Score following using the sung voice

Miller Puckette
Department of Music, UCSD
La Jolla, Ca. 92039-0326
msp@ucsd.edu

 A version of this paper appeared in the 1995 ICMC proceedings.

December 8, 1995

Abstract

While finished peices of music have often relied on score following using the flute, clarinet, trumpet, violin, and piano, little has been written or performed using the sung voice. Consequently, the special opportunities offered by combining the live, sung voice with state-of-the-art electronics remain largely unexplored. This paper describes the special challenges encountered when trying to use score following on the voice and some techniques that can partly overcome them.

1 Score following generalities

Computers are capable of making a much wider range of sounds than one could possibly specify in real time. One important area of research in the field of computer music is how best to map the small amount of information which a live player is capable of expressing, into the much larger space of sounds the computer can generate in response. Among the many strategies which have been proposed, a special niche is occupied by so-called score following.

Independently developed score followers were demonstrated during the 1984 ICMC by [Dannenburg 84] and [Vercoe 84], both of which focussed on the specific problem of extracting tempo from live, solo, monophonic instrumental parts. The underlying asumption was that the solo player would play the notated rhythms accurately, differing from performance to performance mostly in the choice of tempo, which was supposed to change slowly with time. Under these conditions, the computer could actually anticipate the onset of new events by the performer. The computer could thus pre-prepare events which would be played simultaneously with the live player, or provide musical events which fell between events detected from the live player, in a way that respected the player's choice of tempo.

The possibility of other deviations from metrically exact performance besides tempo was studied in [Vercoe 1985]. Musical phrasing seems to be partly communicated by systematic deviations from the exact values of the notes' written durations, which are better described as belonging to the individual note than as a global tempo change. These decisions are "learned" by the computer through repeated rehearsals; they may vary from performer to performer.

Meanwhile, [Dannenburg 86] considered the quite different problem of a soloist playing a polyphonic instrument. In general, Dannenburg's algorithms have proved more robust than Vercoe's, whereas Vercoe's are more responsive than Dannenburg's. Vercoe assumes a high level of musical skill on the part of the musician and assumes that deviations from marked rhythms are made on purpose; Dannenburg does not trust his players (or his pitch detections algorithms) to the same extent.

The use of score following in live concerts was pioneered at IRCAM [Puckette 1992]. The implicit model of the performer which underlies tempo-detecting score followers was found to break down when dealing with contemporary music as it is practiced at IRCAM. In response, a score follower was developed which has no dependence on tempo, and which makes no predictions about the future behavior of the musician to be followed. Rather than use predictions to arrange for the computer and player to act simultaneously, the effort was made to make the delay between the musician's stimulus and the computer's response imperceptibly small.

IRCAM's first score following algorithm still has an important feature in common with those of Dannenburg and Vercoe, in that it relies on a finite alphabet of tempered-scale pitches. This works perfectly for the piano and at least fairly well for the flute and clarinet; not surprisingly, these three instruments figure strongly in IRCAM's recent repertory. This assumption had to be dropped, however, in realizing Philippe Manoury's *En Echo* for soprano and computer, the first version of which was premiered in Summer 1993. That piece catalyzed the research reported here.

2 Instantaneous pitch

The voice is probably the instrument whose output least resembles a sequence of discrete tempered pitches attained at well-defined times. For every other instrument we have encountered, the first step toward score following has been to convert the instrumental performance to a sequence of detected note onsets. In the case of vocal sounds, the pitch changes rapidly and constantly. The onset of a note can have an instantaneous pitch several half-tones away from the note eventually stabilized upon. Even during the "steady-state" of a sung note (if one can be said to exist at all) vibrato can cause excursions two semitones away from the sung pitch, and occasionally even more. The problem of obtaining the pitches of sung notes therefore consists of two sub-problems: getting the instantaneous pitch (a function of time which is sometimes continuous, sometimes not) and then getting the discrete pitch, which corresponds to sung notes.

Obtaining instantaneous pitches of the human voice is a popular subject of study. The particular algorithm we have adopted is related to the one reported in [Rabiner 78], which is attributed in turn to [Noll 69]. Instead of using the Fourier spectrum as Rabiner does we will use the accelerated constant-Q transform reported in [Brown 92]; see also [Brown 93].

If the signal is denoted by $x[k]$, $k = 0, 1, 2, ...$, we define a not-quite-constant-Q spectrum,

$$S[\omega] = \left| \sum_{n=a}^{b} \frac{\exp(-\mathrm{i}\omega\mathrm{n})}{b-a} w\left(\frac{n-a}{b-a}\right) x[n] \right|, \tag{1}$$

where w is the Hanning window function defined from 0 to 1:

$$w[t] = 1/2\,(1 - cos(2\pi t))\,.$$

The sum ranges over a window ranging between sample number a and b, which both depend on ω. The sum is equal to the instantaneous amplitude of the output of a FIR bandpass filter centered about the angular frequency ω. The filter admits frequencies within $4\pi/(b-a)$ radians per sample of the center frequency ω; the 3 DB point is roughly $\pi/(b-a)$ radians per sample distant from ω on either side. The selectivity is thus,

$$Q = \frac{(b-a)\omega}{2\pi}.$$

In order to limit the response time of the filter, the window size $b-a$ is limited to a maximum value N, typically between 20 and 30 milliseconds. Subject to this constraint the window size was chosen so that the passband was a halftone wide, i.e., $Q = 17$:

$$b - a = \min(N, 34\pi/\omega).$$

The spectrum S hus reflects a tradeoff between frequency selectivity and resolution in time.

From the definition of S in Equation 1, we now define a quantity which roughly corresponds to Noll's "Harmonic Sum Spectrum:"

$$L[\omega] = p_1 S(\omega) + p_2 S(2\omega) + ... + p_8 S(8\omega), \tag{2}$$

where the p_i are positive weights. Values of L are computed for frequencies ranging from $\omega = 8\pi/N$ to the Nyquist frequency π. The lower bound is the center frequency at which the best attaiable Q is 4. For high frequencies, some or all harmonics may lie above the Nyquist frequency; their contribution is taken as 0.

The spectrum L can be thought of as estimating the liklihood of seeing a spectrum such as S if the signal x contained (among other possible summands) a signal with period $2\pi/\omega$. To call this a true liklihood function would be a grave abuse of that term; we would first have to propose an underlying model in which the signal's deviation from a periodic one were given by a known stochastic process. The sung voice's deviation from pure periodicity cannot reasonably be modelled by any tractable random process. For a clear exposition of the theory underlying Maximum Liklihood estimation see [Pitman 79].

Nonetheless, we proceed as if we were calculating a maximum liklihood estimate. Our first estimate for ω is to evaluate L for the range of values of ω under consideration, at quarter-step intervals; the frequency is simply that which attains the highest value of L. The weights p_i are found by trial and error in order to give the best output; their value differs from instrument to instrument. A good starting point is $p_i = [1, .9, .8, .7, .7, .7, .7, .7]$.

This will give us some answer or another no matter what sort of signal we analyze; we need a criterion for deciding whether the signal really has a pitch or not. To do this we invent an estimate of the signal's quality, which is the quotient of power of the signal's first eight harmonics (as measured by the appropriate values of S), divided by the signal's total power over the frequency range from 0 to 8ω. If the signal is perfectly harmonic, we would expect this quotient to be one; if it is less than 0.6 or so, we make no estimate for the frequency.

The above method only estimates frequency to the nearest quarter tone. To obtain a sharper estimate of ω we then apply a curve-fitting procedure, which was found by trial and error. If L takes on three consecutive values x, y, z, and if the peak is at y, we calculate

$$f_1 = 1 - x/y, f_2 = 1 - z/y,$$

$$c = (f_1 - f_2)/(2. * (f_1 + f_2) - 3. * f_1 * f_2).$$

The value C, which is between -1/2 and 1/2, is the correction in quartones. (It turns out that the weights p_i used to finf the peak initially are not the best weights to use here; since the accuracy of a harmonic's contribution to the fundamental is proportional to harmonic number, we recalculate L here with weights $p_i = [1, 2, ..., 8]$. If y is then not still at least as great as x and z, we give up and report a correction of 1/2 in the direction of the new peak.) In practice, the corrected result is typically accurate to within five cents.

In order to obtain discrete pitches as needed by the score following algorithm, we will also need an estimate of instantaneous signal power. A good one is given by,

$$P = \sum_{n=a}^{b} \frac{2}{b-a} w\left(\frac{n-a}{b-a}\right) |x[n]|^2,$$

where $b - a = N$.

3 Discrete pitch

We then compare the pitch and power history of the signal to try to identify discrete sung notes. We wish to do so as soon after the note's onset as possible, but without compromising the robustness of the result. In light of the deep vibratos mentioned above, we frequently cannot use a stable frequency estimate to report a note; the vibrato's fleeting moments of apparent stability will be at the endpoints of the vibrato range, not at the true pitch which lies between them.

Our discrete pitch detection algorithm reports two classes of notes, ongoing and *a posteriori*. The algorithm acts differently according to whether it is in the "on" or "off" state. Rules for detecting notes differ depending on this state. The state is changed to "on" if an ongoing note is detected, and to "off" if the pitch and envelope signals do not agree with the last reported pitch.

3.0.1 Ongoing note detection.

As a rule of thumb, vocal vibrato runs at 6 to 7 cycles per second. In order to identify the pitch center of a note with vibrato, we require that the instantaneous pitch be defined for 300 milliseconds so that at least one and preferably two cycles of vibrato are seen. To detect an ongoing note, we must be in the "off" state, and the maximum and minimum values of the instantaneous pitch must be within some maximum allowable excursion such as four half-tones. A note is then reported which is halfway between the maximum and minimum pitch excurstion. The note's reported pitch is not rounded to the nearest half-tone; we will use the exact value of the pitch in the score following stage. When a note is detected we enter the "on" state.

When in the "on" state, either of two possible conditions are regarded as being inconsistent with the note being sung and put us in the "off" state so that a new note may be reported as above. First, the instantaneous pitch may stray outside the permissible range; i.e., may stray more than half the maximum allowable excursion cited above from the note's reported pitch. This includes the possibility of the instantaneous pitch becoming undefined.

Second, the amplitude envelope may fall below a threshold, turning the note off, or it may change in such a way as to suggest that a new note has started (without necessarily having gone below any absolute threshold.) This is defined as a drop in power followed by a rapid rise, typically a factor of two increase in power over a period of 50 msec, or a factor of three rise over 100 msec, or a factor of four over 200 msec. It appears to be necessary to apply separate test for rapid, light attacks and for slower, heavier ones. When a new note onset is thus detected, we do not report a pitch; instead we enter the "off" state and disable ongoing note detection for the required 300 msec.

3.0.2 Note detection *a posteriori.*

Many sung notes never meet the stability criterion for ongoing note detection. If it appears that a note has been sung but if no note was reported using the ongoing note criterion, an attempt is made to find a note em a posteriori.

That some note has been sung is inferred from the power signal. The note's beginning is detected by the note-onset criterion (which also puts the dicrete pitch detector in the "off" state.) THe note's end is detcted either by a falling off of amplitude below the note-off threshold, or oppositely by the onset of yet another note. If either of these two occur after a note onset which was not followed by a stable note, the best pitch candidate found during the note's duration is reported. The report thereore always arrives after the end of the note, usually at the beginning of the folowing one. The best pitch is simply the instantaneous pitch corresponding to the highest instantaneous power at which an instantaneous pitch was present.

4 Score Following

We thus have two pitch signals, one which has very little delay, the other of which is reliable and discretized, but which is typically 1/3 of a second too late. We use the reliable one as input to a discrete-event score follower; this keeps us globally in place. The fast but less reliable signal is then used for triggering computer responses at the beginnings of notes.

The slow-but-reliable algorithm is based on [Puckette 92], but adapted to take into account the fact that the pitches detected do not necessarily fall on notes of the tempered scale. The earlier algorithm, in the case of a monophonic melody, would essentially accept any note that matches one of the next three pitches after the current note. In the algorithm used for the voice, whether to make a match is determined by a scoring system; if the score for going forward exceeds the score for staying put, a match is reported, otherwise not. The algorithm described here would probably benefit from vectorizing it along the lines described in [Dannenburg 84]; however, doing so might constitute an infringement of Dannenburg's patent.

Floating-point pitches and the inexactness of matches between the sung note and the scored one are dealt with by regarding a possible match differently according to how closely the matched note is hit. The match is given a value, which is a function of how closely the desired pitch matches the received pitch. A perfect match is awarded the maximum value; the value falls off linearly as a function of tuning error, with an adjustible slope; typically the slope is set so that the value hits zero when the error reaches a semitone.

The value of a possible match is set against the (negative) value of possibly skipping notes in order to get to to the note matched. Each scores note jumped over contributes a negative value, which can vary from note to note. On the other had, a negative value may be awarded to receiving a note and not matching it. This can also vary from note to note in the performance. If the value of matching a note (counting the negative value of any notes skipped in order to match it) exceeds the (negative or zero) value of not matching the note at all, the match is made and the algorithm moves forward to the new note.

Notes in the score may be weighted differently depending on their liklihood of being hit in the performance, by varying the negative value of jumping over them. This is not only useful in cases where certain notes in the score are more likely to be detected than others, but also permits the inclusion of other events such as rests, specific vowels or consonants, or other gestures which may have a higher or lower liklihood of error in detection than ordinary notes. For example, if we wish absolutely not to jump over a specific note in the score, we attach a high penalty to jumping over it to match a note with a different pitch.

The detection of rests is an example of a situation where the penalty for receiving extra notes should be set to zero. Rests are hard to distinguish from places where the performer puts spaces between the notes of a phrase, to take a breath for example. By setting the penalty to zero we avoid having the algrithm jump to a scored rest on the basis of a falsely detected one.

Whenever a note is matched using the slow algorithm, the following note in the score is awaited using the fast algorithm. The criterion for a match depends on whether the new note has the same pitch as the old one or not. If the pitch is the same, a note onset triggers it; otherwise, any instantaneous pitch within 40 cents of the desired pitch does. This match does not affect the slow algorithm's state; instead, it triggers the computer's response to the new note in advance of when it would have been triggered by the slow algorithm.

5 Practical details

Feedback is especially problematical when using the sung voice for score following. We have found that a headset microphone, set very close to the corner of the singer's mouth but out of the airstream, gives fair but not perfect isolation of the voice signal. The dynamic range of singing is much greater than for most other instruments; it can exceed 55 DB. This makes it harder to use thresholding to detect when the singer is singing a note and when not. We have often found it necessary to raise and lower the thresholds depending on the location in the score.

References

[Brown 92] Brown, J.C., and Puckette, M.S., (1992). "An Efficient Algorithm for the Calculation of a Constant Q Transform", J. Acoust. Soc. Am. 92, 2698-2701.

[Brown 93] Brown, J.C., and Puckette, M.S., (1993). "A high resolution fundamental frequency determination based on phase changes of the Fourier transform", J. Acoust. Soc. Am. 94, 662-667.

[Dannenburg 84] Dannenburg, R. 1984. "An On-line Algorithm for Real-Time Accompaniment", Proceedings, ICMC (Paris, France), P. 187.

[Dannenburg 86] Dannenburg, R., Mukaino, H. 1986. "New Techniques for Enhanced Quality of Computer Accompaniment", Proceedings, ICMC (Cologne, Germany), P. 243.

[Noll 69] Noll, A. M., 1969. "Pitch determination of human speech by the harmonic product spectrum, the harmonic sum spectrum, and a maximum liklihood estimate." Proc. Symp. Computer Proc. in Comm., pp. 779-798.

[Pitman 79] Pitman, E. J. G., 1979. *Some Basic Theory for Statistical Inference.* London: Chapman and Hill.

[Puckette 91] Puckette, M., 1991. "Combining Event and Signal Processing in the MAX Graphical Programming Environment." Computer Music Journal 15(3): pp. 68-77.

[Puckette 92] Puckette, M., and Lippe, A. C. 1992. "Score Following in Practice," Proceedings, International Computer Music Conference. San Francisco: Computer Music Association, pp. 182-185.

[Rabiner 78] Rabiner, L.R., and Schafer, R.W., 1978. *Digital Processing of Speech Signals.* Englewood Cliffs, N.J.: Prentice-Hall.

[Vercoe 84] Vercoe, B. 1984. "The Synthetic Performer in the Context of Live Musical Performance", Proceedings, ICMC (Paris, France), P. 185.

[Vercoe 85] Vercoe, B. and Puckette, M. (1985). "Synthetic Rehearsal: Training the Synthetic Performer", Proceedings, ICMC (Vancouver, Canada) pp. 275-289.