27. bis 29. März 2014, Zürich & Lausanne

m4music

Festival, Conference & Demotape Clinic

WORKSHOP «GENERATIVE VISUAL TOOLS FOR MUSICIANS AND VISUALISTS»

präsentiert von videokultur.ch

Konzept und Realisation

MIGROS kulturprozent

CREATIVE

PROGRAMMING TOOLKITS

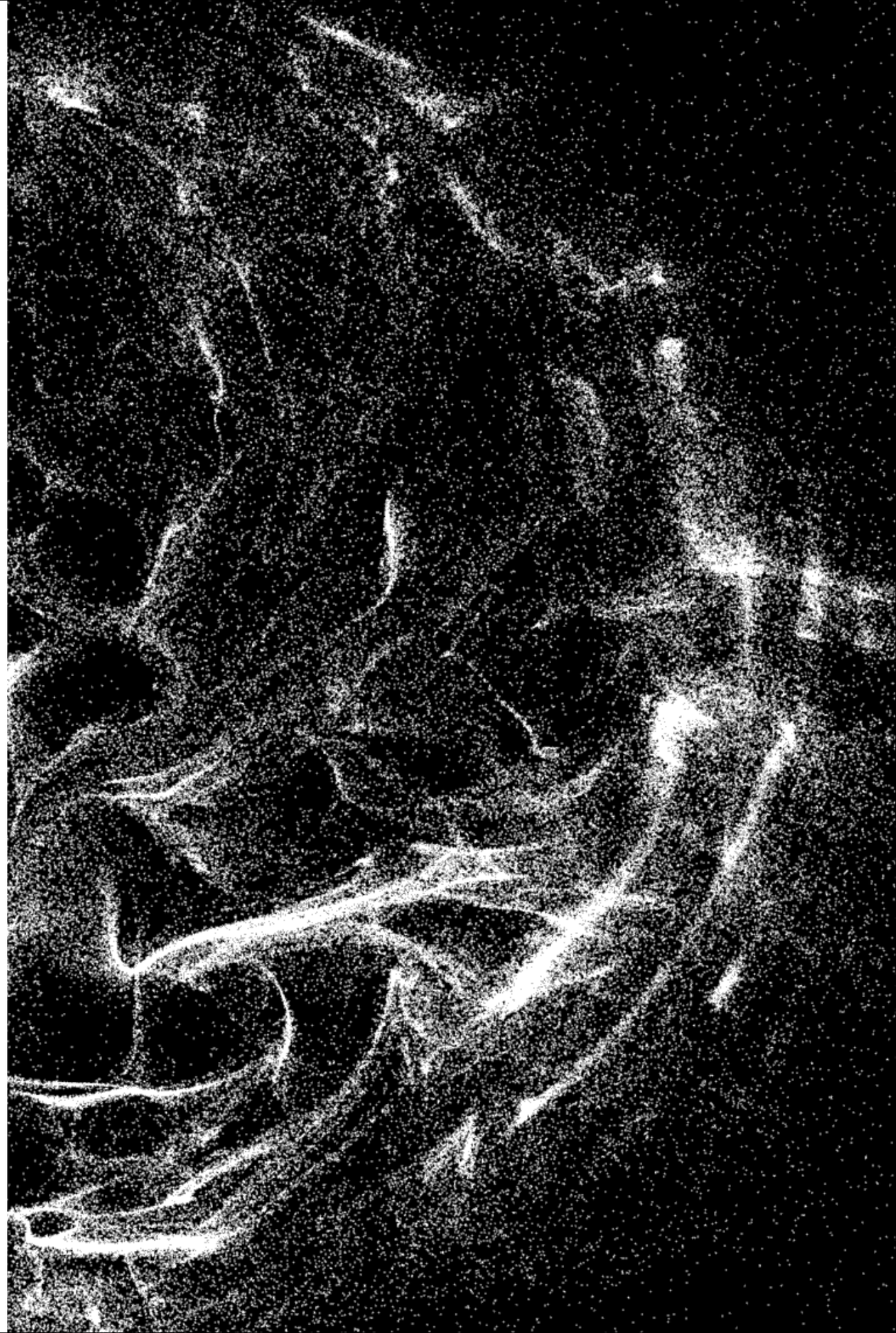
Unter Creative Programming Toolkits verstehen wir Software-Teile welche uns helfen vielfältige Medien-kunst zu erstellen.

Viele dieser Werkzeuge wurden durch Künstler für Künstler erstellt und die bekanntesten sind unter einer open-source Lizenz veröffentlicht.

Solche Werkzeuge bieten grundlegend eine höhere Abstraktionsebene für den Zugriff auf Hardware-nahe Komponenten.

Creative Programming tools werden nicht nur von Künstlern verwendet, sondern auch von Wissenschaftlern und Studenten. Sie können sehr schnell Prototypen erstellen um damit ideen zu visualisieren oder zu entwickeln.

Um creative Programming tools sind meist ganze Ökosysteme entstand, das heisst es findet ein reger Austausch zwischen den Benutzern statt, Benutzer stellen eigene Erweiterungen und Vereinfachungen zur Verfügung, erstellen Tutorials uns Screencasts, helfen in Diskussionsforen, oder nehmen sogar aktiv an der Entwicklung des frameworks selbst teil. es gibt auch solche die weitergehende komplexere Applikation komplett auf einem solchen framework erstellen und weiter verbreiten, zum teil sogar wieder mit eigenen kleinen Ökosystemen.



“**Creative-Coding**“ ist eine Art von Computer-Programmierung, in dem das Ziel ist, etwas ausdrucksvoll anstatt etwas funktionellen erstellen. Es wird verwendet, um Live-Visuals zu erstellen, sowie Erstellung visueller Kunst und Design, Kunstinstallationen, Projektionen und Mapping Projektion, Klangkunst, Werbung, Produkt-Prototypen, und vieles mehr.

```
HUP! 0x7fb97dd9b8d0
HUP! 0x7fb97dd9b920
HUP! 0x7fb97dd9b920
```

make

```
#       ../_/
nothing added to commit but untracked files present (use "git add" to track)
go.c: In function ‘go’:
go.c:27:36: error: expected ‘;’ before ‘}’ token
make: *** [go.so] Error 1
diff --git a/src/go.c b/src/go.c
index 0498e27..b03b737 100644
--- a/src/go.c
+++ b/src/go.c
@@ -24,7 +24,7 @@ typedef struct {

 float go(S *s, float x) {
   Z tick = 0;
-  if (s->tempo < 0.0001) { exit(0) }
+  if (s->tempo < 0.0001) { exit(0); }
   s->time += 1/sr;
   if (s->time > s->tempo) {
     s->time = 0;
[session-2012-09-29-145438 fd5b207] go 2012-09-29T16:22:56+0100
 1 file changed, 1 insertion(+), 1 deletion(-)
diff --git a/src/go.c b/src/go.c
index b03b737..d437880 100644
--- a/src/go.c
+++ b/src/go.c
@@ -29,7 +29,7 @@ float go(S *s, float x) {
   if (s->time > s->tempo) {
     s->time = 0;
     tick = 1;
-    s->tempo *= 0.999;
+    s->tempo *= 0.99;
   }
   R oscs = 0;
   for (Z i = 0; i < COUNT; ++i) {
```

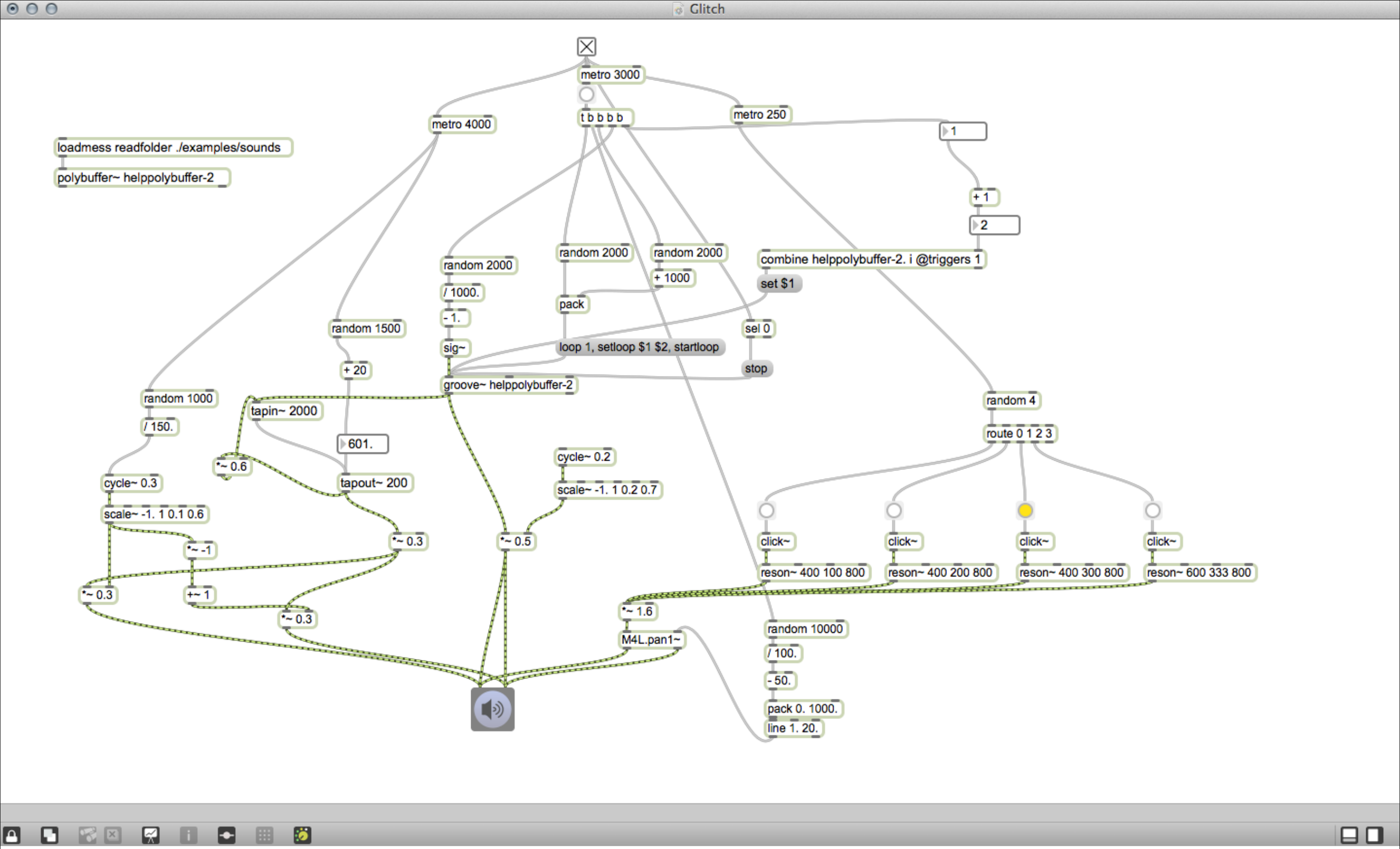
```
#include <stdlib.h>

#define pi 3.1415926
#define sr 48000.0

typedef double R;
typedef long int Z;

#define COUNT 16

typedef struct {
  R time;
  R oscs[COUNT];
  R mods[COUNT];
  R dlts[COUNT];
  R plck[COUNT];
  R dca[COUNT];
  R rng[COUNT];
  R pol[COUNT];
  R rem[COUNT];
} S;

float go(S *s, float x) {
  Z tick = 0;
  if (s->tempo < 0.0001) { exit(0); }
  s->time += 1/sr;
  if (s->time > s->tempo) {
    s->time = 0;
    tick = 1;
    s->tempo *= 0.99;
  }
  R oscs = 0;
  for (Z i = 0; i < COUNT; ++i) {
    if (s->time == 0) {
      s->dlts[i] = rand()/(R)RAND_MAX;
      s->dcay[i] = exp(-0.125 * pow(rand()/(R)RAND_MAX, 4));
      s->rnge[i] = rand() % 512;
```

LIVE -CODING

“**Live-Coding**“ Live coding (sometimes referred to as 'on-the-fly programming', 'just in time programming') is a programming practice centred upon the use of improvised interactive programming. Live coding is often used to create sound and image based digital media, and is particularly prevalent in computer music, combining algorithmic composition with improvisation. Typically, the process of writing is made visible by projecting the computer screen in the audience space, with ways of visualising the code an area of active research. There are also approaches to human live coding in improvised dance. Live coding techniques are also employed outside of performance, such as in producing sound for film or audio/visual work for interactive art installations.

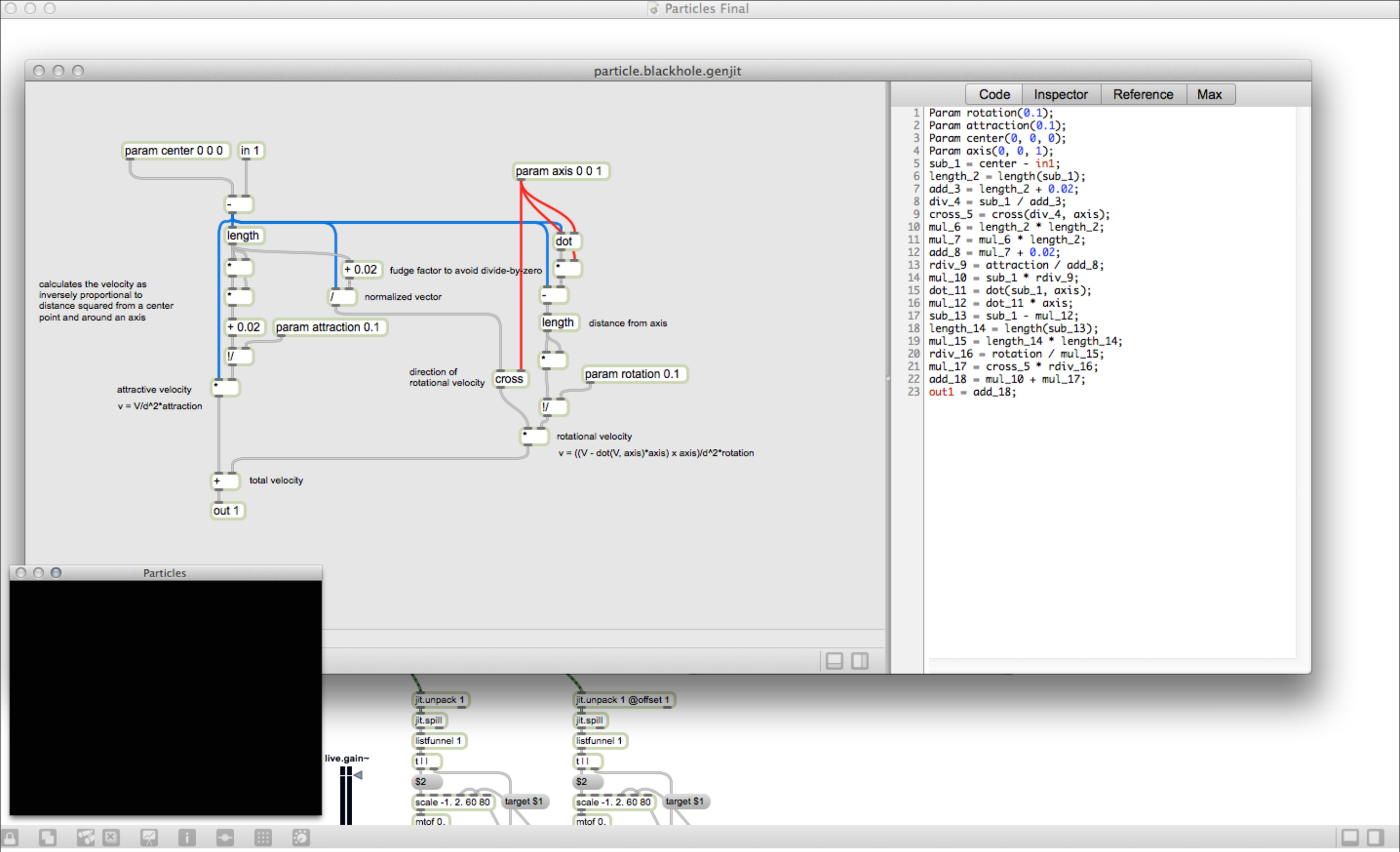
TOOLS

ZWEI WICHTIGE UNTERSCHIEDE

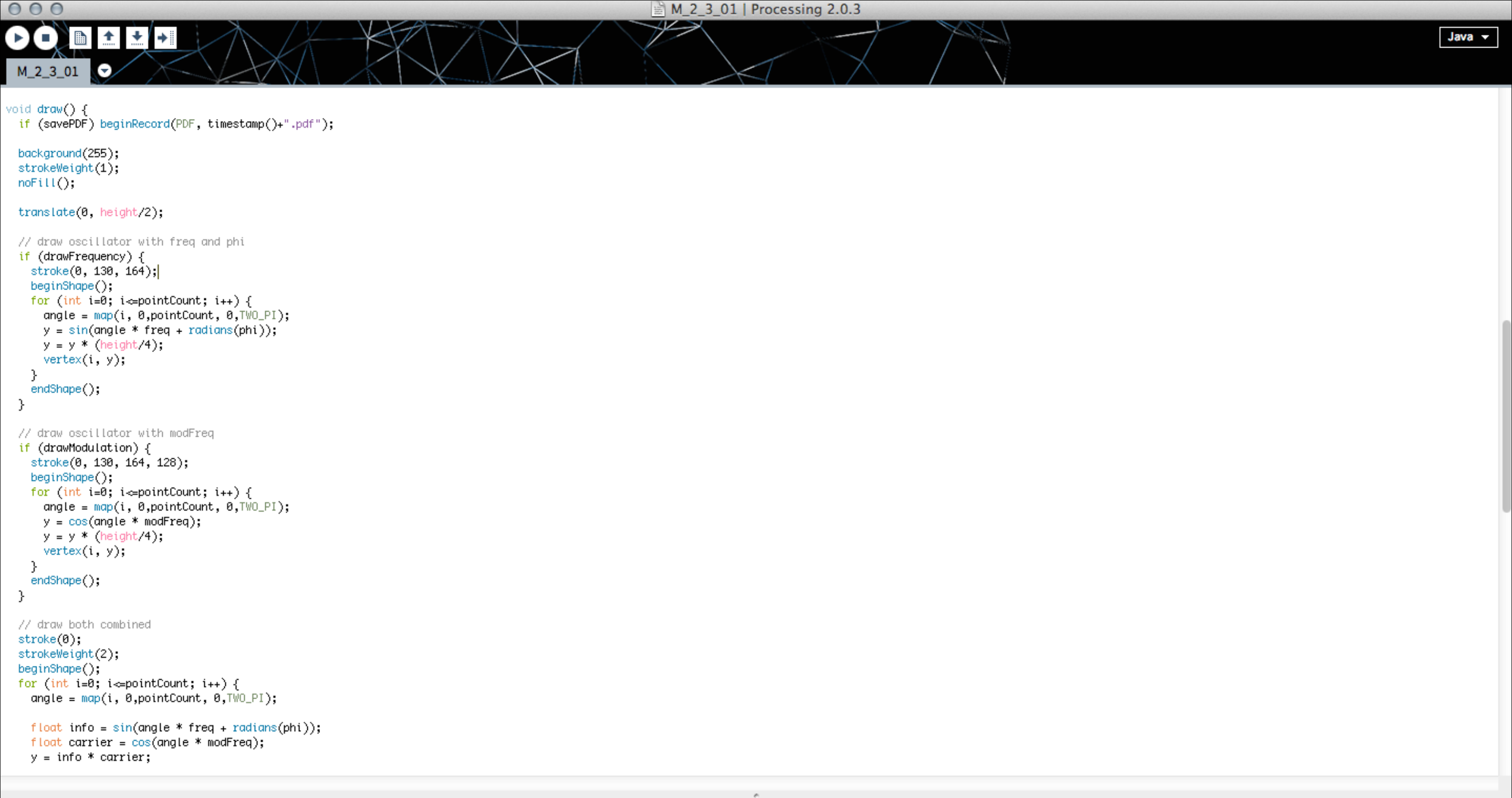


TOOLS BASIEREND AUF **NODES** (PATCHES)

GRAPHISCHE ENTWICKLUNGSUMGEBUNG



“**PATCHES**“ Auf Node basierten Tools entwickelte Anwendungen werden gewöhnlich als Patches bezeichnet . Patches bestehen aus einem Netzwerk von Knoten, die einfache oder komplexe Operationen durchführen. Patches können erstellt, bearbeitet und neu verbunden werden, während sie ausgeführt werden.



TOOLS BASIEREND AUF **CODE** (TEXT)

TEXT ENTWICKLUNGSUMGEBUNG

NODES

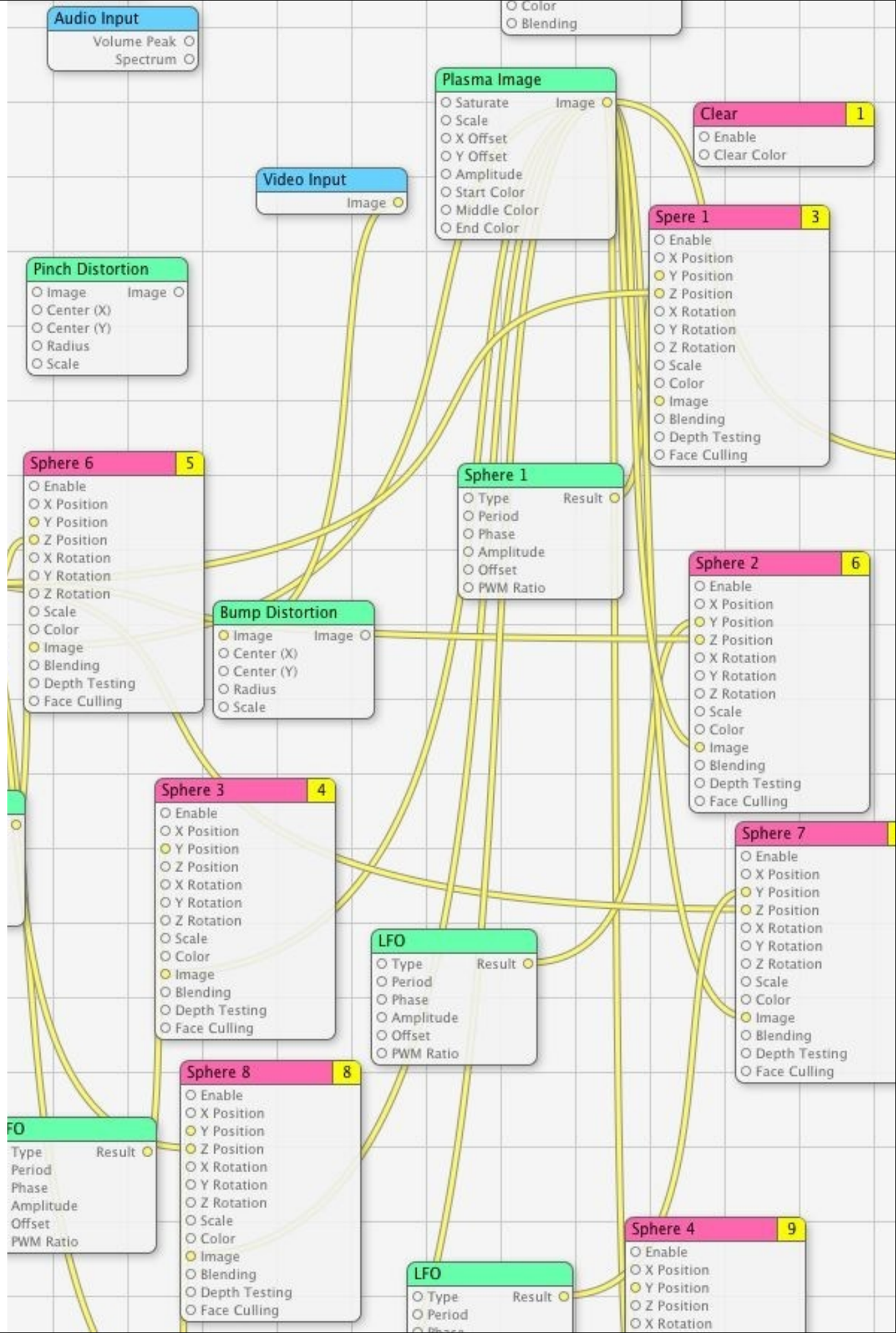
QUARTZ COMPOSER - MAX MSP - VVVV - VUO, PRAXIS LIVE, PURE DATA

QUARTZ COMPOSER

Quartz Composer ist ein Programm von Apple, mit dem man sogenannte Patches entwickeln kann, die dann von der Quartz Grafik-Engine ausgeführt werden.

Quartz ist die Grafikschicht des Apple Betriebssystems Mac OS X. Es handelt es sich um eine **Bibliothek für 2D- und 3D-Grafiken**, die das grundlegende Darstellungsmodell für Mac OS X bildet.

Plattform : Max OS X

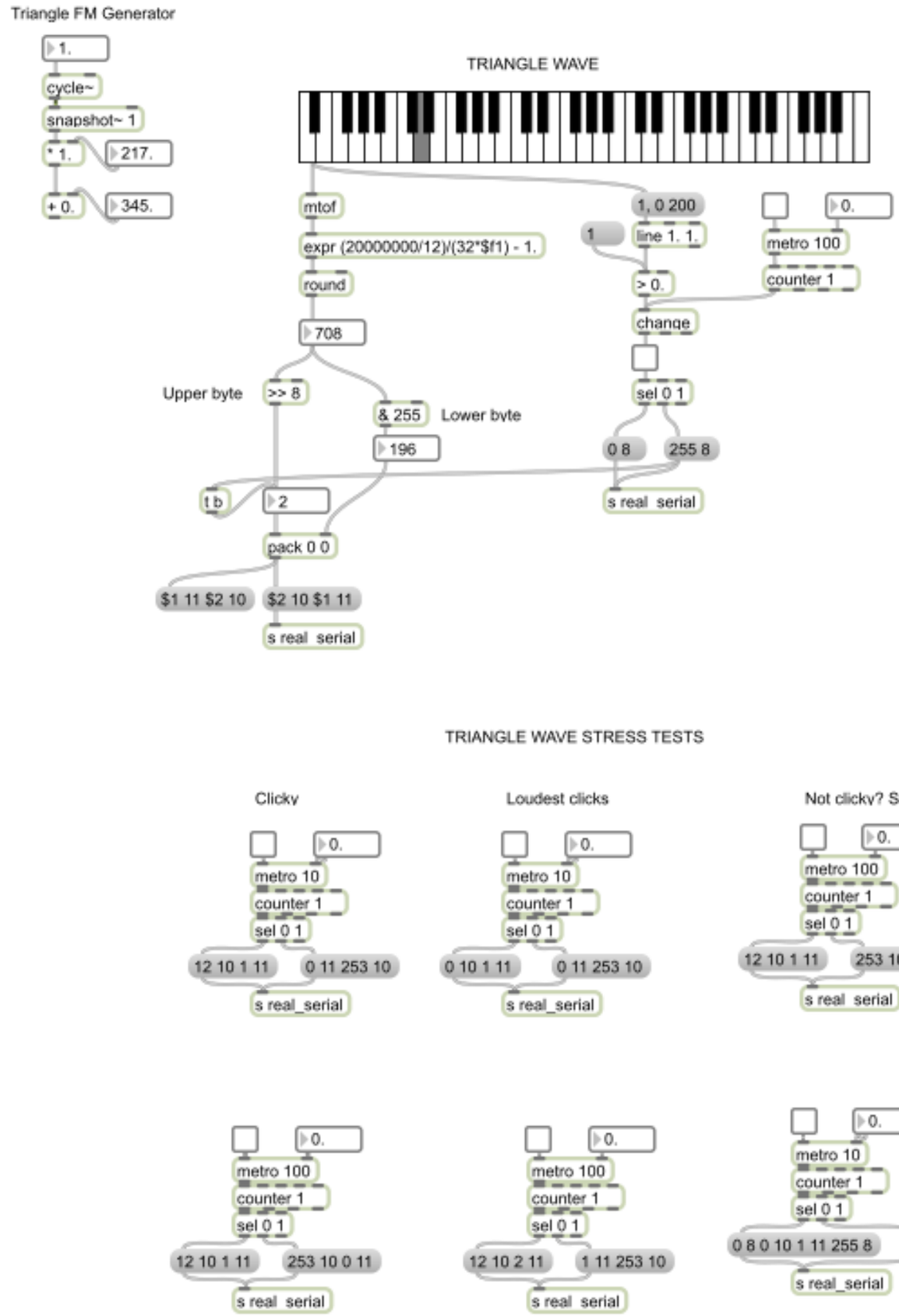


MAX | MSP | JITTER

Max | MSP ist eine graphische Entwicklungsumgebung für **Musik und Multimedia** von Cycling'74, die für **Echtzeitprozesse** ausgelegt ist. Sie wird seit 20 Jahren von Komponisten, Musikern, Softwareentwicklern und Künstlern eingesetzt, um **interaktive Software selbst zu erstellen** – unabhängig von den ästhetischen Vorgaben kommerzieller Produkte.

Insbesondere im **Live-Betrieb** ist diese Sprache in der zeitgenössischen **elektronischen Musik**, im Bereich des **Live-Video** und unter Laptop-Künstlern weit verbreitet.

Plattform: Max OS X, Windows

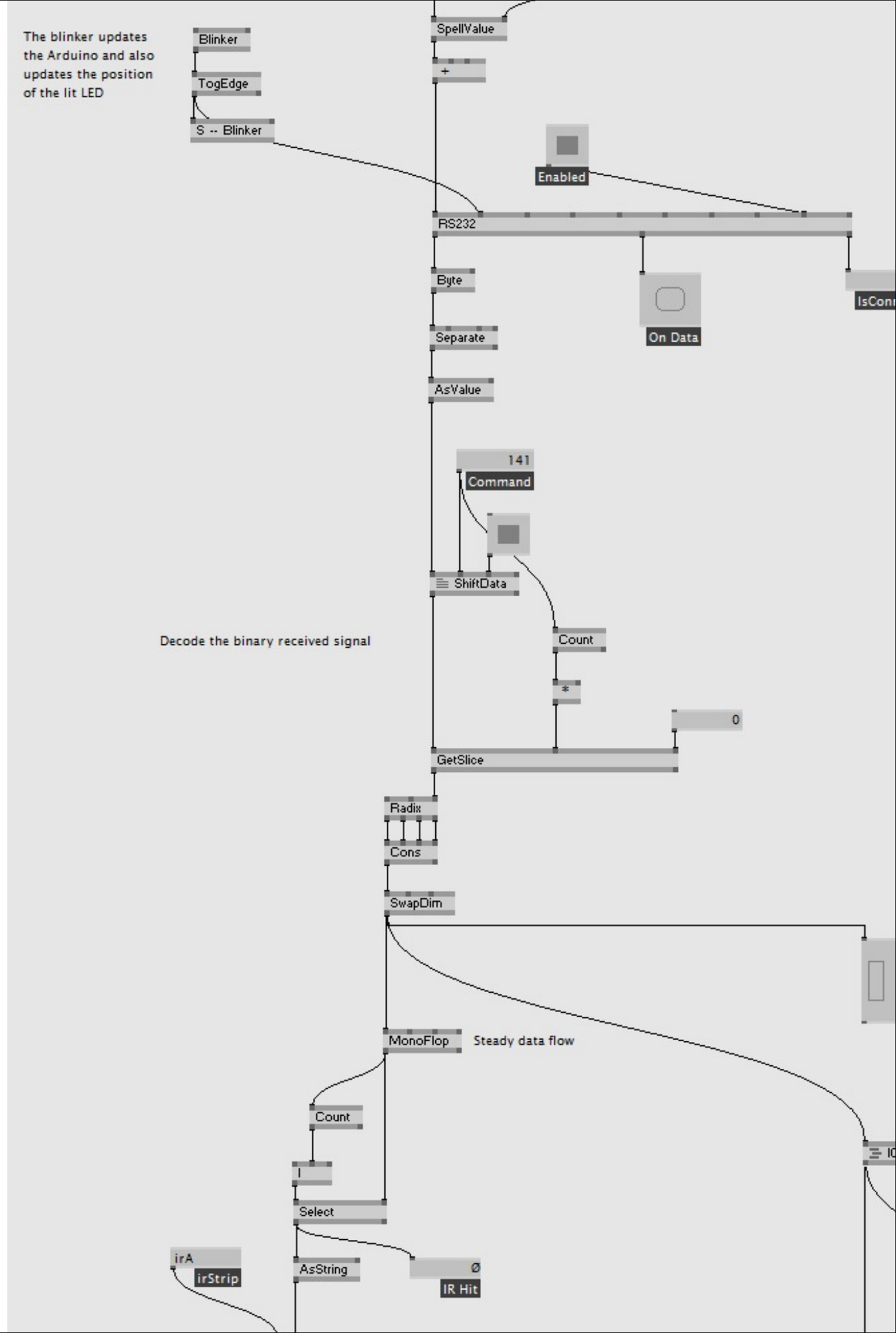


VVVV

VVVV (auch v4 oder v-vier) ist ein Hybrid-visuelle | Text Programmier-Toolkit für einfaches **Prototyping von interaktiven**, **visuellen Installationen** und generativen Kunst.

Es skaliert robust auch für große Umgebungen mit physikalischen Schnittstellen, **Echtzeit -Motion-Grafik-, Audio-und Video** und kann mit vielen Benutzern gleichzeitig interagieren.

Plattform: Windows (OS X und Linux angekündet)



CODE

PROCESSING, OPENFRAMEWORKS, CINDER, SUPERCOLLIDER, POLYCODE, NODEBOX

PROCESSING

Processing ist eine Open Source-Programmiersprache für die Programmierung von **Bildern, Animation und Sound**.

Es wurde speziell für Studenten, Künstler und Designer entwickelt. Durch den einfachen Aufbau eignet sich Processing gut als **Einstieg in die Grundstrukturen des Programmierens**.

Processing wurde von Künstlern und Designern entwickelt, um eine Alternative zu kommerzieller Software zu bieten.

Processing basiert auf einer Java-ähnlichen objektorientierten Programmiersprache.

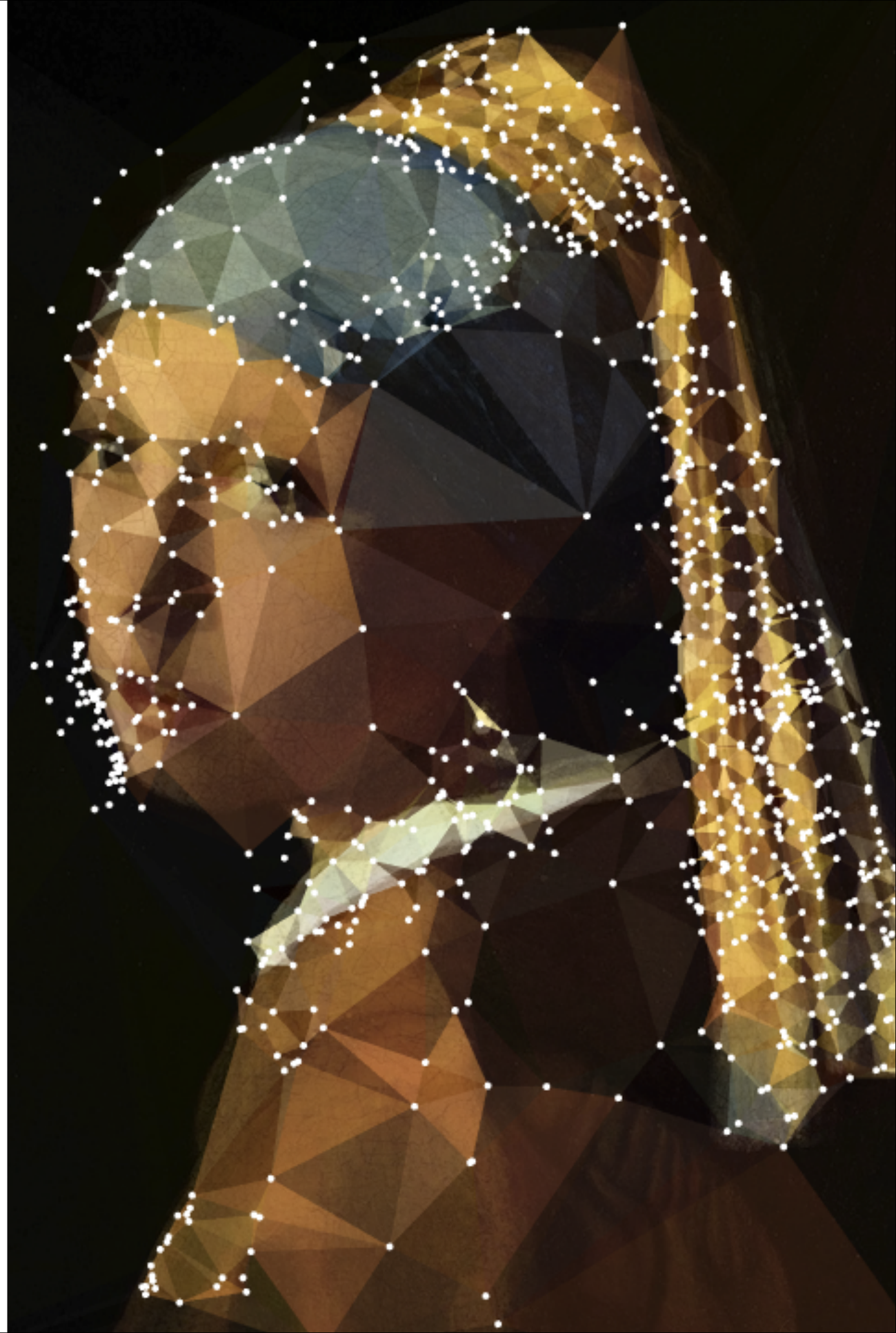
Plattform: Windows, Max OS X, Linux, IOS, Android, Arduino

CINDER

Cinder ist eine Programmierbibliothek entworfen um der Sprache **C++** erweiterte Visualisierung Fähigkeiten zu geben.

Cinder bietet eine leistungsstarke, **intuitiv Toolbox** für die **Programmierung von Grafik-, Audio-, Video-, Netzwerk-, Bildverarbeitung und Rechengeometrie .**

Plattform: Windows, Max OS X, IOS



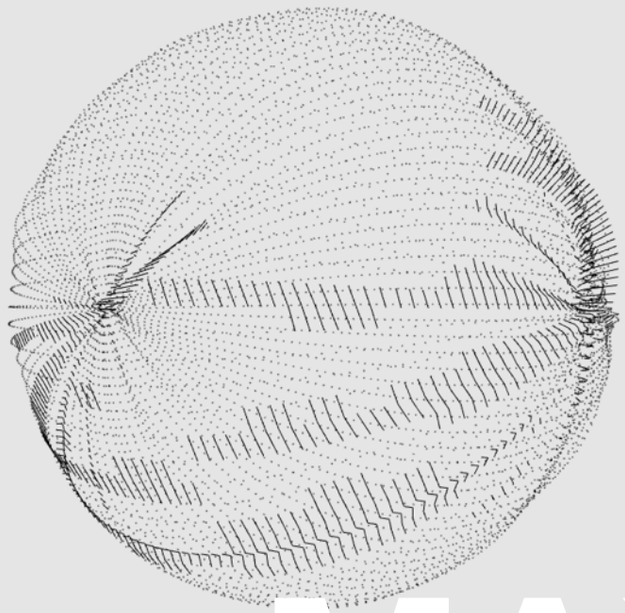
OPENFRAMEWORKS

Openframeworks ist ein Open-Source Toolkit konzipiert für "creative coding“ und ist in **C++** geschrieben.

Openframeworks bietet eine vereinfachte Schnittstelle zu leistungsfähigeren Bibliotheken für **Media Manipulation, Hardware Kontrolle und Netzwerkkommunikation.**

Es besteht aus einem Kern von Funktionen für **2D-und 3D-Grafik-, Sound und Videoverarbeitung.**

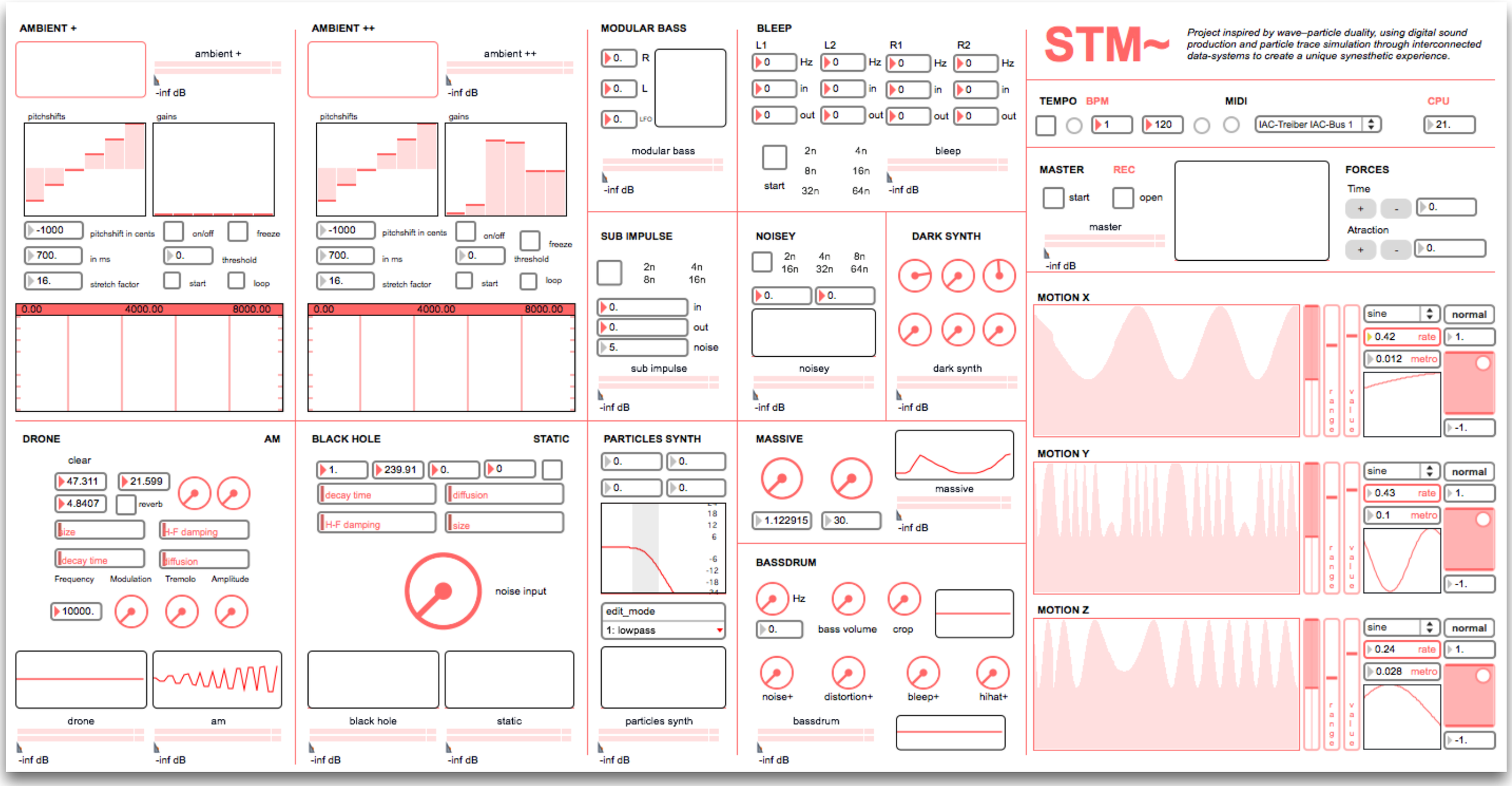
Plattform: Windows, Max OS X, Linux, IOS, Android



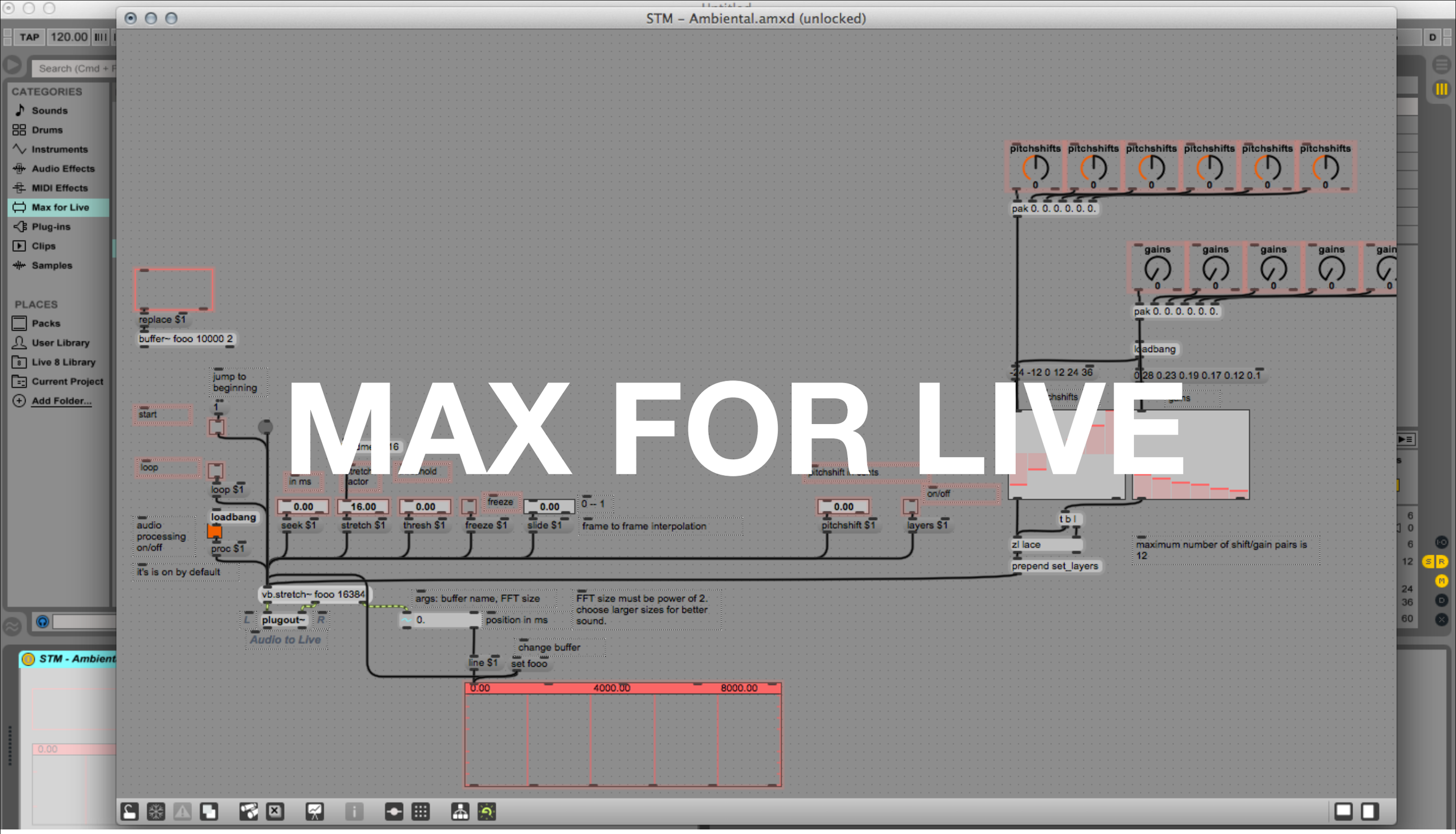
MAX JITTER

Videobearbeitung und 3D-Grafik in Echtzeit

Insbesondere im Live-Betrieb ist diese Sprache in der zeitgenössischen elektronischen Musik, im Bereich des Live-Video und unter Laptop-Künstlern weit verbreitet.



CUSTOM SOFTWARE INTERFACE FOR REAL TIME AUDIO PROCESSING



Max for Live erweitert **Ableton Live** um die aus **Max | MSP** bekannten Möglichkeiten um die Erstellung eigener virtueller Instrumente und Effekte.

Eigens kreierte Synthesizer, algorithmische Kompositions-Werkzeuge, direkt Interaktion mit Hardwaregeräten, generierte Live-Visuals, Licht Steuerung– all das ist möglich und nahtlos in Benutzeroberfläche und Workflow von Ableton Live integriert.