

Gianfranco Pennacchi

Vinícius Menézio

**Realidade Virtual Aplicada ao Desenvolvimento  
de Competências e Habilidades de Raciocínio  
Lógico em Crianças**

São Paulo

2016

Gianfranco Pennacchi  
Vinícius Menézio

**Realidade Virtual Aplicada ao Desenvolvimento de  
Competências e Habilidades de Raciocínio Lógico em  
Crianças**

Trabalho de Formatura apresentado ao Departamento de Engenharia de Computação e Sistemas Digitais da Escola Politécnica da Universidade de São Paulo para obtenção do Diploma de Engenheiro

São Paulo  
2016

## Catalogação-na-publicação

Pennacchi, Gianfranco

Realidade Virtual Aplicada ao Desenvolvimento de Competências e  
Habilidades de Raciocínio Lógico em Crianças /  
G. Pennacchi, V. P. Menézio -- São Paulo, 2016.  
75 p.

Trabalho de Formatura - Escola Politécnica da Universidade de São  
Paulo. Departamento de Engenharia de Computação e Sistemas Digitais.

1.Educação 2.Realidade Virtual 3.Raciocínio Lógico I.Universidade de  
São Paulo. Escola Politécnica. Departamento de Engenharia de Computação e  
Sistemas Digitais II.Pennacchi, Gianfranco III.Menézio, Vinícius Pinto

Gianfranco Pennacchi

Vinícius Menézio

**Realidade Virtual Aplicada ao Desenvolvimento de  
Competências e Habilidades de Raciocínio Lógico em  
Crianças**

Trabalho de Formatura apresentado ao Departamento de Engenharia de Computação e Sistemas Digitais da Escola Politécnica da Universidade de São Paulo para obtenção do Diploma de Engenheiro

Orientador: Prof. Dr. Ricardo Nakamura

Coorientador: Profa. Msc. Lucy Mari Tabuti

São Paulo

2016

# Resumo

O sucesso e a taxa de adoção de jogos eletrônicos na educação foi, até os dias atuais, insuficiente para encorajar a implementação em larga escala de experiências lúdicas como formas de facilitar o aprendizado. Isso decorre principalmente de três fatos: uma divergência entre os objetivos de design de jogos educacionais e os de jogos não-educacionais; dificuldades práticas de prover cada estudante com o *hardware* e o *software* necessários; e a necessidade de supervisão de professores para que os resultados desejados sejam alcançados.

Contudo, conforme a tecnologia móvel se aproxima de seu ponto de maturidade, e dispositivos de realidade virtual tecnologicamente viáveis e financeiramente acessíveis começam a adentrar o mercado em massa, um momento crítico se aproxima no qual experiências educativas verdadeiramente imersivas e cuidadosamente projetadas possam ser desenvolvidas, valendo-se de pouco mais do que os celulares que as crianças já carregam em seus bolsos.

O intuito deste projeto de Trabalho de Conclusão de Curso (TCC) é criar um jogo que, ao fazer uso de tecnologia móvel e de realidade virtual, possa providenciar um ambiente imersivo e fértil para a exploração e desenvolvimento de habilidades motoras, lógicas e do raciocínio espacial de crianças, expandindo a fronteira de conhecimento relacionada a jogos digitais aplicados à educação, alinhado ao projeto da doutoranda Lucy Mari Tabuti.

As tecnologias utilizadas são o *Google Cardboard* para gerar a realidade virtual e o *Leap Motion* para a captura de gestos. O jogo foi desenvolvido na ferramenta *Unity*, em virtude das facilidades propiciadas por ela, em especial lidando com ambientes tridimensionais.

**Palavras-chave:** Educação, imersão, realidade virtual, raciocínio lógico, jogo.

# Abstract

The success and adoption rate of video games in education have, up to this day, not proven sufficient to encourage large-scale implementation of gaming as a means to supplement and facilitate learning. This stems from both a divergence of the design goals when comparing educational to non-educational digital games, as well as from the practical difficulty of providing each student with the necessary hardware, software and support from teachers to ensure the desired results are being achieved from the playing of the games.

However, as mobile technology achieves its point of maturity, and affordable, technically viable virtual reality-capable devices start to enter the market *en masse*, we stand at a critical moment in which truly immersive, carefully designed learning experiences can be developed, making use of little more than the smartphones children already carry in their pockets.

The purpose of this work is to develop a game that, making use of mobile technology and virtual reality, may provide a fertile and immersive environment for exploration and development of motor, logic and spatial reasoning skills, while expanding the frontiers of educational gaming, aligned to the projects of Professor Lucy Mari Tabuti.

The technologies to be employed are the Google Carboard virtual reality headset and the Leap Motion hand gesture detector. The game will be developed using the Unity engine, given its high level capabilities and suitability to 3D environments.

**Keywords:** Education, immersion, virtual reality, logic puzzle, logical reasoning.

# **Lista de ilustrações**

Figura 1 – Modelo baseado na anatomia humana utilizado pelo controlador <i>Leap Motion</i> . . . . .	29
Figura 2 – Ciclo de Boehm. . . . .	32
Figura 3 – Primeiro rascunho do jogo . . . . .	38
Figura 4 – Dois exemplos do <i>minigame</i> “Zelda Boss Key” do jogo <i>Legend of Zelda: Skyward Sword</i> . . . . .	39
Figura 5 – Rascunho de como criar uma praia, utilizando os elementos terra, água e ar . . . . .	42
Figura 6 – Rascunho de uma possível primeira fase . . . . .	42
Figura 7 – Prova de conceito do jogo . . . . .	43
Figura 8 – Mecânicas da água 1 . . . . .	44
Figura 9 – Mecânicas da água 2 . . . . .	44
Figura 10 – Prova de conceito do jogo . . . . .	44
Figura 11 – Alternativa de arquitetura com servidor WebSocket . . . . .	48
Figura 12 – Alternativa de arquitetura com streaming do video . . . . .	49
Figura 13 – Alternativa de arquitetura com conexão direta . . . . .	50
Figura 14 – Crianças testando o jogo durante a no Tech Kids Day . . . . .	59
Figura 15 – Jogo sendo executado apenas com o <i>Leap Motion</i> . . . . .	59

# **Lista de quadros**

Quadro 1 – Competências e habilidades . . . . .	25
Quadro 2 – Elementos manipuláveis pelo jogador . . . . .	41
Quadro 3 – Comparação das alternativas de arquitetura . . . . .	51
Quadro 4 – Gestos definidos . . . . .	53
Quadro 5 – Configurações definidas . . . . .	54
Quadro 6 – Reconhecimento de gestos . . . . .	55

# **Lista de abreviaturas e siglas**

MECC	Consórcio de Computação Educacional de Minnesota
MEC	Ministerio da Educação
PCN	Parâmetros Curriculares Nacionais
GDD	Game Design Document
SDK	Software Development Kit
OCDE	Organisation for Economic Co-operation and Development

# Sumário

<b>1</b>	<b>INTRODUÇÃO</b>	<b>19</b>
1.1	<b>Objetivos</b>	20
1.2	<b>Justificativa</b>	20
1.3	<b>Resultados esperados</b>	21
1.4	<b>Trabalhos relacionados</b>	21
1.5	<b>Estrutura do trabalho</b>	22
<b>2</b>	<b>TEORIA</b>	<b>23</b>
2.1	<b>Jogos digitais</b>	23
2.2	<b>Jogos educativos</b>	23
2.3	<b>Mecânicas, dinâmicas e estética</b>	24
2.4	<b>Realidade virtual</b>	24
2.5	<b>Competências e habilidades do raciocínio lógico</b>	25
2.6	<b>Público alvo</b>	26
2.6.1	Adaptabilidade à realidade virtual	26
2.6.2	Desenvolvimento do raciocínio lógico	27
2.7	<b>Tecnologia</b>	<b>27</b>
2.7.1	Protocolo <i>WebSocket</i>	27
2.7.2	Controlador <i>Leap Motion</i>	28
2.7.3	Reconhecimento de gestos	29
2.7.4	<i>Google Cardboard</i>	30
<b>3</b>	<b>METODOLOGIA</b>	<b>31</b>
3.1	<b>Formulação do problema</b>	31
3.2	<b>Implementação iterativa</b>	31
3.2.1	<i>Brainstorming</i>	31
3.2.2	Escolha da solução	32
3.2.3	Análise de riscos	32
3.2.4	Prototipação	32
3.2.5	Teste	33
3.2.6	Análise e refinamento do design	33
<b>4</b>	<b>DESENVOLVIMENTO</b>	<b>35</b>
4.1	<b>Objetivo e formulação do problema</b>	35
4.1.1	Ferramentas e tecnologias	35
4.1.1.1	Alternativas consideradas	36

4.1.2	Delimitação do escopo . . . . .	37
<b>4.2</b>	<b><i>Brainstorming e seleção do conceito do jogo</i></b> . . . . .	<b>37</b>
4.2.1	Ideias do brainstorm . . . . .	38
4.2.2	Seleção do conceito do jogo . . . . .	40
<b>4.3</b>	<b>Primeira iteração: prova de conceito</b> . . . . .	<b>41</b>
<b>4.4</b>	<b>Segunda iteração: mecânicas básicas</b> . . . . .	<b>43</b>
<b>4.5</b>	<b>Terceira iteração: últimas mecânicas</b> . . . . .	<b>45</b>
<b>4.6</b>	<b>Quarta iteração: refinamentos</b> . . . . .	<b>45</b>
<b>4.7</b>	<b>Requisitos</b> . . . . .	<b>45</b>
4.7.1	Requisitos não funcionais . . . . .	45
4.7.2	Requisitos funcionais . . . . .	46
<b>4.8</b>	<b>Arquitetura</b> . . . . .	<b>46</b>
4.8.1	Identificação dos elementos . . . . .	46
4.8.1.1	Controlador <i>Leap Motion</i> . . . . .	46
4.8.1.2	Lógica do jogo . . . . .	46
4.8.1.3	Processamento de saída do video . . . . .	47
4.8.2	Arquiteturas estudadas . . . . .	47
4.8.2.1	Alternativa utilizando o servidor <i>WebSocket</i> . . . . .	47
4.8.2.2	Alternativa com <i>streaming</i> do video . . . . .	47
4.8.2.3	Alternativa com conexão direta . . . . .	49
4.8.2.4	Comparação das alternativas . . . . .	50
4.8.3	Escolha da arquitetura . . . . .	50
4.8.4	Especificação do <i>hardware</i> e <i>software</i> . . . . .	52
<b>4.9</b>	<b>Reconhecimento de Gestos</b> . . . . .	<b>53</b>
<b>5</b>	<b>TESTES</b> . . . . .	<b>57</b>
<b>5.1</b>	<b>Roteiro de testes</b> . . . . .	<b>57</b>
<b>5.2</b>	<b>Playtesting</b> . . . . .	<b>58</b>
<b>5.3</b>	<b>Observações</b> . . . . .	<b>59</b>
<b>6</b>	<b>CONSIDERAÇÕES FINAIS</b> . . . . .	<b>61</b>
<b>6.1</b>	<b>Principais resultados</b> . . . . .	<b>61</b>
<b>6.2</b>	<b>Contribuições</b> . . . . .	<b>61</b>
<b>6.3</b>	<b>Dificuldades</b> . . . . .	<b>62</b>
<b>6.4</b>	<b>Trabalhos futuros</b> . . . . .	<b>62</b>
	<b>REFERÊNCIAS</b> . . . . .	<b>63</b>

## **APÊNDICES** 67

**APÊNDICE A – EXPERIMENTO - CUBO DE GAIA . . . . . 69**

**APÊNDICE B – GAME DESIGN DOCUMENT . . . . . 73**

# 1 Introdução

O emprego de jogos digitais como ferramenta de auxílio na educação é um conceito estudado há décadas. Tentativas de criar *software* que tome proveito deste tipo de mídia para ensinar determinados tópicos ou fomentar o desenvolvimento de habilidades específicas em crianças existem desde ao menos a década de 1970, com a criação do Consórcio de Computação Educacional de Minnesota (MECC), cujo objetivo era auxiliar escolas do estado de Minnesota, nos EUA, a integrar o uso de computadores ao ensino, de acordo com (LUSSENHOP, 2011).

Além de prover infraestrutura e computadores às escolas locais, o MECC também incentivou a criação de inúmeros jogos educacionais. Dentre esses destaca-se o *Oregon Trail*, possivelmente um dos mais clássicos e conhecidos jogos educacionais, que fora criado por três professores para ensinar aos alunos como era a vida dos colonos do século XIX nos EUA.

Embora, de acordo com (POLI et al., 2012), tenha sido demonstrado em alguns casos que o uso de jogos em sala de aula pode ser benéfico para o envolvimento e desempenho dos alunos, proporcionando um aumento de até 3 horas extras por semana de contato com o material da classe e até 5% de notas mais altas, a utilização de jogos em sala de aula ainda é baixo, quase inexistente. Ainda de acordo com (POLI et al., 2012), as principais causas disso são a dificuldade em manter o engajamento de alunos, a falta de conteúdo complementar e o orçamento limitado.

Considerando que jogar videogames é a quarta atividade mais comum entre adolescentes no Brasil, como visto em (FLEURY; NAKANO; CORDEIRO, 2014), com 35% das crianças entre 9 e 16 anos jogando diariamente, 45% jogando pelo menos uma vez por semana e 19% jogando uma vez ao mês, não se pode negar o potencial deste meio quanto ao seu poder de disseminação de conhecimento e cultura, de uma maneira similar àquela de peças teatrais, literatura e filmes.

Com o amadurecimento e ubiquidade da tecnologia móvel, com 75.2% da população e 49.9% das crianças entre 10 e 14 possuindo um celular no Brasil, segundo dados de (IBGE, 2015), é necessário um estudo aprofundado sobre alternativas para os jogos educacionais tradicionais, que combinem tecnologia e conhecimento para criar uma solução imersiva e efetiva.

Recentemente, o tópico do desenvolvimento de *video games* educativos tem voltado a estar em pauta, sobretudo dada a grande revolução tecnológica que tem ocorrido no campo de mídias interativas: a criação de dispositivos de realidade virtual. Iniciativas como a Nearpod VR (NEARPOD, 2016), que permitem a realização de apresentações e

excursões virtuais, preparam o caminho para o desenvolvimento de novas e mais sofisticadas soluções educativas em realidade virtual.

É inserido neste panorama que este trabalho se propõe a explorar a concepção e desenvolvimento de um *video game* que se valha dos potenciais imersivos de tecnologias que realidade virtual para criar um ambiente fértil para aprendizado lúdico e efetivo.

## 1.1 Objetivos

A partir da realização deste projeto, pretende-se dar os primeiros passos rumo à criação de jogos efetivamente educacionais, capazes de manter o engajamento e interesse de estudantes, ao passo que ensinam sobre tópicos e habilidades relevantes para seu desenvolvimento intelectual e psicológico.

Utilizando tecnologias de realidade virtual, espera-se desenvolver um jogo digital que se mostre recreativo e imersivo, e que permita a alunos aprender e se familiarizar com os conceitos ensinados por conta própria, através de experimentação e exploração, de uma maneira que possa ser supervisionada, auxiliada e acompanhada por profissionais da área da educação, dentro e fora da sala de aula.

Também pretende-se dar preferência ao uso de tecnologias e dispositivos de baixo custo de adoção, de modo a chegar a uma solução final que possa ser implementada em larga escala por escolas e instituições educacionais sem impactos orçamentários proibitivos.

## 1.2 Justificativa

De acordo com a prova *PISA*, feita para mensurar o desempenho acadêmico de alunos de 15 anos, ministrada pela *Organisation for Economic Co-operation and Development* (OCDE), o Brasil ocupa a 58<sup>a</sup> posição dentre os 65 países estudados quanto a performance de seus estudantes em matemática, com uma nota média de 391, contra uma média global de 494, como demonstrado em (OECD, b). É importante notar que a nota da prova é normalizada para que a média de cada categoria seja 500, com desvio padrão de 100 pontos.

O estudo encontrado em (OECD, a), da mesma organização, também revela que dentre os principais problemas que levam um aluno a ter uma performance abaixo da média destacam-se a falta de engajamento com a matéria em questão, o baixo tempo investido em estudo e atividades extra-curriculares, e a baixa auto-estima devido a más experiências anteriores envolvendo a matéria.

Nestes termos, acreditamos ser importante pensar em maneiras de encorajar os

alunos a estimular seu raciocínio lógico, criando um espaço onde crianças possam se engajar e formar conexões positivas com disciplinas na área das ciências exatas, preparando-as rumo a um melhor desempenho nestas matérias ao médio e longo prazo.

### 1.3 Resultados esperados

Ao término deste projeto espera-se ter um produto final no formato de um jogo digital em realidade virtual, que possa ser utilizado por educadores para auxiliar no ensino e retenção habilidades de raciocínio lógico para crianças em idade escolar; bem como um conjunto de especificações técnicas dos *softwares* e *hardwares* necessários para sua implementação em sala de aula.

Espera-se também ter dados no formato de uma pesquisa qualitativa que indiquem uma confirmação ou negação da eficácia do nosso produto como ferramenta de ensino capaz de educar e entreter crianças.

### 1.4 Trabalhos relacionados

(TABUTI; ROCHA; NAKAMURA, 2010) estudam diferentes interações humano-computador, focando em como elas ajudam, ou atrapalham, o aprendizado em ambientes digitais. O estudo foi feito com base no cubo de Rubik (também conhecido como cubo mágico), tanto em sua forma física quanto em várias formas digitais.

(FRADE; ALIXANDRE; SOUZA, 2015) desenvolveram um jogo digital em realidade virtual, motivados pela baixa performance de alunos em aulas de matemática. O foco do jogo é ajudar o jogador a aprender e reter conceitos apresentados em aula, através do uso de ambientes tridimensionais e com o auxílio de personagens e mundo visualmente atraentes. O jogo consiste em um mundo aberto e uma personagem controlável. O objetivo do jogo é conseguir o maior número de moedas o mais rápido possível, enquanto desviando de inimigos. Quando o jogador coleta uma moeda, aparece uma tela com uma questão de múltipla escolha. Caso o jogador acerte a pergunta, recebe a moeda colecionada, caso contrário é teleportado para o início da fase. O jogo pode ser dividido em dois estados: um estado de 'jogo', quando o jogador está caçando as moedas, e um estado de 'aprendizado', quando o jogador está respondendo uma questão.

(ALVES; SOUZA; SOUSA, 2015) também desenvolveram um jogo em realidade virtual, mas focando no ensino de química. Os gráficos são muito mais realistas do que o exemplo anterior, refletindo o público alvo mais maduro. O jogador controla uma personagem que é uma antropomorfização de um átomo de hidrogênio, a qual deve explorar um mundo (baseado na tabela periódica). Cada área do mapa contém elementos diferentes, que interagem com o jogador de maneiras únicas. O objetivo do jogo é auxiliar o ensino de

reações químicas ao jogador. Através da transformação dos elementos em personagens, os autores tinham como objetivo ajudar o jogador a entender a química através de metáforas, contrapondo o exemplo anterior, onde o jogo é dividido em estados.

## 1.5 Estrutura do trabalho

O Capítulo 1 é essa introdução, que apresenta os objetivos, justificativas e motivações desse documento, assim como trabalhos relacionados.

O Capítulo 2 trata da teoria necessária para se estabelecer um vocabulário comum e bem definido, explicando conceitos e as tecnologias utilizadas na solução apresentada por este trabalho.

O Capítulo 3 explicita a metodologia utilizada neste trabalho para se definir, desenvolver e testar o jogo. Explica-se o processo iterativo de design do Ciclo Formal.

O Capítulo 4 relata o desenvolvimento do jogo, desde sua concepção até sua implementação. Contém também especificações importantes, como a definição da arquitetura escolhida e as razões para tal.

O Capítulo 5 é referente aos testes realizados com jogadores, a fim de validar o jogo. Nele se explica o roteiro de testes elaborado, assim como as observações da sessão de testes.

O Capítulo 6 traz as considerações finais do trabalho, mencionando os resultados alcançados, as dificuldades encontradas e trabalhos futuros.

## 2 Teoria

O desenvolvimento de um jogo educacional, que funcione como uma ferramenta de auxílio ao aprendizado em sala de aula, deve envolver a consideração de diversos aspectos pedagógicos, sociais e de *game design*.

Em primeiro lugar, é necessário o estabelecimento de um vocabulário comum e bem definido, delineando conceitos relevantes que possam ser empregados ao longo da formulação e desenvolvimento do projeto.

### 2.1 Jogos digitais

Para (CORREIA et al., 2009), jogos digitais são aqueles aonde exista interação humano-computador, com o uso da tecnologia, e desenvolvidos para serem jogados em computadores, consoles ou outros dispositivos tecnológicos.

Neste tipo de jogo, as regras, restrições, condições de derrota e vitória são estabelecidas por um sistema descrito comumente em *software*, e expostas para o jogador por um dispositivo de *output*, via de regra um *display*, auxiliado de dispositivos de geração de som. O *input* do jogador, descrevendo as ações que este deseja tomar, é enviado por meio de um dentre muitos tipos de dispositivos controladores, como *joypads*, teclados e telas de toque.

### 2.2 Jogos educativos

Jogos podem ser utilizados para auxiliar a assimilação de informações, permitindo o aprendizado através de estratégias de ensino diferentes daquelas exploradas em sala de aula, conforme demonstrado por (FERNANDES, 2012). Isto ocorre, em parte, por incentivar o aprendizado através da prática de atividades.

(CORREIA et al., 2009) comprovam que a utilização de jogos na área de educação é importante não só pela motivação de se jogar um jogo, mas também pelo aprendizado através da exploração que ocorre. O desenvolvimento de jogos educacionais, porém, ainda é um desafio, devido à dificuldade de se juntar dois conceitos que parecem, a primeira vista, ser discordantes: 'Educação' e 'Jogo'.

Para (RAMOS, 2013), a inserção dos jogos digitais na educação contribui para o "desenvolvimento de aspectos cognitivos que são fundamentais para a aprendizagem", favorecendo o desenvolvimento de habilidades relevantes ao processo de aprendizagem. Adicionalmente, devido à grande quantidade de jogos digitais existentes, sua inclusão na

sala de aula permite o acesso a um acervo muito grande e diversificado de novo conteúdo, com um custo baixo, que contribui com a “inclusão e alfabetização digital dos alunos”.

Os jogos que desafiam a lógica, considerado por (RAMOS, 2013) aqueles que “mobilizam o jogador a pensar, levantar hipóteses, experimentar, planejar, testar, realizar cálculos”, contribuem uma melhoria no desenvolvimento do raciocínio lógico, melhorando também habilidades de planejamento e atenção, além da percepção visual.

## 2.3 Mecânicas, dinâmicas e estética

Segundo (HUNICKE MARC LEBLANC, 2004), um jogo, digital ou não, pode ser analisado através de sua decomposição entre três componentes principais: mecânicas, dinâmicas e estética.

Mecânicas são entendidas como as regras efetivas do jogo, compondo as liberdades concedidas e restrições impostas ao jogador, bem como seus objetivos, condições de vitória e derrota. As mecânicas são a porção do jogo sobre a qual o desenvolvedor tem controle direto, e influenciam a percepção das demais porções por parte do jogador.

Chamam-se de dinâmicas os comportamentos emergentes demonstrados por jogadores em função das mecânicas. Englobam desde comportamentos simples, como a reação de um jogador a obstáculos do cenário, até sofisticadas estratégias a respeito de quando cooperar ou competir em jogos multijogador.

Estéticas definem a experiência essencial que o jogador tem ao longo de uma sessão de jogo. Diz respeito às respostas emocionais, ao envolvimento e à imersão do jogador no mundo virtual criado pelo desenvolvedor, e é útil para categorizar o jogo quanto às sensações que ele é capaz de invocar.

## 2.4 Realidade virtual

A realidade virtual é, de acordo com (KIRNER; SISCOUTTO, 2007), uma interface de usuário avançada, que permite que o usuário veja, move e interaja, em tempo real, com um ambiente tridimensional, através de dispositivos especiais. Em geral, a visão tende a ser priorizada em interfaces de realidade virtual, mas a audição e o toque são também muito importantes para a experiência do usuário.

Para (KIRNER; KIRNER, 2011), um mundo virtual visto através de uma tela é considerado não-imersivo, enquanto a percepção deste mundo virtual através de salas com multiprojeção ou por meio de capacetes é considerado imersivo. Desta forma, a realidade virtual:

- a) considera a entrada e saída feita com equipamentos capazes de manipular informação multisensorial em tempo real;
- b) prioriza interação em tempo real sobre a qualidade das informações;
- c) requer processamento gráfico, háptico e sonoro em alta escala;
- d) leva em consideração que ações ocorrem em um espaço tridimensional;
- e) considera que equipamentos especiais para a interação multisensorial sejam utilizados;
- f) entende que o usuário necessita de um período de adaptação para entender a representação virtual do mundo.

## 2.5 Competências e habilidades do raciocínio lógico

O Ministério da Educação (MEC) define um conjunto de parâmetros que devem estar presentes no currículo de escolas em todo o Brasil. Estes são os Parâmetros Curriculares Nacionais (PCN), e abordam várias diferentes áreas importantes.

As competências e habilidades de raciocínio lógico consideradas para análise consistem de um subconjunto dos PCN, em especial, daqueles relativos ao raciocínio lógico, e são apresentados no Quadro 1, retirado de (TABUTI; NAKAMURA, 2015).

Quadro 1 – Competências e habilidades relacionadas ao raciocínio lógico.

Competência	Habilidade	Descrição da habilidade
Análise	H1	Habilidade em resolver um problema complexo dividindo-o em subproblemas mais simples, com soluções mais imediatas
	H2	Habilidade da recomposição dos resultados gerados para a solução do problema mais complexo
Síntese	H3	Habilidade em juntar informações e dados de um problema que sejam de diferentes naturezas
	H4	Habilidade em avaliar a deficiência dessas informações para determinar a solução
	H5	Habilidade em descobrir a falta de outras informações necessárias para a resolução do problema
	H6	Habilidade em priorizar essas informações para o desenvolvimento até atingir a solução
	H7	Habilidade em ordenar essas informações de forma a obter uma sequência de desenvolvimento até atingir a solução
	H8	Habilidade em descobrir padrões em um conjunto de informações
	H9	Habilidade em aplicar novamente determinada informação de forma a agregar novas informações a estrutura de padrões existente
Inferência	H10	Habilidade em aplicar novamente determinada informação de forma a agregar novas informações num conjunto que preserve a estrutura de padrões existente

Fonte: (TABUTI; NAKAMURA, 2015)

As três competências demonstradas na tabela são importantes para a resolução de problemas. O desenvolvimento da competência de análise permite a resolução de problemas através da divisão em problemas menores que tenham soluções mais simples, assim como a recomposição dos resultados gerados para solucionar o problema mais complexo. O desenvolvimento da competência de síntese permite combinar e ordenar diferentes informações obtidas, afim de chegar a uma solução geral. O desenvolvimento da competência de inferência, por sua vez, permite reutilizar informações, padrões e relações aprendidas em situações anteriores para solucionar problemas futuros.

## 2.6 Público alvo

Uma vez estipulado que o foco do projeto seria na criação de um jogo que auxiliasse no aprendizado e retenção de habilidades do raciocínio lógico, e que este jogo seria criado para uma plataforma de realidade virtual, foi necessário refinar cuidadosamente a escolha de público alvo de modo a garantir que jogadores estivessem dentro de uma faixa etária para a qual experiências de realidade virtual fossem manejáveis, e para a qual a aprendizagem de habilidades de raciocínio lógico fosse tão benéfica quanto possível.

### 2.6.1 Adaptabilidade à realidade virtual

Uma consideração a ser feita foi a capacidade de crianças mais novas de compreender e interagir com dispositivos de realidade virtual.

Enquanto muitos dos dispositivos encontrados no mercado atualmente sugerem limites mínimos para a idade de seus usuários na faixa dos 12 e 13 anos de idade (SEETO, 2016), o pesquisador da Universidade da Califórnia Martin Banks afirma que não existem comprovações de malefícios específicos a crianças advindos do uso de equipamentos de realidade virtual, e que quaisquer efeitos adversos que venham a incorrer sobre crianças novas fazendo uso destes equipamentos seriam “os mesmos que aqueles sofridos por adultos” (HILL, 2016).

Banks publicou também em 2008 um artigo (HOFFMAN et al., 2008) detalhando como *displays* tridimensionais e, por extensão, *headsets* de realidade virtual, não acarretariam nos malefícios causados por “*near work*” - ou seja, atividades de leitura ou foco visual em páginas e *displays* muito próximos ao rosto - uma vez que este tipo de *display* gera pontos focais virtuais a uma distância confortável dos olhos, muito além da distância do *display* em si.

Apesar de não haver um limite rígido para a idade mínima para interação com realidade virtual, como será necessário que as crianças consigam compreender e interagir com ambientes tridimensionais, estipulou-se 8 anos de idade como o mínimo necessário para fazer uso do jogo desenvolvido, pois a partir desta idade pode-se considerar completo

o domínio da criança sobre suas habilidades motoras, segundo o *Children's Therapy & Family Resource Centre* (CTFRC, 2016).

### 2.6.2 Desenvolvimento do raciocínio lógico

Conforme a teoria construtivista de Jean Piaget, psicólogo responsável por algumas das teorias mais reconhecidas no campo de desenvolvimento cognitivo infantil, o estágio de desenvolvimento de uma criança compreendido entre os 7 e 11 anos de idade, chamado de estágio operatório concreto, é um momento crucial para o desenvolvimento das noções iniciais do raciocínio lógico.

Em A Teoria do Desenvolvimento Cognitivo de Piaget (GINSBURG; OPPER, 1969), uma análise de diversos textos de Piaget, é postulado que no estágio operatório concreto, crianças começam a ser capazes de utilizar lógica indutiva para formar generalizações e estabelecer relações entre eventos. Estes conceitos são a base das capacidades necessárias para resolver pequenos problemas lógicos, compatíveis com as competências delineadas nos PCNs escolhidos.

Com estas questões em mente, pretendeu-se então projetar um jogo voltado para alunos do ensino fundamental, entre 8 a 12 anos, numa faixa etária suficiente para interagir com as tecnologias empregadas e essencial para a formação de suas capacidades de raciocínio lógico.

## 2.7 Tecnologia

Para se desenvolver este projeto, foi necessário o estudo de algumas tecnologias, apresentadas a seguir.

### 2.7.1 Protocolo *WebSocket*

O *WebSocket* é um protocolo de comunicação full-duplex que pode ser implementado sobre uma conexão TCP, e foi padronizado em 2011 conforme descrito no RFC 6455 (FETTE; MELNIKOV, 2011). Ele foi desenvolvido para ser implementado em navegadores e servidores web, com o intuito de ampliar as possibilidades de interação entre um navegador e um servidor. O protocolo permite que ambos os lados enviem mensagens ao outro a qualquer momento, sem a necessidade de um lado solicitar a mensagem. Mesmo tendo sido desenvolvido para uso em navegadores, o protocolo pode ser utilizado por qualquer aplicação de cliente e servidor.

Outro termo bastante relacionado ao *WebSocket* é o conceito de *Push Notifications*, que são as mensagens enviadas de um servidor para um aplicativo executando em um celular para notificar o usuário de algum evento. O interessante é que o aplicativo não

precisa ficar consultando o servidor (ex. *polling*) para conseguir receber a mensagem, pois esta é enviada ao cliente quando o servidor quiser.

### 2.7.2 Controlador *Leap Motion*

O controlador *Leap Motion*, da empresa de mesmo nome, começou a ser desenvolvido em 2008, contou com várias rodadas de investimento, e a primeira versão do *Software Development Kit(SDK)* foi lançada em maio de 2012. Dois anos depois, foi apresentada a segunda versão do *SDK*, que melhorou bastante o rastreamento das mãos, e que é atualmente a versão mais utilizada. No início de 2016, a mais nova versão, chamada *Orion*, foi lançada, desenvolvida especialmente para a realidade virtual, de acordo com (LEAPMOTION, 2016e)

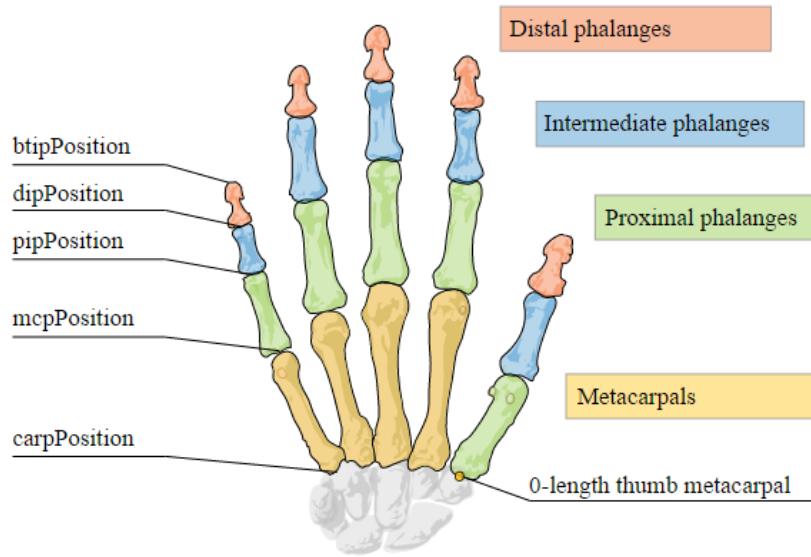
Para utilizar o controlador, são necessários três elementos diferentes: O hardware do controlador em si; um software rodando como um serviço em um computador, agindo como um servidor; e um software cliente que receberá os dados do serviço para utilizá-los, como pode ser visto na arquitetura do *Leap Motion* (LEAPMOTION, 2016d)

O controlador nada mais é do que um conjunto de duas câmeras e três LEDs, todos infravermelhos, de acordo com (LEAPMOTION, 2016b). Os LEDs projetam luz infravermelha, que é refletida pelas mãos do usuário e captadas pelas câmeras. Estas imagens são enviadas ao serviço, que utiliza algoritmos de visão computacional para gerar modelos tridimensionais das mãos do usuário. A versão v2 do software aumentou muito a precisão do controlador por começar a trabalhar com um modelo esquelético da mão, baseado na anatomia humana, ou seja, um modelo composto por cinco dedos, cada um contendo quatro ossos. Este método insere restrições ao modelo, eliminando poses e movimentos que não seriam fisicamente possíveis.

O software cliente, então, pode receber os dados do serviço por duas interfaces: conversando diretamente com o serviço, utilizando uma biblioteca que acompanha o software, ou acessando o servidor *WebSocket* que o serviço roda localmente.

A versão mais recente do *SDK* ainda é nova, e tem focado na otimização dos algoritmos utilizados em várias áreas, tanto no processamento de imagens quanto outros algoritmos de suporte, como a detecção da adição ou remoção de um controlador. Visto a sua orientação para a realidade virtual, uma otimização importante foi em relação à mãos ocluídas, que sumiam em versões anteriores, conforme descrito em (LEAPMOTION, 2016e). É possível que comece a focar em coisas mais específicas para a realidade virtual em pouco tempo.

Figura 1 – Modelo baseado na anatomia humana utilizado pelo controlador *Leap Motion*



Fonte: (LEAPMOTION, 2016c)

### 2.7.3 Reconhecimento de gestos

Conforme mencionado na Subseção 2.7.2, a versão v2 em diante do controlador *Leap Motion* utiliza um modelo da mão baseado na anatomia humana e pode ser visto na Figura 1, retirada de (LEAPMOTION, 2016c). Isto significa que o rastreamento das mãos e dedos ficou mais preciso, pois a imagem capturada pelo controlador é processada e encaixada em um modelo anatomicamente correto. Desta forma, qualquer mão capturada terá cinco dedos, composto do número certo de ossos, e em posições e rotações anatomicamente possível. Isto também melhorou a oclusão de dedos, que antes sumiam do modelo da primeira versão (pois o modelo não teria necessariamente cinco dedos), agora podem ser estimados a partir do resto da mão.

A segunda versão do *SDK* do *Leap Motion* tinha, nativamente, o conceito de ‘gestos’, e o próprio *SDK* disponibilizava alguns métodos para checar se alguns gestos básicos haviam sido executado. Eram quatro o número de gestos reconhecidos: fazer um círculo com o dedo, deslizar o dedo, ‘pressionar’ um botão e tocar a tela, conforme descrito em (LEAPMOTION, 2016a). A versão *Orion*, porém, no momento em que se iniciou o desenvolvimento deste trabalho, ainda não contava com a detecção automática de gestos, sendo necessário que o grupo desenvolvesse o código que o fizesse.

Dados os modelos das mãos, é possível detectar gestos através da posição, rotação e velocidade da palma da mão, dos dedos, ou dos ossos individuais. Para a detecção de um dedo esticado, por exemplo, pode-se checar se o ângulo entre cada par de ossos adjacentes daquele dedo é *pequeno* (aonde *pequeno* deve ser definido com testes). Para se detectar uma palma virada para cima, pode-se verificar o vetor normal da palma (que nos é dada

pelo *SDK* do controlador) e ver se seu componente em y é positivo (tomando y como o eixo perpendicular ao plano do solo). Uma palma aberta apontando para cima, portanto, pode ser detectada com o conjunto da detecção de um dedo esticado e da palma virada para cima.

É importante também notar que um mesmo gesto ou posicionamento da mão pode ser descrita de várias formas. No caso do dedo esticado, por exemplo, poderia-se checar apenas o ângulo entre o primeiro e último ossos. Poderia, também, se calcular a distância entre estes dois ossos. Dependendo do caso, as formas diferentes podem ser equivalentes, mas também podem apenas parecer iguais, mas quando testados em ambientes diferentes, resultarão em saídas diferentes. Por exemplo, uma mão menor poderia não ter o dedo considerado esticado caso se utilizasse a distância entre os ossos.

Mais recentemente, porém, a empresa do *Leap Motion* lançou módulos (chamados de *utilities* no site) que permitem a detecção de gestos simples, não sendo mais necessário desenvolver código que o faça.

#### 2.7.4 Google Cardboard

O *Google Cardboard* é um projeto da empresa Google criado em 2014 para fomentar o desenvolvimento de aplicações de realidade virtual. É composta por um *headset* e de um *software* compatível. A ideia principal do projeto era conseguir proporcionar a experiência de realidade virtual da forma mais barata o possível, conforme dito em (CNET, 2016). Para alcançar tal objetivo, o *headset* é feito de papelão, de tal forma que o único subcomponente mais caro seja as lentes. Adicionalmente, um celular é utilizado tanto como poder de processamento do *headset* quanto como tela. Visto que uma grande parcela da população já tem um *smartphone*, estas partes do *Cardboard* vêm de graça.

Quanto ao software compatível, é necessário que este divida a sua saída de vídeo (que será mostrado na tela do celular) em metades, uma para cada olho. Visto que cada imagem emula a visão de um olho, é necessário que as imagens venham de pontos de origem separados por uma distância equivalente àquela entre as duas pupilas, valor que varia entre 58 mm e 70 mm, de acordo com (DODGSON, 2004). Adicionalmente, devido à distorção causada pelas lentes, é necessário que o software aplique a distorção inversa na imagem, de forma que a imagem fique correta após passar pela lente. É também necessário ler sensores encontrados no dispositivo móvel, como acelerômetros e giroscópios, e disponibilizar estes dados para que se possa implementar o movimento do usuário dentro do jogo ou aplicativo.

O *SDK* do *Google Cardboard*, distribuído pela própria Google e disponível online em (GOOGLE, 2016b), já faz todas estas transformações necessárias na imagem, além de enviar os dados de movimento, já processados, de volta ao software que esteja utilizando o *SDK*.

# 3 Metodologia

A metodologia empregada na realização deste projeto foi o processo iterativo de design do Ciclo Formal (SCHELL, 2010), um método que alia conceitos de desenvolvimento ágil advindos da engenharia de software com noções modernas de *game design*. A adoção desta metodologia se deu pelo seu enfoque em frequentes testes e rápida iteração sobre o projeto, processos imprescindíveis no desenvolvimento de um jogo que atenda aos critérios educacionais estabelecidos, no limitado escopo estipulado.

No contexto deste projeto, a aplicação do Ciclo Formal pode ser descrita nos termos de duas etapas principais: formulação do problema e implementação iterativa.

## 3.1 Formulação do problema

Na etapa inicial de criação do projeto, é necessário enunciar cuidadosamente a problema a ser resolvido, levando em consideração os objetivos do jogo, sua demografia e o contexto em que ele será jogado. Também é necessário analisar os recursos disponíveis e o prazo para a criação do produto final, afim de definir um escopo claro para o projeto. A partir do levantamento dessas informações, têm-se os parâmetros a serem usados nas próximas etapas de criação do projeto.

Este problema é documentado na forma de um *Game Design Document* (GDD), que servirá tanto como um relatório quanto como um ponto de referência para a realização do projeto nas etapas seguintes.

## 3.2 Implementação iterativa

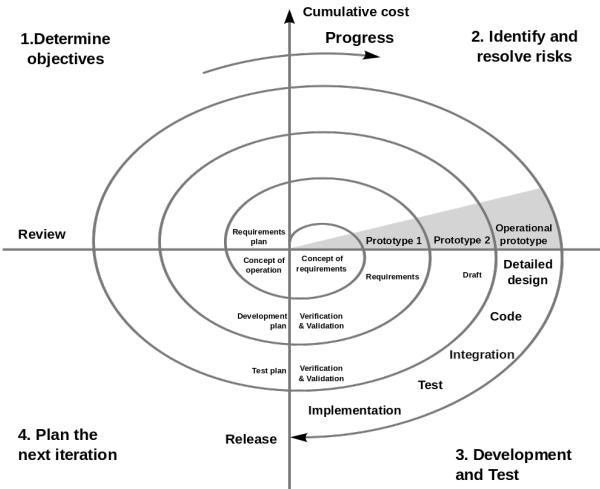
Para a implementação do jogo em si, adota-se a metodologia de design iterativo, consistente de uma aplicação do ciclo de desenvolvimento de software de Boehm (Figura 2) no âmbito de projetos de jogos digitais.

Nos termos dessa metodologia, um jogo digital deve ser implementado por meio de um processo iterativo, ciclando entre seis fase principais: *Brainstorming*, escolha da solução, análise de riscos, prototipação, teste e análise e refinamento do *design*.

### 3.2.1 *Brainstorming*

Nesta fase são propostas diversas potenciais soluções para o problema enunciado, com um enfoque na geração rápida do maior número possível de ideias, sendo evitado levantar críticas. As propostas são então listadas para avaliação quanto a seus méritos e

Figura 2 – Ciclo de Boehm.



Fonte: (SCHELL, 2010)

adequação frete ao problema original elaborado, dentre as quais uma será eleita como o conceito do jogo, aprofundada e descrita formalmente em maiores detalhes.

### 3.2.2 Escolha da solução

Uma das soluções propostas é escolhida, de acordo com sua aderência ao problema enunciado e viabilidade técnica. Sua descrição é então adicionada ao GDD, detalhando as características mecânicas, estéticas e tecnológicas. Esta descrição deverá ser tomada como referência para a implementação do jogo, e alterada conforme necessário durante as fases seguintes.

### 3.2.3 Análise de riscos

Durante a análise de riscos, um elemento específico do design deve ser considerado. Os potenciais riscos e dificuldades de implementação associados a esse segmento devem então ser levantados e listados, considerando as dinâmicas que pretende-se que esse elemento gere, sua aderência aos objetivos estipulados e eventuais problemas técnicos advindos de seu desenvolvimento.

### 3.2.4 Prototipação

Nesta fase, procura-se elaborar rapidamente um protótipo que funcione como prova de conceito para o elemento recortado na fase anterior, de forma a eliminar ou mitigar os riscos observados. A ênfase deve ser na velocidade de implementação do protótipo, não em seu polimento ou reusabilidade dos recursos gerados em etapas posteriores do projeto.

### 3.2.5 Teste

Na fase de testes, o protótipo construído é avaliado quanto aos riscos delimitados na Fase 3 (Subseção 3.2.3). A abrangência dos testes, variará de acordo com a natureza do elemento sendo testado e o atual estado de desenvolvimento do projeto como um todo, sendo que em testes realizados em projetos em etapas mais avançadas de seu desenvolvimento pedirão maior envolvimento de parcelas representativas do público alvo do produto final.

A adequação do elemento considerado aos objetivos esperados e sua eficácia na resolução dos problemas levantados determinarão o rumo das demais iterações do processo.

### 3.2.6 Análise e refinamento do design

Durante esta fase, uma análise das fases anteriores deve ser feita, procurando-se ressaltar os motivos que levaram a eventuais inadequações do protótipo executado aos riscos previamente delimitados. Com base nos resultados observados, as modificações necessárias são feitas ao GDD, uma nova descrição do problema é redigida, e retorna-se à Fase 1 (Subseção 3.2.1) para a construção de um novo protótipo com um maior grau de polimento.

Seguindo este método, o jogo toma forma gradativamente, através de iterações e prototipações sucessivas e aditivas, até que sua implementação esteja de acordo com o estado atual do *GDD*, e todos os objetivos definidos tenham sido atingidos.

# 4 Desenvolvimento

Segundo o processo iterativo de desenvolvimento descrito na metodologia, este projeto evoluiu ao longo de várias etapas de crescente complexidade.

## 4.1 Objetivo e formulação do problema

Foi inicialmente elaborada uma enunciação geral do problema a ser resolvido: a criação de um jogo digital que, valendo-se de técnicas de realidade virtual, oferecesse uma experiência imersiva e educativa a crianças de 8 a 12 anos de idade, de acordo com o recorte selecionado descrito no Capítulo 2. Tendo essa enunciação como base, pôde-se decidir a respeito de restrições, e do escopo que guiariam as etapas seguintes de conceituação do projeto.

### 4.1.1 Ferramentas e tecnologias

Em primeiro lugar, optou-se por restringir os equipamentos e tecnologias a serem empregados, de modo a facilitar a acessibilidade e adoção em sala de aula e diminuir os custos de implantação, sem que a experiência desejada tivesse de ser sacrificada. Isso levou à escolha do uso do *Google Cardboard* como dispositivo principal de realidade virtual, por seu baixo custo comparado a demais alternativas de mercado, e sua integração a dispositivos móveis que, em muitos casos, já estariam ao alcance dos alunos, ou poderiam ser providenciados sem um grande impacto orçamentário.

Em adição ao *Cardboard*, optou-se pela implementação do dispositivo detector de gestos manuais *Leap Motion* como forma de *input* principal para o jogo, com a vantagem de ser uma alternativa de custo aceitável, capaz de providenciar uma experiência bastante direta na interação do aluno com o mundo do jogo, uma vez que permite que movimentos do jogador no mundo real sejam traduzidos em ações do mundo virtual, permitindo abstrair os dispositivos de entrada e saída por completo, e mantendo a imersão e engajamento.

Por fim, como restrição adicional, a ferramenta *Unity* foi escolhida como *engine* de desenvolvimento do jogo, por sua fácil integração ao hardware escolhido, e seu paradigma de desenvolvimento em mais alto nível do que as demais alternativas disponíveis, o que permitiria ao grupo focar-se na conceituação e implementação do jogo em si, sem prender-se a eventuais problemas relativos à configuração e uso do hardware, criação de APIs próprias ou preocupações com elementos mais elementares da criação de um jogo digital.

#### 4.1.1.1 Alternativas consideradas

Alguns outros dispositivos, de função semelhantes aos citados anteriormente, foram considerados como potenciais alternativas àqueles escolhidos para a realização deste trabalho.

Como alternativa para a implementação de controles gestuais, considerou-se duas principais alternativas: o controlador por infra-vermelho *Leap Motion* e o bracelete de controle por contrações musculares *Myo*.

O controlador *Myo*, que funciona medindo a contração e o relaxamento de músculos no braço do usuário, permite detectar gestos e o posicionamento do braço e das mãos. Sua principal vantagem sobre o *Leap Motion* é não estar restrito a uma única área de operação: enquanto o *Leap Motion* precisa que o usuário mantenha suas mãos dentro da área de captura de sua câmera infra-vermelho, o *Myo* é capaz de detectar movimentos como quer que o usuário se posicione.

Contudo, este dispositivo apresentou uma série de restrições que o apontaram como incompatível com o projeto. Primeiramente, enquanto o *Myo* é mais adequado que o *Leap Motion* para a detecção de gestos maiores, que podem extrapolar a restrita área de operação deste, ele é menos capaz de detectar gestos finos e necessários para o controle preciso de um jogo nos moldes projetado.

Além disso o *Myo* tem um custo consideravelmente mais elevado que o *Leap Motion*, custando USD 199,00 (MYO, 2016) por unidade, enquanto o *Leap Motion* está especificado em apenas USD 79,99 (MOTION, 2016). Se considerarmos que para gestos com duas mãos seriam necessários dois braceletes *Myo* por usuário, se custo salta então para USD 398,00.

Quanto a dispositivos de realidade virtual, duas alternativas ao *Google Cardboard* foram trazidas em questão: os *headsets Oculus Rift* e o *HTC Vive*.

Dentre estes o *HTC Vive* oferece a experiência mais precisa, sendo composto por não só um *headset*, mas também sensores individuais a serem dispostos pelo ambiente para detectar movimentação relativa do jogador. Contudo, visto que o jogo contemplado neste projeto visa ser aplicável em sala de aula, um equipamento que necessite um infra-estrutura tão complexa seria incompatível com o que essa estipulação. Há também uma incompatibilidade de preços, pois uma unidade de *HTC Vive* custaria \$799,00 (HTC, 2016), um valor proibitivo para uso em larga escala, como pretendido pelo projeto.

O *Oculus Rift*, por outro lado, não necessita de sensores externos para operar, operando apenas com um *headset*. Apesar de sua performance ser superior à do *Google Cardboard* - visto que aquele é um *hardware* dedicado à realidade virtual enquanto este vale-se somente do uso de um celular para produzir o efeito tridimensional - o *Oculus Rift* ainda possui um preço que extrapola o que foi considerado razoável para implementação

em sala de aula: USD 599,00 (OCULUS, 2016) por unidade.

O dispositivo escolhido, o *Google Cardboard*, pode ser adquirido por a partir de USD 15,00 (GOOGLE, 2016c), uma faixa de preço muito mais compatível com a ideia original do projeto.

#### 4.1.2 Delimitação do escopo

Pelo reduzido tempo de execução disponível para a implementação do projeto, limitada familiaridade com as tecnologias escolhidas, e pelo grupo ser composto apenas por dois integrantes, decidiu-se manter o escopo do projeto pequeno, de forma que o grupo pudesse focar-se em criar um jogo que servisse como prova de conceito, fizesse bom uso da mídia e dos dispositivos empregados, e implementasse mecânicas diversificadas o suficiente para que testes posteriores pudessem ser realizados quanto à sua eficácia.

Enquanto não ficaram previamente estipulados em termos específicos a quantidade de mecânicas a ser implementadas, a extensão do conteúdo criado e a sofisticação artística pretendida para o produto final, ficou claro que o desenvolvimento do jogo deveria ser focado na sua realização técnica, e que não seria possível um alto grau de refinamento e pós-produção equiparável àquele de jogos digitais disponíveis no mercado. Estas restrições moldaram significativamente as decisões tomadas durante as etapas posteriores de *brainstorming* e desenvolvimento do jogo.

### 4.2 *Brainstorming* e seleção do conceito do jogo

Com a enunciação do problema e as restrições de implementação bem definidas e acordadas pelo grupo, pôde-se passar para a etapa seguinte do desenvolvimento, o *brainstorm*, durante o qual várias ideias foram debatidas e exploradas pelos membros do grupo. Nesta fase, diversas propostas foram feitas para uma solução que fizesse um bom uso dos recursos de realidade virtual disponíveis e gerasse um projeto adequado ao público alvo selecionado, resultando em um jogo intuitivo e divertido que fosse capaz de transmitir os conceitos lógicos e cognitivos relevantes. Importante também era que o jogo exercitasse competências e habilidades do raciocínio lógico, apresentados previamente na Quadro 1.

As ideias geradas variaram desde jogos de um ritmo mais acelerado, no qual um jogador teria que categorizar elementos abstratos o mais rápido que pudesse - segundo características como cor, formato e tamanho - até experiências nas quais pressões de tempo e condições de falha estavam completamente ausentes, como jogos de resolução de quebra-cabeças tridimensionais, e são apresentadas a seguir.

#### 4.2.1 Ideias do brainstorm

Na Figura 3 é possível ver o rascunho das ideias feitas durante o *brainstorm*.

Figura 3 – Primeiro rascunho do jogo



Fonte: Autor

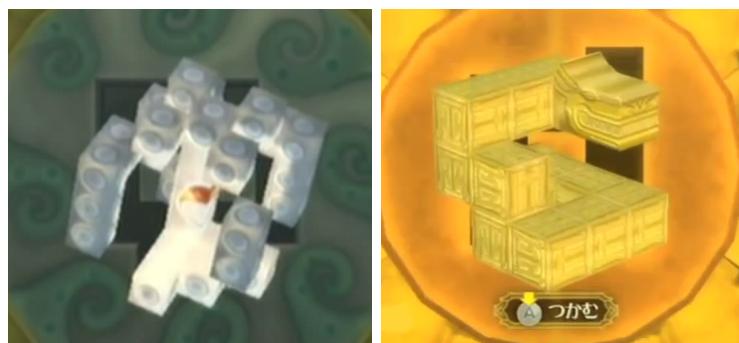
No canto superior direito, vê-se o rascunho de um conceito de jogo aonde peças vão caindo lentamente e devem ser empilhadas em uma bandeja ou superfície controlada pelo movimento das mãos do jogador, que deve movimentar a bandeja de forma que as peças que caem fiquem bem dispostas, em vez de tombar para fora da bandeja. Outra possibilidade é que as peças caindo tenham que se juntar para fazer um formato específico, ou até mesmo uma adaptação do famoso jogo *Tetris*, mas tridimensional, aonde deve-se criar um plano inteiro sobre a bandeja (ao invés de uma linha, como no jogo original). Esta ideia, além de exercitar a coordenação motora do jogador, treina também as competências e habilidades H1, H2 e H7 do Quadro 1, pela necessidade de se resolver o problema atual (de identificar a peça que está caindo e aonde ela deve ser colocada), assim como planejar aonde colocar peças futuras.

No canto superior esquerdo, existe o rascunho de um jogo de classificação. Assim como antes, peças vão caindo, e o jogador deve categorizar estas peças em uma de duas ou mais categorias, através de movimentos das mãos. No exemplo desenhado, o objetivo poderia ser mover todos os cubos azuis para a direita, esferas pretas para a esquerda, e não agir sobre as esferas com espinhos vermelhas. Outras variações poderiam ser feitas, também, como todos os cubos (independente de sua cor) vão para a direita, por exemplo. Existem inúmeras possibilidades, tendo como pré-requisito ser possível categorizar os elementos. Adicionalmente, os exemplos dados foram com formas geométricas, mas po-

dem ser abstraídas para quase qualquer área, como por exemplo elementos (ex. Separe os elementos que reajam com oxigênio), geografia (ex. Categorize os países por continente), história (ex. Classifique estes eventos como causas ou consequências da Segunda Guerra Mundial), ou qualquer outro assunto que se consiga imaginar. Esta alternativa pode exercitar as competências e habilidades H1, H2, H7 e H8 do Quadro 1, dependendo de quanto complexo for o desafio. No caso simples de apenas se classificar formas, por exemplo, apenas a competência H8 será exercitada (além da memória do jogador). Em um caso mais elaborado, aonde a ordem em que se colocam elementos em cada uma das categorias importe, como por exemplo as peças maiores tem que entrar antes de peças menores, as quatro competências listadas podem ser praticadas.

Logo abaixo do rascunho no canto superior esquerdo, estão as frases “Zelda Boss Key” e “Game Show JP”. Estas são referências para uma terceira ideia de jogo, aonde é apresentado ao jogador uma peça tridimensional e uma abertura ou ranhura tridimensional. O jogador deve então, com movimentos da mão, rotacionar ou modificar a peça para que esta encaixe na ranhura/abertura. Dois exemplos da referência do jogo *Legend of Zelda: Skyward Sword* podem ser vistos na Figura 4, aonde é possível ver a peça na frente e, ao fundo, o encaixe. Uma possibilidade mais elaborada é possível, aonde, para se encaixar a peça, é necessário uma série de movimentos, não apenas uma rotação. Nesta opção, seria possível exercitar as competências e habilidades H1, H2 e H8 do Quadro 1, devido à necessidade de se entender a peça tridimensional e como ela encaixa na ranhura. Adicionalmente, este jogo permitiria melhorar o raciocínio espacial do jogador.

Figura 4 – Dois exemplos do *minigame* “Zelda Boss Key” do jogo *Legend of Zelda: Skyward Sword*



Fonte: (YOUTUBE, 2016)(esquerda) e (ZELDAINFORMER, 2016)(direita)

Por último, na Figura 3, no canto inferior esquerdo, é possível ver o rascunho da quarta e última ideia. Neste jogo, os jogadores deparariam-se com um terreno digital composto por acidentes geográficos como planaltos, depressões, corpos d’água e formações magmáticas, e poderiam modificá-lo a partir de gestos manuais simples que controlassem a elevação, precipitação, movimentação do ar, entre outros. O objetivo do jogo é manipular a pequena configuração geológica tridimensional, de forma a chegar a um determinado

estado pré-definido. O terreno seria subdividido em unidades cúbicas que poderiam ser manipuladas individualmente, fornecendo uma resolução suficiente para que configurações diversas e interessantes pudessem ser construídas, mas sem exigir uma precisão incompatível com os dispositivos de entrada e saída escolhidos. Esta última alternativa exercita as competências e habilidades H1, H2, H6 e H7 do Quadro 1, dada a sua necessidade de solucionar desafios por partes, além de conseguir priorizar e planejar os movimentos seguintes necessários para se chegar ao estado pré-definido.

#### 4.2.2 Seleção do conceito do jogo

Por fim, uma análise cuidadosa das limitações das tecnologias escolhidas, aliada a uma preocupação em manter o jogo intuitivo e imersivo levou o grupo a escolher o último conceito dentre os quatro. Os dois primeiros conceitos, por mais interessantes que fossem, tinham uma dependência muito forte sobre a entrada do controlador *Leap Motion*. Dado que o controlador não é completamente preciso e poderia falhar na detecção de movimentos ou na captura da mão do jogador, era necessário prevenir que o jogador fosse punido indevidamente. Visto que os dois primeiros conceitos dependiam dos objetos caindo, uma falha do controlador poderia fazê-los perder, causando frustração. Entre o terceiro e quarto conceito, o grupo escolheu o último, devido a um interesse maior e também pelo número muito maior de possibilidades de interações. O jogador possivelmente perderia o interesse no terceiro conceito após poucas partidas do jogo.

Desta forma, o quarto conceito foi o escolhido para ser desenvolvido. O próximo passo era definir formalmente as mecânicas do jogo, assim como os elementos que o jogador poderia manipular. Estes elementos estão listados no Quadro 2, junto das interações planejadas para cada um.

Para que o jogo não fosse curto demais, foram então imaginados cenários que extrapolassem os usos básicos das mecânicas propostas, de maneiras que se alinhasssem aos conceitos pedagógicos que deveriam ser transmitidos pelo jogo: desafios que dependessem da execução de tarefas em uma determinada ordem, em combinação com outras tarefas paralelas, ou dentro de um determinado limite de tempo. Nestes cenários, era esperado que jogadores conseguissem imaginar soluções mais sofisticadas, indo além das interações básicas às quais teriam acesso direto. Seria necessário movimentar massas de ar para jogar água contra blocos de terra (cujo rascunho é mostrado na Figura 5), formando praias; escavar o solo para alcançar câmaras magmáticos e formar vulcões; ou alterar o curso de corpos d'água para formar cataratas.

Algumas primeiras ideias de possíveis fases também foram discutidas, para dar uma direção para a criação da prova de conceito. A primeira fase lidaria apenas com a mudança da elevação da terra e o jogador teria que criar um buraco e uma torre de terra, algo similar à Figura 6. Desta forma, a primeira prova de conceito já poderia ser jogada

Quadro 2 – Elementos manipuláveis pelo jogador

Elemento	Interação planejada
Terra/Pedra	O principal elemento do mundo. O jogador consegue manipular os blocos de terra modificando sua altura, criando pilares ou buracos
Água	O jogador consegue causar precipitação em uma área pre-determinada, criando corpos d'água, como lagos
Fogo	O jogador pode colocar fogo em uma área pre-determinada, secando corpos d'água que estiverem ali
Vento	O jogador consegue criar correntezas de vento em uma das quatro direções cardinais. Se aplicados em uma configuração específica dos blocos, o vento cria ondas num corpo d'água que erode um bloco de terra/pedra, criando areia
Lava	Esse elemento não pode ser gerado pelo jogador, sendo encontrado no nível mais baixo do mundo em algumas fases. Se um buraco é escavado até este nível mais baixo, a lava começa a subir. Em contato com água, a lava vira terra/pedra
Areia	Criado pela erosão de terra/pedra devido à interação do vento com a água

Fonte: Autor

para se testar a viabilidade do conceito.

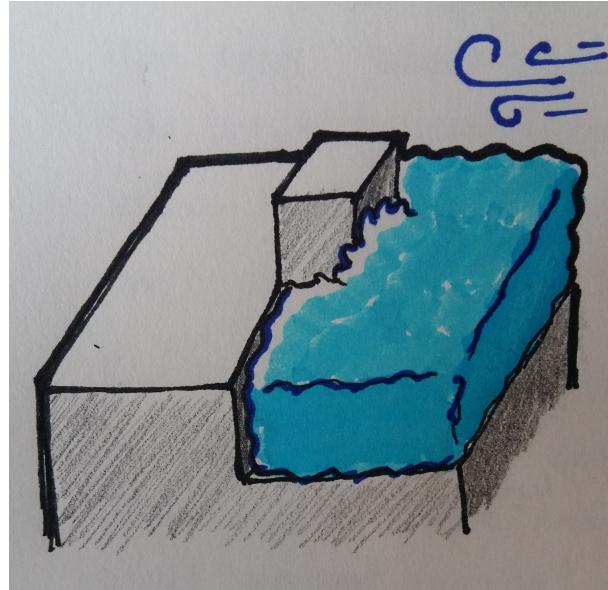
Uma vez escolhido o conceito do jogo, pode-se iniciar o *GDD*, que deve ser atualizado conforme o jogo evolui. O *GDD* pode ser encontrado no Apêndice B.

### 4.3 Primeira iteração: prova de conceito

Uma vez decidido o conceito a ser explorado, um pequeno protótipo foi proposto para dar ao grupo uma noção mais palpável de como esse jogo se pareceria, como seria controlá-lo, que dificuldades imprevistas apareceriam e quais as medidas que precisariam ser tomadas para suplantá-las. Este protótipo inicial tinha como objetivo tanto familiarizar o grupo às ferramentas e tecnologias escolhidas para a realização do trabalho quanto ajudar a nivelar de maneira mais concreta a complexidade de sua implementação.

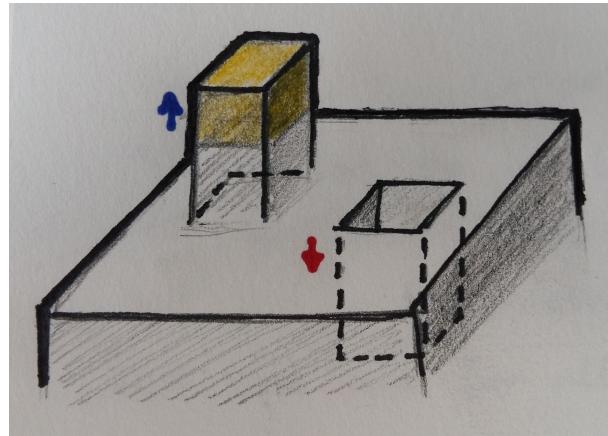
O protótipo deveria se ater apenas a uma mecânica básica da especificação do conceito: controle da elevação do terreno. O jogador teria acesso direto ao jogo - sem passar por telas de menu ou tutoriais - no qual se depararia com uma grade tridimensional de 5x5x5 cubos de terra, os quais poderia elevar ou rebaixar livremente. Um exemplo desta

Figura 5 – Rascunho de como criar uma praia, utilizando os elementos terra, água e ar



Fonte: Autor

Figura 6 – Rascunho de uma possível primeira fase



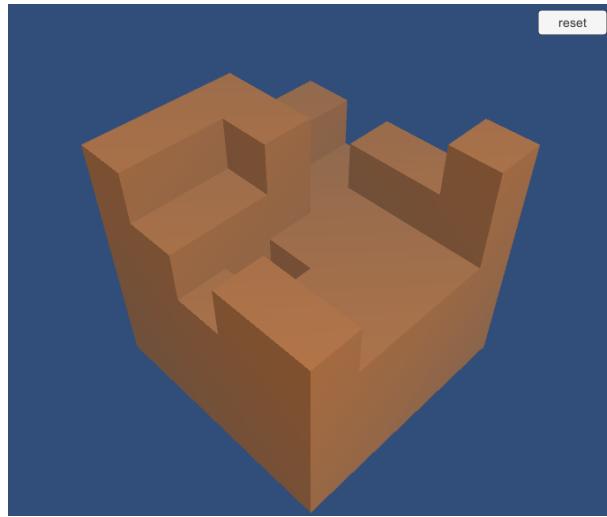
Fonte: Autor

prova de conceito está demonstrado na Figura 7.

Apesar da simples implementação, algumas primeiras dificuldades já puderam ser percebidas nesta iteração, sobretudo na integração dos dispositivos de realidade virtual.

A primeira barreira a ser percebida foi a falta de compatibilidade com dispositivos *Android* da versão mais recente do *SDK* do controlador *Leap Motion*, como havia nas versões anteriores. Isso, aliado ao fato de que a versão anterior do *SDK* não estava mais disponível para download, significou que uma interação direta entre o *Leap Motion* e o jogo compilado para dispositivos móveis não seria possível nessa etapa inicial do projeto. A solução para este problema foi continuar o desenvolvimento diretamente no computador, sem se preocupar em gerar a versão para o *Android* (e, portanto, para o *Google Cardboard*).

Figura 7 – Prova de conceito do jogo



Fonte: Autor

Apesar deste empecilho, *Unity* se mostrou uma plataforma robusta e flexível o suficiente para a continuação do projeto, e as mecânicas básicas puderam ser implementadas rapidamente e sem grandes problemas.

#### 4.4 Segunda iteração: mecânicas básicas

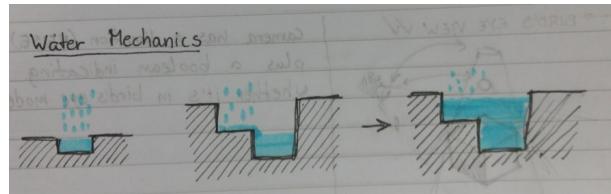
Tendo em mente as facilidades e desafios percebidos durante a criação da prova de conceito, um segundo protótipo foi desenvolvido, desta vez com o intuito de acrescentar uma segunda mecânica básica - controle da precipitação - de modo a permitir que o jogador experimentasse com a interação entre as duas.

Para tal, foi necessário especificar um pouco melhor a mecânica da água e como ela funcionaria em relação às mudanças de altura do terreno. Primeiro, definiu-se que a agua deveria crescer de baixo para cima, de maneira similar a um recipiente sendo preenchido com líquido, demonstrado na Figura 8. Esta era uma definição mais referente ao apelo gráfico do que uma mecânica. Adicionalmente, definiu-se que, dado um bloco (ou conjunto de blocos) selecionados, a ação de precipitação preencheria o espaço aberto até cobrir os blocos selecionados, em vez de preencher de água para que esta fique no mesmo nível do bloco selecionado. Esta mecânica pode ser vista na Figura 9.

Nesta iteração, haveria dois gestos que deveriam ser captados e interpretados pelo *Leap Motion*: abaixar e elevar a mão com a palma aberta para alterar a elevação do terreno e apontar todos os dedos para baixo para fazer chover em uma área específica.

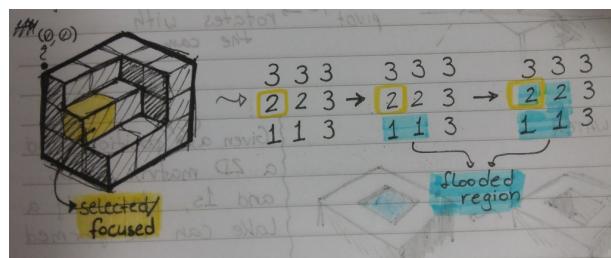
Durante a implementação dessas mecânicas, ficou clara a necessidade de uma mecânica de seleção que permitisse a manipulação de grandes áreas do terreno simultaneamente, então um terceiro gesto foi adicionado: pinçar e arrastar para delimitar áreas.

Figura 8 – Mecânicas da água 1



Fonte: Autor

Figura 9 – Mecânicas da água 2

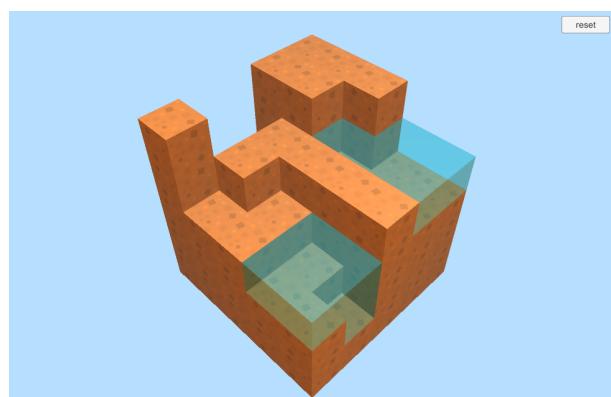


Fonte: Autor

Também nessa etapa, percebeu-se que, devido à posição do *Leap Motion*, alguns dos gestos imaginados para as diferentes mecânicas do jogo - sobretudo o associado a chuva - não poderiam ser captados com a precisão necessária, obrigando o grupo a reimaginar uma parte significativa da interação. Isto resultou na simplificação do gesto para a chuva, bastando apenas fechar o punho com a palma virada para baixo. Desta forma, o gesto era capturado mais facilmente, mas também era ativado muitas vezes quando o jogador não havia feito o gesto, dificultando a jogabilidade. Uma possível razão para isto é ter definido o gesto de maneira não ótima, devido à inexperiência neste assunto por parte dos integrantes do grupo.

Apesar de tudo, o protótipo já estava jogável e era possível modificar o terreno e adicionar ou remover água, como mostrado na Figura 10.

Figura 10 – Prova de conceito do jogo



Fonte: Autor

## 4.5 Terceira iteração: últimas mecânicas

Ao final da segunda iteração, faltavam dois elementos para completar o conjunto de elementos que o jogador poderia criar, de acordo com o que havia sido definido: ar/vento e fogo.

O primeiro é importante para permitir a criação de fases que necessitem uma sequência maior de passos para se chegar ao objetivo e, portanto, criar um desafio maior. Já o segundo é importante por sua reação com a água, que até então, quando criada, não podia ser removida. Desta forma, o fogo serve para desfazer movimentos errôneos de água. Dada a certa imprecisão existente com o gesto de geração de água, aonde o nosso código acusava o jogador de ter feito o gesto erroneamente, o fogo era importante para minimizar a frustração do jogador.

## 4.6 Quarta iteração: refinamentos

Uma vez que os elementos principais foram implementados e o jogo já podia ser jogado, focou-se em possibilitar a execução do jogo no dispositivo *Android* (que estava, até então, rodando no computador, por dificuldades discutidas na Seção 4.3). Visto que a primeira opção, que parecia a mais óbvia, não funcionaria, foi feito um estudo mais a fundo sobre a arquitetura do sistema, que está descrita na Seção 4.8.

Adicionalmente, erros de código foram corrigidos, e melhorias foram feitas na detecção de gestos.

## 4.7 Requisitos

Os requisitos a seguir foram especificados a partir do *GDD*.

### 4.7.1 Requisitos não funcionais

- a) o sistema de reconhecimento de gestos deve tolerar variações de velocidade, ângulo e posição das mãos para aumentar a taxa de reconhecimento dos movimentos executados pelo jogador. Os valores de tolerância serão testados e selecionados para maximizar a usabilidade e a experiência do usuário;
- b) o aplicativo deve ser compatível com *Android* 4.4 (*KitKat*) em diante, por se tratar de 83.5% dos dispositivos *Android*, de acordo com (GOOGLE, 2016a).

### 4.7.2 Requisitos funcionais

- a) o jogador deve poder interagir com o ambiente do jogo por meio de gestos predefinidos;
- b) o jogo deve conter os elementos água, fogo, terra e ar;
- c) deve ser possível selecionar uma área do mundo com a qual se deseja interagir;
- d) o início do jogo deve ser focado em ensinar ao jogador as mecânicas básicas e as interações entre os elementos;
- e) as fases do jogo devem ter sua complexidade aumentada gradativamente, envolvendo cada vez mais mecânicas e interações entre elementos;
- f) algumas fases devem apresentar desafios que envolvam não apenas a escolha adequada de mecânicas e interações, mas também a escolha da ordem em que eles devem ser aplicados;
- g) deve ser possível jogar com o modo de realidade virtual desligado, para os casos aonde o jogador sinta náusea devido ao *headset*;
- h) o mundo deve ser discreto, dividido em blocos bem definidos, que podem ser selecionados e modificados.

## 4.8 Arquitetura

Para se definir a arquitetura do sistema, é antes necessário identificar e entender cada um dos principais elementos, assim como suas interfaces.

### 4.8.1 Identificação dos elementos

#### 4.8.1.1 Controlador *Leap Motion*

Conforme visto na Seção 2.7.2, o controlador pode ser dividido em dois elementos diferentes: O **hardware do controlador** e o **serviço** que aplica algoritmos de visão computacional nas imagens vindas do hardware, transformando-as em dados estruturados.

O serviço pode ser executado no computador ou, no caso de se utilizar a versão v2 do *SDK* do *Leap Motion*, no *Android*.

#### 4.8.1.2 Lógica do jogo

Este é o elemento que age sobre o ambiente virtual a partir de dados de entrada do controlador e do movimento do *headset* de realidade virtual. Este elemento estará implementado no *Unity*, e pode ser executado ou no computador ou no próprio celular.

#### 4.8.1.3 Processamento de saída do video

Este elemento é controlado pelo *SDK* do *Google Cardboard* e cuida de toda a parte de gerar a visão estereoscópica necessária para a visão 3D, além de movimentar a visão do jogo de acordo com o movimento do *headset* de realidade virtual. Este elemento pode ser executado ou no computador ou no próprio celular.

### 4.8.2 Arquiteturas estudadas

As três arquiteturas estudadas têm como entrada as imagens vindas do controlador *Leap Motion* e do movimento do *Google Cardboard*, que são enviadas à lógica de jogo que, então, modifica o estado do jogo e envia para o *Google Cardboard*. As diferentes opções de arquitetura diferem quanto à como a conexão é feita, se é intermediada ou não por um computador, e aonde essa intermediação acontece.

#### 4.8.2.1 Alternativa utilizando o servidor *WebSocket*

Nesta opção, conecta-se o controlador a um computador que esteja rodando o serviço do *Leap Motion*, transformando as imagens do controlador em dados estruturados. O celular que está executando a lógica do jogo então se conecta ao servidor *websocket* gerado pelo serviço do controlador que esta rodando no computador. A partir desta conexão, o servidor envia os dados estruturados ao celular para que estes possam ser utilizados como entradas no jogo. O celular então aplica as transformações necessárias para a saída de vídeo, que é enviada à tela do celular. Esta opção é demonstrada na Figura 11.

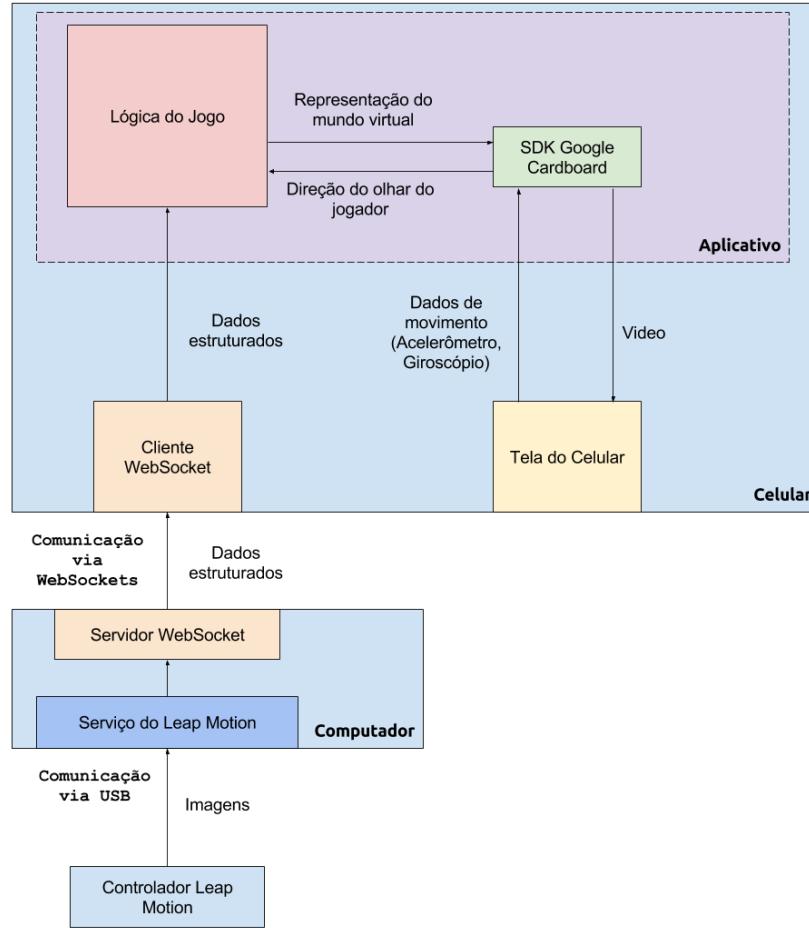
A vantagem desta opção é que podemos utilizar a versão mais recente do *SDK* do *Leap Motion*, que melhorou muito o rastreamento das mãos, comparada à versão v2. Adicionalmente, o serviço já expõe uma interface web para a leitura dos dados estruturados.

Uma desvantagem desta opção, porém, é a necessidade de modificar a biblioteca do *Leap Motion* do *Unity* para que esta aceite conexões web, já que, nativamente, ela aceita apenas conexões via USB. Outra desvantagem é a latência adicionada entre o movimento das mãos do usuário e o momento em que a lógica do jogo recebe tais informações. Se esta latência for alta demais, cria-se uma dissociação entre a mão do jogador e a mão virtual, diminuindo a imersão no jogo.

#### 4.8.2.2 Alternativa com *streaming* do video

O diferencial desta opção é que a lógica do jogo roda no computador, e a saída de vídeo é transmitida para o celular via algum software, como o Riftcat(pago) ou o TrinusVR(gratuíto). Desta forma, o controlador é conectado diretamente ao computador, que está rodando o serviço do *Leap Motion* e lê diretamente dele os dados estruturados. Esta arquitetura pode ser vista na Figura 12.

Figura 11 – Alternativa de arquitetura com servidor WebSocket

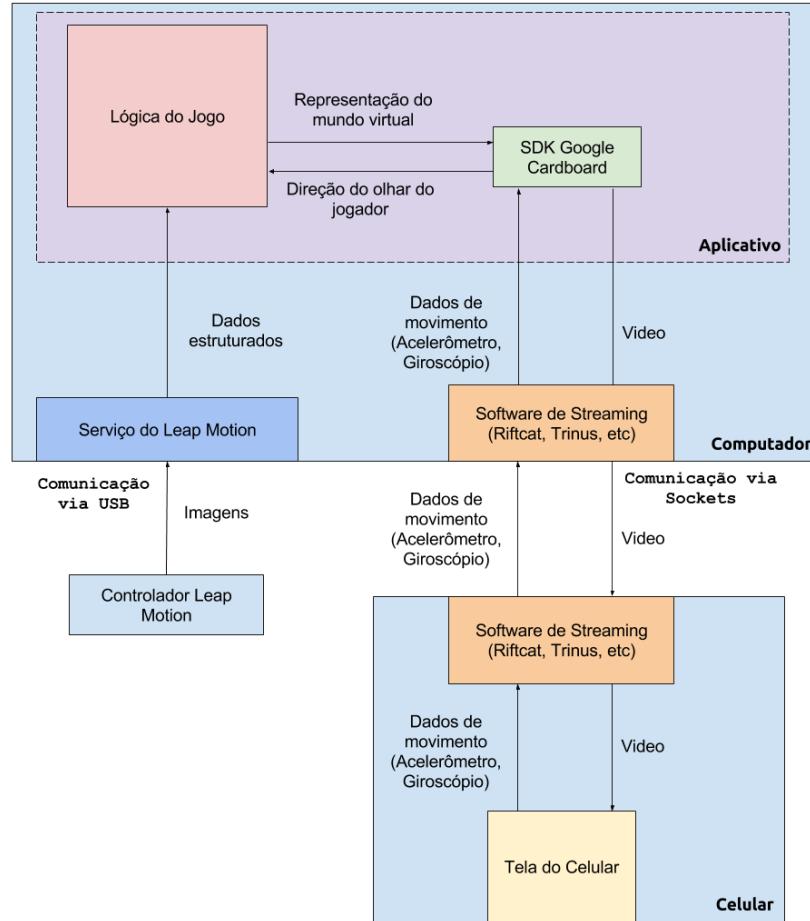


Fonte: Autor

A vantagem desta opção é que, por rodar num computador, o poder de processamento é mais alto e, portanto, existem mais possibilidades quanto a algoritmos utilizados, mecânicas do jogo e processamento gráfico. A conexão com o controlador, por ser direta, também significa uma latência baixa neste quesito.

O principal contraponto desta opção é o aumento da latência entre o movimento da cabeça do usuário e o movimento da cabeça virtual. Esta latência é maior do que a da primeira opção pois é necessário enviar os dados referentes ao movimento da cabeça do celular para o computador, que então deve enviar as imagens de volta para o celular. Essa latência adicional não apenas cria uma dissociação entre o jogador e o mundo virtual, mas também pode causar tontura e enjoos. Adicionalmente, mesmo a latência entre o movimento das mãos do jogador e sua detecção pelo código sendo baixa, a latência adicionada na saída de vídeo pode fazer com que pareça que as mãos estejam sendo detectadas com uma latência mais alta.

Figura 12 – Alternativa de arquitetura com streaming do video



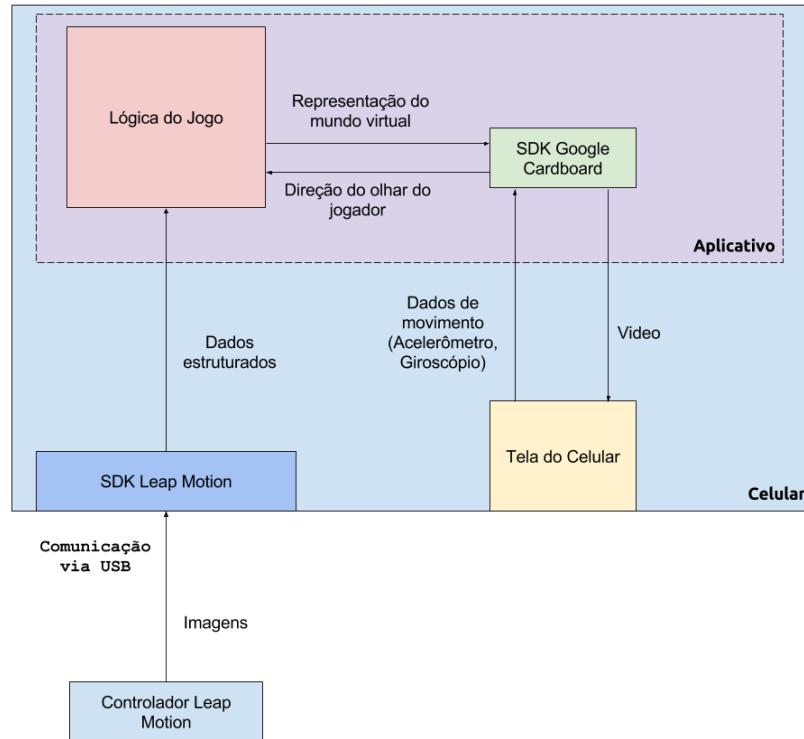
Fonte: Autor

#### 4.8.2.3 Alternativa com conexão direta

Nesta opção, o controlador é conectado diretamente ao celular, conforme mostrado na Figura 13. Desta forma, esta opção provê a maior mobilidade e tem o menor número de subsistemas diferentes. Para tal, é necessário utilizar o serviço do *Leap Motion v2* específica para o *Android*. Neste caso, todo o processamento é feito diretamente no celular (A transformação das imagens em dados estruturados, a lógica do jogo e as transformações necessárias para a saída de vídeo). Por esta razão, esta é a opção que adiciona a menor latência, devido à inexistência de conexões extras entre um computador e o celular, como nas outras opções.

As desvantagens, porém, são a dependência do serviço do *Leap* nativo do *Android*, que estava na versão beta antes de ser descontinuado, e também o fato de se tratar da versão v2, que tem um rastreamento pior.

Figura 13 – Alternativa de arquitetura com conexão direta



Fonte: Autor

#### 4.8.2.4 Comparação das alternativas

A comparação das alternativas pode ser vista no Quadro 3.

É possível ver que as alternativas que não têm a conexão direta introduzem latências no sistema, mas em locais diferentes. O servidor *WebSocket* introduz latência na entrada referente ao movimento das mãos, enquanto o *streaming* de vídeo adiciona atrasos na saída do vídeo para o *headset*. Por outro lado, a alternativa da conexão direta utiliza uma versão anterior do *SDK* do controlador, que tem um rastreamento pior.

#### 4.8.3 Escolha da arquitetura

Inicialmente, havia se optado pela alternativa da conexão direta. Porém, devido à impossibilidade de se utilizar a versão *Orion* do *SDK* do *Leap Motion*, foi necessário mudar de alternativa. A segunda alternativa escolhida foi a do *streaming* de vídeo, visto que a sua implementação não necessitaria de mudanças no código. Bastaria gerar o executável do jogo no Windows e rodar um software de *streaming*, como o Riftcat. Esta alternativa foi testada por um tempo, aonde se constatou alguns defeitos, como a necessidade de se reiniciar o software várias vezes devido a bugs e falta de documentação do Riftcat. Mesmo assim, devido a facilidade de implementação e qualidade aceitável da solução, a alternativa foi escolhida como a que seria utilizada.

Quadro 3 – Comparaçāo das alternativas de arquitetura

Alternativa	Pros	Contras
Servidor Websocket	Versão mais recente do <i>SDK</i> do <i>Leap</i> . Servidor <i>WebSocket</i> já disponível com o serviço <i>Leap Motion</i> .	Maior latência entre o movimento da mão e da mão do jogador, pode diminuir imersão. Necessidade de modificação da biblioteca do <i>Leap</i> no <i>Unity</i> .
<i>Streaming</i> de Video	Versão mais recente do <i>SDK</i> do <i>Leap</i> . Processamento feito no computador Conexão direta da lógica do jogo com controlador	Latência entre movimento da cabeça do jogador e movimento da cabeça virtual, podendo causar desconforto e náusea. Dependência de mais software, podendo ser pago, e que pode ter como requisito um hardware mais caro para funcionar.
Conexão Direta	Menor latência dentre as opções Maior mobilidade Não depende de um computador	Utiliza a versão v2 do <i>SDK</i> do <i>Leap</i> , que tem precisão pior. Necessita da versão beta do serviço para o <i>Android</i> , de difícil acesso.

Fonte: Autor

Porém, descobriu-se posteriormente, ao se testar o software nos computadores dos outros membros do grupo, que para que esta alternativa funcionasse, era necessário uma placa de vídeo relativamente potente, de acordo com as especificações mínimas do software, encontrada em (RIFTCAT, 2016). Inclusive, o software do Riftcat só funcionava em um dos cinco computadores testados. Por esta razão, optou-se por abandonar esta alternativa, visto que diminuiria a acessibilidade do jogo.

Dado este contratempo, decidiu-se verificar a viabilidade das outras duas alternativas. A alternativa da conexão direta havia sido descartada devido à remoção do *SDK* do site do *Leap Motion*. Afortunadamente, a co-orientadora deste projeto havia trabalhado anteriormente com o *Leap Motion* no *Android*, e conseguiu uma cópia dos arquivos necessários. Entretanto, nos testes preliminares feitos, não se obteve sucesso na utilização do serviço: o aplicativo do software no celular reconhecia o controlador conectado, mas aplicativos clientes que utilizariam os dados do controlador não conseguiam conectar-se ao controlador. Por este motivo, esta alternativa foi descartada.

Optou-se então pela alternativa que restou, que utiliza o servidor *WebSocket* gerado pelo serviço do *Leap Motion* no computador. A principal dificuldade desta alternativa é que o plugin disponibilizado pela empresa para o *Unity* só conseguia receber os dados do controlador por meio da conexão direta, e não pelo servidor *WebSocket*. Portanto, foi necessário modificar o plugin distribuído, que é software livre e tem seu código fonte disponível, para que ele se conectasse ao servidor *WebSocket*, recebesse os dados estrutu-

rados em mensagens JSON, processasse-as e então as transformasse na estrutura de dados utilizada pelo *Unity*.

Para implementar tais mudanças no código do *plugin*, foi necessário encontrar e testar bibliotecas que se conectassem a um servidor *WebSocket* (visto que o *Unity* não consegue fazer isto nativamente) e processasse JSON (pois o *parser* JSON nativo do *Unity* só funciona em casos de uso muito limitados). Visto que é necessário que movimentos das mãos sejam refletidos o mais rapidamente o possível no jogo, para se manter a imersão, é essencial que estas bibliotecas fossem rápidas o suficiente para não serem um gargalo. Como, no *Unity*, é o próprio *framework* que faz a gerência da sua memória (ou seja, o *Unity* roda o coletor de lixo de tempos em tempos, em oposição ao programador ter de se preocupar com isto) é necessário também que os algoritmos sejam eficientes em questão de memória e objetos alocados, visto que rodam com frequência alta e que a execução do coletor de lixo pode causar travamentos dependendo de quantos objetos 'mortos' existirem.

Uma dificuldade encontrada nesta parte de modificação do *plugin* é que o *Unity* tem um *profiler* interno, que permite ver a alocação de memória e dados sobre o desempenho do jogo, mas que só permite esse monitoramento no *thread* principal do *Unity*. Visto que a modificação do *plugin* inseria a necessidade de se ficar esperando uma mensagem chegar do servidor *WebSocket*, e que fazer isto na *thread* principal acarretaria em uma grande queda de desempenho, o código que recebe as mensagens do servidor e que as transforma nos dados esperados pelo *Unity* foi colocado para executar em uma *thread* secundária. Portanto, com o *profiler* interno do *Unity* não era possível monitorar diretamente o desempenho da *thread*, sendo necessário estimar o desempenho através das consequências dos resultados do *thread* secundário, ou através da impressão do valor de variáveis no terminal.

#### 4.8.4 Especificação do *hardware* e *software*

A especificação do software e hardware necessário é definido, predominantemente, pelas escolhas feitas de se utilizar o controlador *Leap Motion* e o *headset Google Cardboard*. De acordo com os requisitos mínimos do *Leap Motion*, encontrados em (LEAPMOTION, 2016f), e com as configurações utilizadas na geração do executável do aplicativo, seguem as especificações de *Hardware* e *Software*:

a) *Hardware*

- computador com AMD Phenom™ II ou Intel® Core™ i3/i5/i7 processor;
- computador com pelo menos 2 GB de memória *RAM*;
- duas portas USB 2.0;
- computador com adaptador de rede para a conexão com a internet;
- celular com giroscópio;

- celular com processador Qualcomm Snapdragon 800 em diante;
- celular com pelo menos 2 GB de memória *RAM*;
- o controlador *Leap Motion*.

b) *Software*

- computador com sistema operacional Microsoft Windows 7 ou superior, ou com Apple MacOS X 10.7 em diante;
- celular com sistema operacional Android 4.4 em diante.

## 4.9 Reconhecimento de Gestos

Um dos principais desafios no desenvolvimento do projeto foi a programação do reconhecimento de gestos. Conforme visto na Subseção 2.7.3, o controlador *Leap Motion* capta as mãos por meio de suas câmeras e transforma-as em um modelo baseado na anatomia humana. A partir deste modelo, utilizando a posição e orientação de cada osso do dedo e da palma da mão, é possível extrair gestos.

No projeto, foram definidos os gestos que podem ser vistos no Quadro 4

Quadro 4 – Gestos definidos

Nome	Gesto	Efeito
Selecionar uma área	Para cada mão, juntar o indicador ao polegar e então afastar ou aproximar as mãos	A seção do mundo que está visualmente entre a junção dos dedos das duas mãos é selecionado
Abaixar terra	Movimentar a mão direita de cima para baixo, com a palma apontando para baixo	Abaixa a terra em uma unidade para cada bloco de terra selecionado
Levantar terra	Movimentar a mão direita de baixo para cima, com a palma apontando para cima	Levanta a terra em uma unidade para cada bloco de terra selecionado
Chover	A partir da mão direita parada, com a palma para baixo, fechar a mão em um punho	Cria o elemento água sobre os blocos atualmente selecionados
Fogo	A partir da mão direita parada, com a palma para cima, fechar a mão em um punho	Cria o elemento fogo sobre os blocos atualmente selecionados.
Vento	A partir da mão direita parada, com a palma para a esquerda, fechar a mão em um punho	Cria vento sobre o mundo todo
Girar para esquerda	Fazer um movimento contínuo com a mão direita, começando com a palma apontando para a esquerda e terminando com a palma apontando para o corpo do usuário	Rotaciona ambos os mundos no sentido horário, em relação a um observador que esteja acima dos mundos olhando para baixo.
Girar para direita	Fazer um movimento contínuo com a mão esquerda, começando com a palma apontando para a direita e terminando com a palma apontando para o corpo do usuário	Rotaciona ambos os mundos no sentido anti-horário, em relação a um observador que esteja acima dos mundos olhando para baixo.
Mover câmera	Com ambas as mãos com suas palmas para baixo, fechá-las em punhos e então movimentar as mãos para a esquerda ou direita	No modo sem realidade virtual, serve para movimentar a câmera do jogo para a direita e esquerda, para navegar entre ambos os mundos

Fonte: Autor

Para se detectar os gestos, foi necessário definir alguma configurações das mãos, que podem ser vistos no Quadro 5. Para a definição destas configurações criou-se um

sistema ortonormal de coordenadas, definido em relação ao controlador Leap Motion. Neste sistema, o eixo x fica na direção da dimensão mais longa do controlador, o eixo y é aquele perpendicular ao plano que contém as câmeras do controlador e o eixo z é paralelo ao plano que contém as câmeras e aumenta no sentido que vai para longe do jogador. Para a mão esquerda, o eixo x cresce para a direita, enquanto para a mão direita o eixo cresce para a esquerda.

Adicionalmente, define-se que o vetor normal da mão é o vetor unitário perpendicular à palma da mão, e segue o mesmo sistema de coordenadas definido acima.

Quadro 5 – Configurações definidas

Nome	Descrição	Configuração
Punho	Mão em um punho	Caso o menor ângulo entre o vetor normal da mão e o vetor paralelo à falange proximal do dedo médio seja menor que $25^\circ$ e que o menor ângulo entre o vetor paralelo à falange proximal do dedo médio e o vetor paralelo à falange média do dedo médio seja maior que $60^\circ$
Pinça	Dedo indicador e polegar encostando, com a palma virada para dentro, em relação ao corpo do jogador	Caso a distância entre as pontas dos dedos indicador e polegar seja menor que 3cm, e o vetor normal da mão tenha sua componente em x maior que 0,6 e sua componente em y menor que 0,4
Mão aberta	Mão com todos os dedos esticados, aonde 'esticado' é definido pelo Leap Motion e disponibilizado no modelo	Caso a propriedade booleana 'isExtended' de cada um dos cinco dedos do modelo recebido pelo Leap Motion seja verdadeira
Mão esticada	Mão aberta, com os dedos esticados e juntos	Caso a mão esteja aberta e a distância entre a ponta do dedo mínimo e do dedo indicador seja menor que a soma dos comprimentos das falanges do dedo anelar
Mão virada para dentro	Mão direita com a palma para a esquerda ou mão esquerda com a palma para a direita	Caso a componente x do vetor normal da mão seja maior que 0,9
Mão virada para cima	Mão com a palma para cima	Caso a componente y do vetor normal da mão seja maior que 0,4
Mão virada para baixo	Mão com a palma para baixo	Caso a componente y do vetor normal da mão seja menor que -0,4

Fonte: Autor

Uma vez definido estas configurações, é possível definir os gestos mais facilmente. A definição do reconhecimento de cada gesto pode ser vista no Quadro 6.

Quadro 6 – Reconhecimento de gestos

Nome	Gesto
Selecionar uma área	Ambas as mãos na configuração de pinça. Basta calcular o ponto médio entre as pontas dos dedos indicador e polegar para se obter o ponto exato sendo selecionado
Abaixar terra	Mão direita esticada, com o componente y do vetor perpendicular à palma da mão negativa. A posição inicial quando a mão fica esticada é gravada. Caso a mão se move 5cm para baixo e continue esticada, a ação de 'abaixar terra' é disparada
Levantar terra	Mão direita esticada, com o componente y do vetor normal da mão positiva. A posição inicial quando a mão fica esticada é gravada. Caso a mão se move 5cm para cima e continue esticada, a ação de 'levantar terra' é disparada
Chover	Mão direita aberta, virada para baixo, seguido da mão direita em um punho, ainda virada para baixo
Fogo	Mão direita aberta, virada para cima, seguido da mão direita em um punho, ainda virada para cima
Vento	Mão direita aberta, virada para dentro, seguida de mão direita em um punho, ainda virada para dentro
Mover câmera	Ambas as mãos viradas para baixo. No momento em que ambas se fecharem em punhos, grava-se a posição de cada mão. Enquanto ambas as mãos continuam em punhos, calcula-se quanto cada mão se moveu na direção x e soma-se esta mudanças. Caso o valor seja positivo, a câmera se move para a direita. Caso contrário, para a esquerda. No momento no qual pelo menos uma das mãos deixa de ser um punho, o movimento da câmera para
Girar para esquerda	Mão direita aberta, virada para dentro, seguida de mão direita aberta, com a componente z do vetor normal da mão menor que -0,5
Girar para direita	Mão esquerda aberta, virada para dentro, seguida de mão direita aberta, com a componente z do vetor normal da mão menor que -0,5

Fonte: Autor

# 5 Testes

Para a avaliação da eficácia do produto final deste trabalho, foi desenvolvido um questionário, presente no Apêndice A, assim como um roteiro de testes para a aplicação de tal questionário. Após a concepção do roteiro de testes, houve uma sessão de teste do jogo (*playtesting*) contemplando principalmente representantes do público alvo escolhido - crianças entre 8 e 12 anos de idade.

As experiências dos jogadores conforme observadas e relatadas foram então compiladas em um relatório descrevendo os aspectos positivos e negativos do jogo, bem como dificuldades técnicas encontradas.

## 5.1 Roteiro de testes

A concepção do questionário se deu através de duas iterações.

Na primeira iteração, o foco foi no conteúdo das questões. A preocupação principal foi o quanto as perguntas ajudariam a responder se o jogo era divertido, imersivo e estimulava as habilidades do raciocínio lógico. Foram desenvolvidas 13 questões com uma escala de sete pontos (onde um é “Discordo Plenamente” e sete é “Concordo Plenamente”) e três dissertativas curtas. As questões foram divididas em três seções: experiência, interface e desafios. O primeiro tenta medir a diversão percebida pelo jogador, o segundo a imersão e facilidade de interação com o jogo e o terceiro o estímulo ao raciocínio lógico.

A segunda iteração ocorreu após a verificação do questionário com crianças da faixa etária do público alvo, que foi feita para confirmar que o vocabulário utilizado estava adequado e de fácil compreensão para o jogador. Visto que foram constatadas dificuldades no entendimento em aproximadamente metade das questões, o questionário foi reescrito com uma linguagem mais simples e novamente verificada com crianças.

Após ter sido verificado com crianças da idade do público alvo, foi feita uma validação do questionário com três especialistas que são das áreas de educação, jogos, interface com o usuário e interação humano-computador.

Visto que estamos trabalhando com menores de 18 anos, é necessário que o guardião legal da criança conceda permissão para que ela participe do teste.

Para os testes, o seguinte roteiro foi criado:

- a) fornecimento das informações sobre o motivo da pesquisa;
- b) fornecimento de instruções sobre a tarefa a ser realizada;

- c) fornecimento de informações sobre os riscos ao sujeito de pesquisa;
- d) coleta da permissão, concedida pelo guardião legal da criança, para que esta participe do teste.
- e) conectar o *smartphone* do pesquisador que está aplicando o teste ao computador do pesquisador e executar o software do jogo;
- f) encerrar a sessão de teste após terminar as quatro fases selecionadas, ou após 15 minutos, o que vier antes;
- g) preencher o questionário desta pesquisa.

Esta pesquisa foi aprovada pelo Comitê de Ética em Pesquisa do Hospital Universitário da USP (CEP-HU/USP) com parecer consubstanciado número 1.114.876 datado de 19/06/2015.

## 5.2 Playtesting

A seção de *playtesting* foi realizada durante o Tech Kids Day realizado no museu Catavento Cultural no dia 12 de novembro como parte da São Paulo Tech Week 2016.

Uma versão reduzida e simplificada do jogo foi elaborada, contendo uma parcela suficientemente representativa do conteúdo do jogo, enquanto mantendo o tempo de jogo dentro de um período de 15 minutos ou menos, compatível com as condições de teste.

Essa versão consiste em um conjunto de quatro fases, disposta do seguinte conteúdo:

- a) **fase 1:** introdução ao funcionamento básico do jogo. Apresentação das mecânicas de seleção em área e manipulação da elevação do terreno;
- b) **fase 2:** introdução de mecânicas de criação e manipulação de corpos d'água;
- c) **fase 3:** introdução da mecânica de vento e da criação de blocos de areia;
- d) **fase 4:** combinação de todos os elementos apresentados em um cenário mais complexo.

Duas cópias do jogo foram iniciadas lado a lado, uma ligada tanto ao *Leap Motion* para controle gestual quanto ao *Cardboard* para o efeito de realidade virtual, como pode ser visto na Figura 14, enquanto outra ligada apenas ao *Leap Motion*, com o *output* direto pela tela do computador, mostrado na Figura 15.

Devido à dinâmica do evento, aonde cada bancada com algo exposto era visitada por várias pessoas e a rotatividade delas era alta, não foi possível aplicar o questionário. Mesmo assim, durante a execução do *playtesting* foi notada a reação dos jogadores e seu *feedback* foi coletado após cada sessão do jogo, conforme descrito na Seção 5.3.

Figura 14 – Crianças testando o jogo durante a no Tech Kids Day



Fonte: Autor

Figura 15 – Jogo sendo executado apenas com o *Leap Motion*



Fonte: Autor

### 5.3 Observações

Os resultados dos testes, obtidos através da observação e do *feedback* dos jogadores, trouxeram à tona problemas inesperados, bem como pontos positivos na implementação escolhida.

Em termos gerais, o jogo foi bem recebido pelas crianças, que se mostraram entretidas e imersas durante toda a duração da sessão. Ficou claro também que a versão executando em realidade virtual foi mais popular, despertando bastante interesse nos jogadores.

Contudo, algumas interações mecânicas se revelaram mais difíceis de se executar do que era esperado, exigindo uma familiaridade com os sensores que fugia ao escopo do tempo de teste. Sobretudo as mecânicas de gerar e remover água se comportaram de maneira menos responsiva do que necessário para que o jogo pudesse ser jogado com naturalidade.

Em termos técnicos, os sensores utilizados no projeto apresentaram dificuldades em detectar os gestos e movimentos executados por crianças mais novas, devido tanto à

incapacidade do *Leap Motion* em reconhecer mãos pequenas quanto à impossibilidade das crianças em manter confortavelmente as mão à altura necessária para a captura correta de seus movimentos.

Adicionalmente, o ambiente não colaborou com a seção de testes. A mesa aonde estavam dispostos os jogos se encontrava muito iluminada por fontes de luz, por estar do lado de uma janela. Visto que o controlador funciona com a captura de imagens infravermelhas, o excesso de luz diminuiu a qualidade de captura do controlador, interferindo no jogo.

Apesar dos desafios encontrados, a experiência geral se mostrou positiva, e o *feedback* obtido será muito valioso para desenvolver iterações do projeto.

# 6 Considerações finais

O objetivo deste trabalho foi criar um jogo efetivamente educacional, capaz de manter o engajamento e interesse do jogador, de forma a ajudar a exercitar habilidades de raciocínio lógico, além de desenvolver uma solução final de baixo custo que possa ser implementada em instituições educacionais.

## 6.1 Principais resultados

Foi desenvolvido um jogo digital, utilizando realidade virtual e o controle através das mãos, que entreteve os jogadores que o testaram, mesmo havendo algumas falhas no controlador.

A solução também tem um custo acessível, se comparado a outras soluções de realidade virtual, devido ao uso de componentes de baixo custo e a possível reutilização de celulares *smartphones* e *laptops*. O jogo não necessita de um computador com componentes caros e de alto desempenho (como uma placa de video dedicada topo de linha), e é simples de configurar e executar. Adicionalmente, a latência presente no jogo em relação ao movimento do *headset* foi considerada aceitável, visto que nenhum dos participantes que testaram o projeto reclamaram sobre tontura ou enjoos.

Não foi possível aplicar o questionário criado, e portanto não foi possível obter dados qualitativos sobre o efeito do jogo sobre as competências e habilidades do raciocínio lógico. Porém, durante a sessão de *playtesting* se constatou que os jogadores encaravam o jogo como um desafio lógico, comprovando em parte o exercício do raciocínio lógico.

## 6.2 Contribuições

As contribuições desse trabalho na nossa formação foram:

- a) praticar o gerenciamento de projeto e o trabalho em grupo;
- b) criar uma visão do processo envolvido no desenvolvimento de um jogo do início ao fim;
- c) aprender novas tecnologias, como o controlador *Leap Motion*, *WebSockets* e o *Google Cardboard*;
- d) aplicar teorias aprendidas durante o curso.

Fatores que colaboraram para o projeto foram:

- a) nossa co-orientadora, Profa. Msc. Lucy Mari Tabuti, ter conhecimento e experiência na área de jogos de lógica digitais, jogos educacionais, realidade virtual, e com o controlador *Leap Motion*;
- b) os integrantes do grupo já terem um pouco de experiência com o *Unity*.

### 6.3 Dificuldades

Em relação ao controlador *Leap Motion*, houve dificuldade em detectar movimentos específicos, tanto por inexperiência (conforme dito na Seção 4.4) quanto por dificuldades em expressar os movimentos da forma correta, ao se tratar das relações entre os ossos e vetores que definiam a mão do jogador, conforme descrito na Subseção 2.7.3.

Adicionalmente, houve dificuldade de se aplicar os testes em crianças pois suas mãos, possivelmente pelo seu tamanho menor, não eram detectadas muito bem pelo controlador. Conforme descrito na Seção 5.3, porém, é possível que este problema tenha sido devido à iluminação do local, que atrapalha o controlador.

A definição da arquitetura também encontrou obstáculos, visto que a primeira escolha de arquitetura teve de ser modificada devido a problemas relacionados ao hardware. Também ocorreu problemas devido à falta de documentação de elementos, como da versão *Android* da biblioteca do *Leap Motion* e do software *Riftcat*.

### 6.4 Trabalhos futuros

Dado que não foi possível aplicar o questionário com pessoas pertencentes ao público alvo, um próximo passo seria este. Desta forma, seria possível verificar o quão benéfico o jogo é.

Adicionalmente, novas fases com novas mecânicas podem ser implementadas no futuro, aumentando o tamanho do jogo, de forma a permitir que jogadores passem mais tempo se divertindo, enquanto exercitam suas habilidades de raciocínio lógico.

Melhorias no código, em especial no reconhecimento de gestos utilizados para ações também são um próximo passo importante, por tornar o jogo menos frustrante ao usuário, que não mais precisará repetidamente executar um mesmo gesto para tentar ativar uma ação.

## Referências

- ALVES, G. F.; SOUZA, E. V.; SOUSA, P. M. de. *Realidade Virtual e Aumentada Aplicada na Educacao na Disciplina de Quimica – RVAQ*. 2015. SBGames 2015 - Art & Design Track – Full Papers.
- CNET. *Face has Oculus, Google has Cardboard*. 2016. Disponível em: <<https://www.cnet.com/news/facebook-has-oculus-google-has-cardboard/>>. Acesso em: 2016-09-09.
- CORREIA, A. C. et al. Jogos digitais : possibilidades e limitações : o caso do jogo spore. In: . [S.l.: s.n.], 2009.
- CTFRC. *School Aged Developmental Milestones*. 2016. Disponível em: <<http://www.kamloopschildrenstherapy.org/fine-motor-skills-school-milestones>>. Acesso em: 2016-10-16.
- DODGSON, N. A. Variation and extrema of human interpupillary distance. In: INTERNATIONAL SOCIETY FOR OPTICS AND PHOTONICS. *Electronic imaging 2004*. [S.l.], 2004. p. 36–46.
- FERNANDES, J. C. L. Educação digital: Utilização dos jogos de computador como ferramenta de auxilio à aprendizagem. *FaSCi-Tech*, v. 1, n. 3, 2012.
- FETTE, I.; MELNIKOV, A. Rfc 6455: The websocket protocol. *IETF, December*, 2011.
- FLEURY, A.; NAKANO, D.; CORDEIRO, J. Mapeamento da indústria brasileira e global de jogos digitais. *São Paulo: GEDIGames/USP, BNDES*, 2014.
- FRADE, B. v.; ALIXANDRE, B. F.; SOUZA, P. M. Desenvolvimento de um jogo sério com uso de realidade virtual aplicado ao ensino da matematica. In: . [S.l.: s.n.], 2015.
- GINSBURG, H.; OPPER, S. *Piaget's Theory of Cognitive Development*. Prentice Hall, 1969. ISBN 8535241981. Disponível em: <<https://www.amazon.com/Arte-Game-Design-Livro-Original/dp/8535241981%3FSubscriptionId%3D0JYN1NVW651KCA56C102%26tag%3Dtechkie-20%26linkCode%3Dxm2%26camp%3D2025%26creative%3D165953%26creativeASIN%3D8535241981>>.
- GOOGLE. *Dashboards*. 2016. Disponível em: <<https://developer.android.com/about/dashboards/index.html>>. Acesso em: 2016-11-25.
- GOOGLE. *Google Cardboard SDK*. 2016. Disponível em: <<https://developers.google.com/vr/cardboard/overview>>. Acesso em: 2016-09-05.
- GOOGLE. *Google Cardboard Store*. 2016. Disponível em: <<https://vr.google.com/cardboard/get-cardboard/>>. Acesso em: 2016-12-18.
- HILL, S. *Is VR too dangerous for kids?* 2016. Disponível em: <<http://www.digitaltrends.com/virtual-reality/is-vr-safe-for-kids-we-asked-the-experts/>>. Acesso em: 2016-10-16.

- HOFFMAN, D. M. et al. Vergence-accommodation conflicts hinder visual performance and cause visual fatigue. *Journal of vision*, The Association for Research in Vision and Ophthalmology, v. 8, n. 3, p. 33–33, 2008.
- HTC. *HTC Vive Store*. 2016. Disponível em: <<http://www.vive.com/us/>>. Acesso em: 2016-12-18.
- HUNICKE MARC LEBLANC, R. Z. R. Mda: A formal approach to game design and game research. 2004.
- IBGE. *Acesso à Internet e à Televisão e Posse de Telefone Móvel Celular para Uso Pessoal*. 2015.
- KIRNER, C.; KIRNER, T. G. Evolução e tendências da realidade virtual e da realidade aumentada. *Livro do XIII Pré-Simpósio de Realidade Virtual e Aumentada, Uberlândia*, p. 10–25, 2011.
- KIRNER, C.; SISCOUTTO, R. Realidade virtual e aumentada: conceitos, projeto e aplicações. In: *Livro do IX Symposium on Virtual and Augmented Reality, Petrópolis (RJ), Porto Alegre: SBC*. [S.l.: s.n.], 2007.
- LEAPMOTION. *Gestures*. 2016. Disponível em: <[https://developer.leapmotion.com/documentation/csharp/devguide/Leap\\_Gestures.html](https://developer.leapmotion.com/documentation/csharp/devguide/Leap_Gestures.html)>. Acesso em: 2016-09-23.
- LEAPMOTION. *How Does the Leap Motion Controller Work?* 2016. Disponível em: <<http://blog.leapmotion.com/hardware-to-software-how-does-the-leap-motion-controller-work/>>. Acesso em: 2016-09-09.
- LEAPMOTION. *Introducing the Skeletal Tracking Model*. 2016. Disponível em: <[https://developer.leapmotion.com/documentation/javascript/devguide/Intro\\_Skeleton\\_API.html](https://developer.leapmotion.com/documentation/javascript/devguide/Intro_Skeleton_API.html)>. Acesso em: 2016-09-23.
- LEAPMOTION. *Leap Motion Architecture*. 2016. Disponível em: <[https://developer.leapmotion.com/documentation/cpp/devguide/Leap\\_Architecture.html](https://developer.leapmotion.com/documentation/cpp/devguide/Leap_Architecture.html)>. Acesso em: 2016-09-05.
- LEAPMOTION. *Leap Motion Release Notes & Changeset*. 2016. Disponível em: <[https://developer.leapmotion.com/documentation/Leap\\_SDK\\_Release\\_Notes.html#version-3-0-0](https://developer.leapmotion.com/documentation/Leap_SDK_Release_Notes.html#version-3-0-0)>. Acesso em: 2016-09-05.
- LEAPMOTION. *LeapMotion Requirements*. 2016. Disponível em: <<https://support.leapmotion.com/hc/en-us/articles/223783668-What-are-the-system-requirements->>. Acesso em: 2016-11-15.
- LUSSENHOP, J. *Oregon Trail: How three Minnesotans forged its path*. 2011. Versão arquivada pelo WayBack Machine. Disponível em: <<https://web.archive.org/web/20110123012937/http://www.citypages.com/content/printVersion/1740595/>>. Acesso em: 2016-10-07.
- MOTION, L. *Leap Motion Store*. 2016. Disponível em: <<http://store-world.leapmotion.com/>>. Acesso em: 2016-12-18.

- MYO. *Myo Store*. 2016. Disponível em: <<https://store.myo.com/>>. Acesso em: 2016-12-18.
- NEARPOD. *Nearpod - Easily integrate technology into every lesson*. 2016. Disponível em: <<https://nearpod.com/how-it-works>>. Acesso em: 2016-10-16.
- OCULUS. *Oculus Rift Store*. 2016. Disponível em: <<https://www3.oculus.com/en-us/rift/>>. Acesso em: 2016-12-18.
- OECD. Low-performing students: Why they fall behind and how to help them succeed. OECD Publishing. Disponível em: <[/content/book/9789264250246-en](https://content/book/9789264250246-en)>.
- OECD. Pisa 2012 results - brazil overview. Disponível em: <<http://www.oecd.org/pisa/keyfindings/PISA-2012-results-brazil.pdf>>.
- POLI, D. et al. Bringing evolution to a technological generation: a case study with the video game spore. *The american biology Teacher*, University of California Press Journals, v. 74, n. 2, p. 100–103, 2012.
- RAMOS, D. K. Jogos cognitivos eletrônicos: contribuições à aprendizagem no contexto escolar. *Ciências & Cognição*, Instituto de Ciências Cognitivas, v. 18, n. 1, p. 19–32, 2013.
- RIFTCAT. *Riftcat Requirements*. 2016. Disponível em: <<https://riftcat.com/vridge/requirements>>. Acesso em: 2016-09-07.
- SCHELL, J. *A Arte De Game Design: O Livro Original*. CRC Press, 2010. ISBN 8535241981. Disponível em: <<https://www.amazon.com/Arte-Game-Design-Livro-Original/dp/8535241981%3FSubscriptionId%3D0JYN1NVW651KCA56C102%26tag%3Dtechkie-20%26linkCode%3Dxm2%26camp%3D2025%26creative%3D165953%26creativeASIN%3D8535241981>>.
- SEETO, D. *PlayStation VR Is Not Recommended For People Under 12 Years Old*. 2016. Disponível em: <<http://attackofthefanboy.com/news/playstation-vr-not-recommended-people-12-years-old/>>. Acesso em: 2016-10-16.
- TABUTI, L. M.; NAKAMURA, R. Tabela de habilidades placeholder. In: *Anais do Simpósio Brasileiro de Informática na Educação*. [S.l.: s.n.], 2015. v. 26, n. 1, p. 41.
- TABUTI, L. M.; ROCHA, R. L. d. A. da; NAKAMURA, R. Análise da interação humano-computador de um jogo de lógica com diferentes dispositivos de entrada. 2010.
- YOUTUBE. *The Legend Of Zelda: Skyward Sword - Trouble With The Squid Carving Door Puzzle (Live Commentary)*. 2016. Disponível em: <[https://www.youtube.com/watch?v=7bT2Bn\\_yRB0](https://www.youtube.com/watch?v=7bT2Bn_yRB0)>. Acesso em: 2016-09-15.
- ZELDAINFORMER. *A LOOK BACK AT SKYWARD SWORD: HYPE TRAIN OR MODERN CLASSIC?* 2016. Disponível em: <[http://www.zeldainformer.com/news/a\\_look\\_back\\_at\\_skyward\\_sword\\_hype\\_train\\_or\\_modern\\_classic](http://www.zeldainformer.com/news/a_look_back_at_skyward_sword_hype_train_or_modern_classic)>. Acesso em: 2016-09-15.

# Apêndices

# APÊNDICE A – Experimento - Cubo de Gaia

**Perfil do sujeito de pesquisa**

Idade: \_\_\_\_\_ anos

Sexo: ( ) Masculino ( ) Feminino

**Experiência**

*Selecione a alternativa mais adequada sobre a experiência geral do jogo*

1 – A experiência foi divertida

Discordo plenamente ( ) 1 ( ) 2 ( ) 3 ( ) 4 ( ) 5 ( ) 6 ( ) 7 Concordo plenamente

2 – A experiência foi desafiadora

Discordo plenamente ( ) 1 ( ) 2 ( ) 3 ( ) 4 ( ) 5 ( ) 6 ( ) 7 Concordo plenamente

3 – A experiência foi educativa

Discordo plenamente ( ) 1 ( ) 2 ( ) 3 ( ) 4 ( ) 5 ( ) 6 ( ) 7 Concordo plenamente

4 – A experiência foi injusta em um ou mais momentos

Discordo plenamente ( ) 1 ( ) 2 ( ) 3 ( ) 4 ( ) 5 ( ) 6 ( ) 7 Concordo plenamente

se sim, em que situação ou situações?

---

**Interface**

*Selecione a alternativa mais adequada sobre os controles e estética do jogo*

5 – Os controles do jogo foram fáceis de compreender

Discordo plenamente ( ) 1 ( ) 2 ( ) 3 ( ) 4 ( ) 5 ( ) 6 ( ) 7 Concordo plenamente

6 – Os controles do jogo funcionaram corretamente

Discordo plenamente ( ) 1 ( ) 2 ( ) 3 ( ) 4 ( ) 5 ( ) 6 ( ) 7 Concordo plenamente

7 – O jogo funcionou bem com sensores de movimento

Discordo plenamente ( ) 1 ( ) 2 ( ) 3 ( ) 4 ( ) 5 ( ) 6 ( ) 7 Concordo plenamente

8 – Os gráficos e sons foram agradáveis

Discordo plenamente ( ) 1 ( ) 2 ( ) 3 ( ) 4 ( ) 5 ( ) 6 ( ) 7 Concordo plenamente

9 – Houve confusão para diferenciar os elementos do jogo

Discordo plenamente ( ) 1 ( ) 2 ( ) 3 ( ) 4 ( ) 5 ( ) 6 ( ) 7 Concordo plenamente

**Desafios**

*Selecione a alternativa mais adequada sobre a resolução de fases individuais*

10 – O jogo funcionou bem com sensores de movimento

Discordo plenamente ( ) 1 ( ) 2 ( ) 3 ( ) 4 ( ) 5 ( ) 6 ( ) 7 Concordo plenamente

11 – A dificuldade entre as fases aumentou de forma justa

Discordo plenamente ( ) 1 ( ) 2 ( ) 3 ( ) 4 ( ) 5 ( ) 6 ( ) 7 Concordo plenamente

12 – Conhecimentos obtidos em fases anteriores se mostraram úteis na resolução das fases seguintes

Discordo plenamente ( ) 1 ( ) 2 ( ) 3 ( ) 4 ( ) 5 ( ) 6 ( ) 7 Concordo plenamente

13 – O objetivo apresentado nas fases ficou claro

Discordo plenamente ( ) 1 ( ) 2 ( ) 3 ( ) 4 ( ) 5 ( ) 6 ( ) 7 Concordo plenamente

se não, em que situação?

---

### **Comentários**

Comentários, críticas ou sugestões?

---

---

---

# APÊNDICE B – Game Design Document

Game Design Document

# Cubo de Gaia

Revisão: 25/11/2016

Gianfranco Pennacchi, Vinícius Menézio

# Visão Geral

**Cubo de Gaia** é um jogo educacional em realidade virtual, voltado para crianças em idade escolar - entre 8 e 12 anos - que tem como propósito transmitir conceitos e habilidades básicas de raciocínio lógico.

O jogo foi desenvolvido como parte do Trabalho de Conclusão de curso dos alunos Vinícius Menézio e Gianfranco Pennacchi, do curso de Engenharia de Computação da Escola Politécnica da USP.

## Temática

**Cubo de Gaia** é um *puzzle* com foco em mecânicas de criação e manipulação dos elementos do mundo do jogo. O jogo permite ao jogador explorar a interação entre diferentes elementos naturais, construindo e destruindo livremente. Colocando nas mãos do jogador o poder de controlar diferentes elementos da natureza, como a elevação do terreno, níveis de precipitação e correntes de ar, **Cubo de Gaia** permite controle total sobre o ambiente de jogo.

Em cada fase, o jogador tem acesso a um novo mundo em miniatura, composto de pequenos blocos de diversos materiais, que respondem diferentemente às suas ações. Para prosseguir para a fase seguinte, o jogador deve manipular o mundo, até que este se encontre em um estado especificado.

A cada fase, o jogo introduz novos elementos, novas mecânicas e novas interações que devem ser compreendidas e internalizadas pelo jogador, e usadas em combinação com conhecimentos previamente aprendidos na resolução de *puzzles* cada vez mais complexos.

## Mecânicas Centrais

**Cubo de Gaia** faz uso do dispositivo Leap Motion como controlador principal, traduzindo gestos e movimentos manuais do jogador em interações com o mundo do jogo.

As principais mecânicas do jogo, utilizadas na manipulação do mundo, são:

Mecânica	Descrição	Gesto
Selecionar Área	Permite ao jogador selecionar diversos blocos do jogos para interagir com eles simultaneamente	"Pinçar" com ambas as mãos e movê-las sobre uma área
Alterar Elevação	Permite ao jogador elevar ou rebaixar o terreno	Mover a mão para cima ou para baixo, com a palma estendida para mesmo sentido
Criar Chuva	Permite ao jogador fazer chover sobre uma área, inundando-a	Juntar as pontas dos cinco dedos, apontando-os para baixo
Criar Fogo	Permite ao jogador atear fogo sobre uma área, removendo a água	Juntar as pontas dos cinco dedos, apontando-os para cima
Criar Vento	Permite ao jogador fazer ventar sobre o mundo	Juntar as pontas dos cinco dedos, apontando-os para frente
Rotacionar Mundo	Permite ao jogador observar o mundo do jogo de outros pontos de vista, rotacionando-o	"Abanar" uma das mãos para dentro, dependendo do sentido desejado

## Plataforma e Tecnologia

**Cubo de Gaia** é desenvolvido em Unity, e deve ser executado a partir de um celular Android compatível com Google Cardboard.

Adicionalmente, um computador Windows, conectado a um Leap Motion, deve ser utilizado para intermediar a transmissão do input do usuário para o jogo.

Uma versão alternativa pode ser executada diretamente do computador, sem o uso do Google Cardboard. O uso do Leap Motion é facultativo para essa versão, porém recomendado.

## Influências

Diversos jogos inspiraram a temática, estética e mecânicas empregadas em **Cubo de Gaia**.

Dentre eles destacam-se:

**Dust Game:** jogo *sandbox* de browser, no qual jogadores podem criar diversos elementos que interagem entre si. <http://dan-ball.jp/en/javagame/dust/>

**Grow Cube:** *puzzle* no qual o jogador deve adicionar elementos ao mundo sequencialmente, observando suas interações para encontrar a ordem correta. <http://www.eyezmaze.com/grow/cube/>

**Minecraft:** Jogo de construção e sobrevivência com estética em voxels, que permite ao jogador construir livremente qualquer tipo de estrutura utilizando os blocos básicos que formam o mundo. <https://minecraft.net/en/>

## Gameplay

### Cena de Menu

Ao iniciar **Cubo de Gaia**, o jogador se deparará com uma cena inicial contendo um menu de seleção de fases, dentre as quais apenas a primeira estará inicialmente acessível, com as demais sendo desbloqueadas, uma após a outra, conforme as anteriores são completadas.

Nesta cena, o jogador controlará um cursor, que se manterá fixo no centro de seu campo de visão e poderá ser posicionado sobre outros elementos da tela para interagir com eles conforme o jogador move sua cabeça. Este cursor pode ser utilizado para ligar ou desligar os dispositivos de realidade virtual, e para selecionar uma das fases do menu.

Após a fase ser selecionada, ela começará a ser carregada. Assim que o carregamento for concluído, o jogador será levado para a cena do puzzle correspondente.

## Cena do Puzzle

Uma vez iniciada a fase, uma pequena animação demonstrará o funcionamento de novas mecânicas, caso haja alguma. Essa animação se repetirá até que o jogador selecione o botão dizendo "entendi".

Em seguida, o jogador se encontrará em um extenso espaço, contendo dois "mundos" distintos, à sua direita e esquerda. O mundo à sua esquerda será seu *sandbox*, e poderá ser alterado livremente, através das mecânicas disponíveis para esta fase. O mundo à sua direita representará seu objetivo, indicando o estado final ao qual o *sandbox* deverá ser levado para que a fase seja considerada completa. Este mundo pode ser apenas observado, mas não alterado.

Uma vez que ambos os mundos estejam no mesmo estado, uma mensagem indicará a vitória do jogador, que será levado de volta ao menu inicial, podendo agora escolher a fase seguinte.

## Progressão do Jogo

São ao todo 10 fases, que serão desbloqueadas sequencialmente, ficando incrementalmente mais complexas.

Cada fase contém um *puzzle* distinto que serve tanto para testar e aprofundar as habilidades de raciocínio lógico do jogador quanto para habituá-lo ao próprio jogo e prepará-lo para as fases seguintes.

Os 10 puzzles existentes são os seguintes:

1. O jogador deve controlar a elevação do terreno para construir uma torre e uma vala;
2. O jogador deve controlar elevação e precipitação para criar um lago;
3. O jogador deve controlar elevação e precipitação para criar um oceano com um planalto submerso;
4. O jogador deve controlar elevação e precipitação para criar um rio contendo uma ilha no meio;

5. O jogador deve controlar elevação e precipitação para criar uma ilha sobre um planalto submerso;
6. O jogador deve controlar correntes de ar para erodir pedras e criar uma praia;
7. O jogador deve controlar elevação e precipitação para criar um mar cercado por pedras, e depois gerar uma corrente de ar para criar uma praia;
8. O jogador deve controlar elevação, precipitação, correntes de ar e rotação da perspectiva para criar uma ilha no canto do mundo, coberta por areia;
9. O jogador deve controlar elevação, precipitação, correntes de ar e rotação da perspectiva para criar um mundo contendo areia no centro, cercada por quatro pequenos lagos;
10. O jogador deve controlar elevação, precipitação, correntes de ar e rotação da perspectiva para criar um mundo contendo um lago no centro, e um mosaico de blocos intercalados de terra e areia ao redor.