MUSICANDO, BRINCANDO, PROGRAMANDO

Mestre Alexandre G. Q. Rangel (Universidade de Brasília)

INTRODUÇÃO

Na minha graduação em Artes Plásticas pela Universidade de Brasília (2009), desenvolvi um *software* livre para edição de vídeo ao vivo, no contexto de VJ (Visual Joquei). Batizei o *software* de Quase-Cinema, uma homenagem aos artistas Hélio Oiticica e Neville D'Almeida, que nos anos 60 criaram uma série de obras multimídia intituladas de Quasi-Cinemas. O projeto Quase-Cinema foi contemplado com a Bolsa Funarte de desenvolvimento artístico, quando tive a oportunidade de desenvolver a segunda versão do Quase-Cinema, distribuído como *software* livre. Já no meu mestrado em Artes, com com foco em Arte-Educação, desenvolvi métodos de aplicação do Quase-Cinema para empoderamento e autonomia técnica em criação audiovisual ao vivo. As oficinas do Quase-Cinema foram ministradas no Brasil, Argentina, Espanha, Estados Unidos, Dinamarca e Taiwan. Desde então venho me dedicando à prática de composição audiovisual improvisada, tendo me apaixonado pelo formato de criação e possibilidades criativas da programação ao vivo.

Destaco técnicas e sistemas computacionais e momento mundial de destaque da prática de live coding nas artes, assim como o aparecimento constante de novos sistemas de programação ao vivo, com uso tanto com linguagens textuais quanto de interfaces gráficas de criação.

Em um discurso sobre o fazer artístico usando o computador e suas especificidades, essa pesquisa visa detalhar o meu processo criativo em meios audiovisuais digitais, com foco na produção por meio de criação de programas de computador ao vivo (live coding). A abordagem escolhida para a metodologia foi a realização e análise de oficinas presenciais abertas ao público e a análise de obras criadas no período entre os anos de 2013 e 2019.

Vou discutir os aspectos do live coding sobre a compreensão da linguagem pelo computador e sobre a compreensão por parte do público. Também será levantada a questão da criação paralela em três áreas: música, arte visual e sistema de *software*. Levantarei questões a respeito da necessidade de saber programar *software* para se explorar o processo de criação com live coding, levando em consideração que algumas linguagens de live coding têm, inclusive, um caráter didático, objetivando, além da criação artística, o ensino de conceitos de programação de *software*. O sistema Sonic Pi, a linguagem estudada mais profundamente nesta tese, é direcionada para crianças a partir de 10 anos de idade, sem conhecimento prévio de construção de *software*.

PROGRAMAÇÃO AO VIVO

Antes da invenção da partitura musical, e até mesmo antes da invenção da escrita, grupos se reuniam em situações sociais ou religiosas com a presença de elementos musicais. Desenvolvo técnicas e processos de live coding como ferramenta de composição e performance. Neste formato de produção, a música não é vista como o produto final a ser burilado e refinado e sim como um processo. Até mesmo o fim do processo pode ser considerado como obra aberta, por que costumeiramente, o resultado de uma sessão de programação ao vivo é um sistema vivo, constituído de algoritmos e passível de ser executado ad infinitum ou modificado - pelo autor ou por outras pessoas, se distribuído juntamente com o seu código fonte.

Compreendo o código fonte de programação como desdobramento da partitura musical. Uma

diferença sobre a partitura convencional é a simultaneidade dos processos de composição e instrumento musical de performance.

> "a programação ao vivo consegue mesclar a maioria dos conceitos do discurso musical estabelecido, tais como: compositor, performer e público; instrumento, partitura e obra; composição, performance e improviso; palco e auditório; e instrumento e ferramenta" (MAGNUSSON, 2014, p. 7)

A composição e distribuição da música ocidental desde a Idade Média se baseou no suporte da partitura. Os experimentos de criação com partituras visuais foi uma etapa importante no que culminou no processo estudado nesta tese: o da criação, descrição e distribuição de música como linhas de código de programação de *software*. A descrição de sons por meio de desenhos é um nível de abstração intermediário, nem mais simples nem mais complexo - somente diferente - do que a descrição de sons com notas em uma partitura ou fazendo parte de um algoritmo de programação computacional.

Desde o advento dos computadores pessoais e *softwares* de criação sonora digital, as fronteiras entre composição, registro de peça musical e performance vem se fragmentando e estes campos cada vez se entremeiam mais. Nos primórdios na automatização do processo de composição musical estava presente um jogo de dados, uma espécie de passatempo em eventos sociais, para criação musical em grupos, seja de músicos ou de leigos.

No final do século XVIII, a Europa viu o aparecimento de mais de vinte jogos para criação musical por leigos. Alguns jogos usavam dados, outros simplesmente pediam aos participantes que inventassem números. Em 1789 Mozart criou um jogo, com dados, para composição musical, hoje existe uma versão desse jogo para sistema operacional iOS e uma versão para navegadores de internet. A receita é simples (tacar dois dados 16 vezes e copiar notas de uma tabela), mas podem criar mais de 45 milhões de bilhões de combinações de composições com as regras de Mozart!

Nos sistemas de live coding, uma grande quebra de paradigma de composição é fuga do padrão vigente em composição musical assistida por computador no qual pode-se ver a obra visualmente, organizada em forma de uma linha do tempo, normalmente com várias trilhas, uma para cada instrumento.

Um algoritmo nada mais é do que uma ou mais instruções de procedimentos. O mundo da arte contemporânea já apresentou muitos sistemas dessa natureza, tal como obras do grupo Fluxus. A obra "Composição 1960 # 10", de Bob Morris é um exemplo emblemático, se resumindo a uma frase somente: "Desenhe uma linha reta e siga-a".

Os algoritmos são caixas pretas, mas passíveis de modificações por meio de intervenções. A complexidade é alcançada por meio da fusão de partes mais simples, variáveis e modificação constante de trechos de código fonte. Uma questão importante na prática de live coding, onde a partitura tem a forma de texto, é a compreensão desses textos. Ao se observar uma sessão de programação ao vivo, pode-se pensar, em um primeiro momento, que o código é - somente - um conjunto de instruções a serem seguidas pela máquina. Ao se perceber estilos de construção e palavras chave conhecidas (sejam elas em inglês ou na língua nativa do público), vai-se tomando do processo criativo do artista. Vai-se ganhando percepção do fluxo de processo de pensamento e argumentação com a máquina.

A não-previsibilidade é um ponto essencial no tipo de interação que acontece entre seres humanos e máquinas durante processo criativo de programação ao vivo. É uma mescla entre trabalho intelectual e emocional, uma mescla entre aquisição de controle e se deixar levar pelo fluxo. Oposição entre uma visão instrumentalista, onde o performer deve ser um virtuoso com um instrumento, com uma visão mais intelectual, onde o performer tenta fazer a máquina funcionar criativamente, domando as dificuldades técnicas, das mais abrangentes, como a escolha dos *softwares* e equipamentos para o seu ecosistema pessoal, como as mais minuciosas, com a escolha de comandos e variáveis que irão

trabalhar dentro de um algoritmo específico. Observa-se aqui um diálogo com a música concreta, que trouxe de liberdade ao artista criador sonoro. O compositor usa mais de atributos de inventividade do que de uso da teoria musical.

SUBVERSÕES

A etapa anterior ao uso de uma linguagem de programação é o design da linguagem propriamente dita. É um campo onde a engenhosidade e estilo dos inventores define quais os padrões de pensamento e de expressão dos programadores que irão utilizar os sistemas. Algumas implementações de sistemas são "puramente" brincadeiras, não tendo objetivos práticos de solução de problemas, mas chamam atenção para as decisões de desenho de um ambiente de programação.

A linguagem Piet funciona não com comandos de texto, mas recebendo do programador uma imagem semelhante aos quadros de Piet Mondrian. A imagem neste sistema não é o resultado do programa, e sim o formato de seus comandos.

Na linguagem Cow, cujo nome significa "vaca", todos os comandos são variações de onomatopeias de mugidos do ruminante (Moo, MOo, moO, mOo...). O visual de um programa se assemelha a uma poesia concreta ou a uma partitura de canto minimalista.

O sistema Gertrud funciona da seguinte forma: o programa pode ser escrito com frases em qualquer língua (português, inglês, francês). Para cada frase é calculada uma média do tamanho das palavras. As palavras que não se encaixam na média de tamanho de cada frase são usadas como índices relacionados às funções de processamento do programa (cálculos matemáticos, operações de fluxo do programa, criação de variáveis). O resultado estético do programa é um texto que o autor da linguagem considera similar ao estilo da escritora Gertrud Stein. Aos olhos de quem não sabe o que está examinando o programa é um texto em prosa, sem nenhuma semelhança com o que se associa normalmente a um programa de computador tal como a presença constante de números, sinais matemáticos e pedaços isolados da língua inglesa (IF, THEN, ELSE...).

O sistema Arnoldc utiliza frases de filmes com o ator Arnold Schwarzenegger (Hasta la vista, baby!) e o sistema Shakespeare formata a programação como uma peça de William Shakespeare. Já a linguagem Pikachu tem comandos baseados nos sons emitidos pelo personagem homônimo, do desenho animado Pokémon: pi, ka, pika, chu, pikapi...

Um dos exemplos mia significativos de uso "subversivo" de ferramentas de programação é a prática de Poesia Perl, que engloba a criação de programas na linguagem Perl, com sintaxes que, quando lidas, se traduzem em poesia, conforme o exemplo a seguir, ganhador do Prêmio de Poesia Perl de 2000:

Perl	if ((light eq dark) && (dark eq light)
	&& ($blaze_of_night{moon} == black_hole)
	&& ($ravens_wing{bright} == $tin{bright})){
	my $*love* = $you = $sin{darkness} + 1;
	};

Inglês	If light were dark and dark were light The moon a black hole in the blaze of night A raven's wing as bright as tin Then you, my *love*, would be darker than sin
Português	Se a luz fosse escura e escura fosse a luz, A lua, um buraco negro no brilho da noite, A asa de um corvo brilhante como estanho, Então você, meu amor, seria mais negra que o pecado.

Tabela 1: Poesia de Angie Winterbottom, escrita na linguagem Perl.

Venho desenvolvendo uma série de trabalhos intitulados Esculturas Quânticas, envolvendo a criação de "sistemas escultóricos" com propriedades sonoras e visuais, em forma de móbiles digitais - em forma mutante - animadas.. Ao utilizar na minha produção o Sonic Pi para o controle do *software* de criação visual Blender 3D, também estou subvertendo o uso inicialmente pensado pelo seu desenvolvedor. Isso traz desafios novos e resultados inesperados muito interessantes à exploração artística.

Por que chamá-las de esculturas? O processo de construção é uma investigação quadridimensional (três dimensões espaciais mais a dimensão temporal) sobre a materialidade possível de cada peça. Por que nomeá-las de quânticas? Pelas infinitas possibilidades derivadas dos algoritmos de modelagem, dos processos randômicos e das possíveis interações - com o artista e com o público.

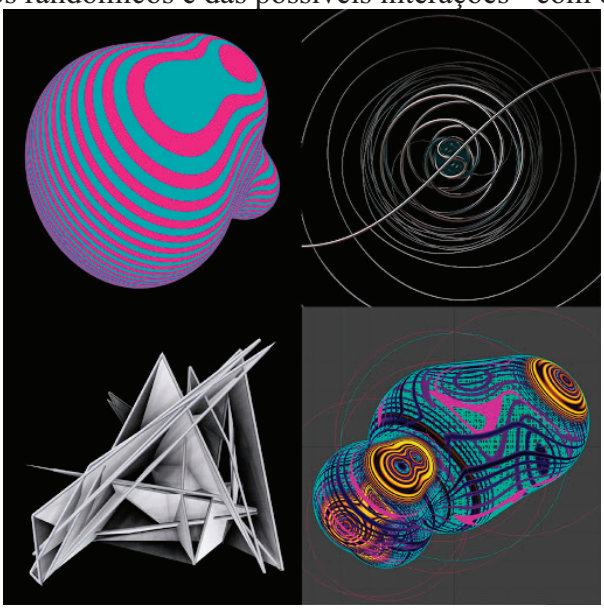

Figura 1: frames de quatros cenas da série "Esculturas Quânticas".

Ao ver os primeiros resultados da fusão entre modelagem, efeitos geométricos e live coding, imagino determinados momentos das animações das formas resultantes como possíveis congelamentos e existência independente da imagem em movimento. Nesse sentido, o processo

de criação de cada “sistema” busca o quesito de materialidades possíveis. Uma inspiração estética sempre pairando no ar durante as etapas de preparação das obra foi o trabalho de Alexander Calder com suas esculturas-móbiles. Compreendo essa pesquisa como uma interpretação contemporânea do conceito de escultura dinâmica.

CONSIDERAÇÕES FINAIS

A expectativa com esta pesquisa baseada em produção artística é a descoberta continuadas de fluxos de trabalho e ferramentas de expressão audiovisual, levando em consideração os aspectos técnicos e poéticos emergentes do processo.

REFERÊNCIAS

MAGNUSSON, Thor. 2014. Scoring with Code: Composing with algorithmic notation. In: Organised Sound, 19. Cambridge: Cambridge University Press, 2014. 268-2