3 Granular Synthesis

> *To stubbornly conditioned ears, anything new in music has always been called noise. But after all, what is music but organized noises?*
> —Edgard Varèse (1962)

Digital sound synthesis techniques inhabit a virtual world more pure and precise than the physical world, and purity and precision have an undeniable charm in music. In the right hands, an unadorned sine wave can be a lush and evocative sonority. A measured pulsation can invite emotional catharsis. Synthesis, however, should be able to render expressive turbulence, intermittency, and singularity; the overuse of precision and purity can lead to sterile music. Sonic grains, and techniques used to scatter the grains in evocative patterns, can achieve these results.

This chapter is devoted entirely to granular synthesis (GS). I present its theory, the history of its implementations, a report on experiments, and an assessment of its strengths and weaknesses. A thorough understanding of the principles of granular synthesis is fundamental to understanding the other techniques presented in this book. This chapter focuses on synthesis with synthetic waveforms. Since granulation transforms an existing sound, I present the granulation of sampled sound in chapter 5 with other particle-based transformations.

Theory of Granular Synthesis

The seeds of granular synthesis can be traced back to antiquity, although it was only after the papers of Gabor and Xenakis that these seeds began to take root (see chapter 2). A grain of sound is a brief microacoustic event, with a duration near the threshold of human auditory perception, typically between one thousandth of a second and one tenth of a second (from 1 to 100 ms). Each grain contains a waveform shaped by an amplitude envelope (figure 3.1).

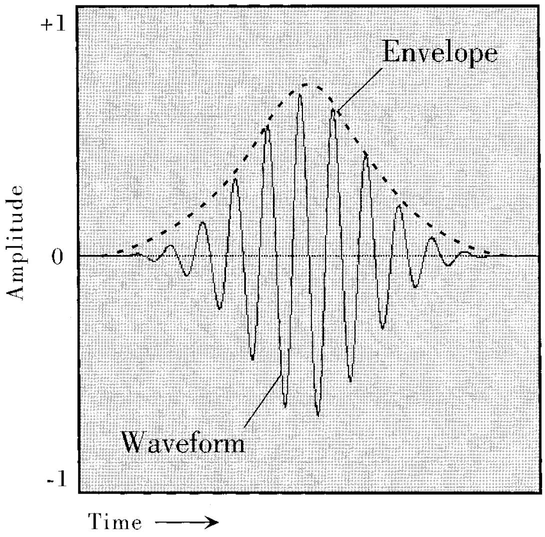

Figure 3.1 Portrait of a grain in the time domain. The duration of the grain is typically between 1 and 100 ms.

A single grain serves as a building block for sound objects. By combining thousands of grains over time, we can create animated sonic atmospheres. The grain is an apt representation of musical sound because it captures two perceptual dimensions: time-domain information (starting time, duration, envelope shape) and frequency-domain information (the pitch of the waveform within the grain and the spectrum of the grain). This stands in opposition to sample-based representations that do not capture frequency-domain information, and abstract Fourier methods, which account only for the frequency domain.

Granular synthesis requires a massive amount of control data. If *n* is the number of parameters per grain, and *d* is the density of grains per second, it takes *n* times *d* parameter values to specify one second of sound. Since *n* is usually greater than ten and *d* can exceed one thousand, it is clear that a global unit of organization is necessary for practical work. That is, the composer specifies the sound in global terms, while the granular synthesis algorithm fills in the details. This greatly reduces the amount of data that the composer must supply, and certain forms of granular synthesis can be played in real time with simple MIDI controllers. The major differences between the various granular techniques are found in these global organizations and algorithms.

Anatomy of a Grain

A grain of sound lasts a short time, approaching the minimum perceivable event time for duration, frequency, and amplitude discrimination (Whitfield 1978; Meyer-Eppler 1959; Winckel 1967). Individual grains with a duration less than about 2 ms (corresponding to fundamental frequencies > 500 Hz) sound like clicks. However one can still change the waveform and frequency of grains and so vary the tone color of the click. When hundreds of short-duration grains fill a cloud texture, minor variations in grain duration cause strong effects in the spectrum of the cloud mass. Hence even very short grains can be useful musically.

Short grains withhold the impression of pitch. At 5 ms it is vague, becoming clearer by 25 ms. The longer the grain, the more surely the ear can hear its pitch.

An amplitude envelope shapes each grain. In Gabor's original conception, the envelope is a bell-shaped curve generated by the Gaussian method (figure 3.2a).

$$p(x) = \frac{1}{\sqrt{2^{\pi}}} e^{-x^2/2}\, dx$$

A variation on the pure Gaussian curve is a *quasi-Gaussian envelope* (Roads 1978a, 1985), also known as a *cosine taper* or *Tukey window* (Harris 1978). This envelope can be imagined as a cosine lobe convolved with a rectangle (figure 3.2b). It transitions smoothly at the extrema of the envelope while maximizing the effective amplitude. This quality persuaded me to use it in my earliest experiments with granular synthesis.

In the early days of real-time granular synthesis, it was necessary to use simple line-segment envelopes to save memory space and computation time (Truax 1987, 1988). Gabor (1946) also suggested line-segment envelopes for practical reasons (figure 3.2c and d). Keller and Rolfe (1998) have analyzed the spectral artefacts introduced by a line-segment trapezoidal window. Specifically, the frequency response is similar to that of a Gaussian window, with the addition of comb-shaped spectral effects. Null points in the spectrum are proportional to the position of the corners of the window.

Figure 3.2e portrays another type of envelope, the band-limited pulse or sinc function. The sidelobes (ripples) of this envelope impose a strong modulation effect. The percussive, exponentially decaying envelope or *expodec grain*

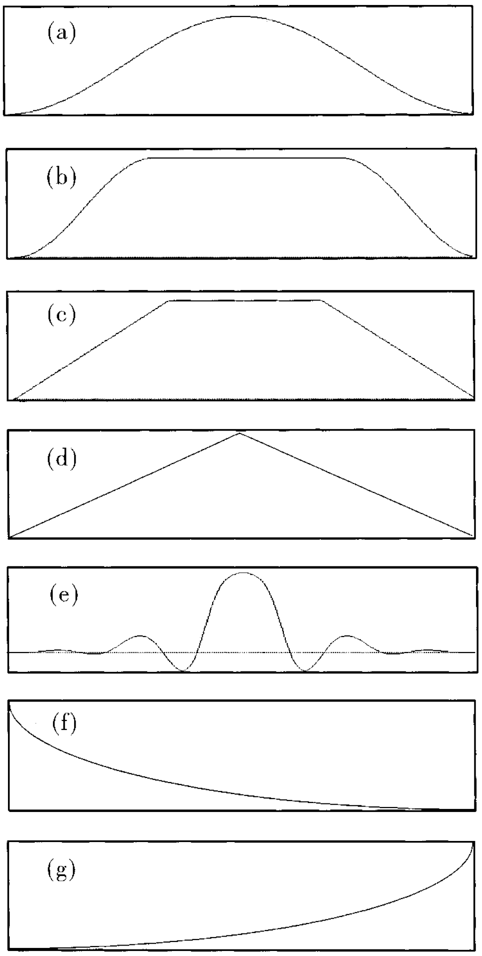

Figure 3.2 Grain envelopes. (a) Gaussian. (b) Quasi-Gaussian. (c) Three-stage line segment. (d) Triangular. (e) Sinc function. (f) Expodec. (g) Rexpodec.

(figure 3.2f) has proven to be effective in transformations such as convolution (described in chapter 5). Figure 3.2g depicts the reversed expodec or *rexpodec* grain. Later we study the strong effect the grain envelope imposes on the spectrum.

The grain envelope and duration can vary in a frequency-dependent manner (shorter envelopes for high frequency sounds); such a correlation is characteristic of the wavelet transform (see chapter 6), and of the grainlet synthesis technique described in chapter 4.

The waveform within the grain is an important grain parameter. It can vary from grain to grain, be a fixed waveform that does not change over the grain's duration, or it can be a time-varying waveform. Typical fixed waveforms include the sine wave and sums of sine waves with increasing harmonic content up to a bandlimited pulse. A time-varying waveform can be generated by frequency modulation or another mathematical technique (Jones and Parks 1988). The grain waveform can also be a single period extracted from a recorded sound. This differs from granulation (chapter 5), which scans across a long sampled waveform, extracting many different grains over time.

Other parameters of the grain include its duration, the frequency of its waveform, its amplitude coefficient, and its spatial location. We examine these parameters later.

The Grain Generator

In its most basic form, the grain generator is a simple digital synthesis instrument. Its circuit consists of a wavetable oscillator whose amplitude is controlled by a Gaussian envelope. The output of the oscillator goes to a spatial panner (figure 3.3).

As a general principle of synthesis, we can trade off instrument complexity for score complexity. A simple instrument suffices, because the complexity of the sound derives from the changing combinations of many grains. Hence we must furnish a massive stream of control data for each parameter of the instrument.

Despite the simplicity of the instrument, we gain two spectral controls for free, owing to the specific properties of the micro time scale. Specifically, variations in the grain duration and the grain envelope affect the spectrum of the resulting signal (as a later of this book section clarifies).

Of course, we can always make the granular synthesis instrument more complex, for instance by adding a local frequency control, per-grain reverberation,

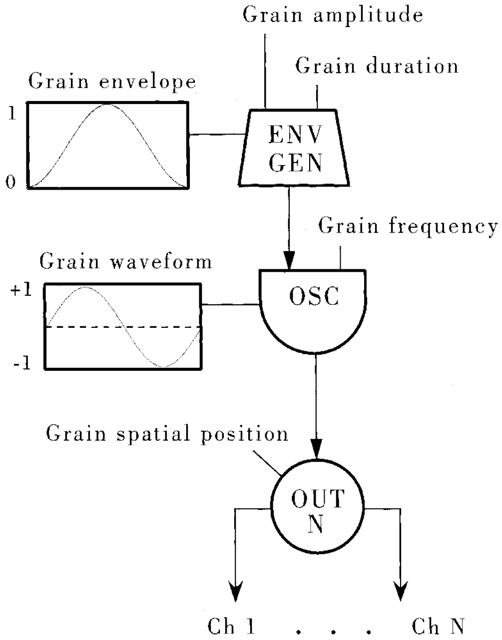

Figure 3.3 The simplest grain generator, featuring a Gaussian grain envelope and a sinusoidal grain waveform. The grains can be scattered to a position in *N* channels of output.

multichannel output, etc. Chapter 4 describes several extensions to the basic granular instrument.

Global Organization of the Grains

The main forms of granular synthesis can be divided into six types, according to how each organizes the grains. They are:

- Matrices and screens on the time-frequency plane
- Pitch-synchronous overlapping streams
- Synchronous and quasi-synchronous streams
- Asynchronous clouds
- Physical or abstract models
- Granulation of sampled sound

Matrices and Screens on the Time-Frequency Plane

The Gabor matrix, shown in chapter 2, is the original time-frequency matrix for sound, based on the analysis of an existing sound. In the same general family are the analyses produced by the short-time Fourier transform (STFT) and the wavelet transform (WT), presented in chapter 6. These operations transform a time-domain signal into a frequency-domain representation that is quantized in both the time and frequency dimensions, creating a two-dimensional matrix. Frequency-domain matrices offer many opportunities for sound transformation.

A related organizational scheme is Xenakis's (1960, 1971) notion of screens (also described in chapter 2). Each screen is like a snapshot of a microsound. It represents a Gabor matrix at a specific moment, divided into cells of amplitude and frequency. Like the frames of a film, a synchronous sequence of screens constitutes the evolution of a complex sound. Rather than starting from an analyzed sound, Xenakis proposed to fill the screen with grains by means of stochastic algorithms. Another proposal suggested that the grains be generated from the interaction of cellular automata (Bowcott 1989; Miranda 1998).

Pitch-Synchronous Granular Synthesis

Pitch-synchronous granular synthesis (PSGS) is an efficient analysis-synthesis technique designed for the generation of pitched sounds with one or more formant regions in their spectra (De Poli and Piccialli 1991). It begins with a spectrum analysis of a sound. Each time-frequency cell corresponds to a grain. As a preparation for resynthesis, at each grain boundary along the frequency axis, a standard algorithm derives the coefficients for a filter. The impulse response of this filter corresponds to the frequency response of the cell. At each grain boundary along the time axis, a pitch detection algorithm determines the fundamental pitch period. In resynthesis, a pulsetrain at the detected frequency drives a bank of parallel minimum-phase finite impulse response filters. The musical signal results from the excitation of the pulsetrain on the weighted sum of the impulse responses of all the filters. At each grain time frame, the system emits a waveform that is overlapped with the previous grain to create a smoothly varying signal. An implementation of PSGS by De Poli and Piccialli features several transformations that can create variations of the original sound. A focus of their implementation was the use of data reduction techniques to save computation and memory space. See De Poli and Piccialli (1991), and Cavaliere and Piccialli (1997) for details.

Synchronous and Quasi-Synchronous Granular Synthesis

> *Granular streams appear naturally from iterative sound production—any kind of roll or trill on drums, percussion, or any sounding material. They are produced vocally by rolled "r" sounds.* (Wishart 1994)

In *synchronous granular synthesis* (SGS), sound results from one or more streams of grains. Within each stream, one grain follows another, with a delay period between the grains. "Synchronous" means that the grains follow each other at regular intervals.

An excellent use of SGS is to generate metric rhythms, particularly when the grain emissions are sparse per unit of time. The density parameter controls the frequency of grain emission, so *grains per second* can be interpreted as a frequency value in Hertz. For example, a density of 1 grain per second indicates that a grain is produced every second. Synchronous densities in the range of about 0.1 to 20 grains per second will generate metrical rhythms. When densities change over time, the listener hears accelerandi and rallentandi.

At higher densities, the grains fuse into continuous tones. Here is found the sweeter side of granular synthesis. These tones have a strong fundamental frequency, and depending on the grain envelope and duration, may also exhibit sidebands. The sidebands may sound like separate pitches or they may blend into a formant peak. At certain settings, these tones exhibit a marked vocal-like quality. In these cases, SGS resembles other techniques such as FOF and Vosim synthesis. Chapter 4 describes these particle-based formant synthesis methods.

The formant shape and strength depend greatly on the grain duration and density which also, under certain conditions, affect the perceived fundamental frequency. We explore this in more detail in the next section.

In *quasi-synchronous granular synthesis* (QSGS), the grains follow each other at unequal intervals, where a random deviation factor determines the irregularity. If the irregularity is great, the sounds produced by this method become similar to those produced by asynchronous granular synthesis. SGS and QSGS are well-adapted to real-time implementations.

Pitch and Noise Perception in Synchronous Granular Synthesis

The fragility of the illusion of pitch is made apparent in SGS. The perceived pitch of a granular stream depends primarily on interactions among three periodicities:

a is the period corresponding to the frequency of the waveform in the grain

b is the period corresponding to the grain envelope

c is the period corresponding to the density, or rate of synchronous grain emission

One or another of these factors may override the others in determining the perceived pitch. In certain cases, modulations caused by their interaction may render the pitch—and especially the specific octave—ambiguous.

Figure 3.4 illustrates these effects. Consider a tone made up of a series of 10 ms grains, where each grain contains two periods of a 200 Hz sine wave. Assume, as in figure 3.4a, that the density of the grains is 50 per second. Here we have:

$a = 5$ ms

$b = 10$ ms

$c = 20$ ms

As figure 3.4a shows, the 10 ms gap between b and c means that there is a *dead interval* between successive grains, leading to a modulation effect with its associated sidebands. The perceived pitch is a buzzy 100 Hz.

A linear increase in grain density (from 50 to 100 grains per second in the above case) causes a pitch doubling effect. The perceived pitch is now 200 Hz. Here the three variables take the values:

$a = 5$ ms

$b = 10$ ms

$c = 10$ ms

In figure 3.4c, the grain density is 200 grains per second. The variables take these values:

$a = 5$ ms

$b = 10$ ms

$c = 5$ ms

Now we have a pure sinusoidal tone. Only one period of the waveform can unfold within the grain repetition period c, and the influence of the grain envelope b is diminished.

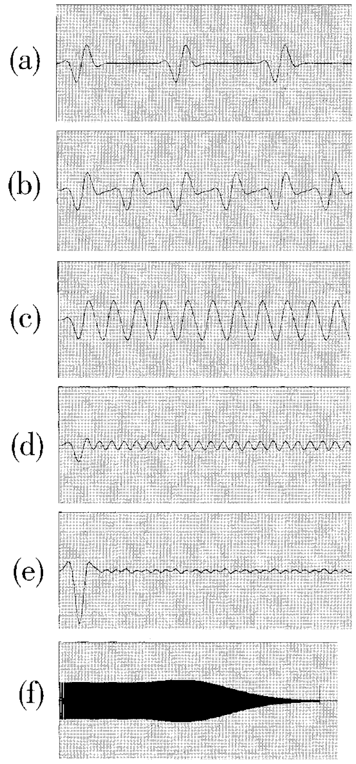

Figure 3.4 Influence of grain density on pitch. The waveforms in (a) through (e) last 59 ms. (a) 50 grains/sec. (b) 100 grains/sec. (c) 200 grains/sec. (d) 400 grains/sec. (e) 500 grains/sec. (f) Plot of a granular stream sweeping from the infrasonic frequency of 10 grains/sec to the audio frequency of 500 grains/sec over thirty seconds.

When the grain density increases to 400 grains per second (figure 3.4d), the perceived pitch doubles to 400 Hz. This is due to the increasing frequency of wavefronts (as in the well-known Doppler shift effect). Notice that the amplitude of tone diminishes after beginning, however, because the density period c is less than the waveform period a. Only the first few samples of the product of the sine wavetable and the grain envelope are being repeated, resulting in a low-amplitude signal.

Finally, at a density of 500 grains per second (figure 3.4e), the signal has almost no amplitude. It is reading only the first few samples of the sinusoid, which are near zero.

Figure 3.4f shows the amplitude profile of a granular stream that sweeps from 10 grains per second to 500 grains per second over thirty seconds. Notice the diminution of amplitude due to the effect shown in figure 3.4e.

Besides pitch changes, other anomalies, such as phase cancellation, can occur when the grain density and envelope duration are at odds with the frequency of the grain waveform.

Even the impression of synchronicity can be undermined. If we widen the frequency limits of a dense synchronous stream slightly, the result quickly truns into a noiseband. The fact that the grain emissions are regular and the frequency changes at regular intervals (for example, every 1 ms), does not alter the general impression of noise. The effect is similar to that produced by asynchronous granular synthesis, described next.

Asynchronous Granular Synthesis

Asynchronous granular synthesis (AGS) abandons the concept of linear streams of grains. Instead, it scatters the grains over a specified duration within regions inscribed on the time-frequency plane. These regions are clouds—the units with which the composer works. The scattering of the grains is irregular in time, being controlled by a stochastic or chaotic algorithm. The composer may specify a cloud with the following parameters:

1. Start-time and duration of the cloud
2. Grain duration—may vary over the duration of the cloud
3. Density of grains per second, with a maximum density depending upon the implementation; density can vary over the duration of the cloud
4. Frequency band of the cloud; specified by two curves forming high and low frequency boundaries within which grains are scattered; alternatively, the frequency of the grains in a cloud can be restricted to a specific set of pitches
5. Amplitude envelope of the cloud
6. Waveform(s) within the grains
7. Spatial dispersion of the cloud, where the number of output channels is implementation-specific

The grain duration (2) can be a constant (in milliseconds), or a variable that changes over the course of a cloud. (It can also be correlated to other parameters, as in the grainlet synthesis described in chapter 4.) Grain duration can also be derived as a random value between an upper and a lower boundary set by the user. The next section explains the effects of different grain durations in more detail.

Parameter (3), grain density, specifies the number of grains per unit of time. For example, if the grain density is low, then only a few grains are scattered at random points within the cloud. If the grain density is high, grains overlap to create rich, complex spectra.

Parameter (6) is one of the most flexible cloud parameters, since each grain may have a its own waveform.

Physical and Algorithmic Models

Physical modeling (PhM) synthesis starts from a mathematical description of acoustic sound production (Roads 1996; Fletcher and Rossing 1991). That is, the equations of PhM describe the mechanical and acoustical behavior of an instrument as it is played. An example of physical modeling applied to granular synthesis is Perry Cook's Physically Informed Stochastic Event Modeling (PhISEM). This suite of programs simulates the sounds of shaken and scraped percussion such as maracas, sekere, cabasa, bamboo windchime, tambourine, sleighbells, and guiro (Cook 1996, 1997). See more about this technique in chapter 4. Going beyond traditional instruments, Keller and Truax created sound models of such physical processes as the bounce of a metallic ball and the rush of a stream of water (Keller and Truax 1998). Physical models of granular processes could be taken still further. A large body of scientific literature centers on granular processes such as grain mixing, granular flow, grain vibration patterns, and grain and fluid interactions. This literature appears in research periodicals such as *Granular Matter, Powders and Grains, Powder Technology, Journal of Fluid Mechanics, Physical Review, Journal of Applied Mechanics*, as well as such Internet web sites as www.granular.com.

Going beyond emulations of the natural world, one can also develop models of virtual worlds through abstract algorithms. To cite an example, Alberto de Campo (1998) proposed a method of grain scattering in time based on a recursive substitution algorithm. Another idea would be using *chaotic functions* to scatter the grains in time (Gleick 1988; Holden 1986; Moon 1987). Chaotic functions produce different results from scattering algorithms based on pseudorandom algorithms. The values produced by pseudorandom number generators tend to be uniform and adirectional, tending toward the mean. To make them directional, they must be filtered through stochastic weightings. In contrast, chaotic functions vacillate between stable and unstable states, between intermittent transients and full turbulence (Di Scipio 1990, 1997b; Gogins 1991, 1995; Miranda 1998). The challenge is to set up a musically compelling mapping between chaotic behavior and the synthesis parameters.

Streams and Clouds of Granulated Samples

The granulation of sampled sounds is a powerful means of sound transformation. To granulate means to segment (or window) a sound signal into grains, to possibly modify them in some way, and then to reassemble the grains in a new time order and microrhythm. This might take the form of a continuous stream or of a statistical cloud of sampled grains.

The exact manner in which granulation occurs will vary from implementation to implementation. Chapter 5 includes a major section on granulation so here we shall limit the discussion to noting, that granulation can be controlled by any of the global control structures described above.

Spectra of Granular Streams

When the intervals between successive grains are equal, the overall envelope of a stream of grains forms a periodic function. Since the envelope is periodic, the signal generated by SGS can be analyzed as a case of *amplitude modulation* or AM. AM occurs when the shape of one signal (the *modulator*) determines the amplitude of another signal (the *carrier*). From a signal processing standpoint, we observe that for each sinusoidal component in the carrier, the periodic envelope function contributes a series of *sidebands* to the final spectrum. (Sidebands are additional frequency components above and below the frequency of the carrier.) The sidebands separate from the carrier by a distance corresponding to the inverse of the period of the envelope function. For grains lasting 20 ms, therefore, the sidebands in the output spectrum will be spaced at 50 Hz intervals. The shape of the grain envelope determines the precise number and amplitude weighting of these sidebands.

The result of modulation by a periodic envelope is that of a formant surrounding the carrier frequency. That is, instead of a single line in the spectrum (a single frequency), the spectrum looks like a sloping peak (a group of frequencies around the carrier). In the case of a bell-shaped Guassian envelope, the spectrum is similarly bell-shaped. In other words, for a Gaussian envelope, the spectrum is an *eigenfunction* of the time envelope.

When the delay interval between the grains is irregular, perfect grain synchronization disappears. The randomization of the onset time of each grain leads to a controllable thickening of the sound spectrum—a "blurring" of the formant structure (Truax 1988).

In its simplest form, the variable-delay method is similar to amplitude modulation using low-frequency colored noise as a modulator. In itself, this is not particularly new or interesting. The granular representation, however, lets us move far beyond simple noise-modulated AM. We can simultaneously vary several other parameters on a grain-by-grain basis, such as grain waveform, amplitude, duration, and spatial location. On a global level, we can also dynamically vary the density of grains per second, creating a variety of scintillation effects.

Parameters of Granular Synthesis

Research into sound synthesis is governed by aesthetic goals as much as by scientific curiosity. Some of the most interesting synthesis techniques have resulted from applied practice, rather than from formal theory. Sound design requires taste and skill and at the experimentation stage, musical intuition is the primary guide.

Grain Envelope Shape Effects

Of Loose Atomes
In every Braine loose Atomes there do lye,
Those which are Sharpe, from them do Fancies flye.
Those that are long, and Aiery, nimble be.
But Atomes Round, and Square, are dull, and sleepie.
(Margaret Cavendish 1653)

This section presents empirical reports on the effects caused by manipulating the grain envelope, duration, waveform, frequency, band, density, and spatial parameters.

Referring back to figure 3.2, the classical grain envelope is the bell-shaped Gaussian curve (figure 3.2a). This is the smoothest envelope from a mathematical point of view. A quasi-Gaussian (Tukey) envelope retains the smooth attack and decay but has a longer sustain portion in the envelope and so increases its perceived amplitude (figure 3.2b). Compared to a pure Gaussian of the same duration, the quasi-Gaussian broadens the spectrum. Its highest side-lobe is only −18 dB down, as opposed to −42 dB for a pure Gaussian curve (Harris 1978). The band-limited pulse or sinc function imposes a strong modulation effect (figure 3.2e). I have used it myself to create "bubbling" or "frying" clouds.

I have carried out numerous experiments using grain envelopes with a sharp attack (typically less than 10 ms) and an exponential decay. These are the expodec grains (figure 3.2f). The percussive attack of the expodec articulates the rhythmic structure. As chapter 5 describes, clouds of expodec grains can be especially useful as impulse responses for convolution with other sounds. Reversed expodec or rexpodec grains feature a long attack envelope with a sudden decay (figure 3.2g). Granulated concrète sounds appear to be "reversed" when they are played with rexpodec grains, even though they are not.

While keeping the envelope shape constant, a change in the grain duration has a strong effect on the spectrum. Furthermore, the grain envelope and duration can vary in a frequency-dependent manner (shorter envelopes for high frequency sounds); such a correlation is characteristic of the wavelet transform and the grainlets.

Experiments in Time Reversal

The concept of time has been thoroughly twisted by modern theoretical physics (Kaku 1995). Barry Truax (1990b) drew an analogy between the world of particle physics—in which time appears to be reversible at the quantum level—and granular synthesis. According to his hypothesis, if a grain is reversed in time, it should sound the same. Moreover, granular synthesis textures should also be reversible. Such a position would also follow on from Trevor Wishart's assertion:

> *Although the internal structure of sounds is the cause of what we hear, we do not resolve this internal structure in our perception. The experience of a grain is indivisible.* (Wishart 1994)

Under special circumstances, all of this is quite true. But if we loosen any one of a number of constraints, time reversibility does not hold. For it to hold at the micro scale, the grain envelope must be symmetrical. This, then, excludes asymmetric techniques such as FOF grains (Rodet 1980), trainlets (chapter 4), expodec, or rexpodec grains. The grain waveform must not alter in time, so excluding techniques such as the time-varying FM grains (Jones and Parks 1988), long glissons (chapter 4), or grains whose waveform derives from a time-varying sampled sound. With texture a grain stream or cloud is time-reversible only if it is stationary in the statistical sense, meaning that its overall amplitude and spectral envelopes are symmetrical, its internal density is constant, and the waveform of all the grains is similar.

Grain Duration Effects

The duration of the grains inside a cloud has profound effects on the resulting audio signal. Within clouds, there are four classes of grain durations:

1. *Constant duration* the duration of every grain in the cloud is the same.
2. *Time-varying duration* the grain duration varies as a function of time.
3. *Random duration* the duration of a grain is random between an upper and lower duration boundaries.
4. *Parameter-dependent duration* the duration of a grain is tied to its fundamental frequency period, as it is in synthesis with wavelets, or any other parameter, as in the grainlet synthesis.

Regarding constant durations, early estimates of the optimum grain duration varied from 10 ms (Gabor 1946, 1947) to 60 ms (Moles 1968).

The grain envelope contributes an amplitude modulation (AM) effect. The modulation spawns sidebands around the carrier frequency of the grain at intervals of the envelope period. If the grain duration is D, the center frequency of the AM is $1/D$. In an asynchronous cloud, the AM sounds like an aperiodic fluttering tremolo when D is around 100 ms (table 3.1).

Table 3.1 Effects of grain durations in asynchronous granular synthesis

Grain duration	Frequency of modulation	Perceived effect
200 μsec	5 KHz	Noisy particulate disintegration
500 μsec	2 KHz	
1 ms	1 KHz	Loss of pitch
10 ms	100 Hz	Fluttering, gurgling
50 ms	20 Hz	Stable pitch formation
100 ms	10 Hz	
200 ms	5 Hz	Aperiodic tremolo, jittering

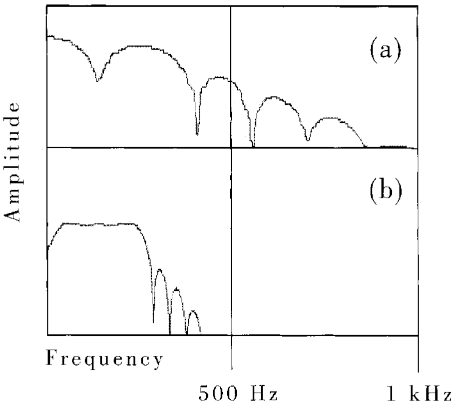

Figure 3.5 Comparison of grain spectra produced by a 7 ms grain duration (top) versus a 29 ms grain duration (bottom). Notice the narrowing of the spectrum as the duration lengthens.

The laws of micro-acoustics tell us that the shorter the duration of a signal, the greater its bandwidth. Thus the width of the frequency band B caused by the sidebands is inversely proportional to the duration of the grain D (figure 3.5).

A dramatic effect occurs when the grain duration is lowered to below the period of the grain waveform. This results in a signal that is entirely unipolar in energy, which is a byproduct of the ratio of the grain duration to the fundamental frequency period P_f of the grain waveform, or D/P_f. The effect is caused by an incomplete scan of the wavetable, where the waveform starts in either the positive or the negative quadrant. It occurs whenever D/P_f is less than 1.0. In the specific case of a 1 ms grain with a fundamental frequency of 500 Hz, the ratio is $0.001/0.002 = 1/2$.

To completely represent one period of a given frequency, the grain duration must be at least equal to the frequency period. If we took this criterion as a standard, grains could last no less than 50 ms (corresponding to the period of 20 Hz) for low frequency signal energy to be captured completely. As it happens however, much shorter grains can represent low frequency signals, but this short grain duration introduces modulation products. Our experiments show that grains shorter than 5 ms tend to generate particulated clouds in which a sense of center-pitch is still present but is diffused by noise as the frequency descends.

Grain Waveform Effects

One of the most interesting features of granular synthesis is that one can insert any waveform into a grain. The waveform can vary on a grain-by-grain basis. This makes possibile micro-animated textures that evolve directionally over time or simply scintillate from the effects of constantly changing grain waveforms.

The simplest grain waveforms are the fixed synthetic types: the sine, saw, square, and sinc (band-limited impulse). In early experiments, I used ten synthetic waveforms created by adding one to ten sine waves in a harmonic relationship.

Interspersing different waveforms in a single cloud leads to cloud *color type* (Roads 1991). Three possibilities for cloud color type are:

- *monochrome* containing a single waveform
- *polychrome* containing two or more waveforms
- *transchrome* the grain waveform evolves from one waveform to another over the duration of the cloud

For a monochrome cloud, we stipulate a single wavetable for the entire cloud. For a polychrome cloud, we specify two or more waveforms which can scatter uniformly in time or according to a time-varying tendency curve. For a transchrome cloud, if we specify a list of N waveforms, the cloud mutates from one to the next, through all N over its duration.

So far we have discussed waveform variations on the time scale of clouds. But even within a single grain, the waveform may be varying in time. The grain waveform could be generated by time-varying frequency modulation, for example. Since the duration of the grain is brief, however, such techniques tend to result in noisy, distorted textures unless the modulating frequencies and the amount of modulation are strictly controlled.

As a practical aside, it has been necessary to use the standard 44.1 or 48 kHz sampling rates for software and hardware compatibility in recording, synthesis, and playback. These sampling rates provide little "frequency headroom," and one must be aware that when the fundamental frequency of a grain is high and the waveform is complex, aliasing can occur. To avoid this, one can constrain the choice of waveform depending on the fundamental frequency, particularly in the region above half of the Nyquist frequency (11.025 or 12 kHz, depending on the sampling rate). Above these limits, waveforms other than sine cause foldover. For this reason, higher sampling rates are better for digital synthesis.

The grain waveform can also be extracted from a sampled sound. In this case, a single extracted waveform is fed to an oscillator, which reads the waveform repetitively at different frequencies. In Cloud Generator, for example, the extracted waveform constitutes the first 2048 samples (46 ms) of a selected sound file (see the appendix). This differs from granulation, which extracts many different segments of a long sample file. See chapter 5.

Frequency Band Effects

Frequency band parameters limit the fundamental frequencies of grain waveforms. Within the upper and lower boundaries of the band, the grain generator scatters grains. This scattering can be aligned to a frequency scale or to random frequencies. When the frequency distribution is random and the band is greater than a small interval, the result is a complex texture, where pitch is ambiguous or unidentifiable. The combined AM effects of the grain envelope and grain density strongly influence pitch and spectrum.

To generate harmonic texture, we can constrain the choice of fundamental frequency to a particular set of pitches within a scale. We distinguish two classes of frequency specifications:

Cumulus The frequencies of the grains scatter uniformly within the upper and lower bounds of a single band specified by the composer.

Stratus The frequencies of the grains align to a set of specified frequencies.

Figure 3.6 depicts a variety of frequency band specifications for granular clouds. When the band centers on a single frequency (figure 3.6a), the cloud produces a single pitch. The frequency can be changed to create a glissando (figure 3.6b). A stratus cloud contains multiple frequency specifications (figure 3.6c). With sampled soundfiles, one can achieve the harmonic effect of a stratus cloud by keeping a database of tones at all the pitches from a desired scale, or by pitch-shifting in conjunction with granulation. When the band is wider than a single pitch, grains scatter randomly between the upper and lower boundaries of a cumulus cloud (figure 3.6d). When the initial and final bands are different, the shape of the cumulus band changes over time (figure 3.6e). In the most flexible case, two time-varying curves shape the bandlimits of the cloud (figure 3.6f).

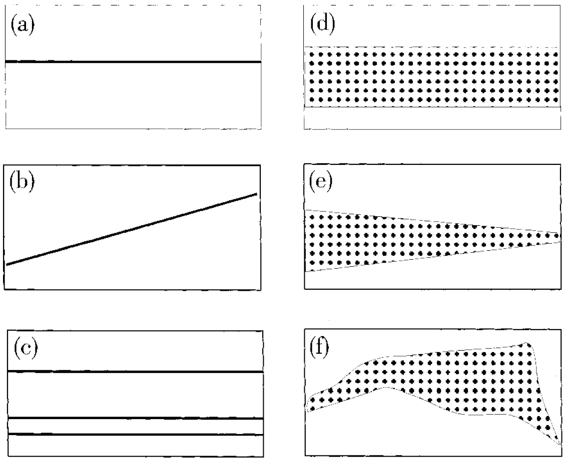

Figure 3.6 Frequency band specifications. (a) The band centers on a single frequency. (b) The center frequency changes over time, creating a glissando effect. (c) Stratus cloud with several frequencies. (d) Cumulus cloud where the grains scatter randomly between the upper and lower boundaries. (e) The shape of the cumulus band changes over time. (f) Time-varying curves shape the bandlimits of the cumulus cloud.

Density and Fill Factor

"Density" is the number of grains per second. If this specification is not linked with the grain duration, however, it tells us little about the resulting texture. Grain duration and density combined produce texture.

A one-second cloud containing twenty 100 ms grains is continuous and opaque, whereas a cloud containing twenty 1 ms grains is sparse and transparent. The difference between these two cases is their *fill factor* (*FF*). The fill factor of a cloud is the product of its density and its grain duration in seconds (*D*). In the cases just cited, the fill factor of the first cloud is $20 \times 0.1 = 2$, and of the second cloud $20 \times 0.01 = 0.2$. These are simple cases, where the density and grain duration are constants, in practice grain density and grain duration can vary over the duration of the cloud. In this case we derive the *average density* and the *average fill factor*, calculated as the mean between any two extremes. These measurements provide these descriptors of fill factor:

- *Sparse FF* < 0.5, more than half the cloud is silence
- *Covered FF* ~ 1.0, the cloud is more-or-less filled by sonic grains
- *Packed FF* > 1.0, the cloud is filled with overlapping grains

In asynchronous granular synthesis, the starting time of a grain is random. One cannot guarantee that fifty 20 ms grains will completely fill a one-second cloud. Some grains may overlap, leaving silences at other points in the cloud. To create what we hear as a solid cloud, a good rule of thumb is to set the density per second of the cloud to at least $2/D$. Hence, for 20 ms grains, it takes about 100 to cover a one-second cloud. Tiny gaps (less than about 50 ms) do not sound as silences, but rather as momentary fluctuations of amplitude.

For a typical grain duration of 25 ms, we can make the following observations concerning grain density as it crosses perceptual thresholds:

<15 grains per sec—Rhythmic sequences.

15–25 grains per sec—Fluttering, sensation of rhythm disappears. If the cloud is asynchronous, we hear intermittencies.

25–50 grains per sec—Grain order disappears. Upper and lower frequency bounds can be inferred. In a synchronous cloud, the perception of synchronicity evaporates if the upper and lower frequency bounds extend beyond several semitones. As the density increases, we no longer perceive an acceleration of tempo in the grain emissions, but rather we feel an increase in the flow of grains.

50–100 grains per sec—Texture band. If the bandwidth is greater than a semitone, we cannot discern individual frequencies.

>100 grains per sec—Continuous sound mass. No space between grains. In some cases resembles reverberation.

Density and frequency band effects are also synergistic, and, depending on the grain density, the musical results of the band parameter will differ. For sparse, pointillist effects, for example, where each grain is heard as a separate event, keep the grain density to less than $0.5/D$, where D is grain duration in seconds. So, for a grain duration of 20 ms, the density should be less than 25 grains per sec (0.5/0.02).

By increasing the grain density, we enrich the texture, creating effects that depend on the bandwidth.

- Narrow bands and high densities generate pitched streams with formant spectra.
- Medium bands (e.g., intervals of several semitones) and high densities generate turgid colored noise.

- Wide bands (e.g., an octave or more) and high densities generate massive clouds of sound.

As we have seen, in the section on grain duration effects, another way to modify the bandwidth of a cloud is by changing the grain duration parameter.

Granular Spatial Effects

Granular synthesis calls for multichannel output, with an individual spatial location for each grain. If the cloud is monaural, with every grain in the same spatial position, it is spatially flat. In contrast, when each grain scatters to a unique location, the cloud manifests a vivid three-dimensional spatial morphology, evident even in a stereophonic configuration.

From a psychoacoustical point of view, the listener's perception of the spatial position of a grain or series of grains is determined by both the physical properties of the signal and the *localization blur* introduced by the human auditory system (Blauert 1997). Localization blur means that a point source sound produces an auditory image that spreads out in space. For Gaussian tonebursts, the horizontal localization blur is in the range of 0.8° to 3.3°, depending on the frequency of the signals (Boerger 1965). The localization blur in the median plane (starting in front, then going up above the head and down behind) is greater, on the order of 4° for white noise and becoming far greater (i.e. less accurate) for purer tones. (See Boerger 1965 for a study of the spatial properties of Gaussian grains.)

Taking localization blur into account, one can specify the spatial distribution of the grains in one of two ways: as an envelope that pans across N channels, or as a random dispersion of grains among N channels. Random dispersion is especially effective in the articulation of long grains at low densities.

Chapter 5 presents more on the spatial effects made possible through particle scattering and other techniques.

Granular Clouds as Sound Objects

A cloud of grains may come and go within a short time span, for example, less than 500 ms. In this case, a cloud of grains forms a tiny sound object. The inner structure of the cloud determines its timbral evolution. I have conducted numerous experiments in which up to fifty grains were generated within a time span of 20 to 500 ms. This is an effective way to construct singular events that cannot be created by other means.

Cloud Mixtures

A granular composition is a flow of multiple overlapping clouds. To create such textures, the most flexible strategy is first to generate each individual cloud. Then to mix the clouds to precisely order and balance their flow in time. To create a polychrome cloud texture, for example, several monochrome clouds, each with a different grain waveform are superimposed in a mixing program.

It is easy to granulate a sound file and take the results "as is." A more sophisticated strategy is to take the granulation as a starting point. For example, one can create a compound cloud—one with an interesting internal evolution—by carefully mixing several granulated sound files.

Mixing is also effective in creating rhythmic structures. When the density of a synchronous cloud is below about 20 Hz, it creates a regular metric pulse. To create a polyrhythmic cloud, one can generate several clouds at different densities, amplitudes, and in different frequency regions to stratify the layers.

Implementations of Granular Synthesis

This section surveys the history of implementations of granular synthesis on computers. It begins with my own first experiments, going on to cover a variety of implementations since.

The Author's Implementations of Granular Synthesis

My involvement with granular synthesis dates back to May of 1972, when I participated in Iannis Xenakis's workshop on music and mathematics at Indiana University. The workshop was based on his book *Formalized Music* (Xenakis 1971, 1992). One chapter of this book described a theoretical approach to sound synthesis based on "elementary sonic particles:"

> *A complex sound may be imagined as a multicolored firework in which each point of light appears and instantaneously disappears against a black sky ... A line of light would be created by a sufficiently large multitude of points appearing and disappearing instantaneously.* (Xenakis 1992 pp. 43–4)

This description intrigued me, but there were no sounds to hear. Granular synthesis remained a theoretical topic at the workshop. Maestro Xenakis took us to the campus computing center to show us experiments in stochastic waveform generation (also described in his book), but he never realized granular synthesis on a computer.

Later that year, I enrolled as a student in music composition at California Institute of the Arts. During this period, I also studied mathematics and computer programming with Leonard Cottrell. For the next two years, I wrote many programs for the Data General Nova 1200, a minicomputer at the Institute. Thus included software for stochastic processes and algorithmic composition based on Xenakis's formulas (Roads 1992a). I spent much time testing the formulas, which fostered in me a deeper understanding of probability theory. The Nova 1200 was limited, however. It lacked memory and had no digital audio converters. Its only peripheral was a teletype with a paper tape punch for storing and reading programs. Digital sound synthesis was out of the question.

In March 1974, I transferred to the University of California, San Diego (UCSD), having learned of its computer sound synthesis facilities. Bruce Leibig, a researcher at UCSD, had recently installed the Music V program (Mathews 1969) on a mainframe computer housed in the UCSD Computer Center. The dual-processor Burroughs B6700 was an advanced machine for its day, with a 48-bit wordlength, virtual memory, digital tape storage, and support for parallel processing. A single language, Extended Algol, provided access to all levels of the system, from the operating system to the hardware. This is not to say that music synthesis was easy; because of the state of input and output technology, the process was laborious.

The Burroughs machine could not produce sound directly. It could, however, write a digital tape that could be converted to sound on another computer, in this case a Digital Equipment Corporation (DEC) PDP-11/20, housed on campus at the Center for Music Experiment (CME). Bruce Leibig wrote the PAL-11 assembly language code that performed the digital-to-analog conversion. This important programming work laid the foundation for my research. I enrolled in an Algol programming course offered by the computer science department. There were no courses in computer sound synthesis, but with help from Bruce Leibig, I learned the Music V language. We programmed on punched paper cards, as there were no interactive terminals.

Owing to storage limitations, my sound synthesis experiments were limited to a maximum of one minute of monaural sound at a sampling rate of 20 kHz. It took several days to produce a minute of sound, because of the large number of steps involved. The UCSD Computer Center scheduled sound calculations for the overnight shift. So I would submit a box of punched cards to a computer operator and return the next day to collect a large digital tape reel containing the previous evening's data. In order to convert this data into sound, I had first to transfer it from the tape to a disk cartridge. This transfer involved setting up an appointment at the Scripps Institute of Oceanography. Surrounded by the pungent atmosphere of the squid tanks of the Neurology Computing Laboratory, I transferred the contents of the tape. Then I would take the disk cartridge to CME and mount it on the DEC minicomputer. This small computer, with a total of 28 kbytes of magnetic-core RAM, had a single-channel 12-bit digital-to-analog converter (DAC) designed and built by Robert Gross. The digital-to-analog converter truncated the four low-order bits of the 16-bit samples.

After realizing a number of short études with Music V, in December 1974 I tested the first implemention of asynchronous granular synthesis. For this experiment, called *Klang-1*, I typed each grain specification (frequency, amplitude, duration) onto a separate punched card. A stack of about eight hundred punched cards corresponded to the instrument and score for thirty seconds of granular sound. Following this laborious experience, I wrote a program in Algol to generate grain specifications from compact, high-level descriptions of clouds. Using this program, I realized an eight-minute study in granular synthesis called *Prototype*. Chapter 7 describes these studies in detail. (See also Roads 1975, 1978a, 1985c, 1987.)

In 1980, I was offered a position as a Research Associate at the Experimental Music Studio at the Massachusetts Institute of Technology. The computing environment centered on a Digital Equipment Corporation PDP-11/50 minicomputer (16-bit word length) running the UNIX operating system. There I implemented two forms of granular synthesis in the C programming language. These programs generated data that could be read by the Music 11 sound synthesis language. The Csound language (Boulanger 2000; Dodge and Jerse 1997; Vercoe 1993) is a superset of Music 11. The initial tests ran at a 40 kHz sampling rate, and used 1024-word function tables for the waveforms and envelopes. The 1980 implementation generated a textual score or note-list for a sinusoidal granular synthesis oscillator. The second, 1981, implementation at MIT granulated sampled sound files using the soundin unit generator of Music 11. I implemented gestures such as percussion rolls by granulating a single stroke on a snare drum or cymbal. Due to the limitations of the Music 11 language, however, this version was constrained to a maximum density of thirty-two simultaneous grains.

An important transition in technology took place in the 1980s with the introduction of personal computers. By 1988, inexpensive computers (less than

$5000 for a complete system including audio converters) had become powerful enough to support stereo 16-bit, 44.1 kHz audio synthesis. In 1988, I programmed new implementations of granular synthesis and granulation of sampled soundfiles for the Apple Macintosh II computer in my home studio (Roads 1992c, d). I called these C programs Synthulate and Granulate, respectively. For playback, I used the Studer Dyaxis, a digital audio workstation with good 16-bit converters attached to the Macintosh II. My synthesis programs worked with a version of the Music 4C language, which I modified to handle the large amounts of data associated with granular synthesis. Music 4C (Gerrard 1989) was a C-language variant of the venerable Music IVBF language developed in the 1960s (Mathews and Miller 1965; Howe 1975). I revised the synthesis programs in 1991 while I was at the Kunitachi College of Music in Tokyo. After moving to Paris in 1992, I modified the grain generator to work with instruments that I wrote for the Csound synthesis language (Boulanger 2000). The revised programs ran on a somewhat faster Macintosh Quadra 700 (25 MHz), but it still took several minutes to calculate a few hundred grains of sound.

Working at Les Ateliers UPIC in 1995, John Alexander and I developed the Cloud Generator program (Roads and Alexander 1995). Cloud Generator is a stand-alone synthesis and granulation program for MacOS computers. The Appendix documents this program. Our implementation of Cloud Generator merged the C code from several of my previous programs (Synthulate, Granulate, etc.) into a single interactive application. Since then, Cloud Generator has served as a teaching aid in the basics of granular synthesis. It has also been used in compositions by many musicians around the world. It provides a variety of options for synthesis and sound processing. I have used it extensively for research purposes, and in composition.

Although Synthulate and its cousins have no graphical interface, they are extensible. For this reason, I have continued to use them when I needed to try an experiment that could not be realized in Cloud Generator. In early 1999, I revised and recompiled Synthulate and its cousins for the Metrowerks C compiler on the Apple Power Macintosh computer.

Between 1996 and 2000, my CREATE colleagues and I also implemented a variety of particle synthesis and sound processing programs using versions 1 and 2 of the SuperCollider language (McCartney 1996, 1998). SuperCollider provides an integrated environment for synthesis and audio signal processing, with gestural, graphical envelope, or algorithmic control. SuperCollider is my synthesis environment of choice at the present time.

Other Implementations of Granular Synthesis

The number of implementations of granular synthesis has increased greatly in recent years. They are happening all over the world, running on different platforms. Although I try here to be comprehensive, this survey is inevitably incomplete.

Working at the Oberlin Conservatory, Gary Nelson carried out an experiment with something similar to granular synthesis in 1974. Beyond a brief mention in (Nelson 1997), there seems to be no further documentation.

According to Clarke (1996), Michael Hinton implemented a type of granular synthesis for a hybrid computer music system called IMPAC, at EMS Stockholm as early as 1984. When launched, the program generated a sequence of short notes with pseudorandom variations on a number of parameters. These included the upper and lower boundaries of a frequency region within which the program scattered the notes. The synthesis was carried out by a combination of custom-made digital frequency modulation oscillators and analog oscillators. A user could control various parameters in real time with either a joystick or a digital pen. By 1988, Clarke had implemented FOF synthesis (a granular technique; see chapter 4), on Atari computers and within the Csound language (Clarke 1996). (See also Clarke 1998.)

The Canadian composer Barry Truax developed a series of important implementations of granular synthesis. In 1986, he wrote a real-time application on the Digital Music Systems DMX-1000 signal processor, controlled by a DEC LSI-11 microcomputer (Truax 1986). By 1987, he had modified his software to granulate a brief sampled sound. He achieved a technical breathrough in 1990, making it possible to perform real-time granulation on an incoming sound source, such as the live sound of an instrumentalist. This technique enabled him to realize a number of pioneering compositions (see chapter 7). Truax later worked with engineers to develop the Quintessence Box for real-time granular synthesis, using a Motorola DSP 56001 chip for signal processing. A prototype of this box was demonstrated at the 1991 International Computer Music Conference. An operational unit was installed in 1993 at the studios of Simon Fraser University, where the composer teaches.

Working at the University of Naples «Federico II,» Cavaliere, Evangelista, and Piccialli (1988) constructed a circuit called the PSO Troll that could realize up to sixteen voices of granular synthesis in real time at sampling rates up to 62.5 kHz.

In the early 1990s, the Marseilles team of Daniel Arfib and Nathalie Delprat created the program Sound Mutations for time-frequency analysis of sound. After analyzing a sound, the program modified and resynthesized it using granular techniques. It could also perform transformations including time-stretching, transposition, and filtering (Arfib and Delprat 1992, 1993).

James McCartney included a granular instrument in his Synth-O-Matic program for MacOS (McCartney 1990, 1994). Users could draw envelopes on the screen of the computer to control synthesis parameters.

Mara Helmuth realized two different implementations of granular synthesis techniques. StochGran was a graphical interface to a Cmix instrument (Helmuth 1991). StochGran was originally developed for NeXT computers, and later ported to the Silicon Graphics Incorporated IRIX operating system. Helmuth also developed Max patches for granular sampling in real time on the IRCAM Signal Processing Workstation (Helmuth 1993).

A group at the University of York implemented granular synthesis with graphical control (Orton, Hunt, and Kirk 1991). A novel feature was the use of cellular automata to modify the output by mapping the automata to the tendency masks produced by the drawing program. Csound carried out the synthesis.

In 1992 and 1993, I presented several lectures at IRCAM on granular synthesis and convolution techniques, After I left the institute, a number of people who had attended these lectures launched granular synthesis and convolution research of their own as extensions of other long-standing projects, namely Chant synthesis and Max on the IRCAM Musical Workstation. The Granular Synthesis Toolkit (GIST) consisted of a set of external objects for the Max programming language, including a sinusoidal FOF grain generator, and a FOG object for granulation (Eckel, Rocha-Iturbide, and Becker 1995; Rocha 1999). (See the description of FOF synthesis in chapter 4, and the description of granulation in chapter 5.) Also at IRCAM, Cort Lippe (1993), developed another Max application for granulation of sound files and live sound.

Recent versions of the Csound synthesis language (Boulanger 2000) provide four unit generators for granular synthesis: fof, fof2, grain, and granule. Another unit generator, fog, was implemented in versions of Csound from the universities of Bath and Montréal. The fof generator reads a synthetic waveform function table and is oriented toward generating formant tones. The fof2 generator adds control over the initial phase increment in the waveform function table. This means that one can use a recorded sound and perform time-stretching, or extract segments. The grain unit generator begins reading a waveform function table from a random point. The granule unit generator handles up to four different grain streams with individual pitches. However, most parameters (including a time-stretch factor) must be set at initialization time (the beginning of each note in Csound), so the only parameter that can be controlled during performance is grain density. The fog generator extracts grains from a sound file. Lee (1995) also implemented a granular unit generator for the Csound language.

The CDP GrainMill granular synthesis program runs on Windows. It derives from Trevor Wishart's command line program Granula, a part of the Composer's Desktop Project System since 1996. The parameters affect each grain individually as it is created. The parameters include size of the grain, density control, time expansion and compression, pitch placement, amplitude, the portion of soundfile from which the grain is extracted, spatial placement, and time placement. The envelope of the grains is variable.

Tom Erbe's Cornbucket (1995) generated a granular synthesis score for Csound. It offered envelope control for all synthesis parameters and was distributed in the form of source code in the C language.

Ross Bencina's Audiomulch is an audio signal processing application also for Windows (Bencina 2000). It includes two modules for granulation of sampled sounds. The Barcelona group of López, Martí, and Resina (1998) developed real-time granular synthesis (again, for Windows), featuring envelope, fader, and MIDI control.

In 1995, R. De Tintis presented a paper to the Italian Computer Music Association (AIMI) on his implementation of granular synthesis and sampling on the IRIS-MARS workstation. The same year, Schnell (1995) and Todoroff (1995) implemented variants of granular synthesis on the IRCAM Musical Workstation.

Kyma (Windows or MacOS) is a commercial package that offers real-time granular synthesis. It is a graphical sound design environment in which one interconnects graphical modules to construct synthesis patches. A synthesizer called the Capybara renders the sound. The 1997 version of Kyma included modules for granular synthesis, granular time expansion, and granular frequency scaling. The parameters of the grains (frequency, pitch deviation, rate of emission, deviation in emission rate, waveform, grain envelope) are controllable in real time through MIDI continuous controllers or faders displayed on the screen, which also allow a source signal to be time-stretched and frequency-scaled.

SuperCollider 2 (McCartney 1996, 1998; De Campo 1999) is a powerful software environment for real-time audio synthesis that runs on MacOS computers. The SuperCollider 2 programming language offers an object-oriented class system, a graphical interface builder for creating a patch control panel, a graphical interface for creating wavetables and breakpoint envelopes, MIDI control, a library of signal processing and synthesis functions, and a library of functions for list processing of musical data. Users can write both the synthesis and compositional algorithms for their pieces in the same high level language. This allows the creation of synthesis instruments with considerably more flexibility than is possible in other synthesis languages. SuperCollider can read and write audio in real time or stream audio to or from a file. The new version, SuperCollider 3, optimizes and extends these capabilities.

Gerhard Behles's real-time Granular program (Technical University Berlin) runs on Silicon Graphics computers. The program reads a sound file and manipulates it in real time. The user moves onscreen faders to change the effects settings. The same author's Stampede allows composers to explore a continuum of sound transformations under MIDI control. It performs granulation operations similar to those in Cloud Generator, but operates in real time. Andre Bartetzki, at the electronic music studio of the Hochschule für Musik Berlin, has written a granular event generator called CMask that generates grain specifications for Csound (Bartetzki 1997a, 1997b). CMask provides numerous options for scattering grains according to probabilistic functions, sieves, and analogies to simple physical processes.

Damiàn Keller and Barry Truax (1998) developed Cmask models for bouncing, breaking, scraping, and filling. The Cmask control functions determine where to scatter the grains in time and frequency. For example, a recursive equation approximated a bouncing pattern. By changing a damping parameter, one could obtain a family of exponential curves with different rates of damping or grain rate acceleration. Starting from samples of water drops, Keller and Truax developed models of droplet patterns and streams, allowing for a smooth transition between discrete droplets and denser aqueous sounds. Chris Rolfe and Damiàn Keller (1999) developed a standalone MacOS program for soundfile granulation called MacPOD.

William Mauchly of the company Waveboy created a granular synthesis plugin for the Ensoniq ASR-10 and EPS-16 Plus samplers. Working as a signal processor, users can granulate any sampled sound or live audio input. This software offers time-scrambling, pitch-shifting, and adjustment of grain duration. Any MIDI controller can modulate the granulation parameters.

Michael Norris (1997) provided four granulation processes in his SoundMagicFX package, which works with the SoundMaker program for MacOS. Entitled Brassage Time Stretch, Chunk Munger, Granular Synthesis, and Sample Hose, these flexible procedures allow multiple-file input, time-varying parameters, and additional signal processing to be applied to soundfiles, resulting in a wide range of granular textures.

Eduardo Miranda developed a Windows application called ChaosSynth for granular synthesis using cellular automata (CA) control functions (Miranda 1998). Depending on how the CA are configured, they calculate the details of the grains. A difficulty posed by this approach is the conceptual rift between the CA controls (number of cell values, resistances of the potential divider, capacitance of the electrical capacitor, dimension of the grid, etc.) and the acoustical results (Correa, Miranda, and Wright 2000).

In 1999, Arboretum Systems offered a scattering granulator effect in its popular Hyperprism effects processing software. The user controls grain size, randomization, speed, as well as density and spread.

Can a standard MIDI synthesizer realize granular synthesis? Yes, in a limited form. The New York–based composer Earl Howard has done so on a Kurzweil K2500 sampling synthesizer. The K2500 lets one create short samples, which can repeat by internal signals as fast as 999 bpm, or about every 10 ms. Howard created granular textures by layering several streams operating at different rates, with each stream having a random delay. Another MIDI-based approach to granular synthesis is found in Clarence Barlow's *spectastics* (spectral stochastics) technique. This generates up to two hundred notes per second to approximate the spectrum of a vocal utterance (Barlow 1997).

Even with all these implementations, there is still a need for an instrument optimized with controllers for the virtuoso performance of granular textures. Apropos of this, see the description of the Creatovox project in chapter 5.

Summary

As regards electric instruments for producing sound, the enmity with which the few musicians who know them is manifest. They judge them superficially, consider them ugly, of small practical value, unnecessary. . . . [Meanwhile, the inventors] undiscerningly want the new electric instruments to imitate the instruments now in use as faithfully as possible and to serve the music that we already have. What is needed is an understanding of the . . . possibilities of the new instruments. We must clearly evaluate the increase they bring to our

> *own capacity for expression . . . The new instruments will produce an unforeseen music, as unlooked for as the instruments themselves.* (Chavez 1936)

Granular synthesis is a proven method of musical sound synthesis, and is featured in important compositions (see chapter 7). Implementations of granular techniques are widespread. Most focus on the granulation of sampled sound files. Pure granular synthesis using synthetic waveforms is available only in a few packages.

At low densities, synchronous GS serves as a generator of metrical rhythms and precise accelerandi/rallentandi. A high-density cloud set to a single frequency produces a stream of overlapping grains. This forms sweet pitched tones with strong formants, whose position and strength depend greatly on the grain envelope and duration.

Asynchronous GS sprays thousands of sonic grains into cloudlike formations across the audio spectrum. At high densities the result is a scintillating sound complex that varies over time. In musical contexts, these types of sounds can act as a foil to the smoother, more sterile sounds emitted by digital oscillators. Granulation of sampled sound—a popular technique—produces a wide range of extraordinary variations, explored in chapter 5. The destiny of granular synthesis is linked both to graphics and to real-time performance.

A paint program offers a fluid interface for granular synthesis. The MetaSynth program (Wenger and Spiegel 1999), for example, provides a spray brush with a variable grain size. A further extension would be a multicolored spray jet for sonic particles, where the color palette corresponds to a collection of waveform samples. (In MetaSynth, the color of the grains indicates their spatial location.)

Analysis/resynthesis systems, such as the phase vocoder, have an internal granular representation that is usually hidden from the user. As predicted (in Roads 1996), the interfaces of analysis/resynthesis systems—which resemble sonograms—have merged with interactive graphics techniques. This merger—sonographic synthesis—is a direct and intuitive approach to sound sculpture. (See chapters 4 and 6 for more on sonographic synthesis and transformation.) One can scan a sound image (sonogram), touch it up, paint a new image, or erase it, with the algorithmic brushes of computer graphics.

My colleagues and I continue to refine our instrument for real-time virtuoso performance of granular synthesis (Roads 1992–1997). The Creatovox research project at the University of California, Santa Barbara has resulted in a prototype of a granular synthesis instrument, playable on a standard musical keyboard and other controllers. (See the description in chapter 5.)

Granular synthesis offers unique opportunities to the composer and suggests new ways of organizing musical structure—as clouds of evolving sound spectra. Indeed, granular representation seems ideal for representing statistical processes of timbral evolution. Time-varying combinations of clouds lead to such dramatic effects as evaporation, coalescence, and mutations created by crossfading overlapping clouds. A striking similarity exists between these processes and those created in computer graphics by *particle synthesis* (Reeves 1983), often used to create images of fire, water, clouds, fog, and grasslike textures, analogous to some of the audio effects possible with asynchronous granular synthesis.