Distortion Didactic Tool

POLITECNICO
MILANO 1863

CMLS HW1

Harry Foley

Eray Ozgunay

Sebastian Mendez

Project Brief & Interpretation

Brief

- Implement a didactic tool to learn and/or teach the distortion effect

Research & Development

Method 1

Static Processing in Interpreter

```
25 ~overdrive = {
26     y = Array.newFrom(a);
27
28     a.do({arg item,i;
29         if ( (abs(item) < 0.33333) && (abs(item) > 0), {y = y.put(i,2*item); "1"});
30         if ( (abs(item) < 0.6666) && (abs(item) >= 0.33333 ), {y = y.put(i,1 - (2-3*item)**0.66666);
   "2"});
31         if ( (abs(item) >= 0.6666) &&( abs(item) <= 1 ), {y = y.put(i,1); "3"});
32         }
33 );
34 "done".postln;
35 };
```

Research & Development

Method 2

Shaper function

```
//hard clip env
(
~sig = Env.new(
    levels: [-0.9,-0.9,0.9,0.9],
        times: [0.1, 0.1, 0.1],
);
~sig = ~sig.asSignal(513);
~w = ~sig.asWavetableNoWrap;
)
//load TF into buffer
~h = Buffer.loadCollection(s, ~w);
```

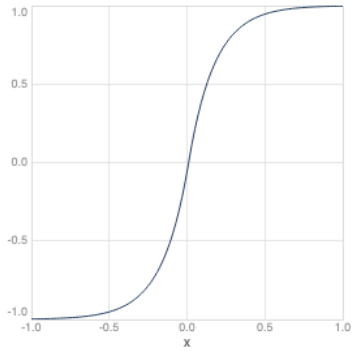
```
// Distortion
(
~playWithShaperMouseControlGain = {
    {
    var sig_distorted, index, distBuffer, mY =0, index_preEQ, sig_postEQ, limited;
        index = PlayBuf.ar(1, b, BufRateScale.kr(d), doneAction: Done.freeSelf, loop: 1.0)*MouseX.kr(0.1, 10.0, 1);
        distBuffer = ~a[0];  //select either hard or soft
        sig_distorted = Shaper.ar(distBuffer.bufnum, index);
        sig_distorted = LeakDC.ar(sig_distorted);
        limited = Limiter.ar(sig_distorted, 1.0, 0.05);

    Out.ar([0,1], limited);
    }.play;
};
)
```

Research & Development

Method 3

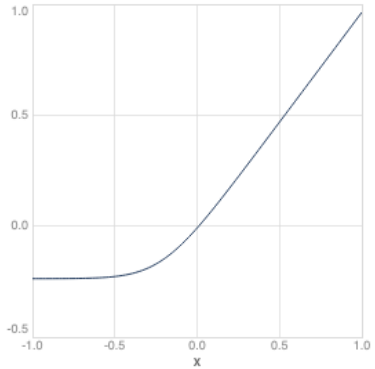
Mathematical Transfer Functions

$$F(x) = \operatorname{sgn}(x)\left(1 - e^{-q|x|}\right)$$



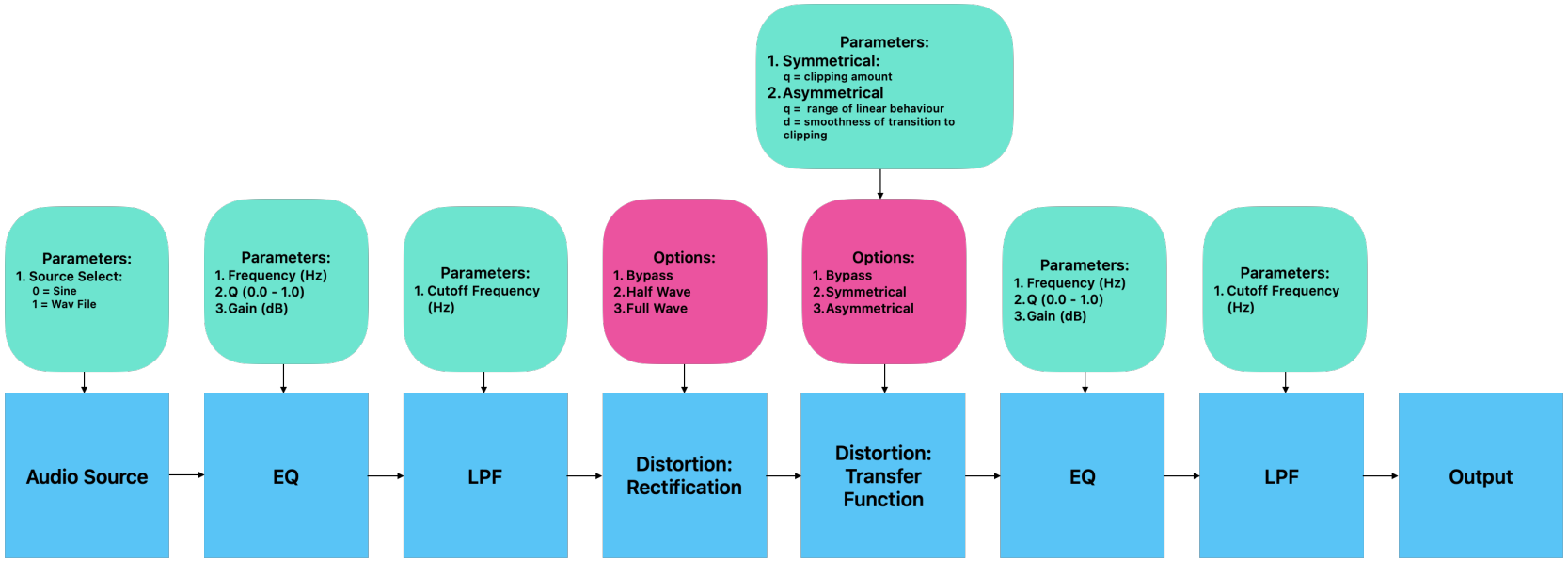
Symmetric Transfer Function

$$F(x) = \frac{x - q}{1 - e^{-d(x-q)}} + \frac{q}{1 - e^{dq}}.$$



Asymmetric Transfer Function

Signal Path


Parameters:
1. Source Select:
0 = Sine
1 = Wav File
Parameters:
1. Frequency (Hz)
2. Q (0.0 - 1.0)
3. Gain (dB)
Parameters:
1. Cutoff Frequency (Hz)
Options:
1. Bypass
2. Half Wave
3. Full Wave
Parameters:
1. Symmetrical:
q = clipping amount
2. Asymmetrical
q = range of linear behaviour
d = smoothness of transition to clipping
Options:
1. Bypass
2. Symmetrical
3. Asymmetrical
Parameters:
1. Frequency (Hz)
2. Q (0.0 - 1.0)
3. Gain (dB)
Parameters:
1. Cutoff Frequency (Hz)
Audio Source
EQ
LPF
Distortion: Rectification
Distortion: Transfer Function
EQ
LPF
Output

DSP Implementation

- SynthDef
- Custom Functions for each signal block
- Realtime (setting the arguments)
- Busses for Scopes and audio

```
~createSynthDef = {
z = SynthDef(
    "master_",
    { arg freq = 110, eqFreq = 2500, eqQue = 1/2, eqGain = 12, lpfFreq =10000, qSym = 0, qAsym = -0.002, dee = 16, audSelect = 0, postLPFFreq =
10000, inpGain = 1.1, outGain = 1;
        var sourceSig, outputSig, afterPreEQ, afterLPF, afterRect, afterTransF, afterPostEQ, afterLPF2;
        sourceSig = ~audioSource.value(~audioSel,freq); // piano sinus change, change online?

        sourceSig = sourceSig * inpGain; // input gain
        afterPreEQ = ~eqGen.value(sourceSig, eqFreq, eqQue, eqGain);
        afterLPF = ~lpfGen.value(afterPreEQ, lpfFreq);
        afterRect = ~rectGen.value(afterLPF, ~rectSelect); // how to change online?
        afterTransF = ~transF.value(afterRect, qSym, qAsym, dee, ~transSelect); // selector issues again
        afterPostEQ = ~eqGen.value(afterTransF, eqFreq, eqQue, -1 * eqGain);
        afterLPF2 = ~lpfGen.value(afterPostEQ, postLPFFreq);
        outputSig = afterLPF2 * outGain;
            {
   [1].do {
      |n|
      n.postln;
      1.wait;
   }
}.fork();
            ScopeOut2.ar(Limiter.ar(outputSig), ~scopeView_in.bufnum);
            Out.ar(b.index, Limiter.ar(outputSig));
            Out.ar([0,1], Limiter.ar(outputSig));

}).play;
```

GUI Implementation

- Modes View
- Principal Objects
- Main methods

```
//Learn Mode
~learn = Button.new(window, Rect(6*w/20, 16*h/20, 2*w/20, h/20))
.states_([
    ["Learn", Color.black,Color.gray(0.8)], //Satate 0
    ["Learn", Color.white,Color.fromHexString("#EE6C4D")]]) //Satate 1
.action_({
    arg obj;
    if(obj.value == 1,{
```


```
//Teach Mode
~teach = Button.new(window, Rect(11*w/20, 16*h/20, 2*w/20, h/20))
.states_([
    ["Teach", Color.black,Color.gray(0.8)], // State 0
    ["Teach", Color.white,Color.fromHexString("#3693D1")]]) //State 1
.action_({
    arg obj;
    if(obj.value == 1,{
```

GUI Objects – Main methods