Journées d'Informatique Musicale, 9e édition, Marseille, 29 - 31 mai 2002

Sounds of the Stars - Gravity-Synthesis with SynthX

Georg Bönn
georg@boenn.de

Résumé

SynthX consists of 26 Modules for the synthesis and processing of digital sounds. Its new method of Gravity-Synthesis is done with a simulation of gravity-effects. The orbits of objects in space are being calculated with numerical integration and translated into soundfiles. The simulation is based upon the definitions of objects through position, mass and velocity. Newton's laws determine the path of these objects through space. The paths are projected on a time-line, thus creating samples of a time-variant signal, which can be written to an audio-stream.

Keywords

Gravity-Synthesis, Law of Gravitation, Newton, Simulation, Space, Physical Modelling, Friction, DSP, Chaos, Microtonality

Introduction

The program SynthX consists of 26 Modules for the synthesis and processing of digital sounds on Apple Macintosh Computers. The Kernel written in C++ is platform-independent and runs in terminal-mode. The first graphical User-Interface has been written for the new MacOS X (10.1). SynthX runs also on OS 9 with CarbonLib 1.4. The modules of SynthX contain classical filters, convolution, signal-generators, moving-sounds-simulation, frequency-shifter, granular synthesis and many others. These modules form a catalogue of sound-tools which I use frequently in my compositions.

With this paper I would like to introduce my method for Gravity-Synthesis, which is programmed in the module PlanetSwing. PlanetSwing was created during my stay as visiting scholar at CCRMA, Stanford, in summer 1999.

The method of Gravity-Synthesis is done with a simulation of gravity-effects. The orbits of objects in space are being calculated with numerical integration and translated into sound-files. The simulation is based upon the definitions of objects through position, mass and velocity. Newton's laws determine the path of such an object through space. There is no consideration of relativistic effects in PlanetSwing.

The very first ideas for this concept came while I was working on moving sounds in space controlled by some algorithmic process. There was an interesting phantasy, that virtual sound-sources on their path through the listening-space would behave, as they would be masses, which act on each other due to the laws of gravitation.

This idea became more general after some time and I stated the hypothesis, that one considers the motion of masses like a time-variant signal, which represents a unique sound or which can control other musical parameters of sound-synthesis. A signal is created by a mass through change of velocity and direction in relation to a specific axis of a coordinate-system in space (x, y or z). This inertial reference frame is given with the eye of an observer. In this way we are scanning three signals per object: The motion of the object relative to the x-, y- and z-axis. You could also say, that the motion of an object is projected on a time-line via the coordinate-system of the observer.

In order to generate sound-files in AIFF or WAVE-format from these informations, I wrote additional C++-classes for the streaming of audio data. This was the origin of the SynthX project.

Scripts for PlanetSwing

To start the sound-synthesis, the user generates first a text-file (ASCII), in which he defines the input-data for the simulation. The module PlanetSwing reads that text-file and then calculates a sound-file plus, if desired, a graphical output of the orbits in PostScript-format. You can view a PostScript-file on the Mac with the free software MacGS from Aladdin or you can print it with a PostScript-Printer. In a future version of SynthX you can open the graphic output or animation within a proper document-window. The graphic output is important in order to view the motion of the objects and to find good constellations for the sound-synthesis.

Here you have an example-script for a simulation with three objects (the lines are enumerated just for explanation purposes):

```
01 PlanetSwing
02 Version 1.0
03
04 constant of gravitation: 0.001
05
06 number of planets: 3
07
08 repulsion: 1.
09
10 friction: 1.
11
12 session: 3L001s.txt
13
14 plot: 3L001p.txt
15
16 soundfile: 3L001.aif
17
18 sample-rate: 44100
19
20 length: 1
21
22 mixing: 2 y 3 x
23
24 planet: 1
25 position: 16 0 0
26 mass: 12000
27 vector: 0 0 0
28 fixed?: 0
29
30 planet: 2
31 position: 8 0 0
32 mass: 1
33 vector:   0 1 0
34 fixed?: 0
35
36 planet: 3
37 position: 7 0 0
38 mass: 1
39 vector:   0 0 1
40 fixed?: 0
41
42 TWO HIGH VOICES -
43 HETEROPHONY
44 WITH DIFFERENCE TONES
```

In the script you find all parameters, which are necessary for the sound-synthesis with PlanetSwing. Line 1 and 2 form the obligational head of the script. The version number ensures compatibility with future versions of the software. Empty lines of the script are being ignored and serve only for clarity. Line 4 gives the first parameter of the simulation, the so-called universal constant of gravitation. Remember, that the gravitation is the force through which two bodies attract each other. That force is proportional to the product of their masses and inverse proportional to the square of their distance. The universal constant of gravitation G defines how large the force in reality is. It is quiet small compared to the electromagnetic force and the two nuclear forces. My simulator allows to play with that scale, so you can change the value of G. Therefore the results become different, because the sum of the forces becomes larger or smaller. Thus it is sufficient to make very small changes of G, around 10^{-4} in order to obtain large variations of the results.
Newton's law of gravitation in vector notation[1]:

$$\vec{F} = GMm / r^2 * \vec{r} / r$$

Vector F is the force on mass m due to mass M and vector r points from M to m.
G has the value: $6.672 * 10^{-11} m^3 kg^{-1} s^{-2}$

Back to our example. Line 6 indicates how many objects are in the simulation. It is important to write the name of the parameters, the tags, exactly as shown. They always end with ':' and need at least one white-space afterwards. I also recommend to use only text-files in Mac-format.

[1] BATE, Roger R., et al., *Fundamentals of Astrodynamics*, Dover, New York, 1971, p.4 ff.

For the moment we put the parameters "repulsion" and "friction" aside. In the lines 12, 14 and 16 we have the names of three files. The "session"-file is a text-file created by the program and contains a log of all in- and outputs of a single synthesis. The advantage of such a log-file is that it records the last positions and vectors of all objects at the end of the simulation. With these data it is possible to launch another synthesis with another script. If you do not want this log, type "none" after the session-tag. Then the generation of a log-file will be suppressed.

The name of the PostScript-file mentioned above comes after the tag "plot". In this case it is also a text-file but with PostScript commands.

Fig. 1 shows the plot of the example above after rendering with MacGS. You see the right-angled orbital planes of the two light objects. One orbit forms the optical illusion of a torus, because the orbit shifts and rotates around the heavy object, which is hidden in the center of the image. Because of the large amount of data it is recommended only to plot simulations of more or less 1 second at 44.1 kHz. If you write the name "none", no plot-file will be genearated, which gives also a faster performance.

The name of the sound-file is given in line 16. The suffix has no relevance for the format of the sound-file. The file-format is defined through the default settings, which can be altered in the Settings-dialog in the menu DSP. The default at program-start is 44100 Hz sample-rate, 16 bit resolution and AIFF-format. Only the sampe-rate can be overwritten by the script (see line 18).

The tag "length" determines the length of the resulting sound-file in seconds.

You can put comments everywhere between the parameter-lines, but you shouldn't use tag-names in comments, unless you write everything in capital letters which I recommend to do so (see line 42).

A single object has 5 different tags:

- "planet" is the index of the object.
  Currently you can define up to 1024 different objects.
- "position" of the object in cartesian coordinates (x y z).
- "mass" of the object. You can work with relative values, e.g. if the sun has the mass 1,
  jupiter has the mass $9.54 * 10^{-4}$
- "vector" is the velocity in relation to the x-, y- and z-axis
- "fixed?" is a switch, which can be used to hold an object on its initial position.
  0 means: the object moves normally, 1 means: the object is fixed, which means it acts with gravitational forces on other objects, but it does not move itself.

Now for the transformation of motion into sound: As mentioned above, the program scans the motion of the objects relative to the three axes in space. It is left to the user, which object and which axis he would like to mix to the samples of a sound-file. In our example, we mix the motion of the second object relative to the y-axis together with the motion of the third object relative to the x-axis. Therefore, we simply give the indices of the desired objects with the name of the desired axis after the tag "mixing" (see line 22). Because the motion of the great mass (Index 1) gives virtually nothing to the sound-synthesis, because it moves only very little, we leave index 1 out of this list. Here we are dealing with, so to speak, the virtual sound-outputs of the objects. There is a maximum number of three sound-ouputs per object corresponding to the three axes in space. In our example we can expect two different tones, despite of the equal velocity of the two objects. The third object is closer to the heavy mass, therefore it reaches a higher angle velocity and we will hear a higher pitch. So one can change the pitches, if one alters the distance of a light mass in an orbit around a heavy mass.

Fig. 2A shows an excerpt of the waveform of our example, fig. 2B the corresponding sonogram. A two-voiced heterophony is created with many internal fluctuations, jumps and difference tones.

The pitfall of making sound-synthesis with PlanetSwing is, that you often have to filter out DC-offsets. This is needed, because the simulated systems are free to move in any direction of the space. So it is very easy to get 0 Hz or other very low frequencies when the system only slightly shifts around. SynthX offers the module DCRemove for filtering these sub-audio frequencies.

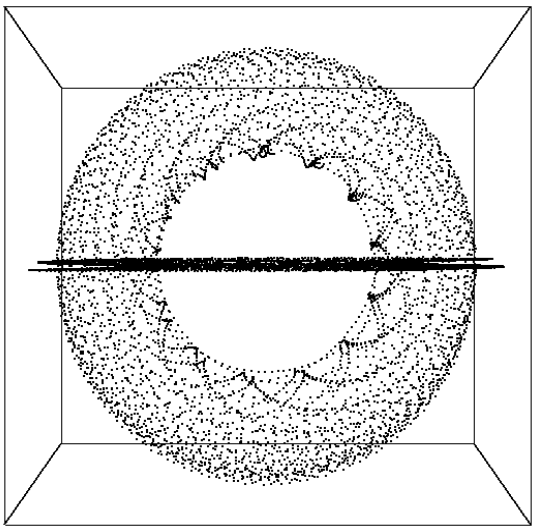

Fig. 1: One second of simulation of our example-script. The heavy mass is hidden in the center. The torus-like object consists of samples of the orbit of the second object. The torus is an optical illusion. In reality the orbital plane is flat. You look towards the edge of the orbital plane of the second object.

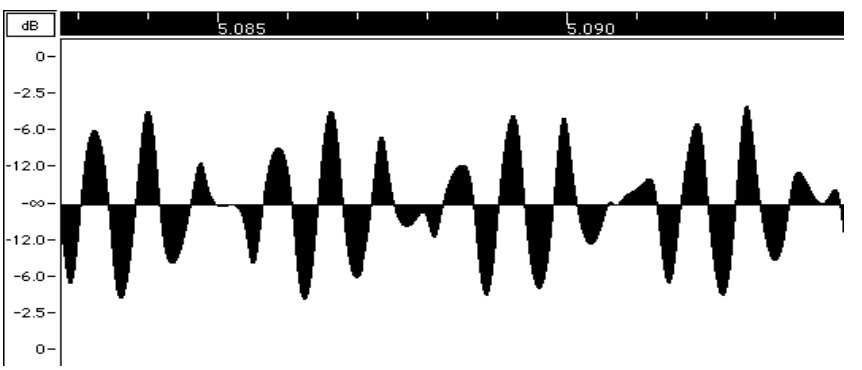

Fig. 2A: Waveform of the synthesized sound of our example.

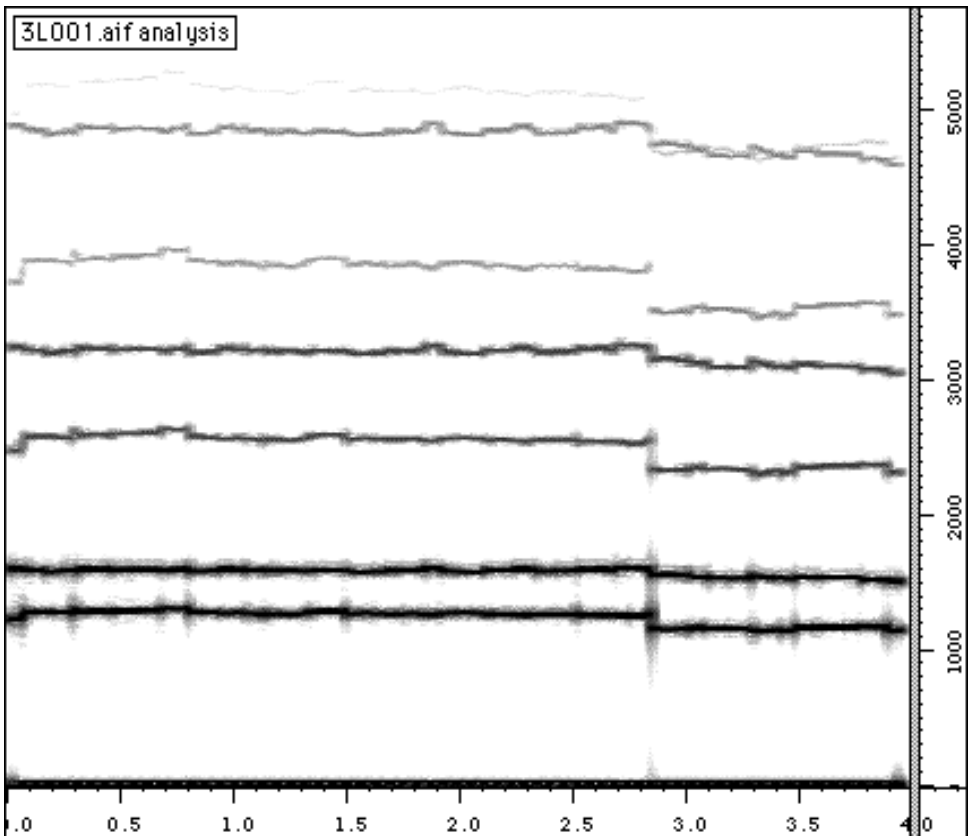

Fig. 2B: Sonogram of the synthesized sound of our example. You can see clearly the heterophonic structure in the overtones.

The solar system

Of course, with a gravity-simulator one can also build real constellations, like our solar system for example. First we need the distances of the planets from the sun in astronomical units (1 AU = mean distance from the earth to the sun), The orbital periods of the planets should be given in days. The velocity is then calculated in AU / day. Finally, the masses of the planets should be written in such unit, where the mass of the sun is equal to1 mass unit.

Example:
Jupiter's distance from the sun is 5.2 AU.
His orbital period is 11.86 years, thus 4332 days.
His velocity is therefore 2 π* 5.2 / 4332 = 0.00754 AU/day

The sun is positioned in the origin of the coordinate-system (0,0,0). The positive x-axis points to the planets, which we set on a line at the beginning of the simulation. We let the planets move counter-clockwise around the sun, therefore we write the velocitiy as a positive value for y in the appropriate vector.

Here the script for the first six planets:

```
PlanetSwing
Version 1.0

constant of gravitation: 0.0002959
number of planets: 7
repulsion: 1.
friction: 1.
session: solsyss.txt
plot: solsysp.txt
soundfile: solsys.aif
sample-rate: 44100
length: 2
mixing: 2 y 3 y 4 y 5 y 6 y 7 y

SUN
planet: 1
```

```
position: 0 0 0
mass: 1
vector: 0 0 0
fixed?: 1

MERCURY
planet: 2
position: 0.39 0 0
mass: 0.00000016552803
vector:   0 0.02793 0
fixed?: 0

VENUS
planet: 3
position: 0.72 0 0
mass: 0.0000024528245
vector:   0 0.01999 0
fixed?: 0

EARTH
planet: 4
position: 1 0 0
mass: 0.0000030096006
vector:   0 0.01721 0
fixed?: 0

MARS
planet: 5
position: 1.52 0 0
mass: 0.00000032202727
vector:   0 0.01392 0
fixed?: 0

JUPITER
planet: 6
position: 5.2 0 0
mass: 0.00095672194
vector:   0 0.00754 0
fixed?: 0

SATURN
planet: 7
position: 9.6 0 0
mass: 0.00028645379
vector:   0 0.0056 0
fixed?: 0
```

One can use the Gaussian constant of gravitation, because the distances, masses and velocities were given in the way I described above.

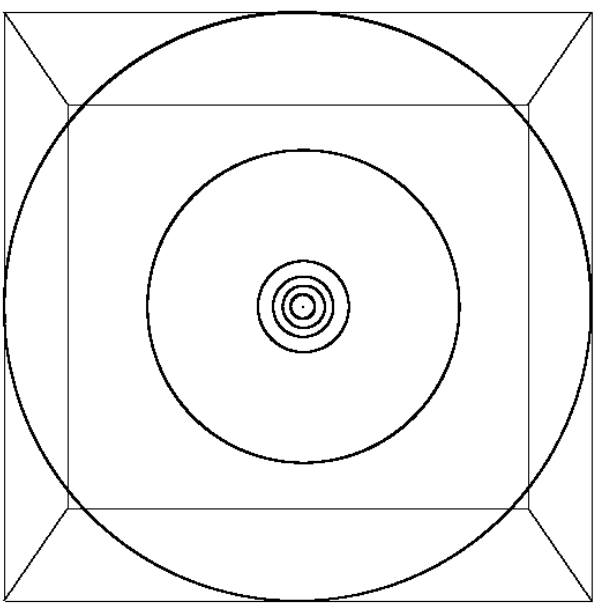

Fig. 3: The first six planets of the solar-system

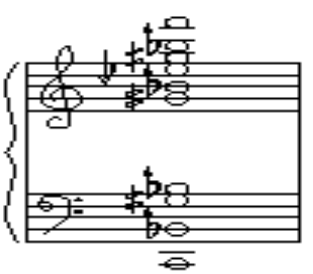

Fig. 4: Chord of the solar-system

The first six planets form a calm and static sound-spectrum at 44.1 kHz sample-rate (see fig.4).

Friction and thrust

In addition to the natural force of gravitation I programmed two other factors in the simulator. The friction-parameter acts on all objects and is not relevant only if its value equals 1. At values greater than 1, the system receives energy via thrust, which means all objects together will be constantly accelerated. The sound makes therefore a glissando downwards. For example, Jupiter escapes the gravitational field of the sun (friction: 1.0001):

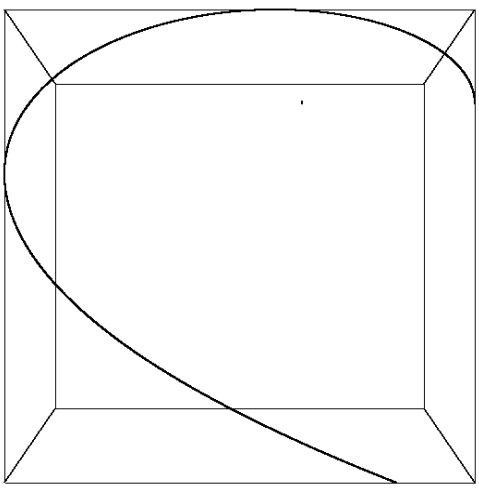

Fig. 5 Escape velocity

At values smaller than 1, all objects will constantly slow down, the gravitational force increases and the sound makes a glissando upwards. In a future version of the software it should be possible, that single objects can have individual and time-related values for friction.

Here the opposite example. Jupiter falls towards the sun (friction: 0.9999):

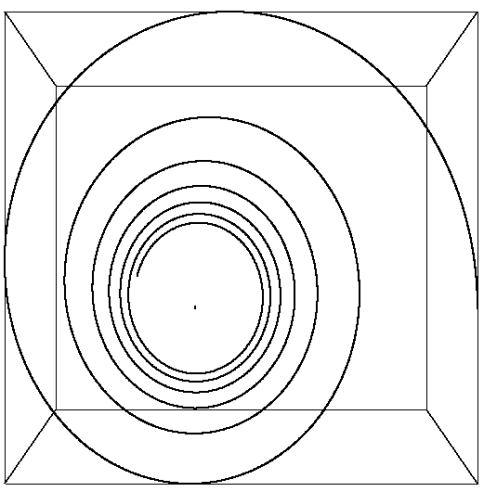

Fig.6 Fall of Jupiter

The repulsion-parameter is only experimental and should always be 1. The idea behind repulsion is negative gravitation, repulsion of masses. In this way it should be possible for example, that a mass swings back and forth on a line between two distant masses.

Synthesis

We have already mentioned, that the program maps the coordinates of selected trajectories into sample-values of an audio-stream. The user selects an object and specifies which axis of the simulation frame will be mapped (x, y or z). How can the resulting audio-stream be characterized?

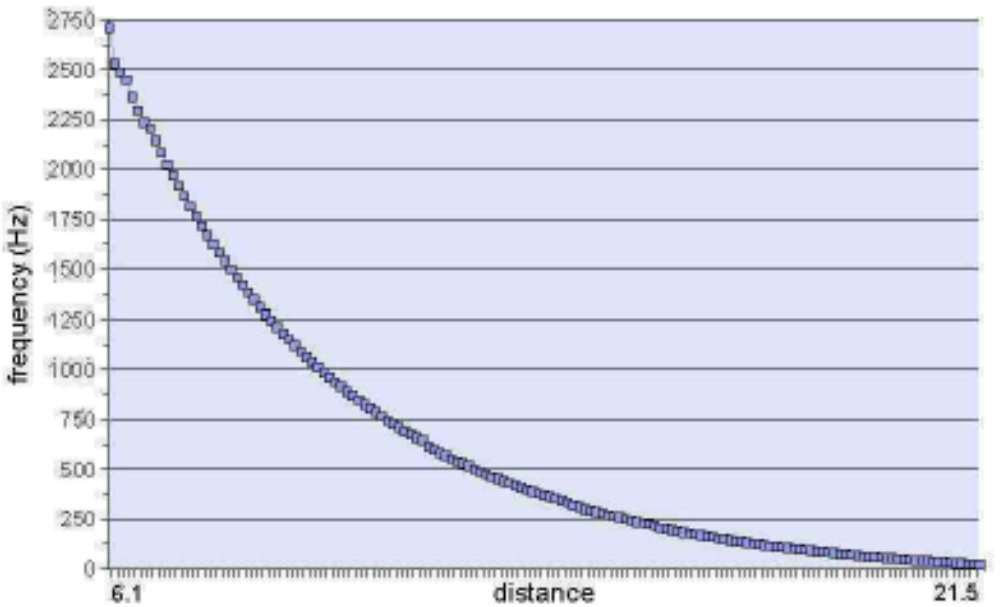

Fig. 7

Figure 7 shows the relationship between the radius of an orbit and the pitch generated by the orbiting mass. As the distance from the heavy mass increases, the pitch of the tone decreases constantly. One gets a similar result for the influence of the initial velocity of the orbiting object on the pitch, while using the same distance from the central mass. The initial mass of the central object has an equal influence on the resulting pitch, as well as the gravitational constant. The parameters of the simulation can directly change the pitch and the phase of the result. The respective amplitudes change if the radius of an orbit changes. In n-body systems, the object closest to the central mass has not only the highest pitch, but also the lowest amplitude. The serie of tests shown in Figure 7 was done with the following script, where the vector of the second object was constantly altered:

```
PlanetSwing
Version 2.0
Sat Apr 18 19:31:36 2002

constant of gravitation: 0.001
number of planets: 2
step-size: 1
repulsion: 1
friction: 1
session: none
plot: t20041249__101_p.txt
soundfile: t20041249__101_a.aif
sample-rate: 44100
length: 1
mixing: 2 x

planet: 1
position: 0 0 0
mass: 12000
vector: 0 0 0
fixed?: 0

planet: 2
position: 16 0 0
mass: 1
vector: 0 1 0
fixed?: 0
```

There are certain limits in which the results become meaningful in terms of a musical application. If we use the script as shown above, and we alter only one variable, then we have the following lower and upper boundaries for that variable:

Limits	low	top
Position of Planet 2:	6	23.9
Initial Velocity of Planet 2:	0.4	1.18
Mass of Planet 1 (center): 8500	41000	
Constant of Gravitation: 0.0007	0.0034	

These limits are only true for the 2-body system as defined with this example script.

Structure of the sound

Here some general remarks about musical properties of the synthesized sounds:

1. One can synthesize monodic or polyphonic sounds. The development of the sounds is either stable or it can evolve with more or less strong fluctuations.

2. The intervals, fluctuations and glissandi of pitches do not belong to a certain scale-system. They can establish micro-intervals, but there is also the possibility for surprising pitch-jumps.

3. The pattern of movements within the polyphonic fluctuating sounds is often heterophonic. The ups and downs of the voices are often surprising but they are not aleatoric, because every change of development within the simulation and therefore within the sounds is at any time based on everything which has happened before. The future of the system is already determined by the initial data. However, it is sufficient to make only slight differences in these data in order to obtain a very different future.

Conclusion

One could characterize the method of Gravity-Synthesis as a combination of sound synthesis with a physical model *and* algorithmic composition. An initial set of physical parameters defines a system of *interactive* sounds, where each voice influences the other voices. Subtle changes of the initial set can have a great impact on the synthesized sound. Within the limits of rounding errors, all moments of the evolving sound are the result of every single moment which came before, and each moment is the germ and the condition for all following states of the sound structure.

The underlying object/force model of the program can be extended in the future in order to experiment with many different physical structures, like fluids, gas, particles, or even membranes, plates, strings etc..

SynthX is shareware an can be obtained from http://www.boenn.de.
For MacGS see http://www.aladdin.com.

Bibliography

BATE, Roger R., et al., *Fundamentals of Astrodynamics*, Dover, New York, 1971
LAYZER, David, *Constructing the Universe*, Scientific American Books, New York, 1989
KRUCKER, Gerhard, *Informatik und angewandte Mathematik*, http://www.krucker.ch/Skripten-Uebungen/IAMSkript/IAMSkript.html, 2001
MOORE, F. Richard, *Fundamentals of Computer Music*, Prentice Hall, Englewood Cliffs, 1990
SANNS, Werner, *Praktische Numerik mit Mathematica*, Teubner, Stuttgart, 2001

You guys will be surprised but this topic of sound design and sound synthesis books has been a place to build a fight club here. Anyways, I'll back the same books I mentioned earlier in that fight club thread: 'Sound Synthesis and Sampling' by Martin Russ, if I'm in a budding phase of synthesis.Â For amazingly useful in depth coverage of analog synthesis try to find the rare publication The Complete Guide to Synthesizers by Devarahi. Other useful books include: Electronic Music by Allen Strange, Synthesizer Basics by the editors of Keyboard Magazine, and Synthesizer Techniques by the editors of Keyboard Magazine.