Low-dimensional parameter mapping using spectral envelopes

Miller Puckette
CRCA, $\mathrm{Cal(it)}^2$, UCSD
msp@ucsd.edu

Reprinted here from *Proceedings*, ICMC 2004.

Abstract

We explore the technique of controlling synthesis using an instrumental or other sound source. The range of spectra available from the sound source, and also that available from the synthesis technique, are estimated and the first is mapped to the second. Suitable synthesis parameters for the synthesis algorithms are found by searching a database of known output spectra. A simple experiment illustrates the technique.

1 Introduction

This paper explores a situation in which an audio stream, produced in real time by a live musician, is used as a source of timbral control over another, synthetic audio stream produced by a computer. Our goal is somehow to project the musician's 'intentionality' onto the synthetic audio output. (This might be possible even if, as seems likely, we can never extract or even understand this 'intentionality' in pure form.) It is by no means necessary (and most often not desirable or even possible) to do so by trying to recreate the time-varying pitch, loudness, and timbre of the musician's sound exactly. Rather, we wish somehow to make the output sound reflect changes in these, in a way the musician can control in order to produce an interesting stream of synthetic sound.

A related possibility is to make synthetic sounds that follow the 'shape' of an unwitting sound source, as a way of highlighting or drawing attention to the music latent in, for example, the voice of a child playing or of a politician dissembling.

A well-known example of this family of techniques is the vocoder, which, in its musical application, estimates the spectral envelope of the incoming sound and applies it as a filter on some other sound. Similarly, it is possible to use a measured spectral envelope to control the amplitudes of an additive synthesis bank. In these two examples, the pitch content of the output, roughly speaking, comes from the filter source or from the frequencies of the sinusoids, whereas variations in timbre and in loudness come from the live performer.

These two examples rely on a high-dimensional timbral estimate, and it is worth considering a situation at the opposite extreme to understand the possibilities, as well as the limitations, that emerge in a very low-dimensional situation. We'll take a somewhat familiar situation in which the synthesis method is simply to retrieve pre-stored music. This has already been explored in a musical context (Moon 2001), and the idea of retrieving sounds according to how well their spectra match a desired spectrum was demonstrated in Xiang (2002). Many ideas from Xiang's work, in which drum patterns were rearranged according to their spectra to imitate other ones, reappear in the project reported here.

Extensive work has also been done in the context of music retrieval, in which a typical challenge is to identify specific, desired, pieces of music in a huge database (Tzanetakis 2002). Here we will use a small enough database that we don't need to use advanced search techniques.

Figure 1 shows diagrammatically a situation in which we want to use a time-varying, live sound to control retrieval of a pre-recorded one. The recorded sound appears as a parametric curve in a (many-dimensional) timbre space; the span of the curve serves as the one available synthesis parameter. The musician's live input appears as another curve in the same space, this one parametrized by real time.

(Although the figure shows the recorded sound as self-intersecting, this is only an artifact of the two-dimensional figure; in a higher-dimensional timbre space the recorded sound might nearly cross itself at many points, but would not be expected to do so exactly.)

As the figure suggests, one possible approach to making the synthetic output follow the live input, assuming we have a reasonable distance measure, would be to output the segment of recorded sound closes to each new point in the 'live' curve. Before discussing the ramifications of this approach we will need to develop a set of reasonable criteria for success.

2 Criteria

Different musical situations will bring different requirements, but several criteria appear likely to recur in work of

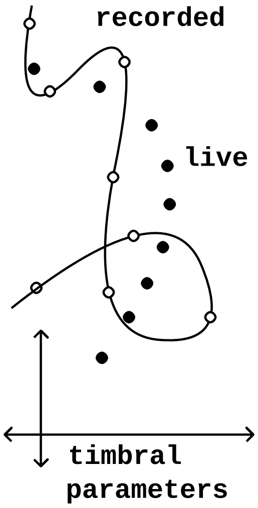

Figure 1: Stored and live sounds as curves in a timbre space.

this sort, perhaps at varying levels of importance. Some of these are:

Perceptibility. Changes in input timbre should be clearly heard as changes in output timbre. This is desirable so that gestures made by the musician will be clearly audible, in some way, in the output.

Robustness. A slight change in the input should not result in a huge change in the perceived output. For example, one should not map loudness to pitch, because the pitch would then always be wavering up and down in a way that the player could not control accurately enough.

Continuity. The synthesis parameters should not jump precipitately if the input sound is changing smoothly.

Correspondence. When a change in playing makes an audible change in the synthetic sound, the two should move in compatible directions. For example, a softer performed sound should not translate into a louder synthetic one.

Fast response. The output should depend on the current, or very recent, input, not on time averages or past gestures, so that the musician can get quickly to any desired sound output.

These criteria may sometimes be traded off against each other. For example, optimizing for ‘fast response’ will often require compromising ‘continuity’.

3 Setup of timbre space

Our working measure of timbre will be the one assumed by the `bonk~` object available in Pd and Max (Puckette 1998). The incoming sound is split into 11 frequency bands, three with center frequency 100, 300, and 500 Hz. and bandwidth 200 Hz, and eight more tuned to each half-octave above 500 Hz., so that the top one is centered at 8 kHz. In each band we estimate a loudness contribution as the fourth root of the power on the band; this is close to a loudness measure suggested in (Rossing, Moore, and Wheeler 2002).

It turns out, of course, that the measured power in these eleven bands is strongly intercorrelated. We decorrelate them in two steps. If the raw timbre vector is

$$R = [r_1, r_2, \cdots, r_{11}]\dagger$$

we first rotate the timbre vector into one component equal to $r_1 + \cdots + r_{11}$ (suitably normalized) and ten other orthogonal components to form another timbre-without-loudness vector, T, of ten dimensions. We then apply multidimensional scaling to T to give yet another timbre vector S, in ten dimensions, so that each component of S has sample mean zero and variance one, and so that the components of S are uncorrelated. To find this transformation, a representative corpus of sounds is analyzed. When analysing synthetic timbres, a systematic sampling is made as each of the usable parameters attain all the values in their domain; for input sounds the corpus is assembled intuitively.

The musician’s controlling signal and a database of possible synthetic sounds are both thus analyzed; each of the two requires its own decorrelating transformation. Associated with each synthetic sound, we also store the synthesis parameters that led to the sound so that we can re-create it later.

By normalizing the timbre vectors of both the input and the available outputs to have the same means and variances, we maximize the closeness of fit between the two; this maximizes the likelihood of finding ‘good’ output parameters. In doing this we are dropping any promise of making the output timbre imitate the input timbre exactly; they should move in roughly the same directions, but each according to its natural span.

4 Recovery

Armed with this analysis, and faced with a new input sound from the musician, we analyze the latter and find the ‘closest’ synthetic point. In the simplest realization, we simply choose the synthetic parameters which minimize the Euclidean distance between the analyzed and the resynthesized timbre. This can be done in real time using a linear search through the parameter set, provided that the number of analysis points does not exceed 10,000 or so; we could thus estimate four synthesis parameters simultaneously to 10% resolution, for example.

To get higher resolution in the synthesis parameters we can use one or another interpolation scheme. In the one-dimensional case this is easy: throw a parabola through three neighboring points in the synthesis data set and find the point of closest approach. In higher dimensions there are many possible schemes, and it is unknown which one works best.

To this we might wish to add a scheme for encouraging continuity of the synthesis parameters that come out. A simple, although not entirely adequate, approach is to favor parameter sets that are close to the previous parameter set that was output, by suitably weighting the timbral distance function (that is minimized above) by the distance travelled in the synthesis parameters from frame to frame.

Finally, the loudness of the output is adjusted to match that of the input, by scaling the output so that the 11-band loudness measures match.

An importaant limitation turns out to be closely related to the continuity problem: it could be that the live input changes in a direction which is perpendicular to the surface of synthesizable sounds. Since the input is changing audibly (we presume), the intentionality principle dictates that the output sound should change as well. The (untested) solution is to cheat, locally rotating the path of input so that *some* change can be heard in the output. At some point, though, we would have to stop rotating; perhaps this is best done at a moment when, despite our efforts to keep the synthesis parameters moving smoothly, it becomes necessary to make a discontinuous change.

No attempt has yet been made in this scheme to deal with the pitch of the sound; it is either considered a byproduct of the process (in which case the performer gives up the possibility of accurate pitch control) or else it must be chosen otherwise (either pre-determined, controlled explicitly, or derived from a separate pitch determination of the performer's input). The desired pitch may be imposed on the synthetic sound either as a post process, or else as a synthesis parameter. In the latter case, it may be desirable to map the synthesis algorithm's timbral behavior for several representative pitches. This will further complicate the problem of making a suitably continuous output.

5 Test

We have made a simple test of these ideas, in which the synthesis technique is simply a phase-vocoder-based time base correction of a forty-second sample of speech. Two such samples were used: the voice of a well-known politician, and a short vocal improvisation by Trevor Wishart. These two were each tested as controls of themselves (it worked) and then as controls of each other. A third control source, a Zeta violin, was also tested, using the politician as output.

Using Pd, each corpus described above was analyzed at 30-millisecond frames, yielding about 1300 analyses. The sample correlations between the eleven channels were measured using Octave.

A Pd extern, `searchvec`, was written to take real-time 11-channel timbre estimates from the `bonk~` object, decorrelate the 11 channels, and look the result up in the database of analyses of the target sound. The Pd phase vocoder (FFT example `10.phaselockedvoc.pd` in the Pd distribution) was used to resynthesize output. The resulting instrument can be played live from the violin, or by playing back either of the two voices.

No attempt was made to make the output continuous, so as to maximize the responsiveness of the output. As a result the resynthesis jumps frequently from one place to another in the soundfile.

Especially when one voice controlled the other, but at least somewhat when the violin was the controller, the shape of the controlling sound could be heard in the output. The timbral nature of the output remained audibly that of the resynthesis sample. Not surprisingly, no semblance of phonetic intelligibility remained in the output.

6 Acknowledgements

This project is a (much belated) continuation of a 1992-3 attempt by Trevor Wishart, Stefan Bilbao, and the author to achieve vocal control over a synthesizer—which proved too ambitious for the technology then available. Many discussions over the years have fed into today's incarnation of the idea, notably with Eric Lindemann, David Wessel, Pei Xiang, and Ted Apel.

The work reported here is in support of an upcoming performance planned for June in Buffalo, 2004.

References

Moon, B. (2001). Temporal filtering: framing sonic objects. In *Proceedings of the International Computer Music Conference*, Ann Arbor, pp. 342–345. International Computer Music Association.

Puckette, M. S. (1998). Real-time audio analysis tools for pd and msp. In *Proceedings of the International Computer Music Conference*, Ann Arbor, pp. 109–112. International Computer Music Association.

Rossing, T. D., F. R. Moore, and P. A. Wheeler (2002). *The Science of Sound* (Third ed.). San Francisco: Addison Wesley.

Tzanetakis, G. (2002). *Manipulation, analysis and retrieval systems for audio signals*. Ph. D. thesis, Princeton University.

Xiang, P. (2002). A new scheme for real-time loop music production based on granular similarity and probability control. In *Proceedings of the International Conference on Digital Audio Effects*, Hamburg, pp. 89–92. www.dafx.de.