INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DE SANTA CATARINA – CAMPUS FLORIANÓPOLIS
DEPARTAMENTO ACADÊMICO DE ELETRÔNICA
CURSO SUPERIOR DE ENGENHARIA ELETRÔNICA

CLEISSON FERNANDES DA SILVA

DESENVOLVIMENTO DE UMA SUPERFÍCIE SINTETIZADORA DE ÁUDIO COM INTERFACE TANGÍVEL PARA USUÁRIO

FLORIANÓPOLIS, 2020

INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DE SANTA CATARINA – CAMPUS FLORIANÓPOLIS
DEPARTAMENTO ACADÊMICO DE ELETRÔNICA
CURSO SUPERIOR DE ENGENHARIA ELETRÔNICA

CLEISSON FERNANDES DA SILVA

DESENVOLVIMENTO DE UMA SUPERFÍCIE SINTETIZADORA DE ÁUDIO COM INTERFACE TANGÍVEL PARA USUÁRIO

Trabalho de conclusão de curso submetido ao Instituto Federal de Educação, Ciência e Tecnologia de Santa Catarina como parte dos requisitos para obtenção do título de engenheiro em eletrônica.

Orientador:
Prof. Fernando Santana Pacheco, Dr.
Coorientador:
Prof. Renan Augusto Starke, Dr.

FLORIANÓPOLIS, 2020

Ficha de identificação da obra elaborada pelo autor.

Silva, Cleisson Fernandes da
Desenvolvimento de uma superfície sintetizadora de áudio com interface tangível para usuário / Cleisson Fernandes da Silva ; orientação de Fernando Santana Pacheco; coorientação de Renan Augusto Starke. - Florianópolis, SC, 2020.
65 p.
Trabalho de Conclusão de Curso (TCC) - Instituto Federal de Santa Catarina, Câmpus Florianópolis. Bacharelado em Engenharia Eletrônica. Departamento Acadêmico de Eletrônica.
Inclui Referências.

1. Sintetizador. 2. Interface tangível. 3. Sistema embarcado. 4. Instrumento musical. I. Pacheco, Fernando Santana. II. Starke, Renan Augusto. III. Instituto Federal de Santa Catarina. Departamento Acadêmico de Eletrônica. IV. Título.

DESENVOLVIMENTO DE UMA SUPERFÍCIE SINTETIZADORA DE ÁUDIO COM INTERFACE TANGÍVEL PARA USUÁRIO

CLEISSON FERNANDES DA SILVA

Este trabalho foi julgado adequado para obtenção do Título de Engenheiro Eletrônico e aprovado na sua forma final pela banca examinadora do Curso de Engenharia Eletrônica do Instituto Federal de Educação, Ciência e Tecnologia de Santa Catarina

Florianópolis, 17/12/2019

Banca Examinadora:

Fernando Santana Pacheco, Dr.

Renan Augusto Starke, Dr.

Alexandre Leizor Szczupak, Dr.

Robinson Pizzio, Dr.

RESUMO

Este trabalho apresenta o desenvolvimento de um sintetizador de áudio com interface tangível. O usuário pode interagir com a aplicação movendo objetos físicos que representam propriedades definidas. Foi desenvolvida uma arquitetura para a interação entre os objetos tangíveis e implementadas interfaces gráficas para a exibição de informações e interação com o usuário. A aplicação utilizou a biblioteca *openFrameworks* e o reconhecimento dos objetos tangíveis é realizado pelo *reacTIVision*. Um protótipo da superfície tangível foi construído utilizando-se a tecnologia de superfície de toque baseada em óptica. Desta forma, o usuário pode tocar na tela para configurar e modificar parâmetros. Na aplicação foram implementados efeitos como filtros, *reverb* e *chorus*. Além disso, é possível gerar sons através de formas de ondas senoidais, quadradas e dente de serra, ou por um acervo de amostras de instrumentos musicais. A aplicação possui também um sequenciador onde é possível selecionar o andamento e as notas a serem tocadas. Por fim, após a integração da aplicação desenvolvida com a superfície tangível implementada, a execução do programa apresentou a taxa de atualização de quadros média de 20 quadros por segundo e latência para o movimento dos objetos tangíveis insignificante.

Palavras chave: Sintetizador, Interface tangível, Sistema embarcado, Instrumento musical.

ABSTRACT

This work presents the development of a tangible user interface audio synthesizer. The user can interact with the application by moving physical objects that represent defined properties. A framework that allows the interaction between tangible objects was developed. Graphical interfaces were implemented for information display and user interaction. The application uses the openFrameworks library and the recognition of tangible objects is performed using reacTIVision. A tangible tabletop prototype was constructed using optics-based touch surface technology. In this way, the user can touch the screen to set and modify parameters. In the application we implemented effects such as filters, reverb and chorus. In addition, it is possible to generate sound through waveforms, which are sine, square and sawtooth, or by a collection of musical instruments samples. The application also has a sequencer where one can select the tempo and notes to play. Finally, after integrating the developed application with the implemented tangible interface tabletop, the program execution presented the average frame refresh rate of 20 frames per second and insignificant latency for the movement of the tangible objects.

Keywords: Synthesizer, Tangible user interface, Embedded system, Musical instrument.

LISTA DE FIGURAS

SUMÁRIO

1 INTRODUÇÃO

1.1 O QUE É UM SINTETIZADOR?

Um sintetizador é um instrumento musical que cria som através do processo chamado de síntese sonora. Este processo pode reutilizar sons existentes processando-os ou gerando o próprio som eletronicamente ou mecanicamente (RUSS, 2009).

Nas últimas décadas, a palavra “sintetizador” passou a significar apenas um instrumento eletrônico que é capaz de produzir uma ampla gama de sons diferentes. As categorias reais de sons que se qualificam para esse rótulo de sintetizador também são muito específicas: sons puramente imitativos são frequentemente vistos como nada além de gravações do instrumento real; nesse caso, o sintetizador é visto como pouco mais do que um dispositivo de repetição. Em outras palavras, o público em geral parece esperar que os sintetizadores produzam sons artificiais. À medida que os sintetizadores se tornam melhores na fusão de elementos de sons reais e sintéticos, os limites do que é considerado “sintético” e do que é “real” estão se tornando cada vez mais difusos (RUSS, 2009).

Seguindo esta mesma linha de pensamento sobre a síntese sonora, Russ (2009) faz a seguinte observação:

> A síntese sonora reúne arte e ciência em uma mistura de habilidade musical e conhecimento técnico. Utilizado com cuidado, pode produzir performances emocionais, que pintam paisagens sonoras com um rico e enorme conjunto de timbres, limitados apenas pela imaginação e conhecimento do criador. Os sons podem ser simples ou complexos, e os métodos usados para criá-los são diversos. A síntese sonora não se preocupa apenas com timbres sofisticados gerados por computador, embora esse seja geralmente o aspecto mais divulgado. Mas a tecnologia nada mais é do que um conjunto de ferramentas que podem ser usadas para produzir sons: as habilidades criativas do intérprete, músico ou técnico ainda são essenciais para evitar que a música se torne mundana. (p. 4, tradução do autor).

1.2 JUSTIFICATIVA

Antigamente, os sintetizadores eram instrumentos musicais de grandes proporções que podiam ocupar o espaço de um piano de cauda. Com o avanço da tecnologia, seu tamanho foi sendo reduzido até caber dentro de um simples teclado musical portátil. Nos dias atuais, os sintetizadores são utilizados extensivamente em

conjunto com softwares para produção musical chamados de estações de trabalho de áudio digital. Estes softwares utilizam *plug-ins* de sintetizadores que se baseiam nos mesmos conceitos dos sintetizadores físicos. Apesar das facilidades trazidas pelo uso de sintetizadores em software, estes possuem tantas possibilidades de configurações com menus, potenciômetros, barras deslizantes e botões (um exemplo é mostrado na Figura 1) que um iniciante neste mundo musical pode se sentir intimidado e avesso a aprender tudo isso. Além disso, esta área musical possui uma grande quantidade de jargões e conhecimentos específicos que dificultam ainda mais a aprendizagem.

Figura 1 – Interface visual do *Ableton Live*

Fonte: Página do *Ableton Live*[1] (2019).

Existem interfaces alternativas que auxiliam na aprendizagem sobre sintetizadores e trazem uma abordagem mais didática deste meio. Um exemplo disto é o *reacTable* (KALTENBRUNNER *et al.*, 2006), um instrumento de música eletrônica que combina uma interface de superfície tangível com conceitos ou técnicas como síntese modular, programação visual e *feedback* visual.

A síntese modular baseia-se no princípio de unidades especializadas separadas, ou seja, cada unidade possui uma função específica e independente. Essas unidades, geralmente chamadas de módulos, podem ser configuradas e

[1] Disponível em: https://www.ableton.com/en/live/. Acesso em nov. 2019.

conectadas livremente para atender a qualquer necessidade musical. Essa liberdade oferece infinitas possibilidades sonoras.

O *reacTable* é baseado nos princípios da síntese modular. Portanto, é também uma ferramenta perfeita para mostrar esses princípios a uma pessoa que está aprendendo. A mesma aprende quais são as formas de onda básicas da síntese, como elas soam e também como os filtros podem ser usados para alterar a qualidade do som. Os princípios básicos do sintetizador podem ser ensinados mostrando os efeitos da alteração das frequências de geradores e filtros de som. Geiger *et al.* (2010) afirmam que o conhecimento sobre a geração de som com sintetizadores faz parte da música moderna, assim como o conhecimento sobre instrumentos tradicionais faz parte da música nos últimos séculos.

1.3 OBJETIVO GERAL

O presente trabalho tem como objetivo a implementação e desenvolvimento de uma superfície sintetizadora de áudio com interface tangível que possui uma estrutura física semelhante à do *reacTable*.

1.4 OBJETIVOS ESPECÍFICOS

Os objetivos específicos citados a seguir representam as etapas necessárias para realização do projeto:

- Estudo das tecnologias disponíveis para o desenvolvimento de uma superfície com interface tangível.
- Escolha dos componentes para a montagem da superfície tangível.
- Implementação e montagem da superfície tangível.
- Escolha do *framework* para o desenvolvimento da aplicação de sintetizador.
- Definição da arquitetura da aplicação de sintetizador.
- Desenvolvimento da aplicação de sintetizador.
- Validação do protótipo.

2 FUNDAMENTAÇÃO TEÓRICA

Este capítulo discorre sobre os elementos teóricos e as definições estudadas e recorridas para o desenvolvimento do projeto. Na seção 2.1, é introduzido o novo conceito de interface de usuário que será utilizado. A seção 2.2 discorre sobre o funcionamento do *reacTable,* o projeto em que a superfície sintetizadora de áudio com interface tangível está sendo baseada. Na seção 2.3, é apresentada a tecnologia de superfície de toque baseada em óptica, necessária para a interação com a superfície através de toques. A seção 3 trata sobre o funcionamento e a escolha de cada componente utilizado neste projeto.

2.1 INTERFACE TANGÍVEL

Uma interface tangível é uma interface de usuário na qual uma pessoa interage com informações digitais através do ambiente físico. Interfaces tangíveis vinculam suas representações à aparência física (ULLMER e ISHII, 1998).

De acordo com Ishii e Ullmer (1997), as interfaces tangíveis fornecem um acoplamento muito mais próximo entre o físico e o digital na medida em que a distinção entre entrada e saída se torna cada vez mais desfocada. Os autores exemplificam que ao se utilizar um ábaco não há distinção entre “entrada” de informação e sua representação e que esse tipo de mistura é previsto pela computação tangível.

Ishii e Ullmer (1997) conceberam as interfaces tangíveis para serem destinadas a aumentar o mundo físico real, acoplando informações digitais a objetos e ambientes físicos cotidianos, permitindo literalmente aos usuários apreender dados com as mãos, fundindo assim a representação e o controle de dados e operações digitais com objetos físicos. Em vez de ter controles físicos, dados digitais e lógica separados, com entrada e saída separadas, as interfaces tangíveis criaram uma representação e controle físico e digital mais tangíveis e acoplados. Desse modo, apenas os dados são puramente digitais.

2.1.1 Interfaces tangíveis *versus* interfaces gráficas

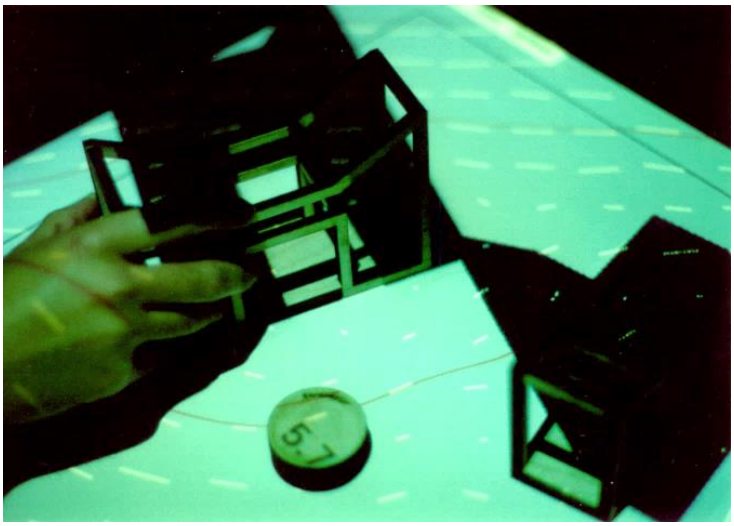
Com as GUIs (*Graphical User Interfaces*), os usuários interagem com as informações digitais selecionando representações gráficas (ícones, janelas etc.) com

um dispositivo apontador (sendo o mais comum o *mouse)*, enquanto as interfaces tangíveis do usuário combinam controle e representação em um único dispositivo físico (ULLMER e ISHII, 2000). Desse modo, as interfaces tangíveis enfatizam a tangibilidade e materialidade, a incorporação física de dados, a interação corporal e a incorporação de sistemas em espaços e contextos reais.

2.1.2 Aplicações

Um exemplo de aplicação com o uso de interface tangível é o Urp (UNDERKOFFLER e ISHII, 1999), sistema para planejamento urbano. Ele permite que os usuários interajam com simulações de fluxo de vento e luz solar através da colocação de modelos e ferramentas de construções físicas sobre uma superfície. Como mostrado na Figura 2, os modelos de construção tangíveis geram sombras (digitais) que são projetadas na superfície. O fluxo de vento simulado é projetado como linhas na superfície. Várias ferramentas tangíveis permitem que os usuários controlem e alterem o modelo urbano. Por exemplo, os usuários podem examinar a velocidade do vento, alterar as propriedades do material dos edifícios (paredes de vidro ou pedra) e alterar a hora do dia. Tais mudanças afetam as sombras digitais projetadas e a simulação do vento (SHAER e HORNECKER, 2010).

Figura 2 – Demonstração da interface tangível do Urp



Fonte: (SHAER e HORNECKER, 2010).

As interfaces tangíveis podem beneficiar o aprendizado por vários motivos. Kim e Maher (2008) estudaram o efeito de uma interface desse tipo comparada com uma interface de usuário gráfica na cognição espacial de *designers* e os resultados mostraram que a manipulação de objetos físicos em 3D melhorou a percepção das relações espaciais e apoiou a exploração e descoberta de diferentes soluções de *design*. Desse modo, os autores concluíram que a utilização de uma interface tangível reduz a carga cognitiva e permite uma imersão mais criativa no problema.

Schneider *et al.* (2011) investigaram como uma interface tangível poderia influenciar uma tarefa de resolução de problemas em comparação com uma interface multitoque e concluíram que a primeira pode ser particularmente adequada para tarefas de aprendizagem colaborativa em comparação com interfaces gráficas clássicas, fornecendo um espaço de trabalho compartilhado e facilitando a visualização das ações entre os usuários da interface tangível.

De acordo com Shaer e Hornecker (2010), um grande número de interfaces tangíveis podem ser classificadas como ferramentas ou ambientes de aprendizado apoiado por computador. Além do mais, os autores afirmam que os ambientes de aprendizagem física que utilizam uma interface tangível envolvem todos os sentidos e portanto podem apoiar o desenvolvimento geral, por exemplo, de uma criança.

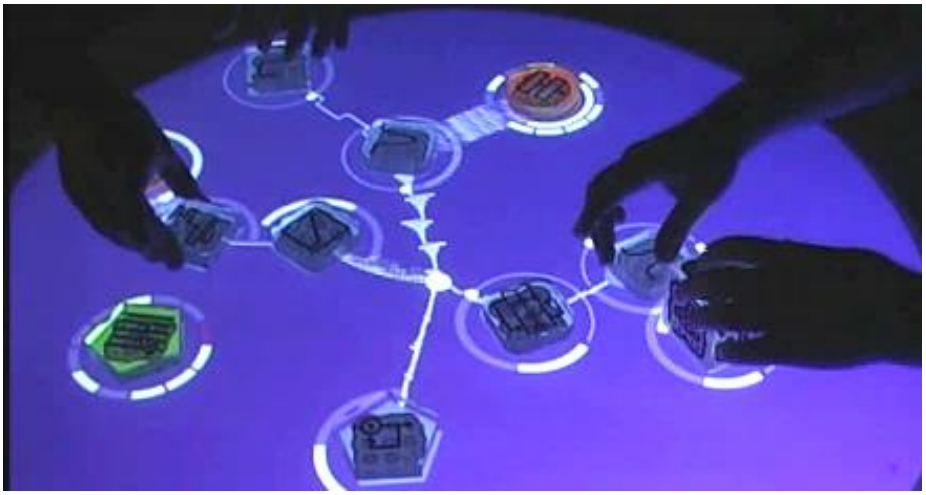
Segundo Julià (2015), desde a concepção de superfícies com interface tangíveis por Ishii e Ullmer (1997), os pesquisadores atribuem intuitivamente qualidades positivas a esse tipo de interface. Além disso, Julià afirma que a interação tangível mudou substancialmente a forma como as pessoas podem se relacionar fisicamente com os computadores.

2.2 REACTABLE

Aplicações para música são uma das áreas mais antigas e populares para interfaces tangíveis. Um exemplo dessa aplicação é o reacTable (JORDÀ *et al.*, 2005).

O reacTable é um instrumento musical eletroacústico multiusuário com uma superfície de interface tangível. Como mostra a Figura 3, vários artistas podem compartilhar simultaneamente o controle sobre o instrumento movendo objetos físicos na superfície enquanto constroem diferentes topologias de áudio em um tipo de sintetizador modular tangível (KALTENBRUNNER *et al.*, 2006).

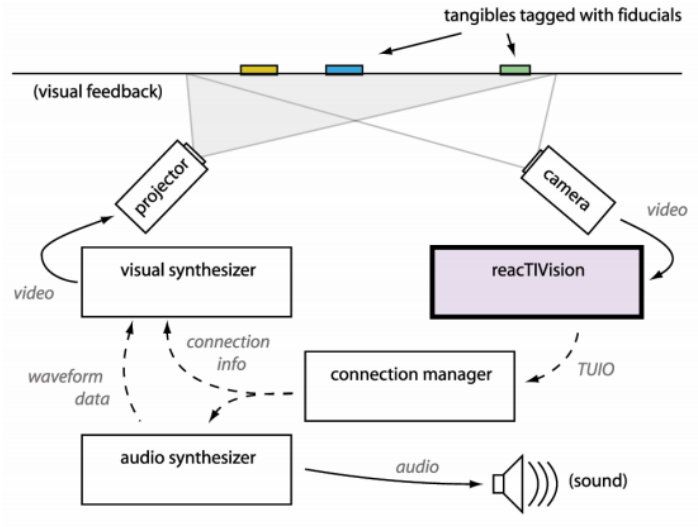
Figura 3 – Demonstração da interface tangível do reacTable



Fonte: (KALTENBRUNNER *et al.*, 2006, p. 4).

O *hardware* do instrumento é constituído por uma mesa redonda translúcida. Como mostrado na Figura 4, uma câmera de vídeo situada abaixo da superfície da mesa analisa continuamente esta superfície, acompanhando a natureza, posição e orientação dos objetos que são distribuídos em sua superfície. Os objetos tangíveis, que são representações físicas dos componentes de um sintetizador modular clássico, são passivos, sem nenhum sensor ou atuador; os usuários interagem movendo-os, alterando sua posição ou orientação. Essas ações controlam diretamente a estrutura topológica e os parâmetros do sintetizador de som. Um projetor, também por baixo da superfície, desenha animações dinâmicas em sua superfície, fornecendo um *feedback* visual do estado, da atividade e das principais características dos sons produzidos pelo sintetizador de áudio (KALTENBRUNNER *et al.*, 2006).

Figura 4 – Arquitetura do reacTable



Fonte: (KALTENBRUNNER *et al.*, 2006).

Um conjunto simples de regras conecta e desconecta automaticamente esses objetos, de acordo com seu tipo, afinidade e proximidade com os outros vizinhos. As topologias sonoras resultantes são representadas permanentemente na mesma superfície da mesa por um sintetizador gráfico responsável pelo *feedback* visual, como mostra a Figura 3. Projeções ao redor dos objetos físicos trazem informações sobre seu comportamento, seus parâmetros, valores e estados de configuração, enquanto as linhas que desenham as conexões entre os objetos, transmitem as formas de onda reais do fluxo sonoro sendo produzido ou modificado em cada nó (KALTENBRUNNER *et al.*, 2006).

Cada objeto do reacTable possui uma função dedicada à geração, modificação ou controle do som. Movendo-os e aproximando-os, os artistas constroem e tocam o instrumento ao mesmo tempo. Como a movimentação de qualquer objeto pela superfície da mesa pode alterar as conexões existentes, é possível obter topologias de sintetizador extremamente variáveis, resultando em um ambiente altamente dinâmico (KALTENBRUNNER *et al.*, 2006).

Geiger (2010), um dos criadores do reacTable, afirma que aprender um idioma não é principalmente ler e escrever, mas se expressar, ouvir e entender. O mesmo é válido para a música, especialmente nos dias de hoje, onde as ferramentas para criá-la são muito mais fáceis de serem acessadas e a sua criação não se restringe aqueles que são capazes de tocar um instrumento. O autor também afirma que o reacTable mostra como a música é gerada e tratada nos computadores, e ensina o que os

músicos modernos sabem sobre a mesma, sem ter que aprender um instrumento. Além disso, o autor aponta que tocar o reacTable tem mais a ver com criatividade musical do que manipular um instrumento, pois o treinamento mecânico não é mais necessário.

2.3 SUPERFÍCIES MULTITOQUE

Muller (2008) define superfícies multitoque como dispositivos gráficos interativos que utilizam tecnologias táteis para manipulação direta na tela. Estas superfícies combinam a tecnologia de exibição com sensores capazes de rastrear múltiplos pontos de entrada. O autor afirma que a ideia é permitir aos usuários interagir com o computador de maneira natural.

Atualmente, existem diversas técnicas para a detecção de toques em superfícies, como a resistiva, a capacitiva e a óptica. No entanto, apenas a óptica, que foi a tecnologia utilizada, será descrita neste trabalho.

2.3.1 Superfícies de toque baseadas em óptica

As abordagens ópticas para multitoque usam o processamento de imagem para determinar a localização e a natureza das interações com a superfície. Esses sistemas geralmente usam iluminação infravermelha e, devido à sua configuração simples, podem ser muito robustos (SCHÖNING *et al.*, 2008).

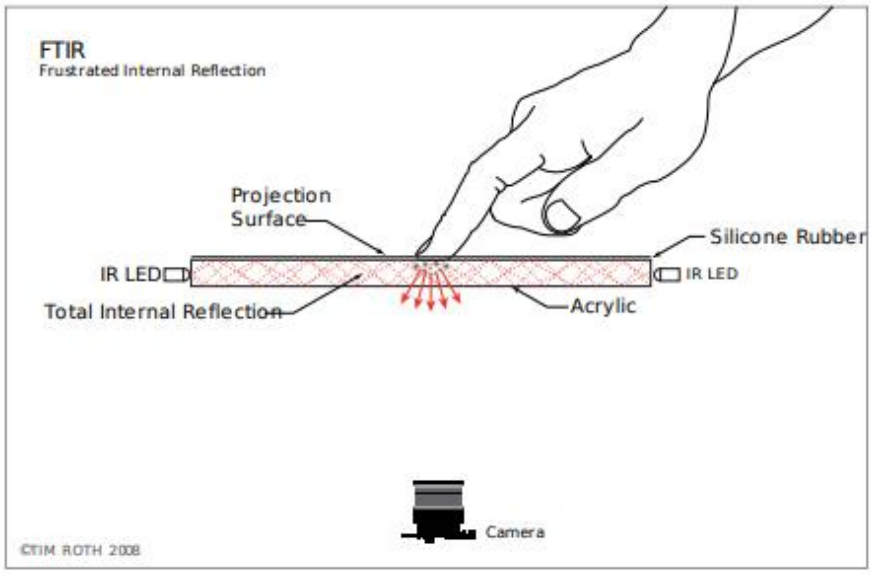
2.3.1.1 Reflexão interna total frustrada

A tecnologia FTIR (*Frustrated Total Internal Reflection)* é baseada na reflexão interna total óptica dentro de uma superfície interativa. Quando a luz encontra uma interface para um meio com um índice mais baixo de refração (por exemplo, vidro para o ar), a luz é refratada até certo ponto que depende do seu ângulo de incidência e, a partir de um certo ângulo crítico, sofre uma reflexão interna total. As fibras ópticas, os tubos de luz solar e outros guias de onda ópticos aproveitam esse fenômeno para transportar a luz com eficiência, com pouquíssimas perdas. No entanto, outro material na interface pode *frustrar* essa reflexão interna total, fazendo com que a luz escape deste guia de ondas (HAN, 2005).

De acordo com Schöning *et al.* (2008), a redescoberta do princípio da reflexão interna total frustrada realizada por Han (2005) pode ser vista como um ponto de partida para sistemas ópticos multitoque.

As configurações comuns de FTIR têm um painel de acrílico transparente com uma moldura de LEDs (*Light Emitting Diodes*) ao redor injetando luz infravermelha. Quando o usuário toca no acrílico, a luz escapa e é refletida no ponto de contato do dedo devido ao seu maior índice de refração; uma câmera sensível ao espectro infravermelho na parte traseira do painel pode captar claramente esses reflexos. Um conjunto básico de algoritmos de visão computacional é aplicado à imagem da câmera para determinar a localização do ponto de contato. A configuração geral de um sistema FTIR é ilustrada na Figura 5 (SCHÖNING *et al.*, 2008).

Figura 5 – Configuração geral de um sistema FTIR



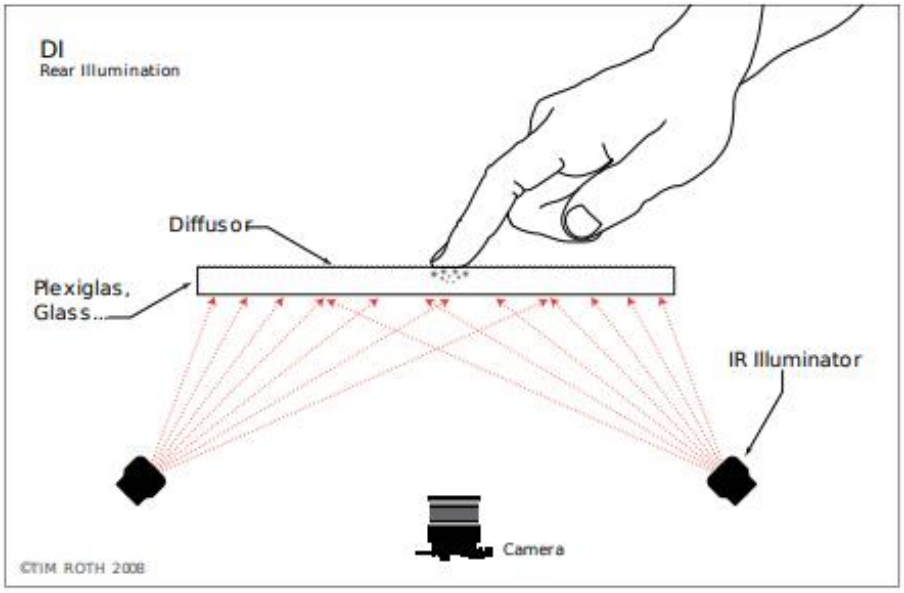
Fonte: (SCHÖNING *et al.*, 2008).

2.3.1.2 Iluminação difusa

O hardware dos sistemas de iluminação difusa é semelhante ao do FTIR. Ambas as técnicas empregam um projetor e uma câmera sensível a infravermelho atrás de uma superfície de projeção. No entanto, para DI (*Diffuse Illumination*), a iluminação infravermelha também é colocada atrás dessa superfície. Isso faz com que a área na frente da superfície seja brilhantemente iluminada no espectro infravermelho

como ilustrado na Figura 6. Consequentemente, a câmera capta todos os objetos nessa área refletindo a luz infravermelha. Isso inclui objetos próximos e objetos que tocam a superfície. A detecção de toque explora o fato de que a superfície da projeção difunde a luz, desfocando as imagens de objetos à distância. Ao contrário do FTIR, o DI permite rastrear e identificar objetos e dedos. Os objetos podem ser identificados usando sua forma ou fiduciais (marcadores visuais reconhecidos por algoritmos de visão computacional) impressos em suas superfícies inferiores. Além disso, qualquer superfície transparente (como vidro de segurança) pode ser colocada entre a tela de projeção e o usuário, pois o sensor não depende do contato da superfície (SCHÖNING *et al.*, 2008).

Figura 6 – Configuração geral de um sistema DI



Fonte: (SCHÖNING *et al.*, 2008).

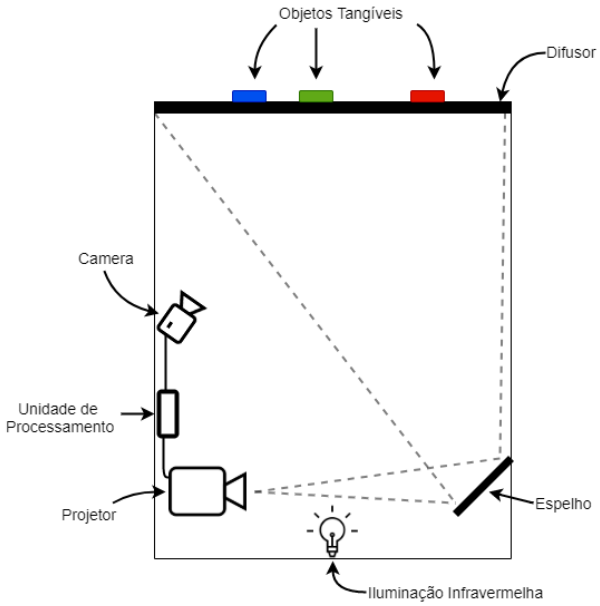
3 IMPLEMENTAÇÃO DE UMA SUPERFÍCIE TANGÍVEL

Esta seção aborda os elementos necessários para a implementação de uma superfície com interface tangível. Os elementos estão divididos em *hardware*, para os componentes físicos utilizados, e *software*, para as bibliotecas e *frameworks* utilizados. A seção 3.3 faz a recapitulação de todos os elementos escolhidos.

3.1 HARDWARE

Uma superfície com interface tangível para usuário é composta por diversos componentes. Nesta seção são analisados e determinados os principais componentes pertinentes à construção da superfície tangível. A Figura 7 apresenta um esquema geral da superfície proposta neste trabalho, destacando os principais componentes.

Figura 7 – Visão esquemática da superfície tangível construída



Fonte: Elaborado pelo autor.

3.1.1 Iluminação

Segundo Schöning *et al.* (2008), as tecnologias para a detecção de toques, com exceção da baseada em óptica, exigem fabricação de qualidade industrial e, portanto, não são adequadas para projetos especiais e personalizados. Com base nisso, a tecnologia baseada em óptica foi a escolhida para ser implementada. Dentre as

possíveis configurações para esta tecnologia, a configuração de iluminação difusa (DI) foi escolhida para ser usada devido ao seu suporte ao uso de fiduciais.

Conforme evidenciado por Kaltenbrunner e Bencina (2007), em um sistema de projetor e câmera, os dois componentes visuais precisam operar em diferentes faixas espectrais para que não interfiram entre si. Como o projetor obviamente precisa operar na faixa visível, a câmera precisa trabalhar apenas no espectro infravermelho. Desse modo, a superfície precisa ser iluminada com luz infravermelha forte e difusa, que é completamente invisível aos olhos e, portanto, não interfere na projeção. Um exemplo de fontes de luz adequadas são matrizes de LEDs infravermelhos disponíveis em diferentes intensidades. Portanto, para a implementação da superfície, que conta com o uso de projetor, foi utilizada uma matriz de LEDs infravermelhos.

Para os casos em que nenhuma projeção é necessária, a instalação pode operar no espectro visível, simplificando significativamente o processo de iluminação.

3.1.2 Processamento

Com o objetivo de criar uma superfície tangível que fosse portátil e independente de conexão a servidores externos, foi escolhida como plataforma de processamento o *Raspberry Pi 3 Model B*, apresentada na Figura 8.

O desenvolvimento de um sistema embarcado utilizando o *Raspberry Pi* traz as vantagens de menor custo, quando comparado a um computador de mesa, espaço e consumo de energia.

Figura 8 – Foto com vista superior da placa Raspberry Pi 3 Model B

Fonte: Raspberry Pi Foundation[2] (2020).

[2] Disponível em: https://www.raspberrypi.org/products/raspberry-pi-3-model-b. Acesso em jan. 2020.

3.1.3 Câmera

Ambas as configurações FTIR e DI necessitam de câmera para detectar os dedos que estão tocando a superfície. As câmeras geralmente contêm um filtro para bloquear a luz infravermelha do ambiente. Essa camada de filtro deve ser removida e, em alguns casos, foi projetada para ser destacável (SCHÖNING *et al.*, 2008).

De acordo com Schöning *et al.* (2008), é necessário aplicar filtros de passagem de luz infravermelha à câmera para filtrar toda a luz visível, principalmente a gerada pelo projetor, pois a imagem projetada sobrepõe e interfere nos símbolos fiduciais. Para um desempenho ideal, deve ser um filtro passa-banda que corresponda ao comprimento de onda infravermelha dos LEDs.

Para satisfazer essa necessidade, foi utilizada a câmera *Raspberry Pi NoIR Infrared Camera Board* v1.3 (Figura 9) e, para reduzir a interferência da iluminação ambiente, um filtro de passagem de banda infravermelha.

Figura 9 – Raspberry Pi NoIR Camera Board

Fonte: Raspberry Pi Foundation [3] (2020).

3.1.4 Projeção

Em uma superfície tangível, geralmente é utilizado um projetor para se obter um *feedback* visual. Isso permite que a aplicação se comunique com o usuário e assim gere uma interação mais natural.

As duas configurações principais para a montagem de um projetor são a projeção frontal ou traseira. Na projeção frontal um único projetor é montado ao longo do eixo normal da tela. Os usuários entre o projetor e a tela produzirão sombras. Essa é uma configuração semelhante à maioria dos projetores montados no teto nas salas

[3] Disponível em: https://www.raspberrypi.org/products/pi-noir-camera-v2. Acesso em jan. 2020.

de conferências. Uma alternativa é utilizar a projeção traseira. Ao usar um único projetor montado atrás da tela, essa solução evita por completo oclusões e sombras, mas requer espaço dedicado extra para o caminho do feixe de luz. Este método é menos afetado por outras fontes de luz e, como a projeção passa pela tela em vez de ser refletida por ela, a luz ambiente não afeta o contraste tanto quanto comparado à projeção frontal.

Como a superfície tangível requer que o usuário interaja com objetos presentes diretamente na tela de projeção, a configuração de projeção traseira é a mais adequada.

Segundo Kaltenbrunner e Bencina (2007), se a câmera ou o projetor não tiver uma lente que resulte em um campo de visão grande o suficiente, colocar um espelho na mesa ajuda a obter uma superfície ativa maior, mantendo uma altura relativamente baixa da superfície.

De acordo com Muller (2008), ao selecionar um projetor, é importante decidir qual a resolução necessária. Dependendo do tipo de aplicação em execução, é recomendável usar uma resolução de pelo menos 1024×768 *pixels*. Qualquer que seja o tipo de projetor, *Digital Light Processing* (DLP) ou *Liquid Crystal Display* (LCD), é importante avaliar o contraste e a quantidade de lúmens (brilho) que cumpram com os requisitos necessários.

O projetor utilizado foi o PJ-Q72 da Exbom (Figura 10). Este projetor cumpriu os requisitos de campo de visão, distância de foco e brilho da projeção. Suas especificações estão apresentadas no Quadro 1.

Figura 10 – Foto com vista frontal do Projetor PJ-Q72

Fonte: next-Trading[4] (2020).

[4] Disponível em: http://www.next-trading.com/index/products/detail?id=3620. Acesso em fev. 2020.

Quadro 1 – Especificações do Projetor PJ-Q72

Propriedade	Valor
Resolução	800x480
Brilho	1.200 lúmens
Tecnologia	LCD
Contraste	1.000:1

Fonte: Adaptado de Mercado Livre[4] (2019).

3.1.5 Difusor

Para a configuração de projeção traseira é necessário que a tela para a projeção seja uma superfície difusa e translúcida. Em uma superfície desse tipo, apenas objetos que estão próximos o suficiente podem ser vistos na imagem da câmera, enquanto os mais distantes permanecem invisíveis para que o software de rastreamento possa detectar objetos apenas em contato com a superfície e não objetos que estão flutuando acima dela. Isso é muito útil para detectar o contato com os dedos, pois, ao tocar a superfície, estes aparecem como pequenos círculos enquanto ocultam o restante da mão (JULIÀ, 2015).

O difusor utilizado foi papel vegetal sobre uma moldura de vidro. A principal vantagem foi a nitidez dos elementos fiduciais quando em contato com a superfície deste tipo de papel.

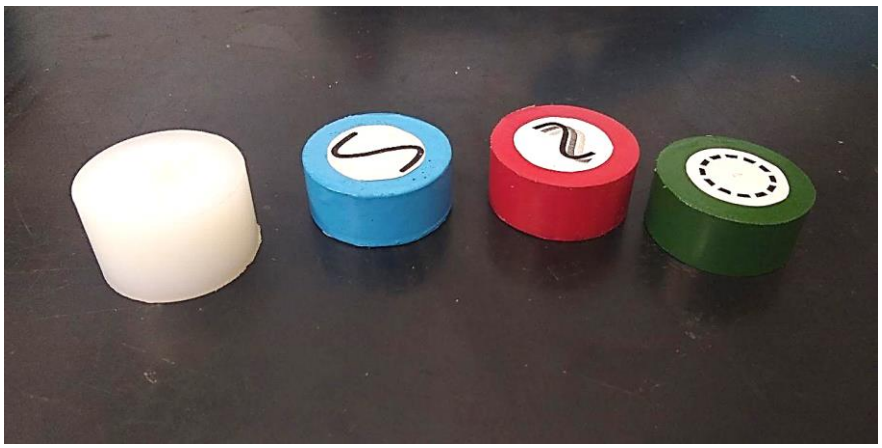
3.1.6 Objetos tangíveis

Segundo Kaltenbrunner e Bencina (2007), quase qualquer objeto, incluindo formas geométricas simples de madeira ou plástico, objetos do cotidiano e até alimentos ou vegetais pode ser transformado em um componente de interface tangível facilmente rastreável, anexando-lhe um marcador fiducial. Idealmente, o símbolo deve ser afixado na parte inferior do objeto para ocultá-lo da atenção do usuário e também para evitar possíveis problemas de oclusão das mãos. O conjunto de símbolos fiduciais pode ser impresso com uma impressora laser em papel branco comum para escritório.

Para criar os objetos tangíveis (exemplos mostrados na Figura 11), foi utilizado um tarugo de *nylon* cortado em cilindros com altura de aproximadamente 2,5 cm. O cilindro mais à esquerda da Figura 11 mostra como os mesmos ficaram após serem

cortados. Os cilindros apresentavam uma coloração branca e, para serem utilizado na superfície tangível, foram pintados de acordo com suas funções (definidas na aplicação desenvolvida). Na face inferior de cada um foi colado um símbolo fiducial e na parte superior, um ícone que representa sua função.

Figura 11 – Demonstração dos objetos tangíveis



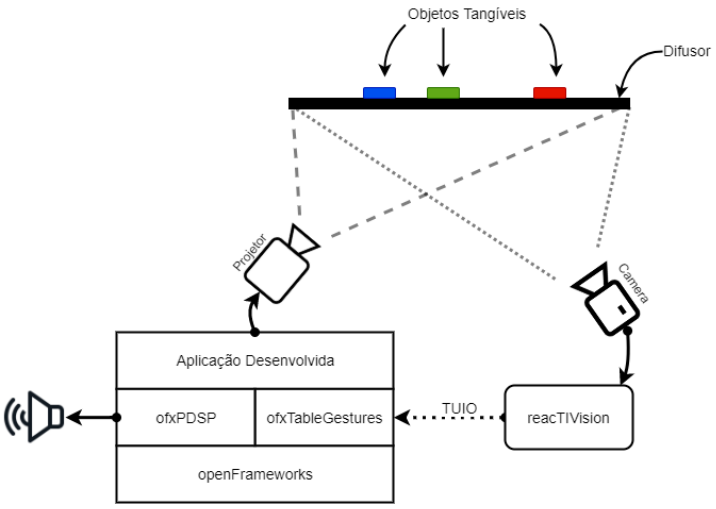
Fonte: Elaborado pelo autor.

3.2 SOFTWARE

Nessa seção é detalhada cada biblioteca selecionada que foi utilizada na implementação da superfície tangível, como ilustrado na Figura 12.

Foi utilizado o *reacTIVision* para o processamento de imagem e a aplicação foi desenvolvida utilizando a biblioteca *openFrameworks* em conjunto com seus *add-ons*.

Figura 12 – Arquitetura das bibliotecas utilizadas



Fonte: Elaborado pelo autor.

3.2.1 reacTIVision

O *reacTIVision* é um *framework* de código aberto multiplataforma de visão computacional projetado principalmente para a construção de superfícies com interface tangível para usuários (KALTENBRUNNER e BENCINA, 2007). Este *framework* foi desenvolvido com o intuito de ser o principal componente sensorial para o *reacTable* (JORDÀ *et al.*, 2005).

O *reacTIVision* obtém imagens da câmera, procura na transmissão de vídeo quadro a quadro símbolos fiduciais e envia dados sobre todos os símbolos identificados através de um *socket* UDP (protocolo de comunicação entre processos) de rede para um aplicativo de escuta (KALTENBRUNNER e BENCINA, 2007).

3.2.1.1 Fiduciais

Fiduciais são marcadores visuais que foram projetados de modo especial para que possam ser anexados a objetos físicos. Os marcadores são reconhecidos e rastreados por um algoritmo de visão computacional otimizado para o design específico destes marcadores, melhorando a velocidade geral e a robustez do processo de reconhecimento (KALTENBRUNNER e BENCINA, 2007).

O mecanismo fiducial mais recente e mais confiável é o de ameba, desenvolvido por Kaltenbrunner e Bencina (2007). Exemplos destes fiduciais se

encontram na Figura 13. Os autores obtiveram as geometrias destes fiduciais utilizando um algoritmo genético que otimizou a aparência do fiducial utilizando um conjunto de funções de ajuste de curva que visava a forma, tamanho, precisão do ponto central e do ângulo de rotação. Ao todo, estão disponíveis 216 fiduciais diferentes para serem utilizados.

Figura 13 – Exemplos de fiduciais amebas

Fonte: (KALTENBRUNNER, 2009).

Segundo Kaltenbrunner e Bencina (2007), nesse mecanismo, o quadro da imagem de origem é primeiro convertido em uma imagem em preto e branco com um algoritmo de limiar adaptativo. Essa imagem é então segmentada em um grafo de adjacência de região, refletindo a estrutura de contenção de regiões preto e branco alternadas. Este grafo é pesquisado por estruturas de árvores únicas, que são codificadas nos símbolos fiduciais. Finalmente, as árvores identificadas são correspondidas a um dicionário para recuperar números únicos de identificação do marcador. A posição de um símbolo ameba é calculada como o centroide de todos os nós das folhas encontrados. A orientação do marcador é calculada como o vetor do centroide do marcador ao centroide de todas as folhas pretas distribuídas na parte superior do símbolo.

3.2.1.2 Protocolo de comunicação TUIO

O *reacTIVision* define seu próprio protocolo de comunicação, TUIO, projetado especificamente para as necessidades de superfícies tangíveis, codificando e transmitindo os atributos de objetos encontrados na superfície da mesa. Para fornecer comunicação rápida e confiável com aplicativos clientes locais e remotos, o protocolo cria uma estrutura de mensagens redundante sobre a camada de transporte UDP. TUIO define um conjunto de mensagens do protocolo *Open Sound Control* (OSC).

Essas mensagens transmitem constantemente a presença, posição e ângulo de todos os símbolos encontrados, além de outros parâmetros derivados. No lado do cliente, essas mensagens são decodificadas para adicionar, atualizar e remover eventos genéricos correspondentes às ações físicas que foram aplicadas a cada objeto tangível (KALTENBRUNNER e BENCINA, 2007).

3.2.2 openFrameworks

openFrameworks é uma biblioteca para linguagem de programação C++, de código aberto, projetada para ser utilizada por artistas e designers que trabalham com design interativo e arte em mídia, fornecendo uma estrutura simples e intuitiva (NOBLE, 2009). O *openFrameworks* agrupa diversas bibliotecas comumente utilizadas para desenhos gráficos, áudio, tipografia e visão computacional.

Devido à sua facilidade, desempenho e suporte à multiplataforma, foi a biblioteca selecionada para se desenvolver a interface da aplicação.

3.2.3 ofxTableGestures

ofxTableGestures[5] é um *framework*, desenvolvido por Carles F. Julià (2015) e Daniel Gallardo, para a criação de aplicações de superfícies com interface tangível. Este *framework* é um *add-on* para *openFrameworks*, ou seja, uma extensão para a biblioteca.

O *framework* possui reconhecimento de gestos multiusuário, ou seja, reconhece gestos relacionados ao uso de uma superfície tangível (como, por exemplo: toque simples, toque duplo, deslizar, etc.) para mais de um usuário ao mesmo tempo. Isso se torna interessante para uma gama de aplicações pois permite que várias pessoas possam utilizar a superfície simultaneamente e que as ações de cada uma não interfiram com as das outras.

Devido ao suporte a superfícies com interface tangível, este *framework* foi escolhido para prover a interação com o usuário na aplicação desenvolvida.

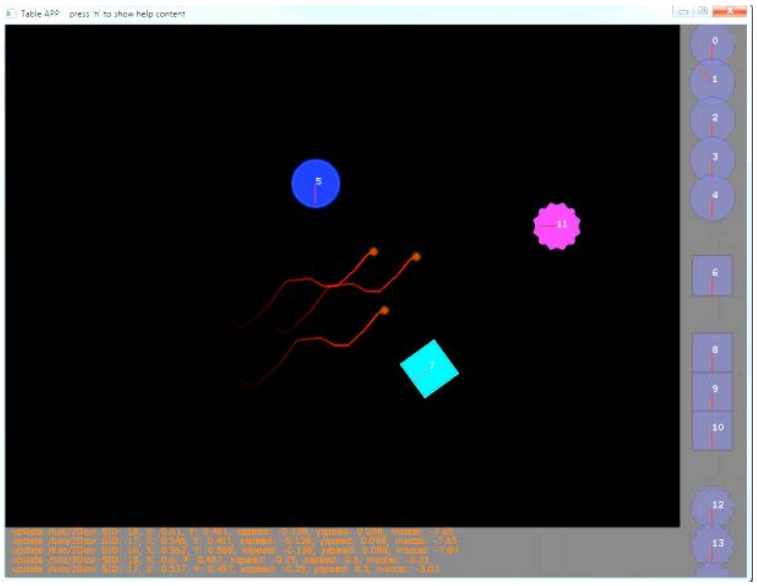
[5] https://github.com/chaosct/ofxTableGestures/

3.2.3.1 Simulação

O *ofxTableGestures* possui simulação em *software* para auxiliar o desenvolvimento. Devido a essa funcionalidade, foi possível testar a aplicação sem a necessidade de executá-la na superfície tangível montada.

Quando o simulador está ativado, como mostrado na Figura 14, um painel com um subconjunto de figuras é mostrado no lado direito da tela. Essas figuras são identificadas com o identificador que será transmitido por mensagens utilizando o protocolo TUIO para o sistema e tem várias formas configuráveis (círculo, quadrado, estrela, etc.) (JULIÀ, 2015).

Figura 14 – Exemplo de uso do simulador embutido



Fonte: (JULIÀ, 2015).

3.2.3.2 Adaptação para Linux embarcado

O *ofxTableGestures* foi desenvolvido com o intuito de ser executado nas principais plataformas de computadores (OS X, Windows, Linux) e não para sistemas embarcados (no caso, Linux embarcado para ARM usado na Raspberry Pi). Portanto, para a utilização desse *add-on* foi necessário adaptar o código para a sua aplicação neste sistema.

O único elemento conflitante foi em relação ao uso de funções do OpenGL, pois o sistema operacional utilizado no *Raspberry Pi* (*Raspbian Stretch*) utiliza OpenGL ES.

OpenGL ES (*OpenGL for Embedded Systems*) é uma extensão para a biblioteca de computação gráfica OpenGL. Ela é projetada para ser utilizada em sistemas embarcados (celulares, tablets, etc.) e portanto possui algumas divergências ao OpenGL (KHRONOS GROUP, 2019).

Após as adaptações, o *framework* compilou com sucesso e teve funcionamento idêntico ao original.

3.2.4 ofxPDSP

Como a aplicação a ser desenvolvida necessita de manipulação e geração de sons, é necessária a utilização de uma biblioteca que forneça essas funcionalidades. O *openFrameworks*, por ser uma biblioteca voltada ao desenvolvimento criativo visual e musical, possui suporte a essa finalidade. Para esse fim, foi utilizado o *ofxPDSP*.

O *ofxPDSP*[6] é um *add-on* para o *openFrameworks* voltado para síntese de áudio, música generativa e processamento digital de sinais. Este *add-on* possui diversos módulos e classes prontos e disponíveis para serem usados, como por exemplo: osciladores, filtros, equalizadores, etc.

3.3 RESUMO GERAL DA ESTRUTURA DE HARDWARE E SOFTWARE

Os componentes escolhidos para a montagem da superfície tangível estão apresentados no Quadro 2. Já o Quadro 3 apresenta as bibliotecas escolhidas para o desenvolvimento da aplicação de sintetizador.

[6] https://github.com/npisanti/ofxPDSP/

Quadro 2 – Componentes escolhidos para a montagem da superfície tangível

Função	Componente Escolhido
Iluminação	Iluminador infravermelho
Processamento	Raspberry Pi 3
Câmera	Raspberry Pi NoIR Infrared Camera Board v1.3
Projeção	PJ-Q72
Difusor	Papel Vegetal
Objetos Tangíveis	Cilindro de Nylon

Fonte: Elaborado pelo autor.

Quadro 3 – Bibliotecas escolhidas para o desenvolvimento da aplicação

Função	Software Escolhido
Rastreamento de dedos e fiduciais	reacTIVision
Interface gráfica	openFrameworks
Reconhecimento de gestos para superfícies tangíveis	ofxTableGestures
Processamento de Áudio	ofxPDSP

Fonte: Elaborado pelo autor.

4 DESENVOLVIMENTO DA APLICAÇÃO

Este capítulo aborda o desenvolvimento da aplicação de sintetizador para ser utilizada na superfície tangível implementada. A seção 4.1 introduz as necessidades da aplicação e a arquitetura estabelecida. A seção 4.2 lista e descreve os componentes concebidos para serem utilizados na interface gráfica da aplicação. Por fim, a seção 4.3 discorre sobre cada tipo de objeto tangível estabelecido e desenvolvido.

4.1 ARQUITETURA DO SOFTWARE

O *software* desenvolvido deve abstrair a interação do usuário com a superfície tangível e ser responsivo com as conexões de áudio. O usuário pode adicionar, remover ou mover os objetos disponíveis para serem usados na superfície tangível e a superfície deve ser responsiva com o *feedback* visual onde é possível obter informações sobre as propriedades do objeto e sobre suas conexões.

Utilizando o *ofxTableGestures*, o *software* desenvolvido recebe as informações dos fiduciais e dedos reconhecidos pelo *reacTIVision* através do protocolo TUIO (*id*, posição, ângulo, etc.) e então gera determinados eventos. O módulo de eventos no *openFrameworks* possui algumas classes e funções que possibilitam ao usuário trabalhar com eventos. Os eventos permitem que um ouvinte receba uma notificação sempre que algo acontecer em outra parte do código ou no hardware.

Os seguintes eventos estão disponíveis:

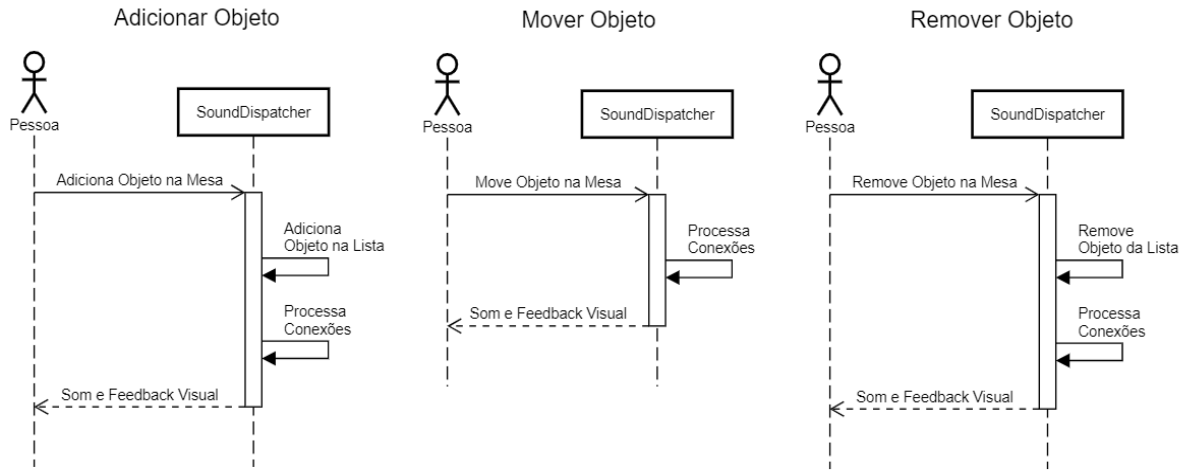
- Eventos relacionados aos Objetos Fiduciais:
    - newObject;
    - removeObject;
    - updateObject;
    - enterObject;
    - exitObject.

- Eventos relacionados ao Multitoque:
  - removeCursor;
  - newCursor;
  - updateCursor;
  - enterCursor;
  - exitCursor.
- Eventos extras relacionados ao Multitoque:
  - Tap;
  - LongPush.

O programa possui uma classe chamada *SoundDispatcher*, responsável pelo gerenciamento de todos os objetos tangíveis presentes e possíveis de serem colocados na superfície tangível. Esta classe é inicializada com uma lista de objetos tangíveis representados pela classe *TableObject.*

A classe *SoundDispatcher* está cadastrada nos eventos relacionados aos objetos tangíveis e, portanto, quando o usuário da superfície tangível coloca, remove ou move um objeto tangível, o evento em questão é acionado e as funções da classe *SoundDispatcher* são chamadas. Essas interações estão ilustradas na Figura 15.

Figura 15 – Interação com a classe *SoundDispatcher*

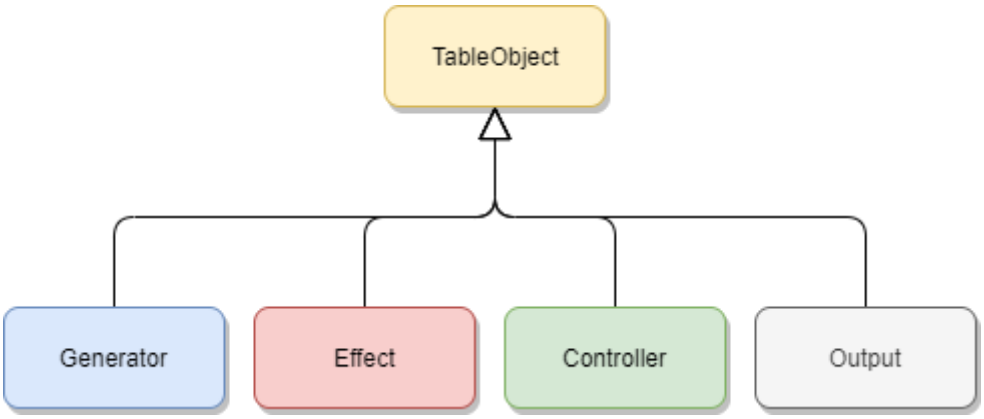


Fonte: Elaborada pelo autor.

A classe *TableObject* é uma classe abstrata e serve apenas como interface para um objeto tangível. A implementação do objeto (ou seja, como ele se comporta)

é feita em classes derivadas da classe *TableObject*. A hierarquia está apresentada na Figura 16.

Figura 16 – Hierarquia da classe *TableObject*



Fonte: Elaborada pelo autor.

As possíveis derivações para *TableObject* são:

- ***Generator***: para objetos que geram ou produzem sons.
- ***Effect***: para objetos que modificam ou alteram os sons.
- ***Controller***: para objetos que controlam ou modificam as propriedades de outros objetos.
- ***Output*****:** para ser usado unicamente como saída do som.

4.1.1 Algoritmo de roteamento

As conexões entre os objetos tangíveis são atualizadas toda vez que ocorre um novo evento relacionado aos objetos fiduciais. As conexões são gerenciadas pelo próprio objeto, ou seja, a classe *TableObject* mantém uma variável que diz com qual objeto ela está conectada.

Para realizar as conexões, foi desenvolvido um algoritmo próprio, apresentado na forma de pseudocódigo no Algoritmo 1. O mesmo faz as conexões baseado nas distâncias entre os objetos.

Algoritmo 1 – Algoritmo de conexão entre os objetos tangíveis

```
INÍCIO
  PARA todos os objetos presentes na superfície
    Procura os dois objetos mais próximos do objeto atual;
    SE existe um objeto mais perto
      SE existe um segundo objeto mais perto
        SE o segundo mais perto está conectado no primeiro mais perto
          Dist01 ← distância do atual para o primeiro;
          Dist02 ← distância do atual para o segundo;
          Dist21 ← distância do segundo para o primeiro;
          SE (Dist21 > Dist01) E (Dist02 > Dist21)
            SE o segundo pode se conectar ao primeiro
              Conecta o segundo no primeiro;
        SE o primeiro mais perto não possui conexão
          Conecta no primeiro mais perto;
    SENÃO
      SE este objeto pode se conectar no primeiro mais perto
        Conecta no primeiro mais perto;
FIM
```

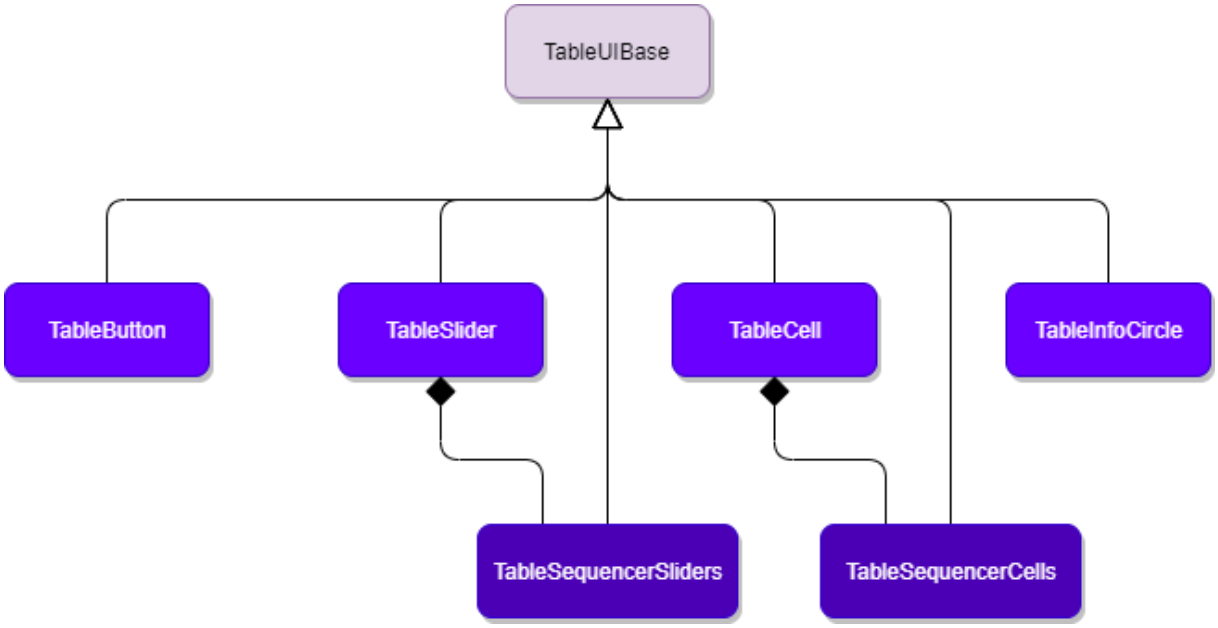
Fonte: Elaborado pelo autor.

4.2 INTERFACE GRÁFICA

A interface gráfica da superfície tangível exerce um papel fundamental na interação dos usuários. É através da interface projetada que os usuários podem obter informações sobre o estado, conexões e propriedades dos objetos tangíveis presentes na superfície.

Todos os componentes gráficos descendem de uma classe-base chamada *TableUIBase*, como ilustrado na Figura 17. Esta classe está cadastrada em todos os eventos relacionados ao multitoque na superfície tangível. No entanto, diferente dos eventos relacionados aos objetos tangíveis, quando uma notificação de multitoque é acionada, apenas o componente que está em colisão com algum ponto do multitoque recebe o evento.

Figura 17 – Hierarquia da classe *TableUIBase*



Fonte: Elaborada pelo autor.

Foram desenvolvidos diversos componentes gráficos primitivos e simples para prover uma interação mais intuitiva. Estes componentes são:

- TableButton.
- TableSlider.
- TableCell.
- TableInfoCircle.

A partir dos componentes primitivos, foi possível agregá-los e desenvolver novos componentes com funcionalidades estendidas:

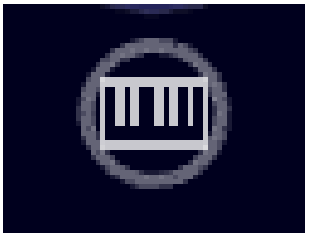
- TableSequencerSliders.
- TableSequencerCells.

4.2.1 TableButton

Este componente representa um botão “clicável”. Com este componente, é possível interagir através de uma simples ação de toque na superfície. Além da interação com o toque simples, ao deixar o dedo em um mesmo lugar em cima do botão, é gerado um evento de toque longo. Portanto, é possível interagir com este componente através destes dois métodos.

Os componentes TableButtons possuem em comum o círculo cinza exterior, como mostrado na Figura 18. No entanto, o desenho central (neste caso um símbolo que representa uma oitava de um teclado musical) pode ser diferente para cada um, dependendo do objeto e do menu atual. Além disso, para os casos em que há menus diferentes de configurações para o objeto tangível, é possível alterar a imagem para simbolizar o menu atual.

Figura 18 – Exemplo de componente *TableButton*



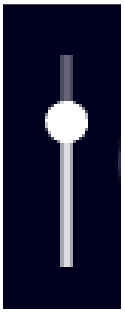
Fonte: Elaborada pelo autor.

4.2.2 TableSlider

Este componente representa um controle deslizante. Ele permite ajustar um parâmetro em um intervalo de valores pré-definidos quando o usuário desliza o dedo sobre ele. Os valores podem ser discretos, ou seja, podem apenas assumir valores inteiros. Além disso, existe a opção de mostrar o valor numérico atual na parte superior do componente.

O *TableSlider* (Figura 19) tem o desenho de uma linha com um círculo (que representa o valor atual) e a linha possui uma diferença de cor entre as direções positiva e negativa. A diferença de cor gera um efeito de “preenchimento” em que o valor máximo resulta na cor branca sobre a linha toda.

Figura 19 – Exemplo de componente *TableSlider*



Fonte: Elaborada pelo autor.

4.2.3 TableCell

Este componente representa um botão radial ao redor do objeto tangível.

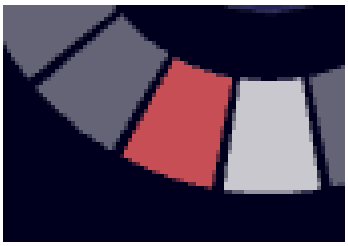
O *TableCell* (Figura 20) possui três cores diferentes:

- **Branco:** Significa que o botão está selecionado.
- **Cinza:** botão não está selecionado.
- **Vermelho:** botão está ativo.

Quando o usuário executa um evento de toque simples neste componente, ele se alterna entre selecionado e não selecionado e, através da aplicação, ao torná-lo ativo, sua cor será vermelha independentemente se estiver selecionado ou não.

Diferente do TableButton, este componente foi desenvolvido para ser utilizado como componente base do *TableSequencerCells* e não de forma independente.

Figura 20 – Exemplo de componentes *TableCell*



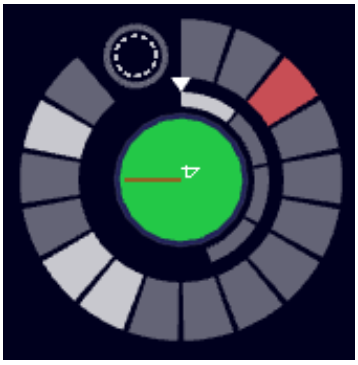
Fonte: Elaborada pelo autor.

4.2.4 TableSequencerCells

Este componente gera uma sequência radial de *TableCells* e serve principalmente para facilitar o gerenciamento deste agrupamento de componentes. Ao inicializar, é possível configurar diversos parâmetros, como, por exemplo, a quantidade de *TableCell*s ou a distância entre eles. Cada *TableCell* recebe eventos independentemente dos outros e pode ser selecionado ou ativado de modo autônomo.

Na Figura 21, o *TableSequencerCells* se encontra na parte mais afastada do círculo central.

Figura 21 – Exemplo de componente *TableSequencerCells*



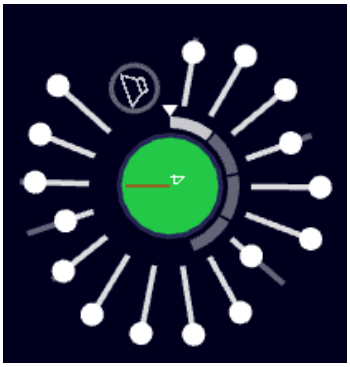
Fonte: Elaborada pelo autor.

4.2.5 TableSequencerSliders

Este componente surgiu da necessidade de se alterar parâmetros adicionais do sinal de controle, como, por exemplo, o volume.

O *TableSequencerSliders* (Figura 22) é formado por uma série de *TableSliders* de modo radial em relação ao objeto tangível e cada *TableSlider* ajusta um parâmetro de modo independente dos outros. Além disso, é possível configurá-lo para mostrar os valores numéricos de cada *TableSlider* na parte superior de cada componente.

Figura 22 – Exemplo de componente *TableSequencerSliders*



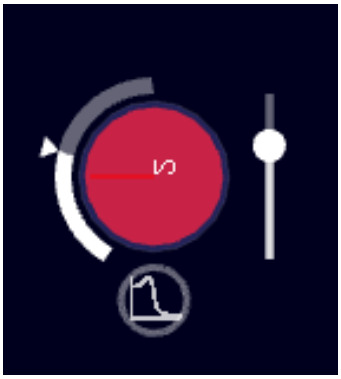
Fonte: Elaborada pelo autor.

4.2.6 TableInfoCircle

Este componente serve para informar e transmitir informações dos objetos tangíveis para o usuário da superfície. Ele é formado por uma barra radial ao redor do objeto tangível e uma seta que indica o valor atual. Ele pode ser configurado para ser tanto contínuo como discreto.

O *TableInfoCircle* (Figura 23) é ligado à rotação do objeto tangível. Ao rotacionar o objeto em um sentido, a barra será preenchida para o mesmo sentido até chegar ao valor máximo (ou mínimo) estipulado pelo programa.

Figura 23 – Exemplo de componente *TableInfoCircle*



Fonte: Elaborada pelo autor.

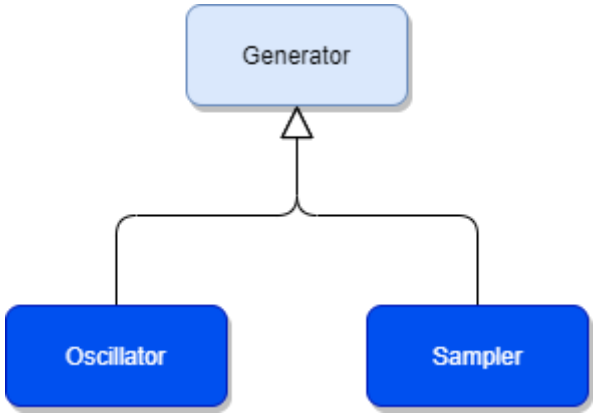
4.3 TIPOS DE OBJETOS TANGÍVEIS

Os objetos tangíveis se conectam de acordo com as possíveis derivações para *TableObject*, no entanto, cada classe derivada de *TableObject* possui suas próprias derivações que especificam seus comportamentos sonoros.

4.3.1 Generator

Os objetos da classe *Generator* são responsáveis por gerar ou produzir som. O processo de gerar som é chamado de síntese sonora. Dentre os diversos métodos de síntese sonora, foram utilizados os dois principais: o oscilador e a reprodução digital (*sampler*). A hierarquia da classe *Generator* está apresentada na Figura 24.

Figura 24 – Hierarquia da classe *Generator*



Fonte: Elaborada pelo autor.

4.3.1.1 Oscillator

A classe *Oscillator* define um oscilador, uma unidade que gera formas de ondas periódicas com uma determinada frequência e amplitude.

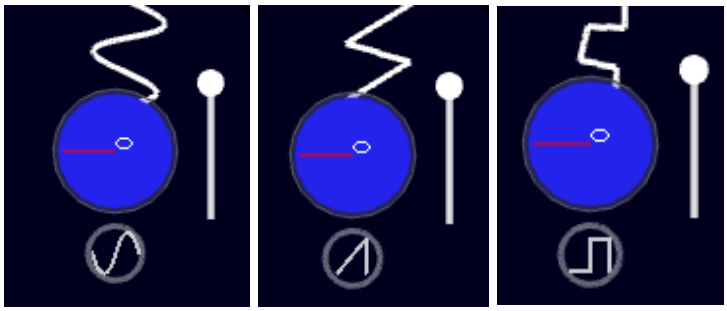
As formas de ondas utilizadas foram:

- Senoidal;
- Quadrada;
- Dente de Serra.

Para mudar a frequência fundamental do oscilador é só girar o objeto tangível. Além disso, é possível trocar a forma de onda, como mostra a Figura 25, dando um

toque simples no *TableButton* presente em torno do objeto tangível e o mesmo mostrará um ícone que representa a forma de onda atual. O objeto conta também com um *TableSlider* em seu entorno que controla o volume.

Figura 25 – Possíveis Modos do *Oscillator*

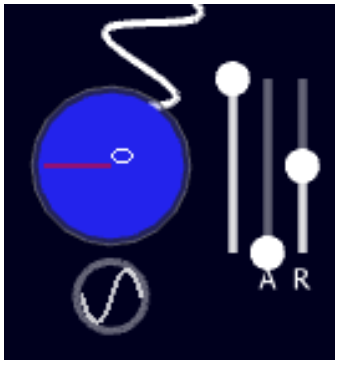


Fonte: Elaborada pelo autor.

Os sons de instrumentos musicais não começam e param instantaneamente. Por exemplo, em um instrumento musical de corda, leva um tempo finito para a sua corda começar a vibrar e um tempo para reduzi-la a um estado estacionário. O tempo decorrido para que um objeto atinja um estado de vibração plena é chamado de tempo de ataque, enquanto o tempo para a vibração decair para um estado estacionário novamente é chamado de tempo de decaimento. Para instrumentos que podem produzir um som contínuo, como um órgão, o tempo de decaimento é definido como o tempo para o som decair para o nível de “sustentação” no estado estacionário, enquanto o fim da vibração é chamado de tempo de repouso ou relaxamento. A combinação de todos os estágios de um som é chamada de envoltória (RUSS, 2009).

Os objetos geradores possuem um menu de configurações que pode ser acessado através de um toque longo no *TableButton* presente no mesmo. Este menu (Figura 26) revela dois novos *TableSlider*s que ajustam os valores de ataque e relaxamento da envoltória, apresentados com as letras A e R embaixo do *TableSlider*, respectivamente. Estes valores são utilizados apenas quando um objeto da classe *Sequencer* está conectado ao mesmo.

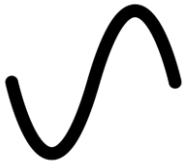
Figura 26 – Possíveis configurações do *Oscillator*



Fonte: Elaborada pelo autor.

Os objetos tangíveis da classe *Oscillator* possuem a cor azul e na face superior deles está presente a Figura 27.

Figura 27 – Ícone do *Oscillator*



Fonte: Elaborada pelo autor.

4.3.1.2 Sampler

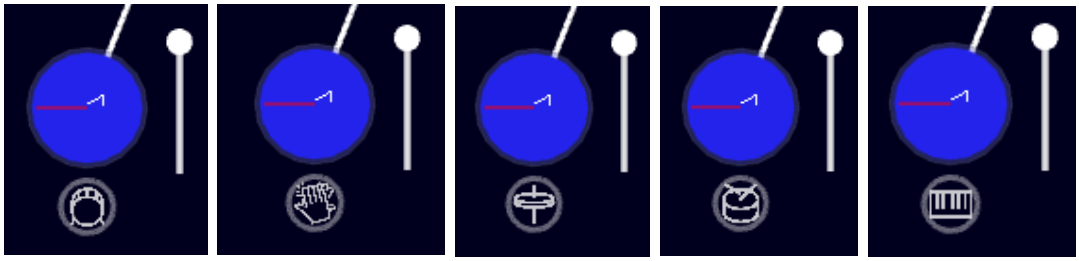
Um *sampler* é o nome dado a um equipamento musical eletrônico que grava um som, armazena-o e o repete sob demanda (RUSS, 2009). Este som gravado (geralmente de curta duração) é chamado comumente de *sample* ou amostra.

Como um *sampler* pode ser carregado com um conjunto de amostras específicas composto de vários sons, até mesmo os produzidos por sintetizadores, ele pode ser usado como uma maneira de produzir sons de uma grande variedade de instrumentos a partir de apenas uma peça de *hardware* ou *software* (RUSS, 2009).

A classe *Sampler* possui cinco modos diferentes. Os modos podem ser alternados por um toque simples no *TableButton* presente no entorno do objeto (Figura 28) e o mesmo mostrará um ícone que representa o modo atual. O modo ativo define o tipo das amostras que são tocadas, sendo estas:

- ***Kick***: percussão de bumbo de bateria.
- ***Clap***: percussão de palmas.
- ***Hat***: percussão de prato de bateria.
- ***Snare***: percussão de caixa de bateria.
- ***Melodic***: sons melódicos, ou seja, sons que não se encaixam na categoria de percussão. Nesse modo se encontram quaisquer amostras que vão desde instrumentos musicais comuns como violino ou xilofone até instrumentos musicais eletrônicos como sintetizadores.

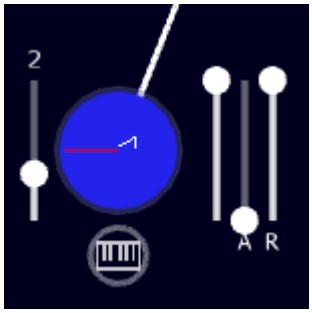
Figura 28 – Possíveis modos do *Sampler*



Fonte: Elaborada pelo autor.

Diferentemente dos objetos da classe *Oscillator*, os objetos da classe *Sampler* necessitam de um *Sequencer* conectado ao mesmo para poder emitir som. O objeto da classe Sampler possui um *TableSlider* para controle de volume e através de um toque longo no *TableButton* do objeto são revelados mais *TableSliders* para configurações adicionais. Além dos *TableSliders* para o ajuste dos valores de ataque e relaxamento, este objeto possui um *TableSlider* com um número em cima (Figura 29) que altera o *sample* atual. O número de *samples* de cada modo só depende do número de arquivos sonoros (em formato *wav* com taxa de amostragem de 44,1 kHz) presentes no diretório do programa.

Figura 29 – Configurações do *Sampler*



Fonte: Elaborada pelo autor.

Os objetos tangíveis da classe *Sampler* possuem a cor azul e na face superior deles está presente a Figura 30.

Figura 30 – Ícone do *Sampler*

Fonte: Elaborada pelo autor.

4.3.2 Effect

Além de escolher qual gerador de som utilizar, existem muitas ferramentas comuns para esculpir o timbre de um som e guiar sua forma ao longo do tempo. Essas ferramentas são chamadas de efeitos ou modificadores. Se considerarmos que o som em sua forma mais básica é uma coleção de ondas em diferentes frequências, fases e intensidades variando ao longo do tempo, é notável que as categorias de efeitos serão aquelas que afetam o conteúdo da frequência, as que afetam a amplitude e aquelas que moldam um ou ambos ao longo do tempo (PEJROLO e METCALFE, 2017).

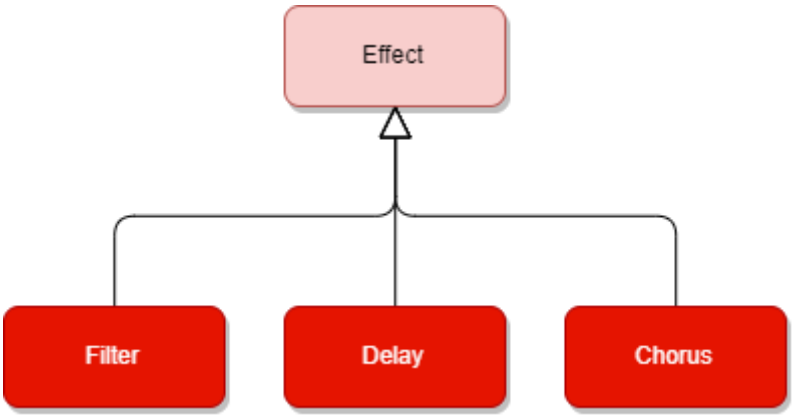
A classe *Effect* representa os modificadores de som e ela pode se conectar com:

- a saída de som;
- entre a conexão de um *Generator* e a saída de som;

- entre a conexão de um *Generator* e um *Effect*;
- entre a conexão de um *Effect* e a saída de som.

Portanto, é possível conectar vários efeitos em série. Sua hierarquia de classes está apresentada na Figura 31.

Figura 31 – Hierarquia da classe *Effect*



Fonte: Elaborada pelo autor.

4.3.2.1 Filter

A classe *Filter* representa um filtro. Os filtros seletores de sinal permitem a passagem de faixas de frequências, mas rejeitam outras (RUSS, 2009).

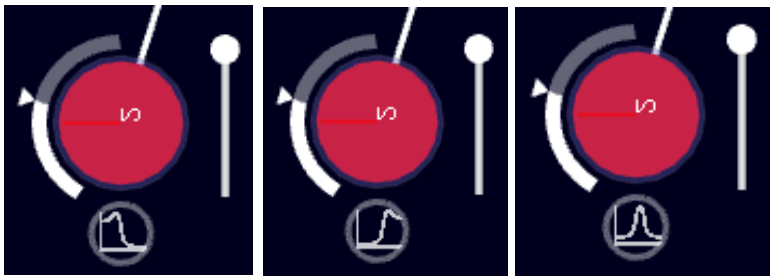
Os tipos de curva de resposta em frequência que foram implementados são:

- **Passa-baixa**: Um filtro passa-baixa tem mais atenuação à medida que a frequência aumenta.
- **Passa-alta**: Um filtro passa-alta tem a ação de filtro oposta a um filtro passa-baixa: atenua todas as frequências que estão abaixo da frequência de corte.
- **Passa-banda**: Um filtro passa-banda apenas permite que uma faixa definida de frequências passe por ele inalterada - todas as outras frequências são atenuadas.

Os possíveis tipos de curva de resposta em frequência da classe *Filter* podem ser alternadas e escolhidas com um toque no *TableButton* presente no entorno do objeto e o mesmo mostrará um ícone que representa o filtro atual (Figura 32). Esta

classe conta também com um *TableInfoCircle,* que representa a frequência de corte de todos os filtros, e um *TableSlider* que determina a intensidade do filtro.

Figura 32 – Possíveis modos do *Filter*



Fonte: Elaborada pelo autor.

Os objetos tangíveis da classe *Filter* possuem a cor vermelha e na face superior deles está presente a Figura 33.

Figura 33 – Ícone do *Filter*

Fonte: Elaborada pelo autor.

4.3.2.2 Delay

Delay é um efeito em que o áudio de entrada é repetido após um determinado período de tempo. Se este tempo estiver dentro de um intervalo de cerca de 100 ms ou menor, o efeito tem um impacto maior no timbre, ou seja, nosso sistema auditivo pode não ser capaz de detectar como dois sons distintos e então irá fundi-los (WÖLFEL e MCDONOUGH, 2009). Com um tempo de atraso maior que 100 ms, é possível ouvir uma repetição distinta do som original.

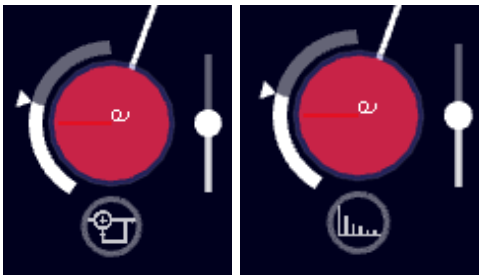
A classe *Delay* possui dois modos:

- ***Delay* com feedback**: neste modo, o áudio é realimentado com uma cópia atrasada e atenuada. Os valores de atraso e atenuação são parâmetros de entrada.

- **Reverberação**: simula um ambiente acústico real. O áudio é realimentado com diversos atrasos e estas realimentações passam por um filtro passa-baixa.

O modo pode ser alterado com um toque simples no *TableButton* presente ao redor do objeto (Figura 34). Além disso, o objeto da classe *Delay* possui um *TableInfoCircle* que indica o valor do tempo de atraso, que pode ser alterado girando o objeto, e um *TableSlider* que modifica a atenuação da realimentação.

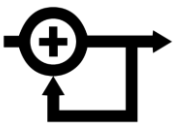
Figura 34 – Possíveis modos do *Delay*



Fonte: Elaborada pelo autor.

Os objetos tangíveis da classe *Delay* possuem a cor vermelha e na face superior deles está presente a Figura 35.

Figura 35 – Ícone do *Delay*



Fonte: Elaborada pelo autor.

4.3.2.3 Chorus

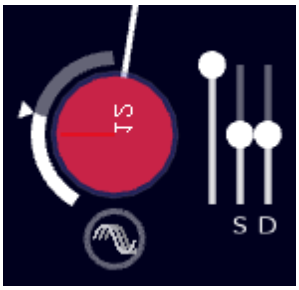
O termo “*chorus*” vem da ideia de tentar fazer uma voz soar como um coro inteiro. O som é obtido por camadas de vários atrasos dentro de um intervalo de 20 a 50 ms. Além disso, os diferentes estágios são levemente desafinados, aumentando ainda mais a complexidade do som (PEJROLO e METCALFE, 2017).

O objeto da classe *Chorus* possui um *TableInfoCircle* que exibe o valor da modulação e esse valor é alterado girando o objeto. Além disso, a classe possui um *TableSlider* que determina a intensidade do efeito (Figura 36). Através de um toque simples no *TableButton* do objeto, sãos revelados dois *TableSliders* para configurações adicionais (Figura 37). O *TableSlider* com a letra S ajusta a velocidade da modulação e o com a letra D ajusta o atraso do efeito.

Figura 36 – Interface do *Chorus*

Fonte: Elaborada pelo autor.

Figura 37 – Possíveis configurações do *Chorus*



Fonte: Elaborada pelo autor.

Os objetos tangíveis da classe *Chorus* possuem a cor vermelha e na face superior deles está presente a Figura 38.

Figura 38 – Ícone do *Chorus*

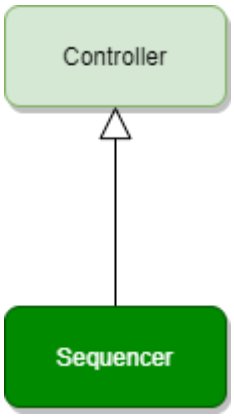
Fonte: Elaborada pelo autor.

4.3.3 Controller

Além de gerar o som e modificá-lo, surge a necessidade de alterar não apenas o timbre do som como também outros parâmetros. Por exemplo, pode-se alterar o andamento, a amplitude ou qual nota musical a ser tocada. O parâmetro mais interessante é o andamento, ou seja, quando tocar a nota. A partir disso, cria-se uma base para a geração de ritmos. Além do mais, se configurarmos também qual nota musical tocar, cria-se uma base para a geração de melodias e assim, com a união de ritmo e melodia, músicas mais interessantes são geradas.

Em vista disso, a classe *Controller* representa controladores que modificam outros parâmetros além do som. Objetos desta classe se conectam apenas a objetos da classe *Generator*. Atualmente a classe *Controller* possui apenas uma derivação e a hierarquia está apresentada na Figura 39.

Figura 39 – Hierarquia da classe *Controller*



Fonte: Elaborada pelo autor.

4.3.3.1 Sequencer

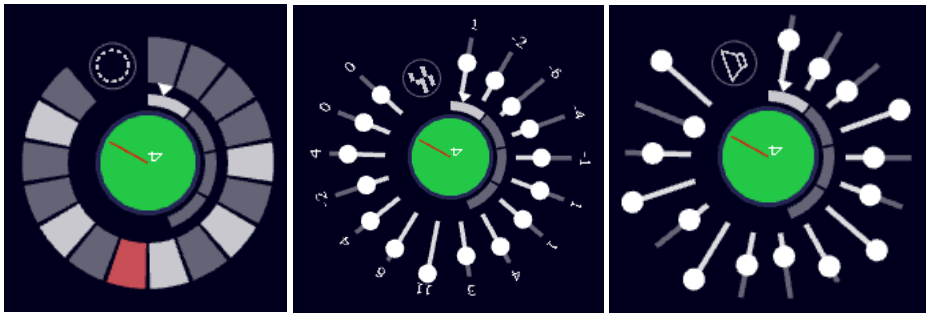
A classe *Sequencer* representa um sequenciador ou, como é mais conhecido, sequenciador de passo. Os sequenciadores de passo produzem *loops* de padrão que normalmente têm 16 notas, com variantes de 8, 12, 24 ou 32 notas em algumas circunstâncias. As sequências repetem-se continuamente uma vez iniciadas, tocando as 16 notas em ordem. Portanto, uma vez configurado o sequenciador, é possível criar composições mais longas que possuem acompanhamentos e que tocam juntos para então criar uma música mais completa.

A classe *Sequencer* possui três menus de configurações (Figura 40):

- **Sequência**: Neste menu é possível selecionar a sequência a ser tocada por meio do componente *TableSequencerCells*. Um toque simples em qualquer uma das *TableCells* presentes no entorno do objeto tangível da classe *Sequencer* irá alterná-lo entre selecionado e não selecionado. Os *TableCells* são ativados, um por vez, no sentido horário seguindo um tempo fixo determinado pelo sistema e um multiplicador desse tempo determinado no próprio objeto (que pode ser modificado pelo usuário). Os *TableCells* selecionados (na cor branca) representam os passos que serão tocados e o *TableCell* ativo (na cor vermelha) mostra o passo atual que está sendo tocado (se ativo).
- **Pitch**: Neste menu é possível ajustar a nota de cada passo do sequenciador. Ele apresenta um componente *TableSequencerSlider* e cada *TableSlider* apresenta um número em cima. Este número representa o *offset,* ou intervalo, da nota a ser tocada, em relação à nota do gerador em que o objeto da classe *Sequencer* está conectado.
- **Volume**: Neste menu é possível ajustar o volume da nota a ser tocada de cada passo do sequenciador.

Para trocar de menu é apenas necessário dar um toque simples no *TableButton* presente no objeto da classe *Sequencer*.

Figura 40 – Possíveis modos do *Sequencer*

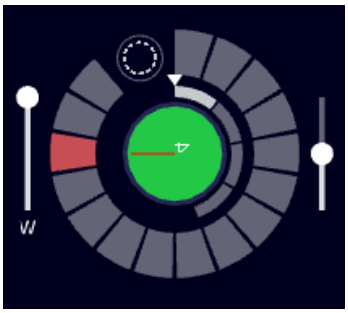


Fonte: Elaborada pelo autor.

Com um toque longo no *TableButton* presente no objeto, são revelados dois *TableSliders* (Figura 41). O *TableSlider* que possui a letra W embaixo é utilizado para ajustar a largura do sinal de pulso enviado quando a nota deve ser tocada. O outro *TableSlider* ajusta o tempo do sequenciador atual. Além disso, como mostra a Figura

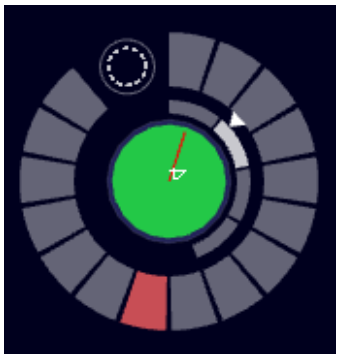
42, esta classe possui um *TableInfoCircle* discreto, presente entre o objeto tangível e o componente *TableSequencerCells,* que indica a sequência atual dentre as quatro possíveis. Para mudar entre sequências salvas é necessário apenas girar o objeto.

Figura 41 – Possíveis configurações do *Sequencer*



Fonte: Elaborada pelo autor.

Figura 42 – Sequências salvas do *Sequencer*



Fonte: Elaborada pelo autor.

Os objetos tangíveis da classe *Sequencer* possuem a cor verde e na face superior deles está presente a Figura 43.

Figura 43 – Ícone do *Sequencer*

Fonte: Elaborada pelo autor.

4.3.4 Output

A classe *Output* representa a saída de som e é considerada como um objeto tangível de posição fixa apenas para fins de conexão. A representação deste objeto é um ponto branco na superfície onde todas as ondas de som desenhadas dos objetos tangíveis conectados convergem. A saída de som sem conexões é mostrada na Figura 44 e a mesma com conexões, na Figura 45.

Figura 44 – Saída de som sem conexões

Fonte: Elaborada pelo autor.

Figura 45 – Saída de som com conexões

Fonte: Elaborada pelo autor.

5 RESULTADOS

5.1 RECONHECIMENTO

Após a montagem da superfície tangível, foram feitos testes para validar o reconhecimento dos fiduciais e do multitoque. A Figura 46 mostra a imagem obtida pela câmera já montada na superfície tangível. Na figura, pode-se ver as paredes da caixa de papelão utilizada para a estrutura e no quadrado central está o difusor. A Figura 47 mostra a grade de calibração presente no programa *reacTIVision*. Esta grade está ajustada para centralização do difusor.

Na Figura 48, pode-se ver o reconhecimento dos dedos da mão, representado pela letra “F” presente no centro de cada dedo. A superfície identificou com alta taxa de acerto os dedos na sua região central, no entanto, as regiões mais próximas às laterais apresentavam perdas na identificação, ou seja, não eram reconhecidas como um dedo pelo programa em alguns momentos.

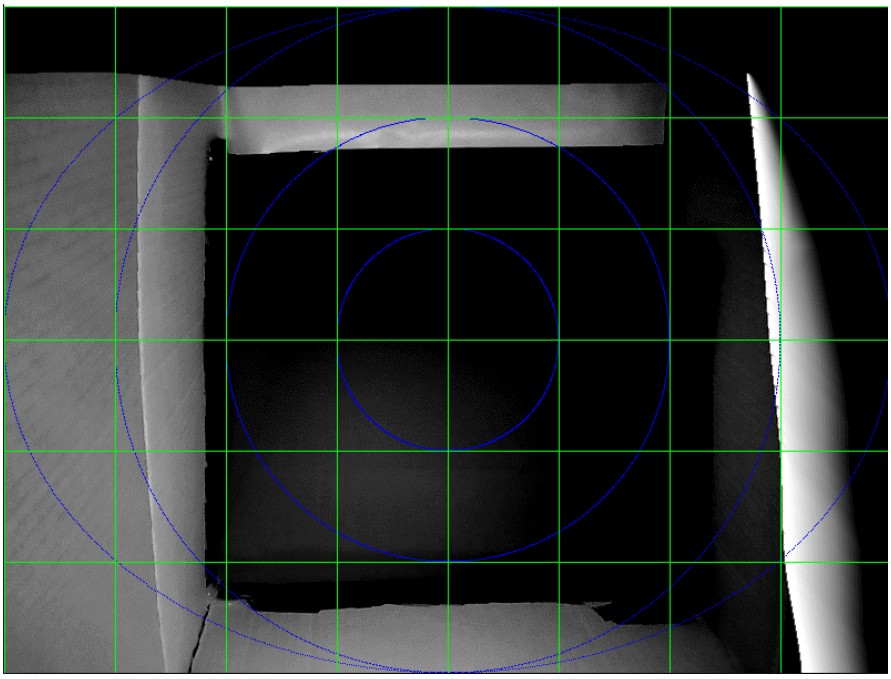
Na Figura 49 está a identificação do fiducial com o seu número, neste caso 1, exibido no interior do próprio fiducial. O reconhecimento dos fiduciais se apresentou robusto por toda a região da superfície.

O *reacTIVision* possui a opção de subtração de fundo que pode simplificar e melhorar o desempenho de reconhecimento de dedos e fiduciais. A subtração de fundo exclui os elementos estáticos da imagem e mantém apenas os elementos que não estavam presentes na imagem de fundo. A Figura 50 mostra a visão da câmera sem a grade de calibração e com a subtração de fundo ativada. Podemos notar que apenas a mão apareceu e as regiões da imagem que são estáticas foram equalizadas para um mesmo valor de cor.

Figura 46 – Visão da câmera na Superfície tangível

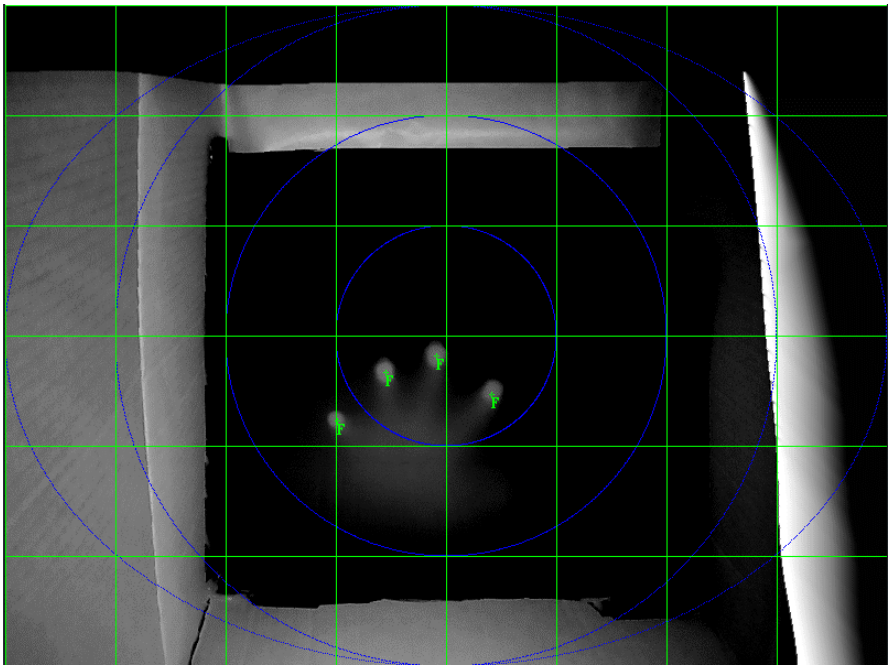
Fonte: Elaborada pelo autor.

Figura 47 – Grade de calibração da Superfície tangível



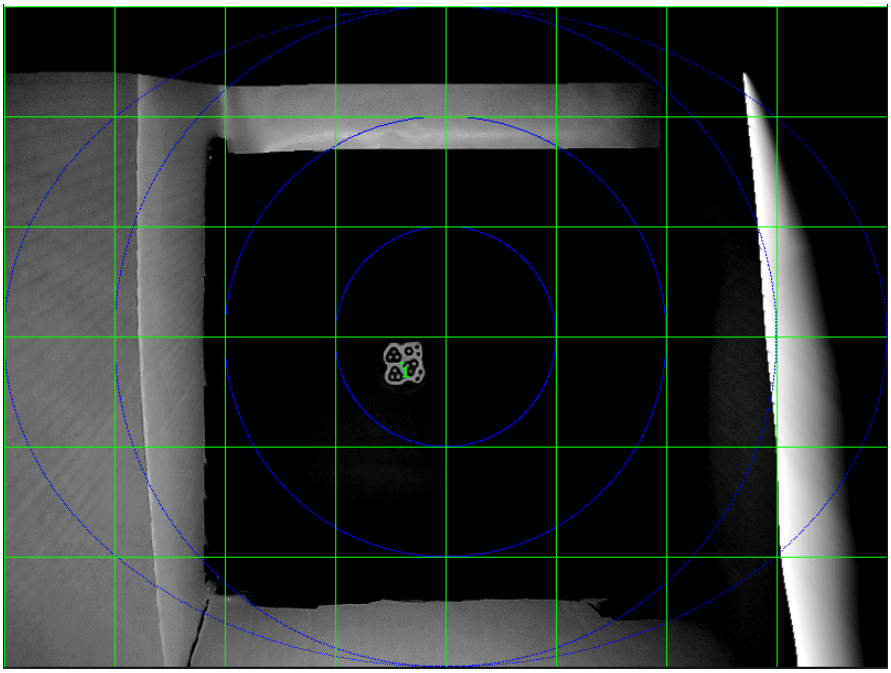
Fonte: Elaborada pelo autor.

Figura 48 – Identificação dos dedos na Superfície tangível



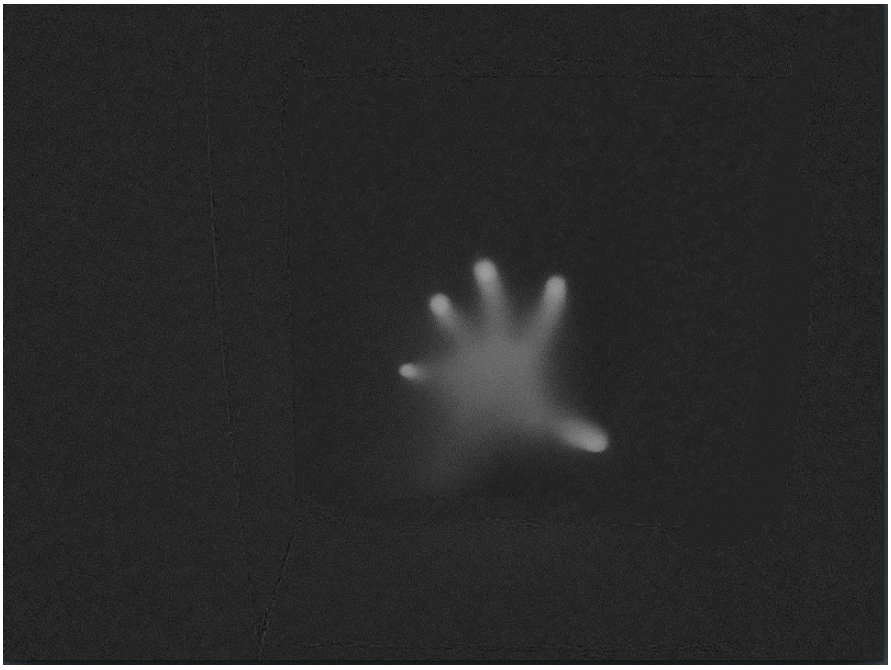
Fonte: Elaborada pelo autor.

Figura 49 – Identificação de fiducial na Superfície tangível



Fonte: Elaborada pelo autor.

Figura 50 – Exemplo da subtração de fundo na Superfície tangível



Fonte: Elaborada pelo autor.

5.2 CONSTRUÇÃO

A Figura 51 mostra a visão exterior da superfície tangível enquanto que a Figura 52 mostra a sua visão interior. A estrutura da superfície, apesar de ser feita de papelão, provou ser bastante robusta e supriu as necessidades para a criação de um protótipo. Os componentes dentro da superfície foram posicionados de modo que não houvesse interferência entre os mesmos.

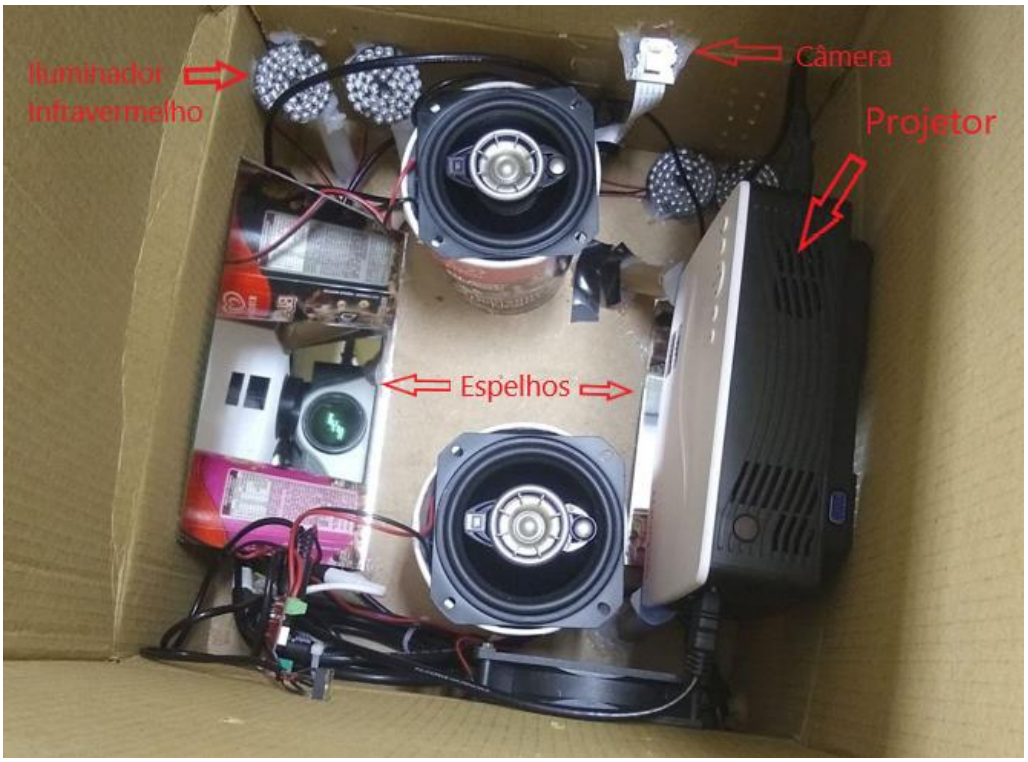
A Figura 53 mostra todos os objetos tangíveis disponíveis para serem utilizados na superfície. Foi decidido criar pelo menos duas peças para cada objeto tangível permitido na aplicação.

As Figuras 54 e 55 mostram o funcionamento da superfície tangível com e sem luz ambiente, respectivamente. Podemos perceber que quando a luz ambiente é reduzida, a projeção se sobressai e fica mais evidente. No entanto, mesmo com luz ambiente é possível utilizar a superfície, com a desvantagem de possuir um contraste menor na projeção.

Figura 51 – Visão do exterior da superfície tangível

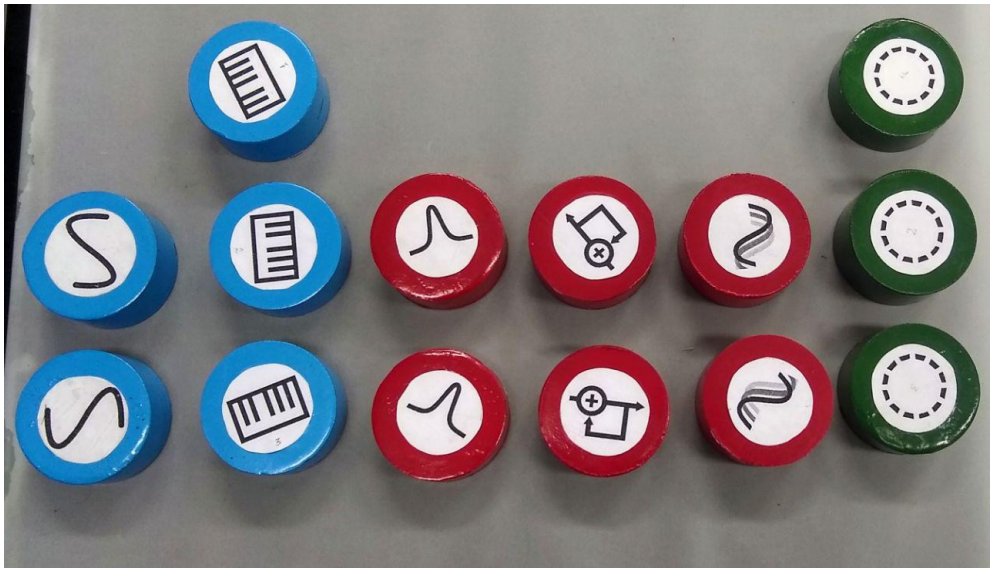
Fonte: Elaborada pelo autor.

Figura 52 – Visão interior da superfície tangível



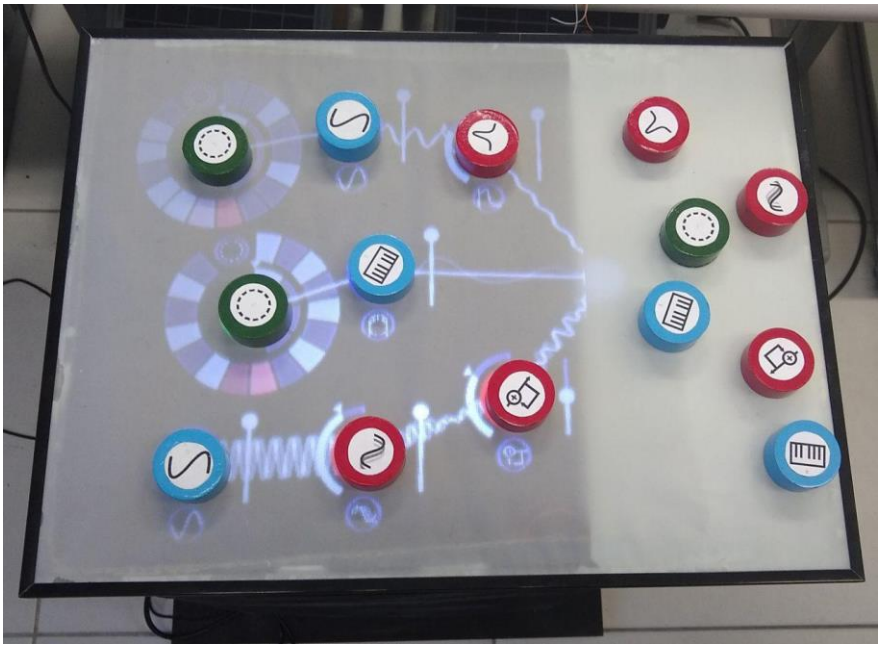
Fonte: Elaborada pelo autor.

Figura 53 – Objetos tangíveis disponíveis



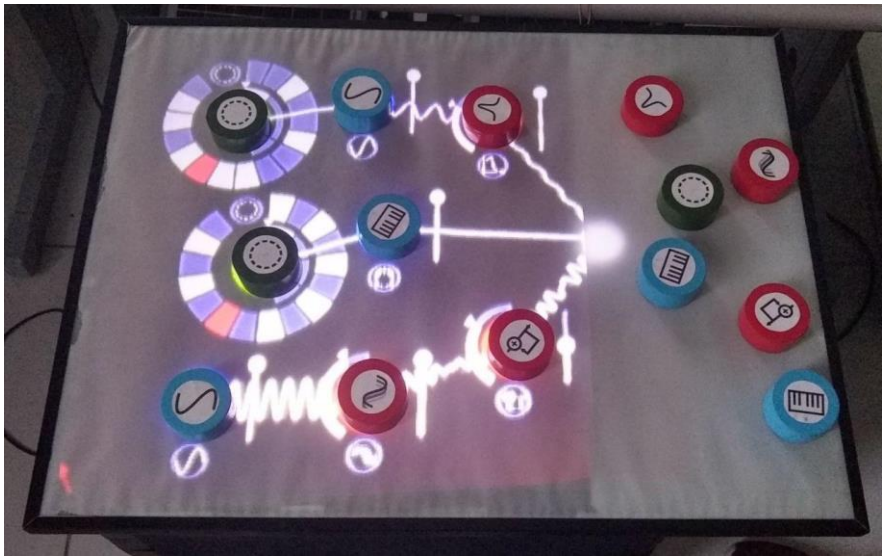
Fonte: Elaborada pelo autor.

Figura 54 – Funcionamento da superfície tangível com luz ambiente



Fonte: Elaborada pelo autor.

Figura 55 – Funcionamento da superfície tangível no escuro



Fonte: Elaborada pelo autor.

5.3 EXPOSIÇÃO

O protótipo desenvolvido foi exposto na 16ª Semana Nacional da Ciência e Tecnologia no Instituto Federal de Santa Catarina (IFSC). Durante três dias inteiros, este projeto ficou em exibição para interação com os visitantes. O projeto era brevemente explicado, mostrando seu funcionamento e, a partir disso, os visitantes podiam mexer para criar seus próprios sons.

Este projeto abriu os olhos de visitantes para novas possibilidades com o uso desta interface e despertou a curiosidade de crianças e adultos que ficavam fascinados com o que viam. Além disso, a exposição serviu como teste de robustez para o *hardware* e *software*. Durante a exposição foi necessário apenas a calibração periódica do *software*, que é realizada a cada vez que o programa é executado. Após a exposição, o protótipo continuou funcional.

5.4 REPOSITÓRIO

O repositório com o código desenvolvido pode ser encontrado em https://github.com/cleissom/Soundnator.

6 CONCLUSÃO

A superfície com interface tangível foi montada em uma estrutura simples para a validação do protótipo e, apesar de apresentar perdas de detecção de dedos nas regiões próximas às bordas da superfície, mostrou-se efetiva no seu funcionamento.

Após determinada a arquitetura da aplicação, o programa foi desenvolvido e satisfez os requisitos de interação com o usuário utilizando uma interface tangível e interações através do multitoque na superfície. A interface da aplicação possui componentes simples, que satisfizeram a necessidade de configurações de parâmetros e apresentação de informações. Além disso, a aplicação mostrou um bom desempenho de processamento mesmo rodando em um sistema embarcado, isto é, a taxa de atualização de quadros média era maior que 20 quadros por segundo e a latência para o movimento dos objetos era insignificante. A detecção pode ser melhorada através de melhorias na estrutura, nos componentes e no difusor utilizados.

Portanto, a superfície com interface tangível e sua aplicação de sintetizador foram desenvolvidas e implementadas com sucesso.

Para trabalhos futuros as possibilidades de ampliação do projeto são extensas:

- Desenvolver novos efeitos, como por exemplo os efeitos de distorção, compressão, *tremolo*.
- Desenvolver novos controladores, como por exemplo o LFO (*Low Frequency Oscillator*).
- Aprimorar os geradores de som para permitir novas opções, como por exemplo permitir o uso de uma forma de onda arbitrária criada pelo usuário.
- Aprimorar o algoritmo de conexão entre objetos tangíveis para uma conexão mais intuitiva e dinâmica, pois o algoritmo atual é baseado na distância entre os objetos e disponibilidade de conexão e, por conta disso, há casos em que a conexão não é feita de modo evidente e imediata.

REFERÊNCIAS

GEIGER, G. *et al.* Music, The Reactable: A Collaborative Musical Instrument for Playing and Understanding. **Her&Mus. Heritage & Museography**, n. 4, p. 36-43, 2010.

HAN, J. Y. **Low-Cost Multi-Touch Sensing through Frustrated Total Internal Reflection**. In UIST ’05: Proceedings of the 18th annual ACM symposium on User interface software and technology. New York: ACM. 23 out. 2005. p. 115-118.

ISHII, H.; ULLMER, B. **Tangible Bits:** Towards Seamless Interfaces between People, Bits and Atoms. Proceedings of the ACM SIGCHI Conference on Human factors in computing systems. Atlanta: ACM. mar. 1997. p. 234-241.

JORDÀ, S. *et al.* **The reacTable***. International Computer Music Conference. Barcelona: [*S.n.*]. 2005. p. 579-582.

JULIÀ, C. F. **Making tabletops useful with applications, frameworks and multi-tasking**. Universidade Pompeu Fabra. Barcelona, p. 224. 2015.

KALTENBRUNNER, M. *et al.* **The reacTable*:** A Collaborative Musical Instrument. Workshops on Enabling Technologies: Infrastructures for Collaborative Enterprises. Manchester: [*S.n.*]. 2006. p. 406-411.

KALTENBRUNNER, M. **reacTIVision and TUIO:** a tangible tabletop toolkit. ACM International Conference on Interactive Tabletops and Surfaces. [*S.l.*]: ACM. 2009. p. 9-16.

KALTENBRUNNER, M.; BENCINA, R. **reacTIVision:** a computer-vision framework for table-based tangible interaction. Proceedings of the 1st international conference on Tangible and embedded interaction. New York City: ACM. 2007. p. 69-74.

KHRONOS GROUP. OpenGL ES Overview. **Khronos Group Web site**, 2019. Disponivel em: https://www.khronos.org/opengles/. Acesso em: 21 nov. 2019.

KIM, M. J.; MAHER, M. L. The impact of tangible user interfaces on designers' spatial cognition. **Human-Computer Interaction**, v. 23, n. 2, p. 101-137, June 2008.

MULLER, L. Y. L. **Multi-touch displays : design , applications and performance evaluation**. University of Amsterdam. Amsterdam, p. 96. 2008.

NOBLE, J. **Programming Interactivity:** A Designer's Guide to Processing, Arduino, and OpenFrameworks. 1. ed. Sebastopol, Califórnia: O'Reilly Media, 2009.

PEJROLO, A.; METCALFE, S. B. **Creating Sounds from Scratch:** A Practical Guide to Music Synthesis for Producers and Composers. 1. ed. New York: Oxford University Press, 2017.

RUSS, M. **Sound Synthesis and Sampling, 3rd Edition**. 3. ed. [*S.l.*]: Focal Press, 2009.

SCHNEIDER, B. *et al.* Benefits of a Tangible Interface for Collaborative Learning and Interaction. **IEEE Transactions on Learning Technologies**, v. 4, p. 222 - 232, out. 2011.

SCHÖNING, J. *et al.* Multi-Touch Surfaces: A Technical Guide Technical Report. **Technical Reports of the Technical University of Munich**, out. 2008.

SHAER, O.; HORNECKER, E. Tangible User Interfaces: Past, Present, and Future Directions. **Foundations and Trends® in Human–Computer Interaction**, v. 3, n. 1-2, p. 4-137, 2010.

ULLMER, B.; ISHII, H. The metaDESK: Models and Prototypes for Tangible User Interfaces. **Proceedings of UIST'97**, fev. 1998.

ULLMER, B.; ISHII, H. Emerging Frameworks for Tangible User Interfaces. **IBM Systems Journal**, Riverton, v. 39, n. 3-4, p. 915-931, jul. 2000.

UNDERKOFFLER, J.; ISHII, H. **Urp:** A Luminous-tangible Workbench for Urban Planning and Design. Proceedings of the SIGCHI Conference on Human Factors in Computing Systems. New York City: ACM. 1999. p. 386-393.

WÖLFEL, M.; MCDONOUGH, J. **Distant speech recognition**. Nova Jersey, EUA: Wiley, 2009.