AMBIENTE DE AULA EM REALIDADE VIRTUAL

Gabriel Garcia Salvador, Dalton Solano dos Reis – Orientador

Curso de Bacharel em Ciência da Computação
Departamento de Sistemas e Computação
Universidade Regional de Blumenau (FURB) – Blumenau, SC – Brasil

gabrielgarciasalvador@outlook.com, dalton@furb.br

Resumo: *Este artigo apresenta uma aplicação de Realidade Virtual (RV) como uma alternativa ou extensão ao ensino online, onde um docente utilizando de óculos de RV pode lecionar a sua aula com algumas das ferramentas necessárias dentro de um ambiente virtual análogo a uma sala de aula. A aplicação foi desenvolvida utilizando do motor gráfico Unity, e então um teste foi realizado com professores e usuários de outras áreas a fim de validar o uso da aplicação. Dentro da aplicação existem exercícios que devem ser completados a fim de instruir o usuário em como utilizar a aplicação. É possível manusear objetos virtuais com uma simulação de física e gravidade, assim como escrever, apagar e carimbar formas pré-desenhadas em um quadro negro. Os resultados foram favoráveis enfatizando a imersão e potencial da tecnologia para a área da educação, mas existe ainda uma curva de aprendizado para utilizar os dispositivos de RV. Concluindo que a tecnologia de RV pode acabar mudando a forma que as aulas onlines e presenciais serão lecionadas.*

Palavras-chave*: Ciência da computação. Realidade Virtual. Computação Gráfica. Educação. Ensino Online.*

1 INTRODUÇÃO

Devido a ameaça posta pela pandemia do vírus COVID-19 em 2020, muitas escolas e faculdades foram forçadas a cessar as atividades letivas presenciais a fim de manter os diretores, colaboradores e estudantes seguros durante essa crise de saúde pública. Levando o seu corpo docente a optar por lecionar de maneira *online* como forma substituta ao ensino presencial a fim de evitar a contaminação e o avanço da doença (HODGES *et al*., 2020).

Para manter as atividades letivas, as instituições educacionais rapidamente optaram por manter a modalidade de ensino a distância o que resultou em um avanço repentino sem precedentes na educação *online*. Com isso, diversas empresas de plataformas *online* de ensino propuseram apoiar e solucionar essa nova alta demanda. Muitas vezes com serviços gratuitos, mas essa rápida adoção mostrou diversas lacunas e deficiências nessas ferramentas (TERÄS *et al.*, 2020).

Além dos déficits das ferramentas, também existe a falta de capacitação por parte dos docentes na utilização de certas tecnologias. É necessário tempo e custo para que os encarregados pudessem treinar os docentes a utilizar as ferramentas e computadores. Outro ponto preocupante, em relação ao ensino *online*, seria a forma que o conteúdo seria disponibilizado e lecionado. Principalmente para conseguir manter o foco e engajamento dos alunos durante as aulas mesmo que a distância. Por fim, a instituição ainda deveria conseguir garantir um ensino acadêmico que esteja à altura do ensino presencial (BASTRIKIN, 2020).

O uso de um computador para lecionar *online*, utilizando de interfaces comuns como o teclado e mouse, se mostram muito limitantes como ferramentas de ensino. Pois as dificuldades do uso dessas ferramentas podem variar muito de acordo com a disciplina. Um exemplo comum onde essas interfaces geram dificuldades é nas situações em que os usuários necessitam criar representações visuais de forma manual através de desenhos como fórmulas e equações, utilizando de um mouse para realizar a tarefa, sem ter o controle e tato que teria ao escrever em um quadro físico (LEE, 2020).

Outro ponto é que no ensino, muitas vezes uma representação bidimensional como as encontradas nos livros didáticos ou em fotografias na internet, são insuficientes na compreensão de certos conteúdos, os quais requerem mais que uma imagem estática em uma página para serem compreendidos. Uma alternativa seria utilizar de modelos tridimensionais (3D) para providenciar aos estudantes um conteúdo muito mais imersivo e interativo. Esses Modelos 3D são ferramentas importantes para garantir uma compreensão rica do conteúdo em sala de aula (TABRIZI, 2008).

Esse artigo então, busca promover o uso da Realidade Virtual (RV) como uma ferramenta alternativa ou complementar aos docentes no ensino *online*. Desta forma se desenvolveu uma aplicação de RV utilizando do motor gráfico Unity. Essa aplicação poderá ser utilizada pela maioria dos Head Mounted Displays (HMD) de RV do mercado consumidor, mas terá como foco o Oculus Quest 2, devido a possibilidade de ser utilizado sem a necessidade de um computador conectado. Nessa aplicação, será providenciada uma sala de aula virtual semelhante a o que é esperado de sua versão real. Onde o professor pode lecionar suas aulas com as devidas ferramentas necessárias dentro dela, como: canetão, quadro, apagador e objetos didáticos. Sendo que tais ferramentas devem ser manuseáveis de forma intuitiva e que sejam de fácil acesso ao usuário, e ainda com uma baixa curva de aprendizado.

Dentro da aplicação, seria possível implementar virtualmente qualquer objeto a fim de dar uma representação visual didática mais compreensível aos alunos como, mostrar objetos e materiais exóticos, reações químicas perigosas, ação e reação de corpos físicos, dentro de virtualmente qualquer ambiente, seja esse uma sala de aula comum, um laboratório de química, ou uma estação espacial na lua. Porém, visto que essa aplicação será um conceito mais generalizado, as ferramentas disponibilizadas serão as de caráter essencial que sejam comuns com o maior número de disciplinas didáticas.

A aplicação foi utilizada por professores e por outras pessoas, a fim de qualificar as impressões, opiniões e críticas em relação a mesma, considerando-as para concluir a viabilidade do uso da tecnologia de RV na educação com a tecnologia que possuímos hoje, assim como extensões propondo possíveis melhorias que possam levar a aplicação a obter outro resultado mais favorável em relação ao seu público-alvo.

2 FUNDAMENTAÇÃO TEÓRICA

Nesta seção será abordado os temas presentes no artigo de forma subdividida. Onde na primeira parte será discutido sobre o que define um material como didático e sua importância dentro de uma sala de aula. Já na segunda parte será abordado o conceito, história e crescimento da tecnologia de Realidade Virtual. Por fim, serão apresentados os trabalhos correlatos.

2.1 FERRAMENTAS EDUCACIONAIS

Dentro de uma sala de aula se utiliza de materiais didáticos a fim de conseguir explicar um conteúdo para a sala de aula. Alguns desses materiais podem ser específicos para certa disciplina, porém outros podem ser comuns entre diversas disciplinas. Como especificado por Freitas (2009, p. 21), “os materiais e equipamentos didáticos são todo e qualquer recurso utilizado em um procedimento de ensino, visando à estimulação do aluno e à sua aproximação do conteúdo.”. Existem diversos recursos visuais, auditivos e audiovisuais que podem ser utilizados para lecionar, sejam estes objetos criados exclusivamente para fins educativos, ou que indiretamente podem ser usados para este fim. Para definir se um material pode ser considerado de uso didático, deve se observar alguns critérios na hora de sua seleção:

a) adequação aos objetivos, conteúdo e grau de desenvolvimento, interesse e necessidade dos alunos;
b) adequação as habilidades que se quer desenvolver, sejam estas cognitivas, afetivas ou psicomotoras;
c) simplicidade, baixo custo e manipulação acessível;
d) qualidade e atração (devem despertar curiosidade).

Um material didático consegue estabelecer uma comunicação entre professor e aluno, transformando uma aula monótona exclusivamente verbal em algo mais cativante. Assim ampliando o campo de experiencia do estudante tendo em vista que agora o mesmo possui uma representação visual do elemento em pauta, que de outro modo permaneceria abstrato (FREITAS, 2009)

Historicamente na educação do Brasil existem alguns recursos considerados universais para o ensino. Entre eles se destacam o quadro de giz ou lousa, por possuir um baixo custo, fácil instalação, ser um ótimo recurso visual acessível para todos os alunos e é versátil para diversas disciplinas. Outro equipamento é o Retroprojetor, capaz de projetar imagens em uma superfície de forma ampliada dando maior visibilidade para toda a sala, e por fim um aparelho de vídeo e DVD como recurso audiovisual permitindo a reprodução de áudio e vídeo dos assuntos necessários para a sala (FREITAS, 2009).

Um exemplo do uso da tecnologia como material didático na educação são as lousas digitais. É possível encontrar esse dispositivo no mercado em versões de grande tamanho e são utilizados em diversas escolas como uma alternativa ao quadro e giz comum. Existem os professores que ficam céticos em relação ao uso dessa lousa e preferem utilizar das ferramentas tradicionais, enquanto outros consideram a tecnologia como algo revolucionário. A lousa permite que a aula seja mais dinâmica, podendo ministrar o conteúdo de forma mais rápida e interativa. Porém, por se tratar de um produto tecnológico, é necessário disponibilizar um curso de capacitação para que os professores possam usufruir da tecnologia de forma adequada.

2.2 REALIDADE VIRTUAL

O conceito de Realidade Virtual (RV) já é datado desde a década de 1800. Dispositivos como o Kinetoscópio e Mutoscópio permitiam ao usuário isolar sua visão de tudo ao seu redor enxergando apenas dentro de outra realidade (42GEARS, 2019). O primeiro capacete de realidade virtual ou Head Mounted Display (HMD) como o que é utilizado em dispositivos de RV hoje em dia, foi desenvolvido somente na década de 1960 por Ivan Sutherland, cujo denominou o mesmo de Ultimate Display (TORI; KIRNER; SISCOUTTO, 2006).

Ao longo dos anos, a tecnologia de RV se encontrava apenas em produtos de um nicho pequeno, estes que raramente se tornavam produtos bem-sucedidos no mercado como o VirtualBoy (FLANAGAN, 2018). Apenas recentemente, essa tecnologia começou a ter um atrativo maior no mercado, com diversos fabricantes desenvolvendo vários dispositivos de HMD de RV cada vez mais acessíveis e com mais recursos tecnológicos. Resultado disso podem

ser observados, e a previsão projeta que as vendas de dispositivos de RV e Realidade Aumentada cheguem a 26 milhões por ano até o ano de 2023 (STATISTA, 2020).

Se tem como exemplo o HMD de RV Oculus Quest 2 desenvolvido pela empresa Oculus que foi fundada por Palmer Luckey e Brendan Iribe em Irvine na California em 2012. A empresa então, foi adquirida pela Facebook em 2014 onde continuou desenvolvendo dispositivos de RV (TAYLOR CLARK, 2014). O Quest 2 foi lançando em outubro de 2020, com um System-On-Chip (SOC) Qualcomm Snapdragon XR2 que faz parte da linha Snapdragon de SOC's para RV e RA. O Oculus Quest 2 conta com 6 gigabytes de memória RAM, seu display é um painel LCD de alta taxa de atualização de até 120hz, com uma resolução de 3664x3840, que equivale a 1832x1920 para cada olho. O Quest 2 acompanha 2 controles Oculus Touch com sensores que permitem ao usuário interagir com o ambiente virtual (FACEBOOK TECHNOLOGIES, 2021a). O diferencial desse dispositivo em relação a outros HMD do mercado é sua capacidade de funcionar de forma autônoma. Enquanto a grande maioria dos outros dispositivos de RV funcionam como uma extensão de um computador pessoal, necessitando de cabos para transferir a imagem do computador até o dispositivo, o Oculus Quest 2 não precisa.

Além disso, o Quest 2 consegue calcular o posicionamento do usuário em relação ao ambiente físico do usuário através das 4 câmeras localizadas nas suas extremidades conhecido como *inside-out tracking*. Ele utiliza dessas câmeras com algoritmos de processamento de imagem para obter a posição e rotação da cabeça do usuário, posição e rotação dos controles, assim como para a sua função de *hand tracking* que possibilita visualizar e controlar o dispositivo virtualmente com as próprias mãos. Enquanto outros dispositivos de RV precisavam de sensores externos para fazer esse posicionamento do HMD e dos controles em relação ao ambiente real (DAVID HEANEY, 2019). Outro fator que impulsionou muito a popularidade desse dispositivo é o seu valor inicial de lançamento de USD$ 299,00, que é um valor muito abaixo dos dispositivos das empresas concorrentes (ROBERTSON, 2020).

Para desenvolver para o dispositivo Oculus Quest 2 assim como outros HMDs da Oculus, a empresa disponibiliza o Oculus PC SDK no seu site para desenvolvedores. O SDK possui todas as APIs necessárias para desenvolver para os dispositivos da plataforma. Caso opte por utilizar de um motor gráfico como Unity ou Unreal Engine para desenvolver, a empresa também fornece pacotes específicos integrados com esses motores (FACEBOOK TECHNOLOGIES, 2021b).

Caso o desenvolvedor queira desenvolver para diversos HMDs de diferentes fabricantes é possível utilizar de bibliotecas terceiras que integram essas APIs do SDK de diferentes fabricantes. Como é o caso do framework XR desenvolvido pela empresa Unity Technologies para ser utilizado em conjunto com o motor gráfico Unity. O framework XR é um pacote que integra as APIs de diferentes plataformas de RV e RA a fim de facilitar o desenvolvimento para o maior número de dispositivos disponíveis de maneira abstraída e genérica (UNITY TECHNOLOGIES, 2021).

Hoje a tecnologia, embora ainda em constante evolução, já consegue entregar uma experiência de uso satisfatória o suficiente para agradar os seus usuários (TCHA-TOKEY *et al*., 2016). O número de vendas de produtos de RV cada vez maior é reflexo disso. Dispositivos como o Oculus Quest 2 com custo não tão alto e bom desempenho são essenciais para impulsionar a tecnologia. O mercado de RV ainda é recente e seu potencial é enorme. Mas é necessário a mobilização dos desenvolvedores para agregar a estas novas plataformas emergentes de RV, para então evoluir essa tecnologia e integrar cada vez mais com a sociedade como uma extensão de como consumimos e entregamos conteúdo para a educação e entretenimento (SANTOS, 2021).

2.3 TRABALHOS CORRELATOS

A seguir serão apresentados os trabalhos que possuem características de alguma maneira semelhante aos objetivos principais do artigo. O primeiro (

Quadro 1) é um trabalho que relata o desenvolvimento de um sistema de Agent based Virtual Reality (AVR) que permite aos professores lecionarem, e aos alunos de estudarem em um ambiente virtual (TABRIZI, 2008). O segundo (Quadro 2) é um artigo que conceitualiza uma sala de aula em RV que possibilite aprimorar a educação através de representações mais interativas para as diversas matérias lecionadas. O terceiro (Quadro 3) mostra o desenvolvimento de um ambiente virtual urbano, com ruas para os usuários aprenderem sobre educação de trânsito.

Quadro 1 – Agent and Virtual Reality-based Course Delivery System

Referência	Tabrizi (2008)
Objetivos	Maximizar a efetividade do ensino *online*, desenvolvendo um sistema de ensino para professores lecionarem e alunos poderem atender as aulas, realizar atividades e avaliações.
Principais funcionalidades	Uma plataforma 3D de RV com um sistema de gerenciamento de cursos, um sistema dinâmico de avaliações, um ambiente de comunicação com base em multimídia, um modelo cliente/servidor efetivo com uma camada de segurança avançada, um quadro branco e apresentador de PowerPoint eletrônico esse ambiente deve ter um aspecto de campus como os laboratórios, e deve poder capturar os movimentos do professor durante as aulas.
Ferramentas de desenvolvimento	O autor não especifica as ferramentas ou linguagens utilizadas para desenvolver o sistema.
Resultados e conclusões	O autor não realiza uma avaliação sobre os resultados, apenas informa que existe trabalhos de outros professores que estão estudando a efetividade de seu sistema. Porém afirma que a atitude dos estudantes em relação a sistemas de jogos é amplamente acordada como algo positivo. Também consta que o sistema AVR providenciará aos estudantes uma plataforma para aprender em um ambiente interativo de multimídia similar ao mundo dos jogos.

Fonte: elaborado pelo autor.

Quadro 2 – An overall solution of Virtual Reality Classroom

Referência	Dong (2016)
Objetivos	O autor Dong (2016) propõe uma solução para a integração geral de uma sala de aula em RV, combinando a tecnologia com as diversas disciplinas. A fim de atingir uma convergência compatível com a sala de aula presencial comum, com a sala de aula de informática, salas de multimídia entre outras.
Principais funcionalidades	Dong (2016) afirma que a sala de aula em RV deve ser constituída pela combinação de animação virtual, espaços virtuais, para criar ambientes imersivos de aprendizado 3D. então. Exemplifica por matérias, de forma pontual e breve, um exemplo de como a RV pode ajudar nas mesmas: a) matemática: a sala RV permite visualizar conceitos matemáticos abstratos e complexos de forma sensorial, mapeamento de coordenadas, porcentagem, objetos geométricos, entre outros; b) biologia: pode-se renderizar o mundo microscópico e observar a multiplicação de células, vírus entre outros; c) física: representando forças, movimento e energia, seja física, térmica, óptica, atômica, mecânica e elétrica de forma visual ajudando a dominar as leis da física; d) química: simular reações químicas sem a necessidade de se preocupar com a obtenção, perda e perigo dos reagentes químicos; e) astronomia: pode levar os alunos a literalmente andar em algum planeta; f) engenharias: visualizar protótipos antes de produzir eles fisicamente. Também aponta que o ensino pela RV não é necessariamente uma alternativa.
Ferramentas de desenvolvimento	O autor apenas conceitualiza um ambiente virtual de RV.
Resultados e conclusões	Concluiu-se pelo autor Dong (2016) que a proposta que a RV traz é extremamente atrativo, e vai mudar a forma que as pessoas pensam em um sentido, e até mudar o entendimento de tempo e espaço. A RV pode desenvolver novos meios de ensino e aprendizado, e terá grande importância na área educacional com o decorrer do tempo.

Fonte: elaborado pelo autor.

Quadro 3 - Developing a Virtual Reality Educational Environment for Traffic Education

Referência	Chatzizisis *et al.* (2019)
Objetivos	Familiarizar os usuários na navegação dentro de um ambiente urbano, com a educação de trânsito, bom comportamento de motoristas, entendimento das prioridades no sistema de ruas e aprimorar a movimentação do usuário dentro do ambiente como pedestre.
Principais funcionalidades	O usuário deve poder se movimentar livremente dentro do ambiente virtual, interagindo com objetos virtuais como: sinalização de trânsito, semáforos, outros veículos, veículos prioritários e personagens não jogáveis.
Ferramentas de desenvolvimento	O ambiente virtual foi implementado utilizando o OpenSim, uma aplicação servidora multiplataforma, multiusuário e tridimensional. Os objetos existentes dentro do mundo virtual foram criados utilizando Blender, uma plataforma open-source para design tridimensional.
Resultados e conclusões	Educação de Trânsito é um tema crucial da educação para humanos já que o conhecimento que é adquirido com essa é aplicado no cotidiano de muitas pessoas. Se locomover de acordo com as leis, com bom comportamento de pedestres e motoristas é o passo mais importante para ruas mais seguras, enfatizando o ensino dessa matéria dês do ensino básico, para que jovens estudantes adotem e apliquem esse conhecimento de bom comportamento no trânsito o quanto antes. Uma forma de alcançar isso é com a aplicação da tecnologia, especialmente da RV e suas vantagens para educação. Com a aplicação desenvolvida, os usuários podem simular situações de perigo no trânsito e como agir nas mesmas sem correr nenhum perigo real, estando em um ambiente confortável.

Fonte: elaborado pelo autor.

3 DESCRIÇÃO DA APLICAÇÃO

A aplicação foi desenvolvida utilizando o motor gráfico Unity usando *scripts* programados na linguagem C# nativa do Unity que definem o comportamento dos objetos e eventos. Também foi utilizado o framework XR desenvolvido pela Unity Technologies para controlar o Head Mounted Display (HMD) de Realidade Virtual (RV), que nesse caso é o Oculus Quest 2. Foi modelado e estruturado uma sala de aula virtual análoga a uma sala de aula física real com um ambiente externo visível pelas janelas da sala, a fim de trazer uma maior imersão ao usuário remetendo um ambiente escolar real (Figura 1).

Alguns aspectos do ambiente foram compostos apenas de formas geométricas primitivas como quadrados e esferas. Dentro do Unity é possível redimensionar, rotacionar e posicionar tais formas a fim de estruturar os objetos desejados como: parede, piso, mesa entre outros objetos. Com esses objetos modelados é possível utilizar de texturas que são arquivos de imagem propícios para serem aplicadas em um objeto 3D para entregar uma representação mais fiel ao que quer se representar. Outras partes do ambiente foram formadas utilizando de modelos 3D disponíveis gratuitamente na internet.

Figura 1 - Sala de aula modelada

Fonte: elaborado pelo autor.

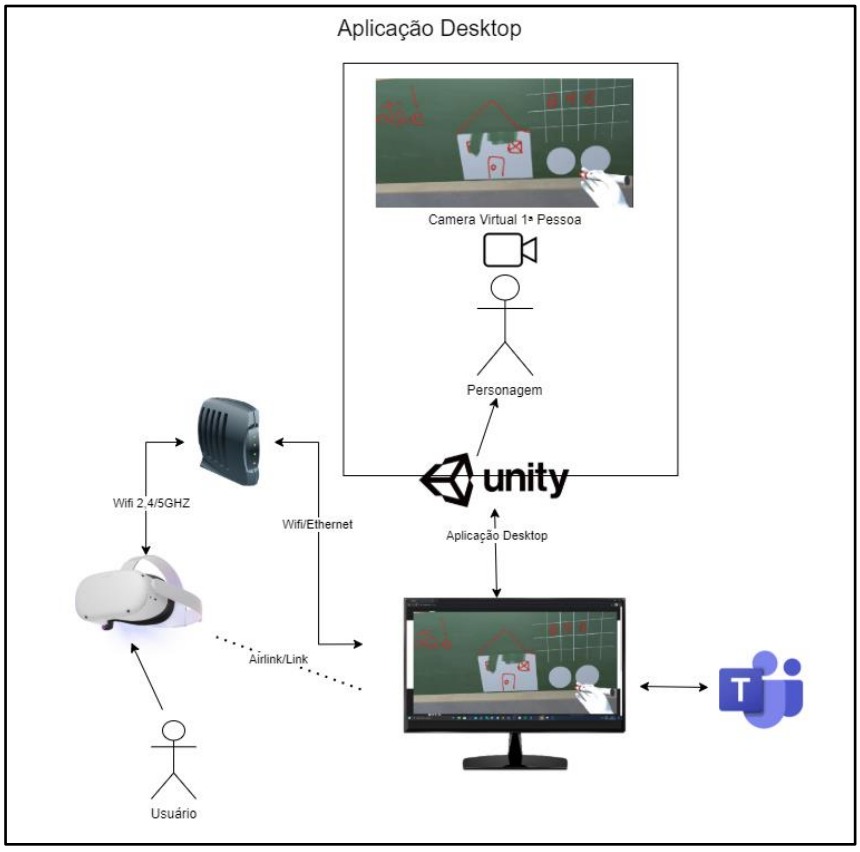
O ambiente desenvolvido teve um quadro negro implementado onde o usuário pode escrever utilizando de 3 canetões disponível nas cores: vermelho, azul e verde. Também foi implementado um apagador e 4 formas pré-definidas para serem desenhadas no quadro, sendo possível alterar sua cor, tamanho e posição no quadro antes de efetuar seu desenho no quadro. É possível andar virtualmente nesse ambiente e o usuário tem mãos virtuais visíveis que respondem aos comandos dos controles. Também é possível pegar todos os objetos e manipular eles com as mãos, simulando colisão e gravidade nos mesmos.

3.1 ESPECIFICAÇÃO

A aplicação pode ser utilizada de duas maneiras diferentes, na sua versão desktop (Figura 2) ou na sua versão móvel (Figura 3). A versão desktop permite que o usuário tenha uma fidelidade visual maior, mas requer um computador com alto poder computacional gráfico. Já na versão móvel a aplicação pode rodar dentro de um Oculus Quest 2 sem qualquer pré-requisito. A seguir serão especificados todos os Requisitos Funcionais (RFs) e Requisitos Não funcionais (RNFs):

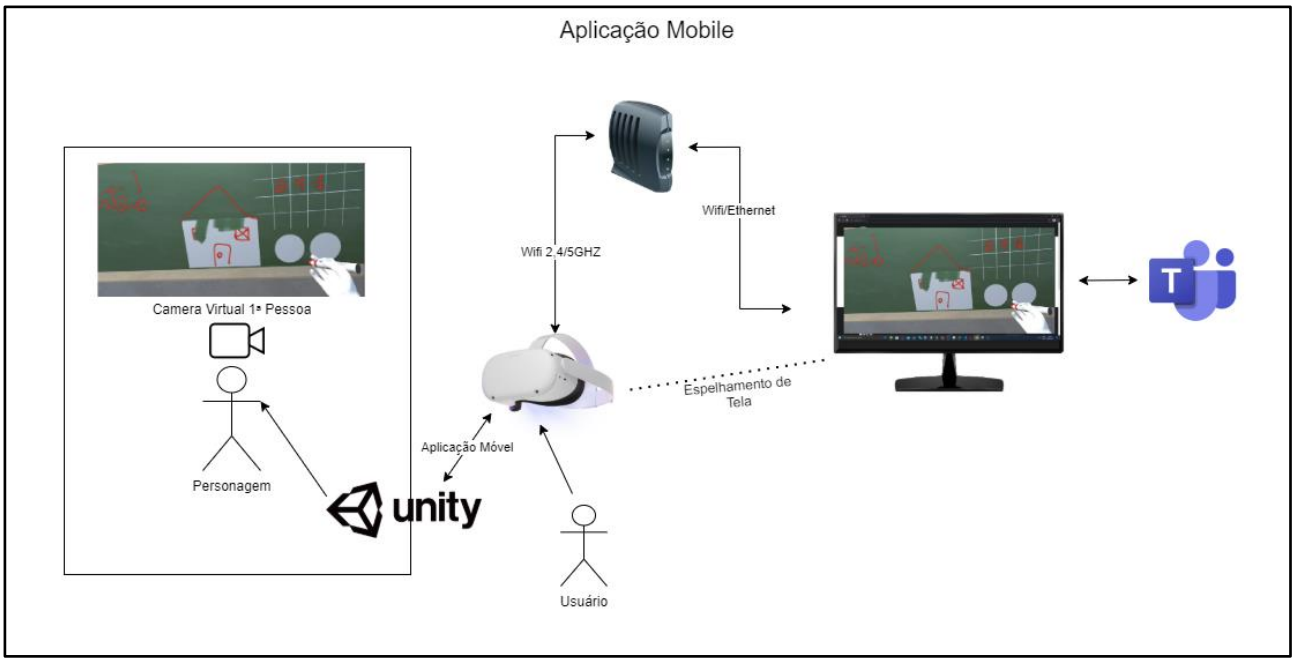
a) permitir que o usuário se locomova dentro do ambiente virtual (RF);
b) permitir que o usuário interaja com os objetos virtuais (RF);
c) permitir que o usuário utilize de canetões para desenhar no quadro negro (RF);
d) permitir que o usuário utilize um apagador para apagar o que escreveu no quadro negro (RF);
e) permitir que o usuário utilize de formas pré-definidas, podendo posicionar, redimensionar, e trocar suas cores para facilmente desenhar essas formas no quadro negro (RF);
f) permitir que o usuário limpe todo o quadro negro instantaneamente (RF);
g) desenvolver um ambiente virtual semelhante a uma sala de aula (RF);
h) desenvolver exercícios pré-definidos para o usuário realizar e se situar com a aplicação (RF);
i) ser desenvolvido na plataforma Unity com seu motor gráfico proprietário (RNF);
j) ser programado na linguagem de programação C# nativa do Unity (RNF).

Figura 2 - Diagrama de uso da aplicação desktop



Fonte: elaborado pelo autor.

Figura 3 - Diagrama de uso da aplicação mobile



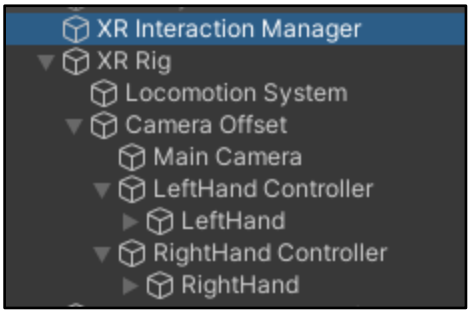
Fonte: elaborado pelo autor.

A logo do Microsoft Teams apresentado no diagrama de uso serve como exemplo de uma plataforma de streaming que permite compartilhar a execução da aplicação seja pela janela da própria aplicação ou pelo espelhamento de tela do Oculus Quest 2 para ministrar o conteúdo para as pessoas. A concepção é de que a aplicação vai disponibilizar esta imagem em tempo real da execução dela do ponto de vista do usuário para ser então compartilhada de n maneiras e não se restringe exclusivamente ao Microsoft Teams.

3.2 IMPLEMENTAÇÃO

Para desenvolver usando Realidade Virtual dentro do Unity se utilizou o *framework* XR Interaction Toolkit. Este *framework* é um pacote de componentes de alto nível para interação com dispositivos de RV desenvolvido pela Unity Technologies, com o intuito de ter um *framework* genérico que consiga ser compatível com o maior número possível de dispositivos de RV. Utilizando esse *framework* é possível preparar uma cena em minutos com grande compatibilidade entre os diversos dispositivos de RV. Utilizando de seu componente `XRRig` (Quadro 4), cujo mesmo já disponibiliza de uma câmera posicionada relativa à posição do HMD, também dispõe de duas âncoras que representam o controle esquerdo e direito, em conjunto com o `XR Interaction Manager` que gerencia os controles. Tendo esses componentes na cena, já é possível ter a posição do seu HMD e dos controles em tempo real no espaço 3D assim como os eventos de todos as entradas dos dispositivos, sendo possível então, estruturar *scripts* customizados em cima desses dados para obter qualquer comportamento desejado para cada caso de uso.

Quadro 4 - Hierarquia do componente XR Rig dentro do Unity



Fonte: elaborado pelo Autor.

Uma funcionalidade importante do trabalho é a manipulação e interação dos objetos virtuais com as mãos. O objeto quando segurado, deve manter suas propriedades físicas, deve colidir com objetos e não atravessar eles, e devem sempre se manter na posição e rotação relativa à posição virtual da mão do usuário.

Para obter esse resultado em vez de optar por transformar os objetos em corpos kinemáticos sem gravidade, e então, a cada quadro estar atualizando a posição do objeto para a mesma posição das mãos, o que acabaria removendo a colisão desse objeto com outros corpos estáticos. Se decidiu então aplicar forças em cima desse objeto a fim de movimentar ele até a mesma posição das mãos. Assim mantendo sua colisão com todos objetos permitindo uma maior interatividade com o ambiente.

Para criar essa funcionalidade se criou um *script* para cada mão do usuário, nomeado de `PickupHand`. Esse *script* vai se encarregar de várias funções, como: escolher o objeto mais próximo a mão, pegar o objeto, movimentar e rotacionar o objeto, entre outras funções mais específicas.

A grande maioria dos *scripts* Unity herdam da classe `MonoBehaviour` e possuem alguns métodos que são chamados em determinados momentos. Entre esses temos o `Awake()` que é similar ao `Start()`. O `Awake()` é um método que vai ser chamado no início do *script*, e é comumente utilizado como inicializador de variáveis. O *script* `PickupHand` utiliza essa função para popular a variável `hand` com o objeto que representa a mão 3D virtual que está localizado hierarquicamente abaixo do objeto que está com o *script*, para manipular esse objeto posteriormente (Quadro 5).

Quadro 5 - Variáveis da classe `PickupHand`

```
Unity Script | 13 references
public class PickupHand : MonoBehaviour
{
    public float distToPickup = 0.3f;
    public bool isHandClosed = false;

    public XRController controller = null;
    public LayerMask pickupableLayer;
    public Rigidbody holdingTarget;
    private GameObject hand;
    private isTrigger isTrigger;
    private GameObject holdingObjectHand;
    private GameObject holdingObject;
    private GameObject objectToHide;
    private GameObject objectHandToHide;
    private GameObject auxObjectToDestroy;
    public GrabbedObjectsManager grabbedObjectsManager;
    public PickupHand otherHand;
    private bool firstMove = true;

    public WhichHand whichHand = WhichHand.RIGHT;

    4 references
    public enum WhichHand { LEFT, RIGHT };

    // Start is called before the first frame update
    Unity Message | 0 references
    void Awake()
    {
        hand = this.transform.GetChild(0).gameObject;
    }
```

Fonte: elaborado pelo autor.

A parte principal do *script* se dá dentro da função `MonoBehaviour.FixedUpdate`. Essa função é chamada de acordo com a frequência definida para o tempo do sistema de físicas do Unity. No caso do projeto essa frequência é de cerca de 143 vezes por segundo, independente da taxa de quadros por segundo. Nessa função primeiramente se verifica, usando o `XRController`, se o botão de garra (que representa o botão que vai fechar a sua mão virtual) está ou não pressionado, e então, se entra no método principal `GrabAndMoveObjects()`.

O *script* `GrabAndMoveObject()` (Quadro 6) terá uma função diferente de acordo com o estado da variável `isHandClosed` que vai dizer se o usuário está atualmente com a mão fechada ou aberta. Isso vai definir se o usuário deveria estar segurando um objeto em suas mãos (mão fechada) ou ainda buscando algum objeto para pegar. A primeira clausula do `if` vai ser executada caso a mão esteja aberta, onde será chamado o método `FindObjectToGrab()`.

Quadro 6 - Método `GrabAndMoveObject()` da classe `PickupHand`

```
1 reference
private void GrabAndMoveObject()
{
    if (!isHandClosed)
    {
        if (auxObjectToDestroy != null)
        {
            Destroy(this.auxObjectToDestroy);
            this.auxObjectToDestroy = null;
        }
        FindObjectToGrab();
    }
    else
    {
        //Se a mão está fechada e possui um objeto selecionado
        if (holdingTarget && holdingTarget != otherHand.holdingTarget)
        {
            if (firstMove)
            {
                Renderer[] childArray = holdingTarget.GetComponentsInChildren<Renderer>();
                foreach (Renderer render in childArray)
                {
                    render.material.SetFloat("_OutlineWidth", 0.000f);
                }
                this.holdingTarget.position = transform.position;
                this.holdingTarget.rotation = transform.rotation;
                firstMove = false;
            }
            //Se a mão está visível, esconde ela.
            if (hand.active)
                {
                    StopCoroutine(showHand());
                    hand.SetActive(false);
                    showHoldingObjectHand();
                }
                bool isObjectAgainstWall = isHoldingTargetAgainstWall();

                MoveAndRotateHoldingObject(isObjectAgainstWall);
            }
        }
    }
```

Fonte: elaborado pelo autor.

O método `FindObjectToGrab()` (Quadro 7) utiliza da biblioteca `Physics` do próprio Unity para calcular uma esfera de raio modificável em relação a posição da mão do usuário. Todo objeto que possuir um `Collider` dentro desse raio será então retornado dentro de um `array` do tipo `Collider`. Após é feito o calculo da distância entre esses objetos, e então se define o objeto que possui o `Collider` com a menor distância em relação a mão. Assim será definido na variável `holdingTarget` como o objeto à ser pegado.

Quadro 7 - Método `FindObjectToGrab()` da classe `PickupHand`

```
1 reference
private void FindObjectToGrab()
{
    //Mostra a mão quando aberta
    if (!hand.active)
        StartCoroutine(showHand());
    if (this.holdingObjectHand != null && this.holdingObjectHand.active)
    {
        this.holdingObjectHand.SetActive(false);
    }
        //Verifica em um raio próximo a mão todos os objetos com colliders
        Collider[] colliders = Physics.OverlapSphere(transform.position, distToPickup, pickupableLayer);
    if (colliders.Length > 0)
    {
        //Se existe objeto dentro do raio, pegue o mais próximo da mão
        float lastDistance = float.MaxValue;
        Collider closestCollider = new Collider();
        foreach(Collider collider in colliders)
        {
            float distance = Vector3.Distance(collider.transform.position, transform.position);
            if(distance < lastDistance)
            {
                lastDistance = distance;
                closestCollider = collider;
            }
        }
        if (closestCollider != null)
        {
            Rigidbody aux = closestCollider.transform.root.GetComponent<Rigidbody>();
            if (otherHand.holdingTarget != aux && aux != this.holdingTarget)
            {
                Renderer[] childArray;
                if (holdingTarget != null)
                {
                    childArray = holdingTarget.GetComponentsInChildren<Renderer>();
                    foreach (Renderer render in childArray)
                    {
                        render.material.SetFloat("_OutlineWidth", 0.000f);
                    }
                }
                holdingTarget = aux;
                childArray = holdingTarget.GetComponentsInChildren<Renderer>();
                Debug.Log(childArray.Length);
                foreach (Renderer render in childArray)
                {
                    render.material.SetColor("_OutlineColor", new Color(0, 255, 244, 1));
                    render.material.SetFloat("_OutlineWidth", 0.005f);
                }
                firstMove = true;
            }
        }
    }
```

Fonte: elaborado pelo autor.

Entrando na outra cláusula `else` do método `GrabAndMoveObject()`, que será executada caso a mão do usuário esteja fechada, vai verificar se previamente fora populado algum objeto na variável `holdingTarget`. Que seria o objeto mais próximo da mão, e então, é chamado o método `MoveAndRotateHoldingObject()` (Quadro 8) que fica encarregado de aplicar forças para rotacionar e translatar o objeto até a posição da mão.

Utilizando de funções `Quaternion` e `Vector3` é aplicado um vetor de velocidade ao objeto. O valor desse vetor se dá pelo delta da distância entre a posição do objeto segurado, e a posição da mão. Também é aplicado uma velocidade angular, que no Unity, se dá por um vetor tridimensional onde cada dimensão representa um eixo de ângulo Euler. Já a rotação de cada eixo é dada pela rotação quatérnion da mão multiplicada pelo inverso da rotação quatérnion do objeto segurado, que então é convertida para ângulos Eulers.

Quadro 8 - Método `MoveAndRotateHoldingObject()` da classe `PickupHand`

```
1 reference
private void MoveAndRotateHoldingObject(bool isObjectAgainstWall)
{
    if (holdingTarget.name.Contains("ShapeParent") && this.auxObjectToDestroy == null){
        this.auxObjectToDestroy = GameObject.Instantiate(holdingTarget.gameObject) as GameObject;
        Destroy(auxObjectToDestroy.transform.root.GetComponent<changeColor>());
        this.auxObjectToDestroy.GetComponent<Rigidbody>().constraints = RigidbodyConstraints.None;
        this.auxObjectToDestroy.GetComponent<Rigidbody>().useGravity = true;
        holdingTarget = this.auxObjectToDestroy.GetComponent<Rigidbody>();
    }
    //Move a posição do objeto na mão
    this.holdingTarget.velocity = (transform.position - holdingTarget.transform.position) / Time.fixedDeltaTime;
    if (!isObjectAgainstWall)
    {
        //Rotaciona o objeto na mão
        this.holdingTarget.maxAngularVelocity = 20;
        Quaternion deltaRot = transform.rotation * Quaternion.Inverse(this.holdingTarget.transform.rotation);
        Vector3 eulerRot = new Vector3(Mathf.DeltaAngle(0, deltaRot.eulerAngles.x), Mathf.DeltaAngle(0, deltaRot.eulerAngles.y),
            Mathf.DeltaAngle(0, deltaRot.eulerAngles.z));
        eulerRot *= 0.95f;
        eulerRot *= Mathf.Deg2Rad;
        this.holdingTarget.angularVelocity = eulerRot / Time.fixedDeltaTime;
    }
}
```

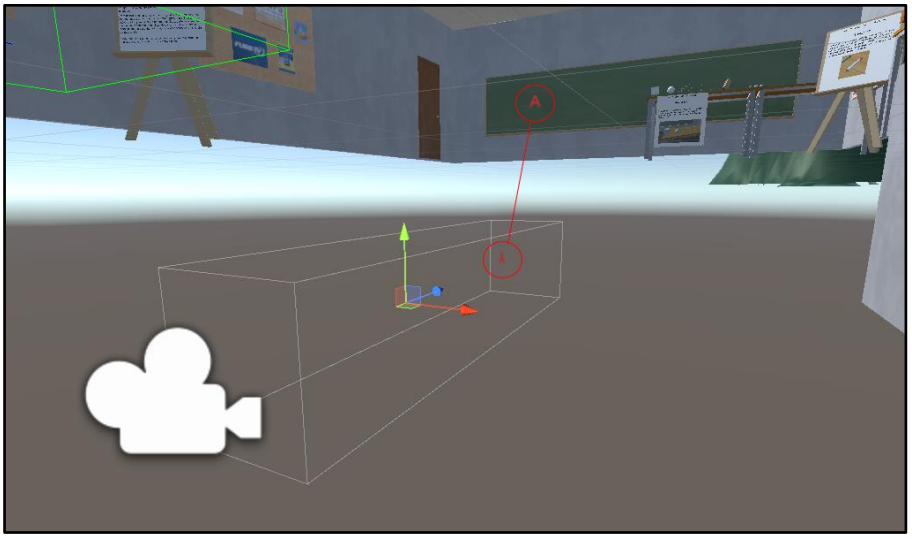
Fonte: elaborado pelo autor.

Esses trechos de códigos tomam conta da parte da localização, e então de tratar a movimentação dos objetos na mão virtual do usuário. Ainda existem mais alguns trechos de código nesse *script* que tomam conta de controlar algumas outras particularidades da aplicação desenvolvida, mas que não carecem de detalhes.

Após a parte de manipulação dos objetos codificada se faz necessário criar uma mecânica de escrita (Figura 4), que permita ter um quadro negro, similar ao que se espera de um quadro negro real. Onde se possa escrever e apagar, e também outras funcionalidades, como a de poder posicionar, redimensionar formas e objetos para serem postas sobre o quadro (como quadrados, círculos, gráficos e matrizes).

Para permitir ter um objeto com uma textura que seria desenhável em tempo real foi utilizado de uma `RenderTexture`. Onde uma câmera virtual ortográfica teria o `RenderTexture` configurado como alvo, como resultado disso, tudo que entrasse dentro do campo de visão da câmera é então desenhado no `RenderTexture`. O `RenderTexture` então, seria aplicado em um material, e esse material aplicado como textura em um objeto. Nesse caso, de um plano translúcido posicionado ligeiramente a frente do quadro negro.

Figura 4 - Representação da solução para desenhar no quadro negro



Fonte: elaborado pelo autor.

Assim se tem uma relação entre, o usuário manipulando esse objeto com o material do `RenderTexture` e a câmera virtual posicionada em outro espaço. Quem controla toda a lógica por trás da funcionalidade de desenhar no quadro é o script `NewTexturePainter`.

Já no `MonoBehaviour.FixedUpdate()` o *script* verifica se o usuário está segurando algum objeto que seria uma caneta ou um apagador, e então, verifica se esse objeto está pressionado contra o quadro negro. Se sim, então o *script* vai iniciar a rotina `DoAction()`.

No método `DoAction()` é feito então o instanciamento de um `sprite` que vai representar efetivamente a pintura do canetão, que representa a textura que será visível ao escrever algo. Então é verificado se o objeto segurado é um apagador, um canetão vermelho, azul, ou verde, e dependendo de sua cor, será alterado a coloração do material assim como do seus componentes necessários. Também é feito um redimensionamento desse `sprite`, para corrigir a distorção de proporção entre o material onde a textura está aplicada que no caso é um retângulo, e a resolução do `RenderTexture` que é um quadrado (Quadro 9).

Quadro 9 - Trecho da rotina `DoAction()` da classe `NewTexturePainter`

```
224                 currentPaintingBrush = (GameObject)Instantiate(Resources.Load("TexturePainter-Instances/BrushEntity"));
225                 currentPaintingBrush.AddComponent<TrailRenderer>();
226                 if (pen.name.Contains("RED"))
227                 {
228
229                     currentPaintingBrush.GetComponent<SpriteRenderer>().color = Color.red;
230                     currentPaintingBrush.transform.localScale = new Vector3(1, 1, 1) * 0.0008f;
231                     trailMaterial.SetColor("_EmissionColor", Color.red);
232                     currentPaintingBrush.GetComponent<TrailRenderer>().material = trailMaterial;
233                     currentPaintingBrush.GetComponent<TrailRenderer>().time = 0.25f;
234                     currentPaintingBrush.GetComponent<TrailRenderer>().startWidth = 1 * 0.002f;
235                     currentPaintingBrush.GetComponent<TrailRenderer>().endWidth = 1 * 0.002f;
236                     currentPaintingBrush.GetComponent<TrailRenderer>().minVertexDistance = 0.005f;
237                     currentPaintingBrush.GetComponent<TrailRenderer>().startColor = Color.red;
238                     currentPaintingBrush.GetComponent<TrailRenderer>().endColor = Color.red;
239                     currentPaintingBrush.GetComponent<TrailRenderer>().shadowCastingMode = UnityEngine.Rendering.ShadowCastingMode.Off;
240                     currentPaintingBrush.GetComponent<TrailRenderer>().textureMode = LineTextureMode.DistributePerSegment;
241                     brushColor = Color.red;
242                     brushColor.a = 1.0f;
243                 }
```

Fonte: elaborado pelo autor.

Após instanciar o objeto que será o `sprite` do desenho é necessário buscar onde esse objeto deve ser posicionado em relação a posição do canetão. Para isso é chamado o método `HitTestUVPosition()` onde é feito um `raycast` partindo do objeto em direção ao eixo Z frontal do mesmo. Caso o `hit` desse `raycast` colida com um objeto que esteja na camada `DrawingGlass` (que seria o objeto desenhável) então calcula-se a partir desse acerto, a posição correspondente na câmera virtual ortográfica que está apontando para o `RenderTexture` (Quadro 10). Tendo essa posição, é então movida a `sprite` do desenho até essa localização, resultando em uma projeção dessa `sprite` no quadro negro. Essa posição é atualizada sempre que o canetão ou outro objeto especificado esteja encostando no quadro negro. O fato da câmera virtual estar posicionada em um local do mundo, onde não existe nenhum fundo que deve ser desenhado pelo motor gráfico, nunca será feita a limpeza da tela a cada quadro renderizado, resultando em uma trilha por onde o `sprite` instanciado foi movido.

Quadro 10 - Método `HitTestUVPosition()` da classe `NewTexturePainter`

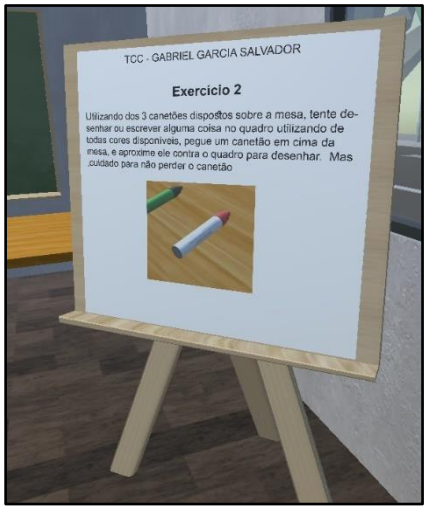
```
305     bool HitTestUVPosition(ref Vector3 uvWorldPosition)
306     {
307         RaycastHit hit;
308         if (pen.GetComponent<isTrigger>().isTriggered && Physics.Raycast(
309             pen.transform.GetChild(0).position, pen.transform.GetChild(0).forward, out hit, 10f, 3 << LayerMask.NameToLayer("DrawingGlass")))
310         {
311             MeshCollider meshCollider = hit.collider as MeshCollider;
312             if (meshCollider == null || meshCollider.sharedMesh == null)
313                 return false;
314             Vector2 pixelUV = new Vector2(hit.textureCoord.x, hit.textureCoord.y);
315             uvWorldPosition.x = pixelUV.x - canvasCam.orthographicSize;//To center the UV on X
316             uvWorldPosition.y = (pixelUV.y - canvasCam.orthographicSize) / 5;//To center the UV on Y
317             uvWorldPosition.z = 0.0f;
318             return true;
319         }
320         else
321         {
322             return false;
323         }
324     }
325
```

Fonte: elaborado pelo autor.

A sala de aula virtual foi modelada em sua maior parte utilizando de primitivas do Unity, como planos, blocos, cilíndros e esferas, em conjunto com `materials` e `textures`. Outras partes da sala e alguns objetos foram adquiridos através de `assets` gratuitos disponibilizados no *website* da loja Unity. Assim como de `assets` e `textures` gratuitos disponibilizadas na internet formando assim um espaço virtual com características próximas a de uma sala de aula real, a fim de trazer uma familiaridade ao usuário.

Dentro do ambiente foram desenvolvidos 3 exercícios básicos para que o usuário possa obter um primeiro contato com a manipulação dos objetos virtuais com os atuadores do dispositivo de realidade virtual. Assim como ensinar a utilizar os recursos disponibilizados, como: canetão, apagador e as formas desenháveis. Esses exercícios foram postos virtualmente em um cavalete para que o usuário os localize e leia as instruções (Figura 5).

Figura 5 - Cavalete contendo o Exercício 2

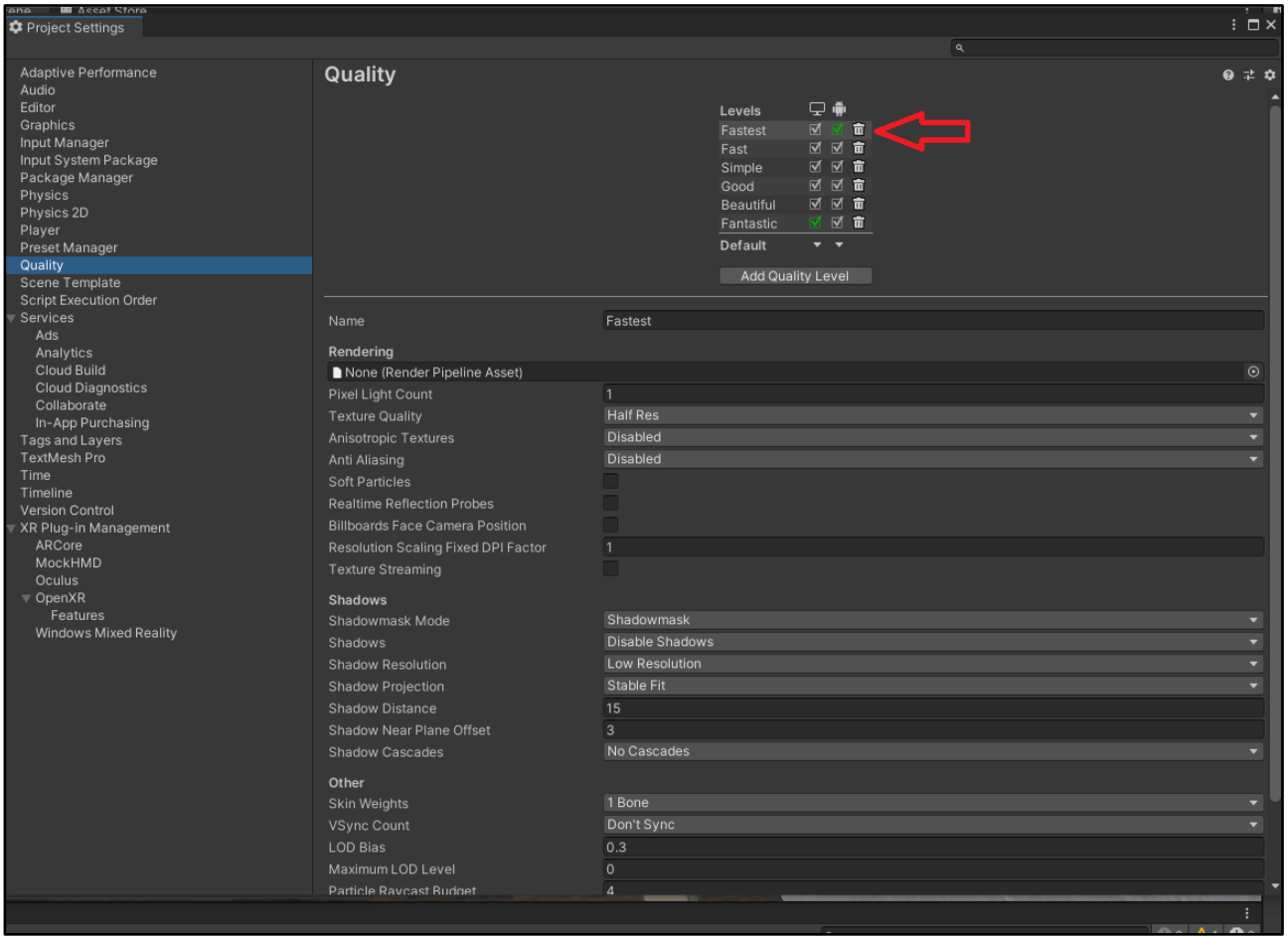


Fonte: elaborado pelo autor.

Ao desenvolver uma aplicação de RV é necessário tomar certos cuidados, pois se deve considerar que a renderização será feita duas vezes, uma para cada olho, assim possibilitando a estereoscopia. Essa imagem será de alta resolução, ainda sendo necessário manter uma taxa estável mínima de 72 quadros por segundo, o que é imprescindível para evitar desconforto ao usuário ao se movimentar. Conseguir atingir essa taxa de quadros por segundo em um dispositívo móvel como o Oculus Quest 2 se torna uma tarefa ainda mais complexa, considerando o poder computacional limitado do mesmo.

Para conseguir atingir uma taxa de quadros aceitável no *deploy* da aplicação foi necessário optar por reduzir ao máximo o número de polígonos dos objetos na cena. Com isso em mente foi utilizado apenas de assets *LowPoly* para modelar o ambiente, principalmente a área externa. Texturas entretanto são importantes que tenham alta resolução, pois é algo que se destaca ao usuário em uma aplicação de RV, a possibilidade de ler as coisas e enxergar os detalhes. Além disso, deve se considerar o custo computacional dos *scripts* que se mantém executando em *background*, calculando a física, posição e comportamento dos eventos na cena até centenas de vezes por segundo. Então para mitigar esse custo a qualidade gráfica do Unity foi definida para a menor fidelidade possível no deploy do APK (Figura 6), assim atingindo o objetivo de uma taxa de quadros estável de no mínimo 72 quadros com pouca flutuação.

Figura 6 - Tela de definições da qualidade no build dentro do Unity

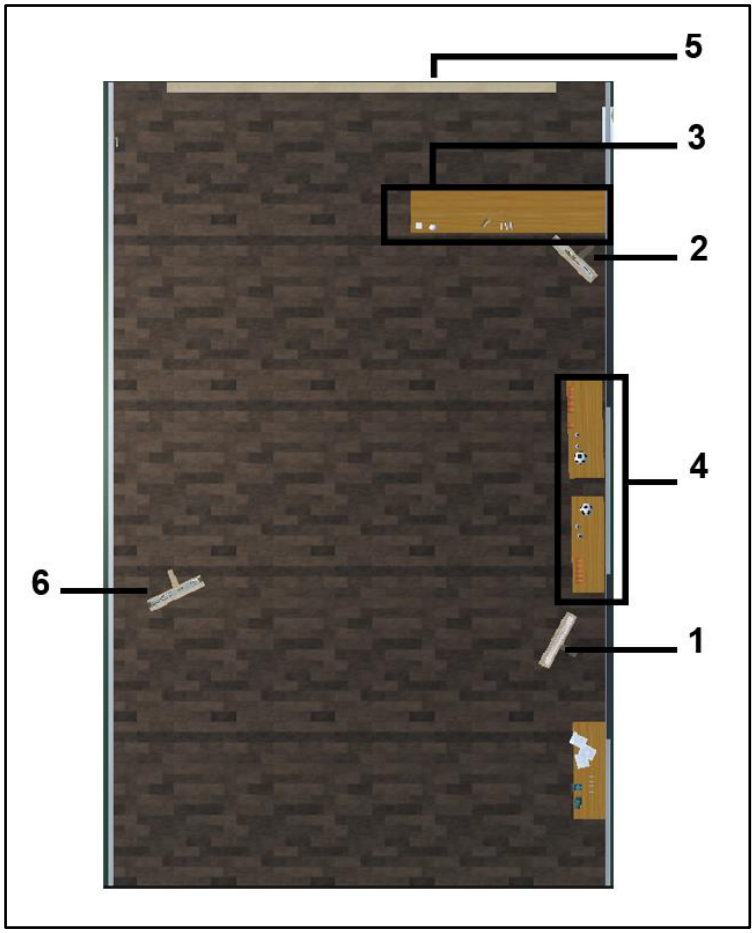


Fonte: elaborado pelo autor.

4 RESULTADOS

Nesse capítulo serão apresentados os testes de uso realizados, assim como a forma que os mesmos foram realizados, as considerações e seus resultados. O teste foi ministrado em 3 cenários diferentes de usuários, no `cenário A` o teste foi realizado dentro da instituição de ensino com professores da área da computação e uma professora de química. Já dentro do `cenário B` o teste foi ministrado sem supervisão em um ambiente residencial por um usuário desenvolvedor com maior afinidade com a tecnologia. Por fim, o `cenário C` foi um teste supervisionado também dentro de um ambiente residencial onde a usuária já possui uma certa afinidade com a tecnologia. Os testes foram conduzidos utilizando do HMD Oculus Quest 2 com um *build* móvel da aplicação rodando no próprio System-on-a-Chip (SOC) do dispositivo sem a necessidade de fios. A planta baixa do ambiente virtual desenvolvido pode ser observada na Figura 7 seguido pelo Quadro 11 onde é descrito a legenda dos pontos de referência da figura.

Figura 7 - Planta baixa da sala de aula virtual



Fonte: elaborado pelo autor.

Quadro 11 - Legenda da planta baixa da sala de aula virtual

Identificador	Descrição
1	Cavalete contendo o exercício 1
2	Cavalete contendo o exercício 2
3	Mesa com os canetões, apagador, formas geométricas e o exercício 3
4	Mesa com objetos manuseáveis
5	Quadro Negro
6	Cavalete contendo as instruções de uso da aplicação

Fonte: elaborado pelo autor.

4.1 CENÁRIO A

Esse teste foi realizado dentro da instituição da FURB - Universidade Regional de Blumenau, onde uma sala foi reservada para a atividade que levou 1 hora e 30 minutos. Dentro desse intervalo cada usuário teve apenas cerca de 15 minutos para testar a aplicação. Nessa reunião um professor da área de computação junto de uma professora da área de química testou a utilização da aplicação. Compareceram também, outros professores que acompanharam os testes.

Antes de iniciar os testes foi delimitado uma área de 3x2 metros como a área segura do sistema de guardião do dispositivo, onde os usuários poderiam se movimentar fisicamente dentro desse espaço, e teriam um alerta caso ultrapassassem esse limite evitando que eles colidam com pessoas e objetos. Também foi configurado o espelhamento do dispositivo, para que tudo que estivesse sendo visualizado pelo usuário fosse também projetado para um computador conectado na mesma rede local, que então projetava a imagem em um retroprojetor para melhor visualização de todos os presentes (Figura 8).

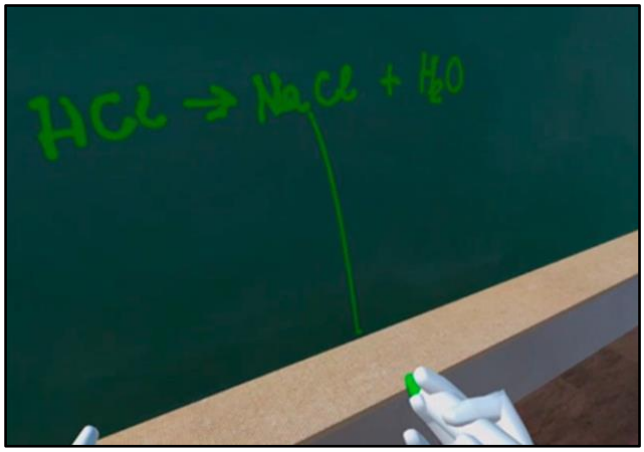
Figura 8 - Professor de computação testando a aplicação

Fonte: elaborado pelo autor.

O primeiro usuário a realizar os testes foi o professor da área de computação. Ao entrar no ambiente virtual ele se mostrou surpreso com a fidelidade da imersão propiciada pelo equipamento e a aplicação. O professor então foi instruído a manipular os controles para conseguir se locomover dentro do ambiente, e então foi dito para o mesmo se locomover até os cavaletes dentro do ambiente virtual que contém as informações de como realizar os exercícios propostos para obter uma noção do uso da aplicação. Daí ele se aproximou do quadro de exercício 1 (Figura 7 – 1), onde foi instruído a tentar pegar blocos com as mãos virtuais e então posicioná-los de maneira a montar uma pirâmide de blocos (Figura 7 – 4). A tarefa foi realizada sem muita dificuldade pelo usuário. Após isso o usuário foi até o segundo cavalete (Figura 7 – 2) onde se localizava o exercício 2, que requisitava para o usuário que desenhasse ou escrevesse algo utilizando os canetões (Figura 7 – 3) no quadro negro (Figura 7 – 5). O usuário não teve muita dificuldade nesse exercício que logo foi concluído. Para o último exercício 3 (Figura 7 – 3) o usuário teria que utilizar as formas dispostas sobre a mesa (Figura 7 – 3), para conseguir desenhar as respectivas formas no quadro negro (Figura 7 – 5). Nessa tarefa foi observada uma maior dificuldade devido ao maior nível de interação necessária com os botões do controle do dispositivo. Após realizar os exercícios o usuário elogiou a imersão e comentou sobre a dificuldade de utilizar os controles mas que isso seria uma questão de se acostumar. Também sugeriu que os objetos poderiam ficar na mão sem ter que manter um botão pressionado para que o mesmo se mantenha na mão. Por fim também relatou ter tido uma leve vertigem devido à locomoção virtual.

O segundo usuário do foi uma professora da área de química, que já estava observando o primeiro usuário realizar as tarefas através do espelhamento de tela do dispositivo. Ao colocar o dispositivo e entrar no ambiente a mesma mostrou uma maior dificuldade em relacionar os botões do controle com as ações no mundo virtual, assim como para se locomover, fazendo com que ela muitas vezes optasse por andar fisicamente até os limites do guardião em vez de utilizar da locomoção virtual. A usuária acabou pulando o exercício 1 (Figura 7 – 1) e foi direto para o exercício 2 (Figura 7 – 2), onde já teve uma dificuldade para saber a distância de sua mão virtual em relação aos objetos para pega-los, possivelmente devido a algum ajuste na distância intra-pupilar dos olhos que não havia sido calibrado antes da usuária utilizar o dispositivo. Após conseguir pegar um canetão (Figura 7 – 3) e se locomover até o quadro negro (Figura 7 – 5) questionou como faria para desenhar. O que deveria ser realizado de maneira intuitiva apenas aproximando o canetão no quadro, mas talvez devido a confusão no uso dos controles acabou pressupondo que seria necessário apertar algo a mais para realizar essa ação. Porém após ser instruída sobre como realizar o desenho, a mesma conseguiu desenhar no quadro negro (Figura 7 – 5), e então questionou como faria para parar de desenhar, onde então foi dito para apenas afastar o canetão do quadro negro. Ela desenhou uma equação de química básica (Figura 9), porém acabou largando o botão que mantém o objeto na mão fazendo com que ele caísse algumas vezes involuntariamente. Após completar o desenho ela se locomoveu até a mesa do exercício 3 (Figura 7 – 3), pegou a forma do cubo para desenhar no quadro, e então aproximou a forma no quadro (Figura 7 – 5) e apertou o botão de gatilho do controle para desenhar, mas com dificuldade e sem utilizar dos recursos que a ferramenta dispõe, como redimensionamento e troca de cores da forma. A usuária não comentou especificamente sobre a sua experiência, porém foi perceptível que seria necessário um maior tempo de uso com o dispositivo para conseguir utilizar do equipamento como um todo de forma mais natural.

Figura 9 - Equação química desenhada pela professora durante os testes



Fonte: elaborado pelo autor.

Em ambos os usuários é possível verificar algumas dificuldades em comuns. A mais notável é a de locomoção dentro do ambiente virtual, que poderia ser resolvido utilizando apenas da locomoção física, dentro de um espaço físico maior ou de um espaço virtual menor de forma que os limites virtuais e físicos coincidam, tornando nula a possibilidade de *motion sickness*. Outra maneira mais prática de evitar o sintoma, seria implementando a locomoção virtual com um sistema de teletransporte, onde o usuário poderia designar no chão onde que ele gostaria de se posicionar virtualmente e após uma transição que bloqueia o campo visual do usuário ele é virtualmente movimentado até o local designado, evitando assim que tenha qualquer tipo de *motion sickness*. Outra dificuldade foi na utilização dos controles do dispositivo, mesmo que a curva de aprendizado não seja grande, para quem nunca utilizou do dispositivo 15 minutos não é o suficiente para se adequar a todos os botões dos controles. A opção de poder apenas apertar uma vez o botão para pegar e manter na mão o objeto poderia ser uma opção customizável de acordo com a preferência do usuário. Por fim, a forma que foi implementada a ferramenta de desenho das formas no quadro também gerou certas dificuldades, mas que em sua grande maioria se deu pelo motivo anterior, da falta de adequação ao uso dos controles, porém uma maneira mais intuitiva de utilizar essa ferramenta pode ser uma alternativa para melhorar a experiência dos usuários. Mais imagens dos testes realizados com o `cenário A` podem ser observados no Apêndice A do artigo.

4.2 CENÁRIO B

Nesse cenário o usuário que é um jovem estudante desenvolvedor de softwares, que possui conhecimento de computação e já tinha afinidade com o dispositivo Oculus Quest 2. O usuário testou o software com seu próprio dispositivo sem supervisão ou observação. Ele teve um pouco de dificuldade em utilizar as formas para desenhar no quadro negro, mas teve grande facilidade para executar todos os 3 exercícios propostos. Seu parecer foi positivo, ressaltando a imersão e o potencial de utilizar essa tecnologia como uma alternativa ao ensino comum no futuro próximo.

Este cenário apresentou resultados similares ao `cenário C` devido ao fato de ambos os cenários terem sido aplicados em usuários que já possuíam uma maior afinidade com o dispositivo. Mas devido a falta de supervisão durante o teste nesse cenário, não foi possível obter observações mais detalhadas sobre o uso da aplicação pelo usuário.

4.3 CENÁRIO C

O teste do `cenário C` foi realizado com uma usuária jovem que não possui nenhuma relação com a área de computação ou educação, e que possuía um pouco de afinidade com o Oculus Quest 2. Para esta usuária foi preparado um guardião em seu ambiente físico de 2x2 metros, e a usuária foi instruída a se locomover até o cavalete do exercício 1 `(`Figura 7 `- 1)` e realizar as instruções contidas nele. A usuária apresentou pouca dificuldade para manusear os objetos virtuais, mas quando tentava posicionar objetos e quando largava os, acabava mantendo a mão no mesmo local involuntariamente, isso fazia com que a mão virtual reaparecesse e colidisse dentro do objeto fazendo com que o mesmo se deslocasse. Porém a usuária conseguiu completar o exercício, então se locomoveu até o segundo exercício `(`Figura 7 `- 2)`, onde conseguiu de maneira intuitiva pegar um canetão `(`Figura 7 `- 3)` e fazer um desenho no quadro `(`Figura 7 `- 5)` sem qualquer dificuldade, completando assim o exercício. Por fim, no último exercício 3 `(`Figura 7 `- 3)`, ao utilizar das formas para desenhar no quadro `(`Figura 7 `- 5)`, apresentou certa dificuldade na compreensão dos controles para conseguir manusear e desenhar a forma no quadro. Porém após alguns minutos de tentativa e erro conseguiu começar a manusear a ferramenta sem dificuldades e assim completando o exercício. A usuária elogiou a forma intuitiva que podia desenhar no quadro e manipular e montar os blocos assim como os outros objetos disponíveis. A usuária relatou ficar com um pouco de *motion sickness* devido ao desconforto do dispositivo na cabeça e com a movimentação virtual que julgou como rápida demais.

4.4 CONSIDERAÇÕES

Alguns resultados foram obtidos com os testes da aplicação mesmo com um número pequeno de usuários. É perceptível que enquanto alguns usuários tiveram pouca dificuldade com o uso do dispositivo, outros já tiveram um nível maior de dificuldade. É possível relacionar essa dificuldade encontrada com a afinidade que cada um desses usuários possuía com tecnologia em geral, quanto maior a afinidade, menor foram as dificuldades encontradas. Muitas das dificuldades encontradas pelos usuários também se tratavam de preferências, é importante considerar acessibilidade em uma aplicação de RV, implementado o maior número possível de alternativas para a locomoção virtual, manuseio de objetos, restrição de visão entre outros a fim de mitigar o efeito negativo do *motion sickness* assim como para garantir estar oferecendo uma solução ideal para cada preferência de cada usuário.

Como colocado por Tabrizi (2008) em seu artigo, a gamificação de uma atividade costuma ser vista de forma positiva, e isso é perceptível pela própria reação obtida dos usuários ao poderem manipular objetos com simulação de física e desenhar no quadro dentro do ambiente virtual. Como colocado pelos autores Chatzizisis *et al.* (2019) a RV permite que possamos estar simulando situações, nesse caso, lecionando uma aula, dentro do conforto da sua própria casa com as ferramentas que só estariam disponíveis nesse ambiente físico. Finalmente, como Dong (2016) propõe, as possibilidades de lecionar dentro de um ambiente virtual são de potencial enorme, a maneira que é possível representar conteúdos de maneira mais concreta com representações tridimensionais abre muitas possibilidades para aprimorar o conhecimento dos alunos durante as aulas. Esse potencial da tecnologia foi observado por alguns usuários durante os testes.

5 CONCLUSÕES

A aplicação desenvolvida atendeu todos os requisitos e os resultados obtidos pelos testes foram favoráveis ao uso da tecnologia de RV na área da educação, assim como também foi destacado pelos diversos autores ao longo do artigo. Esta tecnologia permite que mesmo dentro de suas próprias casas, ainda seja possível estar em um ambiente completamente diferente exercendo sua profissão sem as restrições visuais do mundo físico. Foi perceptível que os usuários submetidos ao teste se sentiram completamente imersos durante o uso da aplicação. Conseguiram realizar os exercícios propostos dentro da aplicação, mostrando que seria possível utilizar da mesma para ministrar uma aula *online*. A tecnologia que encontramos nos dispositivos HMD de hoje ainda não são perfeitas, questões como ergonomia do dispositivo, densidade de pixels, campo de visão, e interfaces para interação com maior imersão e *haptic feedback* para as mãos assim como novas maneiras de se locomover dentro do mundo virtual, são possíveis melhorias para as gerações futuras da tecnologia no mercado consumidor, agregando ainda mais para a experiência e viabilidade do seu uso na educação.

Uma possível extensão desse trabalho seria começar a abranger dentro do ambiente virtual, outros ambientes e objetos mais específicos para cada tipo de disciplina como colocado pelo autor Dong (2016), maximizando assim o potencial do uso da tecnologia em colocar o usuário virtualmente em qualquer local com qualquer objeto implementado. Outra possibilidade seria implementar diferentes tipos de locomoção e acessibilidade, em específico, implementar o *tracking* das mãos disponível de forma nativa com o dispositivo Oculus Quest 2. O que poderia substituir ou ser uma forma alternativa aos controles comuns, onde o uso apenas do *tracking* reduziria o número de formas de interagir com o mundo virtual mas aumentaria a intuitividade reduzindo a curva de aprendizado. Com esses devidos ajustes, seria interessante reaplicar os testes aos usuários finais para qualificar a experiência deles com essas novas funcionalidades, e então comparar as diferenças encontradas nas impressões com as encontradas nesse artigo. Assim como validar o uso da aplicação em um caso de uso de uma aula *online*, onde o professor usaria a aplicação utilizando de alguma plataforma de streaming para dar uma aula para alunos e então avaliando a opinião deles.

REFERÊNCIAS

42GEARS. **The History of VR: 5 Eras Of Evolving A New Reality**, 2019. Disponível em: https://www.42gears.com/blog/the-history-of-vr-5-eras-of-evolving-a-new-reality/. Acesso em: 19 jun 2021

ROBERTSON, Adi. The Verge (ed.). **Oculus Quest 2 Review**: better, cheaper vr. BETTER, CHEAPER VR. 2020. Disponível em: https://www.theverge.com/21437674/oculus-quest-2-review-features-photos. Acesso em: 25 jun. 2021.

BASTRIKIN, Andrej. **Online Education Statistics**. Educationdata, 2020. Disponível em https://educationdata.org/online-education-statistics. 2020.Acesso em: 05 set, 2020.

DAVID HEANEY. Uploadvr. **How VR Positional Tracking Systems Work**. 2019. Disponível em: https://uploadvr.com/how-vr-tracking-works/. Acesso em: 25 jun. 2021.

DONG, Xisong. An overall solution of Virtual Reality Classroom. In: IEEE INTERNATIONAL CONFERENCE ON SERVICE OPERATIONS AND LOGISTICS, AND INFORMATICS, 11., 2016, Beijing. **International Conference on Service Operations and Logistics, and Informatics (SOLI)**. Beijing: Ieee, 2016. p. 120-124. Disponível em: https://ieeexplore.ieee.org/document/7551672. Acesso em: 20 jun. 2021.

FACEBOOK TECHNOLOGIES. **Oculus Device Specifications**. 2021a. Disponível em: https://developer.oculus.com/learn/oculus-device-specs/. Acesso em: 25 jun. 2021.

FACEBOOK TECHNOLOGIES. **Oculus Integration SDK**. 2021b. Disponível em: https://developer.oculus.com/downloads/package/unity-integration/. Acesso em: 25 jun. 2021.

FREITAS, Olga. **Equipamentos e materiais didáticos**. Brasília: Universidade de Brasília, 2009. 132 p. Disponível em: http://portal.mec.gov.br/index.php?option=com_docman&view=download&alias=614- equipamentos-e-materiais-didaticos&Itemid=30192. Acesso em: 19 jun. 2021.

FLANAGAN, Graham. **The incredible story of the 'Virtual Boy' Nintendo's VR headset from 1995 that failed spectacularly**. 2018. Disponível em: https://www.businessinsider.com/nintendo-virtual-boy-reality-3dvideo-games-super-mario-2018-3. Acesso em: 19 jun. 2021.

HODGES, Charles et al. **The Difference Between Emergency Remote Teaching and Online Learning**. 2020. Disponível em: https://er.educause.edu/articles/2020/3/the-difference-between-emergency-remote-teaching-and-online-learning. Acesso em: 18 jun. 2021.

CHATZIZISIS, Ioannis *et al*. Developing a Virtual Reality Educational Environment for Traffic Education. In: ANNUAL INTERNATIONAL CONFERENCE OF EDUCATION, RESEARCH AND INNOVATION, 12., 2019, Itália. **ICERI2019 Proceedings.** [S.L.]: Iated, 2019. p. 240-248. Disponível em: https://www.researchgate.net/publication/338106916_DEVELOPING_A_VIRTUAL_REALITY_EDUCATIONAL_ENVIRONMENT_FOR_TRAFFIC_EDUCATION. Acesso em: 20 jun. 2021.

LEE, Chris. Real Learning in a virtual classroom is difficult. **Arstechnica**, 2020. Disponível em https://arstechnica.com/staff/2020/03/a-crash-course-in-virtual-teaching-real-learning-achieved. Acesso em: 18 jun. 2021

SANTOS, Maria Vanessa. **O Professor na era da lousa digital**. Disponível em: https://educador.brasilescola.uol.com.br/trabalho-docente/o-professor-na-era-lousa-digital.htm. 2014. Acesso em: 26 jun. 2021.

STATISTA. **Augmented reality (AR) and virtual reality (VR) headset shipments worldwide from 2020 to 2025**, 2020. Disponível em: www.statista.com/statistics/653390/worldwide-virtual-and-augmented-reality-headset-shipments/. Acesso em: 19 jun. 2021.

M.H.N., Tabrizi. Agent and Virtual Reality-based Course Delivery System. In: APPLIED COMPUTING CONFERENCE, 8., 2008, Algarve. **Proceedings of the IADIS International Conference on Applied Computing.** Algarve: Iadis, 2008. p. 73-77. Disponível em: https://www.semanticscholar.org/paper/Agent-and-virtual-reality-based-course-delivery-Tabrizi/82406bbc7b549c1bb7adc88f00526c4661352d55?p2df. Acesso em: 05 set. 2020.

TAYLOR CLARK. Smithsonian Magazine (ed.). **How Palmer Luckey Created Oculus Rift**. 2014. Disponível em: https://www.smithsonianmag.com/innovation/how-palmer-luckey-created-oculus-rift-180953049/. Acesso em: 26 maio 2021.

TCHA-TOKEY, Katy et al. A questionnaire to measure the user experience in immersive virtual environments. In: INTERNATIONAL VIRTUAL REALITY CONFERENCE, 3., 2016, Laval. **Proceedings of the 2016 Virtual Reality International Conference**. New York: Association For Computing Machinery, 2016. p. 1-5. Disponível em: https://www.researchgate.net/publication/301558848_A_Questionnaire_to_Measure_the_User_Experience_in_Immersive_Virtual_Environments. Acesso em: 9 out. 2020. The History of VR: 5 Eras Of Evolving A New Reality. 42GE

TERÄS, Marko; SUORANTA, Juha; TERÄS, Hanna; CURCHER, Mark. Post-Covid-19 Education and Education Technology ‘Solutionism’: a seller⁚s market. **Postdigital Science And Education**, [S.L.], v. 2, n. 3, p. 863-878, 13 jul. 2020. Springer Science and Business Media LLC. http://dx.doi.org/10.1007/s42438-020-00164-x. Disponível em: https://link.springer.com/article/10.1007%2Fs42438-020-00164-x. Acesso em: 18 jun. 2021.

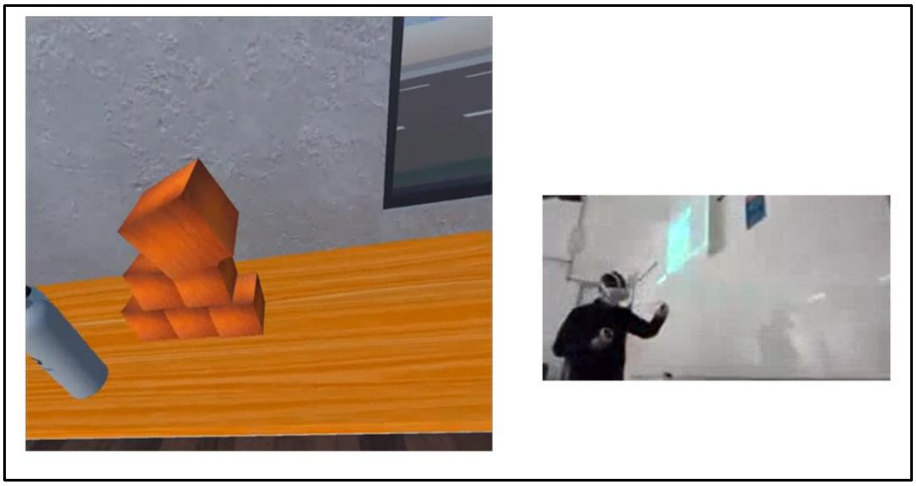
TORI, Romero; KIRNER, Claudio; SISCOUTTO, Robson. **Fundamentos e Tecnologia de Realidade Virtual e Aumentada**. Belém: Sbc, 2006. 318 p. Disponível em: https://webserver2.tecgraf.puc-rio.br/~abraposo/pubs/livro_pre_SVR2006/LivroSVR2006-cap20.pdf. Acesso em: 09 out. 2020.

UNITY TECHNOLOGIES. **Unity Manual**: xr. XR. 2021. Disponível em: https://docs.unity3d.com/2021.2/Documentation/Manual/XR.html. Acesso em: 25 jun. 2021.

APÊNDICE A – IMAGENS DOS TESTES

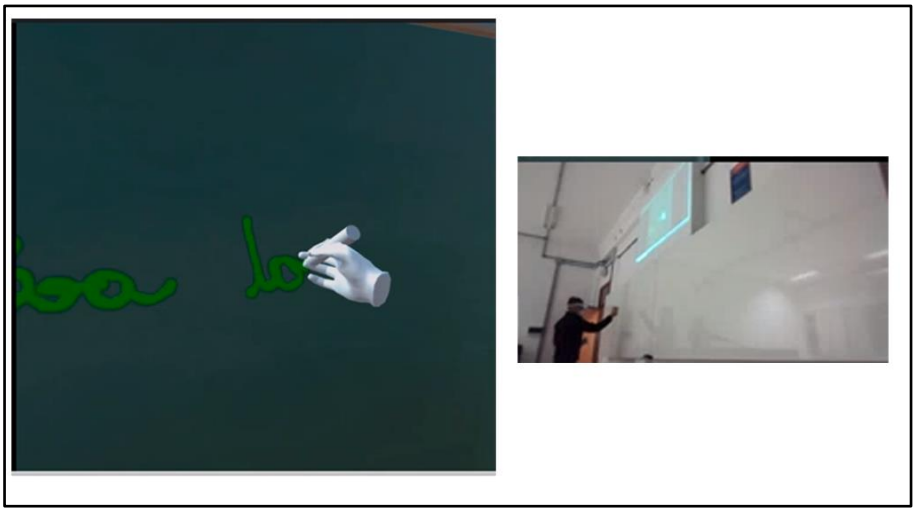
Este apêndice mostra algumas imagens (Figura 10, Figura 11 e Figura 12) que foram retiradas enquanto o aplicativo era testado.

Figura 10 - Professor de computação realizando o exercício 1



Fonte: elaborado pelo autor.

Figura 11 - Professor de computação realizando o exercício 2



Fonte: elaborado pelo autor.

Figura 12 - Professor de computação observando o ambiente virtual

Fonte: elaborado pelo autor.