

UNIVERSIDADE ESTADUAL PAULISTA "JÚLIO DE MESQUITA FILHO"
FACULDADE DE CIÊNCIAS - CAMPUS BAURU
DEPARTAMENTO DE COMPUTAÇÃO
BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO

LUCCA VIEIRA BATISTÃO

**PERCEPÇÃO E INTERAÇÃO NO ESPAÇO PERIPESSOAL DE
REALIDADE AUMENTADA UTILIZANDO HMDS VIDEO
SEE-THROUGH BASEADOS EM SMARTPHONE: UM ESTUDO
DE CASO**

BAURU
Dezembro/2020

LUCCA VIEIRA BATISTÃO

**PERCEPÇÃO E INTERAÇÃO NO ESPAÇO PERIPESSOAL DE
REALIDADE AUMENTADA UTILIZANDO HMDS VIDEO
SEE-THROUGH BASEADOS EM SMARTPHONE: UM ESTUDO
DE CASO**

Trabalho de Conclusão de Curso do Curso de Bacharelado em Ciência da Computação da Universidade Estadual Paulista “Júlio de Mesquita Filho”, Faculdade de Ciências, Campus Bauru.

Orientador: Prof. Associado Antonio Carlos Sementille

BAURU
Dezembro/2020

Lucca Vieira Batistão Percepção e Interação no Espaço Peripessoal de Realidade Aumentada utilizando HMDs *Video See-Through* baseados em *Smartphone*: Um Estudo de Caso/ Lucca Vieira Batistão. – Bauru, Dezembro/2020- 85 p. : il. (algumas color.) ; 30 cm.
Orientador: Prof. Associado Antonio Carlos Sementille
Trabalho de Conclusão de Curso – Universidade Estadual Paulista “Júlio de Mesquita Filho”
Faculdade de Ciências
Bacharelado em Ciência da Computação, Dezembro/2020.

Lucca Vieira Batistão

**Percepção e Interação no Espaço Peripessoal de
Realidade Aumentada utilizando HMDs Video
See-Through baseados em *Smartphone*: Um Estudo de
Caso**

Trabalho de Conclusão de Curso do Curso de Bacharelado em Ciência da Computação da Universidade Estadual Paulista "Júlio de Mesquita Filho", Faculdade de Ciências, Campus Bauru.

Banca Examinadora

**Prof. Associado Antonio Carlos
Sementille**
Orientador
Departamento de Computação
Faculdade de Ciências
Universidade Estadual Paulista "Júlio de
Mesquita Filho"

**Profa. Dra. Simone das Graças
Domingues Prado**
Departamento de Computação
Faculdade de Ciências
Universidade Estadual Paulista "Júlio de
Mesquita Filho"

Prof. Associado Aparecido Nilceu Marana
Departamento de Computação
Faculdade de Ciências
Universidade Estadual Paulista "Júlio de
Mesquita Filho"

Bauru, 15 de Dezembro de 2020.

Dedico esse trabalho à minha mãe, minha primeira e eterna professora.

Agradecimentos

Primeiramente, agradeço meus pais pelo amor, apoio e sacrifícios durante todos esses anos.

Agradeço, também, os meus irmãos pelo incentivo e a ajuda, os quais sempre foram abundantes.

Agradeço, ainda, os meus amigos, por tudo que passamos juntos e por todo o conhecimento dividido.

E por fim, agradeço todos os professores que fizeram parte da minha formação até hoje, em especial ao meu orientador Antonio Carlos Sementille, por toda paciência, leveza e por ter despertado meu interesse na área e na pesquisa.

*"To see a World in a Grain of Sand
And a Heaven in a Wild Flower,
Hold Infinity in the palm of your hand
And Eternity in an hour."*
William Blake

Resumo

Realidade Aumentada é a tecnologia que está relacionada com o enriquecimento do mundo real através da combinação de objetos sintéticos criados por computador, como: texto, imagens e objetos 3D, em tempo real. Por fornecer uma integração simbiótica com os ambientes reais, a Realidade Aumentada possui grande potencial para diversas áreas de aplicação, fornecendo benefícios a qualquer tarefa que se favoreceria por ter acesso à informação. Contudo, ainda existem muitas dificuldades que hão de ser superadas, principalmente no que diz respeito a uma maior difusão da tecnologia. Isso se dá pelo alto custo dos *Head-Mounted Displays*, que são essenciais no uso de Realidade Aumentada. Apesar disso, com o aumento do poder de processamento dos dispositivos móveis, surgem os *Head-Mounted Displays* baseados em *smartphones*, cujo custo é reduzido em relação aos convencionais, porém poucos estudos foram conduzidos usando esse tipo de dispositivo. Portanto, era do interesse desse projeto determinar como a interação e percepção do usuário é afetada pelo uso de *Head-Mounted Displays Video See-Through* baseados em *smartphones* no espaço peripessoal. Os experimentos conduzidos mostraram que apesar da percepção proporcionada por esse tipo de *display* não ser ideal, eles podem ser utilizados em aplicações de RA nas quais a precisão não é um requisito fundamental.

Palavras-chave: Realidade Aumentada; Percepção e Interação; Espaço Peripessoal; *Head-Mounted Display*.

Abstract

Augmented Reality is the technology related to the enrichment of the real world through the combination of synthetic objects created by a computer, such as text, images, and 3D objects, in real-time. By providing a symbiotic integration with real environments, the Augmented Reality has great potential for several application areas, providing benefits to any task that would be favored by having access to information. However, there are still many difficulties to overcome, especially regarding a greater diffusion of the technology. This is due to the high cost of Head-Mounted Displays, which are essential in using Augmented Reality. Nevertheless, with the increase of computational power on mobile devices, appears the smartphone-based Head-Mounted Displays, whose cost is reduced regarding the conventional ones, but few studies were conducted using this type of device. Therefore, it was in the interest of this project to determine how user interaction and perception is affected by the use of smartphone-based Head-Mounted Video See-Through Displays in the peripersonal space. The experiments conducted showed that the perception when making use of this type of display is not ideal, but still has potential for applications where accuracy is not fundamental.

Keywords: Augmented Reality; Perception and Interaction; Peripersonal Space; Head-Mounted Displays.

Lista de figuras

Figura 1 – Configurações típicas de displays (a) <i>optical see-through</i> e (b) <i>video see-through</i> .	16
Figura 2 – HMDs de RA baseados em <i>smartphones</i> . (a) VR BOX 2. (b) Haori AR Headset.	18
Figura 3 – Primeiros HMDs de RA. (a) Primeiro protótipo por Ivan Sutherland e (b) "Super Cabine" da Força Área dos EUA	22
Figura 4 – Livro sendo colorido e modelo 3D correspondente.	23
Figura 5 – Estrutura do olho humano.	26
Figura 6 – Formação de imagem no olho humano.	26
Figura 7 – Taxonomia das pistas de profundidade	27
Figura 8 – Demonstração da vergência: Ângulo θ maior se menor distância do objeto ao observador (menor z).	28
Figura 9 – Projeto para auxílio de realização de cirurgias Buco-Maxilo-Facial. (a) Cirurgião usando o HMD. (b) Rastreamento do dente (c) Visão aumentada.	30
Figura 10 – Aplicação com projetor da GM. Os círculos verdes estão sendo projetados no chassis.	31
Figura 11 – Aplicação de visão indireta. Pode ser visto o monitor no fundo, onde a cena aumentada por ser vista.	32
Figura 12 – Ilustração do eye-relief.	32
Figura 13 – Exemplos de arquiteturas <i>non-pupil Forming</i> : (a) Refrativa (b) Catadióptrico (c) Prisma de forma livre.	33
Figura 14 – Abordagens do rastreamento usando sensores infravermelhos. Em cima: de fora para dentro. Embaixo: de dentro para fora.	40
Figura 15 – Estimação da posição e orientação fazendo uso de um marcador fiducial. Sendo (x_c, y_c, z_c) o sistema de coordenadas da câmera, (x_m, y_m, z_m) sendo o do marcador e T_{cm} a matriz de transformação para a passagem de um pro outro.	41
Figura 16 – Imagem RGB e sua reconstrução 3D pelo Kinect.	41
Figura 17 – <i>Leap Motion</i> internamente.	42
Figura 18 – Volume de interação do <i>Leap Motion</i> .	43
Figura 19 – Representação do espaço peripessoal.	44
Figura 20 – Ambientes virtuais de teste. (a) Sala Escura. (b) Sala com <i>wireframe</i> . (c) Sala rica em detalhes.	45
Figura 21 – Aparato usado no trabalho de Naceri, Chellali e Hoinville (2011).	46
Figura 22 – <i>Feedback</i> da mão virtual.	46
Figura 23 – Disposição dos componentes do projeto de Ballestin, Solari e Chessa (2018).	47
Figura 24 – Mesa Virtual de <i>Shuffleboard</i> .	48
Figura 25 – Configuração em hardware de Batmaz et al. (2019). À esquerda HMD de RA. À direita HMD de RV.	48

Figura 26 – Cenas do projeto Batmaz et al. (2019). À esquerda cena aumentada de RA. À direita cena virtual de RV.	49
Figura 27 – Simulação dos HMDs de RA.	49
Figura 28 – Configuração física de Jiménez (2014).	50
Figura 29 – Exemplo de experimento de Jiménez (2014).	50
Figura 30 – Gesto de pinça agarrando um objeto.	52
Figura 31 – Estrutura hierárquica de software.	54
Figura 32 – Painel de Hierarquia.	55
Figura 33 – Exemplo de um grafo de Cena.	56
Figura 34 – Inspetor mostrando informações do <i>GameObject</i> da Câmera virtual.	56
Figura 35 – Exemplos de marcadores utilizados pelo SDK da Vuforia. (a) Marcador Imagem. (b) Marcador modelo 3D. (c) Mapeamento de ambientes 3D.	57
Figura 36 – <i>Pipeline</i> do <i>Vuforia</i> .	58
Figura 37 – Imagens do VRBOX 2. (a) Visão lateral. (b) Visão traseira.	59
Figura 38 – Arquitetura interna do VRBOX 2.	59
Figura 39 – Arquitetura Lógica do sistema dividida em módulos.	60
Figura 40 – Esfera verde que deveria estar na palma da mão, mas aparece deslocada.	61
Figura 41 – <i>Leap Motion</i> no centro do marcador fiducial.	62
Figura 42 – Arquitetura Física do sistema.	62
Figura 43 – Grafo de Cena do <i>LeapServer</i> .	63
Figura 44 – Fluxograma do script <i>ServerVuforia2</i> .	64
Figura 45 – Grafo de Cena do Menu Principal.	65
Figura 46 – Grafo de Cena do Ambiente de Interação.	66
Figura 47 – Tela onde se coloca o centro do mundo virtual como o primeiro marcador encontrado pelo <i>Vuforia</i> .	67
Figura 48 – Tela onde se adiciona sensores iniciais.	67
Figura 49 – Fluxograma do <i>ClientVuforia2</i> .	68
Figura 50 – Fluxograma do <i>InteractionCube</i> .	69
Figura 51 – Regiões em que os experimentos foram realizados.	71
Figura 52 – Disposição dos objetos e alvos no primeiro cenário de teste.	71
Figura 53 – Mão do usuário realizando gesto de pinça.	72
Figura 54 – Movimento dos dedos sendo obstruído pela própria mão.	72
Figura 55 – Disposição dos objetos e alvos no segundo cenário de teste.	73
Figura 56 – Disposição dos objetos e alvos no terceiro cenário de teste.	73
Figura 57 – Gráficos dos resultados do primeiro caso de teste.	76
Figura 58 – Gráficos dos resultados do segundo caso de teste.	77
Figura 59 – Gráficos dos resultados do terceiro caso de teste.	79

Lista de tabelas

Tabela 1 – Características das tecnologias de rastreamento baseado em sensores	39
Tabela 2 – Características das tecnologias de rastreamento óptico.	42
Tabela 3 – Medidas utilizadas nos testes.	74

Lista de abreviaturas e siglas

BRIEF	<i>Binary Robust Independent Elementary Feature</i>
CAD	<i>Computer Aided Design</i>
CRT	<i>Cathode-ray tube</i>
CT	<i>Computerized Tomography</i>
DOF	<i>Degrees Of Freedom</i>
FOV	<i>Field Of View</i>
GM	<i>General Motors</i>
GPS	<i>Global Positioning System</i>
HMD	<i>Head-Mounted Display</i>
HMPD	<i>Head-Mounted Projective Display</i>
HOE	<i>Holographic Optical Element</i>
IPD	<i>Interpupillary Distance</i>
OST	<i>Optical See-Through</i>
PPD	<i>Pixels Per Degree</i>
RA	Realidade Aumentada
RGB-D	<i>Red Green Blue Depth</i>
RV	Realidade Virtual
SIFT	<i>Scale Invariant Feature Transform</i>
SURF	<i>Speeded Up Robust Feature</i>
TOF	<i>Time Of Flight</i>
UI	<i>User Interface</i>
UX	<i>User eXperience</i>
VRD	<i>Virtual Retinal Display</i>
VST	<i>Video See-Through</i>

Sumário

1	INTRODUÇÃO	16
1.1	Problema	18
1.2	Justificativa	18
1.3	Objetivos	19
1.3.1	Objetivo Geral	19
1.3.2	Objetivos Específicos	19
1.4	Contribuição	19
1.5	Organização do trabalho	20
2	FUNDAMENTAÇÃO TEÓRICA	21
2.1	Realidade Aumentada	21
2.1.1	Histórico	21
2.1.2	Definição de Realidade Aumentada	21
2.1.3	Desafios	23
2.1.3.1	Limitações Tecnológicas	23
2.1.3.2	Falta de Padronização	24
2.1.3.3	Erro de Rastreamento	24
2.1.3.4	Aceitação	25
2.1.3.5	Percepção Visual	25
2.2	Tecnologias de <i>Displays</i>	29
2.2.1	Taxonomia	29
2.2.1.1	Visão Direta	30
2.2.1.2	Visão Indireta	30
2.2.2	Arquitetura dos HMDs	31
2.2.3	Atributos dos HMDs	34
2.2.3.1	Resolução	34
2.2.3.2	Campo de Visão	34
2.2.3.3	Oclusão	35
2.2.3.4	Profundidade de Campo	35
2.2.3.5	Latênciā	35
2.2.3.6	Paralaxe	35
2.2.3.7	Distorções	36
2.2.3.8	Consistência Pictorial	36
2.2.3.9	Multimodalidade	36
2.2.3.10	Sensação	36

2.3	Rastreamento 3D	36
2.3.1	Taxonomia	37
2.3.1.1	Baseado em Sensores	37
2.3.1.2	Baseado em Visão	39
2.3.2	<i>Leap Motion</i>	42
2.4	Espaço Peripessoal	43
3	TRABALHOS CORRELATOS	45
3.1	Método de Naceri, Chellali e Hoinville	45
3.2	Método de Ballestin, Solari e Chessa	46
3.3	Método de Ping, Liu e Weng	47
3.4	Método de Batmaz, Machuca, Pham e Stuerzlinger	48
3.5	Método de Plopski, Moser, Kiyokawa, Swan e Takemura	49
3.6	Método de Jiménez	50
4	MATERIAIS E MÉTODOS	51
4.1	Principais requisitos do projeto	51
4.1.1	Registro	51
4.1.2	Interação	51
4.1.3	Experimentos	52
4.2	Arquitetura Geral do Sistema	53
4.2.1	Materiais	53
4.2.1.1	<i>Unity</i>	54
4.2.1.2	<i>Vuforia</i>	55
4.2.1.3	HMD VST VR BOX 2	58
4.2.1.4	<i>Leap Motion</i>	59
4.2.2	Arquitetura Lógica	60
4.2.3	Arquitetura Física	61
4.3	Implementação do Sistema	62
4.3.1	Grafos de Cena	62
4.3.1.1	<i>Leap Server</i>	62
4.3.1.2	Menu Principal	64
4.3.1.3	<i>Leap Client</i>	65
5	EXPERIMENTOS E ANÁLISE DOS RESULTADOS	70
5.1	Experimentos	70
5.1.1	Caso de Teste 1	70
5.1.2	Caso de Teste 2	72
5.1.3	Caso de Teste 3	73
5.2	Análise dos resultados	74

5.2.1	Resultados do Caso de Teste 1	74
5.2.2	Resultados do Caso de Teste 2	75
5.2.3	Resultados do Caso de Teste 3	78
5.2.4	Discussão	78
6	CONCLUSÃO	81
6.1	Trabalhos Futuros	81
	REFERÊNCIAS	82

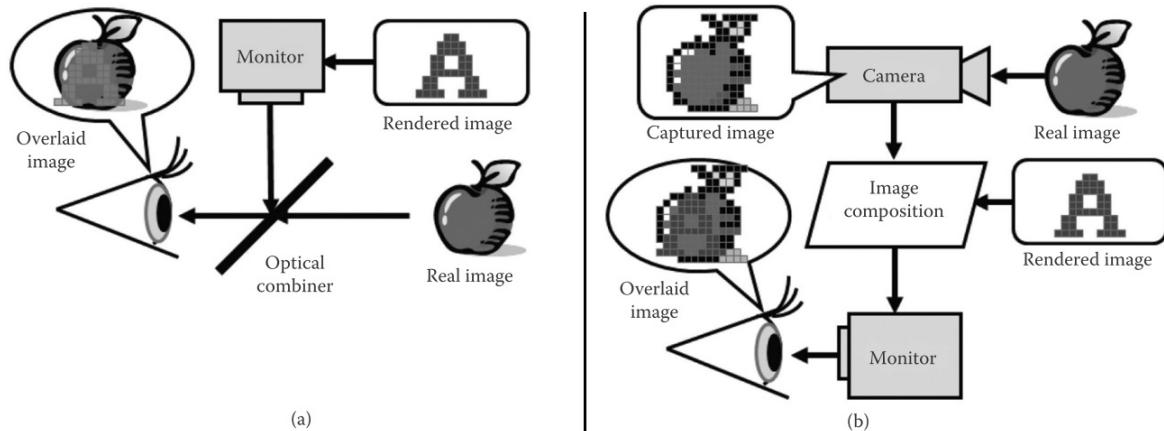
1 Introdução

As aplicações de Realidade Aumentada (RA) têm recebido uma crescente atenção de diferentes áreas de atuação, tais como a Indústria, Educação, Medicina, Entretenimento, entre outras. De acordo com [Azuma \(1997\)](#), o termo Realidade Aumentada refere-se a uma tecnologia que possui três características principais: i) combina conteúdo real e virtual (elementos 2D ou 3D); ii) executa interativamente, em tempo real; e iii) registra (alinha) objetos reais e virtuais de forma coerente no espaço 3D.

Além disso, a RA supera diversas limitações da Realidade Virtual (RV), visto que não necessita que todo o mundo virtual seja modelado, reduzindo, assim, a remodelagem de objetos reais complexos. Porém, a RA também possui diversos desafios científicos/tecnológicos a serem superados.

Voltando à definição de Azuma, para combinar imagens reais e virtuais a fim de que possam ser visualizadas simultaneamente, é necessário algum tipo de tecnologia de visualização (*display*). Em termos de hardware, existem dois tipos principais de tecnologias de visualização em RA: *Optical See-Through* (OST) e *Video See-Through* (VST). A Figura 1 mostra uma configuração típica para ambas as tecnologias ([KIYOKAWA, 2015](#)).

Figura 1 – Configurações típicas de displays (a) *optical see-through* e (b) *video see-through*.



Fonte: [Kiyokawa \(2015\)](#).

Conforme [Kiyokawa \(2015\)](#), os dispositivos OST e VST possuem as características descritas a seguir.

Em um display OST, as imagens reais e virtuais são combinadas por um combinador óptico parcialmente transmissivo e refletivo, tipicamente um espelho semi-prateado. A visão do mundo real é deixada praticamente intacta pelo combinador óptico, enquanto a imagem sintética é opticamente sobreposta à imagem real. Na maioria de *Head-Mounted Displays*

(HMDs) OST, os combinadores ópticos são normalmente posicionados no final do caminho óptico, bem à frente dos olhos do usuário. No caso de dispositivos que utilizam um espelho semi-prateado, a cena real é vista através dele, enquanto as imagens sintéticas são refletidas nele. O dispositivo de imagem não deve bloquear a visão do mundo real, por isso, normalmente está localizado acima do combinador óptico ou ao lado da cabeça do usuário (conforme Figura 1(a)). Entre as principais vantagens dos HMDs OST tem-se: permitem uma visão instantânea e natural da cena real; uma uniformidade entre as visualizações combinadas e periféricas; e, geralmente, são estruturalmente simples e leves. Alguns exemplos de HMDs OST são o *Microsoft Hololens 2*, *Epson Moverio* e *Openinvent Ora*.

A Figura 1(b) mostra a configuração típica de um *display* VST. Nestes *displays*, a imagem do mundo real é primeiramente capturada por uma câmera, depois a imagem capturada e a imagem sintética são combinadas eletronicamente e, finalmente, a imagem combinada é apresentada ao usuário. A fusão eletrônica pode ser realizada por *frame grabbers* (como as câmeras digitais) ou dispositivos de *chroma-keying*. Comparados aos *displays* OSTs, os *displays* VSTs são muito menores. Como resultado, os pesquisadores frequentemente têm que construí-los manualmente, usando um HMD de RV (como por exemplo, o *Oculus Rift*) e uma ou duas pequenas câmeras de vídeo, como as *webcams*. As vantagens dos *displays* VSTs incluem a consistência pictórica entre as visualizações real e sintética e a disponibilidade de uma variedade de técnicas de processamento de imagens. O *Vuzix M400* e *Oculus Rift DK2* for AR são exemplos de HMDs VSTs.

Um aspecto que merece ser destacado, é que a maioria destes *displays* são de relativo alto custo, o que tem dificultado, de modo geral, a popularização das aplicações de RA.

Outro desafio para os sistemas de RA, além da tecnologia de *display* adequada, é a interação com os elementos sintéticos (objetos virtuais 2D ou 3D gerados por computador). Segundo [Billinghurst, Clark e Lee \(2015\)](#), os sistemas de RA podem incorporar vários tipos de entrada e tecnologias de interação. Diferentes métodos de entrada podem ser usados para diferentes tipos de aplicações de RA, dependendo das tarefas de interação do usuário requeridas em uma aplicação. Os principais tipos de interfaces de RA existentes são:

- **Browsers de informação:** interfaces que exibem informação de RA no mundo real;
- **Interfaces 3D para o usuário:** usam técnicas de interação 3D para que o usuário manipule conteúdo no espaço;
- **Interfaces tangíveis com o usuário:** usam objetos reais para interagir com conteúdo virtual;
- **Interfaces naturais com o usuário:** utilizam a entrada natural do corpo, como os gestos;
- **Interfaces multimodais:** usam entrada combinada da fala e dos gestos, por exemplo.

A qualidade das interações permitidas pelas interfaces citadas dependem, também, do tipo de *display* (OST ou VST) utilizado na visualização do ambiente aumentado. Lembrando ainda a definição de [Azuma \(1997\)](#), os sistemas de RA devem ser capazes de alinhar (ou registrar) corretamente o conteúdo virtual com a visão do usuário do mundo real. Este registro só é possível utilizando o rastreamento 3D do ambiente real em torno do usuário. Os pesquisadores têm se esforçado para melhorar as capacidades dos HMDs de RA, sendo que alguns deles permitem a reconstrução de ambientes 3D, estimativa de pose, reconhecimento e rastreamento e obtenção de outras informações contextuais do ambiente circundante ao usuário.

1.1 Problema

Recentemente, os sistemas de RA têm atingido a nova geração de dispositivos móveis, como os *smartphones* e *tablets*. Estes dispositivos móveis podem ser acoplados em *headsets* devidamente projetados, viabilizando a construção de HMDs VST e OST de baixo custo. Exemplos deste tipo de HMDs são o *VR BOX 2* baseado no Google *Cardboard* e o *Haori AR Headset*, conforme mostrado na Figura 2.

Figura 2 – HMDs de RA baseados em *smartphones*. (a) *VR BOX 2*. (b) *Haori AR Headset*.



Fonte: Elaborada pelo autor.

Diversos trabalhos preocuparam-se em investigar o problema da percepção espacial em HMDs convencionais em ambientes de RV e RA, podendo ser citados: [Naceri, Chellali e Hoinville \(2011\)](#), [Plopski et al. \(2016\)](#), [Ballestin, Solari e Chessa \(2018\)](#), [Ping, Liu e Weng \(2019\)](#), [Batmaz et al. \(2019\)](#). No entanto, visto que os HMDs de RA baseados em *smartphones* são equipamentos recentes, são escassas na literatura, informações sobre quais são as principais dificuldades no desenvolvimento de aplicações de RA para os mesmos, especialmente aquelas que dizem respeito a percepção de profundidade e ao tipo de interação do usuário.

1.2 Justificativa

Segundo [Ballestin, Solari e Chessa \(2018\)](#), é conhecido que o sistema visual do ser humano estima profundidade fazendo uso de várias pistas, que interagem entre si para resolver

ambiguidades da percepção. Mas assim como os HMDs de RV, os HMDs de RA também eliminam muitas destas pistas. Isso pode ser observado em maior evidência nos dispositivos VST, pois o usuário é privado da visão direta do mundo real. Portanto, a inibição destas pistas ao fazer uso desses *displays* afetam a percepção de profundidade do usuário.

Considerando o contexto exposto, justifica-se o interesse na determinação de como a percepção de profundidade do usuário é afetada quando HMDs VST baseados em *smartphones* são utilizados, com ênfase nos principais aspectos que influenciam a interação do usuário, principalmente no espaço em torno do corpo do mesmo (espaço peripessoal). O espaço peripessoal refere-se ao espaço que pode ser alcançado pelas mãos do usuário, geralmente envolvendo distâncias de até 1 metro ([CUTTING; VISHTON, 1995](#)).

1.3 Objetivos

1.3.1 Objetivo Geral

O objetivo desse projeto é avaliar, por meio da realização de um estudo de caso, como o uso de HMDs *Video See-Through* baseados em *smartphones*, influenciam a percepção de profundidade e as interações do usuário, via interface baseada em gestos, dentro de seu espaço peripessoal.

1.3.2 Objetivos Específicos

- Desenvolver um sistema que integre um HMD VST para RA baseado em *smartphone* e um sensor de profundidade, a fim de permitir a detecção de gestos do usuário.
- Estruturar e implementar um ambiente de teste, baseado no sistema desenvolvido, voltado à determinação da influência do uso deste tipo de HMD, na percepção de profundidade e na interação do usuário no espaço peripessoal.
- Realizar testes e analisar os resultados obtidos, utilizando métricas objetivas (quantitativas).

1.4 Contribuição

Visto que os HMDs VST baseados em *smartphone* são equipamentos recentes, tem-se poucas informações sobre a influência de sua utilização, no que diz respeito à percepção visual de profundidade e a precisão das interações, em ambientes aumentados. Por isso, a maior contribuição desse trabalho é identificar e mensurar os principais aspectos relacionados ao problema, limitado ao espaço peripessoal do usuário e às interações por meio de gestos. Os

resultados obtidos, portanto, poderiam auxiliar na escolha, ou não, deste tipo de HMD para determinada aplicação de RA.

1.5 Organização do trabalho

O presente trabalho está estruturado da seguinte forma:

No Capítulo 2 tem-se o histórico de RA, sua definição e suas principais dificuldades. Também fornece informações sobre a percepção visual humana, sobre tecnologias de *displays*, de rastreamento 3D e espaço peripessoal no contexto de RA.

O Capítulo 3 apresenta um panorama dos trabalhos desenvolvidos na área, a fim de levantar o estado da arte na área de pesquisa desse trabalho.

O Capítulo 4 descreve o desenvolvimento do projeto, desde sua arquitetura lógica e física até os detalhes de sua implementação.

O Capítulo 5 discute os experimentos realizados com o sistema desenvolvido e seus resultados.

Finalmente, no Capítulo 6 são apresentadas as conclusões e as possibilidades de trabalhos futuros.

2 Fundamentação Teórica

Neste capítulo são apresentados os conceitos considerados mais importantes para este trabalho, tais como, Realidade Aumentada, Tecnologias de *Display*, Rastreamento 3D com ênfase no rastreamento óptico e Espaço Peripessoal.

2.1 Realidade Aumentada

Apesar da recente popularização da RA, não é atual a ideia de combinar objetos virtuais e reais em um mesmo ambiente. Desde o final do século 19 já eram conhecidas técnicas faziam uso de lentes e espelhos para projetar uma imagem virtual no mundo real. Um exemplo disso é a técnica usada no teatro e em apresentações conhecida como "*Pepper's Ghost*", muito difundida por sua aplicação na criação de ilusões de aparições sobrenaturais ou gerar o efeito visual de metamorfose (truque clássico de mágica da Monga, a Mulher Gorila) ([BILLINGHURST; CLARK; LEE, 2015](#)).

2.1.1 Histórico

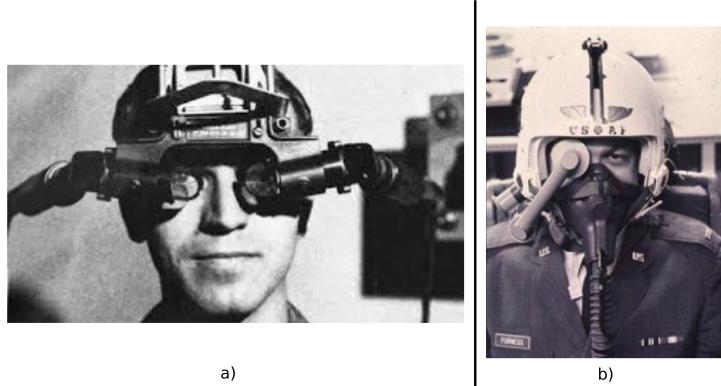
Na década de 1960, havia a fantasia da criação de um "*display* perfeito", que seria capaz de criar objetos virtuais que fossem capazes de interagir com o usuário semelhante a um objeto real. A primeira tentativa de realizar essa fantasia, foi em 1968 com o primeiro HMD de RA, criado por Ivan Sutherland. O protótipo era relativamente primitivo, como pode ser visto na Figura 3 (a), mas era capaz de renderizar imagens 2D que eram projetadas por um pequeno *display* CRT (*Cathode-ray tube*) em duas lentes em frente aos olhos do usuário, mesmo fundamento usado em HMDs OST modernos ([SUTHERLAND, 1968](#)).

Após isso, por muito tempo, segundo [Billinghurst, Clark e Lee \(2015\)](#), a pesquisa em RA ficou limitada à empresas privadas e instituições governamentais, tal como o protótipo de "super cabines", mostrado na Figura 3 (b), para a Força Área dos EUA em 1969, cujo objetivo era arranjar maneiras mais práticas de apresentar informações aos pilotos. Porém, atualmente com a evolução da Tecnologia de Informação e o aumento do poder de processamento, fazendo com que os dispositivos de alta difusão no mercado sejam suficientemente pequenos, baratos e poderosos, a popularização da tecnologia se faz finalmente possível ([HOUNSELL; TORI; KIRNER, 2018](#)).

2.1.2 Definição de Realidade Aumentada

Segundo [Azuma \(1997\)](#), a Realidade Aumentada pode ser definida como uma variação de Realidade Virtual. Porém enquanto que o usuário fica completamente imerso em um sistema

Figura 3 – Primeiros HMDs de RA. (a) Primeiro protótipo por Ivan Sutherland e (b) "Super Cabine" da Força Área dos EUA



Fonte: [Billinghurst, Clark e Lee \(2015\)](#).

de RV, um sistema de RA tenta aumentar a percepção de realidade do usuário. Mas o que caracteriza exatamente um sistema de RA? Retornando à definição de Azuma, apresentada na introdução. Um filme com imagens em computação gráfica não se enquadraria, pois não é um sistema executado interativamente em tempo real.

Portanto, pode-se dizer que essas características de RA são como requisitos que uma aplicação dessa tecnologia se propõe a cumprir. Ou seja, um sistema de RA deve sempre possuir um *display* em que serão apresentados para o usuário a visão combinada do real e virtual (módulo de saída). Também deve possuir um computador que tratará do processamento e renderização dos objetos virtuais (módulo de processamento). E por fim, deve possuir alguma maneira de poder fazer o rastreamento do ambiente a fim de realizar o alinhamento dos objetos (módulo de entrada). E são nesses três módulos que uma aplicação clássica de RA é dividida ([HOUNSELL; TORI; KIRNER, 2018](#)).

Vejamos um exemplo: [Magnenat et al. \(2015\)](#) criaram uma aplicação que muda a coloração de um modelo 3D baseado na coloração feita em uma ilustração, como pode ser visto na Figura 4. Nesse caso, o módulo de entrada é a câmera do *smartphone* ou *tablet* que está capturando a imagem do mundo real e usando a imagem do desenho para realizar o rastreamento da posição da câmera. O módulo de processamento está sendo cuidada pelo processador do dispositivo móvel. E por fim, o módulo de saída é o *display* do dispositivo móvel.

Vale ainda ressaltar a importância de realizar pesquisa nessa área: a possibilidade de mostrar informações ao usuário, que normalmente ele não teria acesso, auxiliando na realização tarefas no mundo real. Isso abre possibilidades para aplicações em múltiplas áreas como treinamento, medicina, arquitetura e educação já mencionadas antes. Basicamente, qualquer área que se beneficiaria da percepção do usuário pode ser melhorada por meio da combinação de objetos 2D ou 3D ([BILLINGHURST; CLARK; LEE, 2015](#)).

Figura 4 – Livro sendo colorido e modelo 3D correspondente.



Fonte: [Magnenat et al. \(2015\)](#).

2.1.3 Desafios

Apesar do grande potencial da área e dos avanços alcançados em recentes anos, ainda existem alguns desafios que devem ser superados a fim de que esse potencial seja realmente realizado. Para [Billinghurst, Clark e Lee \(2015\)](#), os principais obstáculos a serem superados são referentes à limitações tecnológicas, falta de padronização, erros no rastreamento e aceitação. Outro problema muito presente na discussão de RA é como a percepção visual é afetada pelo uso dessa tecnologia. Serão discutidos a seguir com mais detalhes cada um desses tópicos.

2.1.3.1 Limitações Tecnológicas

Seres humanos conseguem, sem muita dificuldade, distinguir o real do virtual. Mesmo que existam muitos estudos que investigam a criação de modelos 3D fotorrealistas como os documentados por [Nakamae e Tadamura \(1995\)](#), ainda se está longe de criar tais modelos consistentemente, principalmente em uma aplicação que deve ser interativa em tempo real, como uma de RA.

De acordo com [Azuma \(1997\)](#), o objetivo final da área de estudo será gerar objetos que terão tamanha fidelidade gráfica que serão indistinguíveis dos objetos reais. Também volta a ressaltar que por mais que essa fidelidade seja possível em filmes ou simulações, realizar isso em uma aplicação interativa em tempo real é muito mais difícil.

Apesar disso, um nível extremo de fidelidade gráfica não é necessária em todo tipo aplicação. Por exemplo, uma aplicação que gera modelos 2D ou 3D (como um projeto de CAD - *Computer Aided Design*) mais abstratos acabam funcionando melhor para arquitetos e engenheiros no auxílio de seus projetos. Ademais, não devemos desconsiderar o valor artístico que pode-se ter em uma modelagem mais cartunesca que também pode acabar servindo para dar ênfase aos objetos aumentados ([HALLER, 2004](#)).

2.1.3.2 Falta de Padronização

Conforme Hounsell, Tori e Kirner (2018), uma das principais desvantagens do desenvolvimento em RA é a falta de padronização no que diz respeito a forma como é promovida a integração dos dispositivos com o processamento e a tarefa, não existindo uma solução pronta para esse problema.

Outro tópico em RA que falta padronização é referente a interface com o usuário. Na Introdução foram apresentadas algumas modalidades de interface e interação usadas em sistemas de RA. As primeiras aplicações usavam tecnologias de interação semelhantes a aplicações *desktop* e de RV, porém há um potencial bem maior para aplicações de RA, principalmente com interações baseadas em gestos. Mas mesmo que haja potencial, as interfaces geralmente possuem somente interações simples e ainda há pouca informação na literatura de como incluir formas mais inteligentes de interação com usuário (BILLINGHURST; CLARK; LEE, 2015).

Ritsos, Ritsos e Gouglis (2011) dizem que geralmente práticas de UX (*User eXperience*) são negligenciadas no desenvolvimento de aplicações de RA a fim de priorizar tempo e simplicidade. Porém, boas práticas em relação a senso de presença, ergonomia, saúde, segurança e usabilidade deveriam ser considerados como parte crucial da padronização dessa tecnologia.

RA é um campo de pesquisa volátil e com muita profundidade, com aplicações variando de jogos e entretenimento até medicina e equipamentos militares. Essa profundidade gera um espaço onde os requisitos são muito dependentes da natureza da aplicação. Apesar disso, criando-se uma padronização, a inovação seria encorajada, usando boas práticas pré definidas como base, gerando confiança de mercado e possibilitando colaboração entre desenvolvedores (PEREY; ENGELKE; REED, 2011).

2.1.3.3 Erro de Rastreamento

Rastreamento é definido como a capacidade do sistema de conseguir identificar a pose do observador em relação ao ambiente aumentado. Investigar os erros e problemas dessa tecnologia é de suma importância para o contexto de RA, já que é através do rastreamento que o registro (alinhamento) dos objetos reais e virtuais é atingido. Logo, por ser uma parte tão importante, o ideal é que o rastreamento seja o mais perfeito possível. Muitos avanços foram alcançados, porém ainda existem algumas tarefas que a tecnologia de rastreamento ainda não consegue realizar muito bem, tais como: rastreamento em grandes áreas, rastreamento ao ar livre, rastreamento ubíquo (BILLINGHURST; CLARK; LEE, 2015).

Segundo Azuma (1997), se o registro não for feito corretamente, a ilusão da combinação do ambiente virtual e real será totalmente quebrada e isso impossibilitaria a aplicação de RA em muitas áreas que pedem uma maior precisão (medicina por exemplo). Erros de registro são de difícil controle por conta da alta necessidade de precisão que essa tarefa requer. As fontes de erro podem ser divididas em duas categorias: erros estáticos e dinâmicos.

Erros estáticos são os erros que ocorrem mesmo quando o usuário e os objetos estão completamente parados. Esses erros costumam ter menor taxa de ocorrência, mas não devem ser deixados de lado. Os causadores mais comuns são: distorção óptica, erros no sistema de rastreamento, desalinhamentos mecânicos e parâmetros de visualização incorretos.

Já os erros dinâmicos ocorrem com maior frequência. São os erros que só começam a ter impacto no problema quando há movimentação do ponto de vista do usuário ou dos objetos. Esse tipo de erro está mais relacionado atrasos no sistema.

2.1.3.4 Aceitação

Enquanto que todos os desafios previamente discutidos tiveram muita pesquisa dedicada a investigar soluções e saídas para seus problemas, de acordo com [Billinghurst, Clark e Lee \(2015\)](#) a questão da aceitação social da RA não recebeu muita atenção, em termos de pesquisa, desde a criação da tecnologia. É dito que a limitação para a difusão das aplicações de RA não é tecnológica, mas sim social. Por mais que, com o avanço tecnológico, os HMDs estejam cada vez menores e mais robustos, ainda é difícil a aceitação destes em meios sociais.

De acordo com [Hong \(2013 apud BILLINGHURST; CLARK; LEE, 2015\)](#), foi realizada uma pesquisa com 4600 adultos e somente 12% disseram que usariam óculos de realidade aumentada de uma marca que eles confiassem. Os motivos dados pelos participantes foram bem diversos como: privacidade, não querer parecer bobo e não querer que outras pessoas pensem que estão sendo filmadas.

Apesar disso, estão começando a aumentar as pesquisas que tentam identificar o estado da aceitação social de RA, como [Perannagari e Chakrabarti \(2019\)](#). E também tentativas de implementação de sistemas em hospitais e em contextos estudantis, como [Nilsson e Johansson \(2008\)](#), e nesse contexto o resultado tem sido bem positivo. Porém ainda falta maior investigação quanto a aceitação desse tipo de aplicação em contextos mais abrangentes.

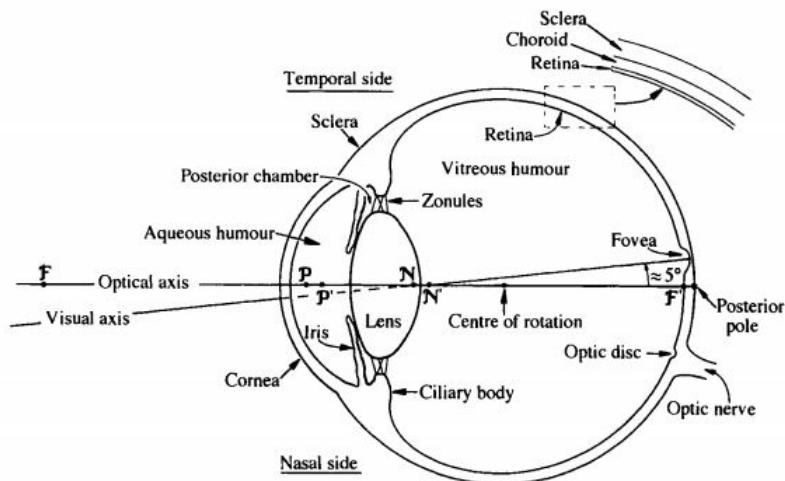
2.1.3.5 Percepção Visual

Uma aplicação de RA tem a possibilidade de estimular múltiplos sentidos, até mesmo simultaneamente, como a audição através do uso de sons 3D ou tato através de dispositivos com retorno haptico. Porém o sentido com maior foco é inegavelmente a visão, com o uso de *displays* que tentam emular a percepção 3D. Antes de discutir as especificidades da Percepção Visual quando envolvidos HMDs de RA, será tratado brevemente como o Sistema Visual humano funciona e percebe a profundidade.

Primeiro, tem-se a estrutura básica do olho humano, ilustrado na Figura 5. Como pode ser visto, a parte mais externa do olho é dividida em duas partes, a córnea (transparente) mais a frente e a esclera (branca e opaca) mais atrás. Mais internamente temos a úvea que é composta por corpo ciliar, coroide e íris. A íris tem o papel importante de fazer o controle

da quantidade de luz que entra pelos olhos através de sua abertura variável e o corpo ciliar tem importante papel no processo de acomodação que será detalhado depois. A camada mais interna é a retina, que é a extensão do sistema nervoso, ela é responsável por receber os estímulos e enviá-los ao cérebro. É na retina também que se localizam os foto-receptores, os chamados cones e os bastonetes (ATCHISON; SMITH, 2000).

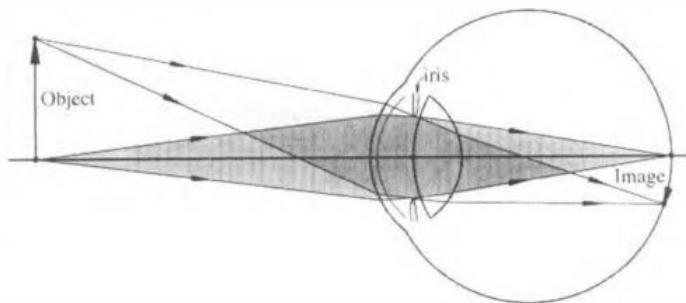
Figura 5 – Estrutura do olho humano.



Fonte: Atchison e Smith (2000).

Voltando a Atchison e Smith (2000), os princípios da formação da imagem no olho humano, conforme pode ser visto na Figura 6, são semelhantes aos das lentes de câmeras. A luz entra através da córnea e é refratada por ela e pelo cristalino para ser focada na retina. A refração provida pela córnea é constante, mas já a do cristalino não. Cristalino ou lente é uma lente biconvexa e altamente flexível e com a ajuda do corpo ciliar, o cristalino pode contrair ou relaxar a fim de focar em diferentes distâncias. Esse processo é chamado de acomodação e é uma das pistas de profundidade.

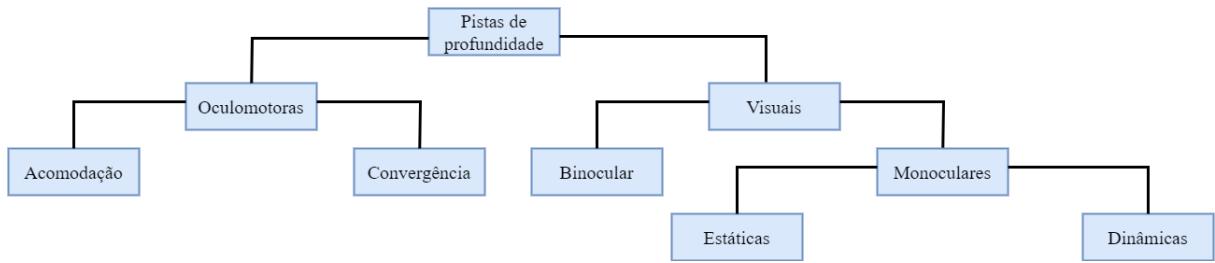
Figura 6 – Formação de imagem no olho humano.



Fonte: Atchison e Smith (2000).

Como já comentado no Capítulo 1 desse trabalho, o sistema visual humano depende de pistas para estimar profundidade. E de acordo com Reichelt et al. (2010), as pistas são divididas em oculomotoras e visuais, enquanto que a segunda é dividida em monoculares e binoculares. As monoculares também são divididas em estáticas e dinâmicas. Tal classificação está ilustrada na Figura 7.

Figura 7 – Taxonomia das pistas de profundidade



Fonte: Baseado em uma Figura de Reichelt et al. (2010)

Segundo Drascic e Milgram (1996), as pistas de profundidade monoculares são as que podem ser detectadas através de uma simples foto, elas são as seguintes:

- **Tamanho Relativo:** Conhecido o tamanho do objeto, se ele aparentar estar maior, ele está mais perto e se aparentar estar menor, estará mais longe.
- **Oclusão:** Objetos que estão na frente dos outros, estão mais perto.
- **Perspectiva Linear:** Tem relação com a aparência de que as linhas convergem no horizonte, objetos que aparentam estar mais perto dessa convergência, estão mais longes.
- **Perspectiva Aérea:** Essa pista tem efeito quando o observador está a grandes distâncias dos objetos. É referente a perda de nitidez e cor da visão do objeto por conta da atmosfera.
- **Brilho Relativo:** Quando objetos estão mais afastados das fontes de luz, eles aparentam estar mais escuros.
- **Gradiente de Textura:** Relaciona-se com a pista de perspectiva linear, em que a densidade da textura aumenta junto com a distância.

As pistas monoculares dinâmicas estão relacionadas com a mudança do ponto de vista e movimento, elas são:

- **Paralaxe de Movimento:** Quando o observador está se movendo, os objetos fixos que ele observa tem a ilusão de estarem andando na direção oposta a ele. A velocidade com a qual esses objetos "andam" dão uma pista de profundidade, se eles estão perto terão a ilusão de se mover mais rápido.

- **Efeito de Profundidade Cinética:** Muito semelhante a paralaxe de movimento, mas requer que somente o objeto esteja em movimento e não o observador.

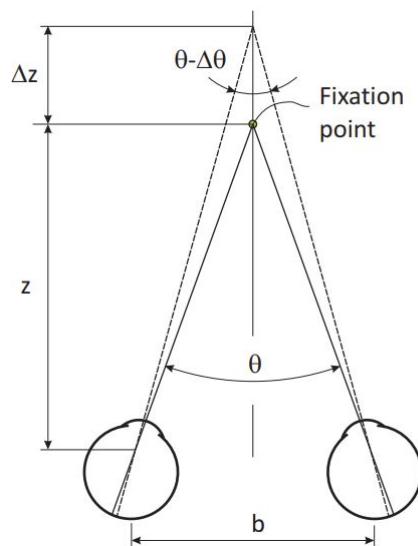
A pista binocular é a:

- **Disparidade Binocular:** A paralaxe de movimento providênciaria informações de profundidade graças aos múltiplos pontos de vista gerados pelo movimento, porém os seres humanos possuem dois olhos e portanto possuem dois pontos de vista a todo momento sem precisar se mover. O sistema visual consegue extrair muitas informações sobre a profundidade dos objetos analisando as disparidades das imagens dos dois olhos.

As pistas oculomotoras estão relacionadas com o movimento dos dois olhos, existem duas delas:

- **Acomodação:** Essa foi a pista que foi descrita anteriormente que possui relação com o cristalino do olho. Basicamente, o cristalino vai alterar dinamicamente a distância focal do olho se contraíndo (perto) ou relaxando (longe).
- **Vergência:** É referente ao ato dos olhos rotacionarem em direções diferentes para fixarem sua visão em um objeto, como visto na Figura 8, quanto mais perto o objeto está do observador, mais os olhos se angularão para dentro para focá-lo.

Figura 8 – Demonstração da vergência: Ângulo θ maior se menor distância do objeto ao observador (menor z).



Fonte: [Reichelt et al. \(2010\)](#).

Dadas essas pistas, as mais predominantes na estimativa de profundidade no espaço peripessoal são: disparidade binocular, convergência, acomodação, tamanho relativo e oclusão.

Para reproduzir disparidade binocular em HMDs VST de RA é necessário que seja renderizada uma imagem diferente para cada olho com a distância interpupilar (IPD - *Interpupillary Distance*) do usuário. Porém, sem que haja duas câmeras no HMD, essa tarefa se torna complicada e pode acabar custando parte do campo de visão (FOV - *Field Of View*), já que a imagem da única câmera precisa ser manipulada para gerar duas de dois pontos de vista diferentes. Por outro lado, os HMDs OST não negam essa pista para os objetos dos mundos reais, mas os objetos virtuais precisam ser projetados de maneira correta na lente.

As pistas monoculares são em grande parte preservadas entre os objetos virtuais (tamanho relativo e oclusão) se a cena virtual for organizada corretamente, porém se a oclusão não for tratada na aplicação, ela também será eliminada.

O verdadeiro problema ocorre no que chamamos de conflito vergência-acomodação ao se usar HMDs VST de RA, que completamente negam a contribuição dessas pistas. Ademais, visto que essas pistas se baseiam na distância em que os objetos estão do observador, sendo que em um HMD, os objetos virtuais estarão a uma distância fixa em frente os olhos do usuário, é criado o conflito mencionado, que ao expor o usuário por muito tempo de uso, pode gerar dores de cabeça e fadiga ([BALLESTIN; SOLARI; CHESSA, 2018](#)).

2.2 Tecnologias de *Displays*

Houve muito avanço em tecnologias de *displays* em recentes anos, mas não foi atingido o "display perfeito" imaginado por Sutherland. Atualmente, HMDs OST possuem algumas desvantagens como: não possuir um grande FOV e não serem capazes de fazer oclusão dos objetos virtuais com o mundo real. Enquanto que os *displays* VST, ainda possuem muitos problemas que afetam a percepção visual, como já detalhados anteriormente.

Porém, voltando à definição de Azuma, não é especificado que aplicações de RA devam usar HMDs. Existem mais meios pelos quais uma aplicação pode ser executada, como será visto mais a frente.

Também vale comentar as especificidades da arquitetura dos HMDs e também os atributos que devemos nos atentar ao analisar um modelo desses capacetes.

2.2.1 Taxonomia

Segundo [Hounsell, Tori e Kirner \(2018\)](#), a RA pode ser classificada quanto a tecnologia de visualização nas seguintes categorias e suas divisões: visão direta (dividido em óptica e por vídeo) e visão indireta (projetor e monitor).

2.2.1.1 Visão Direta

A visão direta é definida pelo controle do ponto de vista ser de domínio do usuário. Está geralmente atrelada ao uso de HMDs, sendo que a visão direta óptica é caracterizada pelo OST e a visão direta por vídeo, o VST.

Como já dito no Capítulo 1, em um *display* VST, a visão do mundo real é digitalizada por uma ou duas câmeras e as imagens dos objetos virtuais são combinadas a cada *frame*. Um exemplo disso pode ser visto na Figura 9, onde foi feita uma aplicação para o auxílio de realização de cirurgias Buco-Maxilo-Facial. Normalmente antes da cirurgia é realizada uma Tomografia Computadorizada (CT - *Computerized Tomography*) e o cirurgião faz simulações se baseando no resultado do exame. Na hora da cirurgia de fato, o profissional deve lembrar os procedimentos com base nas simulações, porém com o uso dessa aplicação ele pode ser guiado com o uso de imagens aumentadas ([WANG et al., 2017](#)).

Figura 9 – Projeto para auxílio de realização de cirurgias Buco-Maxilo-Facial. (a) Cirurgião usando o HMD. (b) Rastreamento do dente (c) Visão aumentada.



Fonte: [Wang et al. \(2017\)](#)

Diferentemente dos *displays* VST, nos *displays* OST, um combinador óptico sobrepõe as imagens virtuais sobre a visão direta do mundo real. Exemplo de uma aplicação que faz uso dessa modalidade pode ser encontrado em [Chen et al. \(2015\)](#), também da área da saúde. Se trata de um sistema de que auxilia nos treinamentos pré-cirurgias.

Segundo [Kiyokawa \(2015\)](#), HMDs também podem ser classificados quanto a sua ocularidade. Existem três categorias: monocular, biocular e binocular. Um *display* monocular é aquele que apresenta a imagem para um olho só, isso pode ser preferível em uma situação ao ar livre, visto que ele é menos intrusivo. Um *display* biocular é o que apresenta uma única imagem para os dois olhos, bons para aplicações casuais para entretenimento, onde a visão estéreo não é tão necessária. E por fim, o *display* binocular apresenta uma imagem para cada olho e é o mais usado em aplicações.

2.2.1.2 Visão Indireta

A visão indireta é o oposto da direta, visto que o ponto de vista não é controlado diretamente pelo usuário. Essa modalidade ocorre quando o módulo de saída da aplicação é um

monitor ou um projetor. Quando o *display* é um monitor, a aplicação funcionará semelhante a um HMD VST, porém não diretamente atrelado ao ponto de vista do usuário, o ponto de vista seria dado por uma câmera em um ponto fixo do ambiente.

O módulo de saída também poderá ser um projetor (ganhando nome de RA espacial), que fará uma projeção diretamente sobre um objeto físico com a intenção de aumentar a percepção com alguma finalidade. Útil em casos em que se queira incorporar detalhes a certos objetos ou mostrar suas partes internas ([HOUNSELL; TORI; KIRNER, 2018](#)).

Um exemplo de uma aplicação de RA espacial é a mostrada na Figura 10. Essa aplicação foi feita para facilitar o processo de solda ponto no chassi dos carros da *General Motors* (GM). Nesse caso eram projetados círculos verdes, indicando instruções aos funcionários, sem que houvesse a necessidade de cada um deles se equipar com um HMD para realizar a tarefa com o auxílio da aplicação ([DOSHI et al., 2017](#)).

Figura 10 – Aplicação com projetor da GM. Os círculos verdes estão sendo projetados no chassi.



Fonte: [Doshi et al. \(2017\)](#).

Um exemplo de uma aplicação de RA baseada em monitor se vê na Figura 11. Usando essa aplicação, o autor propõe um método de se medir a interação no espaço peripessoal ([JIMÉNEZ, 2014](#)).

2.2.2 Arquitetura dos HMDs

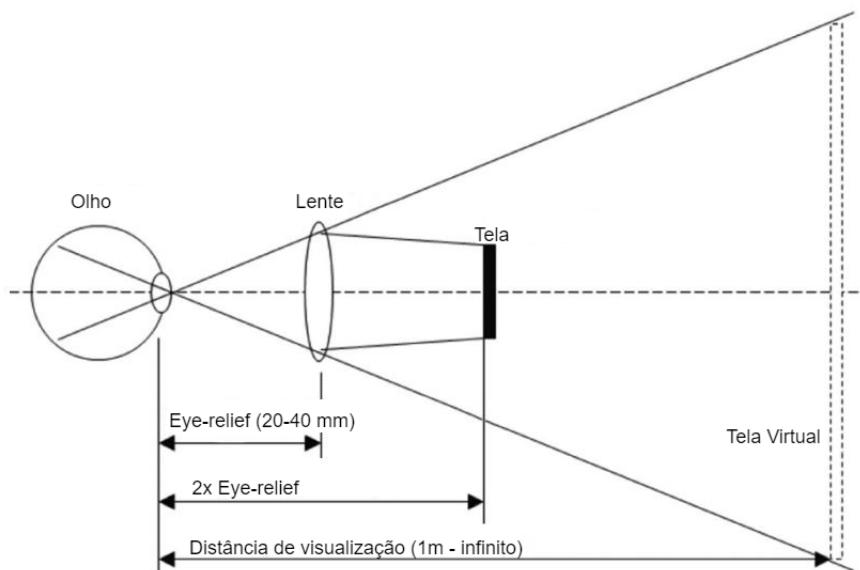
Tipicamente, os HMDs precisam ampliar a imagem vinda da tela, para que a visão do usuário seja realmente coberta. Para que isso seja feito, uma lente é posicionada entre o olho do usuário e a tela a uma distância que recebe o nome de *eye-relief*, ilustrado pela Figura 12. Essa distância acaba determinando muitos aspectos do HMD, como seu tamanho e seu FOV. Se o *eye-relief* for muito grande, o HMD pode acabar ficando grande e pesado, porém se for muito pequena o FOV diminui ([KIYOKAWA, 2015](#)).

Figura 11 – Aplicação de visão indireta. Pode ser visto o monitor no fundo, onde a cena aumentada por ser vista.



Fonte: Jiménez (2014).

Figura 12 – Ilustração do *eye-relief*.

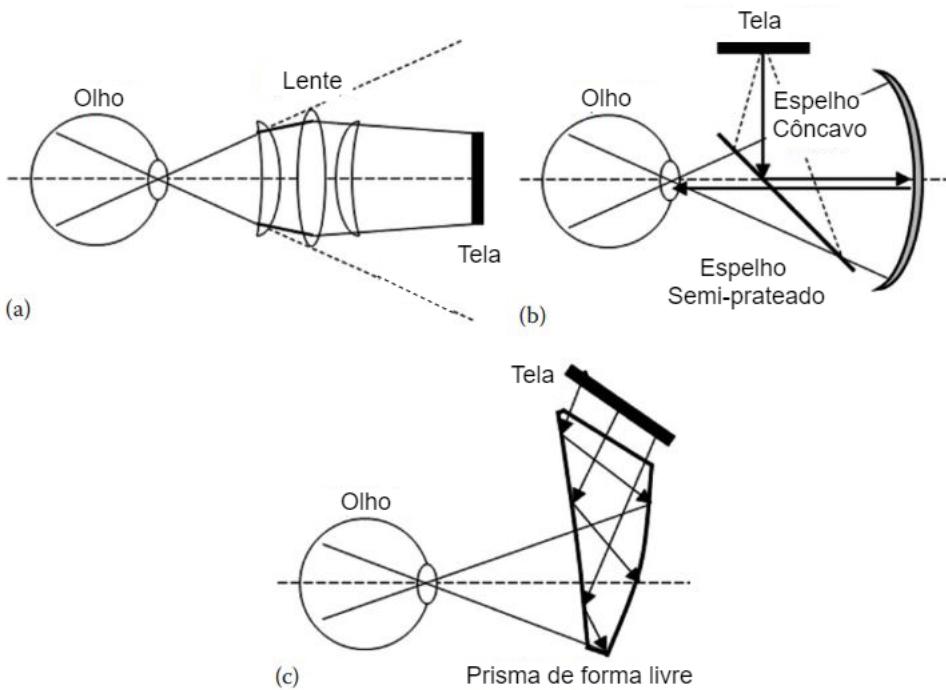


Fonte: Kiyokawa (2015).

Arquiteturas de HMD podem ser divididas em dois tipos: *pupil forming* e *non-pupil forming*. Arquiteturas baseadas em *pupil forming* geram HMDs com um FOV, tamanho e peso elevados, enquanto que arquiteturas *non-pupil forming*, que são mais usadas atualmente, geram um FOV modesto com um baixo peso e tamanho.

Na Figura 13, podem ser vistos três exemplos de arquiteturas *non-pupil Forming*. Figura 13(a) ilustra uma arquitetura refrativa, essa que era muito usada no ínicio do desenvolvimento de HMDs, as três lentes são necessárias para corrigir deformações. Já a Figura 13(b) mostra um esquema para um HMD catadióptrico, nesse caso a luz vinda da tela reflete primeiramente no espelho semi-prateado e, depois, no espelho côncavo (espelho côncavo semi-transparante se o HMD for OST) indo até o olho. Essa configuração diminui o tamanho e o peso, porém ao custo do olho receber somente 25% da luminosidade da tela, já que a luz passa pelo espelho semi-prateado duas vezes. Finalmente, na Figura 13(c), é apresentada uma configuração que usa um prisma de forma livre. Assim sendo, esse exemplo possui menor peso sem pagar o preço da perda da luz da tela, visto que a luz acaba passando por duas reflexões totais dentro do prisma antes de chegar ao olho.

Figura 13 – Exemplos de arquiteturas *non-pupil Forming*: (a) Refrativa (b) Catadióptrico (c) Prisma de forma livre.



Fonte: [Kiyokawa \(2015\)](#).

Outra possibilidade é a de fazer uso de um elemento holográfico óptico (HOE - *Holographic Optical Element*). Graças a capacidade de realizar difração nas ondas de luz (espalhamento de ondas ao encontrar um obstáculo), o HOE é capaz de ajudar a produzir HMDs pequenos e leves. Eles também funcionam como um combinador óptico transparente

devido a sua seletividade com certos comprimentos de onda.

Existem outros exemplos e ideias de arquiteturas para HMDs como o *Virtual Retinal Display* (VRD), cuja premissa é projetar imagens diretamente na retina, a fim de evitar uso de displays e ópticas complexas, atingindo um grande FOV e resolução ao mesmo tempo. Outro modelo é o *Head-mounted projective display* (HMPD), que possui dois pequenos projetores ao invés de fazer uso de telas ([KIYOKAWA, 2015](#)).

2.2.3 Atributos dos HMDs

Nesta subseção, são discutidas as características de um HMD, como resolução, campo de visão, oclusão, profundidade de campo, latência, paralaxe, distorções, consistência pictorial, multimodalidade e sensação ([KIYOKAWA, 2015](#)).

2.2.3.1 Resolução

Uma característica bastante importante de um HMD é sua resolução, pois é um dos fatores que vão definir a fidelidade da imagem que será apresentada ao usuário. Isso é especialmente importante em HMDs VST, já que o usuário verá até mesmo o mundo real através do *display*, ademais nessa situação, a resolução da câmera usada também é importante. Segundo [Kiyokawa \(2015\)](#), para competir com a visão humana os HMDs precisariam ter uma resolução de 12000 x 7000 *pixels*.

Um termo que precisa ser considerado é o de resolução angular, que é o número de *pixels* por grau de FOV (PPD - Pixel Per Degree). Portanto uma escolha é feita ao se projetar um HMD: ter um amplo FOV e uma baixa resolução angular ou ter um FOV menor por mais resolução angular. Mesmo com a resolução nas telas dos dispositivos aumentando cada vez mais, essa troca entre o FOV e a resolução vai ser difícil de contornar.

2.2.3.2 Campo de Visão

Campo de visão ou FOV é a extensão angular em que o usuário é capaz de enxergar. O FOV de um HMD é essa extensão que o usuário é capaz de ter ao colocá-lo. De acordo com [Kiyokawa \(2015\)](#), há uma subárea dentro do FOV que é chamada de FOV ajudado ou de sobreposição, que é a seção do FOV em que os objetos virtuais vão estar ativos na visão do usuário.

Quando um HMD possui um FOV de menos de 60°, é normal que a taxa de sobreposição dele seja de 100%. Enquanto que quando o FOV é mais amplo, com algo a cima de 80°, a taxa será aproximadamente 50%. Além do FOV ajudado, existe o FOV periférico, onde não aparecerá nenhum objeto virtual. Em HMDs OST o FOV ajudado não consegue atingir grandes valores, a fim de evitar distorções.

2.2.3.3 Oclusão

Como foi dito na Seção 2.1.3.5, oclusão é uma pista de profundidade de suma importância, portanto seu tratamento amplia muito a percepção do usuário. Porém isso não é uma tarefa fácil, já que se faz necessário que sejam conhecidas as informações de profundidade de ambos os mundos, o real e o virtual. Do segundo é fácil de conseguir, mas já o primeiro exige a utilização de sensores externos que sejam capazes de extrair essas informações. Uma maneira de se atingir isso é fazendo uso de câmeras RGB-D (*Red, Green, Blue, Depth*), que são câmeras que capturam informações de profundidade além da imagem RGB.

Vale ressaltar que fazer o tratamento da oclusão em HMDs OST é bem dificultado, em virtude do fato que o usuário tem visão direta do mundo real (Kiyokawa, 2015).

2.2.3.4 Profundidade de Campo

De acordo com Kiyokawa (2015), profundidade de campo refere-se ao intervalo de distâncias em que o objeto aparece em foco em uma câmera ou olho. No olho humano, como explicado na Seção 2.1.3.5, o cristalino se ajusta para mudar a distância focal do olho, a fim de focar em objetos a diferentes distâncias.

É possível reproduzir o efeito do olho humano em imagens sintéticas, borrando os objetos virtuais de acordo com a distância deles do ponto de foco. Geralmente esse cálculo é realizado usando o valor de profundidade dos objetos virtuais. Porém esse método é fisicamente incorreto. Kán e Kaufmann (2012) propõe um método que utiliza *Ray-Tracing* para correção desses erros.

2.2.3.5 Latência

O termo latência no contexto de RA tem relação com a diferença de tempo de alguma mudança no sistema acontecer até ele ser apresentado ao usuário. Essa mudança pode ser um movimento da cabeça do usuário, um movimento dos objetos virtuais, o resultado de uma interação, o registro dos objetos virtuais e etc.

A latência pode passar mais despercebida em displays VST já que o usuário vê o mundo real através das câmeras e um possível atraso no registro pode ser compensado por atrasar a imagem da câmera também. Já em um OST, os objetos virtuais ficarão em posições erradas na visão do usuário o que pode gerar confusão e incômodo (Kiyokawa, 2015).

2.2.3.6 Paralaxe

Paralaxe é o problema que ocorre quando o ponto de vista da câmera é um ponto diferente dos olhos do usuário, ou seja, se a câmera de um HMD VST está localizada acima de onde os olhos do usuário ficam, isso pode gerar um sensação de altura. O problema pode ser eliminado por usar a arquitetura de prisma de forma livre (Seção 2.1.3.5) (Kiyokawa, 2015).

2.2.3.7 Distorções

Segundo Kiyokawa (2015), ao usar essa variedade de elementos ópticos na arquitetura dos HMDs, é normal que sejam inseridos distorções. Como, por exemplo, as aberrações cromáticas ocorrem dados os diferentes índices de refração para diferentes comprimentos de onda ou as aberrações circulares que são introduzidas graças ao formato esférico das lentes. Essas distorções podem ser corrigidas por software sem que haja a necessidade da inclusão de outros elementos ópticos.

2.2.3.8 Consistência Pictorial

Consistência pictorial é o conceito referente a manter uma certa consistência de brilho e contraste entre o mundo real e os objetos sintéticos. Nos HMDs OST, a consistência é difícil de ser atingida, já que é difícil para o *display* conseguir atingir os níveis de brilho das luzes do mundo real. No entanto, em HMDs VST, é mais simples (KIYOKAWA, 2015).

2.2.3.9 Multimodalidade

Kiyokawa (2015) diz que a maioria das aplicações de RA é baseada no sentido da visão (como dito na Seção 2.1.3.5). Porém um HMD pode possuir multimodalidade quanto a quantidade de sentidos que são alvo do aumento da percepção. Alguns sentidos são mais difíceis de se reproduzir do que outros, contudo existem muitas possibilidades no estudo de diferentes tipos de HMD que poderiam manipular os outros sentidos além da visão.

2.2.3.10 Sensação

No contexto de computação vestível, os HMDs devem tender a ser dispositivos que podem ser usados por longos períodos de tempo sem que precisem ser colocados ou retirados com muita frequência. Porém, além de precisar superar as limitações de hardware para que eles seja suficientemente pequenos e não intrusivos a usuário para que isso seja possível, os HMDs devem ser adaptáveis ao contexto que cercam ao usuário, a fim de que ele mostrasse somente o necessário para o usuário a cada momento e evitasse excesso de informações (KIYOKAWA, 2015).

2.3 Rastreamento 3D

Conforme foi definido na seção 2.1.3.3, rastreamento é a capacidade do sistema de encontrar a pose do observador em relação a alguma âncora no mundo real. Essa âncora pode ter várias formas, dependendo da tecnologia utilizada. Ela pode ser uma imagem impressa em uma folha de papel, que é o que pode ser chamado de marcador fiducial, um sensor ou uma posição definida usando o *Global Positioning System* (GPS). Após a determinação da pose, o sistema usa isso para fazer o alinhamento dos objetos virtuais em relação ao ambiente real e

permitir a interação. Existem muitas maneiras de realizar esse cálculo e elas serão discutidas a seguir ([BILLINGHURST; CLARK; LEE, 2015](#)).

Um fator que dificulta a popularização de RA é devido aos requerimentos necessários para começar a utilizá-la. De acordo com [DiVerdi e Hollerer \(2007\)](#), aplicações tradicionais de RA costumam fazer muitas suposições de como o ambiente deve estar e ser configurado. O objetivo é que com o tempo, sejam diminuídas essas barreiras, fazendo aplicações que só necessitem de hardware barato e popular para funcionarem e permitir que RA seja utilizado independente do contexto que o usuário está inserido. Melhorar a tecnologia de rastreamento é um aspecto chave para isso, já que é um dos limitantes.

Ao se analisar um método de rastreamento tem-se que nos atentar para as seguintes características:

- **Graus de Liberdade (DOF - *Degrees of Freedom*):** se refere a quantidade de maneiras que o sensor consegue rastrear o observador se movendo no espaço, podendo ter 6 graus no total. Os três primeiros graus correspondem aos movimentos de rotação ao redor dos eixos x, y e z, enquanto que os 3 últimos são as translações nesses eixos.
- **Precisão:** Capacidade do método de identificar a posição e a orientação do observador corretamente.
- **Taxa de Amostragem:** Frequência com a qual as informações sobre o observador são transmitidas ao sistema.
- **Latência:** Atraso levado para detectar uma mudança na pose do observador.

2.3.1 Taxonomia

Entre os vários modos em que o rastreamento pode ser realizado, existem duas categorias: baseado em sensores (focados no uso de sensores como GPS, sensores acústicos, sensores iniciais, sensores magnéticos, entre outros) e rastreamento baseado em visão (focados no uso de câmeras e algoritmos de Visão Computacional).

2.3.1.1 Baseado em Sensores

De acordo com [Azuma \(1997\)](#), para realizar o rastreamento, é necessário identificar continuamente a localização da cabeça do observador e de outros objetos no ambiente. Para isso se faz preciso sensores com precisão e longo alcance. Serão vistos agora alguns sensores que já foram utilizados para rastreamento.

- **GPS:** Boa opção para rastreamento ao ar livre, porém é pouco precisa e somente permite o rastreamento da posição, negligenciando a orientação. O Rastreamento baseado em

GPS é melhor usado quando o registro não é tão importante ou em aplicações que usam um método de rastreamento híbrido ([HOUNSELL; TORI; KIRNER, 2018](#)).

- **Sensores Acústicos:** Faz rastreamento utilizando uma técnica que leva o nome de *Time of Flight* (TOF). Isso funciona da seguinte forma: um transmissor acústico lança uma onda de ultrassom e calcula o tempo levado para onda ir e voltar até o receptor (tempo de voo), como a velocidade do som é conhecida, pode-se calcular a distância do objeto até o sensor. Ao se colocar múltiplos transmissores e receptores podemos determinar a posição e a orientação do observador. É um sensor leve, pequeno e só é sensível a ruídos que estejam na faixa de frequência do ultrassom (a cima de 20kHz). No entanto, a taxa de amostragem não é muito alta, pois depende da velocidade do som, que também é variável a alguns fatores, como pressão e umidade ([MEYER; APPLEWHITE; BIOCCHA, 1992](#)).
- **Sensores Inerciais:** Alguns exemplos de sensores inerciais são: acelerômetros, giroscópios e magnetômetros. Esse método de rastreamento utiliza esses três sensores em conjunto. O acelerômetro fica responsável por medir a posição através da aceleração que é calculada a partir da deformação de molas. Já o giroscópio e o magnetômetro calculam a orientação. Esses sensores são muito úteis por não terem limite de alcance, não são influenciados por luz ou falta de luz, não sofrem interferência de campos magnéticos e nem ruídos sonoros. O grande problema desse sensores é que eles são extremamente suscetíveis a deterioramento ao longo do tempo, produzindo erros. Geralmente os sensores inerciais são usados em conjunto com outros sensores a fim de mitigar esses erros ([BILLINGHURST; CLARK; LEE, 2015](#)).
- **Sensores Magnéticos:** São sensores que usam propriedades de campos magnéticos para calcular a posição referente a um transmissor, que funcionará como âncora. São gerados campos magnéticos com bobinas e então os sensores determinam o tamanho e a direção. Ao usar em um sistema de realidade aumentada, o gerador do campo funcionam como a origem do mundo virtual e ao colocar um receptor no HMD, é possível determinar sua posição. Eles são capazes de fazer rastreamento de algumas partes do corpo. Sensores magnéticos são pequenos, tem uma boa taxa de amostragem, não sofrem interferências ópticas. Porém funcionam a uma distância limitada e são sujeitos a ruídos causados por outros campos magnéticos (comuns em cenários industriais) ([BILLINGHURST; CLARK; LEE, 2015](#)).

A Tabela 1 apresenta uma comparação baseada em uma tabela apresentada por [Carmigniani e Furht \(2011\)](#). Na tabela, a primeira coluna tem o nome do tipo de rastreamento. Na segunda, o alcance em metros. Na terceira, o tempo necessário para preparar o sistema em horas. Na quarta, a precisão em milímetros. Na quinta, tempo em que a medida é retornada sem erro em segundos. E na sexta, ambiente onde o rastreamento pode ser usado.

Tabela 1 – Características das tecnologias de rastreamento baseado em sensores

Tecnologia	Alcance(m)	Preparo(h)	Precisão(mm)	Tempo(s)	Ambiente
GPS	∞	0	5000	∞	dentro/fora
Magnético	1	1	1	∞	dentro/fora
Inercial	1	0	1	10	dentro/fora
Ultrassom	10	1	10	∞	dentro
Híbrido	30	10	1	∞	dentro/fora

Fonte: [Carmignani e Furht \(2011\)](#).

2.3.1.2 Baseado em Visão

Segundo [Ballestin, Solari e Chessa \(2018\)](#), rastreamento baseado em visão pode ser definido como o cálculo da pose do observador através de sensores ópticos. Essa modalidade tem crescido muito em popularidade devido pequena quantidade de hardware exigida e o aumento do poder computacional dos dispositivos populares. Pode-se separar em três categorias: sensores infravermelhos, sensores de luz visível e sensores de estrutura 3D. Essas categorias serão detalhadas a seguir.

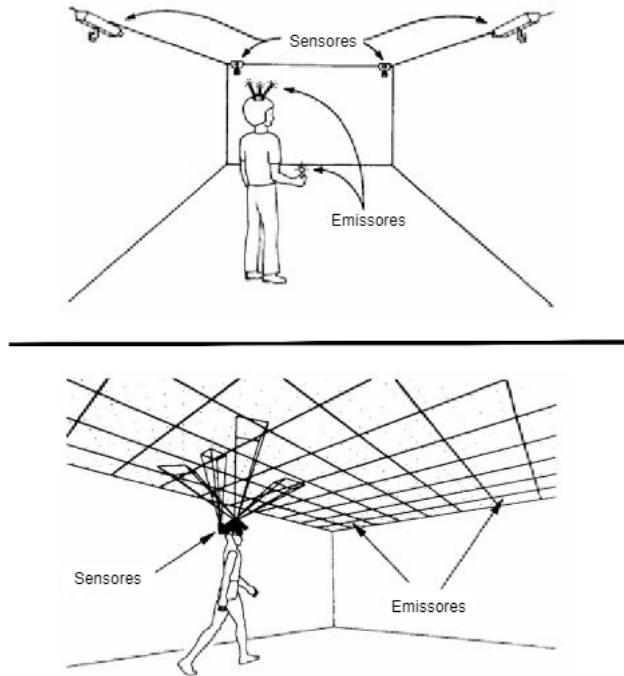
Primeiro tem-se os o rastreamento usando sensores infravermelhos, esse tipo de rastreamento é semelhante ao rastreamento utilizando sensores acústicos. São usados LEDs infravermelhos como transmissores e câmeras infravermelhos como receptores. Existindo duas abordagens: colocar os LEDs no HMD e as câmeras no ambiente (de fora para dentro) ou colocar câmeras no HMD e LEDs no ambiente (de dentro para fora), como pode ser visto na Figura 14. As configurações sensores que eram de dentro para fora tinham mais precisão que a outra, mas tendo como desvantagem o tempo necessário para montar o sistema e o peso extra no HMD pelo uso da câmera infravermelho acoplada nele. Esses tipos de rastreamento infravermelho oferecem precisão, mas são um pouco invasivos por conta da dificuldade de preparação do sistema ([BILLINGHURST; CLARK; LEE, 2015](#)).

Há uma outra possibilidade de sensores de infravermelho, que são aqueles que utilizam técnicas de *Time of Flight*, assim como sensores ultrassônicos, mas sem o problema trazidos ao usar ondas sonoras.

Outra possibilidade, que é a mais usada entre os sensores ópticos, é fazer o uso de câmeras para fazer o rastreamento baseado em luz visível. Essa modalidade é muito popular já que um sensor de luz visível nada mais é que uma câmera. Portanto é possível utilizar câmeras dos *smartphones*, *tablets*, e notebooks para fazer o rastreamento. Essa técnica pode ser dividida em três categorias: rastreamento com marcador fiducial, rastreamento de características naturais e rastreamento de estrutura 3D.

- **Rastreamento com Marcador Fiducial:** Marcadores fiduciais são pontos de referência que são colocados no ambiente, que variam de padrões impressos em folhas (marcador

Figura 14 – Abordagens do rastreamento usando sensores infravermelhos. Em cima: de fora para dentro. Embaixo: de dentro para fora.



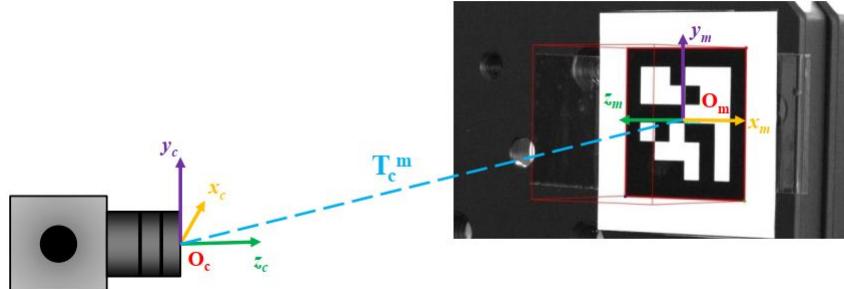
Fonte: Meyer, Applewhite e Biocca (1992).

passivo) até um grupo de LEDs (marcador ativo), que ajudam a realizar o rastreamento óptico. Os marcadores são criados na intenção de que seja de fácil reconhecimento pelo software de Visão Computacional. O grande problema do uso de marcadores é que exige a alteração do ambiente e isso nem sempre será possível e não é o ideal. O funcionamento da técnica pode ser observada na Figura 15. Na Equação 2.1, pode-se observar como o cálculo é feito, a primeira matriz é o sistema de coordenadas da câmera, a segunda é a matriz de transformação que é composta pela matriz de rotação e matriz de translação, por último a matriz que representa o sistema de coordenadas do observador (POPESCU; CERNAIANU; DUMITRACHE, 2018).

$$\begin{pmatrix} x_c \\ y_c \\ z_c \\ 1 \end{pmatrix} = \begin{pmatrix} r_{11} & r_{12} & r_{13} & T_x \\ r_{21} & r_{22} & r_{23} & T_y \\ r_{31} & r_{32} & r_{33} & T_z \\ 0 & 0 & 0 & 1 \end{pmatrix} * \begin{pmatrix} x_m \\ y_m \\ z_m \\ 1 \end{pmatrix} \quad (2.1)$$

- **Rastreamento de Características Naturais:** Como já comentado, nem sempre é possível a alteração do ambiente para o uso de algoritmos de Visão Computacional. Porém, felizmente, surgiram algoritmos que não dependem da existência de um padrão conhecido no ambiente para funcionar, tal qual o SIFT (*Scale Invariant Feature Transform*), SURF (*Speeded Up Robust Features*) e BRIEF (*Binary Robust Independent Elementary*

Figura 15 – Estimação da posição e orientação fazendo uso de um marcador fiducial. Sendo (x_c, y_c, z_c) o sistema de coordenadas da câmera, (x_m, y_m, z_m) sendo o do marcador e T_{cm} a matriz de transformação para a passagem de um pro outro.



Fonte: Popescu, Cernaianu e Dumitache (2018).

Feature). Os algoritmos tentam encontrar características a cada *frame*, como pontos, linhas, cantos e calcula um descritor com cada uma. Então são feitas comparações entre as características encontradas no ambiente com as do objeto a ser encontrado, igualando o problema a um estado semelhante ao que se tivesse um marcador no ambiente (BILLINGHURST; CLARK; LEE, 2015).

- **Rastreamento de Estrutura 3D:** Por volta de 2015, surgiram sensores infravermelhos que são capazes de identificar estruturas 3D no ambiente. São esses que usam a técnica de TOF (mencionada anteriormente) para obter informações sobre a posição e orientação de pontos no ambiente. Alguns exemplos famosos dessa tecnologia é o *Microsoft Kinect* e o *Leap Motion*. Com esses sensores é possível calcular a pose fazendo uma reconstrução 3D do ambiente usando as informações de profundidade, mas não somente isso, como também podem ser mapeados ambientes ou objetos inteiros, abrindo muitas possibilidades (como evidenciado na Figura 16) (BILLINGHURST; CLARK; LEE, 2015).

Figura 16 – Imagem RGB e sua reconstrução 3D pelo Kinect.



Fonte: Billinghamurst, Clark e Lee (2015).

Na Tabela 2 pode-se observar uma comparação semelhante a realizada com os métodos de rastreamento baseados em sensores, mas dessa vez comparando o rastreamento óptico com e sem uso de marcadores.

Tabela 2 – Características das tecnologias de rastreamento óptico.

Tecnologia	Alcance(m)	Preparo(h)	Precisão(mm)	Tempo(s)	Ambiente
C/ marcadores	10	0	10	∞	dentro/fora
S/ marcadores	50	0-1	10	∞	dentro/fora

Fonte: [Carmignani e Furht \(2011\)](#).

2.3.2 *Leap Motion*

Nessa seção, será detalhado o funcionamento e principais características do sensor *Leap Motion*, devido a sua importância para este trabalho. O sensor foi utilizado já que além de ser uma solução boa e barata para a determinação da pose do observador, como já comentado, o *Leap Motion* também é capaz de fornecer interação gestual em tempo real através do rastreamento infravermelho das mãos ([COSTA; KAYATT; BOGONI, 2018](#)).

O sensor contém duas câmeras infravermelhas e três LEDs infravermelhos, como pode ser visto na Figura 17. As câmeras infravermelho captam luz com comprimento de onda de cerca de 850nm, que é fora do espectro de luz visível. O volume de interação possível é de cerca de 0.2 m^3 que toma a forma de uma pirâmide invertida (com a ponta saindo do sensor) de 80 cm de altura, como pode ser visto na Figura 18 ([COLGAN, 2014](#)).

Figura 17 – *Leap Motion* internamente.

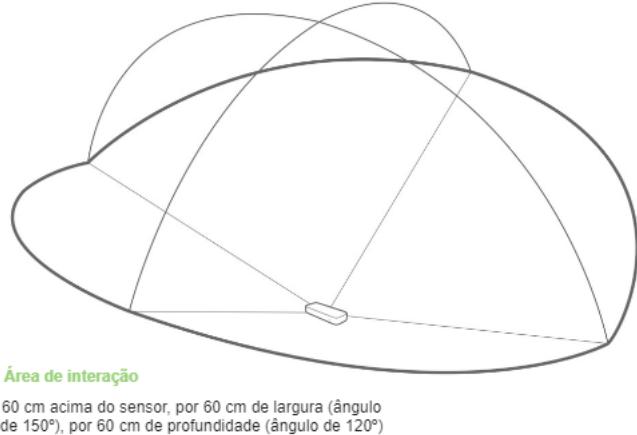


Fonte: [Colgan \(2014\)](#).

Segundo [Colgan \(2014\)](#), *Leap Motion*, ao contrário do *Kinect*, não constrói um mapa de profundidade, ele aplica algoritmos avançados nas leituras brutas do sensor. Esses cálculos

são realizados pelo suporte *Leap Motion Service* que deve ser instalado no computador e irá executar em segundo plano. O suporte então faz uma reconstrução 3D do que o sensor está vendo e extraí informações sobre os dedos, mãos e ferramentas reconhecidas pelo *Leap Motion*. Após isso, o suporte fornece as informações em forma de *frames* para a aplicação cliente utilizando o sensor.

Figura 18 – Volume de interação do *Leap Motion*.



Fonte: [Colgan \(2014\)](#).

Em resumo, o *Leap Motion* não utiliza técnicas de TOF e nem constrói mapas de profundidade como outros sensores semelhantes. Ele simplesmente captura imagens estéreas do ambiente usando seus transmissores e receptores de infravermelho a fim de fazer um rastreamento de gestos e da mão muito semelhante ao rastreamento óptico de características naturais, porém com a vantagem de trabalhar com um espectro específico de luz, o que diminui consideravelmente o ruído que deve ser tratado da imagem.

2.4 Espaço Peripessoal

Seres humanos mantém uma representação mental do espaço que imediatamente os cercam. Essa região recebe o nome de espaço peripessoal e é definida como o espaço que é alcançável pelos braços, ou seja, é a região onde os objetos podem receber uma ação do dito ser humano, como pode ser notado na Figura 19 ([HUNLEY; LOURENCO, 2018](#)).

O espaço peripessoal pode variar de pessoa para pessoa, no que diz respeito ao seu comprimento, porém em média ele se estende em cerca de 1m. Porém, além disso, entra o espaço extrapessoal que vai de 1m até 30m. A função do espaço extrapessoal é fornecer informações sobre navegação e localização, enquanto que o espaço peripessoal é focado em tarefas precisas. Vale salientar, como já mencionado anteriormente, que as pistas de profundidade agem de maneira diferente dependendo da distância, portanto algumas pistas vão funcionar melhor no espaço peripessoal do que no extrapessoal. Isso é bem claro com as pistas binoculares

Figura 19 – Representação do espaço peripessoal.

PPS = Espaço Peripessoal



Fonte: [Amemiya \(2020\)](#).

e oculomotoras, que não surtem tanto efeito no espaço extrapessoal ([NACERI; CHELLALI; HOINVILLE, 2011](#)).

Enfim, estudar a percepção no espaço peripessoal é de suma importância, visto que a maior parte das interações que o ser humano realiza ocorre nesse espaço. Então se faz necessária a investigação de como o uso de HMDs e outras tecnologias de RA afetam a interação nesse espaço.

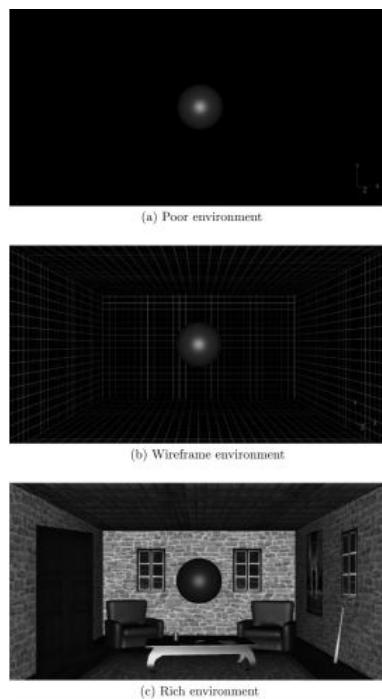
3 Trabalhos Correlatos

O presente capítulo apresenta os principais trabalhos encontrados na literatura científica, a fim de estabelecer o estado da arte no que diz respeito a interação e percepção no uso de HMDs em ambientes de RA e RV.

3.1 Método de Naceri, Chellali e Hoinville

O trabalho de [Naceri, Chellali e Hoinville \(2011\)](#) propõe medir a percepção de profundidade no espaço peripessoal em três ambientes distintos de Realidade Virtual (Figura 20): uma sala escura, uma sala com somente *wireframes* (contornos das paredes, não possui texturas) e uma sala rica com objetos, iluminação, texturas e sombras. A medição era feita através de uma tarefa de apontar com característica malha aberta (*open loop*), que não apresenta *feedback* para o usuário. Na tarefa, uma esfera era apresentada ao usuário, e ele deveria estimar sua posição apontando para ela.

Figura 20 – Ambientes virtuais de teste. (a) Sala Escura. (b) Sala com *wireframe*. (c) Sala rica em detalhes.



Fonte: [Naceri, Chellali e Hoinville \(2011\)](#).

Foi usado o HMD *Cybermind Visette45 SXGA*, de RV com 45° de FOV e resolução de 1280 × 1024. Para o rastreamento foi utilizado um sistema de rastreamento óptico infravermelho

de abordagem de fora para dentro (câmeras no ambiente e LEDs no HMD), chamado *WorldViz PPT*.

Para que a posição da mão apontando para o alvo fosse capturada, foi colocado um marcador na ponta do dedo do usuário. Os participantes sentavam em uma cadeira no centro da sala e um aparato fixava a posição da sua cabeça, mostrado na Figura 21.

Figura 21 – Aparato usado no trabalho de [Naceri, Chellali e Hoinville \(2011\)](#).



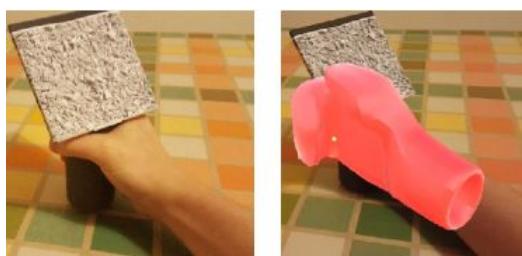
Fonte: [Naceri, Chellali e Hoinville \(2011\)](#).

De acordo com os testes feitos, os usuários erravam menos no ambiente rico em detalhes.

3.2 Método de Ballestin, Solari e Chessa

O projeto de [Ballestin, Solari e Chessa \(2018\)](#) foi realizada uma comparação entre dois HMDs de RA, um VST e o outro OST, visando medir a percepção no espaço peripessoal. Semelhante ao trabalho apresentado anteriormente, os usuários deveriam posicionar a mão no ponto em que eles perceberam o alvo. Outro fator analisado pelo estudo, foi se a presença ou ausência de um *feedback* (posição da mão virtual) para o usuário afetaria o resultado de alguma forma. Na Figura 22, pode ser visto como o *feedback* da mão era mostrado. A estrutura na mão do usuário é um marcador reconhecido pelo sistema.

Figura 22 – *Feedback* da mão virtual.

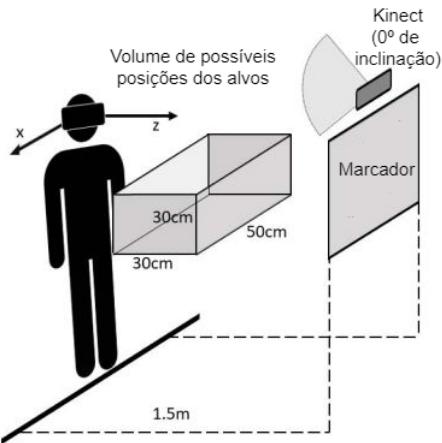


Fonte: [Ballestin, Solari e Chessa \(2018\)](#).

Para o HMD OST foi escolhido o *Metavision Meta* (90° de FOV e 2560x1440 de resolução) e para o HMD VST, o *Google Cardboard* (com aproximadamente 95° de FOV e

com a resolução do *display* do *smartphone galaxy S6* de 2560x1440). O rastreamento baseado em luz visível foi utilizado, visto que um marcador fiducial plano era situado em uma parede em frente ao usuário. Ao passo que para poder rastrear a mão do usuário eram usados dois sistemas: o aparato mostrado na Figura 22 e o sensor RGB-D *Kinect*. As duas medidas eram então comparadas, sendo que a do *Kinect* era usado como referência (*ground-truth*). A disposição dos componentes é mostrada na Figura 23.

Figura 23 – Disposição dos componentes do projeto de [Ballestin, Solari e Chessa \(2018\)](#).



Fonte: [Ballestin, Solari e Chessa \(2018\)](#).

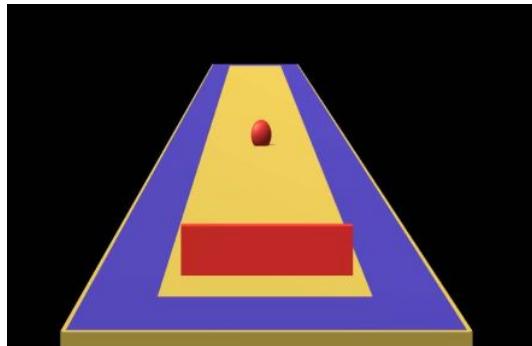
O estudo conclui, após a análise dos experimentos, que a estimativa de profundidade acabou tendo resultados melhores em testes que utilizaram o HMD OST. Com relação ao *feedback*, no HMD VST houve uma aparente melhora na introdução do mesmo, enquanto que no HMD OST, não houve diferença significativa nos experimentos com e sem *feedback*.

3.3 Método de Ping, Liu e Weng

O projeto de [Ping, Liu e Weng \(2019\)](#) realiza uma comparação, com relação a percepção de profundidade, entre ambientes de RA e RV usando HMDs. Para medir a percepção foram criados ambientes virtuais que simulam uma mesa de *shuffleboard*, um jogo em que o participante arremessa um disco de forma que ele pare em uma área triangular no fim da pista. No projeto, o usuário ficava a um metro de uma mesa virtual no caso do RV (Figura 24) e real no caso de RA, então eram posicionadas determinadas bolas e o usuário controlava uma barra virtual, através do teclado, e tentava colocá-la na posição das bolas.

Os HMDs selecionados para realização dos experimentos foram: o NED+ para RA, sendo um HMD OST com 45° de FOV e 1280x1920 de resolução, e para RV foi usado o mesmo HMD, mas com um pano preto para simular a imersão dos ambientes de RV. Não houve nenhuma forma de rastreamento das mãos, o ambiente era controlado através do teclado. Foi notado que os usuários cometiam menos erros no ambiente de RA.

Figura 24 – Mesa Virtual de *Shuffleboard*.



Fonte: [Ping, Liu e Weng \(2019\)](#).

3.4 Método de Batmaz, Machuca, Pham e Stuerzlinger

Assim como o projeto da Seção 3.3, o projeto de [Batmaz et al. \(2019\)](#) é proposto uma comparação entre HMDs de RA e RV, no que diz respeito a medição da percepção no espaço peripessoal. O método utilizado para as medidas foi fazer um teste basado na Lei de Fitts, que envolve contar a quantidade de tempo para o usuário apontar para um alvo. Em questão de hardware, o HMD de RV usado foi o *HTC Vive Pro headset* que tem aproximadamente 110° de FOV e 2880x1600 de resolução. Já para o ambiente de RA, o HMD foi o *Meta AR headset*, com 90° de FOV e 2560x1440 de resolução. Apesar desses dois HMDs possuírem sistemas de rastreamento próprios, foi empregado um sistema externo único, a fim de garantir a uniformidade entre as medidas. O sistema utilizado foi um de rastreamento óptico infravermelho chamado *OptiTrack S250e*, que consiste em um grupo de câmeras infravermelhas que foram posicionadas acima da cabeça do usuário. Para o rastreamento ambos da cabeça e das mãos, foram colocados marcadores nos HMDs e nos capacetes. Todo a configuração do sistema pode ser observada na Figura 25

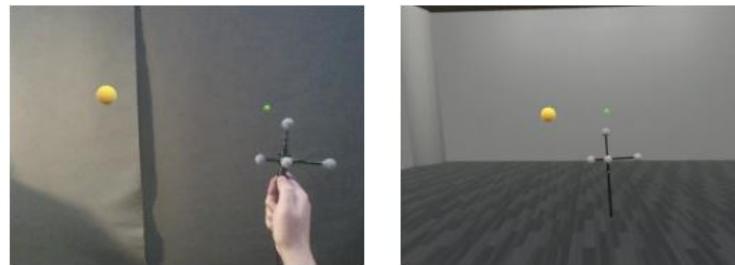
Figura 25 – Configuração em hardware de [Batmaz et al. \(2019\)](#). À esquerda HMD de RA. À direita HMD de RV.



Fonte: [Batmaz et al. \(2019\)](#).

Como pode ser visto na Figura 26 para os testes em RV, foi criado um ambiente virtual completo, incluindo uma réplica 3D do marcador segurado pelo usuário, enquanto que em RA, somente as esferas (alvos) eram renderizados para o usuário.

Figura 26 – Cenas do projeto [Batmaz et al. \(2019\)](#). À esquerda cena aumentada de RA. À direita cena virtual de RV.



Fonte: [Batmaz et al. \(2019\)](#).

Os resultados obtidos indicaram que, embora não tenha havido uma diferença grande em questão de precisão entre os dois ambientes dos usuários ao realizar os testes, houve uma clara vantagem em questão de velocidade no ambiente de RA em relação com o de RV.

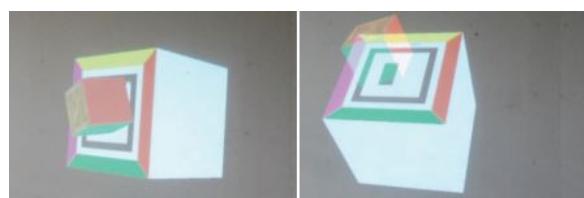
3.5 Método de Plopski, Moser, Kiyokawa, Swan e Takemura

De acordo com [Plopski et al. \(2016\)](#), não há como alcançar um alinhamento perfeito entre os objetos virtuais e os objetos reais na cena, então o trabalho se propõe a fazer uma análise de como os possíveis erros afetam a percepção dos usuários. Porém, ao invés de usar HMDs de RA para fazer os experimentos, foi construído um ambiente virtual que simula uma aplicação de RA, a fim de que os autores tenham maior controle sobre os erros. Para fazer essa simulação, eram usados dois projetores *SANYO PDG-DWL2500J*, com resolução de 1280x800 e brilho de 2500 lúmens.

Para a simulação do HMD VST, duas imagens eram projetadas em uma mesma parede, em sobreposição uma a outra. Uma imagem representaria o marcador e a outra, o objeto virtual. Na Figura 27, à esquerda, pode ser visto o resultado.

Já para a simulação do HMD OST, a mesma ideia era empregada, com a diferença de acrescentar uma transparência no objeto virtual, que geralmente é vista em HMDs OST. Na Figura 27, à direita, é mostrado o resultado.

Figura 27 – Simulação dos HMDs de RA.



Fonte: [Plopski et al. \(2016\)](#).

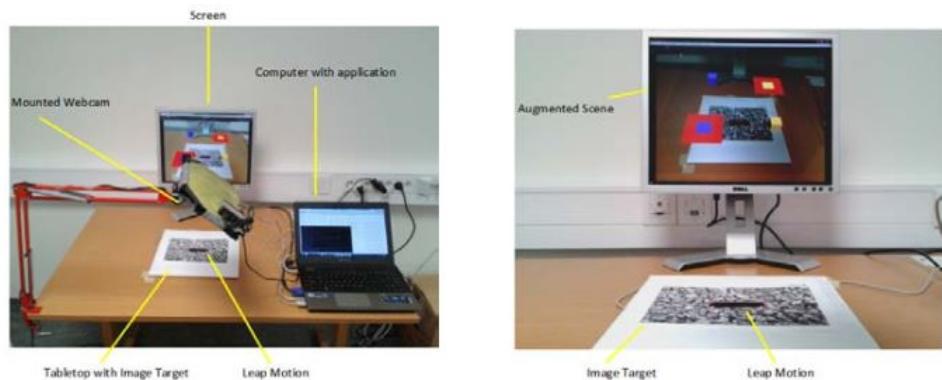
Os resultados dos experimentos, indicaram que os usuários são menos perceptivos aos

erros de rotação em geral e que erros em translação são menos importantes em sistemas OST.

3.6 Método de Jiménez

O projeto de [Jiménez \(2014\)](#) compromete-se a implementar um sistema de RA capaz de realizar interações com os objetos virtuais através de uma interface de gestos baseada em sensores RGB-D. O sistema de RA é de visão indireta baseado em monitores e o rastreamento é óptico baseado em luz visível. Uma câmera era posicionada acima do usuário para que ela pudesse capturar o ambiente e realizar o rastreamento do marcador. Para o reconhecimento dos gestos, foi usado o *Leap Motion*, apresentado em [2.3.2](#). A configuração do ambiente pode ser vista na Figura [28](#).

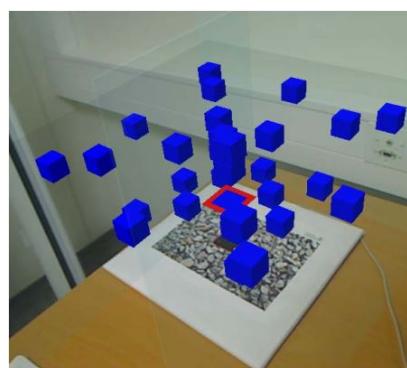
Figura 28 – Configuração física de [Jiménez \(2014\)](#).



Fonte: [Jiménez \(2014\)](#).

Foram construídos alguns testes diferentes para medir como a interação para testar o ambiente experimental construído. Os experimentos eram baseados em tarefas de pegar, arrastar e soltar cubos em diferentes alvos e configurações diferentes. A Figura [29](#) mostra um exemplo de um desses experimentos.

Figura 29 – Exemplo de experimento de [Jiménez \(2014\)](#).



Fonte: [Jiménez \(2014\)](#).

4 Materiais e Métodos

Neste capítulo são apresentados os principais requisitos do projeto, bem como o detalhamento de seu desenvolvimento, no que diz respeito à sua arquitetura lógica, física e materiais utilizados.

4.1 Principais requisitos do projeto

Como apresentado na Seção 1.1, são escassas na literatura as informações sobre o desenvolvimento de sistemas de RA para HMDs Video See-Through baseados em *smartphones*, principalmente no que diz respeito a interação e percepção de profundidade no espaço peripessoal. Sendo assim, o objetivo do projeto consistiu em estruturar e implementar um sistema de RA que viabilizasse a avaliação do uso desse tipo de HMD, no que tange a interação por meio de gestos e considerando o espaço peripessoal do usuário.

Nas próximas subseções, os requisitos de desenvolvimento serão discutidos com maior detalhamento, tendo como base a contextualização exposta.

4.1.1 Registro

Retornando a Seção 2.1.2 na definição de [Azuma](#), uma aplicação de RA necessita atender certos requisitos para que o aumento da realidade seja atingido. Logo, o sistema desenvolvido precisa executar em tempo real e combinar os objetos virtuais e reais de maneira coerente. Para que isso seja feito, o sistema deve ser capaz de realizar o registro dos elementos virtuais, o que envolve o uso de técnicas de rastreamento 3D, conforme discutido no Capítulo 2.

Segundo [Jiménez \(2014\)](#), para que haja interação direta com objetos virtuais é necessário que seja usado um rastreamento com 6 DOF. E já que o objetivo é criar uma aplicação RA *mobile*, a câmera do *smartphone*, que já serviria para dar a visão do mundo real ao usuário, também pode ser a responsável pelo rastreamento, tornando o Rastreamento Óptico Baseado em Luz Visível o mais ideal, como discutido na Seção 2.3.

4.1.2 Interação

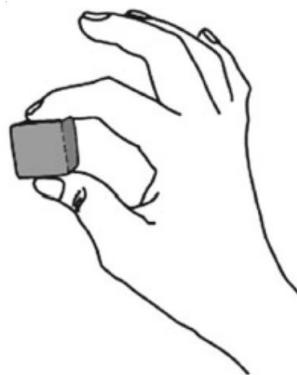
Como discutido no Capítulo 1, uma aplicação de RA pode fazer uso de diferentes modalidades de interface, por isso é importante que a diferença entre os objetos virtuais e reais seja mínima, sendo interessante que se busque o uso de interações mais naturais, como gestos, ao invés do uso de teclados convencionais, por exemplo. Portanto, é importante que, além do

rastreamento da pose da cabeça do usuário, também sejam rastreados os movimentos e gestos de sua mão.

Nas Subseções [2.3.1.2](#) e [2.3.2](#), foram descritos sensores de profundidade (câmeras RGB-D) que são capazes de capturar a movimentação da mão e interpretar os seus gestos. Porém, tais sensores não possuem integração direta com *smartphones*, o que precisou ser considerado na elaboração do presente projeto.

Um gesto pode ser definido como um movimento do corpo que carrega algum tipo de significado. O grande problema de tentar interpretar gestos computacionalmente é que não há um único significado universal para eles, ou seja, para algumas pessoas podem representar algo e para outras não. Entretanto, podem ser encontrados gestos com significados de fácil entendimento e grande difusão em interfaces digitais, como o gesto de pinça. Apesar do uso da pinça para aproximação (*zoom*) em outras sistemas, no mundo real fazemos esse movimento para pegar e mover objetos (Figura 30). O gesto de pinça é de fácil reconhecimento computacional, já que há pouca ambiguidade gerada quando o dedão encosta no dedo indicador, fazendo da pinça um bom candidato para a metáfora da interação com objetos sintéticos ([JIMÉNEZ, 2014](#)).

Figura 30 – Gesto de pinça agarrando um objeto.



Fonte: [Kita et al. \(2013\)](#).

4.1.3 Experimentos

Vale recordar que um dos propósitos do projeto era adquirir medidas quantitativas através de cenários de testes, que colocassem a percepção e a capacidade de interação do usuário a prova. Logo, além de ser capaz de reconhecer gestos do usuário, o sistema também deve coletar medidas do quanto preciso o usuário consegue ser durante essas interações. Mais detalhes de como os experimentos foram formulados e aplicados serão discutidos mais adiante.

4.2 Arquitetura Geral do Sistema

O sistema foi construído segundo uma arquitetura modular. Essa arquitetura divide o sistema em módulos, que são agrupamentos de componentes que cumprem uma determinada função. Os módulos são dependentes entre si e comunicam-se através de interfaces. Tal modalidade de arquitetura foi adotada a fim de facilitar o desenvolvimento do projeto, pois assim as funções poderiam ser estruturadas, implementadas, e reutilizadas separadamente. Com isso a manutenção também é facilitada.

Como mencionado anteriormente, a maioria dos sensores de profundidade não foi projetada para ser acoplada diretamente a dispositivos móveis. Mesmo que tal acoplamento pudesse ser realizado, os requisitos em termos de consumo de energia inviabilizaria seu uso prático. Tal situação entrou em conflito com a proposta do projeto que é fundamentada no uso de HMDs baseados nesses dispositivos. A solução adotada foi a criação de duas aplicações, uma que executasse em um computador e outra no *smartphone*. A aplicação no computador é responsável por coletar as leituras do sensor e enviá-las à outra aplicação. Essa solução, apesar de afetar a mobilidade do sistema, não fere o seu propósito, que é testar a interação e percepção no espaço peripessoal.

4.2.1 Materiais

Considerando tais requisitos, foi feita a escolha das seguintes ferramentas:

Software

- Sistema Operacionais: Windows 10 64bit (PC) e *Android* (*smartphone*);
- Ambiente de desenvolvimento: *Microsoft Visual Studio 2019*;
- Motor de jogos: *Unity3D*;
- Linguagem de programação: C#;
- Biblioteca de RA: *Vuforia*;
- Ferramentas de modelagem/reconstrução 3D: *Blender*.

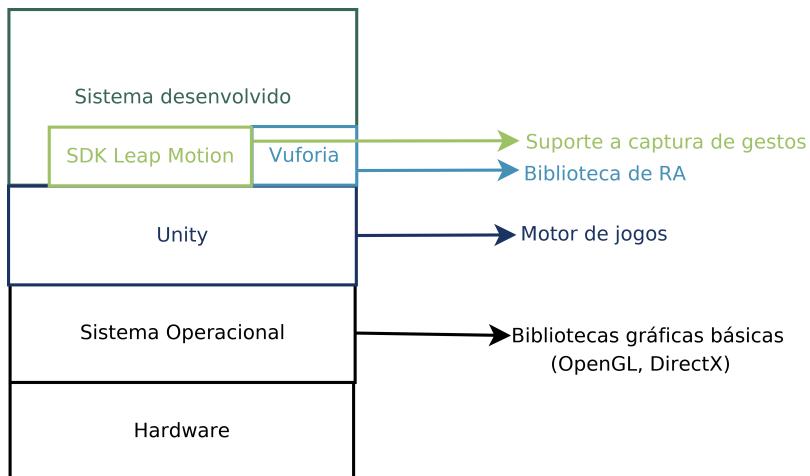
Hardware

- Computador pessoal com: processador *Intel Core i5-7200U* CPU; disco rígido de 1TB; 8.0 GB DDR4 de Memória RAM; monitor LED de 15.6" de alta resolução;
- *smartphone* com sistema operacional *Android*;
- HMD VST: *VR BOX 2*;

- Sensor RGB-D: *Leap Motion*.

Com a intenção de ilustrar o estado da estrutura da aplicação em um ponto de vista de vista de software, tem-se a Figura 31. No nível mais baixo tem-se o hardware interagindo com o sistema operacional que contém as bibliotecas gráficas básicas que são responsáveis pelo pipeline gráfico, como o *DirectX* e *OpenGL*, portanto elas tratam somente de operações mais básicas. Apesar de elas possuírem maior flexibilidade, seria muito trabalhoso usar elas diretamente. Por isso são usadas ferramentas que auxiliam na organização do ambiente virtual, como o *Unity*, os quais são conhecidos como editores gráficos ou motores de jogos. Eles são responsáveis pela organização do ambiente virtual através de uma estrutura hierárquica chamada grafo de cena. O *Unity* também facilita a interação com bibliotecas de Realidade Aumentada, tais como o *Vuforia*. O *Vuforia* possui algoritmos prontos de registro e rastreamento, que são de suma importância para o desenvolvimento de aplicações de RA. Por sua vez, o SDK do *Leap Motion* fica encarregado de obter as leituras do sensor para que se possa fazer o reconhecimento dos gestos.

Figura 31 – Estrutura hierárquica de software.



Fonte: Elaborada pelo autor.

O sistema desenvolvido nesse trabalho baseia-se no uso das camadas abaixo dele nessa hierarquia. Os detalhes sobre estas ferramentas são apresentados nas subseções seguintes.

4.2.1.1 *Unity*

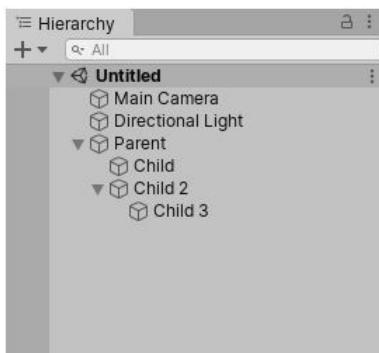
Unity é uma plataforma de desenvolvimento de aplicações interativas em tempo real, muito utilizado na criação de jogos digitais. Graças a isso, ele conta com uma grande comunidade de desenvolvedores e pacotes gratuitos. Além disso é compatível com múltiplas linguagens como: C#, *UnityScript* (baseado em *JavaScript*) e *Boo* (baseado em *Python*).

Uma grande vantagem do *Unity* é sua versatilidade de desenvolvimento devido a capacidade de exportar seus projetos para múltiplas plataformas com extrema facilidade (incluindo

Android), compatibilidade com softwares de modelagem como o *Blender* e compatibilidade com diversos formatos de arquivo (imagem, audio e vídeo) ([UNITY, 2020](#)).

O *Unity* tem a importante função de organizar o ambiente virtual através de uma estrutura hierárquica. No editor, essa estrutura é representada pelo Painel de Hierarquia (Figura 32), que pode ser manipulada livremente. Essa hierarquia tem diversas funções ao se desenvolver uma aplicação, como auxiliar no *pipeline* gráfico, organizar a cena de forma coesa e juntar objetos para que eles se movam como um. Na Figura 32 por exemplo, se o objeto chamado *Parent* se mover, todos os seus objetos filhos (estão abaixo dele na hierarquia) se moverão com ele.

Figura 32 – Painel de Hierarquia.



Fonte: [Unity \(2019a\)](#).

Toda hierarquia de uma cena no *Unity* possui um grafo de cena correspondente. O grafo da Figura 33 é referente a cena cujo Painel de Hierarquia foi apresentado na Figura 32. Cada nó é um *GameObject* e cada aresta indica a relação de parentesco entre dois objetos.

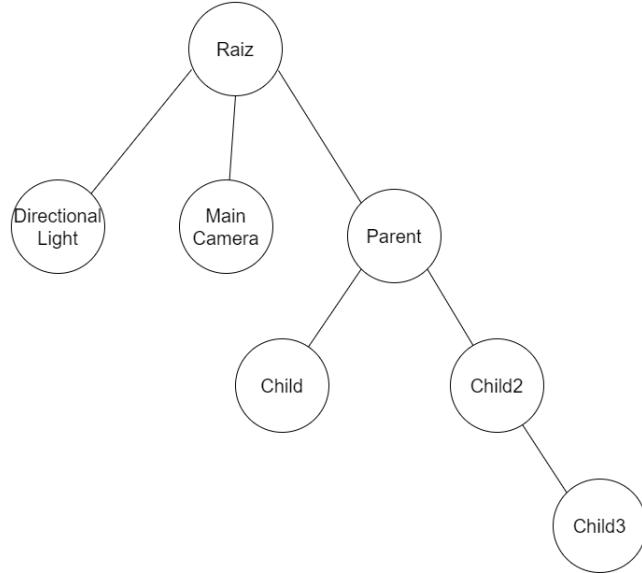
Vale discutir ainda sobre conceito de *GameObjects* e de componentes dentro do *Unity*. Todos os objetos dentro de uma cena são considerados *GameObjects*, e eles recebem os componentes e *scripts*, os quais os dotam de suas propriedades específicas. Já os componentes são os agrupamentos de funcionalidades e parâmetros que definem o que o *GameObject* faz. Esses componentes podem ser uma transformada (posição, rotação e escala), colisores, simuladores de física, *scripts* e muitos outros. Um *GameObject* vazio só possui a transformada como componente.

Os componentes de um *GameObject* podem ser observados na janela do Inspetor, ilustrado pela Figura 34.

4.2.1.2 Vuforia

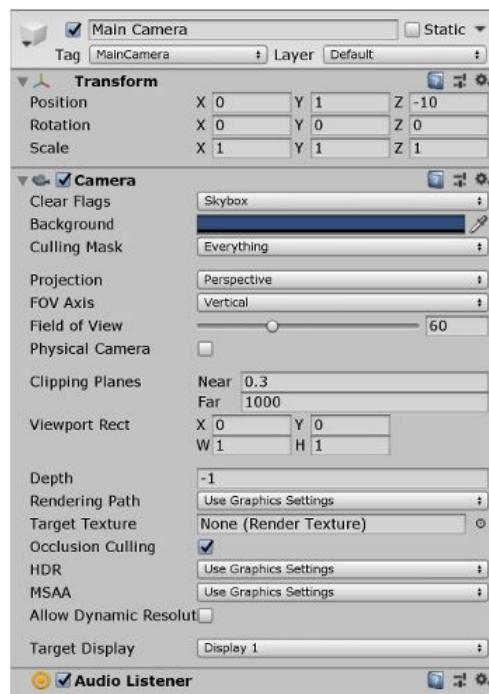
Uma outra grande vantagem de utilizar o *Unity* é a extrema facilidade de integração com inúmeras bibliotecas de RA. Usar o *ARCore* com o *Unity*, por exemplo, é tão fácil quanto importar uma textura. E com o *Vuforia* isso é mais facilitado ainda, visto que é preciso somente

Figura 33 – Exemplo de um grafo de Cena.



Fonte: Elaborada pelo autor.

Figura 34 – Inspetor mostrando informações do *GameObject* da Câmera virtual.

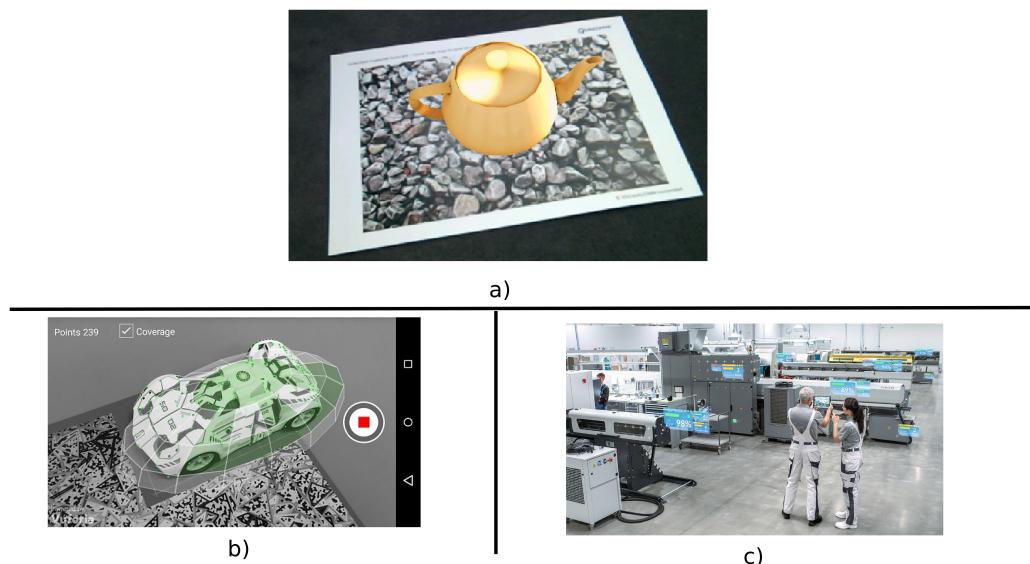


Fonte: [Unity \(2019b\)](#).

ativar uma configuração dentro do próprio *Unity* que seu pacote é automaticamente instalado e fica pronto para uso.

Vuforia é o SDK mais utilizado de realidade aumentada e possui essa íntima relação com o motor gráfico escolhido, então é uma ótima alternativa. Ele também gera conteúdos para múltiplas plataformas com extrema facilidade e possui muitos algoritmos de registro de imagem. O SDK é capaz reconhecer marcadores fiduciais em formato de imagens (Figura 35 (a)) e modelos 3D (Figura 35 (b)). Também consegue fazer o rastreamento de planos e fazer mapeamento de ambientes 3D (Figura 35 (c)) ([VUFORIA, 2020](#)).

Figura 35 – Exemplos de marcadores utilizados pelo SDK da Vuforia. (a) Marcador Imagem. (b) Marcador modelo 3D. (c) Mapeamento de ambientes 3D.



Fonte: [Vuforia \(2020\)](#).

Outra vantagem de usar o *Vuforia* é a facilidade de exportar aplicações para HMDs, sejam eles VST ou OST. Algo raro de se ver em outras bibliotecas de RA compatíveis com o *Unity*.

De acordo com [Ibañez e Figueras \(2013\)](#), o SDK é dividido nos seguintes elementos:

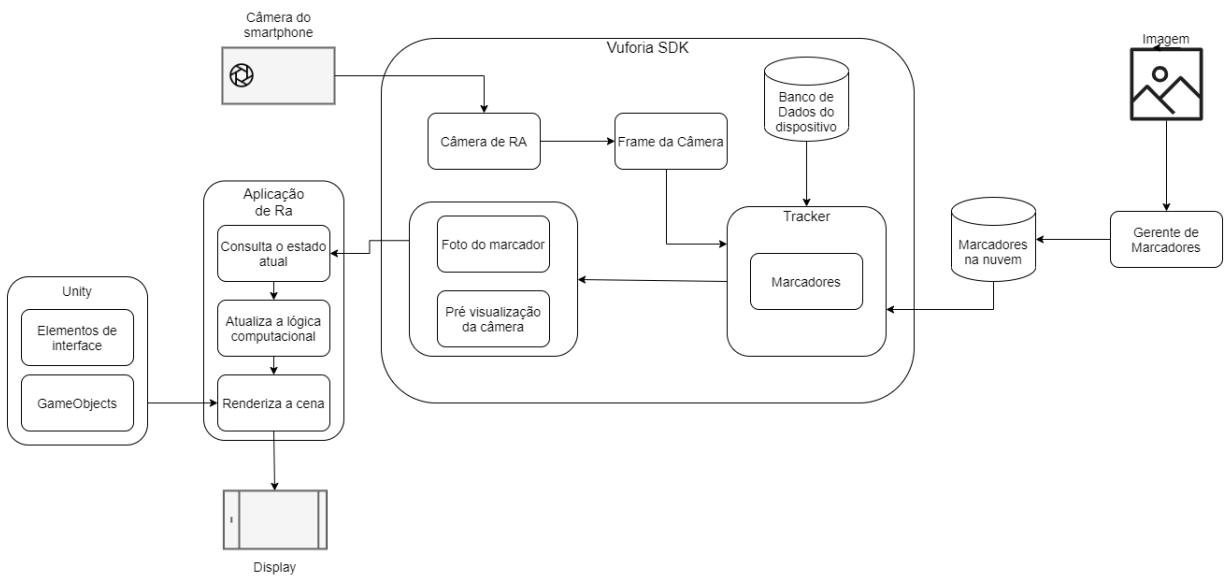
- **Câmera:** Captura a cena e garante que o *frame* será transmitido para passar pelo processo de rastreamento. O programador somente se preocupa em dizer quando a câmera começa e termina a captura.
- **Conversor de Imagem:** Realiza a conversão da imagem para um formato aceito pelo *OpenGL* e pelos algoritmos de rastreamento. Também é responsável por fazer a compressão do arquivo se necessário.
- **Tracker:** Unidade do SDK que é responsável pelo processo de rastreamento. Contém algoritmos de Visão Computacional que irão rastrear os marcadores e objetos na cena.

Os resultados são armazenados no objeto de estado que serão usados pelo Renderizador de *Background* e podem ser acessados pelo código da aplicação.

- **Renderizador de *Background*:** Responsável por renderizar a imagem da câmera que está armazenada no objeto resultado do *Tracker*.
- **Aplicação:** A cada *frame* o objeto de estado é atualizado e é dever da aplicação realizar as seguintes etapas: procurar no objeto de estado por marcadores recém detectados ou por mudanças de estado nos outros; atualizar a lógica da aplicação e renderizar os objetos sintéticos. No caso desse trabalho, isso será realizado pelo *Unity*.
- **Gerenciador de Alvos:** Trata-se da base de dados com os marcadores que o programador pode customizar.

A Figura 36 apresenta o *pipeline* da biblioteca do *Vuforia*.

Figura 36 – *Pipeline* do *Vuforia*.



Fonte: Baseado em Alonso-Rosa et al. (2020).

4.2.1.3 HMD VST VR BOX 2

VR BOX 2 é baseado no *Google CardBoard* e pode funcionar como um HMD de RV ou como um HMD VST de RA (ao abrir seu painel deslizante). Imagem dele pode ser vista na Figura 2 (a). Assim como o *CardBoard*, é uma alternativa de baixo custo aos HMDs, porém tem mais algumas vantagens como poder ajustar a posição da lente e ajustar a distância interpupilar (IPD). Também possui um melhor acabamento em plástico ABS e almofadas, ao invés do papelão do *Cardboard*. Imagens do HMD, podem ser vistas na Figura 37.

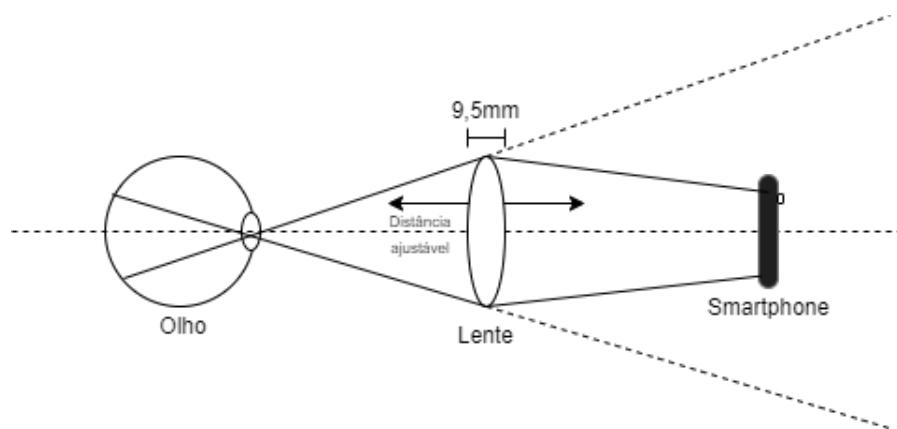
Figura 37 – Imagens do VRBOX 2. (a) Visão lateral. (b) Visão traseira.



Fonte: Elaborado pelo autor.

Quanto a sua arquitetura interna, o HMD possui somente uma lente de 9,5mm espessura e comprimento focal de 55mm. A distância interpupilar pode ser alterada em uma faixa de 58 a 72mm. O FOV admitido é de 80° (horizontal). E o peso é de 330g. A arquitetura interna pode vista na Figura 38, possui somente uma lente para cada olho cuja função é aumentar a visão da tela do *smartphone*. O *smartphone* utilizado com o HMD foi o *Galaxy S9*, que possui 1440 x 2960 de resolução em seu *display* e uma câmera 4K.

Figura 38 – Arquitetura interna do VRBOX 2.



Fonte: Elaborado pelo autor.

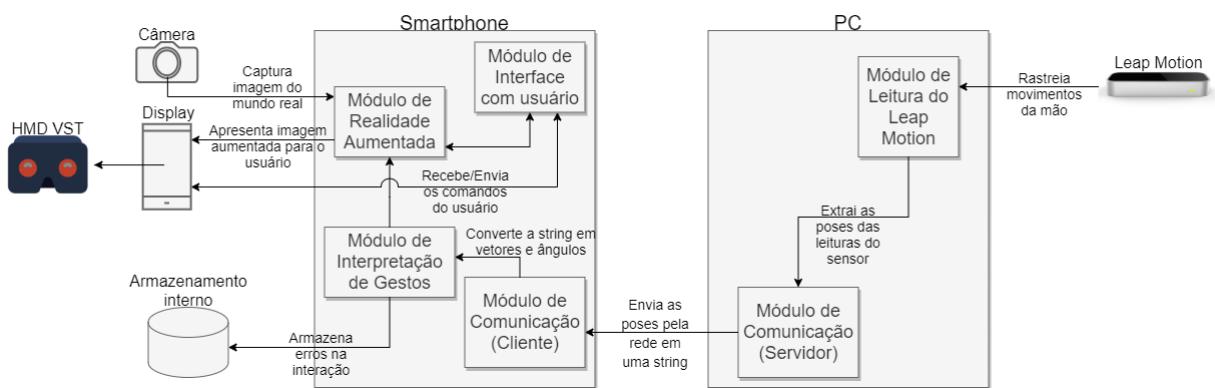
4.2.1.4 Leap Motion

Como já mencionado anteriormente, o *Leap Motion* é uma opção para o rastreio preciso do movimento das mãos, por isso se torna um ótima escolha para o projeto. Ademais, o *Leap Motion* também possui funções prontas para o reconhecimento do gesto de pinça, que é uma ótima metáfora para interação com objetos 3D em RA (Seção 4.1.2).

4.2.2 Arquitetura Lógica

O sistema desenvolvido nesse trabalho foi dividido em módulos, porém devido a natureza do problema, existem duas aplicações separadas que cumprem funções distintas e portanto são compostos por módulos diferentes. Os que constituem a aplicação do PC são: Módulo de Leitura do *Leap Motion*, Módulo de Comunicação (Servidor). Já os do *smartphone* são: Módulo de Comunicação (Cliente), Módulo de Interpretação de Gestos, Módulo de Interface com Usuário e Módulo de Realidade Aumentada. A Arquitetura Lógica do sistema está ilustrado pela Figura 39.

Figura 39 – Arquitetura Lógica do sistema dividida em módulos.



Fonte: Elaborado pelo autor.

As funções principais de cada módulo, bem como as relações entre eles, são descritas a seguir.

- **Módulo de Leitura do *Leap Motion*:** É o módulo responsável por coletar as leituras do sensor de profundidade que são fornecidas através da aplicação *Leap Motion Services* que roda em segundo plano, tal aplicação faz parte do SDK do *Leap Motion*. Após coletar as leituras, ele as armazena em uma cadeia de caracteres que será acessado pelo Módulo de Comunicação (Servidor).
- **Módulo de Comunicação (Servidor):** Esse módulo é encarregado de estabelecer uma conexão pela rede com o *smartphone* e transmitir a ele (na forma de uma cadeia de caracteres), o rastreamento dos gestos obtidos do Módulo de Leitura do *Leap Motion*.
- **Módulo de Comunicação (Cliente):** Recebe a cadeia de caracteres relativa ao rastreamento de gestos e a transforma em poses (posições e orientações) para as partes do modelo da mão. Também é o responsável por atualizar a pose do modelo da mão.
- **Módulo de Interpretação de Gestos:** Analisa a cena e verifica se houve alguma colisão entre os objetos virtuais e o modelo da mão. Além disso também verifica se o gesto de

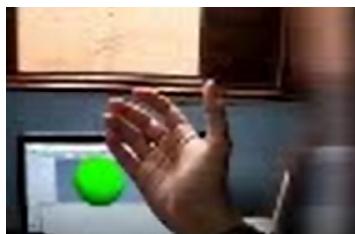
pinça foi realizado. Também armazena as medidas dos experimentos em arquivos de texto na memória interna do *smartphone*, conforme será detalhado no próximo capítulo.

- **Módulo de Realidade Aumentada:** Tem a função de fazer o rastreamento da cena real através das imagens recebidas da câmera do *smartphone* e o registro dos objetos reais e virtuais de maneira coerente. Além disso, também apresenta a imagem aumentada para o *display* do *smartphone* do usuário. Tem acesso as poses do modelo da mão fornecidos pelo Módulo de Interpretação de Gestos para que possa colocar o modelo corretamente no *frame* aumentado.
- **Módulo de Interface com Usuário:** Módulo responsável por fazer o controle das cenas virtuais através da entrada do usuário no *display touch* do *smartphone*, enquanto no menu principal da aplicação.

4.2.3 Arquitetura Física

Ao se usar a câmera para fazer o rastreamento da cabeça do usuário e o *Leap Motion*, outro sensor separado, para fazer o rastreamento das mãos é criado um problema. A diferença do ponto de vista dos dois sensores gera um erro ao se colocar o modelo da mão na cena aumentada. Esse efeito pode ser observado na Figura 40. Era então necessária a combinação do sistema de coordenadas da câmera e do *Leap Motion*.

Figura 40 – Esfera verde que deveria estar na palma da mão, mas aparece deslocada.

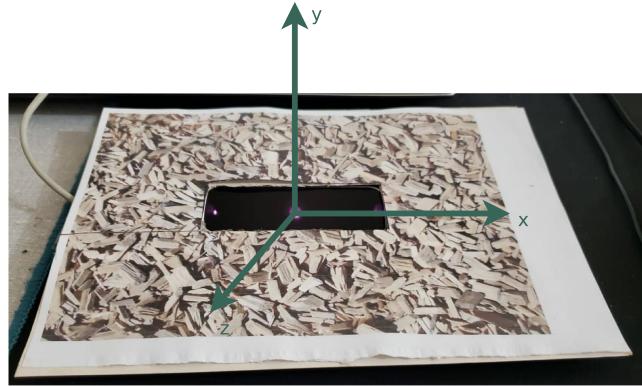


Fonte: Elaborado pelo autor.

A solução adaptada de Jiménez (2014) foi posicionar o sensor no centro de um marcador fiducial reconhecido pelo *Vuforia* (Figura 41), combinando o sistema de coordenadas do marcador e do *Leap Motion*. E então, ao fazer o centro do mundo virtual ser o centro do sistema de coordenadas do marcador, tem-se, também, o alinhamento do sistema de coordenadas do *Leap Motion*.

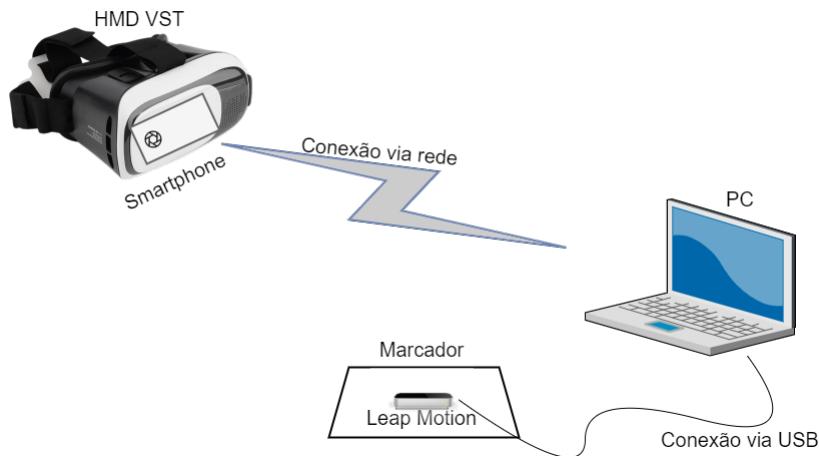
Em relação ao hardware, o sistema está organizado como pode ser visto na Figura 42. O PC está conectado ao *Leap Motion* através de um USB 3.0 devido ao fato de quanto maior energia é alimentada ao sensor, mais luz o LED infravermelho será capaz de emitir. O *smartphone* está fisicamente dentro do HMD VST (VR BOX 2) e está conectado via rede com o PC. E por fim, o *Leap Motion* é posicionado no centro do marcador para fazer a combinação do sistema de coordenadas.

Figura 41 – *Leap Motion* no centro do marcador fiducial.



Fonte: Adaptado de Jiménez (2014).

Figura 42 – Arquitetura Física do sistema.



Fonte: Elaborado pelo autor.

4.3 Implementação do Sistema

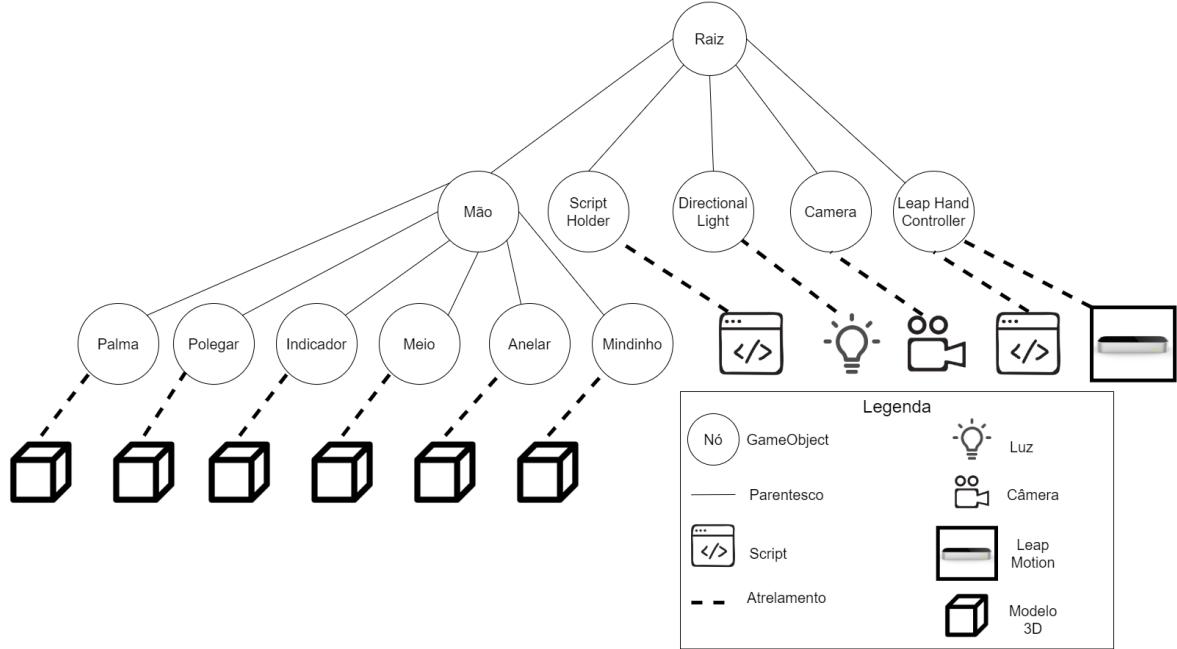
Uma forma de se dar detalhes da implementação de um projeto em *Unity* é criar uma versão gráfica do grafo de cena, pois só com isso já é possível inferir a maioria das informações sobre as cenas que compõem o ambiente aumentado. Portanto, serão apresentados os principais grafos de cena que compõem a base do sistema implementado.

4.3.1 Grafos de Cena

4.3.1.1 *Leap Server*

Primeiramente, na Figura 43, é mostrado o grafo da principal cena da aplicação executada no PC. Essa cena é responsável por capturar as leituras do sensor e enviá-las, pela rede, à aplicação executando no *smartphone*. Os nós e seus componentes serão detalhados a seguir.

Figura 43 – Grafo de Cena do *LeapServer*.

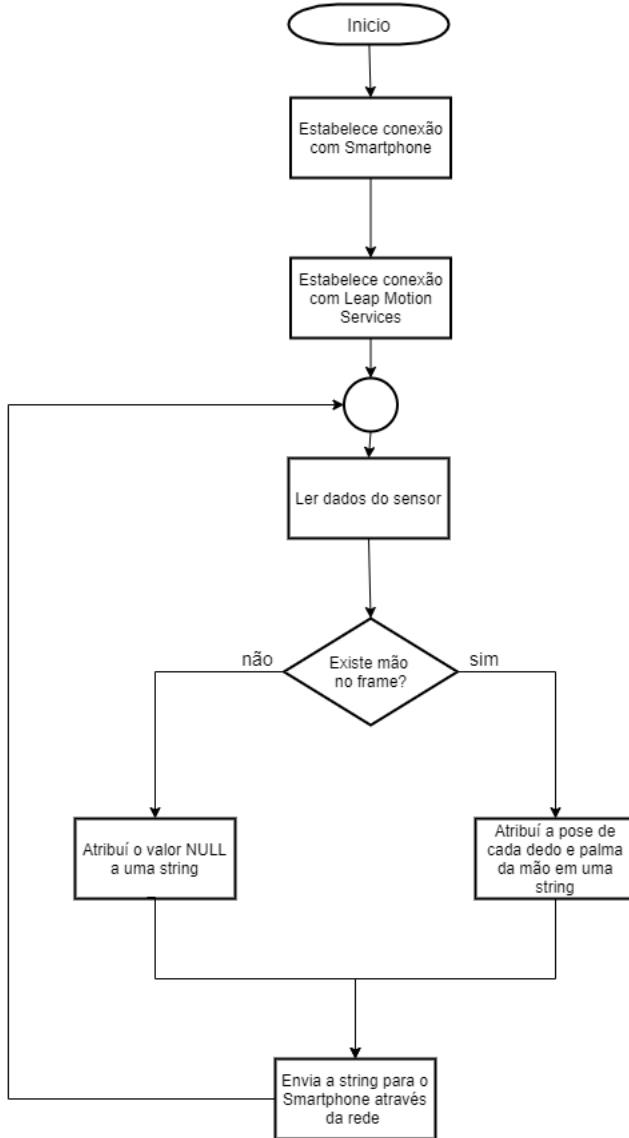


Fonte: Elaborado pelo autor.

- **Nó Mão:** Trata-se de um objeto vazio que engloba todos os objetos 3D que representarão o modelo da mão (esferas para as pontas dos dedos e um paralelepípedo para a palma da mão).
- **Nós Dedos (Polegar, Indicador, Meio, Anelar, Mindinho) e Palma:** Esses nós são bem semelhantes entre si. Todos possuem um modelo 3D, porém somente o polegar e o indicador tem colisores com gatilhos que são capazes de interagir com objetos. Isso se deve ao fato de que o gesto de pinça é feito somente usando esses dedos. Os outros estão representados para visualização.
- **Script Holder:** Outro objeto vazio, porém nesse caso carregando o *script* controla a maior parte da cena. Nesse *script*, chamado *ServerVuforia2*, é feita a conexão com a aplicação em segundo plano do *Leap Motion*, extrai as informações, as formata e envia pela rede para o *smartphone*. O processo geral feito por esse *script* pode ser visto na Figura 44.
- **Directional Light:** *GameObject* com o componente de luz. Sua finalidade é iluminar a cena virtual.
- **Camera:** Câmera Virtual nos permite ter uma visão da cena, apesar disso não ser importante para aplicação já que o usuário estará focado na experiência em RA no HMD. Entretanto, ter essa visão auxilia no desenvolvimento.

- **Leap Hand Controller:** Segundo [LeapMotion \(2020\)](#), esse é o *Prefab* (molde de *GameObject*) que deve ser utilizado ao se utilizar o *Leap Motion* estacionário na mesa, sendo a outra opção colocá-lo no HMD. O *Prefab* contém um *script* que utiliza funções internas do SDK do *Leap Motion* que não cabem nesse contexto. Além disso fazem o controle de um modelo de mão pronto, porém este foi desativado, já que a visão do *smartphone* é o que importa.

Figura 44 – Fluxograma do *script ServerVuforia2*.

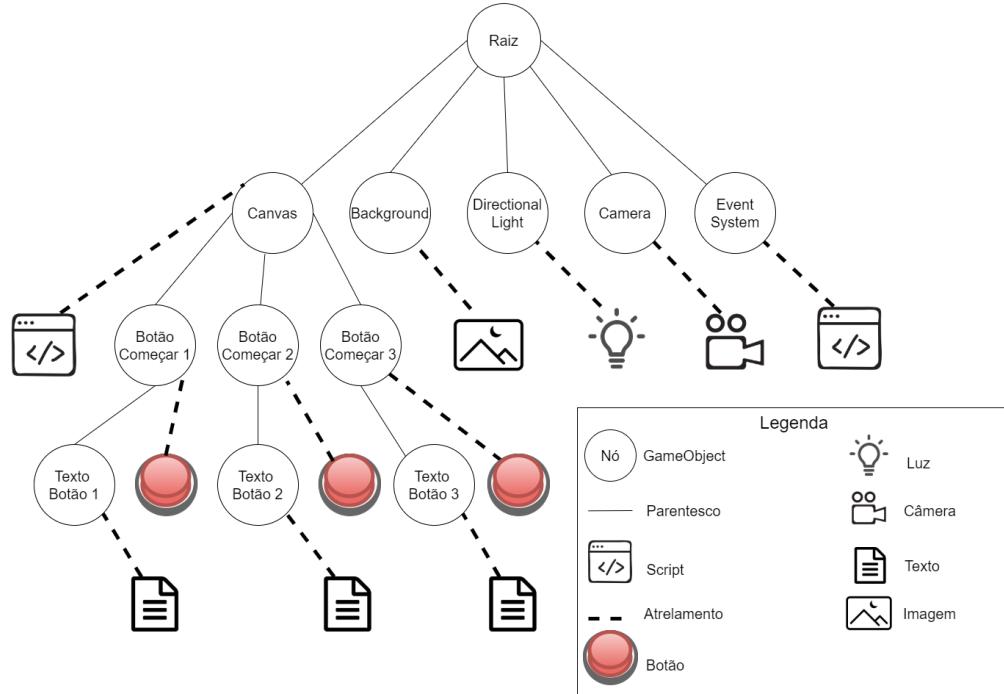


Fonte: Elaborado pelo autor.

4.3.1.2 Menu Principal

O próximo grafo de cena representa o Menu Principal da aplicação *mobile*. O Menu foi feito de maneira simples, ao passo que sua função era somente fazer o controle das cenas de teste. O grafo pode ser visto na Figura 45.

Figura 45 – Grafo de Cena do Menu Principal.



Fonte: Elaborado pelo autor.

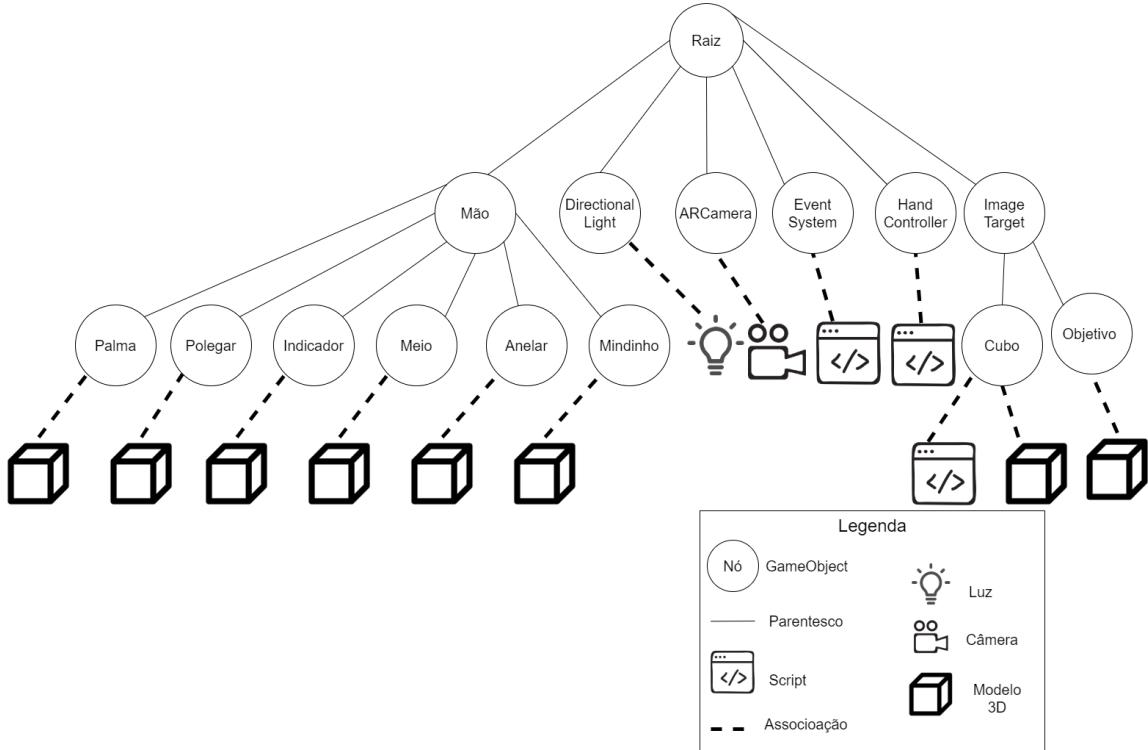
- **Canvas:** Elemento de User Interface (UI), permite a inserção de botões e texto na tela da aplicação. Nessa cena o *script* que faz o controle das mudanças entre cenas está atrelado ao Canvas, juntamente com 3 botões que levam para cada Caso de Teste.
- **Botões Começar 1-3:** São compostos pelo botão em si, que dá gatilho em um evento ao ser clicado, e por um texto que dá nome ao botão.
- **Background:** *GameObject* com um componente imagem para servir de plano de fundo da UI.
- **Directional Light:** Luz ambiente.
- **Camera:** Apontada ortogonalmente para o *Background* para dar o efeito de imagem de plano de fundo ao Menu.
- **Event System:** Responsável pelo tratamento de eventos da UI.

4.3.1.3 Leap Client

E por fim, na Figura 46, o Grafo de Cena de um ambiente de interação genérico que precede os Casos de Teste propriamente ditos, mas tem todos os componentes principais.

- **Nó Mão:** Funciona da mesma forma do *Leap Server*.

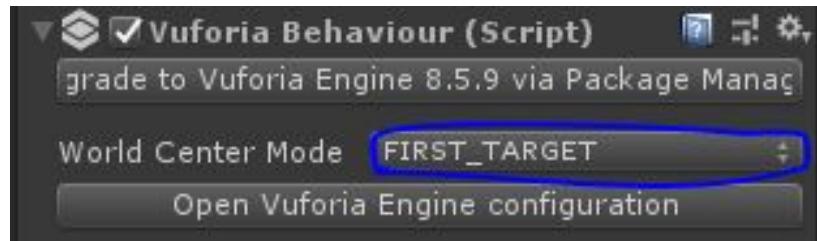
Figura 46 – Grafo de Cena do Ambiente de Interação.



Fonte: Elaborado pelo autor.

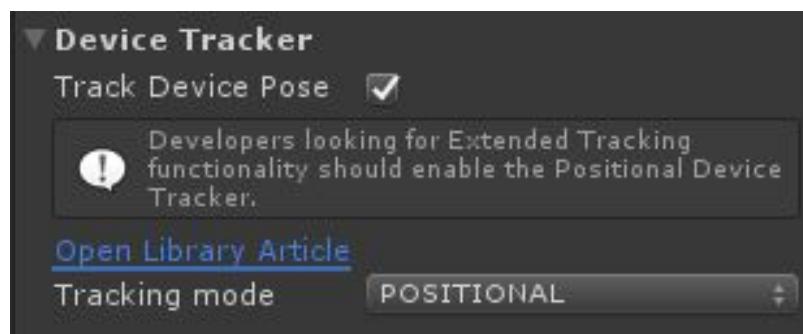
- **Nós Dedos (Polegar, Indicador, Meio, Anelar, Mindinho) e Palma:** Funcionam da mesma forma do *Leap Server*.
- **Directional Light:** Luz ambiente.
- **ARCamera:** Essa câmera é diferente das apresentadas até agora. Enquanto que as outras mostravam a visão do mundo virtual somente, a ARCamera é uma das *Prefabs* do *Vuforia* que permite inserir a visão do mundo real no *background* da cena, possibilitando a criação do ambiente aumentado. É também no *script* desse *GameObject* que é feita a configuração do centro do mundo virtual como sendo o marcador fiducial (Figura 47). Outra configuração importante é referente aos sensores inerciais do *smartphone*. É possível acioná-los como um método secundário de rastreamento, ativando a opção de *positional device tracking* (Figura 48). Auxilia um pouco nesse projeto, visto que há um grande nível de oclusão do marcador e da câmera.
- **Event System:** Responsável pelo tratamento de eventos da UI.
- **Hand Controller:** Objeto vazio que carrega o *script* mais importante da cena, o *ClientVuforia2*. Ele realiza todo o controle das informações chegadas pela rede e pela formatação dessa informação. A estrutura lógica desse *script* pode ser vista na Figura 49.

Figura 47 – Tela onde se coloca o centro do mundo virtual como o primeiro marcador encontrado pelo *Vuforia*.



Fonte: Elaborado pelo autor.

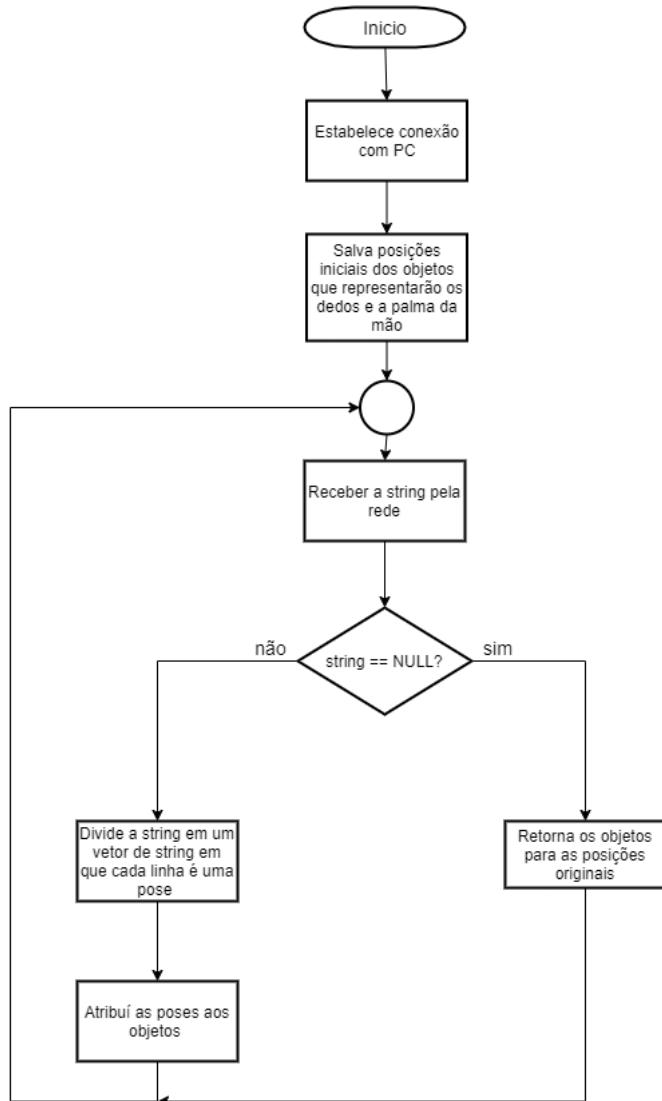
Figura 48 – Tela onde se adiciona sensores inerciais.



Fonte: Elaborado pelo autor.

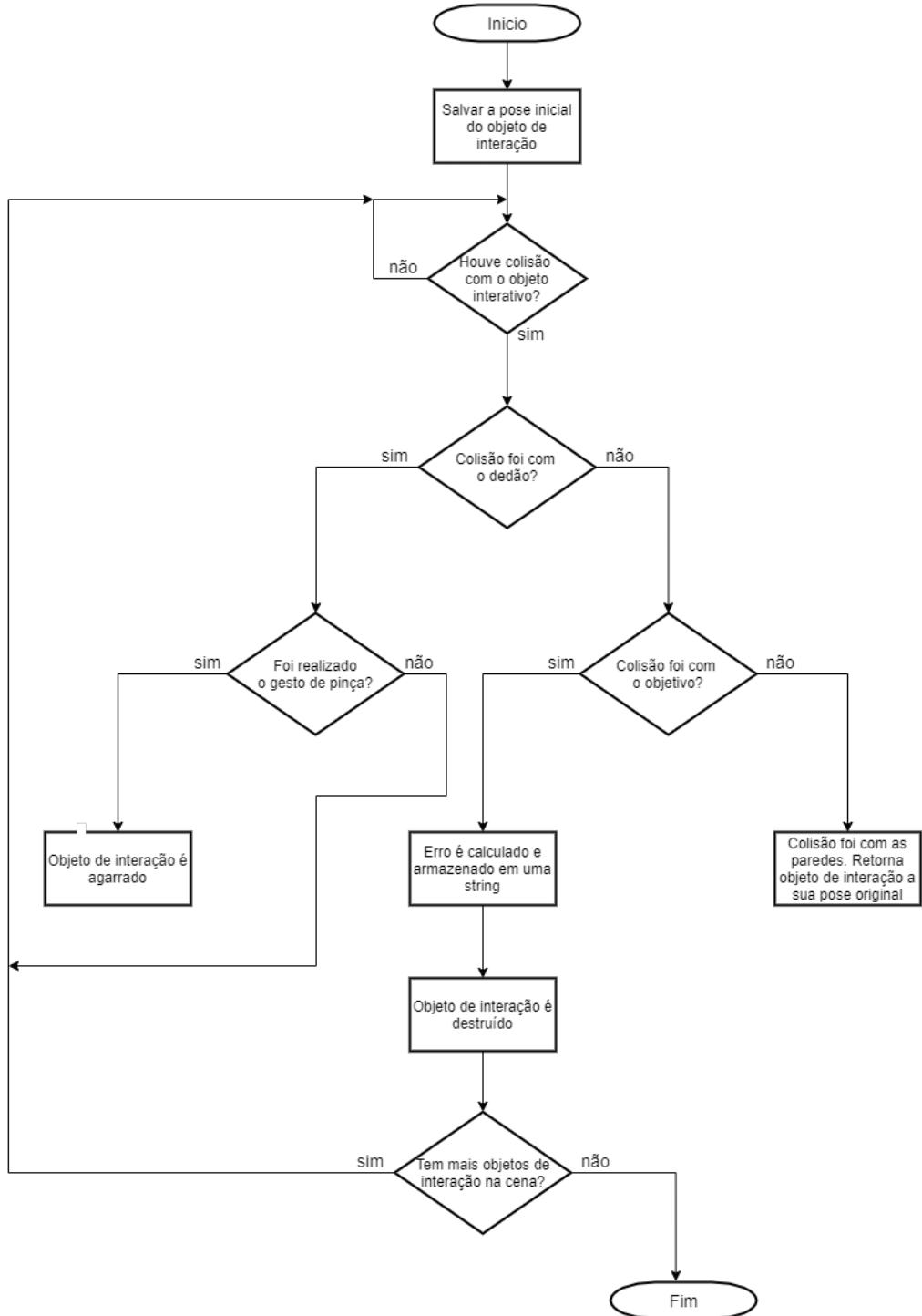
- **Image target:** Objeto que representa o marcador fiducial. Qualquer objeto que está associado ao marcador vai aparecer na cena aumentada com a mesma posição e orientação do mundo virtual.
- **Cubo:** Objeto que é capaz de sofrer interações. Possui um colisor que dá gatilhos a eventos. Quando ocorre uma colisão, o *script* associado a ele verifica qual objeto que causou isso, se forem os dedos da pinça e o gesto da pinça for detectado, o objeto é agarrado. Para fazer o objeto agarrar outro foi usado o conceito de hierarquia apresentado, ou seja, quando o cubo é "agarrado" ele na verdade vai se tornar filho do objeto mão e acompanhar seus movimentos. O *script* associado a este cubo é chamado de *InteractionCube* e serve de base para todos os Casos de Teste. O fluxograma que o representa está na Figura 50.
- **Objetivo:** Grupo de objetos que representam o alvo que o cubo interativo deve ser colocado. Trate-se de um *prefab* que é composto por 4 cubos deformados em paredes de um pequeno alvo com forma quadrangular. As suas paredes tem colisores que mandam o objeto interativo para sua posição original, indicando que o usuário errou o alvo. E no seu centro existe outro colisor que apaga o objeto.

Figura 49 – Fluxograma do *ClientVuforia2*.



Fonte: Elaborado pelo autor.

Figura 50 – Fluxograma do *InteractionCube*.



Fonte: Elaborado pelo autor.

5 Experimentos e Análise dos Resultados

Neste capítulo são apresentados os procedimentos e testes utilizados para a medição de percepção e interação no espaço peripessoal. Ademais, também serão analisados os resultados destes testes.

5.1 Experimentos

A fim de analisar a interação e percepção no espaço peripessoal, foi decidido o uso de medidas quantitativas através da criação de casos de teste.

Vale ressaltar que o objetivo do projeto é a realização de um estudo de caso referente ao uso desse HMD VST baseado em *smartphones*, portanto não é possível a generalização dos resultados que serão apresentados para todos HMDs VST.

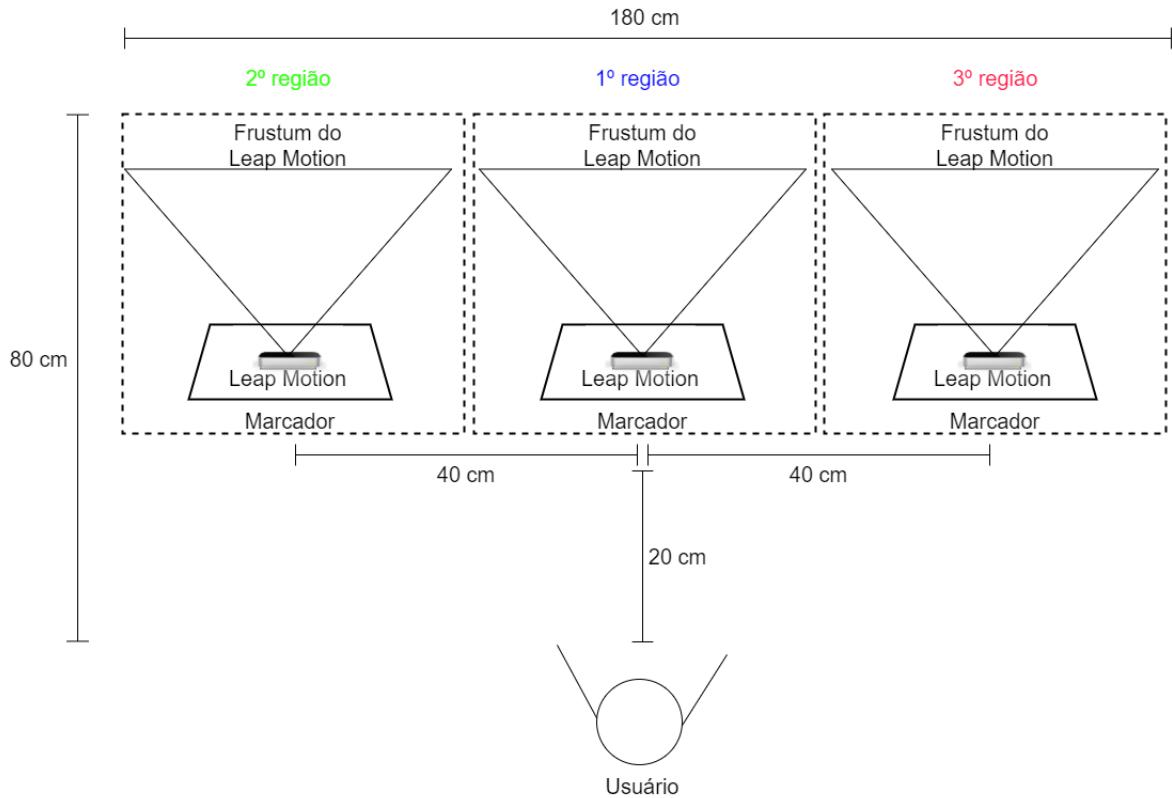
A metodologia para os testes foi baseada no trabalho de [Moser e Swan \(2016\)](#), em que os testes eram realizados por somente um usuário especializado, já que a intenção não é testar a facilidade de uso do sistema, mas sim a percepção e interação no espaço de RA. Foram formulados três cenários de teste, cada um com seus objetivos e especificidades, que serão detalhados a seguir. Cada caso de teste foi executado em sete ciclos em três regiões diferentes: diretamente em frente do usuário, à esquerda do usuário e à direita do usuário. Tais regiões foram criadas para que o espaço peripessoal do usuário fosse testado em sua completude, cobrindo um volume de interação de 180 cm x 80 cm x 60 cm (C x P x A), como mostrado na Figura 51. Um ciclo de testes envolveu a execução do teste cinco vezes consecutivas. É importante notar que não foram usados três sensores simultaneamente, o ciclo é feito em uma região de cada vez, ou seja, após realizar-se o ciclo na 1º região, o marcador e o *Leap Motion* foram movidos para a posição da próxima região.

5.1.1 Caso de Teste 1

O primeiro cenário foi projetado usando um experimento feito por [Jiménez \(2014\)](#) como base, já que esse primeiro teste foi confeccionado para uma maior familiarização das ferramentas do ambiente. Nesse teste, haviam dois cubos e dois alvos de cores diferentes, em que os cubos deveriam ser agarrados e levados para os alvos com as cores correspondentes. Se o usuário errasse qual cubo deveria ir em qual alvo, o cubo voltava para a posição original e o erro era gravado.

O objetivo desse teste era fazer uma primeira avaliação do gesto de pinça e da percepção de profundidade em um ambiente simples, onde o único obstáculo é identificar as cores dos

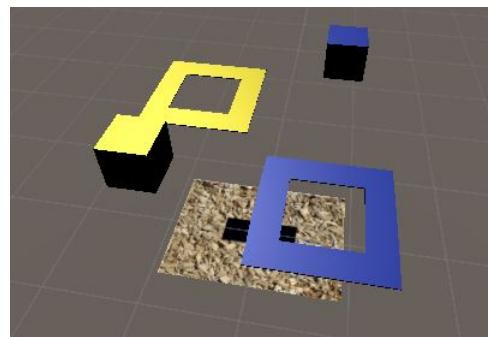
Figura 51 – Regiões em que os experimentos foram realizados.



Fonte: Elaborado pelo autor.

alvos e dos cubos, portanto também estava sendo avaliada a percepção de cores no uso desse HMD. Uma imagem da disposição cena virtual pode ser vista na Figura 52.

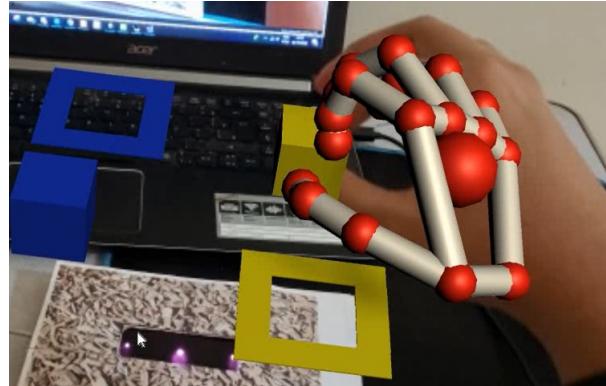
Figura 52 – Disposição dos objetos e alvos no primeiro cenário de teste.



Fonte: Elaborado pelo autor.

O teste usou todas as medidas apresentadas na Tabela 3, com exceção de "Acerto de Obstáculo", já que não existe nenhum obstáculo físico na cena para o usuário acertar. Os dados das medidas eram armazenados em arquivo de texto, na memória interna do *smartphone*, no final da cena. Na Figura 53, pode-se ver a mão do usuário, na cena aumentada, interagindo com um cubo.

Figura 53 – Mão do usuário realizando gesto de pinça.

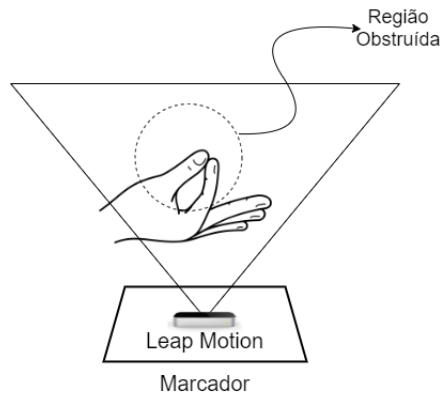


Fonte: Elaborado pelo autor.

5.1.2 Caso de Teste 2

Já o segundo cenário de testes foi elaborado para expandir a ideia do primeiro. Nesse caso, foi implementada uma analogia ao brinquedo de encaixe de formas. Eram dispostos, na cena virtual, um tetraedro, um cubo e um cilindro que deveriam ser encaixados nos alvos que correspondiam a suas formas. Assim como no caso 1, se o usuário se enganasse com relação a qual objeto iria para qual alvo, o objeto voltava a sua posição original e o erro era gravado. Nesse ambiente, também era testado se o usuário conseguia manter a rotação correta dos objetos para o encaixe, uma vez que por mais que os alvos eram grandes, formas como o tetraedro são de difícil encaixe com rotações erradas. Porém, a rotação em torno de alguns eixos não é muito bem reconhecida pelo sistema, pois em algumas posições a mão obstruía a captura da pose dos dedos pelo sensor, dificultando seu reconhecimento, graças a isso o sistema não obriga os usuários a ajeitarem o objeto dessa forma. O fenômeno descrito pode ser visto na Figura 54.

Figura 54 – Movimento dos dedos sendo obstruído pela própria mão.

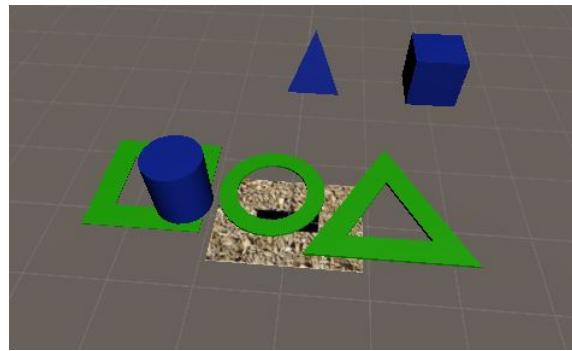


Fonte: Elaborado pelo autor.

O arranjo dos objetos virtuais na cena do segundo cenário está apresentado na Figura

55. Pode ser visto que os objetos e os alvos possuem a mesma cor, portanto podem ser diferenciados somente pela sua forma. Ao contrário do primeiro caso, em que um alvo era posicionado próximo e o outro afastado do usuário, todos os alvos se localizam no centro da cena, e os objetos aparecem distribuídos aleatoriamente pelo ambiente.

Figura 55 – Disposição dos objetos e alvos no segundo cenário de teste.



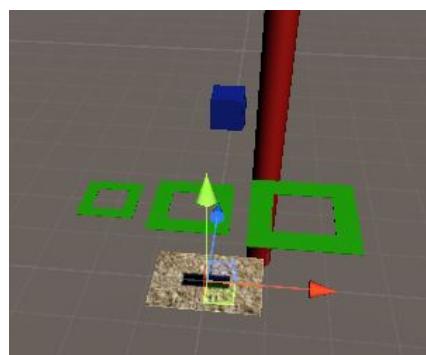
Fonte: Elaborado pelo autor.

O objetivo desse teste era medir se o usuário é capaz de diferenciar os objetos pela forma e se consegue realizar o encaixe de objetos com formatos mais complexos, mantendo um controle da rotação do objeto.

5.1.3 Caso de Teste 3

Por fim, tem-se o terceiro caso de teste, no qual são distribuídos três alvos quadrados de tamanhos diferentes na cena, um obstáculo cilíndrico e três cubos de tamanhos diferentes. Assim como os outros cenários, o objetivo do usuário é encaixar os objetos nos alvos corretos, porém agora há um obstáculo cilindro móvel entre os cubos e os alvos. A disposição dos objetos da cena pode ser observado na Figura 56 (os cubos estão todos na mesma posição, porque a sua posição é aleatoriamente selecionada entre três possíveis opções no começo da cena).

Figura 56 – Disposição dos objetos e alvos no terceiro cenário de teste.



Fonte: Elaborado pelo autor.

O obstáculo faz um movimento repetitivo, indo da esquerda para a direita, e depois volta, ele funciona como as paredes dos alvos, se for tocado, o objeto retorna a posição original e o erro é gravado. Ademais as posições dos cubos são trocada a cada execução do caso de teste, a fim de evitar que o usuário consiga simplesmente memorizar a posição dos objetos, como foi notado nos outros cenários.

O obstáculo recebe uma coloração diferente dos outros objetos, posto que ele não é um objeto de interação e será ignorada a colisão da mão do usuário com o mesmo. Já os objetos que são capazes de receber interação, não só possuem a mesma cor, mas também possuem a mesma forma, sendo diferenciados somente pelo seu tamanho.

Logo, o objetivo desse caso de teste era verificar se a inclusão de um obstáculo, que acaba fazendo oclusão com os objetos de interação na cena, impacta de alguma forma a interação e percepção do usuário. E também, descobrir se o usuário conseguia diferenciar os objetos e alvos somente através de seu tamanho.

5.2 Análise dos resultados

Cada caso de teste foi executado 35 vezes em cada região, resultando em um total de 105 medidas quando considerado o experimento como um todo. Como mencionado anteriormente, o experimento foi realizado por somente um usuário, ao longo de quatro dias, por cerca de duas horas por dia. Vale ressaltar que as regiões foram executadas na seguinte ordem: meio, esquerda e por último direita. A métrica utilizada nos experimentos é descrita na Tabela 3.

Tabela 3 – Medidas utilizadas nos testes.

Medida	Descrição
DistanciaPinch	Ao realizar o gesto de pinça, grava a distância da mão ao objeto
Tempo	Tempo decorrido para concluir um teste
ErroDistanciaObjetivo	Distância do centro do objeto até o centro do alvo, ao acertá-lo
Erro de Alvo	Usuário tentou encaixar o objeto no alvo errado
Acerto de Parede	Usuário não fez o encaixe corretamente, porque acertou a parede
Acerto de Obstáculo	Usuário acerta um obstáculo
ContadordePinch	Quantidade de gesto de pinça realizadas pelo usuário

Fonte: Elaborada pelo autor.

5.2.1 Resultados do Caso de Teste 1

Na Figura 57, são apresentados os gráficos referentes ao primeiro cenário de teste. O gráfico do tempo (Figura 57(a)) mostra que o usuário levou em média aproximadamente

26 segundos para completar o teste na 1º região, porém a média caiu significativamente nas regiões seguintes. Isso provavelmente se deve ao fato do usuário estar se familiarizando com o gesto da pinça e com a tarefa proposta, provando que realmente é útil a mudança de posição dos objetos virtuais para amenizar isso. O gráfico mostrado na Figura 57(b), apresenta a média da quantidade de gestos de pinça utilizados em cada posição, e seu formato é extremamente semelhante ao gráfico do tempo, aumentando a confirmação de que o usuário estava se acostumando com o gesto e acabou precisando de menos tentativas para completar a tarefa.

Já na Figura 57(c) e (d) são gráficos muito importantes para esta análise, já que são estes que irão mostrar se o usuário estava estimando a posição dos objetos virtuais corretamente. Como mostrado na seção anterior, nesse caso de teste, dois cubos eram apresentados ao usuário, um perto (azul) e um longe (amarelo). Nesse ponto pode-se observar um padrão interessante, em média o usuário estimava a posição do cubo azul com maior facilidade observado pela uniformidade das três regiões, pois o cubo azul sempre ficava perto do usuário independente da região. Já o cubo amarelo sofre de uma diferença principalmente na 3º região, onde o cubo amarelo ficava mais distante que o normal do usuário, aumentando a média de erro para 7,5 cm. Como foi visto nas Seções 2.1.3.5 e 2.4, as pistas de profundidade vão perdendo sua eficácia quanto mais distante o objeto observado está do observador.

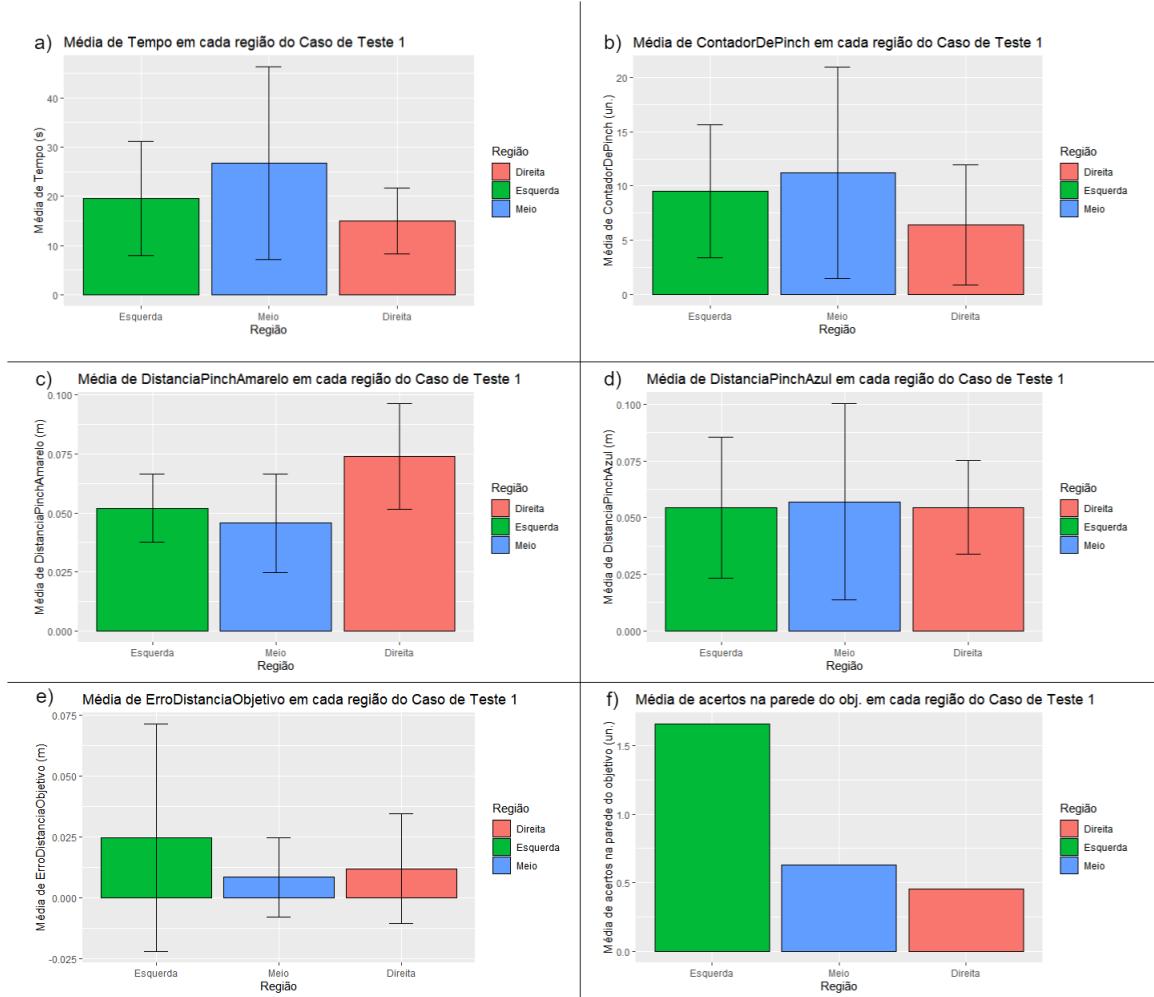
Apesar da curva de aprendizado do usuário, no gráfico da Figura 57(e), os resultados mostram uma maior precisão na tarefa de encaixar o objeto no alvo na região do meio, que é a mais próxima do usuário, conseguindo a média de 0,8 cm, que é bem mais baixa que em trabalhos semelhantes, como será discutido posteriormente. Por fim, no gráfico da Figura 57(f), são ilustradas as médias do número de acertos na parede do alvo, ou seja, quando o usuário falha ao encaixar o objeto. Nesse caso é mostrada uma aparente dificuldade na 2º região, possivelmente devido ao fato do usuário ser destro e ter escolhido fazer uso da mão direita durante todos os experimentos, causando uma grande distância do usuário aos objetos virtuais, dificultando uma interação mais precisa. Porém, mesmo nesse caso, a média foi de menos de 2 erros por cena.

Não foi criado um gráfico para a medida de erro de alvo, já que o número de vezes que o usuário cometeu esse erro foi desprezível. Portanto, identificar forma, cor e tamanho não é um problema no uso desse HMD.

5.2.2 Resultados do Caso de Teste 2

Agora com o usuário já familiarizado com o ambiente e os gestos, não houve nenhuma aparente melhora em quantidade de tempo e de gestos realizados ao longo da execução das três regiões, como mostrado pelos gráficos da Figura 58(a) e (b). Somente nota-se uma ligeira vantagem na região do meio e da direita, presumidamente pelos motivos apresentados anteriormente.

Figura 57 – Gráficos dos resultados do primeiro caso de teste.



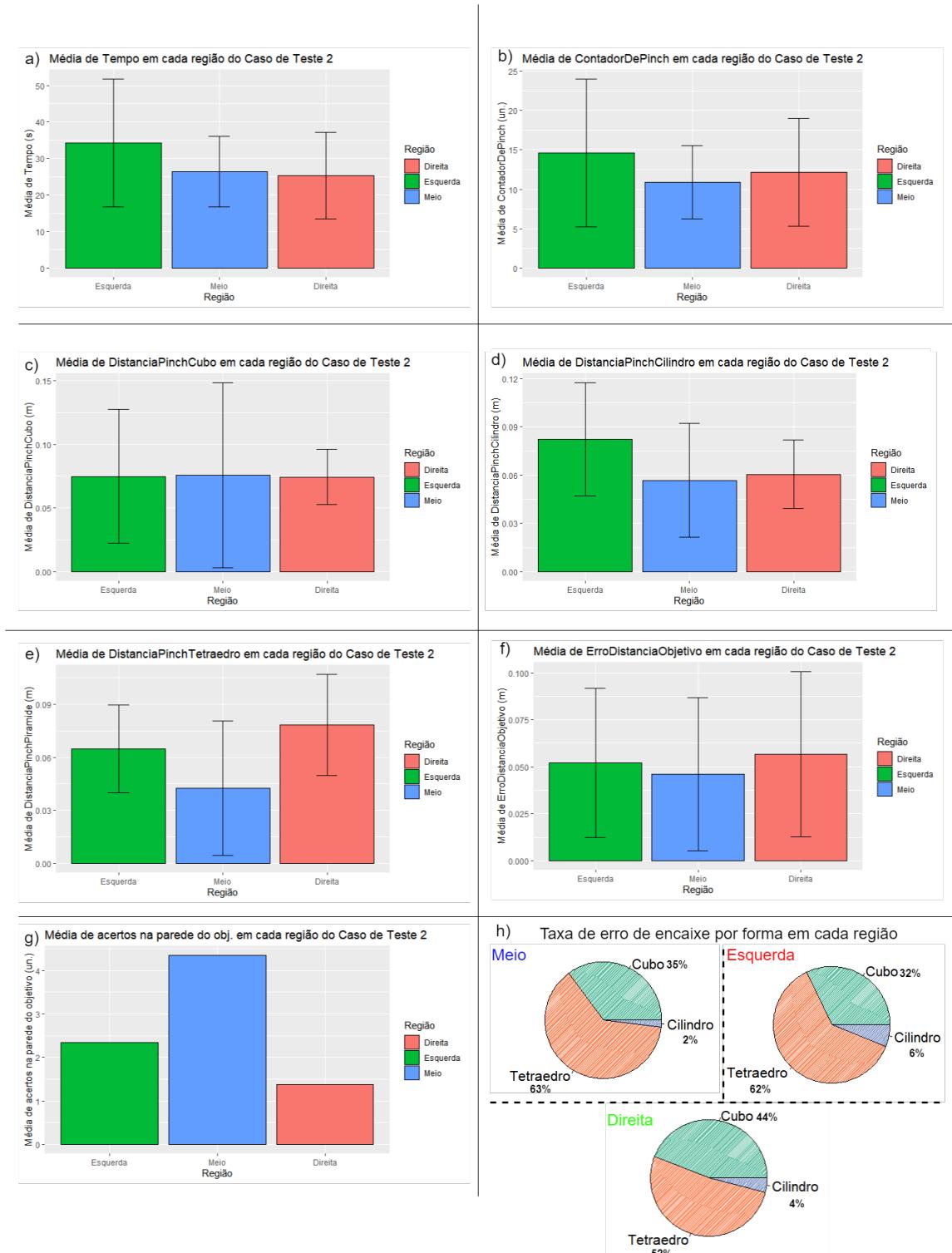
Fonte: Elaborado pelo autor.

Nos gráficos na Figura 58(c), (d) e (e) houve uma certa uniformização dos valores das médias entre 5 e 6 cm, salvo gráfico do cilindro na região da esquerda (muito longe do usuário) com um aumento e o gráfico do tetraedro na região do meio com uma redução (muito perto do usuário).

Graças ao desafio de manipular novas formas e encaixá-las em seus alvos com formato específico, no gráfico da Figura 58(f), pode-se ver que as médias dos erros ficaram em torno de 5 cm, grande diferença do experimento anterior que todos ficaram abaixo de 2,5 cm. Pode-se ver também uma uniformização das três regiões, possivelmente devido a disposição dos alvos na cena, permitindo que sempre houvesse um perto do usuário (na esquerda, alvo triangular por exemplo). Pode-se ver também o impacto dessa mudança no gráfico da Figura 58(g), em que se pode observar o dobro de encaixes mal sucedidos em comparação com o último caso de teste. Para demonstrar isso, nos gráficos presentes na Figura 58(h), pode-se observar a taxa de erros de cada objeto 3D, região por região. Em todas as condições, o tetraedro teve o maior número de erros de encaixamento, somente se igualando nos testes na região da direita,

enquanto que o cilindro teve a menor taxa por uma grande margem.

Figura 58 – Gráficos dos resultados do segundo caso de teste.



Fonte: Elaborado pelo autor.

5.2.3 Resultados do Caso de Teste 3

Por último tem-se a Figura 59, na qual são mostrados os resultados obtidos nos experimentos no caso de teste 3.

Nos dois primeiros gráficos, era esperado uma maior uniformidade entre as três regiões, já que agora a posição dos objetos de interação eram trocados a cada execução da cena. E como pode ser visto na Figura 59(a) e (b), isso foi atingido com sucesso, mesmo que haja uma média de tempo mais elevada na 1º região, o seu desvio padrão também é maior do que o dos outros. Algo a ser notado é que mesmo com o obstáculo, em média o tempo gasto não teve um aumento em relação ao experimento anterior.

A uniformidade dos gráficos continua nos apresentados na Figura 59(c), (d) e (e). Nesses gráficos é mostrado que a estimativa da posição dos objetos se encontrou geralmente entre 7 e 9 cm nas três regiões, com exceção do cubo médio que por volta de 9 e 11 cm.

Como no primeiro caso de testes, o alvo e os objetivos tinham formatos quadrados. Graças a isso, as médias dos erros de encaixe, mostrados pela Figura 59(f), tiveram um resultado semelhante ao primeiro caso de teste, ficando entre 1,1 cm e 2,2 cm.

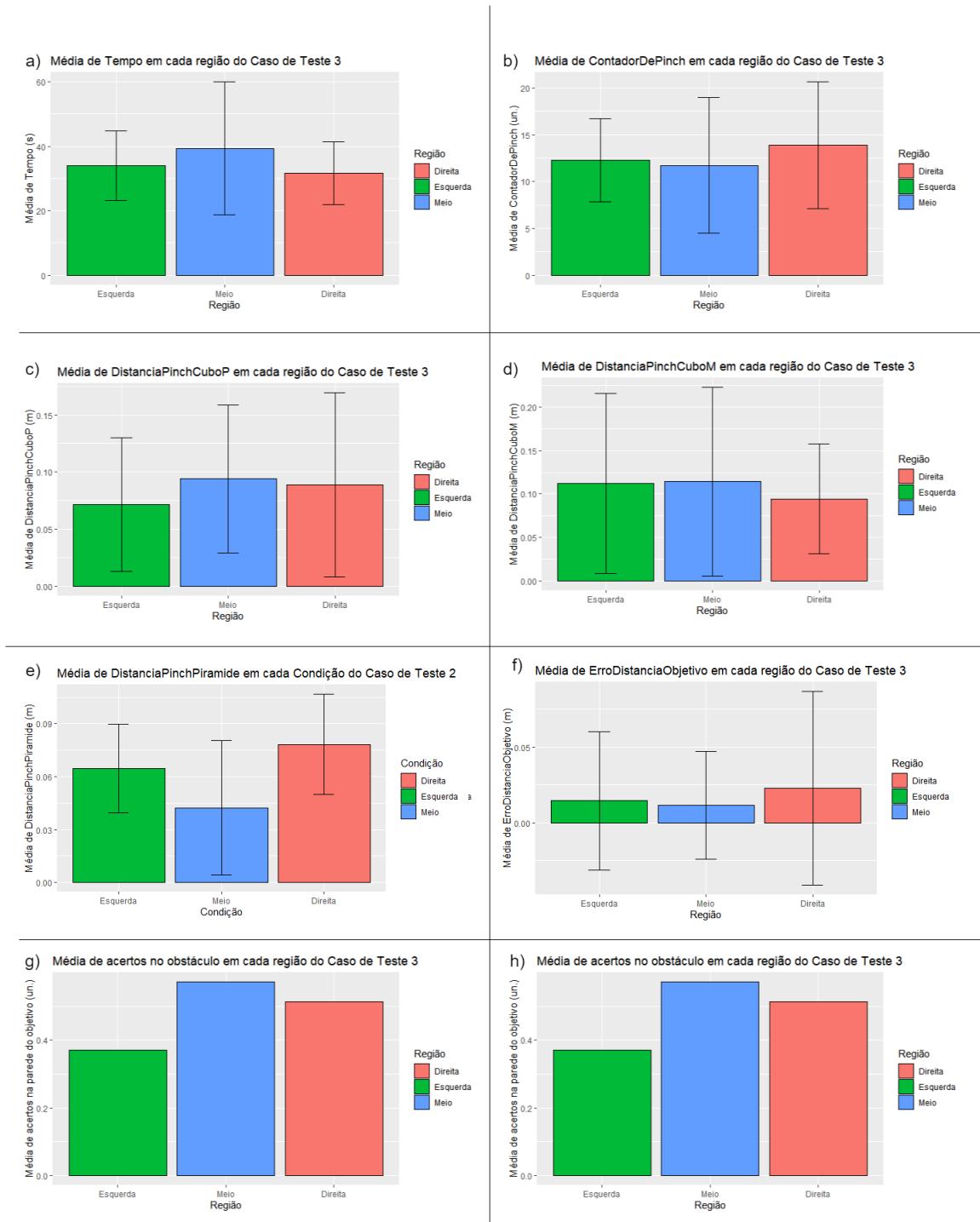
Finalmente, nos dois últimos gráficos, presentes na Figura 59(g) e (h), são expostos as baixíssimas médias de erros ao acertar as paredes do objetivo e o obstáculo. Isso prova que com a perspectiva do HMD, o usuário consegue facilmente observar e desviar do obstáculo e encaixar o objeto corretamente. Também mostra que a interação é robusta o suficiente ao ponto de desviar de um obstáculo dinâmico.

5.2.4 Discussão

No Capítulo 3 foi revisado o trabalho de Ballestin, Solari e Chessa (2018), que propõe uma medição bem semelhante a que foi apresentada nesse trabalho, chegando até mesmo a fazer uso de um HMD baseado em *smartphone*. A estimativa de profundidade usando esse HMD, quando sem *feedback*, chegou a ter erros de 28cm e 18cm com *feedback*. No trabalho em questão era pedido ao usuário que somente estimasse a posição de um cubo. Em relação a estimativa da posição do objeto virtual, as medidas encontradas aqui tem uma significativa melhora, com erros variando somente de 5 a 12 cm. Essa diferença pode ser devido a grande interatividade do sistema, já que somente o acréscimo de *feedback* resultou em uma redução de 10 cm no trabalho de Ballestin, Solari e Chessa. Outra concordância, entre os dois trabalhos, foi a conclusão de que quanto maior a distância pior a estimativa de profundidade.

Também houve a comparação entre RA e RV no sistema de Ping, Liu e Weng (2019). Neste trabalho era pedido que o usuário estimasse a profundidade em grandes distâncias variando de 1.75 m até 7.35 m, ou seja além do espaço peripessoal. Portanto conseguiram encontrar uma média de erro entre 15 m e 20 m no ambiente de RA. Se for comparado com as medidas encontradas neste trabalho, podemos verificar que as pistas de profundidade tem

Figura 59 – Gráficos dos resultados do terceiro caso de teste.



Fonte: Elaborado pelo autor.

maior impacto no espaço peripessoal. Concluindo então que quanto maior a distância, maior o erro encontrado na estimativa de profundidade.

Apesar da semelhança do sistema desse projeto com o de Jiménez (2014), o interesse daquele trabalho era investigar a performance do gesto de pinça e a usabilidade do gesto com inúmeros usuários. Nenhuma análise foi feita com relação a percepção. Além da diferença de modalidade de *display* utilizada por aquele trabalho, que era baseado em monitor, enquanto nesse é usado um HMD, possibilitando testes mais complexos como o caso de teste 3 que possui um obstáculo dinâmico que obstrui outros objetos virtuais.

Apesar do resultado mais promissor do que os trabalhos anteriores, graças as deficiências de se usar um *display* VST com somente uma câmera, o resultado ainda está distante da precisão de 1cm obtida em HMDs OST encontrado por Ballestin, Solari e Chessa, colocando ainda os HMDs VST como um equipamento não indicado para aplicações hápticas ou que exijam uma alta precisão, como na área médica e indústria (BALLESTIN; SOLARI; CHESSA, 2018).

Algumas limitações desse sistema envolvem o sacrifício da mobilidade do *smartphone* devido a dependência do PC, para uma aplicação realmente de fácil acesso, uma melhor integração com o sensor RGB-D deveria ser pensada, como o uso de um *smartphone* que contenha o sensor embutido (MICALI, 2016).

Finalmente, outra limitação é a impossibilidade de rotacionar o objeto virtual em torno do eixo x e do eixo z, devido ao posicionamento do *Leap Motion*.

6 Conclusão

O objetivo principal desse trabalho foi investigar e avaliar o uso de HMDs VST baseados em *smartphones*, no que diz respeito à percepção de profundidade no espaço próximo ao usuário. A escolha deste tipo de tecnologia de *display* deveu-se ao seu baixo custo e a sua simplicidade em termos de arquitetura óptica. Optou-se pelo uso da interação baseada em gestos por ser uma forma de interface natural, a qual, por ser intuitiva, exige pouco treinamento para ser utilizada. Além disso, foram estudadas as pistas de profundidade e seu funcionamento, e como podem afetar a percepção visual humana. Também foram revisadas arquiteturas existentes de HMDs, técnicas de rastreamento, limitações e exemplos da tecnologia de RA.

Neste trabalho foram, também, identificadas as principais dificuldades no desenvolvimento de aplicações de RA para HMDs VST baseados em *smartphone*. Durante o processo foi criado um ambiente experimental capaz de realizar interações naturais baseadas em gestos, e por mais que a percepção de profundidade do espaço peripessoal não tenha sido ideal, os resultados dos experimentos indicaram que o uso deste tipo de HMD VST pode ser adequado para aplicações de RA que não exijam alta precisão, como, por exemplo, aplicações na área da Educação, comércio eletrônico, visualização científica e games.

6.1 Trabalhos Futuros

Considerando uma futura continuidade desse trabalho, podem ser citados os seguintes aspectos a serem explorados:

- Utilização de smartphones com sensor de profundidade embutido, viabilizando uma aplicação móvel;
- A investigação de HMDs OST baseados em *smartphone* e comparação entre as duas modalidades;
- A integração de mais gestos e maior complexidade nesses gestos e interações;
- A realização de uma avaliação qualitativa; e
- O estudo da aceitação social desse tipo de HMD.

Referências

- ALONSO-ROSA, M.; CASTRO, A. Gil-de; MORENO-MUNOZ, A.; GARRIDO-ZAFRA, J.; GUTIERREZ-BALLESTEROS, E.; CAÑETE-CARMONA, E. An iot based mobile augmented reality application for energy visualization in buildings environments. *Applied Sciences*, Multidisciplinary Digital Publishing Institute, v. 10, n. 2, p. 600, 2020.
- AMEMIYA, T. Resizing of the peripersonal space for the seated for different step frequencies of vibrations at the soles. In: IEEE. 2020 IEEE Conference on Virtual Reality and 3D User Interfaces Abstracts and Workshops (VRW). [S.I.], 2020. p. 287–288.
- ATCHISON, D. A.; SMITH, G. The human eye: an overview. In: *Optics of the human eye*. [S.I.]: Butterworth-Heinemann, 2000. p. 3–10.
- AZUMA, R. T. A survey of augmented reality. *Presence: Teleoperators & Virtual Environments*, MIT Press, v. 6, n. 4, p. 355–385, 1997.
- BALLESTIN, G.; SOLARI, F.; CHESSA, M. Perception and action in peripersonal space: A comparison between video and optical see-through augmented reality devices. In: IEEE. 2018 IEEE International Symposium on Mixed and Augmented Reality Adjunct (ISMAR-Adjunct). [S.I.], 2018. p. 184–189.
- BATMAZ, A. U.; MACHUCA, M. D. B.; PHAM, D. M.; STUERZLINGER, W. Do head-mounted display stereo deficiencies affect 3d pointing tasks in ar and vr? In: IEEE. 2019 IEEE Conference on Virtual Reality and 3D User Interfaces (VR). [S.I.], 2019. p. 585–592.
- BILLINGHURST, M.; CLARK, A.; LEE, G. A survey of augmented reality. 2015.
- CARMIGNIANI, J.; FURHT, B. Augmented reality: an overview. In: *Handbook of augmented reality*. [S.I.]: Springer, 2011. p. 3–46.
- CHEN, X.; XU, L.; WANG, Y.; WANG, H.; WANG, F.; ZENG, X.; WANG, Q.; EGGER, J. Development of a surgical navigation system based on augmented reality using an optical see-through head-mounted display. *Journal of biomedical informatics*, Elsevier, v. 55, p. 124–131, 2015.
- COLGAN, A. *How Does the Leap Motion Controller Work*. 2014. Disponível em: <<https://blog.leapmotion.com/hardware-to-software-how-does-the-leap-motion-controller-work/>>. Acesso em: 18/11/2020.
- COSTA, R. M.; KAYATT, P.; BOGONI, T. Hardware. In: *Introdução a Realidade Virtual e Aumentada*. [S.I.]: Editora SBC, 2018. p. 92–102.
- CUTTING, J. E.; VISHTON, P. M. Perceiving layout and knowing distances: The integration, relative potency, and contextual use of different information about depth. In: *Perception of space and motion*. [S.I.]: Elsevier, 1995. p. 69–117.
- DIVERDI, S.; HOLLERER, T. Groundcam: A tracking modality for mobile mixed reality. In: IEEE. 2007 IEEE Virtual Reality Conference. [S.I.], 2007. p. 75–82.

- DOSHI, A.; SMITH, R. T.; THOMAS, B. H.; BOURAS, C. Use of projector based augmented reality to improve manual spot-welding precision and accuracy for automotive manufacturing. *The International Journal of Advanced Manufacturing Technology*, Springer, v. 89, n. 5-8, p. 1279–1293, 2017.
- DRASIC, D.; MILGRAM, P. Perceptual issues in augmented reality. In: INTERNATIONAL SOCIETY FOR OPTICS AND PHOTONICS. *Stereoscopic displays and virtual reality systems III*. [S.I.], 1996. v. 2653, p. 123–134.
- HALLER, M. Photorealism or/and non-photorealism in augmented reality. In: *Proceedings of the 2004 ACM SIGGRAPH international conference on Virtual Reality continuum and its applications in industry*. [S.I.: s.n.], 2004. p. 189–196.
- HONG, J. *Considering privacy issues in the context of Google glass*. [S.I.]: ACM New York, NY, USA, 2013.
- HOUNSELL, M. da S.; TORI, R.; KIRNER, C. Realidade aumentada. In: *Introdução a Realidade Virtual e Aumentada*. [S.I.]: Editora SBC, 2018. p. 36–74.
- HUNLEY, S. B.; LOURENCO, S. F. What is peripersonal space? an examination of unresolved empirical issues and emerging findings. *Wiley Interdisciplinary Reviews: Cognitive Science*, Wiley Online Library, v. 9, n. 6, p. e1472, 2018.
- IBAÑEZ, A. S.; FIGUERAS, J. P. *Vuforia v1. 5 SDK: Analysis and evaluation of capabilities*. Dissertação (Mestrado) — Universitat Politècnica de Catalunya, 2013.
- JIMÉNEZ, S. *Physical Interaction in augmented environments*. [S.I.]: sl: Gjøvik University College, 2014.
- KÁN, P.; KAUFMANN, H. Physically-based depth of field in augmented reality. In: *Eurographics (Short Papers)*. [S.I.: s.n.], 2012. p. 89–92.
- KITA, K.; OTAKA, Y.; TAKEDA, K.; SAKATA, S.; USHIBA, J.; KONDO, K.; LIU, M.; OSU, R. A pilot study of sensory feedback by transcutaneous electrical nerve stimulation to improve manipulation deficit caused by severe sensory loss after stroke. *Journal of neuroengineering and rehabilitation*, Springer, v. 10, n. 1, p. 55, 2013.
- KIYOKAWA, K. Head-mounted display technologies for augmented reality. *Fundamentals of Wearable Computing and Augmented Reality*, CRC Press, p. 59–84, 2015.
- LEAPMOTION. *Leap Motion: Unity Modules*. 2020. Disponível em: <<https://leapmotion.github.io/UnityModules/>>. Acesso em: 25/11/2020.
- MAGNENAT, S.; NGO, D.; ZUND, F.; RYFFEL, M.; NORIS, G.; ROTHLIN, G.; MARRA, A.; NITTI, M.; FUÀ, P.; GROSS, M.; SUMNER, R. Live texturing of augmented reality characters from colored drawings. *IEEE transactions on visualization and computer graphics*, v. 21, 07 2015.
- MEYER, K.; APPLEWHITE, H. L.; BIOCCHI, F. A. A survey of position trackers. *Presence: Teleoperators & Virtual Environments*, MIT Press, v. 1, n. 2, p. 173–200, 1992.
- MICALI, B. *RealSense, celular da Intel que tem câmeras 3D, entra em pré-venda*. 2016. Disponível em: <<https://www.tecmundo.com.br/intel/92671-realsense-cellular-intel-tem-cameras-3d-entra-pre-venda.htm>>. Acesso em: 1/12/2020.

- MOSER, K. R.; SWAN, J. E. Evaluation of user-centric optical see-through head-mounted display calibration using a leap motion controller. In: IEEE. *2016 IEEE Symposium on 3D User Interfaces (3DUI)*. [S.I.], 2016. p. 159–167.
- NACERI, A.; CHELLALI, R.; HOINVILLE, T. Depth perception within peripersonal space using head-mounted display. *Presence: Teleoperators and Virtual Environments*, MIT Press, v. 20, n. 3, p. 254–272, 2011.
- NAKAMAE, E.; TADAMURA, K. Photorealism in computer graphics—past and present. *Computers & graphics*, Elsevier, v. 19, n. 1, p. 119–130, 1995.
- NILSSON, S.; JOHANSSON, B. Acceptance of augmented reality instructions in a real work setting. In: *CHI'08 extended abstracts on Human factors in computing systems*. [S.I.: s.n.], 2008. p. 2025–2032.
- PERANNAGARI, K. T.; CHAKRABARTI, S. Factors influencing acceptance of augmented reality in retail: insights from thematic analysis. *International Journal of Retail & Distribution Management*, Emerald Publishing Limited, 2019.
- PEREY, C.; ENGELKE, T.; REED, C. Current status of standards for augmented reality. In: *Recent trends of mobile collaborative augmented reality systems*. [S.I.]: Springer, 2011. p. 21–38.
- PING, J.; LIU, Y.; WENG, D. Comparison in depth perception between virtual reality and augmented reality systems. In: IEEE. *2019 IEEE Conference on Virtual Reality and 3D User Interfaces (VR)*. [S.I.], 2019. p. 1124–1125.
- PLOPSKI, A.; MOSER, K. R.; KIYOKAWA, K.; SWAN, J. E.; TAKEMURA, H. Spatial consistency perception in optical and video see-through head-mounted augmentations. In: IEEE. *2016 IEEE Virtual Reality (VR)*. [S.I.], 2016. p. 265–266.
- POPESCU, D.; CERNAIANU, M.; DUMITRACHE, I. Automatic rough alignment for key components in laser driven experiments using fiducial markers. In: IOP PUBLISHING. *Journal of Physics: Conference Series*. [S.I.], 2018. v. 1079, n. 1, p. 012013.
- REICHELT, S.; HÄUSSLER, R.; FÜTTERER, G.; LEISTER, N. Depth cues in human visual perception and their realization in 3d displays. In: INTERNATIONAL SOCIETY FOR OPTICS AND PHOTONICS. *Three-Dimensional Imaging, Visualization, and Display 2010 and Display Technologies and Applications for Defense, Security, and Avionics IV*. [S.I.], 2010. v. 7690, p. 7690B.
- RITSOS, P. D.; RITSOS, D. P.; GOUGOULIS, A. S. Standards for augmented reality: A user experience perspective. In: *International AR standards meeting*. [S.I.: s.n.], 2011. p. 1–9.
- SUTHERLAND, I. E. A head-mounted three dimensional display. In: *Proceedings of the December 9-11, 1968, fall joint computer conference, part I*. [S.I.: s.n.], 1968. p. 757–764.
- UNITY. *The Hierarchy window*. 2019. Disponível em: <<https://docs.unity3d.com/Manual/Hierarchy.html>>. Acesso em: 24/11/2020.
- UNITY. *Introduction to components*. 2019. Disponível em: <<https://docs.unity3d.com/Manual/Components.html>>. Acesso em: 24/11/2020.
- UNITY. *Unity*. 2020. Disponível em: <<https://unity.com/>>. Acesso em: 24/11/2020.

VUFORIA. *Vuforia Engine Features*. 2020. Disponível em: <<https://library.vuforia.com/content/vuforia-library/en/features/overview.html>>. Acesso em: 24/11/2020.

WANG, J.; SUENAGA, H.; YANG, L.; KOBAYASHI, E.; SAKUMA, I. Video see-through augmented reality for oral and maxillofacial surgery. *The international journal of medical robotics and computer assisted surgery*, Wiley Online Library, v. 13, n. 2, p. e1754, 2017.