Ssynth: a Real Time Additive Synthesizer With Flexible Control

V. Verfaille, J. Boissinot, Ph. Depalle, M. M. Wanderley
Sound Processing and Control Laboratory[*]
Schulich School of Music, McGill University
Montréal, Québec, Canada, H3A 1E3
{vincent,julien,depalle,mwanderley}@music.mcgill.ca

Abstract

This research project concerns the simulation of interpretation in live performance using digital instruments. It addresses mapping strategies between gestural controls and synthesizer parameters. It requires the design and development of a real time additive synthesizer with flexible control, allowing for morphing, interpolating and extrapolating instrumental notes from a sound parameters database. The scope of this paper is to present the synthesizer, its additive and spectral envelope control units, and the morphing they allow for.

1 Introduction

This paper presents a real time additive synthesizer with advanced and flexible control functionalities that we developed in the context of gestural control of sound synthesis. Nowadays, personal computer power and the wide range of gestural controllers permit to use costly computational synthesis techniques with gestural control devices in performance situation. This combination offers the sound quality of offline applications together with the control quality of real time applications. It however requires to consider synthesis from the control viewpoint, in terms of design and implementation.

Additive synthesis is a complex parametric method that can reach a high sound quality. Since it is based on analysis/synthesis schemes, it can synthesize perceptually identical instrumental notes. Additive synthesis however requires the control of a greater number of parameters, *e.g.* hundreds of partials' frequencies and amplitudes for instrumental sounds.

When addressing the gestural control of a sound synthesizer, the following sequence of items have to be considered: the performer controls the sound via a gestural transducer; then, he/she controls the synthesizer parameters for simulating performance. This implies to develop a specific structure of a synthesizer. It also implies that the control of synthesizer parameters reflects the articulation in time of synthesis data (*e.g.* requiring interpolation in a sound parameters database). We want to be able to generate good quality sounds from an instrumental sound parameters database, with a simple, clear and coherent control, and also to provide interpolation as well as extrapolation of musical playing of digital instruments. This is then performed by articulating in time sets of control parameters. The database is constructed from additive analysis of instrumental sounds, and the synthesis by interpolating or extrapolating in this database is then equivalent to what is usually called sound morphing. Gestural control of additive synthesis is then considered through the angle of mapping strategies, defining how many and which mapping layers exist between the gestural transducer and the additive synthesizer.

As an example of such synthesizer design, the *Escher* system was developed for studying gestural control in interpolation of digital musical instruments playing. Identically, *Ssynth* uses two mapping layers and provides fundamental frequency, intensity, and dynamics as intermediary abstract parameters (Wanderley, Schnell, and Rovan 1998) for modularity purpose in the design of digital musical instruments.

Abstract parameters are derived from the synthesizer (amplitude and frequencies of sinusoids) and from the gesture transducer, via two mapping or parameter conversion layers: from gesture data to abstract parameters and from abstract parameters to synthesis parameters. A common way to simplify the control of the additive parameters is to use the spectral envelope as a shaping curve to modulate the amplitude of partials: the provided control parameters of the additive synthesis are then the frequencies of partials (additive representation) and the spectral envelope representation parameters (substractive representation).

In the next sections, we present *Ssynth* and review low-level techniques used to control additive as well as source-filter models, allowing for morphing, interpolating and extrapolating instrumental notes.

[*] *Ssynth* is part of the project '*Simulation et Extrapolation Numériques d'Instruments de Musique en Situation de Jeu*', funded by the FQRNT (*Fond Québécois pour la Recherche sur la Nature et les Technologies*).

2 Description of *Ssynth*

The real time additive synthesizer with flexible control we developed implements 3-order phase polynomial model (McAulay and Quatieri 1986), with scalar, vectorized and recursive formulation implementations. Morphing between N notes in the database according to fundamental frequency, dynamics and instrument is provided by weighting several pitch-shifted additive frames with different fundamental frequency and dynamics of various instruments. *Ssynth* allows for interpolating and extrapolating data from the database, synthesizing polyphonic sounds, and handling OSC messages (Wright and Freed 1997) to carry control information.

The sound parameters database contains additive analysis of wind, wood and brass instrument notes (clarinet and oboe as in *Escher*, plus saxophone and trumpet) from the *McGill master samples database* (Opolko and Wapnick 1987). The additive analysis (peak extraction and tracking) as well as the fundamental frequency estimation[1] were performed using standard techniques implemented in *Additive*[2]. The large number of additive synthesis control parameters can be reduced to a smaller set of abstract parameters. Once the additive analysis is performed, frames parameters are organized as a 3-dimensional mesh as in (Haken, Tellman, and Wolfe 1998) according to pitch — 7 notes, with fundamental frequencies being as evenly spaced as possible, covering the whole pitch range, and therefore depending on the instrument —, dynamics — 3 levels: ***pp***, ***mf*** and ***ff***; related to loudness but also to spectral envelope — and instrument — clarinet, oboe, trumpet and saxophone up to now. The current dynamic parameter controls both intensity and brightness; it is however interesting to have a more precise and direct control of the spectral envelope parameters. For that reason, spectral envelope models (see section 4) were added to the database.

In order to provide a gestural control of *Ssynth*, the synthesizer structure is made of two parts. The first part is a set of *Pd* patches that implements the different mapping strategies and layers. In the case of clarinet for instance, those mapping layers convert the input data from the transducer into abstract parameters by using a set of rules that renders the acoustical coupling that exist between lip pressure, air pressure and finger in order to provide fundamental frequency, intensity and dynamics (Wanderley, Schnell, and Rovan 1998). The second part is the additive synthesizer with both abstract parameters (fundamental frequency, intensity, dynamics, instrument) and spectral envelope parameters input as well as commands to indicate which control structures are used. From the given input controls, an internal mapping layer converts the abstract parameters into additive parameters (partials frequencies and amplitudes) by interpolating/extrapolating the database. The obtained amplitudes of partials can also be modified via the spectral envelope control. As regards the practical implementation, the synthesizer is implemented in C and can be compiled as a stand alone program[3] or as a Pd object, using the Pd scheduler to have output audio.

criteria	***Escher***	***Ssynth***
model	spectral	temporal
	FFT^{-1}	3-order pol. phase
extrapolation	—	√
interpolation	pitch, loudness, dynamic, instrument	
instruments	clar., oboe	clar., oboe, sax., trumpet
directivity	√	√
polyphony	—	√
mapping	in *jMax*	in *Pd*
messages	MIDI	OSC

Table 1: *Comparison between* Ssynth *and* Escher.

Since *Ssynth* and *Escher* were developed for similar purposes, we provide in Table 1 a comparison highlighting the improvements proposed. *Escher* was controlled using MIDI and performed additive synthesis through FFT^{-1} algorithm (Rodet and Depalle 1992). *Ssynth* benefits from the OSC protocol and uses 3-order phase polynomial model to synthesize hundreds of partials in real-time. The sound parameters database is getting bigger, and morphing strategies have been refined, to allow for both interpolation and extrapolation.

3 Controlling Additive Synthesis

The gestural control of additive synthesis from sound parameters database requires to infer new sounds. In terms of fundamental frequency, intensity and dynamics, and for a given instrumental timbre, not all sounds exist in this database. Since we want to provide smooth transitions during interpolation on abstract parameters (separately or all at once) between sets of parameters. This implies to morph sounds to create intermediate configurations.

Among the various definitions of morphing (Verfaille and Depalle 2004), we consider the timbral morphing or hybridization of 2 or more sounds according to their spectral properties. It is used for singing voice (Depalle, Garcia, and Rodet 1995) as well as instrumental sounds (Haken, Tellman, and Wolfe 1998). Usually, the two parts of the sinusoid + noise model (Serra and Smith 1990) are morphed: harmonics[4] (ad-

[1] The fundamental frequency is estimated using maximum likelihood harmonic matching and Hidden Markov's Models (Doval and Rodet 1993).

[2] Additive documentation is provided online at http://recherche.ircam.fr/equipes/analyse-synthese/DOCUMENTATIONS/additive/index-e.html

[3] The code has been kept as portable as possible, controlled exclusively through OSC messages. In practice, it would be fairly easy to port to any audio plug-in platform architecture.

[4] We consider perfectly harmonic as well as quasi-harmonic partials.

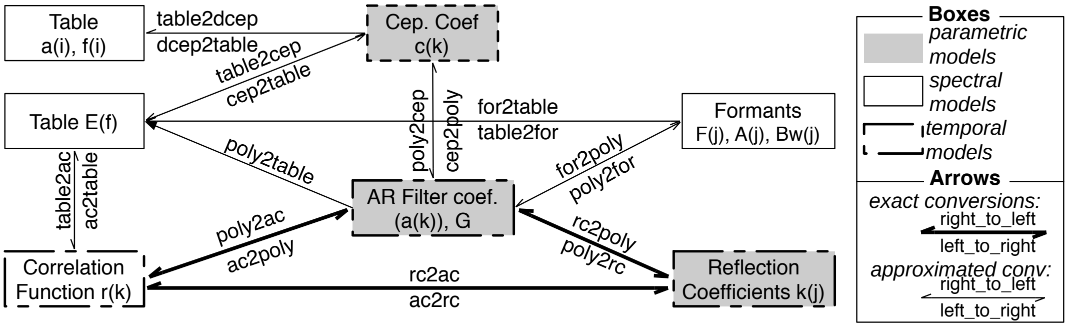

Figure 1: *Conversions between spectral envelope models. Algorithms are named using their* Matlab *counterpart's convention.*

ditive model) and residual noise (source-filter model). However, for the chosen instrumental sounds, the synthesis of harmonics only in a first step provides a reasonably good sound quality. As regards morphing strategies differ depending on wether it concerns harmonics during sustain and attack parts.

Morphing is applied to harmonics' frequencies and amplitudes and involves harmonic matching between at least two sets of spectral data (should missing harmonics be created) and harmonic weighting. Shared harmonics are matched by their rank p, the nearest integer to $f_p(k)/f_0(k)$. We generalize weighting of shared harmonics from 2 (Tellman, Haken, and Holloway 1995) up to L sets of frequency/amplitude trajectories: $(f_{l,p}(k), a_{l,p}(k))$: common harmonics (with index p) to all frequency sets, at time index k, have their amplitudes and frequencies weighted after pitch-shifting with spectral envelope preservation $(\tilde{a}_{l,p}(k))$ as:

$$\hat{f}_p(k) = \prod_{l=1}^{L} \left[f_{l,p}(k)\right]^{w_l(k)}, \quad \hat{a}_p(k) = \prod_{l=1}^{L} \left[\tilde{a}_{l,p}(k)\right]^{w_l(k)} \quad (1)$$

Ignoring unpaired harmonics can lower the computational cost, but can generate sudden harmonics birth depending on the morphing ratios $\rho_{l,p}(k) \in [0, 1]$ with $\sum_{l=1}^{L} \rho_{l,p}(k) = 1$. This unpleasant effect is avoided and better morphing with continuous transition are created by first creating the missing harmonic of unpaired harmonics (Tellman, Haken, and Holloway 1995). Its frequency and amplitude are estimated from neighbor harmonics of the same set. Then, harmonics are morphed between I instrument timbres by:

1. pitch-shifting $4 \times I$ frames (2 neighbor fundamental frequencies & 2 neighbor dynamics) with spectral envelope preservation;
2. computing mean spectral envelope by harmonics amplitude or spectral envelope weighting of the I frames;
3. intensity morphing by controlling the sound level.

The abovementioned morphing strategies apply as long as the sustain part of a note is played. The strategy is however modified under 2 circumstances. First, a proper morphing of attack shapes and durations requires to time-warp the additive data (Tellman, Haken, and Holloway 1995). Indeed, amplitudes weighting of two harmonics with different birth times would otherwise create a two-step birth amplitude curve instead of a single morphed birth time. Moreover, the partials' frequencies are not stable nor perfectly harmonic during the attack, so any amplitude gain may results in unrealistic and unpleasant gliding partials. Second, the sustain part of the morphed sound potentially being longer than the database sounds (of unequal duration), each individual note from the database has a start and an end loop times, used in order to loop this sound for as much time as needed.

4 Controlling the Spectral Envelope

The spectral envelopes are functions of frequency noted $E(f)$. In practice, the definition of an envelope is a way to sort information and depends on the context: for instance, various levels of smoothness will provide various shapes of envelope (Schwarz and Rodet 1999). When adequately computed, it simplifies the amplitude control of partials in *Ssynth*, and its modification is useful to morph sounds, at the condition that good models of spectral envelope are computed, preventing instabilities. Therefore, *Ssynth* uses various models and conversion methods between those models.

Spectral envelope models can be classified as auto-regressive filters, cepstral models, sampled representations (evenly or logarithmically spaced), geometric models (*e.g.* breakpoint functions, splines), and formantic models (Schwarz and Rodet 1999). Another classification sorts models according to their properties[5] (parametric aspect and model domain), even though it is sometimes redundant:

[5] *Ssynth* only implements AR, RC and cepstral coefficients (by cepstrum and discrete cepstrum), autocorrelation function, piece-wise linearly sampled magnitude of frequency response, and formantic models.

- parametric models: auto-regressive (AR) filter and reflection coefficients (RC), cepstrum[6];
- sinusoidal models: cepstrum (Noll 1964) and discrete cepstrum (Galas and Rodet 1990) are also sinusoidal models in the exponential domain;
- temporal models (LPC class): AR filter, autocorrelation function (AC), and RC;
- spectral models (related to the frequency response magnitude of spectral envelope) include sampled representations, formantic models, and geometric models.

Gestural control of the spectral envelope may require conversions from one model into another, more suited to provide a spectral envelope corresponding to a stable filter for a given control. Fig. 1 depicts the implemented conversions, classified as exact (for LPC class) and approximated. Indirect conversions are then derived by combination of basic conversions.

Depending on the type of parameters, conversion might be exact or only approximated. They are systematically represented in Figure 1 by bold arrows, whereas approximated conversions are in thin arrows. Exact computation conversions are possible between AR coefficients (noted '*poly*' in Fig. 1), reflection coefficients (noted '*rc*') and autocorrelation function (noted '*ac*'), and are based on Durbin-Levinson resolution of Yule-Walker equations (Kay 1988). Approximated conversions are related to a sampled model (involving frequency response magnitude of spectral envelope, noted '*table*') and/or an estimation of parameters (involving formants). The '*for2table*' conversion samples a parallel set of second-order FIR filters. Conversions between AR and cepstral coefficients use a recursive formulation (Oppenheim and Schafer 1975), usually with $p \geq 10$ auto-regressive coefficients and $q = 3p/2$ cepstral coefficients.

5 Conclusions

In the context of gestural control of additive synthesis for interpolating and extrapolating instrumental notes, we reviewed techniques used to control additive and source-filter models. We explained how interpolation/extrapolation/articulation can be viewed as morphing in those models, and more specifically conversions of spectral envelope parameters. Our contribution lies in the systematic design of the synthesis environment for allowing flexible control. This implies a potential control of the additive part by spectral envelope, parameterized in various forms. This was implemented in the *Ssynth* additive synthesizer that includes flexible control of additive and source-filter models of sound.

[6] A cepstral envelope is considered then as a parametric modeling of envelopes because it reduces the number of control coefficients.

References

Depalle, P., G. Garcia, and X. Rodet (1995). Reconstruction of a castrato voice: Farinelli's voice. In *Proc. IEEE Workshop Appl. of Digital Sig. Proc. to Audio and Acoustics, New Palz, USA*, pp. 242–5.

Doval, B. and X. Rodet (1993). Fundamental frequency estimation and tracking using maximum likelihood harmonic matching and HMM's. In *Proc. IEEE Int. Conf. Acoust., Speech, and Sig. Proc. (ICASSP'93),* Minneapolis, USA, Volume 1, pp. 221–4.

Galas, T. and X. Rodet (1990). An improved cepstral method for deconvolution of source-filter systems with discrete spectra: Application to musical sounds. In *Proc. Int. Comp. Music Conf. (ICMC'90),* Glasgow, Scotland, pp. 82–8.

Haken, L., E. Tellman, and P. Wolfe (1998, Spring). An indiscrete music keyboard. *Computer Music J. 22*(1), 30 – 48.

Kay, S. M. (1988). *Modern Spectral Estimation: Theory Application*. Prentice-Hall.

McAulay, R. J. and T. F. Quatieri (1986). Speech analysis/synthesis based on a sinusoidal representation. *IEEE Trans. Acoust., Speech, and Sig. Proc. 34*(4), 744–54.

Noll, A. M. (1964). Short-time Spectrum and "Cepstrum" Techniques for Vocal Pitch Detection. *J. Acoust. Soc. Am. 36*(2), 296–302.

Opolko, F. and J. Wapnick (1987). McGill University Master Samples. [online] http://www.music.mcgill.ca/resources/mums/html/, Montreal, Canada.

Oppenheim, A. V. and R. W. Schafer (1975). *Digital Signal Processing*. Prentice Hall, Englewood Cliffs.

Rodet, X. and P. Depalle (1992). Spectral envelopes and inverse FFT synthesis. In *93rd Conv. Audio Eng. Soc.,* San Francisco, USA, AES preprint 3393 (H-3).

Schwarz, D. and X. Rodet (1999). Spectral envelope estimation and representation for sound analysis-synthesis. In *Proc. Int. Comp. Music Conf. (ICMC'99),* Beijing, China, pp. 351–4.

Serra, X. and J. O. Smith (1990). A sound decomposition system based on a deterministic plus residual model. *J. Acoust. Soc. Am., sup. 1, 89*(1), 425–34.

Tellman, E., L. Haken, and B. Holloway (1995). Timbre morphing of sounds with unequal numbers of features. *J. Audio Eng. Soc. 43*(9), 678–89.

Verfaille, V. and P. Depalle (2004). Adaptive effects based on STFT, using a source-filter model. In *Proc. Int. Conf. on Digital Audio Effects (DAFx-04),* Naples, Italy, pp. 296–301.

Wanderley, M., N. Schnell, and J. B. Rovan (1998). Escher - modeling and performing composed instruments in real-time. In *Proc. IEEE Int. Conf. Systems, Man and Cybernetics (SMC'98),* San Diego, USA, pp. 1080–4.

Wright, M. and A. Freed (1997). Open Sound Control: A new protocol for communicating with sound synthesizers. In *Proc. Int. Comp. Music Conf. (ICMC'97),* Thessaloniki, Greece, pp. 101–4.