

Defesa do TCC

Link da defesa: https://teams.microsoft.com/l/meetup-join/19%3ameeting_MmM5YjM3MGYtNzQ0Zi00ZjYxLWEzNDgtNDZkYmUzNDFmZDI4%40thread.v2/0?context=%7b%22Tid%22%3a%220c2d222a-ecda-4b70-960a-acef6ced3052%22%2c%22Oid%22%3a%22e2602793-81ee-4f40-ac4e-f7a7f9d1e175%22%7d

Inicio da defesa: 20:00

Fim da defesa: 20:24

A curto prazo tudo vai estar no SmartPhone ...

Explorar mais ferramentas de desenho “Flood Fill”.

->> Ter uma sequência de imagens mostrando o comportamento da UI.

Acho que se teria que melhorar a interface para indicar quais ações cada opção do menu faz. Assim ter mais igualdade nas comparações com a interface tradicional. Mas não pensar em somente “levar” a UI tradicional para esta “nova” proposta.

->> Não mostra no artigo e nos slides tela com estereoscopia.

->> mudar RV p/ RVi.

Sugestão de explorar a estereoscopia, Google CardBoard ... pois acredito que exploraria mais o conceito da Realidade Virtual Imersiva. Fiquei em dúvida se mudaria para RVi no seu texto. Pois com estereoscopia melhoria a noção de profundidade ... exemplo abrir um trinco de porta com um só olho aberto.

O LeapMotion permite aumentar a imersão que é peça fundamental de RVi.

Sugestão fazer uma imagem para representar a forma como conecta a aplicação com Leap e o Smartphone.

Porque a escolha das cenas de “praia” e “mini fazenda”? Eu acho interessante popular a cena, mas quem sabe escolher algum espaço próximo de uma sala de aula, uma parede de uma praça (com o comentário que está autorizado para pintar), um estúdio de pintura ... e ter os desenhos numa galeria de arte (algumas salas parecidas com uma galeria). A ideia é aumentar a imersão.

Ou ainda explorar cenário que tragam um contexto que possa trazer algum conteúdo didático e o usuário acrescentar “desenhos” para responder questões ..

UM APLICATIVO DE DESENHO EM REALIDADE VIRTUAL UTILIZANDO O LEAP MOTION

Gabriel Brogni Bento, Mauricio Capobianco Lopes – Orientador

Curso de Bacharel em Ciência da Computação

Departamento de Sistemas e Computação

Universidade Regional de Blumenau (FURB) – Blumenau, SC – Brasil

gbbento@furb.br, mclopes@furb.br

Resumo: Este artigo apresenta o desenvolvimento de uma aplicação para desenho livre em realidade virtual (RV) utilizando o sensor Leap Motion. Foi utilizado o motor gráfico Unity para o desenvolvimento da aplicação e as ferramentas Riftcat e Steam para executar em RV, espelhando para o smartphone o que está sendo apresentado no computador. Os resultados foram obtidos a partir de testes com usuários em dois momentos diferentes de desenvolvimento o primeiro teste teve como objetivo encontrar melhorias na aplicação e no segundo para validar a implementação. Conclui-se que os usuários tiveram uma boa experiência com o aplicativo, mas que o conjunto de recursos tecnológicos necessários para executá-lo pode limitar sua utilização.

Palavras-chave: Leap Motion. Realidade Virtual. Unity. RiftCat. Desenho.

1 INTRODUÇÃO

Segundo Moran (2013), as tecnologias estão cada vez mais presentes na educação, ampliando as possibilidades de recursos para os professores usarem em suas práticas. Moran (2013) ressalta que, com as tecnologias atuais, a escola pode transformar-se em um conjunto de espaços ricos de aprendizagens significativas, presenciais e digitais, que motivem os alunos a aprender ativamente, a pesquisar o tempo todo, a serem proativos, a saberem tomar iniciativas e a interagir. Ainda, em outra obra, Moran (2012) consolida a opinião de que essas tecnologias desafiam as instituições a modificar o ensino tradicional, no qual os professores são o centro, para uma aprendizagem participativa e integrada, com momentos presenciais e a distância. O mesmo autor destaca que as tecnologias móveis geram desafios, descentralizam os processos de construção do conhecimento, com aprendizagens diversas que ocorrem a qualquer hora e em qualquer lugar, podendo ser em grupos ou individualmente.

Para Patten *et al.* (2006), os recursos possibilitados pelos dispositivos móveis não visam somente reproduzir ou ampliar os atuais cenários de aprendizagem, mas gerar novas oportunidades que não seriam possíveis sem a tecnologia móvel: celulares, notebooks, tablets, dentre outras. Com isso, as inovações desenvolvidas em escolas que têm tecnologias móveis na sala de aula são mais promissoras uma vez que podem ser utilizadas por professores e alunos, sinalizando mudanças na forma de ensinar e aprender. Uma das possibilidades de uso das tecnologias móveis na Educação é para a realização de desenhos. Existem diversas aplicações para isso, desde aplicativos simples para colorir até colaborativos na web como o Desenhos do Google. Em comum entre esses aplicativos está a possibilidade de inserir e colorir formas geométricas ou realizar desenhos livres possibilitando aos seus usuários utilizar todo seu potencial imaginativo.

O desenho é uma representação de algo visto ou imaginado. “A percepção do espaço é dada por um conjunto de linhas e de manchas que irão sugerir formas reais ou abstratas. Nos desenhos realizados com programas de computador, a sua especificidade leva o observador a assumir parecenças quase reais”. (MOREIRA PINTO, 2012, p. 23). Nesse sentido, o presente trabalho propõe a criação de um aplicativo para desenho utilizando o sensor Leap Motion acoplado a um smartphone para a criação de desenhos virtuais. O Leap Motion é um sensor de rastreamento óptico que captura os movimentos das mãos com precisão tornando natural a interação com o conteúdo digital (ULTRALEAP, 2021). O uso do sensor permite interagir com ambientes em realidade virtual em movimentos executados livremente no ar, sem a necessidade de dispositivos como mouse ou teclado. O acoplamento do sensor com o smartphone requer softwares específicos para comunicação entre os aparelhos. Para o presente trabalho foi escolhido o VRidge. “O VRidge é uma aplicação que transforma o smartphone em um óculos de realidade virtual barato como o Google Cardboard” (RIFTCAT, 2020).

Com base nisso, este trabalho tem por objetivo disponibilizar um aplicativo em realidade virtual que possibilite a criação de desenhos virtuais utilizando o Leap Motion. Como objetivos específicos destacam-se: analisar potenciais e dificuldades no uso do Leap Motion e VRidge; identificar e comparar o aplicativo com os trabalhos correlatos; avaliar a experiência dos usuários no uso do aplicativo.

A seguir são detalhados os aspectos que fundamentam o trabalho, bem como os correlatos pesquisados. Em seguida é relatado o desenvolvimento do aplicativo, com destaque para a especificação e implementação, seguido da análise dos resultados. Por fim, são apresentadas as conclusões e sugestões para trabalhos futuros.

2 FUNDAMENTAÇÃO TEÓRICA

Nesta seção são fundamentadas bibliografias sobre Realidade Virtual, o sensor Leap Motion e sobre Desenho. Em seguida são apresentados os trabalhos correlatos.

2.1 REALIDADE VIRTUAL

A Realidade Virtual (RV) é uma interface avançada do usuário para acessar aplicações executadas no computador, propiciando a visualização, movimentação e interação do usuário, em tempo real, em ambientes tridimensionais gerados por computador. O sentido da visão costuma ser preponderante em aplicações de realidade virtual, mas os outros sentidos, como tato, audição, entre outros, também podem ser usados para enriquecer a experiência do usuário (KIRNER; SISCOUTTO, 2007). O termo RV é bastante abrangente e acadêmicos, desenvolvedores de software e pesquisadores tendem a defini-lo com base em suas próprias experiências. Espinheira Neto (2004 apud SILVA *et al.*, 2017, p. 9) define RV como “uma experiência imersiva e interativa baseada em imagens gráficas 3D, geradas em tempo real por computador”.

Segundo Kirner e Siscoutto (2007), os ambientes virtuais mais simples são aqueles nos quais o usuário apenas faz a navegação, sem executar nenhum tipo de interação. Já nos ambientes mais complexos há algum tipo de alteração no espaço virtual que podem ser realizadas por meio de diferentes tipos de dispositivos não convencionais como controles, capacetes de RV, luvas e até o próprio corpo usando gestos ou comando de voz. No caso específico do presente trabalho, utiliza-se o Leap Motion, que será detalhado na seção seguinte.

Além disso, a RV também pode ser classificada como imersiva ou não imersiva (TORI; KIRNER; SISCOUTTO, 2006). A RV imersiva ocorre quando o usuário está em um mundo virtual utilizando sensores que permitem captar seus movimentos e reações, porém ele está com a sensação de estar no mundo real. Já a RV não-imersiva é quando o usuário é levado parcialmente para o mundo virtual e consegue perceber que está no mundo real. Conforme Baierle e Gluz (2017), a RV imersiva tem um potencial alto, mas depende de fatores como facilidade de uso e custo acessível para tornar-se mais popular. No presente trabalho, uma vez que é utilizado o dispositivo de Head Mounted Display, o usuário terá uma experiência imersiva.

2.2 LEAP MOTION

Conforme citado na seção anterior, a interação do usuário com a aplicação no presente trabalho é realizada pelo Leap Motion. O Leap Motion é um dispositivo que funciona como um sensor que visa traduzir os movimentos das mãos em comandos do computador. O controlador em si é uma unidade de oito por três centímetros que se conecta à porta USB do computador. Sua área de captação permite o reconhecimento das mãos em uma angulação de até 120 graus, sendo possível captar apenas os movimentos das mãos e não o corpo como um todo (ULTRALEAP, 2021). Para realizar a captação dos movimentos, o aparelho utiliza-se de sensor infravermelho junto com um conjunto de câmeras, os quais são utilizados como entrada de dados para o algoritmo de rastreamento que estima a posição das mãos e dedos (WOZNIAK *et al.*, 2016). A Figura 5 apresenta uma imagem do aparelho Leap Motion.

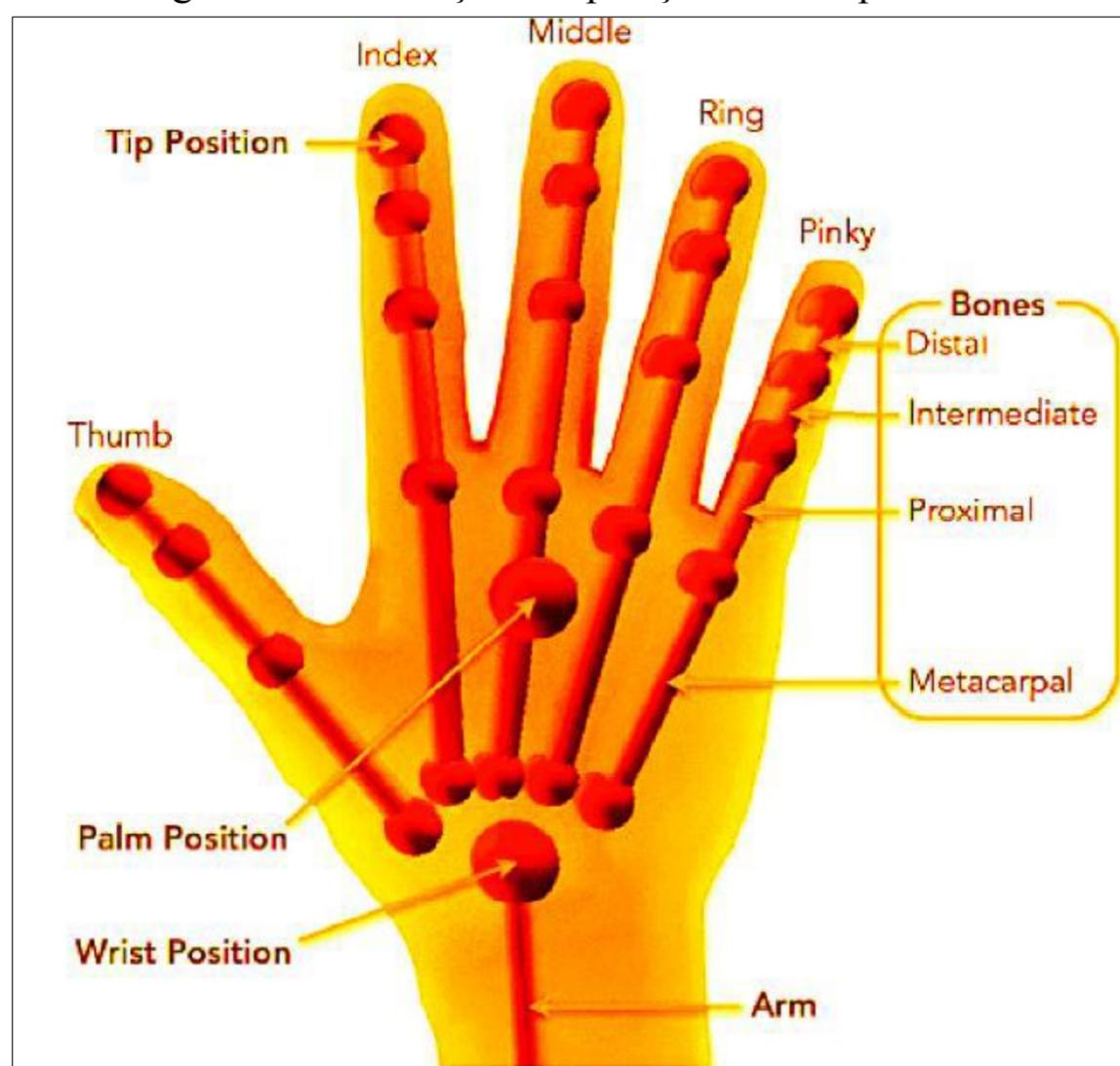
Figura 1 - Leap Motion Controller



Fonte: Ultraleap (2021).

Segundo Vaitkevičius *et al.* (2019), com o aparelho Leap Motion é possível obter o movimento individual da palma, do pulso, dos cinco dedos e de 11 juntas das mãos, conseguindo saber a distância e a posição de cada um deles (Figura 6). Entretanto, o aparelho Leap Motion não realiza o reconhecimento de gestos, tais como apontar o dedo indicador para determinada posição. Neste caso, é possível realizar validações por algoritmos para captar três Graus de Liberdade (Degrees of Freedom - DoF) das mãos, permitindo verificar qual gesto o usuário está realizando, tais como: o movimento de pinça entre dois dedos, apontar um dedo para determinada posição e o movimento dos punhos (VAITKEVIČIUS *et al.*, 2019).

Figura 2 - Informação das posições do Leap Motion



Fonte: Vaitkevičius *et al.* (2019).

Segundo Wozniak *et al.* (2016), o algoritmo de reconhecimento possui algumas falhas, sendo uma delas a quantidade de luminosidade no ambiente ao redor do Leap Motion: se tiver luz em excesso ou a falta dela o aparelho pode não reconhecer os movimentos, ocasionando falhas no rastreamento das mãos. Outro problema citado pelo mesmo autor é a grande variação no tamanho das mãos: se elas forem muito pequenas ou muito grandes os movimentos dos dedos podem não ser reconhecidos.

O Leap Motion disponibiliza um [SDK \(Software Developer Kit\)](#) que facilita o desenvolvimento de novas aplicações que o utilizam. Com esse kit, um desenvolvedor não precisa trabalhar com os dados vindo diretamente do aparelho, mas com as informações previamente processadas. (LEAPMOTION, 2020). Além de facilitar o processamento dos dados do aparelho, o SDK possui um conjunto de modelos 3D de mãos humanas que o desenvolvedor pode utilizar em suas aplicações.

2.3 DESENHO

No presente trabalho, o Leap Motion é utilizado para desenhar. “O desenho é provavelmente a forma de expressão que sintetiza melhor a nossa relação com o mundo. Ele permite-nos, com a elaboração mental, o desenvolvimento de ideias e a descoberta do que ainda desconhecemos de nós mesmos”. (CARNEIRO, 2001, p.34). A Base Nacional Comum Curricular (BNCC) expressa a importância do desenho desde a Educação Infantil, citada especificamente no campo de experiência que trata dos traços, sons, cores e formas, tendo como uma das aprendizagens esperadas que a criança consiga “Expressar-se por meio das artes visuais, utilizando diferentes materiais”. (BRASIL, 2018, p. 54). Também é citada na Educação Fundamental, no componente de Arte, mais especificamente na unidade temática de Artes Visuais.

“As Artes Visuais são um meio de explorar, aprimorar e qualificar a observação, percepção e exploração do mundo” (FERREIRA, 2020, p.15). A arte é uma junção de linguagens e materiais que possibilita revelar algo, sem saber o que exatamente seja, permitindo expressar ações e sentimentos que de outra forma não seria possível. (FERREIRA, 2020). Segundo a mesma autora, as Artes Visuais vinculadas com o desenho infantil permitem entrar no universo da criança facilitando a compreensão de seu mundo e permitindo ajudá-la a se encaixar no nosso.

Ainda segundo Ferreira (2020), as Artes Visuais têm como objetivo estimular diferentes linguagens por meio de atividades que provocam a capacidade de descobrir, conhecer e aprender das crianças. Elas precisam de estímulos que as levam a desenvolver suas coordenações motoras e a arte atua nesse processo como principal facilitadora (FERREIRA, 2020). A autora ressalta ainda que é importante as crianças entrarem no mundo da arte o mais cedo possível pois isso ajuda no desenvolvimento de sua aprendizagem. Assim, o presente trabalho apresenta uma proposta de experimentar uma nova forma de relação com o desenho, utilizando as tecnologias.

2.4 TRABALHOS CORRELATOS

A seguir são apresentados três trabalhos que possuem características semelhantes aos principais objetivos do proposto e que utilizam algumas das técnicas apresentadas anteriormente. O Quadro 1 apresenta o trabalho de Falcao,

Lemos e Soares (2015) que demonstra um teste de usabilidade utilizando o Leap Motion e o Photoshop. O Quadro 2 apresenta uma aplicação que permite desenhar utilizando os movimentos da mão em um ambiente de duas dimensões, desenvolvido por Lyu *et al.* (2017). O Quadro 3 apresenta o trabalho de integração do óculos DK2 e o Leap Motion em um aplicativo de desenho em realidade virtual realizado por Gerry (2017).

Quadro 1 – Trabalho Correlato 1

Referência	Falcao, Lemos e Soares (2015)
Objetivos	Realizar testes de usabilidade para medir a experiência do usuário utilizando o Leap Motion para interagir com a interface do Photoshop.
Principais funcionalidades	Interagir com os recursos de menu do Photoshop utilizando o Leap Motion.
Ferramentas de desenvolvimento	Não houve desenvolvimento de uma aplicação. Utiliza o hardware Leap Motion.
Resultados e conclusões	Os usuários não consideraram os resultados agradáveis nos testes e afirmaram que a usabilidade do Leap Motion junto com a ferramenta Photoshop não é natural pois, para executarem uma ação simples, foi preciso muito mais tempo do que o necessário em relação às atividades realizadas com o uso do mouse e teclado.

Fonte: elaborado pelo autor.

Quadro 2 – Trabalho Correlato 2

Referência	Lyu et al. (2017)
Objetivos	Desenvolver uma ferramenta que utiliza o Leap Motion para desenhar em uma tela de duas dimensões (2D).
Principais funcionalidades	Permite desenhar em uma tela virtual ao realizar o gesto de apontar com o indicador e movimentar a mão, enquanto afasta e aproxima o dedo para criar formas e traços distintos.
Ferramentas de desenvolvimento	Não especifica a linguagem de desenvolvimento. Utiliza o hardware Leap Motion.
Resultados e conclusões	Essa aplicação possibilita imersão mesmo em uma tela 2D, permitindo criar desenhos com diversas formas e tamanhos variados. Entretanto segundo relatos de usuários a aplicação apresentou alguns problemas no rastreamento das mãos criando desenhos incoerentes com seus movimentos, além disso, para que o usuário consiga parar de desenhar, é preciso retirar a mão da área de detecção do Leap Motion, assim retirando um pouco da imersão do usuário.

Fonte: elaborado pelo autor.

Quadro 3 – Trabalho Correlato 3

Referência	Gerry (2017)
Objetivos	Desenvolver uma aplicação que permita desenhar em realidade virtual utilizando o óculos SK2 e o Leap Motion.
Principais funcionalidades	Permite ao usuário desenhar em um ambiente de realidade virtual enquanto um pintor no mundo real realiza movimentos simulando o seu trabalho.
Ferramentas de desenvolvimento	Desenvolvido em C# utilizando o motor gráfico Unity. Utiliza os hardwares Leap Motion e o óculos SK2.
Resultados e conclusões	Essa aplicação diminui a curva de aprendizado entre um novato e um profissional quase que para zero instantaneamente, tornando o aprendizado de pinturas tão simples quanto seguir os movimentos de outra pessoa.

Fonte: elaborado pelo autor.

3 DESCRIÇÃO DO APlicATIVO

Nesta seção é apresentado o desenvolvimento do aplicativo. A seção 3.1 apresenta a especificação, com destaque para as funcionalidades e diagramas de estrutura e comportamento. A seção 3.2 descreve a implementação do aplicativo apresentando a interface gráfica a partir da qual o usuário interage com o aplicativo, além dos principais trechos de código.

3.1 ESPECIFICAÇÃO

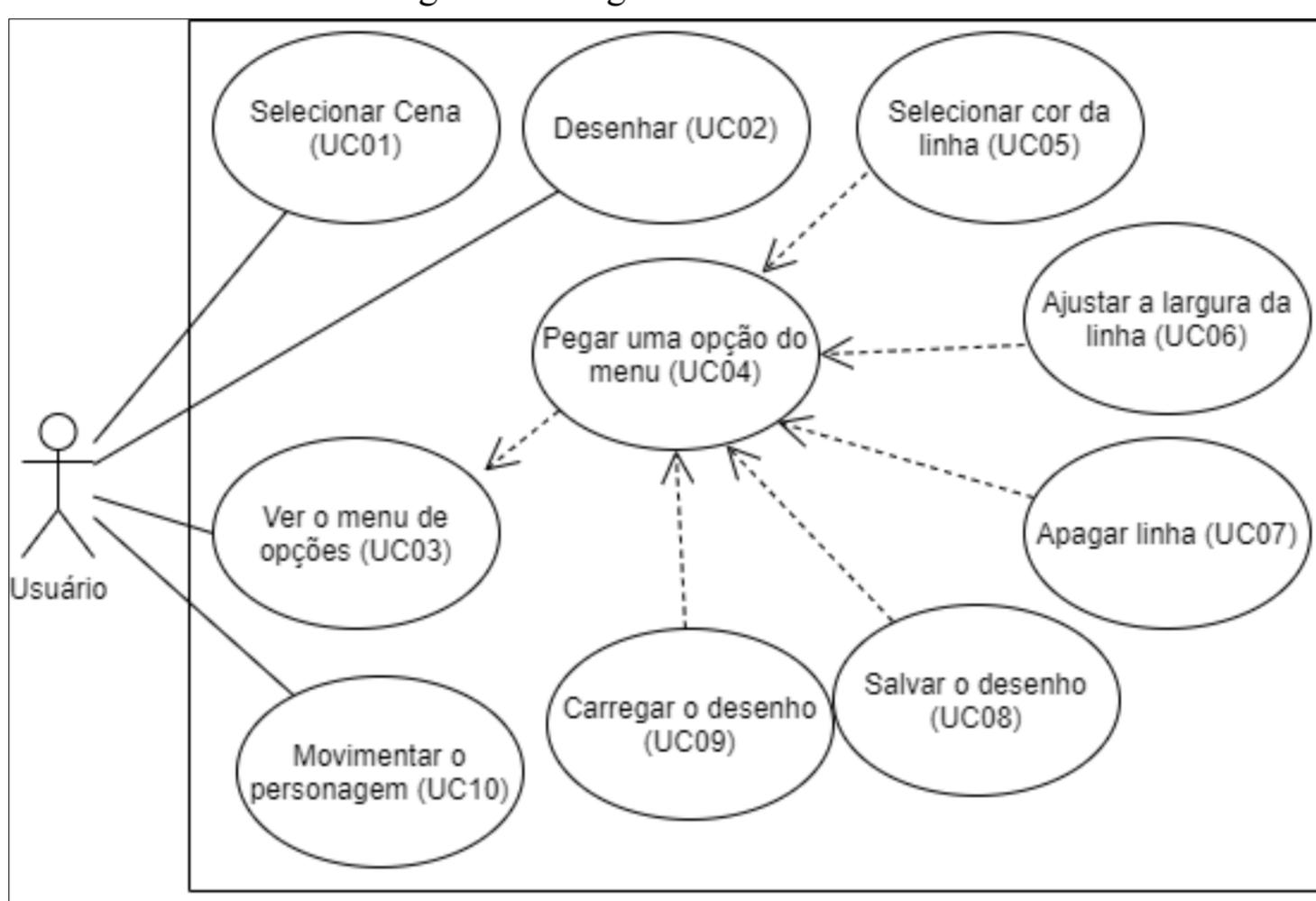
O aplicativo proposto tem por objetivo permitir ao usuário fazer desenhos e se movimentar em um cenário utilizando o LeapMotion e um óculos de realidade virtual. Assim, os Requisitos Funcionais (RF) e Requisitos Não Funcionais (RNF) definidos são:

- a) permitir ao usuário selecionar cena em 3D (RF);
- b) permitir ao usuário desenhar em um ambiente de RV com o Leap Motion (RF);
- c) permitir ao usuário ver o menu (RF);
- d) permitir ao usuário pegar uma opção do menu (RF);
- e) permitir ao usuário selecionar a cor do desenho (RF);

- f) permitir ao usuário escolher a largura da linha (RF);
- g) permitir ao usuário apagar partes do desenho (RF);
- h) permitir ao usuário salvar o desenho (RF);
- i) permitir ao usuário abrir um desenho salvo (RF);
- j) permitir ao usuário se movimentar na cena (RF);
- k) o aplicativo deve ser compatível com um aparelho smartphone com Android 8 ou superior (RNF);
- l) o aplicativo deve ser compatível com o aplicativo VRidge (RNF);
- m) o aplicativo deve utilizar o aparelho Leap Motion para detecção das mãos (RNF);
- n) o aplicativo deve ser executado no *framework Steam*.
[REDACTED]
- o) o aplicativo deve funcionar para várias envergaduras de mãos, desde crianças até adultos (RNF);
- p) o aplicativo deve ser especificado no Draw.io (RNF);
- q) o aplicativo deve ser desenvolvido em C# e Unity (RNF).

A Figura 3 apresenta o diagrama de casos de uso da aplicação com suas funcionalidades. Uma vez que foi criado um caso de uso para cada requisito funcional não foi necessário apresentar a matriz de rastreabilidade.

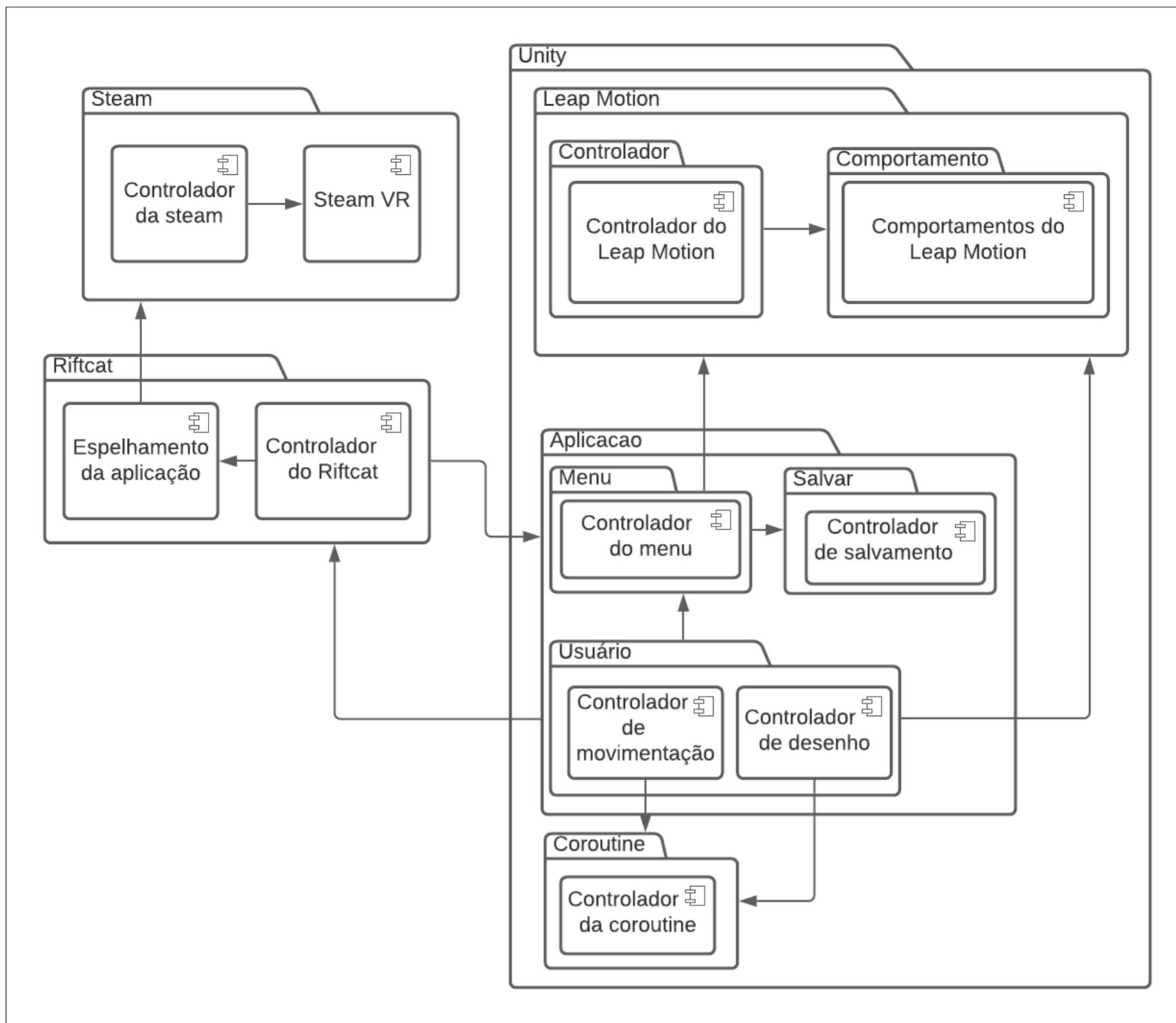
Figura 3 - Diagrama de casos de uso



Fonte: elaborado pelo autor.

A Figura 4 apresenta o diagrama de pacotes, com seus respectivos componentes, o qual é dividido em quatro módulos. O pacote *Aplicacao* gerencia os comportamentos realizados pelo usuário, como desenhar e se movimentar. É com ele que se faz a comunicação com o aparelho Leap Motion. O pacote *Leap Motion* providencia os comportamentos padrões do aparelho que são utilizados nas ações dos usuários. O *Riftcat* controla o espelhamento da aplicação para o aparelho smartphone e a troca de informações dos sensores. O *Steam* é o *framework* de RV onde a aplicação é executada. O *Unity* é a ferramenta de desenvolvimento da aplicação. Esses pacotes constituem um conjunto de recursos demandados pela aplicação, bem como os que foram customizados ou desenvolvidos.

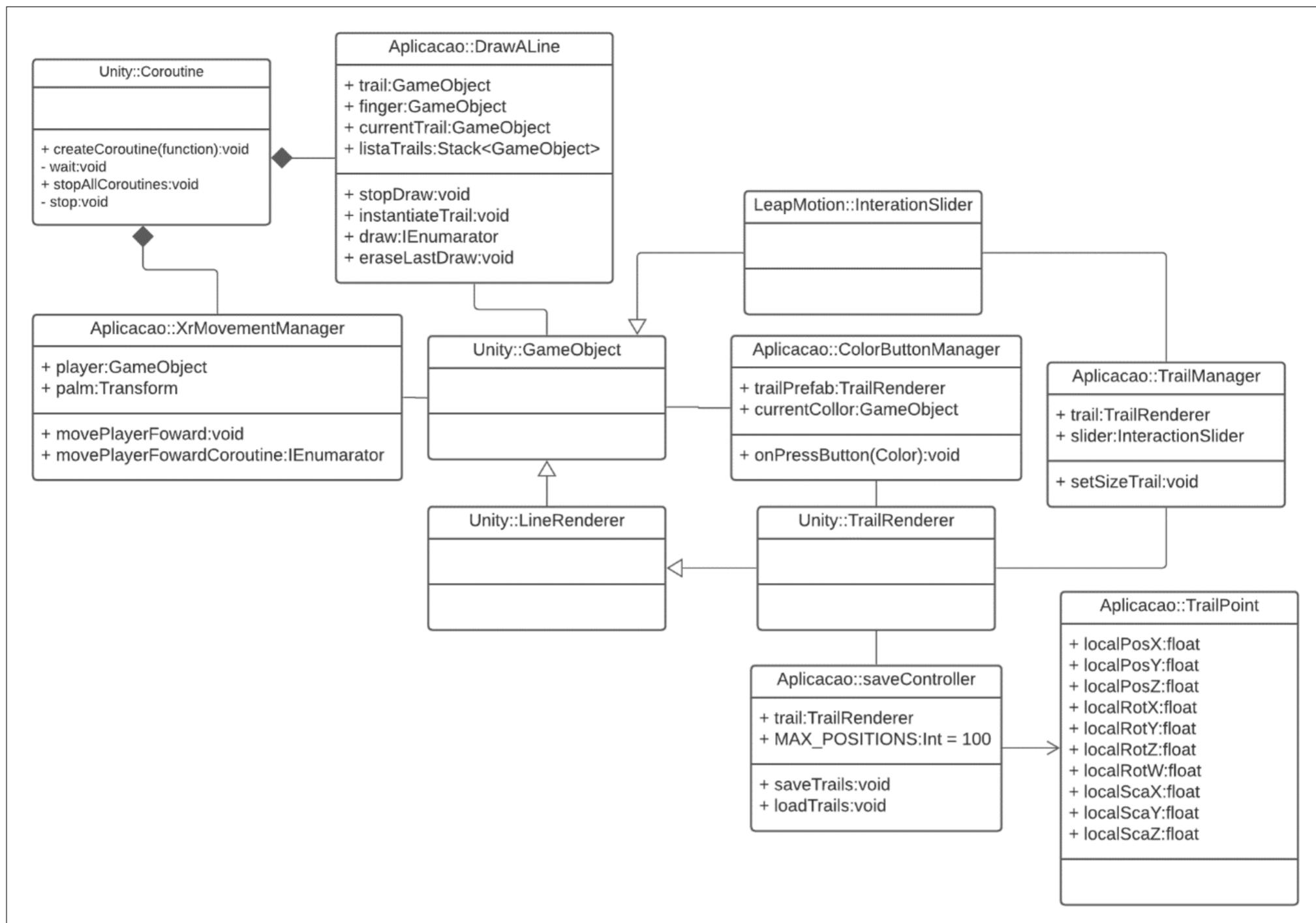
Figura 4 - Diagrama de pacotes



Fonte: elaborado pelo autor.

A Figura 5 apresenta o diagrama de classes e demonstra as principais estruturas presentes na aplicação. A classe `DrawALine` e `XrMovementManager` utilizam os métodos da `coroutine` para controle de execução nos laços de repetição. A Classe `GameObject` da Unity é utilizada para acessar os objetos presentes na cena sendo o `TrailRenderer` uma especialização de objeto com comportamentos específicos. A classe `TrailManager` utiliza a `InteractionSlider`, uma classe do `LeapMotion` para criação de menus com controles deslizantes. A classe `saveController` é responsável por processar os algoritmos para salvar e carregar o desenho, sendo utilizada a classe `TrailPoint` como um objeto de pontos. A classe `ColorButtonManager` gerencia o menu de cores da aplicação e realiza a troca dos atributos do objeto `TrailRenderer`.

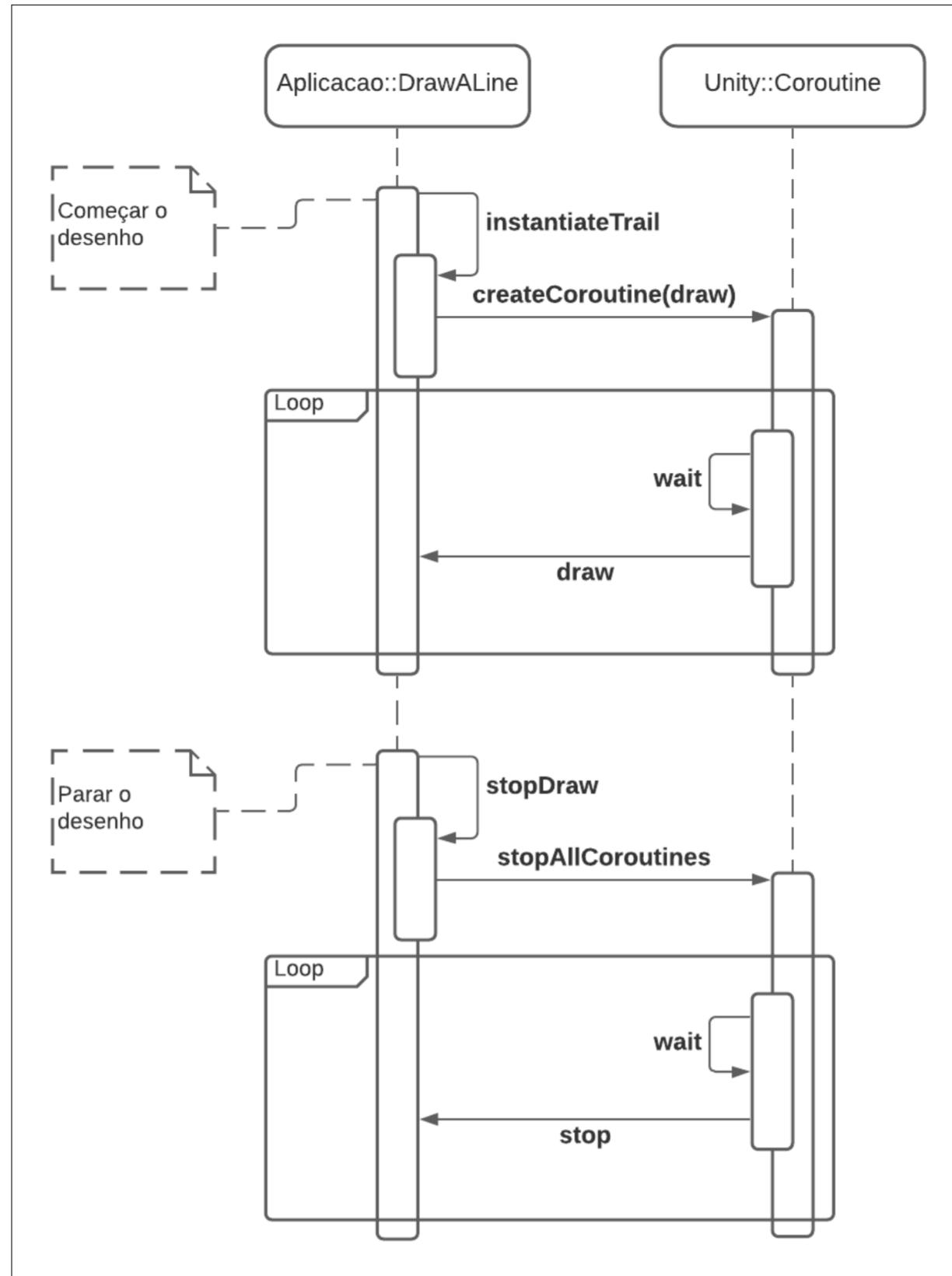
Figura 5 - Diagrama de classes



Fonte: elaborado pelo autor.

O diagrama de sequência relativo ao caso de uso desenhar (UC02) é apresentado na Figura 6. O diagrama representa duas situações do desenho: uma quando o usuário está desenhando e outra quando para de desenhar. No primeiro caso é feita a validação se o comportamento para começar a desenhar foi realizado pelo usuário. Assim, quando o usuário junta o polegar com o indicador é requisitada a chamada do método `instantiateTrail`, responsável por iniciar o desenho. Nesse método é criado uma *coroutine* responsável por instanciar o objeto na cena, sendo executada até que se pare de desenhar. Esta rotina inicia com a validação se o usuário desfez o movimento de pinça sendo, então, chamado o método `StopAllCoroutines` que vai parar a execução da *coroutine*.

Figura 6 - Diagrama de sequência da ação de desenhar



Fonte: elaborado pelo autor.

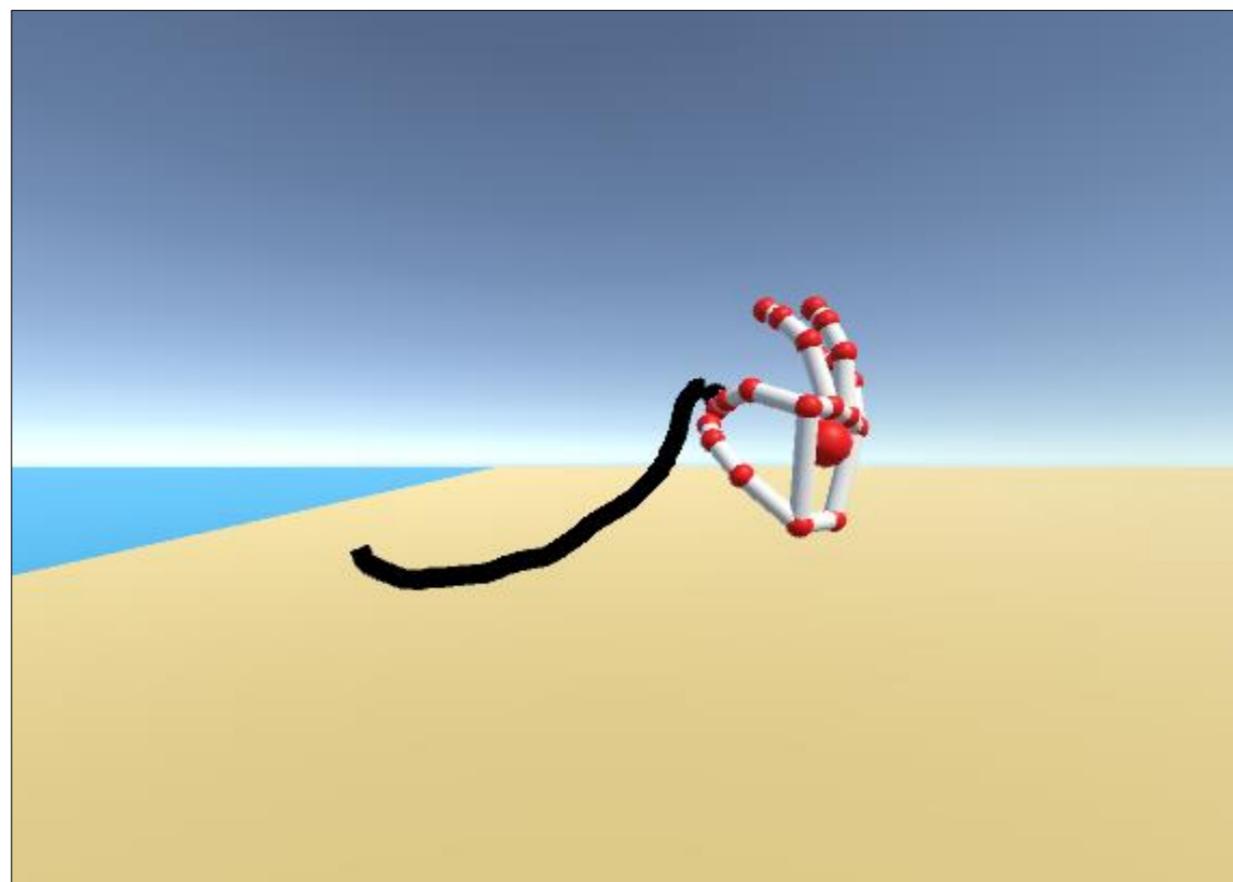
O caso de uso movimentar o personagem (UC10) possui uma sequência de ações parecido com o diagrama apresentando na Figura 6. A diferença é o comportamento que realiza a chamada dessa ação, pois para movimentar o personagem é preciso validar mais do que apenas um movimento do usuário. Outra diferença é no método da *routine* que, ao invés de instanciar um objeto na cena, realiza a movimentação do personagem para frente ou para trás dependendo do gesto do usuário.

3.2 IMPLEMENTAÇÃO

Para o desenvolvimento da aplicação foi utilizado o motor gráfico Unity, sendo que a estruturação geral e os algoritmos foram desenvolvidos na linguagem de programação C#, enquanto a comunicação entre o computador e o dispositivo móvel foi realizada por meio da aplicação RiftCat e de um *plugin* para o *framework* Steam VR.

Uma das primeiras ações que o usuário pode realizar ao iniciar a aplicação é a de desenhar, sendo que para executar esse comportamento o usuário deve fazer o movimento de pinça entre o dedo indicador e o polegar, como demonstrado na Figura 7. Ao mover a mão com o gesto de pinça um traço é iniciado, permitindo criar um desenho em 3D. Para o usuário terminar de desenhar, é preciso afastar os dois dedos, desfazendo o gesto de pinça, permitindo movimentar a mão livremente ou realizar outra ação.

Figura 7 - Ação de desenhar



Fonte: capturado pelo autor.

Para a implementação dos algoritmos que necessitam de ações do usuário, como a ação de desenhar, foi utilizado como base os algoritmos de comportamento do Leap Motion. Esses algoritmos consistem em realizar validações nas ações do usuário para definir se determinado ato executado se encaixa no comportamento predefinido. Para essa ação em questão é utilizado o `PinchDetector`, que valida se o usuário está realizando o movimento de pinça entre o dedo indicador e o polegar. Se esse algoritmo retorna positivo, então é chamado o algoritmo para realização desse comportamento. No Quadro 4 é apresentado o método que retorna a distância entre os dois dedos.

Quadro 4 - Algoritmo de validação da distância

```
private float GetPinchDistance(Hand hand) {
    var indexTipPosition = hand.GetIndex().TipPosition.ToVector3();
    var thumbTipPosition = hand.GetThumb().TipPosition.ToVector3();
    return Vector3.Distance(indexTipPosition, thumbTipPosition);
}
```

Fonte: adaptado de Leap Motion (2021).

O retorno do método `GetPinchDistance` é utilizado no algoritmo de validação, que pode ser visto no Quadro 5. Esse algoritmo valida se a distância entre dedos é menor que a configurada. Se positivo, realiza a chamada do método para desenhar em `ChangeState`, passando como parâmetro um valor booleano verdadeiro para realizar o desenho. Se a validação da distância retornar falso então o desenho é finalizado.

Quadro 5 - Algoritmo de controle do comportamento

```
_distance = GetPinchDistance(hand);
_rotation = hand.Basis.CalculateRotation();
_position = ((hand.Fingers[0].TipPosition + hand.Fingers[1].TipPosition) * .5f).ToVector3();

if (IsActive) {
    if (_distance > DeactivateDistance) {
        changeState(false);
        //return;
    }
} else {
    if (_distance < ActivateDistance) {
        changeState(true);
    }
}
```

Fonte: adaptado de Leap Motion (2021).

O retorno positivo na chamada anterior inicializa o método `instantiateTrail` que instancia um objeto na tela na posição da mão do usuário e cria uma *coroutine* chamada `Draw` (Quadro 6). A utilização de *coroutines* nesse trabalho tem como objetivo reduzir a utilização de laços de repetição pois, como a Unity utiliza a execução de códigos em paralelo, um *loop* se torna algo muito complexo de se executar. Com isso, ao se utilizar uma *coroutine* é possível parar a execução e voltar conforme demanda da aplicação. Essa técnica se assemelha ao conceito de uma *thread*, porém com a diferença de que diferentes *coroutines* não executam em paralelo.

Quadro 6 - Algoritmo desenhar

```

public void instantiateTrail(){
    CurrentTrail = Instantiate(trail, Finger.transform.position, Quaternion.identity);
    listaTrails.Push(CurrentTrail);
    StartCoroutine(Draw());
}

IEnumerator Draw(){
    while (true){
        Vector3 Position = new Vector3(Finger.transform.position.x,
                                         Finger.transform.position.y,
                                         Finger.transform.position.z);
        CurrentTrail.transform.position = Position;
        yield return null;
    }
}

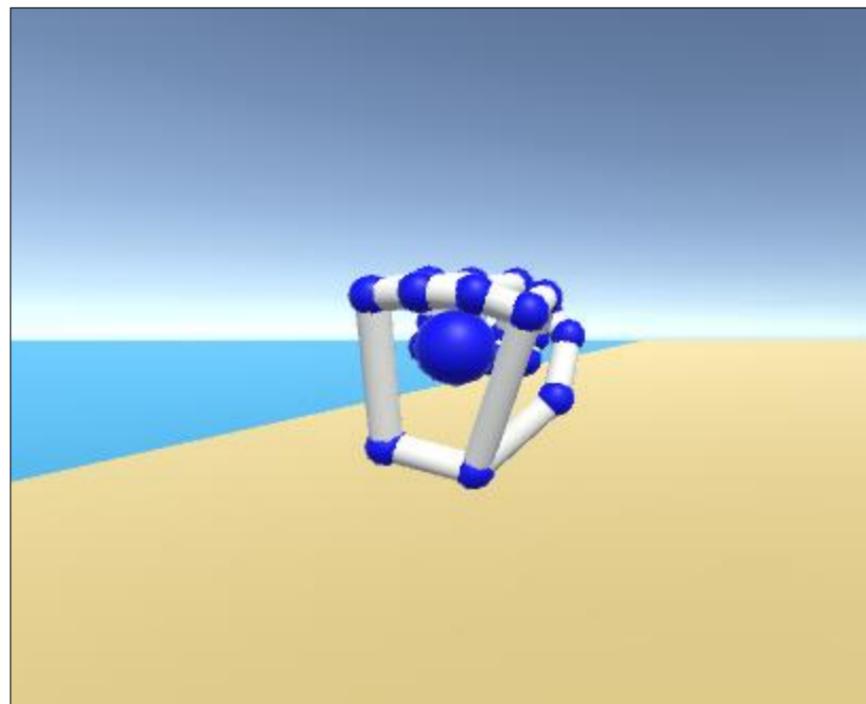
```

Fonte: elaborado pelo autor.

O método `Draw` realiza a criação de pontos de uma reta conforme o usuário movimenta a mão pela cena e realiza a ligação dessas retas por meio de um objeto renderizado nessas posições predefinidas. Esse método é executado enquanto a *coroutine* estiver ativa, ou seja, enquanto o usuário estiver realizando a posição de pinça com os dedos e movimentando a mão são gerados os pontos que representam o desenho. A junção desses pontos representa o objeto desenhado.

Outra possível ação do usuário é a de se movimentar pela cena. Para realizar esse comportamento é preciso apontar a palma da mão esquerda para frente e fechar o pulso, como mostra a Figura 8. Ao realizar esse gesto, o personagem se movimenta para frente seguindo a direção em que ele está olhando. Para se movimentar para trás, a palma da mão precisa estar apontada na direção do seu corpo.

Figura 8 - Ação Movimentar



Fonte: capturado pelo autor.

O algoritmo de movimentação se assemelha com o algoritmo de desenhar, porém a principal diferença é a utilização de portas lógicas na validação dos comportamentos. Para isso, são utilizados os algoritmos `PalmDirectionDetector` e o `ExtendedFingerDetector`, em conjunto com o `DetectorLogicGate`. O algoritmo `PalmDirectionDetector` tem como principal método o `palmWatcher` (Quadro 7), uma *coroutine* que valida se a angulação da mão está apontada para a direção configurada. Se estiver, então retorna positivo no método `Activate`. O algoritmo `ExtendedFingerDetector` valida se os todos os dedos da mão esquerda estão apontando para baixo e encostados na palma da mão. Seu código é parecido com o algoritmo anterior e também retorna positivo caso seu comportamento esteja valido. Já o `DetectorLogicGate` é responsável por chamar os métodos de movimentação e somente fará isso quando ambos os algoritmos anteriores tiverem validado o comportamento.

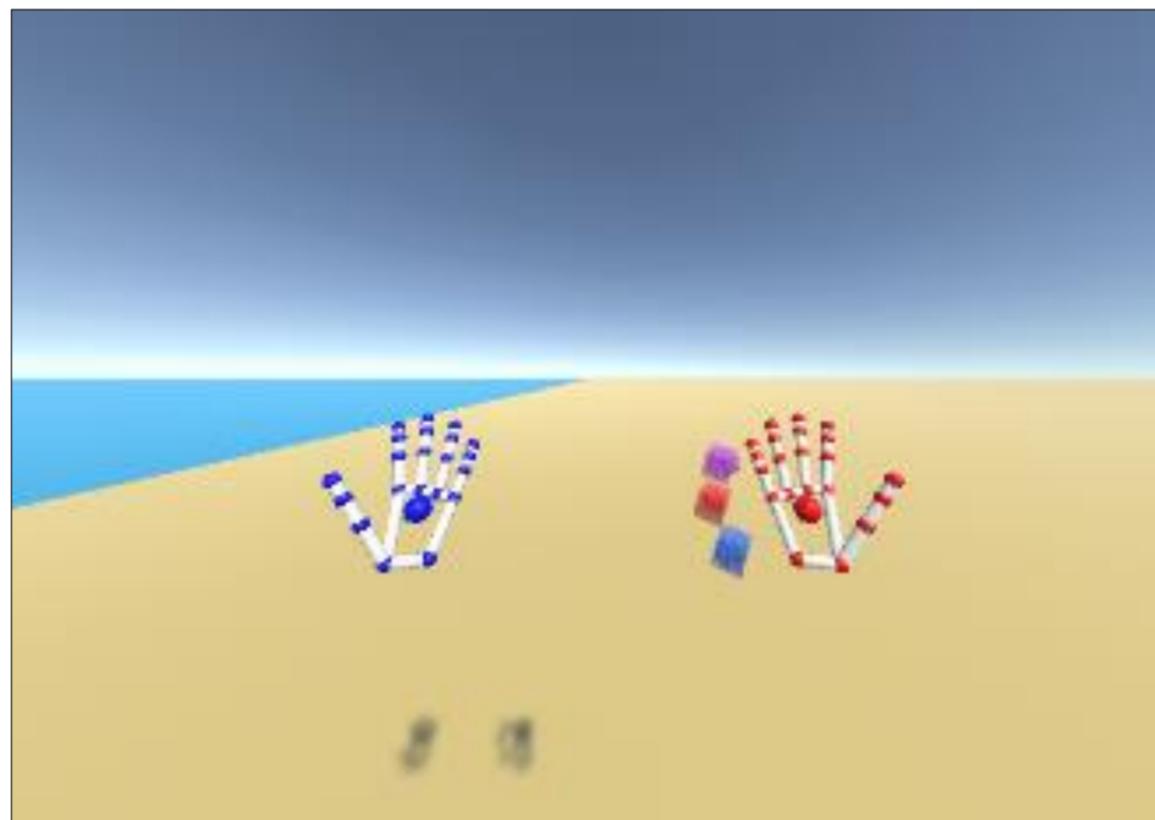
Quadro 7 – Algoritmo PalmDirectionDetector

```
private IEnumerator palmWatcher() {
    Hand hand;
    Vector3 normal;
    while(true){
        if(HandModel != null){
            hand = HandModel.GetLeapHand();
            if(hand != null){
                normal = hand.PalmNormal.ToVector3();
                float angleTo = Vector3.Angle(normal, selectedDirection(hand.PalmPosition.ToVector3()));
                if(angleTo <= OnAngle){
                    Activate();
                } else if(angleTo > OffAngle) {
                    Deactivate();
                }
            }
        }
        yield return new WaitForSeconds(Period);
    }
}
```

Fonte: adaptado de Leap Motion (2021).

Para que seja possível acessar as outras ações do aplicativo, tais como, escolher cores, mudar espessura da linha, apagar, mudar de cena ou salvar o desenho é preciso abrir o menu de opções. Para isso, o usuário deve apontar a palma da mão direita em direção ao seu corpo, como mostra a Figura 9. Assim que o usuário realizar essa ação, três blocos aparecem ao lado da mão. Para interagir com esses blocos, o usuário deve pegar um deles com a mão esquerda, arrastar para longe da mão direita e soltar. Um novo menu é criado dependendo da opção selecionada. O primeiro bloco possui as opções para o desenho, como mudar a cor, largura da linha e apagar o último traço. O segundo possui as opções para mudar o cenário e o terceiro para salvar o desenho.

Figura 9 - Menu de opções



Fonte: capturado pelo autor.

O primeiro bloco é o mais utilizado durante a aplicação, pois possui os botões básicos para realizar as alterações no desenho. Como pode ser visto na Figura 10, os botões coloridos representam a ação de mudar a cor da linha, o botão branco com um símbolo na parte esquerda da imagem representa a ação de apagar e o botão deslizante é responsável por determinar a largura da linha.

Figura 10 - Menu de opções do desenho



Fonte: capturado pelo autor.

Para apagar um desenho é realizada a chamada do método `eraseLastDraw`, que pode ser visto no Quadro 8. Esse método retira do vetor o último objeto adicionado e remove da tela a instância desse objeto. Já a mudança da cor ou da espessura implicam apenas na mudança de atributos do desenho.

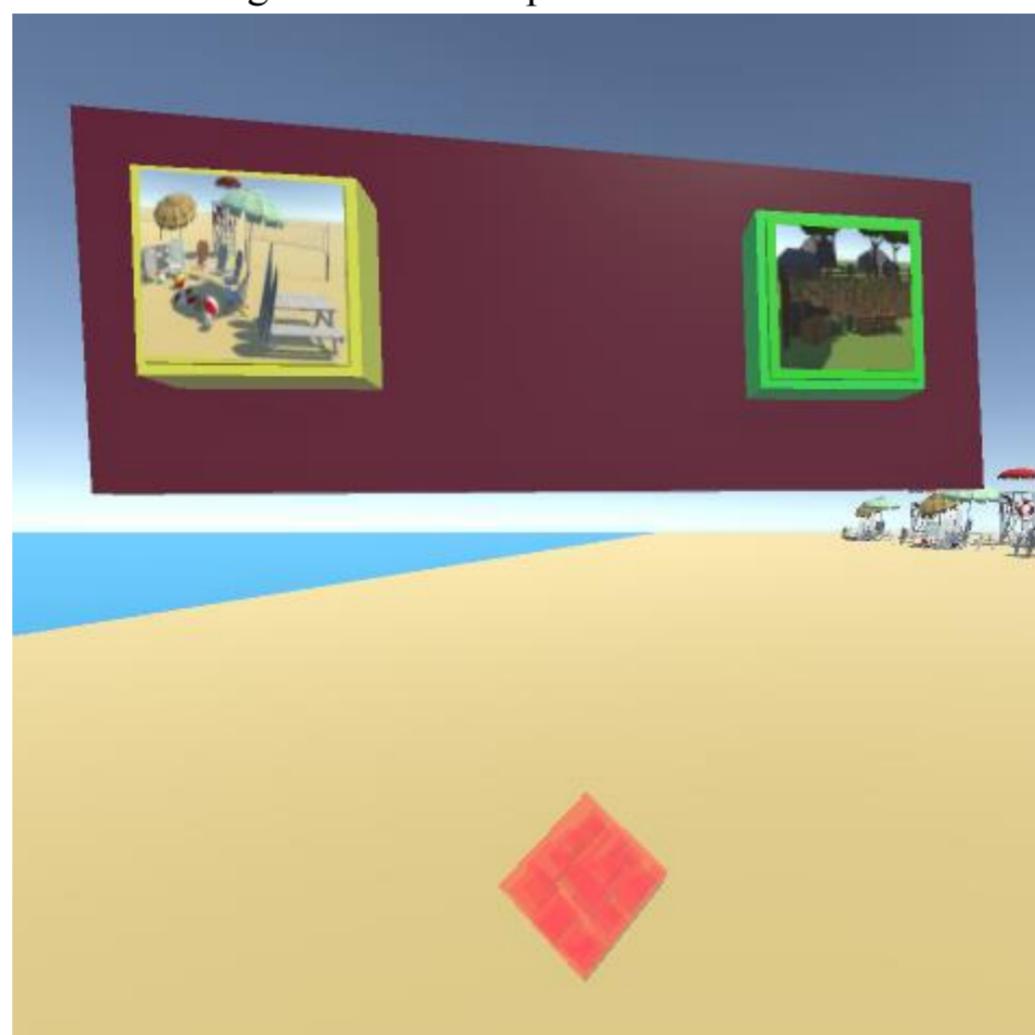
Quadro 8 - Método da ação de apagar

```
public void eraseLastDraw(){
    if(listaTrails.Peek() != null){
        GameObject erase = listaTrails.Pop();
        Destroy(erase);
    }
}
```

Fonte: elaborado pelo autor.

O segundo bloco é utilizado para controlar as cenas da aplicação, como pode ser visto na Figura 11. Existem dois botões, os quais ao serem pressionados faz a troca para a respectiva cena, sendo o primeiro botão a de uma praia e o segundo a de uma mini fazenda. Ao trocar a cena os desenhos já presentes nela continuam persistidos e não são apagados.

Figura 11 - Menu para trocar de cena



Fonte: capturado pelo autor.

O terceiro bloco é responsável por armazenar o comportamento de salvar o desenho. Esse menu possui apenas um botão que, ao ser pressionado, começa o processamento chamando o método `saveTrails` (Quadro 9). Esse método cria um arquivo no formato `JSON` contendo os pontos de todos os traços desenhados pelo usuário. Para isso, é criado um objeto chamado `TrailPoint` que armazena a estrutura desses pontos para que possa ser carregado novamente.

Quadro 9 - Algoritmo para salvar

```

public void saveTrails(){
    StreamWriter sw = File.CreateText(@"Path");
    GameObject[] trails = GameObject.FindGameObjectsWithTag("trail");
    foreach(GameObject trail in trails)
    {
        Vector3[] TrailRecorded = new Vector3[trail.GetComponent<TrailRenderer>().positionCount];
        int positions = trail.GetComponent<TrailRenderer>().GetPositions(TrailRecorded);

        TrailPoint myObject = new TrailPoint();
        myObject.pos = TrailRecorded;
        myObject.LocalPosX = trail.transform.position.x;
        myObject.LocalPosY = trail.transform.position.y;
        myObject.LocalPosZ = trail.transform.position.z;

        myObject.LocalRotX = trail.transform.rotation.x;
        myObject.LocalRotY = trail.transform.rotation.y;
        myObject.LocalRotZ = trail.transform.rotation.z;
        myObject.LocalRotW = trail.transform.rotation.w;

        myObject.LocalScaX = trail.transform.localScale.x;
        myObject.LocalScaY = trail.transform.localScale.y;
        myObject.LocalScaZ = trail.transform.localScale.z;

        sw.WriteLine(JsonUtilityToJson(myObject));
    }

    sw.Close();
}

```

Fonte: elaborado pelo autor.

Para carregar o desenho, o arquivo `Json` é lido e seus dados são apresentados na tela.

4 RESULTADOS

Nesta seção são apresentados os resultados obtidos com o desenvolvimento desse trabalho, sendo subdividida em três partes. Na primeira é comentado sobre o uso das tecnologias adotadas, na segunda a comparação entre os trabalhos correlatos previamente apresentados e a terceira os testes realizados para validação deste trabalho.

4.1 EXPERIÊNCIA COM AS TECNOLOGIAS

Para esse trabalho foi optado em dar um foco em três tecnologias em específico: o Leap Motion, a RV e o aparelho smartphone. De início a maior dificuldade foi que o Leap Motion não possui nenhum suporte para smartphone, não sendo possível a conexão direta entre os dois. Assim, optou-se por executar a aplicação no computador, espelhar a tela para o celular e então utilizar um óculos de RV para visualizar a cena no smartphone. Porém, como o usuário precisa movimentar a câmera durante a aplicação, é necessário captar os movimentos do sensor giroscópio do smartphone, sendo necessário para isso a utilização da aplicação RiftCat da empresa VRidge que permite recuperar essas informações sem estar executando o código no aparelho móvel. Essa solução permitiu as condições para que as funcionalidades da aplicação funcionassem da forma esperada.

Para que a comunicação entre a Unity e o RiftCat funcione é preciso instalar um *plugin* da empresa Valve que se chama Steam VR. Esse *plugin* permite executar a aplicação no ambiente de RV da empresa Valve. Além disso, nas configurações do projeto na Unity é necessário selecionar qual é o aparelho VR no qual a aplicação executa, nesse caso o Steam VR. Além disso, para que o RiftCat funcione é necessário instalar no celular e no computador seus respectivos softwares e, então, configurar o smartphone como ponto de ancoragem da rede. Assim, ao iniciar a aplicação na Unity, o ambiente Steam VR é iniciado e transmitido para o smartphone.

Ressalta-se, ainda, que o Leap Motion permite o desenvolvimento de aplicações não convencionais, possibilitando a interação direta com a cena sem o uso de um controle na mão do usuário, proporcionando uma experiência mais natural e fluida. Porém, para que o Leap Motion consiga captar os movimentos com uma boa precisão, existem alguns fatores que precisam ser atendidos. O primeiro deles é o fato que o dispositivo precisa ser calibrado constantemente, principalmente ao trocar o local em que é utilizado e esse processo, na maioria das vezes, precisa ser executado mais de uma vez. Outro fator importante é que o ambiente no qual ele será utilizado tenha uma boa iluminação de modo que o aparelho consiga captar corretamente os movimentos das mãos. Além disso, o Leap Motion tem dificuldade para captar movimentos específicos de usuários que possuem mãos muito pequenas. Por exemplo, para a ação de

desenhar, em algumas situações o Leap Motion não consegue identificar o gesto que o usuário está realizando. Para minimizar essa questão, é necessário ajustar os parâmetros de determinados métodos com valores menores que o habitual.

4.2 COMPARAÇÃO COM OS CORRELATOS

Os correlatos foram importantes fonte de informação para a construção do presente aplicativo. As funcionalidades de cada um deles possibilitou refletir sobre as características da aplicação proposta. A partir do Quadro 10 percebe-se que o trabalho realizado por Gerry (2017) possui foco mais preciso sobre o desenho que o usuário está realizando, sendo a principal fonte dessa afirmação o uso de Realidade Virtual (RV). Com o uso de RV, o usuário pode visualizar seu desenho em diversos ângulos. Além disso, o rastreamento das mãos é facilitado pois, uma vez que o sensor está preso ao óculos de RV, consegue captar melhor os movimentos devido ao ângulo de inclinação. No trabalho de Lyu *et al.* (2017), o sensor foi utilizado na mesma base que o computador e, por esse motivo, somente foi possível verificar a distância entre a tela e o dedo do usuário, impedindo um maior controle sobre o desenho. O trabalho realizado por Falcao, Lemos e Soares (2015) teve como objetivo testar a eficiência do Leap Motion em uma ferramenta já existente no mercado, entretanto os resultados demonstram que o sensor não foi muito eficaz ao realizar essa tarefa. Um desses motivos é em função do Photoshop não possuir compatibilidade nativa com o aparelho. Nos trabalhos realizados por Lyu *et al.* (2017) e Gerry (2017) esse problema não ocorre por se tratar de aplicações desenvolvidas especificamente para o sensor. Tal dificuldade também não foi observada na aplicação proposta.

Quadro 10 - Comparativo entre correlatos

Trabalhos Características	Falcao (2015)	Lyu <i>et al.</i> (2017)	Gerry (2017)	Proposto
Permite desenhar?	Sim	Sim	Sim	Sim
Utiliza um software próprio?	Não	Sim	Sim	Sim
Utiliza realidade virtual?	Não	Não	Sim	Sim
Possui controle detalhado sobre o desenho?	Não	Não	Sim	Sim

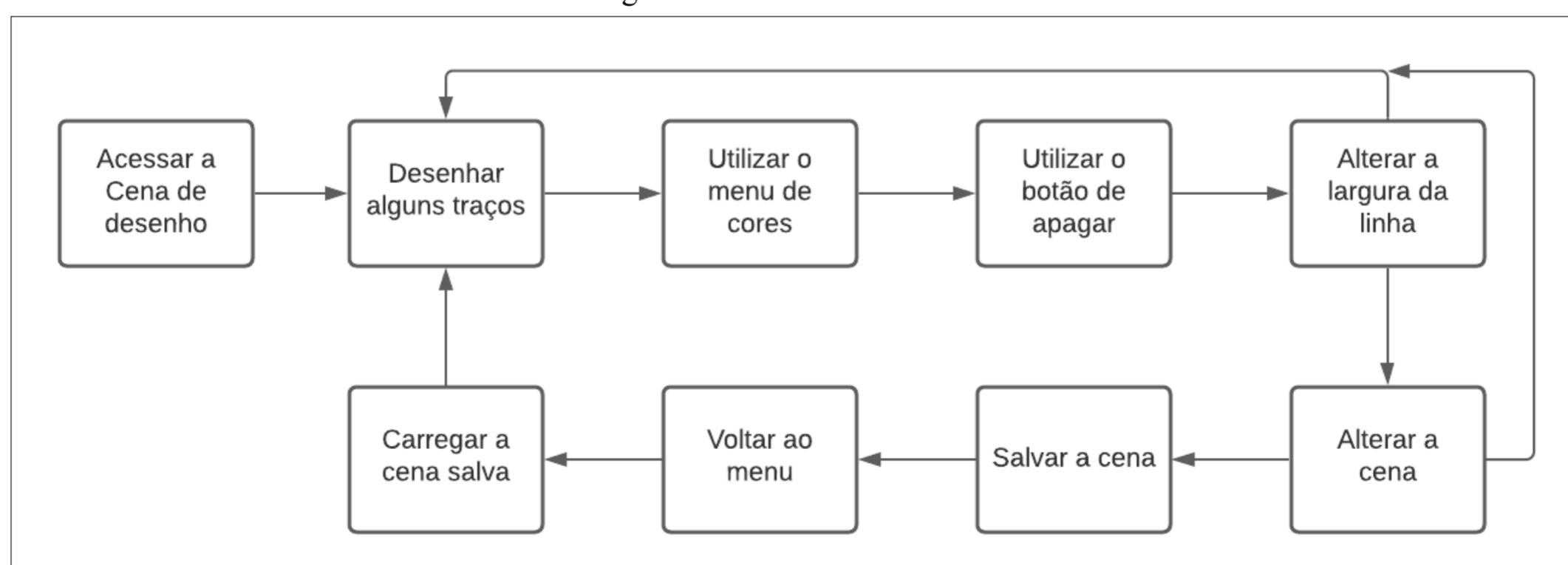
Fonte: elaborado pelo autor.

A principal relação da aplicação desenvolvida com os trabalhos correlatos, pode ser observada na comparação com o trabalho de Gerry (2017) em que o uso de RV possibilitou mais precisão no desenho. Já o trabalho de Lyu *et al.* (2017) demonstrou mais opções para melhorar a maneira como o usuário desenha e o trabalho de Falcao (2015) apresentou a importância de uma interface feita especificamente para a tecnologia alvo. Todos esses resultados foram unificados e desenvolvidos na aplicação apresentada, permitindo ao usuário uma nova experiência de desenho com o Leap Motion, como também aconteceu nos trabalhos correlatos. Destaca-se, ainda, que o conjunto de tecnologias utilizadas na presente aplicação diferem das utilizadas nos correlatos.

4.3 TESTE DE USABILIDADE

Para verificar se os usuários realmente tiveram uma experiência diferente de desenho, foram realizados dois testes de usabilidade em dois períodos diferentes, de modo a avaliar a evolução da aplicação e aplicar possíveis melhorias. Os testes foram realizados com quatro usuários de idades variando entre 11 e 14 anos que nunca tinham utilizado o Leap Motion. De início, no primeiro teste foi apresentado o Leap Motion e demonstrado como funciona, utilizando uma aplicação da própria empresa. Após os usuários terem conhecido o hardware, foi demonstrada a aplicação proposta e, então, seguiu-se um roteiro de testes solicitando a execução das atividades como demonstra a Figura 12.

Figura 12 - Roteiro de testes

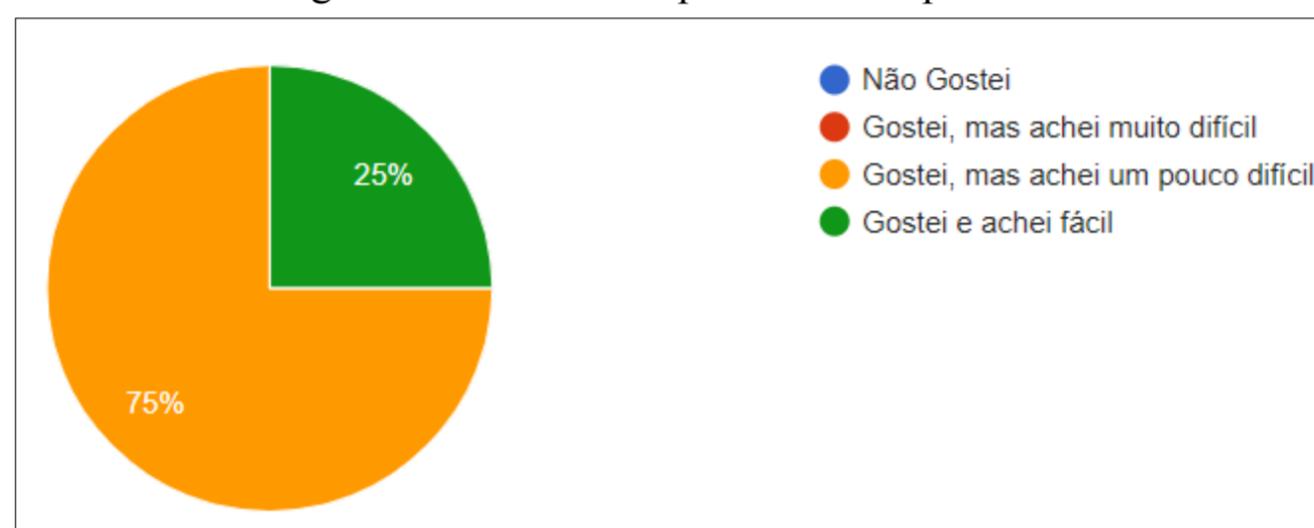


Fonte: elaborado pelo autor.

Após a realização do roteiro de testes foi aplicado um questionário no qual os usuários deveriam avaliar as ações realizadas na aplicação. Para cada ação, os usuários deveriam selecionar uma opção em uma escala de 1 a 10, sendo que 1 representava muito difícil e 10 muito fácil. Analisando as respostas do primeiro teste, pode-se notar que os usuários acharam muito fácil utilizar as ações de desenhar e movimentar o personagem, com notas entre 9 e 10. As ações de mudar a cor, espessura da linha e salvar o desenho, tiveram notas variando de 7 a 9, representando que não tiveram muita dificuldade de utilizar. Por outro lado, a ação de apagar teve resultados variando entre 5 a 7, sendo considerada, portanto, a que gerou mais dificuldade.

Também foi questionado se eles gostaram e acharam fácil utilizar a aplicação com o Leap Motion e com RV. Em relação ao Leap Motion, três deles disseram que gostaram, mas acharam um pouco difícil e o outro disse que gostou e achou fácil, como demonstra a Figura 13. Já para a RV as respostas ficaram divididas igualmente entre gostei, mas achei um pouco difícil e gostei eachei fácil. Ao final do questionário, foi perguntado de forma opcional o que poderia melhorar no aplicativo, tendo sido obtidas duas respostas: uma que indicou a necessidade de melhorar a ação de apagar e outra que a mão desaparecia às vezes.

Figura 13 - Uso do Leap Motion no aplicativo



Fonte: elaborado pelo autor.

Uma vez que no processo de aplicação foram observados erros e problemas não relatados pelos usuários, optou-se por fazer melhorias na aplicação de modo a tentar deixar a experiência mais interessante e imersiva. Após a implementação de algumas mudanças no aplicativo, o teste foi repetido com os mesmos usuários. As perguntas sobre as ações geraram resultados muito próximos aos anteriores, sendo a pergunta sobre a ação de desenhar a que teve a maior diferença. A Tabela 1 apresenta a média dos resultados em cada teste.

Tabela 1 - Média dos resultados de usabilidade

	Primeiro teste	Segundo teste
O que você achou da ação de desenhar?	8	9
O que você achou da ação de mudar a cor?	8	9
O que você achou da ação de mudar a espessura da linha?	8	9
O que você achou da ação de apagar o desenho?	6	8
O que você achou da ação de movimentar o personagem?	9	10
O que você achou da ação de mudar a cena?	8	8
O que você achou da ação de salvar a cena?	8	8

Fonte: elaborado pelo autor.

As questões sobre o Leap Motion e RV também apresentaram resultados parecidos, com uma pequena melhora. Para a utilização do Leap Motion, as respostas ficaram divididas igualmente entre gostei, mas achei um pouco difícil e gostei eachei fácil. Para RV teve uma resposta em gostei, mas achei um pouco difícil e três em gostei eachei fácil. Nesse caso, entretanto, os resultados podem ter sido influenciados pela experiência anterior que os usuários já haviam tido com os recursos.

5 CONCLUSÕES

Esse trabalho apresentou o desenvolvimento de uma aplicação para desenho utilizando o Leap Motion e RV, que foi desenvolvido utilizando o motor gráfico Unity e a linguagem de programação C#. O uso da Unity facilitou o desenvolvimento de aspectos gerais da aplicação, tais como a interface gráfica e interação entre os algoritmos e componentes gráficos. De forma geral, o uso do Leap Motion com RV se tornou desafiador de utilizar sem um aparelho com integração nativa com o *hardware*, sendo necessário o uso de uma aplicação como o Riftcat para acessar as funções do smartphone e utilizar a RV com o aparelho. Entretanto, seguir as instruções da empresa em como o aparelho funcionou corretamente e não apresentou problemas graves no uso, apenas pequenos defeitos que foram corrigidos na implementação.

Os resultados obtidos foram considerados bons, tanto em relação ao uso das tecnologias escolhidas quanto nos testes de usabilidade, sendo apenas a ação de apagar que obteve notas mais baixas no questionário aplicado. Acredita-se que essa ação não obteve um agrado alto dos usuários pela maneira em que foi implementada, pois permitir remover apenas o último traço realizado pelo usuário retira a imersão que a RV permite criar. Em relação ao último comentário realizado no questionário do primeiro teste, em que às vezes a mão desaparecia, esse problema acontece por causa do Leap Motion, pela facilidade em perder o rastreamento das mãos especialmente em um ambiente não controlado. Para melhorar esse aspecto acredita-se ser necessário alterar os parâmetros internos do Leap Motion de modo que funcione para um grupo específico de usuários. Na aplicação desenvolvida, esses valores foram readequados nas configurações dos comportamentos.

O aplicativo ofereceu uma nova experiência de desenho aos usuários permitindo usar a criatividade e sua percepção de mundo. Uma limitação verificada é a quantidade de tecnologias necessárias para que a aplicação funcione, as quais não são tão facilmente acessíveis. Assim, uma vez que se supere a questão tecnológica, é possível oferecer um recurso que alie a representação livre de ideias possibilitadas pelo desenho com recursos tecnológicos avançados em RV.

A partir destas discussões, levantam-se algumas extensões para expandir o trabalho realizado, sendo elas:

- a) adicionar as ações de mover e expandir o desenho;
- b) alterar a ação de apagar, para que possa ser possível excluir pontos da reta;
- c) adicionar a possibilidade de misturar as cores presentes no menu;
- d) reduzir a quantidade de tecnologias utilizadas, para permitir ficar mais acessível aos usuários.

REFERÊNCIAS

- BAIERLE, I. L. F.; GLUZ, J. C.; Watt: Imersão 3D Compartilhada e Acessível na Realidade Virtual do Surgimento da Revolução Industrial, In: BRAZILIAN SYMPOSIUM ON COMPUTERS IN EDUCATION (SIMPÓSIO BRASILEIRO DE INFORMÁTICA NA EDUCAÇÃO - SBIE), 10, 2017. Recife. **Anais...** Recife: SBC, 2017. p. 585-594.
- BRASIL. Ministério da Educação. **Base Nacional Comum Curricular**: educação é a base. Brasília: MEC, 2018.
- CARNEIRO, A. O Desenho, projeto da pessoa. In: SEMINÁRIO: OS DESENHOS DO DESENHO NAS NOVAS PERSPECTIVAS SOBRE ENSINO ARTÍSTICO, 1, 2000, Porto. **Actas...** Porto: EPCE, 2001. p.34.
- FALCAO, C.; LEMOS, A.; SOARES, M. Evaluation of natural user interface: a usability study based on the leap motion device. **Procedia Manufacturing**, v. 3, p. 5490-5495, 2015.
- FERREIRA, T. C.; **Artes visuais na educação infantil**: o desenho e o seu papel no desenvolvimento da criança. São Paulo: Contagem, 2020.
- GERRY, L. Paint with me: stimulating creativity and empathy while painting with a painter in Virtual Reality. **IEEE Transactions on Visualization and Computer Graphics**, v. 23, n. 4, p. 1418-1426, Apr. 2017.
- KIRNER, C.; SISCOUTTO, R. **Realidade virtual e aumentada**: conceitos, projeto e aplicações. Rio de Janeiro: Petrópolis, 2007.
- LEAP MOTION. **Leap motion documentation**. [S.I.], [2021]. Disponível em: <https://developer.leapmotion.com/documentation>. Acesso em: 16 jun. 2021.
- LYU, R. *et al.*, A flexible finger-mounted airbrush model for immersive freehand painting. In: 2017 IEEE/ACIS 16TH INTERNATIONAL CONFERENCE ON COMPUTER AND INFORMATION SCIENCE (ICIS), 16, 2017, Wuhan. **Proceedings...** Wuhan: IEEE, 2017. p. 395-400.
- MORAN, J. M. Desafios que as tecnologias digitais nos trazem. In: Moran, J. M.; Masetto, M. T.; Behrens, M. A. **Novas tecnologias e mediação pedagógica**. 21. ed. Campinas, SP: Papirus, 2013.
- MORAN, J. M. **Tablets e netbooks na educação**. [S.I.], 2012. Disponível em: http://www.eca.usp.br/prof/moran/site/textos/tecnologias_eduacacao/tablets.pdf. Acesso em: 12 de out. de 2020.
- MOREIRA PINTO, L. M. M. **Desenho, percepção e forma**. Lisboa: Colibri, 2012.
- PATTEN, B. *et al.* Designing collaborative, constructionist, and contextual applications for handheld devices. **Computers & Education**, Oxford, UK: Elsevier Science Ltd, v. 46, p. 294-308, 2006.
- RIFTCAT. **VRidge**: play PC VR on your cardboard. [S.I.], 2020. Disponível em: <https://riftcat.com/vridge>. Acesso em: 10 out. 2020.
- SILVA, V. A. F. *et. al.* Realidade Virtual. **Revista Interface Tecnológica**, [S. l.], v. 14, n. 2, p. 7-18, 2017.
- TORI, R.; KIRNER, C.; SISCOUTTO, R. **Fundamentos e tecnologia de realidade virtual e aumentada**. Porto Alegre: Editora SBC, 2006. 412 p.
- ULTRALEAP. **Leap motion controller**. [S.I.], [2021?]. Disponível em: https://www.ultraleap.com/datasheets/Leap_Motion_Controller_Datasheet.pdf. Acesso em: 10 jun. 2021.
- VAITKEVIČIUS, A. *et al.* Recognition of american sign language gestures in a virtual reality using leap motion. **Applied Sciences**, [S. l.], v. 9, n. 3, p. 1-26, 2019.

WOZNIAK, P. *et al.* Possible applications of the LEAP motion controller for more interactive simulated experiments in augmented or virtual reality. In: SPIE OPTICAL ENGINEERING + APPLICATIONS, 2016, San Diego. **Proceedings ...** San Diego, California: SPIE, set. 2016. p. 1-13.