FOSSILAR: EXPLORANDO A INTERAÇÃO TÁTIL EM MODELOS 3D DE FÓSSEIS ATRAVÉS DA REALIDADE AUMENTADA

Julio Vicente Brych, Dalton Solano dos Reis – Orientador

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

jbrych@furb.br, dalton@furb.br

Resumo: *Esse artigo apresenta o desenvolvimento de um aplicativo de realidade aumentada em conjunto ao aparelho Leap Motion. Com objetivo de criar uma experimentação interativa para manipular peças do acervo da Exposição de História Natural Fritz Müller – FURB, utilizando as mão livres do usuário para interagir com componentes virtuais A aplicação foi desenvolvida usando o motor gráfico Unity em conjunto com as bibliotecas Vuforia, Ultraleap Tracking e Mirror. A aplicação foi testada com visitantes da exposição no qual foi aplicado um questionário para a coleta de dados. Os resultados obtidos foram satisfatórios de forma que a aplicação atingiu o seu objetivo.*

Palavras-chave*: Augmented Reality, Leap Motion, Fossil, Museum.*

1 INTRODUÇÃO

Desde que entramos no século XXI os avanços tecnológicos abriram caminhos para novas mídias, como bandas maiores de internet e dispositivos móveis com uma maior capacidade de processamento e visualização. Com isso, os museus vêm buscando se adaptar às novas demandas do público adaptando as tecnologias aos seus espaços expositivos, como exposições que permitem a interação dos visitantes com tablets, smartphones, tela táteis e com óculos de realidade virtual (SILVA, 2018). Como por exemplo, o Museu do Louvre em Paris, que possui um passeio virtual disponível para tablets e smartphones, no qual se pode ter uma experiencia de ver as obras do museu em alta resolução cheia de detalhes, e ainda poder ter acesso a vídeos com explicações sobres as obras.

Para Costa (2020), tanto a Realidade Virtual (RV) quanto a Realidade Aumentada (RA) são tecnologias particularmente uteis para promover a interação remota com as obras de arte. Portanto, possibilitando o interesse do público a espaços de exposição artística, como também podendo proporcionar experiências únicas, que não seriam possíveis no espaço físico real. Deste modo a RV e RA tem se tornando mais populares, pela sua capacidade de auxiliar usuários em tarefas e contribuindo para a melhoria de seu desempenho por oferecer informações que o usuário não teria normalmente (ARAÚJO, 2018).

Já para Cardoso *et al.* (2014), a RA e RV vem ganhando destaque em diversas áreas de conhecimento, no qual o uso dessa tecnologia estimula e facilita a sua aquisição. Assim, possibilitando diversas maneiras de se ensinar e até de maneiras que fisicamente seriam complexas de se fazer, ou até impossíveis, podendo se adaptar diversos assuntos e temas a essa tecnologia. Com tudo, esse recurso se torna extremamente eficiente, pois possui a capacidade de exibir objetos com detalhes, no qual, não se torna necessário que se imagine como esse objeto seria ou agiria. Além de também permitir a interação com esses objetos.

Com tudo, um dos pontos mais importantes de qualquer aplicação de RV ou RA, é a forma de interação com o usuário, que traz os problemas que envolvem o desenvolvimento de tais aplicações, pois uma interação malfeita pode tornar o uso da aplicação complexa e pouco intuitiva. Uma forma de contornar esse problema é utilizar os gestos manuais para a interação humano-computador, pois esse tipo de interação torna a aplicação mais amigável, assim reduzindo o desconforto do usuário (ARAÚJO, 2018). Um bom exemplo de uma aplicação que proporciona essa interação é o Cat Explorer, que se trata de uma aplicação de RV, em que o usuário pode manipular um gato virtual possibilitando ver dentro do gato e analisar seus sistemas internos entre outros. Essa manipulação se dá por meio de menus e botões que são acionados pelo movimento das mãos, no qual o Leap Motion e essencial, pois é ele faz o *tracking* das mãos do usuário (ULTRALEAP, 2021).

O Leap Motion é um dispositivo que usa duas câmeras infravermelhas para capturar cerca de 2.000 fotos por segundo, e assim essas fotos passam por um software que identifica as mãos, dedos e antebraço, fazendo o *tracking* das mãos. *O tracking* e tão eficiente que o tempo de processamento para identificar as mãos, os dedos e os antebraços ocorrem em milésimos de segundo, fazendo o atraso ser imperceptível ao olho humano (ULTRALEAP, 2020)

Com isso, o intuito deste trabalho é disponibilizar uma aplicação para experimentar o uso de RA e Leap Motion para inspecionar peças do acervo da Exposição de História Natural Fritz Müller - FURB. Onde a RA seria utilizada para sobrepor modelos virtuais nas peças do acervo. Já, o uso do Leap Motion seria para explorar se gestos das mãos permitem interagir com estes modelos virtuais. Assim, o objetivo principal desse trabalho é disponibilizar uma aplicação para experimentar o uso de RA e Leap Motion para inspecionar peças do acervo da

Exposição de História Natural Fritz Müller - FURB. Já os objetivos específicos são: avaliar o uso de modelos virtuais sobrepostos as peças do acervo usando RA; verificar se gestos da mão possibilitam uma interação com os modelos virtuais; e analisar a eficácia da interação usando peças do acervo da Exposição de História Natural Fritz Müller – FURB.

2 FUNDAMENTAÇÃO TEÓRICA

Nessa seção são apresentados os assuntos que fundamentam o desenvolvimento da aplicação. A primeira subseção trata sobre museus, a segunda subseção trata sobre a RA e Leap Motion, a terceira subseção trata sobre modelos virtuais, e por fim, é abordado os trabalhos correlatos.

2.1 MUSEUS

De acordo com a ICOM (2022), os museus são instituições sem fins lucrativos que coletam, pesquisam, conservam, interpretam e expõem patrimônio material e imaterial. Além disso, os museus abertos ao público promovem a diversidade e a sustentabilidade, comunicam ética e, em conjunto com a comunidade, oferecem experiências diversas para educação, fruição, reflexão e compartilhamento de conhecimento. O patrimônio de um museu, também denominado acervo, pode conter uma ampla variedade de itens, como obras, móveis, locais ou qualquer outro objeto que represente algum valor histórico, cultural e/ou material. Com base nos tipos de itens presentes nos acervos, os museus podem ser classificados em diversas categorias, tais como museus de arte, museus tecnológicos, museus de ciência, museus ecológicos, museus históricos, entre outros (CONCEITO, 2020).

Um exemplo destacado de museu é o Museu de História Natural Charles Darwin, criado com o objetivo de divulgar as teorias de Charles Darwin sobre a origem e evolução da vida em nosso planeta. Além disso, o museu expõe acervos abrangendo temas como arqueologia, geologia, invertebrados terrestres e aquáticos, uma exposição sobre baleias e golfinhos, e anatomia animal (BOMBINHAS, 2019).

2.2 RA E LEAP MOTION

Ao abordar a RA, é necessário situá-la no contexto mais amplo da realidade misturada, definida como a sobreposição de objetos virtuais gerados por computação com o ambiente físico. Esta sobreposição é realizada por meio de dispositivos que mostram ao usuário os objetos virtuais em tempo real (TORI *et al*., 2006). A RA proporciona um ambiente que integra elementos virtuais ao ambiente físico, permitindo que o usuário, ao permanecer em seu ambiente físico, interaja de forma natural, sem a necessidade de treinamento ou adaptação. Essa abordagem visa enriquecer o ambiente do usuário, como exemplificado pela possibilidade de mobiliar virtualmente um apartamento vazio ou visualizar informações sobre um restaurante ao olhar para sua placa (TORI *et al*., 2006).

A RA envolve quatro aspectos fundamentais: renderização de alta qualidade, alinhamento preciso, orientação no mundo real e interação em tempo real. Para alcançar renderização de alta qualidade e interação em tempo real, é necessário o uso de dispositivos que atendam a esses requisitos (TORI *et al*., 2006). Quanto ao alinhamento e orientação precisos, três métodos são reconhecidos pela maioria dos autores: uso de marcadores de RA, definição de posição no espaço físico e baseado na localização do usuário (COSTA, 2020).

A interação em tempo real é crucial na RA, e diversos dispositivos facilitam essa interação, como o Leap Motion. O o Leap Motion é um dispositivo óptico de rastreamento de mão de 8 por 3 cm, que captura o movimento das mãos e dedos para permitir a interação natural com conteúdo digital. O Leap Motion rastreia mãos em uma zona interativa 3D, estendendo-se até 60 cm, distinguindo 27 elementos distintos das mãos, como ossos e articulações, mesmo quando obstruídos por outras partes da mão (ULTRALEAP, 2020).

2.3 MODELOS VIRTUAIS

De acordo com Tori *et al*. (2006), a criação de um modelo virtual requer a declaração de pontos que definem sua estrutura, conhecidos como vértices, com três coordenadas (x, y, z). Vale ressaltar que, na maioria das aplicações gráficas, a representação ocorre em uma superfície, implicando a conversão de dados tridimensionais para bidimensionais. No entanto, os vértices sozinhos não são suficientes para descrever modelos virtuais; eles servem como ancoragem para entidades geométricas, como retas e curvas, que, organizadas, originam os modelos virtuais. A modelagem visa impor ordem ao que inicialmente parece caótico.

A representação mais simples e tradicional de um modelo virtual é através de linhas que delimitam seu exterior, técnica conhecida como modelo de fio de arame. No entanto, apenas as ligações entre vértices podem gerar um modelo virtual que aparenta mostrar todas as suas superfícies. Assim, para uma representação mais precisa, é necessário ocultar arestas e superfícies não visíveis, definindo a ordem das ligações dos vetores de cada face. Isso permite mostrar somente as faces em orientação horária ao observador, enquanto as faces em orientação contrária permanecem invisíveis (TORI *et al*., 2006).

Para aplicações de realidade virtual, é desejável manipular dados de forma eficiente para agilizar a renderização dos modelos virtuais. Portanto, geralmente opta-se por definir apenas uma "casca" para o modelo, mostrando apenas o que será visível para o observador. Modelos virtuais baseados em superfícies manipulam principalmente o exterior do modelo e podem assumir diversas formas. As superfícies poligonais, paramétricas e quádricas são as mais utilizadas na literatura (TORI *et al*., 2006).

Superfícies poligonais são as mais simples e intuitivas de modelar, compostas por superfícies planas, como triângulos e quadriláteros. Elas são adequadas para modelar objetos com representação fácil, como caixas e prismas. A representação paramétrica utiliza três polinômios para as dimensões (x, y, z), variando com base em um parâmetro comum, gerando um traçado curvo entre os vértices para um contorno mais fidedigno. A quádrica é formada apenas por superfícies quadráticas, definidas por uma função do tipo `f(x, y, z) = 0`, sendo amplamente utilizada para representar esferas, cilindros, elipsoides e troncos paraboloides (TORI *et al*., 2006).

Para além da modelagem de superfícies, há a modelagem de sólidos, dividida em duas categorias: representações exatas e aproximadas. A primeira utiliza equacionamento matemático de volumes e superfícies para descrever a forma do modelo virtual de maneira precisa, embora apresente alto custo computacional. Já a segunda utiliza superfícies simples para representar modelos virtuais, combinando essas superfícies para que se assemelhem ao desejado, podendo também fazer uso de uma malha de polígonos conectados (TORI *et al*., 2006).

Um ponto crucial na modelagem é a iluminação, pois ela complementa o realismo dos modelos. A iluminação é classificada conforme sua origem, e para cada caso é definido um tipo de comportamento. Um comportamento amplamente utilizado é o da luz ambiente, onde a energia luminosa provém de todas as direções. Trata-se de um modelo de iluminação simples que utiliza a cor do modelo virtual refletida no ambiente. No entanto, o modelo de luz ambiente pode não proporcionar o realismo desejado. Nesse sentido, pode-se recorrer ao modelo de luz refletida, no qual o modelo virtual não emite luz própria, mas reflete a luz irradiada sobre ele. Essa luz refletida varia dependendo da composição, direção e geometria da fonte de luz (TORI *et al*., 2006).

Além da iluminação, é possível adicionar texturas às superfícies. A adição de textura consiste no processo de extrair informação gráfica de uma imagem, seja real ou não, e aplicá-la em determinadas superfícies, adicionando realismo ao modelo virtual. Assim, a textura, aliada à iluminação, proporciona maior realismo e possibilita um melhor desempenho no processo de renderização. Contudo, em ambientes virtuais, a adição de luz e textura começa a se tornar um fator de alto consumo de recursos computacionais, uma vez que as aplicações demandam movimento, exigindo uma constante atualização na renderização (TORI *et al*., 2006).

2.4 TRABALHOS CORRELATOS

Nessa seção, são apresentados os trabalhos correlatos que possuem características semelhantes ao objetivo proposto. No primeiro é descrito o trabalho de Cardoso *et al*. (2014), que desenvolve uma aplicação web para auxílio no ensino fazendo uso de RA para mostrar imagens aos alunos (Quadro 1). O segundo descreve o trabalho de Bento (2021), no qual teve como objetivo desenvolver uma aplicação de RV para desenho permitindo o usuário desenhar no ar com as mãos (Quadro 2). No terceiro é relatado um estudo feito por Valentini (2018) para a utilização do leap motion e RA no processo de aprendizagem de montagem virtual interativa (Quadro 3).

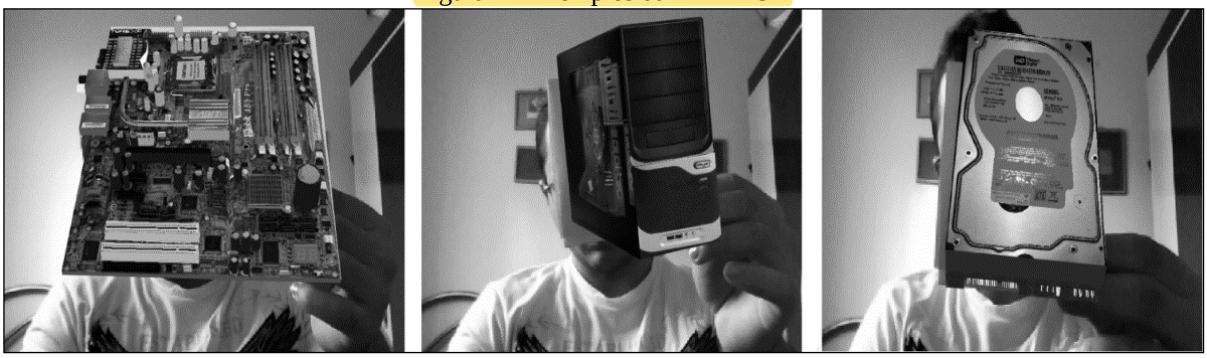
Quadro 1 - Uso da Realidade Aumentada em Auxílio à Educação

Referência	Cardoso *et al*. (2014)
Objetivos	Desenvolvimento da aplicação web utilizando RA, chamada RAINFOR, para o auxiliar os discentes no aprendizado por meio da interatividade proporcionada por RA
Principais funcionalidades	Permite o usuário ver imagens utilizando um marcador
Ferramentas de desenvolvimento	Foi desenvolvido na IDE *Flash Builder* utilizando a linguagem Adobe ActionScript, juntamente com a API FLARToolkit. Como também as linguagens PHP e Html para o desenvolvimento web
Resultados e conclusões	A aplicação foi testada em uma sala de aula em que foi constatado dos docentes o aumento da motivação ao aprenderem. Como também a facilitação da aprendizagem/fixação do conteúdo exposto.

Fonte: elaborada pelo autor.

A aplicação consiste em permitir que o usuário possa visualizar imagens do conteúdo que está sendo demonstrado utilizando RA. Se o usuário possui acesso a um computador com internet e uma webcam, ele pode acessar o site onde a aplicação está localizada e, ao mostrar o marcador para a webcam, é exibida na tela a imagem de algum dos componentes sobre o marcador.

Figura 1 - Exemplos da RAINFOR



Fonte: Cardoso (2014).

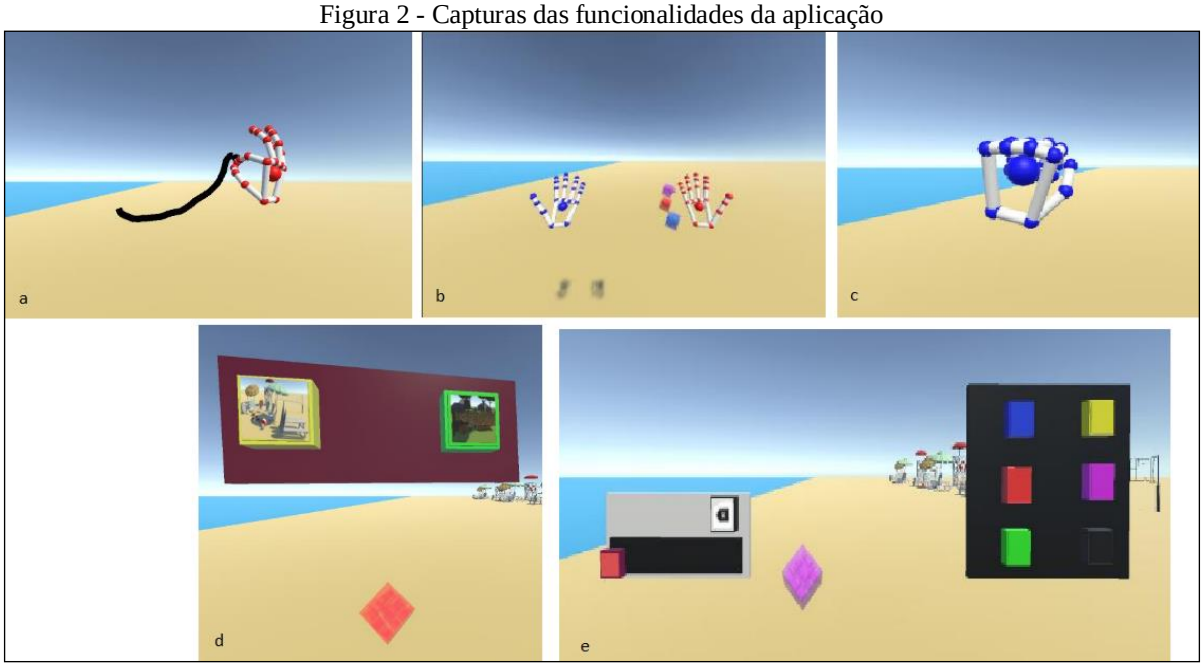
Quadro 2 - Um Aplicativo de Desenho em Realidade Virtual Utilizando o Leap Motion

Referência	Bento (2021)
Objetivos	Desenvolvimento de uma aplicação de realidade virtual utilizando o Leap Motion para a criação de desenhos virtuais.
Principais funcionalidades	Permite o usuário desenhar livremente no ar usando as mãos em um espaço virtual
Ferramentas de desenvolvimento	Desenvolvido no motor de jogos Unity.
Resultados e conclusões	Foram feitas duas rodadas de testes com usuário de 11 a 14 anos para testar a usabilidade da aplicação. Conseguindo bons resultados na maioria das funcionalidades, fazendo os usuários terem uma nova experiencia de desenho permitindo utilizar sua criatividade e percepção de mundo.

Fonte: elaborada pelo autor.

Na aplicação desenvolvida por Banto (2021), o usuário é inserido em um ambiente virtual que o possibilita usar as mãos para desenhar e interagir com menus, como pode ser visualizado na Figura 2. Ao fazer o movimento de pinça, o usuário pode desenhar no ar, e ao virar a palma da mão para si, é possível visualizar cubos que, ao interagir usando a outra mão, abrem menus para mudar a cor e espessura da linha desenhada, apagar a última linha desenhada, mudar o ambiente virtual e salvar o desenho.

Figura 2 - Capturas das funcionalidades da aplicação



Fonte: Bento (2021).

Quadro 3 - Natural interface for interactive virtual assembly in augmented reality using Leap Motion Controller

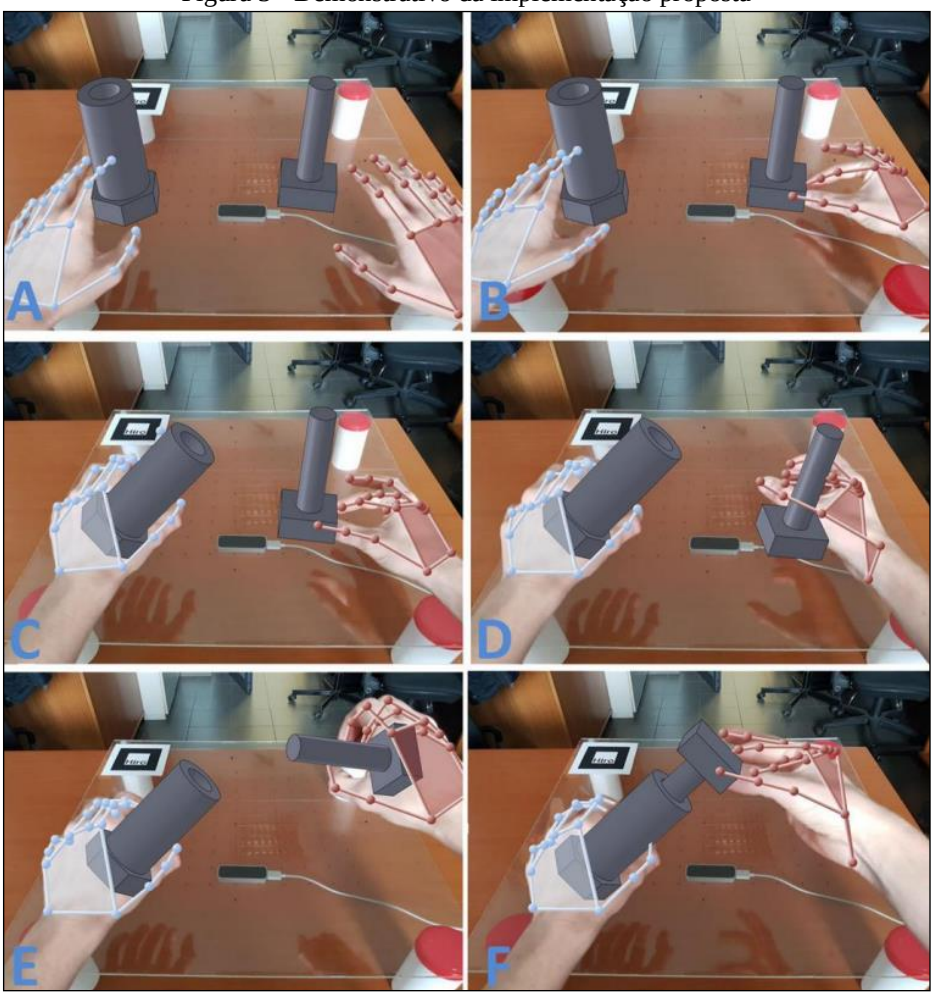
Referência	Valentini (2018)
Objetivos	Discutir a interação do Leap Motion com a RA para implementar uma metodologia de montagem virtual interativa.

Principais funcionalidades	Interagir com modelos 3D virtuais usando as mãos na montagem de peças.
Ferramentas de desenvolvimento	IDE não foi especificado. Foi utilizado o ARToolkit.
Resultados e conclusões	A eficácia da metodologia foi testada com um grupo de usuários de 20 a 40 anos que não possuíam experiencia anterior com o Leap Motion. Foi notado que após alguns minutos de utilização grande parte dos usuários se sentiram à vontade no ambiente de RA. Em que o autor finaliza considerando viável a possibilidade de utilização de RA e o Leap Motion para substituir ou ajudar no processo de aprendizagem de processos de montagem.

Fonte: elaborada pelo autor.

No trabalho de Valentini (2018), foi primeiramente realizado um estudo para definir os principais movimentos das mãos para manipular objetos a serem captados usando o Leap Motion. No qual foram definidas três poses de mão: a pose de pinça, cilíndrica e esférica. Em seguida, foi montada uma experimentação em RA contendo duas peças virtuais, e o usuário teria que encaixá-las usando as mãos, utilizando os gestos definidos, como pode ser visualizado na Figura 3.

Figura 3 - Demonstrativo da implementação proposta



Fonte: Valentini (2018).

3 DESCRIÇÃO DA APLICAÇÃO

Essa seção apresenta os detalhes de especificação e implementação da aplicação desenvolvida. Na primeira seção apresenta uma visão geral das principais funcionalidades da aplicação. Já na segunda seção são apresentados aspectos que englobam a implementação da aplicação.

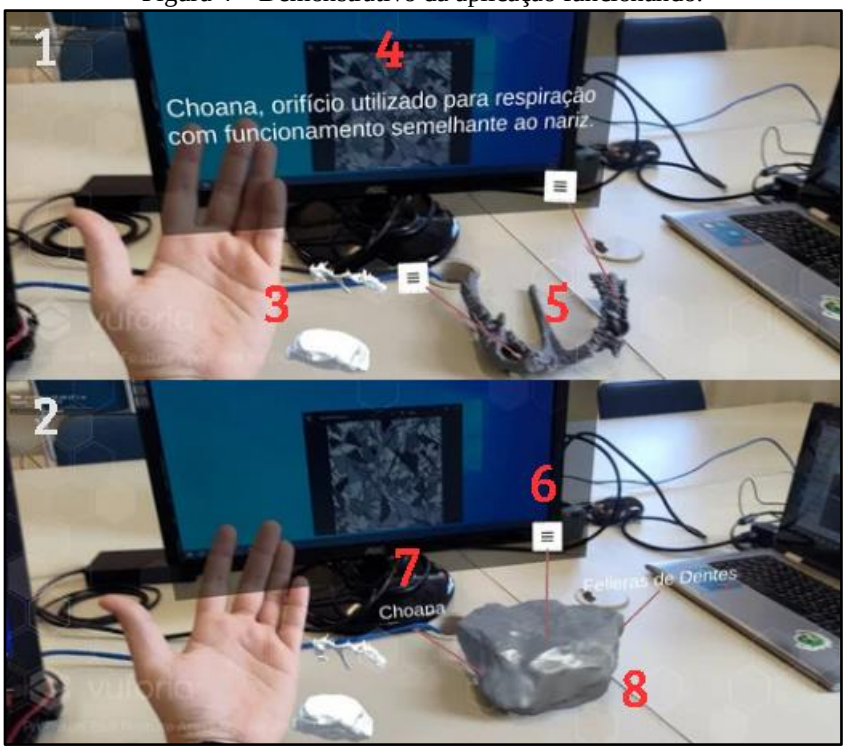
3.1 ESPECIFICAÇÃO

O FossilAR, aplicação desenvolvida, permite que o usuário possa interagir com modelos de objetos virtuais sobrepostos à visão do ambiente real em que ele se encontra. Para isso, a experimentação conta com um Head Mounted Display (HMD) equipado com o Leap Motion, uma impressão 3D de um fóssil utilizada como

marcador principal e um marcador secundário que mostra um painel virtual com informações. Contudo, a aplicação não possui um roteiro a ser seguido, deixando livre para o usuário interagir com o que quiser na ordem que desejar a partir do momento em que coloca o HMD.

Ao usuário colocar o HMD, pode-se pegar o fóssil nas mãos e manipulá-lo livremente, mas ao visualizar o fóssil, aparecerão botões ao redor dele para que o usuário possa pressioná-los usando as mãos, mostrando assim informações sobre o fóssil. As informações são exibidas em um painel virtual que pode ser visualizado tendo o segundo marcador no campo de visão do usuário, bem como a possibilidade de mudar o que está sendo mostrado na cena e no painel, ao selecionar miniaturas que estão ancoradas na mão esquerda do usuário, mas que só são mostradas quando a palma da mão esquerda está virada para o usuário, como pode ser visto na Figura 4.

Figura 4 – Demonstrativo da aplicação funcionando.



Fonte: Capturado pelo autor.

Na Figura 4, é possível visualizar as duas possibilidades de cenas, representadas pelos itens (1) e (2). No item (1), o usuário pode interagir com o fóssil e obter informações sobre características visíveis do mesmo, enquanto no item (2) é adicionado um modelo do fóssil ainda na rocha, permitindo ao usuário obter informações sobre a descoberta do mesmo. Já no item (3) são demonstradas as miniaturas ancoradas na mão esquerda do usuário, as quais só são visíveis caso o usuário vire a palma da mão esquerda para si. Essas miniaturas controlam qual das duas possíveis cenas o usuário verá e interagirá. Ao utilizar a mão direita para transpassar a miniatura do fóssil, a cena do item (1) torna-se visível, e ao transpassar a miniatura do bloco, a cena do item (2) torna-se visível.

O item (4 da Figura 4 apresenta o marcador ao qual o painel virtual está ancorado. Esse painel virtual serve para mostrar informações ao usuário, dependendo de qual cena está visível e de qual dos botões em volta do marcador do fóssil foi pressionado. Além disso, as informações do painel são removidas a cada mudança de cena. Já o item (5) mostra o marcador principal da aplicação, que é o principal componente de interação. Esse marcador consiste em um modelo 3D impresso de um fóssil, extraído da rocha onde ele se encontra por meio de tomografia.

E por fim, o item (6) ainda da Figura 4 demonstra os botões com os quais o usuário pode interagir. Na cena do item (1), existem dois botões com linhas que os ligam às partes, onde cada um dos botões mudará para informações específicas no painel virtual. Contudo, na cena o item (2), há apenas um botão que se liga à rocha e serve para mostrar as informações sobre ela também no painel virtual. Além disso, surgem dois textos que se ligam às partes apontadas pelos botões da cena do item (1), como pode ser visto no item (7). Por fim, no item (8) demonstra como o modelo da rocha fica sobreposto ao marcador do fóssil.

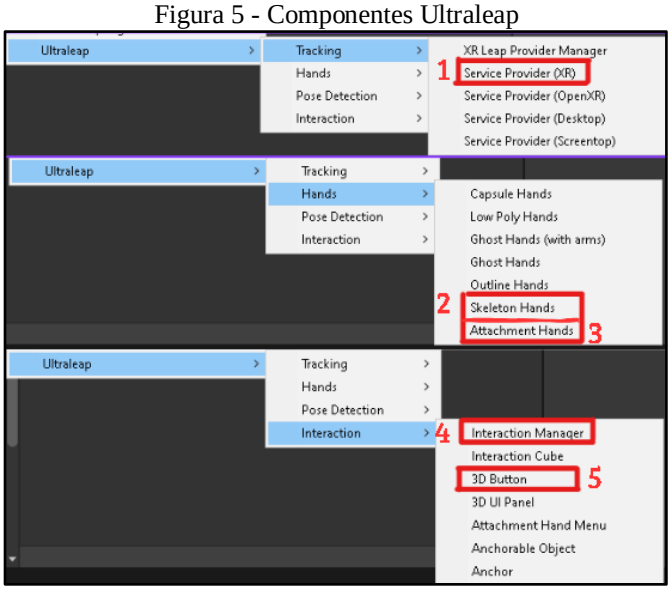
3.2 IMPLEMENTAÇÃO

A aplicação foi desenvolvida utilizando o motor gráfico Unity (versão: 2021.3.11f1) juntamente com o ambiente de desenvolvimento Visual Studio Community 2019 (versão: 16.11.29), ambos disponíveis de maneira gratuita. Além disso, foram utilizadas três principais bibliotecas: Ultraleap Tracking para a captura da posição das

mãos, Vuforia Engine AR para a parte da câmera e marcadores de AR, e o Mirror para a comunicação entre os dispositivos.

A biblioteca Ultraleap Tracking disponibiliza GameObjects e *scripts* prontos para permitem a comunicação entre a aplicação e o software externo (também chamado Ultraleap Tracking), bem como para utilizar as informações na criação das interações. Foram utilizados cinco principais GameObjects dessa biblioteca, como demonstrado na Figura 5.

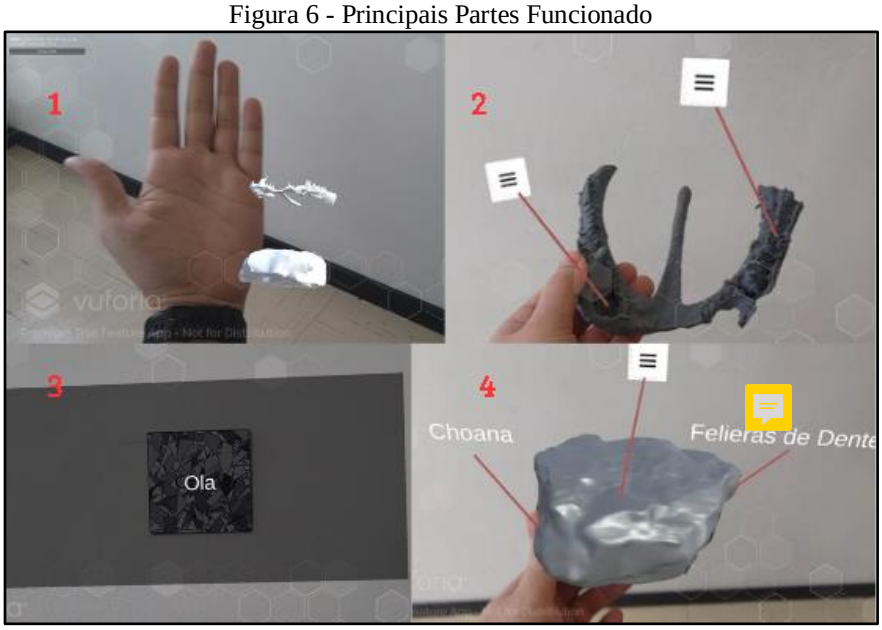
Figura 5 - Componentes Ultraleap



Fonte: Elaborado pelo Autor.

O `GameObject (1)` serve como entrada dos dados provenientes do software Ultraleap Tracking, permitindo a opção de selecionar como o aparelho Leap Motion seria posicionado no mundo real. Para a posição anexada na frente do HMD, foi usada a opção XR. Já o `GameObject (2)` representa as mãos reais no mundo virtual, controladas pelo `Service Provider`, sendo escolhida a opção `Skeleton Hands` em detrimento das demais, devido à sua *hitbox* mais próxima das mãos reais e à preferência do autor. O componente `(3)` funciona como uma extensão das `Skeleton Hands`, permitindo ancorar outros `GameObjects` em vários pontos das mãos, sendo utilizado para ancorar os botões na mão esquerda do usuário. O componente `(4)` é responsável por gerenciar quais dedos serão utilizados para as interações, se eles podem colidir com outros componentes e se podem agarrar outros `GameObjects`, sendo possível configurar cada mão separadamente. Por fim, o `(5)` é um botão pronto que possui um *script* próprio para lidar com as interações das mãos, podendo adicionar eventos que chamam métodos de outros *scripts*, tais como `OnPress()` para quando for pressionado e `OnUnpress()` para quando deixar de ser pressionado. Uma demonstração do uso dessa biblioteca pode ser vista no `(1)` da Figura 6, onde dois modelos ficam ancorados na mão do usuário.

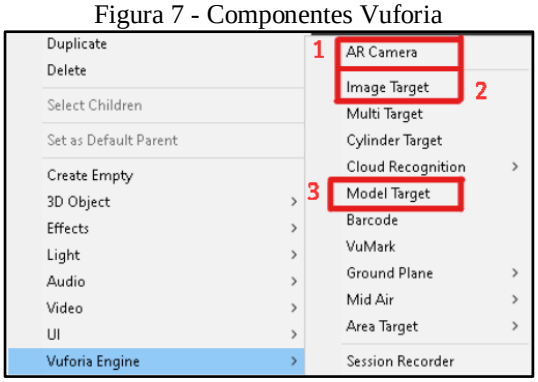
Figura 6 - Principais Partes Funcionado



Fonte: Elaborado pelo Autor.

A biblioteca Vuforia disponibiliza diversos `GameObjects`, bem como *scripts* que permitem a criação de aplicações e jogos empregando RA. Na aplicação proposta, foi necessário aplicar duas principais funcionalidades: a da câmera, que serviria de entrada para a RA, e os marcadores, para que a aplicação pudesse sincronizar o mundo virtual e o real. Os `GameObjects` utilizados podem ser vistos na Figura 7.

Figura 7 - Componentes Vuforia



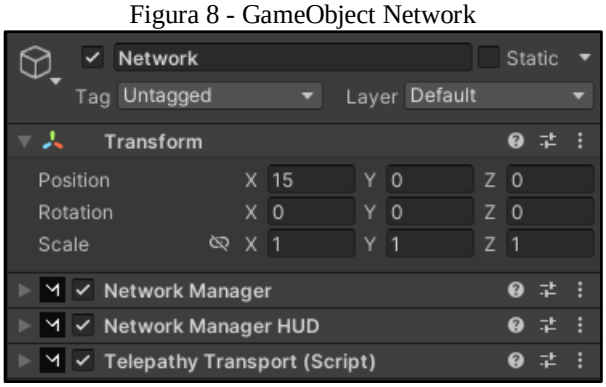
Fonte: Elaborado pelo Autor.

O `GameObject` (1) da Figura 7 refere-se ao `GameObject` responsável por criar uma câmera que realizará o trabalho de criar a visualização da RA. Esta câmera se conecta com a engine do Vuforia, que captura as imagens da câmera real e as coloca como fundo do que a câmera virtual está vendo. Já o `GameObject` (2) trata-se de um marcador de imagem, permitindo colocar outros `GameObjects` como filhos dele para que sejam mostrados quando esse marcador for identificado. Em contrapartida, o `GameObject` (3) funciona de forma semelhante ao `Image Target`, mudando apenas que não será mais usada uma imagem para fazer a identificação, mas sim um modelo 3D. Uma demonstração do funcionamento dos marcadores pode ser visualizada nos itens (2), (3) e (4) da Figura 6.

A biblioteca Mirror não adiciona `GameObjects`, mas vários componentes que permitem criar conexões entre instâncias da mesma aplicação, transferindo dados entre elas. Na aplicação, foi criado um `GameObject` chamado `Network` que continha certos componentes do Mirror responsáveis por criar a conexão entre as instâncias da aplicação.

O `Network Manager` serve para configurar e gerenciar a troca de informações de cada instância, além de ser responsável por criar os `players` no mundo virtual. Também temos o `Network Manager HUD`, que cria uma pequena interface com opções de iniciar um Anfitrião e cliente (*host and client*), permitindo que essa instância atue tanto como servidor para que as outras instâncias se conectem, quanto como um jogador. Há também uma segunda opção de se conectar como um jogador em um servidor, indicando o Ipv4 de destino, contanto que ambas as instâncias estejam na mesma rede. Por último, uma terceira opção que inicia somente o servidor. Além disso, temos o componente `Telepathy Transport`, que se refere ao tipo de transferência de dados que ocorrerá entre o servidor e os jogadores, sendo este o mais recomendado pela própria documentação do Mirror para grandes e pequenas transferências de dados, garantindo a entrega dos dados. O `GameObject` Network e os componentes citados podem ser visualizados na Figura 8.

Figura 8 - GameObject Network



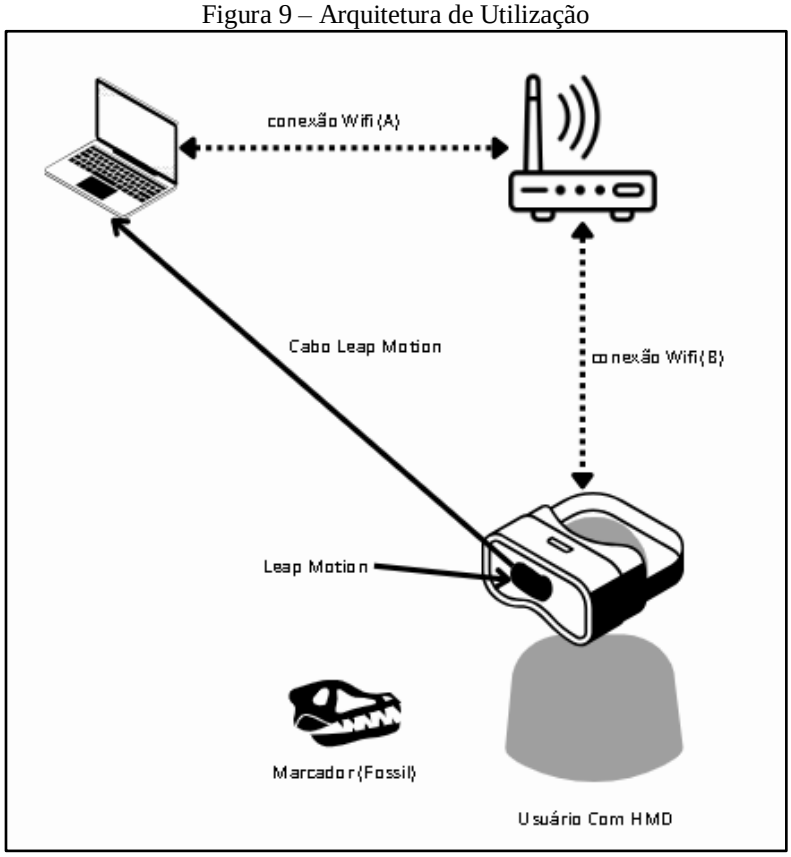
Fonte: Elaborado pelo Autor.

A aplicação necessita de duas instâncias para funcionar, uma no Windows e outra no Android. A instância do Android fica responsável por realizar a captura das imagens da câmera real, bem como identificar os marcadores e posicioná-los no mundo virtual. Já no Windows, sua principal função é capturar a posição das mãos do usuário e posicioná-las no mundo virtual, além de detectar as interações das mãos com os demais objetos. Para que as duas instâncias funcionem corretamente, é necessário que haja uma sincronização entre as duas partes, pois cada uma delas obtém as posições dos principais `GameObjects` com base no mundo real, como a câmera, os

marcadores e as mãos. Cada instância precisa enviar dados diferentes uma para a outra, sendo que o Android envia os dados de posição e rotação de sua câmera e do marcador do fóssil, enquanto o Windows envia a posição dos botões que ficam ancorados na mão esquerda do usuário e as interações das mãos com os botões.

Essa arquitetura pode ser visualizada na Figura 9. Nessa configuração, o computador atua como servidor e cliente Windows, conectando-se com o celular por meio de uma rede Wi-Fi. O Leap Motion se conecta ao computador por cabo e fica posicionado na frente do HMD. Por fim, o celular está dentro do HMD, de forma que sua câmera fique posicionada em uma abertura feita para captar o que está situado na frente do usuário.

Figura 9 – Arquitetura de Utilização



Fonte: Elaborado pelo Autor.

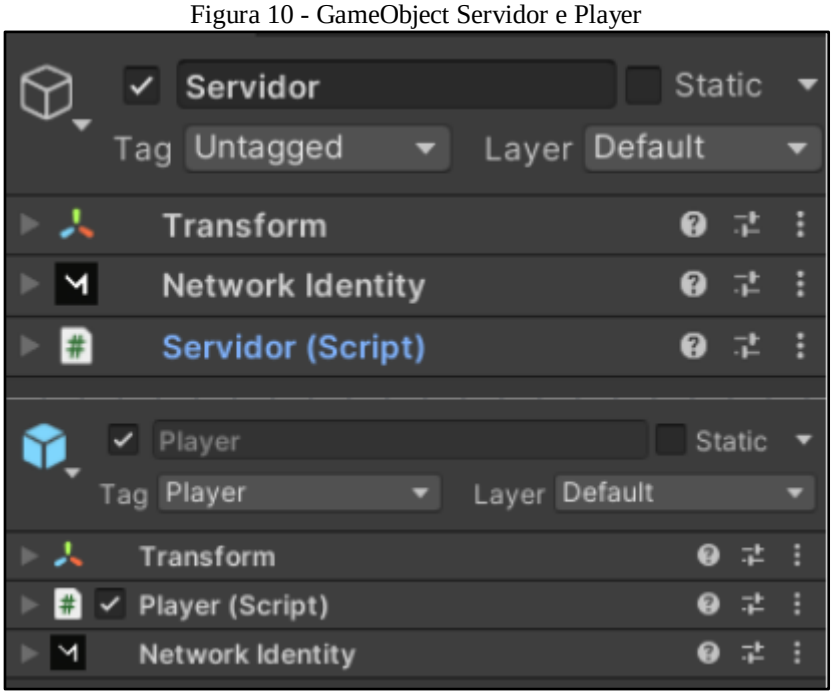
Para a sincronização, foram criados dois `GameObjects`: o `servidor` e o `player`. O `servidor` tem a função de manter as variáveis sincronizadas com o servidor e, caso uma delas seja alterada, executar os métodos adequados para atualizar a cena. Por outro lado, o `player` é responsável por fazer as alterações, pois é ele que tem a autoridade de alterar as variáveis sincronizadas no `servidor`. Conforme pode ser visualizado na Figura 10, o `servidor` é composto por dois componentes: o `Network Identity` e o *script* do `servidor`. O primeiro é um componente do Mirror que tem a função de autenticar esse `GameObject` no servidor, permitindo que a comunicação aconteça.

O *script* `servidor` armazena as variáveis necessárias para a sincronização entre as instâncias, que são a posição e rotação da câmera virtual, marcador, botões do marcador e o painel de botões na mão, assim como as variáveis que indicam qual botão foi pressionado e se os botões da mão estão visíveis. Além disso, o `servidor` também armazena referências a `GameObjects` que precisam ser alterados caso determinadas variáveis tenham seu valor alterado, e métodos vinculados a cada variável para executar as alterações. O `servidor` também armazena uma variável *booleana* que serve para saber se a aplicação está sendo executada no Windows ou no Android. A necessidade dessa variável se dá devido ao funcionamento da sincronização de dados no Mirror, em que ao atualizar o valor de uma variável todos os usuários executam o método para atualizar o correspondente `GameObject`. No entanto, na aplicação, era necessário que apenas uma das instâncias atualizasse a variável e a outra apenas atualizasse o `GameObject`. Por isso, cada método de atualização de `GameObjects` possui uma verificação para determinar se devem ou não atualizar o `GameObject`, sendo o *script* `player` responsável por determinar qual instância deve atualizar as variáveis.

O `Player` não está presente na cena desde o início; na verdade, ele é um `Prefab`, ou seja, um `GameObject` que pode ser criado por meio de código. O componente `Network Manager` é quem cria esses `GameObjects`, criando um para sua instância quando se conecta com o `servidor`. Ele é composto por dois componentes: o `Network Identity` e o *script* `player`, como demonstrado na Figura 10. O *script* `player`

fica responsável por verificar as mudanças de posições dos `GameObjects` marcador, câmera e botões, assim como verificar se algum dos botões foi pressionado, acionando o comando para alterar as devidas variáveis no `servidor`. Essas verificações ocorrem no método `Update()`, fazendo uso da variável booleana do `servidor` que indica se está no Windows ou Android para chamar os devidos métodos de verificação. No caso de estar sendo executado no Windows, uma das verificações que o *script* realiza é a posição do painel de botões da mão; em contrapartida, se for no Android, verifica a posição do marcador da câmera e dos botões do marcador. Para tal funcionamento, o *script* precisa da referência do `servidor` de sua instância e dos `GameObjects` que precisam ser verificados, sendo o único *script* com permissão para alterar as variáveis sincronizadas no `servidor`.

Figura 10 - GameObject Servidor e Player

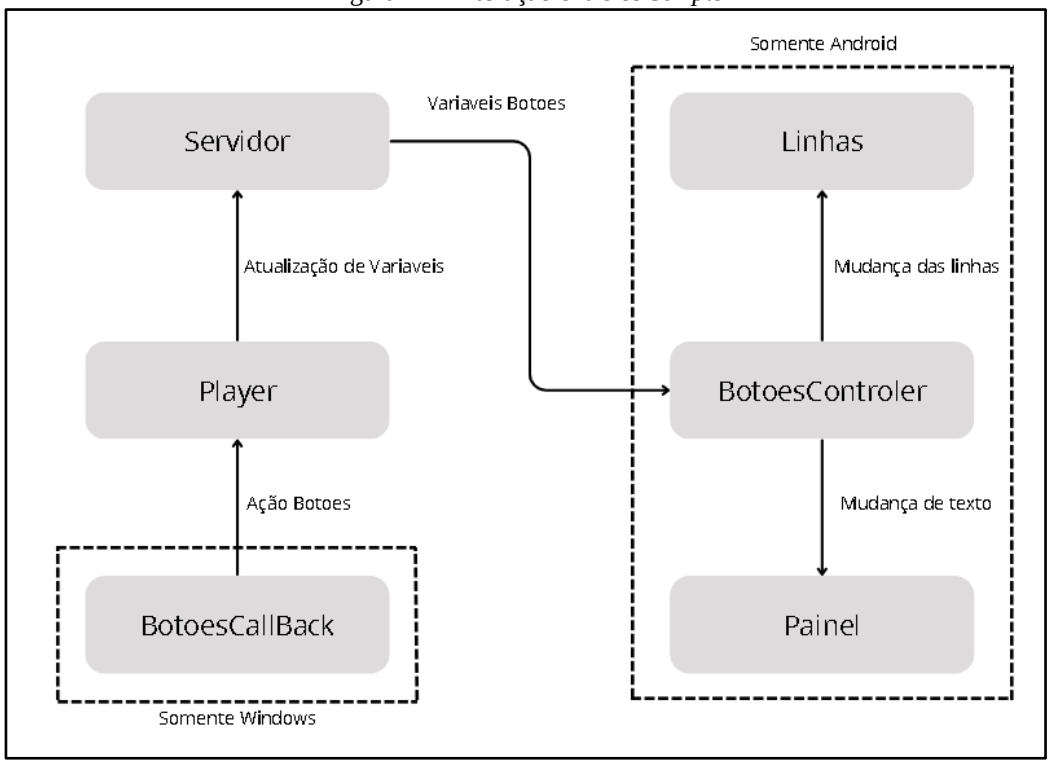


Fonte: Elaborado pelo Autor.

Além desses dois *scripts*, foram criados mais quatro que lidam com as interações do usuário, que são o `BotoesCallBack`, `BotoesControler`, `Painel` e o `Linhas`. O `BotoesCallBack` tem o propósito de servir como uma ponte entre o *script* do `Button 3D` do Leap Motion e o `player`, pois o *script* do botão possibilita adicionar uma chamada de um método de um `GameObject` caso o botão seja pressionado, deixe de ser pressionado, seja desabilitado, entre outros. Como um `player` só existe a partir do momento que uma instância se conecta com o servidor, não teria como adicionar a chamada de um método do `player`. Para que esse *script* funcionasse, os botões foram divididos em dois grupos: os botões do marcador e os da mão, em que cada um dos botões do grupo possui um índice que começa em 1. O `BotoesCallBack` possui um método para cada grupo que recebe um valor *int*, em que ao botão ser pressionado passa o índice correspondente, mas quando o botão deixa de ser pressionado ele passa o índice 0 para que possa saber que não tem mais nenhum botão daquele grupo pressionado. Por fim, esses métodos chamam seus respectivos métodos no `player` que atualiza a devida variável no `servidor`, propagando a informação até que os outros *scripts* que geram a resposta visual ao usuário. A interação entre os *scripts* criados pode ser visualizada na Figura 11.

O *script* `BotoesControler` tem a função de controlar como cada ação de um botão vai alterar os objetos e informações que o usuário vai visualizar. Ao acionar um dos botões no marcador, é acionado o método `AcionaBotaoMarcador` que recebe o índice do botão e realiza a ação de pressionar o botão, como mudar o texto que está sendo mostrado no painel. Caso o índice seja 0, ele desfaz a ação de pressionar. De forma similar, o método `AcionaBotaoPainel` lida com o pressionar dos botões da mão, mas, neste caso, não há a ação de pressionar o botão, mas sim de mudar a cor dos objetos ancorados na mão do usuário. Caso o fóssil seja selecionado, as informações no painel de informações são removidas, e dois botões são exibidos ao redor do marcador do fóssil, cada um com uma linha apontando para uma determinada parte do fóssil com informações relevantes. Se o modelo da rocha for selecionado, é adicionado um modelo 3D do fóssil ainda dentro da rocha, além de ficar somente um dos dois botões com uma linha apontando para a rocha. Também surgem dois textos que se referem às partes do fóssil que possuem informações relevantes, usando linhas para apontar para elas.

Figura 11 - Interação entre os *Scripts*



Fonte: Elaborado pelo Autor.

O *script* `Painel` tem o propósito de guardar os textos com as informações que serão mostradas ao usuário, assim como dispor essas informações no painel. Ele possui apenas dois métodos simples: um que recebe qual foi o último botão da mão pressionado, juntamente com o último botão do marcador pressionado, para saber qual informação colocar no painel e um método para remover as informações do painel.

Já o *script* `Linha` tem a função de desenhar linhas entre determinadas posições, onde possui dois vetores: um para guardar a posição inicial da linha e outro a posição final. Ele possui somente um método público que o `BotoesControler` pode chamar, que altera a posição de alguns dos elementos do vetor que guarda as posições finais para que cada um dos botões e textos aponte para os locais certos.

4 RESULTADOS

Esta seção apresenta os testes realizados com o jogo e a comparação com os correlatos. Na primeira seção são discutidos os testes de funcionalidades do jogo, realizados durante o desenvolvimento. Na segunda seção são apresentados os resultados dos testes realizados com usuários. Na terceira é apresentado um comparativo da aplicação desenvolvida com os trabalhos correlatos.

4.1 TESTES DE FUNCIONALIDADES

Os testes de funcionalidade foram realizados de forma contínua durante o desenvolvimento da aplicação, a fim de assegurar a conformidade das funcionalidades com as expectativas estabelecidas. Estes testes foram predominantemente conduzidos no ambiente de desenvolvimento do Unity, tanto para a instância que seria executada no sistema Windows, quanto para a destinada ao sistema Android. Além disso, foram realizados testes de comunicação e execução entre as instâncias do Windows e do Android, sendo executados em um notebook e em um smartphone, respectivamente.

A principal finalidade da aplicação era utilizar o celular exclusivamente como meio de captura de câmera e visualização para o usuário, deixando todo o processamento de colisões, interações, rastreamento das mãos e detecção dos marcadores a cargo do computador. Essa abordagem inicial foi escolhida devido à maior capacidade de memória e processamento do computador, bem como à necessidade de utilizar a aplicação Ultraleap Tracking, que realiza a comunicação com o dispositivo Leap Motion e só pode ser utilizada em sistemas operacionais Windows, Linux e OS, com exceção de dispositivos Android com um processador específico.

No entanto, essa abordagem inicial revelou-se falha devido à incompatibilidade das bibliotecas Leap Motion e Vuforia em relação às possibilidades de exportar a aplicação do ambiente de desenvolvimento Unity para o Windows. O Unity disponibilizava duas maneiras de exportar uma aplicação para Windows: a versão standalone, que também poderia ser executada em outros sistemas operacionais, e a versão Universal Windows Platform (UWP). Foram realizados testes de exportação em ambas as versões, e constatou-se que a biblioteca Leap

Motion funcionava na versão standalone, mas a biblioteca Vuforia não; na versão UWP, ocorria o oposto. Isso se devia ao fato de que a biblioteca Vuforia necessita de permissão para acessar os dispositivos de câmera do Windows, permissão essa concedida apenas pela versão UWP. Por outro lado, a biblioteca Leap Motion não funcionava na versão UWP devido à impossibilidade da aplicação estabelecer comunicação com a aplicação Ultraleap Tracking.

Então se adotou uma outra abordagem, mantendo o uso do computador e do celular. Em vez de o celular servir apenas como dispositivo de captura e visualização, ele passou a desempenhar o papel de detecção dos marcadores, visualização e posicionamento dos `GameObjects`. O computador, por sua vez, ficou responsável por detectar os movimentos das mãos e as interações das mãos com os demais `GameObjects`. Para isso, foi necessária uma sincronização entre as duas instâncias, para a qual se optou por utilizar a biblioteca Mirror, que permite estabelecer uma conexão entre as instâncias (ver 3.2 – Figura 9).

Com a adoção dessa nova abordagem, foi identificado um grande problema de sincronização entre as duas instâncias, resultando em `GameObjects` posicionados de maneira a não serem visíveis para o usuário, ou em posições que não permitiam interações. Para resolver esse problema, foram adotadas três soluções principais. A primeira consistiu em manter uma hierarquia semelhante entre os `GameObjects` em ambas as instâncias. A segunda solução envolveu a realização de testes para determinar se era necessário utilizar a posição do `GameObject` em relação à referência global da cena ou à referência local do `GameObject`. Por fim, foram criados `GameObjects` intermediários na hierarquia dos `GameObjects`, cujas posições foram sincronizadas para permitir pequenos ajustes manuais de posição, rotação e escala.

Foi observado durante os testes que, devido ao constante cálculo da posição dos marcadores pelo Vuforia, as transformações dos `GameObjects` apresentavam pequenas variações. Cada mudança nos valores resultava em vários comandos de sincronização na outra instância, levando os `GameObjects` a oscilações constantes. Para resolver esse problema, foram criados códigos que verificavam se a mudança na posição dos `GameObjects` sincronizados ficava abaixo de um determinado valor. Quando essa variação excedia esse valor, os comandos de sincronização eram acionados.

Além disso, durante os testes, foi observado que o marcador do fóssil não era detectado em determinadas posições. Isso ocorreu devido à falta de pontos de referência para algumas partes do fóssil, bem como à semelhança visual entre certas posições e suas posições opostas, tornando difícil a distinção entre elas.

4.2 TESTES COM USUÁRIOS

Após o desenvolvimento, a aplicação foi instalada na Exposição de História Natural Fritz Müller – FURB, ficando disponível para que qualquer visitante pudesse utilizar. Os testes foram realizados com 5 participantes na presença do autor. Antes dos testes, foi dada uma breve explicação dos componentes da aplicação e sua posição; em seguida, os participantes puderam usar a aplicação. Após terem utilizado todos os recursos disponíveis na aplicação, foi solicitado o preenchimento de um questionário que continha duas etapas. A primeira etapa consistia em recolher informações sobre o perfil dos participantes; as respostas podem ser visualizadas no Quadro 4

Quadro 4 - Perfil dos participantes

Idade	18	20%
	21	20%
	25	40%
	33	20%
Você utiliza dispositivos móveis com qual frequência?	Nunca utilizei	0%
	Às vezes	20%
	Frequentemente	80%
Indique seu grau de familiaridade com Realidade Aumentada:	Nunca ouvi falar	20%
	Conheço, mas nunca utilizei	20%
	Já utilizei	60%

Fonte: elaborado pelo autor.

Na segunda etapa do questionário, foram incluídas perguntas relacionadas à proposta da aplicação e à usabilidade das funções. Para as respostas de usabilidade, foi utilizada uma escala de 1 a 5, onde 1 representa "ruim" e 5 representa "bom", além de possuir um campo de observações. As perguntas juntamente com as respostas dos participantes podem ser visualizadas no Quadro 5.

No que diz respeito à proposta da aplicação, todos concordaram que a aplicação ajudou a despertar interesse e melhorar a compreensão sobre o fóssil, e que a utilização dessa abordagem pode ajudar nas explicações e compreensão de assuntos relacionados a este tema. A maioria dos participantes achou a aplicação intuitiva e fácil de usar, e não sentiram desconforto ao utilizar o HMD. As observações feitas sobre ele foram de uma primeira estranheza ao utilizá-lo, mas que passava após alguns minutos de utilização.

Os participantes classificaram sua experiência com os botões do fóssil e os da mão como mediana a boa. Durante os testes, foi necessário oferecer um pequeno auxílio para que fosse possível realizar as interações com mais facilidade, pois foi notada uma dificuldade em perceber a real posição dos botões devido à falta de oclusão sobre as mãos. Em relação à visualização das informações no painel virtual, todos os participantes a classificaram como boa, informando durante a utilização que não apresentavam dificuldades para ler as informações tanto do painel como as do fóssil. Além disso, mencionaram que a qualidade gráfica dos componentes da aplicação foi satisfatória.

Um dos problemas notados durante a montagem e os testes foi que o comprimento dos cabos limitava a movimentação dos participantes, sendo o cabo do Leap Motion o responsável. Ao tentar utilizar um extensor, o aparelho não funcionava, assim seria necessário comprar um cabo mais comprido e que tivesse uma taxa de transferência de dados superior a um USB 2.0. Além disso, o smartphone não conseguia se manter carregado durante a execução constante da aplicação, mesmo estando conectado ao carregador, sendo um dos motivos o constante processamento do Vuforia.

Quadro 5 - Opinião dos entrevistados sobre o aplicativo

Você acha que a aplicação ajudou a melhorar a compreensão do conteúdo relacionado ao fóssil?	Sim Não	100% 0%
A ferramenta conseguiu despertar em você interesse em conteúdos ou assuntos relacionados a Fósseis?	Sim Não	100% 0%
Você acha que com esta abordagem para a demonstração de conteúdos relacionados ao Fósseis, possa ajudar na explicações e compreensão de assuntos relacionados a este tema?	Sim Não	100% 0%
De modo geral, você achou a aplicação intuitiva e fácil de usar?	1 4	20% 80%
Como você classificaria a sua experiência com os botões de interação em volta do fóssil?	3 4	40% 60%
Como você classificaria a sua experiência com os objetos de seleção da mão esquerda?	4 5	80% 20%
Você se senti-o confortável utilizando o Head Mount Display?	Sim Não	60% 40%
Qual o grau de facilidade de visualização das informações no painel virtual da aplicação?	5 4	80% 20%
O que você achou da qualidade dos gráficos da aplicação?	5 4	80% 20%
Como você avaliaria a sua experiência com o FossilAR?	4 5	60% 40%

Fonte: elaborado pelo autor.

No geral, os resultados foram considerados satisfatórios, visto que a maioria dos participantes demonstrou interesse em poder interagir com o fóssil usando as mãos. Alguns participantes tiveram dificuldades em interagir com os botões, e em saber quais eram as possíveis interações que a aplicação oferecia. Esses pontos, em conjunto com os comentários ao final do questionário, demonstram que a aplicação possui vários aspectos passíveis de melhoria que podem ser explorados em trabalhos futuros.

4.3 COMPARATIVO COM OS TRABALHOS CORRELATOS

O Quadro 6 apresenta um comparativo entre os trabalhos correlatos. Como pode ser observado, o trabalho de Cardoso et al. (2014) e Valentini (2018) propõe-se a criar aplicações que permitam aos usuários experimentarem uma maneira mais interativa de adquirir novos conhecimentos por meio da RA. Por sua vez, Bento (2021) se propõe a criar uma experiência de imersão em RV que estimule a criatividade do usuário, permitindo-o desenhar usando as mãos.

Essas aplicações utilizam diferentes ferramentas para criar os ambientes de interação. Cardoso et al. (2014) faz uso do FLARToolkit e Papervision 3D para proporcionar uma aplicação que possa ser executada em navegadores, proporcionando uma maior facilidade de acesso a ela. A aplicação de Bento (2021) utiliza o Unity, um motor gráfico robusto que permite alterar cenas e adicionar objetos, além de várias bibliotecas para as mais diversas funcionalidades, permitindo assim a adição de dispositivos à aplicação com o Leap Motion. Por sua vez,

Valentini (2018) faz uso do ARToolkit, uma biblioteca feita para facilitar a criação de aplicações de RA, proporcionando várias funções prontas para agilizar e facilitar o processo de criação em RA. Quanto ao FossilAR, este utiliza o Unity para a criação do ambiente interativo com o usuário e a conexão com o Leap Motion, bem como o Vuforia para detecção dos marcadores.

Quadro 6 - Comparativo com os correlatos

Trabalhos Correlatos / Características	Cardoso *et al.* (2014)	Bento (2021)	Valentini (2018)	FossilAR
Desenvolve uma aplicação de RA	Sim	Não	Sim	Sim
Utiliza o Leap Motion	Não	Sim	Sim	Sim
Utiliza de marcador de RA	Sim	Não	Sim	Sim
API de RV / RA	FLARToolkit	-	ARToolkit	Vuforia
Renderização dos modelos virtuais	Papervision 3D	Unity	-	Unity
Linguagem utilizada	Adobe ActionScript, Html e PHP	C#	-	C#

Fonte: elaborado pelo autor.

5 CONCLUSÕES

Com base nos resultados obtidos, a aplicação alcançou seu objetivo de utilizar Realidade Aumentada em conjunto com o Leap Motion para criar uma experiência de interação usando as mãos para a Exposição de História Natural Fritz Müller - FURB. Os participantes também se mostraram motivados e interessados em aprender mais sobre os fósseis. No entanto, é importante ressaltar que muitos pontos de melhoria foram encontrados nos testes, o que confere à aplicação um potencial significativo de melhoria, sendo um dos principais pontos a adição da oclusão das mãos do usuário, para que possa melhorar o entendimento do usuário em relação à posição dos botões e demais componentes.

O Unity se provou uma ferramenta mais do que suficiente para o desenvolvimento em Realidade Aumentada, atendendo todas as necessidades com excelência. No entanto, a utilização de certos componentes, `GameObjects` e configurações apresenta uma curva de aprendizado mais acentuada, mas a comunidade existente ao redor do Unity facilitou o aprendizado e a correção de erros. Outro ponto a ser comentado é o fato do Unity utilizar a linguagem C#, que, por ser similar à linguagem Java, na qual já possuía um conhecimento prévio, facilitou o entendimento de *scripts* existentes e a criação de novos.

Quanto às bibliotecas utilizadas, o Vuforia se mostrou muito simples e fácil de utilizar e modificar, sendo o ponto que apresentou a maior dificuldade a geração e utilização de um marcador com base em um modelo 3D. O Ultraleap Tracking foi capaz de atender às necessidades de gerar interações das mãos do usuário com os demais objetos em cena. Por fim, o Mirror se apresentou mais do que capaz de permitir uma conexão entre dispositivos diferentes e manter cenas e objetos sincronizados.

Para possíveis extensões desse trabalho, ressalta-se principalmente a adição da oclusão das mãos do usuário, adição de novos fósseis para interação, procurar alternativas para melhorar a identificação do marcador do fóssil, adição de efeitos sonoros, procurar maneiras diferentes de demonstrar as informações ao usuário (como as da aplicação Cat Explorer), tornar o HMD mais confortável, acionar a estereoscopia na visualização do usuário e procurar maneiras de sincronizar as cenas de maneira mais precisa.

6 REFERÊNCIAS

ARAÚJO, Alexandre de Carvalho. **Interação gestual usando o Leap Motion para visualização em Realidade Aumentada através do Meta 2**. 2018. 33 f. Monografia (Especialização) - Curso de Ciência da Computação, Universidade Federal do Maranhão (Ufma), São Luiz, 2018.

BENTO, Gabriel Brogni. **Um Aplicativo de Desenho Em Realidade Virtual Utilizando o Leap Motion**. 2021. 17 f. TCC (Graduação) - Curso de Ciência da Computação, Departamento de Sistemas e Computação, Universidade Regional de Blumenau (FURB), Blumenau, 2021.

BOMBINHAS, Portal de Turismo de. **Museu de História Natural Charles Darwin**. 2019. Disponível em: https://turismo.bombinhas.sc.gov.br/o-que-fazer/item/museu-de-historia-natural-charles-darwin. Acesso em: 24 nov. 2022.

CARDOSO, Raul G. S. *et al*. **Uso Da Realidade Aumentada Em Auxílio À Educação**. Computer On The Beach 2014, São Luís, p. 330-339, 2014. Disponível em: https://periodicos.univali.br/index.php/acotb/article/view/5337. Acesso em: 21 nov. 2022.

CONCEITO. **Conceito de Museu**. 2020. Disponível em: https://conceito.de/museu. Acesso em: 23 nov. 2022.

COSTA, Maria João Pascoal Rodrigues Gomes da *et al*. **A Realidade Virtual e a Realidade Aumentada na Exposição de Obras de Arte:** A Pandemia de COVID-19. 2020. 98 f. Dissertação (Mestrado) - Curso de Mercados da Arte, Iscte-Instituto Universitário de Lisboa, Lisboa, 2020.

ICOM, O Conselho Internacional de Museus. **ICOM aprova Nova Definição de Museu**. 2022. Disponível em: https://www.icom.org.br/. Acesso em: 22 nov. 2022.

SILVA, Sâmia Siqueira Neves da. **Realidade virtual em museus:** Estudo de caso do NewsMuseum em Sintra. 2018. 96 f. Dissertação (Mestrado) - Curso de Mestre em Empreendedorismo e Estudos da Cultura, Especialização em Entretenimento e Indústrias Criativas, Departamento de História, Instituto Universitario de Lisboa, Lisboa, 2018.

TORI, Romero *et al* (ed.). **Fundamentos e Tecnologia de Realidade Virtual e Aumentada**. Belém: Sbc, 2006. 412 p.

ULTRALEAP. **Cat Explorer**. 2021. Disponível em: https://gallery.leapmotion.com/cat-explorer/. Acesso em: 10 ago. 2022.

ULTRALEAP. **Leap Motion Controller**. 2020. Disponível em: https://www.ultraleap.com/product/leap-motion-controller/. Acesso em: 20 ago. 2022.

VALENTINI, Pier Paolo. **Natural interface for interactive virtual assembly in augmented reality using Leap Motion Controller**. International Journal On Interactive Design And Manufacturing (Ijidem), [S.L.], v. 12, n. 4, p. 1157-1165, 5 mar. 2018. Springer Science and Business Media LLC. http://dx.doi.org/10.1007/s12008-018-0461-0. Disponível em: https://link.springer.com/article/10.1007/s12008-018-0461-0. Acesso em: 21 nov. 2022.