Termpot: criação, edição e execução de funções no navegador em tempo de execução .

Guilherme Lunhani[1] **, Flávio Luiz Schiavoni**[2]

[1]Instituto de Artes e Design – Universidade Federal de Juiz de Fora
Juiz de Fora, MG

gcravista@gmail.com, ttonmeister@gmail.com

[2]Departamento de Computação – Universidade Federal de São João Del Rei
São João Del Rei, MG

fls@ufsj.edu.br

Abstract. *Researchs on sound synthesis using Internet browsers aren't new, but a group of developers used algebraic representations of waves as a online compositional strategy. Javascript language is used to process in a single data vector. With the research on this approach, we have developed a software with a different compositional viewpoint, using synthesis process. We'll present motivations, current, technical details, problems and future plans.*

Resumo. *Pesquisas sobre síntese sonora utilizando navegadores de internet não são novas, mas um grupo de desenvolvedores, Wavepot.com, utilizou representações algébricas de ondas como estrategia composicional online. Em linguagem javascript é processado um unico vetor de dados. Com a pesquisa desta abordagem, desenvolvemos um software com um ponto de vista da composicional diferente, utilizando o processo de síntese. Apresentaremos motivações, detalhes técnicos, problemas e planos futuros.*

1. Introdução

A *Web Audio API* [W3C 2012] possibilitou o processamento de sinais digitais de áudio em navegadores de rede, como por exemplo o Mozzilla Firefoz, Google Chrome ou Apple Safari [Roberts et al. 2013, Wyse and Subramanian 2014]. Esta API traz diversos *nós de áudio* padrões, que podem ser concatenados em um grafo de DSP. Um exemplo desta concatenação é demonstrado na Figura 1. São utilizados três instâncias de nós diferentes, em sequência, para construir um simples sintetizador. Um oscilador de ondas (OscilatorNode), ganho (*GainNode*), e alto-falantes (*DestinationNode*).

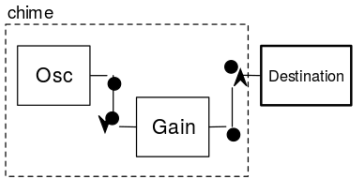

Figura 1: Estrutura de síntese da API webaudio. Fonte: [Srikumar 2013].

Um sintetizador mais complexo pode ter diversos nós concatenados, entre eles um nó "especial" chamado *ScriptProcessorNode*. Uma instância deste nó permite o cálculo

customizado de cada *sample*. Baseado neste nó, alguns *frameworks* são eles mesmos os instrumentos musicais. Por exemplo, *Gibber* [1]; ou *Wavepot* [2].

Neste sentido, nossa motivação foi desenvolver um *software* de luteria composicional, *Termpot*, como uma experiência técnico-estética de uma pesquisa mais ampla. Esta pesquisa envolve a prática *livecoding* [Collins 2014]. O resultado não é o objetivo da pesquisa, mas envolveu uma proposta histórica de Max Mathews, [Mathews and Moore 1970], GROOVE, pouco discutida pela comunidade acadêmica. Esta abordagem foi fundamental para a concepção do que é *livecoding*, bem como a estrutura do *software* desenvolvido.

2. Termpot

O Termpot (ver Figura 2) é um ambiente para improvisação de luteria composicional [Iazzetta 2009, Soares 2015], mais especificamente, um ambiente *web* de *livecoding*. Adapta o ambiente *Wavepot* (apresentado na nota-de-rodapé 2 da Seção 1) para um emulador de terminal, controlado pela linguagem *coffeescript*[Burnham 2011][3]. O ambiente possibilita a improvisação de códigos com a extensão média de uma linha (ver Código 1).

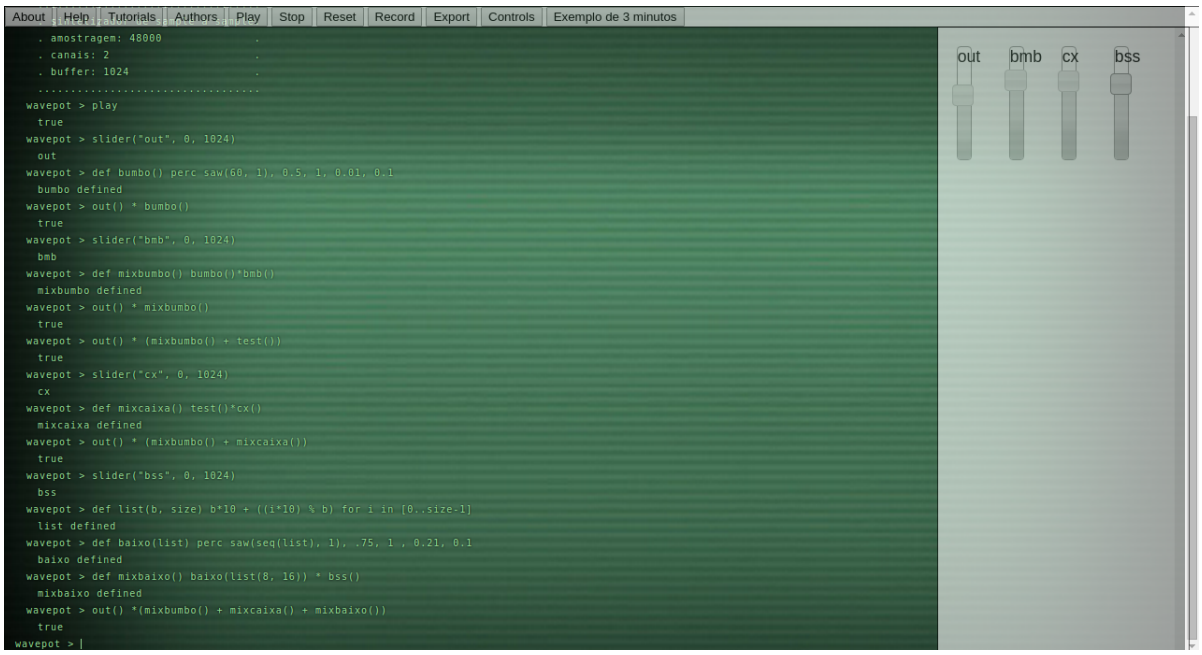

Figura 2: Aplicativo *Termpot*. Fonte: autores.

Em tese, é possível criar sonoridades semelhantes ao *Wavepot*. Porém, dado o curto tempo de desenvolvimento, pré-definimos apenas ruído (branco), senóide, dente-de-serra, triangular, pulso, envelope e sequenciamento.

É possível prototipar rapidamente novas funções para encapsular novas funcionalidade sonoras ao ambiente.

Além da atividade de síntese, elaboramos uma maneira para a criar e utilizar GUIs de controle (*sliders*), como uma mesa de som (ver Código 2

Uma característica singular do *Wavepot* original, é a possibilidade de separação da programação-partitura em dois arquivos, muito semelhante ao método *Instrumento-Orquestra* descrito por Max Mathews e utilizado no CSound [Mathews 1963,

[1]disponível em `http://gibber.mat.ucsb.edu/`)

[2]Disponível em `http://www.wavepot.com`)

[3]`http://coffeescript.org/`, acessado em 4 de novembro de 2015.

```
.....................................
. Virtual machine started at            .
. Thu Sep 03 2015 13:32:15 GMT-0300 (BRT).
. type help for instructions            .
.....................................
$\$$ |
$\$$ wavepot 1024
..............................
. sintetizador de sample a sample.
. amostragem: 44100             .
. canais: 2                     .
. buffer: 1024                  .
..............................
wavepot > play
wavepot > 0.71 * sin 440, sin(330, sin(220, 0.5)
true
```

Código 1: Console do *termpot* aguardando dados de entrada do improvisador; o improvisador inicia o ambiente wavepot com um buffer de 1024 pontos flutuantes; o improvisador inicia o processamento de áudio; o improvisador define o processamento de áudio de uma cascata de Modulção de amplitude.

```
wavepot > def tresAM(f, f1, f2, a) a * sin f, sin(f1, sin(f2, 0.5)
true
wavepot > inspect tresAM
var tresAM;

tresAM = function(f, f1, f2, a) {
  return a * sin(f, sin(f1, sin(f2, 0.5)))
};
wavepot > tresAM 440, 330, 220, 0.71
```

Di Nunzio 2010]. Isso é possível adicionando um marcador $@module$ aos comentários iniciais de um código. Desta forma, serão reconhecidos dois arquvivos durante a performance de improvisação: $index.js$ e $test.js$. O primeiro permite elaborar os instrumentos, enquanto o segundo realiza o DSP (teste).

Já no Termpot, buscamos utilizar outro método de codificação focado em uma abordagem mais performática.

Arriscamos a comentar uma inspiração no GROOVE de [Mathews and Moore 1970, Nunzio 2010], quando este propõe a criação de novas funções em tempo de execução. Ao mesmo tempo em que utilizamos a biblioteca *Ptty.js* dando ao *Termpot* as características de um emulador de terminal no que tange a utilização de comandos em tempo de execução, como em um terminal, integrado com controles manuais. Por esta razão, acreditamos que esta ferramenta é inspirada no conceito de compor, memorizar, editar e controlar funções do tempo, algoritmicamente e manualmente [Mathews and Moore 1970].

Referências

Burnham, T. (2011). Coffeescript: Accelerated javascript development.

Collins, N. (2014). Origins of live coding.

Di Nunzio, A. (2010). Genesi, sviluppo e difusione del software ”MUSIC n”nella storia della composizione informatica.

Iazzetta, F. (2009). *Música e mediação tecnológica*. Ed. Perspectiva-Fapesp.

Mathews, M. V. (1963). The digital computer as a musical instrument. *Science*, 142(3592):553–557.

```
wavepot > slider("left", 0, 1024)
true
wavepot > slider("right", 0, 1024)
true
wavepot > stereo sin(440,0.5)*left(), sin(330, 0.5)*right()
true
```

Código 2: Exemplo de código do Wavepot

Mathews, M. V. and Moore, F. (1970). GROOVE a program to compose, store, and edit functions of time. *Bell Telephones Laboratories*, page 7.

Nunzio, A. d. (2010). Groove.

Roberts, C., Wakefield, G., and Wright, M. (2013). The web browser as synthesizer and interface. page 6.

Soares, G. R. (2015). Luteria composicional de algoritmos pós-tonais v1.1final.

Srikumar, S. (2013). Tamming the script processor node.

W3C (2012). Web audio API.

Wyse, L. and Subramanian, S. (2014). The viability of the web browser as a computer music platform. *Computer Music Journal*, 37(4):10–23.