ON THE DETECTION OF MELODY NOTES IN POLYPHONIC AUDIO

Rui Pedro Paiva **Teresa Mendes** **Amílcar Cardoso**

CISUC – Centre for Informatics and Systems of the University of Coimbra
Department of Informatics Engineering, Pólo II – Pinhal de Marrocos
P 3030 – 290 Coimbra, Portugal
ruipedro@dei.uc.pt tmendes@dei.uc.pt amilcar@dei.uc.pt

ABSTRACT

This paper describes a method for melody detection in polyphonic musical signals. Our approach starts by obtaining a set of pitch candidates for each time frame, with recourse to an auditory model. Trajectories of the most salient pitches are then constructed. Next, note candidates are obtained by trajectory segmentation (in terms of frequency and pitch salience variations). Too short, low-salience and harmonically related notes are then eliminated. Finally, the notes comprising the melody are extracted. This is the main topic of this paper.

We select the melody notes by making use of note saliences and melodic smoothness. First, we select the notes with highest pitch salience at each moment. Then, by the melodic smoothness principle, we exploit the fact that tonal melodies are usually smooth. Thus, long music intervals indicate the presence of possibly erroneous notes, which are substituted by notes that smooth out the melodic contour.

Finally, false positives in the extracted melody should be eliminated. To this end, we remove spurious notes that correspond to abrupt drops in note saliences or durations. Additionally, note clustering is conducted to further discriminate between true melody notes and false positives.

Keywords: Melody detection, melodic smoothness, feature extraction, note clustering

1 INTRODUCTION

Query-by-humming (QBH) is a particularly intuitive way of searching for a musical piece, since melody humming is a natural habit of humans. This is an important research topic in an emergent and promising field called Music Information Retrieval (MIR). Several techniques have been proposed in order to attain that goal, e.g., [1]. However, this work is presently restricted to the MIDI domain, which places important usability questions. In fact, usually we look for recorded songs, which can be obtained from CDs or are stored in audio formats such as mp3. Additionally, in musical pieces in the MIDI format, the melody is usually available in a separate channel. The main issues are, then, to extract the notes from the hummed query (a well-known monophonic pitch extraction problem) and to match the query to the melody (an information retrieval problem).

On the other hand, querying "real-world" polyphonic recorded musical pieces requires the analysis of polyphonic musical waveforms. This is a rather complex task since many types of instruments can be playing at the same time, with severe spectral interference between each other. So far, only little work has been conducted to tackle the problem of melody detection in polyphonic audio, e.g., [2, 3, 4, 5]. Additionally, most of the work is only concerned with the extraction of melodic pitch lines, rather than melody notes.

In our approach, we put the focus on the melody, no matter what other sources are present. Thus, we base our strategy in two main assumptions that we designate as the "salience principle" and the "melodic smoothness principle". By the salience principle, we assume that the melody notes are, in general, salient in the mixture (i.e., in terms of their intensity). As for the melodic smoothness principle, we exploit the fact that note frequency intervals tend, generally, to be small. Finally, false notes present in the obtained melody are deleted by setting out the ones that correspond to abrupt salience or duration decreases and by performing note clustering to further separate true melody notes from false positives.

2 MELODY DETECTION SYSTEM

Our melody detection algorithm comprises five stages, as illustrated in Figure 1. The general strategy was described previously, e.g., [5] and, thus, only a brief presentation is provided here, for the sake of completeness. New improvements to the melody extraction stage are described in more detail.

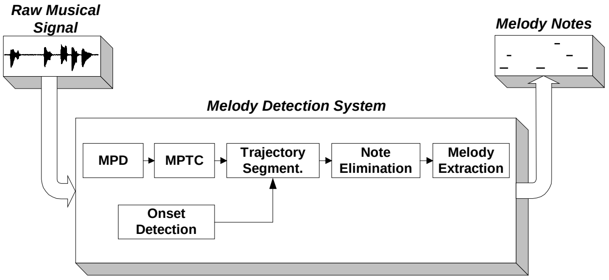

Figure 1. Melody detection system overview.

In the Multi-Pitch Detection (MPD) stage, the objective is to capture the most salient pitch candidates, which constitute the basis of possible future notes. We perform pitch detection in a frame-based analysis, with a 46.44 ms frame length and a hop size of 5.8 ms. For each obtained pitch, a pitch salience is computed, which is approximately equal to the energy of the corresponding fundamental frequency. Our approach is based on Slaney and Lyon's auditory model [6].

Multi-Pitch Trajectory Construction (MPTC), in the second stage, aims to create a set of pitch tracks, formed by connecting consecutive pitch candidates with similar frequency values. To this end, we based ourselves on the algorithm proposed by Serra [7]. 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 behaviours were kept within a single track. Thus, each trajectory may contain more than one note and should, therefore, be segmented.

The segmentation of tracks resulting from the MPTC stage is performed in two phases: frequency segmentation, aiming to separate notes with different MIDI values, and salience segmentation, with the objective of dividing consecutive notes at the same MIDI note number. Our trajectory segmentation algorithm is described with detail in [8].

In the fourth stage, irrelevant note candidates are eliminated, based on their saliences, durations and on the analysis of harmonic relations. We make use of perceptual rules of sound organization, namely "harmonicity" and "common fate" [9], where common frequency and amplitude modulation are exploited.

In the last stage, 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, many notes are still present. Therefore, we have to extract the ones that convey the main melodic line. This is the main topic of this paper and is described in the following section.

3 EXTRACTION OF MELODY NOTES

The definition of the notes comprising the melody of a song under analysis, being probably the most important task of any melody detection algorithm, is also the most difficult one to carry out. In fact, 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 order to limit the scope of our study, we focus this analysis on Western tonal music, where a clear solo is present, as in [9].

We base our strategy on the assumptions that i) the main melodic line often stands out in the mixture (salience principle) and that ii) melodies are usually smooth in terms of the note frequency intervals, which tend to be small (melodic smoothness principle). In addition, we attempt to eliminate false notes in the resulting melody by removal of spurious notes and note clustering.

3.1 Selection of the Most Salient Notes

In the first step of the melody extraction stage, we select the most salient notes at each time as initial melody candidates. The used criteria for comparing the salience between notes, as well as algorithmic details, were described previously, e.g., [5]. Namely, notes below MIDI number 50 (146.83 Hz) are excluded. This is motivated by the fact that the notes comprising the melody are, usually, in a middle frequency range. Moreover, bass notes usually contain a lot of energy and so, if no frequency limit was set, such notes would probably be selected as part of the melody. Anyway, this restriction will be relaxed later on, when melodic smoothness is applied.

In the implemented algorithm, some of the selected notes were truncated, since melody notes are not allowed to overlap in time.

The results of melody extraction by selecting the most salient notes are illustrated in Figure 2, for an excerpt from Pachelbel's Kanon. There, the correct notes are depicted in grey and the black continuous lines denote the obtained melody notes. The dashed lines stand for the notes that result from the note elimination stage. We can see that some erroneous notes are extracted, whereas true melody notes are excluded. Namely, some octave errors occur. As a matter of fact, one of the limitations of only taking into consideration note saliences is that the notes comprising the melody are not always the most salient ones. In this situation, wrong notes may be selected as belonging to the melody, whereas true notes are left out. This is particularly clear when abrupt transitions between notes are found, as can be seen in Figure 2. Hence, we improved our method by smoothing out the melody contour, as follows.

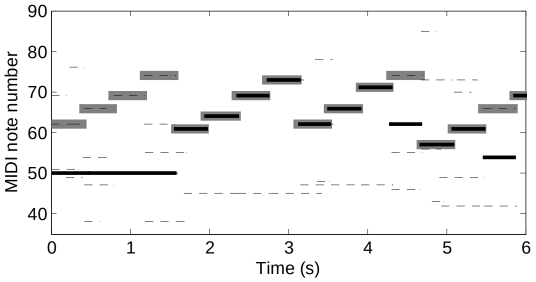

Figure 2. Extraction of the most salient notes (excerpt from "Pachelbel's Kanon").

3.2 Melody Smoothing

As referred above, taking into consideration only the most salient notes has the limitation that, frequently, non-melodic notes are more salient than melodic ones. As a consequence, erroneous notes are often picked up, whereas true notes are excluded. Particularly, abrupt transitions between notes give strong evidence that wrong notes were selected. In fact, small frequency transitions favour melody coherence, since smaller steps in pitch hang together better [9]. In an attempt to demonstrate that musicians generally prefer to use smaller note steps, the psychologist Otto Ortmann counted the number of sequential intervals in several songs by classical composers, having found that the smallest ones occur more frequently and that their respective number roughly decreases in inverse proportion to the size of the interval [9]. So being, we improved the melody extraction stage by taking advantage of this melodic smoothness principle. This is a culturally dependent principle, which is particularly relevant for Western tonal music.

We started to improve the initial melody by performing octave correction. In fact, in the note elimination stage not all harmonically related notes are eliminated and, thus, some octave errors occur when sub or superharmonic notes are more salient than the right notes. In order to correct octave errors, we select all notes for which no octaves (either above or below) are found and compute their average MIDI values. Then, we analyse all notes that have octaves with common onsets: if the octave is closer to the computed average, the original note is replaced by the corresponding octave. This simple first step already improves the final melody significantly. However, some octave errors, as well as abrupt transitions, are still kept, which will be worked out in the following stages.

In the second step, we analyse the obtained notes and look for regions of smoothness, i.e., regions where there are no abrupt transitions between consecutive notes. Here, we define a transition as being abrupt if the intervals between consecutive notes are above a fifth, i.e., seven semitones, as illustrated in Figure 3. There, the bold notes (a_1, a_2 and a_3) are marked as abrupt. In the same example, four initial regions of smoothness are detected (R_1, R_2, R_3 and R_4).

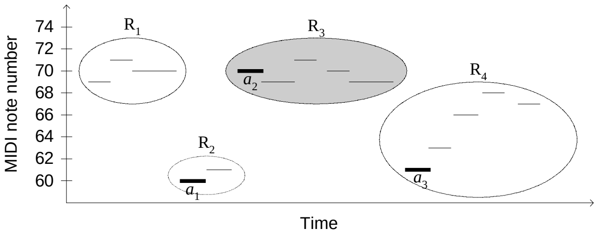

Figure 3. Regions of smoothness.

Then, we select the longest region as a correct region (region R_3, in Figure 3, filled in grey) and define the allowed note range for its adjacent regions (R_2 and R_4).

Regarding the left region, we define its allowed range based on the first note of the correct region, e.g., MIDI value 70 in this example. Keeping in mind the importance of the perfect fifth, the allowed range for the left region is 70 ± 7, i.e., [63, 77]. As region R_2 contains no note in the allowed range, this region is a candidate for elimination. However, before deletion, we first check if each of its notes contains an octave in the allowed range. If so, the corresponding notes are substituted by the found octaves. If at least one octave is found, no note is deleted in this iteration. On the contrary, if no octave is found, all the notes are eliminated.

As for the right region, we proceed likewise. Hence, we define the allowed range based on the last note of the correct region, e.g., 69 in this example, resulting the range [62, 76]. Since region R_4 contains notes in the allowed range, its first note, i.e., note a_3, is marked as non-abrupt. However, we still look for an octave of the referred note in the allowed range. In case it is found, the abrupt note is substituted, as before.

In short words, regions that correspond to sudden movements to different registers are interpreted as being incoherent and are, consequently, eliminated. However, abrupt transitions are allowed if adjacent regions are both coherent in melodic terms, as happens in Figure 3 for regions R_3 and R_4. This situation occurs in some musical pieces as, for example, Pachelbel's Kanon, as can be seen in Figure 2 and Figure 4.

If no notes are substituted/eliminated for the current region, the following regions are analysed in the same way, in descending length order. If no change at all is performed for all regions, the algorithm stops. Otherwise, whenever a change is performed, the procedure for definition of regions of smoothness, analysis of neighbours and deletion/substitution is repeated until no change is done. In the successive iterations, regions of smoothness are defined taking into consideration notes previously marked as non-abrupt, e.g., the notes in region R_4. Therefore, in a following iteration, regions R_3 and R_4 will not be divided.

Finally, since some regions are eliminated, their notes need to be substituted by other notes that are more likely to belong to the melody, according to the smoothness principle. Thus, we fill each gap in the melody with the most salient note candidates that are in the allowed range for that region. In this gap filling procedure, the previous restriction on the minimum allowed note no longer applies: the most salient note in the allowed range is selected, no matter its MIDI value. In fact, such restriction was imposed as a necessity to prevent the selection of too many erroneous notes (particularly bass notes), which would jeopardize melody smoothing. Therefore, we kept the general assumption that melodies are contained in middle frequency ranges, but allowing now the selection of low-frequency notes, as long as the smoothness requirement is fulfilled.

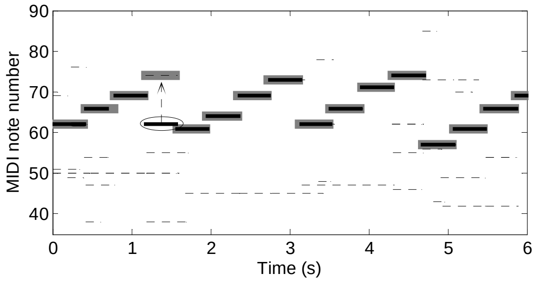

Figure 4. Extracted melody after melodic smoothness (excerpt from "Pachelbel's Kanon").

The results of the implemented procedures are illustrated in Figure 4, for the same excerpt from Pachelbel'

Kanon. We can see that only one erroneous note resulted (signalled by an ellipse), which corresponds to an octave error. This example is particularly challenging to our melody-smoothing algorithm due to the periodic abrupt transitions present. Yet, the performance was very good.

Since our proposed melody extraction approach outputs the most salient notes at each time in the allowed note range, false positives may arise. Such notes may be output both when pauses between melody notes are sufficiently long and when the solo is absent (e.g., singing has stopped and another instrument dominates for some time). Thus, spurious notes should be removed, as well as notes that are obtained when the solo is absent.

3.3 Elimination of Spurious Notes

As referred, when pauses between melody notes are fairly long, spurious notes, resulting either from noise or background instruments, may be included in the melody. We observed that, usually, such notes have lower saliences and shorter durations, leading to clear minima in the pitch salience and duration contours.

As for the pitch salience contour, we start by computing the average pitch salience of each note in the extracted melody and, then, look for deep valleys in the pitch salience sequence. Since saliences were normalized to the [0, 100] in the MPD stage, we defined a valley as being deep if it is at least 30 units below the respective left and right global maxima. Hence, notes in deep valleys of the pitch salience contour are disposed.

A jazz excerpt (jazz3 sample, see Table 1), where the solo is often absent, was chosen to illustrate the conducted procedure.

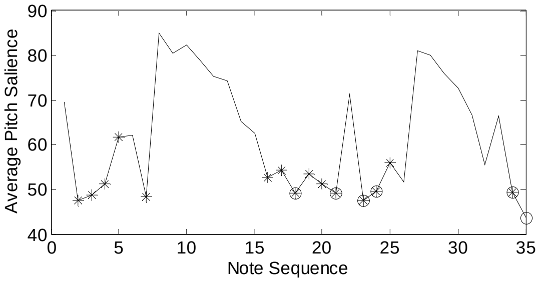

Figure 5. Illustration of pitch salience contour (jazz3 excerpt).

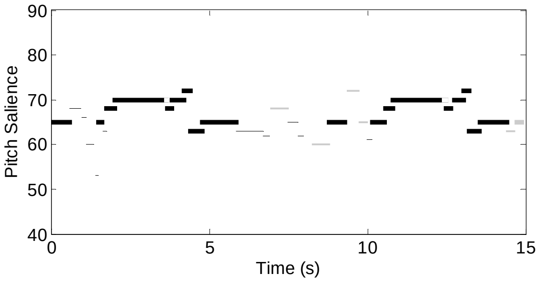

Figure 6. Illustration of elimination of spurious notes based on pitch salience (jazz3 excerpt).

Figure 5 depicts the pitch salience contour, where ‘*’ denote false positives and ‘o’ represent the deleted notes. It can be seen that one true note (the last one) was, nevertheless, removed. Besides, with a lower elimination threshold, a few more false notes would have been deleted. However, best overall results were obtained with the defined threshold.

The extracted melody notes are visualized in Figure 6. There, the thick lines denote true melody notes, whereas the thin ones stand for false positives. The grey lines represent deleted notes. It can be seen that, though some extra notes are disposed, some false positives remain present in this excerpt.

Regarding the duration contour, we proceeded likewise. However, we observed that duration variations are much more common than pitch salience variations. In this way, we decided to eliminate only isolated abrupt duration transitions, i.e., isolated notes delimited by much longer notes, where a note is too short if its duration is at least 20% its neighbours’. Additionally, in order not to inadvertently delete short ornamental notes, a minimum difference of two semi-tones was defined.

3.4 Note Clustering

As observed above, when the solo is absent, notes from the dominant accompaniment are output. It can be argued that this behaviour corresponds to the way humans memorize songs: a continuous “line” that comprises both melody per se and dominant accompaniments. However, since our goal is to extract the melody in a strict sense (not a predominant pitch line), the accompaniment should be eliminated. To this end, true notes and false positives are discriminated via note clustering.

This work is related to the classification of musical instruments in a polyphonic context. Only little work has been conducted in this field, e.g. [10], so far with limited accuracy. In fact, this is a complex task, since, in one hand, it is difficult to define acoustic invariants that are good timbre correlates and, on the other hand, the proposed features are difficult to measure in a polyphonic context due to spectral overlapping between sources.

The conducted procedures, namely feature extraction and selection, dimensionality reduction and clustering, are described as follows.

3.4.1 Feature Extraction

In order to acquire information on the source of each note, we use a set of features that aim to capture sound pitch, intensity and timbre content in both the attack and steady-state parts of each note. Namely, the following features were used, based on related work, e.g., [11, 12]:

- 1) *Spectral centroid*, which correlates well with the perceived sound brightness;
- 2) *Relative spectral centroid*, calculated as the ratio of the centroid to the fundamental frequency
- 3) *Pitch salience*, which is closely related to the intensity of the sound;
- 4) *Pitch stability*, measured as the frequency variation over successive time frames, related to aspects such as pitch jitter or modulation;

- 5) *Harmonic magnitude*, which gives a measure of spectral shape;
- 6) *Relative harmonic magnitude ratio*, the same as before, except that now relative values are used;
- 7) *Spectral irregularity*, calculated as the average difference between the magnitude of a harmonic and its two neighbours;
- 8) *Spectral inharmonicity*, computed as the sum of differences of each harmonic frequency from its theoretical value;
- 9) *Spectral skewness*, which is the magnitude of the harmonics weighed by their respective inharmonicities;
- 10) *Harmonic frequency*, whose absolute values give also information on inharmonicity;
- 11) *Relative harmonic frequency ratio*, the same as before, except that relative values are used here;
- 12) *Harmonic onset time*, calculated as the absolute time delay of each harmonic compared to the note onset; a measure of onset asynchrony;
- 13) *Relative onset time*, the same as before, except that relative timings are used here;
- 14) *Attack duration*, which correlates to the type of coupling between the excitation and resonant structures; short attacks indicate tight coupling;
- 15) *Frequency slope in the attack*, which measures the amount of glissando before pitch stability;
- 16) *Note duration*;

The listed features were extracted on top of the auditory front-end used in the MPD stage. In this way, the harmonic frequencies and magnitudes of each pitch candidate in each time frame are obtained directly from a correlogram frame, by using the respective correlogram columns [5]. Then, for each column, local peaks are detected and matched to the expected frequencies of each harmonic. If no peak is found in the allowed range of the frequency partial, the filter-bank channel whose centre frequency is closest to the theoretical value is selected. Hence, in this case the harmonic frequencies and magnitudes are the ones of the filter channel. This is carried out much in the same way as Martin did [11].

Furthermore, for partials above the sixth, several of them may be mapped to the same cochlear channel [11]. In this case, some upper harmonics get the same magnitude values. Coincidentally or not, we tested our approach with different numbers of harmonics and best results were obtained with exactly six. Therefore, only six frequency partials were used. Spectral features are then computed based on the obtained harmonic frequencies and magnitudes in the steady-part of the signal.

Finally, instead of storing sequences of feature values, we computed statistical summaries for each feature. Namely, mean and standard deviation were used, except for those features that have a sole value for each note (e.g., frequency slope, duration). In addition, each feature vector was normalized to the [0, 1] interval, so as to avoid numerical problems resulting from the different feature ranges.

The computation of some of the features was problematic, as a consequence of the polyphonic context we are working in. Namely, the frequency slope was difficult to measure for notes with many missing frequency values in the beginning. Therefore, the slope was simply calculated by interpolating the first and last frequency values in the attack. Also, some harmonic magnitudes may be corrupted due to spectral collisions. Therefore, those elements should be discarded and clustering should be conducted following a missing feature strategy [10]. We will address this issue in future developments.

3.4.2 Feature Selection and Dimensionality Reduction

The number of implemented features is very high compared to the number of notes available in each song excerpt. In addition, a high number of features may lead to the so-called curse of dimensionality [13]. Hence, feature selection and dimensionality reduction were performed prior to clustering.

As referred, it is important to select the best combination of features to include. Since it is impractical to analyse every different combination of features, forward selection was conducted [13]. In this way, starting from an empty feature set, the algorithm adds, step by step, the feature that leads to the best model accuracy. The combination of features that gives the best overall performance is then selected.

In addition, the dimension of the feature space was reduced with recourse to Principal Component Analysis (PCA) [13]. This is a widely used technique whose basic idea is to project the computed feature matrix into an orthogonal basis that best expresses the original data set. Moreover, the resulting projected data is decorrelated. As for the selection of the principal components, we kept the ones that retained 90% of the variance.

3.4.3 Clustering

Finally, after feature extraction, selection and dimensionality reduction, true notes and false positives are discriminated via clustering. To this end, we used Gaussian Mixture Models (GMMs) [13].

GMMs are extensively used for unsupervised clustering of data. Basically, Gaussian distributions are fitted to the observed data and so GMMs model the probability density of observed features by multivariate Gaussian mixture densities.

In order to separate false positives from true melody notes, we defined only two clusters (a “melody cluster” and a “garbage cluster”), initialised with the *K*-means clustering algorithm [13].

Next, the parameters of the model (mean vector, covariance matrix - diagonal, in this case - and mixing coefficients) are iteratively estimated with recourse to the Expectation-Maximization algorithm [13]. The algorithm stops when the likelihood function stabilizes for consecutive iterations.

After that, each note is allotted to a cluster based on the posterior probabilities in each: the cluster with the highest probability is selected.

Finally, the melody is assigned to the cluster with maximum salience, where cluster salience is computed as the sum of the average pitch salience of each note multiplied by its duration.

The procedure for note clustering is illustrated in Figure 7, for the same jazz excerpt used before. As can be seen, all false positives were eliminated. However, three true melody notes were erroneously deleted. In fact, there seems to be a trade-off between keeping all the true melody notes and removing all false positives.

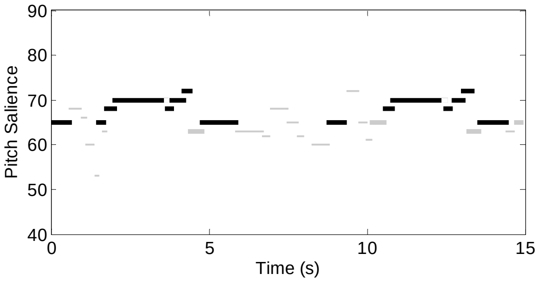

Figure 7. Illustration of note clustering (jazz3 excerpt).

3.4.4 Clustering the Whole Note Set

We also decided to test a different approach, where clustering was performed on the whole note set that resulted from the note elimination stage. Some constraints should be imposed on the performed clustering (e.g., no overlap between notes) [4]. However, we ignored this issue since the procedures for detection of salient notes and melody smoothing guarantee the consistency of the results. Furthermore, harmonically related notes may come from the same source and, thus, such constraints are problematic in this situation.

Therefore, notes were clustered with the GMM algorithm, using now five clusters. Then, for each cluster, salient notes were detected, melody smoothing was performed and spurious notes were eliminated.

Finally, the melody was assigned to the cluster with the highest salience, as before.

4 EXPERIMENTAL RESULTS

One difficulty regarding the evaluation of MIR systems results from the absence of standard test collections and benchmark problems. This problem was partly solved through the creation of a set of evaluation databases by researchers from the Music Technology Group of University Pompeu Fabra (MTG - UPF), Barcelona, Spain, for the ISMIR 2004 Audio Description Contest (ADC) [14]. Several competitions were organized as part of it. Naturally, we are more interested in the database created for the Melody Extraction Contest (MEC-04).

In this way, we evaluated the proposed algorithms with both the MEC-04 database and a test-bed we had previously created (see Table 1, where the top 11 lines correspond to our test-bed and the next 10 refer to the MEC-04 database). Both databases were defined taking into consideration its diversity and musical content. Therefore, the selected song excerpts contain a solo (vocal or instrumental), and accompaniment parts (guitar, bass, percussion, other vocals, etc.). In addition, in some excerpts the solo is absent for some time. In our test-bed, we collected excerpts of about 6 seconds from 11 songs, which were manually annotated. As for the MEC-04 database, 10 excerpts, each of around 20 seconds, were automatically annotated based on monophonic pitch estimation from multi-track recordings [14].

For accuracy computation, the detected melody notes were compared with the correct notes. To this end, we used two of the metrics defined in [14], with some adaptations. In the first metric, the pitched accuracy (*PA*), i.e., the accuracy regarding only the notes comprising the melody, was performed. In the second one, a global accuracy (*GA*) was computed taking into consideration also the matching of frames where the melody is absent. Additionally, another frame-based metric was considered, where octave errors were ignored [14]. For this one, only summary results are presented.

In terms of frame comparison, we defined the target frequency values for each time frame as the reference frequencies of the corresponding MIDI notes. In the same way, the extracted frequencies were defined from the reference frequencies corresponding to the extracted melody notes. The accuracy was calculated as the percentage of correctly identified frames. In the original metric defined in [14], exact frequency values were used. However, since we do not know the precise frequency values for the excerpts in our test-bed, reference MIDI frequencies were used for the sake of uniformity. Also, the determination of exact frequency values does not seem very relevant in a melody detection context.

Five evaluations were performed: i) extraction of salient notes only (*SN*); ii) note salience plus melodic smoothness (*MS*); iii) elimination of spurious notes (*ES*); iv) note clustering *(NC*); and v) note clustering in the whole note set (*NCW*). For each evaluation, results for the two used metrics were computed.

The obtained results are summarized in Table 2. Short descriptions of the used song excerpts are presented in Table 1.

Regarding the *MS* evaluation, we can see that good results were achieved. There, an average accuracy of 84.0 / 75.4% (*PA* / *GA*, respectively) was attained. Without melody smoothing, the average accuracy was 74.7 / 66.2% (*SN* evaluation) and so our implementation of the melodic smoothness principle amounts for an average improvement of 9.3 / 9.2%. A high number of octave errors was corrected, especially in the excerpts from Battlefield Band and Pachelbel’s Kanon.

The results from the choral sample were also interesting, since four simultaneous voices are present, plus orchestral accompaniment. Still, the algorithm could reasonably detect the melody, which we defined as corresponding to the soprano. The use of this example contradicts our previous assumptions, but we were interested in the results for a special situation like this one.

As for the MEC-04 database, the results were also good, except for the opera excerpts. These samples seem

ID	*Song Title*	*Genre*	*Solo Type*
1	Pachelbel’s Kanon	Classical	Instrumental
2	Handel’s Hallelujah	Choral	Vocal
3	Enya - Only Time	Neo-Classical	Vocal
4	Dido - Thank You	Pop	Vocal
5	Ricky Martin - Private Emotion	Pop	Vocal
6	Avril Lavigne - Complicated	Pop / Rock	Vocal
7	Claudio Roditi - Rua Dona Margarida	Jazz / Easy	Instrumental
8	Mambo Kings - Bella Maria de Mi Alma	Bolero	Instrumental
9	Compay Segundo - Chan Chan	Son Cubano	Vocal
10	Juan Luis Guerra - Palomita Blanca	Bachata	Vocal
11	Battlefield Band - Snow on the Hills	Scottish Folk	Instrumental
12	daisy2	Synthesized singing voice	Vocal
13	daisy3	Synthesized singing voice	Vocal
14	jazz2	Saxophone phrases	Instrumental
15	jazz3	Saxophone phrases	Instrumental
16	midi1	MIDI synthesized	Instrumental
17	midi2	MIDI synthesized	Instrumental
18	opera_fem2	Opera singing	Vocal
19	opera_male3	Opera singing	Vocal
20	pop1	Pop singing	Vocal
21	pop4	Pop singing	Vocal

Table 1. Description of used song excerpts.

ID	*Salient Notes*		*Melody Smoothing*		*Elim. Spurious*		*Note Clustering*		*Note Clust. Whole*	
	PA	*GA*	*PA*	*GA*	*PA*	*GA*	*PA*	*GA*	*PA*	*GA*
1	59.3	58.3	89.5	88.1	89.5	88.1	89.5	88.1	89.5	88.1
2	62.6	54.9	78.7	67.9	78.7	67.9	81.5	76.8	81.5	72.3
3	94.0	90.9	94.0	90.9	94.0	90.9	94.0	90.9	94.0	90.9
4	92.0	74.2	94.9	73.5	94.9	73.5	94.9	73.5	94.9	73.5
5	64.4	44.2	72.0	53.7	72.0	53.7	75.9	58.6	72.0	53.7
6	75.6	68.8	93.7	84.2	93.7	88.6	93.7	88.6	93.7	88.6
7	89.0	83.0	98.3	91.7	98.3	91.7	98.3	91.7	98.3	91.7
8	87.7	81.0	90.8	83.8	90.8	83.8	90.8	83.8	90.8	83.8
9	82.4	63.3	82.4	65.0	82.4	65.0	82.4	69.6	82.4	65.0
10	73.5	51.8	80.2	57.2	80.2	57.2	80.2	57.2	80.2	57.2
11	47.1	46.9	93.6	92.3	93.6	92.3	93.6	92.3	93.6	92.3
12	91.6	79.5	92.3	82.0	92.3	84.9	87.1	80.4	92.3	84.9
13	84.4	84.3	97.2	97.1	97.2	97.1	97.2	97.1	97.2	97.1
14	69.6	65.0	73.6	70.4	76.1	73.7	73.4	71.2	76.1	73.7
15	82.4	59.8	86.6	63.8	85.5	74.3	78.8	84.6	85.5	74.3
16	64.1	62.2	85.9	83.8	86.1	85.4	86.1	85.4	88.2	88.4
17	97.9	96.3	97.9	96.3	97.9	96.3	97.9	96.3	97.9	96.3
18	64.8	57.8	69.8	62.0	69.8	62.0	69.8	62.0	69.8	62.0
19	38.5	37.1	41.3	38.7	41.3	39.4	41.3	39.4	42.5	40.6
20	69.9	62.6	70.2	65.3	70.7	69.6	69.3	68.4	70.7	69.6
21	78.6	69.0	82.0	75.1	82.2	76.7	82.2	76.7	82.2	76.7
Avg	74.7	66.2	84.0	75.4	**84.1**	**76.8**	83.7	77.7	84.4	77.2

Table 2. Results of the melody detection system.

to pose additional difficulties to the pitch detection algorithm, in the first stage of our system. We plan to address this issue in the near future.

Another interesting fact is that the proposed approach is almost immune to octave errors. Indeed, disregarding octave errors, the accuracy for SM raised to 85.0 / 76.2%, i.e., an improvement of only 1.0 / 0.8%.

Regarding the elimination of spurious notes (*ES* evaluation), we can see that *GA* improved slightly (1.4%). As a consequence of note elimination, the original durations of some notes were restored (recall that some of them were truncated in the note salience stage), which led to a slight improvement of *PA* (0.1%).

As for note clustering, the global accuracy improved a bit more (0.9%, comparing to the *ES* evaluation and 2.3%, regarding *SM*), but some excerpts still have many false positives. In fact, different songs prefer different features combinations. For example, almost all false positives from Juan Luis Guerra's sample were eliminated with a particular feature set. However, best overall results were achieved using the following features, in order of insertion from the forward selection algorithm: 5, 6, 2, 8, 1, 3, 7, 10, 11, 9, 15 and 14. From the obtained results we can see that, while many false positives were deleted, a few true notes were also wrongly eliminated, leading to a 0.4% drop in the pitched accuracy.

Clustering the whole note set (*NCW* evaluation) led to similar results: 84.4% for *PA* and 77.2% for *GA*. Again, different excerpts prefer different features combinations, but best global results were attained with only these: 5, 6, 10, 11 and 1.

The main limitation of the note clustering stage is its lack of robustness. In fact, the best set of features varies from sample to sample and some particular feature combinations simply cannot discriminate between true notes and false positives, leading to a notorious fall in melody detection accuracy. Therefore, so far, robustness cannot be guaranteed after the elimination of spurious notes. However, longer song excerpts could possibly improve the accuracy for note clustering.

Finally, for the sake of comparison with the results from the ISMIR 2004 ADC, we also tested our approach with the exact frequency values used there. As a consequence, the accuracy for the *ES* evaluation, taking into consideration only the MEC-04 database, dropped from 79.9 / 75.5% to 75.0 / 69.9%, i.e., approximately 5%. In our opinion, when the goal is predominant pitch estimation, exact frequency values are important. However, accuracy computation in terms of MIDI reference values seems more relevant for melody detection tasks, where exact frequency values are not needed in the output.

5 CONCLUSIONS

We propose a system for melody detection in polyphonic musical signals. This is a main issue for MIR applications, such as QBH in "real-world" music databases. The work conducted in this field is presently restricted to the MIDI domain, and so we guess we make an interesting contribution to the area, with some encouraging results. Additionally, we tackled the problem of false positives. As expected, this proved to be a very difficult task, and so only slight improvements were achieved.

Regarding future work, we plan to further work out some of the described limitations, namely in what concerns the reliability of feature computation and the improvement of the elimination of false positives.

ACKNOWLEDGEMENTS

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

REFERENCES

[1] Bainbridge, D., Nevill-Manning, C., Witten, I., Smith, L. and McNab, R. "Towards a digital library of popular music", Proceedings of the ACM International Conference on Digital Libraries, 1999.

[2] Eggink, J., and Brown, G. J. "Extracting melody lines from complex audio", Proceedings of ISMIR, 2004.

[3] Goto, M. "A predominant-F0 estimation method for CD recordings: MAP estimation using EM algorithm for adaptive tone models", Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2001.

[4] Marolt, M. "On finding melodic lines in audio recordings", Proceedings of the International Conference on Digital Audio Effects, 2004.

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

[6] Slaney, M., and Lyon, R. F. "On the importance of time - a temporal representation of sound". In Cooke, Beet and Crawford (eds.), Visual representations of speech Signals, 1993.

[7] Serra, X. "Musical sound modeling with sinusoids plus noise". In Roads, C., Pope, S., Picialli, A., De Poli, G. (eds.), Musical signal processing, 1998.

[8] Paiva, R. P., Mendes, T., and Cardoso, A. "On the definition of musical notes from pitch tracks for melody detection in polyphonic recordings", accepted for presentation at the International Conference on Digital Audio Effects, 2005.

[9] Bregman, A. S. Auditory scene analysis: the perceptual organization of sound. MIT Press, 1990.

[10] Eggink, J., and Brown, G. J. "Application of missing feature theory to the recognition of musical instruments in polyphonic audio", Proceedings of the International Conference on Music Information Retrieval (ISMIR), 2003.

[11] Martin, K. D. Sound-source recognition: a theory and computational model. PhD Thesis, Massachusetts Institute of Technology, 1999.

[12] Tzanetakis, G. Manipulation, Analysis and Retrieval Systems for Audio Signals. PhD Thesis, Princeton University, 2002.

[13] Bishop, C. M. Neural networks for pattern recognition. Oxford University Press, 1995.

[14] MTG - UPF. "ISMIR 2004 Audio Description Contest". ISMIR, 2004. http://ismir2004.ismir.net/ISMIR_Contest.html