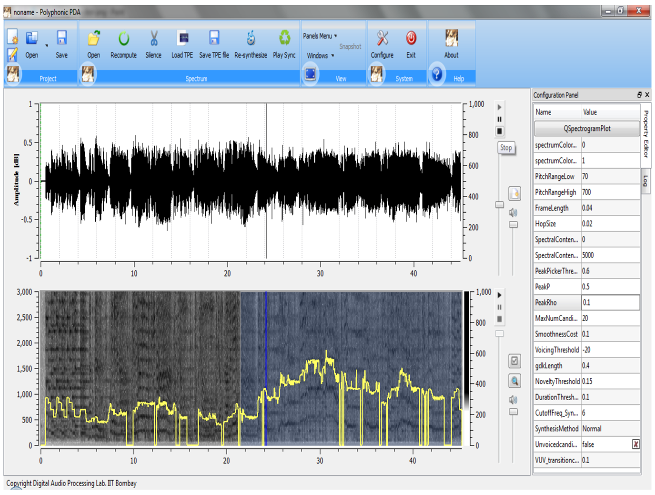
To emphasize the use of this feature i.e. parameter variation for different audio segments, we consider the analysis of a specific excerpt from a duet Hindi film song. The audio clip has two phrases, the first sung by a male singer and the latter by a female singer. Fig. 2 shows a snapshot of our interface when a single set of parameters is used on the entire audio clip. It can be seen that the pitch has been correctly estimated for the part (first half) sung by the male singer, but there are errors for the female part (second half). This becomes more evident by observing the spectrogram display closely or also by listening to the re-synthesis of the extracted pitch contour. Selecting the female portion from the song and computing its pitch using a slightly modified set of parameters (reduced frame-length and lower octave bias) leads to much better estimate of female voice-pitch contour (as shown in Fig. 3).

[1] These annually-held evaluations provide a framework for formally assessing different Music Information Retrieval (MIR) systems for a wide variety of tasks including melody extraction, chord detection, and artist classification, on common test data sets. Detailed results are available at music-ir.org/mirex/2008/index.php/Audio_Melody_Extraction_Results



Name	Value
QSpectrogramPlot	
spectrumColor...	0
spectrumColor...	1
PitchRangeLow	70
PitchRangeHigh	700
FrameLength	0.04
HopSize	0.02
SpectralConten...	0
SpectralConten...	5000
PeakPickerThre...	0.6
PeakP	0.5
PeakRho	0.15
MaxNumCandi...	20
SmoothnessCost	0.1
VoicingThreshold	-20
gdkLength	0.4
NoveltyThreshold	0.15
DurationThresh...	0.1
CutoffFreq_Syn...	6
SynthesisMethod	Normal
Unvoicedcandi...	☐
VUV_transitionc...	0.1

Fig. 2. Analysis of Hindi film duet song clip showing incorrect pitch computation i.e. octave errors, in the downward direction, in the extracted pitch contour (yellow) are visible towards the second half (female part)



Name	Value
QSpectrogramPlot	
spectrumColor...	0
spectrumColor...	1
PitchRangeLow	70
PitchRangeHigh	700
FrameLength	0.04
HopSize	0.02
SpectralConten...	0
SpectralConten...	5000
PeakPickerThre...	0.6
PeakP	0.5
PeakRho	0.1
MaxNumCandi...	20
SmoothnessCost	0.1
VoicingThreshold	-20
gdkLength	0.4
NoveltyThreshold	0.15
DurationThresh...	0.1
CutoffFreq_Syn...	6
SynthesisMethod	Normal
Unvoicedcandi...	false
VUV_transitionc...	0.1

Fig. 3. Analysis of Hindi film duet song clip showing correct pitch computation. The pitch contour (yellow) of the selected segment was recomputed after modifying some parameters.

D. *Non-Vocal Labeling*

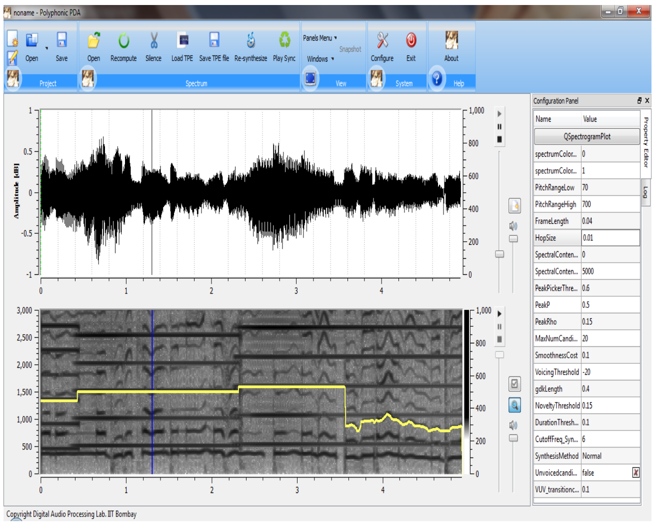
Even after processing, there may be regions in the audio which do not contain any vocal segments but for which melody has been computed. This occurs when an accompanying, pitched instrument has comparable strength as the voice because the vocal segment detection algorithm is not very robust to such accompaniment. In order to correct such errors we have provided a user-friendly method to zero-out the pitch contour in a non-vocal segment by using the ✂ tool from the menu (C).

E. *Saving Final Melody and Parameters*

The melody computed can be saved and later used for comparison or any MIR tasks. The log component (F) records parameters used for analysis with time-stamps representing the selected regions. By studying these log files for different audio clips we gain insight into optimal parameter settings for different signal conditions. For example, one observation made was that higher frame-lengths resulted in more accurate pitch contours for audio examples in which a lot of instrumentation (polyphony) was present but was found to be detrimental to performance when rapid pitch modulations were present. This has led us to investigate the use of a signal-driven adaptive time-frequency representation in the melody extraction back-end.

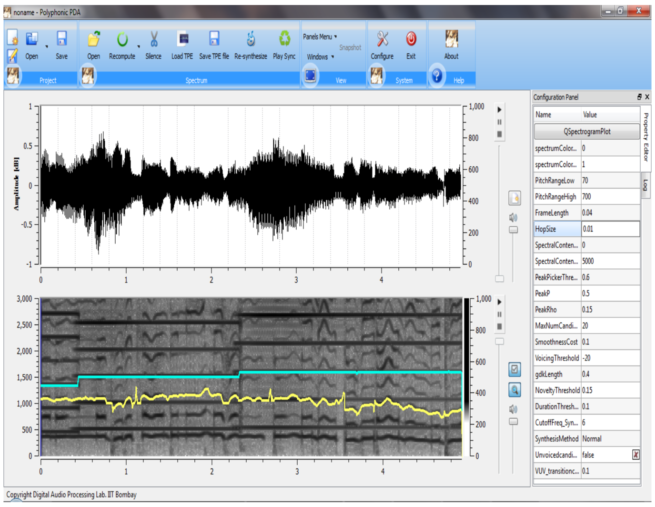
F. *Error Correction by Selective Use of Dual-F0 Back-end*

State-of-the-art melody extraction algorithms have been known to incorrectly detect the pitches of loud, pitched accompanying instruments as the final melody, in spite of the voice being simultaneously present. Recently, however, we have shown that attempting to track two, instead of a single, pitch contours can result in a significant improvement in system performance [13]. Specifically, the path finding technique in the melody extraction algorithm is modified to track the path of an ordered pair of possible pitches through time. Pairing of pitches is done under harmonic constraints i.e. two pitches that are integer (sub) multiples of each other cannot be paired.



Name	Value
QSpectrogramPlot	
spectrumColor...	0
spectrumColor...	1
PitchRangeLow	70
PitchRangeHigh	700
FrameLength	0.04
HopSize	0.01
SpectralConten...	0
SpectralConten...	5000
PeakPickerThre...	0.6
PeakP	0.5
PeakRho	0.15
MaxNumCandi...	20
SmoothnessCost	0.1
VoicingThreshold	-20
gdkLength	0.4
NoveltyThreshold	0.15
DurationThresh...	0.1
CutoffFreq_Syn...	6
SynthesisMethod	Normal
Unvoicedcandi...	false
VUV_transitionc...	0.1

Fig. 4. Analysis of an audio clip containing voice and loud harmonium using the single-F0 option. The extracted pitch contour (yellow) mainly tracks the harmonium pitch and only switches to the voice pitch towards the end of the clip.



Name	Value
QSpectrogramPlot	
spectrumColor...	0
spectrumColor...	1
PitchRangeLow	70
PitchRangeHigh	700
FrameLength	0.04
HopSize	0.01
SpectralConten...	0
SpectralConten...	5000
PeakPickerThre...	0.6
PeakP	0.5
PeakRho	0.15
MaxNumCandi...	20
SmoothnessCost	0.1
VoicingThreshold	-20
gdkLength	0.4
NoveltyThreshold	0.15
DurationThresh...	0.1
CutoffFreq_Syn...	6
SynthesisMethod	Normal
Unvoicedcandi...	false
VUV_transitionc...	0.1

Fig. 5. Analysis of an audio clip containing voice and loud harmonium using the dual-F0 option. The system outputs two pitch contours (yellow and blue). The yellow contour in this case represents the voice pitch.

The use of the above 'dual-F0 tracking' approach results in a considerable increase in computation time and may not be practically viable for long audio segments. However, we have provided the option for the user to selectively apply such an analysis approach i.e. track 2 F0s. On selecting this option (by selecting the dual-F0 option in the drop-down menu under the ↻ button) the system will output 2 possible, melodic contours. This is much cleaner than presenting the user with multiple locally-salient F0 candidates, as this will clutter up the visual display. The user can listen to the re-synthesis of each of these contours and select any one of them as the final melody. Typically we expect users to use this option on segments on which the single-F0 melody extractor always outputs some instrument pitch contour despite trying various parameter settings.

To emphasize the performance improvement on using the dual-F0 tracking option we consider the analysis of an audio clip in which the voice is accompanied by a loud harmonium. Fig. 4 displays the result of using the melody extractor in single-F0 mode. The resulting pitch contour can be seen to track the harmonium, not the voice, pitch for a major portion of the file. It is not possible to make any parameter changes in order to correct the output. By using the dual-F0 tracking option (Fig. 5) we can see that now two contours are output: the yellow represents the voice pitch and the light blue represents the harmonium pitch. The user can now select one of these two contours as the final output melody.

IV. Development Details

The graphical interface has been developed using Qt [14] and Qwt [15] toolkit. The interface uses generic component framework (GCF) [16] developed by VCreateLogic. The interface is written in C++. Qt because of its cross-compilation capabilities enables deployment of our system on a variety of Platforms. GCF on the other hand provides component based architecture making development and deployment easier.

V. Summary

In this paper we have presented a graphical user interface for semi-automatic melody extraction based on a recently proposed algorithm for voice-pitch extraction from polyphonic music. This interface is shown to display several novel features that facilitate the easy extraction of melodic contours from polyphonic music with minimal human intervention. It is presently being used for melody extraction from large durations of Indian classical music to facilitate studies on *Raga* identification [17] and also on Hindi film music for extraction of reference templates to be used in a query-by-humming system [18]. We are working towards making this interface available to fellow researchers who are interested in analyses of polyphonic music signals.

The spectrogram display (B) is useful for validation of the extracted melodic contour, as described in Section II.B. However, the 'melodic range spectrogram' (MRS) as used in the 'Sonic Visualiser' program [6] would be much more appropriate. The MRS is the same as the spectrogram except for different parameter settings. It only displays output in the range from 40 Hz to 1.5 kHz (5.5 octaves), which most usually contains the melodic content. The window sizes are larger with heavy overlap for better frequency resolution. The vertical frequency scale is logarithmic i.e. linear in perceived musical pitch. Finally the colour scale is linear making noise and low-level content invisible but making it easier to identify salient musical entities. The integration of the MRS into our interface will be taken up in the future.

VI. References

[1] G. Poliner, et. al., "Melody transcription from music audio: Approaches and evaluation," *IEEE Trans. Audio, Speech, Lang., Process*., vol. 15, no. 4, pp. 1247–1256, May 2007.

[2] Y. Wang and B. Zhang, "Application-specific music transcription for tutoring," *IEEE Multimedia*, vol. 15, no. 3, pp. 70–74, 2008.

[3] V. Rao and P. Rao, "Melody extraction using harmonic matching," in *MIREX Audio Melody Extraction Contest Abstracts*, Philadelphia, 2008.

[4] P. Boersma and D. Weenink, "Praat: Doing phonetics by computer (Version: 5.0.44)," [Computer program], Retrieved August 1, 2009, from http://www.praat.org.

[5] P. Brossier, "Aubio: a library for audio labeling," [Computer program], Retrieved Aug 27, 2009, from http://aubio.org

[6] C. Cannam, "Sonic Visualiser," [Computer program], Retrieved: August 27, 2009, from http://www.sonicvisualiser.org.

[7] R. Zhou, "Polyphonic transcription VAMP plugin," [Computer program], Retrieved: August 27, 2009, from http://isophonics.net/QMVampPlugins.

[8] Celemony, "Direct Note Access: the new Melodyne dimension," Online: http://www.celemony.com/cms/index.php?id=dna

[9] T. Nakano, M. Goto and Y. Hiraga, "MiruSinger: A singing skill visualization interface using real-time feedback and music CD recordings as referential data," in *Proc. of the 9th IEEE Intl. Symposium on Multimedia Workshops, 2007*.

[10] M. Goto, "A real-time music scene description system: predominant-F0 estimation for detecting melody and bass lines in real-world audio signals," *Speech Comm.,* vol. 43, no. 4, pp. 311–329, 2004.

[11] G. Poliner and D. Ellis, "A classification approach to melody transcription," in *Proc. Intl. Conf. Music Information Retrieval,* London, 2005.

[12] H. Fujihara et. al. "F0 estimation method for singing voice in polyphonic audio signal based on statistical vocal model and Viterbi search," in *Proc. IEEE Intl. Conf. Audio Speech and Sig. Process*., Toulouse, France, 2006.

[13] V. Rao, and P. Rao, "Improving polyphonic melody extraction by dynamic programming based multiple F0 tracking," in *Proc. 12th Intl. Conf. Digital Audio Effects (DAFx-09),* Como, Italy, Sept. 2009.

[14] Qt, A cross-platform application and C++ UI framework http://qt.nokia.com/

[15] Qwt Library *http://qwt.sourceforge.net*

[16] GCF-A custom component framework *http://www.vcreatelogic.com/products/gcf/*

[17] S. Belle, P. Rao and R. Joshi, "Raga identificiation by using swara intonation," *Frontiers of Research in Speech and Music* (*FRSM*), Gwalior, India, 2009.

[18] M. Raju, B. Sundaram and P. Rao, "TANSEN: A query-by-humming based music retrieval system," in *Proc. National Conf. on Communications*, Chennai, 2003.