On the Transcription of Musical Sound by Computer

by James A. Moorer
Stanford University

ABSTRACT

An examination of the problem of producing a written score from a piece of polyphonic music has been done with the result that a program to accomplish this end for a restricted class of input samples has been written and debugged. The program uses bandpass filtering to extract individual harmonic data. Two examples are presented here that show the viability of the system given the restrictions on the music under analysis.

INTRODUCTION

Musical dictation is commonly taught to freshman and sophomore music students at music schools everywhere. We might ask if it is possible to program the computer to do as well at this task as people can do. Although the answer at this time seems to be negative, we can make a program which does this to some extent, given certain restrictions on the music used.

This problem is somewhat analogous to the "cocktail party" problem in speech recognition, that is, the perception of a speech line in the presence of contaminating noise of equal or greater amplitude than the signal of interest. The approach taken here is to decompose the signal as much as possible to separate the voices, and then to reconstruct the signal by inference on this analysis.

Our first restriction on the type of music to be analysed is that the instruments are all *harmonic*. We must exclude drums, chimes, gongs, cymbals, and other such instruments with partials which are not nearly-integral multiples of the fundamental frequency (i.e., the pitch of the note). This restriction, like most of the following restrictions, is not inherent in the process, but is merely for the purpose of simplifying the immediate task. The problem with non-harmonic tones is that there is no way to infer a small number of possible fundamental frequencies from the pitch of a given partial. Simultaneity of the partials is the only cue for this type of instrument. This demands that the exact beginning time of each partial be measured very accurately, so that coincidence can be adequately judged.

The next restriction is that the pitches of the tones be *piecewise constant.* This eliminates vibrato and glissando. This restriction is quite severe, in that almost all strings and vocalists use vibrato. The problem is that an adaptive technique must be used to track the frequency as it changes with time. This was out of the scope of the current investigation.

One last restriction is that *the fundamental of a note must not be at the same frequency as a harmonic of another note sounding simultaneously.* This is an extremely strong restriction because this eliminates most common musical intervals, like octaves, twelfths, etc. Musical intervals tend to be ratios of small integers.

For the time being, we will address only duets. This is not an inherent limitation of the technique, but again is only a convenience.

ACOUSTIC PROCESSING

To begin, we digitize a piece of musical sound with an analog to digital converter. The one used for this project was an Analogic 14-bit converter. We used 25 kc sampling rate throughout, with a 10 kc TTE aliasing filter.

Once the sound has been digitized, our first task is to filter out each harmonic of each note separately. To do this, we need to know what frequency to filter. We can use the "shotgun" approach and make a dense covering of the frequency spectrum with bandpass filters, but the computation involved is then immense. It can be reduced somewhat by taking advantage of a feature of musical sound. We hope to create a method by which many of the frequencies in a dense covering may be eliminated.

Any two notes of the diatonic scale played simultaneously have a frequency ratio that may be approximated by a ratio of integers. Historically, our scale evolved from just intonation, where the notes were, in fact, ratios of small integers. What this means is that any two notes has a least common fundamental frequency, such that the notes are integral multiples of this fundamental. This means that any two notes played together produces a waveform that is periodic with a period consisting of this low common fundamental. Let us call this frequency a "root" of the two notes.

As an example of this, Figure 1 shows the waveform of two violins playing together. One is playing at about 194 Hz, the other at about 165 Hz. This is an interval of a minor third. You can see that the waveform is periodic with a period of about 31 milliseconds. This corresponds to about a 32.2 Hz signal. 165/5 is about 33, and 194/6 is about 32.3, so indeed, there is a root related to a common factor of these tones. With just intonation, the interval of a minor third represents a frequency ratio of exactly 6/5.

We can use a periodicity detector [Gold, Miller, Moorer 1974, Noll, Sondhi] to determine this period for us. Figure 2

shows the output of one such detector [Moorer 1974]. The deep minimum at about 31 milliseconds indicates the period. Using such a detector as a preprocessor, we can estimate the roots of each part of the music. We can then set the filter frequencies to integral multiples of this root and be assured that we will capture each of the harmonics in some filter. This can save us as much as a factor of three in filter applications from the dense covering.

This technique also works with more than two notes only if the notes are in an unambiguous harmonic (used here in a musical sense) relation. If there are notes that form an ambiguous harmony, such as a diminished triad, then the root will not be unique, and savings may not be realized.

Once we have the roots for each part of the sound sample, we may apply the bandpass filters. The output of each bandpass filter can be processed to discover if it is an active harmonic. We can tell this by determining if the filter output consists of a periodic signal of a frequency within the passband of the filter. This is done, again, by use of a periodicity detector. Figure 3 shows the output of a bandpass filter in the top trace. The second trace shows the output of the periodicity detector. The third trace shows the periodicity detector output again with all the easily detected spurious traces omitted. The straight line through the plot indicates the average pitch of the harmonic. The vertical bar at the end of the straight line is two sample standard deviations high.

Figure 4 shows a plot in a case where there really was not any harmonic present at that frequency. The variance of the pitch estimate was so high that the trace was eliminated.

INTERMEDIATE-LEVEL PROCESSING

At this point, we have traces of actual harmonics, and noise traces. We must have a way of separating them. We do this by assigning a score to the trace that is a reflection of its quality. This can be done by generating a power plot of the filter output and by fitting the power and the frequency of each harmonic with low order (6 for the power, 2 for the frequency) polynomials. The residual error of the power polynomial is a measure of how smooth and simple the power

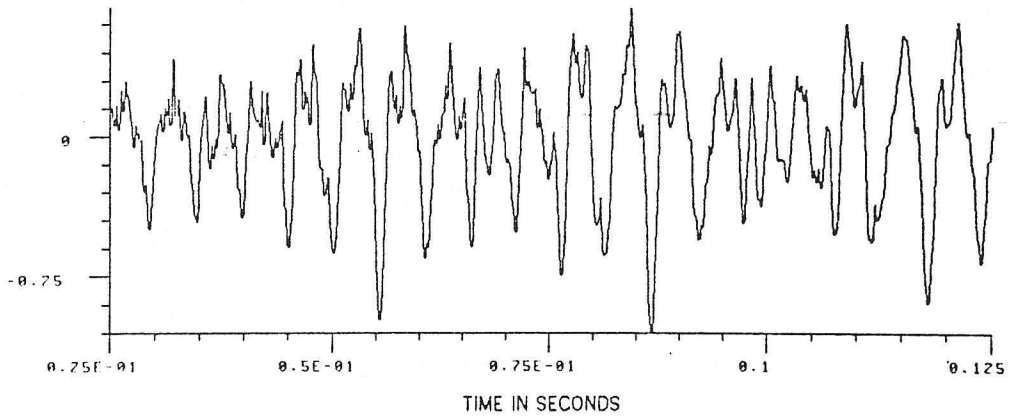

Figure 1. A segment of a waveform of a violin duet. The first violin is playing a 194 Hz. note, the other is playing a 165 Hz. note. We see that the composite waveform is periodic with a period of about 31 milliseconds (32.2 Hz.). This is because the frequencies are approximately in a ratio of 6/5, such that 195/6 is about 32.3 Hz., and 165/5 is about 33 Hz. The interval being played is a minor third.

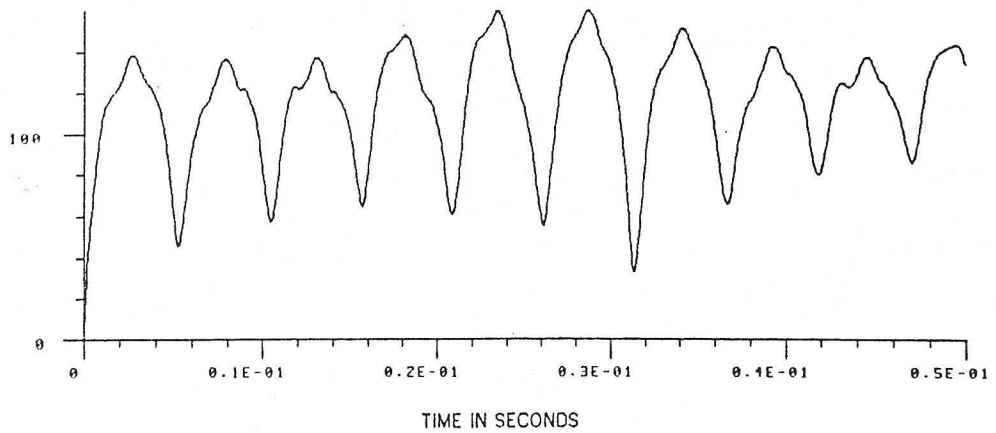

Figure 2. Output of the optimum-comb pitch detector when applied to the waveform of Figure 1. We see a deep minimum at about 31 milliseconds, representing the period of the waveform.

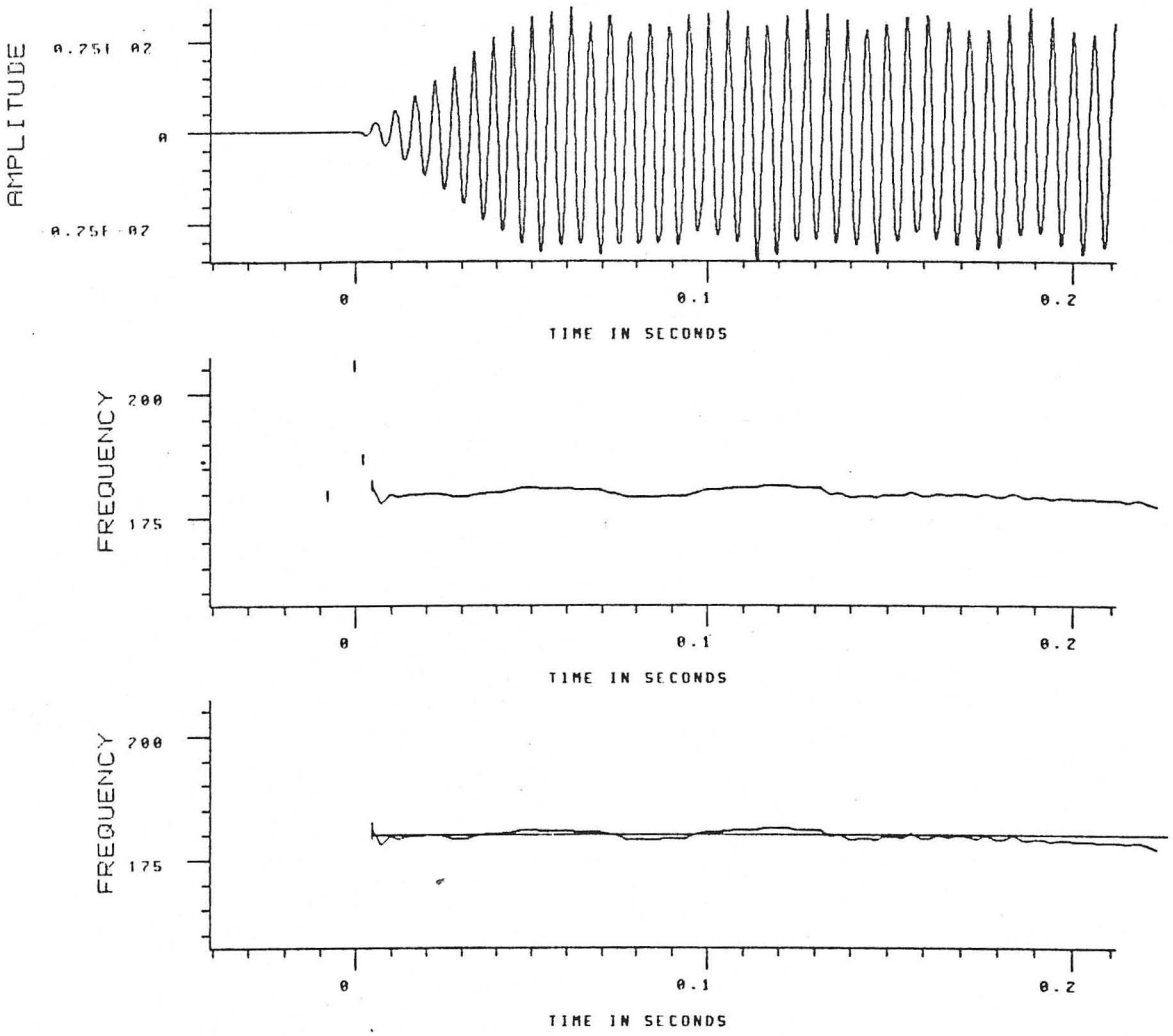

Figure 3. Processing on the output of a bandpass filter. The upper plot is the output of a bandpass filter which is centered over the fundamental frequency of a violin tone. The center plot shows the output of a pitch detector when applied to this signal. The line denotes the pitch at each point in time. The lower plot shows the output of the pitch detector after the clearly spurious traces have been eliminated. The horizontal line represents the average pitch of this harmonic. The vertical bar on the end of the horizontal line is two sample standard deviations high.

curve is. Noisy traces will get a much higher residual error. Since harmonics may be of different duration, we must normalize the residual error for length. This is done by dividing by the residual by the χ-squared value for the number of points in the harmonic.

Since the frequencies are restricted to be piecewise constant, the existence of first and second order coefficients is a measure of deviation of the frequency curve from constancy.

To compute the score of a harmonic, we take a weighted average of the normalized residual error, and the magnitudes of the first and second order frequency coefficients. When we take the average energy in the harmonic and divide it by this score, we get a number that is larger for better harmonics.

As often as not, each actual harmonic will get several traces. Application of filters to adjacent frequencies sometimes results in a poor tracing of the harmonic. This can easily be recognized and lumped into a single trace. We are then ready to infer the notes from the harmonics.

To get the notes, we start with the strongest harmonic, and look at one-half and one-third of its frequency for a possible fundamental frequency. If there is a lower harmonic with reasonable strength, we use it as the fundamental of the note, otherwise we assume we have the fundamental. We then look at integer multiples of the fundamental frequency for its harmonics. This forms a *note hypothesis.* This hypothesis is tested for viability in a number of ways. There are simple tests which can eliminate many spurious cases. One is to check the even harmonics. If there are no odd harmonics, except the fundamental, chances are that the fundamental is a noise trace and the note is really an octave higher. Likewise with harmonics that are multiples of the third harmonic. The final test involves looking at all the hypotheses that occur simultaneously and eliminating ones that are more than an order of magnitude weaker than the others. This is important for eliminating

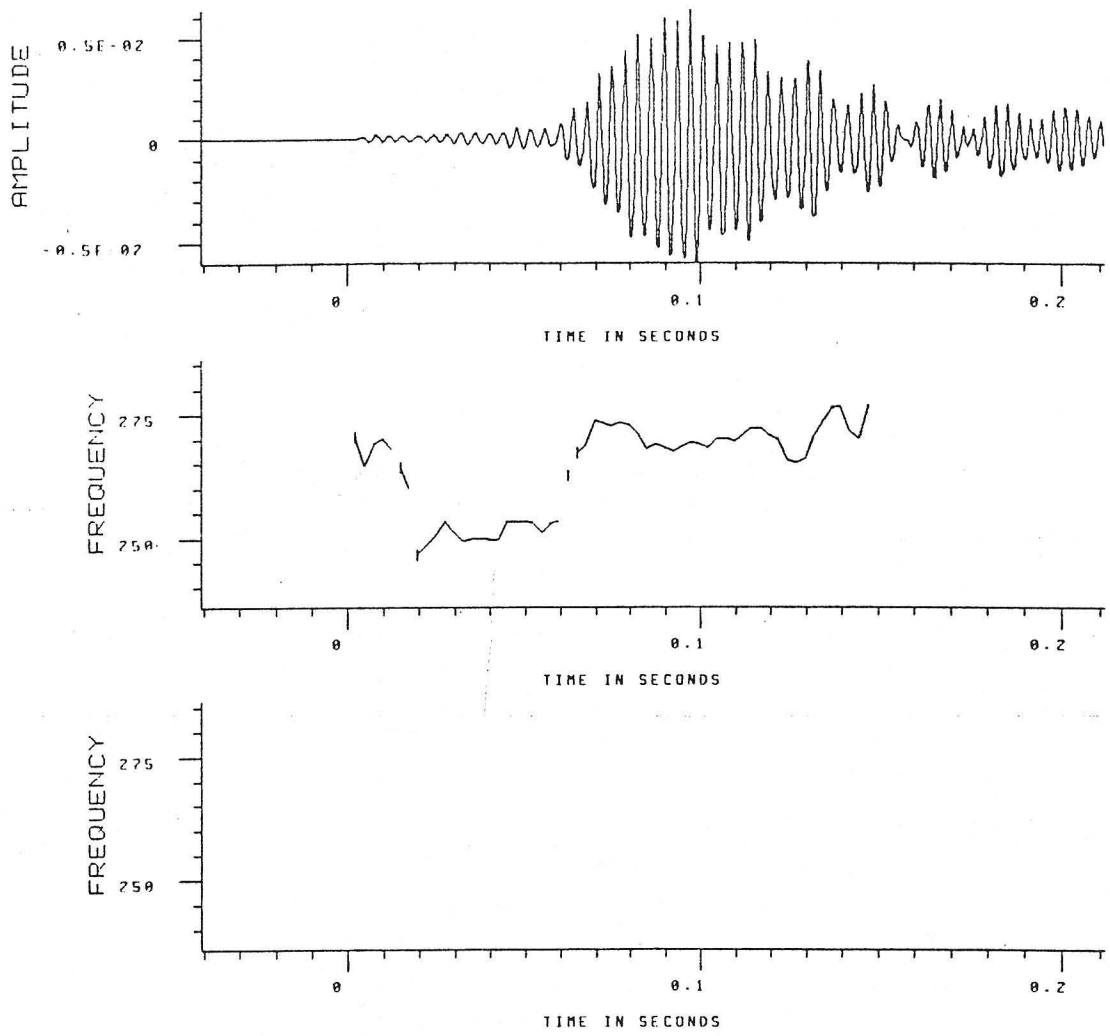

Figure 4. Processing on the output of a bandpass filter. As in Figure 3, the upper plot is the output of a bandpass filter which is centered over the fundamental frequency of a violin tone. The center plot shows the output of a pitch detector when applied to this signal. The line denotes the pitch at each point in time. The lower plot shows the output of the pitch detector after the clearly spurious traces have been eliminated. This is a case where there is in fact no harmonic present. In this case, all traces are eliminated.

things that are in the signal but are not audible, like open strings on a guitar or violin that resonate on their own without being struck.

MELODIC GROUPING AND PRINTING

The next step is to group the notes into melodic lines. This is the first place in the program that the assumption of two voices is used explicitly.

The problem is to assign each note to a voice. The melody is then constructed as the concatenation of consecutive notes belonging to a single voice. The problem is somewhat complicated by inaccuracies in the judgement of the starting and ending times of notes. Sometimes consecutive notes in a given voice will overlap and consequently exist at the same time. We get around this by requiring that notes overlap considerably before they are judged to be in different voices.

We start by finding all the notes that overlap completely, such that one note occurs entirely within the duration of another note. This way, we establish "islands" of known correct labeling. Oh yes, we also assume that the voices do not cross! (for simplicity). We then look at the spaces between the islands. A number of these can be labeled by logical extension from the islands. Those remaining after this are searched combinatorially for a suitable labeling. Luckily, there are very few such notes remaining. One 16-second piece of music with 54 notes had only 7 unlabeled notes left. This means only 128 cases must be explored.

We can recognize the "best" labeling by eliminating the clearly spurious labelings, like voices that cross or move backwards in time, and using simple frequency difference between adjacent notes as a measure of goodness. We produce an overall score by summing the magnitudes of the differences

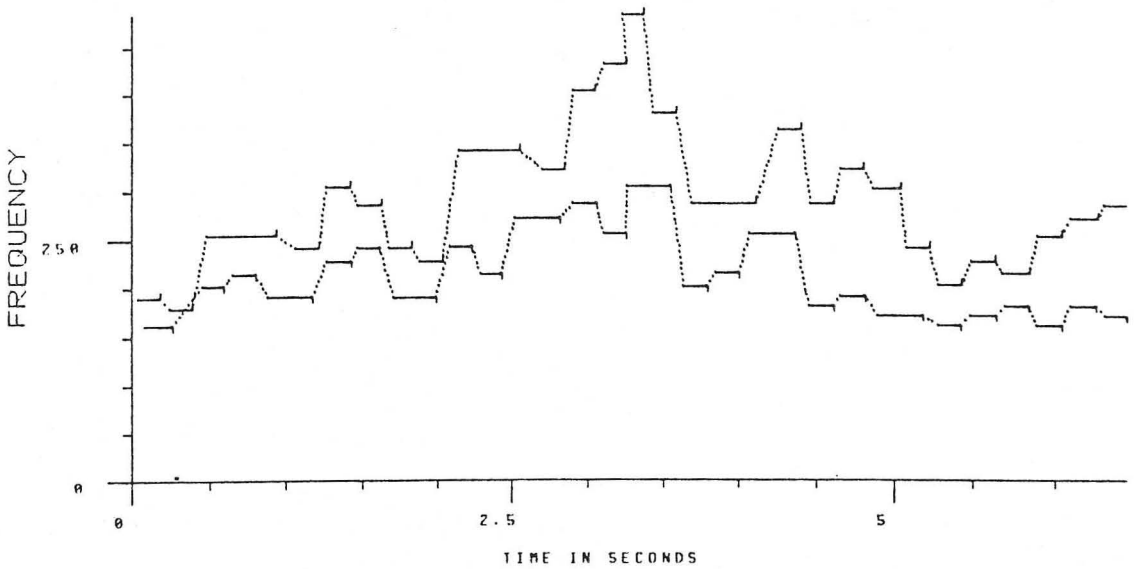

Figure 5. Melodic grouping for a pseudo-violin duet, the score of which is shown in Figure 6. The horizontal lines represent notes. The dotted lines represent connections between notes and indicate melodic connectedness.

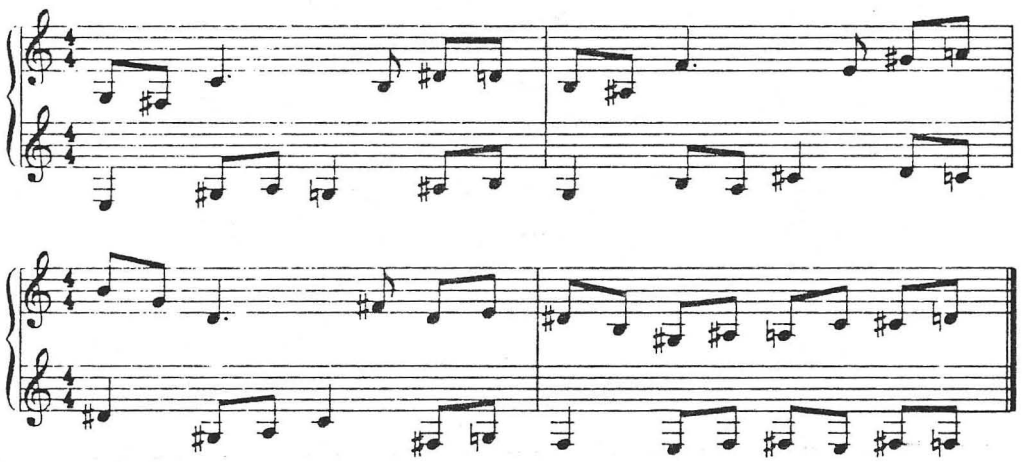

Figure 6. Original score of the first 4 measures of a violin duet.

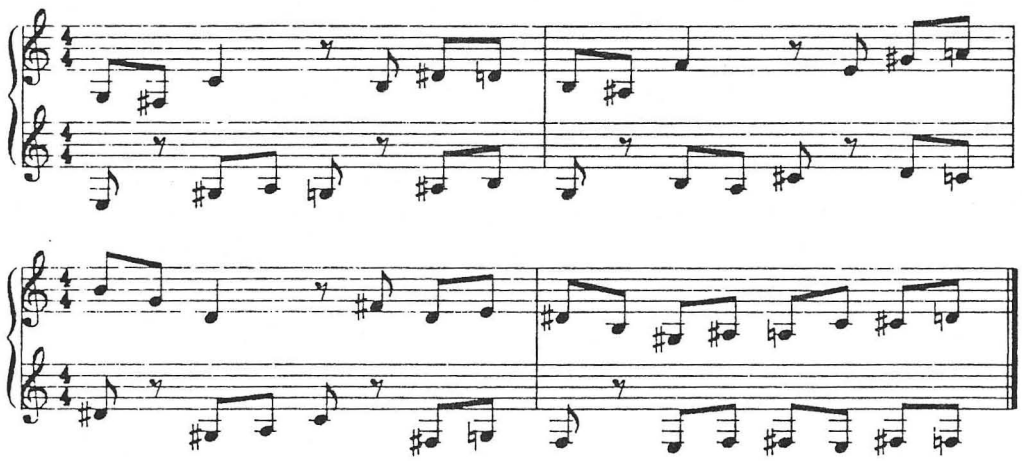

Figure 7. Final output of the program for the score in Figure 6 above. The piece was synthesized by the computer to provide noise - free signals. Notice that the note durations are consistantly underestimated. This is because the noise rejection threshold is set quite high to eliminate noise tones.

between adjacent notes of each proposed labeling. The best score (lowest total frequency difference) seems to reliably indicate proper voicing.

Figure 5 shows the results of a melodic grouping for a two-violin piece. The horizontal solid bars denote a note of that duration. The dotted lines between the bars indicate melodic adjacency. We can see from this that a small amount of overlapping is allowed.

The next task is the actual manuscripting. Luckily, we have a program that already automates much of this tedious task. This is Leland Smith's marvelous manuscripting system [Smith]. Our job is to convert these melodic lines into traditional note names simply by taking the name of the nearest note of the diatonic scale. To allow for slight mistunings, we first normalize by looking at the intervals between adjacent notes. We then label one note, and judge all the rest by their intervallic relation to this one "cornerstone" note. This way, a mistuning that places us exactly halfway between two notes a half-step apart will not cause some notes to get rounded up and others rounded down.

To get note durations, we must accept something as the smallest note duration allowed. Otherwise, the floating-point note durations would result in extremely complex notations. In fact, the notation would be non-terminating. We must ask the user for the minimum note duration printed and what the time signature is. For some kinds of music, the key and time signature can be inferred [Longuet-Higgins and Steedman], but not in general.

We must find the "fundamental duration", of which every other note is an integral multiple (ignoring such things as triplets for the time being). This is done by making a histogram of the durations of notes and the number of notes at that duration. This histogram is then searched for its peak. We check also one-half and one-third of that time to see if there are notes that exist at those durations. If there are, then their length is taken to be the fundamental duration, and all other notes are represented as multiples of that duration.

There are some problems even in deciding how to print the note durations. For instance, a note that is 1.5 beats long in 4/4 time is printed as a dotted quarter if it occurs on the

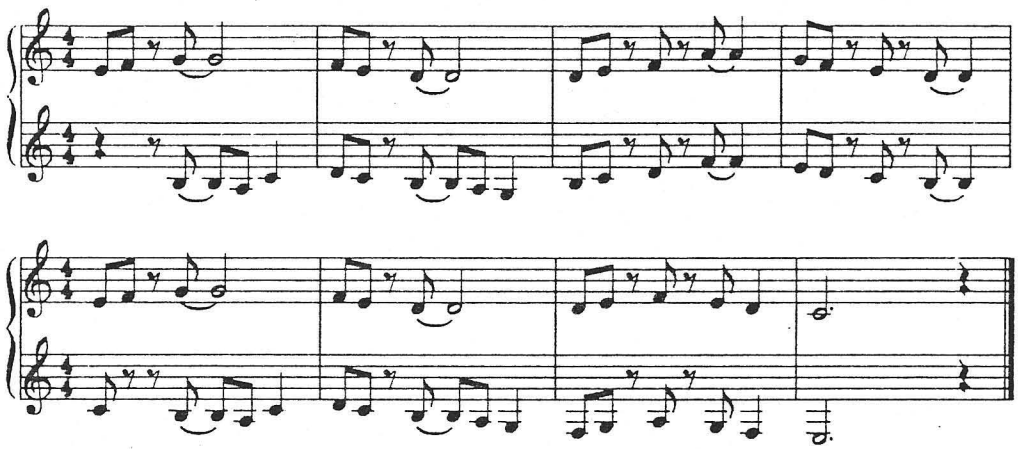

Figure 8. Original score of the first 8 measures of a guitar duet.

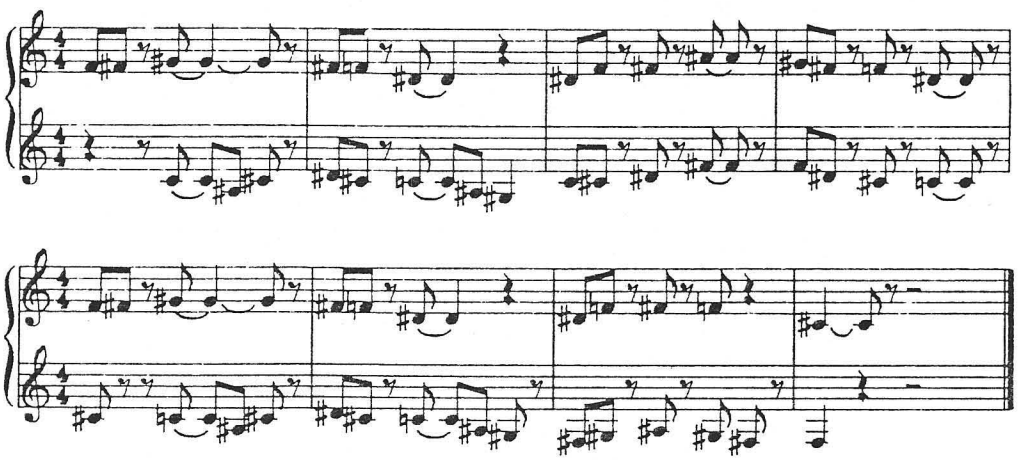

Figure 9. Final output of the program for the score in Figure 8 above. This piece was played by the author, recorded, digitized, and processed. Notice how literal-minded the computer is. The guitar was mistuned at about a half-step high. The resulting score is, naturally, in the key of C-sharp!

beat, but is printed as an eighth note tied to a quarter note if it occurs on the half-beat.

In printing pitches, we must remember that once a note has been printed with an accidental (sharp, flat, or natural, etc.), then that state sticks throughout the measure. Any other note between that time and the end of the measure with the same frequency need not have the accidental printed by it. This must be reset at the end of the measure.

We must be careful to only print so much on a line so as not to crowd the line. Although this can be edited later in the manuscripting system, it seemed reasonable to try to automate this as much as possible.

YES, BUT DOES IT WORK?

We will show here two pieces that the system was tested with. These two are a pseudo-violin duet and a guitar duet. The violins are *pseudo* because the piece was synthesized by the computer. This does not make this piece trivial for many reasons. One is that the violins were synthesized from analysis data derived from the analysis of actual violin sounds, so the notes displayed all the time-variant and transient behavior of real violins. Furthermore, the notes were quite fast, further complicating the analysis. The only thing that makes this piece easier than a natural piece is that there is no recording noise. This was extremely helpful for debugging. The guitar piece was full of noise of various kinds.

Figure 6 shows the score of the violin piece that was synthesized. Figure 7 shows the score derived by the computer. As we see, there is a note missing near the end. The durations of the notes were consistently underestimated. This is because of the noise-suppression algorithms which trim off the ends of the notes.

Figure 8 gives the score of the guitar duet that was played. Figure 9 gives the computer score. Unfortunately, the guitar was mistuned by almost a half-step, so the score reads one half-step high throughout. The intervals, however, have been faithfully copied. Again there is a note missing near the end, and the durations are consistently underestimated.

These results, on the whole, seem to be quite good. We must remember, however, that these were somewhat special pieces.

SHORTCOMINGS AND IMPROVEMENTS

At this point is it reasonable to ask which of the restrictions are fundamental and which are just convenient? As it turns out, only one restriction is fundamental. That is, there does not seem to be any way to determine whether a note has a fundamental that is coincident with a harmonic of another note. We do not know how people do this. Perhaps people use slight frequency or timing mismatches as distinguishing cues. Certainly different vibrato rates are important. We cannot use just the amplitudes of the harmonics and the knowledge of what instrument is playing, because sometimes the overlapping harmonics will constructively interfere and sometimes destructively interfere, depending on the relative phases. This is a problem that needs more study, both from a signal-processing point of view and a psychoacoustic one. The question is *when do a group of harmonics fuse into a single percept (note in our usage) and when do they remain distinct?* This is a question we cannot hope to answer here.

Other restrictions, like vibrato and multiple voices, can be dealt with simply by use of more sophisticated versions of these same techniques: bandpass filtering, harmonic grouping, and voice identification. Adaptive filters can be used to track changing pitches, vibrato rates can be matched up to infer harmonic groupings, etc. In general, there is quite a bit of improvement that can be made using just these techniques.

It is interesting that we have not required the computer to identify the instrument involved. There does not seem to be any knowledge as to how to do this. We do not know how people do this. It is often said that recognition is done on the basis of the attack transients and the steady-state harmonic amplitudes. This is easy to say but somewhat difficult to implement. The same instrument will exhibit widely varying harmonic amplitudes at different times, different loudnesses, and most certainly for different pitches. Each pitch of each instrument has a radically different set of harmonic amplitudes. In any case, as soon as harmonics overlap, the amplitude of the resultant, as mentioned above, is changed greatly. It would be an interesting research project to see if human instrument identification behavior could be simulated.

CONCLUSIONS

A set of programs to transcribe musical sound and produce a written score has been described and demonstrated. These programs will work as they stand on a class of music that is restricted fairly strongly to simple two-part pieces. The methods used are not inherently limited to this class of pieces and can be upgraded to handle much more complicated music. The only restriction that can not be easily lifted is the denial of notes whose harmonics overlap entirely. We do not know how people do this, much less how a computer might do it.

REFERENCES

Gold, Bernard. "Computer Program for Pitch Extraction." *J. Acoust. Soc. Amer.*, 34:916, 1962.

Longuet-Higgins, H.C., Steedman, M.J. "On Interpreting Bach", in Bernard Meltzer and Donald Michie, ed. *Machine Intelligence VI.* Edinburgh University Press, Edinburgh, 1971, pp. 221-241.

Miller, Neil J. "Pitch Detection by Data Reduction." *IEEE Symposium on Speech Recognition,* IEEE Catalog No. 74CH0878-9 A E, pp. 122-130.

Moorer, James A. "The Optimum Comb Method of Pitch Period Analysis of Continuous Digitized Speech." *IEEE Transactions on Acoustics, Speech, and Signal Processing,* Vol. ASSP-22, No. 5, October 1974, pp. 330-338.

Moorer, James A. *On the Segmentation and Analysis of Continuous Musical Sound by Digital Computer.* Ph.D. Thesis, Stanford University, 1975. Distributed as Dept. of Music Report No. STAN-M-3.

Noll, A. Michael. "Cepstrum Pitch Determination." *J. Acoust. Soc. Amer.*, 41(2):293, 1967.

Smith, Leland C. "Editing and Printing Music by Computer." *J ournal of Music Theory*, Fall 1973, pp. 292-308.

Sondhi, Man Mohan. "New Methods of Pitch Extractions." *IEEE Trans. on Audio and Electroacoustics*, Vol. AU-16, No. 2, June 1968.