ON THE DEFINITION OF MUSICAL NOTES FROM PITCH TRACKS FOR MELODY DETECTION IN POLYPHONIC RECORDINGS

Rui Pedro Paiva, Teresa Mendes and Amílcar Cardoso

CISUC – Center for Informatics and Systems of the University of Coimbra
Department of Informatics Engineering, University of Coimbra (Polo II), P3030, Coimbra, Portugal
{ruipedro, tmendes, amilcar}@dei.uc.pt

ABSTRACT

The present study addresses the problem of defining musical notes from pitch tracks, in the context of a system for melody detection in polyphonic musical signals. This is an important issue for melody transcription, as well as melody-based music information retrieval. Previous work in the area tackled mainly the extraction of melodic pitch lines, without explicit determination of musical notes. Therefore, in this paper we propose an approach for the creation of musical notes based on a two-stage segmentation of pitch tracks. In the first step, frequency-based segmentation is carried out through the detection of frequency variations in pitch tracks. In the second stage, salience-based segmentation is performed so as to split consecutive notes with equal value, by making use of salience minima and note onsets.

1. INTRODUCTION

As a result of recent technological innovations, there has been a tremendous growth in the Electronic Music Distribution industry. Factors like the widespread access to the Internet, bandwidth increasing in domestic accesses or the generalized use of compact audio formats with CD or near CD quality, such as mp3, have given a great contribution to that boom. Presently, it is expected that the number of digital music archives, as well as their dimension, grow significantly in the near future, both in terms of music database size and in number of genres covered.

However, any large music database is only really useful if users can find what they are looking for in an efficient manner. Today, whether it is the case of a digital music library, the Internet or any music database, search and retrieval is carried out mostly in a textual manner, based on categories such as author, title or genre. This approach leads to a certain number of difficulties, namely in what concerns database search in a transparent and intuitive way. Therefore, in order to overcome the described limitations, research is being conducted in an emergent and promising field called Music Information Retrieval.

Query-by-humming (QBH) is a particularly intuitive way of searching for a musical piece, since melody humming is a natural habit of humans. Several techniques have been proposed in order to attain that goal, e.g., [3]. However, presently, this work is being carried out only in the MIDI domain, which places important usability questions. Querying "real-world" polyphonic recorded musical pieces requires a melody representation of the songs. This is a complex task since many types of instruments can be playing at the same time, with severe spectral interference between each other.

Previous work concerned with obtaining symbolic representations from musical audio has concentrated especially on the problem of full music transcription, which requires accurate multi-pitch estimation for the extraction of all fundamental frequencies (FF) present in a song under analysis, e.g., [1], [7]. However, the present solutions are neither sufficiently general nor accurate. In fact, the proposed approaches impose several constraints on the music material, namely on the maximum number of concurrent instruments, musical style or type of instruments present.

Only little work has been conducted in the problem of melody detection in polyphonic audio, e.g., [4],[5],[8],[9]. Additionally, most of the work is only concerned with the extraction of melodic pitch lines. Therefore, we propose an approach for melody detection with explicit note determination, in terms of note values, onsets and offsets. An overview of the system is given in Section 2. Definition of musical notes, the main topic of this paper, is the subject of Section 3. In Section 4, evaluation results are discussed.

2. MELODY DETECTION SYSTEM

Our multi-stage strategy for melody detection consists of five modules, as illustrated in Figure 1. The general strategy was described previously, e.g., [9], [10]. Thus, only a brief description is offered here, for the sake of completeness. New improvements to the trajectory segmentation stage are described in more detail.

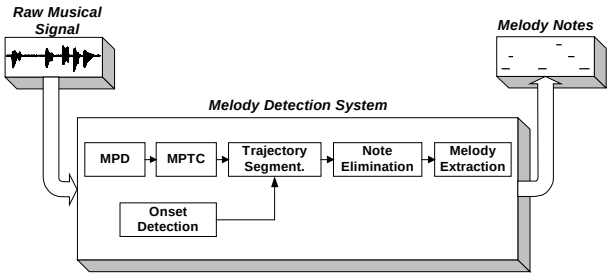

Figure 1: *Overview of the melody detection system.*

2.1. Multi-Pitch Detection (MPD)

In the first stage of the algorithm, the objective is to capture a set of pitch candidates, which constitute the basis of possible future notes. The MPD algorithm receives as input a raw musical signal (monaural, sampling frequency f_s = 22050 Hz, 16 bits quantization) and outputs a set of pitch candidates and respective saliences, related to the energy of the FF. We perform pitch detection in a frame-based analysis, defining a 46.44 ms frame length and a hop size of 5.8 ms.

Then, we conduct an auditory model based analysis of each frame, in order to detect the most salient pitches in each. This analysis is performed in four stages: i) conversion of the sound waveform into auditory nerve responses for each frequency channel, using a model of the ear with particular emphasis on the cochlea, obtaining a so-called cochleagram; ii) detection of the main periodicities in each frequency channel using auto-correlation, from which a correlogram results; iii) detection of the global periodicities in the sound waveform by calculation of a summary correlogram (SC); and iv) detection of the pitch candidates in the frame by looking for the most salient peaks in the SC.

The approach undertaken is based on an auditory model proposed by Slaney and Lyon [14].

2.2. Multi-Pitch Trajectory Construction (MPTC)

The second stage of our algorithm aims to create a set of pitch trajectories, formed by connecting consecutive pitch candidates with similar frequencies. To this end, we based ourselves on the algorithm proposed by Serra [13]. The general idea is to find regions of stable pitches, which indicate the presence of musical notes. In order not to loose information on the dynamic properties of musical notes, e.g., frequency modulations, glissandos, we had especial care in guaranteeing that such behaviors were kept within a single track. Therefore, each trajectory may contain more than one note and should, therefore, be segmented in the next stage.

2.3. Trajectory Segmentation

The trajectories that result from the MPTC algorithm may contain more than one note and, therefore, have to be segmented. Such segmentation is performed in two stages: frequency and salience segmentation. This is the main topic of this paper and is described in the following section.

2.4. Note Elimination

The objective of the fourth stage of the melody detection algorithm is to delete irrelevant note candidates, based on their saliences, durations and on the analysis of harmonic relations.

First, low-salience notes are eliminated. Next, all the notes that are too short are also deleted. Finally, we look for harmonic relations between notes, based on the fact that some of the obtained pitch candidates are sub or super-harmonics of real pitches in the sound wave. Therefore, we make use of perceptual rules of sound organization, namely "harmonicity" and "common fate" [2].

In the "harmonicity" rule, if two notes have approximately the same onset times and are harmonically related, it is possible that they have come from the same source. This is further validated by the rule of "common fate", where harmonically-related notes can be grouped by exploiting aspects such as common modulation, both in frequency and in amplitude.

2.5. Melody Extraction

In the final stage of the present melody detection system, our goal is to obtain a final set of notes comprising the melody of the song under analysis. In fact, although a significant amount of irrelevant notes are eliminated in the previous stage, there are still many notes present. Hence, we have to extract the ones that convey the main melodic line.

Many aspects of auditory organization influence the perception of melody by humans, for instance in terms of the role played by the pitch, timbre and intensity content of the sound signal. In our approach, we do not attack the problem of source separation. Instead, we base our strategy on two assumptions that we designate as the "salience principle" and the "melodic smoothness principle".

In the salience principle, we exploit the fact that the main melodic line often stands out in the mixture. Thus, in the first step of the melody extraction stage, we select the most salient notes as the initial melody candidates.

One of the limitations of only taking into consideration pitch salience is that the notes comprising the melody are not always the most salient ones. In this situation, erroneous notes may be selected as belonging to the melody, whereas true notes are excluded. This is particularly clear when abrupt transitions between notes are found. In fact, small frequency intervals favor melody coherence, since smaller steps in pitch hang together better [2]. Hence, we improved melody extraction by smoothing the initial melodic contour.

3. DEFINITION OF MUSICAL NOTES

As referred above, the trajectories that result form the MPTC stage may contain more than one note. Hence, they must be segmented, so as to explicitly define musical notes, characterized by a MIDI value and onset and offset times.

Past work in melody detection addressed especially the issue of extracting melodic pitch lines, without explicit definition of musical notes, or using ad-hoc algorithms for segmentation (e.g., segment as soon as MIDI note values change). As Klapuri refers [7], this has turn out to be a difficult problem for some signals, particularly for singing. In this way, we propose a robust algorithm for the segmentation of the obtained pitch tracks, with recourse to frequency and pitch salience segmentation.

The proposed algorithm, though used in the perspective of a system for melody detection in polyphonic musical signals, is envisioned as general approach for segmentation of pitch tracks, either in monophonic or polyphonic contexts.

3.1. Frequency Segmentation

The objective of frequency segmentation is to separate all the notes with different values that are present in each pitch track. This is performed taking into consideration the presence of glissandos and frequency modulation.

The main issue is, then, to approximate a frequency curve by a set of piecewise-constant functions (PCFs), as a basis for track segmentation. However, this is not a trivial task, since musical notes, besides containing regions of approximately stable frequency, also contain regions of transition, where frequency evolves until (pseudo-)stability, e.g., glissando. Moreover, frequency modulation can occur, and so no stable frequency exists. Yet, an average stable fundamental frequency can be determined.

Our problem, could, thus, be characterized as one of finding a set of PCFs that best approximate a frequency curve. As unknown variables we have the number of functions, their respective parameters (only bias, for constant functions), and start and end points.

We have investigated some approaches for PCF, or, generally, piecewise-linear, approximation. Two main paradigms are defined: "characteristic points" (CPs) and "minimum error" (ME) [11]. Algorithms based on CPs do not suit well our needs, e.g., in the

case of frequency modulation. In fact, the optimal function may not pass at the obtained CPs. In this way, we constrained the possible approaches to the second paradigm. As for the ME paradigm, it can be further categorized into two main approaches. In the first one, an upper bound for the global error is specified and the minimum number of functions that satisfies it, and respective parameters, is computed. In the second (less studied) approach, a maximum number of functions is specified, and optimization is conducted in order to minimize the global approximation error. However, these algorithms either require that an analytic expression of the curve be known, or need to test different values for the number of functions.

Therefore, we propose an algorithm for approximation of frequency curves from musical notes by PCFs, taking advantage of some peculiarities of musical signals.

The algorithm starts by filtering the frequency curves of all tracks, in order to fill in missing values, as a result of the MPTC algorithm, which allows for a maximum number of "sleeping" frames in peak continuation. This is carried out by a simple zero-order-hold (ZOH), as in (1). There, $f[k]$ is the frequency value of the k^{th} frame in the current track, in a total of N frames, and $f_F[k]$ denotes the filtered curve.

$$f_F[k] = \begin{cases} f[k], & if\ f[k] \neq 0 \\ f_F[k-1], & if\ f[k] = 0 \end{cases}, \forall_{k \in \{1,\cdots,N\}} \qquad (1)$$

After that, the filtered frequency curve is approximated by PCFs through the quantization of each frequency value to the corresponding MIDI note number, according to (2). There, $f_{MIDI}[k]$ represents the MIDI value corresponding to frequency $f_F[k]$ and F_{ref} is the frequency of MIDI note number zero.

$$f_{MIDI}[k] = \text{round}\left(\frac{\log\left(f_F[k] / F_{ref} \right)}{\log \sqrt[12]{2}} \right),\ F_{ref} \approx 8.1758 \text{ Hz} \qquad (2)$$

Then, PCFs can be directly defined as sequences of constant MIDI values. Generally, it comes (3), where, PC_i represents the i^{th} PCF, defined in the domain D_i and characterized by a sequence of constant MIDI values equal to c_i. Also, the special case of singleton domains is allowed. The total number of PCFs is denoted by *nPC*.

$$\forall_{i \in \{1,\cdots,nPC\}},$$

$$1: \quad D_i = \{a_i, \cdots, b_i\} = \left\{ \begin{matrix} k \in \{1,\cdots,N\} : f_{MIDI}[k] = c_i \ \text{ and} \\ f_{MIDI}[k] = f_{MIDI}[k \pm 1] \end{matrix} \right\} \qquad (3)$$

$$2: \quad PC_i[k] = c_i \quad , \forall_{k \in D_i}$$

However, due to frequency variations resulting from modulation, as well as frequency errors from the MPD stage, fluctuations of MIDI values may be present. Also, glissandos should be kept within one single function. Therefore, $f_{MIDI}[k]$ must be filtered, so as to allow for a more robust determination of PCFs that may represent musical notes. Three stages of filtering are applied in order to appropriately deal with too short note candidates. Hence, PCFs whose length (i.e., the number of elements in the domain) is below 125ms (22 frames, using the defined hop size) are dealt with.

In the first filtering stage, the general idea is that short sequences delimited by long sequences with the same note number, e.g., {70,…,70,71,71,70,70,71,70…,70}, are interpreted as possible frequency modulation regions. So, they should be filtered, keeping the value of the delimiting sequence (70 in this example. This is exemplified in Figure 2.

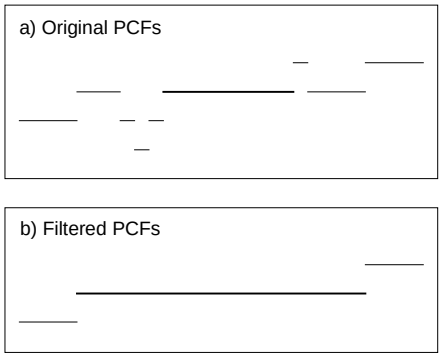

Figure 2: *Filtering of delimited sequences.*

However, some short PCFs are still kept. Therefore, a similar filtering is applied, much in the same way as in Figure 2, with the difference that no long sequences need to be found.

At this moment, the only short PCFs present correspond to glissandos. Therefore, we look for a succession of increasing or decreasing short notes (corresponding to the transition region) and possibly ending with a long note. Here, if the final PCF in the sequence is long, the new PCF keeps its the value, since there is strong evidence that the glissando evolves until the final note. Otherwise, if the sequence only contains short notes and if the duration of the whole sequence is long enough to form a note, the new PCF receives the value of the last note in the sequence. Glissando filtering is illustrated in Figure 3.

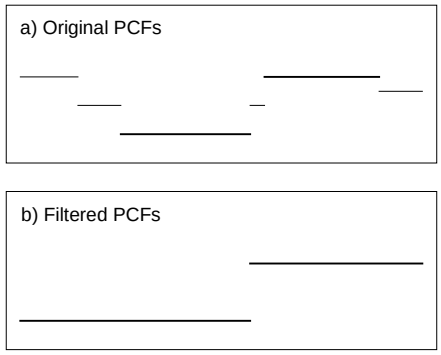

Figure 3: *Glissando* filtering.

After PCF filtering, the precise timings for each note candidate must be adjusted. In fact, as a consequence of MIDI quantization, there is a delay in the moment where transitions start, since the frequencies at the beginning of transitions may be converted to the same MIDI value, instead of the next MIDI value. In this way, we define the start of the transition as the point of maximum derivative of $f[k]$, after it starts to move towards the next note, i.e., the point of maximum derivative after the last occurrence of the median value. The median, md_i, is calculated only for non-empty frames (zero frequency) whose original MIDI note numbers are equal to the MIDI numbers after filtering, according to (4). In this way, the median, md_i, is obtained in a more robust way, since possibly noisy frames are not considered.

$$md_i = \text{median}\left(f[k] \right) \quad , \forall_{k \in D_i : f_{MIDI}[k] = c_i \text{ and } f[k] \neq 0} \qquad (4)$$

Furthermore, we compute the discrete derivative using the original frequency curve instead of the filtered one. As a conse-

quence of the empty frames present, the derivative in a given point k is calculated as the difference between $f[k]$ and $f[p]$, where p denotes the first non-empty frame immediately before frame k. Formally, it comes, (5):

$$\dot{f}[k] = \begin{cases} f[k] - f[p] & , f[k] \neq 0 \text{ and } f[p] \neq 0 \\ & \text{and } \forall_{i \in \{p+1, \cdots, k-1\}:\, f[i]=0} \\ 0 & , f[k] = 0 \end{cases} \quad (5)$$

Finally, we have to assign a MIDI note number to each of the obtained PCFs for each track. In order to increase the robustness of the assignment procedure, we deal with ambiguous situations, where it is not totally clear which is the correct MIDI value. This happens, for instance, when the median frequency is close to the frequency border of two MIDI notes.

In this way, we compute the initial MIDI value from the median frequency of each function, according to eq. (2). Then, we get the reference frequency associated to the obtained MIDI value, by inverting eq. (2). This is carried out in order to check if the median does not deviate too much from the reference frequency. Here, we define a maximum deviation, *maxDev*, of 30 cents. Therefore, if the median is in the allowed frequency range for the defined MIDI value (eq. (6)), we conclude that there is evidence that the assigned MIDI value is correct and, so, keep it.

$$\begin{aligned} iniMIDI_i &= \text{MIDI}(md_i) \\ refF_i &= \text{frequency}(iniMIDI_i) \\ range_i &= \left[refF_i \cdot 2^{-maxDev/1200}; refF_i \cdot 2^{maxDev/1200} \right] \end{aligned} \quad (6)$$

In (6), $iniMIDI_i$ represents the initial MIDI value of the i^{th} PCF, $refF_i$ stands for the corresponding reference frequency, $range_i$ denotes the allowed frequency range, 'MIDI' is the function corresponding to eq. (2) and 'frequency' is a function for obtaining the reference frequency from a MIDI value (inversion of eq. (2)).

However, when the median deviates significantly from the reference, it is not clear whether the initial MIDI value is correct or not. In order to clarify this ambiguity, we use a simple heuristic for the definition of the final MIDI value. Basically, if the median is higher than the upper range limit, the final MIDI value may need to be incremented.

In this way, we determine the frequency value in the frontier of the two candidate MIDI values, $borderF_i$, which corresponds to the frequency 50 cents above the reference frequency, (7):

$$borderF_i = refF_i \cdot 2^{50/1200} \quad (7)$$

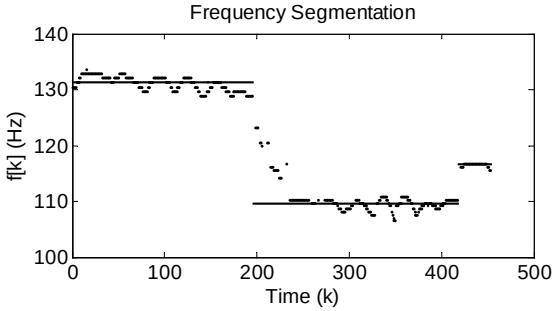

Figure 4: *Illustration of frequency segmentation.*

Then, we count i) the number of frames, *numH*, for which the frequency is above the frontier, i.e., the number of frequency values corresponding to the incremented MIDI value and ii) the number of frames, *numL*, where the frequency is below the median. If *numH* is higher than *numL*, there is evidence that the final MIDI value should be changed to the incremented value. Otherwise, it is left unchanged. We follow a similar path if the median is below the lower range limit, except that here the MIDI value may have to be decremented.

The procedure for frequency segmentation is illustrated in Figure 4, for a pitch track from The Mambo King's "Bella Maria de Mi Alma". There, the constant lines represent the obtained PCFs.

3.2. Pitch Salience Segmentation

As for pitch salience segmentation, the objective is to separate consecutive notes with the same value, which the MPTC algorithm may have interpreted as forming one single note. This requires segmentation based on pitch salience minima, which mark the limits of each note. In fact, the salience value depends on the evidence of pitch for that particular frequency, which is strongly correlated to the energy of FF, though not exactly equal. Therefore, saliences are lower at the onsets and offsets. Consequently, the envelope of the salience curve is similar to an amplitude envelope: it grows at the note onset, has then a more steady region and decreases at the offset. In this way, notes can be segmented by detecting clear minima in the salience curve.

As in the frequency segmentation stage, the algorithm starts by filtering the salience curve with a ZOH, due to missing values. Additionally, as the salience curve may be somewhat noisy, we add a low-pass filtering stage in order to smooth it. A zero-phase Blackman-sinc filter [15] is used, as follows, (8):

$$\begin{aligned} s_F[k] &= s[k] * h[k], \\ h[n] &= K \frac{\sin(2\pi f_c n)}{n\pi} \cdot b[n] \quad , n \in \{-W/2, \cdots, W/2\} \end{aligned} \quad (8)$$

In (8), $s[k]$ is the k^{th} pitch salience value in the current track, '*' denotes discrete convolution, $b[n]$ represents the Blackman window, centered at the origin, and $s_F[k]$ is the smoothed curve. The parameters f_c = 100Hz and W = 9 stand for the cutoff frequency and the length of the filter kernel, respectively. K is chosen so as to provide unity filter gain at DC.

Then, we iteratively look for all clear local minima and maxima of $s_F[k]$. First, all local minima and maxima are found, coping with plateaus. In these situations, the indexes of the corresponding minima/maxima are assigned to the middle of the plateau.

Then, only clear minima are selected. This is accomplished in a recursive procedure that starts by finding the global minimum of $s_F[k]$. Followingly, the set of all local maxima is divided into two subsets, one to the left and another to the right of the global minimum. The global maximum for each subset is then obtained.

After that, the global minimum is selected as a clear minima if its prominence, i.e., the minimum distance from its amplitude and that of both the left and right global maxima, is above the defined minimum peak-valley distance, *minPvd*.

Finally, the set of all local minima is also divided into two new intervals, to the left and right of the global minimum. The described procedure is then recursively repeated for each of the new subsets until all clear minima and respective prominences are found.

One difficulty of the proposed approach is its lack of robustness. In fact, the best value for *minPvd* was found to vary from

track to track, along different song excerpts. In fact, a unique value for that parameter leads to both missing and extra segmentation points. Also, it is sometimes difficult to distinguish between note endings and amplitude modulation in some performances. Therefore, we improved our method by performing onset detection and matching the obtained onsets with the candidate segmentation points that resulted from our prominent valley detection algorithm.

In this way, *minPvd* should receive a low value so that missing segmentation points are unlikely. In addition, this parameter ought to be adaptive in order to cope with notes having different salience ranges. Hence, *minPvd* was set to 10% of the maximum amplitude range of the salience curve under consideration (whose values belong to the [0; 100], after the normalization conducted in the MPD stage). As a result of its low value, extra false segmentation points occur, which are eliminated later on via onset matching.

Figure 3 illustrates our prominent valley detection algorithm for a pitch track of Claudio Roditi's "Rua Dona Margarida", where 'o' represents correct segmentation candidates and '*' denotes extra segmentation points.

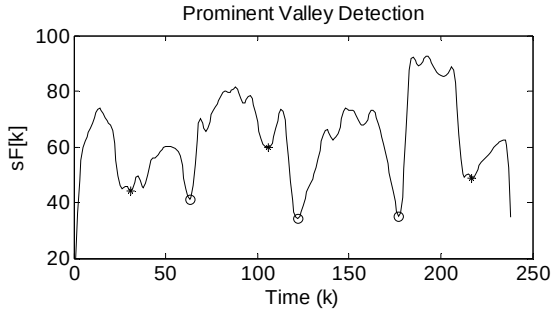

Figure 5: *Pitch salience segmentation: candidate points.*

As for onset detection, we based ourselves on [6] and [12], where onsets are detected following a band-wise processing approach. A bank of nearly critical band filters is chosen, which covers the frequencies from 44Hz to the Nyquist frequency, in a total of 18 filters. Elliptic filters are employed, so as to guarantee a maximally sharp cutoff in the transition band. Since it is important to maintain the temporal properties of the signal in each band, zero-phase should be a requirement. Thus, we perform bi-directional filtering [15], (9):

$$S_B^i(z) = S(z) * H^i(z) * H^i\left(z^{-1}\right) \tag{9}$$

Here, $S_B^i(z)$, $H^i(z)$ and $S(z)$, represent, respectively, the filtered output and the filter transfer function at band *i*, and the original signal, all in the *Z*-domain. Due to bi-directional filtering, we specified filter parameters in terms of the desired transfer function: 3rd order filters, with 1.5dB ripple in the pass-band and 20dB of rejection in the stop-band. The design parameters are approximately doubled as a result of bi-directional filtering, e.g., 6th order filters result. As for the cutoff frequencies, the lowest three filters are one-octave band-pass filters, whereas the remaining are third-octave band-pass filters, with no overlapping. Bi-directional filtering slightly changes the cutoff frequencies, which was not problematic in this case.

After filtering, onset components are computed. First, we extract the amplitude envelope of each output via rectify-and-smooth [12]. In order to ease calculations, the output of each band is decimated to 200 Hz. Then, the outputs are full-wave rectified and smoothed with a 100ms zero-phase half-Hanning window [12], *w*[*n*], of corresponding width *W*/2, as in (10). Again, we guarantee zero-phase by shifting the window.

$$\begin{aligned} s_H^i[k] &= \left|s_B^i[k]\right| * w[k - W/2] \\ w[i] &= 0.5 - 0.5\cos\left(\frac{2\pi i}{W-1}\right),\ i = 1,2,\cdots,W/2 \end{aligned} \tag{10}$$

After rectify-and-smooth, we extract information regarding energy variations by computing the derivatives of the amplitude envelopes $s_H^i[k]$, $\dot{s}_H^i[k]$. We half-wave rectify the derivative curve in each band, since we are only interested in the points of positive energy variations. Then, we linearly sum the computed derivatives and look for points of maximum in the resulting derivative curve. In other words, we look for clear maxima in the summed derivative. This is accomplished in a similar way to the above procedure for detection of clear minima, except that now we look for peaks instead of valleys. Here, we normalize the derivative curve to the [0; 1] interval, and select all peaks, whose saliences are above 0.05. Such peaks form the set of original onset candidates.

Finally, since some peak neighborhoods may be very dense, we delete components that are closer than 50 ms to a more intense component [6]. The remaining onset components are then normalized to the [0; 1] interval.

After onset detection, our goal is to validate the candidate segmentation points obtained above. First, all clear salience minima, i.e., whose valley prominence is at least 30 units, are kept as definitive segmentation points. Then, for all unclear valleys, onset matching is performed. Hence, if a candidate valley has a close and clear onset, i.e., an onset that differs from it by less than 20 ms and whose magnitude is above 0.4, that valley is kept as a segmentation point. The defined segmentation points are then adjusted to the locations of the detected onsets.

Finally, the obtained segmentation points are used to further segment the notes that resulted from the frequency-based segmentation stage. Also, the starting times of the original unsegmented notes are adjusted when close clear onsets are found before the original note beginnings (20 ms distance).

The described procedure for pitch salience segmentation is illustrated in Figure 6, for an excerpt from Claudio Roditi's "Rua Dona Margarida".

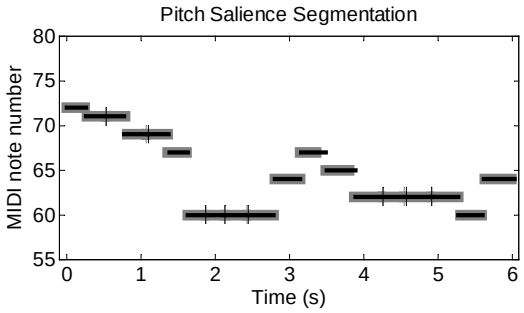

Figure 6: *Illustration of p itch salience segmentation.*

There, the gray horizontal lines represent the original annotated notes, whereas the black lines denote the extracted notes. The small gray vertical lines stand for the correct segmentation points and the black vertical ones are the obtained results of our algorithm. An almost perfect match is observed in this excerpt.

4. EVALUATION EXPERIMENTS

In order to evaluate the melody detection system, and particularly the segmentation algorithm, we collected excerpts of about 6 seconds from 12 songs, encompassing several different genres (pop, rock, jazz, classical, latin, …) [9]. The selected songs contain a solo (either vocal or instrumental) and accompaniment parts (guitar, bass, percussion, other vocals, etc.). Also, the selected excerpts contain glissando and frequency modulated notes, as well as consecutive notes with the same MIDI value.

The results for frequency segmentation were very good. All glissandos and frequency-modulated notes were correctly captured except for a short ornamental note found in one excerpt from Battlefield Band's "Snow on the Hills". As for the timings, they matched very well our manually annotated database. Most deviations were smaller than 20ms, which may even have resulted from annotation errors. Only a few higher deviations occurred in tracks with transitions zones with many missing frequency values.

As for pitch salience segmentation, the results were also generally good (e.g., Figure 5 and Figure 6). However, some tracks were more problematic, as a result of missing and extra onsets in some excerpts (e.g., Enya's "Only Time", with a lot of reverb). Onset detection in polyphonic recordings is itself a complex task, and so salience segmentation may be improved if onset detection algorithms become more reliable.

As a final word, our melody extraction system obtained 88.3% average pitch detection accuracy for the annotated melody notes. Some extra notes are still present, a problem that will be addressed in the future. Yet, the obtained results are encouraging.

5. CONCLUSIONS

We proposed a method for the definition of musical notes from pitch tracks. This is an important requirement for melody detection in polyphonic musical signals, which is usually ignored. However, the explicit definition of notes is an important issue for melody-based music information retrieval, as well as melody transcription. The obtained results were very good, especially for frequency segmentation. Pitch salience segmentation may be improved as soon as more robust onset detection algorithms for polyphonic signals become available.

6. ACKNOWLEDGEMENTS

This work was partially supported by the Portuguese Ministry of Science and Technology, under the program PRAXIS XXI.

7. REFERENCES

[1] J. P. Bello, *Towards the Automatic Analysis of Simple Polyphonic Music: A Knowledge-based Approach*, PhD Thesis. University of London, 2003.

[2] A. S. Bregman, *Auditory Scene Analysis: the Perceptual Organization of Sound*, MIT Press, 1990.

[3] W. Chai, *Melody Retrieval on the Web*, MSc Thesis, Massachusetts Institute of Technology, 2001.

[4] J. Eggink and G. J. Brown, "Extracting melody lines from complex audio", in *Proc. Intl. Conf. on Music Information Retrieval (ISMIR'2004)*, 2004.

[5] M. Goto, "A predominant-F0 estimation method for CD recordings: MAP estimation using EM algorithm for adaptive tone models", in *Proc. International Conference on Acoustics, Speech and Signal Processing (ICASSP' 2001)*, 2001.

[6] A. Klapuri, "Sound onset detection by applying psychoacoustic knowledge", in *Proc. ICASSP'1999*, 1999.

[7] A. Klapuri, *Signal Processing Methods for the Automatic Transcription of Music*, PhD Thesis. Tampere University of Technology, 2004.

[8] M. Marolt, "On finding melodic lines in audio recordings", in *Proc. Intl. Conf. on Digital Audio Effects (DAFX'2004)*, 2004.

[9] R. P. Paiva, T. Mendes, and A. Cardoso, "An auditory model based approach for melody detection in polyphonic musical recordings", in U. K. Wiil (ed.), *Computer Music Modelling and Retrieval – CMMR 2004*, Lecture Notes in Computer Science, vol. 3310, 2005.

[10] R. P. Paiva, T. Mendes, and A. Cardoso, "Exploiting melodic smoothness for melody detection in polyphonic audio", accepted for presentation in Intl. Computer Music Conf. (ICMC'2005), Barcelona, 2005.

[11] J.-C. Pérez and E. Vidal, "An algorithm for the optimum piecewise linear approximation of digitized curves", in *Proc. Intl. Conf. on Pattern Recognition (ICPR'1992)*, 1992, pp. 167–170.

[12] E. D. Scheirer, "Tempo and beat analysis of acoustic musical signals", *Journal of the Acoustical Society of America*, vol. 103, no. 1, pp. 588–601, Jan. 1998.

[13] X. Serra, "Musical Sound Modeling with Sinusoids Plus Noise". In Roads, C., Pope, S., Picialli, A., De Poli, G. (eds.), *Musical Signal Processing*, 1997.

[14] M. Slaney and R. F. Lyon, "On the Importance of Time – A temporal Representation of Sound", In Cooke, Beet and Crawford (eds.), *Visual Representations of Speech Signals*, 1993.

[15] S. W. Smith, *The Scientist and Engineer's Guide to Digital Signal Processing*, California Technical Publishing, 1997.