

3

v-Glove: um dispositivo de interação para aplicações imersivas de realidade virtual

A proposta desse trabalho é a construção de uma luva para interação com uma tela de toque virtual mapeada em um sistema imersivo. Neste capítulo serão detalhados os componentes da solução proposta e, por isso, optou-se por uma abordagem narrativa na sequência em que o trabalho de prototipação foi desenvolvido. Assim os componentes criados serão apresentados na ordem em que foram desenvolvidos até que se chegasse à solução mais adequada. Situações avaliadas e que tenham sido descartadas posteriormente serão igualmente mencionadas com a finalidade de demonstrar o aprendizado adquirido durante cada uma das etapas do trabalho.

3.1.

Princípio de funcionamento

O princípio de funcionamento da v-Glove baseia-se no rastreamento da posição da ponta do dedo indicador da luva e do seu mapeamento para coordenadas em duas dimensões (X e Y), de forma similar a um mouse de mesa tradicional. Ao movimentar o dedo para o lado direito, por exemplo, o cursor do mouse desloca-se na tela seguindo esse mesmo padrão de movimento. A operação de clicar e arrastar objetos na superfície virtual funciona também baseada no rastreamento, sendo que neste caso é usado um terceiro eixo de referência (Z). Ao aproximar a ponta do dedo de um ponto pré-determinado em Z, o sistema detecta a intenção de clicar no objeto apontado pelo cursor naquele instante. Para reproduzir o movimento de arrastar basta manter o dedo nessa área de clique enquanto movimenta-se a mão nos eixos X e Y.

3.2.

Visão geral

A solução elaborada neste trabalho é composta pelos componentes de hardware mostrados na Figura 3.1. Todos os elementos construídos nesse

trabalho foram projetados pensando-se na sua utilização em uma Cave e, por isso, foram adaptados a características ergonômicas desse tipo de ambiente como a necessidade de mobilidade do usuário e a baixa luminosidade. Nas próximas seções serão apresentados em detalhes cada um desses componentes.

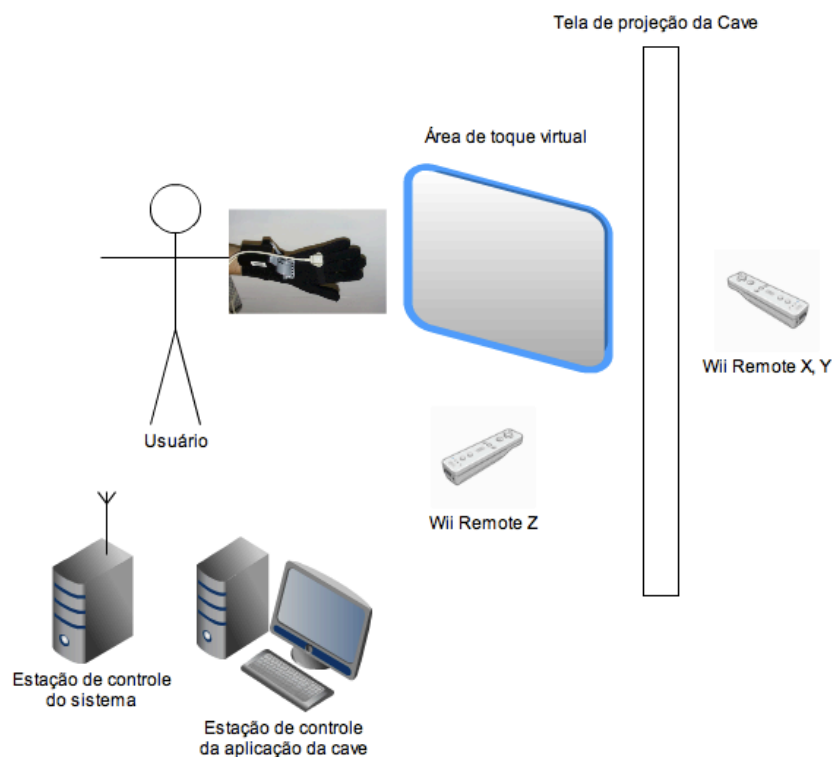


Figura 3.1: Visão geral da solução proposta

3.3. Luva

A primeira etapa deste trabalho foi a confecção de uma luva que permita a interação do usuário com a tela de toque virtual. A luva é feita de microfibra e foi costurada a partir de um molde de mão de um homem adulto, de forma a garantir um tamanho adequado para um maior número de usuários (Figura 3.2). Nada impede que futuramente sejam confeccionados modelos em tamanhos diferenciados conforme a necessidade. A escolha do tecido levou em consideração sua maleabilidade e espessura, pois é fundamental que a luva não restrinja a movimentação das mãos do usuário ou sua capacidade de manusear

uma *wand* sem ter que retirar a luva. Mesmo considerando-se que esses componentes serão utilizados em ambientes com controle de temperatura, é importante considerar que a luva não esquite demasiadamente a mão do usuário, por isso a preocupação com a espessura do tecido.



Figura 3.2: Primeiro protótipo da v-Glove

À luva foram inicialmente costurados os componentes eletrônicos responsáveis pelo sistema de feedback tátil. A costura dos componentes foi feita utilizando-se uma linha de costura com propriedades condutivas especialmente criada para essa finalidade [SparkFun 2011]. A linha de costura é banhada em prata e apresenta resistência elétrica de aproximadamente 45 Ohms por metro. A Figura 3.3 mostra um diagrama esquemático do circuito costurado na luva.

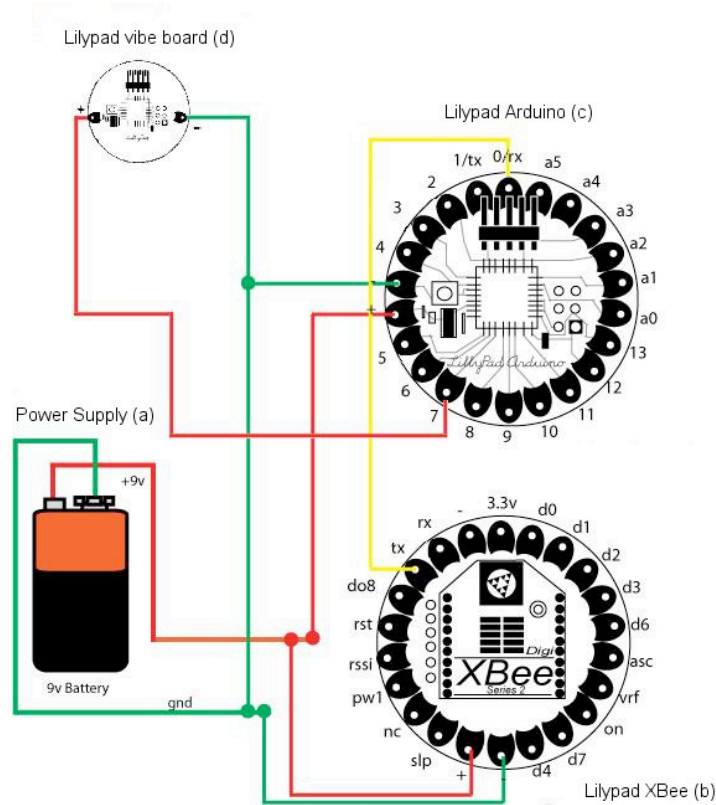


Figura 3.3: Diagrama esquemático do circuito da luva [Tune Glove 2011]

Uma bateria de 9V (a) provê a alimentação de todos os componentes do circuito. O módulo XBee acoplado ao adaptador Lilypad (b) recebe através de sua antena o comando proveniente da estação de controle. Os bytes são então enviados à porta de transmissão (tx) que está ligada diretamente à porta de recepção (0/rx) do micro-controlador Lilypad Arduino (c). O software armazenado na memória do micro-controlador recebe essa informação e envia sinal para a porta de número 7, que, por sua vez, está diretamente conectada ao polo positivo do atuador de vibração (d). Este, ao receber o sinal, faz com que o usuário sinta uma leve vibração na ponta dos dedos. A duração da vibração é controlada pelo software executado pelo micro-controlador Lilypad Arduino e dura 300 milissegundos. Maiores informações sobre este programa podem ser encontradas no Apêndice A.

3.3.1.

Micro controlador Arduino

O micro-controlador Lilypad Arduino é uma versão do já consagrado processador Arduino [Arduino 2011] desenvolvida pela empresa norte-americana SparkFun Electronics [SparkFun 2011] com a finalidade de ser usado em roupas [Buechley et al 2008]. Este componente é composto por um microprocessador programável modelo Lilypad Arduino 328 (modelo ATmega328V) montado sobre uma placa de circuito impresso da linha de componentes Lilypad (Figura 3.4).

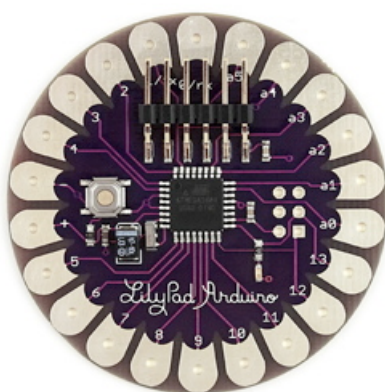


Figura 3.4: Micro-controlador Lilypad Arduino 328 [SparkFun 2011]

O Lilypad Arduino pode ser alimentado com tensões de 2.7 a 5.5 volts, possui quatorze portas de entrada e saída digital e seis portas de entrada analógica. Ele possui uma memória interna não volátil de 16 KB para armazenamentos dos programas e velocidade de *clock* é de 8 MHz. O módulo pode ser alimentado através de uma conexão USB utilizando um cabo FTDI/USB ou por uma fonte de alimentação externa.

3.3.2.

Antena e modem XBee

O modem XBee é um componente fornecido pela empresa Digi International Inc. [Xbee 2011] e que pode ser acoplado a um módulo Lilypad XBee fornecido pela SparkFun Electronics para sua utilização em circuitos vestíveis de forma similar ao microprocessador Lilypad Arduino. O XBee implementa o protocolo de comunicação sem fio ZigBee [ZigBee 2004] e pode ser configurável através de uma conexão FTDI/USB de forma similar a modems

tradicionais. Mais detalhes sobre a configuração do XBee e os parâmetros utilizados nesse trabalho podem ser encontrados no Apêndice B. O modelo utilizado neste trabalho foi o XBee 1mW Chip Antenna, cuja antena é em formato de micro-chip, possui potência de 1mW, alcance de até 100 metros e taxa de transmissão de dados de 250 kbps.

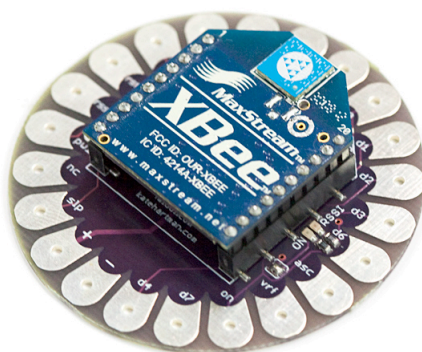


Figura 3.5: Modem XBee 1mW Chip Antenna [SparkFun 2011]

O módulo XBee montado na luva foi configurado como receptor. Um outro módulo idêntico foi conectado à estação de controle via porta USB, sendo a única diferença neste caso que o componente foi configurado como transmissor. Ao detectar um evento de toque, a aplicação executada na estação de controle envia um comando para o módulo XBee transmissor acoplado a ela. Este, por sua vez, envia o comando para o XBee receptor acoplado à luva que repassa a informação ao controlador Arduino, gerando o efeito de feedback tátil.

3.3.3. Feedback tátil

O componente Lilypad Vibe Board, também fornecido pela SparkFun Electronics, é um elemento atuador capaz de gerar uma vibração quando alimentado por um circuito de 5 Volts. Também construído em uma configuração vestível ele pode ser costurado de forma a funcionar com outros componentes similares como o Lilypad Arduino.



Figura 3.6: Lilypad Vibe Board [SparkFun 2011]

Para possibilitar o efeito desejado com o recurso de feedback tátil, o Lilypad Vibe Board foi costurado à ponta do dedo indicador da v-Glove.

3.3.4. Marcadores retroreflexivos

Na ponta do dedo indicador da luva é aplicada uma fita refletiva 3M modelo 8910 [3M 2011] (Figura 3.7(a)). Essa fita tem como finalidade refletir a luz infravermelha emitida pelo iluminador infravermelho posicionado próximo às câmeras e auxiliar no sistema de rastreamento dos dedos do usuário. A luz refletida pela fita é captada pela câmera IR e assim a posição do dedo do usuário pode ser estimada.



Figura 3.7: (a) Tecido refletivo 3M 8910 [3M 2011], (b) Tecido refletivo aplicado à v-Glove

Como sistema proposto utiliza duas câmeras IR, uma para captura das coordenadas X e Y e a outra para a coordenada Z, é importante que a fita retro-reflexiva seja visível pelas duas câmeras. Assim, a maneira mais adequada de se posicionar a fita para garantir uma boa visibilidade é envolvendo-a na ponta do dedo, conforme demonstrado na Figura 3.7(b).

Durante o desenvolvimento desse trabalho foram avaliadas algumas opções de materiais para serem usados como marcadores reflexivos. Uma primeira análise foi referente à utilização do dedo sem qualquer tipo de revestimento, mas viu-se que a pele humana reflete uma quantidade muito baixa de luz infravermelha. Uma segunda tentativa foi feita com papel alumínio convencional de uso doméstico. Este material possui uma alta capacidade de reflexão de luz convencional e também infravermelha, entretanto pela sua formação irregular a reflexão da luz infravermelha é captada com muito ruído pelas câmeras. A terceira opção avaliada foi a fita refletiva 3M 8910. Essa fita é bastante utilizada como elemento de sinalização urbana por sua alta capacidade refletiva à luz convencional e sua grande mobilidade (ela pode ser facilmente costurada a roupas, como por exemplo em uniformes de agentes de trânsito). O tecido refletivo 3M 8910 é composto por microesferas de vidro com grande angularidade que possibilitam alta visibilidade noturna e elevados índices de retro-refletividade. Nossos testes demonstraram que seu comportamento com luz infravermelha segue o mesmo padrão da luz convencional.

3.4. Câmeras de rastreamento

A câmera escolhida para o sistema de rastreamento foi a que acompanha o controle remoto do vídeo game Wii, conhecido como Wii Remote [Wii Remote 2011]. Essa câmera já possui um filtro para capturar luz apenas na faixa infravermelha, com maior sensibilidade para fontes de IR com comprimento de onda próximos a 940nm. O Wii Remote utiliza tecnologia Bluetooth de comunicação sem fio [Bluetooth 2011]. O modelo utilizado neste trabalho pode ser visto na Figura 3.8.



Figura 3.8: Wii Remote [Wii Remote 2011]

As principais vantagens da utilização do Wii Remote em relação a uma câmera convencional são: ele é sem fio, já possui filtro IR, possui bibliotecas que fornecem as coordenadas X e Y dos pontos de luz infravermelha capturados. Uma desvantagem do Wii Remote é que ele é capaz de identificar apenas quatro pontos de luz infravermelha simultaneamente, o que pode limitar uma aplicação que tenha a intenção de rastrear todos os dedos da mão. Isso não será um problema no nosso caso, pois a prova de conceito proposta nessa dissertação utiliza apenas o dedo indicador do usuário.

3.4.1. Posicionamento das câmeras

Como a câmera do Wii Remote fornece as coordenadas dos pontos em um plano de duas dimensões, acrescentamos uma segunda câmera posicionada lateralmente para a captura do eixo de profundidade. Dessa forma a solução final ficou sendo composta por duas câmeras independentes dispostas conforme o diagrama da Figura 3.9.

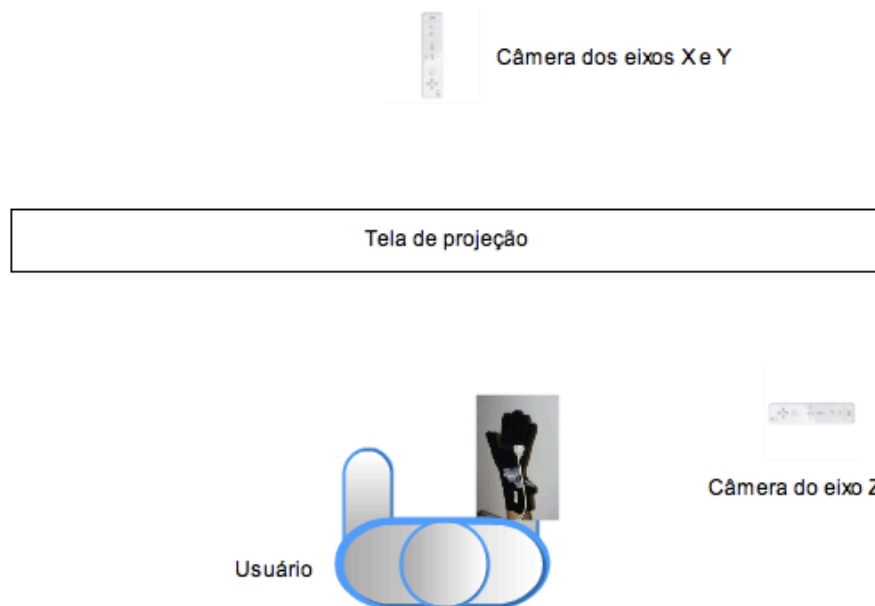


Figura 3.9: Posicionamento das câmeras IR em relação ao usuário

A primeira câmera é responsável pelo rastreamento nos eixos X e Y. Ela indica o posicionamento dos dedos do usuário nessas duas dimensões. A segunda câmera fica responsável pelo eixo Z. Ela fica posicionada na lateral do usuário e faz o mapeamento da superfície de toque virtual. Quando o dedo indicador do usuário se aproxima desse plano mapeado no espaço o software reconhece um toque na superfície e dispara um evento de feedback tátil para o software da luva. Assim é acionada a vibração no dedo do usuário.

Durante a construção do protótipo avaliamos a possibilidade de utilizar as câmeras com posicionamento baseado nos conceitos de geometria epipolar, [Hartley e Zisserman 2004]. Esta técnica tem como vantagem permitir a reprodução do espaço virtual em três dimensões, facilitando o rastreamento de múltiplos dedos, o que é impossível na topologia de câmeras atual. Isso ocorre porque a posição lateral da câmera do eixo Z faz com que ocorra oclusão entre dedos próximos, impossibilitando a detecção de profundidade dos dedos. Entretanto, como inicialmente não há necessidade de se identificar mais de um dedo e a solução atual não requer calibração das câmeras, procedimento necessário ao utilizar-se geometria epipolar, esta solução acabou sendo descartada. Além disso, como nosso interesse resumia-se a identificar se a v-Glove atingiu um determinado ponto no plano de toque virtual, a solução com a

câmera lateral atendeu aos requisitos da prova de conceito sem impactar na complexidade do software construído. Ela não é afetada por variações no posicionamento das câmeras e reduz o tempo de preparação do ambiente para a realização dos testes uma vez que não demanda calibração prévia.

3.4.2. Iluminador infravermelho

A câmera infravermelha presente no Wii Remote é sensível a uma certa intensidade de luz infravermelha. Considerando-se que essa aplicação será utilizada em um ambiente imersivo com iluminação reduzida, o componente iluminador infravermelho é fundamental. Ele é formado por oito LEDs infravermelhos de 5mm e comprimento de onda de 940nm alimentados por uma fonte de alimentação de 12 volts (Figura 3.10).

Os LEDs utilizados no iluminador são do mesmo tipo utilizado por dispositivos de controle remoto domésticos e podem ser facilmente encontrados em lojas de componentes eletrônicos.

Esse componente iluminador tem como finalidade a emissão de luz infravermelha que será refletiva pela fita retro-reflexiva presente na luva sendo então capturada pela câmera do Wii Remote.

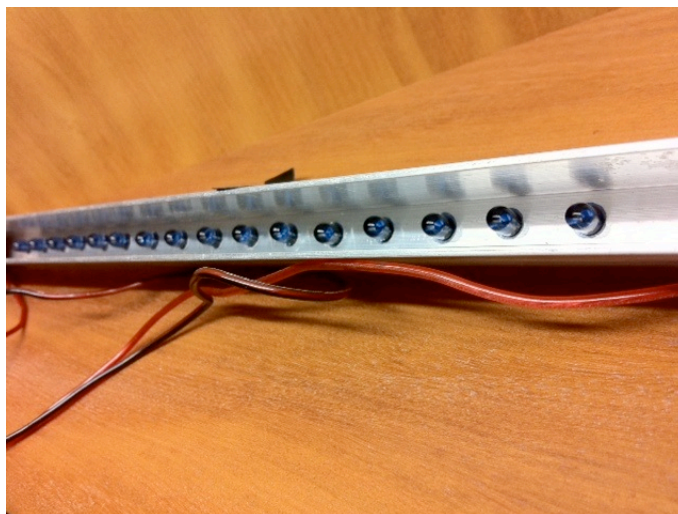


Figura 3.10: Iluminador infravermelho

Uma alternativa à utilização do iluminador infravermelho seria a colocação de um LED infravermelho diretamente na ponta do dedo indicador da luva, em

substituição à fita retro-reflexiva. Nesse caso a luz emitida pelo LED seria diretamente capturada pela câmera do Wii Remote, evitando-se o processo da reflexão. O problema dessa abordagem é que a presença de um LED na luva aumentaria o consumo de bateria da mesma, diminuindo-se assim sua autonomia. Pensando-se ainda na utilização de mais de um ponto de rastreamento esse problema aumentaria ainda mais.

3.5.

Estações de controle

As estações de controle representam os microcomputadores utilizados no ambiente imersivo para controle do software da aplicação e da v-Glove. Neste trabalho definiremos conceitualmente duas estações de controle, que serão descritas nas subseções a seguir. Apesar de serem duas estações de controle, nada impede que ambas compartilhem a mesma plataforma de hardware.

3.5.1.

Estação de controle de rastreamento e feedback tátil

A estação de controle do sistema de rastreamento e feedback tátil é o microcomputador onde o software desenvolvido é executado. Maiores detalhes sobre o software instalado nesta máquina serão discutidos na seção 3.7 (Arquitetura de Software). A Figura 3.11 mostra a estação de controle de rastreamento e feedback tátil com o modem XBee acoplado a uma das portas USB disponíveis na traseira do gabinete.



Figura 3.11: Estação de controle de rastreamento e feedback tátil com o modem XBee

Essa estação de controle é o componente do sistema responsável por gerenciar as coordenadas X, Y e Z enviadas pelas câmeras bem como o subsistema de feedback tátil. Além disso, o software presente nessa estação é responsável pela integração com as aplicações que são executadas na Cave de forma a fornecer eventos de entrada que possam permitir a interação do usuário com essas aplicações.

3.5.2. Estação de controle da aplicação

Este elemento do sistema é responsável pela execução da aplicação no sistema imersivo de realidade virtual. Em geral a estação de controle da aplicação é um microcomputador com uma elevada capacidade computacional e uma placa controladora gráfica de alto desempenho com capacidade para renderização de imagens em tempo real e suporte estereoscópico. No contexto deste trabalho a estação de controle da aplicação será responsável pela execução do programa Environ, que será melhor detalhado no capítulo 4 (Estudo de Caso: Environ).

3.6. Sistema da Cave

Cruz-Neira et al (1992) definem a Cave como uma interface de realidade virtual composta por uma sala cujas paredes, o teto e o piso possuem imagens projetadas. Um usuário presente neste ambiente tem seu entorno coberto por essas projeções. Ainda segundo os autores, o conceito da Cave se baseia em dois princípios: *suspension of disbelief*¹ e a perspectiva centrada no espectador. *Suspension of disbelief* indica a capacidade de um usuário de “se deixar levar” pela simulação, ou seja, de aceitar sua condição imersiva como real dentro do contexto da aplicação que está sendo utilizada. A perspectiva centrada no espectador propõe a visualização na perspectiva da localização do usuário, fazendo uso para isso de sensores que monitoram continuamente sua posição no espaço, ajustando a imagem projetada de acordo com essas informações.

O trabalho de Cruz-Neira et al (1992) foi o primeiro a propor essa idéia de telas ao redor do usuário. A partir dele, outras propostas surgiram com algumas variações e adaptações a contextos específicos de aplicação. Bowman et al (2004) citam alguns exemplos de trabalhos similares, como o CUBE (interface com 360 graus de visibilidade composta por 4 telas de projeção que ficam a cerca de 30 centímetros do rosto do usuário), a RAVE (dispositivo com tela flexível que pode ser configurado como uma tela plana de 9 metros ou como um ambiente imersivo com 4 telas), dentre outros.

¹ Suspensão da descrença seria uma tradução livre do termo *suspension of disbelief*. Entretanto, por não haver referência a este termo na literatura em português optamos por manter no texto o termo original em inglês.

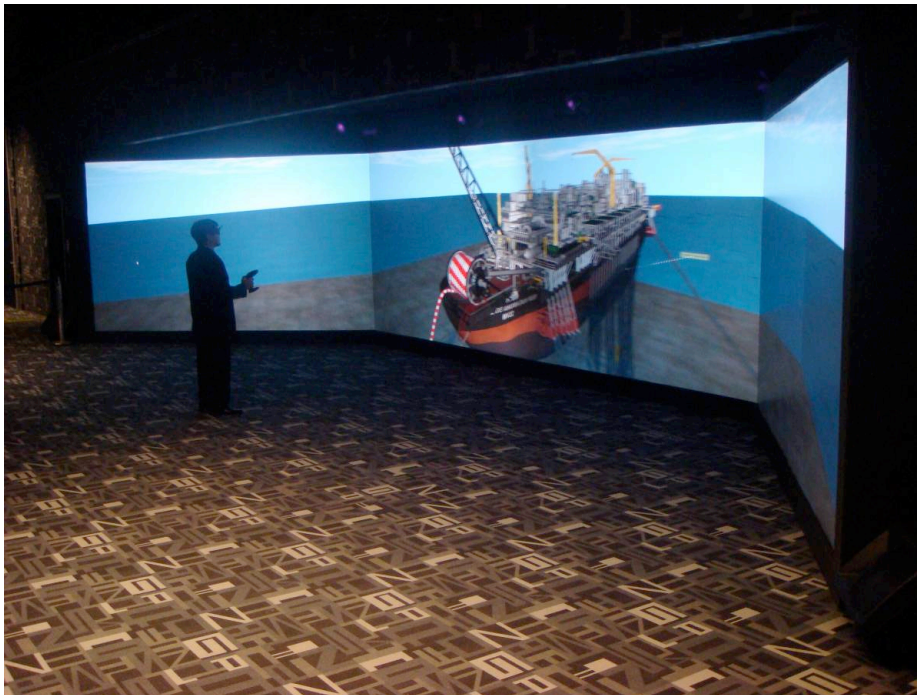


Figura 3.12: Cave montada no CENPES da Petrobras

A Figura 3.12 mostra a Cave construída no Centro de Pesquisas da Petrobrás (CENPES), no Rio de Janeiro. Como pode ser observado na imagem, esta Cave possui três telas com projeção estereoscópica de alta definição, não possuindo projeção no piso nem no teto. Um sistema de rastreamento permite ainda que a localização da cabeça do usuário seja rastreada por câmeras dispostas na parte superior das telas provendo suporte à perspectiva centrada no espectador.

3.7. Arquitetura de software

Todo o software desenvolvido neste trabalho usou como base o paradigma da orientação a objetos e a linguagem de programação escolhida foi Java. As bibliotecas de terceiros utilizadas seguem este mesmo princípio e possuem na sua maioria código fonte aberto ou possuem autorização de seus responsáveis para utilização em projetos sem fins lucrativos como este.

A arquitetura do software foi dividida em 5 módulos principais que interagem de forma a prover todas as funcionalidades necessárias para a aplicação da luva. Nas próximas seções esses módulos são apresentados em detalhes.

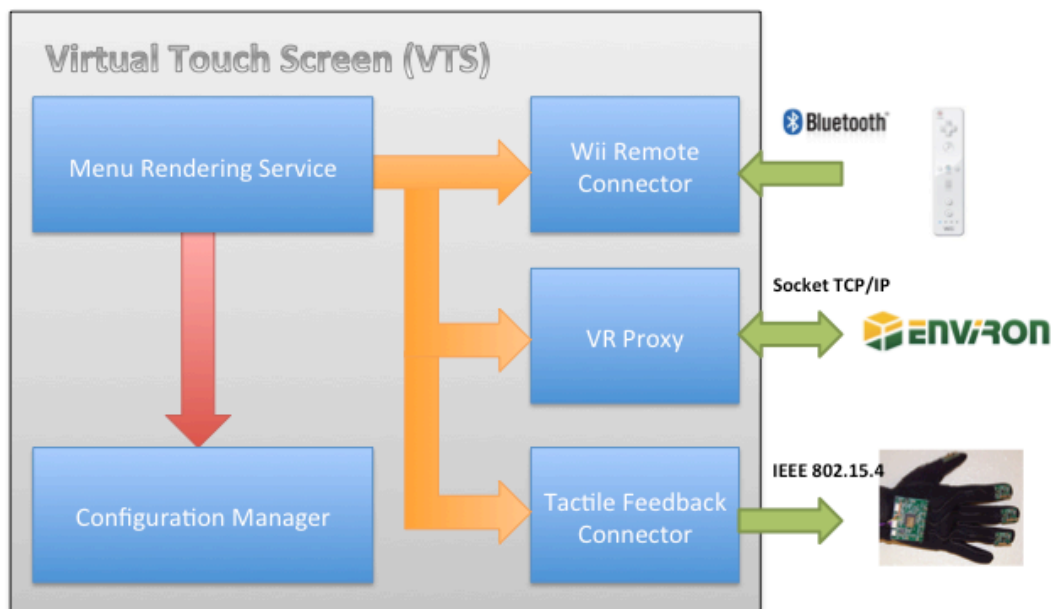


Figura 3.13: Diagrama de arquitetura do software

O diagrama de arquitetura da Figura 3.13 indica quais são esses módulos e mostra os elementos externos com os quais cada um deles interage, informando em cada caso a forma como é feita a integração. O componente VR Proxy interage com o programa Environ através de uma conexão socket TCP/IP. O conector de integração com o Wii Remote, por sua vez, realiza a comunicação através do protocolo Bluetooth, enquanto o módulo de controle do feedback tátil se comunica com a luva através do protocolo ZigBee (IEEE 802.15.4).

3.7.1. Menu Rendering Service

Este componente pode ser definido como o módulo central da aplicação. Ele é o primeiro a ser executado e é a partir dele que os módulos que interagem com os componentes externos do sistema são acionados. O serviço de renderização de menus é responsável também pela construção da interface gráfica de controle do software. A Figura 3.14 mostra o diagrama de classes desse componente do sistema.

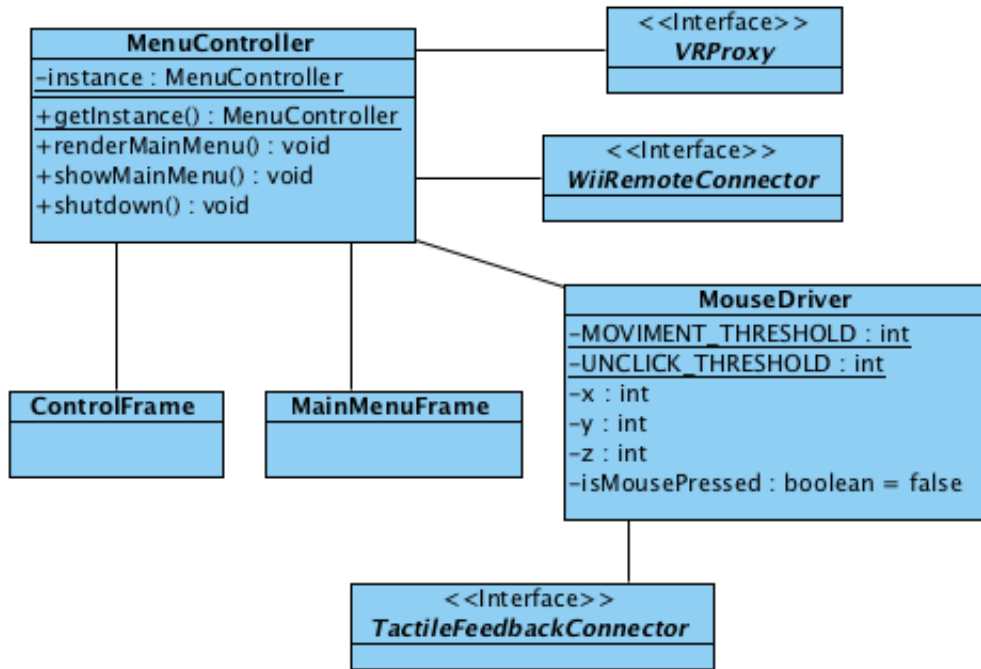


Figura 3.14: Diagrama de classes do Menu Rendering Service

A classe MenuController interage com os outros módulos do programa como VRProxy, WiiRemoteConnector e MouseDriver de forma a disparar ações de acordo com a forma de interação do usuário com os menus e controles da aplicação. A classe MouseDriver é responsável por gerenciar o movimento do cursor da v-Glove na tela e acionar o recurso de feedback tátil através da interface TactileFeedbackConnector. Existem ainda as classes ControlFrame e MainMenuFrame que representam painéis da interface gráfica com os quais o usuário interage.

3.7.2. Wii Remote Connector

Este módulo é o responsável pela comunicação do software com o Wii Remote para a captura das informações provenientes da câmera infravermelha. O software suporta a utilização de duas bibliotecas de conexão com o Wii Remote: Wiigee [Wiigee 2011] e WiiRemoteJ [WiiRemoteJ 2011]. As duas bibliotecas foram escritas em Java e utilizam o driver Bluetooth BlueCove versão 2.1.0 [BlueCove 2011]. Ambas foram testadas e possuem recursos similares, sendo que a WiiRemoteJ possui como vantagem o fato de permitir o ajuste de sensibilidade da câmera infravermelha do Wii Remote. Por esse motivo a

biblioteca WiiRemoteJ foi utilizada na versão final do software. A Figura 3.15 mostra o diagrama de classes do Wii Remote Connector.

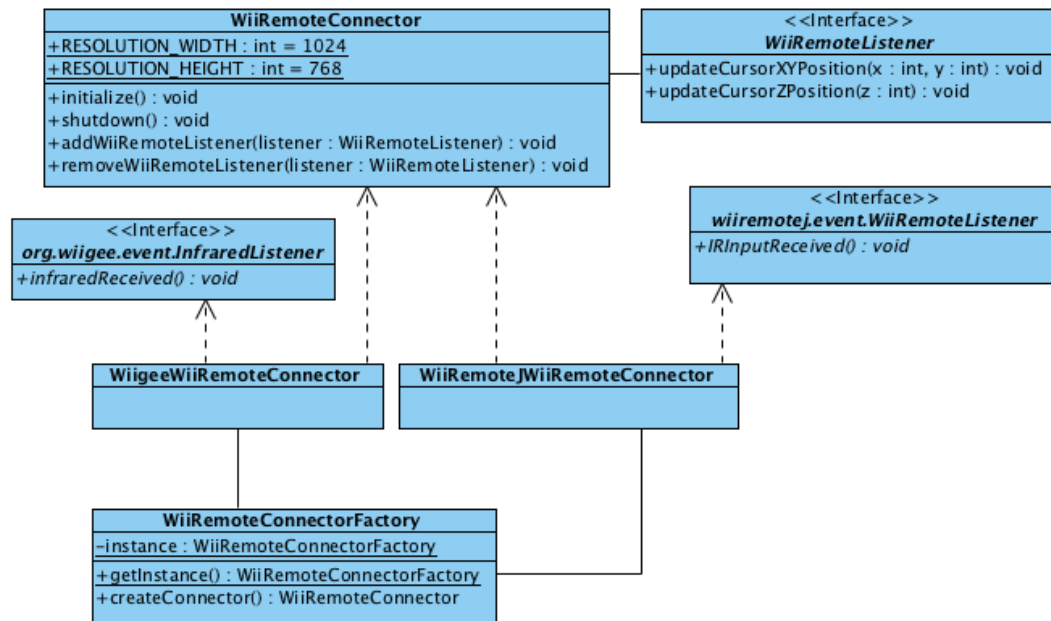


Figura 3.15: Diagrama de classes do Wii Remote Connector

A interface *WiiRemoteConnector* define as operações necessárias para uma implementação de conector para o Wii Remote. Como já foi mencionado, foram criadas duas implementações, uma utilizando a biblioteca Wiigee (e representada pela classe *WiigeeWiiRemoteConnector*) e outra utilizando a biblioteca *WiiRemoteJ* (representada pela classe *WiiRemoteJWiiRemoteConnector*). A classe *WiiRemoteConnectorFactory* é responsável pela criação de instâncias de conectores, enquanto as interfaces de *listener* são utilizadas para a recepção das coordenadas dos pontos de infravermelho a serem tratados pela aplicação.

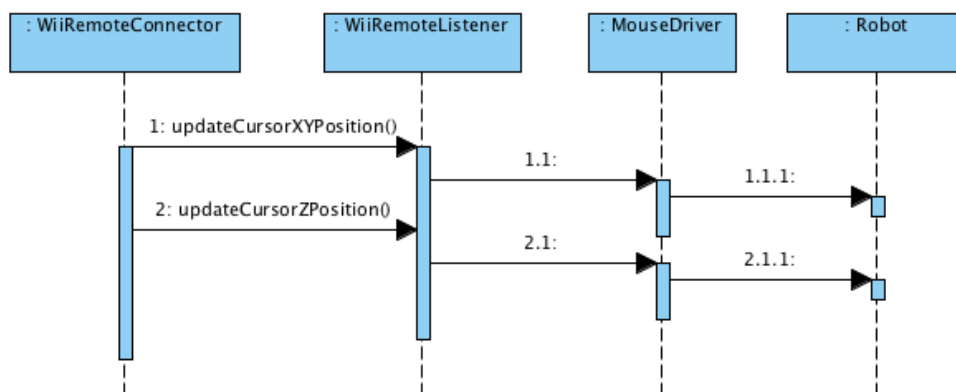


Figura 3.16: Diagrama de sequência da recepção de um evento de infravermelho

A Figura 3.16 mostra o diagrama de sequência referente ao fluxo de recepção de um evento de atualização de coordenadas desde o WiiRemoteConnector até seu tratamento pela aplicação. A classe MouseDriver implementa a interface WiiRemoteListener e recebe, através dela, eventos de atualização das coordenadas X e Y pelo método updateCursorXYPosition() e Z pelo método updateCursorZPosition(). Ela então aciona as chamadas de sistema operacional que controlam o posicionamento e o estado do cursor do mouse através da classe Robot, fornecida pela API padrão da linguagem Java.

3.7.3. VR Proxy

Este componente é responsável pela abstração da aplicação que está sendo utilizada na Cave em relação à luva. Ele define uma interface padronizada para a comunicação entre os módulos da aplicação e as aplicações imersivas de realidade virtual que serão utilizadas com a luva. A implementação de referência foi baseada no programa Environ que será alvo do estudo de caso apresentado no capítulo 4 (Estudo de caso: Environ). A Figura 3.17 mostra o diagrama de classes deste componente.

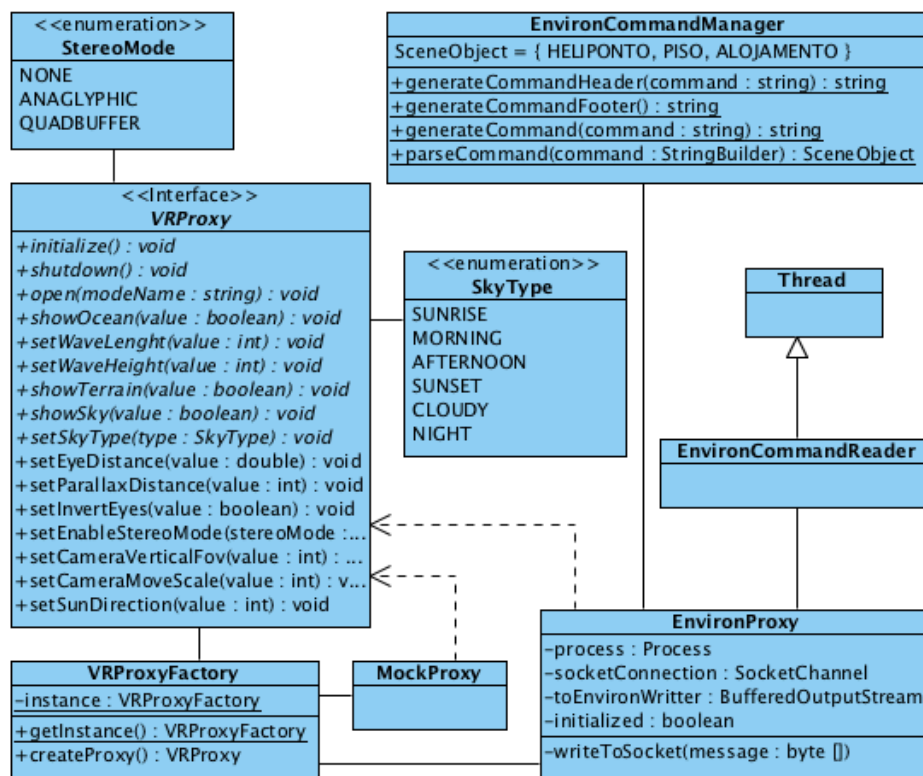


Figura 3.17: Diagrama de classes do VR Proxy

A interface VRProxy define as operações necessárias para uma implementação de proxy para aplicações de realidade virtual. A classe EnvironProxy representa a implementação de referência para o Environ e a classe MockProxy uma implementação para testes internos. A classe EnvironCommandManager é responsável pela conversão dos comandos de realidade virtual para funções no formato suportado pelo Environ. A classe EnvironCommandReader faz o tratamento dos comandos enviados pelo Environ em resposta aos comandos enviados pela aplicação. A classe VRProxyFactory é responsável pela criação de instâncias de proxy, enquanto o enumerador SkyType define os tipos de céu suportados pela aplicação e o enumerador StereoMode define os modos de operação estereoscópicos suportados.

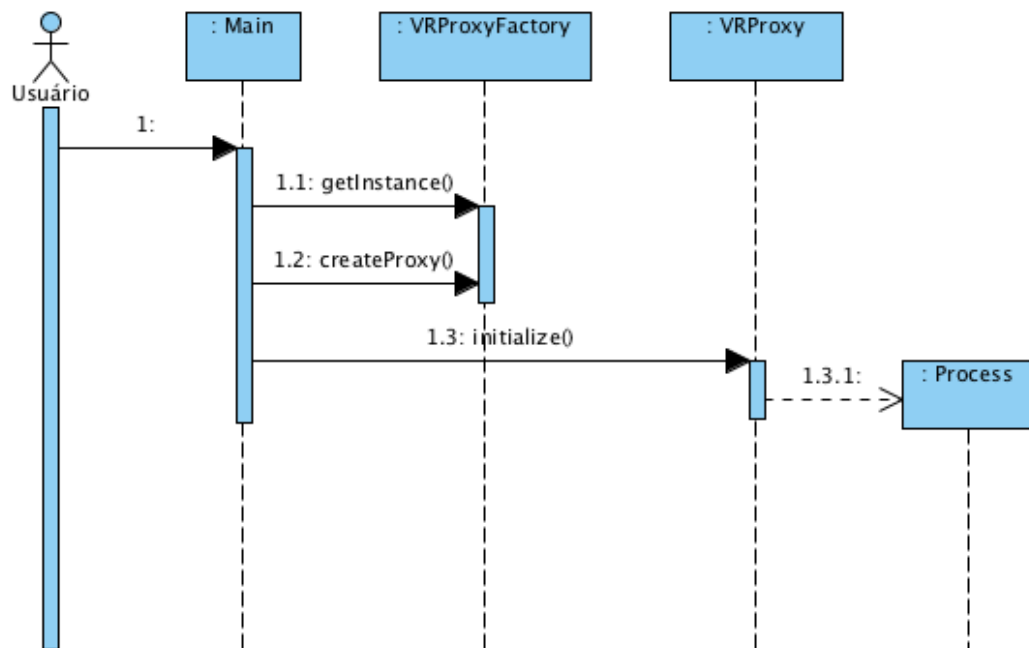


Figura 3.18: Diagrama de sequência do carregamento do Environ

A Figura 3.18 mostra o diagrama de sequência da operação de carregamento do Environ. Ao iniciar o software a classe MenuController é acionada e inicializa o componente VRProxy a partir da VRProxyFactory. Uma instância de EnvironProxy é devolvida e o proxy é iniciado, fazendo com que um novo processo seja criado no sistema operacional executando o arquivo binário do Environ.

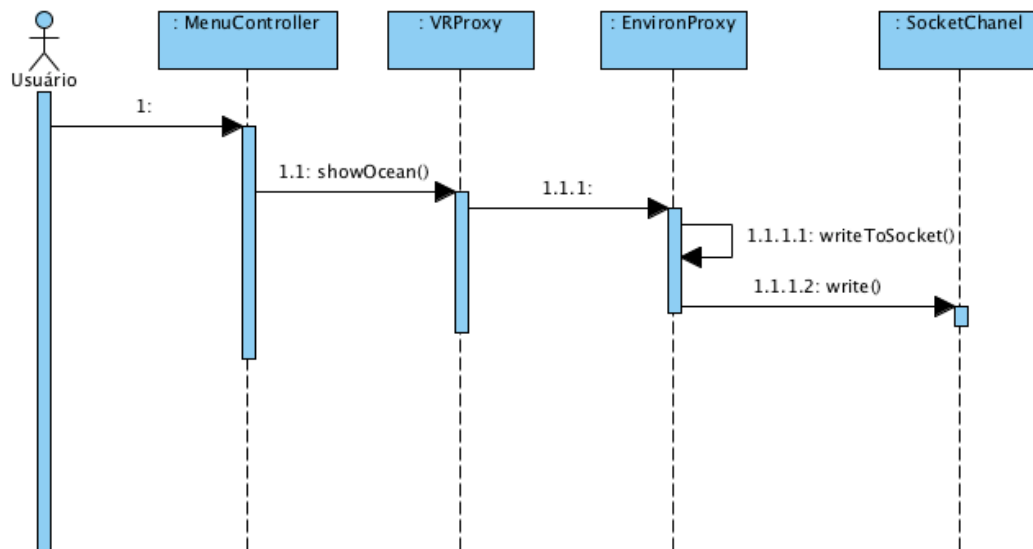


Figura 3.19: Diagrama de sequência do envio de um comando para o Environ

A Figura 3.19 mostra o diagrama de sequência da operação de envio de um comando da aplicação para o Environ. Ao clicar em uma opção no menu da aplicação é gerado um evento de interface gráfica de usuário. Esse evento é tratado pelo MenuController que, por sua vez, encaminha a execução do comando associado ao evento ao VRProxy. O EnvironProxy então envia um comando executável através da interface socket TCP/IP para o Environ, que finalmente executa o comando e responde informando se a execução foi concluída com sucesso ou não. Em caso de erro uma mensagem contendo o código e uma breve descrição do erro é retornada para o usuário.

3.7.4. Tactile Feedback Connector

Este componente é responsável pela geração do feedback tátil que provê ao usuário da luva a sensação de estar de fato tocando em uma superfície ao interagir com o sistema. Ele foi concebido de forma a suportar diversas estratégias de feedback tátil, tendo como implementação de referência a já mencionada com os componentes Lilypad Arduino (Figura 3.20).

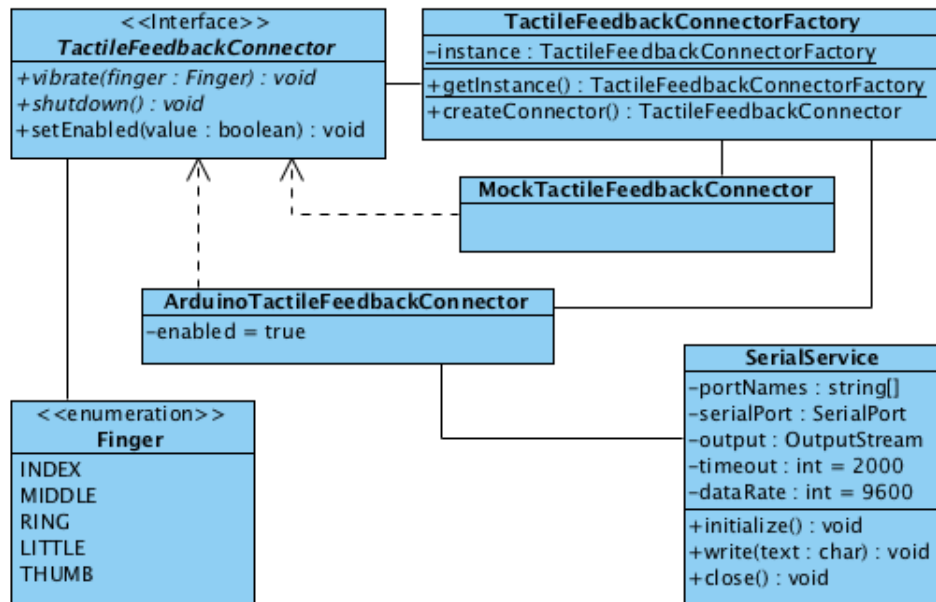


Figura 3.20: Diagrama de classes do Tactile Feedback Connector

A interface `TactileFeedbackConnector` define as operações que podem ser executadas por um conector de feedback tátil. A classe `ArduinoTactileFeedbackConnector` é a implementação de referência que suporta os componentes LilyPad Arduino, enquanto a classe `MockTactileFeedbackConnector` é utilizada para testes internos. A classe `TactileFeedbackConnectorFactory` é responsável pela criação de instâncias de conectores de feedback tátil. Já a classe `SerialService` representa um serviço para comunicação via porta serial que é utilizado pela classe `ArduinoTactileFeedbackConnector` na comunicação com a luva. A enumeração `Finger` representa os dedos da mão suportados pelo software.

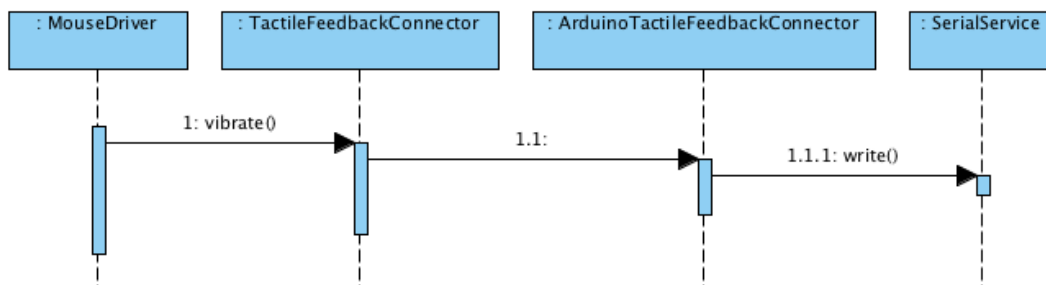


Figura 3.21: Diagrama de sequência da operação de acionar o feedback tátil

A Figura 3.21 mostra o diagrama de sequência da operação de acionar o feedback tátil na v-Glove. Ao identificar que o usuário se aproximou da interface de toque virtual, a classe `MouseDriver` dispara um evento de vibração através do

TactileFeedbackConnector, que por sua vez envia o comando de vibração para o SerialService, indicando qual dedo deve ser acionado. O SerialService então envia essa mensagem via conexão serial para o XBee presente na estação de controle, que repassa o comando para a v-Glove.

3.7.5. Configuration Manager

Este módulo é responsável por gerenciar os parâmetros de configuração necessários para a execução do software. O Configuration Manager pode ser acessado por qualquer classe do programa que precise fazer a leitura de um parâmetro de configuração.