KOMPUTEROWE NARZĘDZIA MUZYCZNE

Komputerowe narzędzia muzyczne

- Mamy na myśli wszelkie oprogramowanie, które jest użyteczne do tworzenia komputerowej muzyki (*computer music*).
- Znamy już z poprzednich wykładów:
  - samplery – jako instrumenty muzyczne oraz jako programy do tworzenia banków instrumentów,
  - sekwencery MIDI – programy do rejestracji, edycji i odtwarzania kodów MIDI – sterowania instrumentami muzycznymi (sprzętowymi i programowymi).

VST

Virtual Studio Technology (VST) – standard firmy Steinberg

- *VST plugin* (wtyczki):
  - *VST effects*: efekty brzmieniowe, otrzymują cyfrowy dźwięk, wysyłają na wyjście przetworzony dźwięk,
  - *VST instruments* (VSTi): otrzymują na wejściu kody MIDI, wytwarzają dźwięk cyfrowy (synteza, sampling, itp.) i wysyłają go na wyjście,
  - *VST MIDI effects*: otrzymują kody MIDI, przetwarzają je i wysyłają na wyjście.
- *VST host*: program wysyłający dane do wtyczek VST i odbierający od nich wyniki działania.

Instrumenty VSTi

Zadania programisty instrumentu VSTi:

- napisanie algorytmu, który generuje (dowolną metodą) dźwięk cyfrowy o parametrach zdefiniowanych w kodach MIDI otrzymanych od hosta,
- stworzenie interfejsu użytkownika (GUI),
- zdefiniowanie i interpretacja parametrów *MIDI Control Change*, które wpływają na sposób wytwarzania dźwięku.

Programista nie musi martwić się o funkcje wejścia/wyjścia, to załatwia za niego host!

Host VST - DAW

Współczesne hosty VST to DAW – *Digital Audio Workstation*

- ścieżki audio:
  - dźwięk cyfrowy (nagrania instrumentów, wokalisty),
  - wykorzystanie efektów VST,
- ścieżki MIDI:
  - zarejestrowane kody MIDI,
  - sterowanie instrumentem (VSTi lub sprzętowym),
  - funkcje edycyjne sekwencera MIDI,
  - dźwięk cyfrowy powstaje w procesie masteringu.

Zalety wykorzystania VSTi

Co daje VSTi w porównaniu z nagraniem dźwięku cyfrowego:

- można łatwo edytować nagrane kody MIDI,
- można modyfikować brzmienie instrumentu VSTi poprzez zmiany jego programów i ustawień,
- można łatwo wymieniać instrumenty VSTi bez zmiany nagranych kodów MIDI,
- można korzystać z wielu instrumentów jednocześnie, ograniczeniem jest tylko moc komputera.

Uruchamianie instrumentów VSTi

Jak uruchomić jeden instrument VSTi bez potrzeby wykorzystywania skomplikowanego hosta DAW?

- Pobieramy program *SAVIHost*
  (http://www.hermannseib.com/english/savihost.htm)
- Plik savihost.exe umieszczamy w katalogu instrumentu.
- Zmieniamy jego nazwę na taką, jaka ma wtyczka,
  - np. plik wtyczki: *Synth1 VST.dll*
  - zmieniamy *SAVIHost.exe* na *Synth1 VST.exe*
- Uruchamiamy plik exe i możemy grać.

Komputerowe języki muzyczne - CSound

- Języki programowania ogólnego zastosowania (C++, Java, Python, itp.) nie mają wygodnych procedur do tworzenia muzyki.
- Powstały specjalne języki programowania (zwykle skryptowe) służące do tworzenia muzyki.
- CSound – jeden z najczęściej używanych języków programowania dla muzyki. Kod programu definiuje:
  - *orchestra* – instrukcje tworzące dźwięki (instrumenty),
  - *score* – instrukcje tworzące muzykę z tych dźwięków.
- Duże możliwości, ale dość trudny język programowania.
- Dużo literatury i przykładów.

Csound - przykład

Przykład skryptu
Csound
– synteza
simple FM

http://booki.flossmanuals.net/csound/d-frequency-modulation/

```
<CsoundSynthesizer>
<CsInstruments>
sr = 48000
ksmps = 32
nchnls = 2
0dbfs = 1
instr 1

kCarFreq = 440     ; carrier frequency
kModFreq = 440     ; modulation frequency
kIndex = 10        ; modulation index

kIndexM = 0
kMaxDev = kIndex*kModFreq
kMinDev = kIndexM*kModFreq
kVarDev = kMaxDev-kMinDev
kModAmp = kMinDev+kVarDev

; oscillators
aModulator poscil kModAmp, kModFreq, 1
aCarrier poscil 0.3, kCarFreq+aModulator, 1

outs aCarrier, aCarrier
endin
</CsInstruments>

<CsScore>
f 1 0 1024 10 1     ; Sine wave for table 1
i 1 0 15
</CsScore>

</CsoundSynthesizer>
; written by Alex Hofmann (Mar. 2011)
```

SuperCollider

- Nowszy język programowania muzyki.
- Również język tekstowy.
- Przeznaczenie:
  - synteza dźwięku w czasie rzeczywistym,
  - komponowanie algorytmiczne.
- Działa w architekturze „klient-serwer”.
- Umożliwia tworzenie GUI.
- Mniejsza liczba przykładów niż dla CSound.
- Dostępny dla wielu systemów operacyjnych.
- Możliwość rozszerzania o własne instrumenty.

SuperCollider - przykład

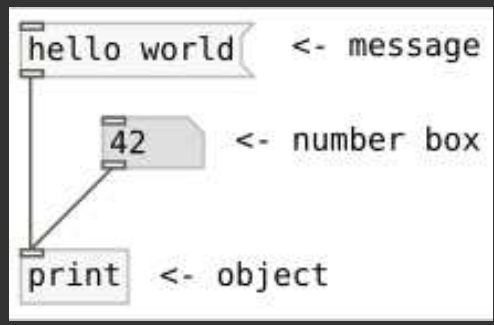
Przykład skryptu *SuperCollider* – synteza *simple FM*

http://danielnouri.org/docs/SuperColliderHelp/Tutorials/Mark_Polishook_tutorial/Synthesis/14_Frequency_modulation.html

```
(
SynthDef("fm1", { arg bus = 0, freq = 440, carPartial = 1, modPartial = 1, index = 3, mul = 0.05;
            // carPartial :: modPartial => car/mod ratio

            var mod;
            var car;

            mod = SinOsc.ar(freq * modPartial, 0,
                        freq * index * LFNoise1.kr(5.reciprocal).abs);

            car = SinOsc.ar( (freq * carPartial) + mod, 0, mul);

            Out.ar(bus, car)
}).load(s);
)

(
Synth("fm1", [\bus, 0, \freq, 440, \carPartial, 1, \modPartial, 1, \index, 10]);
)

(
s.queryAllNodes;
)
```

Pure Data (pd) / Max/MSP

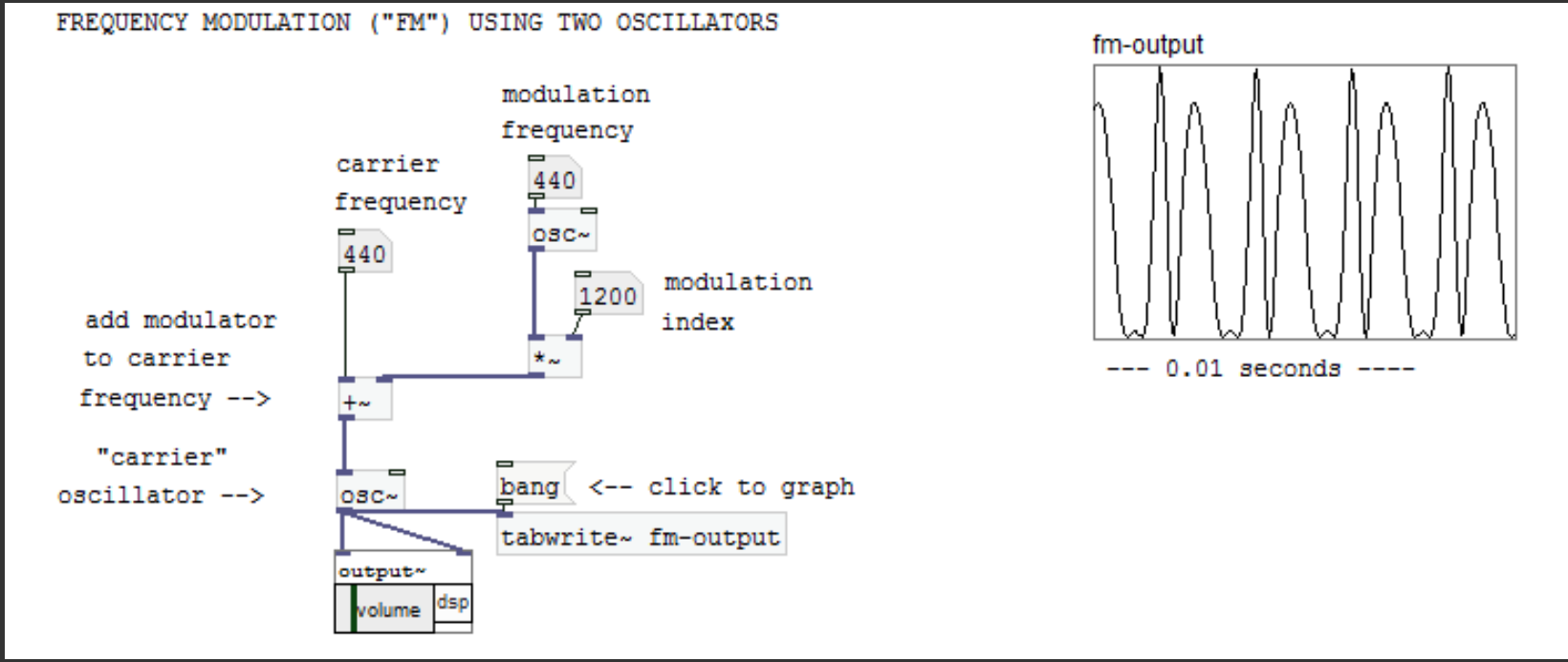
- Alternatywne podejście: zamiast tekstowego kodu - wizualne programowanie za pomocą schematu.
- Implementacje:
  - *Pure Data* (pd) – open source (Linux, Windows, Mac)
  - *Max/MSP* – komercyjna, MacOS
- Przetwarzanie dźwięku, synteza, sampling, obsługa MIDI.
- Dość ascetyczny interfejs użytkownika.
- Bardziej „poglądowy” sposób programowania, ale skomplikowane układy mogą być „poplątane”.
- Możliwość rozszerzania o własne bloki.
- Sygnały: *audio* i *control*.



Pure Data - przykład

Układ syntezy *Simple FM* w programie *pd*

z przykładu dostarczonego z programem (*A09.frequency.mod.pd*)

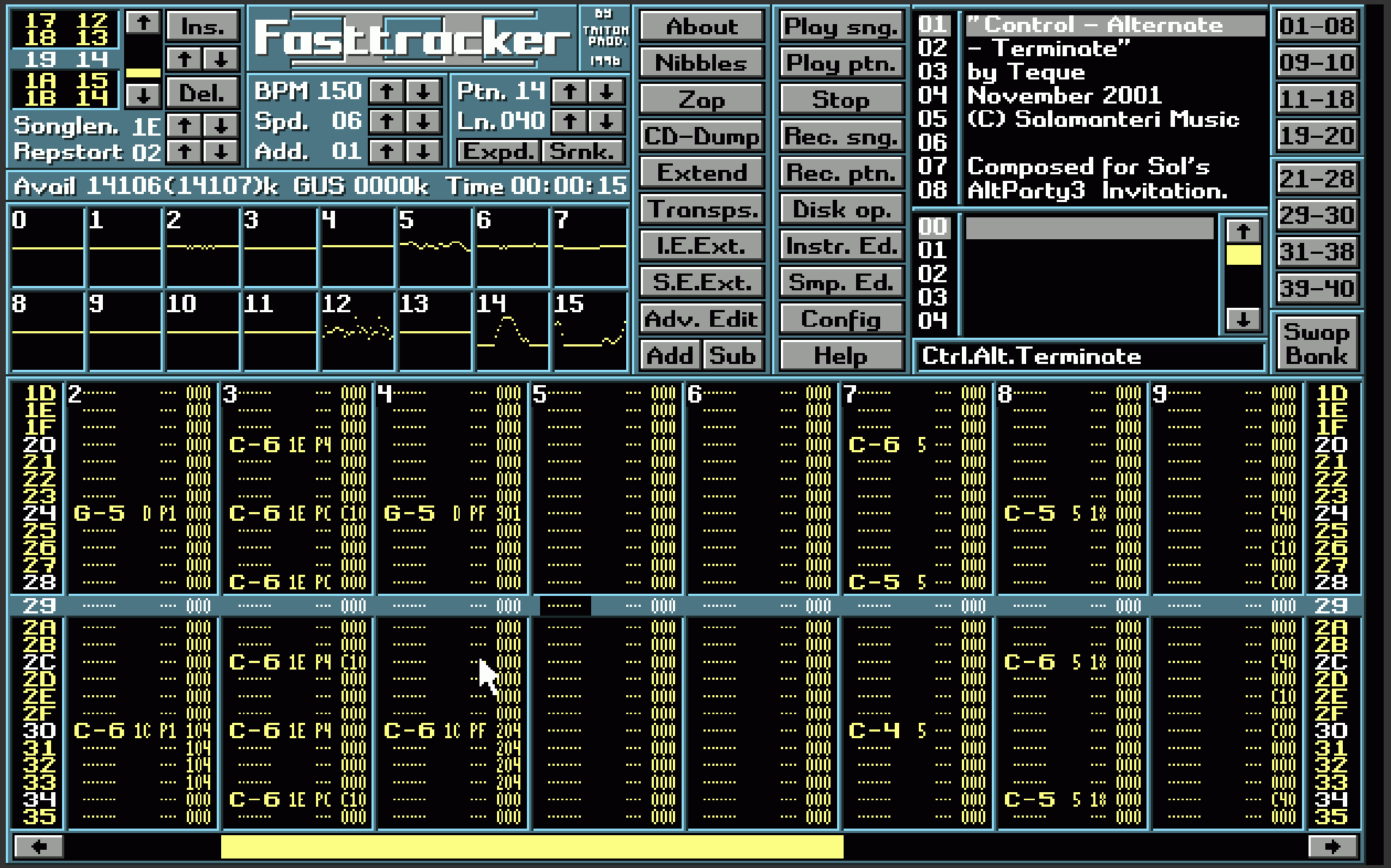


Trackery

Tracker – typ oprogramowania wywodzący się z lat 80. 20. wieku i komputerów 16-bitowych (np. Amiga).

- Łączy funkcje samplera i sekwencera.
- Próbki – bardzo krótkie, zwykle zapętlane.
- Instrumenty – próbki z nałożonymi efektami i obwiednią.
- Wzorzec (*pattern*) – sposób odtwarzania dźwięków (rodzaj zapisu nutowego).
- Sekwencja (*order*) – sposób odgrywania wzorców, zwykle z ich powtórzeniami.
- Moduł (*module*, MOD) – plik zawierający wszystkie te dane, mały rozmiar – zwykle poniżej 100 KB!

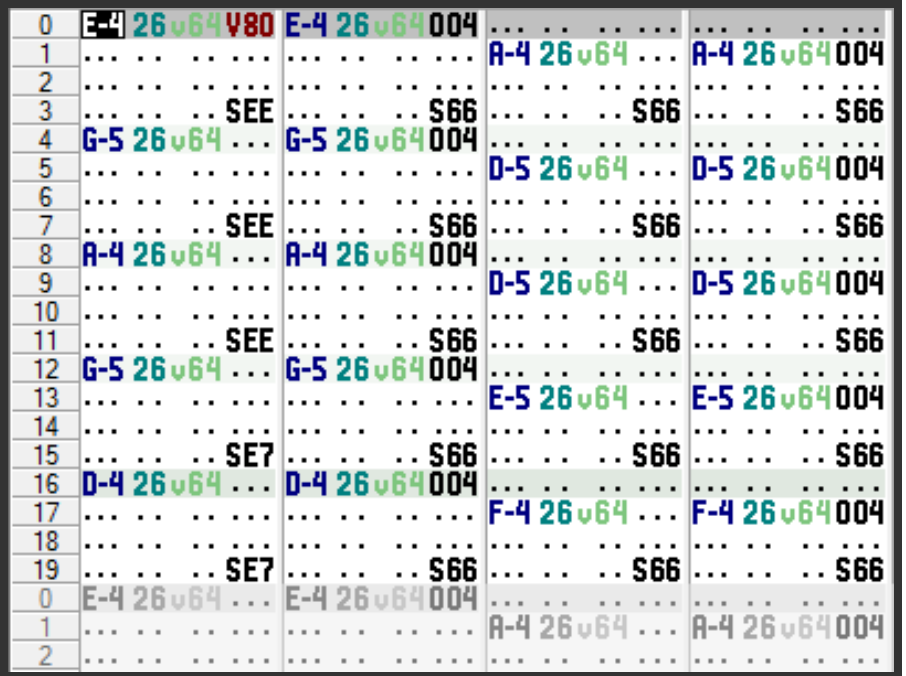
Fasttracker (Amiga)



Wzorce

Wzorzec (*pattern*) opisuje sposób generowania muzyki.

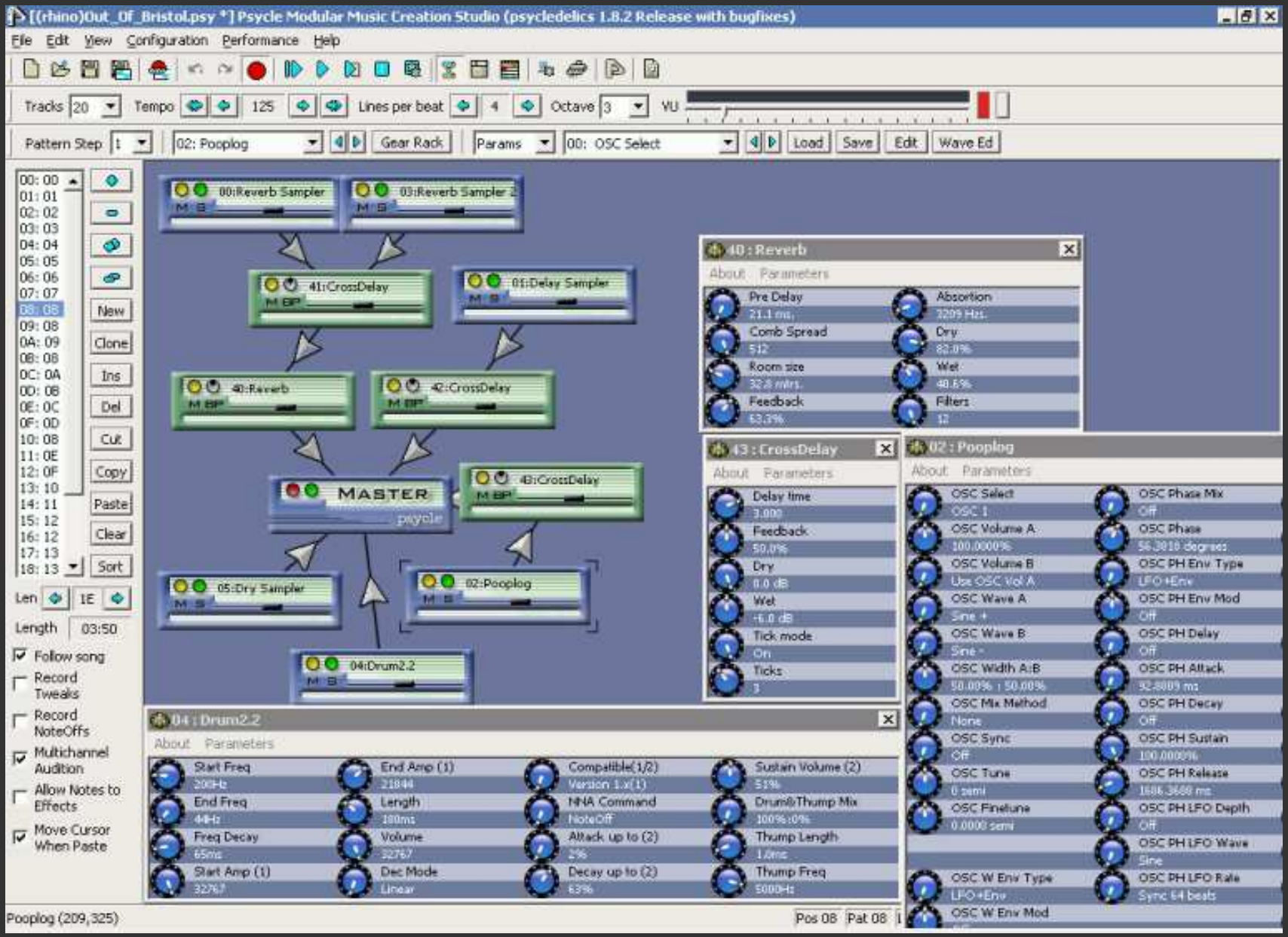
- Kolumny – ścieżki (*tracks*), opisują głosy (wytwarzane równocześnie).
- Rzędy – linie (*lines*), dźwięki na skali czasu.
- Komórki wzorca – nuty (*notes*):
  - wysokość dźwięku,
  - numer instrumentu (próbki),
  - głośność,
  - opis efektu brzmieniowego.



Trackery modularne

- Współczesne trackery są często zmodyfikowane: zamiast prostych instrumentów z próbek, budowane są złożone układy modułów – maszyn (machines):
  - generatory – syntezatory, samplery,
  - efekty – przetwarzanie brzmienia,
  - wzmacniacze i miksery.
- Sterowanie takim układem odbywa się tak samo jak w tradycyjnych trackerach.
- Znacznie większe możliwości brzmieniowe.
- Możliwość tworzenia własnych maszyn.

Psycle – przykład modularnego trackera



Przykłady trackerów i układów modularnych

Współczesne trackery:

- *OpenMPT* – tradycyjny tracker, darmowy, Windows
- *Psycle*, *Buzz* – modularne trackery, darmowe
- *Renoise* – komercyjny DAW oparty na zasadzie trackera

Jest też wiele programów modularnych, które nie wykorzystują zapisu trackera, ale np. MIDI.

- Przykład: *NI Reaktor* (komercyjny).

Programy do kompozycji algorytmicznej

Kompozycja algorytmiczna (*algorithmic composition*) polega na tworzeniu muzyki, nie pojedynczych dźwięków.

Komputer komponuje utwory muzyczne.

Podejścia algorytmiczne (używa się kilku naraz):

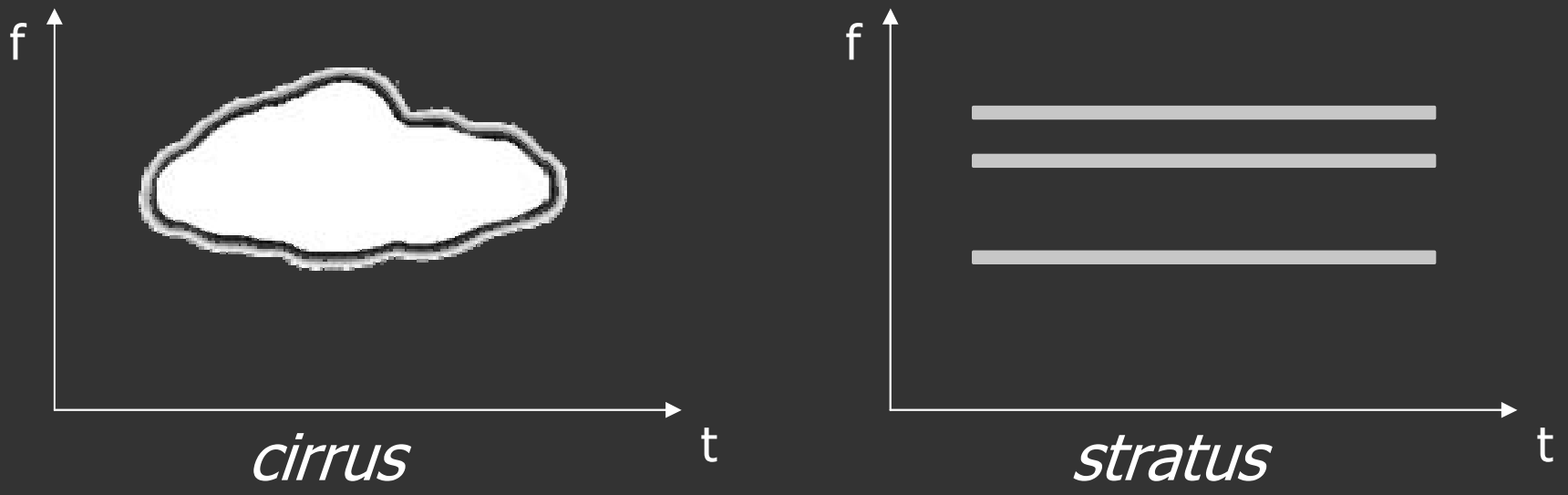
- algorytmy oparte na teorii muzyki, gramatyczne,
- oparte na modelach statystycznych,
- losowe,
- fraktalne (chaos deterministyczny),
- modele sztucznej inteligencji.

Przykład: *cgMusic*
(http://codeminion.com/blogs/maciek/2008/05/cgmusic-computers-create-music/)

Synteza granularna

Synteza granularna (*granular synthesis*) łączy cechy syntezy dźwięku z komponowaniem algorytmicznym.

- Podstawą są granulki – bardzo krótkie fragmenty nagrań dźwiękowych (do 50 ms).
- Granulki są opisane przez wysokość i głośność.
- Duża liczba granulek jest układana w chmury (*soundscape*) na płaszczyźnie czas-wysokość.



Synteza granularna

- Możemy manipulować wysokością, głośnością, szybkością odtwarzania granulek.
- Odpowiednie chmury granulek tworzą dźwięki muzyczne.
- Układy mogą być tworzone zarówno przez muzyka, jak i przez algorytm sterujący układem chmur.
- Metoda nadaje się dobrze do „psychodelicznych” efektów dźwiękowych.
- Odmianą jest synteza falkowa (*wavelet synthesis*), która wykorzystuje analizę falkową do uzyskania granulek dających pożądaną wysokość dźwięku.

Literatura

- VST SDK (Steinberg), dla twórców wtyczek VST: http://ygrabit.steinberg.de/~ygrabit/public_html/index.html
- SAVIHost (host wtyczek VST): http://www.hermannseib.com/english/savihost.htm
- CSound: http://www.csounds.com/
- SuperCollider: http://supercollider.github.io/
- Pure Data (pd): https://puredata.info/
- Tracker *OpenMPT*: https://openmpt.org/
- *Psycle*: psycle.pastnotecut.org
- Programy do kompozycji algorytmicznej (Wikipedia): https://en.wikipedia.org/wiki/Algorithmic_composition