

Do Barro ao Berro: Cerâmica, Algoritmos, Gambiarras e Composição Colaborativa em “Hybris: Ocarinas do Mestre Nado”

**From Clay to Scream:
Ceramics, Algorithms, Makeshift Works and
Collaborative Composition in “Hybris: Ocarinas do
Mestre Nado”**

Henrique Maia Lins Vaz¹

Universidade Federal de Juiz de Fora (UFJF)

henrique.maia.vaz@ufjf.br

<https://orcid.org/0009-0005-4918-9689>

João Pedro Mendes de Oliveira²

Universidade Federal de Juiz de Fora (UFJF)

joaopedro.oliveira@estudante.ufjf.br

<https://orcid.org/0000-0001-8756-8771>

Submetido em 30/04/2025

Aprovado em 18/09/2025

Resumo

O artigo apresenta os itinerários de uma pesquisa artística, um percurso metodológico investigativo, técnico e poético que articula quatro eixos principais: a composição colaborativa por desenho vetorial, o desenvolvimento de um sistema de notação gráfico-proporcional xenarmônico, a concatenação como síntese coletiva e espaços compositivos pela modelagem computacional. Esses procedimentos convergem na realização do álbum *Hybris: Ocarinas do Mestre Nado*, lançado pelo selo independente Seminal Records, como síntese fonográfica do processo. A pesquisa é situada no contexto do grupo Gambioluteria (CNPq/PPGACL/UFJF), que investiga as interfaces entre algoritmos, escuta, luteria pós-digital e pedagogias críticas. A obra propõe uma reconceitualização da escritura e escrita musical como gesto expandido, onde escuta, espaço e memória se entrelaçam.

Palavras-chave: composição algorítmica; notação xenarmônica; luteria pós-digital; escuta expandida; síntese por convolução.

Abstract

The article outlines the itineraries of an artistic research project, tracing a methodological, technical, and poetic investigative path. This inquiry is articulated through four principal axes: collaborative composition via vector drawing, the development of a xenharmonic graphic-proportional notation system, concatenation as a form of collective synthesis, and the construction of compositional spaces through computational modeling. These methodological procedures converge in the realization of the album *Hybris: Ocarinas do Mestre Nado*, released by the independent label Seminal Records, which stands as the phonographic synthesis of the entire process. The research is situated within the context of the Gambioluteria group (CNPq/PPGACL/UFJF), which investigates the interfaces between algorithms, listening, post-digital lutherie, and critical pedagogy. Ultimately, the work proposes a reconceptualization of musical scoring and writing as an expanded gesture wherein listening, space, and memory are intertwined.

Keywords: algorithmic composition; xenharmonic notation; post-digital lutherie; expanded listening; convolution synthesis.

Prelúdio

O trabalho “Do Barro ao Berro: Cerâmica, Algoritmos, Gambiarras e Composição Colaborativa em Hybris: Ocarinas do Mestre Nado” nasce de uma imersão do grupo de pesquisa “Gambioluteria – da programação orientada à gambiarra ao entalhe da luteria pós-digital” (CNPq/PPGACL-UFJF) – no ateliê do artesão e músico popular Mestre Nado, em Olinda, PE.

Reconhecido como Patrimônio Vivo pelo IPHAN e pela Fundarpe, Mestre Nado é escultor de ocarinas e outras sonoridades ancestrais. Sua prática combina técnica, espiritualidade e escuta, moldando o barro com sopro e intuição até transformá-lo em corpo sonoro. Essa abordagem não apenas conforma a materialidade do álbum, mas também inspira sua epistemologia: uma técnica que ritualiza a escuta e corporifica o gesto criativo. A Figura 1 ilustra esse trabalho.

¹ Henrique Vaz, professor substituto de composição e música e tecnologia do Departamento de Música do Instituto de Artes e Design da Universidade Federal de Juiz de Fora na vigência de prestação de serviços entre abril de 2022 a junho de 2023, atualmente "professor permanente" (Edital N 08/2024) na linha "Música e Artes Sonoras" do Programa de Pós-Graduação em Artes, Culturas e Linguagens da UFJF. É autor dos livros *Ensaio{Algorítmico}* e *Crônicas {Algorítmicas}* (Estranhas Ocupações, 2023). Doutor em Processos e Práticas Composicionais pela Universidade Federal da Paraíba (bolsa CAPES - FAPESQ) e bacharelado em Ciências da Computação, coordena o grupo de pesquisa "Gambioluteria - da programação orientada à gambiarra ao entalhe da luteria pós-digital" e "O Código Criativo enquanto Chave Pedagógica", voltado aos itinerários da pedagogia da "programação criativa" sob o prisma dos algoritmos dos designs acústicos, eletrônicos, digitais e culturais de instrumentos tanto analíticos quanto desembocados em artefatos aurais tangíveis para a expressão musical. Coordenou os laboratórios de "simulação computacional e modelagem", de "síntese analógica em sistemas modulares", de "desenhos de escuta, fisiologias do som e neuromusicologia informatizada", de "culturas integradas de gravação, engenharias do áudio e práticas de estúdio" e de interatividade e iteratividade dos paradigmas audiovisuais à luz da "computação cognitiva". É diretor musical do Movimento Baía Vive, do Laboratório de Imagem e Criação em Dança (LICRID) da UFRJ, integrante do Coletivo Difluência associado às graduações do IAD/UFJF, colaborador da Camerata JF, do Coral da UFJF e da Liga Acadêmica de Neurociências da UFJF. Atua como professor colaborador do PPGACL da UFJF na linha "Música e Artes Sonoras".

² João Pedro Mendes de Oliveira é mestre em Ciência da Computação pela Universidade Federal de São João del-Rei (UFSJ), bacharel em Engenharia Mecatrônica pelo Instituto Federal do Sudeste de Minas Gerais (IF Sudeste MG) e graduando em Bacharelado em Música (Composição) pela Universidade Federal de Juiz de Fora (UFJF). Devido à sua formação interdisciplinar, possui interesse em pesquisas que envolvem música e tecn



Figura 1: Mãos do Mestre Nado sob a argila.

Fonte: adaptadas e modificadas com base em três fotografias de Eric Gomes.

Mais do que documentar, a pesquisa integrou-se ao ambiente do mestre, promovendo uma experiência de escuta coletiva, registro acústico e modelagem computacional. Dessa convivência emergiu um percurso composicional que articula tradição oral e luteria pós-digital, fundindo ocarinas artesanais, instrumentos híbridos, algoritmos, convoluções e manufatura de notações. A Figura 2 apresenta algumas das ocarinas concebidas e adquiridas quando da vivência no ateliê.



Figura 2: Ocarinas construídas no ateliê.

Fonte: acervo do autor.

A culminância da pesquisa se dá no advento do álbum “Hybris: Ocarinas do Mestre Nado” (VAZ; MEDEIROS; ALENCAR; CAMPELLO, 2024). A obra cristaliza a pesquisa, integrando os seus eixos principais: (1) aplicação algorítmica na “composição colaborativa pelo desenho” (Seção 1); (2) uso de notação gráfico-proporcional xenoharmônico (Seção 2); (3) concatenação como síntese e método de montagem (Seção 3) e (4) convolução e deconvolução em “espaços compositivos” via modelagem computacional (Seção 4).

O ato criativo articulou escuta, gesto, escrita e gambioluteria, explorando um território onde som, código e memória se cruzam. A manufatura artesanal, associada aos espaços compositivos via produto da convolução, constrói espaços acústicos arborescentes e enraizados, entrelaçando

instrumento, mediação digital e reverberação simbólica. O álbum propõe um itinerário crítico às sintaxes normativas de notação concorrentes ao *habitus conservatorial* (PEREIRA, 2012), composição e escuta, operando um sistema modular, histerético e topológico de representação.

A escrita emerge ceterior e ulterior à escuta, à síntese e ao ritual encarnado, instaurando tanto uma pedagogia da “sonicidade” (ERNST, 2019) quanto dos modos de existência de uma “fisiologia do som” (DEL NUNZIO, 2011) e uma epistemologia do campo eletromagnético: escrever o que já foi escutado e escutar o que será escrito anunciam um corpo que é adjuvante e uma histerese sistêmica como fanal. O escriturar-se verte-se em cartografia de forças, curvatura de gestos e reconstrução simbólica da memória: o álbum se afirma como prolongamento metodológico e proposição política, expandindo registros e deslocando auralidades dentro de uma luteria onde a modelagem de instrumentos vincula-se à arquitetura de escutas e modos de presença.

Inspirado na mitologia grega, o conceito de *hýbris* transcende a moralização do “erro”, sendo tratado como uma força ambivalente de “criação” e “transgressão”. Segundo (FERNANDES, 2023), (VERNANT, 2002) e (BRANDÃO, 2016), a *hýbris* é um “impulso desmedido”, poético e epistemológico que atua como um “motor de criação”. No álbum, a *hýbris* se traduz em uma poética onde a luteria e o “artesanal” evoca o “excesso criador” — como a argila que se dobra, a voz que se curva e a nota que escapa: o da terra originado, o da água amaciado, o do fogo consagrado e o que pelo sopro é ativado. Esta abordagem pressupõe uma ontologia diferencial e relacional operativa do micro ao macro, concorde diagrama ilustrado na Figura 3, na qual o “ser” sonoro não é substância indexada, mas um “entre”: um ser relacional, transitivo, não totalizável — condição de possibilidade para alteridade e para o vir-a-ser. Ao mesmo tempo, manifesta-se como “resto”: um ser que resiste, que escapa, que permanece como tensão — a memória do excluído e da possibilidade. A obra celebra uma escuta que se reinventa, um “berro do barro”, um ritual de transfiguração sonora.

Ainda que reificadas a eficiência algorítmica e a estrutura modular, o sistema proposto se distende do neorracionalismo automatizante, tensionando a racionalidade com uma relação ambivalente ao “real”. A *hýbris* permite a inclusão de curva, erro e excesso no processo criativo, transformando a escrita em um campo de disputa sensível e política, onde as materialidades do som, gesto e linguagem são reconfiguradas sob regimes outros de escuta. Nesse contexto, a histerese opera como mediação crucial: um ser temporal, marcado por diferença persistente — condição de deformação criadora e memória atuante. Ela tensiona o “entre” e o “resto”, gerando uma zona de indeterminação onde o processo composicional resiste à totalização. Desse

**Do Barro ao Berro: Cerâmica, Algoritmos, Gambiarras e
Composição Colaborativa em “Hybris: Ocarinas do Mestre Nado”**

movimento tensivo emerge uma ontologia da emergência: um-ser-que-se-torna — irreduzível à causa, mas condicionado por forças — resultando no “ato estético” como verdade singular. A emergência não é forma imposta, mas acontecimento gerado pela fricção entre precisão/eficácia/eficiência/efeito e excesso, apagamento e persistência.

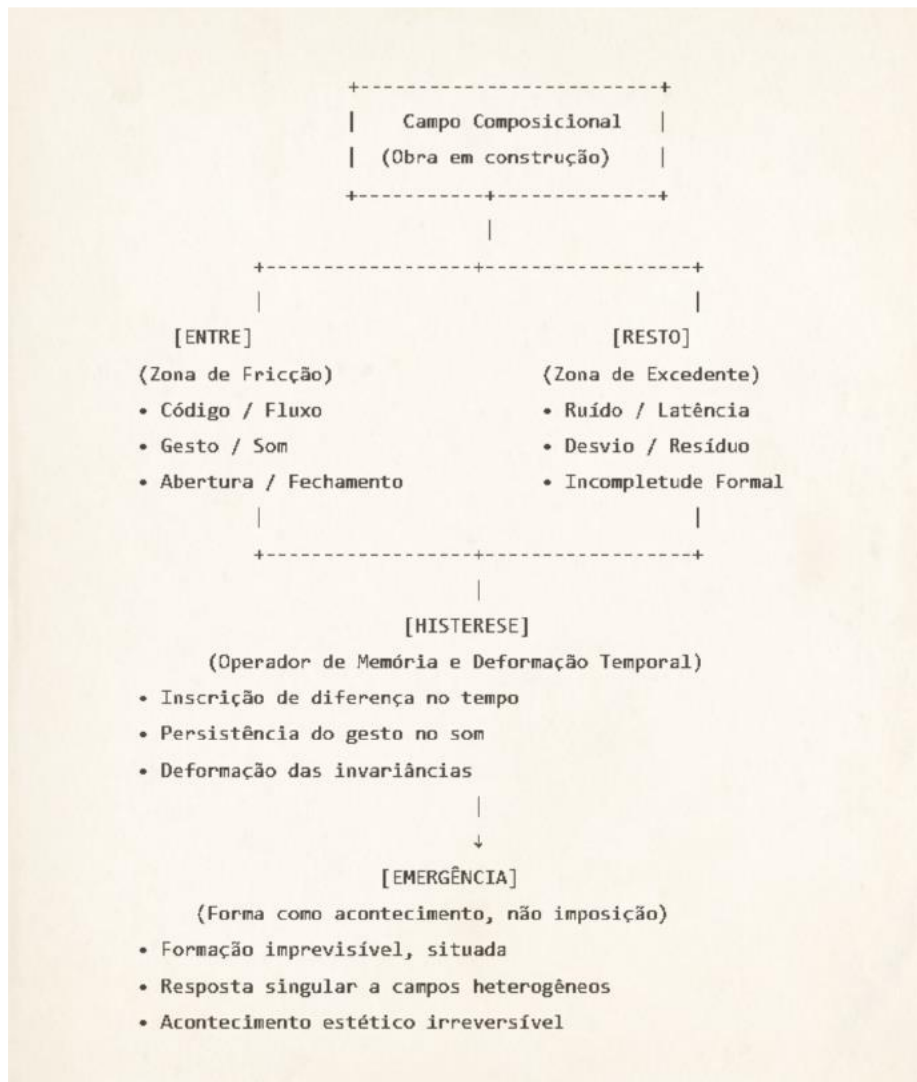


Figura 3: Ontologia diferencial, relacional e modular em Hybris: Ocarinas do Mestre Nado

Fonte: acervo do autor.

Do trânsito que atravessa o macro e o micro iteradamente e recursivamente, a obra instaura uma ontologia diferencial e relacional na qual o próprio processo — com suas fricções, restos, memórias, emergências, dados e *clocking* — encontra no registro fonográfico não uma mera representação, mas uma cristalização técnica e sensível cronopoética. O álbum *Hybris: Ocarinas do Mestre Nado* opera, assim, como um arquivo vivo desse processo, onde a sonacidade se manifesta menos como conteúdo expressivo e mais como campo de forças materiais permeado por quatro regimes ontológicos entrelaçados. A “ontologia do entre” revela-se nas zonas de mediação entre o

gesto artesanal e o algoritmo, entre o sopro e a modelagem computacional — um espaço relacional onde a sonicidade é tensionada e transformada. Desse jogo de alteridades surge a “ontologia do resto”: os resíduos sonoros não assimilados, as frequências marginais, as texturas que escapam à captura simbólica — traços de uma irreducibilidade que persiste como memória material do excluído. Mediando essas camadas, a “ontologia da histerese” imprime sua marca temporal: a deformação das invariâncias ao longo do processo, a latência dos gestos de luteria, a persistência de ressonâncias e artefatos nas ocarinas — uma escuta técnica que registra não significados, mas durações e diferenças. Por fim, desse campo emerge a “ontologia da emergência”: o “álbum” como acontecimento singular, resultado de suas interações tensionadas. O fonograma, longe de ser um documento estático, é a síntese tecnopoética dessas quatro camadas — um anti-organicismo estruturalizado que cristaliza o processo sem esgotá-lo, oferecendo à escuta não uma forma fechada, mas um mapa de tensões entre código e gesto, precisão e excesso, apagamento e persistência, onde a própria sonicidade se revela como protocolo que obista ontologias de um “dever-ser”.

A estrutura textual aqui adotada oscila entre artigo acadêmico e relatório científico-poético. Essa opção é intencional: misturar códigos, descrições técnicas, instruções de uso e considerações conceituais reflete a própria natureza do projeto, que articula prática artística, investigação computacional e reflexão crítica. A forma híbrida não deve ser entendida como dispersão, mas como parte do método — uma escrita histerética e expandida que busca tornar visível o processo.

1. Composição Colaborativa pelo Desenho

A partir dessa configuração conceitual, desenvolvem-se modos técnicos de viabilizar a escrita como experiência colaborativa e modular, iniciando-se com um protótipo de aplicação que transforma o desenho livre em vetor composicional. Tal como o barro nas mãos do oleiro, o traço, na aplicação, converte-se em som, deslocando a ideia de composição como paradigma solipsista para um modelo relacional, no qual o desenho é partilhado, interpretado e reconfigurado por módulos visuais e sintáticos. Esse gesto gráfico abandona a linearidade da escrita para reconquistar o espaço da imagem (FLUSSER, 2007), convertendo-o em escuta.

A “composição colaborativa pelo desenho” inaugura a dimensão operatória do sistema em *Hybris: Ocarinas do Mestre Nado*, indo além da ferramenta ao propor um gesto comunitário e um ato de tradução topológica, onde o traço se torna vetor de diferença. Nesse processo, a escrita

nasce da escuta do outro, e a composição emerge como negociação entre curva, código e comunidade, afirmando um modelo de criação placitada coletivamente.

1.1 Mediação gráfica e cognição musical: considerações epistemológicas

A aplicação desenvolvida para a “composição colaborativa pelo desenho”, com inspirações nos trabalhos de (CASTELÕES, 2016) e (GATO, 2016), propõe mais do que uma ferramenta técnica: ela articula uma forma de pensar e produzir música mediada por gestos gráficos. Esse deslocamento epistemológico se insere nos debates sobre cognição musical, mediação tecnológica e estética computacional, sugerindo uma transformação no modo como a composição é concebida e praticada.

Utilizando curvas *Break-Point Function* (BPF) desenhadas livremente como *input* composicional, o sistema transforma informação gráfica em informação musical. Tal abordagem se aproxima da noção de mediação técnica proposta por (LATOURE, 1994), segundo a qual os instrumentos não apenas servem à intenção do usuário, mas reconfiguram seu modo de agir e pensar. Assim, a escrita gráfica deixa de ser mera representação para tornar-se ato composicional primário.

Inspirando-se em (HUTCHINS, 1995) e (CLARK, 2001), a interação entre sujeito, interface gráfica e algoritmo é compreendida como um exemplo de cognição distribuída. O raciocínio musical emerge da relação entre olhar, gesto, manipulação visual e retroalimentação sonora. No sentido de Clark, a composição se configura como prática cognitiva estendida no ambiente, com a aplicação e os dados gráficos integrando o sistema de pensamento.

A organização sonora resultante dos atributos topológicos do desenho — como proximidade, direção, curvatura e repetição — aproxima-se da composição morfogenética descrita por (XENAKIS, 1992) e (WISHART, 1996). Nesse modelo, a estrutura musical nasce de propriedades visuais e formais do material, comutando a escrita tradicional, baseada em símbolos codificados, por um espaço contínuo de variação: curvas tornam-se acordes, vetores tornam-se polifonias.

Por permitir a participação simultânea de múltiplos usuários e utilizar uma linguagem acessível como o desenho, o sistema abre possibilidades pedagógicas e coletivas. Essa perspectiva dialoga com a pedagogia crítica de (FREIRE, 1970), que valoriza a aprendizagem pela ação, e com o design participativo (BJÖRGVINSSON; EHN; HILLGREN, 2012), no qual a tecnologia é construída de forma colaborativa, incorporando múltiplos agentes no ato de criação.

A aplicação pormenorizada não apenas traduz graficamente a música, mas reconfigura o campo epistemológico da composição, deslocando o foco para o gesto, do código para o desenho, da partitura para o campo gráfico interativo. A captura dos gestos gráficos é realizada por meio de uma mesa digitalizadora, utilizando uma interface vinculada ao *OpenMusic* — uma extensão do *Common Lisp* para música. A Figura 3 apresenta essa ferramenta.



Figura 4: Aplicação "composição colaborativa pelo desenho".
Fonte: acervo do autor.

1.2 Análise crítica da aplicação

Esta subseção descreve a aplicação desenvolvida para a composição colaborativa a partir de entradas gráficas, permitindo a criação de estruturas polifônicas e multicanais a partir da interação entre três processos fundamentais: (1) a análise espectral-harmônica de amostras — via *Sound Description Interchange Format* (SDIF) e *chord-seq*; (2) representações gráficas desenhadas livremente (BPF), que atuam como filtros e máscaras; (3) e um processamento polifônico parametricamente inteligente que organiza a saída final em múltiplas vozes com durações equalizadas e estrutura temporal coerente.

Inserida no campo da composição algorítmica colaborativa, a aplicação explora gestos visuais como vetores de organização musical. Sua arquitetura modular, composta por *subpatches* interconectados e *OM-Loops*, implementa um fluxo que abrange extração de coordenadas, normalização, quantização, filtragem, simplificação e distribuição polifônica, transformando gestos visuais em eventos musicais “harmonicamente organizados” e oferecendo uma ponte intuitiva entre desenho e som.

A estrutura da aplicação é baseada em núcleos interconectados que desempenham diferentes funções dentro de sua arquitetura. Esses núcleos são:

- Núcleo da interface: permite que o usuário opere gestos gráficos com uma mesa digitalizadora. Nele, curvas e formas livres podem ser desenhadas em um editor visual para controlar parâmetros musicais, atuando como máscaras ou filtros aplicados sobre as saídas do núcleo de análise, integrados ao núcleo gráfico;
- Núcleo de análise: realiza a extração *frame to frame* de silhuetas harmônicas, controlada por “limiar”, garantindo a continuidade dos parciais. A análise temporal assegura coesão nas sequências geradas, enquanto as curvas BPF, derivadas da análise pelo SDIF, operam como filtros para dinâmicas expressivas. Esse núcleo também utiliza material espectral extraído do das amostras de entrada como base para a composição gráfica.
- Núcleo gráfico: transforma o gesto desenhado em operador composicional, funcionando como filtro ou reorganizador da sequência harmônica extraída do material sonoro. Ele integra de forma dinâmica a linguagem visual e musical. Seu módulo de saída quantiza os dados para uma sintaxe musical historicamente informada, distribuindo-os em múltiplas vozes paralelas e garantindo ritmo coerente, duração mínima adequada e *voicing* consistente. Para verificação e validações em tempo real, o sistema se conecta ao SuperCollider via protocolo OSC, utilizando sínteses por modelagem física.

A Figura 4 apresenta um modelo representativo da arquitetura da aplicação, indicando a interação entre núcleos descritos.

Do Barro ao Berro: Cerâmica, Algoritmos, Gambiarras e Composição Colaborativa em “Hybris: Ocarinas do Mestre Nado”

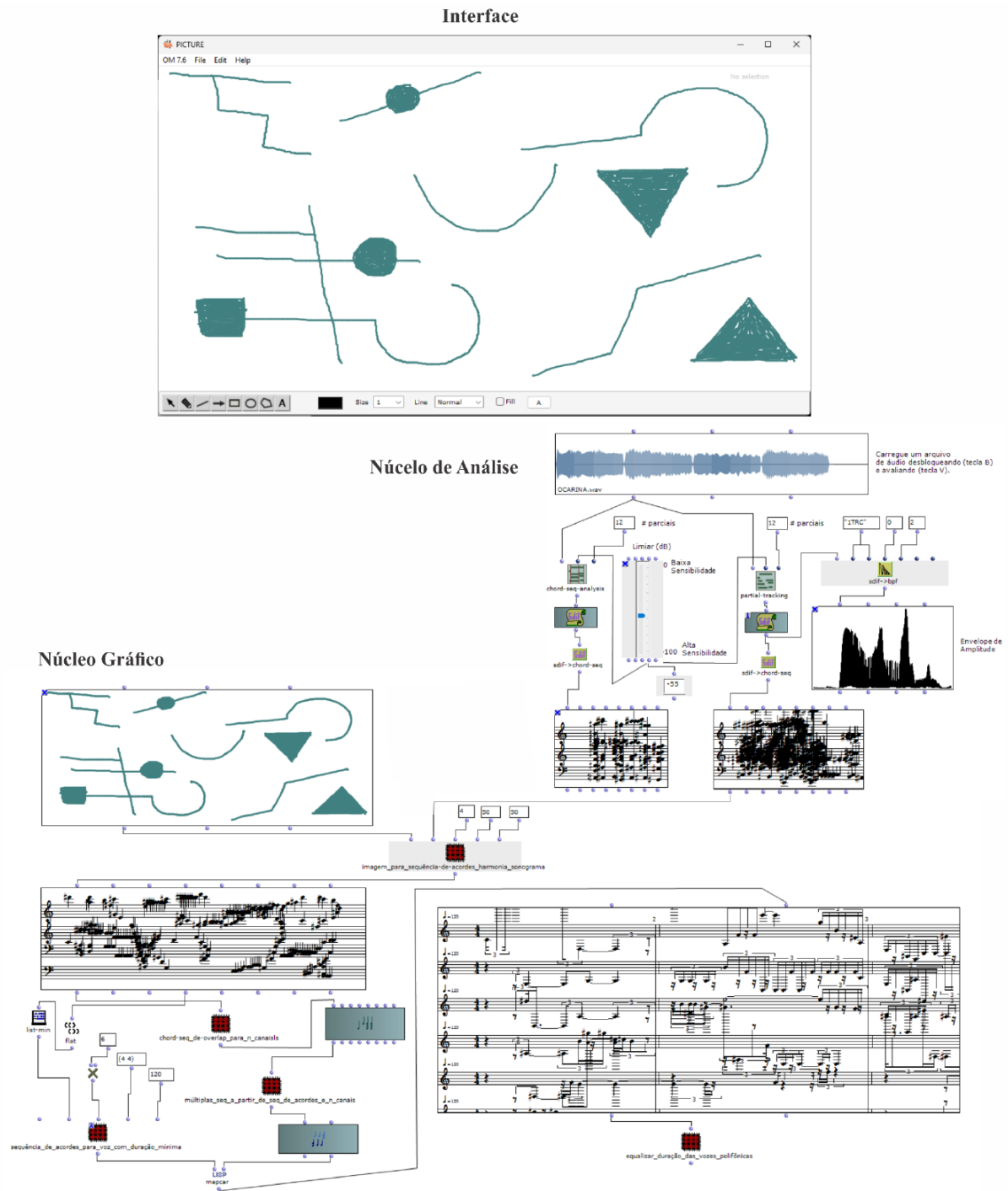


Figura 5: Arquitetura da aplicação "Composição Colaborativa pelo Desenho".

Fonte: Acervo do autor.

O sistema propõe uma forma de composição visual e compartilhada, destacando-se pela acessibilidade e caráter educativo ao permitir que usuários, mesmo sem formação musical, criem estruturas sonoras por meio do desenho. Sua abordagem gráfica e modular favorece práticas colaborativas em camadas distintas ou em tempo real, apontando para desdobramentos como

interfaces online, ferramentas de educação inclusiva e ambientes interativos para performances e instalações sonoras.

1.3 Pré-processamento harmônico via análise espectral (núcleo de análise)

Antes da inserção do gesto gráfico, a aplicação estabelece uma trama, uma treliça ou um “campo harmônico” referencial a partir da análise espectral de uma amostra de entrada. Utilizando o objeto *chord-seq-analysis* e a biblioteca SDIF, o sistema extrai parciais significativas de um arquivo sonoro, convertendo-as em objetos *chord-seq* que servem como um “gradiente harmônico” para a atuação dos gestos gráficos. Três rotas de análise estão disponíveis: análise direta com *threshold* ajustável, *partial tracking* que organiza parciais ao longo do tempo e extração de envelope de amplitude (SDIF→BPF).

A partir dessas rotas, as estruturas harmônicas distendidas temporalmente podem ser utilizadas de três formas principais: como correlação visual com dados gráficos, como matriz harmônica (filtro, máscara ou campo), ou integradas à polifonia gerada no processo gráfico. Essa abordagem cria uma ponte entre análise sonora e síntese musical, permitindo tanto a visualização de relações harmônicas quanto a aplicação em estruturas polifônicas dinâmicas. O “timbre” ou conteúdo sonoro original guia os gestos gráficos, aproximando a escuta analítica da composição assistida por computador.

A seção de análise espectral da aplicação começa com o carregamento de um arquivo de áudio, exibindo o espectrograma na interface central para referência visual. A análise segue dois caminhos: o primeiro utiliza o módulo *chord-seq-analysis* para extrair parciais com limiar ajustável, processando-as *frame to frame* para gerar uma sequência harmônica quantizada e visível no domínio simbólico. O segundo caminho, baseado em *tracking* temporal (SDIF “1TRC”), rastreia as trajetórias das parciais ao longo do tempo, oferecendo controle refinado sobre a densidade e a dinâmica espectral. Esse método gera uma “partitura” densa, expressiva e temporalmente coesa, sendo particularmente útil em contextos que exigem maior “fidelidade estrutural” à morfologia do arquivo de entrada.

A análise também inclui um módulo de envelopamento, onde a curva BPF derivada da análise SDIF representa o envelope de amplitude do áudio original. Esta curva pode ser utilizada como filtro dinâmico ou sincronizador temporal, permitindo selecionar regiões de maior intensidade sonora como base composicional. As saídas geradas podem ser integradas ao módulo principal, onde o

gesto gráfico do usuário atua como filtro ou reorganizador do material harmônico discretizado, criando uma composição resultante da interação entre o material espectral e a máscara gráfica.

Importa esclarecer que a referência ao mapeamento das alturas para valores MIDI (0–127) não indica uma limitação à resolução de 7 bits, mas sim uma estratégia de indexação e compatibilidade para intercâmbio com ambientes externos (OSC, SuperCollider). Internamente, as curvas BPF preservam a resolução contínua em *Hz*, assegurando a possibilidade de representar intervalos microtonais. Além disso, as curvas de amplitude das parciais extraídas na análise espectral são utilizadas como máscaras dinâmicas, modulando ganho, ponderando a relevância de determinados parciais e selecionando regiões de maior densidade espectral. Essa etapa confere maior organicidade ao material, pois as variações de intensidade captadas no áudio original influenciam diretamente a configuração polifônica final, aproximando o sistema da expressividade tímbrica do gesto acústico.

1.4 Integração gráfico-harmônica (núcleo gráfico)

A interligação entre os núcleos de análise e de desenho gráfico possibilita uma composição orientada simultaneamente pelo gesto e pelo espectro. O núcleo gráfico, organizado em uma arquitetura modular e hierárquica, converte gestos visuais em estruturas musicais harmonicamente organizadas. O sistema é composto por *subpatches* especializados conectados em

cascata, onde cada módulo realiza transformações específicas sobre os dados extraídos das curvas BPF, conforme ilustrado na Figura 5.

Esta abordagem assegura inteligibilidade sonora, correspondência semântica entre forma visual e organização musical, além de promover flexibilidade composicional e colaboração criativa. O fluxo do sistema parte do desenho livre do usuário, realizado sobre a interface vetorial, e passa por múltiplas etapas de processamento algorítmico que traduzem o gesto gráfico em material

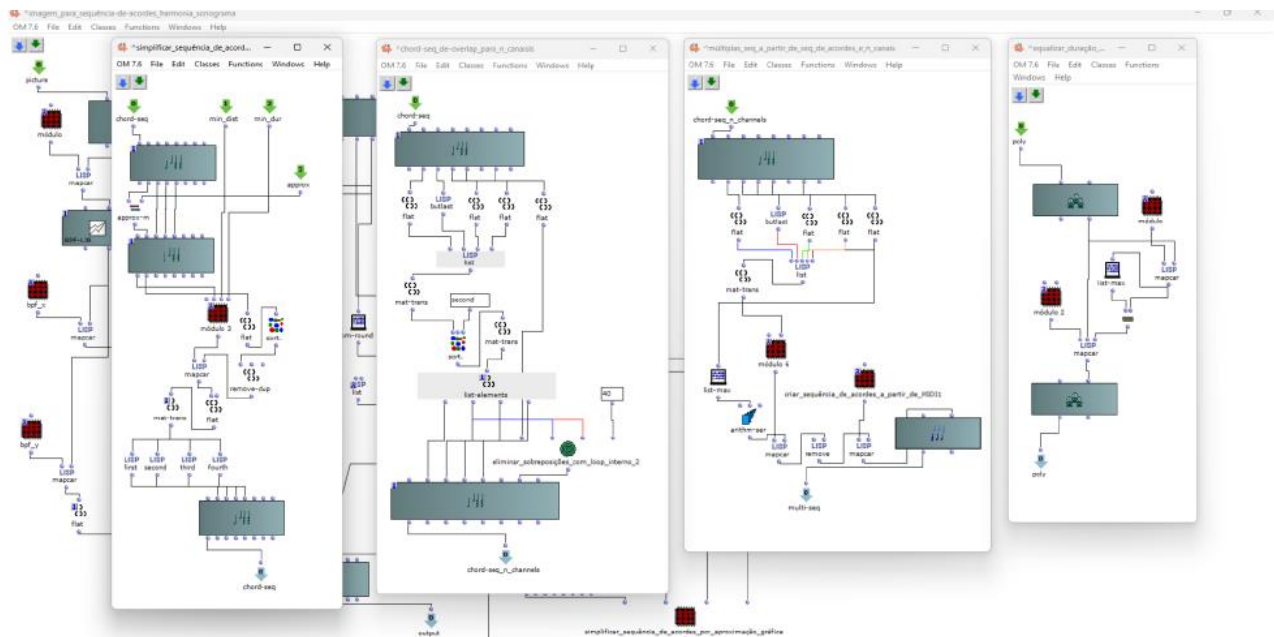


Figura 6: Representação de subpatches que integram parte do núcleo gráfico.

Fonte: acervo do autor.

sonoro estruturado e “inteligível”.

O processamento começa pela representação do desenho em curvas BPF, onde o eixo X corresponde ao tempo e o eixo Y à estruturação frequencial. Os *subpatches* *bpf_x* e *bpf_y* extraem as listas de coordenadas numéricas, que contêm as informações implícitas do gesto composicional. Para tornar esses dados utilizáveis musicalmente, aplica-se um processo de normalização e escalonamento, mapeando alturas para valores indexados em MIDI (0–127) e tempos para durações musicais padronizadas.

Após a normalização, os dados são quantizados para discretizar alturas e durações segundo convenções musicais, como semínimas ou colcheias, e escalas com razões intervalares demandadas. Essa transformação assegura que o gesto visual se traduza em unidades musicais reconhecíveis, preservando a coerência harmônica e abrindo espaço para experimentações composicionais com vetores mais complexos.

O sistema aplica então mecanismos de filtragem e simplificação: o *subpatch* “simplificar_sequência_de_acordes_por_aproximação_gráfica”, reduz eventos redundantes, mantendo apenas estruturas essenciais. Posteriormente, a organização polifônica é realizada usando algoritmos iterativos (*OM Loops*), nos quais eventos simultâneos são distribuídos entre canais independentes pelo *subpatch* “chord-seq_de-overlap_para_n_canais”, preservando identidade “melódica” e coesão temporal.

Dando sequência ao processo, a polifonia gerada é processada em duas etapas. Primeiramente, pelo *subpatch* “múltiplas_seq_a_partir_de_seq_de_acordes_e_n_canais” e, em seguida, pelo “equalizar_duração_das_vozes_polifônicas”, que asseguram durações consistentes e atribuição de vozes distintas. Com isso, o sistema gera uma “saída musical” bem estruturada, pronta para ser reproduzida, analisada ou utilizada como base para novas criações.

O sintetizador “\ocarina” foi desenvolvido em SuperCollider para simular o comportamento acústico da ocarina, comunicando-se via OSC com o OpenMusic. Baseado em princípios de modelagem física, o projeto utiliza um modelo de excitação por jato de ar turbulento combinado a ruído branco filtrado, técnica fundamentada nos trabalhos de (COOK, 1997) e expandida por (SCAVONE, 2003). A excitação do sistema é formada pela soma do ruído filtrado (simulando o fluxo de ar) e de uma onda senoidal de baixa amplitude (representando o jato de ar no bocal), ambos suavizados por uma função *tanh* para atenuar não-linearidades.

A estrutura do sintetizador organiza o som da ocarina em três componentes principais: a excitação (*airflow + jet*), a ressonância principal do corpo do instrumento (representada por um filtro passa-banda centrado na frequência fundamental) e um reforço de harmônicos altos que simula os formantes característicos do instrumento. A cavidade de Helmholtz da ocarina é modelada por um filtro de banda estreita com controle de *rq*, conferindo estabilidade espectral e aderência às propriedades físicas do instrumento físico.

Por fim, o uso de uma envoltória articulada e a aplicação de um vibrato de baixa profundidade garantem maior expressividade na execução. O diagrama apresentado na Figura 6 ilustra a integração entre este módulo de modelagem física e os núcleos de análise e síntese anteriormente descritos, consolidando a ponte entre os gestos gráficos, o processamento algorítmico e a síntese sonora para fins de verificação e validação.

Do Barro ao Berro: Cerâmica, Algoritmos, Gambiarras e Composição Colaborativa em “Hybris: Ocarinas do Mestre Nado”

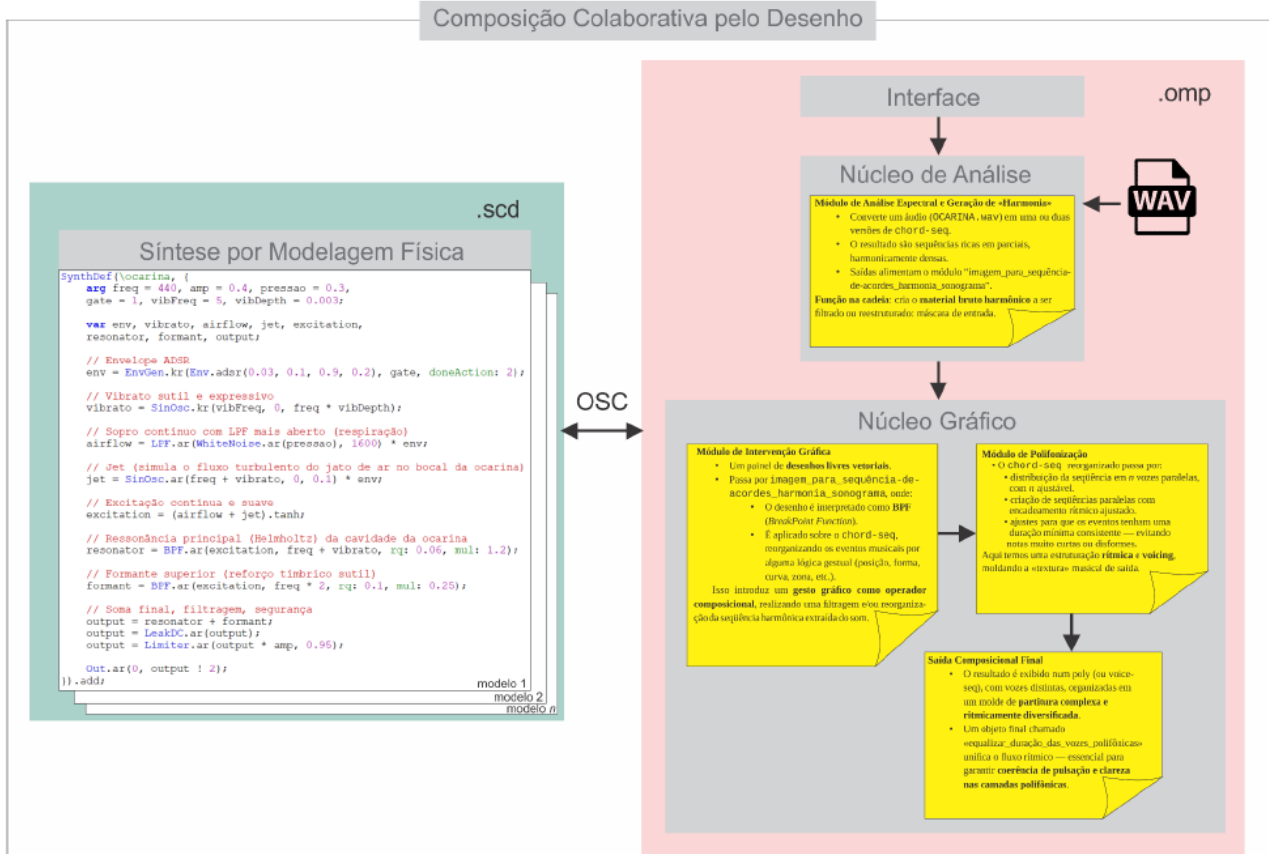


Figura 7: Diagrama analítico da integração modular da aplicação.

Fonte: acervo do autor.

O sistema proposto configura-se como um ambiente de mediação criativa no qual a tecnologia não atua meramente como ferramenta, mas como agente epistêmico ativo. Inspirado por (SIMONDON, 1958) e (FLUSSER, 2002), ele mobiliza o áudio real, o gesto gráfico e o código como modos complementares de cognição, favorecendo processos composicionais baseados em relação, fluxo e transformação.

A cognição musical é abordada de maneira multimodal, articulando escuta analítica (aural), desenho e leitura gráfica (visual), lógica e abstração (simbólica), além da percepção espacial e temporal das vozes polifônicas. Tal abordagem converge com teorias de cognição enativa e distribuída, segundo as quais o conhecimento emerge da ação situada e da interação com representações externas (VARELA, 1991; NOË, 2004).

O sistema opera segundo uma lógica de interferência e coautoria: a gestualidade do desenho e a estrutura do código não se subordinam uns aos outros, mas compõem uma rede de forças onde gesto, som, máquina, análise e algoritmo se entrelaçam. O resultado disso é um sistema composicional “coerente”, modular, sensorialmente multirreferenciado e aberto à intervenção.

2. Topologias para além do tom: uma proposta gráfico-proporcional de notação xenoarmônica

A transição da aplicação “composição colaborativa pelo desenho” para o desenvolvimento do sistema gráfico-proporcional de notação representa uma mudança técnica e conceitual importante na pesquisa. A primeira etapa foca na transdução algorítmica de desenhos livres em eventos sonoros organizados, enquanto a segunda se dedica à criação de um sistema notacional que possa representar com “precisão” as relações intervállicas emergentes dessas composições. Essa transição não é apenas funcional, mas também epistemológica, envolvendo uma transformação na forma de pensar a relação entre som, gesto e notação incorporada.

De acordo com (SIMONDON, 1958), os processos técnicos devem ser vistos como mediadores ontológicos, não como simples ferramentas. O sistema de notação proposto surge da necessidade de dar forma simbólica à experiência gerada pela aplicação ceterior, funcionando como uma escrita posterior à escuta, baseada não em “sons ideais”, mas em sons organizados por dados gráficos e máscaras espectrais.

O gesto desenhado é reinterpretado em termos de proporções, curvas e relações espaciais, permitindo o surgimento de uma notação que se baseia em critérios topológicos e harmônicos abertos. Essa notação emerge do que pode ser chamado de “*design* emergente da escuta”, registrando e formalizando “razões” que surgem da prática colaborativa iniciada pelo desenho.

A intersecção entre composição algorítmica, escuta crítica e notação simbólica reflete a ideia de “tecnogênese” proposta por (STIEGLER, 1994), segundo a qual pensamento e percepção “co-evoluem” com a técnica. O sistema de notação, portanto, não é uma simples adição, mas constitui um novo regime de sentido, onde gesto, máscara, espectro e símbolo se conectam. A notação não apenas registra os resultados da aplicação, mas os reinscreve de forma expressiva, abrindo caminho para novas interpretações e transcrições pedagógicas.

O sistema implementado foca na transcrição e análise de intervalos utilizando curvas, contornos e proporções espaciais como vetores simbólicos e expressivos. A proposta de notação se destaca por representar uma ampla gama de intervalos e relações harmônicas não convencionais. Sua estrutura baseada em curvas contínuas e relações espaciais permite representar sistemas de afinações mais complexas, como as das séries xenharmônicas (XENHARMONIC WIKI, 2024).

Do Barro ao Berro: Cerâmica, Algoritmos, Gambiarras e Composição Colaborativa em “Hybris: Ocarinas do Mestre Nado”

As séries xenarmônicas, que envolvem frequências baseadas em lógicas não-temperadas e proporções derivadas de séries subharmônicas, superparticulares, logarítmicas ou irracionais, geralmente carecem de uma notação adequadamente robusta. O sistema proposto busca resolver esse problema representando diretamente as relações intervalares, permitindo que a leitura se baseie em forma, proporção e contorno, em vez de rótulos fixos ou onomástica. A Figura 7 ilustra o mapeamento de frequências e relações intervalares, apresentando um modelo tridimensional em projeção bidimensional, com polígonos que representam interseções de razões e funções escalares.

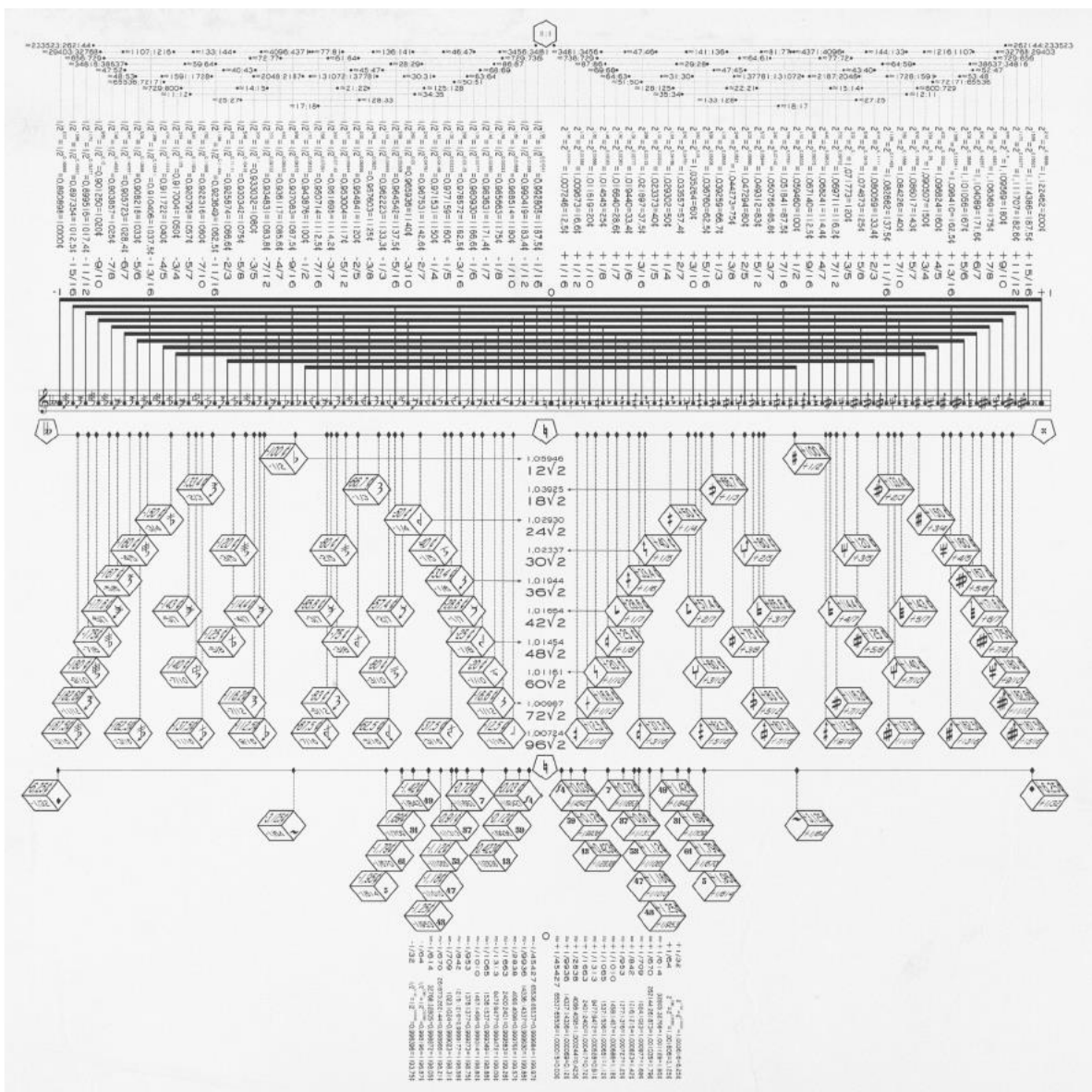


Figura 8: Diagrama organizacional do sistema gráfico proposto.

Fonte: acervo do autor.

O campo de cobertura, ou domínio de representação, que o sistema gráfico-proporcional proposto abrange é amplo, incluindo as séries:

- Harmônica clássica $f_n = n \cdot f_0 (n = 1, 2, 3, \dots)$;
- Sub-harmônica $f_n = \frac{f_0}{n} (n = 1, 2, 3, \dots)$;
- “Overtonal” $f_n = n \cdot f_0 (n = 2, 3, 4, \dots)$;
- “Undertonal” $f_n = \frac{f_0}{n} (n = 2, 3, 4, \dots)$;
- “Utonal” $f_n = \frac{f_0}{r_n} (r_n \in \mathbb{Q})$;
- Delta-racional $f_{n+1} - f_n = \frac{k}{m} \cdot f_0 (k, m \in \mathbb{Z})$.
- Hiper-harmônica $f_n = (n + c)^k \cdot f_0 (c, k \in \mathbb{R})$;
- Hipo-harmônica $f_n = \frac{f_0}{(n+c)^k} (c, k \in \mathbb{R})$;
- Harmônica com múltiplos primos da fundamental $f_n = p_n \cdot f_0 (p_n = 2, 3, 5, 7, 11, \dots)$;
- Harmônica “livre de quadrados” $f_n = q_n \cdot f_0 (q_n = 1, 2, 3, 5, 6, 7, \dots)$;
- Harmônica restrita com apenas múltiplos ímpares $f_n = (2n - 1) \cdot f_0 (n = 1, 2, 3, \dots)$;
- Harmônica com os múltiplos pares da fundamental $f_n = 2n \cdot f_0 (n = 1, 2, 3, \dots)$;
- Baseada na sequência de Fibonacci $f_n = F_n \cdot f_0 (F_n = 1, 1, 2, 3, 5, 8, \dots)$;
- Harmônica fatorial $f_n = n! \cdot f_0 (n = 1, 2, 3, \dots)$;
- Harmônica exponencial $f_n = k^n \cdot f_0 (k \in \mathbb{R}, n \in \mathbb{N})$;
- Harmônica polinomial $f_n = P(n) \cdot f_0 (P(n) \text{ é um polinômio em } n)$;
- Alíquota $f_n = d_n \cdot f_0 (d_n \text{ são divisores de } N)$;
- Harmônica de Bessel $f_n = J_0(n) \cdot f_0 (J_0 \text{ é a função de Bessel de primeira espécie})$;
- Harmônica esticada $f_n = n^\alpha \cdot f_0 (\alpha > 1)$;
- Harmônica comprimida $f_n = n^\beta \cdot f_0 (0 < \beta < 1)$.

O sistema proposto demonstra grande flexibilidade, sendo capaz de representar tanto afinações estabelecidas quanto configurações especulativas e emergentes. No álbum, os intervalos “microtonais” são representados como gestos gráficos inspirados em modificadores (flechas e símbolos combinatórios) inscritos diretamente na curva gráfica. Essa abordagem abrange uma ampla gama de sistemas intervalares, incluindo *Just Intonation* (JI) (FOKKER, 1955) com razões harmônicas complexas como 5:4, 7:4, 11:8 e 13:10.

O método também contempla relações subharmônicas, sistemas xenharmônicos e temperamentais não convencionais, como as afinações derivadas de Bohlen-Pierce (MATHEWS,

1988) e microtonalidades assimétricas. Além disso, permite a representação de divisões temperadas incomuns, razões racionais de limites primos elevados (como 17-limit, 19-limit e 23-limit) e estruturas entropicamente distribuídas. A Tabela 1 apresenta um comparativo entre o sistema implementado e outros sistemas tradicionais e alternativos de notação.

Sistema de Notação	Base Epistêmica	Representação de Intervalos Microtonais	Limitações
Ben Johnston (+/–) (JOHNSTON, 2006)	Entoação justa tradicional	Modificadores simbólicos sobre alturas convencionais	Escala fixa; curva de aprendizado íngreme
Helmholtz–Ellis (SABAT, 2004)	Entoação justa expandida	Diacríticos combinatórios de alta granularidade	Leitura densa e sobrecarregada de símbolos
Sagittal (KEENAN, 2025)	Temperamentos e EDOs múltiplos	Sistema vetorial de setas com precisão centesimal	Exige domínio de convenções padronizadas
Partitura UPIC (Xenakis)	Gesto gráfico-contínuo	Representação visual livre sem notação convencional	Voltado à síntese, não à execução instrumental tradicional
Sistema proposto	Proporções e curvas topológicas	Combinação flexível de signos e trajetórias gráficas	Requer familiarização com leitura topológica e notação combinada

Tabela 1: Comparativo entre sistemas de notação tradicionais e alternativos.

O sistema gráfico-proporcional descrito distingue-se por sua natureza agnóstica em relação ao sistema de afinação, permitindo a representação de qualquer intervalo que possa ser mapeado proporcionalmente. Sua cobertura interválica é, portanto, teoricamente ilimitada, restringindo-se apenas à resolução gráfica e à capacidade perceptiva e interpretativa do leitor, o que amplia significativamente seu potencial de uso em contextos experimentais.

A Figura 8 exemplifica a aplicação prática desse aparato simbólico, mostrando a transcrição de razões intervalares específicas (como 31:26, 87:82, 16:13) em uma pauta adaptada. Cada célula utiliza signos visuais personalizados para codificar informações como vetores interválicos, paridades, classes modulares, direções e identidade de série (harmônica, subharmônica, utonal,

etc.), possibilitando a anotação simultânea de valores racionais exatos, aproximações em *cents* e posições relativas na série fundamental.

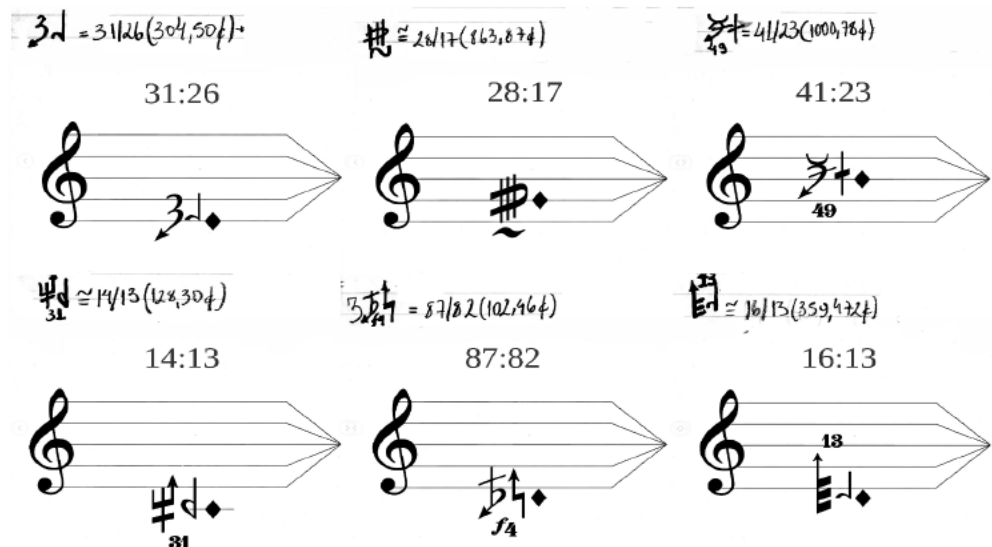


Figura 9: Representação de instâncias práticas da aplicação do aparato simbólico proposto.

Fonte: acervo do autor.

Cada razão ilustrada na imagem pode ser representada através da combinação de signos microtonais com as notas temperadas estipuladas pelo sistema 12-FET. Os exemplos abaixo ilustram tal fenômeno.

- 31:26 (304,5¢): partindo da nota mi no sistema 12-TET ($2^{400/1200} = 2^{1/3} \approx 1.26, 400\text{¢}$)
 - . Signo para -1/16 de tom $\rightarrow (\frac{1}{2})^{\frac{1}{96}} = (\frac{1}{2})^{0,01041} \approx 0,992805 \rightarrow -12,5\text{¢}$
 - . Signo para -5/12 de tom $\rightarrow (\frac{1}{2})^{\frac{1}{72}} = (\frac{1}{2})^{0,06944} \approx 0,953004 \rightarrow -83\text{¢}$ $400\text{¢} - 12,5\text{¢} - 83\text{¢} = 304,5\text{¢}$
- 28:17 (863,87¢): partindo da nota sol no sistema 12-TET ($2^{700/1200} = 2^{7/12} \approx 1.4983, 700\text{¢}$)
 1. Signo para +5/6 de tom $\rightarrow 2^{\frac{5}{36}} = 2^{0,13888} \approx 1,101056 \rightarrow +167\text{¢}$
 2. Signo para -1/64 de tom $\rightarrow (\frac{1}{2})^{\frac{1}{384}} = (\frac{1}{2})^{0,00260} \approx 0,998196 \rightarrow -3,125\text{¢}$ $700\text{¢} + 167\text{¢} - 3,125\text{¢} = 863,88\text{¢}$
- 41:23 (1000.788¢): partindo da nota si no sistema 12-TET ($2^{1100/1200} = 2^{11/12} \approx 1.8877, 1100\text{¢}$)

1. Signo para -4/7 tom $\rightarrow (\frac{1}{2})^{\frac{4}{42}} = (\frac{1}{2})^{0,09523} \approx 0,936117 \rightarrow -114,4\text{¢}$
2. Signo para +1/12 de tom $\rightarrow 2^{\frac{1}{72}} = 2^{0,01388} \approx 1,009673 \rightarrow +16,6\text{¢}$
3. Signo para -1/842 de tom $\rightarrow 1215:1216$ (pertencente ao 49-limit, embora contido já no 19-limit) $\rightarrow -1,424\text{¢}$

$$1100\text{¢} - 114,4\text{¢} + 16,6\text{¢} - 1,424\text{¢} = 1000,776\text{¢}$$

- 14:13 (128.298¢): partindo da nota dó no sistema 12-TET (0¢)

- . Signo para +9/10 de tom $\rightarrow 2^{\frac{9}{60}} = 2^{0,15} \approx 1,109569 \rightarrow +180\text{¢}$
- . Signo para -1/4 de tom $\rightarrow -50\text{¢}$
- . Signo para -1/709 de tom $\rightarrow 1023:1024$ (que pertence ao 31-limit) $\rightarrow -1,69\text{¢}$

$$0\text{¢} + 180\text{¢} - 50\text{¢} - 1,69\text{¢} = 128,31\text{¢}$$

- 87:82 (102.47¢): partindo da nota ré no sistema 12-TET ($2^{1/6} \approx 1.12246$, 200¢)

- . Signo para -11/16 de tom $\rightarrow (\frac{1}{2})^{\frac{11}{96}} = (\frac{1}{2})^{0,114583} \approx 0,923649 \rightarrow -137,5\text{¢}$
- . Signo para +1/5 de tom $\rightarrow 2^{\frac{1}{30}} = 2^{0,033333} \approx 1,023373 \rightarrow +40\text{¢}$
- . Signo para -1/45427 aproximado à proporção 65536:65537 (representado no sistema como f_4) $\rightarrow -0,03\text{¢}$

$$200\text{¢} - 137,5\text{¢} + 40\text{¢} - 0,03\text{¢} = 102,47\text{¢}$$

- 16:13 (359.47¢): partindo da nota ré no sistema 12-TET ($2^{1/6} \approx 1.12246$, 200¢)

- . Signo para +6/7 de tom $\rightarrow 2^{\frac{6}{42}} = 2^{0,142857} \approx 1,104089 \rightarrow +171,6\text{¢}$
- . Signo para -1/16 de tom $\rightarrow (\frac{1}{2})^{\frac{1}{96}} = (\frac{1}{2})^{0,01041} \approx 0,992805 \rightarrow -12,5\text{¢}$
- . Signo para +1/2838 de tom $\rightarrow 4096:4095$ (representado com o signo 13) $\rightarrow 0,423\text{¢}$

$$200\text{¢} + 171,6\text{¢} - 12,5\text{¢} - 0,423\text{¢} = 359,52\text{¢}$$

A notação apresentada revela uma flexibilidade, permitindo representar o mesmo grau temperado de diferentes formas, através da combinação de signos microtonais derivados de

proporções justificáveis no âmbito da entoação justa. Na Figura 9, três soluções distintas são apresentadas para representar o primeiro grau em 19-EDO ($\approx 63,157\text{¢}$).

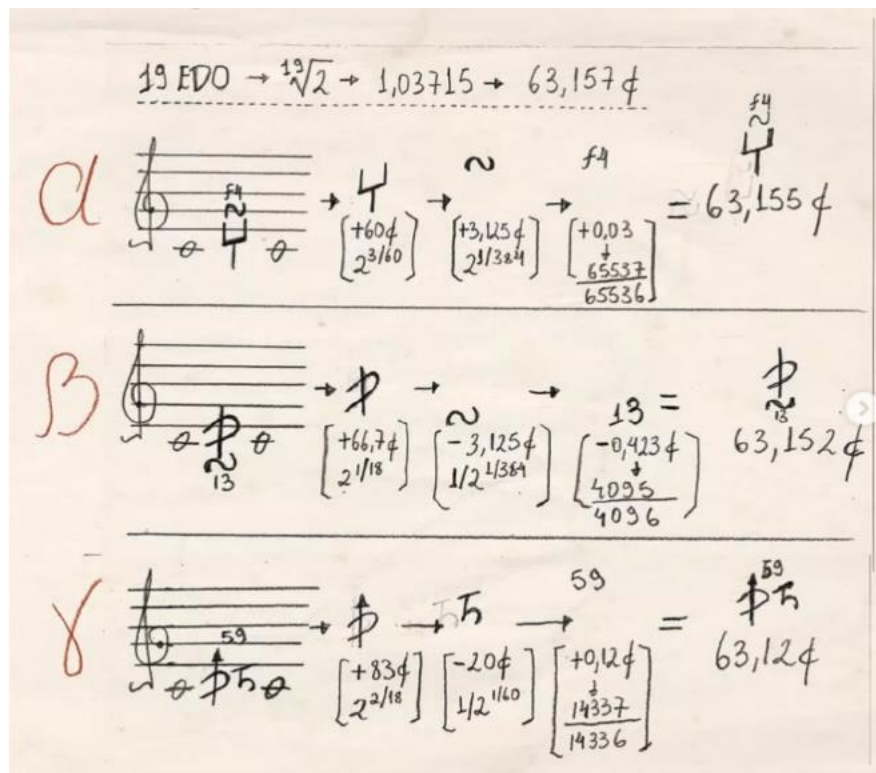


Figura 10: Representações distintas o primeiro grau do 19EDO.

Fonte: acervo do autor.

A primeira (α) parte de um dó temperado, desloca-se por signos sucessivos de $+60,4\text{¢}$, $+3,125\text{¢}$ e $+0,03\text{¢}$ ($65537:65536$), somando aproximadamente $63,155\text{¢}$; a segunda (β) inicia em $+66,7\text{¢}$, $-3,125\text{¢}$ e $-0,423\text{¢}$ ($4096:4095$), resultando em cerca de $63,152\text{¢}$; já a terceira (γ) explora signos mais extremos, como o 59-limit, com alterações de $+83\text{¢}$, -20¢ e $+0,124\text{¢}$ ($14337:14336$), atingindo aproximadamente $63,124\text{¢}$.

O sistema de notação gráfico-proporcional apresenta grande plasticidade, ofertando a navegação entre aproximações de diferentes limites harmônicos mantendo rastreável a procedência intervalar. No ensino, o sistema favorece a prática de séries harmônicas e espectros sonoros, permitindo aos estudantes comparar curvas espectrais e desenvolver intuições sobre simetria, densidade e comportamento não-linear das alturas. Essa abordagem torna mais claras as diferenças entre afinação justa e temperada, consolidando o sistema como recurso para a escrita, leitura e compreensão da xenoharmonia (DARREG, 1974).

No contexto de *Hybris: Ocarinas do Mestre Nado*, a notação gráfico-proporcional não se restringe a uma função analítica, mas é utilizada como modelo de “desubjetivação interpretativa” onde a interpretação musical se desloca de sua função histórica de reconstituição da intenção autoral e da coerência formal, para compreendê-la como um gesto aberto a múltiplas temporalidades e materialidades sonoras. Nesse sentido, a interpretação deixa de ser o lugar privilegiado da subjetividade que valida a obra pela sua unidade orgânica e passa a operar como vetor de diferença, risco e alteridade, acolhendo o imprevisto, a falha e o ruído como dimensões constitutivas da experiência. A desubjetivação interpretativa, portanto, não implica a eliminação do intérprete, mas a suspensão de sua centralidade teleológica: o intérprete já não “realiza” uma forma preexistente, mas inscreve-se em um campo de forças onde gesto e resultado não coincidem plenamente, instaurando um espaço crítico em que a obra acontece como acontecimento singular.

3. A concatenação como síntese colaborativa

A etapa seguinte à criação, sonificação e notação gráfico-proporcional dos desenhos marca um ponto central da obra: a montagem colaborativa dos fragmentos gráfico-sonoros. Inspirada na síntese concatenativa (SCHWARZ, 2006; LAZIER; COOK, 2003; STOWELL; PLUMBLEY, 2007) — técnica que organiza pequenos grãos sonoros por “similaridade” —, essa fase propõe uma “concatenação como síntese colaborativa”, expandindo o conceito para incluir gráficos, ideias harmônicas, ritmos e gestos visuais como elementos manipuláveis. A montagem emerge como processo criativo coletivo, orientado por relações qualitativas e afetivas, e não por uma linearidade temporal rígida.

A “partitura” final, construída artesanalmente com os trechos produzidos em sessões, configura-se como uma colagem polifônica: blocos visuais-sonoros autônomos funcionam como camadas interdependentes. Essa montagem combina critérios rítmico-visuais, espaciais e lúdicos, como cartas sonoras organizadas manual e algoritmicamente. Na dimensão performativa, os desenhos sonificados foram transpostos por improvisos com o ambiente de *live coding* TidalCycles (McLEAN, 2025), cuja lógica de padrões temporais favorece a concatenação expressiva e dinâmica dos eventos. Assim, a concatenação colaborativa adquire um caráter algorítmico-improvisativo, com manipulação em tempo real dos módulos gráficos convertidos em parâmetros sonoros. A Figura 10 apresenta um fragmento da partitura resultante desse processo.

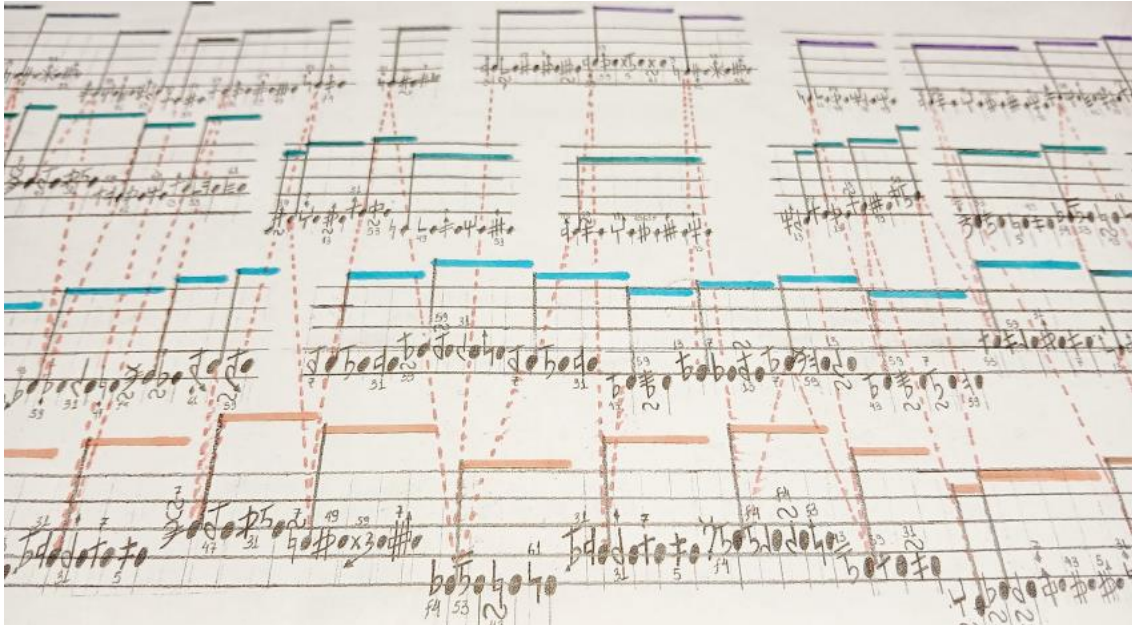


Figura 11: Fragmento da partitura resultante do processo de “concatenação como síntese colaborativa”.
Fonte: acervo do autor.

A montagem colaborativa, orientada por padrões e manipulação de eventos, cria texturas dinâmicas, polirritmias e camadas justapostas que refletem a diversidade gráfica dos desenhos. Cada participante, ao inserir um fragmento visual, adiciona uma unidade sonora potencial, que será interpretada, distorcida, deslocada e remixada pelo algoritmo implementado, transformando a colaboração do gesto gráfico sonificado em uma performance algorítmica.

Um trecho de improvisação com código, comentado abaixo, ilustra a concatenação como síntese colaborativa. Nesse exemplo, fragmentos sonoros derivados de gestos gráficos, declarados no excerto como “desenho”, são reorganizados em tempo real pela microlinguagem TidalCycles, utilizando o paradigma funcional em Haskell para articular padrões temporais complexos, sobreposições rítmicas e variações. A recombinação dos elementos transforma o gesto gráfico original em um tecido polifônico, promovendo improvisação e múltiplas significações coletivas.

```
-- Define o andamento (cycles per second): tempo base da sessão
cps (11/110)

do
  -- Função auxiliar: ativa trechos com base em ciclos de modulação
  let whenmod' n s e = when ( (\c -> (c >= s) && (c < e)).(`mod` n) )
  -- Define a duração de referência para oscilações e envelopes
  let len = 8
  -- Define o tempo base de deslocamento entre camadas concatenadas
  let time = (8/3)
  -- Define a célula sonora básica a ser reutilizada: fragmentos "desenho" como base
  rítmica
```

```
let des = stack [sound "[desenho:2, desenho:3]" # shape 0.5 # speed 0.8 # coarse 2 #
gain 1.1, sound "desenho" # gain 0.8 # speed 0.8 # coarse 6 # legato 0.5]

-- Stream principal (d1): estrutura concatenada de fragmentos gráficos sonoros
d1 $ fast 9 $ trunc 0.36 $ brak
  $ stack [
    -- Seção com três camadas deslocadas temporalmente,
    -- simulando a concatenação de ideias rítmicas
    stack[
      ((time*0) ~>) $ e 4 4 $ des |*| gain (slow len $ scale 0 1 sine1),
      ((time*1) ~>) $ e 6 3 $ des |*| gain (slow len $ scale 0 1 sine1),
      ((time*2) ~>) $ e 3 8 $ des |*| gain (slow len $ scale 0 1 sine1)
    ],
    -- Seção com elementos mais texturais: camada coletiva, heterogênea
    every 3 (within (0.25, 0.75) (slow 1.5))
    $ stack [
      -- Fragmentos diferentes de "desenho", combinando com randomização leve
      sound "desenho:4?" # shape 0.5 # speed 0.7 # gain 1 # coarse 4,
      sound "desenho:5?" # speed 0.8,
      sound "desenho:6?" # speed "0.7",
      -- Textura granular com variações
      stut 4 0.5 0.25 $ sound "desenho:7*4" # hpf 200 # speed (slow 2 $ scale 1 1.1
sine1) # lpf (slow 4 $ scale 800 1200 tri1) # lpq 0.15 # shape 0.2 # coarse 6 # gain 1,
      -- Efeitos de eco/dobra, aplicando atrasos
      stut 12 0.5 1 $ (0.125 ~>) $ sound "desenho?" # shape 0.5 # up (-3) # lpf 150 #
legato 0.2 # gain 0.9,
      -- Layer lenta de transição textural, reforçando a ideia de colagem sonora
      slow 1.5 $ sound "desenho:9(1,4,1)" # gain 1.25 # coarse 4 # shape 0.5
    ],
    -- Filtro condicional que controla entrada/saída do "desenho:8"
    -- Exemplo de controle de presença em bancos colaborativos
    whenmod' 32 0 16 (# gain 0) $ gain "[0.6 1]*4" # sound "desenho:8" # speed 1.1 #
legato 0.1
  ]
  -- Parâmetros globais de espacialização e modulação
  # room 0.1 # speed (rand) # cut "1" # pan (rand) |*| gain 1.3
-- Stream secundária (d2): manipulação probabilística de fragmentos gráficos
d2 $ sometimesBy 0.73 (|+| up (rand))
  $ stack [
    -- Fragmento direto
    sound "desenho/2" # gain 1.1,
    -- Ações estocásticas com base em padrões rítmicos binários
    stut 6 0.5 1 $ gain "0 1 0 0 0 0 0 0" # sound "desenho:9" |*| gain 0.9,
    -- Fragmentos concatenados dinamicamente com variação de ganho e velocidade
    gain "0 1 0 0 0 0 0 0" # sound "desenho:7 desenho:19" |*| speed (rand) |*| gain
(slow 4 $ scale 0.8 1.1 sine1),
    -- Camada lenta de modulação com escolha aleatória de velocidade
    slow 1.5 $ gain "0 0 0 0 0 0 1" # sound "desenho:5" # speed (choose [0.5, 0.75,
1]) |*| gain 1.3
```

```

]
-- Espaço acústico e modelagem espectral global
  # room 1 # size (slow 8 $ scale 0.19 0.95 tri1) # orbit 1 |*| gain 0.9 # cut "0.5"
# accelerate "[0.05 -0.05 0.03 -0.03 0.01 -0.01]/3"

```

No excerto de improvisação referendado no código, o arcabouço conceitual da “concatenação” assim se apresenta: seja D o conjunto finito de fragmentos sonoros de “desenho” gerados colaborativamente constituinte da matéria-prima da performance nesta fase: $D = \{d_1, d_2, d_3, \dots, d_n\}$ onde cada $d_i \in D$ corresponde a uma “composição” única (e.g., síntese do “desenho”) associado ao gesto gráfico de um participante. O processo de “transformação” é governado por três conjuntos de operadores: T para transformações determinísticas (e.g., *speed*, *gain*, *shape*, *stut*, *slow*, deslocamento temporal $\sim >$), E para operadores estocásticos (e.g., *rand*, *sometimesBy*, *choose*) e C para operadores condicionais (e.g., *whenmod*).

Uma *stream* $S_i(t)$ é definida como uma função que combina elementos de D com operações de T , E e C , ou seja: $S_i(t) \in F(D, T, E, C)$. O compósito resultante $A(t)$ é a soma de k *streams* independentes: $A(t) = \sum_{i=1}^k S_i(t)$. A *stream* principal $S_i(t)$ é estruturalmente complexa, organizada em camadas concorrentes e transformadas globalmente:

$$S_1(t) = [(\sum_{a=0}^2 P_a(t + \Delta_a)) \oplus P_4(t) \oplus P_5(t)]|G$$

onde: $P_a(t + \Delta_a)$ concerne a camadas rítmicas deslocadas temporalmente. Cada P_a é uma “célula sonora” básica (e.g., o agregado *desenho*) modulada por uma envoltória de ganho senoidal $gain = f(sine_1)$. Os deslocamentos $\Delta_a = \{0, time, 2 \cdot time\}$ implementam uma concatenação de ideias rítmicas. A camada textural-estocástica $P_4(t)$ inclui: seleção estocástica de elementos de D (e.g., “*desenho:3*”), granulação (*stut*) e efeitos de “eco” (*stut* com deslocamento), desaceleração (*slow*) e modulação por um operador de periodicidade $E_{3,(0.25,0.75)}$ (*every 3 (within (0.25, 0.75))*). O estrato condicional $P_5(t)$ é controlada por um operador $C_{32,0,16}$ (*whenmod' 32 0 16*) que a silencia fora do intervalo de ciclos especificado. A camada de transformações globais G é aplicada à *stream* completa: $G = T_{room(0.1)} \circ T_{cut(1)} \circ T_{pan(rand)} \circ T_{gain(1.3)}$.

A *stream* secundária é caracterizada por uma aplicação probabilística de transformações sobre um conjunto de padrões binários:

$$S_2(t) = P_{0.73}(\sum_{j=1}^4 P_j(t))|G'$$

onde: $P_{0.73}$ é o perador probabilístico (*sometimesBy* 0.73) que aplica uma transformação estocástica $|+| \text{ } up(rand)$ com 73% de chance; $\sum P_j(t)$ perfaz-se na soma de quatro sub-camadas de padrões, onde cada $P_j(t)$ é um padrão binário (e.g., *gain* "0 1 0 0 0 0 0") aplicado a elementos específicos de D , frequentemente combinado com variações estocásticas de velocidade e ganho e G' é o espaço acústico global definido por: $G' = T_{room(1)} \circ T_{size(f(tri1))} \circ T_{cut(0.5)} \circ T_{gain(0.9)}$. A performance algorítmico-improvisativa aqui modelada neste excerto de código pode ser compreendida com a função:

$$A(t) = S_1(t) + S_2(t) \in F(D, T, E, C)$$

Este modelo, com sua “partitura de processos” ilustrada na Figura 12, demonstra que a complexidade do resultado emergente $A(t)$ surge não do intrincado agregado inerente dos materiais em D , mas da aplicação — determinística, estocástica e condicional — de um conjunto de operações de transformação sobre estes materiais.

A “concatenação colaborativa” é, formalmente, um sistema de funções que operam sobre um espaço de estados (“desenhos”) dentro de um domínio temporal. A noção de concatenação ganha uma dimensão estética e política: cada fragmento carrega a identidade de seu criador, mas transforma-se ao se aglutinar aos demais. Esse processo ecoa a ideia de uma autoria difusa e compartilhada, onde o gesto individual só ganha sentido em sua articulação com o todo. A escuta é também aqui uma escuta da montagem, do vínculo, da costura. A “partitura” final, remanescente da estética de *Topofonica* de Bogusław Schaeffer (SCHAEFFER, 1960), apresenta uma organização em camadas densas e móveis, onde o tempo é tratado como campo de sobreposição, e não de sucessão linear.

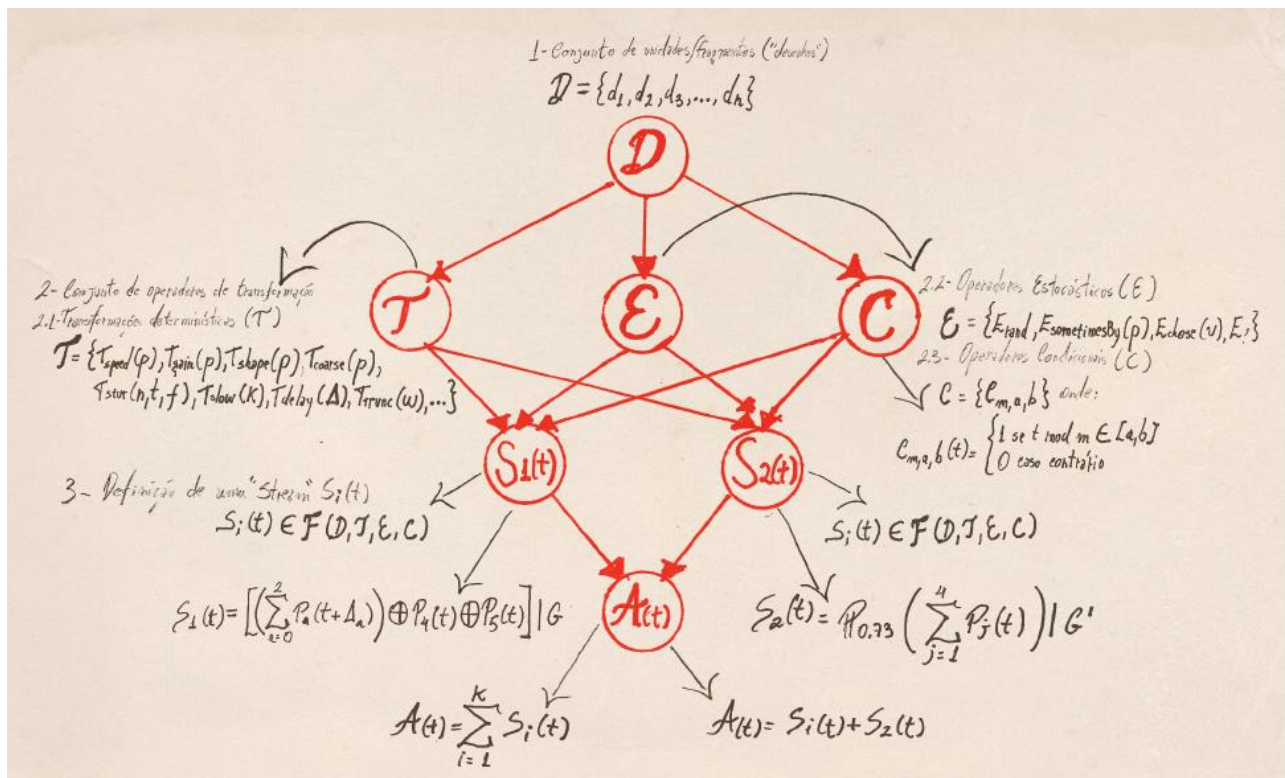


Figura 12: Diagrama em blocos ou "partitura de processos" do trecho do código em Haskell

Fonte: acervo do autor

Como em Schaeffer, a notação gráfica não fixa, mas ativa: sugere gestos, abre caminhos de leitura múltipla, convoca o "intérprete" à criação. A concatenação como síntese colaborativa revela, portanto, um gesto composicional que já não separa montagem de invenção, codificação de escuta, técnica de política. Cada camada é uma escolha; cada sobreposição, um convite à escuta do outro.

A concatenação atua como a ponte entre o gesto gráfico e a "realização partitural", funcionando simultaneamente como uma fase intermediária de montagem, performatização e escuta colaborativa, um momento de estruturação macro em que os desenhos ou fragmentos dos gestos gráficos sonificados são combinados e transformados, e um eixo coletivo e algorítmico que conecta a abstração da notação à irre realidade concreta dos desenhos transduzidos.

Essa etapa não apenas amplia a lógica da composição colaborativa, mas também materializa a potencialidade da notação gráfica, preparando o terreno para processos de espacialização e gravação das ocarinas enquanto formas de escritura acústica. A Figura 13 apresenta o excerto da partitura resultante do processo de concatenação como síntese colaborativa. Os fragmentos gráficos, derivados de desenhos livres e posteriormente sonificados, foram reorganizados segundo padrões temporais modelados em improvisações com TidalCycles. A transcrição musical preserva tanto a expressividade gráfica dos gestos originais quanto a lógica rítmica emergente da interação algorítmica, culminando em uma escrita híbrida entre escuta, código e traço.

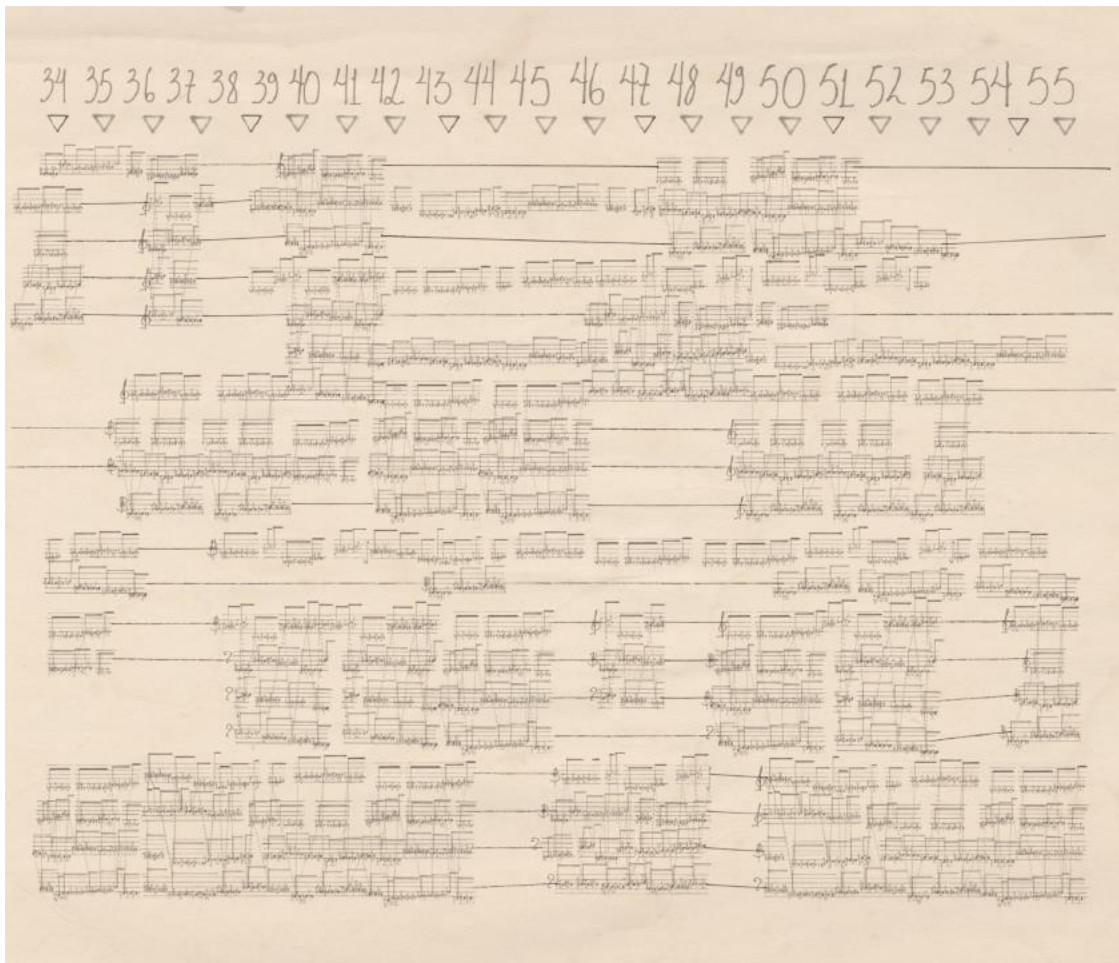


Figura 13: Excerto da partitura resultante do processo de concatenação.

Fonte:acervo do autor

4. Espelhos do sopro: ressonâncias convolutivas do arquétipo cerâmico

A partir das etapas anteriores — que traduziram o gesto gráfico em estruturas harmônicas, formalizaram-no em notação gráfico-proporcional e o reorganizaram como síntese colaborativa — , a investigação avança para o espaço acústico como matéria de composição. Agora, a escuta se desdobra em topologia vibrátil, na qual os ambientes ressoam como dispositivos de inscrição sonora. O desenvolvimento de um reverberador de convolução via FFT, a partir de respostas impulsivas captadas em objetos cerâmicos confeccionados por Mestre Nado, permite não apenas simular espaços acústicos, mas virtualizar memórias materiais. O espaço reverberante, assim, não é mero efeito estético, mas uma camada de sentido, onde a escuta se aloja como experiência sensorial e simbólica.

Inspirado na “matéria-forma técnica” de (SIMONDON, 1958), este módulo atua como luteria pós-digital, fundindo eficiência algorítmica à “materialidade cultural”. A técnica aqui não se limita à manipulação de sinais: opera como individuação simbólica entre corpo, som e ambiente, aproximando-se da ideia de (FLUSSER, 2007) de que o digital não representa o real, mas o reconstrói. A cerâmica, mais que suporte, é superfície vibratória e memória mineral do gesto. Ao capturar seus modos de ressonância, o projeto ativa uma poética do espaço como volume sensível (MALASPINA, 2021), no qual reverberam gestos, matérias e inscrições cultivadas. O reverberador torna-se, assim, ferramenta de escavação sonora — como propõe (PARIKKA, 2015) —, transformando a escuta em escultura e a convolução em ritual de recomposição ontológica, onde o digital prolonga, em vez de suprimir, a fisicalidade.

4.1 O método *overlap-add* e a convolução por FFT: técnicas eficientes para processamento segmentado de sinais

O processamento digital de sinais (PDS) enfrenta limitações relacionadas à memória e ao tempo de resposta, especialmente ao lidar com sinais longos. Duas técnicas eficazes para contornar esses desafios são o método *overlap-add* (OPPENHEIM; SCHAFER, 2009) e a convolução por FFT, que permitem o processamento segmentado no domínio da frequência. O método *overlap-add* divide o sinal em blocos menores, processa cada segmento separadamente e os recombina, mantendo a integridade do sinal. Ao utilizar a FFT, substitui-se a convolução direta por uma multiplicação espectral, reduzindo significativamente a carga computacional.

A convolução por FFT explora a equivalência entre a convolução no tempo e a multiplicação no domínio da frequência. Em tempo real, como em sistemas de telecomunicações, sua baixa latência torna-a imprescindível. A convolução de um sinal de 300 amostras com um *kernel* de 101 gera uma saída de $N + M - 1$ amostras (400 nesse caso), e o método *overlap-add* lida com essa expansão segmentando a entrada (por exemplo, em blocos de 100 amostras), adicionando zeros e somando regiões sobrepostas. Essa abordagem assegura equivalência matemática à convolução direta, com o benefício de distribuir a carga de processamento. A sobreposição intencional entre segmentos de saída no método *overlap-add* exige soma precisa para preservar a integridade do sinal. Apesar da complexidade algorítmica envolvida, sua combinação com a convolução por FFT oferece alta eficiência no processamento de sinais longos e aplicações em tempo real.

A convolução via FFT baseia-se no princípio de que multiplicação no domínio da frequência equivale à convolução no tempo. Embora esse conceito seja antigo, sua aplicação prática só se

tornou viável após o algoritmo FFT desenvolvido por (COOLEY; TUKEY, 1965), que reduziu a complexidade da transformada discreta, DFT, de $O(N^2)$ para $O(N\log N)$. Essa inovação estabeleceu a chamada *convolução de alta velocidade*, composta por três etapas: FFT do sinal, multiplicação espectral e retorno ao tempo com FFT inversa (IFFT). Apesar do percurso algorítmico distinto, o resultado é matematicamente idêntico à convolução linear, com enorme ganho em eficiência — o que viabilizou aplicações em tempo real antes inviáveis.

A Figura 14, adaptada de (SMITH, 2013) apresenta uma aplicação do método *overlap-add*:

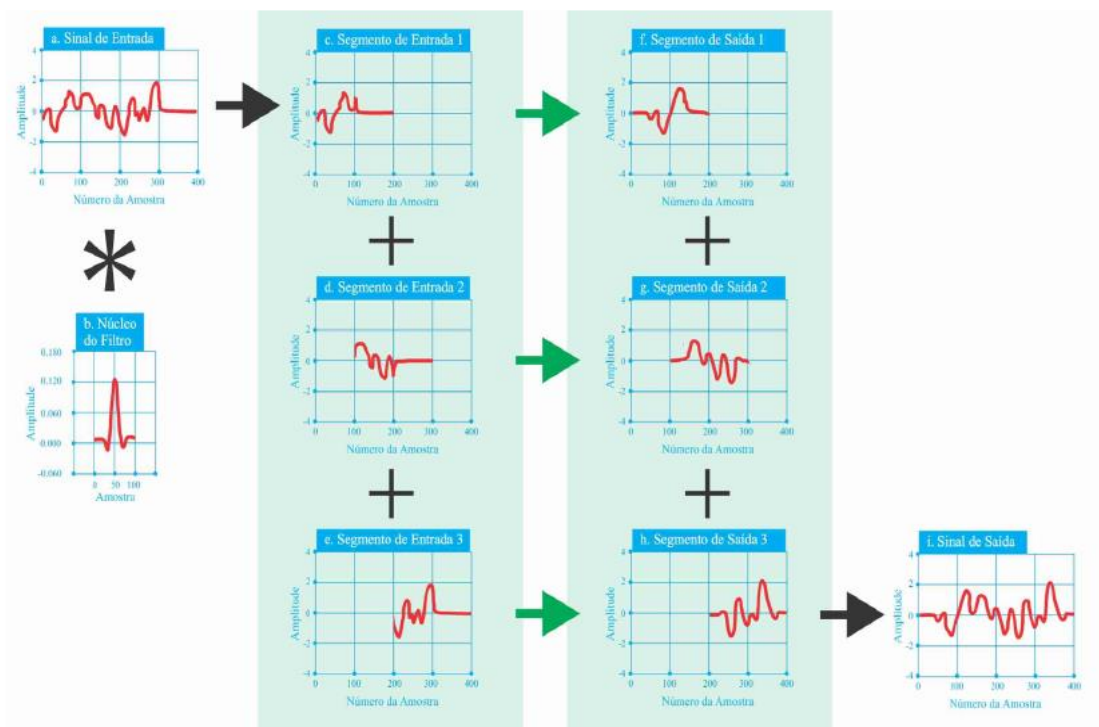


Figura 14: Exemplo de aplicação do método *overlap-add*.

No exemplo, um sinal de entrada discreto (a) é convoluído junto a um *kernel* de filtro (b), especialmente útil para sinais longos ou sistemas com recursos limitados. O sinal é dividido em segmentos de comprimento fixo (c), (d) e (e), com zeros adicionados ao final de cada segmento para acomodar a expansão da convolução, que resulta em $N + M - 1$ amostras (onde N são as amostras do segmento e M os coeficientes do filtro). Esse *zero-padding* preserva a informação espectral durante o processamento.

Cada segmento ampliado passa pela operação de filtragem com o *kernel* (b), gerando saídas sobrepostas (f), (g), (h). A soma das regiões sobrepostas resulta no sinal de saída completo (i), matematicamente idêntico à convolução direta. Essa abordagem é garantida pela equivalência da convolução segmentada e direta, assegurada pelo teorema da convolução linear.

As vantagens computacionais são claras, pois o método permite processar sinais longos de maneira eficiente, dividindo-os em blocos gerenciáveis e reduzindo os requisitos de memória, o que é essencial em sistemas embarcados e de tempo real. Além disso, preserva a integridade do sinal sem distorções, sendo amplamente adotado em aplicações como processamento de áudio, telecomunicações e análise de sinais biomédicos, onde o equilíbrio entre desempenho e precisão é crucial.

A Figura 15, adaptada de (SMITH, 2013), ilustra um processo convolucional por FFT:

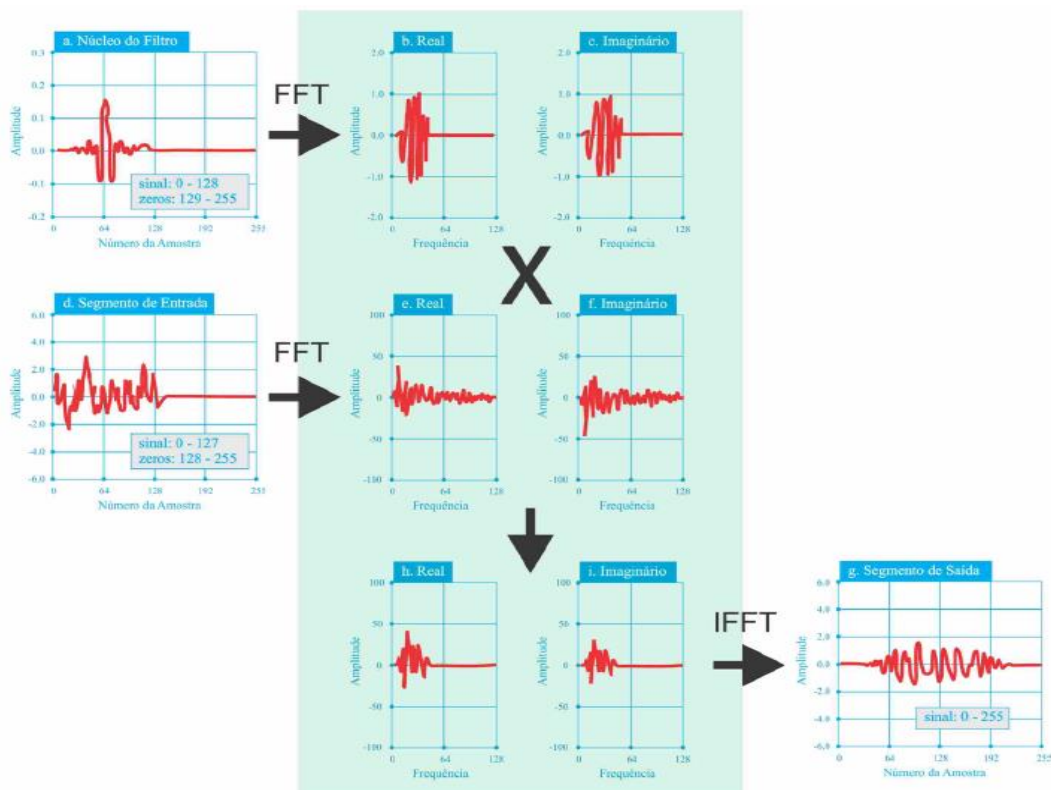


Figura 15: Exemplo de convolução por FFT.

O processo começa com a preparação do filtro, representado pelo *kernel* (a), que pode ser um filtro passa-banda do tipo “janela-sinc”. Este filtro é transformado para o domínio da frequência usando a FFT, gerando as componentes real e imaginária mostradas em (b) e (c). Esses dados são armazenados para uso posterior.

Quando analisamos um segmento do sinal de entrada (d), aplicamos a FFT novamente, gerando seu espectro de frequência (e) e (f). A multiplicação ponto a ponto entre o espectro do sinal e a resposta do filtro resulta no espectro de saída (h) e (i), uma operação matematicamente equivalente à convolução no domínio do tempo, mas muito mais eficiente computacionalmente.

Um ponto crucial é o dimensionamento adequado da FFT para evitar distorções devido à convolução circular. Para isso, o tamanho da FFT deve corresponder ao comprimento do segmento

de saída. No exemplo ilustrado, o *kernel* de 129 pontos e segmentos de 128 pontos requerem FFTs de 256 pontos. Para isso, são adicionados zeros a eles. Em casos práticos, como sinais longos com filtros de 600 pontos, podemos usar segmentos de 425 pontos com FFTs de 1024 pontos ou segmentos de 1449 pontos com FFTs de 2048 pontos. Essas opções equilibram eficiência computacional, memória e latência, adaptando-se às necessidades específicas de cada aplicação.

A escolha entre a convolução tradicional e por FFT depende do tamanho do *kernel* e dos requisitos de precisão. Para *kernels* curtos (≈ 60 pontos), a tradicional é mais eficiente, mas para *kernels* maiores, a por FFT se torna muito mais eficiente. Um *kernel* de 16.000 pontos, por exemplo, requer apenas o dobro do tempo de processamento de um *kernel* de 64 pontos com FFT. A precisão numérica também é superior na convolução por FFT, reduzindo erros de arredondamento.

Enquanto a convolução padrão com *kernel* de 1.000 pontos em precisão simples pode introduzir ruídos na ordem de 1 parte em 20000, a abordagem por FFT atinge uma precisão dez vezes superior, reduzindo esse ruído para 1 parte em 200000. Dessa forma, recomenda-se reservar a convolução por FFT para cenários específicos, como o processamento de sinais com centenas de milhares a milhões de amostras combinado com *kernels* que ultrapassem centenas de pontos. A Tabela 2 compara os métodos apresentados.

Característica	Convolução via FFT	Convolução Direta
Complexidade	$O(N \log N)$ (rápido)	$O(N^2)$ (lento)
Uso em tempo real	Sim (baixa latência)	Não (alta latência)
Tamanho da IR	Adequado para IRs longas	Limitado a IRs curtas
Recursos	Consome mais RAM (<i>buffers</i> FFT)	Menos RAM

Tabela 2: Comparativo entre ambos os métodos de convolução.

4.2 Implementação de um reverberador de convolução via FFT

A implementação de um reverberador por convolução via FFT exige traduzir os princípios do método *overlap-add* em estruturas de algoritmos eficientes. O processamento baseia-se na divisão do sinal de entrada e da resposta impulsiva (IR) em partições de tamanho fixo — geralmente

potências de 2, como 2048 ou 4096 amostras — para aproveitar a eficiência da FFT. Matematicamente, esse particionamento pode ser descrito como:

$$h[n] = \sum_{i=0}^{P-1} h_i[n - iL]$$

onde h_i representa a i -ésima partição da IR de tamanho L , e P é o número total de partições.

A integração da convolução particionada no SuperCollider (COLLINS, 2008) é uma UGen (geradora de unidade) especializada que utiliza o método de partição uniforme para reduzir a latência em convoluções de longa duração. No método `preprocessIR()`, a IR armazenada em `m_irBuffer` é dividida em partições de tamanho `m_partitionSize` (potência de 2), calculando antecipadamente os espectros de cada partição. Cada bloco é convertido para o domínio da frequência através da FFT:

$$H_i[k] = \sum_{n=0}^{N-1} h_i[n] e^{\frac{-j2\pi kn}{N}}$$

com N sendo o tamanho da FFT (incluindo *zero-padding* para evitar convolução circular). A FFT é calculada para cada partição e armazenada em `m_partitions` onde `m_fft->forward()`. Essa pré-computação é crucial para garantir baixa latência durante o processamento em tempo real.

```
void PartitionedConvolution::preprocessIR() {
    // ... (inicializações e verificações)

    // Divide a IR em partições e calcula as FFTs
    for (int i = 0; i < m_numPartitions; ++i) {
        int offset = i * m_partitionSize; // Deslocamento para a i-ésima partição
        int remainingSamples = m_irBuffer.size() - offset;
        int samplesToCopy = std::min(m_partitionSize, remainingSamples);

        // Preenche o buffer de entrada da FFT com zeros (zero-padding)
        std::fill(m_fftInput.begin(), m_fftInput.end(), 0.f); // Zero-padding
        std::copy_n(m_irBuffer.data() + offset, samplesToCopy, m_fftInput.begin());

        // Calcula a FFT da partição atual
        m_fft->forward(m_fftInput.data(), m_fftOutput.data());

        // Armazena o espectro da partição para uso posterior
        std::copy_n(m_fftOutput.begin(), m_fftSize, m_partitions[i].begin());
    }
}
```

O tamanho da FFT ($m_fftSize$) é um parâmetro crítico: valores maiores reduzem a frequência de cálculos, mas aumentam a latência. Por exemplo, para $m_fftSize = 2048$ a 44.1 kHz, a latência mínima é de $2048/44100 \approx 46.4$ ms. Tal parâmetro é definido na inicialização:

```
: m_fftSize(fftSize),  
  m_partitionSize(partitionSize) {  
  // ...  
}
```

Durante a execução do método *next()*, o sinal de entrada é acumulado em um *buffer* circular até preencher um bloco completo. Quando pronto, a FFT é aplicada:

$$X_m[k] = \sum_{n=0}^{N-1} x_m[n] e^{-j2\pi kn/N}$$

onde $x_m[n]$ é o m -ésimo bloco de entrada.

```
/// Método next() - Processa uma amostra de entrada por vez  
void PartitionedConvolution::next(float inSample) {  
  // --- Passo 1: Acumulação no Buffer Circular ---  
  // Armazena a amostra de entrada na posição atual do buffer  
  m_inputBuffer[m_inputPosition] = inSample;  
  
  // Avança a posição do buffer (com wraparound usando módulo)  
  m_inputPosition = (m_inputPosition + 1) % m_fftSize; // Ex.: Se m_fftSize = 1024, o  
  buffer "circular" volta para 0 após a posição 1023  
  
  // --- Passo 2: Verificação de Bloco Completo ---  
  // Incrementa o contador de amostras processadas  
  if (++m_samplesProcessed >= m_partitionSize) {  
    // Reset do contador quando atinge o tamanho do bloco  
    m_samplesProcessed = 0;  
  
    // --- Passo 3: Processamento do Bloco ---  
    processInputBuffer(); // Método que executa: FFT → Multiplicação → IFFT  
  }  
}
```

O método *next()* é o coração do processamento em tempo real, acumulando amostras em um *buffer* circular até formar um bloco completo, disparando a FFT quando o bloco está pronto ($m_partitionSize$ amostras) e executando a convolução no domínio da frequência (multiplicação de FFT). A Figura 16 ilustra o fluxo de processamento por blocos com *buffer* circular, onde: (1) o fluxo principal mostra a entrada de amostras no *buffer* (tamanho $m_fftSize$), a verificação de bloco

completo e o processamento via FFT/convolução/IFFT; (2) o detalhe do *buffer* exemplifica a estrutura circular com 8 posições (0 a 7), destacando em vermelho a posição atual do ponteiro ($m_inputPosition = 2$); e (3) o *loop* de acumulação mantém o preenchimento contínuo até atingir $m_fftSize$. As cores codificam funções: azul (entrada/dados), verde (processamento), vermelho (posição ativa) e tracejado (fluxo de controle). A notação pontilhada indica continuidade do *buffer* para tamanhos arbitrários.

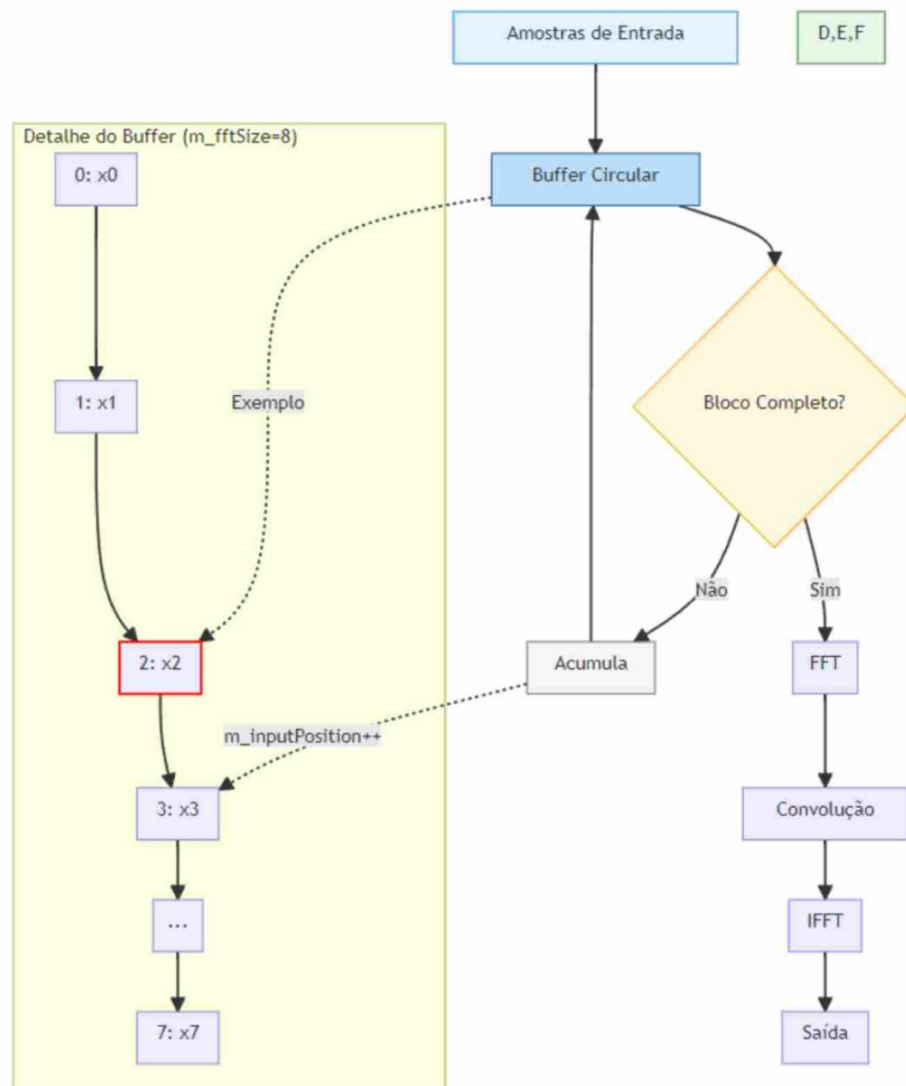


Figura 16: Diagrama do fluxo de processamento por blocos com buffer circular.

Fonte: acervo do autor.

A convolução no domínio da frequência reduz-se a uma multiplicação complexa, ponto a ponto, entre $X_m[k]$ e cada $H_i[k]$:

$$Y_{m,i}[k] = X_m[k] \cdot H_i[k]$$

implementada algoritmicamente pela função *complexMultiply()*, que executa a multiplicação complexa entre o espectro do sinal de entrada (*m_fftOutput*) e o da IR (*m_partitions[i]*):

$$Y = \begin{pmatrix} X_R \cdot H_R - X_I \cdot H_I \\ X_R \cdot H_I + X_I \cdot H_R \end{pmatrix}$$

```
void complexMultiply(float* a, float* b, float* out, int size) {  
    for (int i = 0; i < size; i += 2) {  
        float re_a = a[i], im_a = a[i + 1];  
        float re_b = b[i], im_b = b[i + 1];  
        out[i] = re_a * re_b - im_a * im_b; // Parte real  
        out[i + 1] = re_a * im_b + im_a * re_b; // Parte imaginária  
    }  
}
```

Os resultados são convertidos de volta ao domínio do tempo via IFFT e acumulados no *buffer* de saída com sobreposição (*overlap-add*), garantindo a reconstrução do sinal. A sobreposição é calculada para cobrir exatamente *L* amostras (metade de *m_fftsize*), seguindo a propriedade da convolução linear:

$$y[n] = \sum_{i=0}^{P-1} y_i[n - iL]$$

onde y_i são as saídas parciais de cada partição. Após a IFFT, as saídas são somadas com sobreposição:

```
m_fft->inverse(m_convOutput.data(), m_ifftOutput.data()); // IFFT  
  
// Overlap-add: acumula no buffer de saída  
for (int i = 0; i < m_fftSize; ++i) {  
    m_outputBuffer[i] += m_ifftOutput[i] * m_window[i]; // Janela de sobreposição  
}
```

A janela de Hann (*m_window*) é aplicada conforme:

$$w[n] = 0.5(1 - \cos(\frac{2\pi n}{N-1})), 0 \leq n \leq N-1$$

A técnica de *overlap-add* com janelamento, essencial no processamento de áudio via FFT/IFFT, é realizada em três etapas: a IFFT reconverte o sinal convoluido para o domínio do tempo (armazenado em *m_ifftOutput*); em seguida, cada amostra é multiplicada por uma janela de Hann pré-computada (*m_window*), suavizando bordas e evitando descontinuidades entre blocos; por fim, o resultado é acumulado no *buffer* de saída (*m_outputBuffer += m_ifftOutput * m_window*), assegurando reconstrução contínua. Essa abordagem reduz artefatos como cliques e vazamentos espectrais. O *overlap*, geralmente de 50% ou 75%, garante a integridade do sinal, e embora a janela

de Hann seja ideal por suavizar extremidades, outras como Hamming ou Blackman também podem ser empregadas conforme a necessidade de atenuação.

Para evitar artefatos, o algoritmo implementado opera *zero-padding* implícito, estendendo cada bloco para $N = 2L$, garantindo que a convolução no domínio da frequência seja equivalente à linear no tempo. A escolha de $m_fftsize$ como potência de 2 permite usar FFT *radix-2*, reduzindo a complexidade de $O(N^2)$ para $O(N\log N)$. IR longas, como reverberações de 3 segundos (~132k amostras a 44,1 kHz), exigem particionamento hierárquico, que economiza RAM mas aumenta a latência. O SuperCollider contorna isso com *buffers* circulares e alocação dinâmica, mantendo a qualidade mesmo em sistemas com poucos recursos de processamento.

Em reverberadores algorítmicos, respectiva técnica permite convoluir sinais com IR acústicas complexas em tempo real, preservando suas características temporais e espectrais com o método *overlap-add*. Para IR com mais de 10k amostras, a convolução por FFT pode ser até 50× mais rápida que a implementação direta no domínio do tempo.

O código-fonte do algoritmo enfrenta vários desafios práticos:

- Gerenciamento de IR longas: para uma IR de 3 segundos a 44.1 kHz (132.300 amostras), o número de partições é:

$$P = \lceil 132300/1024 \rceil = 130 \text{ partições}$$

O consumo de memória é otimizado armazenando apenas os espectros FFT (N pontos complexos por partição);

- Precisão numérica: a implementação usa aritmética de ponto flutuante de 32 *bits*, com erro relativo da ordem de 10^{-7} por operação. Para P partições e K blocos processados, o erro acumulado é:

$$\epsilon_{total} \approx K \cdot P \cdot 10^{-7}$$

- Eficiência em tempo real: o uso da biblioteca FFTW (com planejamento prévio) reduz o custo computacional para:

$$C \approx P \cdot (2N\log N + 6N) \text{ operações por bloco}$$

Para simular o ambiente reverberante de uma catedral — cuja IR possui 5 segundos de duração —, o sistema divide a resposta impulsiva em aproximadamente 215 partições de 1024 amostras cada, processa blocos de entrada a cada 23.2 milissegundos (garantindo uma baixa latência) e mantém o uso da CPU abaixo de 15% mesmo em processadores modernos, assegurando

eficiência computacional sem comprometer a qualidade do efeito. A Tabela 3 e o Algoritmo 1 resumem o processo.

Conceito Teórico	Implementação no Código
Particionamento da IR	<i>Loop em preprocessIR() com m_partitionSize</i>
Pré-computação das FFT	<i>m_fft → forward() + armazenamento em m_partitions</i>
Multiplicação complexa	Função <i>complexMultiply()</i>
<i>Overlap-Add</i>	Acumulação em <i>m_outputBuffer</i> após IFFT
Latência da FFT	Parâmetro <i>m_fftSize</i> no construtor

Tabela 2: Métodos de implementação dos conceitos discutidos.

O gerenciamento de memória na convolução por FFT otimiza o processamento de IR longas com alocação dinâmica e armazenamento esparsos. A implementação da *UGen* divide a IR em blocos de tamanho fixo, convertidos para o domínio da frequência via FFT e armazenados como espectro complexo em *m_partitions*, reduzindo o uso de memória em ~50%. Buffers circulares reutilizáveis (*m_inputBuffer* e *m_outputBuffer*) evitam alocações frequentes, enquanto o *zero-padding* é aplicado de forma eficiente durante a cópia dos dados. Para uma IR de 3 segundos (132k amostras), essa estratégia limita o consumo de RAM a aproximadamente 2.1 MB, contra os 528 MB no domínio do tempo. A liberação de memória é tratada automaticamente pelos *destructors* dos *std::vector*, garantindo ausência de vazamentos.

```
// Alocação dos buffers principais
std::vector<float> m_irBuffer;           // Armazena a IR original
std::vector<float> m_fftInput;          // Buffer de entrada para FFT (com zero-padding)
std::vector<float> m_fftOutput;         // Buffer de saída da FFT (espectro complexo)
std::vector<std::vector<float>> m_partitions; // Armazena espectros FFT das partições

// Buffers circulares para processamento contínuo
std::vector<float> m_inputBuffer;       // Buffer circular de entrada
std::vector<float> m_outputBuffer;      // Buffer de saída com overlap-add

void preprocessIR() {
    // Configura buffers com tamanho otimizado (potência de 2)
    m_fftInput.resize(m_fftSize, 0.0f);
    m_fftOutput.resize(m_fftSize);
    m_partitions.resize(m_numPartitions);
}
```

```
// Processa cada partição
for (int i = 0; i < m_numPartitions; ++i) {
    std::fill(m_fftInput.begin(), m_fftInput.end(), 0.f); // Zero-padding
    std::copy_n(m_irBuffer.data() + i*m_partitionSize,
        std::min(m_partitionSize, (int)m_irBuffer.size()-i*m_partitionSize),
        m_fftInput.begin());

    m_fft->forward(m_fftInput.data(), m_fftOutput.data()); // FFT
    m_partitions[i] = m_fftOutput; // Armazena apenas o espectro
}
}
```

As otimizações da FFT são cruciais para desempenho em tempo real no processamento de áudio. A implementação utiliza bibliotecas altamente otimizadas como a FFTW, que emprega algoritmos adaptativos (como o Cooley-Tukey) para reduzir a complexidade computacional. Um aspecto crítico é o *planejamento prévio* (“*plan*”), onde a melhor estratégia de cálculo é determinada durante a inicialização, baseando-se no tamanho da transformada e na arquitetura do processador.

```
// Criação do plano FFT durante a inicialização
m_fft = std::make_unique<FFT>(m_fftSize);
m_fft->createPlan(); // Otimizações específicas para o tamanho

// Uso durante o processamento:
void processBlock() {
    m_fft->forward(m_fftInput.data(), m_fftOutput.data()); // Execução rápida}
}
```

Esta abordagem elimina sobrecarga de *runtime*, aproveitando instruções *Single Instruction Multiple Data* (SIMD) como AVX/SSE e cache de CPU. Para tamanhos comuns (ex.: 2048 pontos), o planejamento prévio pode acelerar a FFT em até 10×. O uso de potências de 2 (ex.: 1024, 2048) permite a utilização de algoritmos *radix-2*, mais eficientes que implementações genéricas.

No processamento por blocos, o uso de janelas é crucial para suavizar transições. A janela de Hann é preferida por equilibrar características espectrais e desempenho computacional. Ela atenua 18dB/oitava, zera as bordas do bloco e minimiza vazamentos espectrais e artefatos de “clique”. Comparada à Hamming (resíduos de 0,08%) e à Blackman (maior atenuação, mas mais custo computacional), a Hann oferece bom compromisso entre qualidade e leveza. Pré-computada, é multiplicada ponto a ponto com a saída da IFFT na etapa de *overlap-add*, garantindo soma unitária nas sobreposições e preservação espectral. A implementação computacionalmente leve (apenas

três operações por amostra) aliada à sua eficácia fazem da Hann a escolha padrão em sistemas de convolução por FFT para áudio.

```
// Pré-computação da janela de Hann
m_window.resize(m_fftSize);
for (int n = 0; n < m_fftSize; ++n) {
    m_window[n] = 0.5f * (1 - cos(2 * M_PI * n / (m_fftSize - 1)));
}
// Aplicação durante o overlap-add
for (int i = 0; i < m_fftSize; ++i) {
    m_outputBuffer[i] += m_ifftOutput[i] * m_window[i]; // Atenua bordas
}
```

No processamento em tempo real, é crucial equilibrar desempenho e latência para evitar *underruns* (quando a CPU não processa o áudio a tempo). O tamanho do bloco é o principal fator: blocos menores reduzem a latência, mas exigem mais da CPU; blocos maiores aliviam o processamento, mas aumentam a latência.

```
void next(float inSample) {
    m_inputBuffer[m_inputPos] = inSample;
    m_inputPos = (m_inputPos + 1) % m_blockSize;

    if (++m_samplesProcessed >= m_blockSize) {
        m_samplesProcessed = 0;
        processBlock(); // FFT → Multiplicação → IFFT
    }
}
```

Para evitar *underruns*, usam-se três técnicas: (1) pré-computação das FFT da resposta impulsiva, (2) *threads* dedicadas para processamento e (3) escalonamento dinâmico com prioridade para a *thread* de áudio. A latência depende do tamanho do bloco (ex: 1024 amostras a 44.1 kHz = 23.2 ms), sendo ≈ 50 ms o limite aceitável para aplicações interativas. Sistemas críticos, como VoIP, operam com blocos de 128–256 amostras (2.9–5.8 ms), exigindo otimizações como SIMD e *buffers lock-free*.

4.3 Integração com a interface de usuário

A interface gráfica foi projetada para facilitar o controle interativo do reverberador de convolução implementado, integrando-se com os aspectos técnicos discutidos. Ela permite selecionar sinais de entrada, ajustar parâmetros em tempo real e carregar IR acústicas pré-gravadas. A janela principal organiza os controles em seções lógicas:

- Seleção de entrada: um menu suspenso (*PopupMenu*) permite escolher entre fontes mono, estéreo ou entrada ao vivo (*Live Input*). O código abaixo verifica a seleção e direciona o sinal para os *buffers* apropriados (*~monoBus* ou *~stereoBus*);

```
• ~chooser = PopupMenu(~labelView, Rect(25, 20, 110, 20))  
• .items_([ "Mono", "Stereo", "Live Input" ])  
• .action_({ |menu| ~updateInputSource.value(menu.value) });
```

- Controles de reverberação: um *slider* exponencial (*~revAmpSlider*) ajusta a amplitude do efeito, mapeado para uma faixa de 0.001 a 1 (útil para evitar distorções em ganhos altos). Botões de *play/stop* (*~revButton*) ativam o sintetizador de reverberação (*Synth(\reverb)*), que utiliza *PartConv.ar* para aplicar a convolução com a IR carregada;
- Filtros de pós-processamento: dois *knobs* controlam filtros passa-alta (*~hiPassKnob*) e passa-baixa (*~loPassKnob*), aplicados ao sinal reverberado para moldar o timbre. Os valores são convertidos para escalas exponenciais (*ControlSpec*) para maior precisão em frequências extremas.

```
~hiPassKnob.action_({ |obj|  
  ~revSynthL.set(\loFreq, ~hiPassSpec.map(obj.value)) // Atualiza filtro em tempo real  
});
```

A interface inclui um sistema robusto para carregar IR a partir de um diretório especificado. O código verifica erros (como arquivos corrompidos ou diretórios inválidos) e pré-processa os *buffers* para convolução via FFT. O método *preparePartConv* aloca *buffers* com tamanhos otimizados (calculados por *PartConv.calcBufSize*), garantindo compatibilidade com o *~fftsize* definido.

```
~bufList.action_({ |i|  
  ~irSpectrumL.preparePartConv(~bufs[i.value * 2], ~fftsize); // Pré-cálculo do  
  espectro  
  ~irSpectrumR.preparePartConv(~bufs[i.value * 2 + 1], ~fftsize);  
});
```

Um *toggle* personalizado (*~toggle*) permite adicionar o sinal seco à mixagem, útil para comparações A/B. Quando ativado, um *synth* adicional (*\dryStereo* ou *\dryMono*) é criado.

```
~toggle.mouseDownAction_({  
  if(~togValue, { ~drySynthStereo = Synth(\dryStereo, [\in, ~stereoBus] ) });
```

```
});
```

Um *FreqScopeView* (*~fs*) exibe o espectro do sinal processado, enquanto mensagens de erro são exibidas no console para diagnóstico. *Sua integração modular com o núcleo do reverberador combina eficiência, baixa latência e usabilidade, servindo a fins criativos e educativos ao demonstrar na prática a convolução no domínio da frequência.*

Um aspecto essencial da integração é o gerenciamento dinâmico de *buffers* e a sincronização entre processos, garantindo robustez no uso em tempo real. O carregamento de IR é tratado com verificações de erro e pré-alocação dos *buffers* espectrais (*~irSpectrumL/R*), etapa fundamental para a convolução eficiente via FFT.

```
~bufList.action_({ |i|
  Routine({
    if(~bufs[i.value * 2].isNil.not) {
      ~irSpectrumL.preparePartConv(~bufs[i.value * 2], ~fftsize); // Pré-processa a IR
      s.sync; // Sincroniza com o servidor de áudio
      "IR carregada com sucesso (FFT: %)".format(~fftsize).postln;
    } { "Erro: Buffer inválido".postln };
  }).play(AppClock);
});
```

A lógica de ativação/desativação dos *synths* (como *~revSynthL* e *~pbmSynth*) através de botões (*~revButton*, *~playButton*) oferece controle preciso sobre os recursos do servidor, prevenindo vazamentos de memória e conflitos. Ao pressionar “PARAR”, os *synths* são liberados (*free*) e, ao reiniciar, são recriados com parâmetros atualizados, garantindo eficiência e integridade do processamento em tempo real, mesmo em sistemas com recursos limitados. O algoritmo abaixo detalha esse processo, enquanto a Figura 17 ilustra a interface do reverberador.

Algoritmo: Reverberador por Convolução com <i>Impulse Response</i> em tempo real
Entradas: - <i>audio_input</i>: sinal de áudio em tempo real (microfone ou outra fonte) - <i>ir_buffer</i>: buffer com a resposta ao impulso (carregada de arquivo externo) - <i>dryWet</i>: controle de mistura entre sinal original (dry) e processado (wet) - <i>gain</i>: fator de amplificação aplicado à saída - <i>roomSim</i>: controle espacial da reverberação

Saídas:

- *audio_output*: sinal convolvido com resposta ao impulso, com mixagem dry/wet

Inicialização:

1. Carregar IR de arquivo externo para *ir_buffer*
2. Inicializar os parâmetros de controle: *dryWet*, *gain*, *roomSim*
3. Criar estrutura de síntese com *PartConv* ou *Convolution2*:
 - a. Estabelecer tamanho da FFT baseado no comprimento do IR
 - b. Inicializar *delay* interno, se necessário
4. Criar interface gráfica (GUI) para controle em tempo real dos parâmetros

Processamento em tempo real:

Para cada bloco de áudio:

1. Capturar áudio de entrada: *input = AudioIn.ar(bus)*
2. Processar convolução: *convOutput = Convolution2.ar(input, ir_buffer, fftSize)*
3. Aplicar mixagem dry/wet:
 $wet = convOutput * dryWet$
 $dry = input * (1 - dryWet)$
 $output = (dry + wet) * gain$
4. Aplicar panoramização ou simulação de sala se *roomSim* estiver ativado
5. Enviar o sinal para os alto-falantes ou gravação

Interface Gráfica:

- Deslizadores para controle de *dryWet*, *gain*, *roomSim*
- Botões para carregar novos arquivos IR dinamicamente



Figura 17: Interface gráfica do reverberador de convolução.

Fonte: acervo do autor

4.5 Método de geração de IR via deconvolução de *sweeps* exponenciais

A geração de IR por deconvolução de *sweeps* exponenciais baseia-se no princípio de que a resposta ao impulso de um sistema linear invariante no tempo pode ser obtida pela divisão espectral entre o sinal gravado e o sinal de excitação original (FARINA, 2000), conforme descrito matematicamente por:

$$h(t) = \mathcal{F}^{-1}\left(\frac{Y(f)}{X(f)}\right)$$

onde $X(f)$ é a transformada de Fourier do *sweep* exponencial original, $Y(f)$ é a transformada do *sweep* gravado no ambiente e $h(t)$ é a resposta impulsiva estimada.

O código em Python implementa uma *pipeline* otimizada para deconvolução, com etapas críticas descritas abaixo:

- Pré-processamento dos sinais:
- Os sinais original e gravado são carregados e normalizados (*load_audio()*), garantindo mono e faixa dinâmica adequada com aplicação de *pre-emphasis* (filtro FIR) para compensar a atenuação de altas frequências.

```
def load_audio(file_path):  
    """Carrega áudio garantindo mono e normalizado"""  
    data, sample_rate = sf.read(file_path, dtype='float32') # Leitura do arquivo  
    if data.ndim > 1: # Conversão para mono se estéreo  
        data = np.mean(data, axis=1)  
    return data / np.max(np.abs(data)), sample_rate # Normalização para amplitude  
máxima = 1
```

- Na função *compute_ir()*, *pre_emphasis=0.5* (valor padrão) compensa-se a atenuação de altas frequências.

```
original = signal.lfilter([1, -pre_emphasis], [1], original) # Filtro FIR (Eq. 1)  
recorded = signal.lfilter([1, -pre_emphasis], [1], recorded) # Aplicado a ambos os  
sinais
```

- O *sweep* original é invertido no domínio espectral (*exponential_sweep_inversion*) e multiplicado por uma janela de *fade-out* para minimizar artefatos de borda.

```
def exponential_sweep_inversion(sweep, sr):  
    """Inversão espectral para sweep exponencial"""  
    t = np.arange(len(sweep)) / sr # Vetor de tempo  
    fade_out = np.logspace(0, -6, len(sweep)//10) # Fade-out exponencial (60 dB de  
atenuação)  
    envelope = np.concatenate([np.ones(len(sweep)-len(fade_out)), fade_out]) # Combina  
partes plana e fade  
    return sweep * envelope # Multiplica pelo sweep original
```

Essas etapas são acionadas em:

```
original, sr = load_audio(original_path) # Passo 1  
recorded, _ = load_audio(recorded_path) # Passo 1  
original_inv = exponential_sweep_inversion(original, sr) # Passo 3  
# Passo 2 ocorre dentro de compute_ir()
```

- Deconvolução no domínio da frequência:
- A divisão espectral é realizada com proteção contra divisão por zero (adicionando $\varepsilon = 10^{-10}$). Implementada na função correspondente `compute_ir()` o termo `np.max(np.abs(fft_original))` escala a proteção conforme a magnitude do espectro.

```
fft_original = np.fft.fft(original_inv) # Transformada do sweep invertido
fft_recorded = np.fft.fft(recorded)     # Transformada do sinal gravado

# Divisão protegida:
ir_freq = fft_recorded / (fft_original + EPSILON * np.max(np.abs(fft_original)))
```

- Imediatamente após a divisão espectral, na mesma função a suavização do espectro é operada com uma janela de Hanning para reduzir ruído. A convolução (`np.convolve`) com a janela age como um filtro passa-baixas no domínio da frequência e `mode='same'` mantém o comprimento original do sinal.

```
window = np.hanning(100) # Janela de Hanning de 100 pontos
window /= np.sum(window) # Normaliza a janela
ir_freq = np.convolve(ir_freq, window, mode='same') # Suavização
```

Essas etapas são partes críticas da função principal `compute_ir()`, que é chamada por:

```
ir = compute_ir(original, recorded, sr) # Gera a IR bruta
```

- Pós-processamento da IR:
- A IR é truncada com base na energia acumulada, removendo regiões de silêncio, encontrado na função `post_process_ir`.

```
energy = np.cumsum(ir**2) # Calcula energia acumulada
threshold = 0.0005 * energy[-1] # Threshold = 0.05% da energia total
start = np.argmax(energy > threshold) # Primeiro ponto acima do threshold
end = len(energy) - np.argmax(energy[::-1] > threshold) # Último ponto relevante
ir_cut = ir[start:end] # Corte das regiões de silêncio
```

- Na mesma função, logo após o corte, um janelamento é operado pela aplicação de *fades* de entrada/saída (janela de Hanning) para evitar *clicks*.

```
fade_samples = min(512, len(ir_cut)//4) # Define tamanho do fade (até 512 amostras)
fade_window = np.hanning(2 * fade_samples) # Janela de Hanning simétrica
ir_cut[:fade_samples] *= fade_window[:fade_samples] # Fade-in
ir_cut[-fade_samples:] *= fade_window[fade_samples:] # Fade-out
```

- Na última linha da função *post_process_ir*: computa-se a normalização final para amplitude máxima unitária.

```
return ir_cut / np.max(np.abs(ir_cut)) # Normaliza para amplitude máxima = 1
```

Essas etapas são acionadas durante a chamada principal:

```
ir = compute_ir(original, recorded, sr) # Deconvolução bruta  
ir_processed = post_process_ir(ir, sr) # Aplica corte, janelamento e normalização
```

- Análise e validação:
- A primeira, realizada visualmente, é implementada em *plot_ir()*, que exibe a IR nos domínios temporal (em segundos) e frequencial (escala logarítmica no eixo X e magnitude em decibéis no Y, com faixa dinâmica de 80 dB). A segunda, realizada quantitativamente, calcula a relação entre sinal e ruído (SNR) por intermédio da conversão $20 \cdot \log_{10}(\text{np.max}(\text{np.abs}(\text{ir_processed}))/\text{np.std}(\text{ir_processed}[:1000]))$, comparando o pico do sinal útil com o ruído na região de pré-chegada. A relação sinal-ruído (SNR) é exibida no terminal antes do salvamento final (via *generate_ir()*), fornecendo uma medida objetiva da qualidade da IR, enquanto os gráficos complementam com uma análise visual detalhada.

```
# Na função plot_ir():  
magnitude = 20 * np.log10(np.abs(np.fft.rfft(ir)) + EPSILON) # Conversão para dB  
plt.semilogx(freqs, magnitude) # Gráfico logarítmico  
  
# Na função generate_ir():  
print(f"SNR                                     estimado:  
{20*np.log10(np.max(np.abs(ir_processed))/np.std(ir_processed[:1000])):.1f} dB")
```

O método proposto de geração de IR combina robustez e eficiência. A resistência ao ruído decorre da inversão espectral com termo de regularização (*epsilon*) e da suavização com janela de Hanning no domínio da frequência, minimizando distorções não lineares e artefatos. A eficiência computacional resulta do uso de operações vetorizadas com *NumPy* e FFT, viabilizando o processamento em tempo real mesmo para IR de até 10 segundos. Além disso, o método é adaptável: parâmetros como *pre_emphasis* e critérios de corte baseados em energia (ex.: *threshold* = $0.0005 \times \text{energy}[-1]$) podem ser ajustados conforme o ambiente acústico.

Contudo, o método apresenta limitações. A sincronização entre os sinais original e gravado é crítica, tratada no código por corte automático ao menor comprimento comum ($min_len = \min(len(original), len(recorded))$), exigindo cuidado na captura. A resolução em baixas frequências também é limitada pela duração do *sweep* — por exemplo, um *sweep* de 10 segundos permite análise até ~ 0.1 Hz, conforme o princípio de incerteza tempo-frequência. Essas limitações devem ser consideradas em medições que exigem alta precisão em baixas frequências ou em ambientes acusticamente complexos.

O método de geração de IR via deconvolução de *sweeps* exponenciais oferece vantagens marcantes em relação a abordagens tradicionais. Diferente da excitação por impulso direto ou sequências pseudoaleatórias, o *sweep* exponencial distribui a energia de forma contínua ao longo do tempo, evitando picos que poderiam gerar distorções e garantindo uma excitação uniforme em todas as faixas de frequência. A deconvolução isola eficientemente a resposta linear do sistema, removendo distorções harmônicas e separando componentes não lineares com precisão. A técnica também permite capturar IR longas com alta resolução temporal, sem perda de qualidade, além de ser computacionalmente eficiente, dispensando médias repetidas como no método *Maximum Length Sequence* (MLS).

4.6 Aplicação: virtualização de espaços acústicos em cerâmica

A cerâmica, enquanto matéria e ofício, retém em sua estrutura uma memória acústica — uma assinatura vibratória inscrita pela relação entre argila, fogo e forma. Quando uma ocarina é soprada, seu corpo oco não apenas projeta notas, mas revela, em ressonâncias, histórias de sua manufatura: espessura, porosidade, curvatura. Esta aplicação propõe uma arqueologia invertida: em vez de escavar fragmentos, extraímos espectros — virtualizando os espaços internos do “bum d’água” (tambor de cerâmica similar ao Udu nigeriano), através de IR geradas por *sweeps* exponenciais, para então recompor esses ambientes como paisagens sonoras habitáveis. A Figura 16 ilustra o “bum d’água”.



Figura 18: vistas em perspectivas múltiplas do “bum d’água” manufaturado por Mestre Nado cujas dimensões e peso podem variar de 20 x 27 x 20 cm, 10,8 litros e 1,350 Kg ao dobro com 21,6 litros. Classificado no sistema Hornbostel-Sachs como 412.13 – aerofone de vaso percutido (vessel flute without duct, struck), pois seu som é produzido pela vibração do ar no interior do corpo ressonante, ativado pelo impacto manual na abertura lateral.

Fonte: acervo do autor.

O método parte de um princípio físico: todo corpo cerâmico é um ressonador passivo cuja resposta em frequência revela sua forma. Ao capturar IRs dentro do “bum d’água”, mapeamos não só reflexões, mas a textura acústica da cerâmica — o sussurro de moléculas sinterizadas, o eco de fissuras. Virtualizar o espaço torna-se reanimar matéria. As IRs, mais que filtros, tornam-se espectros. A “memória acústica” não é metáfora, mas dado: a transformada de Fourier de um artefato cerâmico expõe, em picos e vales, o gesto do oleiro.

A técnica de geração de IR por deconvolução de *sweeps* exponenciais foi aplicada para virtualizar o ambiente acústico do “bum d’água” e criar reverberações realistas para ocarinas em tempo real. O processo inicia com a síntese de um *sweep* exponencial, utilizando um gerador otimizado (ilustrado na Figura 19) que garante excitação espectral uniforme e controle preciso de parâmetros como frequência inicial (ex.: 20 Hz), final (ex.: 20 kHz), duração (ex.: 23.6 s) e amplitude (ex.: 0.5). O código abaixo destaca a síntese do *sweep*, que emprega uma fase logarítmica: e um envelope suave para evitar transientes abruptos.

```
SynthDef(\cleanSweep, {  
  var t = Line.ar(0, safeDur, safeDur, doneAction: 2);  
  var phase = 2pi * safeStart * safeDur / ratio.log * (ratio.pow(t/safeDur) - 1);  
  var sig = SinOsc.ar(phase, 0, ~params.amp);  
  var env = Env([0,1,1,0], [0.05, safeDur-0.1, 0.05]).ar(2);  
  Out.ar(0, sig ! 2 * env); // Saída estéreo (ajustável)  
}).add;
```

Do Barro ao Berro: Cerâmica, Algoritmos, Gambiarras e Composição Colaborativa em “Hybris: Ocarinas do Mestre Nado”

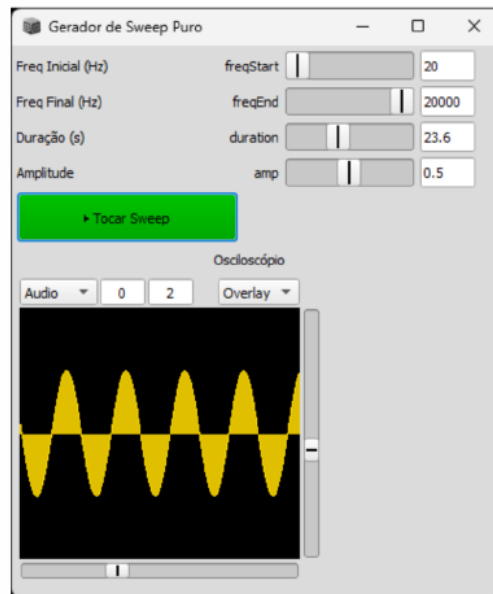


Figura 19: Interface do gerador de sweep exponencial.

Fonte: acervo do autor.

O sweep é reproduzido em um orifício do “bum d’água” por um sistema Output Capacitor-Less (OCL) customizado (500 W RMS, THD < 0.1% a 1 kHz), com filtragem ativa baseada no CI 4558 para garantir planicidade espectral (± 0.5 dB) e supressão de aliasing. A captura é feita pelo segundo orifício, utilizando transdutores customizados montados em falantes de pequeno diafragma e circuitos “op-amp/hex inverter”. A Figura 20 ilustra esse processo.

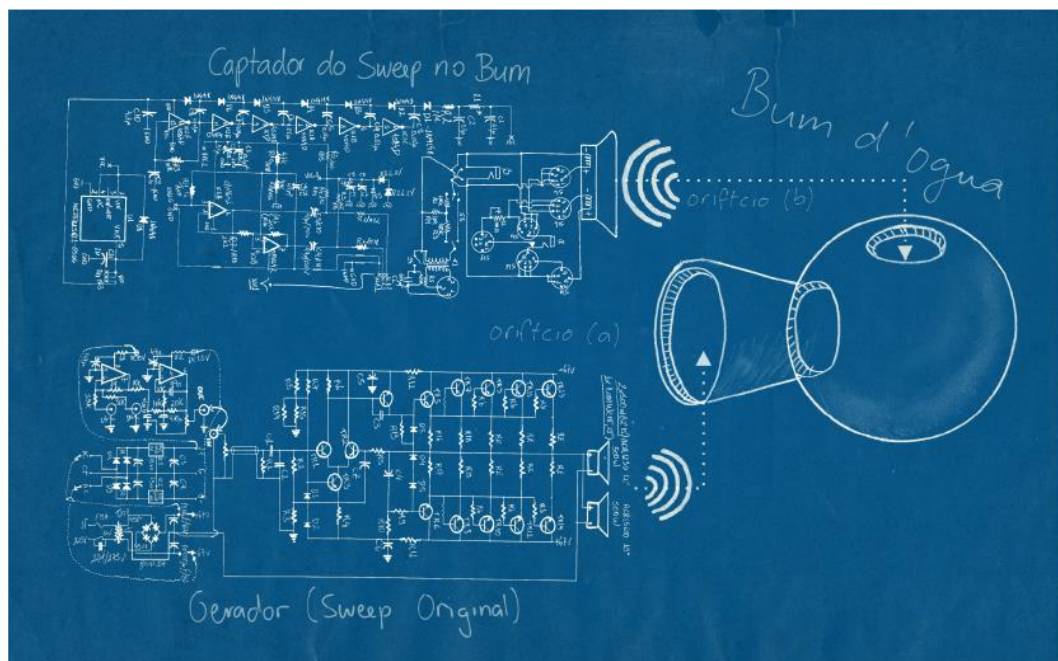


Figura 20: Instrumentação do processo de captura realizado no bum d’água.

Fonte: acervo do autor

No cianótipo do sistema está expresso o “capitador do sweep no bum”, o reproduutor ou “gerador do sweep original” e seus posicionamentos nos orifícios (a, b) do “bum d’água”. O “capitador do sweep no bum” (orifício b) verte-se em um transdutor customizado com uma etapa de alimentação regulada baseada em um regulador (NCV4274) e diodos de proteção (1N4148), responsável por fornecer tensão estável aos demais blocos. A seção de pré-amplificação utiliza amplificadores operacionais (TL074 ou LM358) configurados para ganho e filtragem do sinal captado. O circuito também incorpora retificação, estabilização de tensão com zener de 6.2V e proteção contra sobretensões. A saída é balanceada, com transformadores de isolamento conectados a conectores XLR de 5 pinos. Além disso, há circuitos de controle com interruptores, relés e transistores, destinados a funções como chaveamento de alimentação, corte de graves (low cut) ou atenuação (pad). Outras adaptações com pequenos diafragmas (frequência de ressonância > 25 kHz), acoplados a um circuito de condicionamento com estágio *op-amp* (ganho de 40 dB, impedância de entrada > 10 MΩ) e inversores hexadecimais para compensação de fase foram testados. O reproduutor ou “gerador do sweep original” (orifício a) é montado em amplificador caseiro de alta potência com topologia OCL (Output Capacitor-Less), capaz de entregar 500 watts RMS, sendo alimentado por uma fonte simétrica de $\pm 67V$. O projeto inclui um estágio de pré-amplificação com transistores diferenciais e uma saída push-pull com transistores de potência complementares, garantindo eficiência e fidelidade na reprodução sonora. A seção de entrada conta com um módulo de filtro ativo baseado no CI 4558, responsável por modelar a resposta em frequência do sistema. Além disso, a fonte de alimentação robusta utiliza retificação de alta corrente, filtragem com capacitores de 10.000 μF e proteção contra surtos.

O registro das características acústicas do espaço no “bum d’água” (reflexões, ressonâncias e reverberação) sofre a deconvolução (entre o *sweep* original e o gravado) implementada em Python, isolando a IR do ambiente através de divisão espectral protegida contra ruído ($\text{fft_recorded} / (\text{fft_original} + \text{EPSILON})$) e pós-processamento com corte baseado em energia e janelamento (Hanning) como discutido. A IR resultante foi integrada ao reverberador de convolução via FFT descrito, que utiliza o método *overlap-add* para aplicar a reverberação em tempo real com baixa latência. A Figura 21 compara ambos os *sweeps* e apresenta a IR resultante do processo.

Do Barro ao Berro: Cerâmica, Algoritmos, Gambiarras e Composição Colaborativa em “Hybris: Ocarinas do Mestre Nado”

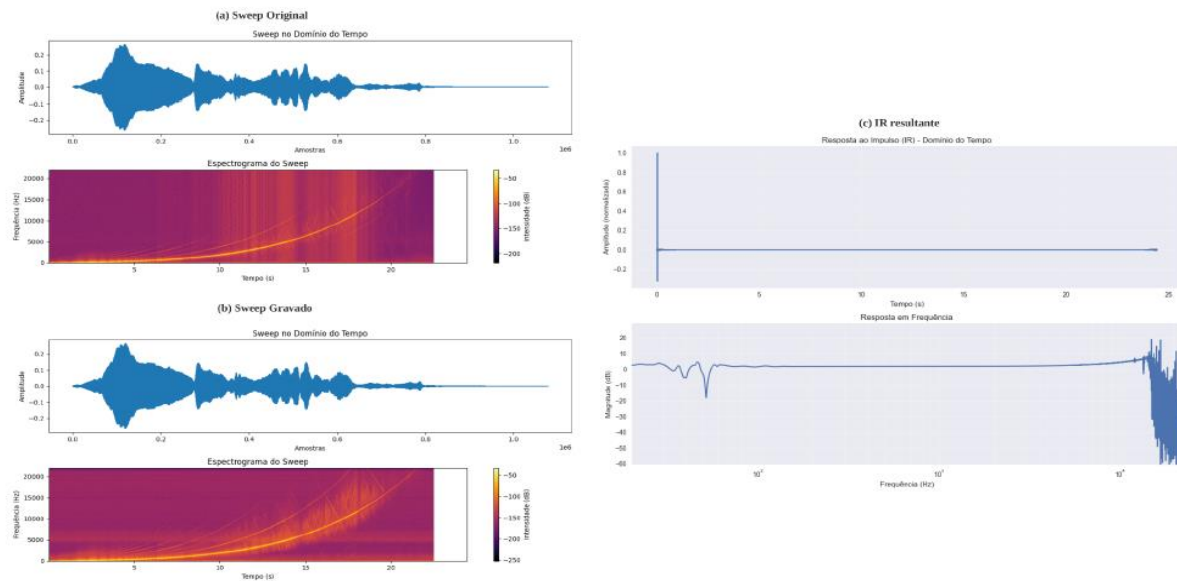


Figura 21: “sweep original” (a) gerado pela aplicação “gerador de sweep puro” com as frequências inicial de 20 Hz, final de 20 kHz, duração de 23.6 segundos e 0.5 de amplitude fixada reproduzido por altifalantes montados em um amplificador caseiro de topologia OCL (500watt RMS + módulo de filtro sob CI 4558);

“sweep gravado” (b) no interior do “bum d’água” (25 x 34 x 25 cm) por transdutores montados em pequenos diafragmas e alguns sob circuito “op-amp/hex inverter”; “IR resultante” (c) pela deconvolução implementada em

Na primeira representação (a) do “sweep original”, a varredura exponencial é ideal para excitação de sistemas acústicos, pois proporciona uma excitação equilibrada por oitava, favorecendo uma análise precisa da resposta em frequência do sistema. Na segunda representação (b), “sweep gravado”, o mesmo padrão de varredura está presente, mas são evidentes distorções harmônicas e não linearidades, visíveis como múltiplas trilhas paralelas no espectrograma. Essas distorções indicam a presença de componentes não lineares introduzidos pelo caminho eletroacústico: o amplificador OCL de alta potência e o módulo de filtro ativo com CI 4558, além da possível coloração acústica imposta pela ressonância do “bum d’água” utilizado como caixa de ressonância. Adicionalmente, há um aumento expressivo da densidade espectral em certas bandas médias e agudas, sugerindo realce ou acoplamentos ressonantes nessas regiões. A terceira representação (c) apresenta a Resposta ao Impulso (IR) resultante da deconvolução entre o “sweep original” e o “sweep gravado”, já pós-processada com janelamento e normalização. No domínio do tempo, a IR mostra um pico claro seguido de um decaimento rápido, indicando boa resolução temporal e baixa persistência reverberante — o que é coerente com um ambiente de captação altamente amortecido (como a câmara globular cerâmica do “bum d’água”). No domínio da frequência, a curva apresenta relativa linearidade até cerca de 10 kHz, com pequenas oscilações e um notável *notch* em torno de 60–70 Hz, possivelmente relacionado à ressonância própria do

invólucro cerâmico ou a efeitos de cancelamento na posição do microfone. Acima de 10 kHz, observa-se queda abrupta da magnitude, consistente com limitações dos transdutores e do sistema de gravação caseiro. A razão sinal-ruído (SNR) estimada em mais de 20 dB reforça a eficiência do processo de deconvolução, considerando a natureza artesanal do sistema.

5. Escutar com os olhos e ver com os ouvidos: a obra como ressonância

O ciclo de partilha e investigação culmina na publicação fonográfica “*Hybris: Ocarinas do Mestre Nado*” (VAZ; MEDEIROS; ALENCAR; CAMPELLO, 2024), lançada em 2024 pelo selo independente Seminal Records. Mais que um registro documental, o álbum condensa simbolicamente o percurso: escuta expandida, gesto gráfico transmutado em som, espacialização convolutiva dos corpos cerâmicos e notação como cartografia do excesso. O que se ouve é o sopro ancestral do barro, o espectro algorítmico do código, o eco trágico do herói e o gesto pedagógico que ensina a escutar o invisível. A Figura 20 ilustra a capa e contracapa do álbum.



Figura 22: capa e contracapa do álbum *Hybris: Ocarinas do Mestre Nado* (2024).

Fonte: acervo do autor.

Cada faixa do álbum é um ensaio de fronteira, articulando curvas gráficas, estruturas harmônicas e espacializações baseadas em respostas impulsivas cerâmicas — todas derivadas da presença ritual de Mestre Nado, Patrimônio Vivo de Pernambuco. O som das ocarinas foi inscrito como forma e espaço, multiplicado pelo desenho e pela escuta.

Lançado pela Seminal Records — selo voltado ao experimentalismo e às poéticas do sul global — o álbum reafirma o compromisso da pesquisa com práticas editoriais fora do eixo, livres de lógicas mercadológicas. Publicar aqui é também um gesto político: devolver à escuta sua potência crítica, criativa e coletiva.

A obra responde de forma profunda à chamada do dossiê “Escrever, escutar, compor”, ao não apenas tensionar os limites entre escrita e notação, mas propor um ciclo em que compor é escutar, escutar é escrever, e escrever é modo de estar no mundo. Neste álbum, o som não representa: encarna. Curva, estilhaça, reconstrói, inscreve. É intervalo e borda. É barro vibrante. Escutar *Hybris* é atravessar — e talvez, começar de novo.

Todo o percurso — do ateliê ao estúdio, do estúdio ao laboratório — forma uma cadeia estética e epistêmica onde o artesanal, o composicional e o tecnológico se contaminam mutuamente. *Hybris* é mais que um álbum: é um objeto de pensamento construído pela escuta, fruto de uma invenção partilhada entre tradição e experimentação.



Figura 23: *A Forma do Som I, II & II, série de esculturas do Mestre Nado, 2024.*

Fonte: adaptadas e modificadas com base em três fotografias do Estúdio em Obra

REFERÊNCIAS BIBLIOGRÁFICAS

BJÖRGVINSSON, Erling; EHN, Pelle; HILLGREN, Per-Anders. *Design things and design thinking: Contemporary participatory design challenges. Design Issues*, Cambridge, v. 28, n. 3, p. 101–116, 2012.

BRANDÃO, Junito de Souza. *Mitologia grega: o nascimento dos deuses*. 24. ed. Petrópolis: Vozes, 2016. v. 1.

CASTELÕES, L. *Musicalizing Sonification: Image-to-music Conversion using OpenMusic*. In: BRESSON, J.; AGON, C.; ASSAYAG, G. (Org.). *OM Composer's Book*. v. 3. 1. ed. Paris: Éditions Delatour, 2016. p. 1-16.

CLARK, Andy. *Being there: putting brain, body, and world together again*. Cambridge: MIT Press, 2001.

COLLINS, Nick. PartitionedConvolution.cpp: implementation of partitioned convolution for real-time audio processing. SuperCollider, GitHub, out. 2008. Disponível em: <<https://github.com/supercollider/supercollider/blob/develop/server/plugins/PartitionedConvolution.cpp>>. Acesso em: 5 abr. 2025.

COOK, Perry R. *Physically informed sonic modeling (PhISM): Synthesis of percussive sounds*. Computer Music Journal, v. 21, n. 3, p. 38–49, 1997.

COOLEY, J. W., & TUKEY, J. W. *An algorithm for the machine calculation of complex Fourier series*. Mathematics of Computation, 19(90), 297-301. 1965. <https://doi.org/10.1090/S0025-5718-1965-0178586-1>

DARREG, Ivor. *Xenharmonic Bulletin*, n. 1. San Diego: Xenharmonicon, 1974

DEL NUNZIO, Mário Augusto Ossent. *Fisicalidade: potências e limites da relação entre corpo e instrumento em práticas musicais atuais*. 2011. 211 f. Dissertação (Mestrado em Música) – Escola de Comunicação e Artes, Universidade de São Paulo, São Paulo, 2011.

ERNST, Wolfgang. *Digital Memory and the Archive*. University of Minnesota Press, Minneapolis, 2015.

ERNST, Wolfgang. *Sobre a “Sonicidade”*. Tradução de Eduardo Harry Luersen. Revista GEMInIS, São Carlos, UFSCar, v. 10, n. 1, pp.4-26, jan. / abr. 2019

FARINA, A. *Simultaneous Measurement of Impulse Response and Distortion with a Swept-Sine Technique*. AES. 2000.

FERNANDES, Isabela. *Hybris: o herói trágico e a poética da desmedida*. 2023.

FLUSSER, Vilém. *Filosofia da caixa preta: ensaios para uma futura filosofia da fotografia*. São Paulo: Annablume, 2002.

FLUSSER, Vilém. *O mundo codificado: por uma filosofia do design e da comunicação*. São Paulo: Ubu Editora, 2007.

FOKKER, Adriaan. *Just Intonation and the 31-tone System*. In: *Selected Writings*. Amsterdam: Fokker Foundation, 1955.

FREIRE, Paulo. *Pedagogia do oprimido*. 17. ed. Rio de Janeiro: Paz e Terra, 1970.

GATO, Gonçalo Alves Gato Lopes. *Algorithm and Decision in Musical Composition*. Tese (Doutorado em Musical Composition) – Guildhall School of Music and Drama, City University London, Londres, 2016.

HUTCHINS, Edwin. *Cognition in the Wild*. Cambridge: MIT Press, 1995.

JOHNSTON, Ben. *Maximum Clarity and Other Writings on Music*. Urbana: University of Illinois Press, 2006.

KEENAN, Dave; SECOR, George D. *The Sagittal Microtonal Notation System, 2025*. Disponível em: <https://sagittal.org>. Acesso em: 14 abr. 2025.

LATOUR, Bruno. *Jamais fomos modernos: ensaio de antropologia simétrica*. Trad. Carlos Irineu da Costa. Rio de Janeiro: Editora 34, 1994.

LAZIER, Aaron; COOK, Perry R. *MOSIEVIUS: Feature-driven interactive audio mosaicing*. In: *Proceedings of the International Computer Music Conference (ICMC)*. 2003.

MALASPINA, Cécile. *An Epistemology of Noise*. London: Bloomsbury Academic, 2021.

MATHEWS, Max V.; PIERCE, John R. *The Bohlen–Pierce Scale*. In: *Journal of the Acoustical Society of America*, v. 84, n. 4, 1988, p. 1214.

McLEAN, Alex. *TidalCycles*. Projeto de live coding musical. 2025. Disponível em: <https://tidalcycles.org>. Acesso em: 13 abr. 2025.

NOË, Alva. *Action in perception*. Cambridge, MA: MIT Press, 2004.

OPPENHEIM, Alan V.; SCHAFER, Ronald W. *Discrete-Time Signal Processing*. 3. ed. Upper Saddle River, NJ: Prentice Hall, 2009.

PARIKKA, Jussi. *A Geology of Media*. Minneapolis: University of Minnesota Press, 2015.

PEREIRA, Marcus Vinícius Medeiros. *Ensino superior e as licenciaturas em música (pós Diretrizes Curriculares Nacionais 2004): um retrato do habitus conservatorial nos documentos curriculares*. 2012. 281 f. Tese (Doutorado em Educação) – Programa de Pós-Graduação em Educação, Universidade Federal de Mato Grosso do Sul, Campo Grande, 2012.

SABAT, Marc. *The Extended Helmholtz–Ellis II Pitch Notation*. [online], 2004. Disponível em: <http://www.marcsabat.com/pdfs/notation.pdf>. Acesso em: 14 abr. 2025.

SCAVONE, Gary P.; COOK, Perry R. *Real-time physical models for flute and clarinet*. In: *Proceedings of the International Computer Music Conference (ICMC)*, Beijing, China, 1999. Cf.

SCAVONE, Gary P. *An acoustic analysis of single-reed woodwind instruments with an emphasis on design and performance issues and digital waveguide modeling techniques*. 2003. Thesis (Ph.D.)—Stanford University, Stanford, CA, 2003.

SIMONDON, Gilbert. *Du mode d’existence des objets techniques*. Paris: Aubier, 1958.

SMITH, Steven. *Digital Signal Processing: A Practical Guide for Engineers and Scientists*. Newnes, 2013.

STIEGLER, Bernard. *Technics and Time 1: The Fault of Epimetheus*. Stanford: Stanford University Press, 1998. (traduzido do original francês *La technique et le temps 1: La faute d’Épiméthée*, 1994).

STOWELL, Dan; PLUMBLEY, Mark. *A framework for the provision of contextual metadata for concatenative sound synthesis*. Organised Sound, v. 12, n. 3, p. 229–238, 2007.

SCHAEFFER, Bogusław. *Topofonica*. [S.l.]: [s.n.], 1960. Composição orquestral para 40 instrumentos.

SCHWARZ, Diemo. *Concatenative sound synthesis: The early years*. Journal of New Music Research, v. 35, n. 1, p. 3–22, 2006.

VARELA, Francisco J.; THOMPSON, Evan; ROSCH, Eleanor. *The embodied mind: cognitive science and human experience*. Cambridge, MA: MIT Press, 1991.

VAZ, Henrique; MEDEIROS, Hugo; ALENCAR, Lucas; CAMPELLO, Marcelo. *Hybris: Ocarinas do Mestre Nado*. Recife: Seminal Records, 2024. Álbum digital. Disponível em: <https://seminalrecords.bandcamp.com/album/hybris-ocarinas-do-mestre-nado>

VERNANT, Jean-Pierre. *Mito e pensamento entre os gregos: estudos de psicologia histórica*. Trad. Haakon Chevalier. 3. ed. Rio de Janeiro: Paz e Terra, 2002.

XENAKIS, Iannis. *Formalized Music: Thought and Mathematics in Composition*. Revised edition. Stuyvesant, NY: Pendragon Press, 1992.

XENHARMONIC WIKI. *Xenharmonic Series*, 2024. Disponível em: https://en.xen.wiki/w/Xenharmonic_series. Acesso em: 10 abr. 2025.

WISHART, Trevor. *On Sonic Art*. Amsterdam: Harwood Academic Publishers, 1996.