

A Survey of Software Designs from the Sonic Art Group

Peter McIlwain

School of Music - Conservatorium, Monash University

email: Peter.McIlwain@arts.monash.edu.au

Abstract

A survey is presented of a range of approaches to software design that have been developed in the Sonic Art Group, an electro-acoustic music ensemble based in the School of Music - Conservatorium at Monash University. The software is focused on real-time application and therefore interface design is an important element in the discussion of this work. This will be discussed in the context of the *Max/MSP* programming environment, which was used to create the software. Techniques covered in the survey include: methods for real-time additive and subtractive synthesis, spatialisation (with an 8 channel sound system), and automated musique concrète techniques.

1 Introduction

The Sonic Art Group is an ensemble that is based in the School of Music – Conservatorium at Monash University. One of its functions is to provide a research base for the development of software that can be used for the creation of electro-acoustic and computer music. The ensemble mainly works with real-time sound generation using the *Max/MSP* programming environment (which is especially suited to these applications).

A number of software applications have been developed during 2001 and 2002. These include; *Matrixsynth* – a software synthesizer, *Additive Gendèr* – an additive synthesis instrument, *Fractured Delay* – a gate based delay application, and *Cubist Audio* – automated musique concrète software that incorporates *haas* panning. Screen shots of the software interfaces are included in the appendix.

The function and operation of these applications will be described and discussed. As all of these applications are intended for real-time use, efficient and useable interface design is critical. Comments

will therefore be made regarding the various user interface designs that have been adopted.

2 Matrixsynth

Matrixsynth is a software synthesiser based on the synthesis architecture of the *VCS3* synthesiser (Hilton, 1998). Like other synthesisers of its time, the *VCS3* incorporates a modular design in which almost all possible signal paths between its various synthesis components can be configured. The *VCS3* is unique in that module configuration is done using a pin matrix system whereas other systems employ lead and socket connections. Many software synthesis environments, such as: *Max/MSP*, *Pd*, *Reason*, *Audio Mulch* and *Turbosynth*, have used the latter paradigm in the design of the user interface. This design paradigm affords a high level of flexibility in that a large number of modules and configurations are possible, however it is difficult to implement and, for the user, somewhat time consuming to configure.

Matrixsynth on the other hand, offers a limited range of synthesis modules, but its advantage lies in the implementation of the *matrix* object (a standard *Max/MSP* object) that offers a succinct overview of the configuration of the synthesiser (which is not possible with patch cords) and it enables rapid, real-time patch configuration. Figure 1 below shows a section of the matrix interface.

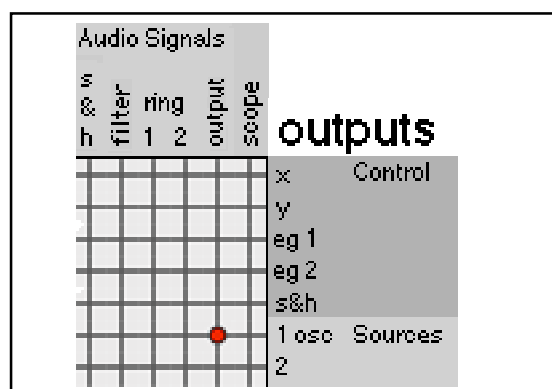


Figure 1. Part of the matrix control interface. Connections are made by clicking on the intersection between inputs, arranged on the horizontal axis and outputs, arranged on the vertical axis. In the example above the output of oscillator 1 is sent to the main output.

2.1 Matrixsynth Components

Matrixsynth comes in two versions Matrixsynth-1 and Matrixsynth-2 (the second version is a simplified design for training applications). The components of Matrixsynth-1 include:

Sound Generators

- Oscillators (with selectable waveforms, sine, triangle and square waveforms)
- Noise Generators
- Audio file and external analogue input signals

Modifiers

- Filter (with selectable highpass, lowpass, bandpass and notch functions)
- Ring Modulator
- Stereo Delay

Controllers

- Envelope Generators (with attack, decay, sustain and release stages)
- Sample and Hold
- X/Y control square (both outputs are scalable)

The oscillator design allows for control of the pulsewidth of the three waveform types. Thus a sawtooth wave can be obtained from extreme settings of the pulsewidth parameter when the triangle wave is selected. Other significant features include a preset facility and selectable triggering sources for the envelope generators. These are: spacebar, tab key, cycle (where the envelope generator operates as an oscillator), mouse, and MIDI (yet to be implemented).

Any output signal can be used as a modulation signal. A wide range of parameters can be modulated in Matrixsynth. These are:

- Oscillator frequency
- Oscillator amplitude
- Oscillator pulsewidth
- Noise amplitude
- Audio file and external analogue input signal amplitude
- Filter cutoff frequency

- Delay time

2.2 Evaluation

While the synthesis modules and control sources are relatively simple, the design affords a high level of predictable control. These design characteristics, together with the flexible patching that the matrix interface offers, enables the creation of very mobile sonic textures. This makes Matrixsynth especially suited to improvisation, a process that is a prominent feature of the music created in the Sonic Art Group.

In comparison with the original VCS3, Matrixsynth offers a greater range of possibilities and a greater level of predictability in the sonic result. One disadvantage that Matrixsynth has, in its current design, is that it only allows for the control of a maximum of two simultaneous parameters with the mouse via the x/y control space. In the VCS3 three parameters can be controlled at once (one hand controlling two parameters with the joystick and the other controlling the one of the various knobs and dials on the synthesizer). This can be addressed by the use of MIDI slider controllers however it has to said that the knobs and dials on the original instrument are more satisfying to use.

3 Additive Gendèr

The *gendèr* is a metallophone found in the Javanese gamelan orchestra¹. It is pronounced “gndair”. The Additive Gendèr is a software instrument - its name referring to the synthesis method used and the characteristic sounds that it produces.

The software uses an additive synthesis approach that utilizes an oscillator bank consisting of 256 oscillators. The amplitude of each oscillator is controlled by the standard Max/MSP *multislider* object (configured with 256 sliders). The frequency of each oscillator is allocated according to the harmonic series based on 20 Hz.

The multislider object functions in three ways: 1) as a trigger that causes a specific oscillator to sound, 2) as a table in which the amplitude values of all the oscillators are stored, and 3) as a visual representation of the power spectrum of the synthesized sound. The multislider object is illustrated in Figure 2 below.

¹ For more information about this instrument see Gendèr (Kartomi and Ornstein, 1984).

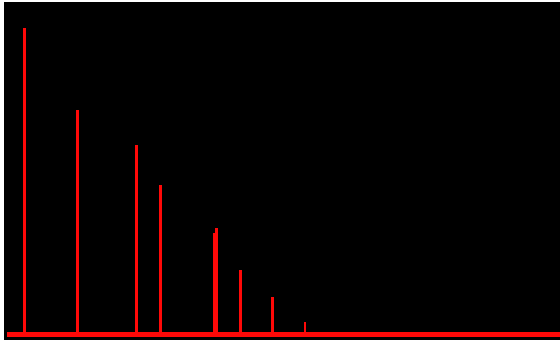


Figure 2. An illustration of the multislider object showing bars (which are independent slider controls). Each bar represents the amplitude of the corresponding oscillator in the oscillator bank.

3.1 Oscillator Triggering and Decay

Each oscillator is triggered when the corresponding slider is set in the multislider object. This is done by clicking in the multislider edit space. Clicking and dragging across the multislider will set and trigger multiple oscillators. The amplitude value that is set is then continuously altered by multiplication with a decay/growth factor. So that the oscillator bank is kept up to date with the current amplitude values, all the values in the multislider object are sent to the oscillator bank at 100 ms intervals.

If the decay/growth factor is below 1 the function will progressively decay the amplitude values in the multislider and oscillator bank. Alternatively values above 1 will cause the amplitude values to rise. The rate of decay or growth is determined by the distance from 1 of the decay/growth factor. Boundaries are set on the function so that no amplitude value can be greater than 1 or less than 0.

The function also includes a randomising feature that alters the decay/growth value. With no randomisation set, the decay curve will be a smooth exponential shape. A high randomising value causes a staggered or erratic decay/growth curve. The randomisation occurs separately for each of the 256 oscillator envelopes. When multiple oscillators are triggered the randomisation results in a shimmering or ringing effect reminiscent of the timbre of a metallophone.

3.2 Stereo Treatment

A stereo effect is created by the use of two oscillator banks (consisting of 256 oscillators each)

that output to left and right channels. As mentioned above, the frequency of each oscillator is set according to the harmonic series however these values are slightly randomized according to a user controlled value. This causes a complex detuning effect between the two oscillator banks and accentuates the stereo space.

Another method of spatial articulation used is the implementation of a multi-tap stereo delay line. An interesting feature of the delay line design is the incorporation of a ring modulator that is positioned after the output stage of the delay line and before the feedback function. A simplified signal flow diagram of this design is shown below in Figure 3.

The effect of a ring modulator inserted in a delay line, as shown in Figure 3, is easiest to describe if we look at a simple scenario in which the input signal is an impulse consisting of a sinusoidal wave at 400Hz and the oscillator is producing a sinusoidal wave at 300Hz. Here the component frequencies in the output waveform will be as follows:

- 1st pass) 700 (sum), 100 (difference)
- 2nd pass) 1000, 400, -200
- 3rd pass) 1300, 700, 100, -500
- 4th pass) 1600, 1000, 400, -200, -800,
- 5th pass) 1900 1300 700 100 -500 -1100
- 6th pass) 2200 1600 1000 400 -200 -800 -1400

Here the terms *first pass*, *second pass* etc. refer to the number of times the same signal has passed through the delay as a result of the feedback function. As can be seen in the frequency list above, the ring modulator produces two effects, 1) alternating frequencies, and 2) an upwards cascade of additional higher frequencies.

The implementation of the ring modulator in Additive Gendèr is more complex than is described above. For each of the two stereo processing chains there are three tapped delay lines. This produces more complex patterns of modulation; however the upwards cascade effect is still present. An additional complication is that the frequency of the oscillator is slowly shifted in response to a random number generator. This produces a rich metallic sound.

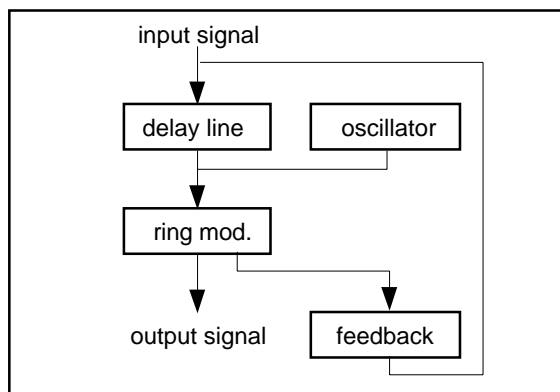


Figure 3. Delay line with a ring modulator and feedback function.

3.3 Additive Gendèr Performance

When the software is set so that amplitudes will decay after triggering, the sounds that are produced are similar to that of a metallophone. The initial amplitude is determined by the level set in the multislider so, in the case of decaying envelopes, triggering in the higher regions works well. For a growth envelope setting, triggering works best when it occurs in the lower regions. This produces a kind of bowed sound.

If single oscillators are triggered in succession they will be heard as pitches. Multiple oscillators triggered at once causes a singular sound with a complex harmonic structure.

As the triggering occurs in response to mouse clicks, the rate of successive triggering is slow. This is not necessarily a problem as it can be seen as a characteristic of the instrument, just as it is not possible to quickly trigger a succession of gongs. Music performed on gongs is by necessity slow, however this characteristic works well because there is time between gong strikes to appreciate the rich and complex evolution of the sound. The same can be said for the Additive Gendèr however it may be worth exploring other forms of interface that could allow for faster triggering rates. A MIDI keyboard has been tried as an alternative triggering source. This of limited use however as the MIDI specification has a 128 as the maximum number of pitches (or frequency units) whereas the Additive Gendèr requires twice that number. One interesting possibility here might be the use of a proximity sensor to set frequency and a pressure pad to set amplitude.

3.4 Performance Gestures

The most interesting, and arguably successful, aspect of the Additive Gendèr software is the integration of control interface with sonic display via the multislider object. Here the object that triggers the sound, also to displays the current state of the sound (as a power spectrum). This makes performance gestures feel natural because the control interface gives feedback to the performer about the current state of the sound generator. The same kind of process occurs when one is performing on a physical, acoustic instrument.

This approach has a secondary benefit in that the multislider object can be displayed via video projection so that an audience can see the performance gestures and possibly divine the sound production process.

4 Fractured Delay

Fractured Delay is a software program that distributes a monophonic signal to eight separate delay lines. The output of these delays are then panned across a stereo field or, in a alternative design, sent to 8 discreet outputs for use in multi-speaker arrays. There are two versions of Fractured Delay; *Fractured Delay 2* for stereo output and *Fractured Delay 8* for eight-channel output.

Each of the eight delay lines has an input gate that is designed to open and close at varying times, and at varying intervals. This causes successive fragments of the input sound to be distributed to the eight delay lines.

The on and off interval values for each gate (called *duration* and *rest* in the program) are generated by two random number generator modules that incorporate the *Gauss* object by Peter Castine (Castine, 1997). Output from gauss is distributed in a bell curve around a center value that is determined by the user. The width of the distribution can also be set by the user.

Fractured Delay is comprised of several components organized in the signal path as shown in Figure 4.

4.1 Input, Filter and Gates

The input signal first passes through an FFT based filter that allows for graphic equaliser style filtering of 253 frequency bands. It then passes to the eight input gates. At the start command for the program,

the gates are successively turned on at random intervals ranging from 0 to 6 sec. They are also assigned duration and rest values. Upon completion of the rest stage (in each delay module) a message is sent from the gate to the two random number generator modules. This initiates the generation of new duration and rest values and the gate is opened again. The gates will cycle continuously through the duration and rest stages until the program is stopped.

An x/y control surface is used to set the centre point for the generation of the duration and rest values. The range of possible values is from 14ms to 22 sec. The use of an X/Y control interface enables rapid control of these two values. This is desirable as the relationship between duration and rest times greatly influences the texture that is generated.

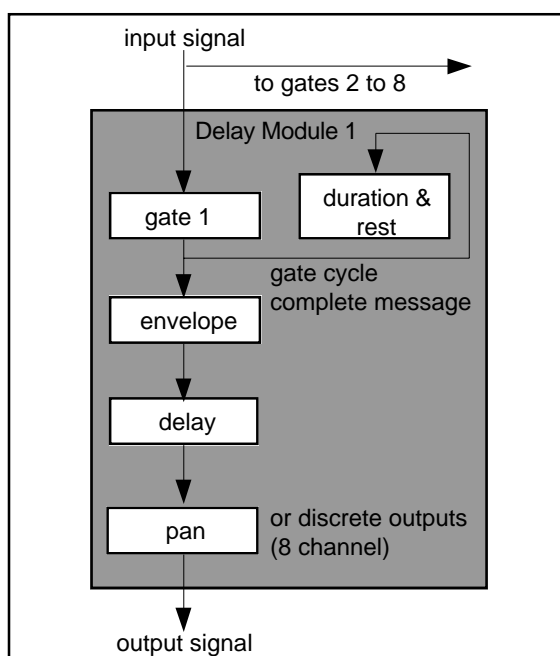


Figure 4. The signal path of the various components of Fractured Delay.

4.2 Envelope

An envelope generator is included in each delay module. The function of the envelope generator is to alter the amplitude envelope of the input signal. When a gate is opened, a message is sent to trigger the start of the envelope generator. The total duration of the envelope is equal to the duration time (the on time of the gate). Control of the envelope shape is done using the standard Max/MSP *function* object that allows for the creation of multiple breakpoint envelopes.

4.3 Delay

Each delay line has a maximum delay of 6 sec. The design of the delay line incorporates a feedback function and delay time modulation using a sine wave oscillator. These functions are controlled by four parameters; 1) delay time, 2) feedback, 2) modulation frequency, and 4) modulation depth.

As there are 8 separate delay lines, there is a total of 32 parameters that must be set. An interface is provided for this using simple number box entry. Due to the large number of parameters this interface is not practical for real-time control. Therefore an alternative interface has also been provided that sets the four parameters for all eight delay lines from four main number controls (one for each parameter). This approach utilises a random number generator that distributes varied values around the value set in the main control. The width of the distribution can also be set. This enables rapid control over all 32 parameters, which is very useful in live performance situations. The interface is illustrated in Figure 5 below.

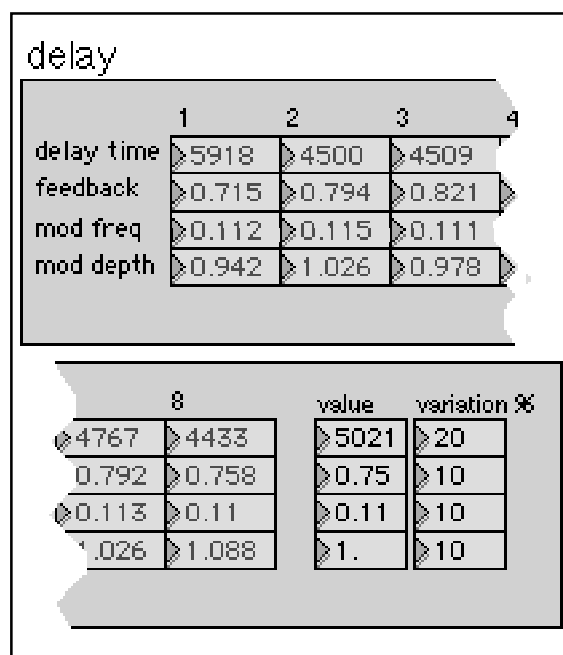


Figure 5. The control interface for the eight delay lines. The value number boxes set the centre value for the distribution of values to the corresponding parameter for each delay line. The variation number boxes set the width of the distribution.

4.4 Output

In the stereo version, Fractured Delay 2, the output of the delay lines are panned to left and right outputs. In the 8 channel version, Fractured Delay 8, the output of each delay is sent to one of the 8 output channels. Due to the action of the input gates, as described above, the input sound is distributed across the virtual stereo space or through an 8 channel speaker array. In order to ensure an even spatial distribution, the generation of duration and rest values is done using the random number module (as discussed above).

The sonic textures generated by Fractured Delay are various however they can be characterised as sounding similar to those created by delay loops. The difference however is that the textures generated by Fractured Delay are more complex and less repetitious. When very short intervals, in the 40 to 100 ms range are used for the duration and rest parameters, the process is similar to granular synthesis.

5 CubistAudio

One of the most significant techniques developed in the *Cubist* painting movement (as practiced by Picasso, Braques and Gris), was the presentation of multiple views of an object, or scene, in the one image (Hughes, 1980). This technique is a manifestation of the idea that our concept of reality is as much a function of memory as it is of instantaneous perception. For example our concept of the reality of an object is informed by the memory of multiple perspectives of that object and other similar objects. It is through our memories that we assemble and assimilate the multiple perspectives of our experience.

Unlike painting however, the cubist approach applied to sound, is much more difficult to achieve. This is because it is possible to visually represent simultaneous and discrete views of an object or scene whereas a sound is an object that must exist in time. Simultaneous and complete representations of the same sound are not possible. However parallel presentations of a sound object can be made and this approach is relatively common. The basic compositional technique of the *canon* is an example of this. Minimalist techniques can present multiple aspects of the sound or sound series. This is achieved by selectively presenting portions of a sound series

against other portions, as is found in techniques such as phasing and additive repetition.

CubistAudio explores some aspects of the cubist technique by simultaneously presenting different sections of the same sound and rendering them in different locations in a virtual space. As there are several similarities between the techniques used in musique concrète and cubist painting, the program employs some of these techniques as well. These include: retrogradation, pitch shifting, the isolation and reordering of events in a sound or series of sound objects, and the alteration of the amplitude envelope. As the same sonic material is presented in various forms, and positioned in a virtual space, the musical textures that are created have continuity as a sonic landscape².

5.1 Segment Selection

The program includes eight modules that select and play segments from an audio file that is loaded into a buffer. This enables a maximum of eight segments to be played simultaneously. A segment is defined by the:

- Start position within the audio file,
- Duration (of the segment),
- Pitch, or transposition (this influences the actual duration),
- Orientation – whether the segment is played forwards or backwards.

The start position is generated in two processes:

- 1) A scan factor is applied. When the program is run, the start position of each segment generator is at the beginning of the audio file. When the next segment is generated the start position is advanced along the timeline of the audio file. The distance that the start position is advanced by is determined by the scan factor (a user controlled variable that is measured as a percentage of the total length of the audio file).
- 2) The position determined by the scan factor is randomized by the use of the *wprob10* object (see 5.3 below).

The maximum length of a segment is equal to the total length of the audio file. Segment durations are

² For further discussion on the issue of sonic landscapes, see Spatialised Sound: the listener's perspective (McIlwain, 2001) and On Sonic Art (Wishart, 1996)

generated stochastically (see section 5.3) with an upper limit that is set by the user and is measured as a percentage of the total length of the audio file. As used by the author, segment lengths tend to be in the range of 10 – 15% of audio files that are around 20 seconds long.

While the process of segmenting a sound object has similarities with that used in granular synthesis the effect is different due to the longer intervals involved. In granular approaches, segment durations tend to be around the threshold at which a listener can discern successive separate events, therefore combinations of grains aggregate into a continuous timbre. With longer segments the identity of the segment, as a sound object, is retained so that the aggregations tend to be heard as textures consisting of several streams.

5.2 Segment Processing and Output

After the segment is selected, it is played and altered according to several processes. Firstly, the amplitude envelope is altered according to a user-controlled interface (featuring the Max/MSP function object). Secondly the segments are processed for positioning in a virtual space using the *haaspan* object (see 5.4 below). Lastly the output from the program can either be recorded or sent to monitoring equipment only.

5.3 wprob10

Most of the parameters for segment selection and spatialisation are generated using the *wprob10* object (which is created by the author). This object generates random values in 10 bands. Values within a band have an equal probability of being generated, however the bands are selected according to probability weights. Therefore the generation of values by *wprob10* is a two step process:

- 1) a band is selected based on weightings assigned by the user,
- 2) a value within the range of the selected band is generated using a simple random number generator.

Figure 6. illustrates the user interface for *wprob10* together with a graph illustrating the frequency distribution for all possible values that are generated from various settings of the weightings. The height of the bars (which can be altered by the user) determines the relative weightings for the various bands. The use

of weights in the generation process described above enables a wide range of distribution patterns to be generated.

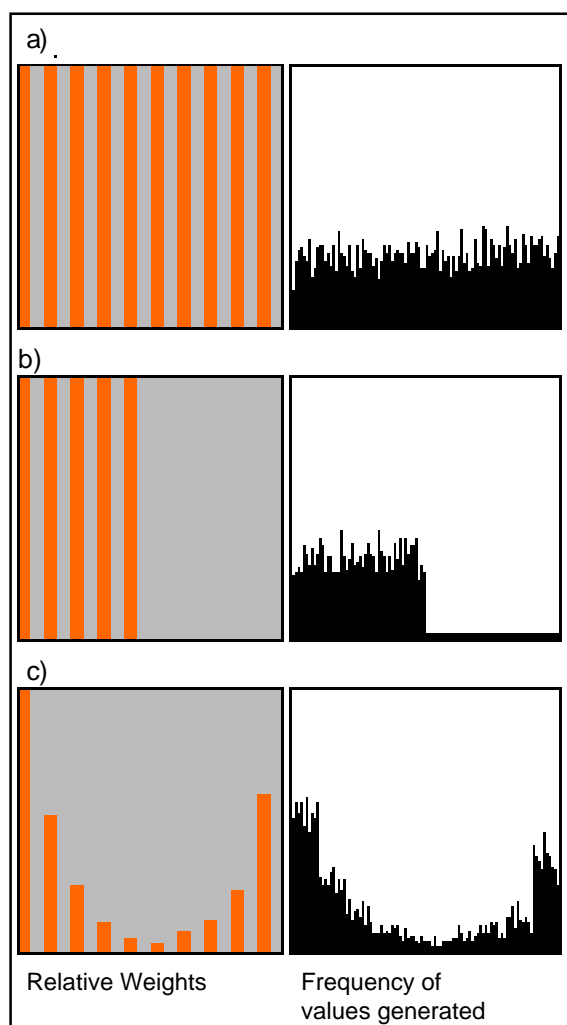


Figure 6. Various distributions patterns for *wprob10*. a) even distribution, b) limited distribution, and c) weighted distribution.

5.4 haaspan

Each of the eight segment generators uses an instance of the *haaspan*³ object to position the playback of the segment in a stereo virtual space. The *haaspan* object is created by the author according to the design outlined in “Spatialisation of Sounds over Loudspeakers” (Moore, 1989). It creates a panning effect via interaural time difference and through changes in amplitude. There are two user controlled parameters, *x* and *y*, that determine the position of the

³ The name of this object is a reference to the *haas effect* after Haas who is credited with discovering the spatial effect described in this paper (Haas. 1951).

sound in a cartesian space. A third variable, *speaker width*, controls the distance that the loudspeakers are apart. Figure 7 shows the relationship of the virtual space created by haaspan to the position of the loudspeakers.

The object also includes a reverb send that is scaled according to distance. The scale adopted is inversely proportional so that the balance between direct and reverberated sound is altered in favor of reverberation the further away a sound is located from the speakers. Reverb is created using a VST plugin, the one currently used in CubistAudio is Freeverb (Jezar, 2000).

Apart from dynamic variations in the original audio file, the amplitude scaling inherent in the spatialisation process used in haaspan is primary way in which dynamic variation occurs.

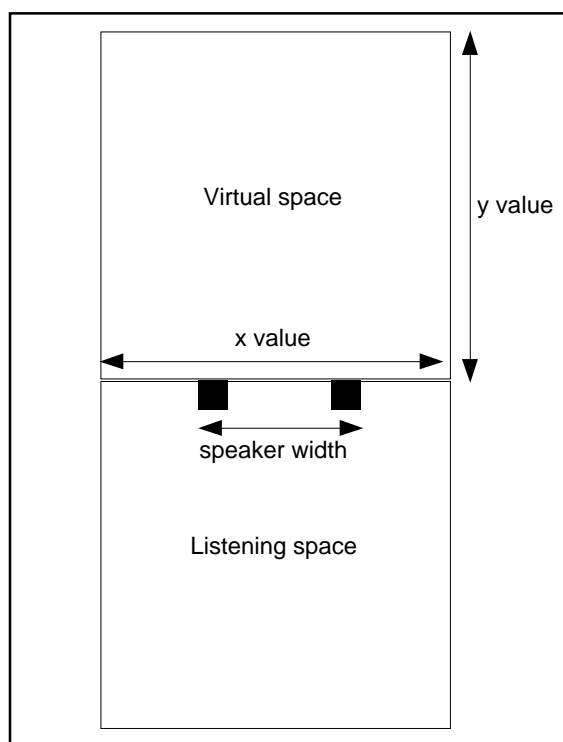


Figure 7. The relationship between the three user variables in haaspan and virtual space.

5.5 Landscapes

The textures that are generated by Cubist Audio have similarities with those that occur as a result of Minimalist techniques. Here textures are characterised by recurring gestures that are combined in a variety of ways by processes such as the addition, subtraction, re-ordering, and displacement of notes, and the scaling of temporal elements. The textures

generated by the software however are significantly different in two respects; 1) metrical structures are not adhered to so that they have more in common with the *micropolyphonic* textures of Ligeti or Lutoslawski's *aleatoric counterpoint*, and 2) the elements in the texture are spatialised. The resultant textures from the process tend to create a unified sonic landscape inhabited by aggregations of like gestures (as they are all sourced from the same audio file). For example, if the source audio file is a recording of a dog barking, the resultant texture (if segments are in the 1 to 2 second range) will sound like a field of dogs distributed around the virtual space.

6 Acknowledgments

Thanks to Paul Doornbusch for knowledge and assistance in a number of ways and to the members of the Sonic Art Group; Matt Baxter, Tom Dunstan, Eden Krumins, Kim Lajoie, Andrew Ross, Clint Small and Simon.

References

- Castine, P. 1997 "The Litter Package" *Peter Castine's Software*
<http://www.bek.no/~pcastine/Music/programs.html>
 Accessed 29/04/02
- Haas, H. 1951. "Über den Einfluss eines Einfahechos auf die Hörsamkeit von Sprache" *Acustica* 1:49-58.
- Hilton, G. *Electronic Studios Home Page*,
<http://www.ems-synthi.demon.co.uk/> Accessed 6/5/02.
- Hughes, R. 1980 *The Shock of the New*. London, British Broadcasting Corporation. 20
- Jezar, 2000. *Freeverb*, <http://www.dreampoint.co.uk>
 Accessed 6/5/02.
- Kartomi, M.J., R. Ornstein. 1984. "Gendèr" In S. Sadie (Ed.) *The New Grove Dictionary of Musical Instruments*, London: Macmillan Press. Vol. 2: 35.
- Mcilwain, P. A. 2001 "Spatialised Sound:the listener's perspective" *Proceedings of the Australasian Computer Music Conference 2001 ACMA Melbourne*. 57-65
- Moore, F. R. 1989. "Spatialization of Sounds over Loudspeakers" in M. V. Mathews & J. R. Pierce (Eds) *Current Directions in Computer Research*, Cambridge, Massachusetts: MIT Press. 89-103.
- Wishart, T. 1996 *On Sonic Art*, Harwood, Amsterdam

Appendix: Screen shots of the software

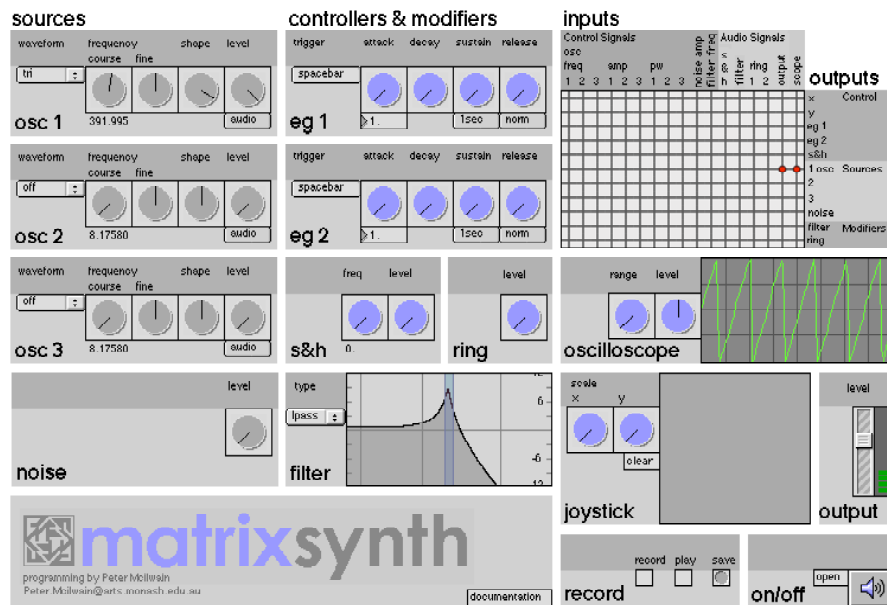


Figure 9. The interface for Matrixsynth-2.

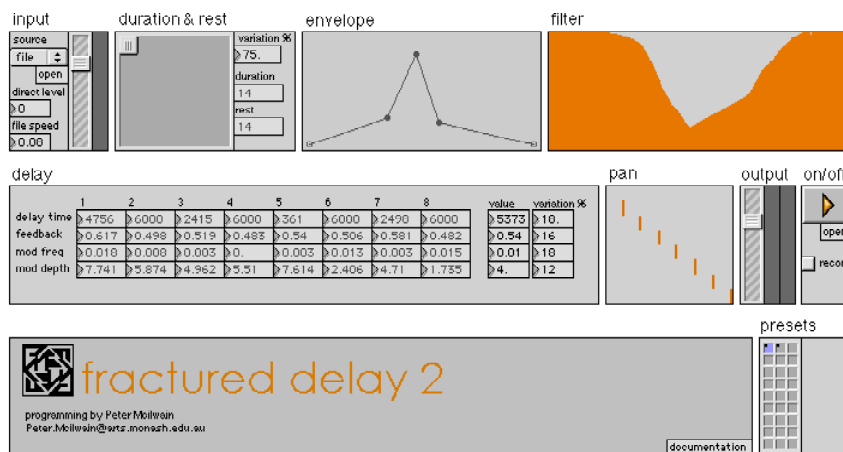


Figure 10. The interface for Fractured Delay 2.

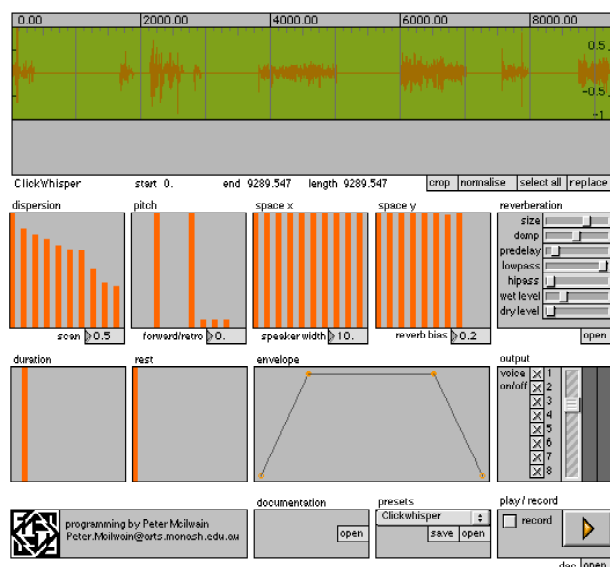


Figure 11. The interface for CubistAudio.

