UNIVERSIDADE ESTADUAL DE MARINGÁ
PROGRAMA INSTITUCIONAL DE BOLSAS DE INICIAÇÃO CIENTÍFICA – PIBIC/CNPq-Fundação Araucária-UEM
DEPARTAMENTO DE MÚSICA E ARTES CÊNICAS

O MÉTODO DE SÍNTESE POLIGONAL DIGITAL DE ONDAS SONORAS VIA ORDENAÇÃO CONTÍNUA DE HOHNERLEIN, REST & SMITH: INVESTIGAÇÃO, IMPLEMENTAÇÃO E EXPERIMENTAÇÃO COMPOSICIONAL

Relatório contendo os resultados finais do projeto de iniciação científica vinculado ao PIBIC/CNPq-Fundação Araucária - UEM.

Orientador:
Prof. Dr. Marcus Alessi Bittencourt

Bolsista:
Danilo Pires Lucio

Maringá
2021

RESUMO

Este projeto de pesquisa teve o objetivo de estudar e implementar em uma peça de software original o método de síntese poligonal digital de ondas sonoras via ordenação contínua de Hohnerlein, Rest & Smith (2016). Este projeto se justifica na medida em que se integra de maneira expressiva nas atividades de pesquisa, ensino, extensão e criação artística do Laboratório de Pesquisa e Produção Sonora (LAPPSO) do Departamento de Música da UEM, além de contribuir para a pesquisa na área da síntese sonora digital e para a ampliação da paleta de técnicas disponível aos compositores do laboratório. A metodologia utilizada na pesquisa incluiu o levantamento, estudo e fichamento do material bibliográfico para a sua fundamentação, incluindo a programação computacional de áudio digital e a síntese sonora digital, os ambientes de programação de áudio Pure Data e RTcmix, e o método de síntese poligonal digital de ondas sonoras via ordenação contínua. A partir destes estudos, a síntese poligonal foi implementada computacionalmente utilizando-se o ambiente Pure Data e, como experimentação do método, o software resultante foi utilizado na criação de fragmentos musicais que demonstram as suas possibilidades timbrísticas. Ao final, esta pesquisa foi formalizada com a preparação de um artigo científico e todo o material bibliográfico, computacional e de criação musical produzido foi acrescentado ao site de documentação do Laboratório de Pesquisa e Produção Sonora (LAPPSO) da UEM.

1. INTRODUÇÃO.

Este projeto de pesquisa de iniciação científica teve como objetivo investigar, experimentar e implementar na forma de um software original um método recentemente descoberto de síntese de áudio denominado Síntese Poligonal. A síntese poligonal de ondas via ordenação contínua (*continuous order polygonal waveform synthesis*) é um método de geração de ondas sonoras por meio da travessia de polígonos utilizando-se de um fasor rotativo para realizar a amostragem em espaço polar das coordenadas geométricas do perímetro de um polígono variável. Devido à velocidade constante do fasor, o resultado sonoro do processo possui uma altura definida e fixa e o ajuste em tempo real dos parâmetros geométricos do polígono produz alterações e variações timbrísticas igualmente em tempo real e bastante ricas do ponto de vista musical (HOHNERLEIN, REST & SMITH, 2016, p. 533). Este método de síntese, desenvolvido por Christoph Hohnerlein, Maximilian Rest e Julius Smith III e apresentado e descrito em 2016 nos Anais da 42ª International Computer Music Conference (ICMC) em Utrecht, Holanda (HOHNERLEIN, REST & SMITH, 2016), foi implementado na forma de um produto comercial pela empresa alemã E-RM Erfindungsbüro (https://www.e-rm.de) de Berlin. Este produto, um módulo para acréscimo a sintetizadores modulares do tipo Eurorack, foi lançado em 2019 sob a denominação de Polygogo (E-RM ERFINDUNGSBÜRO, 2019).

Uma vez que o método matemático para efetuar tal síntese sonora digital encontra-se descrito em detalhes em HOHNERLEIN, REST & SMITH (2016), a implementação dele é absolutamente viável e possível em ambientes de programação algorítmica de áudio tais como o Pure Data (PUCKETTE, 1996) e o RTcmix (GARTON & TOPPER, 1997), ambos disponíveis gratuitamente na internet e já costumeiramente instalados e utilizados nos computadores do Laboratório de Pesquisa e Produção Sonora (LAPPSO) da UEM.

Desta maneira, este projeto de pesquisa pretendeu fazer uso do ambiente gráfico Pure Data, que serve para processamento em tempo real de áudio e vídeo, para projetar e construir uma peça de software capaz de sintetizar ondas sonoras por meio do método poligonal de Hohnerlein, Rest & Smith, utilizando ao final também o software resultante para efetivamente realizar experimentos

composicionais musicais originais que demonstrassem as possibilidades timbrísticas daquele método de síntese.

2. OBJETIVOS, JUSTIFICATIVA E METODOLOGIA.

2.1. OBJETIVOS.

- Objetivo Geral:
    1. Implementar em uma peça de software original o método de síntese poligonal digital de ondas sonoras via ordenação contínua de Hohnerlein, Rest & Smith.

- Objetivos Específicos:
    1. Estudar métodos clássicos de síntese sonora;
    2. Estudar o método de síntese poligonal de ondas sonoras de Hohnerlein, Rest & Smith;
    3. Estudar a linguagem computacional de programação de áudio Pure Data;
    4. Desenvolver por meio do Pure Data um software original de síntese utilizando-se do método de síntese poligonal de ondas sonoras de Hohnerlein, Rest & Smith;
    5. Experimentar com o dispositivo de síntese criado, por meio da composição de fragmentos musicais originais que demonstrem as possibilidades timbrísticas do método de síntese poligonal;
    6. Escrever um artigo científico formalizando as pesquisas realizadas;
    7. Acrescentar todo o material bibliográfico, computacional e criativo produzido pela pesquisa do website de documentação do Laboratório de Pesquisa e Produção Sonora (LAPPSO) da UEM;

2.2. JUSTIFICATIVA.

Este projeto de Iniciação Científica se integrou de maneira expressiva nas atividades de pesquisa, ensino, extensão e criação artística do Laboratório de Pesquisa e Produção Sonora (LAPPSO) do Departamento de Música da UEM, criado em 2006 e cadastrado no diretório de grupos de pesquisa do CNPq, que desempenha papel fundamental nas pesquisas da linha de pesquisa “Processos e Práticas de Construção e Expressão Musicais” do Programa de Pós-Graduação em Música da UEM (PMU). Tendo se somado às atividades de pesquisa do LAPPSO e do PMU, este estudo contribuiu com os esforços de produção de material bibliográfico do laboratório, acrescentando os fichamentos, resumos, escritos, softwares e criações artísticas originais produzidos pela pesquisa ao website de documentação do LAPPSO. Além de contribuir em geral para a pesquisa na área da síntese sonora digital, a implementação do método de síntese poligonal de Hohnerlein, Rest & Smith no LAPPSO contribui para o aumento da paleta de técnicas de criação de sons musicais implantada no laboratório e consequentemente disponível para utilização nos projetos composicionais dos músicos discentes e docentes do grupo de pesquisas do LAPPSO.

2.3. METODOLOGIA.

Esta pesquisa se iniciou com o levantamento, estudo e fichamento do material bibliográfico que fundamenta: a) a programação computacional de áudio digital (BOULANGER & LAZZARINI, 2011) e a síntese sonora digital (RUSS, 2004; PEJROLO & METCALFE, 2017; PUCKETTE, 2007) b) os ambientes de programação de áudio Pure Data (PUCKETTE, 1996, 2007; FARNELL, 2010) e RTcmix (GARTON & TOPPER, 1997; SOMMERFELDT, 2016); e c) o método de síntese poligonal digital de ondas sonoras via ordenação contínua de Hohnerlein, Rest & Smith (HOHNERLEIN, REST & SMITH, 2016; E-RM ERFINDUNGSBÜRO, 2019). Após esta pesquisa bibliográfica de base, o método matemático utilizado na síntese poligonal de Hohnerlein, Rest & Smith foi implementado computacionalmente utilizando-se a linguagem de programação C para criar uma biblioteca externa e *patches* para o ambiente Pure Data. Seguindo a implementação, ocorreu a experimentação com o

software desenvolvido, com a utilização dele na criação de fragmentos musicais que demonstrassem as possibilidades timbrísticas do método de síntese poligonal, sob os parâmetros composicionais estabelecidos em ROADS (2015). Os elementos tecnológicos para realizar esta pesquisa, dentre hardware e software, estavam todos disponíveis no Laboratório de Pesquisa e Produção Sonora (LAPPSO) do Departamento de Música eArtes Cênicas da UEM e seu Programa de Pós-Graduação em Música (PMU). O projeto finalizou-se com a formalização da pesquisa realizada em formato de artigo científico, o que incluiu ainda a transferência dos materiais bibliográficos, computacionais e criativos gerados pela pesquisa para o website de documentação do Laboratório de Pesquisa e Produção Sonora da UEM.

3. RESULTADOS E DISCUSSÃO.

3.1. DESCRIÇÃO DO MÉTODO.

O processo de síntese poligonal de Hohnerlein, Rest & Smith (2016) parte da ideia de um polígono inscrito com seus vértices em um círculo imaginário de raio de 1 unidade em um plano cartesiano. A partir deste cenário, ocorre a leitura dos pontos do perímetro desse polígono à partir do ângulo percorrido por um ponto posicionado no perímetro do círculo que inscreve o polígono. Este ponto gira no perímetro do círculo em velocidade constante, sendo que para todo posição do ponto no perímetro do círculo corresponde uma coordenada x e y no perímetro do polígono. A velocidade fixa desse giro determina a altura do som gerado. O fato de que a forma do polígono se mantém inalterada providencia uma periodicidade na variação dos valores dos pontos x e y à medida que o ponto gira no círculo, gerando um timbre harmônico. Os valores das coordenadas x se tornarão os valores de amplitude do áudio digital de um canal, e as coordenadas y, de outro canal, assim gerando um som estéreo. O desenho do polígono – e consequentemente o som que ele gera – é alterado por meio de alguns parâmetros: o número de lados, o ângulo em radianos do ponto no perímetro do círculo, e os chamados “dentes”, que basicamente rotacionam os lados do polígono, criando figuras parecidas com estrelas.

Sendo G o giro em radianos do polígono, sendo α o ângulo em radianos do ponto no perímetro do círculo em relação ao eixo horizontal do plano, sendo T o valor de *offset* dos lados do perímetro (que basicamente rotaciona os lados do polígono criando "dentes" no polígonos, ou seja, figuras parecidas com estrelas), sendo n o número de lados do polígono (e não necessariamente um número inteiro), os pontos x e y do perímetro do polígono são achados com as seguintes expressões (adaptadas de RASKOLNIKOV, 2011 e de HOHNERLEIN, REST & SMITH, 2016):

$$x = R \cdot \cos(\alpha + G) \quad ; \quad y = R \cdot \sin(\alpha + G) \quad \text{, em que:}$$

$$R = \frac{\cos(\frac{\pi}{n})}{\cos[\frac{2\pi}{n} \cdot fmod(\frac{\alpha n}{2\pi}, 1) - \frac{\pi}{n} + T]} \quad ; \text{ sendo } \quad fmod(a, b) = resto\,de\ (\frac{a}{b})$$

3.2. TRADUÇÃO DA FÓRMULA PARA A LINGUAGEM C.

A ideia inicial era a de traduzir a equação matemática (RASKOLNIKOV, 2011) para a linguagem de programação C, pois seria a linguagem utilizada para a fabricação de uma biblioteca externa para o software Pure Data.

Após algumas pesquisas, encontramos uma implementação da fórmula na linguagem de programação R e fizemos a tradução para C.

Código em R:

```
n=5;
theta=(0:999)/1000;
r=cos(pi/n)/cos(2*pi*(n*theta)%%1/n-pi/n);
plot(r*cos(2*pi*theta),r*sin(2*pi*theta),asp=1,xlab="X",ylab="Y",
main=paste("Regular ",n,"-gon",sep=""));
```

(RASKOLNIKOV, 2011)

Código traduzido em C:

```
double d_n = 3 ; // número de lados
double T = 0.0 ; // intensidade dos dentes
double M_PI = 3.14159265358979323846 ;
double X , Y ; // coordenadas do ponto do perímetro do polígono

double theta = 0. ; // ângulo em radianos do ponto no círculo
```

```
    while (theta <= (2.*M_PI)*3. )
    {
        R = cos(M_PI/d_n)/cos( ( ((2.*M_PI)/d_n) *
fmod((theta*d_n)/(2.*M_PI),1.)) - (M_PI/d_n) + T ) ;

        X = R*cos(theta) ;
        Y = R*sin(theta) ;

        fprintf(write_file,"%lf %lf\n",X , Y ) ;

        theta += .001 ;
    }
```

Então testamos o código utilizando o software GnuPlot, que imprime desenhos, para averiguar se o polígono pretendido era desenhado corretamente e se os parâmetros (lados, dentes e rotação) se alteravam conforme o esperado com a substituição dos parâmetros. Para isso, realizamos algumas impressões no software com os mesmos parâmetros de exemplos encontrados no artigo usado como base (HOHNERLEIN, REST & SMITH, 2016) e conferimos se os nossos resultados eram os mesmos. A seguir, as Figuras de 1 a 6 mostram o resultado destes testes usando como base alguns modelos que constam no artigo usado como base para a pesquisa:

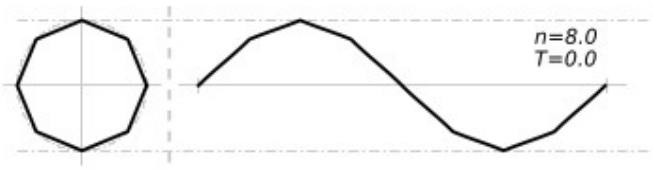

Figura 1: Exemplo de polígono de n=8.0 (correspondente aos lados), e T=0.0 (correspondente aos dentes): (HOHNERLEIN; REST; SMITH 2016: p. 534)

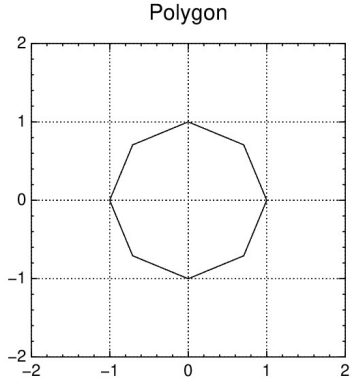

Figura 2: Nosso teste com os mesmos parâmetros (n=8.0, T=0.0):

Figura 3:Exemplo do artigo (n=5.4, T=0.1): (HOHNERLEIN; REST; SMITH 2016: p. 534)

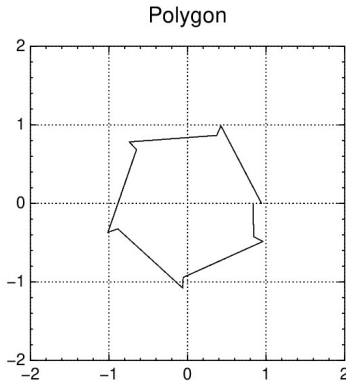

Figura 4: Nosso teste com os mesmos parâmetros (n=5.4, T=0.1).

Figura 5: Exemplo do artigo (n=2.3, T=0,1):

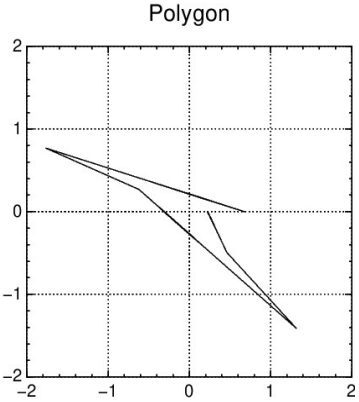

Figura 6:Nosso teste com os mesmos parâmetros (n=2.3, T=0.1).

3.3. COMPREENSÃO DO FUNCIONAMENTO DO FILTRO FIR LOW-PASS DE ORDEM 128 E TESTAGEM.

Nos diferentes processos de síntese sonora digital, um problema comumente encontrado é o de *aliasing,* que resulta na adição ao áudio de frequências espúrias indesejadas e algumas distorções sonoras quando se tenta representar digitalmente vibrações de frequências acima do limite de *nyquist* relativo à frequência de amostragem utilizada (BOULANGER & LAZZARINI, 2011, p. 261). Desse problema surge a necessidade do uso da estratégia de aliar um *oversampling* com um filtro *low-pass*, seguido de um *downsampling* (HOHNERLEIN, REST & SMITH, 2016).

A lógica do uso do filtro *low-pass* nesse caso é a seguinte: o áudio será gerado em 4 vezes mais amostras por segundo do que o valor padrão desejado (no caso, 44100 amostras por segundo), técnica chamada de *oversampling,* e então esse sinal de áudio é processado pelo filtro, que deve eliminar grande parte do espectro sonoro mais agudo desse som (os ¾ superiores do espectro), em que aparecem as frequências que causarão *aliasing* na frequência de amostragem padrão original desejada. Após esta filtragem, ocorre o processo de *downsampling*, em que as amostras de áudio são reamostradas para o seu patamar padrão original, porém agora com a devida filtragem em todo o espectro de parciais que causariam *aliasing*, com a preservação da parte do espectro onde as características sonoras importantes da síntese estão.

Encontramos em (KLOSTERMANN, 2016) o código em C para a implementação do filtro Low-Pass de ordem 128 que precisávamos. Compreendemos o seu funcionamento juntamente com o cálculo de seus coeficientes e realizamos testes para conferir se os resultados eram de acordo com o que esperávamos.

A adaptação em C do código encontrado foi a seguinte:

```
double Sinc(double x)
{
    if(x > -1.0e-5 && x < 1.0e-5) return(1.0);
    return(sin(x)/x);
}
```

```
    void RectWinFIR(double *FirCoeff, int NumTaps, double OmegaC,
double BW)
    {
        int j;
        double Arg, OmegaLow, OmegaHigh;


        for(j=0; j<NumTaps; j++)
        {
                Arg = (double)j - (double)(NumTaps-1) / 2.0;
            FirCoeff[j] = OmegaC * Sinc(OmegaC * Arg * M_PI);
        }

    }


    void FilterWithFIR(double *FirCoeff, int NumTaps, double *Signal,
double  *FilteredSignal, int NumSigPts)
    {
        int j, k, n, Top = 0;
        double y, Reg[MAX_NUMTAPS];

        for(j=0; j<NumTaps; j++)Reg[j] = 0.0;

        for(j=0; j<NumSigPts; j++)
         {
            Reg[Top] = Signal[j];
            y = 0.0;
            n = 0;

// The FirCoeff index increases while the Reg index decreases.
            for(k=Top; k>=0; k--)
            {
                y += FirCoeff[n++] * Reg[k];
                }
            for(k=NumTaps-1; k>Top; k--)
                {
                    y += FirCoeff[n++] * Reg[k];
            }
            FilteredSignal[j] = y;

            Top++;
            if(Top >= NumTaps)Top = 0;
         }

    }
```

Adaptado de (KLOSTERMANN, 2016)

Realizamos alguns testes que consistiram na geração de alguns segundos de ruído branco e filtragem pelo filtro FIR programado, para após isso averiguar por meio de uma análise espectral se o filtro funcionava como o esperado, ou

seja, com uma grande redução da parte mais aguda do espectro, restando de forma consistente somente o primeiro quarto mais grave do espectro.

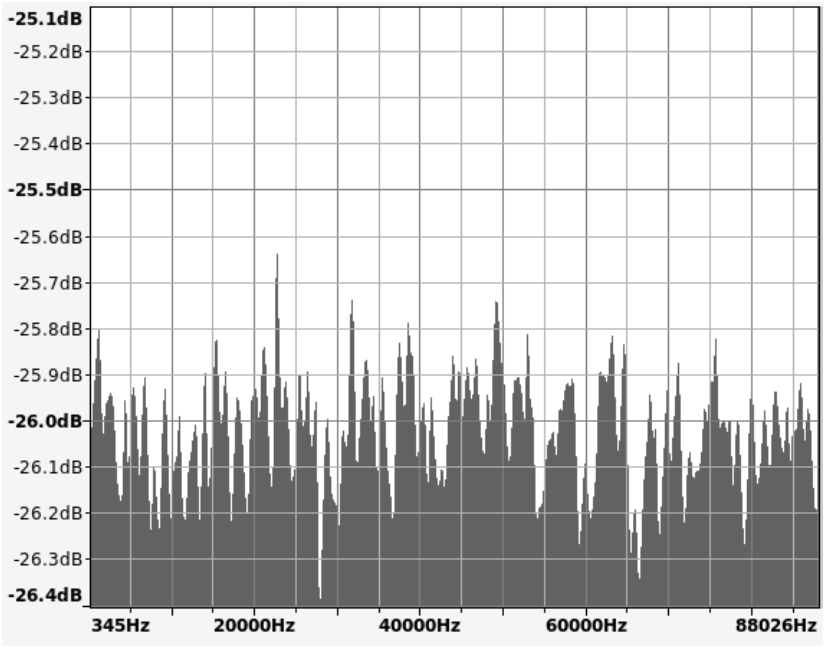

***Figura 7**: Análise espectral de fragmento de alguns segundos de ruído branco sem filtragem.*

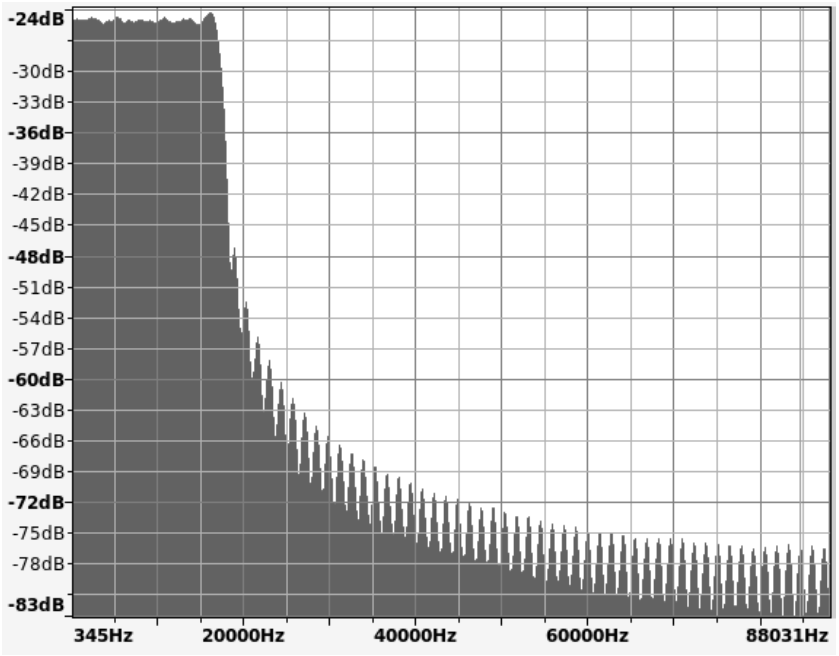

***Figura 8**: Análise espectral de alguns segundos de ruído branco agora após o filtro Low Pass. Nota-se que, se comparada à análise anterior, 3/4 do espectro é reduzido em grande parte, como esperado.*

3.4. CRIAÇÃO DA BIBLIOTECA NO SOFTWARE PURE DATA.

A criação da biblioteca aconteceu em algumas fases: primeiramente foi criada a biblioteca que continha os objetos básicos de desenho do polígono e foram realizados os testes para conferir se os desenhos eram criados corretamente. Como base, foi usado um objeto que já havia sido criado anteriormente para realizar interfaces gráficas no Pure Data e utilizado em outros projetos do professor Marcus Bittencourt.

Exemplos dos objetos em funcionamento:

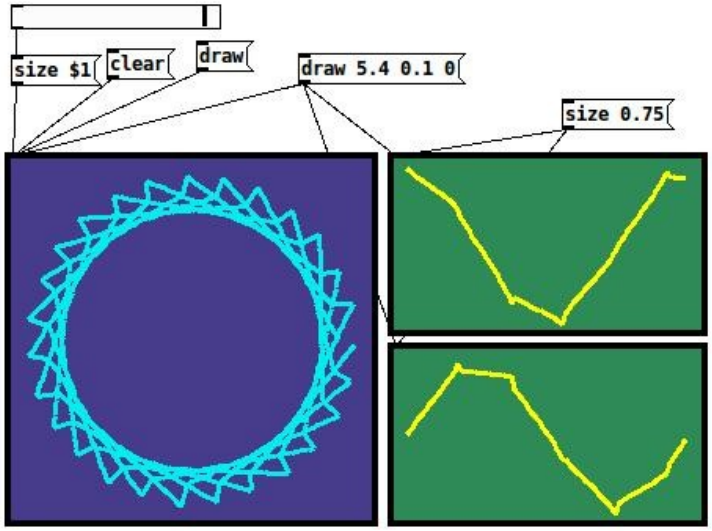

Figura 9: *Exemplo 1: n=5.4, T=0.1;*

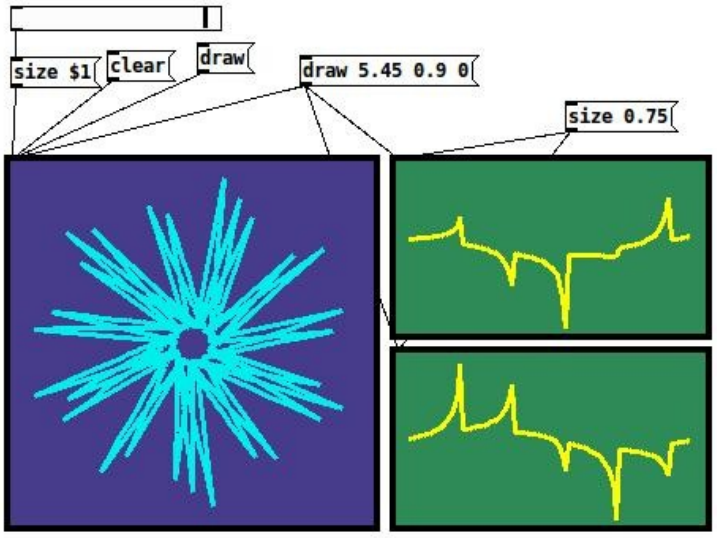

Figura 10: *n=5.45, T=0.9;*

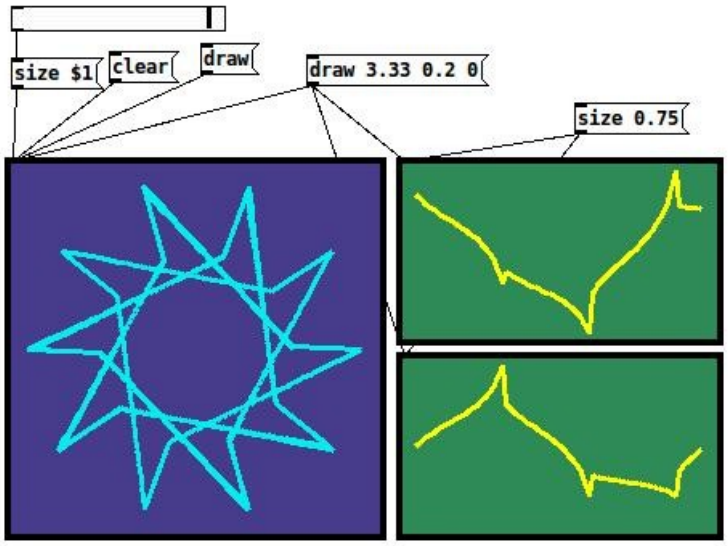

Figura 11: *n=3.33, T=0.2;*

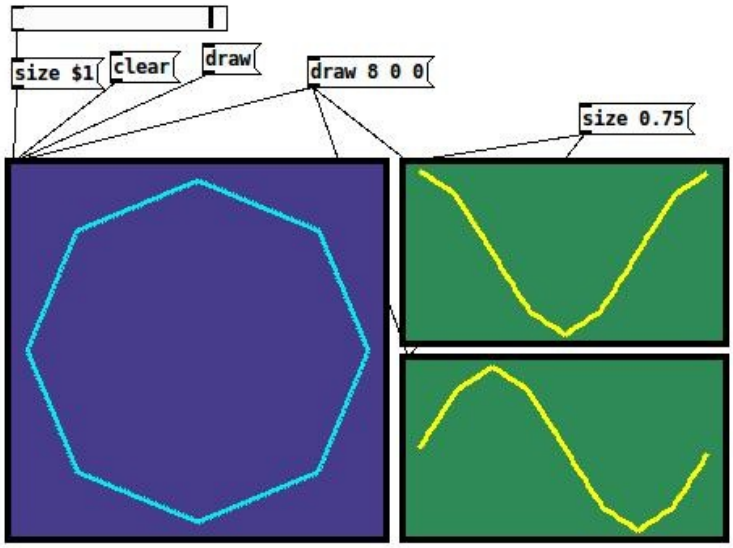

Figura 12: *n=8, T=0;*

Após o funcionamento do primeiro objeto, foi iniciada a fase de estruturação do objeto de síntese sonora. Foram criadas primeiramente todas as *inlets* e *outlets* do objeto e foram realizados testes no software a fim de conferir se os parâmetros passados estavam sendo transmitidos normalmente ao objeto.

A seguir foram implementados os códigos que realizavam a síntese sonora, de forma que os parâmetros fossem inseridos por meio de sinais de áudio, para que esses parâmetros pudessem ser alterados em tempo real durante a operação da síntese sem que ocorressem interrupções no processo de síntese, causando clipagens sonoras indesejadas.

Os pares de figuras 13-14, 15-16, 17-18 e 19-20 mostram, respectivamente, a visualização no aplicativo Pure Data do processo de síntese e a visualização do áudio gerado como resultado, confirmando que a síntese ocorreu de acordo com o esperado (note que nem sempre foi possível colocar na janela de visualização do software de edição de áudio o mesmo intervalo de fase das ondas).

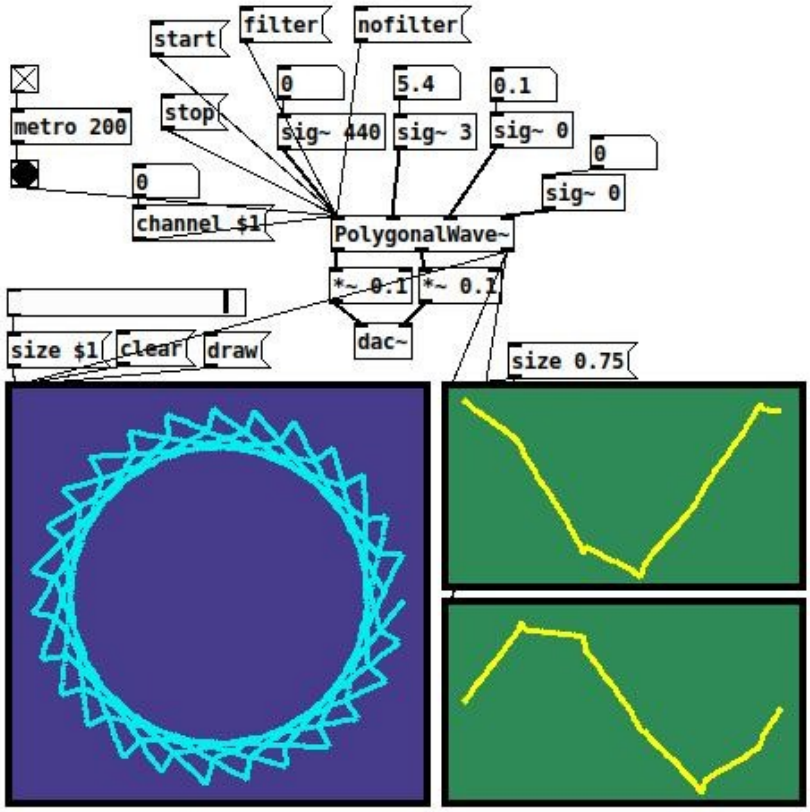

Figura 13: *n=5.4, T=0.1. Nota-se que todos os valores precisam passar pelo objeto "sig~", que os transforma em sinal.*

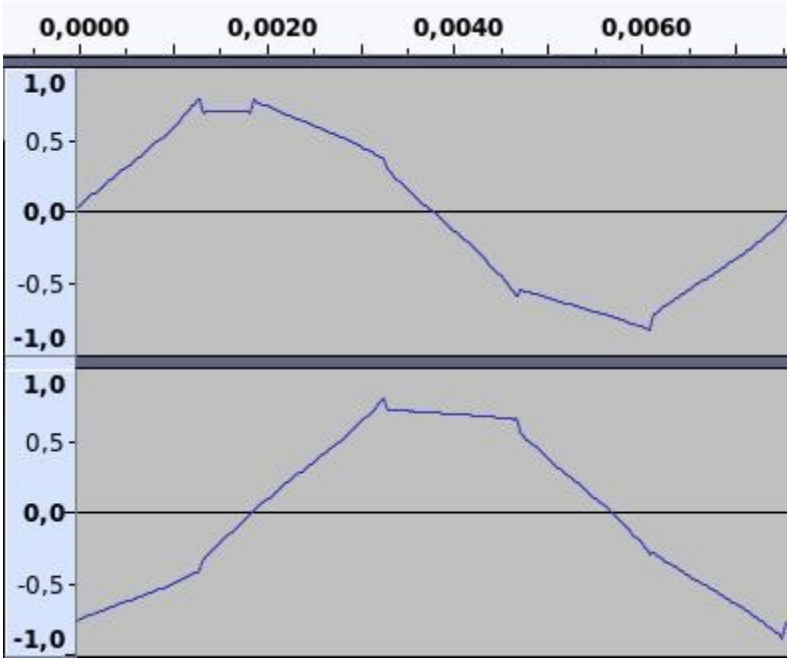

Figura 14: *Análise no software Audacity de uma gravação do som resultante com os parâmetros da Figura 11 (um período de onda completo).*

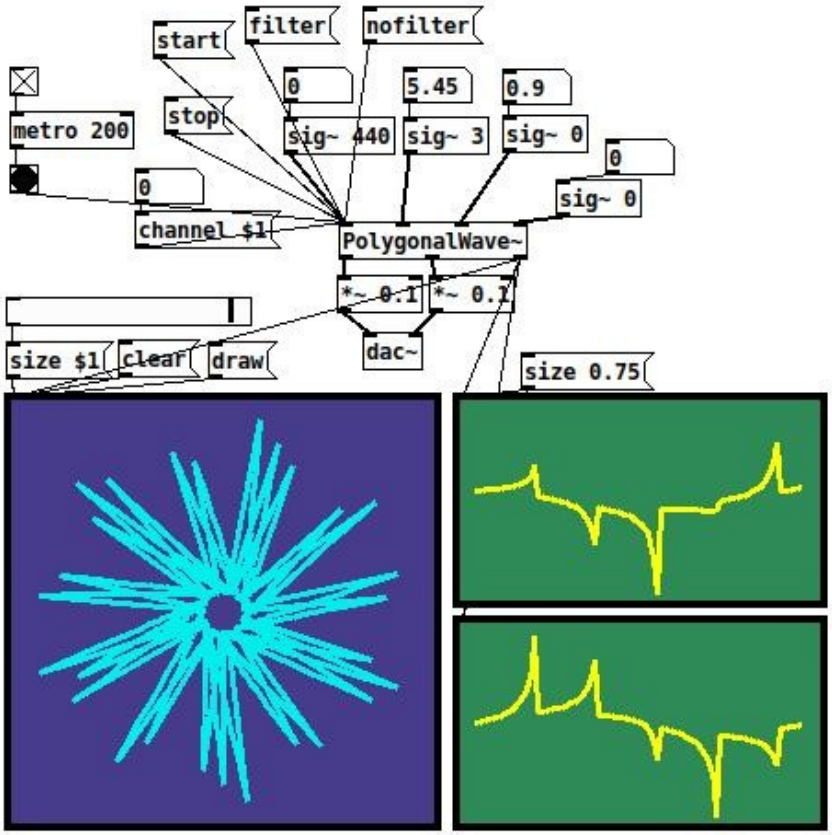

Figura 15: *n=5.45, T=0.9.*

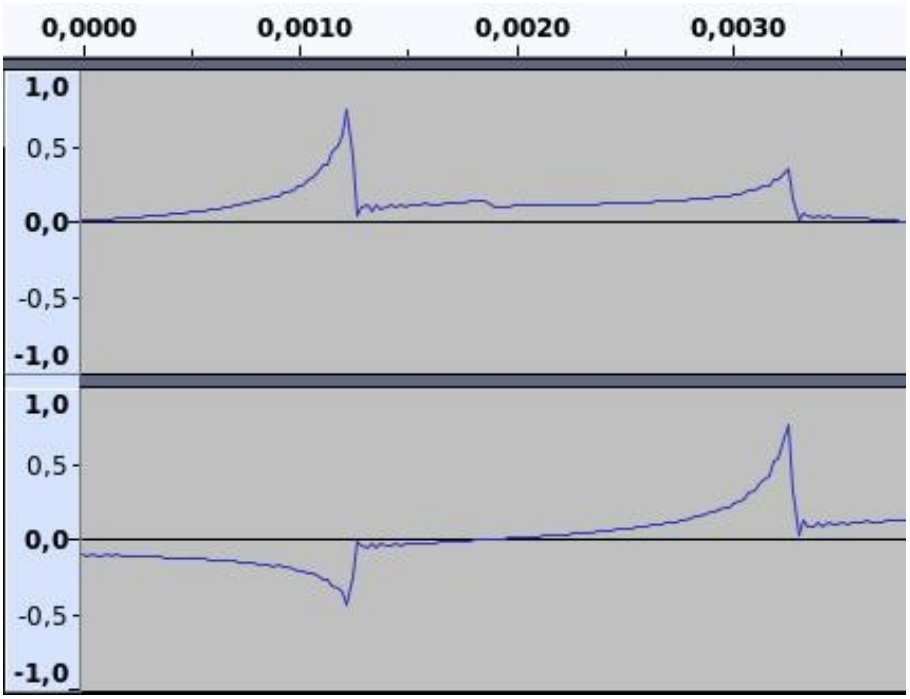

Figura 16: *análise no software Audacity de uma gravação do som resultante com os mesmos parâmetros da figura 13 (um período completo).*

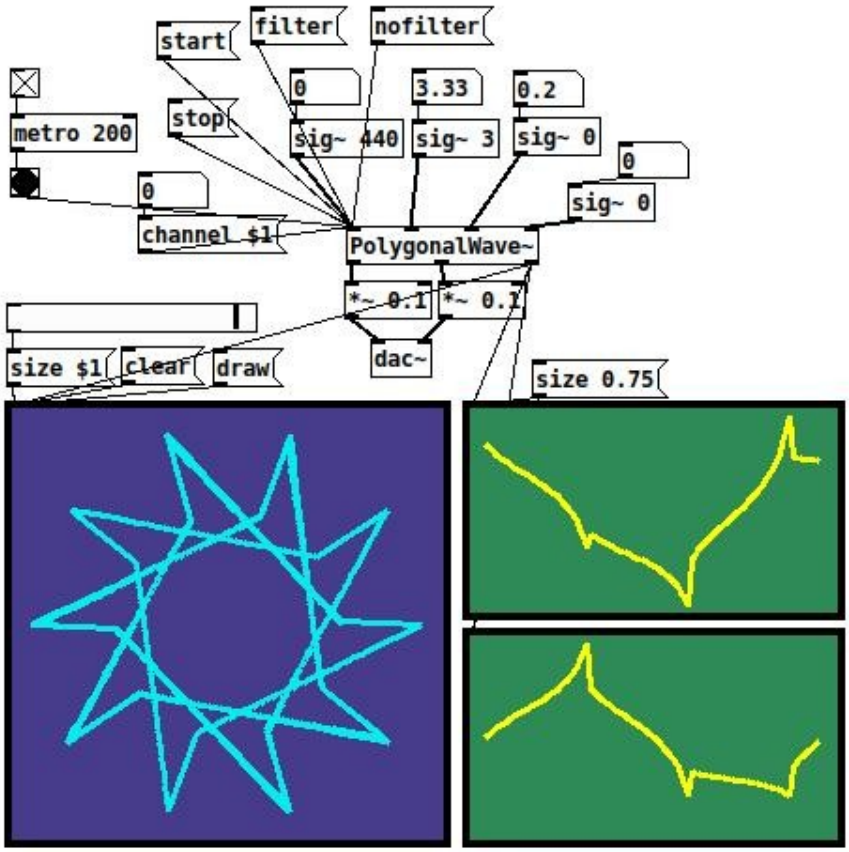

Figura 17: *n=3.33, T=0.2.*

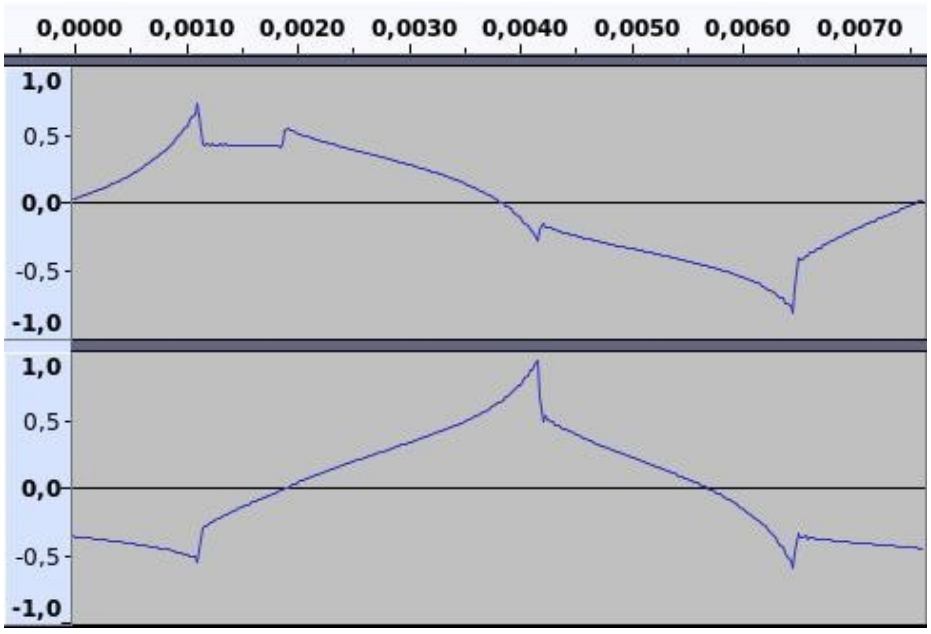

Figura 18: *análise no software Audacity de uma gravação do som resultante com os mesmos parâmetros da figura 15 (um período completo).*

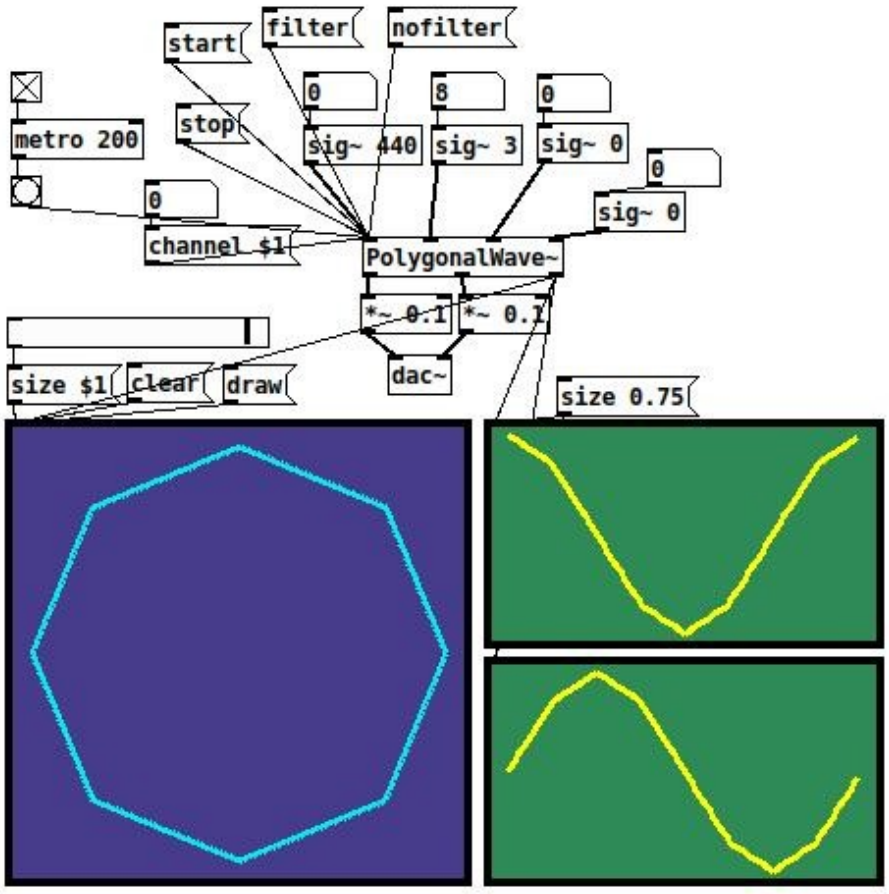

Figura 19: *n=8, T=0.*

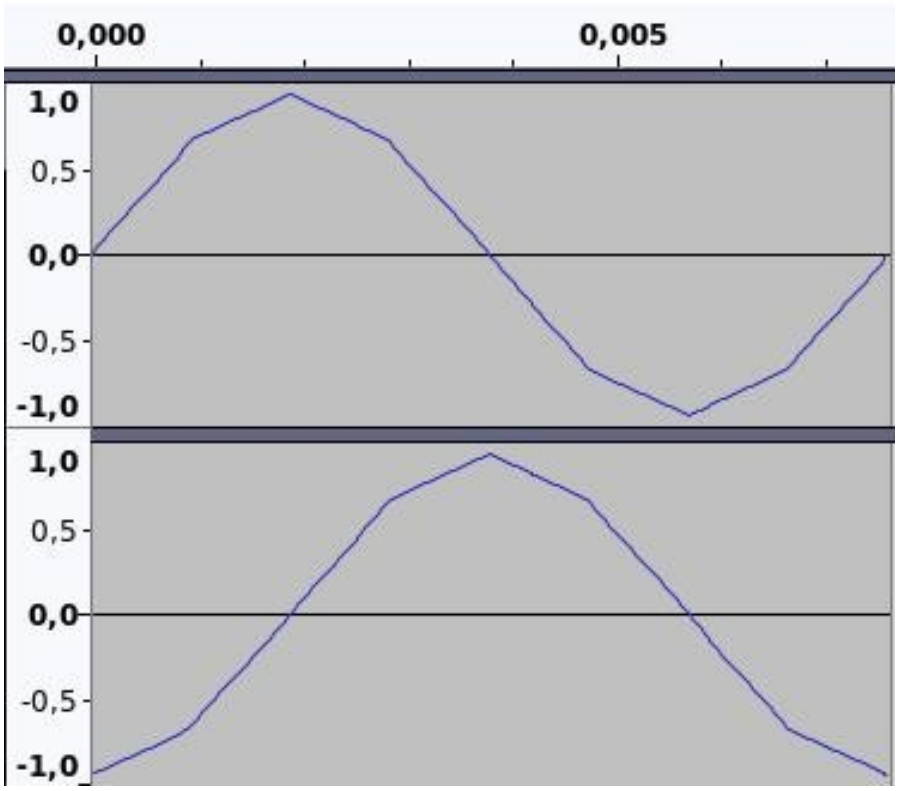

Figura 20: *análise no software Audacity de uma gravação do som resultante com os mesmos parâmetros da figura 17 (um período completo).*

Com o objeto principal de síntese sonora (PolygonalWave~) já em funcionamento, foi implementada então a última etapa para a conclusão dos componentes básicos do sintetizador poligonal: a estratégia de *oversampling*.

Foram inseridos no código fonte do objeto as linhas de código para que o áudio fosse então gerado com uma taxa de amostragem 4 vezes maior que a do resultado final esperado (44100 amostras por segundo) e inserido no código o filtro *low-pass* já antes mencionado.

3.5. IMPLEMENTAÇÃO DA POLIFONIA.

Com o objetivo de implementar uma polifonia de no mínimo 10 vozes, realizamos o teste de processamento para avaliar se o objeto de síntese sonora iria sobrecarregar o processador do computador.

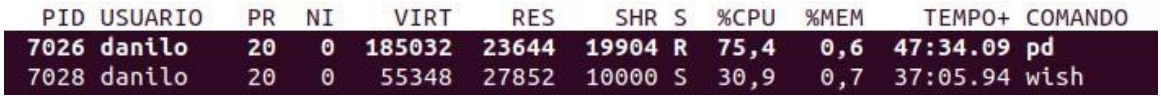

Figura 21: *processamento de 7 objetos de síntese em funcionamento resultam em 75,4% de uso de processamento. A observar também que 30,9% de processamento estão sendo utilizados pelo objeto que realiza os desenhos do polígono.*

Foi constatado que o processador de fato é sobrecarregado pelo procedimento de síntese. Também observamos que esse sobrecarregamento acontece graças ao filtro Low Pass de ordem 128. Isso acontece pois esse filtro realiza operações de multiplicação 128 vezes para cada amostra. Com a taxa de amostragem configurada em 44100 amostras por segundo, somado o fato de que a síntese é feita em estéreo (2 canais), resulta em 11.289.600 operações de cálculo por segundo por voz de síntese, ocasionando no sobrecarregamento do processador.

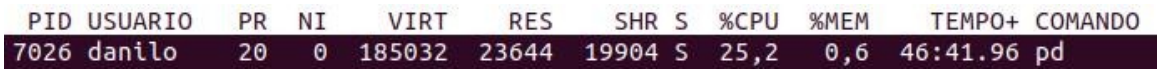

Figura 22: *Uso do processador em 25% com os mesmos 7 objetos em funcionamento, porém com os filtros desligados.*

3.6. NOVA ESTRATÉGIA DE OVERSAMPLING E FILTRAGEM.

Como a constatação foi de que o sobrecarregamento do processador é causado pela ativação de vários filtros ao mesmo tempo, e não somente pela ativação dos objetos de síntese (PolygonalWave~), a solução pensada foi de que o filtro deveria não ficar internamente ao objeto, mas sim de forma externa para que um filtro só servisse para vários objetos de síntese simultaneamente.

Para que fosse implementado dessa forma, outro problema foi encontrado. O objeto PolygonalWave~ foi pensado para funcionar em qualquer *patch* de Pure Data que trabalhe em uma taxa de amostragem de 44100, taxa padrão de trabalho. Porém, para que o filtro cumpra a função desejada de *anti-aliasing* ele precisa realizar a filtragem em uma taxa de amostragem de 4 vezes esse valor.

Para resolução desse problema foi pensado então que, para que funcionassem da forma mais eficiente esperada, ambos os objetos de síntese (PolygonalWave~) e um novo de filtragem (OverSampleFIR~) precisariam ser dois processos trabalhando separadamente e estar em um *subpatch* no Pure Data em que a taxa de amostragem fosse a do *oversampling*.

A realização disso aconteceu graças ao objeto *block~*. Esse objeto, quando colocado em um *subpatch*, faz com que a taxa de amostragem do mesmo varie em relação à taxa de amostragem do *patch*-mãe, fazendo com que o *up-sampling* ocorra a todo o sinal de áudio que entra nesse *subpatch* e um *down-samplig* ao sair do *subpatch* após todas as alterações feitas ou não neste sinal.

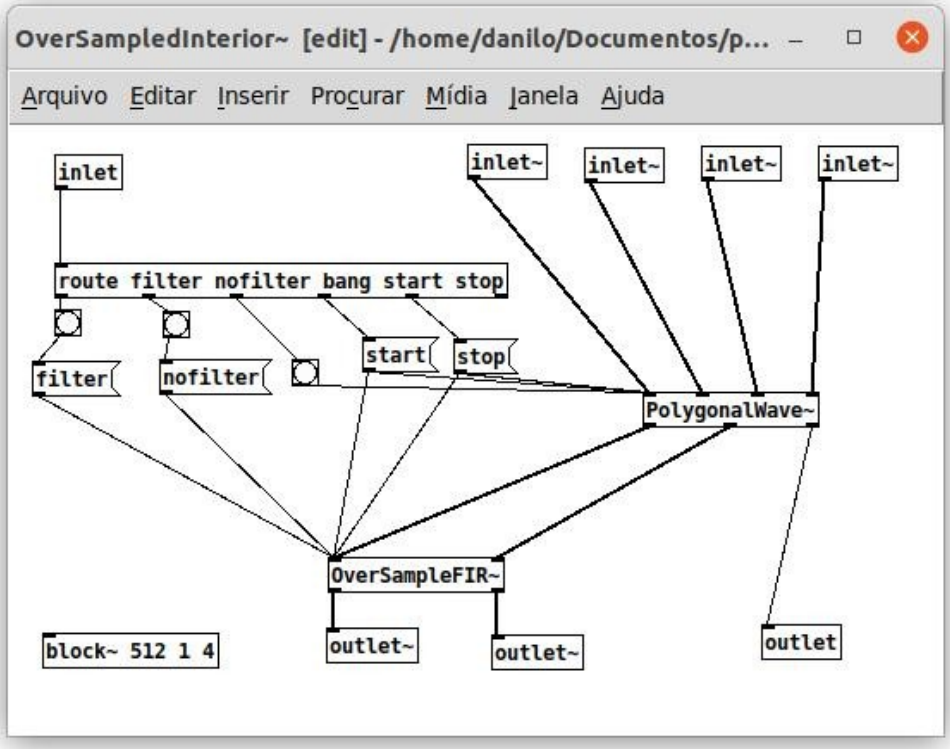

Figura 23: *Subpatch no Pure Data em que o objeto PolygonalWave~ funciona separadamente do objeto de filtragem OverSampleFIR~ em um ambiente em que a taxa de amostragem acontece em 4 vezes a taxa de amostragem do patch-mãe.*

3.7. CRIAÇÃO DE UM GERADOR DE ENVELOPES.

A fim de aumentar ainda mais a capacidade do projeto de criar timbres novos e variados, foi idealizado um gerador de envelopes um pouco mais complexo que os geradores de envelopes comuns: um gerador que servisse para gerar movimentos interessantes para todos os parâmetros que o sintetizador recebe (lados, intensidade de dentes e rotação), mas que também servisse para a amplitude, e que tivesse nele a capacidade de randomizar segundo uma margem estabelecida os valores resultantes em sua saída de dados, fazendo que, mesmo que configuradas com os mesmos parâmetros estabelecidos no gerador de envelopes, as diferentes vozes quando tocadas ao mesmo tempo ou em sequência pudessem ter pequenas (ou grandes) diferenças entre elas, o que resulta em mais organicidade e naturalidade no resultado sonoro final.

Os parâmetros utilizados no gerador de envelopes são:

- **Valor inicial (ini):** valor do ponto de partida (quando o gerador for utilizado para amplitude esse valor muito provavelmente será 0, quando utilizado para outros parâmetros do polígono provavelmente iniciará em um valor muito diferente).
- **Objetivo A (ObjA):** Meta inicial do envelope. Ao iniciar, o envelope vai do valor inicial ao valor do objetivo A em tempo X (parâmetro tIni que vem a seguir). Combinado ao seu tempo, normalmente é chamado de *attack* (ataque) nos geradores de envelopes comuns.
- **Objetivo B (ObjB):** Meta seguinte ao objetivo A. Após atingir o Objetivo A, o envelope vai ao Objetivo B no seu tempo pré determinado (tObjB). Também juntamente ao seu tempo, normalmente é o que é chamado de *decay* (decaimento).
- **Fim (end):** Valor final do envelope. Normalmente – combinado ao seu tempo – é chamado de *release.*
- **Tempo Inicial (tINI):** Tempo em que o envelope vai do Valor Inicial (ini) ao Objetivo A (ObjA).
- **Tempo do Objetivo B (tObjB):** Tempo que o envelope leva para atingir o Objetivo B (ObjB) após atingir o ObjA.
- **Tempo do Fim (tEnd):** Tempo que o envelope leva para atingir o valor final (end) quando gatilhado o acontecimento do *release*.

Após atingir o Objetivo B, o envelope chega na etapa comumente chamada de *sustain*, período em que a nota é sustentada e não estão acontecendo mais os movimentos iniciais (normalmente chamados de *attack* e *decay*); nesta posição, o sintetizador ainda não recebeu o comando final para desligar a nota, portanto ainda não foi iniciado o movimento final (*release*). Para que aconteça ainda mais dinamicidade e a fim de aumentar as possibilidades com esse envelope, nesse período em que a nota é sustentada implementamos um LFO, um oscilador de baixa frequência. Para que ele funcione, também foram necessários os parâmetros:

- **Tempo do LFO (tLFO):** Tempo que o LFO leva para ser iniciado após o Objetivo B ser atingido.
- **Frequência do LFO (fLFO):** Frequência em Hertz que o LFO terá.
- **Âmbito do LFO (bLFO):** O âmbito em porcentagem que o LFO funcionará. Exemplo: se o envelope deveria permanecer no valor 10 no período do sustain, o LFO, se configurado com a fLFO de 1(Hz) e bLFO de 0.1, irá oscilar a cada 1 segundo entre 0.9 e 1.1.

Além dos parâmetros, também foram implementados os âmbitos individuais de randomização para cada um desses parâmetros. Esses âmbitos funcionam também em porcentagem.

3.8. EXPERIMENTOS TÍMBRICOS COMPOSICIONAIS.

Com a composição de pequenos fragmentos composicionais, buscou-se demonstrar um pouco a variedade de timbres possíveis com o sintetizador poligonal e, principalmente, demonstrar a criação de timbres complexos que podem ser gerados sem que seja necessária a interação do sintetizador com outras ferramentas, como efeitos, distorções, etc.

Todos os exemplos criados – disponíveis na página do projeto no *site* do Laboratório de Pesquisa e Produção Sonora (LAPPSO) da UEM no endereço *<http://www.dmc.uem.br/lappso/projetos/computacao-musical/sintesepoligonal>* – em alguma circunstância mostram uma certa movimentação nos parâmetros do sintetizador ao longo do tempo, seja no número de lados, na rotação ou na intensidade dos dentes. Alguns, como o exemplo 3, mostram bem as possibilidades sonoras que podem ser alcançadas quando, ao utilizar o gerador de envelopes criado durante o projeto para alterar os parâmetros do sintetizador, um LFO altera a intensidade dos dentes em uma certa frequência, de forma com que essa frequência é randomizada e independente para cada nota tocada. Isso gera batimentos, novas frequências e novas interações entre as notas. É aberta assim a possibilidade da criação de uma nova camada de textura sonora. Interessante também apontar que essa nova camada pode ser muito ou pouco ligada ao acaso e à aleatoriedade: isso depende do grau de randomização e de

âmbito em que o LFO foi configurado. Isso é demonstrado de forma mais imprevisível no exemplo 3, no qual cada nota pode ter uma variação grande de velocidade e âmbito do LFO, e menos no exemplo 1, no qual cada nota, apesar de também ter um desenvolvimento diferente de tempo de início e de velocidade de LFO, neste exemplo essas diferenças acontecem com âmbitos menores de variabilidade, o que resulta em uma sensação maior de semelhança entre as notas. Isso faz com que essa ferramenta traga organicidade nos sons criados, estratégia inspirada no fato de que todas as notas tocadas em instrumentos acústicos, mesmo que intencionalmente iguais, sempre terão pequenas variações e diferenças entre si. A tentativa de trazer esse aspecto de pequenas variações entre uma nota e outra, mesmo que tocadas com os mesmos parâmetros, é aplicada em todos os exemplos – mesmo que em alguns casos isso não seja explorado ao ponto de ser facilmente percebido.

Outro resultado interessante que o sintetizador poligonal traz é alcançado quando se alia a polifonia ao gerador de envelopes de forma em que este esteja configurado para alterar amplamente algum parâmetro (sejam os dentes ou lados do polígono), de forma com que isso aconteça de forma devagar. Isso faz com que notas tocadas rapidamente tenham timbres muito diferentes das notas que soam por bastante tempo, que por vezes resulta na sensação de que são dois instrumentos tocando juntos, mesmo que sejam os mesmos parâmetros . O exemplo 5 – que também tem mostra a interação do sintetizador com um efeito de delay – demonstra claramente um resultado com timbres muito diferentes entre as notas longas e curtas. No exemplo, as notas graves se desenvolvem ao longo do tempo de forma que a intensidade dos dentes do polígono fica maior, gerando um timbre mais rico harmonicamente. Enquanto isso, os acordes que são feitos com as notas agudas, que são mais curtas e não sofrem o efeito de aumento de dentes, mantêm um timbre mais doce, resultante de um polígono com muitos lados, o que transforma o polígono mais parecido com um círculo e consequentemente gera um som similar ao de uma onda senoidal. Isso também acontece de forma parecida no exemplo 6, no qual as notas que continuam pressionadas por um tempo sofrem um aumento de número de lados, assim como de dentes. Esse tipo de movimento e alteração de timbre em notas longas também pode ser encontrado nos exemplos 2 e 7. Como já mencionado, em

todos os exemplos buscou-se chegar a resultados não estáticos, com um certo movimento de alterações timbrísticas que podem ser mais ou menos afetadas pelo fator de aleatoriedade possível pelo gerador de envelopes. Assim como tentamos demonstrar a aliança possível entre esses recursos e a polifonia possível com o sintetizador poligonal.

4. CONCLUSÕES.

A implementação criada por esta pesquisa no Pure Data do método de síntese poligonal digital de ondas sonoras via ordenação contínua de Hohnerlein, Rest & Smith mostrou-se bem sucedida em gerar sonoridades musicalmente interessantes, com timbres variados e dinâmicos, conforme pode ser ouvido nos exemplos de pequenas criações musicais que trabalhamos (disponíveis no link *<http://www.dmc.uem.br/lappso/projetos/computacao-musical/sintesepoligonal>*).

É importante salientar que, ao contrário de outras implementações pesquisadas do mesmo método de síntese (por exemplo, E-RM ERFINDUNGSBÜRO, 2019), que são monofônicas, a implementação criada por esta pesquisa é polifônica, o que permite uma gama maior de possibilidades de utilização musical em tempo real. Inclusive, esta utilização em tempo real se mostrou segura, sem grandes dificuldades de processamento. Acreditamos que a possibilidade de polifonia da nossa impĺementação, de seu processamento em tempo real e o fato de ser feito em software livre, grátis a qualquer pessoa, ao contrário de outras implementações encontradas que são proprietárias e de cunho comercial (E-RM ERFINDUNGSBÜRO, 2019), trazem diversos diferenciais positivos ao projeto.

5. REFERÊNCIAS.

BOULANGER, Richard & LAZZARINI, Victor. *The Audio Programming Book*. Cambridge, Massachusetts: The MIT Press, 2011.

E-RM ERFINDUNGSBÜRO. *Polygogo, graphical stereo oscillator with original Polygonal Synthesis*. Berlin, Alemanha, 2019. Disponível em: <https://www.e-rm.de/polygogo>. Acesso em: 10 abril 2020.

FARNELL, Andy. *Designing Sound*. Cambridge, MA: The MIT Press, 2010.

GARTON, Brad; TOPPER, Dave. RTcmix - Using CMIX in Real Time. In: ICMC 1997, International Computer Music Conference, vol. 1997, Thessaloniki, Greece. *Proceedings of the International Computer Music Conference 1997*. San Francisco: International Computer Music Association, 1997, p. 224-227.

HOHNERLEIN, Christoph; REST, Maximilian; SMITH, Julius. Continuous Order Polygonal Waveform Synthesis. In: ICMC 2016, 42nd International Computer Music Conference 12th–16th September 2016, Utrecht, The Netherlands. *Proceedings of the International Computer Music Conference 2016*. San Francisco: International Computer Music Association, 2016, p. 533-536.

PEJROLO, Andrea & METCALFE, Scott B.. *Creating sounds from scratch:* a practical guide to music synthesis for producers and composers. New York: Oxford University Press, 2017.

PUCKETTE, Miller. Pure Data. In: ICMC 1996, International Computer Music Conference, vol. 1996, Hong Kong, China. *Proceedings of the International Computer Music Conference 1996*. San Francisco: International Computer Music Association, 1996, p. 269-272.

PUCKETTE, Miller. *The Theory and Technique of Electronic Music*. Singapore: World Scientific Press, 2007.

ROADS, Curtis. *Composing Electronic Music:* A New Aesthetic. New York: Oxford University Press, 2015.

RUSS, Martin. *Sound Synthesis and Sampling*. Burlington, MA: Focal Press, 2004.

SOMMERFELDT, Jerod. *Computer Music Composition with RTcmix*. USA: edição do autor, 2016.

RASKOLNIKOV. Parametric equation for regular n-gon. [Online]. Available at: <http://math.stackexchange.com/questions/41940/is-there-an-equation-to-describe-regular-polygons>. 2011.

KLOSTERMANN, Daniel. Example C Code for FIR and IIR Filters. Disponível online em: <http://www.iowahills.com/A7ExampleCodePage.html>. Acessado em 17/03/2021. 2016.

O MÉTODO DE SÍNTESE POLIGONAL DIGITAL DE ONDAS SONORAS VIA ORDENAÇÃO CONTÍNUA DE HOHNERLEIN, REST & SMITH: INVESTIGAÇÃO, IMPLEMENTAÇÃO E EXPERIMENTAÇÃO COMPOSICIONAL

Danilo Pires Lucio (PIBIC/CNPq/FA/UEM) e-mail: danilopireslucio@gmail.com, Marcus Alessi Bittencourt (Orientador), e-mail: mabittencourt@uem.br.

Universidade Estadual de Maringá / Centro de Ciências Humanas, Linguística, Letras e Artes/Maringá, PR.

Área e subárea do conhecimento conforme tabela do CNPq/CAPES: 8.03.03.03-0 Artes; Música; Composição Musical

Palavras-chave: Síntese Sonora Digital, Computação Musical, Pure Data.

Resumo:

Este projeto de pesquisa teve o objetivo de estudar e implementar em uma peça de software original o método de síntese poligonal digital de ondas sonoras via ordenação contínua de Hohnerlein, Rest & Smith (2016). Para isso, a pesquisa partiu do estudo de elementos de programação computacional de áudio digital e de síntese sonora digital (BOULANGER & LAZZARINI, 2011), além do método específico de síntese poligonal digital (HOHNERLEIN; REST; SMITH, 2016). Para a implementação deste método de síntese, foi utilizado o ambiente de programação Pure Data (PUCKETTE, 1997) por meio da programação em linguagem C de uma biblioteca de objetos externos. A implementação de síntese criada mostrou-se bem-sucedida em gerar, via software livre, sonoridades polifônicas musicalmente interessantes que, ao contrário de outras implementações monofônicas e proprietárias pesquisadas, permite uma gama maior de possibilidades de acesso e utilização musical em tempo real.

Introdução

Este projeto de pesquisa teve o objetivo de estudar e implementar em uma peça de software original o método de síntese poligonal digital de ondas sonoras via ordenação contínua de Hohnerlein, Rest & Smith (2016). Este projeto se justificou na medida em que se integrou de maneira expressiva nas atividades de pesquisa, ensino, extensão e criação artística do Laboratório de Pesquisa e Produção Sonora (LAPPSO) do Departamento de Música e Artes da Cênicas da UEM, além de contribuir para a pesquisa na área da síntese sonora digital e para a ampliação da paleta de técnicas disponível aos compositores do laboratório.

O processo de síntese poligonal de Hohnerlein, Rest & Smith (2016) parte da ideia de um polígono inscrito com seus vértices em um círculo imaginário

de raio de 1 unidade em um plano cartesiano. A partir disso, um ponto imaginário gira no perímetro do círculo em velocidade constante, sendo que para toda posição do ponto no perímetro do círculo corresponde uma coordenada x e y de um segundo ponto no perímetro do polígono, pela projeção de uma linha reta do ponto no perímetro do círculo ao centro do círculo. A velocidade fixa desse giro determina a altura do som gerado, e o fato de que a forma do polígono se mantém inalterada providencia uma periodicidade na variação dos valores dos pontos x e y à medida que o ponto gira no círculo, gerando um timbre harmônico. Os valores das coordenadas x se tornarão os valores de amplitude do áudio digital de um canal, e as coordenadas y, de outro canal, assim gerando um som estéreo. O desenho do polígono – e consequentemente o som que ele gera – é alterado por meio de alguns parâmetros: o número de lados do polígono, o ângulo em radianos de rotação do polígono em seu eixo e os chamados "dentes", que basicamente reposicionam os lados do polígono segundo um certo *offset*, criando figuras parecidas com estrelas.

Materiais e métodos

A metodologia utilizada na pesquisa incluiu inicialmente o levantamento, estudo e fichamento do material bibliográfico para a sua fundamentação, incluindo a programação computacional de áudio digital e a síntese sonora digital (BOULANGER & LAZZARINI, 2011), o ambiente de programação de áudio Pure Data (PUCKETTE, 1997), além do método específico de síntese poligonal digital de ondas sonoras via ordenação contínua (HOHNERLEIN; REST; SMITH, 2016).

Para a implementação deste método de síntese, foi utilizado o ambiente de programação Pure Data, por meio da programação em linguagem C de uma biblioteca de objetos externos, que operacionalizam os cálculos necessários para que os polígonos sejam obtidos, assim como as suas representações gráficas e sonoras. Para implementar em objetos externos o método de síntese envisionado, foi utilizada uma equação para descrever polígonos (RASKOLNIKOV, 2011), juntamente com uma descrição matemática de como associar as coordenadas dos pontos do perímetro do polígono às amplitudes do áudio digital gerado pelo processo (HOHNERLEIN; REST; SMITH, 2016).

Com a biblioteca de objetos externos de síntese implementada, foram preparados *patches* de Pure Data criando versões polifônicas do sintetizador, juntamente com outros dispositivos típicos de síntese sonora para controlar os parâmetros mencionados anteriormente da síntese, tais como LFOs (*low frequency oscillators*) e geradores de envelopes diversos.

Após isto, foi experimentado o software de síntese criado, por meio de experimentos efetivos de criação musical, fazendo uso do sintetizador virtual implementado.

Ao final, esta pesquisa foi formalizada com a preparação de um artigo científico, e todo o material bibliográfico, computacional e de criação musical

produzido foi ainda acrescentado ao site de documentação do Laboratório de Pesquisa e Produção Sonora (LAPPSO) da UEM (*link* abaixo).

<http://www.dmc.uem.br/lappso/projetos/computacao-musical/sintesepoligonal>

Resultados e Discussão

A ideia inicial era a de traduzir a equação matemática que descrevia polígonos (RASKOLNIKOV, 2011) – que foi encontrada na linguagem R – para a linguagem de programação C, para que com ela fosse fabricada a biblioteca externa para o Pure Data. Após isso, testamos o código utilizando o software GnuPlot**,** que imprime gráficos cartesianos, para averiguar se o polígono pretendido era desenhado corretamente e se os parâmetros (lados, dentes e rotação) se alteravam conforme o esperado.

Na sequência, fizemos a implementação da estratégia de *oversampling*, sugerida no trabalho original estudado (HOHNERLEIN; REST; SMITH, 2016), que se utiliza de um filtro *low-pass* de ordem 128 para resolver o problema de *aliasing*, comumente encontrado na síntese sonora digital e que resulta em adição de frequências espúrias indesejadas e algumas distorções sonoras. A lógica dessa implementação é a de que o áudio é gerado em 4 vezes mais amostras por segundo do que o padrão de amostragem desejado (44100 Hz), e então esse sinal de áudio é processado pelo filtro, que elimina grande parte do espectro agudo desse som – a parte do espectro onde aparecem as frequências indesejadas que gerarão o *aliasing* – e então, após a filtragem, o som passa pelo processo de *downsampling*, no qual o sinal de áudio é reamostrado ao seu patamar padrão original, mas agora sem as frequências ofensoras. Após testagem por meio de espectrogramas, foi constatado que o filtro funcionava como havíamos previsto.

Com os códigos para gerar a síntese poligonal e o filtro já em funcionamento, foi fabricada em C a biblioteca que cria os objetos externos que, no Pure Data, fazem a síntese sonora e a representação visual do polígono em tempo real. Assim, implementamos no Pure Data um processo de síntese polifônica com 10 vozes (com a possibilidade de aumentar esse número), que estrategicamente foi posicionado antes da filtragem *anti-aliasing*, de forma que a filtragem aconteça apenas uma vez, sem que seja necessário um filtro individual para cada voz. Pensou-se assim pois, após testes preliminares, foi constatado que filtros independentes causavam sobrecarregamento extremo do processador, o que inviabilizava o uso do sintetizador em tempo real.

A fim de aumentar ainda mais a capacidade do projeto de criar timbres novos e variados, idealizamos um gerador de envelopes que servisse para gerar movimentos interessantes para todos os parâmetros que o sintetizador recebe (número de lados, intensidade dos dentes e rotação do polígono em seu eixo) e que também servisse para envelopes de amplitude. O gerador de envelopes tem também nele a capacidade de randomizar seus parâmetros segundo uma margem estabelecida, fazendo com que diferentes vozes, mesmo operadas por geradores de envelope configurados com os mesmos

parâmetros, pudessem apresentar pequenas (ou grandes, dependendo do caso) diferenças entre elas, o que resulta em mais organicidade e naturalidade no resultado sonoro final.

Conclusões

A implementação criada por esta pesquisa no Pure Data do método de síntese poligonal digital de ondas sonoras via ordenação contínua de Hohnerlein, Rest & Smith mostrou-se bem-sucedida em gerar sonoridades musicalmente interessantes, conforme pode ser ouvido em exemplos de pequenas criações musicais que trabalhamos (disponíveis no *link* mencionado anteriormente). É importante salientar que, ao contrário de outras implementações pesquisadas do mesmo método de síntese (por exemplo, E-RM ERFINDUNGSBÜRO, 2019), que são monofônicas e proprietárias, a implementação criada por esta pesquisa é polifônica e via software livre, o que permite uma maior abrangência de utilização e uma maior gama de possibilidades de utilização musical em tempo real.

Agradecimentos

Agradeço ao Prof. Dr. Marcus Alessi Bittencourt pelo grande apoio e incentivo que me foram dados, além do enorme conhecimento generosamente compartilhado. Também agradeço ao CNPq e à Fundação Araucária pelo incentivo à pesquisa científica.

Referências

BOULANGER, Richard & LAZZARINI, Victor. **The Audio Programming Book**. Cambridge, Massachusetts: The MIT Press, 2011.

E-RM ERFINDUNGSBÜRO. **Polygogo, graphical stereo oscillator with original Polygonal Synthesis**. Berlin, Alemanha, 2019. Disponível em: <https://www.e-rm.de/polygogo>. Acesso em: 10 abril 2020.

HOHNERLEIN, Christoph; REST, Maximilian; SMITH, Julius. Continuous Order Polygonal Waveform Synthesis. In: ICMC 2016, 42nd International Computer Music Conference 12th–16th September 2016, Utrecht, The Netherlands. **Proceedings of the International Computer Music Conference 2016**. San Francisco: International Computer Music Association, 2016, p. 533-536.

PUCKETTE, Miller. Pure Data. **Proceedings, International Computer Music Conference**, vol. 1997. San Francisco: International Computer Music Association, p. 224-227, 1997.

RASKOLNIKOV. **Parametric equation for regular n-gon**, 2011. Disponível em: <http://math.stackexchange.com/questions/41940/is-there-an-equation-to-describe-regular-polygons>. Acesso em: 10 out. 2020.