

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

Guilherme Lunhani¹ , Flávio Luiz Schiavoni²

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

gcravista@gmail.com

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

fls@uufs.j.ou.br

Abstract. *This panel reports the development of a sound synthesis program, Termpot. The Web Audio API technology was used in a reinterpretation of a [Mathews and Moore 1970]’s work, GROOVE. The resulting work is still in its infancy, but allows create and edit sound functions at runtime, in a web browser.*

Keywords: GROOVE; Web Audio API; DSP.

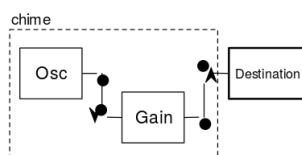
Resumo. *Este painel reporta o desenvolvimento de um programa de síntese sonora, Termpot. A tecnologia Web Audio API foi usada em uma reinterpretação de um trabalho de [Mathews and Moore 1970], GROOVE. O trabalho resultante ainda é incipiente, mas possibilita criar e editar funções no tempo de execução, em um navegador web.*

Palavras-chave: GROOVE; Web Audio API; DSP.

1. Introdução

A síntese sonora em *web browsers* é sumarizada por [W3C 2012, Roberts et al. 2013, Wyse and Subramanian 2014]. [Srikumar 2013] exemplifica uma concatenação de *nós de áudio* em um grafo de DSP. Três instâncias diferentes, *OscillatorNode*, *GainNode*, *DestinationNode*, são apresentadas na figura . Existe um outro nó, *ScriptProcessorNode* (figura 2) que possibilita customizações. Por exemplo, pode ser usado para externalizar para um(a) improvisador(a) suas próprias customizações de *funções de transferência* [Mathews and Moore 1970]. Esta abordagem foi utilizada no desenvolvimento do *Termpot*.

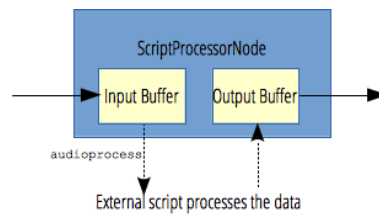
Figura 1: Estrutura básica de um sintetizador webaudio. **Fonte:** [Srikumar 2013]



2. Trabalho relacionado: GROOVE

Este artigo também envolve uma interpretação do GROOVE, *Generated Real-time Operations On Voltage-controlled Equipment*, de [Mathews and Moore 1970]. A

Figura 2: “A interface do *ScriptProcessorNode* permite a geração, processamento ou análise de áudio usando JavaScript”. **Fonte:** [W3C 2012]



compositora Laurie Spiegel (figura 3) sumariza características, durante a produção de *The Expanding Universe* (1975) ¹:

Todas as músicas no GROOVE eram representadas na memória digital como funções abstratas do tempo, séries paralelas de dois pontos, cada ponto sendo um instante no tempo e um valor instantâneo. A taxa de amostragem para essas funções, usada principalmente como controle de voltagem, era cronometrada por um grande e antiquado oscilador analógico que era normalmente fixado em 100 Hertz, cada ciclo do oscilador pulsando à frente do código, o computador lia, em cada uma das funções, naquele ponto do tempo, todos dispositivos de entrada e executava todas amostras. ²



Figura 3: Laurie Spiegel configurando a saída analógica do GROOVE, durante a produção de *The Expanding Universe*. **Fonte:** [Spiegel 1975].

3. Objetivo

Sumarizar um programa estruturado no *ScriptProcessorNode* segundo uma interpretação do GROOVE.

¹Disponível em <https://www.youtube.com/watch?v=dYUZmsfm4Ww>.

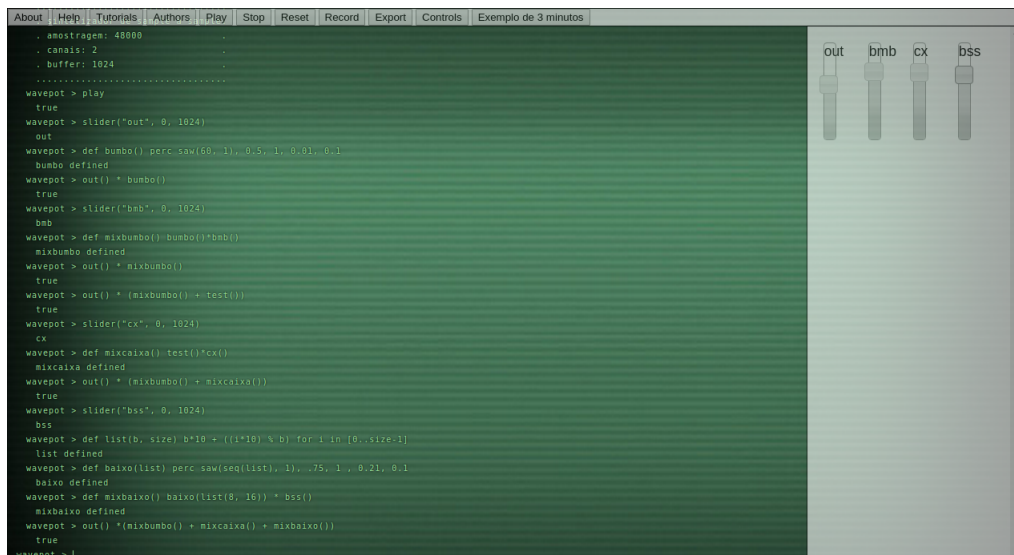
²Tradução de *All music in GROOVE was represented in digital memory as abstract functions of time, parallel series of point pairs, each point being an instant in time and an instantaneous value. The sampling rate for these functions, which would be used mostly as control voltages, was clocked by a big old-fashioned analog oscillator that was usually set to 100 Hertz, each cycle of the oscillator pulsing one run through the code, the computer reading all of the real time input devices and playing of all of the samples at that time point in each of the time functions.*

4. Metodologia de desenvolvimento

1) Customização um emulador terminal *Ptty.js* (<http://code.patxipierce.com/jquery-plugin/ptty/>). 2) Definição de um ambiente interno, baseado no ambiente *Wavepot* (Código-fonte disponível em <https://www.github.com/wavepot/wavepot>). 3) Definição de comandos deste ambiente interno: inspeção de funções, definição de novas funções, tocar, parar, pausar, gravar e download, criação de controles gráficos (*jQueryUI*) e gravação (<https://github.com/mattdiamond/Recorderjs/blob/master/recorderWorker.js>).

5. Resultados

O *Termpot* (ver figura 4)³ é uma customização do ambiente *Wavepot*. Novas funções sonoras podem definidas em linguagem *coffeescript* [Burnham 2011], encapsuladas em outras funções, e então executadas e gravadas. Apresentamos um sumário no código 1.



```
. amostragem: 48000
. canais: 2
. buffer: 1024
.....
wavepot > play
true
wavepot > slider("out", 0, 1024)
out
wavepot > def bumbo() perc.saw(60, 1, 0.5, 1, 0.01, 0.1)
bumbo defined
wavepot > out() * bumbo()
true
wavepot > slider("bmb", 0, 1024)
bmb
wavepot > def mixbumbo() bumbo()*bmb()
mixbumbo defined
wavepot > out() * mixbumbo()
true
wavepot > out() * (mixbumbo() + test())
true
wavepot > slider("cx", 0, 1024)
cx
wavepot > def mixcaixa() test()*cx()
mixcaixa defined
wavepot > out() * (mixbumbo() + mixcaixa())
true
wavepot > slider("bss", 0, 1024)
bss
wavepot > def list(b, size) b*10 + ((i*10) % b) for i in [0..size-1]
list defined
wavepot > def baixo(list) perc.saw(seq(list), 1, .75, 1, 0.21, 0.1)
baixo defined
wavepot > def mixbaixo() baixo(list(0, 16)) * bss()
mixbaixo defined
wavepot > out() * (mixbumbo() + mixcaixa() + mixbaixo())
true
wavepot > |
```

Figura 4: Aplicativo *Termpot*. Fonte: autores.

Problemas Técnicos

Existem *xruns*. Um *xrun* “(...) pode ser um estouro de buffer ou de uma saturação de um buffer. Um aplicativo de áudio ou não foi rápido o suficiente para transmitir dados (...) ou não rápido o suficiente para processar os dados”[User:Markc 2013]⁴. Existe uma hipótese, não verificada, que a possível fonte dos *xruns* é a biblioteca *jQuery*.

6. Conclusão

O *Termpot* está em estágio inicial de desenvolvimento. Ao revisitarmos uma abordagem histórica, esbarramos com questões relativas à síntese sonora, performance, e linguagem de programação textual simplificada para músicos. Por outro lado, ainda existem problemas técnicos. Neste sentido, o desenvolvimento de um *software* para criação musical baseada em funções matemáticas, aguarda o auxílio de contribuições, com interesse em uma abordagem pedagógica para o ensino de música eletroacústica.

³Disponível em <https://jahpd.github.io/termpot>.

⁴Tradução nossa de *An "xrun" can be either a buffer underrun or a buffer overrun. In both cases an audio app was either not fast enough to deliver data (...) or not fast enough to process data.*

```

1 $ | (1)
2 $ wavepot 1024 (2)
3 .....
4 . sintetizador de sample a sample. (3)
5 . amostragem: 44100 .
6 . canais: 2 .
7 . buffer: 1024 .
8 .....
9 wavepot > play (4)
10 true
11 wavepot > def AM440(f, a) sin f1, sin(440, a) (5)
12 AM defined
13 wavepot > inspect (6)
14 funcoes definidas
15 tau tmod mute stereo sin sin2 saw ramp ttri
16 tri sqr pulse noise perc test seq bpm nextevent
17 AM
18 wavepot > slider "f", 1, 1025 (7)
19 true
20 wavepot > slider "a", 0, 1024
21 true
22 wavepot > record (8)
23 true
24 wavepot > AM440 f()*1000, a() (9)
25 true
26 wavepot > export (10)

```

Código 1: Console do *Termpot* aguardando dados de entrada do improvisador (1). *Boot* do ambiente *wavepot* com um buffer de 1024 amostras por ciclo de DSP (2). Informações diversas do sistema (3). Início do processamento de áudio (4). O improvisador define uma função *AM440* (5). O sistema informa as funções disponíveis (6). Definição de interfaces gráficas controladoras (7). Comando para gravar o processamento em um arquivo (8). Execução da função *AM440* com controles (9). Realizar o download da gravação (10)

6.1. Trabalhos Futuros

i) criar um servidor; *ii)* otimizar o emulador, talvez substituindo o Ptty ou propondo melhorias; *iii)* suporte para amostras pré-gravadas.

6.2. Agradecimentos

Os autores agradecem ao Guilherme Rafael Soares e ao *labMacambira* pelas sugestões, aos desenvolvedores do *Wavepot* pelo código-aberto, e a FAPEMIG por subsidiar a pesquisa

Referências

- Burnham, T. (2011). Coffeescript: Accelerated javascript development. *Pragmatic Bookshelf*.
- Mathews, M. V. and Moore, F. (1970). GROOVE a program to compose, store, and edit functions of time. *Bell Telephones Laboratories*, page 7.
- Roberts, C., Wakefield, G., and Wright, M. (2013). The web browser as synthesizer and interface. page 6.
- Spiegel, L. (1975). The expanding universe: 1970s computer music from bell labs by laurie spiegel. disponível em http://www.retiary.org/ls/expanding_universe/. *Retiary*.
- Srikumar, S. (2013). Tamming the script processor node. disponível em <http://sriku.org/blog/2013/01/30/taming-the-scriptprocessornode/>.
- User:Markc (2013). Xruns: From the alsa wiki. disponível em <http://alsa.opensrc.org/Xruns>.
- 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.