GMU, A FLEXIBLE GRANULAR SYNTHESIS ENVIRONMENT IN MAX/MSP

Charles Bascou and Laurent Pottier

GMEM
Centre National de Creation Musicale
15, rue de Cassis
13008 MARSEILLE FRANCE
www.gmem.org
charles.bascou@free.fr dvlpt@gmem.org

ABSTRACT

Granular Synthesis has found to be a particularly rich framework to synthesize varieties of different sound colours and textures. Our goal at GMEM is to design and develop a flexible environment, called "GMU" (for GMEM Microsound Universe), dedicated to this synthesis method including analysis, transformations and synthesis algorithm. The proposal of intuitive control strategies is also an important part of the project. This paper will focus on the synthesis tool written for the Max/MSP Graphical Programming Environment and will present several applications.

1. INTRODUCTION

In 1946, Dennis Gabor introduced the idea that any sound could be decomposed into a set of simple acoustical events called quantum [1]. This granular model of sound was perceptually validated according to the limitations of the auditory system dealing with time-frequency discrimination. Each of these acoustical events corresponds to a local Time-Frequency component of the sound and thus can represent a large variety of signal structures from transients to pitched sustained parts.

This theory of signal and perception introduced a new approach to synthesize sounds called Granular Synthesis. The main principle of this technique is the accumulation of a large amount of basic parametric sonic events called grains. Iannis Xenakis was one of the first music composers who used the grain as the basic symbolic component of some of his pieces (mostly instrumental) and thus broke the wall between micro and macro musical structure. Since the 1970s, Curtis Roads has explored many aspects and applications of this synthesis technique from real-time pitch shifting of sounds to complex textures generation [2]. He particularly studied the perception effects of the different synthesis parameters and proposed an exhaustive categorization of the diverse applications according to the constraints/relations applied to these parameters.

At GMEM, we want to create a flexible environment dedicated to the granular representation of sounds. It includes analysis algorithms performing granular decomposition of natural sounds[4], treatments and transformations of analysis data, and off-line/real-time granular synthesis tools. In other words, we would like to design a granular analysis/synthesis tool.

This idea comes with the application of such techniques to natural noisy sounds in mind. It points to sounds that can be defined as the accumulation of more or less complex sonic grains, with their proper temporal and spectral variability. For example, the sound of rice falling onto a metal plate is composed of thousands of elementary "ticks"; the rain produces, in the same way, the accumulation of a large amount of water droplet microsounds...

From here comes our need of a flexible synthesis environment which would supports generation of high density complex grain streams with a precise control of their parameters. Thus, we have developed a set of real-time Max/MSP DSP objects we will present in the next sections.

2. BUFFER BASED ENHANCED GRAIN GENERATOR

Our first need was to have a flexible granular synthesis engine which would be efficient and highly configurable. Max/MSP provides ways to make synthesis system in abstractions directly in the Max graphical langage. But for our application, *eg.* an high polyphony and accurate grain generator, we need an efficient and optimized synthesis engine which can only be done by programming an external in C langage. Moreover, the current objects available in Max/MSP have found to be too much specialized in a specific application of this wide synthesis scheme. We thus developed a granular synthesis MSP object called *bufGranul*$\sim$ performing multi buffer granulation (*ie.* recorded sound granulation) with buffer based envelope.

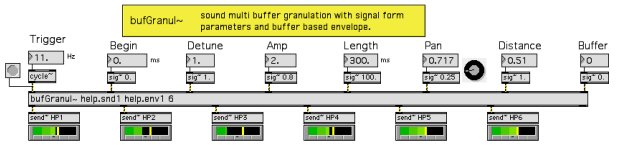

Figure 1. *Parameters of the MSP bufGranul∼ object.*

2.1. Parameters

As seen in figure 1, this object provides the commonly used granular synthesis parameters which are :

- *begin* : time position in milliseconds where the grain waveform is picked in the sound buffer.
- *detune* : transposition factor (*ie.* read speed) of the grain waveform. For example, 1. corresponds to original speed, 0.5 to lower octave, 2. to upper octave. Note that if the speed is negative the buffer waveform is read backward.
- *amp* : linear amplitude of grains.
- *length* : length in milliseconds of the grains. Note that when length is negative, the envelope is read backward.
- *pan-dist* : spatialization parameters of the grains.
- *buffer* : sound buffer number where grain waveforms are picked from.

2.2. Buffer based ...

The work with sound and envelope buffers have found to be the most general framework for granular synthesis. It allows granulation of the most varied types of sound materials. Indeed, sound buffers can be filled with synthetic waveforms like sinusoids allowing synthesizing elementary grains in the Gabor sense. It also gives the user the ability to generate complex grains streams from recorded natural sounds like environmental sounds and to apply transformations like pitch-shift/time-stretch on voices or instruments. We will see later that buffer based granulation can also be adapted for real-time granulation of live inputs.
Amplitude envelope of each grains are also defined as buffers. This specification has multiple advantages. The first is related to efficiency of the synthesis. Indeed, common grain envelopes like gaussian envelope use complex mathematics operations which are very cpu time consuming. Precalculating these envelopes in buffers and then reading them at synthesis time gives a great efficiency improvement with no audible artefacts (with a 512 samples or higher envelope in the buffer). The second advantage is the ability for the user to define arbitrary shape envelopes. They can be generated from complex formulas, drawn by hand or with breakpoint function editors extending the sound possibilities of this synthesis.

In order to enhance the generation of complex textures, the object can refer to multiple buffers for sound and envelope. You can then choose for each grains in which sound buffer to pick waveform and which envelope buffer to use. This idea has been primarily motivated by the possible generation of complex textures based on multiples sound files. In this application, each sound was an isolated natural acoustic grain (little rocks knocking against each others, wood breaking microsounds, ...) randomly distributed to recreate and extend natural acoustic sound textures.

2.3. Trigger by signal zero crossing

In order to keep the *bufGranul∼* object the most generic, no grain triggering algorithm is included in it, allowing the user to define his own according to his needs. Indeed, the different applications of the granular synthesis often require their own triggering methods and parameters. The Pitch Synchronous Granular Synthesis (PSGS) [2] trigs grain periodically with a control of the inter-grain time, while complex texture generation requires, requires for trigger frequency most advanced controls like random generators based on statistcal laws. In that purpose grain triggering by signal zero crossing (first inlet of the object) allows a wide variety of external control possibilities. The figure 2 show two very simple examples of triggering methods.

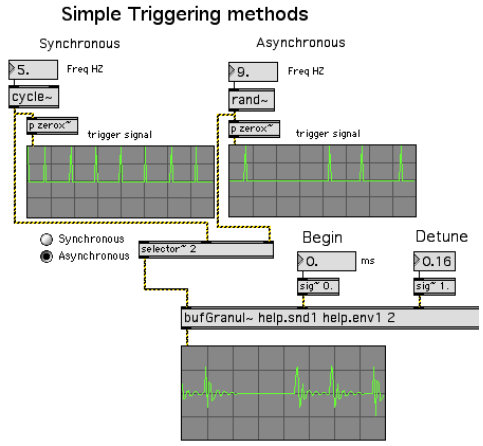

Figure 2. *Examples of grain triggering methods.*

As we will see next, the advantage of working with signal for triggering is that you can generate very high density grain streams (up to *samplerate* / 2 grains per seconds) with a high time accuracy.

2.4. Signal form parameters

Indeed, the generation of complex grain stream implies high frequency fluctuations of the grain parameters. The message manager in Max is not well adapted for very high frequency message flux ($>$ 1000 Hz) introducing timing irregularities. It results in grain streams highly correlated with the max message scheduler clock. To bypass this

artefact, we implemented signal form grain parameters which provides many advantages for synthesis reliability.

As we saw before, the grains are triggered at the signal rate when the trigger signal crosses zero. The parameter signals are sampled at this exact time to generate the grain. You can though have a deterministic control of the parameters of each generated grains. As an example, it allows to trig a grain with its own specific parameters into a high density grain stream. This functionality is illustrated in figure 3. This Max patch shows the accuracy of the work with signal parameters. The standard parameter signals (inlets) are "replaced" by the desired ones by a kind of "impulse routing". The impulse is sent to the trigger inlet of the *bufGranul∼* to generate the desired grain.

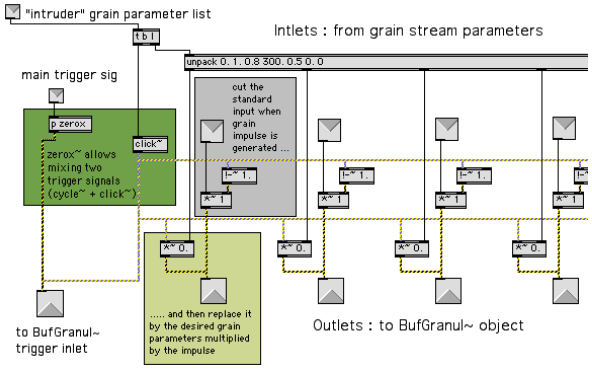

Figure 3. *Parameter filter to insert a specific grain in a grain stream.*

Another main advantage to work with signal parameters is the possible implementation of versatile parameter dependency system. As shown in [2], dependency between parameters is an important specification for some specialized granular synthesis like the wavelet synthesis. Message form parameter can produce undesired side effect especially with high trigger frequency grain stream. The signal parameter ensure that two different parameters are sampled at the same time to generate the grain and thus allows linking two or more parameters, with more or less complex mapping, obtaining a deterministic result.

2.5. Spatialization

The *BufGranul∼* object also provides an internal spatialization system. Each grain can have his own position in the virtual space. It allows to integrate the space into the sound texture creation process. It functions on the same scheme as the HoloSpat system [6] developed at GMEM for sound spatialization on multiple speaker system. You can specify the number of output (up to 8) of the object depending on your multi-speaker system. Position of each grain is controlled by the *pan* (angle) and *dist* (distance) parameters. It can be directly connected to the HoloEdit [5] sound trajectory editor to create complex spatial behaviors.

2.6. Computation Load

The computation load of the *bufGranul∼* object is directly related to the number of active grains playing together. This is dependent on trigger frequency and grain length. A few CPU load samples are represented in table 1. It has been done on a PowerBook 1,3 Ghz G4 Macintosh with a 44.1 kHz signal sampling rate.

Active voices	CPU load %
4	3%
50	15%
100	30%
300	75%

Table 1. *bufGranul∼ computation load samples.*

These results were independents of the ratio between trigger frequency and grain length. Indeed, high trigger frequency ($>$ 10000 Hz) with short grains is as computationally expensive as low trigger frequency with long grains, with the same overlap factor (active voices number). To save RAM memory , the polyphony of the object is hardwired limited to 512 voices.

3. ENHANCED RANDOM GENERATORS

The possibility to randomize grain parameters is an important specification of the Granular synthesis. Indeed, it allows to create complex sound textures such as granular spectral clouds [2]. We found the need to extend the currently available random generators in Max/MSP, which only support equiprobable values (object *random* or *noise∼*), to custom probability function generator.

3.1. Algorithm

The idea is to define probability functions as lists of values, each value corresponding to the probability of the choice of its index. Let $\{p_k\}$ be the list of length K corresponding to the probability function with:

$$\sum_{k=0}^{K} p_k = 1$$

We want :

$$P(Y = k/K) = p_k$$

With Y random value output. Let X be equiprobable random variable with $X \in [0; 1.]$. If $(a, b) \in [0; 1.]^2$ and $a < b$:

$$P(a < X < b) = b - a$$

besides we introduce :

$$S_k = \sum_{n=0}^{k} p_n$$

Thus :

$$P(S_k < X < S_{k+1}) = p_{k+1}$$

we can deduce Y :

$$Y = (\arg_{k\in[0;K]}[S_k < X < S_{k+1}])/K$$

3.2. Implementations

In other words, we have to create the list of the p_k successive sums and to compare each elements with an equiprobable variable. The index of the first higher element is the desired output. The figure 4 shows the Max Patch of a message version of this random generator.

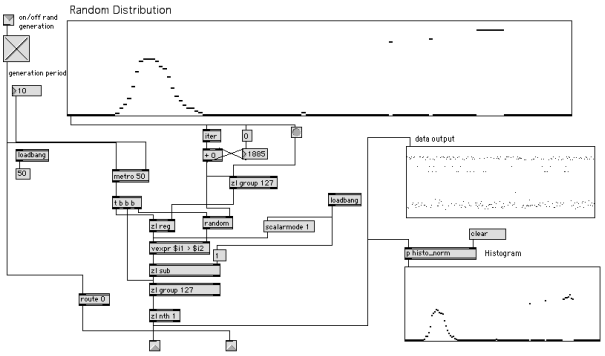

Figure 4. *Max Patch of the random generator message version.*

The *vexpr* converts the list S_k into a binary value list according to the comparison value. *zl sub* gives the index of the true value of the last list and *zl nth* returns the first element of the index list. We can specify the random generation frequency by modification of the *metro* period.

This patch gives good results for moderately high generation frequency ($f < 100$ Hz). But, as we saw before, message timing in Max can have irregularities especially for high frequency generation. With high density granular stream, these irregularities can produce audible artifacts. In order to resolve these problems, considering the signal parameter capability of *bufGranul*$\sim$, we developed an external MSP object *rand_dist_list*$\sim$ of random signal generation on the same scheme. The help patch of this object is shown in figure 5.

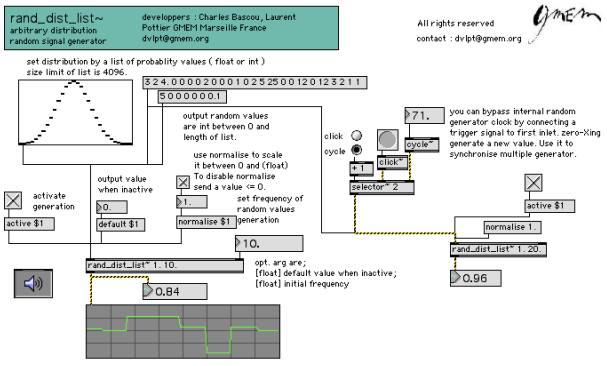

Figure 5. *Help patch of the rand_dist_list*$\sim$ *object.*

The probability function is input as a list of values. As in the message version, output is a signal going from 0 to *length_of_list*. An internal clock, set by frequency (2^{nd} inlet) in Hz, controls the generation of new values. Working with signal allows generating new values every sample. An important feature is its signal synchronisation capability. Indeed, you can bypass internal generation clock by connecting a trigger signal to the first inlet. In this case, a new value is generated at each zero crossing of the trigger signal. It is very useful if you want to synchronize multiple generators and/or synchronise the generator with grain triggering of the *bufGranul*$\sim$ object.

3.3. Graphical user interface

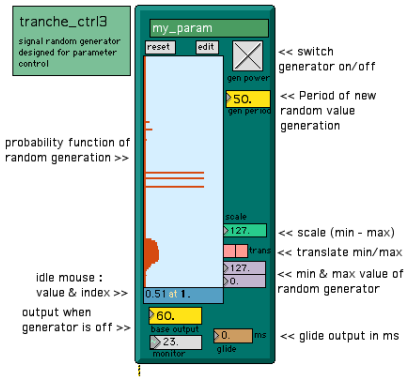

Figure 6. *Graphical user interface of the random parameter control.*

Shown in figure 6, a graphical user interface has been written to fit to the need of granular synthesis parameter control. The output of the random generator are linearly mapped from a low to a high bounds corresponding to the first and last value of the probability list. The probability function is edited by an extended multislider object. We especially added idle value preview and some high level graphical editing functions such as translating, scaling, smoothing curve.

4. APPLICATIONS

4.1. Sound design

Thanks to the richness of the Granular Synthesis scheme, a great variety of sounds can be produced with this system. The control of sound synthesis in a stochastic way is very powerful. You can combine stochastic control of some parameters and deterministic control of other parameters. The sound result can go from pitched sound to noisy sound texture. Making very long sinusoidal grains (several seconds) with moderate trigger frequency (about 10 grains/second) produces random spectrum content sustained sound. In this case, the detune curve of the random generator is very meaningful and allows to constrain the pitch/tone at a high level. More radical synthesis values

like very short grains (lower than 2 milliseconds) with high trigger frequency (about 5000 grain/seconds) produce quasi-constant coloured noises. Here, the detune curve is directly related to the colour produced. Moreover, Microsound synthesis allows you to explore the transition between rhythm and frequency (triggering the grains near 10 to 50 Hz), between crunchy grains (envelope sizes less than 50 ms) and smooth grains and between spatially localised grains (fusion) and dispersed grains. All these explorations can produce very strange and interesting effects on the perception.

4.2. Real Time Granulation

Recently, this system has been successfully adapted to live input granulation. Technically, you can record a sound in a *buffer~* (memory space in RAM) and play with *bufgranul~* the wave stored in this buffer. Obviously, there can be a delay between writing and reading, especially if you want to read faster than you write (*detune* value higher than 1). The *bufgranul~* object can loop in a buffer. That means that if you begin to read a grain at the end of a buffer, it jumps to the beginning of the same buffer. A synthesis parameter filter patch has been developed to avoid that the reading point of grains cross the recording point in the buffer by constraining length. The patch for real-time granulation showing the filter and the random parameter GUI controlers is represented in figure 7.

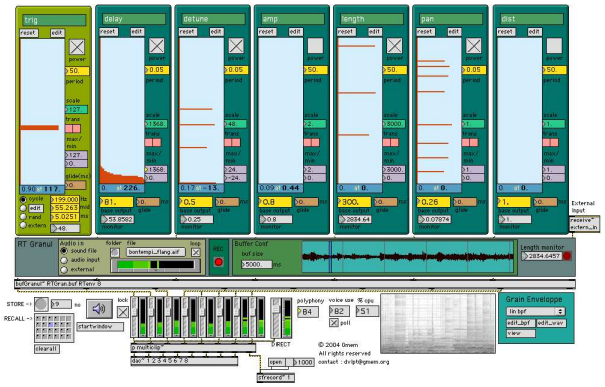

Figure 7. *Patch for real time granulation with bufgranul~*

It was especially used in the Francois Narboni piece [1] ".....nouveau et particulier..." for realtime delaying and transposing sound produced by acoustical instruments. We recorded the sound in a buffer (in loop mode) and then triggered grain at a *begin time* minus the desired delay. The triggering period was set to 25 ms and the length of the grain to 75 ms (3 times the period). The detune parameter was set to the desired transpositions. Other parameters were constant or statistically distributed.

An interesting feature of this system is the ability to stop live input recording and then to achieve granulation on the last few seconds of sound. It creates live "freezing" effect allowing to make a percussive note last more for example.

[1] for string quartet, electronic and video - 2005 - Commande d'Etat/GMEM

4.3. Analysis/synthesis

An important part of the GMU project is the possibility to deduce microsound synthesis parameters from sound analysis that is to say to set up a granular analysis/synthesis tool. Several sound analysis techniques has been studied. We used the Terhardt algorithm, extracting the most pertinent partials from musical sounds, to control the probability law of detune random generator and their corresponding amplitudes. This algorithm initially written by G. Assayag, has been ported on Max/MSP by Todor Todoroff (*Iana~* object) and thus performs analysis in real time. Interesting sound results can be obtained applying the input sound specific spectral colour to complex sound textures.

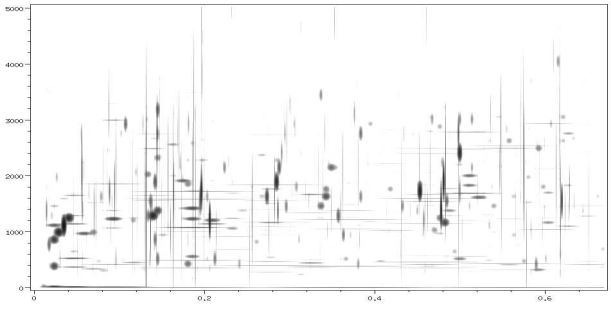

Figure 8. *The representation of a water sound Matching Pursuit decomposition.*

Other research has been made on sound analysis techniques closer to the granular sound model. The Matching Pursuit Algorithm [7] initially introduced by S. Mallat and Z. Zhang performs sound decomposition onto a set of elementary Gabor grains. Each grains are indexed by frequency and duration. It gives very good result on sound where sinusoidal partials vary rapidly in amplitude and frequency. A Matching Pursuit water sound decomposition from the *Lastwave* software [9] is showed in figure 8. This figure shows grains according to their Wigner Time-Frequency distribution. A new OpenMusic library has been developed to allow transformations of the analysis results. From the parameters of each grains detected, we can deduce local stochastic laws for frequency, amplitude and duration and then uncorrelate time with the grain density of the original sound. Interfaces has been developed to resynthesize sound in the Max/MSP environment using *bufGranul~* and *rand_dist_list~* , performing either direct resynthesis or stochastically controlled sound textures.

The application of such techniques are promising allowing to deduce sonic behaviours from the analysis of natural, instrumental or electronic sound materials.

5. IMPROVEMENTS AND FUTURE WORK

To improve this synthesis system, further researches have to be made especially on a parameter dependency system and strategies involved. An intuitive environment for managing parameter relations is an important specification to go further in complex sound structure generation. Another feature we have to work on is an ergonomic system for parameter evolution in time. This includes managing fluctuation of the random generator probability laws with a special attention on ways to interpolate these distributions along time. It will probably be interesting to set up a kind of "image based sequencer" to achieve this, proposing musically meaningful graphical tools to edit them.

6. REFERENCES

[1] Gabor D., "Acoustical Quanta and the Theory of Hearing", *Nature*, 159(4044):591-594, 1947.

[2] Roads C., *Microsound*, Cambridge, Massachusetts: MIT Press, 2002.

[3] Pottier L., "GMU - An Integrated Microsound Synthesis System" *Proceedings of the Computer Music Modeling and Retrieval Conference*, Montpellier (France), 2003.

[4] Bascou C., Pottier L., "New sound decomposition method applied to Granular Synthesis", *ICMC 2005* , to be published in 2005.

[5] Pottier L., "Dynamical spatialization of sound. HOLOPHON : a graphic and algorithmic editor for Sigma1", *Dafx98 Proceedings*, Barcelone, 1998

[6] Pottier L., "Holophon : projet de spatialisation multi-sources pour une diffusion multi-haut-parleurs", *JIM2000 Proceedings*, Bordeaux, 2000.

[7] Mallat S. and Zhang Z., "Matching pursuits with time-frequency dictionaries", *IEEE Transactions on Signal Processing*,vol 41, no 12, p 3397–3415, 2004.

[8] Gribonval R., Rodet X., Depalle P., Bacry E. and Mallat S., "Sound signal decomposition using a high resolution matching pursuit", *Proceedings of ICMC'96*, Clear Water Bay, Hong-Kong, 1996.

[9] Gribonval R., Bacry E., LastWave 2 Software `http://www.cmap.polytechnique.fr/~Bacry/LastWave/`, 2003.