A PARTIAL SEARCHING ALGORITHM AND ITS APPLICATION FOR POLYPHONIC MUSIC TRANSCRIPTION

Xue Wen, Mark Sandler
Centre for Digital Music,
Department of Electronic Engineering,
Queen Mary, University of London,
Mile End Road, London, E1 4NS
{xue.wen,mark.sandler}@elec.qmul.ac.uk

ABSTRACT

This paper proposes an algorithm for studying spectral contents of pitched sounds in real-world recordings. We assume that the 2nd-order difference, w.r.t. partial index, of a pitched sound is bounded by some small positive value, rather than equal to 0 in a perfect harmonic case. Given a spectrum and a fundamental frequency f0, the algorithm searches the spectrum for partials that can be associated with f0 by dynamic programming. In section 3 a background-foreground model is plugged into the algorithm to make it work with reverberant background, such as in a piano recording. In section 4 we illustrate an application of the algorithm in which a multipitch scoring machine, which involves special processing for close or shared partials, is coupled with a tree searching method for polyphonic transcription task. Results are evaluated on the traditional note level, as well as on a partial-based sub-note level.

Keywords: sinusoids, spectral harmonic model, dynamic programming, polyphonic music transcription.

1 INTRODUCTION

Real-world tonal sounds from acoustical instruments depart more or less from the perfect harmonic model, in which partial frequencies are multiples of a fundamental frequency (f0) [1]. Partial frequencies are crucial for estimating other spectral parameters [2] that can be basic to higher-level MIR tasks. Often it is desirable to find out individual partial frequencies rather than assuming perfect harmonicity. In this paper we propose a method that finds partials from the *signal*. Given the power spectrum and an estimate of f0, it searches for partials that can be associated with f0 under assumptions weaker than the perfect harmonic model. It neither requires the input sound to be noise free, nor asks partials from different events to be well separated. It's also robust with missing or weak partials. However, interfering partials are given a *summary* amplitude only. Section 4 gives an example on how we can make use of this amplitude.

2 THE PARTIAL SEARCHING ALGORITHM

Partials of most tonal sounds can be viewed as *nearly* harmonic. Departure of *true* partial frequencies from multiples of f0 is known as inharmonicity. We denote the frequency of the p^{th} partial as k_p(f0), a function of fundamental f0 and partial index p. In particular, the fundamental frequency is k_1(f0)[1]. Since inharmonicity is something highly dependent on individual instruments, a parametric *a priori* modeling similar to [3] will be hard when we don't have enough knowledge on the instrument involved. Here we take a *posterior* approach, in which partial frequencies are estimated from the signal, while only weak assumptions are imposed on how the partials distribute.

Rather than considering the difference between k_p(f0) and pk_1(f0), as does [4], we focus on the intervals between consecutive partials. Define the 1st- and 2nd-order differences of partial frequencies with respect to p

$$\Delta k_p(\mathrm{f0}) = \begin{cases} k_1(\mathrm{f0}), & p=1 \\ k_p(\mathrm{f0}) - k_{p-1}(\mathrm{f0}), & p>1 \end{cases}, \qquad (1)$$

and

$$\Delta^2 k_p(\mathrm{f0}) = \begin{cases} 0, & p=1 \\ \Delta k_p(\mathrm{f0}) - \Delta k_{p-1}(\mathrm{f0}), & p>1 \end{cases}. \qquad (2)$$

In perfect harmonic case, we have $\Delta k_p(\mathrm{f0})=k_1(\mathrm{f0})$ and therefore $\Delta^2 k_p(\mathrm{f0})=0$ for all $p\geq 1$. With real-world signals, we assume that

$$\delta_l(p)\cdot \mathrm{f0} < \Delta^2 k_p(\mathrm{f0}) < \delta_u(p)\cdot \mathrm{f0},\ \forall p \qquad (3)$$

where $\delta_l(p)<0$ and $\delta_u(p)>0$ take small absolute values. We also assume that inharmonicity grows larger for higher partials, so $|\delta_l(p)|$ and $|\delta_u(p)|$ are allowed to increase with p. What assumption (3) does is hard-limiting the error of a 1st-order linear prediction of partial frequencies. It's trivial to extend (3) to a higher order by taking into account differences of $\Delta^2 k_p$(f0). We use 1st-order only.

Partial searching starts from the discrete sound spectrum $x=(x_k)_{k=1,2,\ldots}$, typically obtained by DFT, and a given fundamental f0. The sound may have multiple pitches, either from the same instrument or not. Constraints (1)~(3) help to prevent well separated partials from disturbing each other. The output of partial searching are partial frequencies k_p(f0), p=1, 2, …, as

[1] f0 is not quantitatively defined in this article, while k_1(f0) is its quantitative model.

well as amplitudes $a_p(\text{f0})$, p=1, 2, …. In the case one single partial being shared by multiple pitches, $a_p(\text{f0})$ is a *summary* amplitude of that partial. For simplicity we omit "(f0)" from these notations in what follows.

The searching process is one of optimization. We denote any candidate partial frequency sequence ψ as $k[\psi]=(k_p[\psi])_{p=1,2,...}$, subject to constraints (1)~(3). Given a $k[\psi]$, a reference spectrum $h[\psi]=(h_k[\psi])_{k=1,2,...}$ is constructed as

$$\begin{aligned} h_k[\psi] &= C\sum_p c_p w_k(k_p[\psi]) \\ &= C\sum_p c_p w(k-k_p[\psi]),\ k=1,2,3... \end{aligned} \tag{4}$$

where $w(k)$ is the amplitude spectrum of the window function used for the DFT, and $C = 1\Big/\sqrt{\sum_p c_p^2}$ a normalizing factor. In practice $(c_p)_{p=1,2,...}$ is selected so that $c_p \geq 0$ and $\sum_{p=1}^{\infty} c_p < \infty$. The number of partials being considered can be roughly decided by f0 and the sampling rate. Notice that while $w_k(k_p[\psi])$ is discrete, $w(k)$ takes a continuous domain. $h[\psi]$ approximates the amplitude spectrum of a signal whose partials fall at $k[\psi]$ and amplitudes are given as $(c_p/C)_{p=1,2,...}$. We define our objective function as the inner product of the data spectrum x and the reference spectrum $h[\psi]$:

$$\begin{aligned} <x, h[\psi]> &= C\sum_p c_p <x, w(k_p[\psi])> \\ &= C\sum_p c_p \sum_k x_k w_k(k_p[\psi]) \end{aligned}, \tag{5}$$

The searching algorithm tries to find an optimal ψ that maximizes this inner product:

$$\hat{\psi} = \arg\max_{\psi} <x, h[\psi]> \tag{6}$$

The term $<x, w(k_p[\psi])>$ in (5) sets up a frequency-domain matched filter to detect the p^{th} partial locally. We put special focus on local spectral peaks, i.e. maxima of $<x, w(k_p[\psi])>$. This is done by introducing another constraint on k_p:

$$\begin{aligned} &\text{"}k_p \text{ either locally maxizes} <x, w_k(k_p)>, \\ &\text{or is selected from a predefined discrete set"} \end{aligned} \tag{7}$$

The "either" condition requires a partial to fall on a spectral peak. However, when no peak is located in the searching interval, typically with missing or masked partials, we artificially add some candidates so that the search can continue. A plausible suggestion is to place a k_p candidate at $2k_{p-1}$-k_{p-2}, which is its 1st-order predicted position. Adding more candidates may slightly improve the result, at the cost of more computation.

Constraint (7) confines candidates for each partial to a discrete set. In this case the optimization can be formulated as a route finding problem: here *route* refers to a sequence of (index, frequency) pairs $\{(p, k_p)\}_{p=1,2,...}$, and each pair contributes a *mileage* of $c_p <x, w(k_p)>$, determined solely by k_p. A route starts from p=1, and terminates where p is big enough so that the tail sum $\sum_{p'>p} c_{p'} <x, w(k_{p'})>$ is ignorable. The frequency of the p^{th} partial k_p must be selected from a set determined by k_{p-1} and k_{p-2} through the constraints. Finding the optimal $\hat{\psi}$ is equivalent to finding the longest route with these settings.

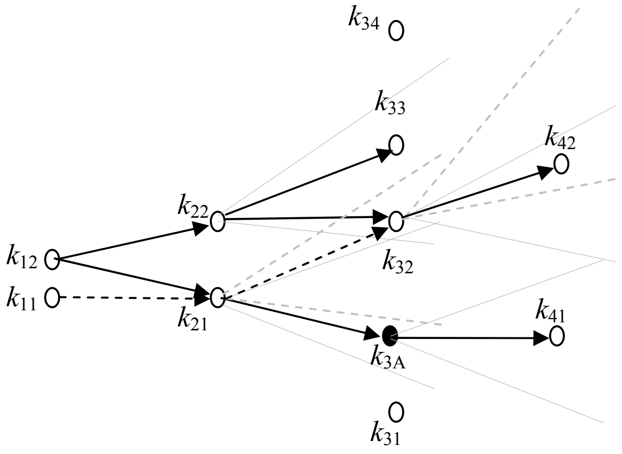

Figure 1 Partial searching routes

Figure 1 gives an illustration on how the constraints work. Each circle in the graph stands for a spectral peak, with k_{pn} denoting the n^{th} peak found for the p^{th} partial. Circles are connected by arrows to form a route. A light-coloured line pair from a circle gives the feasible range to find the next partial when the last partial comes by the arrow in the same line style. E.g. the solid line pair from k_{21} encloses the range to search for a next partial after $k_{12}\rightarrow k_{21}$, and the dotted line pair from k_{21} encloses the range to search for the next partial after $k_{11}\rightarrow k_{21}$. Of the four peaks found for partial 3, k_{32} falls in both ranges; and k_{33} falls in the successor range of $k_{12}\rightarrow k_{22}$ only. This means route $k_{12}\rightarrow k_{22}$ may lead to both k_{32} and k_{33}, while route $k_{11}\rightarrow k_{21}$ may only lead to k_{32}. As for route $k_{12}\rightarrow k_{21}$, although it also reaches k_{21}, it finds no matched peak for partial 3. However, it is extended to a *temporary lodge* at k_{3A}, from which it may continue with further peaks in higher partials, such as k_{41}. Route $k_{11}\rightarrow k_{21}$ joining $k_{12}\rightarrow k_{22}$ at k_{32} does not imply that they have become one immediately. However, if a spectral peak at the next partial, say k_{42}, falls in the intersection of feasible successor ranges of $k_{21}\rightarrow k_{32}$ and $k_{22}\rightarrow k_{32}$, then routes $k_{21}\rightarrow k_{32}\rightarrow k_{42}$ and $k_{22}\rightarrow k_{32}\rightarrow k_{42}$ are bound as one in future searching for higher partials.

We solve the constrained optimization by *dynamic programming* (DP). However, we can not apply DP directly on (p, k_p), as the candidate set for $(p+1)^{th}$ partial depends on both k_p and k_{p-1}, while the standard DP recursion allows only 1-step dependency to derive the $(p+1)$-step optima from p-step optima. To fix this problem we tie $(p\text{-}1, k_{p-1})$ and (p, k_p) together to form an *extended partial* $(p, k_p, \Delta k_p)$. We define the optimal partial route length as

$$S_p(k_p, \Delta k_p) = \max_{k_{p-1}=k_p-\Delta k_p} \Big(\sum_{p'<=p} c_{p'} <x, w(k_p)>\Big). \tag{8}$$

This is interpreted as the maximal length of all routes that terminate at (p, k_p, Δk_p). From another point of view, an extended partial is a *connection* between consecutive partials, corresponding to an arrow in Figure 1. Since the inward connection at a peak is enough to determine the outward connection candidates, 1-step dependency is satisfied and DP is directly applicable with connections. The complete algorithm is given as follows:

Algorithm 1: harmonic partial finding

```
A1.1° Set the root node $p=0$, $k_0=0$, $\Delta k_0=f0$, $S_0(k_0, \Delta k_0)=0$.
A1.2° For $p=1, 2, \dots$, do A1.3°~1.5° until the stop condition is met, i.e. $p$ is large enough.
A1.3° For each node ($p-1$, $k_{p-1}$, $\Delta k_{p-1}$) in the last iteration,
    A1.3a° calculate the feasible interval of its successor partial frequency as ($k_{p-1}+\Delta k_{p-1}+\delta_l(p)f0$, $k_{p-1}+\Delta k_{p-1}+\delta_u(p)f0$);
    A1.3b° do a maximum search of $< x,w_k(k_p) >$ regarding $k_p$ on that interval, call the found maxima $k_{p,1}$, $k_{p,2}$, …;
    A1.3c° define the feasible successor set of ($p-1$, $k_{p-1}$, $\Delta k_{p-1}$) as {($p$, $k_{p,1}$, $k_{p,1}-k_{p-1}$), ($p$, $k_{p,2}$, $k_{p,2}-k_{p-1}$), ($p$, $k_{p,3}$, $k_{p,3}-k_{p-1}$),… }, or in the case no maximum is found in (3b), as {($p$, $k_{p-1}+\Delta k_{p-1}$, $\Delta k_{p-1}$)}.
A1.4° Collect all feasible successors generated in 3° together and re-label them as ($p$, $k_{p,l}$, $\Delta k_{p,l}$), $l=1, 2, 3,\dots$; these are node candidates for partial $p$.
A1.5° For each new feasible node ($p$, $k_{p,l}$, $\Delta k_{p,l}$), calculate the optimal partial route length
    $S_p(k_{p,l},\Delta k_{p,l}) = \max_{\Delta k_{p-1,*}} S_{p-1}(k_{p,l}-\Delta k_{p,l},\Delta k_{p-1,*}) + c_p < x, w(k_{p,l}) >$
    where the maximum is taken over all its predecessors with the same $k_{p-1}$ but different $\Delta k_{p-1}$'s. Denote the $\Delta k_{p-1}$ which maximizes (8) as $\Delta^- k_p(p, k_{p,l}, \Delta k_{p,l})$.
A1.6° For the final iteration $p=P$, find $\hat{l}$ as the $l$ that maximizes $S_P(k_{P,l}, \Delta k_{P,l})$; set $k_p=k_{P,\hat{l}}$, $\Delta k_p=\Delta k_{P,\hat{l}}$, $\Delta^- k_p=\Delta^- k_p(p, k_{P,\hat{l}}, \Delta k_{P,\hat{l}})$.
A1.7° For $p=P-1, P-2, \dots, 1$, calculate $k_p=k_{p+1}-\Delta k_p$, $\Delta k_p=\Delta^- k_{p+1}$, $\Delta^- k_p=\Delta^- k_p(p, k_p, \Delta k_p)$.■
```

The algorithm searches for k_1(f0) in a vicinity of the given f0, which copes with accuracy problems of the given fundamental rather than with inharmonicity. The choice of $\delta_l(1)$ and $\delta_u(1)$ therefore differs from that for higher partials. One can force k_1(f0) at f0 by setting $\delta_l(1)= \delta_u(1)=0$, which is equivalent to starting searching at $p=2$ from root (1, f0, f0).

The amplitude of the pth partial a_p can be estimated as

$$a_p \cong < x, w(k_p) > \Big/ \|w\|^2 \quad (9)$$

where $\|w\|^2 = \sum_{k\in Z} w_k^2$ is a positive constant. Compare (9) with (8), it's apparent that

$$a_p \cong \frac{S_p(k_p,\Delta k_p) - S_{p-1}(k_{p-1},\Delta k_{p-1})}{c_p \|w\|^2}. \quad (10)$$

Equation (10) shows how partial amplitude estimation can be integrated into the partial searching algorithm.

For a stationary sound source with constant partial frequencies, one may wish to use spectra calculated from multiple frames for better estimation. Let the spectra be x_1, x_2, …, we rewrite the objective function as

$$\sum_n b_n < x_n, h[\psi] > = C \sum_p c_p < \sum_n b_n x_n, w(k_p[\psi]) > \quad (11)$$

where b_n, n=1, 2, …, are weights assigned to the frames. Equation (11) implies that we use $\sum_n b_n x_n$ instead of x for partial searching.

Input data x appears in the algorithm only in the form of inner product $<x, w_k(k_p)>$. This means if $<x, w_k(k_p)>$ is available as input, we don't have to know x. We'll show how we can make use of this property later.

3 REVERBERANT BACKGROUND: A PIANO EXAMPLE

The partial searching algorithm is tested on a piano recording of Bach's Prelude in C, BWV 846a, in which the instrument is supposed to be well-tuned on a perfect well-tempered scale with A4 at 440Hz. The recording is of high quality, yet extra sounds like pedalling and singing are heard. The piece is *partially monophonic* in that only one note is played at a time (except the last chord). However, a note may last a long period and overlap the coming-up ones, which creates polyphony.

Like many other polyphonic analyzers, our system prefers a *sparser* input with fewer concurrent sounds. In common sense, a mixture of two sounds is no sparser than any of the two. For a piano recording, we try to reduce polyphony by breaking the sound into a foreground part and a background part in a note-by-note manner. The most recent note is modeled as the foreground, and sustaining previous notes are modeled as the background. Given a note onset where a new note (i.e. the foreground) starts, we denote the spectrum before and after the onset as x_- and x_+ respectively. x_- is interpreted as the summary spectrum of all previous notes immediately before the new note, while x_+ is a combination of the new note, whose spectrum we denote as y, with those notes carried over from x_-, whose spectrum after the onset we denote as $\tilde{x}$. We make three further assumptions:

1) for any bin index k,

$$0 \le \tilde{x}_k \le x_{-k}, \quad y_k \ge 0 \quad (12)$$

2) for any bin index k,

$$x_{+k}^2 = \tilde{x}_k^2 + y_k^2 \quad (13)$$

3) y is made up of partial spectra, each in the form of $a_p w_k(k_p)$:

$$y_k = \sum_p a_p w_k(k_p), \quad a_p > 0 \quad (14)$$

By (12) we assume that the power of a sustaining note does not increase. By (13) we assume that energy is preserved in every bin. (14) is a common assumption in sinusoidal models. Apparently these are approximations only, and their solution $(a_p, k_p)_{p=1,2,\dots}$ can be non-existing or non-unique. We get around the existence problem by allowing a spectral error r in assumption 2. We combine (12)~(14) and rewrite with the error term:

$$x_{+k}^2 = \lambda_k x_{-k}^2 + \left(\sum_p a_p w_k(k_p)\right)^2 + r_k, \qquad 0 < \lambda_k < 1, \quad a_p > 0 \tag{15}$$

We minimize the residue $r=(r_k)_{k=1,2,\ldots}$ by its Euclidian norm. If $r=0$, it indicates non-unique solutions. One way to deal with the uniqueness problem is to use an additional objective function. We choose to maximize $\lambda=(\lambda_k)_{k=1,2,\ldots}$ by its Euclidian norm on the constraint $r=0$, which implies *maximal removal* of the background. With fixed $(k_p)_{p=1,2,\ldots}$, by minimizing r (and maximizing λ when $r=0$) we get $a=(a_p)_{p=1,2,\ldots}$. We write

$$< y, w(k_p) > = \sum_k y_k w_k(k_p) = \sum_k \sum_p a_p w_k^2(k_p) \cong a_p \|w\|^2 \tag{16}$$

where inter-partial *spectral leakage* has been assumed ignorable. Equation (16) explicitly evaluates $<y, w_k(k_p)>$, which enables partial searching on the foreground signal y using Algorithm 1.

In our test we measure partial frequencies of every note, using multiple frames starting from the onset. The ideal f0, e.g. A4=440Hz, is used to start partial searching. The means μ_p(f0) and standard deviations σ_p(f0) of partial frequencies are calculated for notes that appear at least 9 times. We have no ground truth on the exact partial frequencies. However, by assuming partial frequencies being constant for a given key, we can study how reliable the searching is using σ_p(f0). In general the envelope of σ_p(f0) increases slowly with p, until after some point the increase becomes dramatic. Figure 2 depicts σ_p(f0) against k_p for C4, $^\#$F4 and A4, both axes are measured in DFT bins (1bin=10.77Hz). While minor σ_p(f0) may be credited to local noises, large ones generally imply searching failure.

We set 1 bin as the threshold for judging whether a partial is reliably measured, and define p'(f0) as the maximal P that satisfies σ_p(f0)<1, $\forall p \leq P$. Results are given in Table 1. Frequencies are given in bins. The first 10 partials are successfully captured most of the time. More than 99% of total energy is enclosed in the first p'(f0) partials.

Table 1 Evaluating partial frequency measurement

Pitch	p'(f0)	$k_{p'}$(f0)	Pitch	p'(f0)	$k_{p'}$(f0)
G2	22	204.01	E4	10	313.68
D3	24	340.16	F4	8	264.29
F3	16	265.72	$^\#$F4	12	430.22
G3	15	280.69	G4	12	457.17
A3	17	362.35	A4	10	425.48
B3	12	281.42	B4	5	232.07
C4	16	405.62	C5	3	147.05
D4	17	490.79	D5	8	454.57
E4	10	313.68			

It's also interesting to look at how partial frequencies depart from perfect harmonic model. We collect $\Delta^2 k_p$ and the difference between k_p and pk_1 in Table 2. Results are given for the first 12 partials of keys A3, C4 and G4. All frequencies are given in bins. For all three keys $\Delta^2 k_p$

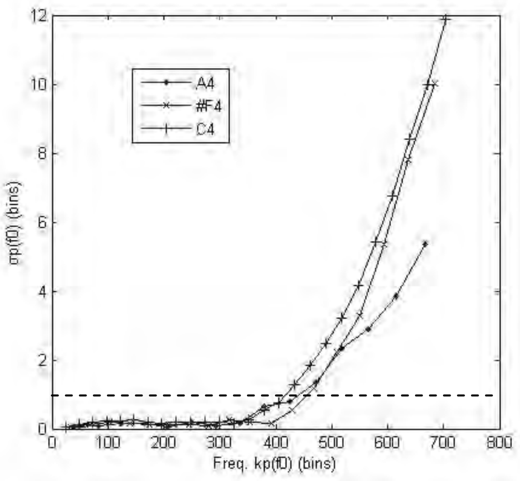

Figure 2 Standard deviation of partial frequencies

are always positive and increasing in the long trend, which supports the positive adaptation of $\delta_u(p)$ with p. k_p-pk_1 gives a hint on how much will be lost when using a perfect harmonic model. With most popular window functions (Hann, Hamming, Kaiser, etc), a frequency error above 1 bin usually implies a big error in amplitude estimation, and an error above 2 bins usually means that the partial is lost.

Table 2 Evaluating inharmonicity

p	A3		C4		G4	
	$\Delta^2 k_p$	k_p-pk_1	$\Delta^2 k_p$	k_p-pk_1	$\Delta^2 k_p$	k_p-pk_1
2	0.059	0.06	0.033	0.03	0.152	0.15
3	0.019	0.14	0.030	0.10	0.005	0.31
4	0.037	0.25	0.079	0.24	0.264	0.73
5	0.049	0.42	0.068	0.45	0.208	1.36
6	0.087	0.67	0.082	0.74	0.407	2.39
7	0.138	1.06	0.247	1.28	0.409	3.84
8	0.122	1.57	0.087	1.91	0.380	5.66
9	0.069	2.16	0.188	2.72	0.584	8.07
10	0.110	2.84	0.302	3.84	0.723	11.2
11	0.107	3.63	0.110	5.06	0.535	14.9
12	0.342	4.77	0.246	6.53	0.560	19.1

4 APPLICATION FOR POLYPHONIC MUSIC TRANSCRIPTION

By *transcription* we mainly mean pitch or multipitch identification. The partial searching algorithm associates spectral peaks, either perfectly or nearly harmonic, to a *hypothesis* f0. The results on frequencies and amplitudes compose an informative point to start pitch estimation. In [5] we have shown how a *partially monophonic* piece can be effectively transcribed using the partial searching algorithm only. However, as the algorithm is designed for single pitch, it's not directly applicable for transcribing polyphonic music. Instead, we start from its outputs for polyphonic transcription .

Before we proceed with multipitch estimation, it's helpful to remove those unlikely pitches from further consideration. To do this, we require at least one of the first three partials to appear as a spectral peak with amplitude above a threshold *th*. A pitch candidate is removed if no peaks are located in step A1.3b° for its first three partials, or if the amplitudes of located peaks all fall below *th*. This trimming can be integrated into the partial searching Algorithm 1.

We build a scoring machine in the form $S(\psi, x_+, x_-)$, where ψ is a hypothesis pitch set, x_+ and x_- are spectra before and after the onset. The larger $S(\psi, x_+, x_-)$, the more likely ψ being the solution. The score is calculated as the sum of individual contributions of all partials of ψ. The p^{th} partial of the n^{th} pitch with amplitude a_p contributes $c_p\alpha(a_p)$ to the score, where $c_p>0$, $p=1,2,\ldots$, are partial weights, and $\alpha(\bullet)$ is a nondecreasing function. We let c_p decrease slowly with small p's for which the partial searching results are more valid, and approach zero when p is large. $\alpha(\bullet)$ is designed as

$$\alpha(a_p)=\begin{cases}\tanh\left(\dfrac{a_p-Floor(k_p)}{A}\right), a_p > Floor(k_p)\\ 0, \quad otherwise\end{cases} \tag{17}$$

for controlling the magnitude of individual contributions, where $Floor(k)$ is a floor level at spectral channel k, and A is a relatively large constant. $A\rightarrow\infty$ implies a linear $\alpha(\bullet)$.

However, if a partial is shared by multiple pitches, or if certain partials of multiple pitches are very close, the frequencies tend to be less valid and the amplitudes are *summary* amplitudes of all partials involved. When calculating the score, we make sure that a shared partial is summed only once, and very close partials go through a *masking* process before subjected to $\alpha(\bullet)$, as follows.

Suppose we have n partials with close frequencies k_1, k_2, …, k_n, their summary amplitudes given by a_1, a_2, …, a_n. Let one partial located at k_l have a *true* amplitude b_l, i.e. it contribute $b_lw_k(k_l)$ to the spectrum x. Accordingly, it contributes an amount of $< b_lw(k_l), w(k_m) > / \|w\|^2$ to the summary amplitude a_m. We can write

$$\begin{aligned}&\frac{< b_lw(k_l), w(k_m) >}{\|w\|^2}\\ &\cong b_l\frac{< w(0), w(k_m-k_l) >}{\|w\|^2} = b_lr_w(k_m-k_l)\end{aligned} \tag{18}$$

where $r_w(k)$ is determined solely by the window function w. The summary amplitude a_m is modeled as the sum of the contributions of all n partials, i.e.

$$a_m \cong \sum_l b_lr_w(k_m-k_l), \; m=1,2,\ldots,n \tag{19}$$

We derive a *masking* process from (19), in which the largest partial, say k_l, is chosen as the masker; then for any $m\neq l$, a_m is decreased by $a_lr_w(k_m-k_l)-a_mr_w^2(k_m-k_l)$, or set to 0 if it's smaller than that amount. This masking may go on with the 2nd largest partial, etc. For our task, we mask with the largest partial only.

Multipitch searching works in an iterative way: given the m_N best N-pitch solutions $\{\psi_m^N\}_{m=1,2,\cdots,m_N}$, it searches for m_{N+1} best (N+1)-pitch candidates by adding a new pitch to an N-pitch one. We start by testing all single pitches, find the best m_1; then test all pitch pairs that contain one of the m_1 best pitches, find the best m_2; then test all 3-pitch sets that contain one of the m_2 best pitch pairs, etc. The complete process is given as follows.

Algorithm 2: multipitch searching

```
A2.0° Given $x_+$, $x_-$, $(m_N)_{N=1,2,\ldots}$.
A2.1° Derive trimmed pitch candidate set $\mathcal{P}$ using Algorithm 1, recording all frequencies and amplitudes; set
    $\Psi_1=\{\{k_{1m}\}|\ m=1,2,\ldots,m_1\}$, where $k_{11}$, $k_{12}$, … are the best $m_1$ pitches;
A2.2° For N=1, 2, …, do A2.3°~2.5° until $\Psi_N=\emptyset$ or N meets a preset upper bound;
A2.3° Set $\Psi_{N+1}=\emptyset$;
A2.4° For $m$=1, 2, …, $m_N$,
    A2.4a° for all $k\in\mathcal{P}\setminus\psi_m^N$, do A2.4b°~2.4f°;
    A2.4b° if $S(\psi_m^N+\{k\})$ has been calculated once, skip A2.4c°~2.4f°;
    A2.4c° calculate score $S(\psi_m^N+\{k\}, x_+, x_-)$;
    A2.4d° if $S(\psi_m^N+\{k\}, x_+, x_-) < S(\psi_m^N, x_+, x_-)$, skip A2.4e°~2.4f°;
    A2.4e° if the size of $\Psi_{N+1}$ is smaller than $m_{N+1}$, insert $\psi_m^N+\{k\}$ into $\Psi_{N+1}$, skip A2.4f°;
    A2.4f° denote the element in $\Psi_{N+1}$ with the smallest score as $\psi_{\tilde{m}}^{N+1}$; if $S(\psi_m^N+\{k\}, x_+, x_-) > S(\psi_{\tilde{m}}^{N+1}, x_+, x_-)$, then
        replace $\psi_{\tilde{m}}^{N+1}$ from $\Psi_{N+1}$ with $\psi_m^N+\{k\}$;
A2.5° If $|\Psi_{N+1}|<m_{N+1}$, set $m_{N+1}=|\Psi_{N+1}|$;
A2.6° Output results. ■
```

A2.4b° is a step to stop multiple calculations on the same multipitch. Algorithm 2 can be summarized as growing a tree: each branch from a node adds a pitch, meanwhile brings an increase in the score. A limit m_N is imposed on the number of nodes on level N so that only those *predominant* branches grow on.

Algorithm 2 outputs a set sequence $(\Psi_N)_{N=1,2,\ldots}$, where Ψ_N contains m_N best N-pitch candidates. If the number of concurrent notes is known as n, we can select the best element in Ψ_n as the result. Finding this number, however, is not trivial. A naïve way is to compare the scores. In favour of less notes, we start from N=1 and allow N to increase as long as the best score increases by more than some preset level. We also look at the total amplitude of all partials, which measures how much of the amplitude spectrum has been resolved.

We run our test on a recording of Bach's Fugue in C, BWV 846b, of high quality with the real-world noises like pedalling and singing. The piece is a 4-part fugue. A maximum of 4 keys are played at a time. Altogether 736 notes are played in 406 note groups (by *note group* we mean notes played at the same time), forming 20 4-pitch chords, 79 3-pitch chords, 112 2-pitch chords, as well as 195 single notes. We assume that the note onsets are known. For onset detection and verification using the partial searching Algorithm 1, one may refer to [5, 6]. At each onset, we calculate the background and foreground spectra x_- and x_+. 72 keys from A1 to #G7 are considered as note candidates.

Evaluation is done both note-wise and note-group-wise. A note is *correctly identified* (abbr. CI) if the detected pitch coincide with the labelled one. Note-wise errors are classified into harmonic and non-harmonic ones. Harmonic errors include harmonic replacement (HR) in which a pitch is replaced by another harmonic[1] pitch, harmonic insertion (HI) in which a spurious pitch, harmonic to a CI pitch, is found, and harmonic missing (HM) in which a labelled pitch, harmonic to a CI pitch, is not found. Non-harmonic errors are insertion (I) and missing (M) errors excluded from harmonic ones. Note-wise errors of the same type within a note group are counted as *one* note-group-wise error of that type. A note group is CI if it has no errors.

Before presenting the more conventional note-level evaluation, we do a sub-note level evaluation of the error types, in which we count percentages of missing and inserted partials, rather than those of notes. Results are given in Table3 separately for the 5 error types and for groups of 1, 2, 3 and 4 (column N) notes. For example, 11.1/27.8 in the top left says that with all HR type errors in single-pitch groups, 11.1% of all partials of the replaced notes are not found in the identified notes, and 27.8% partials of the replacing notes are not found in the labelled notes. For all HI errors, only 2.01% partials of inserted notes are truly spurious, while for non-harmonic insertion the ratio is 83.8%. For note missing types, corresponding ratios are 25.4% compared to 62.5%. These support our classification of error types: on partial level, harmonic errors do less harm than non-harmonic ones.

Table 3 Partial-level evaluation for notes

N	HR(I/M, %)	HI(%)	HM(%)	I(%)	M(%)
1	11.1 / 27.8	1.39	-	88.0	-
2	0 / 37.7	2.03	50	80.4	70
3	1.99 / 31.6	4.17	15.4	72.0	65.8
4	4.31 / 25.4	0	28.9	70.8	43.3
1~4	2.81 / 32.1	2.01	25.4	83.8	62.5

Table 4a lists the note-group-wise results. Numbers of errors of each type are given separately for groups of 1, 2, 3 or 4 notes. Since a group may have multiple errors, the total number of CI and errors may exceed the number of note groups.

Table 4b lists the note-wise results. Numbers of errors of each type are given separately in Table 4b for note groups with 1, 2, 3 or 4 pitches. In this table the equality total=CI+HR+HM+M holds.

Table 4a Note-level evaluation for note groups

N	total	CI	HR	HI	HM	I	M
1	195	108	12	44	0	49	0
2	112	45	32	29	4	21	5
3	79	15	33	20	13	13	12
4	20	1	13	2	8	2	3
1~4	406	169	90	95	25	85	20

[1] We call two pitches *harmonic* if one is a rough multiple of the other. This includes both harmonic and subharmonic cases in the strict sense.

Table 4b Note-level evaluation for notes

N	total	CI	HR	HI	HM	I	M
1	195	183	12	64	0	64	0
2	224	182	33	37	4	22	5
3	237	173	39	20	13	14	12
4	80	50	17	2	9	2	4
1~4	736	588	101	123	26	102	21

A note group identification rate of 42% is obtained at 22% HR, 24% HI, 6.2% HM and 21% non-harmonic insertion errors. A note identification rate of 80% is obtained at 14% HR, 17% HI, 3.5% HM and 14% non-harmonic insertion errors. In both cases insertions are several times more than missing errors, implying the result may be improved a bit by adjusting the threshold.

5 CONCLUSION

In this paper we propose a partial searching algorithm based on 1st-order frequency prediction using a revised dynamic programming method. Results show that the algorithm is able to correctly locate partials of a piano recording when perfect harmonic model would probably fail. We also give a tree-searching method for multipitch identification, using the partial searching algorithm at front end. We evaluate the transcriber with a piano recording both on note level and on sub-note level. In the latter case we propose to measure how harmful an error is in polyphonic transcription by counting partials. Results support our classification of transcription errors into harmonic and non-harmonic ones.

6 ACKNOWLEDGMENTS

This work was supported by EU-FP6-IST-507142 project SIMAC (acronym for Semantic Interaction with Music Audio Contents).

REFERENCES

[1] White H E, White D H. Physics and music: the science of musical sound. Saunders Coll. Pub. Philadelphia. 1980.

[2] F Keiler, S Marchand. Survey on extraction of sinusoids in stationary sounds. 5th Int. Conference on Digital Audio Effects (DAFx). Germany. 2002.

[3] A Klapuri. Wide-band pitch estimation for natural sound sources with inharmonicities. AES 106th Convention. Munich. 1999.

[4] S Godsill, M Davy. Bayesian harmonic models for musical pitch estimation and analysis. ICASSP. 2002.

[5] Wen X, Sandler M. Transcribing piano recordings using signal novelty. AES 118th Convention. Barcelona. 2005.

[6] Wen X. Acoustical onset detection using phase information. 18th Int. Cong. Acoustics. Kyoto. 2004.