Spectral Granular Synthesis

Stefano Fasciani
Faculty of Engineering and Information Sciences
University of Wollongong in Dubai
stefano@fasciani.xyz

ABSTRACT

This article introduces a granular synthesis algorithm in which grains of sound are processed in the frequency domain and combined at spectral level. In particular, each grain contributes to the sound synthesis with its magnitude spectrum only. Phase is reconstructed using spectrogram inversion techniques which support real-time computation. With this method, windowing and overlapping grains of regular size is no longer required. The algorithm presents a large number of parameters available for the dynamic manipulation of the synthetic timbre. Algorithm implementations are detailed in the paper, including comparison with traditional time-domain against the proposed frequency-domain granulation strategies.

1. INTRODUCTION

1.1 Granular Synthesis

Granular synthesis techniques are based on 1946 Gabor's research [1], which described complex sounds as composed by simple and small acoustic particles or grains. Gabor's aim was to compress audio minimizing bandwidth requirements in telecommunication. A few decades later, Xenakis [2] imported this approach to the musical context, rearranging and pasting together small slices of magnetic tape. After, Roads extensively experimented granular synthesis on computers [3]. In the late 80's, supported by advances in computing technologies, Truax implemented real-time granular synthesis [4], [5]. Thenceforth, granular synthesis availability and popularity grew significantly among musicians and composers.

In sound synthesis, the generated timbre depends on a set of user-defined parameter. In most implementations, these parameters can be manipulated at runtime to produce dynamic variations of the timbre. Common parameters in granular synthesis implementations are the size of audio grains, the amplitude envelope, the grain density, and the grain sequencing mode. The method of generating grains or extracting these from selected sources also has a significant impact on the sound, however. Roads classifies granular synthesis methods according to five different grain organization approaches: Fourier and wavelet grids; pitch-synchronous overlapping streams; quasi-synchronous streams; asynchronous clouds; and time-granulated or sampled-sound streams with overlapping quasi-synchronous or asynchronous playback [6]. The latter category is the most popular among the various granular synthesis implementations proposed to date[1], in which grains are extracted using different schemes from one or more audio files [7].

Besides extracting and recombining grains, granular synthesis implementations often offer the possibility to modify the sonic contents of the grains, such as shifting the pitch. Based on the same principle, audio effects such as granular delay have also been developed, where grains are extracted from a live audio stream. Variations of granular synthesis that allow users to define features of the output timbre, include an offline stage analyzing grains extracted from a corpus of sound [8]–[10].

1.2 Constraints of Granular Synthesis

Granular synthesis methods performing granulation in the time domain are required to address the unpredictability of the phase at the beginning and at the end of the grains. In general, grain selection and extraction do not include criteria ensuring that initial and final phases are zero, or that the initial phase of a grain matches the final phase of the previous grain for all frequency components. Introducing such a requirement would significantly complicate the grain retrieval process, and it may constrain to work with grains of variable size. With non-matching phases, the generated sound presents amplitude discontinuities determining audible clicks. The common approach to address this issue is the use of an amplitude envelope smoothing grains' edges. An exception is grainlet synthesis, which combines granular and wavelet synthesis ideas [11]. Grainlet synthesis presents a stricter definition and construction, ensuring zero phase at the edges of the grains.

Shape and width of the amplitude envelope is usually one of the parameters available to users. The application of the envelope is equivalent to windowing a signal in the time-domain, resulting in spectral leakage due to the convolution of window and signal in the frequency domain. Window shape selection is a tradeoff between width of the main lobe and height of the side lobe of the frequency response of the window function. The amplitude envelope blurs the spectral contents of the signal, and sonic sub-components at the edges of the grains are greatly attenuated. Moreover, smooth amplitude envelopes must be overlapped in consecutive grains to avoid audible amplitude modulations. The simple sequencing of smoothed

[1] https://granularsynthesis.com/software.php

grains determines audible gaps, or noticeable amplitude modulation, in the generated sound.

1.3 Spectral Modeling Synthesis

Spectral models, suitable to represent human auditory perception, have been used for sound synthesis purposes. A detailed survey on these techniques is provided by Smith in [12]. These methods aim at synthesizing the short-time Fourier transform of sound, which is then inverted using overlap-and-add technique. Following Fourier's theorem, the sinusoidal modeling approach, equivalent to additive synthesis, is sufficiently generic to model periodic sounds. To overcome limitations with noisy and transient components, this approach has been extended to the sines+ noise+ transient modeling [13], suitable to generate most types of sound.

Inspired by 1949 'Pattern Playback' by Franklin S. Cooper, and by 1958 'ANS Synthesiser' by Yevgeny Murzin, there are synthesis techniques that use images and inverse short-time Fourier transform to generate sound. In particular, vertical column of pixels are mapped to the magnitude of frequency bins. Since phase information is missing, this synthesis does not generate crisp or sharp transient sounds. Spectral techniques have also been used in conjunction with granular synthesis to morph the spectral contents of grains [14].

2. SOUND SYNTHESIS FROM SPECTRAL GRAINS

The synthesis method we proposed here is based on the simple idea of sequencing spectrums, from which we reconstruct grains that are then combined to generate the sound signal. There are two key difference with the time-domain granulation. First, the actual size of the grain that we combine in the time domain, can be derived from a segment of the sound source with arbitrary size. Indeed, with an intermediate step in the frequency domain, we can interpolate or decimate the frequency bins to reduce or increase their number, but preserving the overall spectral shape. Second, when computing the Fourier transform of segments from the audio source, we preserve only the magnitude spectrum but not the phase. Phase is reconstructed aiming to provide continuity between consecutive grains, which may contain significantly different sonic data. If phase reconstruction provides a perfect match, using edge-smoothing amplitude envelopes and grain overlap is not necessary anymore, although possible to increase synthesis density. This approach, illustrated in Figure 1, includes the following steps: extraction of grains with arbitrary size from the source signal; computation of Fourier transform; computation of the spectrum of the required size (accordingly to the output grain size) and phase reconstruction (block SP-PR); computation of inverse Fourier transform; sequencing of grains to generate the output signal. In the figure, we omitted the application of amplitude envelope before the FFT and after the IFFT. In the figure, output grains are not overlapped although this is still possible in order to increase the grain density or to cope with the smoothed edges if amplitude envelope is used.

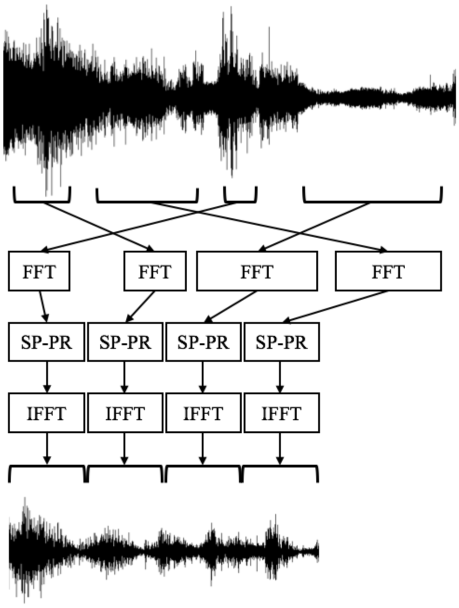

Figure 1. Overview of granular spectral synthesis, with the block SP-PR computing the spectrum with a size matching the one of the output grain and reconstructing the phase. In the figure, application of amplitude envelope prior FFT and after IFFT is omitted. Output grain density can be increased by overlapping.

Compared to time-domain grain sequencing, the intermediate step in the frequency domains enables further manipulation of grains' sonic content, and allows to expose additional synthesis options to users. These can be dynamically varied to further develop timbral complexity. This synthesis approach shares similarities with the phase vocoder [15], [16]. However, the phase vocoder performs interpolation across spectrums of the same size to change the timescale (i.e. time-stretch) of an audio signal. Instead, in the proposed synthesis, spectra related to segments of the source file are individually modified (resampled) to match a fixed size. Also, there are differences in the phase reconstruction technique as detailed in Section 2.2.

2.1 Grain Size

Generally, in both synchronous and asynchronous granular synthesis, the size of the grains is fixed or controlled by users, who may change this value at a rate significantly lower compared to the grain sequencing process. In our synthesis algorithm, sequenced grains have user-defined fixed size N samples, but these are derived from grains of arbitrary size from the sound source. We draw the grain size from a Gaussian distribution. Users can set the mean M and variance S of this distribution according to their preference. With zero variance, the size is fixed to M samples. We compute the N-point DFT using the Cooley–Tukey FFT algorithm. When N is smaller than M, we perform upsampling of the frequency bins (or zero padding in the time domain), while when N is larger than M we perform decimation. The resulting M point spectrum is then used to reconstruct the phase, and finally the grain to be sequenced is computed using a M-point IDFT. The idea behind this computational step is to preserve the overall spectral shape of the extracted grain while mapping it onto a lower or higher number of frequency bins. As interpolating in the frequency domain is equivalent to zero-padding in the time domain, a significant upsampling of the spectrum will introduce zero samples at the center of the grain resulting in audible amplitude modulations. This can be considered as a side-effect that can be constrained with appropriate synthesis parameters, or an additional control dimension to generate richer textures. The process of decimating (or upsampling) the frequency bins and discarding the phase does not produce significant pitch fluctuations, but it contributes in blurring the timing of the frequency components in the original audio source.

2.2 Spectrogram Inversion

After computing the spectrum of each grain, we need to reconstruct the phase, required to compute the *M* grain samples via IDFT. This step is equivalent to reconstructing a signal from a spectrogram (or sonogram), where phase information is not available. This problem has been extensively addressed in the literature for more than three decades [17], [18]. A survey on key techniques is available in [19]. In this context, suitable methods are those supporting online computation, enabling real-time sound synthesis, i.e., phase can be reconstructed using information from previous grains only. Other selection criteria are related to the computational load, which should be deterministic. The Single Pass Spectrogram Inversion (SPSI) [20] and the Real-Time spectrogram inversion using Phase Gradient Heap Integration (RTPGHI) [21] are suitable techniques in this context. The first is less computationally expensive and it works similarly to phase vocoders. It includes a phase-locking mechanism at peak frequency bins, but with phase rate at the peak being estimated by interpolation of the magnitude spectrum. The phases of non-peak bins are locked to those of the peaks. Besides the higher computational cost, RTPGHI provides a synthesis output sound with sharper transient sounds. After the phase is estimated and IDFT performed, amplitude envelope has to be reapplied before the overlap-and-add stage. As the length of source segments can vary, we compute the window function at each iteration, using different lengths but consistent shape.

3. IMPLEMENTATION

We developed two proof-of-concept versions of the spectral granular synthesis. The first supports only identical *N* and *M* values, i.e. there is no spectrum upsampling or decimation. The second version supports different values of *N* and *M*. The first version computes time-domain granulation using identical settings and grains for comparison purpose. Time-domain granulation with variable grain size is not univocally defined and requires additional assumptions which may bias the comparison. Both versions have been implemented in MATLAB. Synthesis computation is offline, but the algorithm is real-time ready, i.e. it computes a buffer of output samples per iteration using past data only. The computational complexity depends on the synthesis settings, especially grain density and size. Using 8192 samples per grain, 95% overlap, and the largest grain size deviation, we were able to synthesize 10 seconds of audio at 48 kHz in less than 3 second of MATLAB time. Benchmarking was performed on a quad-core Intel i7 with clock speed of 2.4 GHz, 6 MB of L3 cache, 256 KB of L2 cache per core and a front bus speed of 1333 MHz. However, when the overlap gets close to 100%, the required number FFT per second does not permit real-time computation. A simplified version of the spectral granular synthesis has also been implemented for the browser using the Web Audio API and JavaScript. The current browser implementation presents reduced features and less parameter flexibility due to major limitations in FFT packages for JavaScript. Both version and implementations are available as open-source software[2].

3.1 Additional Features

In current implementation, we included three modes to extract grains: random, fixed position with random component, and sequential with random component. In random mode, to determine the grain position, we use a uniformly distributed random variable spanning the entire range of the source file. In the second mode, we use central position plus a random component drawn from a Gaussian distribution with user-defined variance. Users can force the distribution to be unilateral, drawing random position only before or after the fixed position. Finally, in sequential mode grains are drawn at position iteratively incremented the size of the grain by a user-defined rate. The system supports any fractional rate including negative values (i.e., moving backward). Position can be randomly varied using a unilateral or bilateral Gaussian random component with user defined variance.

Before reconstructing the phase and computing the IDFT, we allow user to modify the spectrum by applying a circular shift by a number of bins drawn from a Gaussian random variable with user-defined mean and variance. Amplitude envelopes for the grains are generated using the Kaiser window, which present a parameter allowing the adjustment of the edge smoothing. When the parameter is zero the resulting window is rectangular.

3.2 Synthesis Parameters

The current implementation of the synthesis algorithm can be controlled using the following parameters. *Sampling Frequency*; *Audio Source File*, resampled to match the system sampling frequency, and converted to mono; *Grain Size* in samples, representing the size of audio frames extracted from the audio file; *Spectrum Size*, indicating the number of bins used for the spectrum of grains after resizing, hence half of the grain size after IDFT prior to overlap-and-add; *Grain Size Deviation*, denoting the standard deviation of the Gaussian distribution used to vary the *Grain Size* from its mean; grain *Extraction Mode*, as detailed in the previous subsection, including *Central Position*, *Standard Deviation*, *Laterality* of the random component, and *Rate* for the sequential mode;

[2] http://www.stefanofasciani.com/sgs.html

Kaiser Beta, which is the parameter defining the shape of the Kaiser window we use as amplitude envelope; *Spectral Shift Mean* and *Standard Deviation* for the circular shift of the spectrum bins; *Phase Reconstruction Mode* to select between SPSI and RTPGHI. In the current implementation, we sequence grains at regular intervals only, such as in pitch synchronous granular synthesis. When dynamically varying the size of extracted grains (i.e., *Grain Size Deviation* different than zero), we obtain clearly audible irregularities in the synthesis output similar to asynchronous granulation.

4. RESULTS AND FUTURE WORK

Figure 2 and Figure 3 include two comparisons between time-domain granulation (top spectrogram) and the proposed spectral granular synthesis (bottoms spectrogram). In both examples, we used a voice sample as source file. Identical sequences of randomly extracted grains were used for in both type of synthesis. In the example of Figure 2, we use grains with a size of 43 ms, whereas the size for Figure 3 is 11ms. To emphasize the differences between the two methods we used no amplitude envelope and no overlap between consecutive grains. In order to allow fair comparison, for the spectral granular synthesis we set *M* equal to *N*. Files associated with the figures and additional synthesis examples are available online[2].

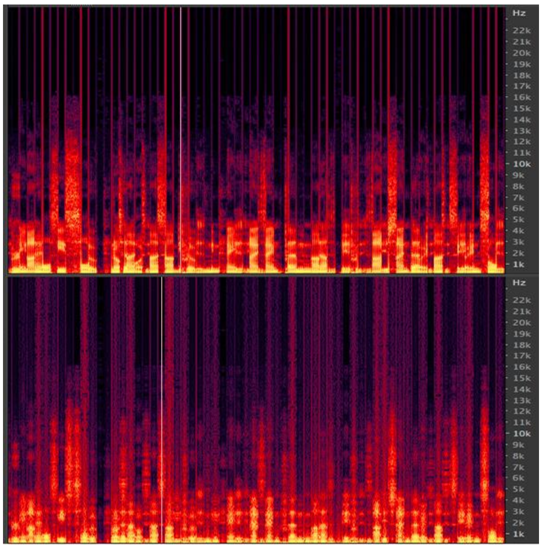

Figure 2. Comparison between time-domain granulation synthesis (top) and proposed spectral granular synthesis (bottom), using identical grain sequence from voice file, with size of 43 ms, no amplitude envelope and no overlap between consecutive grains.

In the time-domain granulation of Figure 2, audible clicks due to absence of edge-smoothing amplitude envelope and no grain overlap are evident (periodic vertical lines). These are significantly reduced in the spectral granular synthesis, but contents of the grains appear to be slightly blurred and transient smearing is audible. When reducing grain size, as in the example of Figure 3, overall spectral artifacts are reduced while discontinuities are still minimized. These effects can be further reduced just by overlapping the output grain by 10 to 20%.

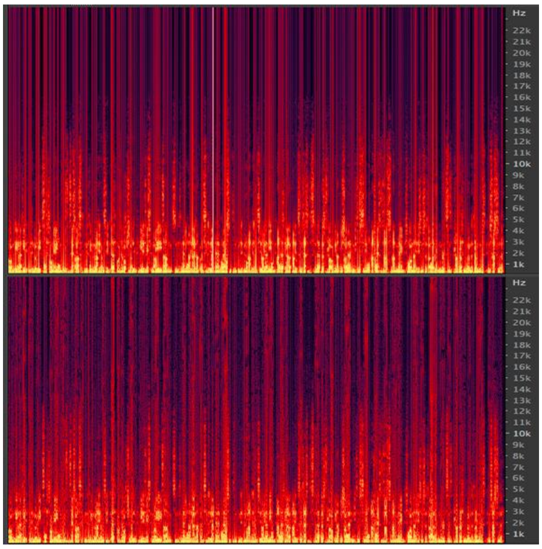

Figure 3. Same as Figure 2, but using grains of 11 ms.

As described in Section 3, the computational load of the synthesis algorithm enables real-time computation on modern computers, but is much higher compared to other granular synthesis techniques. Moreover, in the current implementation, reduction of the spectrum size is implemented with a sub-optimal integer decimation. Using resampling with rational factor is too demanding for real-time computation, especially with large grain size variance. As the grain size varies amplitude envelope must be recomputed all time. These issues remain to be addressed using alternative techniques, or finding optimal tradeoff between degraded synthesis features and computational complexity. In future work, additional functionalities will be integrated, including asynchronous granulation and enhanced possibilities to manipulate the grains, such as filtering or combining spectra from different grains. Technical improvement of browser implementation for will be carried out to remove current functional limitations. This will enable deployment of the proposed synthesis technique for qualitative evaluation.

5. REFERENCES

[1] D. Gabor, “Theory of communication. Part 1: The analysis of information,” *Journal of the Institution of Electrical Engineers-Part III: Radio and Communication Engineering*, vol. 93, no. 26, pp. 429–441, 1946.

[2] I. Xenakis, *Formalized Music: Thought and Mathematics in Composition*. Indiana University Press, 1971.

[3] C. Roads, “Introduction to Granular Synthesis,” *Computer Music Journal*, vol. 12, no. 2, pp. 11–13, 1988.

[4] B. Truax, “Real-Time Granular Synthesis with a Digital Signal Processor,” *Computer Music Journal*, vol. 12, no. 2, pp. 14–26, 1988.

[5] B. Truax, “Composing with Real-Time Granular Sound,” *Perspectives of New Music*, vol. 28, no. 2, pp. 120–134, 1990.

[6] C. Roads, *The Computer Music Tutorial*. MIT Press, 1996.

[7] Ø. Brandtsegg, S. Saue, and T. Johansen, “Particle synthesis–a unified model for granular synthesis,” in *Linux Audio Conference*, 2011.

[8] D. Schwarz, “A system for data-driven concatenative sound synthesis,” in *Proc. of Digital Audio Effects (DAFx)*, Verona, Italy, 2000.

[9] D. Schwarz, “Concatenative sound synthesis: the early years,” *Journal of New Music Research*, vol. 35, no. 1, pp. 3–22, 2006.

[10] J.-S. Lee, F. Thibault, P. Depalle, and G. P. Scavone, “Granular Analysis/Synthesis for Simple and Robust Transformations of Complex Sounds,” in *Proc of 49th Int. AES Conf.: Audio for Games*, 2013.

[11] R. Kronland-Martinet, “The Wavelet Transform for Analysis, Synthesis, and Processing of Speech and Music Sounds,” *Computer Music Journal*, vol. 12, no. 4, pp. 11–20, 1988.

[12] J. O. Smith III, *Spectral audio signal processing*. W3K publishing, 2011.

[13] S. N. Levine and J. O. SMITH III, “A compact and malleable sines+ transients+ noise model for sound,” in *Analysis, Synthesis, and Perception of Musical Sounds*, Springer, 2007, pp. 145–174.

[14] S. Siddiq, “Morphing of granular sounds,” *Proc. of Digital Audio Effects (DAFx), Norway*, 2015.

[15] J. L. Flanagan and R. Golden, “Phase vocoder,” *Bell Labs Technical Journal*, vol. 45, no. 9, pp. 1493–1509, 1966.

[16] J. Laroche and M. Dolson, “New phase-vocoder techniques for pitch-shifting, harmonizing and other exotic effects,” in *Applications of Signal Processing to Audio and Acoustics, 1999 IEEE Workshop on*, 1999, pp. 91–94.

[17] S. Nawab, T. Quatieri, and J. Lim, “Signal reconstruction from short-time Fourier transform magnitude,” *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 31, no. 4, pp. 986–998, Aug. 1983.

[18] D. Griffin and J. Lim, “Signal estimation from modified short-time Fourier transform,” *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 32, no. 2, pp. 236–243, Apr. 1984.

[19] N. Sturmel and L. Daudet, “Signal reconstruction from STFT magnitude: A state of the art,” in *Proc of Digital Audio Effects (DAFx)*, 2011, pp. 375–386.

[20] G. T. Beauregard, M. Harish, and L. Wyse, “Single Pass Spectrogram Inversion,” in *2015 IEEE International Conference on Digital Signal Processing (DSP)*, 2015, pp. 427–431.

[21] Z. Pruša and P. L. Søndergaard, “Real-Time Spectrogram Inversion Using Phase Gradient Heap Integration,” in *Proc. of Digital Audio Effects (DAFx-16)*, 2016.