

STOCHOS : A Real Time Stochastic Sound Synthesis application

Designed and Programmed by Sinan Bokesoy in collaboration with Gerard Pape

During my studies in 2001/02 through the wonderful book of Xenakis' 'Formalized Music' in CCMIX, I have discovered that there were still a lot of ideas of him remained untouched with the contemporary electronic music tools. His ideas seem to have endless openings and deliver a lot of inspirations for designing computer music composition tools. For instance, the *Achorripsis* mechanism (described in detail in Formalized Music) distributes a certain number of note events in a certain time interval by assigning stochastic functions to density, onset time and duration parameters. Then for each note event, envelope functions are generated for the parameters like pitch (glissandi) and intensity. This and further mechanisms in Xenakis music deliver a certain attitude in sound composition and the creation of musical form.

The implementation of *Achorripsis* in a real time operating platform like Max/Msp drove me to new openings which lead to *Stochos*. While developing the application design, many features in the computer music history, especially composition of sound masses and types of granular synthesis could be fully reached within the same mechanism.

By having a time scale of millisecond resolution and having a modulation matrix assigning multiple curve generators to multiples synthesis parameters at the same time, *Stochos* delivers enormous data flow within the control of the composer. I would say, a real computer music tool, where the computer creates and processes the data and brings the sonic result.

About this version :

I introduce you this version for educational purposes; it is prohibited to use the application under commercial conditions. You are welcome to contact me for developments in special projects.

The first version of *Stochos* was finished early in 2002 and then by collaborating with Gerard Pape (CCMIX), different features have been added for compositional ideas (*LoHi* barriers and *LogisticMap* function were his ideas).

It was a fruitful period where all the potentials of the software were tested with compositions, which is very good for creating a composers tool. In this version, you will have all the materials to shape the sound in continuum. Especially the synthesis engine I have designed in 2004 for my piece RuinsA39, combines many possibilities of sound synthesis of a single waveform up to advanced granular synthesis.

I recommend using a fast G4 and preferably G5 for rich sound results.

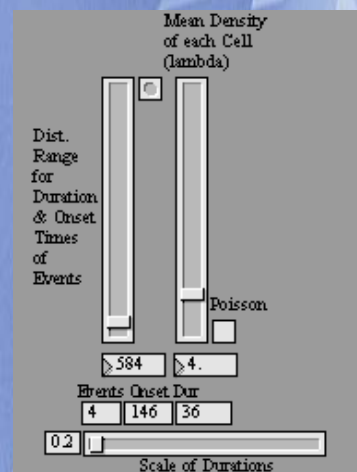
When you launch *Stochos* the program will ask you your sample directory, in order make a list in the sample menu.

The preset number 1 will be automatically loaded and ready to play.

You should activate the DAC switch first and then the On/Off switch.

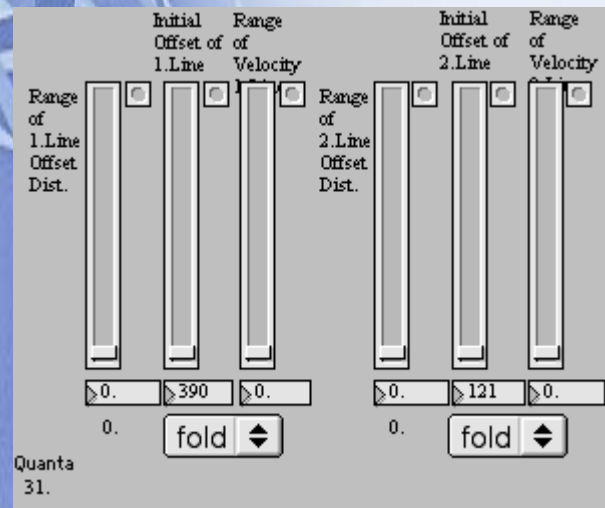
In case you have difficulties in hearing a sound, you should control the parameters for amplitude and pitch carefully, since *Stochos* can give unexpected results in combination of many parameters because of its stochastic nature.

Stochos parameters :



This section has the controls for the event distribution process inside in the cell. The left slider sets the cell length which can be as low as 10ms. The right slider sets the mean density in the corresponding cell. If the distribution box is selected, then the events will be distributed according the Poisson distribution with the mean density set by this slider. If the box is not checked, then the distribution will have constant density value.

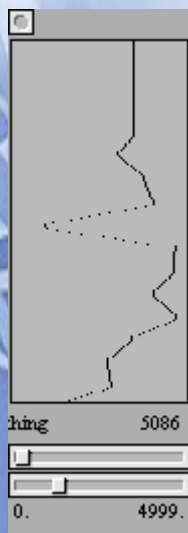
You can see the event density, onset time interval between events and the duration of each event on the number boxes below. The horizontal slider sets the duration scale value. This value will be multiplied with the value of each calculated event value.



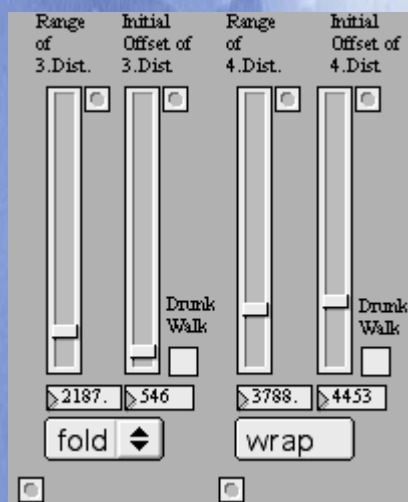
These are the curve (glissandi) generators as implemented in the Achorripsis model of Xenakis. There are two glissandi generators, which can be assigned as a modulation source to synthesis parameters for each running event. Each curve generator consists of 3 sliders. The left one controls the range of the stochastic function, which will be added to the offset value set by the middle slider. The right slider sets the velocity range of the glissandi. The velocity can be controlled with other stochastic function then the curve generator function as you can see on the 1.Matrix on the right side of the screen.

The menu below has two states; fold and wrap. They are the modes of how the curve generators do behave when they reach the low and barrier limits. Their either fold back from the barrier or wrap around.

The little button at the top-right side of each slider delivers a zooming function for easy editing of the slider value.

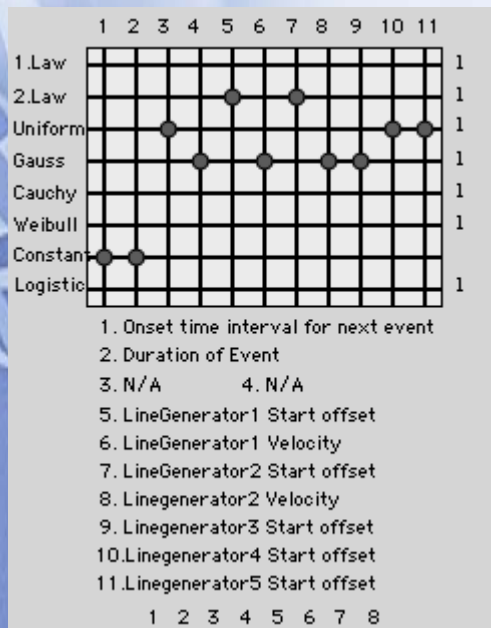


This display shows the evolution of the curve generator above like a flow of data. The sliders below the display are the low and high barriers limiting the curve generator output.

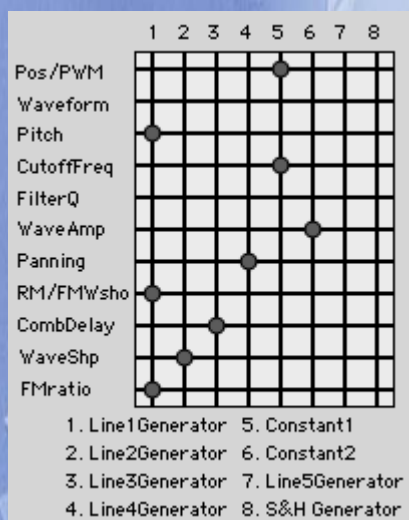


These are also the curve generators but without the velocity function. They float around an offset value set by the right slider, and change the range the stochastic function with the left slider. Each of them can be assigned to a unique stochastic function on the 1.Matrix. The 'drunk walk' button switches to the 'drunk' random walk mode.

There is also a third curve generator of this type, and two constant value generators labelled with constan1&2. In total, there are 5 curve generators and 2 constant value generators assigned as modulation generators for each running event.



The 1.Matrix assigns each stochastic function to a destination as given in the list below. When clicked on the 'Weibull' and 'Logistic' labels, a window opens showing the parameters for these distributions. The 1.Law is 'exponential distribution' and the 2.Law is the 'Linear distribution'. The numbers on the right side of the 1.matrix down-sample the relevant stochastic function on that row, such as you can have different clock frequency for each stochastic function.



This is the 2. Matrix, where you assign the curve generators to the synthesis parameters following each event, you would like to modulate. On some rows you will see two parameters labelled at the same time, but in fact it depends also on the operation mode, whether they are active or not. For instance, Waveform Position on the 1.Row won't work if you are in the SynthesisObj mode of *Stochos*.

In this version of *Stochos*, WaveShaping, Waveform and S&H Generator do not exist.



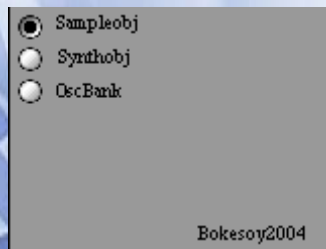
This section has various controls regarding the audio driver and the preset select/store mechanism.

Stochos has two cascade running filters. The cut-off frequency and the resonance can be controlled by the stochastic curve generators. The button switches the filters on and off. The slider provides between the comb filter and the cascade filter outputs. In the menus, one can select the filter types for each filter in cascade.

Regarding the audio driver section; to be able to work with small events one has to choose small signal vector sizes in the cost of the CPU consumption. The 'Voices' menu assigns the maximum voices which *Stochos* allocates. It depends on your CPU power whether it can be played or not. After changing the number of voices, one has to load the preset again. The default voice number is 5. The 'voice stealing' button has the same function as in the implementation of the 'poly' object.

The preset buttons help to select and store the preset parameters (all the parameters on the user interface) to slots. The small buttons after the slot numbers make it possible to load and save preset banks to disk.

The small button left to the 'voice allocation' menu opens a tuning map, which defines custom tuning scales. It gets the data from the LineGenerator7 and 'sieves' its output.



There are three operating modes on 'Stochos'. 1. *SampleObj* mode
2. *SynthesisObj* mode 3. *OscBank* mode


In the first mode, 'Stochos' uses samples assigned to each event as a sound source. In the second mode, you can assign several synthesis sources *noise*, *PWM* and 44 partial *additive synthesizer* as a sound source and even morph between them in continuously. In the third mode, you can use an oscillator bank, as implemented in the *oscbank* object of Max/Msp to generate and manipulate a cloud of sound.



Here you can assign a sample waveform for the *Stochos* events. The small button on the right of the sample menu lets you select your sample directory on the 'haddisk'. You can also select the base frequency of your sample with the number box below. The two small horizontal sliders are attack and release parameters applied to the amplitude envelope function. When the 'LoopSw' is selected, the sample will be looped when the event duration exceeds the sample length. You can specify the loop start and end points with the relevant sliders. The 'forward' menu assigns the playback direction of the wavetables. It can be forward/backward/random.

When you are operating on the SynthObj mode of *Stochos*, you can click on the menu 'additiv' to open a window where you apply additive synthesis with 44 partials. The slider on the right side of the menu makes it possible to morph from additive synthesis to PWM synthesis. The slider on the left side makes it possible to morph ro sound from either PWM or additive to noise.

The buttons FM and RM activate frequency modulation and ring modulation of the event, which happens before the cascade filters.



The OSC Bank operates likewise with pitch and amp modulation only, which comes from the curve generators of *Stochos*. The pitch of each oscillator in the cloud will be modulated by LineGenerator3 and the amplitude by LineGenerator2. With the small button on the bottom left corner, you can assign a waveform for the oscillators. There are also control sliders for spatialization and filtering of the cloud.

Bokesoy 2005,
sinan@sonic-disorder.com
www.sonic-disorder.com