**Rui Pedro Paiva, Teresa Mendes, and Amílcar Cardoso**
Center for Informatics and Systems of the University of Coimbra (CISUC)
Department of Informatics Engineering
University of Coimbra, Pólo II
3030 Coimbra, Portugal
{ruipedro, tmendes, amilcar}@dei.uc.pt

# Melody Detection in Polyphonic Musical Signals: Exploiting Perceptual Rules, Note Salience, and Melodic Smoothness

Melody extraction from polyphonic audio is a research area of increasing interest. It has a wide range of applications in various fields, including music information retrieval (MIR, particularly in query-by-humming, where the user hums a tune to search a database of musical audio), automatic melody transcription, performance and expressiveness analysis, extraction of melodic descriptors for music content metadata, and plagiarism detection, to name but a few. This area has become increasingly relevant in recent years, as digital music archives are continuously expanding. The current state of affairs presents new challenges to music librarians and service providers regarding the organization of large-scale music databases and the development of meaningful methods of interaction and retrieval.

In this article, we address the problem of melody detection in polyphonic audio following a multistage approach, inspired by principles from perceptual theory and musical practice. Our system comprises three main modules: pitch detection, determination of musical notes (with precise temporal boundaries, pitches, and intensity levels), and identification of melodic notes. The main contribution of this article is in the last module, in which a number of rule-based systems are proposed that attempt to extract the notes that convey the main melodic line among the whole set of detected notes. The system performs satisfactorily in a small database collected by us and in the database created for the ISMIR 2004 melody extraction contest. However, the performance of the algorithm decreased in the MIREX 2005 database.

Computer Music Journal, 30:4, pp. 80–98, Winter 2006
© 2006 Massachusetts Institute of Technology.

## Related Work

Previous work on the extraction of 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 the fundamental frequencies present (Martin 1996; Bello 2003; Klapuri 2004). 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.

Little work has been conducted on melody detection in polyphonic audio. However, this is becoming a very active area in music information retrieval, confirmed by the amount of work devoted to the ISMIR 2004 and MIREX 2005 evaluations. Several different approaches have been proposed in recent years (Goto 2001; Brossier, Bello, and Plumbey 2004; Eggink and Brown 2004; Marolt 2004, 2005; Paiva, Mendes, and Cardoso 2004, 2005b; Dressler 2005; Poliner and Ellis 2005; Ryynänen and Klapuri 2005; Vincent and Plumbey 2005; Gómez et al. 2006). (A few of these systems were originally published as non-peer-reviewed online proceedings of MIREX 2005 and, to our knowledge, were not published elsewhere.)

Generally, most current systems, including ours, are based on a front-end for frequency analysis (e.g., Fourier Transform, autocorrelation, auditory models, multi-rate filterbanks, or Bayesian frameworks), peak-picking and tracking (in the magnitude spectrum, in a summary autocorrelation function, or in a pitch probability density function), and post-processing for melody identification (primarily rule-based approaches based on perceptual rules of sound organization, musicological rules, path-finding in networks of notes, etc.). One exception is Poliner and Ellis (2005), where the authors follow a different strategy by approaching the melody-detection problem as a classification task using Support Vector Machines.

## Melody Definition

Before describing our system, it is important to clarify what we mean by the term *melody.* An important aspect regarding the perception of the main melodic stream in an ensemble is the phenomenon of figure-ground organization in audio. This is related to the "tendency to perceive part of . . . the auditory scene as 'tightly' organized objects or events (the figure) standing out against a diffuse, poorly organized background (the ground)" (Handel 1989, p. 551). In this respect, Leonard Meyer wrote

> the musical field can be perceived as containing: (1) a single figure without any ground at all, as, for instance, in a piece for solo flute; (2) several figures without any ground, as in a polyphonic composition in which the several parts are clearly segregated and are equally, or almost equally, well shaped; (3) one or sometimes more than one figure accompanied by a ground, as in a typical homophonic texture of the eighteenth or nineteenth centuries; (4) a ground alone, as in the introduction to a musical work—a song, for instance—where the melody or figure is obviously still to come; or (5) a superimposition of small motives which are similar but not exactly alike and which have little real independence of motion, as in so-called heterophonic textures" (Meyer 1956, quoted in Tsur 2000).

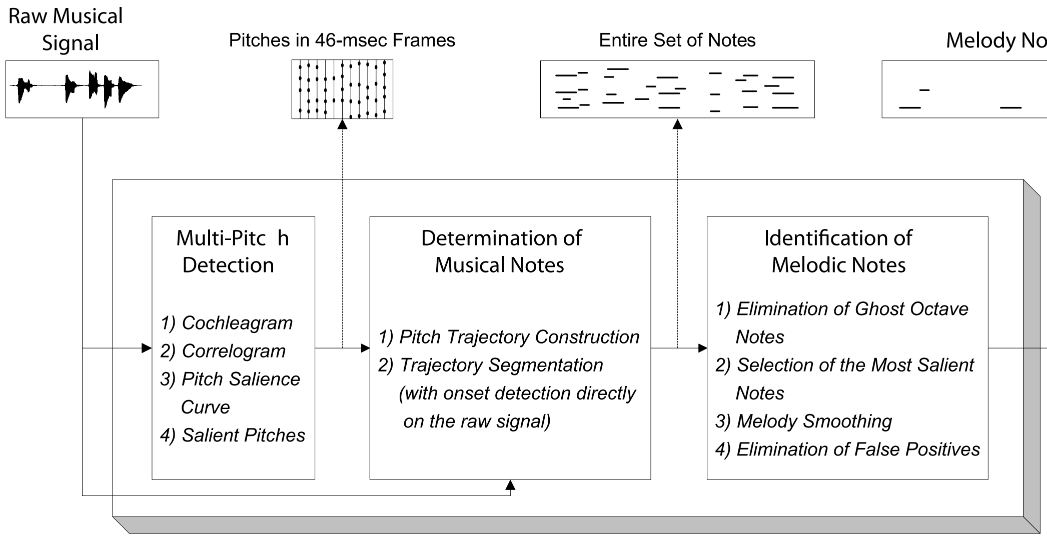
Naturally, our work does not aim to contemplate all the aspects around the concept of melody. Hence, we focus this article on the analysis of songs in which a single figure dominates and is accompanied by pitched and/or percussive background instruments. In this way, we define melody as "the dominant individual pitched line in a musical ensemble." In this definition we exclude unpitched percussion instruments from the main melody (the "pitched" requirement); emphasize the analysis of polyphonic music, i.e., music in which several instruments are playing concurrently (the "ensemble"); require the presence of a "dominant" lead voice, where by dominant we mean part that usually stands out in a mixture, e.g., lead vocals in pop music, lead saxophone in jazz, or a vocal soloist in opera; and exclude from the main melody the accompaniment parts that dominate when the lead voice is absent (the "individual line"). Also, we define "line" as a sequence of musical notes, each characterized by specific temporal boundaries as well as a corresponding MIDI note number and intensity level. Moreover, we assume that the instrument carrying the melody is not changed through the piece under analysis.

## System Overview

Our melody-detection algorithm is illustrated in Figure 1. Different parts of the system were described in previous publications (Paiva, Mendes, and Cardoso 2004, 2005a, 2005b); thus, only a brief presentation is provided here for the sake of completeness. Improvements to the last module (identification of melodic notes) are described in more detail.

In the multi-pitch detection stage, the objective is to capture the most salient pitch candidates in each time frame that constitute the pool of possible future notes. Our pitch detector is based on Slaney and Lyon's auditory model (Slaney and Lyon 1993), using 46.44-msec frames with a hop size of 5.8 msec. For each frame, a cochleagram and a correlogram are computed, after which a pitch-salience curve is obtained by adding across all autocorrelation channels. The pitch salience in each frame is approximately equal to the energy of the corresponding fundamental frequency. We follow a strategy that seems sufficient for a melody-detection task: instead of looking for all the pitches present in each frame, as happens in general polyphonic pitch detectors, we only capture those that most likely carry the main melody. These are assumed to be the most salient pitches, corresponding to the highest

Figure 1. Overview of the melody-detection system.



Raw Musical Signal
Pitches in 46-msec Frames
Entire Set of Notes
Melody Notes
Multi-Pitc h Detection
1) Cochleagram
2) Correlogram
3) Pitch Salience Curve
4) Salient Pitches
Determination of Musical Notes
1) Pitch Trajectory Construction
2) Trajectory Segmentation (with onset detection directly on the raw signal)
Identification of Melodic Notes
1) Elimination of Ghost Octave Notes
2) Selection of the Most Salient Notes
3) Melody Smoothing
4) Elimination of False Positives

peaks in the pitch-salience curve. A maximum of five pitch candidates is extracted in each frame. This value provided the best trade-off between pitch-detection accuracy and trajectory-construction accuracy in the following stage. Details on the pitch-detection algorithm can be found in Paiva, Mendes, and Cardoso (2004).

Unlike most other melody-extraction systems, we attempt to explicitly distinguish individual musical notes (in terms of their pitches, timings, and intensity levels), maintaining as well the exact frequency values that might be necessary for the analysis of performance dynamics or timbre. This is the goal of the second stage of the algorithm (Determination of Musical Notes, in Figure 1). To this end, we first create pitch tracks by connecting pitch candidates with similar frequency values in consecutive frames (the pitch trajectory construction, or PTC, step). We based our approach on the algorithm proposed by Xavier Serra (1997). The general idea is to find regions of stable pitches that indicate the presence of musical notes. The number of pitches in each frame is small, and so they are clearly spaced most of the time. Hence, the number of resulting trajectories is significantly lower compared to approaches based only on sinusoidal tracks (e.g., Serra 1997). Therefore, our approach minimizes ambiguities in trajectory construction.

To avoid losing information on the dynamic properties of musical notes, we took special care to keep phenomena such as vibrato and glissando within a single track. Thus, each trajectory may contain more than one note and should, therefore, be segmented in time. This is performed in two phases, namely frequency-based segmentation and salience-based segmentation. In frequency-based segmentation, the goal is to separate all notes of different pitches that might be present in the same trajectory. This is accomplished by approximating the pitch sequence in each track by a set of piecewise constant functions, handling glissando, legato, vibrato, and frequency modulation in general. Each detected function will then correspond to a MIDI note. Despite this quantization effect, the original pitch sequences are still kept so that the information on note dynamics is not lost.

The algorithm for frequency segmentation is based on a minimum note duration of 125 msec. This threshold was set based on the typical note durations in Western music. As Albert Bregman points out, “Western music tends to have notes that are rarely shorter than 150 msec in duration” (1990, p. 462). We experimented with a range between 60 and 150 msec, but the defined threshold of 125 msec led to the best results. It is noteworthy that this value is close to the one mentioned by Mr. Bregman.

With segmentation based on pitch salience variations, the objective is to separate consecutive notes at the same frequency that the PTC algorithm may have mistakenly interpreted as forming only one note. This requires trajectory segmentation based on pitch-salience minima, which mark the temporal boundaries of each note. To increase the robustness of the algorithm, note onsets are detected directly from the audio signal and used to validate the candidate salience minima found in each pitch track. The procedures conducted at the note-determination stage are described in Paiva, Mendes, and Cardoso (2004, 2005b).

In the last stage, our goal is to identify the final set of notes representing the melody of the song under analysis. This is the main topic of this article and is described in the following sections.

## Identification of Melodic Notes

After the first two stages of our system (see Figure 1), several notes from each of the different instruments present in the piece under analysis are obtained, among which the main melody must be identified. The separation of the melodic notes in a musical ensemble is not a trivial task. In fact, many aspects of auditory organization influence the perception of the main melody by humans, for instance in terms of the pitch, timbre, and intensity content of the instrumental lines in the sonic mixture.

Robert Francès investigated the figure–ground relationship in music (1958, cited in Uitdenbogerd 2002, p. 15). The main conclusion of his studies was that a musical part is heard as the figure if it is higher in pitch than the accompanying parts. However, this rule fails in some instances. For example, if the upper notes are more or less constant and the lower ones form a more interesting pattern, the lower notes will more easily catch a listener's attention and will thus be heard as the figure. Besides pitch height, other factors affect the perception of a part as the figure, namely frequency proximity and intensity (Uitdenbogerd 2002, p. 15).

In our algorithm, we have made particular use of intensity- and frequency-proximity aspects, as will be described. Namely, we based our strategy on the following assumptions: regarding intensity, the main melodic line often stands out in the mixture (the salience principle); and melodies are usually smooth in terms of note-frequency intervals (the melodic smoothness principle). Also, we attempt to eliminate non-melodic notes, i.e., false positives, in the resulting melody. Prior to the identification of the main melody, "ghost octave" notes are eliminated. The strategy for melody identification comprises four steps, as illustrated in the rectangle marked Identification of Melody Notes in Figure 1.

### Elimination of Ghost Octave Notes

The set of candidate notes resulting from trajectory segmentation typically contains several ghost octave notes. The partials in each such note are actually multiples of the true note's harmonics (if the ghost octave note is higher than the true note) or submultiples (if it is lower). Therefore, the objective of this step is to dispose of such notes.

In short, we look for harmonic relations between all notes, based on the fact that some of the obtained pitch candidates are actually harmonics or sub-harmonics of true fundamental frequencies in the sound wave. Therefore, we make use of the perceptual rules of sound organization designated as harmonicity and common fate (Bregman 1990). Namely, we look for pairs of octave-related notes with common onsets or endings and with common modulation, i.e., whose frequency and salience sequences change in parallel. We then delete the least-salient note if the ratio of its salience to the salience of the other note is below a defined threshold. This procedure is summarized in Algorithm 1 (Figure 2).

In this algorithm, while looking for octave-related notes with common onsets and endings (step 2.1a), some relaxation is introduced. In quantitative terms, two notes are said to have common onsets if their beginnings differ at most by *maxOnsetDist,* i.e., 62.5 msec. The same maximum difference applies when comparing the two notes' endpoints. This somewhat high value was experimentally set to handle timing inaccuracies that may result from noise and frequency drifting at the beginnings and

*Figure 2. Algorithm 1: Elimination of ghost octave notes.*

```
1. Sort all notes in ascending onset time order.
2. For each note i
    2.1. Look for a note, j, such that:
        a)(|onset(i) – onset(j)| ≤ maxOnsetDist or |ending(i) –
              ending(j)| ≤ maxOnsetDist) and
        b) |MIDI(i) – MIDI(j)| = 12k or 12k ± 1 and
        c) the two notes have parallel changes in frequency and
              salience.
    2.2 If note j was found,
        2.2.1. Compute the average salience of the two notes in
              their common time interval, avgSal.
        2.2.2. If avgSal(j) / avgSal(i) ≤ 0.4/k then
              delete note j and repeat step 2.1 until no more
              notes are found.
        2.2.3. If avgSal(i) / avgSal(j) ≤ 0.4/k then
              delete note i and repeat step 2 for the next note.
```

endings of notes. In actual notes, the onset asynchronies between partials do not exceed 30–40 msec, because at that point each partial may be heard as a separate tone (Handel 1989, p. 214).

In step 2.1b, we test if the two notes are octave-related (i.e., their MIDI note numbers, MIDI($i$) and MIDI($j$), differ by multiples of twelve semitones), tolerating deviations of one semitone to handle possible semitone errors. Here, $k$ is an integer that represents the number of octaves that separate them.

In step 2.1c, we exploit the fact that frequency sequences belonging to the same note tend to have synchronized and parallel changes in frequency and intensity (here represented by pitch salience). Thus, we measure the distance between frequency curves for pairs of octave-related note candidates. Similarly, we measure the distance between their salience curves.

Formally, the distance between frequency curves is calculated according to Equation 1, based on Virtanen and Klapuri (2000):

$$d_f(i,j) = \frac{1}{t_2 - t_1 + 1}\sum_{t=t_1}^{t_2}\left(\frac{f_i(t)}{avg\left(f_i(t)\right)} - \frac{f_j(t)}{avg\left(f_j(t)\right)}\right)^2 \qquad (1)$$

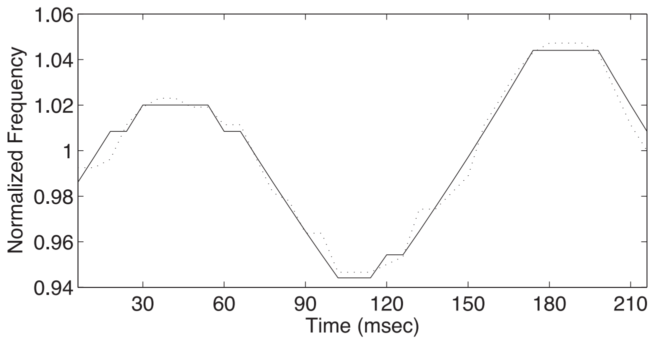
where $d_f$, represents the distance between two frequency trajectories, $f_i(t)$ and $f_j(t)$, during the time interval $[t_1, t_2]$, where both trajectories exist. Equation 1 attempts to scale the amplitude of each curve by its average, thereby normalizing it. An identical process is followed for the salience curves.

This procedure is illustrated in Figure 3 for the frequency sequences of two octave-related note candidates from an opera excerpt with extreme vibrato. (See Table 1, which appears later in this article, in the section entitled "Experimental Results.") We can see that the normalized frequency curves are very similar, which provides good evidence that the two sequences are both part of the same note.

Additionally, we found it advantageous to measure the distance between the normalized derivatives of frequency curves (and, likewise, the

Figure 3. *Example of similarity analysis of frequency curves.*

Figure 4. *Example of similarity analysis of salience trends.*



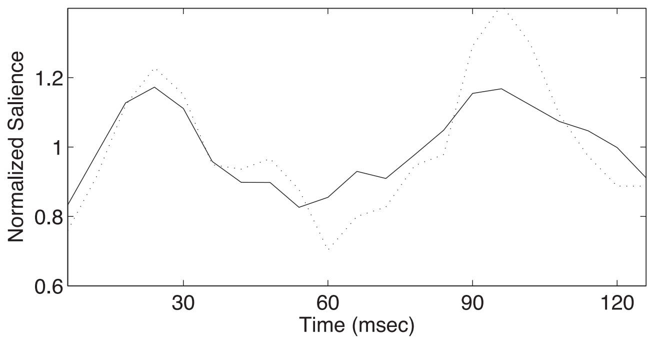


**Table 1. Description of Used Song Excerpts**

| *ID* | *Song Title* | *Category* | *Solo Type* |
|---|---|---|---|
| 1 | Pachelbel's *Canon* | Classical | Instrumental |
| 2 | Handel's *Hallelujah* | Choral | Vocal |
| 3 | Enya - *Only Time* | New Age | 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 | 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 |

Rows 1–11 are from our test bed and rows 12–21 are from the MEC-04 training set.

derivatives of salience curves). In fact, it is common that these curves have high absolute distances despite exhibiting the same trends. The distance between derivatives is used as another measure of curve similarity. This is illustrated in Figure 4 for the pitch-salience curves of two notes from the same opera excerpt. Note that, although the depicted saliences differ somewhat, their trends are very similar, i.e., the distance between the normalized derivates is small. Thus, it is also likely that they both belong to the same note.

To conclude the common modulation analysis, we assume that the two candidate notes have parallel changes if any of the four computed distances (i.e., in frequency, salience, or their derivatives) are below a threshold of 0.04. Finally, in step 2.2 of Algorithm 1 (see Figure 2), we eliminate one of the notes if its salience is less than 40% of the most salient note if they differ by one octave, 20% if they differ by two octaves, and so forth.

The values for curve-distance and salience-ratio thresholds were experimentally set so that the elimination of true melodic notes was minimal, while still deleting a significant amount of ghost octave notes. This is motivated by the fact that missing notes cannot be recovered in later stages but, instead, false candidates can be eliminated afterward.

In the testing database, an average of 38.1% of notes that resulted from the note-determination stage were eliminated. Moreover, only 0.3% of true melodic notes were deleted. Although many ghost notes were discarded at this point, a high number of non-melodic notes were still present. Namely, only 25.0% of all notes belonged to the melody. This posed interesting challenges to the next steps of the algorithm.

## Selection of the Most Salient Notes

As previously mentioned, intensity is an important cue in melody identification. Therefore, we select the most salient notes as an initial attempt at melody identification. The procedures conducted at this stage are described in Algorithm 2 (Figure 5).

In Algorithm 2, notes below MIDI note number 50 (146.83 Hz) are excluded (step 2). This constraint was motivated by the fact that bass notes usually contain a great deal of energy, and so if no frequency limit were set, these notes would probably be selected. To prevent the selection of too many erroneous notes (which would put at risk melody smoothing in the next stage), we first dispose of such notes. Thus, our algorithm is biased toward selecting middle- and high-frequency notes, which indeed corresponds to most real situations. Low-frequency notes may still be selected in the next stage, where this restriction will be relaxed provided melodic smoothness is guaranteed. Alternatively, we could have filtered bass notes in the front-end of the system; however, this would lead to the irrecoverable loss of true low-frequency notes.

In step 3, non-dominant notes are discarded. To this end, we segment the song excerpt under analysis, as illustrated in Figure 6, where, $s_k$ denotes for the *k*th segment. Then, we delete all notes that are non-dominant, i.e., that are not in the most salient segment more than 35% of their total number of frames or are not in the three most salient segments for more than 80% of their total number of frames.

It often happens that the remaining notes overlap in time, which should not be allowed. We handle such situations in step 4, where we first analyze the possible types of time overlaps between pairs of notes. We have identified six overlapping types, illustrated in Figure 7.

The first considered overlapping type, corresponding to a situation where the two notes have approximately the same onsets and endings. In this case, the maximum allowed distance between the onsets and endings is *maxOnsetDist,* as previously defined. In the second and third overlapping types, the two notes have common onsets but different endings. In the fourth possibility, notes under comparison have equal endings but different onsets. Finally, in the fifth and sixth overlapping types, the two notes have neither common onsets nor common endings. The fifth type denotes inclusions, where the second note starts after and ends before the reference note. The sixth type corresponds to the situations where the notes intersect, i.e., the second note starts and ends after the beginning and the ending of the reference note, respectively, considering again the maximum allowed difference.

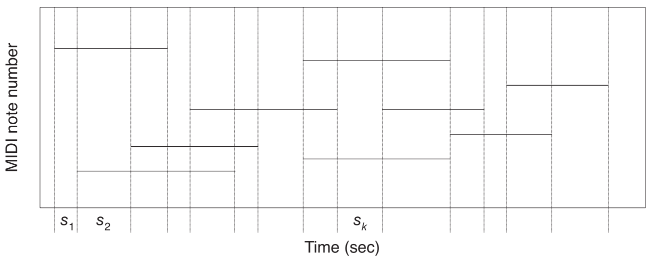
Each candidate note is then compared with the notes with temporal overlapping, and the overlap type is determined (step 4.2.2). In short, preference is given to the note with the highest average salience in their common time interval (step 4.2.3), leading to the elimination or truncation of the other note (steps 4.2.4 and 4.2.5). For example, in the second overlap type in Figure 7, if the reference note has the highest average salience in the common time interval, the second note is deleted. In the opposite case, the second note is kept unchanged, whereas the reference note is truncated at its beginning (i.e., it will start later, immediately after the second note ends). The same reasoning applies to all situations except for inclusions (the fifth situation). Here, if the second note is the strongest one, only the beginning or the ending of the reference note is

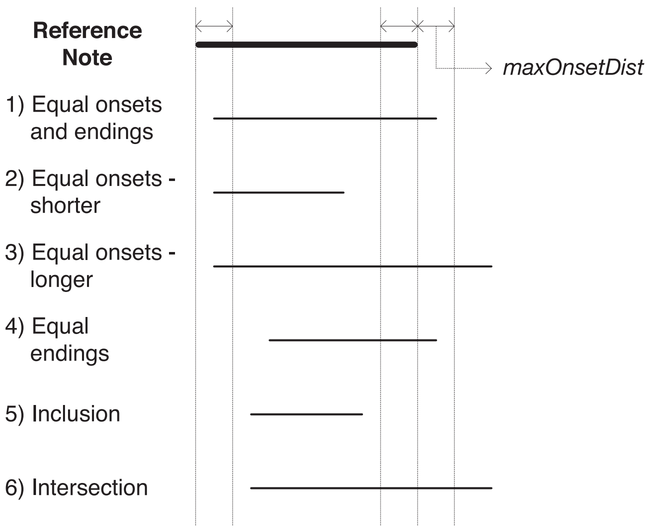
*Figure 5. Algorithm 2: Selection of the most salient notes.*

```
1. Sort all notes in ascending onset time.
2. Discard notes i such that MIDI(i) < 50.
3. Eliminate non-dominant notes
      3.1. Define song segments according to Figure 6.
      3.2. For each segment s,
            2.2.1. Calculate the average salience of each note in s.
      3.3. For each note i,
            3.3.1. Find the segments where note i is the most
                  salient one and calculate the corresponding
                  percentage of frames, percTop (comparing to the
                  note's total number of frames).
            3.3.2. Compute percTop3 in a similar way, by considering
                  the segments where note i is in the 3 most salient
                  ones.
            3.3.3. If percTop < 35% and percTop3 < 80%, delete
                  note i.
4. Resolve note overlaps.
      4.1. Save the original note beginnings and endings.
      4.2. For each resulting note i (reference note),
            4.2.1. Look for a note j that overlaps note i, i.e.,
                  onset(j) ≤ ending(i)
            4.2.2. Determine the overlap type (as in Figure 7).
            4.2.3. Calculate the average salience of each note in
                  the common time interval avgSal.
            4.2.4. If avgSal(j) ≤ avgSal(i) eliminate or truncate
                  note j (based on the overlap type) and repeat step
                  4.2.1 until no more notes are found (see text).
            4.2.5. Otherwise, delete or truncate note i and repeat
                  step 4.2 for the next note.
```

Figure 6. Definition of segments in a song excerpt. The horizontal lines denote the notes present after eliminating ghost notes. The vertical lines determine segment boundaries, which correspond to note beginnings and endings.

Figure 7. Types of note overlapping. The reference note (thick line) is, by definition, the one with the earliest onset time. The other horizontal lines represent time spans of hypothetical notes that overlap in time with the reference note.





kept, depending on the salience of each. Additionally, the original note timings are saved for future restoration (step 4.1), in case any of the selected notes are removed in the following stages.

The results of the implemented procedures are illustrated in Figure 8 for an excerpt from Pachelbel's famous *Canon in D.* We can see that some erroneous notes are extracted, whereas true melody notes are excluded. Namely, some octave errors occur. In fact, one of the limitations of only taking into consideration note salience 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 the previous figure. Hence, we improved our method by smoothing out the melody contour, as discussed in the next subsection.

## Melody Smoothing

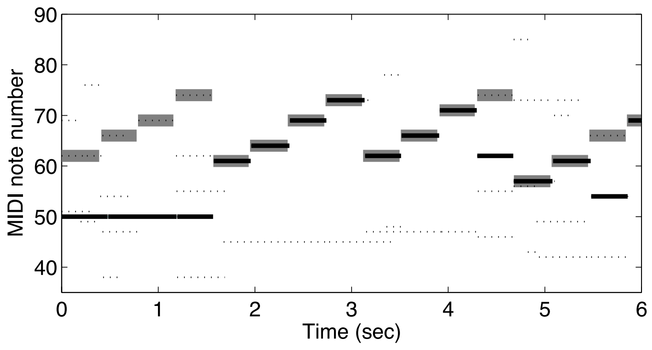
In an attempt to demonstrate that musicians generally prefer to use smaller note steps, the psychologist Otto Ortmann counted the number of sequential intervals in several works by classical composers; he found that the smallest ones occur more frequently and that their respective number roughly decreases in inverse proportion to the size of the interval (Bregman 1990, p. 462). In fact, small-frequency transitions favor melody coherence, because smaller steps in pitch "hang together" better (Bregman 1990, p. 462). Thus, we improved the melody-extraction stage by taking advantage of the melodic-smoothness principle. Although this might be a culturally dependent principle, it is relevant at least in Western tonal music.

The process of melody smoothing is presented in Algorithm 3 (shown in Figure 9). The basic idea is to detect abrupt note intervals and replace notes corresponding to sudden movements to different registers by notes that smooth out the extracted melody. In this stage, we start to improve the tentative melody that results from the selection of the most salient notes by performing octave correction (step 1 of Algorithm 3).

In fact, octave errors might occur because Algorithm 1 typically does not eliminate all ghost octave notes. This simple first step already improves the final melody significantly. However, a few octave errors, as well as abrupt transitions, are still present; these are worked out in the following steps.

In the second step, we smooth out the melodic contour by deleting or substituting notes corresponding to sudden movements to different registers. To this end, we first define regions of smoothness (step 2.1), i.e., regions with no abrupt note intervals. Here, intervals above a fifth, i.e., seven semitones, are defined as abrupt, as illustrated in Figure 10. This maximum-note interval was set based on the importance of the perfect fifth in Western music. Other intervals were evaluated as well, but, in the used excerpts, the best results were obtained with the fifth.

*Figure 8. Results of the algorithm for extraction of salient notes. The actual melody's notes are gray; the notes selected by the algorithm are black; dashed lines represent notes that are kept after the elimination of ghost notes.*



In the example in Figure 10, four initial smooth regions are detected: $R_1$, $R_2$, $R_3$, and $R_4$. We then select the longest region as a correct region (region $R_3$ in Figure 10, filled in gray) and define the allowed note range for its adjacent regions ($R_2$ and $R_4$). Region $R_2$ is a candidate for elimination, because it contains no notes in the allowed range (MIDI note number of note $a_2 \pm 7$, i.e., [63, 77]). However, before deletion, we first look for octave multiples of each of its notes in the allowed range. In case at least one octave is found, the original notes are replaced by the corresponding octaves, and no note is deleted in this iteration. Otherwise, all the notes in the region are eliminated.

As for region $R_4$ (i.e., the region immediately after the longest one), we perform a similar analysis. Hence, we define the allowed range based on the last note of the correct region, e.g., 69 in this example, resulting in the range [62, 76]. Because region $R_4$ contains a few notes in the allowed range, its first note, i.e., note $a_3$, is marked as non-abrupt, and regions $R_3$ and $R_4$ are joined together. (Still, if an octave of note $a_3$ is found in the allowed range, it is used instead of the original note.) In this way, abrupt transitions are allowed in case adjacent regions have notes in similar ranges. This situation occurs in some musical pieces as, for example, Pachelbel's *Canon* (Figures 8 and 11).

As a result of region elimination, the respective notes must be replaced by other notes that are more likely to belong to the melody according to the smoothness principle. Thus, in step 3, we fill in each gap with the most-salient note candidates that begin in that time interval and are in the allowed range. Again, we do not permit note overlaps. In this gap-filling procedure, the previous restriction on the minimum-allowed pitch (in the selection of the most-salient notes) no longer applies: the most salient note in the allowed range is selected, regardless of its MIDI note number. In fact, that 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 permitting now the selection of low-frequency notes, as long as the smoothness requirement is fulfilled.

The results of the implemented procedures are illustrated in Figure 11 for the same excerpt from Pachelbel's *Canon* presented before. We can see that only one erroneous note resulted (signaled by an ellipse), corresponding to an octave error. This example is particularly challenging to our melody-smoothing algorithm owing to the periodic abrupt transitions present. However, the performance was quite good.

## Elimination of False Positives

When pauses between melody notes are fairly long, non-melodic notes, resulting either from noise or background instruments, may be included in the extracted melody. (In particular, the gap-filling step described in the previous stage is often a cause of this phenomenon.) We observed that such notes typically have lower saliences and shorter durations, leading to clear minima in the pitch-salience and duration contours. In this way, we attempt to eliminate false positives by detecting and discarding the notes corresponding to the aforementioned minima, as described in Algorithm 4 (see Figure 12).

Regarding the pitch-salience contour, we start by deleting notes corresponding to clear minima in the salience contour (step 1). There, a given local minima is defined as clear if its prominence (i.e., the difference between its amplitude and that of both the global maxima before and after it) is at least 30. This value is based on the fact pitch saliences were normalized into the [0, 100] interval in the pitch-detection stage. Moreover, it was experimentally set

*Figure 9. Algorithm 3: Melody smoothing.*

```
1. Correct octave errors.
        1.1. Select all notes with no octaves (either above or below).
        1.2. Compute the average of their pitches, $avgMIDI$.
        1.3. For each note $i$,
                1.3.1. Look for a note $j$ such that:
                        a) $(|\text{onset}(i) - \text{onset}(j)| \leq maxOnsetDist$
                        b) $\text{MIDI}(i) - \text{MIDI}(j)| = 12k$ and
                1.3.2 Replace the current note $i$ by the found note $j$
                        (the octave) if it is closer to $avgMIDI$, i.e.,
                        if $|\text{MIDI}(j) - avgMIDI| < |\text{MIDI}(i) - avgMIDI|$
2. Smooth out the melodic contour by deleting or substituting notes
        corresponding to sudden movements to different registers.
        2.1. Define regions of smoothness, which are delimited by
                abrupt notes, i.e., notes $i$ such that
                $|\text{MIDI}(i) - \text{MIDI}(i-1)| > 7$ (see Figure 10).
        2.2. Select the longest region $r$ as a correct region.
        2.3. Analyze the regions immediately before and after region $r$
                for possible note deletion or octave substitution (see
                text).
        2.4. If any deletions or substitutions are performed in step
                2.3, repeat from step 2.1; otherwise, repeat from step
                2.2 for the next-longest region, until all regions are
                analyzed.
3. Repeat step 2 until no deletions or substitutions are performed
        (takomg into consideration notes marked as non-abrupt).
4. Fill in gaps.
        4.1. Look for empty intervals $g$ such that $\text{duration}(g) >$
                minimum note duration (125 msec).
        4.2. For each interval $g$,
                4.2.1. Look for a set of notes $i$ such that:
                        a) $\text{onset}(i)$ occurs during $g$ and
                        b) $\text{MIDI}(i)$ is in $\{\text{MIDI(last note before gap)} \pm 7\}$
                                or $\{\text{MIDI(first note after gap)} \pm 7\}$
                4.2.2. Select the most-salient notes in gap $g$, as in
                        Algorithm 2 (except that step 2 is not run).
```

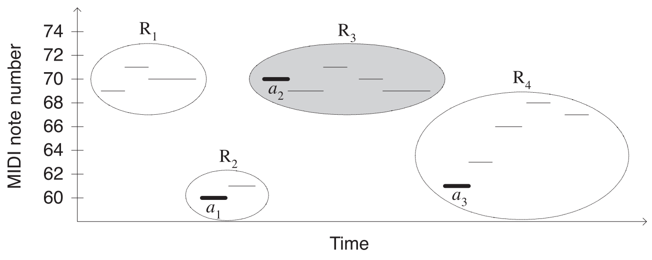


*Figure 10. Regions of smoothness. $R_3$ is the longest region (the length of each region is the sum of the lengths of all its notes); bold notes correspond to abrupt transitions.*

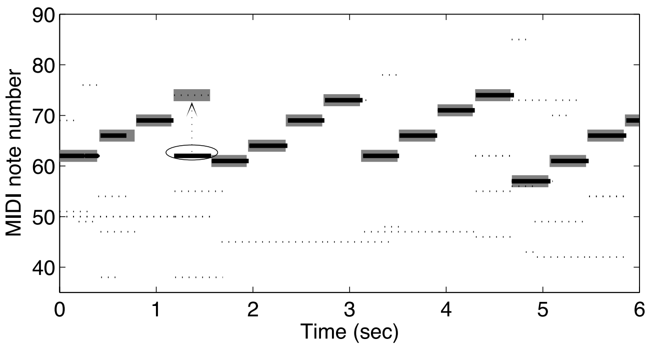


*Figure 11. Results of the melody-smoothing algorithm.*

so that the trade-off between the elimination of false positives and true melodic notes was satisfactory.

A jazz excerpt (the "jazz3" sample in Table 1; see the Experimental Results section), where the solo is often absent, was chosen to illustrate the conducted procedure. The salience contour of the employed sample is depicted in Figure 13. It can be seen that two true notes were, nevertheless, removed. Besides, with a lower elimination threshold, a few more false positives would have been deleted, but best overall results were obtained with the defined threshold.

As for the duration contour (step 2), we proceeded likewise. However, we observed that duration variations are much more common than pitch-salience variations. This was expected, because tone lengths tend to vary significantly. In this way, we decided to eliminate only isolated abrupt duration transitions, i.e., individual notes whose adjacent notes are significantly longer. Here, we define a note as being too short if its duration is at most 20% of its neighbors. Additionally, a minimum difference of three semitones was defined to prevent inadvertently deleting short ornamental notes, such as commonly used whole-step grace notes. Finally, in step 3, the time intervals of previously truncated notes (Algorithm 2, step 4.1, in Figure 5) that are adjacent to the deleted ones are restored. The melody that results after elimination of false positives is illustrated in Figure 14 for the same jazz excerpt in Figure 13.

It can be seen that, even though a few extra notes are disposed of (including two true melodic notes), some false positives remain. In this way, we have conducted a pilot study aiming to further discriminate between melodic and accompaniment notes. This was based on feature extraction and note clustering using Gaussian Mixture Models. The average results for melody segregation improved slightly, but so far the method is not sufficiently robust, because the best set of features varies significantly from sample to sample. In fact, if in some excerpts melody segregation improved notoriously, in other samples the accuracy dropped. In any case, the approach seems promising, but more general conclusions required evaluation on a larger corpus. Details on the algorithm can be found in Paiva, Mendes, and Cardoso (2005b).

## Experimental Results

One difficulty regarding the evaluation of MIR systems comes from the absence of standard test collections and benchmark problems. This was partly solved through the creation of a set of evaluation databases for the ISMIR 2004 Melody Extraction Contest (MEC-04) and for MIREX 2005 (see Table 1).

### Ground Truth Data

We evaluated the proposed algorithms with both the MEC-04 database and a small database we had previously created (see Table 1). Each of these databases were designed taking into consideration diversity and musical content. Therefore, the selected song excerpts contain a solo (either vocal or instrumental, corresponding to the main melody, according to our previous definition) and accompaniment parts (guitar, bass, percussion, other vocals, etc.). Additionally, in some excerpts, the solo is absent for some time. In our test bed, we collected excerpts of

*Figure 12. Algorithm 4: Elimination of false positives.*

```
1. Delete notes with lower salience.
      1.1. Define the pitch salience contour.
            1.1. Calculate the average pitch salience of each
note,
                  $i$, avgSal($i$).
            1.2. Form de salience contour as the sequence of
                  average note saliences.
      1.2. Delete notes corresponding to deep valleys in the
            salience contour.
            1.2.1. Detect all local minima in the contour.
            1.2.2. For each local minimum, $i$,
                  a) Determine the global maximum after it, $ma$,
and
                        the global maximum before it, $mb$.
                  b) If $|\text{avgSal}(i) - \text{ma}| \geq 30$ and
                        $|\text{avgSal}(i) - \text{mb}| \geq 30$, delete note $i$.
2. Delete notes with lower duration.
      2.1. Define the duration contour (similarly to step 1.1)
      2.2. Delete isolated notes corresponding to deep valleys in
            the duration contour.
            2.2.1. Detect all local minima in the contour.
            2.2.2. For each local minimum, $i$,
                  a) If $\text{duration}(i) < 0.2 \cdot \text{duration}(i+1)$ and
                        $\text{duration}(i) < 0.2 \cdot \text{duration}(i-1)$ and
                        $|\text{MIDI}(i) - \text{MIDI}(i+1)| > 2$ and
                        $|\text{MIDI}(i) - \text{MIDI}(i-1)| > 2$, delete note $i$.
3. Restore the time intervals of previously truncated notes that
are
      adjacent to the deleted notes.
>
```

*Figure 13. Pitch salience contour (jazz3 excerpt), where "*" denotes false positives, and "o" represents deleted notes. Pitch saliences are normalized to the interval [0, 100].*

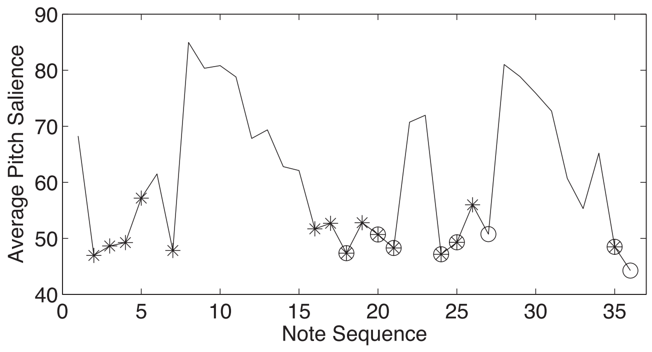
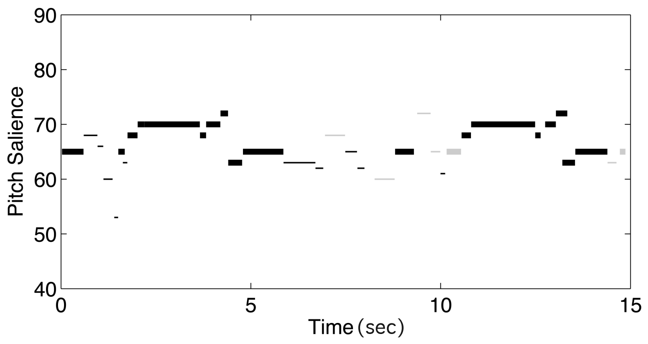


*Figure 14. Results of the algorithm for elimination of spurious notes. Thick black lines denote true positives, thin black lines represent false positives, thick gray lines denote deleted true notes, and thin gray lines represent deleted non-melodic notes.*



about 6 sec from 11 songs that were manually annotated with the correct notes. As for the MEC-04 database, 20 excerpts, each around 20 sec, were automatically annotated based on monophonic pitch estimation from multi-track recordings, as described in Gómez et al. (2006). From these, we employed the defined training set, consisting of 10 excerpts.

Contradicting our previous assumptions, we have selected a choral piece (ID 2), consisting of four simultaneous voices plus orchestral accompaniment. The idea was to observe the behavior of the algorithm in this situation, where we defined the solo as corresponding to the soprano. Additionally, we evaluated the algorithm on the Music Information Retrieval Evaluation eXchange (MIREX 2005) database (25 excerpts of 10–40 sec). The employed audio files were not made public, and so only average results are provided here.

## Evaluation Metrics

Depending on the application, the melody might be extracted as a sequence of MIDI notes or as a continuous pitch contour. For example, in tasks such as audio-to-MIDI conversion or query-by-humming, notes should be explicitly determined. In other tasks, e.g., analysis of performance dynamics (vibrato, glissando, etc.), pitch contours are preferred.

In our system, we are particularly interested in obtaining musical notes, although pitch-track contours are also available. Moreover, in our test bed, we do not know the exact target frequencies, and so we evaluate MIDI note-extraction accuracy. Regarding the MEC-04 database, because both exact frequencies and quantized notes are available, the two possibilities are evaluated.

Regarding pitch-contour accuracy, we used the two metrics defined in Gómez et al. (2006). The first metric, melodic pitch accuracy (MPA), uses only melodic frames. Here, the pitch detection error is measured by averaging the absolute difference between the annotated pitches and the extracted ones. This error is bounded to a maximum of one semitone, i.e., 100 cents, and is computed as

$$err[i] = \begin{cases} 100, & if \left| f_{cent}^{ext}[i] - f_{cent}^{ref}[i] \right| \geq 100 \\ \left| f_{cent}^{ext}[i] - f_{cent}^{ref}[i] \right|, & otherwise \end{cases} \quad (2)$$

Here, $f_{cent}^{ext}[i]$ and $f_{cent}^{ref}[i]$ denote, respectively, the extracted and annotated fundamental frequencies (in cents) in the $i$th frame, and $err[i]$ denotes the absolute pitch-detection error in the same frame. Then, the final score (on a 0–100 scale) is computed by subtracting the bounded mean absolute difference from 100:

$$score = 100 - \frac{1}{N} \cdot \sum_{i=1}^{N} err[i] \quad (3)$$

where $N$ is the total number of frames in the excerpt under analysis.

The second metric, overall pitch accuracy (OPA), uses both melodic and non-melodic frames in the calculation of the error. There, the target value of non-melodic frames is set to 0 Hz. This metric indirectly evaluates the capability of separating the melody from the accompaniment. Therefore, inaccurate melody discrimination will lead to a decrease in this metric compared to MPA.

Regarding note accuracy, we could simply count the number of correctly extracted notes and divide it by the total number of annotated notes. However, to have a more precise figure that could cope with notes with different lengths, duration mismatches, etc., we decided to compute the note-accuracy metric as the percentage of correctly identified frames. There, the target and the extracted frequency values in each time frame were defined as the equal-temperament frequencies (ETF) of the corresponding notes. Again, both the melodic note accuracy (MNA) and overall note accuracy (ONA) are computed.

Finally, we compute two other statistics to evaluate the ability of the system to separate the melody from the accompaniment: recall (i.e., the percentage of annotated non-melodic frames that the system classifies correctly as non-melodic) and precision (i.e., the percentage of extracted non-melodic frames that are indeed non-melodic).

## Results and Discussion

Before presenting the results for the melody-identification stage, we summarize the results obtained in the previous stages for completeness. For multi-pitch detection, we achieved 81.0% average pitch accuracy (nearly the same, i.e., 81.2%, if octave errors are ignored). In this evaluation, we compared the annotated frequency in each frame with the closest extracted pitch. In case only the most salient pitch were selected in each frame, the accuracy dropped to 53.0% (65.6%, if octave errors were ignored). In our test bed, the extracted pitches were quantized to the closest ETFs, which best suits the conducted manual annotation. (Naturally, the actual pitch-detection accuracy is slightly distorted if one single annotated note spans several MIDI note numbers, e.g., in the case of glissando and vibrato.)

Regarding note determination, pitch tracks were segmented with reasonable accuracy. In terms of frequency-based segmentation, average recall (i.e., the percentage of annotated segmentation points correctly identified, considering a maximum difference of *maxOnsetDist*) was 72%, and average precision (i.e., the percentage of identified segmentation points that corresponded to actual segmentation points) was 94.7%. Moreover, the average time error was 28.8 msec (which may be slightly distorted by annotation errors), and the average semitone error rate for the melodic notes was 0.03%. In terms of MIDI note accuracy, the average accuracy was 89.2%; in other words, among all the extracted notes, 89.2% correspond to the melody (for pitch contours, the average accuracy was 86.6%). These values are higher than that for pitch detection, because undetected pitches are partly recovered by substituting empty frames in the defined notes with the corresponding ETF.

Most of the encountered difficulties in frequency-based segmentation came from operatic samples with extreme vibrato. In those cases, the number of false negatives and semitone errors was clearly above the average. In any case, in excerpts with moderate vibrato, results were quite satisfactory. As for salience-based segmentation, many false positives resulted, with a consequent decrease in average precision (41.2%), against 75.0% average recall. Finally, the results for the identification of melodic notes are summarized in Table 2 (for the song excerpts presented in Table 1).

As for the elimination of irrelevant notes (first column), we can see that 88.9% of the melodic notes are still kept after this stage. Moreover, as previously mentioned, an average of 38.1% of notes from the note-identification stage were eliminated, among which only 0.3% of true melodic notes were inadvertently deleted.

Without melody smoothing (the second and third columns in Table 2), the average MNA was 74.6, and the average ONA was 66.0%. Therefore, a high number of melodic notes are missing. After melody smoothing (the fourth and fifth columns), the average accuracy improved to 84.4% and 75.6%, respectively. Hence, our implementation of the melodic-smoothness principle amounts for an average improvement of 9.8% and 9.6% in the melodic and overall note metrics, respectively.

Several octave errors were corrected, especially in the excerpts from Battlefield Band and Pachelbel's *Canon* (IDs 11 and 1, respectively). In fact, in the presented experiments, the proposed approach was almost immune to octave errors. Indeed, disregarding these errors (i.e., performing a chroma-based

**Table 2. Results for the Melody-Identification Stage, Showing Melodic Note Accuracy (MNA) and Overall Note Accuracy (ONA)**

| | *Elim.* | *Salient Notes* | | *Melody Smoothing* | | *Elimination of FP* | | |
|---|---|---|---|---|---|---|---|---|
| *ID* | *MNA* | *MNA* | *ONA* | *MNA* | *ONA* | *MNA* | *ONA* | *%kept* |
| 1 | 98.0 | 54.1 | 53.2 | 90.5 | 89.0 | 90.5 | 89.0 | 92.3 |
| 2 | 82.2 | 70.6 | 56.9 | 80.5 | 65.5 | 80.5 | 65.5 | 97.9 |
| 3 | 95.4 | 94.2 | 91.1 | 93.6 | 90.6 | 93.6 | 90.6 | 98.1 |
| 4 | 96.6 | 86.9 | 67.5 | 95.6 | 74.2 | 95.6 | 74.2 | 99.0 |
| 5 | 85.3 | 67.4 | 51.2 | 78.1 | 57.2 | 78.1 | 57.2 | 91.5 |
| 6 | 93.6 | 70.4 | 61.4 | 90.8 | 80.6 | 90.8 | 80.6 | 97.1 |
| 7 | 98.8 | 93.5 | 87.2 | 98.2 | 91.7 | 98.2 | 91.7 | 99.5 |
| 8 | 93.6 | 90.3 | 83.4 | 93.6 | 86.4 | 93.6 | 86.4 | 100.0 |
| 9 | 86.5 | 81.3 | 64.6 | 81.3 | 64.6 | 81.3 | 64.6 | 94.1 |
| 10 | 81.1 | 75.1 | 53.8 | 75.1 | 53.8 | 75.1 | 53.8 | 92.6 |
| 11 | 95.4 | 31.6 | 31.7 | 94.2 | 92.8 | 94.2 | 92.8 | 98.7 |
| 12 | 96.5 | 91.3 | 79.2 | 92.0 | 81.8 | 95.4 | 86.8 | 98.8 |
| 13 | 98.1 | 84.7 | 84.6 | 97.5 | 97.5 | 97.5 | 97.5 | 99.4 |
| 14 | 79.5 | 71.6 | 66.9 | 73.6 | 70.4 | 74.1 | 72.7 | 93.1 |
| 15 | 90.8 | 83.1 | 61.2 | 87.3 | 64.1 | 83.6 | 74.7 | 92.1 |
| 16 | 91.7 | 69.6 | 67.6 | 87.4 | 85.5 | 86.4 | 85.8 | 94.3 |
| 17 | 98.4 | 93.3 | 91.9 | 96.7 | 95.2 | 96.7 | 95.2 | 98.2 |
| 18 | 78.4 | 66.5 | 59.8 | 66.6 | 64.2 | 66.6 | 64.2 | 84.9 |
| 19 | 69.5 | 42.5 | 40.7 | 47.4 | 44.2 | 47.4 | 45.0 | 68.1 |
| 20 | 71.5 | 69.7 | 61.9 | 69.6 | 65.6 | 70.1 | 70.8 | 98.0 |
| 21 | 87.0 | 78.9 | 69.3 | 82.3 | 74.1 | 83.0 | 78.1 | 95.4 |
| *Avg* | 88.9 | 74.6 | 66.0 | 84.4 | 75.6 | 84.4 | 77.0 | 94.4 |

evaluation), the MNA for melody smoothing was 85.2%, leading to an improvement of only 0.8%.

Regarding the elimination of false positives (sixth and seventh columns), we can see that the average overall note accuracy improved only slightly (1.4%). However, a few true melodic notes were erroneously deleted in some excerpts (jazz3 and midi1; IDs 15 and 16). The value for the MNA metric increased in some samples as a result of restoring previously truncated notes after elimination of false positives. Overall, the average MNA remained the same.

Finally, in the last column, we observe that an average of 94.4% of melodic notes were kept among those available after the elimination of ghost octave notes (first column). Only the operatic samples (IDs 18 and 19) stayed much below the average, because too many non-melodic notes were initially selected, thus misleading the melody-smoothing algorithm. In fact, in these cases, long smooth regions with several non-melodic notes are defined, which the smoothing algorithm leaves untouched.

We can see that, in the given excerpts, the overall accuracy is always lower than the melodic accuracy. In fact, our algorithm shows a limitation in disposing of false positives (i.e., accompaniment or noisy notes): 31.0% average recall and 52.8% average precision. This is a direct consequence of the fact that the algorithm is biased detecting the maximum of melodic notes, no matter if false positives are included. As mentioned, a pilot study was conducted to improve this limitation, which needs to be further elaborated.

Comparing the two databases, it can be seen that the results in our database were generally higher than those for the MEC-04 train set. This is a consequence of the fact that, in our samples, the signal-to-noise ratio (SNR, hereafter defined as the ratio between the energy of the melody and the accompaniment parts) was generally higher. In fact, a higher SNR leads to more accurate pitch detection, because fewer melodic pitches are masked. In any case, except for the operatic samples, the metric in the last column of Table 1 was nearly the same for both databases.

To compare the present results with those from the ISMIR 2004 melody-extraction contest, we also evaluated our approach with the exact frequency values used there. As a consequence, the accuracy after eliminating false positives, taking into consideration only the MEC-04 database, dropped from 80.1% (for MNA) and 77.1% (for OPA) to 75.1% and 71.1%, respectively, i.e., approximately 5–6%.

The parameters used in our algorithm were tuned using the excerpts in Table 1. Some of the defined thresholds were based on common musical practice (e.g., minimum note duration and maximum note interval), as previously defined. However, other values were empirically set, although our initial guesses were usually close to the final values (e.g., the parameters for the elimination of non-dominant notes). To evaluate the effect of parameter variance in the final results, parameter values were individually modified, typically in a ±50% range from the defined thresholds (up to ±100% in some parameters, e.g., maximum number of notes in each segment, as in Figure 6). In the conducted experiments, we observed a maximum average decrease of 7% in the MNA metric. However, a few individual excerpts had higher variations. For instance, in Juan Luis Guerra's sample (ID 10), we observed accuracy oscillations of up to +5% and –15%.

The final melody-extraction accuracy is obviously influenced by the results obtained in the first two stages of the system (depicted in Figure 1). Namely, inaccurate pitch detection automatically constrains the final results of the system. However, this has more to do with the nature of the used song excerpts than to algorithmic decisions in pitch detection (e.g., strong percussion may lead to significant pitch masking). Regarding determination of musical notes, different parameterizations have some influence on the accuracy of the following stages of the algorithm, particularly the minimum note-duration parameter. Namely, the maximum decrease in average melody note accuracy was 6.5%, which resulted from a minimum note duration of 60 msec.

**Table 3. Summary of the Results in the ISMIR 2004 Evaluation**

| *Participant* | *OPA* | *OPA (chroma)* |
|---|---|---|
| Paiva | 69.1 | 69.7 |
| Poliner | 56.1 | 57.1 |
| Bello | 50.8 | 57.7 |
| Tappert | 42.2 | 55.9 |

Bibliographic references of each system are provided in the References section (as cited in the Related Work section). Brief descriptions of Bello's and Tappert's systems can be found in Gómez et al. (2006).

To assess the generality of our algorithm and the specific parameter tuning, we evaluated it with the test set used in the ISMIR 2004 melody extraction contest, consisting of ten additional samples. The achieved results for MPA and ONA were 72.1% and 70.1%, respectively. For MNA and OPA, the accuracy was 77.4% and 75.1%, respectively. Hence, the obtained results are only slightly below those achieved in the MEC-04 training set.

For the MIREX 2005 evaluation, the pitch-contour accuracy was evaluated by measuring the percentage of correctly identified frames, which correspond to a maximum separation of a quarter-tone from the annotated frequencies. In this test bed the melodic and overall pitch contour accuracy of our algorithm dropped to 62.7% and 57.8%, respectively. Although we did not have access to the used excerpts, three representative samples were provided that allow us to deduce that this decrease in efficiency is mostly due to the use of excerpts with lower SNRs. In this case, too many non-melodic notes might have been initially selected (a consequence of basing our algorithm on the salience principle), which the smoothing algorithm was unable to fix.

For comparison, the following tables summarize, respectively, the ISMIR 2004 (Gómez et al. 2006) and MIREX 2005 evaluations. (For full details, see www.music-ir.org/evaluation/mirex-results/audio-melody/index.html.) In Table 3, the metrics shown for all participants correspond to an average between the training set and the test set. Our average

**Table 4. Summary of the Results in the MIREX 2005 Evaluation**

| *Participant* | *OPA* | *MPA* | *MPA (chroma)* |
|---|---|---|---|
| Dressler | 71.4 | 68.1 | 71.4 |
| Ryynänen | 64.3 | 68.6 | 74.1 |
| Poliner | 61.1 | 67.3 | 73.4 |
| Paiva2 | 61.1 | 58.5 | 62.0 |
| Marolt | 59.5 | 60.1 | 67.1 |
| Paiva1 | 57.8 | 62.7 | 66.7 |
| Goto | 49.9 | 65.8 | 71.8 |
| Vincent1 | 47.9 | 59.8 | 67.6 |
| Vincent2 | 46.4 | 59.6 | 71.1 |
| Brossier* | 3.2 | 3.9 | 8.1 |

Bibliographic references of each system are provided in the References section (as cited in the Related Work section). *Scores for Brossier are artificially low owing to an unresolved algorithmic issue.

results using the two sets are at present slightly higher (70.6% in the OPA metric).

As for Table 4, the algorithm corresponding to Paiva2 extracts several pitches in each frame, whereas in Paiva1, only one pitch is extracted. Selecting only one pitch led to a better overall accuracy, because the number of false positives decreased. Unlike our test-bed, the melodic accuracy is only slightly below that of Paiva1. This is a consequence of the aforementioned difficulties in detecting the correct melodic notes in this dataset.

## Conclusions

In this article, we proposed a system for melody detection in polyphonic musical signals. The achieved results are encouraging given the current initial stage of research in this topic. However, one present limitation of our approach is the inefficient discrimination between melody and accompaniment. To this end, we have conducted a pilot study on clustering of melodic notes that must be further refined. Moreover, limitations were also present in the analysis of operatic excerpts with extreme vibrato and in songs with low signal-to-noise ratios, given our initial assumption that the melody is usually salient in a mixture. As for database size, we hope to create a larger musical corpus in the future to more rigorously evaluate our system, namely in terms of some of the involved algorithmic decisions and parameter tuning.

## Acknowledgments

This work was partially supported by the Portuguese Ministry of Science and Technology under the program PRAXIS XXI. We also thank the reviewers and editors for their valuable comments and suggestions.

## References

Bello, J. P. 2003. *Towards the Automatic Analysis of Simple Polyphonic Music: A Knowledge-Based Approach.* Ph.D. Thesis, University of London.

Bregman, A. S. 1990. *Auditory Scene Analysis: The Perceptual Organization of Sound.* Cambridge, Massachusetts: MIT Press.

Brossier, P., J. P. Bello, and M. D. Plumbey. 2004. "Fast Labeling of Notes in Music Signals." *Proceedings of the 2004 International Conference on Music Information Retrieval.* Barcelona: Audiovisual Institute, Universitat Pompeu Fabra, pp. 331–336.

Dressler, K. 2005. "Extraction of the Melody Pitch Contour from Polyphonic Audio." *Proceedings of the 2005 Music Information Retrieval Exchange.* Available online at www.music-ir.org/evaluation/mirex-results/articles/melody/dressler.pdf.

Eggink, J., and G. J. Brown. 2004. "Extracting Melody Lines from Complex Audio." *Proceedings of the 2004 International Conference on Music Information Retrieval.* Barcelona: Audiovisual Institute, Universitat Pompeu Fabra, pp. 84–91.

Francès, R. 1958. *La perception de la musique* [*The Perception of Music*], trans. W. J. Dowling, 1988. Hillsdale, New Jersey: Erlbaum.

Gómez, E., et al. 2006. "A Quantitative Comparison of Different Approaches for Melody Extraction from Polyphonic Audio Recordings." Technical Report MTG-TR-2006-01. Barcelona: University Pompeu Fabra, Music Technology Group.

Goto, M. 2001. "A Predominant-F0 Estimation Method for CD Recordings: MAP Estimation Using EM Algorithm for Adaptive Tone Models." *Proceedings of the*

*2001 IEEE International Conference on Acoustics, Speech and Signal Processing.* New York: Institute of Electrical and Electronics Engineers, pp. 3365–3368.

Handel, S. 1989. *Listening: An Introduction to the Perception of Auditory Events.* Cambridge, Massachusetts: MIT Press.

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

Marolt, M. 2004. "On Finding Melodic Lines in Audio Recordings." *Proceedings of the 2004 International Conference on Digital Audio Effects.* Naples: Federico II University of Naples, pp. 80–83.

Marolt, M. 2005. "Audio Melody Extraction Based on Timbral Similarity of Melodic Fragments." *Proceedings of the 2005 Eurocon Conference.* Serbia and Montenegro: Institute of Electrical and Electronics Engineers, pp. 1288–1291.

Martin, K. D. 1996. "Automatic Transcription of Simple Polyphonic Music: Robust Front End Processing." MIT Media Lab Perceptual Computing Technical Report #399. Cambridge, Massachusetts: MIT Press.

Meyer, L. 1956. *Emotion and Meaning in Music.* Chicago: University of Chicago Press.

Paiva, R. P., T. Mendes, and A. Cardoso. 2004. "An Auditory Model Based Approach for Melody Detection in Polyphonic Musical Recordings." *Proceedings of the 2004 International Symposium on Computer Music Modeling and Retrieval.* Vienna: Springer, pp. 21–40.

Paiva, R. P., T. Mendes, and A. Cardoso. 2005a. "On the Definition of Musical Notes from Pitch Tracks for Melody Detection in Polyphonic Recordings." *Proceedings of the 2005 International Conference on Digital Audio Effects.* London: Queen Mary University of London, pp. 208–213.

Paiva, R. P., T. Mendes, and A. Cardoso. 2005b. "On the Detection of Melody Notes in Polyphonic Audio." *Proceedings of the 2005 International Conference on Music Information Retrieval.* London: Queen Mary University of London, pp. 175–182.

Poliner, G., and D. Ellis. 2005. "A Classification Approach to Melody Transcription." *Proceedings of the 2005 International Conference on Music Information Retrieval.* London: Queen Mary University of London, pp. 161–166.

Ryynänen, M. P., and A. Klapuri. 2005. "Note Event Modeling for Audio Melody Extraction." *Proceedings of the 2005 Music Information Retrieval Exchange.* Available online at www.music-ir.org/evaluation/mirex-results/articles/melody/ryynanen.pdf.

Serra, X. 1997. "Musical Sound Modeling with Sinusoids Plus Noise." In C. Roads, S. Pope, A. Picialli, and G. De Poli, eds. *Musical Signal Processing.* Cambridge, Massachusetts: MIT Press, pp. 91–122.

Slaney, M., and R. F. Lyon. 1993. "On the Importance of Time: A Temporal Representation of Sound." In M. Cooke, S. Beet, and M. Crawford, eds. *Visual Representations of Speech Signals.* New York: Wiley, pp. 95–116.

""Tsur, R. 2000. "Metaphor and Figure-Ground Relationship: Comparisons from Poetry, Music, and the Visual Arts." *PsyArt: An Online Journal for the Psychological Study of the Arts.* Available online at www.tau.ac.il/~tsurxx/Figure-ground+sound.html.

Uitdenbogerd, A. L. 2002. *Music Information Retrieval Technology.* PhD Thesis, RMIT University, Department of Computer Science, Melbourne, Australia.

Vincent, E., and M. D. Plumbey. 2005. "Predominant-F0 Estimation Using Bayesian Harmonic Waveform Models." *Proceedings of the 2005 Music Information Retrieval Exchange.* Available online at http://www.music-ir.org/evaluation/mirex-results/audio-melody/index.html.

Virtanen, T., and A. Klapuri. 2000. "Separation of Harmonic Sound Sources Using Sinusoidal Modeling." Paper presented at the 2000 IEEE International Conference on Acoustics, Speech, and Signal Processing, 8 June, Istanbul.