A SYSTEM FOR ADVANCED ADDITIVE SYNTHESIS IN MAX/MSP

Ircam Composer in Research Report 2011

Andrea Agostini
Composer
and.agos@gmail.com

ABSTRACT

Possibly the main interest composers find in sound synthesis is the ability to define sound in its subtlest details, up to the behavior of every spectral component. Such a level of precision is typically achieved by explicitly controlling a very large number of parameters, but this kind of approach has the disadvantage of being extremely unintuitive and slow.

In this document I will present some sound synthesis techniques I employed within a large-scale musical project I undertook in 2011. I will also present the graphical environment I have developed for controlling sound synthesis (mainly additive synthesis) in Max, as it is strictly related to the way I conceived the music itself, and the kind of aural and aesthetical problems I wished to face.

1. INTRODUCTION

In 2010 I received a commission from the Louvre Museum, for a new music for the 1925 film *Wunder der Schöpfung* (also known as *Our Heavenly Bodies*), directed by Hans-Walther Kornblum. The commission was for an entirely electronic composition; the duration of the film is 92 minutes, which made the composition itself a rather massive work. IRCAM would coproduce the work, in the framework of the *Electrons Libres* program, within which several other productions of this kind have been undertaken – including works by Yan Maresz, Mauro Lanza and others.

In the initial phases of the conception and composition of the score I had planned to employ a wide array of techniques for producing the actual musical sounds: several synthesis paradigms, including additive synthesis, classic subtractive synthesis, resonating filters and more; sampling; hybrid techniques, such as granulation and concatenative synthesis; and a comprehensive palette of sound tools for sound transformation and spatialization.

The subject of my musical research would be the conception of an intuitive, high-level representation of sensible musical parameters for additive synthesis. I was particularly focused on the control of micro-articulations of the synthetic sound, which I felt particularly necessary to infuse liveliness in it. In addition, I wanted my synthesis system to be able to run in real-time, so that I could really "play" the instrument and interact with it during the composition process, rather than being forced to completely formalize the parameters of every sound before being able to hear it. For these reasons, I chose to work within the real-time graphical environment Max.

2. BACKGROUND

Max offers a large number of tools that can easily be employed to set up an interactive system for simple additive synthesis. On the other hand, with this kind of tools I have found impossible to manage the spectral richness and complexity of behavior I was aiming at. This level of complexity is typically managed through text-based, non-real-time systems such as Csound. A very interesting control system for Csound is Marco Stroppa's *OMChroma* library for OpenMusic [1]. *Chroma* allows programming the Csound orchestra by taking advantage of all the advanced graphical facilities of OM.

For my previous work *Legno Sabbia Vetro Cenere* I built a custom system for additive synthesis in Max. The core of the system is a very optimized multi-threaded external able to manage a large number of oscillators with individual control of amplitude envelope and frequency. All those parameters are controlled graphically through a custom Max external, roughly behaving as the *multislider* object, but provided with some extra features such as the abilities to zoom onto a certain area of the multislider (useful when dealing with hundreds of bars, each representing a single sinusoidal component of the sound) and to yield a logarithmic representation of its values. The system is based around the concept of presets, where each preset represents a complete set of parameters for a sound. As long as certain compatibility requirements are respected, it is extremely simple to morph between different presets.

The main limitation I felt within this system was a certain lack of micro-articulation in the sound. I did not consider this to be a problem with respect to the composition of *Legno Sabbia Vetro Cenere*, where I felt this "smoothness" to be well balanced with respect to the rugosity of the string quartet. On the other hand, the large-scale form of *Wunder der Schöpfung* demanded a wider range of sonic possibilities, and the absence of physical instruments called for the ability to overcome an intrinsically "artificial" quality of sound that is often associated with sound synthesis, in order to induce in the audience a sort of suspension of disbelief towards sound.

3. RESEARCH

The research project was centered upon the development of a new synthesis environment. It is necessary now to expose its architecture and features.

The system can be seen as built by two distinct layers: the actual synthesis engine, which is essentially a Max external provisionally called *aa.polysynth~* [2], and a graphical interface to control it.

The core concept for the synthesis engine is the voice. A voice can be seen as a combination of a group of sinusoidal oscillators (one of which will be called "main", and the other "ghosts") tuned at different frequencies and a resonant filter tuned at the frequency of the main oscillator. The mix of the additive sinusoidal oscillators and resonant filter is then fed in a classic ADSR envelope generator, whose parameters are specific to the voice, and sent to one of the object's output channels (currently up to 8) in a round-robin fashion. The number of manageable voices varies greatly depending on the number of ghosts and, of course, the processing power of the computer. I have often been able to manage hundreds – sometimes thousands – of simultaneous voices on a 8-core Mac Pro.

The signal to be fed in the filtering section is global to all the voices. There is also a number of global controls: the number of ghosts per voice, the balance between the main and ghosts oscillators, the output amplitude of the additive synthesis and the output amplitude of the filtering.

This kind of architecture has been conceived after the following observations:

1. the spectral contents of natural sounds usually varies over time. The amplitude of each individual partial varies independently from the others, including the neighbouring ones. The morphology of this amplitude variation can often be modeled as the combination of a simple amplitude envelope and an amplitude modulation; the amplitude modulation itself can easily be modeled by adding sinusoids at random frequencies, within a proximity range from the main frequency;
2. in addition to the sinusoidal component, natural sounds can be seen as constituted by an inharmonic, noisy component as well. The spectral envelope of the noisy component is usually related to the spectral envelope of the sinusoidal component. This noisy component has been obtained by feeding pink noise in the bank of resonant filters. The choice of assigning the output of each filter the same amplitude envelope as the corresponding set of sinusoidal oscillator can be seen as an oversimplification, but has allowed me to reduce the already massive number of control parameters and to avoid a second bank of envelope generators – thus greatly improving the efficiency of the system. On the other hand, the resonance of each filter can be independently controlled and morphed in real-time, as well as the global output amplitudes of the oscillator and filter banks.

A Max patch built around the synthesis engine [3] provides a set of graphical facilities for controlling the synthesis parameters. As stated above, a set of *multislider*-like controllers allows to control most synthesis parameters. Each voice's parameters are represented by a set of sliders. Each set of sliders is in turn associated with an harmonic of a given fundamental frequency, whose parameters affect a single voice of the synthesis engine. Those parameters are: attack time, decay time, sustain level, release time, amplitude, deviation from the true harmonic frequency, Q factor of the resonant filter.

In addition to that, there is a set of global controls:

1. total number of voices;
2. number of ghosts for each voice;
3. maximal deviation of each ghost from the frequency of the main oscillator, expressed in half-steps: the frequency of each ghost is randomly chosen within the assigned range;
4. global transposition;
5. global pitch shift;
6. number of output channels the synthesized sound should be spread over;
7. a toggle to control the envelope generators;
8. a toggle to choose whether the phase of a silent oscillator should be zeroed when its envelope is retriggered;
9. global amplitude of the additive synthesis;
10. global amplitude of the resonant filters;
11. global spectral envelope.

At the end of the chain is a non-linear waveshaping module. The use of this module has been twofold: on one hand it has been useful for producing heavily distorted sounds, which play a major role in my work. On the other hand, when set to moderate values, it creates harmonic and subharmonic frequencies at low amplitudes, in what I perceive as a very musical way – quite similar to how physical instrument behave.

Moreover, I have found very musically satisfying the application of waveshaping distortion to filtered noise, as this enhances its spectral richness and intrinsic instability, thus directly addressing the problem of excessive "smoothness" and lack of microarticulation I found in my previous, exclusively additive synthesis engine. Although not completely deterministic, this kind of instability is controllable to a certain extent through careful setting of (and morphing between) the Q factor of each resonant filter.

It should be remarked, as a final note, that my first attempt to overcome the smoothness problem through fast moving interpolation among several vectors of amplitudes was not satisfactory, as the result quickly appeared predictable and mechanic, and reminded too closely a typical filter sweep. Nonetheless, interpolating among amplitude vectors with very slow movements has proven musically interesting, and was widely exploited in one section of the work.

4. FUTURE PERSPECTIVE

The system described here does not aim to be a full-fledged, general tool. It has been a very useful instrument for a specific project, and I have shaped it according to my specific artistic needs. My perception of the system itself is more of an actual instrument, with strong specificities and limitations, than a general-purpose synthesis environment.

That being said, my main goal in my work with synthesis is being able to produce sounds that are not imitations of existing sounds, but whose richness and complexity can match that of physical instruments and, in general, real-world sounds. With this perspective, improvements and extensions of the system can be imagined in a number of directions, including the following:

1. The amplitude envelope model is extremely crude. Although it can be tweaked by means of morphing and re-triggering, classic ADSR can be improved in a number of ways. The difficulty in this is imagining an intuitive, expressive user interface for controlling more complex envelopes.
2. The paradigm of the system is centered around the idea of single sound objects that are triggered, evolve and end. While this paradigm can be partly bypassed through accurate retriggering and morphing, it informs the very conception of the system. It would be interesting to imagine ways to model more complex transitions between sound objects, and to complexify the behavior of the objects themselves over time (see also the previous point).
3. For simplicity's sake, now the additive and subtractive subsystems are very strictly connected – they share all the synthesis parameters, except of course for Q factor. It would be interesting to experiment with new, subtler and looser relationships between the two subsystems, so to be able to model different relationships between the sinusoidal and noisy components of sound.
4. The periodic behavior of each partial's amplitude modulation could be refined, and possibly linked to the rate of an overall amplitude modulation. This could be useful to create a naturally-sounding vibrato.
5. A second bank of fixed resonant filters could be implemented, in order to emulate a resonant body. This would further improve the naturalness of vibrato.

Some of these ideas are adressed in commercial synthesis systems, although they are generally applied to the imitation of acoustic instrument rather than the creation of new sounds. For an interesting overview on these and other topics, see [3]. It should anyway be remarked that several conceptual simplifications have been introduced in the system, as it is now, in order to improve its efficiency. Each of the new possible features listed above would significantly raise the computational cost of the synthesis, which in turn would reduce the number of synthesis voices available: this may or may not be acceptable, depending on the intended use and the processing power of the machine running the synthesis system.

Finally, one last, very important step would be a thorough cleanup and documentation of all the system. This will be necessary in order to make it publicly releaseble – which is an outcome I would definitely be interested in.

5. ACKNOWLEDGEMENTS

I wish to thank Arshia Cont for the logistic and conceptual support he has given me thruoghout this whole research; Jean Bresson and Norbert Schnell for the questions they rose, challenging my initial view of the project and providing me with new problematics and points of view; Emmanuel Jourdan and Joshua Kit Clayton for their technical help with advanced programming topics and the inner workings of Max; and Xavier Rodet for his extremely useful practical suggestions and hints about how to make synthesis sound good.

.

6. REFERENCES

[1] Marco Stroppa, OMChroma. Available online: http://repmus.ircam.fr/cao/omchroma

[2] See repository

[3] See repository

[4] Lindemann Eric, Music Synthesis with Reconstructive Phrase Modeling, in *IEEE Signal Processing Magazine,* March 2007, pp. 80-91.