

**CENTRO UNIVERSITÁRIO BELAS ARTES DE SÃO PAULO**

**BACHARELADO EM ARTES VISUAIS**

Pablo Vieira Pereira

***Pure Data* e suas aplicações nas artes visuais**

**São Paulo**

**2010**

Pablo Vieira Pereira

***Pure Data* e suas aplicações nas artes visuais**

Trabalho de Iniciação Científica

apresentado à FEBASP – Centro Universitário

Belas Artes de São Paulo

Curso: Artes Visuais: Bacharelado em Pintura, Gravura e Escultura

**ORIENTADOR:**

**Profº. Ms. Paulo Mattos Angerami**

***São Paulo***  
**2010**

Pereira, Pablo Vieira

*Pure Data* e suas aplicações nas artes visuais/ Pablo Vieira Pereira, 1ª Edição.

São Paulo: FEBASP, 2010

## **Agradecimentos**

Agradeço ao Mestre Paulo Mattos Angerami, à FEBASP, pela oportunidade concedida, e ao leitor.

## Sumário

<b>Lista de Imagens.....</b>	
<b>Resumo.....</b>	
<b>Ao leitor.....</b>	
<b><i>Abstract</i>.....</b>	
<b>CAPITULO I - As mudanças de paradigmas no campo da recepção midiática e o surgimento de PURE DATA.....</b>	<b>10</b>
<i>Introdução.....</i>	
<i>História do PD.....</i>	
<i>Open source X Proprietário.....</i>	
<b>CAPITULO II – <i>Pure Data</i>.....</b>	<b>15</b>
<i>Instalação.....</i>	
<i>Janela Inicial e Patch.....</i>	
<i>Objects.....</i>	
<i>GUIObjects.....</i>	
<b>CAPITULO III - Análise de alguns trabalhos de arte sobre o ponto de vista do determinismo tecnológico.....</b>	<b>22</b>
<b>CAPITULO IV - Campeonato mundial de sério – comentário a respeito da pesquisa prática .....</b>	<b>27</b>
Regras para um campeonato mundial de sério.....	
Visão técnica do trabalho Campeonato mundial de sério.....	
<b>Conclusão.....</b>	<b>33</b>

<b>Bibliografia.....</b>	<b>34</b>
<b>ANEXOS.....</b>	<b>36</b>

## Lista de Imagens

Figura 1 – Janela inicial do *Pure Data*. Fonte: Pereira. (2010).

Figura 2 – *Objects*. Fonte: Pereira.(2009).

Figura 3 – *GUI Objects*. Fonte: Pereira(2010).

Figura 4 – *Patch* campeonato mundial de sério. Fonte: Pereira (2010).

Figura 5 – *Patch* campeonato mundial de sério. Fonte: Pereira (2010).

Figura 6 – Croqui arena. Fonte: Pereira (2010).

## Ao leitor

Esta pesquisa investiga o uso do programa *open source Pure Data* no campo das artes visuais. Os textos a seguir estão divididos entre explicações técnicas relativas ao funcionamento do software estudado e a teorização sobre suas aplicações baseada em experimentações práticas.

O presente texto não é direcionado a profissionais com conhecimento específico, mas pretende apresentar uma ferramenta passível de uso por profissionais de outras áreas, como os artistas visuais.

Palavras-chave: *Pure Data*, arte-tecnologia, *open source*



## ***Abstract***

To the reader

This research investigates the use of the open source software Pure Data in the area of visual arts. The texts are divided into technical explanations on the operation of the software studied and theory of its applications based on practical experiments. The present text is not directed to professionals with specific knowledge, but aims to present a tool that can be used by professionals of other areas, such as visual artists.

Key words: *Pure Data*, art-technology, *open source*

## **CAPITULO I - As mudanças de paradigmas no campo da recepção midiática e o surgimento de *Pure Data***

Em 1973, em seu trabalho *Global Groove*, Nam June Paik declarou que no futuro o catálogo de canais de televisão seria maior que a lista telefônica de Manhattan e que o espectador seria livre para criar sua própria programação<sup>1</sup>. Quase meio século depois desta “previsão”, parecemos viver em uma época posterior ao futuro visualizado por Paik. Não só temos um número muito maior de canais de TV “tradicional”<sup>2</sup> (alguns pacotes de TV a cabo oferecem 250 opções de canais, contra apenas 13 possibilidades existentes até o final dos anos 1980), como também, se considerarmos as transmissões de vídeo veiculadas pela internet, esse número se torna quase infinito. Isso porque, entre outros motivos, após o surgimento de serviços *online*, como o “youtube.com”<sup>3</sup>, as possibilidades técnicas de gravação caseira (que já haviam se tornado notoriamente mais baratas devido ao desenvolvimento do sistema de captura digital, que se tornou popular nos anos 1990) encontraram o meio de distribuição e divulgação perfeito, transformando o espectador em produtor.

Assim, a internet cria uma fissura no estado de passividade do espectador diante de do produto “contemplado”, instituindo um novo paradigma ante as mídias criadas até então. Cumpre lembrar que os que usufruem dos produtos dessa mídia são comumente denominados “usuários”, ou seja, a internet permite que além de espectador o usuário se coloque na posição de criador e disseminador de uma

---

<sup>1</sup> A parte referida no texto pode ser encontrada em: <http://www.youtube.com/watch?v=lnLcRXfd3NI>

<sup>2</sup> Entenderemos aqui o termo “televisão tradicional” pela seguinte definição: s.f.(Do ingl. Television.) “1. Transmissão, por cabo ou por ondas radioelétricas, de imagens que podem ser reproduzidas sobre uma tela. 2. Aparelho receptor dessas imagens.” Grande Enciclopédia Larousse Cultural. São Paulo, 1998.

<sup>3</sup> *Youtube.com* é um site de *upload* e divulgação de vídeo criado em 2005 por Chad Hurley, Steve Chen e Jawed Karim, e que desde 2006 é de propriedade do gigante da internet *google*. *Youtube* funciona graças à tecnologia de *player* desenvolvida pela empresa Adobe, o *Adobe Flash Player* e ao codec *Sorenson Spark* (H.263). Para mais informações: <http://pt.wikipedia.org/wiki/YouTube>.

produção midiática<sup>4</sup>. Para Arlindo Machado<sup>5</sup>, esse fenômeno se dá porque as novas gerações não se acomodam mais em papéis passivos, apresentando uma necessidade de participação ativa na construção do pensamento audiovisual dos nossos tempos. Em contrapartida, nas palavras de Carlos Farinha, “todo esse movimento é empolgante, o resultado é que não é empolgante até agora, o produto deste movimento (...) em grande parte é porcaria.”<sup>6</sup>

Com a facilidade de criação e a possibilidade de troca de experiências entre estes produtores, surge também uma tentativa de horizontalização da produção tecnológica. Segundo Otávio Savietto<sup>7</sup>, na última década houve uma busca pela humanização da tecnologia, que se dá devido à constatação coletiva do fato de que não se pode fugir da convivência com os meios tecnológicos a que estamos sujeitos no dia-a-dia em diversas instâncias de nossas vidas. Assim, são criadas comunidades de usuários, que objetivam a aproximação do homem com a máquina, através da interação entre eles<sup>8</sup>.

---

<sup>4</sup> Este é um processo que se inicia ao longo da história da televisão e tem sua acentuação a partir do período pós-guerra e ao longo dos anos, devido a desenvolvimentos técnicos, há uma individualização da programação, ou pelo menos do modo como se organiza a recepção dessa programação, por exemplo, é depois do surgimento do controle remoto que se constata o efeito *zapping* (ato de trocar de canal diversas vezes), que é uma maneira de o telespectador burlar a imposição de comerciais feita pela grade de programação. Também podemos citar o efeito *zipping* (ato de avançar ou retroceder um filme em um vídeo), como mais uma conquista do espectador em relação a como receber as informações da programação. Assim, a maneira como os usuários de sites como chatroullete.com e youtube.com se comportam diante do conteúdo por eles apresentados como assistir mais de um vídeo ao mesmo tempo (*grazing*) ou não assistir o conteúdo todo, saltando de um vídeo para outro (*flipping*), já podia ser identificado no público da televisão.

<sup>5</sup> A entrevista completa com Machado pode ser encontrada em <http://www.youtube.com/watch?v=zbQXGuSPh3o>

Acessado pela última vez em 19/3/2010 23:43.

<sup>6</sup> A entrevista completa com Farinha pode ser encontrada em <http://www.youtube.com/watch?v=hAIZ31rF7hM&feature=related> Acessado pela última vez em 19/3/2010 23:43.

<sup>7</sup> Diálogos Transversais 10/12/2010.

<sup>8</sup> HCI (*human computer interaction*) é a sigla usada para descrever esta interação. Tal relação entre usuário e máquina é mediada por uma interface, que é composta tanto por software como hardware; essa interação é propiciada em um computador pessoal, por exemplo, pelos periféricos como mouse, teclado e *joysticks*. Em arte é comum o desenvolvimento de interfaces tangíveis que possibilitem múltiplas experiências sensoriais aos seus usuários, como a *reactables* ([http://www.youtube.com/watch?v=vm\\_FzLya8y4](http://www.youtube.com/watch?v=vm_FzLya8y4)) por exemplo.

É nesse contexto que floresce o desenvolvimento de programas *open source* como o *Pure Data*.

### **HISTORIA DO PD**

Na metade do século XX, técnicas eletrônicas começaram a ser usadas para construção musical. Com o desenvolvimento da tecnologia e a popularização dos computadores pessoais, essa construção, que era feita basicamente usando *hardwares* ligados por fios, nos dias atuais está intimamente ligada ao uso de *softwares*<sup>9</sup>.

Em 1988, Miller Puckette desenvolveu um *software* que pode ser considerado o avô do *Pure Data* (PD), chamado Max<sup>10</sup>. Naquela época, os computadores disponíveis não eram suficientemente avançados para lidar diretamente com áudio. Por isso, Max lidava apenas com sinais musicais para serem sintetizados.

Entre 1993 e 1994, Puckette, em conjunto com David Zicarelli, implementou ao Max o programa *MSP*. Nessa época, os computadores pessoais já possuíam placas de som e processadores suficientemente desenvolvidos, possibilitando que um computador pessoal se transformasse em um instrumento musical capaz de realizar *performances* musicais ao vivo.

Posteriormente, David Zicarelli transformou Max/MSP em um *software* proprietário, que é comercializado como produto. Puckette, por sua vez, baseado na programação de *Max/MSP*, escreveu o *Pure Data* e o tornou um programa *open source*, que pode ser baixado e modificado livremente (discutiremos as diferenças entre *open source* e programas proprietários a seguir). Tanto o MAX/MSP quanto o *Pure Data* possuem versões estendidas que vão além do que foram originalmente programados para fazer (desenvolvimento de sons eletrônicos e MIDI<sup>11</sup>).

---

<sup>9</sup> O nascimento das *performances* musicais usando computador é atribuída ao ano de 1957 quando um computador IBM 704 tocou uma música de 17 segundos escrita em um dos programas da família MUSIC, o Music I.

<sup>10</sup> Max é uma homenagem a Max Vernon Matthews (nascido em 1926, em Columbus, Nebraska), Matthews foi o primeiro a escrever um programa de geração de áudio para computador na década de 1950, chamado MUSIC.

<sup>11</sup> MIDI (**M**usical **I**nstrument **D**igital **I**nterface) é um formato de arquivo desenvolvido nos anos 1970 que permite a comunicação de instrumentos musicais com equipamentos eletrônicos, arquivos MIDI não contêm áudio e sim instruções que são lidas pelos aparelhos eletrônicos para produzir som.

## **OPEN X PROPRIETARIO**

*Softwares* como *Max/MSP* e sistemas como o *Windows* são criados e desenvolvidos por programadores contratados por grandes empresas que visam à venda do programa como mercadoria. As linhas de código que formam esses programas se encaixam nas leis de proteção aos direitos intelectuais. Por essa razão, qualquer modificação feita nesses softwares sem a prévia autorização dos seus idealizadores é considerada crime, assim como a cópia e comercialização sem os devidos préstimos aos fabricantes.

Há, porém, uma outra tendência muito forte que teve início nos primeiros anos do século XXI, ligada à invenção de programas de compartilhamento de arquivos por P2P<sup>12</sup> e de disseminação livre de dados e informações via internet. Desde o surgimento do *Napster*<sup>13</sup>, grandes empresas de informática, gravadoras, companhias de cinema e editoras de livros parecem ir na contramão desta tendência.

Assim, como aponta José Chispiniano (2002), “o grande símbolo da globalização capitalista: a internet (...) permitiu que pequenos grupos em cidades diferentes divulgassem seus discursos, trocassem informações e articulassem ações conjuntas ou simultâneas, sem a necessidade de uma pesada estrutura de organização” (p.18). Softwares de código aberto (*open source*) não visam lucros com a venda de um “objeto de consumo”. Sua documentação é de domínio público e está disponível em vários lugares da rede. Muitos deles permitem que os próprios usuários modifiquem sua programação, o que, em termos práticos, possibilita um desenvolvimento muito mais rápido, se comparados com os programas proprietários. Isso ocorre porque no lugar de uma equipe de programadores existem milhares de usuários espalhados em todo o

---

<sup>12</sup> P2P vem do inglês “*peer to peer*”, que significa par por par, é um tipo de arquitetura de rede onde os computadores estão ligados entre si sem nenhum servidor principal, já que todos os ligados nessa rede fazem o papel de servidor e cliente.

<sup>13</sup> *Napster* foi um programa de compartilhamento de arquivos criado por Shawn Fanning em 1999, em sua casa, quando ele tinha apenas 19 anos. Seu programa era revolucionário, pois usava a arquitetura P2P para compartilhamento de arquivos MP3, o que mudou o modo de consumir música para sempre e também causou grandes implicações legais. Em janeiro de 2001, *Napster* chegou a ter 8 milhões de usuários, movimentando uma biblioteca de 20 milhões de músicas. Em março daquele mesmo ano, *Napster* foi fechado, graças aos processos legais movidos pela indústria fonográfica.

mundo conectados entre si através de fóruns e listas de discussão, pensando ao mesmo tempo em novas soluções e caminhos para o *software*.

Programas proprietários possuem travas de segurança. É ilegal quebrar essas travas, inventar maneiras de quebrá-las e divulgar maneiras de burlar esses sistemas de segurança. Porém, programas de código aberto contornam esse problema, pois não existe nada a ser protegido, já que o que temos é a troca de informação e não a venda dela (DOCTOROW,2009).

## CAPÍTULO II – *Pure Data*

Além de ser *open sourcer*, *Pure Data* se diferencia dos outros programas do mesmo gênero por ser um *software* de programação com interface gráfica. Normalmente, *softwares* de programação usam linhas de comando<sup>14</sup> para criar aplicativos, ou seja, o programador deve escrever um texto seguindo uma sintaxe específica para se comunicar com o programa e assim fazê-lo seguir suas ordens. Isso exige dos seus usuários certa dedicação para a aprendizagem da linguagem destes programas. É como aprender uma língua estrangeira, como inglês ou francês, tornando a programação difícil e pouco atraente para pessoas como os artistas e músicos, que não estão acostumadas a lidar com este tipo de paradigma.

Por ser um programa com interface gráfica, *Pure Data* não possui linhas de código. Todos os comandos foram substituídos por objetos gráficos chamados de “caixas” ou “box”, o que torna a aprendizagem muito rápida e a construção de aplicativos muito mais intuitiva. Claro, *Pure Data* ainda deve ser entendida tal como qualquer outra linguagem de programação, ou seja, o usuário deve aprender sua sintaxe e regras de construção para que assim possa produzir algum resultado.

### Instalação

A instalação de *Pure Data* no sistema *Windows* é semelhante a qualquer outro programa instalado nessa plataforma. Apenas no *Linux* ela pode apresentar algumas particularidades, dependendo da versão do sistema que o usuário tem instalado em seu computador. Presumiremos que os usuários conheçam o método de instalação de seus sistemas e seguiremos em frente. Porém, deixaremos os *links* a seguir, com o passo a passo desta operação, caso haja alguma dúvida restante.

*Instalação windows*: <http://en.flossmanuals.net/PureData/InstallingWindows>

---

<sup>14</sup> Linhas de comando a grosso modo são os códigos de uma determinada linguagem que permitem a execução de uma determinada tarefa pelo programa.

Instalação linux Ubuntu: <http://en.flossmanuals.net/PureData/InstallingUbuntu>

## **Janela inicial e *Patch***

(Anexo A Figura 1)

Quando o usuário abre o *Pure Data* em seu computador, encontra uma janela inicial. Essa janela possui informações importantes sobre o funcionamento do programa. Ela registrará se algum erro ocorrer durante seu uso. O usuário também pode usar essa janela para imprimir informações durante a execução do *patch*.

Se o usuário estiver usando o *Pure Data* para sintetizar áudio, essa janela possui algumas ferramentas úteis:

- a) “computer audio” – se esta opção estiver marcada, o som do PD estará ligado. Se estiver desmarcada, o programa não produzirá nenhum som.
- b) Há do lado esquerdo duas caixas com as marcações IN OUT. Essas caixas marcam a altura do som que entra e sai do computador. Se a marcação for maior que 100, o som estará “estourado” e o usuário o ouvirá distorcido. Quando isso acontece os botões CLIP ficam vermelhos.
- c) Por último, o botão “DIO erros” (Digital IN OUT erros). Ao ser clicado, esse botão mostra o número de vezes que *Pure Data* teve problemas para se comunicar com a placa de som do computador.

Para que todas essas ferramentas sejam usadas, deve-se deixar marcada a opção “peak meters”.

Os aplicativos criados pelos usuários no *Pure Data* são chamados de “patches”, assim como as páginas em branco para criar esses aplicativos. Segundo a referência oficial do *Pure Data*, esse nome se deve à história do uso de equipamentos eletrônicos na música, quando o som era produzido e transformado ligando cabos (*patch cables*) a *hardwares*.

Para produzir um novo *patch* em *Pure Data*, o usuário deve criar uma série de objetos gráficos (*objects* e *GUI objects*) e interligá-los de diferentes maneiras, por linhas gráficas, como se faziam com os *patch cables*. Para isso, como em muitos outros



programas de interface gráfica, o usuário deve ir em “file”, clicar em “new” e abrir uma nova janela.

Nessa nova janela, como dito anteriormente, chamada de patch, o usuário encontrará:

- File: como as opções comuns à maioria dos programas para criar um novo arquivo; salvar; abrir; imprimir, etc., mais as opções:
  - a) *message*: que manda mensagem direto para a janela inicial do *Pure Data*;
  - b) *path*: que serve basicamente para dizer em que parte do computador *Pure Data* deve procurar os arquivos externos que fazem parte da sua biblioteca e permitem que certos objetos sejam criados. Essas localidades, como o usuário poderá perceber, podem ser mudadas, deletadas e outros locais podem ser incluídos.
  - c) *Startup*: serve para indicar ao *Pure Data* quando ele é iniciado, quais *paths* usar, dar informações sobre *drives* de áudio, etc.
  
- Edit: com as funções comuns de desfazer, cortar, copiar, colar e algumas funções específicas como:
  - a) *toggle console*: que desaparece com a área onde são impressas informações na janela inicial de *Pure Data*;
  - b) *clear console*: que apaga as mensagens antigas na janela inicial;
  - c) *edit mode* (tecla de atalho ctrl+e): esta função serve para entrar e sair do modo de edição de um *patch*. Quando ela está ativada, a seta do mouse muda para a representação de uma mão com o dedo indicador apontado. Só no modo de edição o usuário pode fazer modificações na estrutura da *patch*, como criar novos objetos, mudá-los de lugar ou deletá-los. Importante frisar que a *patch* pode continuar sendo executada mesmo quando o usuário está no *edit mode*, ou seja, pode-se executar alterações na estrutura do *patch*, mesmo quando ele está rodando. Porém, pode ser controlado com o *edit mode* desabilitado.

- Put: onde se encontram as citadas estruturas gráficas que substituem as linhas de comando na linguagem do *Pure Data*, elas são chamadas de *objets* e *gui objects*. Trataremos disso com mais detalhes à frente.
- Find: uma ferramenta de busca de *Pure Data* que também pode ser acessada clicando em *edit*.
- Windows: útil quando se tem muitas janelas de *patch* abertas ao mesmo tempo, pois podem ser acessadas por esse recurso.
- Media: aqui encontram-se:
  - a) *audio on* (ctrl+/,), que liga o áudio e *audio off* (ctrl+.) que desliga o áudio.
  - b) uma lista de *drivers* de áudio, que o usuário pode escolher<sup>15</sup>.
  - c) *audio settings*: a configuração de áudio, aqui podem ser mudados o número de canais de entrada e saída, *delay*, etc.
  - d) *MIDI settings*: para configuração de *drivers* MIDI.
  - e) *Test audio and midi*: ao clicar nessa opção uma janela de *patch* do *Pure Data* se abrirá. Nela se encontram basicamente duas opções de áudio: o *noise* e o *tone* e um número de canais de 1 a 8. Essa opção serve para detectar defeitos nas saídas de áudio e na placa de som.
- Help: *Help* possui a documentação de *Pure Data*, além de uma série de *links* na internet, como lista de email, fóruns, e *irc channel*, além da opção *report bug*, onde o usuário pode indicar para os programadores responsáveis os erros da programação.

## **Objects**

(Anexo A figura 2)

---

<sup>15</sup> No Windows são MMIO e ASIO, no Linux jack, ALSA e OSS e no Mac portaudio e Jack (esses *drives* variaram, dependendo da configuração do computador em que *Pure Data* está instalado).

Objects box (caixas de objetos): são os comandos do PD. O PD possui centenas de objetos em sua versão estendida. Quando se cria um objeto, ele aparece em branco na tela. Assim, cabe ao usuário escrever qual a sua função. Normalmente, o nome de um objeto está ligado à sua função. Por exemplo, se o usuário deseja criar um retângulo, ele deve escrever “*rectangle*” dentro do objeto. Em alguns casos o nome do objeto pode ser precedido por um sufixo como “pix”, por exemplo, ou ser apenas uma sigla como dac~ (Digital to Analog Converter). Esse objeto é responsável pelo acesso do PD à placa de som. Quando a *objects box* está vazia ou com o nome incorreto, ela aparece como um quadrado com linhas incompletas. Quando isso acontece na janela inicial de *Pure Data* aparecerá a mensagem “... couldn't create”. *Tecla de atalho ctrl + 1.*

Message box (caixa de mensagem): message são instruções escritas dadas diretamente aos objetos. Ao clicar sobre uma caixa de mensagem escrita, o usuário envia o conteúdo da caixa para um destino. Assim, por exemplo, podemos criar uma mensagem “teste de mensagem” e ligá-la a um objeto “print”. Ao clicar sobre a mensagem a frase aparecerá na janela do PD. *Tecla de atalho ctrl + 2.*

Number box (caixas de número): caixas de número permitem que o usuário use números em seus *patches*. Os números podem ser negativos, quebrados ou inteiros. Podem ser controlados por comandos de teclado ou com o mouse, e usados em operações matemáticas. *Tecla de atalho ctrl + 3.*

Symbol box (caixas de símbolo): caixa de símbolos serve para registrar o *output* de alguns objetos. Por exemplo, o usuário pode ligá-la ao objeto “keyname” e ela indicará que tecla está sendo pressionada no teclado. *Tecla de atalho ctrl + 4.*

Coment (comentário): comentários permitem que o usuário escreva textos de anotação na janela do *patch*. Assim, o próprio usuário ou outras pessoas que abram o *patch* terão a possibilidade de entender como os objetos ali presentes se articulam. *Tecla de atalho ctrl + 5.*

## **GUI objects**

(Anexo A Figura 3)

GUI *objects* (*Graphical User Interface objects*) são equivalentes virtuais para alguns tipos de controladores do mundo real, como por exemplo, botões, potenciômetros, etc. GUI *objects* também são úteis para aperfeiçoar visualmente as *patches*. Os GUI *objects* são:

Bang: *bang* é um objeto que funciona como uma campainha. Cada vez que é clicado ele manda um sinal de ataque, ou seja, ele diz para outros objetos iniciarem uma ação. Ligado a um metrônomo, por exemplo, ele pode ser usado para registrar graficamente quantas vezes por segundo um sinal é enviado dentro do *patch*. Tecla de atalho ctrl + shift + b.

Toggle: *toggle* é um botão do tipo liga/desliga. Por padrão, ele trabalha com dois valores, 0 e 1. Porém, isso pode ser mudado nas propriedades de *Toggle*. Ele pode ser usado para acionar e cancelar uma ação, por exemplo. Tecla de atalho ctrl + shift + t .

Number2: *number2* tem basicamente as mesmas funções da caixa de número, com a vantagem de que quando o *patch* é fechado, *Number2* pode guardar o número atribuído a ele. Tecla de atalho ctrl + shift + n.

Vslider e hslider: são potenciômetros verticais e horizontais, respectivamente. Por padrão, seus valores vão de 0 a 127, o que pode ser mudado em suas propriedades. Seus valores são mudados usando-se o mouse. Tecla de atalho ctrl + shift + v e ctrl + shift + h, respectivamente.

Vradio e hradio: são, respectivamente, botões de rádio vertical e horizontal, compostos por uma sequência de oito botões, o que pode ser mudado em suas propriedades. Quando um botão está ativo os outros sete permanecem desligados. Tecla de atalho ctrl + shift + d e ctrl + shift + i, respectivamente.

VU: VU é um medidor de volume de sinais de áudio. Os valores da escala podem ser mudados na propriedade. Tecla de atalho ctrl + shift + u.

Canvas: *Canvas* é um retângulo cinza que pode ter sua área e cor mudadas em suas propriedades. Ele serve para organizar e deixar visualmente mais legível a *patch*. Tecla de atalho ctrl + shift + c.

.....

Todos os objects e os GUI *objects*, além das teclas de atalho, podem ser inseridos em uma *patch* clicando-se em *put*, na parte superior da janela.

Os *inlets* (*hot e cold*) são pequenos retângulos localizados na parte superior e inferior dos objetos, os quais permitem que eles se liguem e interajam entre si. Assim, *inlets* que ficam na parte superior permitem que o objeto sofra influência do objeto acima e *inlet* na parte inferior permite que o objeto influencie o que vem abaixo, pois os objetos são ligados entre si por linhas.

### **CAPÍTULO III – Análise de alguns trabalhos de arte sobre o ponto de vista do determinismo tecnológico**

“ Constata-se em nosso entorno como os aparelhos se preparam a programar, com automação estúpida, as nossas vidas; como o trabalho está sendo assumido por máquinas automáticas, e como os homens vão sendo empurrados rumo ao setor terciário, onde brincam com símbolos vazios; (...) como o viver passa a alimentar aparelhos e a ser alimentado por eles.”(Flusser, 1985, p. 40).

*Pure Data*, assim como todas as criações tecnológicas pós-industriais, é um aparelho<sup>16</sup>. Para alguns autores como Flusser, o aparelho é um brinquedo e o homem que o manipula um jogador que joga contra seu brinquedo, tentando adentrar seu programa, esgotando por completo todos os seus segredos, “a fim de descobrir-lhe as manhas”.

No entanto, o aparelho possui um programa infinito, sendo impossível para o seu manipulador esgotar por completo suas possibilidades; diz ele:

“(...) a competência do fotógrafo deve ser apenas parte da competência do aparelho”, ou seja, o número de possibilidades potencial de uso do aparelho deve ser maior que o número de aplicações de que o fotógrafo faz uso, a fim de que esse último siga jogando e descobrindo esses potenciais” (FLUSSER, 1985, p 15).

Já no livro filosofia da tecnologia, Val Dusek dedica um capítulo ao que ele chama de determinismo tecnológico. Segundo o autor, as teorias deterministas dizem que as formas da cultura, como a arte e a religião, dependem especificamente de fatores científicos, o que coloca em cheque as noções de livre arbítrio do ser humano.

Por exemplo, deterministas biológicos acreditam que as decisões que uma pessoa toma na vida, como as relações pessoais e a escolha da carreira de trabalho, etc., são

---

<sup>16</sup> Segundo o Glossário de Flusser, em A Filosofia da Caixa Preta, Aparelho: Brinquedo que simula um tipo de pensamento; Brinquedo: Objeto para jogar;Jogo: Atividade que tem fim em si mesmo.

determinadas de acordo com uma pré-programação que visa à perduração de alguns genes durante gerações.

Para explicar o determinismo tecnológico, usemos um dos exemplos do próprio autor. Uma sociedade capitalista só foi possível graças às forças dos motores a vapor – sem eles teríamos uma sociedade baseada na exploração do trabalho escravo. Na visão de alguns pesquisadores, seria impossível pensar em uma sociedade sem escravos na Grécia antiga ou em Roma, pois não existia tecnologia avançada o bastante para fabricar materiais resistentes à pressão do vapor, podendo assim liberar os homens da tarefa que a partir da revolução industrial começou a ser feita pelas máquinas. Igualmente a partir desse período não havia mais sentido, economicamente falando, manter potenciais consumidores privados dos seus direitos de liberdade e de compra.

Como diz o autor : “Na Idade Média, o conflito – liberdade *versus* determinismo – foi formulado em relação com a ideia de Deus (...) que podia prever e controlar tudo, mas deixou Adão pecar (...) Com a ascensão das leis científicas e ideias deterministas, o conflito com a liberdade assumiu uma nova forma. Se tudo o que fazemos é fisicamente, causalmente determinado, podemos ser livres?”

Exposições com foco em arte tecnologia como a File na FIESP e Emoções Artificiais do Itaú Cultural costumam apresentar diversas obras que utilizam do HCI e que pretendem viabilizar a aproximação entre público e tecnologia. *Skinstrument*, do holandês Daan Brinkmann, exposta na FILE de 2009<sup>17</sup>, é uma destas obras. Trata-se de uma superfície tangível que tenta transformar o espectador em parte da obra, por meio do toque do espectador que produz alterações sonoras no ambiente, o que resulta na ilusória sensação de participação. Nesse processo, a interação entre homem e máquina se dá em níveis superficiais, não existindo troca efetiva entre as partes envolvidas, pois o “ativador” da obra é colocado na posição de apertador de botão.

---

17

“Skinstrument” é um instrumento musical que pode ser tocado por duas pessoas. Através de uma pequena e imperceptível corrente elétrica, os jogadores tornam-se parte de um circuito. Quando eles se tocam na pele, esse circuito começa a gerar som. A intensidade do toque determina a frequência do som.” ( [http://www.file.org.br/file2009/press\\_sp/](http://www.file.org.br/file2009/press_sp/))

Já a obra *Moving Mario* do chinês Keith Lam<sup>18</sup>, mostra-se como uma grande sátira crítica ao HCI<sup>19</sup>, contrapondo a abordagem *high-tech* vista em *Skinstrument* a uma pequena gambiarra mecânica de papelão e sucata eletrônica. O “ativador” controla com um *joystick* um pequeno desenho do personagem Mario que se encontra preso em um carrinho que se locomove pelo cenário de papel dentro de uma televisão. Podemos então entender *Moving Mario* como uma demonstração irônica do que realmente o público é dentro da exposição: um jogador sem objetivo destinado a ser um mero operador da obra, apertando botões, puxando alavancas e se entregando ao estado de êxtase. O que nos leva a perguntar: haveria espaço para as ações inter-relacionais (homem/homem, homem/máquina, máquina/homem, máquina/máquina) dentro da produção artística tecnológica hoje, ou estamos destinados a percorrer caminhos já traçados quando nos deparamos com uma obra de arte tecnológica?

Em algumas obras da artista Vivian Cacurri esta tensão entre o livre arbítrio e o determinismo, representados pela dicotomia do fator humano e a acertividade da arte-tecnologia, é valiosamente explorada. Diz a artista sobre seu trabalho *Maquiagem Aumentada*: “Nas artes plásticas tradicionais você tem muito espaço para improvisar e criar coisas do nada, mas na arte-tecnologia, tudo é mais planejado, mais exato, então eu queria humanizar as coisas que eu tava fazendo, deixar as coisas mais flexíveis, depende-se do momento, a performance vem como um fator psicológico muito sutil, você tem a coisa tecnológica que é bem preparada (premeditação), mas se coloca muito do sangue da pessoa para criar aquilo (improvisação)”.<sup>20</sup>

Tanto em *Maquiagem Aumentada* quanto em *Canções Submersas*, outro trabalho em que a artista usa *Pure Data* para criar a experiência sonora de sua obra, Vivian Cacurri

---

<sup>18</sup> “Moving Mario definitivamente não reproduz *Super Mario Bros* de qualquer maneira. Ao aproveitar parcialmente o conceito e alguns elementos-chaves do jogo para TV, *Moving Mario* tenta desafiar alguns elementos tradicionais dos jogos. Durante todo o processo, os jogadores podem repensar a relação entre o jogador e o jogo”. ( [http://www.file.org.br/file2009/press\\_sp/](http://www.file.org.br/file2009/press_sp/)) acessado em 23 de julho de 2010 às 23:23 h.

<sup>19</sup> Mais informações sobre HCI podem ser encontradas no capítulo I.

<sup>20</sup> Entrevista realizada para um minidocumentário da exposição *Rumos do Itaú Cultural* pode ser encontrada aqui: [http://www.youtube.com/watch?v=J6ZyMZ47\\_sM](http://www.youtube.com/watch?v=J6ZyMZ47_sM) acessado em 23 de julho de 2010 às 20:03 h.



cria um espaço de leis rígidas, representado fisicamente em Canções Submersas por um aquário – objeto destinado a limitar e dar forma a matéria amorfa, a qual proporciona vida a seres não racionais – o aquário é limitador e dita o espaço de atuação. Porém, dentro deste espaço se cria um microcosmo de possibilidades, criando uma pequena abertura para o acaso dentro da rigidez tecnológica.

Também podemos analisar os trabalhos apresentados pelo coletivo *The Champions*, na PDCON 09, sob o ponto de vista das questões do livre arbítrio do homem e o determinismo tecnológico. Tanto “Do Sinusoids Dream of electric Sweeps?” quanto “He boxed regularly and was strong and very brave and always a perfect gentleman” são performances de *live coding*<sup>21</sup>, portanto, as opções tomadas pelos artistas/programadores durante a performance podem (e vão) mudar os rumos que a obra vai tomar.

Em *Do Sinusoids Dream of electric Sweeps?*<sup>22</sup> existem *patches* dotados de inteligência artificial que são automodificáveis; isso significa que além da ação do artista que cria a programação, o próprio programa “se cria”, agindo de modo “político”, ligando *patches* entre si, excluindo-se, enviando mensagens que definem novas funções, criando outros *patches*, etc. Assim, o trabalho coloca lado a lado as escolhas tomadas de intuitivo e imediato por humanos e máquina (mesmo que artificialmente).

Nos dois casos citados acima podemos dizer que os artistas trabalham com uma “imprevisibilidade determinada”, ou seja, cria-se um pequeno espaço para o inesperado, porém este espaço é contornado pelas condições que o possibilita e tais condições no campo virtual e tecnológico serão sempre inferiores às encontradas no mundo palpável. Nunca se esgotarão as possibilidades de percurso que os peixes no aquário de Vivian Cacuri podem fazer e ninguém pode obrigá-los a escolher nadar por

---

<sup>21</sup> *Live Coding* é uma “modalidade” de programação, diferente da praticada por Vivian Cacuri por exemplo, artistas/programadores que utilizam o *live coding* criam o programa ao vivo, como uma apresentação de improviso musical; neste tipo de programação, frequentemente o programa é considerado a própria obra.

<sup>22</sup> Esta obra pode ser vista no seguinte endereço:  
[http://fff.murspace.net/Members/umlaeute/videos/sinusoids\\_piksel08.ogg/view](http://fff.murspace.net/Members/umlaeute/videos/sinusoids_piksel08.ogg/view) acessado em 14 de maio de 2010 às 14:01 h

um caminho ou por outro, porém as possibilidades programadas das mudanças de espacialidade que cada um desses peixes pode fazer na música ambiente são restritas. O mesmo acontece em “He boxed regularly and was strong and very brave and always a perfect gentleman”, os boxeadores podem optar por intuição a fazer determinado movimento ou não, porém, para cada ação imprevisível de um ser humano, a máquina terá uma resposta específica e direcionada.

## CAPÍTULO IV - Campeonato mundial de sério – comentário a respeito da pesquisa prática

---

ATENÇÃO: o leitor pode estranhar o conteúdo deste capítulo; portanto, cabe a nós a função de dar algumas explicações prévias.

Este capítulo é composto por uma coletânea de textos escritos ao longo do desenvolvimento da obra “Campeonato mundial de sério”.

Em *Visão técnica do trabalho Campeonato mundial de sério* foi adotado o estilo usualmente encontrado para se referir aos componentes de um *patch* em *Pure Data*; portanto, sempre que o leitor encontrar uma palavra entre este símbolo [ ] estará se referindo a um objeto do programa; quando uma palavra aparecer entre { } significa que ela é uma mensagem (para mais informações sobre os componentes do PD leia os capítulos anteriores).

*Regras para um campeonato mundial de sério* é um texto explicativo sobre como se configura o trabalho. **Este texto é parte integrante de uma obra de arte**, apenas sendo alterado em alguns pontos para melhor se enquadrar no corpo desta publicação; portanto, ele segue regras próprias de construção que em alguns pontos diferem do adotado no resto da pesquisa.

---

O trabalho *Campeonato mundial de sério* se utiliza dos meios tecnológicos, mas não tem a tecnologia como fim principal. Sob o ponto de vista do programador da obra, o *patch Campeonato Mundial de Sério* é a contestação da estética tecnológica contemporânea, através de um comentário sarcástico a respeito dos procedimentos frequentemente adotados pelas obras contempladas em salões e exposições específicas sobre o “tema”.

A obra é planejada para funcionar como um game interativo entre duas pessoas. Os participantes não interagem com a máquina de forma direta, e mais ainda, quando dão sinais visíveis de interação entre si são automaticamente eliminados da partida.

“Campeonato Mundial de Sérió” foi construído como um programa de vigilância e monitoramento – muito similar aos sistemas de vídeo internos dos prédios e das áreas de segurança – desenvolvido para acusar aquele que desobedece a regra. Os usuários da obra (competidores) devem aceitar previamente a condição de serem analisados pelos que controlam o programa de computador e terem sua imagem exposta para uma plateia. Como recompensa, aquele que “ganha” o jogo recebe uma quantia grande de um objeto de consumo desejado – essa quantia é tão descomunal que se torna absurdo e inútil possuir tantas peças daquele determinado objeto.

A obra tenta subverter o conceito de participação e interatividade do usuário do aparato tecnológico, já que a “participação” e interação daqueles que de fato estão no jogo é extremamente limitada; se o participante possui autocontrole continua na partida, se não, é eliminado. Os competidores do “Campeonato Mundial de Sérió” são mais uma das engrenagens que movem o trabalho e nutrem a vivência da plateia. Os competidores são, por assim dizer, parte do programa, um mecanismo orgânico que dá sentido ao aparato digital. Existem pouquíssimas vantagens em participar do jogo como competidor, assim como não existem vantagens reais em se vencer o jogo.

Por outro lado, cabe ao público (plateia) que assiste à competição vivenciar diversos tipos de relações criadas no microuniverso do campeonato: apostas, torcidas, consumo durante a partida. A plateia tem a liberdade de vivenciar toda uma sorte de acontecimentos que só são proporcionados durante aquele evento específico. Nesse sentido, os que estão teoricamente “fora” da obra têm a possibilidade de experiência muito maior dos que, de fato, usam o aparato tecnológico.

\*

### **Regras para um campeonato mundial de sério**

\* As seguintes determinações devem ser seguidas à risca para que a obra Campeonato Mundial de Sérió se configure:

#### *Do campeonato*

Campeonato mundial de serio é uma obra que se dá em uma arena.

Dois competidores são colocados frente a frente e se encaram olhando um para o rosto do outro; aquele que sorrir primeiro é eliminado.

Os embates são filmados e transmitidos ao vivo para o público.

### Dos competidores

Os competidores devem estar previamente inscritos na competição, e serão organizados em chaves eliminatórias.

Cada embate deve durar no máximo 2 minutos; se nenhum competidor rir nesse tempo estipulado, os dois serão eliminados.

Não é permitido ao competidor desviar o olhar de seu oponente assim como também não serão toleradas tentativas de induzir o oponente à derrota utilizando-se de subterfúgios, como caretas e ou gestos. Cada desvio a essa regra será considerado como uma falta; ao se somarem 2 faltas o competidor estará automaticamente eliminado da competição.

O grande vitorioso ganhará o montante de 100 (cem) reais em balas sabor *tutti-frutti*.

### Dos juízes

As competições serão mediadas por dois juízes a quem será entregue a função de monitorar os competidores, penalizar faltosos e identificar sorrisos.

O juiz de campo estará localizado na arena junto aos competidores e portará um apito e cartões nas cores amarela e vermelho (os cartões tem a função penalizante – amarelo para primeira falta e vermelho para eliminação).

O juiz de mesa estará localizado fora do perímetro da arena e acompanhará a competição por meio de um sistema de vídeo-monitoramento; a ele também cabe a função de retransmitir o evento para o público via telão.

Tanto o juiz de mesa quanto o juiz de campo terão plenos poderes inalienáveis.

### Do sistema de monitoramento

O sistema de monitoramento será operado pelo juiz de mesa, e tem duas funções principais: indentificar a expressao e movimentação facial dos competidores e retransmitir o evento do ponto de vista dos oponentes para o público presente.

Este sistema é composto por duas câmeras localizadas no próprio corpo dos competidores, um projetor, um telão, e por um *software* desenvolvido exclusivamente para esse fim.

O *software* conta com um medidor de tempo, um editor de mensagens escritas e a própria imagem das câmeras – todos esses componentes foram construídos utilizando o programa *Pure Data*.

### Da arena

A arena (Anexo B) de combates deve ter forma circular como as arenas de luta de sumô. Haverá marcações indicando onde os competidores deverão estar localizados.

Não será permitida a presença do público dentro da arena de embates.

### Do público presente

O público presente poderá acompanhar o embate de seus lugares reservados, localizados na diagonal da arena e do lado oposto ao telão que retransmite o evento.

Ao público é permitido que aposte nos competidores.

\*

## **Visão técnica do trabalho Campeonato mundial de sério**

Campeonato mundial de sério é um trabalho desenvolvido sobre a biblioteca GEM<sup>23</sup> do *Pure Data* (Anexo A, fig.04 e 05).

---

<sup>23</sup> *Graphics Environment for Multimedia*, escrito por Mark Danks, biblioteca de computação gráfica em tempo real. <http://gem.iem.at/documentation/manual/manual/gem-introduction> acessado em 20/8/2010 15:37h.

Basicamente, é um gerenciador de câmera criado a partir da modificação do *patch* de ajuda *[pix\_video]*. Há também um medidor de tempo para as competições, e um letreiro onde se podem escrever mensagens em tempo real para a *gemwin*<sup>24</sup> enquanto o trabalho está sendo executado.

*CÂMERA PLAYER 1 e CÂMERA PLAYER 2*: São os gerenciadores de vídeo. Funcionam graças a uma *box* chamada *[pix\_video]*<sup>25</sup>. Ela é ligada a *[gemhead]* (que define que ela será usada como um objeto GEM) e a uma *box* gráfica chamada *[rectangle]* que delimita o espaço onde a imagem será projetada na tela<sup>26</sup>. Ainda temos um *Hradio* (ver capítulo sobre *GUI objects*), fazendo a função de seletor dos sinais das câmeras.

*MEDIDOR DE TEMPO*: é um cronômetro rústico. Ele funciona transformando os sinais de *[pulse]* em imagens gráficas na *gemwin*. *[pulse]*<sup>27</sup> é um objeto desenvolvido a partir de *[metro]*<sup>28</sup>. Os dois cumprem a mesma função; porém, *pulse* possui maior precisão e a possibilidade de “afinação” durante o processo.

*LETREIRO*: é um *patch* que possibilita serem escritas e enviadas mensagens em tempo real na *[gemwin]*<sup>29</sup>. Ele funciona ligando *[keyup]*, objeto que identifica o botão que está sendo pressionado no teclado ao objeto *[entry]* (objeto semelhante a uma caixa de texto). Os textos digitados dentro de sua área são entendidos como símbolos gráficos pelo objeto *[textoutline]* e enviados para a *gemwin*.

---

<sup>24</sup> *Gemwin* é um objeto que cria a janela onde as ações gráficas irão acontecer.

<sup>25</sup> *pix\_video* é um objeto GEM que permite acesso a componentes de vídeo ligados ao computador como *webcams* e placas de captura.

<sup>26</sup> *pix\_video* pode ser ligado a qualquer forma geométrica; o tamanho dessas formas pode ser definido pelo usuário previamente e redefinido em tempo real.

<sup>27</sup> *Pulse* é um objeto escrito por James MacCartney originalmente para o programa Max.

<sup>28</sup> *[pulse]* e *[metro]* são objetos que trabalham com a medição de tempo, a cada x mil segundos (definido pelo usuário) *[pulse]* e *[metro]* mandam um sinal positivo para o *patch*.

<sup>29</sup> *[gemwin]* é um objeto que cria a janela onde as ações gráficas acontecem.

PD VERSUS e PD TEMPO: são duas mensagens estáticas. Com os dizeres {X} e {medidor de tempo} servem para ajudar a leitura da projeção pelo espectador. Funcionam ligando uma *box de menssagem* um objeto [text2d]<sup>30</sup>.

---

<sup>30</sup> [textoutline] e [text2d] são objetos da biblioteca GEM responsáveis pela criação de textos na *gemwin*.



## CONCLUSÃO

Uma das características marcantes do trabalho com *softwares open source* é o ideal de seus usuários de compartilhar livremente informações com os interessados, fazendo com que cada pessoa se alimente das experiências umas das outras, para corresponder a esse forte caráter ideológico de troca. Após o término desta pesquisa, estará sendo elaborado um projeto de oficina que visa divulgar e fomentar o trabalho artístico em *Pure Data*, para que assim outros possam se alimentar do que foi aprendido nesta pesquisa.

Por fim, gostaríamos de dizer que, como citado acima, *Pure Data* é uma ferramenta em constante atualização, e qualquer tentativa de retratá-lo se mostrara apenas útil momentaneamente; assim, para informações mais recentes sobre o assunto, o leitor pode visitar a página oficial na internet : <http://puredata.info/>.

## BIBLIOGRAFIA

<http://puredata.info/> acessado pela última vez em 13/03/2010 19:19h

MAFFE, CORINTO. *O Software público e a economia dos bens intangíveis* – Texto disponível em: <http://www.cultura.gov.br/site/2008/05/05/o-software-publico-e-a-economia-dos-bens-intangiveis-por-corinto-meffe/> acessado pela última vez em 9/3/2010 23:39h

PUCKETTE, Miller S. *The Theory and Technique of Electronic Music* – Texto disponível em: <http://www-crca.ucsd.edu/~msp/techniques.htm> acessado pela última vez em 13/3/2010 16:59h

DOCTOROW, Cory. *Content* - Texto disponível em <http://ibrahimcesar.com/palestra-sobre-drm-na-microsoft-research/comment-page-1/> acessado pela última vez em 25/02/09 15:28h

FLUSSER, Vilém. *Filosofia da caixa preta - Ensaios para uma futura filosofia da fotografia*. São Paulo: Hucitec, 1985.

ADAMS, Ansel. *A Câmera*. São Paulo: Senac, 2000.

BENJAMIN, Walter. *Magia e técnica, arte e política: ensaios sobre literatura e história da cultura, obras escolhidas*. São Paulo: Brasiliense, 1994.

KRAUSS, Rosalind E. *O Fotográfico*. Trad. Anne Marie Davée. Barcelona: GG, 2002.

Vitamin Ph: *New perspectives in photography*. PHAIDON, 2006.

VIRILIO, Paul. *Guerra e cinema*. São Paulo: Página Aberta, 1993.

Critical Art Ensemble. *Distúrbio eletrônico*. Trad. Leila de Souza Mendes. São Paulo: Conrad Editora do Brasil, 2001.

ARANTES, Priscila. *@rte e mídia: perspectivas da estética digital*. São Paulo: Editora SENAC, 2005.

GUARNACCIA, Matteo. *Provos: Amsterdam e o nascimento da contra cultura*. Trad. Leila de Souza Mendes. São Paulo: Conrad Editora do Brasil, 2001.

BOURRIAUD, Nicolas. *Estética relacional*. São Paulo: Editora Martins Fontes, 2009.

CHRISPINIANO, José. *A guerrilha Surreal*. São Paulo: Conrad Editora do Brasil, 2002.

## Anexo A

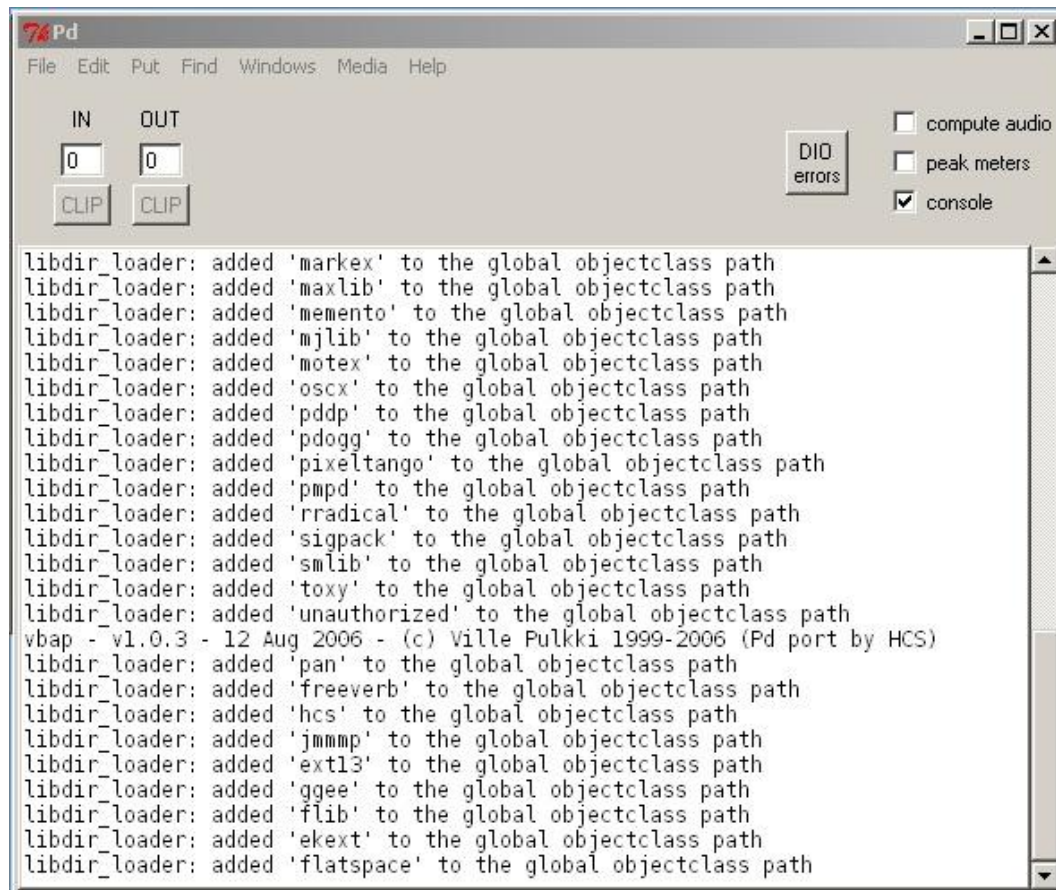


Figura 1 – Janela inicial do *Pure Data*. Fonte: Pereira. (2010).

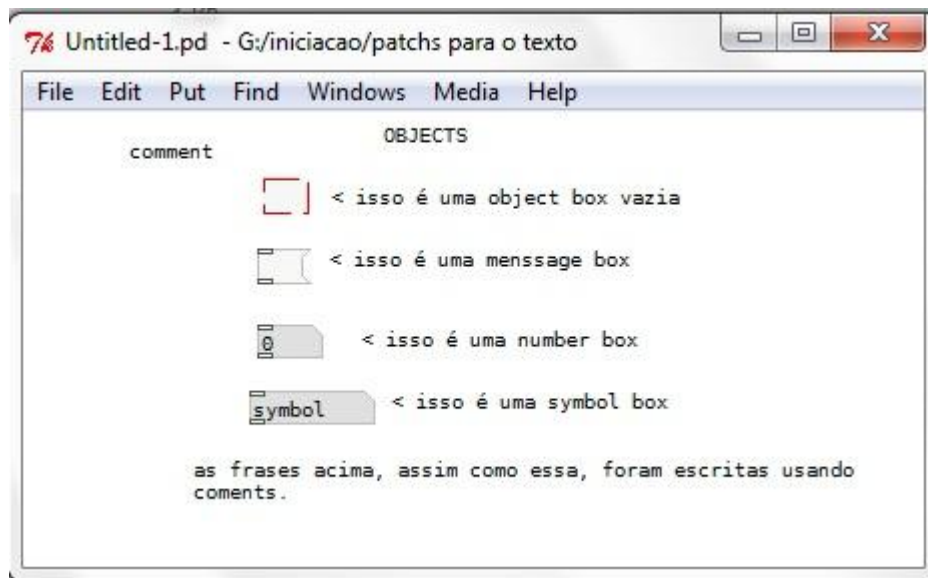


Figura 2 – Objects. Fonte: Pereira.(2009).

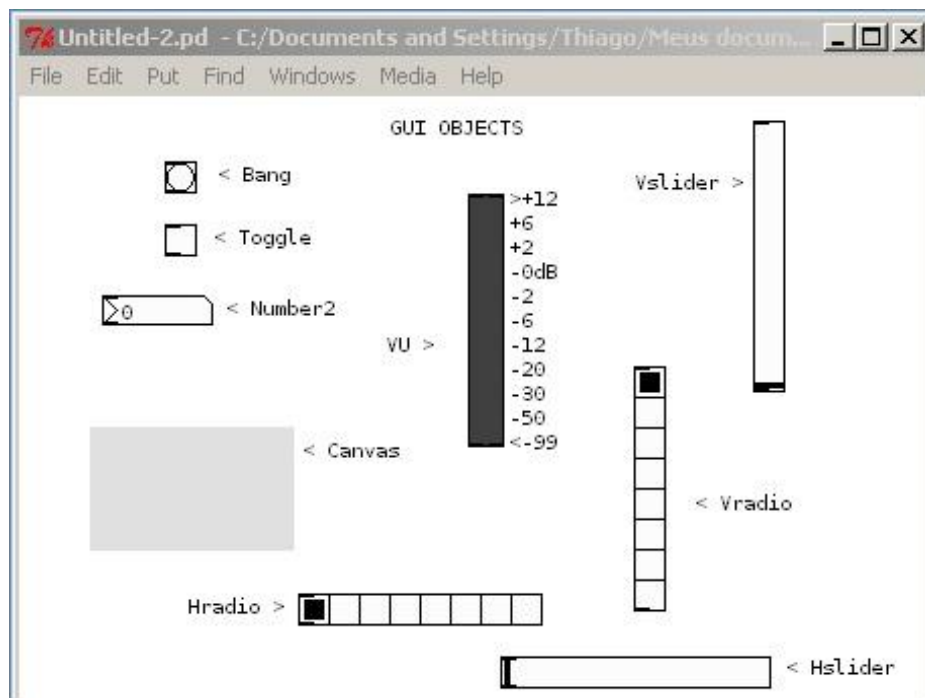


Figura 3 – GUI Objects. Fonte: Pereira(2010).

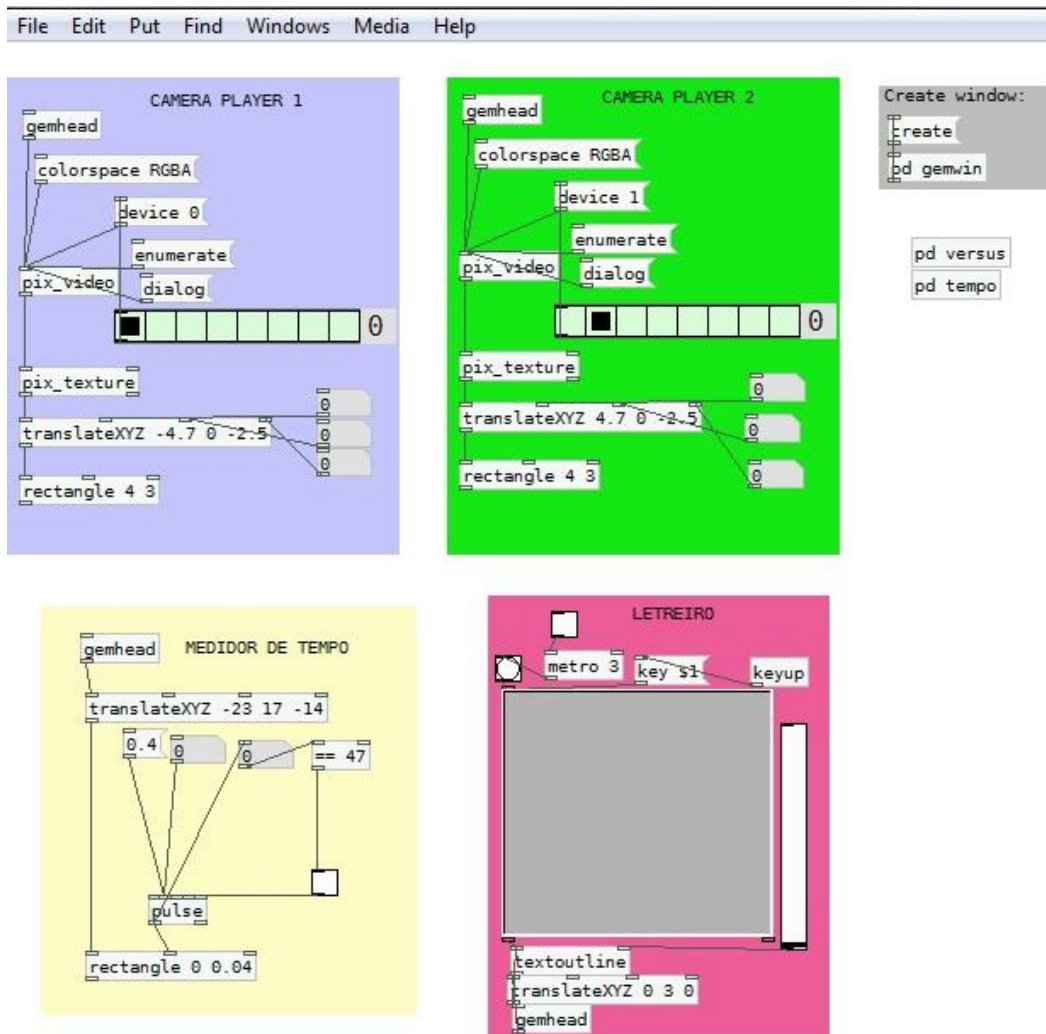


Figura 4 – Patch campeonato mundiao de sério. Fonte: Pereira (2010)

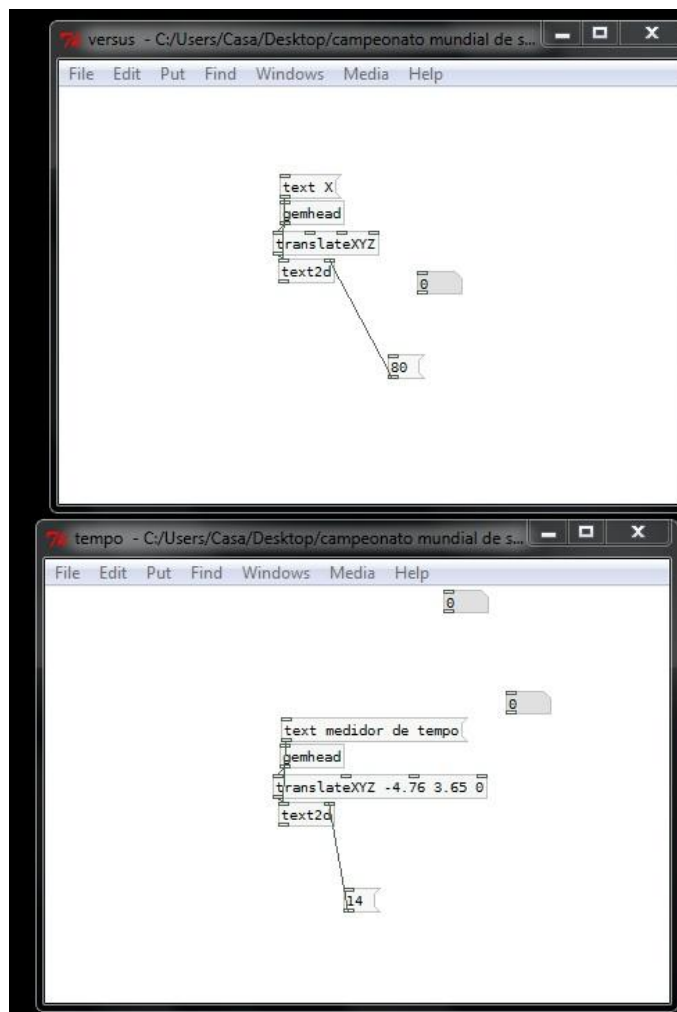


Figura 5 – *Patch* campeonato mundial de sério. Fonte: Pereira (2010)

## Anexo B

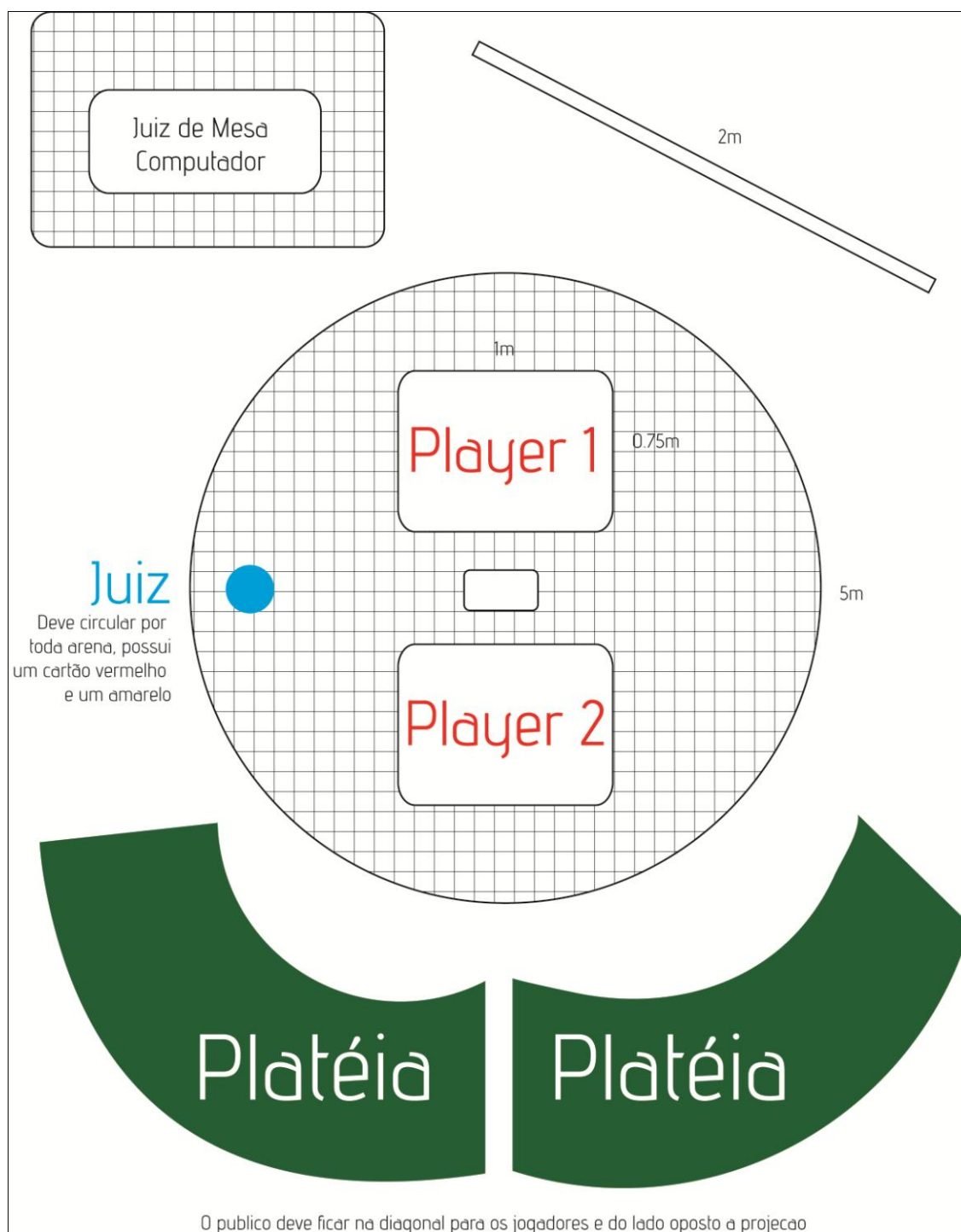


Figura 6 – Croqui arena. Fonte: Pereira (2010)