Score following using the sung voice

Miller Puckette
Department of Music, UCSD
msp@ucsd.edu

ABSTRACT: While finished pieces 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. The special opportunities offered by combining the live, sung voice with 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.

Score following generalities

Among the many control strategies which have been proposed for real-time computer music performance, a special niche is occupied by the technique of score following. Independently developed score followers were demonstrated during the 1984 ICMI by [Dannenberg 84] and [Vercoe 84], both of which focused on the specific problem of extracting tempo from live, solo, monophonic instrumental parts. The underlying assumption was that the solo player would play the annotated 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 accurately 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 deviations are "learned" by the computer through repeated rehearsals; they may vary from performer to performer.

Meanwhile, [Dannenberg 86] considered the quite different problem of a soloist playing a polyphonic instrument. In general, Dannenberg's algorithms have proved more robust than Vercoe's, whereas Vercoe's are more responsive than Dannenberg'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; Dannenberg 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 92]. 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 to play simultaneously, an 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 Dannenberg 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, three-three instruments figure strongly in IRCAM's recent repertory. This assumption had to be dropped, however, in realizing Philippe Manoury's En Ech's for soprano and computer, the first version, of which was premiered in Summer 1993. That piece catalyzed the research reported here.
Pitch

The voice is probably the instrument whose output least resembles a sequence of discrete tempered pitches at a well-defined time. 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 pitches, 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 [Rabinet 78], which is attributed in turn to [Noll 69]. Instead of using the Fourier spectrum as [Rabinet] does we used the accelerated constant-Q transform reported in [Brown 92]; see also [Brown 95].

We then examine 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 end-points of the vibrato range, not at its 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.

Vocal vibrato typically runs at 6 to 7 cycles per second. In order to identify one 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 excursion. The note's reported pitch is not weighted 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, 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, so 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 ten in power over a period of 50 msec, or a factor of three over 100 msec, or a factor of four over 200 msec. It appears to be necessary to apply separate tests 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.

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 identify the pitch of the note *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 discrete pitch detector in the "off" state). The note's end is detected 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 therefore always arrives after the end of the note, usually at the beginning of the following one. The best pitch is simply the instantaneous pitch corresponding to the highest instantaneous power at which an instantaneous pitch was present.

**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 phase. 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 [Pockette 92], but adapted to take into account the fact that the pitches detected do not necessarily fall on notes of the tempered scale. The 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 [Dussenberg 94]; however, doing so might constitute an infringement of Dussenberg'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 adjustable 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 the note matched. Each scored note jumped over contributes a negative value, which can vary from note to note. On the other hand, 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 likelihood 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 other gestures which may have a higher or lower likelihood of detection than ordinary note. 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 subsequent note.

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 unscored spaces between notes, to take a breath for example. By setting the penalty to zero we avoid having the algorithm jump to a scored rest on the basis of a falsely detected one.

[ICMC Proceedings 1997]
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.

Bibliography


