

Fractal 1

Version of November 6, 2024

Reiner Zorn

Fractal 1 is an instrument based on a fractal sequence of pulses. It resembles a bit Etude # 3, but the spacings between the pulses are chosen deterministically¹.

1 Pulse sequence

The elementary unit of the sequence are pulses at $t = 0$ and $t = t_0$ in a frame of length $2t_0$. Instead of starting the next frame at $2t_0$, it is started at $t_1 = at_0$ with $a > 2$. This four-pulse pattern is repeated at $t_2 = at_1 = a^2t_0$ and so on (Figure 1). It can easily be shown that the Hausdorff dimension of the resulting pulse sequence is

$$d_f = \frac{1}{\log_2 a} . \quad (1)$$

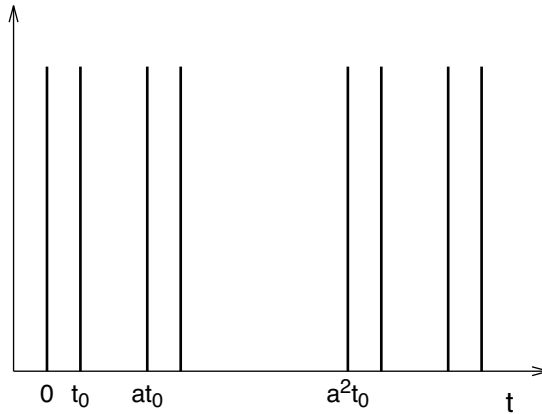


Figure 1: Pulses of the second generation of the fractal.

Of course, for $a = 1$ the procedure results in an equidistant pulse sequence sounding like what is generated by the Csound opcode `buzz`. In that case, $d_f = 1$. By increasing a the dimension can be reduced down to zero. Decreasing a below 2 should have the opposite effect. But the problem in realising the sound is that, after some generations, pulses of the next frame will fall before those of the preceding. Since the generating algorithm is based on pulse spacings in time, it will not work because negative numbers appear.

¹Of course, also for Etude # 3 these are deterministic because they are generated by *pseudorandom* generators. But you know what I mean.

2 Anti-aliasing

The solution for avoiding anti-aliasing is basically the same as for Etude # 3. For convenience, the description is copied here.

In the case of sequences of ‘ideal’ (i.e. delta function shaped) pulses there is the possibility to obtain an aliasing-free signal. Ideally, each pulse should be replaced by the inverse Fourier transform of a box window (in frequency) whose width is half the sampling frequency f_s ,

$$\frac{\sin(\pi f_s t)}{\pi t}. \quad (2)$$

The disadvantage of this function is that it decays only with t^{-1} in time. So, a large sample would have to be played for each instance of a pulse which would probably lead to a prohibitive CPU load at high frequencies. The solution is to multiply the pulse function (2) by a window in time. Admittedly, this leads to a softening of the cut-off in frequency potentially allowing aliasing to reappear. But from signal processing theory several window functions are known to perform well in that respect. Here, the Kaiser window [1] with $a = 2$ was chosen for that purpose:

$$\frac{I_0\left(a\pi\sqrt{1-t^2/T^2}\right)}{I_0(a\pi)} \text{ for } |t| \leq T \quad (3)$$

Here, I_0 denotes the zeroth-order Bessel function of the first kind. The cut-off time T was chosen as $6/f_s$ so that there are six oscillations of the sine in equation (2) represented. The product of (2) and (3) was calculated at 128 points and stored in table 1010 in the score. It turns out that generating the audio signal from this table is sufficient to completely suppress aliasing audibly and monitored by a spectrum analyser. On the other hand, the limit of the time range ensures that only six lookups from the table per pulse are necessary.

3 Controls

The knob **Frequency**, in the simplest case, sets the elementary distance to $t_0 = 1/f$. f is given as the MIDI note number. There are two modifiers for this knob. **F** (follow) instructs the instrument to follow the MIDI note played. In that case the frequency is interpreted as an detune with respect to A4/MIDI 69/440 Hz. **C** (correct) applies a correction to the frequency which makes the effect of a change of the fractal dimension more natural (see below).

df = ... controls the fractal dimension d_f . Using the button **M** this can be coupled to the mod wheel. A non-linearity² is built into this control to allow the user fine control in the range $d_f > 0.9$ which already has a significant effect, but is still ‘nice’ to hear.

In principle, the fractal extends to infinite generations and thus over an infinite time. But as soon as the time for starting a new frame falls below about 10 Hz, the effect is perceived as a rhythm — not as part of the sound anymore. Even if this is the desired effect, the

²For DAW control you may need the formula: $d_f = 1.0 - \text{kfract}^2$ with **kfract** being the position of the slider between zero and one.

pauses in that rhythm will strongly grow with time and at some point become too long. Therefore, the construction of the fractal is stopped after n generations, determined by the **depth** knob at i-time. Unless the button labelled **One Shot** is pressed, after the n th generation the sequence is repeated periodically. The total time for a period is $a^n t_0$, implying that the fundamental frequency is $a^{-n} f$. For d_f close to one, the amplitude of the fundamental will be small and the dominating frequency will be that of the n th octave above, which is close to the original fundamental (of the non-fractal $d_f = 1$ sound, f). The frequency of that harmonic is lower than the original frequency, f , by a factor $(2/a)^n < 1$. This results in a perceived flattening of the pitch when d_f is decreased. To avoid this, a frequency correction with the inverse of this factor will be applied when button **C** is pressed. In the limit $d_f \rightarrow 0$, the effect of this correction is that an equidistant pulse sequence is produced again, but with a frequency n octaves below, thus $2^{-n} f$.

The effect of the **Bipolar** button is to invert the polarity of every second pulse. For $d_f = 1$, the obvious consequence is that all even harmonics vanish. For the true fractal ($1 > d_f > 0$), the fundamental is lowered to $a^{-n+1} f$ only, thus one octave higher after correction. In the limit $d_f \rightarrow 0$ with frequency correction, only silence remains because neighbouring pulses cancel each other.

(The following controls are mostly identical to those of Etude # 3.) **Attack**, **Decay**, **Sustain**, and **Release** have the usual meanings. Presets can be selected in the combobox. New ones can be defined by $+$ and existing deleted by $-$.

The pulse sequence can be filtered by **Low Pass** which is inactive if the knob is turned to the right end. Also here coupling to the MIDI note is possible and a simple modulation sweeping the filter. If the **Low Pass** frequency lies significantly below the fundamental frequency, the effect results in a ‘fractalised’ sawtooth (or square wave if **Bipolar** is selected). The **Resonance** knob allows to control the second parameter of the Csound filter `lowpass2`. This allows a second way to ‘play’ the instrument, namely producing pulses at low frequency which are converted into pitched sounds through the resonance of the filter (as used in the preset “Heartbeat”).

Room Size, **Damping**, **Dry/Wet** are the controls for `freeverb`. The **Volume** knob has a large range (± 40 dB) because the instrument tends to produce largely different amplitudes depending on the filter settings which have to be compensated to avoid clipping. This can be checked on the VU meter. To the left of the piano keyboard, a button **P** is added to permanently play a note which is useful to study the effect of varying controls continuously. (Note that a change of the depth n will only take effect if this button is toggled again, because it is an i-type variable.)

4 Known bugs and problems

It would be nice if the ‘frequency’ would be set to MIDI 69 (no transposition) and the filter to 440 Hz (filter frequency = f) when the respective **F** buttons are pressed. As mentioned for Etude # 3, this conflicts with the combobox-snaps preset system. The resulting (mis)behaviour annoyed me so much that I disabled this feature. It may work with the ‘new’ presets system. But frankly speaking, I am too lazy to try.

Also as in Etude # 3, there is the issue that the whole instrument runs at `ksmps = 1`, thus very CPU-intensive. This may be resolved by separating it into a main instrument

with higher `ksmps` and a subinstrument. That would be quite some work with the risk to break the already precarious code.

The fact that the depth is an i-time variable and a note has to be re-triggered for a change to have effect sometimes surprises me myself. But also this would be difficult, to change it to a k-rate variable.

References

- [1] *Kaiser Window*, https://en.wikipedia.org/wiki/Kaiser_window.