Luc Döbereiner

Innenraum

für Altsaxophon, Klavier, Schlagzeug und Live-Elektronik

2015

Innenraum

für Altsaxophon, Klavier, Schlagzeug und Live-Elektronik

dem ensemble werktag gewidmet

Allgemeines

Das Stück beschäftigt sich mit der Idee, die drei Instrumente als akustische Räume zu behandeln, die durch die Elektronik miteinander verbunden werden können. Das Spielen der Instrumente wird also nicht in erster Linie als Tonerzeugung aufgefasst, sondern als Transformation eines akustischen Raums. Das Saxophon, das Klavier, die Udu und das Tamtam werden durch Lautsprecher im Inneren und Körperschallwandler akustisch angeregt. Der Klang wird mit Mikrofonen abgenommen, die – wie die Lautsprecher – zum Teil zu einem spielbaren Teil des Instruments werden können. Alle elektronisch erzeugten bzw. verarbeiteten Klänge entstehen durch gesteuerte Rückkopplungen im Inneren eines oder mehrerer durch die Elektronik verbundener Instrumente.

Aufbau und Zeichenerklärung

Saxophon

Im Schallbecher des Altsaxophons wird ein diesen ausfüllender Lautsprecher angebracht. Bei der Entwicklung der Elektronik wurde ein Miniatur-Lautsprecher mit einem Durchmesser von 92mm (8 Ohm, 1 Watt, breitband) verwendet. Der Rand des Lautsprechers sollte mit Schaumstoff gepolstert werden, um Beschädigungen des Instruments und Störgeräusche zu vermeiden. Ein Elektret-Mikrofon wird im Inneren des Instruments am unteren Ende des S-Bogens angebracht. Der Klang wird häufig durch die Rückkopplung im Inneren des Saxophons erzeugt und durch bloßes Greifen (nicht Blasen) transformiert.

Eine frei gewählte Taste soll in dem durch den Kasten angzeigten Zeitraum langsam geöffnet werden.

Drei frei gewählte Tasten sollen in dem durch den Kasten angzeigten Zeitraum langsam geöffnet oder geschlossen werden (für jede Taste frei wählbar).

Vier frei gewählte Tasten sollen plötzlich geöffnet oder geschlossen werden (für jede Taste frei wählbar).

Vier frei gewählte Tasten sollen plötzlich geöffnet oder geschlossen werden (für jede Taste frei wählbar).

Schnelle, frei gewählte Griffwechsel während des angezeigten Zeitraums.

Zahntöne, der ungefähre Tonhöhenverlauf wird durch die Linie angezeigt.

Klavier

Mit Doppelklebeband wird von unten am Resonanzboden des Flügels ein Körperschallwandler angebracht (Visaton EX 45 S). Ein Elektret-Mikrophon hängt ca. 10 cm ungefähr über den Saiten des Tons C (c3). Der Klavierteil wird häufig in zwei Systemen notiert. Das untere System beschreibt die Aktionen auf der Tastatur oder die Saiten, die gespielt werden. Das obere System beschreibt die Art der Aktionen und den Ort auf den Saiten, der (im Inneren des Klaviers) gedämpft oder berührt wird.

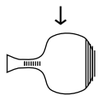
Mit einem dicken Gitarrenplektrum zwischen den angegenen Saiten reiben. Das Plektrum muss die entsprechende Größe haben (z.B. Dunlop JD Jazztone 207). Durch variieren des Drucks und der Geschwindkeit können verschiedene Texturen und Klangfarben erzeugt werden.

Mit der Handfläche im angegebenen Bereich auf die Saiten schlagen.

Taste(n) stumm drücken.

Udu

Die Udu (Schlagzwerk, U 80 Neck Udu) wird so gehalten, dass der Nacken nach links weist. In der linken Hand wird ein Miniatur-Lautsprecher (92mm, 8 Ohm, 1 Watt, breitband) gehalten, der auf Höhe der Nackenöffnung bewegt wird (der Abstand von der Öffnung ist im unteren System notiert). Ein Elektret-Mikrofon hängt durch das Spielloch in der Mitte der Udu auf Höhe der Nackenöffnung. Die rechte Hand spielt die Udu. Zwei Finger der rechten Hand tragen Fingerhüte.



Der Pfeil deutet den Ort der Aktion auf der Udu an.

Ein langsames Streichen der angebenen Stelle mit den Fingerhüten.

Ein schnelles Reiben oder Kratzen der angebenen Stelle mit den Fingerhüten.

Resonanter Schlag mit der Handfläche auf das Spielloch.

Wirbeltextur mit Fingerhüten und Finger.

Tamtam

Auf der Rückseite des Tamtams wird in der Mitte ein Körperschallwandler (Visaton EX 45 S) mit Doppelklebeband angebracht. Das Tamtam wird nie geschlagen. Der Schlagzeuger hat ein Elektretmikrofon in der Hand und spielt das Tamtam nur durch Veränderungen des Abstands zwischen Mikrofon und Tamtam und beinflusst somit die entstehenden Rückkopplungen. In einem Sytem wird die "horizontale" Position des Mikrofons beschrieben (zwischen Rand und Mittelpunkt des Tamtams). Im anderen System wird der Abstand des Mikrofons vom Tamtam in der Tiefe beschrieben. Außerdem wird an einer Stelle eine kreiförmige Bewegung über dem Rand des Tamtam notiert.

Innenraum

für Altsaxophon, Klavier, Schlagzeug und Live-Elektronik

Luc Döbereiner (2015)

Zeit in Sekunden

A

0 23 24 25 31 32 33 39 40 41 47 48 49 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 76 81 88

Altsaxophon

⬆1 ⬆1 ⬆1 ⬆1 ⬇1 ⬇1 ⬇1 ⬆2 ⬆2 ⬇2

Bb, C

Nur greifen
Mundstück nicht im Mund

Mundstück in den Mund nehmen
Mit der Zunge das Rohrblatt drücken

max:
Zungendruck
min:

88 97 102 109 118 126 134 142 150 154 159 165 173 183 195 210

A.S.

⬆/⬇3 ⬆/⬇3 ⬆/⬇3 ⬆/⬇3

Zirkularatmung
Zahntöne

C1, B, B, Bb

langsam umgreifen →

Nur greifen
Mundstück nicht im Mund

pp f p ff pp ff p | p f p | mp ff mp | p ff p

B

210 217 224 232 235 240 257 265 275 295 305

Klavier

Innen

Saiten mit der rechten Hand dämpfen/Flageoletts erzeugen

Max. Reichweite

Druck:

Tastatur

Hammer

p mf f mp fff | fff p fff p mf | pp f pp f pp ff pp

Ped.

C

305 310 315 323 329 337 340 341 349 352 365 373 375 385 390 400

Klv.

Innen

Saiten mit der rechten Hand dämpfen/Flageoletts erzeugen

Max. Reichweite

Druck:

Mit dem Plektrum zwischen den beiden Saiten reiben

Mit der Hand auf die Saiten schlagen, dann dämpfen

Tastatur

Hammer

mf f p mp | f pp (nicht spielen) | ff | ff | ff

Ped.

Udu

Rechte Hand
Udu

Durch gelegentliche Schläge mit dem Fingerhut auf die Schlagfläche unterbrechen

Spielloch nach dem Schlag schließen, dann langsam heben

Entfernung vom Loch
15 cm
Spielloch

f

Übergang →

Übergang →

p f mp fff

Entfernung vom Loch
15 cm
Spielloch

ff

50 cm

Linke Hand
Lautsprecher

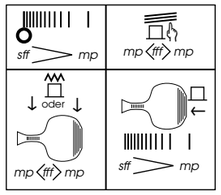
Öffnung

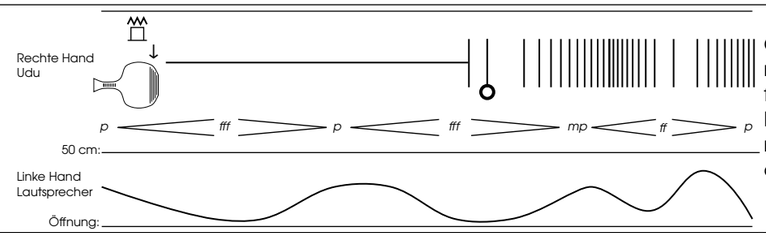
D

400 435 455 470 510 518

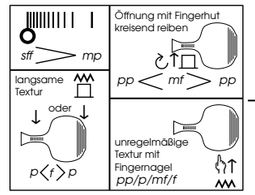
Udu

Gesten und Texturen (2-7 Sekunden) mit Pausen (1-5 Sekunden), Materialien frei wählen und wechseln. Den Lautsprecher während der Gesten nah an der Öffnung halten, während der Pausen entfernt halten.





Gesten und Texturen (3-8 Sekunden) mit Pausen (1-5 Sekunden), Materialien frei wählen und wechseln. Den Lautsprecher während der Gesten nah an der Öffnung halten, während der Pausen entfernt halten.



Lautsprecher ca. 15 cm von der Öffnung entfernt halten.

Klv.

Ped.

E

518 522 533 538 550 557 568 578 584 597 601 616 621 633 640 656 668 674 678 683 687 693 698 702 708 718

A.S.

Während der Pausen nur Greifen, Mundstück nicht im Mund

Luft "pp" "f" "pp" Luft "pp" "f" "pp" Luft "pp" "f" "pp" Luft "pp" "f" "pp" Luft "pp" "f" "pp" Luft "pp" "f" "pp" Luft "pp" "f" "pp" "sff" "pp" "pp" "f" "pp"

langsam umgreifen langsam umgreifen langsam umgreifen

pp f mf ff pp f pp

⬆/⬇4 ⬆/⬇4 ⬆/⬇4 ⬆/⬇4 ⬆/⬇4

Klv. Innen / Tastatur

Max. Reichweite

Mit dem Plektrum langsam zwischen den beiden Saiten reiben, hoher Druck, knarzend.

Hammer

sfz p pp

Sost.Ped. (nicht spielen) 1/2 1/2 Sost.Ped. f

Ped.

F

718 728 738 748 758 768 788 808 818 838 873 938

Tamtam

Abstand des Mikros vom Mittelpunkt des Tamtams in der Breite

Abstand des Mikros vom Tamtam in der Tiefe

Rand

Das Mikrofon langsam über dem Rand des Tamtams kreisen lassen, 1 Umdrehung in 20 Sekunden.

Mittelpunkt

1 m

Tamtam

A.S.

Tremolo, Geschwindikeit variieren, nur Greifen, Mundstück nicht im Mund

so schnell wie möglich

Wechsel 2 Sek

Klv.

Ped.

```
////////// INNENRAUM -- LIVE-ELECTRONICS
////////// (c) 2015 by Luc D bereiner

(

s.waitForBoot({

s.meter;

//////// INPUTS and OUTPUTS
~saxIn = 3;
~uduIn = 2;
~beckenIn = 4;
~pianoIn = 5;

~saxOut = 3;
~uduOut = 0;
~beckenOut = 1;
~pianoOut = 2;

// SYNTHS & BUFFERS
z = Buffer.sendCollection(s, Env.perc(0.1,1).discretize, 1);
~grain1 = Buffer.alloc(s,44100 * 15,1);
~grain2 = Buffer.alloc(s,44100 * 15,1);

// SAX
~saxRec = Buffer.alloc(s,44100*20);

SynthDef("saxSec1", {arg out=0,trigFreq=(1/10),end = 0, overlap=1,input = 0,
                compmax=0.2,ampFac=0.9,minFreq=1880,maxFreq=2500,lagTime=3,
                lfFreq=0.02,fisMin=1,fisMax=4,ringFreq=180,
                ring=(-1),timeDev=1,rec=0,recB;

                var endEnv, freqs,sound,sound2,env,trig,in = SoundIn.ar(input);
                endEnv = EnvGen.ar(Env([1,1,0,0],[2,8,10],[1,1,-2]),end,
                        doneAction: 2);

trig = Impulse.ar(trigFreq.lag2(lagTime) *
        LFNoise1.kr(0.2).range(1/timeDev.lag2(lagTime),timeDev
                .lag2(lagTime)));
env = EnvGen.ar(Env([0.002,1,1,0.002],[0.15,0.6,0.25],
        [4,2,-4]),trig,timeScale:TExpRand.kr(0.5,1) *
        (1/trigFreq.lag2(lagTime))*overlap.lag2(lagTime));

sound = in * env * endEnv;
sound = XFade2.ar(sound,SinOsc.ar(ringFreq,0,0.5,0.5) *
        sound,ring.lag2(lagTime));
sound = SinOsc.ar(0,sound*LFNoise2.ar(lfFreq.lag2(lagTime))
        .exprange(fisMin.lag2(lagTime),fisMax.lag2(lagTime)));
sound = Compander.ar(sound,sound,0.09,1,0.4);
sound = EnvGen.ar(Env.new([0,1],[3])) * sound;
freqs = LFDNoise3.kr(lfFreq.lag2(lagTime)!13)
.exprange(minFreq.lag2(lagTime),maxFreq.lag2(lagTime));
sound = BHiPass4.ar(sound,
        [freqs[0],freqs[1],freqs[2],freqs[3],freqs[4],freqs[5],
                freqs[6],freqs[7],freqs[8],
                freqs[9],freqs[10],freqs[11]]);
sound = BLowPass4.ar(sound,
        [freqs[1],freqs[2],freqs[3],freqs[4],freqs[5],
                freqs[6],freqs[7],freqs[8],
                freqs[9],freqs[10],freqs[11],freqs[12]]);
sound = Compander.ar(sound,sound,0.001,5,1); // noise gate
sound2 = Compander.ar(sound,sound,0.09,0.2,-1.5,4,7) +
(sound * 0.3);
sound2 = sound2 * LFTri.kr(LFNoise2.kr(0.005!12)
        .exprange(0.03,0.1),Rand(0!12,2))
.exprange(0.35,1.25);
sound2 = Mix.ar(sound2) +(in*0.15);
sound2 = Compander.ar(sound2,sound2,0.11,1,0.5) * 2.5;
sound2 = Compander.ar(sound2,sound2,compmax.lag2(lagTime),
        0.8,-0.25,0.01,3);
sound2 = (BLowPass.ar(sound2,2500,4) * 0.6) +
```



```
                (BLowPass.ar(sound2,12000) * 0.52);
                sound2 = EnvGen.ar(Env.new([0,0,1],[4,3])) * sound2;
                sound2 = Limiter.ar(sound2);
                RecordBuf.ar(sound2, recB, 0, 1, 0, rec);
                Out.ar(out, sound2*ampFac.lag2(lagTime)*endEnv);
}).load(s);

SynthDef("saxPiano", {arg out=0, amp=1, in=0, overlap = 0.5,
                trigFreq = 0.25,timeDev=2, grainB,shift=5000,
                grainAmp = 0.05,maxRing=800,minFis=10,maxFis=80,lagTime=10,
                end=0,inFac=1;
                var freq,grain,onset,noiseSound,env,trig,endEnv,
                sound = SoundIn.ar(in)*inFac;

                sound = Compander.ar(sound,sound,0.002,5,1,0.01,0.2);
                sound = Compander.ar(sound,sound,0.1,0.45,-1.0,0.01,2)*2;
                RecordBuf.ar(sound*1.2,grainB);
                onset = Onsets.kr(FFT(LocalBuf(2048),sound),0.2);

                trig = Impulse.ar(trigFreq.lag2(lagTime) *
                        LFNoise1.kr(0.2).range(1/timeDev.lag2(lagTime),
                                timeDev.lag2(lagTime)));
                env = EnvGen.ar(Env([0.002,1,1,0.002],[0.15,0.6,0.25],
                        [4,2,-4]),trig,
                        timeScale:TExpRand.kr(0.5,1)*(1/trigFreq.lag2(lagTime))
                        * overlap.lag2(lagTime));
                endEnv = EnvGen.ar(Env([1,1,0,0],[2,8,6],[1,1,1]),end,
                        doneAction: 2);

                sound = sound * env;

                // noiseSound
                freq = Demand.kr(onset,0,
                        Dshuf([500,2000,200,100,80,1000,700,3000], inf))
                .lag2(0.01).clip(50,7000);
                noiseSound = BBandPass.ar(sound,freq,1);
                noiseSound = noiseSound *
                SinOsc.ar(TExpRand.kr(10,maxRing.lag2(lagTime),onset),
                        0,0.5,0.5);
                noiseSound = SinOsc.ar(0,noiseSound*LFNoise2.ar(0.25)
                        .range(minFis.lag2(lagTime),maxFis.lag2(lagTime)));
                noiseSound = Compander.ar(noiseSound,noiseSound,
                        0.1,1,0.2,0.01,0.01);

                // sound
                sound = BLowPass.ar(sound,18000);
                sound = BHiPass.ar(sound,20);
                sound = FreqShift.ar(sound,
                        TRand.kr(shift*(-1).lag2(lagTime),shift
                                .lag2(lagTime),onset));

                // grain
                grain = GrainBuf.ar(1,Dust.ar(TRand.kr(10,100,onset)),
                        TRand.kr(0.01,0.07,onset),grainB,TRand.kr(0.25,2,onset),
                        TRand.kr(0,1,onset),envbufnum: z.bufnum);

                // mix
                sound = sound + (noiseSound * 0.6) + (grain * grainAmp);
                sound = BPeakEQ.ar(sound,6000,2,-9);
                sound = FreeVerb.ar(sound*endEnv,0.05,0.95);
                sound = Limiter.ar(sound, 0.9);

                Out.ar(out, sound*amp.lag2(lagTime));
}).send(s);

// PIANO
~pianobuf = Buffer.alloc(s,44100 * 40,1);
~pianograin = Buffer.alloc(s,44100 * 10,1);

SynthDef("pianoSec2", {arg out=0,trigFreq=0.2,overlap=2,
                input=0,compmax=0.2,ampFac=1,pianobuf,timeDev=1,end=0,
```

```
minFreq=1880,maxFreq=2500,lagTime=3,lfFreq=0.02,fisMin=1,
fisMax=4,ringFreq=180,ring=(-1);
var freqs,sound,ringSound,onset,sound2,env,trig,endEnv,
in = SoundIn.ar(input) * 1.1;

endEnv = EnvGen.ar(Env([1,1,0.75,0],[1,14,10],[1,-2]),
        end,doneAction: 2);

trig = Impulse.ar(trigFreq.lag2(lagTime) *
        LFNoise1.kr(1).range(1/timeDev.lag2(lagTime),
                timeDev.lag2(lagTime)));
env = EnvGen.ar(Env([0.002,1,1,0.002],[0.15,0.6,0.25],
        [4,2,-4]),trig,
        timeScale:TExpRand.kr(0.5,1)*(1/trigFreq.lag2(lagTime))
        * overlap.lag2(lagTime));
sound = in * endEnv;
sound = XFade2.ar(sound,SinOsc.ar(ringFreq,0,0.5,0.5)
        * sound,ring.lag2(lagTime));
ringSound = sound;
sound = SinOsc.ar(0,sound*LFNoise2.ar(lfFreq.lag3(lagTime))
        .exprange(fisMin.lag3(lagTime),fisMax.lag2(lagTime)));
sound = Compander.ar(sound,sound,0.2,1,0.5);
sound = EnvGen.ar(Env.new([0,1],[3])) * sound;
freqs = LFDNoise3.kr(LFDNoise3.kr(0.2)
        .exprange((lfFreq.lag3(lagTime)*0.75)!13,lfFreq
                .lag3(lagTime)*1.5))
.exprange(minFreq.lag2(lagTime),
        maxFreq.lag2(lagTime));

sound = BHiPass4.ar(sound,
[freqs[0],freqs[1],freqs[2],freqs[3],freqs[4],
                freqs[5],freqs[6],freqs[7],freqs[8],
                freqs[9],freqs[10],freqs[11]]);
sound = BLowPass4.ar(sound,[freqs[1],freqs[2],freqs[3],freqs[4],
        freqs[5],freqs[6],freqs[7],freqs[8],
        freqs[9],freqs[10],freqs[11],freqs[12]]);
                sound = Compander.ar(sound,sound,0.001,5,1); // noise gate
                sound2 = Compander.ar(sound,sound,0.09,0.25,-1.3,3,6)
                + (sound * 0.25);

                sound2 = sound2*LFTri.kr(LFNoise2.kr(0.005!12)
                        .exprange(0.03,0.1),Rand(0!12,2))
                .exprange(0.25,1.25);

                sound2 = Mix.ar(sound2) +(in*0.1);
                sound2 = Compander.ar(sound2,sound2,0.2,1,0.5) * 2.0;

                sound2 = Compander.ar(sound2,sound2,compmax.lag2(lagTime),
                         0.6,-0.25,0.01,3);
                sound2 = (BLowPass.ar(sound2,3500,4) * 0.6) + (sound2 * 0.4);
                sound2 = FreeVerb.ar(sound2,0.01,0.95);
                sound2 = EnvGen.ar(Env.new([0,0,1],[4,3])) * sound2;
                sound2 = Limiter.ar(sound2,0.95);
                RecordBuf.ar(sound2,pianobuf);
                ringSound = Compander.ar(ringSound,ringSound,0.1,1,-1.0,0.01,1);
                ringSound = BLowPass.ar(BPeakEQ.ar(ringSound,6000,2,-8),12000,1);
                sound2 = (sound2 * 0.8) + (ringSound * 0.75 * env);
                Out.ar(out, sound2*ampFac.lag2(lagTime)*endEnv*0.9);
}).load(s);

SynthDef("pianoDuo", {arg out=0, in=0,amp=0.9,lagTime=3,minRing=10,
                maxRing=800,minShift=(-5000),maxShift=5000,soundAmp=1,
                noiseSoundAmp=0.05,minFis=20,maxFis=80,
                onsetThreshold=0.25,maxFreq=7000,end=0;
                var endEnv,env,freq,freq2,onset,noiseSound,
                sound = SoundIn.ar(in);

                endEnv = EnvGen.ar(Env([1,1,0.25,0],[1,4,10],[1,-2]),end,
                        doneAction: 2);

                sound = Compander.ar(sound,sound,0.001,5,1,0.01,0.2);
```



```
sound = Compander.ar(sound,sound,0.2,0.4,-1.2,0.01,0.1)*3.5;

onset = Onsets.kr(FFT(LocalBuf(2048),sound),onsetThreshold);

// noiseSound
freq = Demand.kr(onset,0,
        Dshuf([500,2000,200,100,80,1000,700,3000], inf))
.lag2(0.01).clip(50,maxFreq);
freq2 = Demand.kr(onset,0,
        Dshuf([500,2000,50,100,1000,700,3000], inf)).lag2(0.01)
.clip(50,7000);
noiseSound = BBandPass.ar(sound,freq,1);
noiseSound = noiseSound *
SinOsc.ar(TExpRand.kr(minRing,maxRing,onset),0,0.5,0.5);
noiseSound = SinOsc.ar(0,noiseSound*
        LFNoise2.ar(0.25)
        .range(minFis.lag2(lagTime),maxFis.lag2(lagTime)));
noiseSound = Compander.ar(noiseSound,noiseSound,
        0.1,1,0.2,0.01,0.01);

// sound
sound = BLowPass.ar(sound,18000);
sound = BHiPass.ar(sound,20);
sound = FreqShift.ar(sound,TRand.kr(minShift,maxShift,onset));

env = Select.kr(Demand.kr(onset,0,
        Dwrand([0,1,2],[0.35,0.35,0.3],inf)),[EnvGen.kr(
                Env.linen(0.01,2,2),onset,
                timeScale:TRand.kr(0.5,2,onset)),
                EnvGen.kr(Env.linen(0.01,2,2),onset,
                        timeScale:TRand.kr(0.5,2,onset)),
                EnvGen.kr(Env.linen(2,2,1),onset,
                        timeScale:TRand.kr(0.5,2,onset))]);

// mix
sound = (sound * soundAmp.lag2(lagTime))
                + (noiseSound * noiseSoundAmp.lag2(lagTime));
                sound = BPeakEQ.ar(sound,5000,2,-8);
                sound = FreeVerb.ar(sound*env*endEnv,0.03,0.93);
                sound = Limiter.ar(sound*amp.lag2(lagTime), 1);

                Out.ar(out, sound);
}).send(s);

// UDU
~recb = Buffer.alloc(s,44100*8);
~uduRec = Buffer.alloc(s,44100*20);

SynthDef("uduS1", {arg out=0, in=0, amp=0.9,lagTime=3,minRing=10,
                maxRing=800,minShift=(-5000),maxShift=5000,soundAmp=1,
                noiseSoundAmp=0.05,minFis=20,maxFis=80,
                onsetThreshold=0.45,end=0,maxFreq=7000;
                var endEnv,freq,freq2,onset,noiseSound,
                sound = SoundIn.ar(in);

                endEnv = EnvGen.ar(Env([1,1,0.25,0],[1,4,10],[1,-2]),end,
                        doneAction: 2);

                sound = Compander.ar(sound,sound,0.0015,5,1,0.01,0.2);
                sound = Compander.ar(sound,sound,0.28,0.4,-1.1,0.01,0.1) * 3.5;

                onset = Onsets.kr(FFT(LocalBuf(2048),sound),onsetThreshold);

                // noiseSound
                freq = Demand.kr(onset,0,
                        Dshuf([500,2000,200,100,80,1000,700,3000], inf))
                .lag2(0.01).clip(50,maxFreq);
                freq2 = Demand.kr(onset,0,
                        Dshuf([500,2000,50,100,1000,700,3000], inf))
                .lag2(0.01).clip(50,7000);
                noiseSound = BBandPass.ar(sound,freq,1);
                noiseSound = noiseSound *
```

```
                SinOsc.ar(TExpRand.kr(minRing,maxRing,onset),0,0.5,0.5);
                noiseSound = SinOsc.ar(0,noiseSound * LFNoise2.ar(0.5)
                        .range(minFis.lag2(lagTime),maxFis.lag2(lagTime)));
                noiseSound = Compander.ar(noiseSound,noiseSound,
                        0.1,1,0.2,0.01,0.01);

                // sound
                sound = BLowPass.ar(sound,18000);
                sound = BHiPass.ar(sound,20);
                sound = FreqShift.ar(sound,TRand.kr(minShift,maxShift,onset));

                // mix
                sound = (sound * soundAmp.lag2(lagTime))
                + (noiseSound * noiseSoundAmp.lag2(lagTime));
                sound = BPeakEQ.ar(sound,5000,2,-11);
                sound = FreeVerb.ar(sound*endEnv,0.03,0.85);
                sound = Limiter.ar(sound*amp.lag2(lagTime), 0.95);

                Out.ar(out, sound);
}).send(s);

SynthDef("grainUdu", {arg out=0,in=0,minShift=(-5000),maxShift=5000,
                lagTime=8,amp=0.5,end=0,minTrig=1,maxTrig=80,
                minRate=0.1,maxRate=20,buf1,buf2,envbuf,maxDur=0.1;
                var endEnv,soundAmp,onset,sound1,sound2,sound3,trig,trig2,
                sound = SoundIn.ar(in);
                soundAmp =      Amplitude.ar(sound).lag2(0.5);
                sound = Compander.ar(sound,soundAmp,0.18,1,0.2,1.0,1.0)*12;
                RecordBuf.ar(sound,buf1);

                RecordBuf.ar(Compander.ar(sound*0.9,sound,0.4,1,0.5),buf2);

                endEnv = EnvGen.ar(Env([1,1,0,0],[2,15,6],[1,-2,1]),
                        end,doneAction: 2);

                onset = Onsets.kr(FFT(LocalBuf(2048),sound),0.25);

                sound = sound * SinOsc.ar(LFNoise2.kr(0.1).exprange(0.25,16));
                sound = [BBandPass.ar(sound,
                        LFNoise1.ar(0.1).exprange(30,10000),0.4) * 2 *
                        LFNoise2.ar(0.1).exprange(0.25,1),
                        BBandPass.ar(sound,LFNoise1.ar(0.1)
                                .exprange(30,10000),0.4) * 2 *
                        LFNoise2.ar(0.1).exprange(0.5,1),
                        BBandPass.ar(sound,LFNoise1.ar(0.1)
                                .exprange(30,10000),0.4) * 2 *
                        LFNoise2.ar(0.1).exprange(0.5,1),
                        BBandPass.ar(sound,LFNoise1.ar(0.1)
                                .exprange(30,10000),0.4) * 2 *
                        LFNoise2.ar(0.1).exprange(0.5,1),
                        BBandPass.ar(sound,LFNoise1.ar(0.1)
                                .exprange(30,10000),0.4) * 2 *
                        LFNoise2.ar(0.1).exprange(0.5,1),
                        BBandPass.ar(sound,LFNoise1.ar(0.1)
                                .exprange(80,10000),0.4) * 2 *
                        LFNoise2.ar(0.1).exprange(0.5,1)];
                sound = Compander.ar(sound,sound,0.1,1,-1.4,0.01,0.1);

                sound = Mix.ar(sound);
                sound = FreqShift.ar(sound,TRand.kr(minShift.lag2(lagTime),
                        maxShift.lag2(lagTime),onset));

                sound = (BLowPass.ar(sound,1000) + (sound * 0.5));
                sound = Compander.ar(sound,sound,0.4,1,0.4,0.1,0.1);
                trig = Impulse.kr(LFNoise1.kr(LFDNoise3.kr(0.1)
                        .exprange(0.05,2))
                        .exprange(minTrig.lag2(lagTime),maxTrig.lag2(lagTime)));
                trig2 = Impulse.kr(LFNoise1.kr(LFDNoise3.kr(0.1)
                        .exprange(0.05,2))
                        .exprange(minTrig.lag2(lagTime),maxTrig.lag2(lagTime)));

                sound = (GrainBuf.ar(1,trig,LFNoise2.kr(0.2)
```



```
                        .range(0.01,maxDur.lag2(lagTime)),buf1,
                        TExpRand.kr(minRate.lag2(lagTime),maxRate
                                .lag2(lagTime),onset),
                        LFNoise2.kr(0.1).range(0,1),envbufnum:envbuf)
                        * LFNoise2.ar(0.1).exprange(0.35,1) ) +
                (GrainBuf.ar(1,trig2,LFNoise2.kr(0.2)
                        .range(0.01,maxDur.lag2(lagTime)),buf1,
                        TExpRand.kr(minRate.lag2(lagTime),maxRate
                                .lag2(lagTime),onset),
                        LFNoise2.kr(0.1).range(0,1),envbufnum:envbuf)
                        * LFNoise2.ar(0.1).exprange(0.35,1) ) +
                (sound * LFNoise2.ar(0.1).exprange(0.7,1.0));

                sound = (AllpassC.ar(sound,0.2,LFNoise1.kr(0.15)
                        .exprange(0.01,0.2),2.5) * 0.35) + sound;
                sound = Limiter.ar(sound,0.9);
                Out.ar(out, sound*amp.lag2(lagTime)*endEnv*1.2);
}).send(s);

// BECKEN
SynthDef("becken1", {arg out=0,in=0,amp=1,minFreq=85,maxFreq=13200,
                end=0,maxFis=6,speed=0.1,lagTime=10,maxSpeed=2.0;
                var trigEnv,soundAmp,sound,endEnv;
                sound = SoundIn.ar(in);

                endEnv = EnvGen.ar(Env([1,1,0,0],[2,8,10],[1,1,-2]),end,
                        doneAction: 2);

                sound = (Compander.ar(sound,sound,0.3,0.7,-0.5,0.1,1.0) * 0.3)
                + (sound * 0.7);
                sound = ((sound * SinOsc.ar(110,0,0.5,0.5) ) * 0.3)
                + (sound * 0.7);

                sound = SinOsc.ar(0,sound*LFNoise2.ar(LFDNoise3.ar(0.2)
                        .exprange(0.2,3))
                        .exprange(1,maxFis.lag2(lagTime)));
                sound = RemoveBadValues.ar(BBandPass.ar(sound,LFNoise2.ar(
                        LFNoise2.ar(speed.lag2(lagTime))
                        .range(0.05,maxSpeed.lag2(lagTime)))
                        .exprange(minFreq.lag2(lagTime),maxFreq.lag2(lagTime)),0.5))
                + RemoveBadValues.ar(BBandPass.ar(sound,
                        LFNoise2.ar(LFNoise2.ar(speed.lag2(lagTime))
                        .range(0.05,maxSpeed.lag2(lagTime)))
                        .exprange(minFreq.lag2(lagTime),maxFreq.lag2(lagTime)),0.5))
                + RemoveBadValues.ar(BBandPass.ar(sound,
                        LFNoise2.ar(LFNoise2.ar(speed.lag2(lagTime))
                                .range(0.05,maxSpeed.lag2(lagTime)))
                        .exprange(minFreq.lag2(lagTime),maxFreq.lag2(lagTime)),0.5));

                sound = Compander.ar(sound,sound,0.4,0.8,-1.5,1,7);
                sound = BPeakEQ.ar(sound, 6000, 2, -6);
                sound = BLowPass.ar(sound, 12000, 2);
                sound = sound;
                Out.ar(out, sound*2*amp.lag2(lagTime)*endEnv);
}).send(s);

// DELAY
SynthDef("delayGranSynth", {arg out, amp, bufnum, end=0;
                var env = DemandEnvGen.kr(Dshuf([1,1,0.5,0.001,0.001],inf),
                        Dshuf([1,3,5,7],inf),2,2,1);
                var endEnv = EnvGen.ar(Env([1,1,0],[2,8],[1,-2]),end,
                        doneAction: 2);
                var snd = FreqShift.ar(PlayBuf.ar(1,bufnum,
                        startPos:Rand(0,44100*20),loop:1),Rand(-20,20));
                var onset = Dust.kr(LFDNoise3.kr(0.2).exprange(0.15,2));
                Out.ar(out, snd * amp.lag2(2) * env * endEnv);
}).send(s);


~saxSec1 = Task({
                ~sax = Synth("saxSec1", [\input, ~saxIn, \out, ~saxOut,
                        \trigFreq, (1/8),
```

```
                \overlap, 0.8, \minFreq, 1200,
                \maxFreq, 3000, \recB, ~saxRec.bufnum,
                \compmax, 0.47, \ampFac, 0.88]);
        16.wait;
        ~sax.set(\lagTime, 32, \overlap, 1.2, \minFreq, 1000);
        40.wait;
        ~sax.set(\lagTime, 32, \overlap, 3,
                \minFreq, 500, \maxFreq, 4000);
        62.wait;
        "multi".postln;
        ~sax.set(\lagTime, 20, \ampFac, 0.95, \minFreq, 80,
                \maxFreq, 1600, \compmax, 0.53, \ring, 0);
        32.wait;
        "noise".postln;
        ~sax.set(\lagTime, 16, \minFreq, 80,
                \maxFreq, 17600, \compmax, 0.64,
                \ring, 0, \fisMin, 10, \fisMax, 200, \ampFac, 1);
        ~sax.set(\rec,1);
        16.wait;
        ~saxOnGong = Synth("delayGranSynth", [\out, ~beckenOut,
                \bufnum, ~saxRec.bufnum,
                \amp, 1.75]);
        ~saxOnPiano = Synth("delayGranSynth", [\out, ~pianoOut,
                \bufnum, ~saxRec.bufnum,
                \amp, 0.8]);
        8.wait;
        "gest".postln;
        ~sax.set(\lagTime, 8, \overlap, 1, \minFreq, 80,
                \maxFreq, 17600, \compmax, 0.45, \ring, 0, \fisMin, 1,
                \fisMax, 200, \lfFreq, 0.2);
        26.wait;
        "end".postln;
        ~saxOnGong.set(\end, 1);
        ~saxOnPiano.set(\end, 1);
        ~sax.set(\end, 1);
});
// 210
~pianoSec2 = Task({
        "piano start".postln;
        ~piano = Synth("pianoSec2", [\out, ~pianoOut, \input, ~pianoIn,
                \minFreq, 260, \maxFreq, 4000,
                \lfFreq, 0.08, \compmax, 0.2, \lagTime, 1, \fisMin, 1,
                \fisMax, 10, \ringFreq, 80, \ring, 0.0, \overlap, 0.3,
                \trigFreq, 0.15, \timeDev, 1]);
        20.wait;
        ~piano.set(\lagTime, 5,\compmax, 0.15);
        10.wait;
        "transition".postln;
        ~piano.set(\lagTime, 0.5, \minFreq, 30, \maxFreq, 6000,
                \fisMax, 100, \lfFreq, 0.3, \overlap, 2, \compmax, 0.8);
        30.wait;
        ~piano.set(\lagTime, 10, \fisMax, 400);
        10.wait;
        ~piano.set(\end, 1);
        "end".postln;
});
// 305
~pianoUduP = Task({
        "startPiano".postln;
        ~p1 = Synth("pianoDuo", [\out, ~pianoOut, \in, ~pianoIn,
                \minShift,-3000,\maxShift,-650, \amp, 0.34, \soundAmp, 0.7,
                \noiseAmp, 0.02, \minFis,1,\maxFis,5,
                \minRing,10, \maxRing,100]);
        35.wait;
        ~p1.set(\lagTime, 5, \out,  ~pianoOut, \in, ~pianoIn,
                \minShift,-5000,\maxShift,5000, \amp, 0.4, \soundAmp, 0.45,
                \noiseAmp, 0.48, \minFis,5,\maxFis,300, \minRing,5,
                \maxRing,700,\maxFreq, 7000);
        25.wait;
        ~p1.set(\lagTime, 15, \out,  ~pianoOut, \in, ~pianoIn,
                \minShift,-200,\maxShift,5000, \amp, 1.5, \soundAmp, 0.3,
                \noiseAmp, 1.5, \minFis,1,\maxFis,700, \minRing,5,
```



```
                \maxRing,700);
        30.wait;
        ~p1.set(\end, 1);
});
~pianoUduU = Task({
        "start Udu".postln;
        ~udu1 = Synth("uduS1", [\out, ~uduOut, \in, ~uduIn,
                \minShift,-2000,\maxShift,-650, \amp, 0.35, \soundAmp, 1.2,
                \noiseAmp, 0.03, \minFis,2,\maxFis,15, \minRing,10,
                \maxRing,100, \maxFreq, 200]);
        35.wait;
        ~udu1.set(\lagTime, 5, \out, ~uduOut, \in, ~uduIn,
                \minShift,-5000,\maxShift,5000, \amp, 0.55, \soundAmp, 0.5,
                \noiseAmp, 0.5, \minFis,5,\maxFis,200, \minRing,5,
                \maxRing,700,\maxFreq, 7000);
        25.wait;
        ~udu1.set(\lagTime, 20, \out, ~uduOut, \in, ~uduIn,
                \minShift,-500,\maxShift,5000, \amp, 2, \soundAmp, 0.35,
                \noiseAmp, 1.5, \minFis,1,\maxFis,540, \minRing,5,
                \maxRing,700);
        30.wait;
        ~udu1.set(\end, 1);
});
// 400
~uduSolo = Task({
        ~recb.zero;
        ~udu2 = Synth("grainUdu", [\out, ~uduOut, \in, ~uduIn, \buf1,
                ~recb.bufnum,\buf2, ~uduRec.bufnum,
                \envbuf, z.bufnum, \amp, 1.2]);
        35.wait;
        ~ud2.set(\amp, 1, \minTrig, 20, \maxTrig, 100, \minRate, 0.1,
                \maxRate, 0.5);
        20.wait;
        // adjust amps
        ~uduOnPiano = Synth("delayGranSynth", [\out, ~pianoOut, \bufnum,
                ~uduRec.bufnum, \amp, 0.8]);
        ~uduOnSax = Synth("delayGranSynth", [\out, ~saxOut, \bufnum,
                ~uduRec.bufnum, \amp, 0.4]);
        15.wait;
        ~udu2.set(\lagTime, 5, \amp, 0.8, \minTrig, 1, \maxTrig, 120,
                \minRate, 0.1, \maxRate, 2.5, \minShift, -5000,
                \maxShift, 5000,\maxDur, 0.05);
        15.wait;
        ~udu2.set(\amp, 0.9, \minTrig, 1, \maxTrig, 60, \minRate, 0.1,
                \maxRate, 20, \maxDur, 0.1, \minShift, -2000,
                \maxShift, 2000);
        20.wait;
        ~uduOnPiano.set(\end,1);
        ~uduOnSax.set(\end,1);
        ~udu2.set(\end,1);
});
// 518
~saxPiano = Task({
        ~grain1.zero;
        ~grain2.zero;
        // piano
        ~s1 = Synth("saxPiano", [\out, ~pianoOut, \in, ~saxIn,
                \grainB,~grain1.bufnum, \overlap, 0.12,
                \trigFreq, 0.1, \timeDev, 1.5, \grainAmp,0.2, \amp, 1,
                \maxRing, 200, \shift, 300, \minFis, 1,
                \maxFis, 30,\inFac,1.1]);
        // sax
        ~s2 = Synth("saxPiano", [\out, ~saxOut, \in, ~pianoIn,
                \amp, 0.8, \grainB, ~grain2.bufnum, \overlap, 0.12,
                \trigFreq, 0.1, \timeDev, 1.5, \grainAmp,0.2,
                \maxRing, 200, \shift, 300, \minFis, 1, \maxFis, 30,
                \inFac,2.25]);
        30.wait;
        ~s1.set(\lagTime, 20); ~s2.set(\lagTime, 20);
        ~s1.set(\overlap, 0.2, \timeDev, 2.2);
        ~s2.set(\overlap, 0.2, \timeDev, 2.2);
        15.wait;
```

```
                ~s1.set(\maxRing, 500, \shift, 1000, \overlap, 0.7);
                ~s2.set(\maxRing, 500, \shift, 1000, \overlap, 0.7);
                20.wait;
                ~s1.set(\maxRing, 800, \shift, 5000, \grainAmp,0.02, \amp, 1.45,
                        \trigFreq, 0.2, \overlap, 1, \trigFreq, 0.3);
                ~s2.set(\maxRing, 800, \shift, 5000, \grainAmp,0.02, \amp, 0.8,
                        \trigFreq, 0.2, \overlap, 1, \trigFreq, 0.3);
                10.wait;
                ~s1.set(\overlap, 1.5);
                ~s2.set(\overlap, 1.5);
                20.wait;
                ~s1.set(\overlap, 2, \maxFis, 100, \amp, 1.1, \minFis, 20);
                ~s2.set(\overlap, 2, \maxFis, 100, \amp, 0.7, \minFis, 20);
                15.wait;
                ~s1.set(\grainAmp, 0.75);
                ~s2.set(\grainAmp, 0.75);
                25.wait;
                ~s1.set(\overlap, 0.12, \timeDev, 1.9, \trigFreq, 0.15);
                ~s2.set(\overlap, 0.12, \timeDev, 1.9, \trigFreq, 0.15);
                35.wait;
                ~s2.set(\end, 1);
                20.wait;
                // piano later, because of overlap with cymbal-sax duo
                ~s1.set(\end, 1);
});
// 718
~ending = Task({
                ~becken1 = Synth("becken1", [\in, ~beckenIn, \out, ~beckenOut,
                        \amp, 0.0, \maxFis, 1, \minFreq, 40, \maxFreq, 5000]);
                5.wait;
                "start".postln;
                ~becken1.set(\amp, 0.4);
                40.wait;
                "freq up".postln;
                ~becken1.set(\maxFreq, 15000);
                5.wait;
                ~becken1.set(\amp, 0.5);
                25.wait;
                "fis up".postln;
                ~becken1.set(\maxFis, 250);
                15.wait;
                "min up".postln;
                ~becken1.set(\minFreq, 1000, \speed, 0.25,
                        \maxSpeed, 5, \amp, 1);
                "transition to sax".postln;
                25.wait;
                ~becken1.set(\minFreq, 30, \maxFreq, 2000,
                        \amp, 0.35, \maxFis, 2);
                10.wait;
                ~becken1.set(\end, 1);
                "enter sax".postln;
                ~becken1 = Synth("becken1", [\in, ~saxIn, \out, ~beckenOut,
                        \amp, 0.6, \maxFis, 4, \minFreq, 30, \maxFreq, 2500]);
                ~sax = Synth("saxSec1", [\input, ~beckenIn,
                        \out, ~saxOut, \trigFreq, (1/4),
                        \overlap, 2, \minFreq, 80,\maxFreq, 8000,
                        \recB, ~saxRec.bufnum, \compmax, 0.15, \fisMin, 10,
                        \fisMax, 700]);
                35.wait;
                "trio".postln;
                ~becken1.set(\end, 1);
                ~sax.set(\end, 1);
                ~becken1 = Synth("becken1", [\in, ~saxIn, \out, ~beckenOut,
                        \amp, 0.7, \maxFis, 4, \minFreq, 30, \maxFreq, 2500]);
                ~piano = Synth("pianoSec2", [\input, ~beckenIn,
                        \out, ~pianoOut, \minFreq, 260, \maxFreq, 6000,
                        \lfFreq, 0.05, \compmax, 0.8, \lagTime, 1, \fisMin, 1,
                        \fisMax, 20, \ringFreq, 80, \ring, 0.0, \overlap, 0.9,
                        \trigFreq, 0.15, \timeDev, 2, \ampFac, 1.5]);
                ~sax = Synth("saxSec1", [\input, ~pianoIn, \out, ~saxOut,
                        \trigFreq, (1/8), \overlap, 0.9, \minFreq, 80,
                        \maxFreq, 6000, \recB, ~saxRec.bufnum, \compmax, 0.3,
```



```
                        \fisMin, 1, \fisMax, 100]);
                20.wait;
                "freq narrow".postln;
                ~becken1.set(\lagTime, 20, \minFreq, 1200, \maxFreq, 2000,
                        \amp, 0.6);
                ~piano.set(\lagTime, 20, \minFreq, 1200, \maxFreq, 2000,
                        \compmax, 0.7);
                ~sax.set(\lagTime, 20,\minFreq, 1200, \maxFreq, 2000, \fisMax, 1,
                        \compmax, 0.3);
                25.wait;
                ~becken1.set(\end, 1);
                ~piano.set(\end, 1);
                ~sax.set(\end, 1);
        });
});
)



// start electronics + counter window

(
~stueck = Task({
~saxSec1.start;
210.wait;
~pianoSec2.start;
95.wait;
~pianoUduP.start; ~pianoUduU.start;
95.wait;
~uduSolo.start;
118.wait;
~saxPiano.start;
200.wait;
~ending.start;
});
)

~counter = CounterWindow.new(-10,{~stueck.start},0);

~counter.start;
```