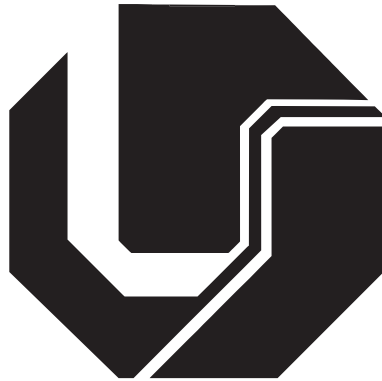


**UNIVERSIDADE FEDERAL DE UBERLÂNDIA
FACULDADE DE ENGENHARIA ELÉTRICA
PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA**



**Manipulação de Objetos 3D em Aplicações de
Realidade Aumentada por Meio da Movimentação da
Mão**

Eduardo de Souza Santos

Julho

2011

Manipulação de Objetos 3D em Aplicações de Realidade Aumentada por Meio da Movimentação da Mão

Eduardo de Souza Santos

Texto apresentado à Universidade Federal de Uberlândia como parte dos requisitos para obtenção do título de Mestre em Ciências.

Prof. Edgard A. Lamounier Junior, PhD.
Orientador

Alexandre Cardoso, Dr.
Co-orientador

Prof. Alexandre Cardoso, Dr.
Coordenador do curso de Pós-Graduação

**UNIVERSIDADE FEDERAL DE UBERLÂNDIA
FACULDADE DE ENGENHARIA ELÉTRICA
PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA**

**Manipulação de Objetos 3D em Aplicações de
Realidade Aumentada por Meio da Movimentação da
Mão**

Eduardo de Souza Santos

Texto apresentado à Universidade Federal de Uberlândia, perante a banca de examinadores abaixo, como parte dos requisitos necessários para a obtenção do título de Mestre em Ciências.

Banca Examinadora:

Prof. Edgard A. Lamounier Junior, PhD- Orientador (UFU)

Prof. Alexandre Cardoso, Dr- Co-orientador (UFU)

Prof. Augusto W. Fleury Veloso da Silveira, Dr. (UFU)

Prof. José Remo Ferreira Brega, Livre-docente (UNESP)

Agradecimentos

Chegar até aqui não foi um processo fácil. O caminho para alcançar esse ponto da pesquisa de mestrado, foi longo e árduo. Mas graças a algumas pessoas e circunstâncias, esse processo tornou-se ameno e rico em descobertas. Pessoas essas que foram fundamentais para a minha formação tanto pessoal, quanto profissional. Pessoas essas as quais algumas eu devo a vida, dom básico para a existência humana, e pessoas as quais devo os meus conhecimentos. E outras pessoas que devo o deleite da convivência no aprimoramento do conceito de humanidade.

Primeiramente quero agradecer a Deus e a Jesus, que me deram força para construir um caminho de buscas e conquistas de vitórias e conhecimentos acadêmicos.

Quero agradecer a minha Mãe que me deu conceitos básicos de civilidade, conceitos básicos dos valores éticos primordiais, mas principalmente quero agradecer com especial reverência a pessoa do meu avô José. Figura engraçada, amiga, que se tornou meu espelho de homem. Na figura dele quero agradecer todos os membros da minha consanguinidade.

De maneira bem clara e honesta quero agradecer ao meu orientador Edgard Lammounier, que junto com o professor Alexandre Cardoso, deram-me a oportunidade de aprofundar meu saber e lapidar minhas aptidões.

Gostaria de agradecer aos meus companheiros e amigos do Laboratório da Computação Gráfica, que fizeram com que as horas de dedicação fossem aliviadas com palavras de estímulo e risadas.

Queria agradecer a minha noiva Pricilla, pelo incentivo e motivação de sempre.

E ficando a certeza de que mais uma etapa foi cumprida, e já pronto para outras que virão, quero agradecer os meus passos trilhados para que eu chegasse até aqui.

Obrigado a todos vocês.

Resumo

A necessidade de tornar o acesso e a manipulação de informações digitais mais intuitivo tem feito com que novas tecnologias surjam. Neste aspecto pode-se destacar o crescimento de aplicações de Realidade Virtual e Aumentada, as quais passaram a demandar melhorias das metodologias de interação utilizadas.

Dessa forma, este trabalho tem o objetivo de investigar metodologias de interação e desenvolver uma aplicação de Realidade Aumentada que fizesse uso de uma forma mais intuitiva de manipulação de objetos virtuais.

Com a realização do estudo do estado da arte, percebeu-se que a utilização das mãos, através do posicionamento e gestos, consiste em uma interface intuitiva e cognitivamente vantajosa para utilização.

Com essas informações, foi determinado que a aplicação a ser desenvolvida deveria apresentar como recursos a possibilidade de interagir com os objetos virtuais utilizando as mãos sem a necessidade de utilizar qualquer aparato acoplado ao corpo, levando em conta, as necessidades de ser compatível com diversas cores de pele e de garantir a utilização em ambientes não controlados, ou seja, com fundos complexos (contendo pessoas e objetos diversos) e com pouca dependência de iluminação.

Para tal, foi utilizado o dispositivo Kinect, juntamente com o *framework* OpenNI e o *middleware* NITE, os quais consistem em módulos de *software* de uso específico para aplicações com interação via gestos. A exibição da imagem aumentada foi realizada através da implementação de um módulo de tratamento de eventos, atuação e renderização, que utilizou como motor de renderização a biblioteca Ogre3D.

Por fim, discutiu-se o desenvolvimento da aplicação com os requisitos propostos, a qual tem como função permitir a interação em um ambiente de RA simplificado, que permite a criação de objetos virtuais e sua manipulação (rotação, escala, translação e deleção) através de gestos pré-definidos e menus.

Palavras-chave: Realidade Aumentada. Interação natural. Kinect. Interação sem aparatos.

Abstract

The need to make access and manipulation of digital information has made more intuitive with which new technologies emerge. In this respect we can highlight the growth of applications of Virtual and Augmented Reality, which began to demand improvements in methods of interaction used.

Thus, this study aims to investigate methods of interaction and develop an augmented reality application that makes use of a more intuitive way of manipulating virtual objects.

With the study of the state of the art, it was noted that the use of hands, by positioning and gestures, consists of an intuitive interface and cognitively advantageous for use.

Using this information, it was determined that the application to be developed as resources should provide the possibility to interact with virtual objects using their hands without using any apparatus attached to the body, taking into account needs to be compatible with different skin colors skin to ensure use in environments with complex backgrounds (containing various objects and people) with little dependence on illumination.

For this, we used the Kinect device together with the OpenNI framework and NITE middleware, which consist of software modules for specific applications with interaction via gestures. The display of the augmented image was carried through the implementation of a module event handling, and rendering operations, which used as a rendering engine the Ogre3D library.

Finally, we discussed the development of the application with the proposed requirements, which is to allow interaction in a simplified environment of AR, which allows the creation of virtual objects and their manipulation (rotation, scaling, translation and deletion) via pre-defined gestures and menus.

Keywords: Augmented Reality. Natural Interaction. Kinect. Bare Hand Interaction.

Publicações

A seguir são apresentadas as publicações resultantes desse trabalho:

Santos, E. S.; Lamounier, E.; Cardoso, A. **Interação em Ambientes de Realidade Aumentada Utilizando o Dispositivo Kinect**. In: SVR 2011 - XIII Symposium on Virtual and Augmented Reality, 2011, Uberlândia, MG, Brazil.

Santos, E. S. ; Silva, M. D.; Arantes, W. V.; Lima, L.; Cardoso, A.; Lamounier, E. **Proposta de uma Interface não Convencional para Reabilitação Motora Suportada por Ambientes de Realidade Virtual**. In: 6º Workshop de Realidade Virtual e Aumentada, 2009, Santos. WRVA, 2009.

Arantes, W. V.; Silva, M. D.; Santos, E. S.; Oliveira, M. R.; Yamanaka, K.; Cardoso, A.; Lamounier, E. **Proposta de uma Interface não Convencional Baseada em Processamento de Imagens e Redes Neurais Artificiais**. In: IX Congresso Brasileiro de Redes Neurais/Inteligência Computacional (IX CBRN), 2009, Ouro Preto, MG. Anais do IX Congresso Brasileiro de Redes Neurais/Inteligência Computacional (IX CBRN), 2009.

Sumário

Lista de Figuras	vi
Lista de Tabelas	ix
Lista de Abreviaturas	x
1 Introdução	1
1.1 Contextualização	1
1.2 Motivação	2
1.3 Objetivo	3
1.4 Organização da dissertação	3
2 Fundamentos de Realidade Aumentada	5
2.1 Introdução	5
2.2 Campos de Aplicação da Realidade Aumentada	6
2.2.1 Medicina	6
2.2.2 Entretenimento e Publicidade	7
2.2.3 Aplicações Militares	9
2.2.4 Manutenção e Reparo de Equipamentos	9
2.3 Tecnologias para Visualização de Ambientes de Realidade Aumentada	11
2.3.1 Visualização Baseada em Monitor	11
2.3.2 Visualização com <i>Head Mounted Displays</i>	13
2.4 Interação em Ambientes de Realidade Aumentada	15
2.5 Bibliotecas para Desenvolvimento de Aplicações de Realidade Aumentada	18

2.5.1	ARToolKit	18
2.5.2	ARTag	19
2.5.3	FLARToolKit	20
2.6	Considerações Finais	21
3	Estado da Arte	22
3.1	Introdução	22
3.2	Metodologia para a Análise	22
3.3	Trabalhos Relacionados	23
3.3.1	Finger Tracking for Interaction in Augmented Environments	23
3.3.2	Assembly Design and Evaluation based on Bare-Hand Interaction in an Augmented Reality Environment	25
3.3.3	Manipulation of Virtual Objects in Marker-less AR System by Fingertip Tracking and Hand Gesture Recognition	27
3.3.4	FingARtips: Gesture Based Direct Manipulation in Augmented Reality	29
3.3.5	Opportunistic Tangible User Interfaces for Augmented Reality	32
3.3.6	Using Haar-like Feature Classifiers for Hand Tracking in Tabletop Augmented Reality	34
3.4	Considerações Finais	35
4	Análise e Especificação do Sistema	39
4.1	Introdução	39
4.2	Especificação do Sistema	39
4.2.1	Requisitos do Sistema	39
4.2.2	Diagrama de Caso de Uso do Sistema	40
4.2.3	Diagrama de Atividade do Sistema	43
4.2.4	Arquitetura do Sistema	45
4.3	Considerações Finais	47

5	Detalhes de Implementação	48
5.1	Introdução	48
5.2	Tecnologias de Apoio	48
5.2.1	Hardware	48
5.2.2	Software	51
5.3	Ambiente de Desenvolvimento	54
5.3.1	Interface Gráfica	56
5.4	Comunicação com o Sensor e Detecção de Gestos	58
5.5	Interação com o Usuário	64
5.5.1	Navegação em Menus	64
5.5.2	Seleção de Objetos Virtuais	66
5.5.3	Translação, Rotação e Escala	67
5.6	Considerações Finais	68
6	Análise dos Resultados	69
6.1	Introdução	69
6.2	Escolha dos Módulos do Sistema	69
6.3	Detecção da Mão Nua e Compatibilidade com Múltiplas Cores de Pele .	71
6.4	Suporte a Variações de Luz e Independência do Ambiente de Trabalho	72
6.5	Determinação do Gesto Realizado	74
6.6	Atuação no Ambiente de RA e Visualização da Cena Aumentada	75
6.7	Considerações Finais	76
7	Conclusões e Trabalhos Futuros	79
7.1	Introdução	79
7.2	Conclusões	79
7.3	Trabalhos Futuros	80

Lista de Figuras

2.1	Aplicação de Realidade Aumentada na medicina proposta por (PALOC et al., 2004).	7
2.2	Simulação de próteses de membros superiores utilizando RA (NOGUEIRA, 2007).	7
2.3	Resultados obtidos na utilização da aplicação “Jogo de Palavras” (ZORZAL et al., 2006).	8
2.4	Aplicação de RA - Campanha BMW (BMW, 2010).	8
2.5	Aplicação de RA - Campanha Portuguesa (Portuguesa de Desportos, 2010).	9
2.6	Ilustração do protocolo para visualização de objetos oclusos (LIVINGSTON et al., 2002).	10
2.7	Processo de manutenção auxiliado por RA (HENDERSON; FEINER, 2009).	10
2.8	Sistema de Realidade Aumentada baseada em monitor, adaptado de (VALLINO, 1998).	12
2.9	Exemplo de configuração espelho para sistemas de RA.	12
2.10	Esquema de HMD que utiliza a tecnologia “ <i>video see-through</i> ”, adaptado de (VALLINO, 1998).	13
2.11	Exemplo de HMD com a tecnologia “ <i>video see-through</i> ”, adaptado de (BROLL et al., 2004).	14
2.12	Esquema de HMD que utiliza a tecnologia “ <i>optical see-through</i> ”, adaptado de (VALLINO, 1998).	14
2.13	Exemplo de HMD a tecnologia “ <i>optical see-through</i> ” (NVIS Inc., 2010).	15
2.14	Exemplo de interação espacial em ambiente de RA (FISCHER; BARTZ; STRASSER, 2005).	16
2.15	Exemplo de interação baseada em comandos (SEO et al., 2008).	16

2.16 Exemplo de interação por controle virtual (BROLL et al., 2004).	17
2.17 Exemplo de aplicação de Realidade Aumentada utilizando ARToolKit (KATO et al., 2010).	18
2.18 Exemplos de aplicações de Realidade Aumentada utilizando ARTag (FIALA, 2010).	19
3.1 Luva com marcadores retrorreflexivos (DORFMÜLLER-ULHAAS; SCHMALSTIEG, 2001).	23
3.2 Experimento com Realidade Aumentada mostrado em (DORFMÜLLER-ULHAAS; SCHMALSTIEG, 2001).	24
3.3 Plataforma de montagem da aplicação de RA com interação através das mãos (Adaptada de (WANG et al., 2009)).	25
3.4 Segmentação da mão e determinação das pontas dos dedos (WANG et al., 2009).	26
3.5 Interação com o ambiente de RA (WANG et al., 2009).	27
3.6 Fases do sistema proposto em (LEE; CHUN, 2009) (adaptado de (LEE; CHUN, 2009)).	27
3.7 Correlação entre a imagem da mão e o padrão quadriculado (a). Sistema de coordenadas da mão do usuário (b) (LEE; CHUN, 2009).	28
3.8 Gestos cadastrados no experimento de (LEE; CHUN, 2009) (rotação, escala e translação, respectivamente).	29
3.9 Ambiente de trabalho da aplicação utilizada em (BUCHMANN et al., 2004). . .	30
3.10 Luva para interação proposta em (BUCHMANN et al., 2004).	30
3.11 Modelo simplificado da mão (BUCHMANN et al., 2004).	31
3.12 Solução para oclusão da mão por marcadores de (BUCHMANN et al., 2004). .	31
3.13 Exemplo de utilização de um OC (HENDERSON; FEINER, 2010).	33
3.14 Ambiente de trabalho de (FERNANDES; FERNÁNDEZ, 2010) e movimentação de objeto virtual.	34
3.15 Operações de rotação e escala mostradas em (FERNANDES; FERNÁNDEZ, 2010).	35

4.1	Diagrama de caso do uso para o sistema de interação com RA.	41
4.2	Diagrama de atividade para a interação.	44
4.3	Diagrama de atividade para a renderização.	45
4.4	Arquitetura do sistema.	46
5.1	Abstração das camadas do framework OpenNI (adaptado de (OPENNI, 2011)).	52
5.2	Exemplo de configuração baseada em monitor para o sistema de RA proposto.	56
5.3	Elementos da interface gráfica (painéis e objeto 3D).	57
5.4	Elementos da interface gráfica (menu).	58
5.5	Movimentação para inicialização da posição da mão (<i>wave</i>).	64
5.6	Indicação da detecção iniciada.	65
5.7	Movimentação para inicialização de menus (círculo).	65
5.8	Navegação no menu.	66
5.9	Objeto selecionado e menu de operações.	66
5.10	Procedimento de translação.	67
5.11	Procedimento de rotação.	67
5.12	Procedimento de escala.	68
6.1	Exemplo de reconhecimento prévio de usuários.	71
6.2	Amostras de testes com pessoas de cores de peles distintas.	72
6.3	Amostras de testes com variação de luminosidade.	73
6.4	Amostras de testes em ambientes complexos.	74
6.5	Usuário utilizando o sistema.	75
6.6	Níveis de reconhecimento - ambiente claro.	77
6.7	Níveis de reconhecimento - ambiente escuro.	77
6.8	Nível de satisfação do usuário.	77

Lista de Tabelas

3.1	Comparação entre o <i>hardware</i> necessário, o espaço de cores e a <i>biblioteca</i> de RA utilizada nos trabalhos relacionados.	36
3.2	Comparação das características analisadas nos trabalhos relacionados.	38
4.1	Descrição do caso de uso “Interagir com Menus”.	41
4.2	Descrição do caso de uso “Criar Objetos 3D”.	42
4.3	Descrição do caso de uso “Manipular Objetos 3D”.	42
6.1	Quantidade de pessoas que participaram dos testes e suas respectivas cores de pele.	76
6.2	Comparação das características analisadas nos trabalhos relacionados.	78

Lista de Abreviaturas

AS3 ActionScript 3.0

BARS Battlefield Augmented Reality System

CEGUI Crazy Eddies GUI System

CIE Comission International de l'Éclairage

FPS Frames per Second

GPS Global Positioning System

HCI Human-Computer Interaction

HIT Lab Human Interface Technology Laboratory

HMD Head Mounted Display

HSI Hue, Saturation, Intensity

IHM Interação Homem-Máquina

IR Infra Red

MFC Microsoft Foundation Classes

OC Opportunistic Controls

OGRE Object-Oriented Graphics Rendering Engine

OpenNI Open Natural Interaction

OST Optical See-Through

RA Realidade Aumentada

RdP Redes de Petri

RGB Red, Green, Blue

RIA Rich Internet Applications

RV Realidade Virtual

SO Sistema Operacional

UML Unified Modeling Language

USAR Urban Search and Rescue

USB Universal Serial Bus

VST Video See-Through

1 *Introdução*

1.1 Contextualização

O início da era computacional foi marcado pela necessidade de se obter um processamento preciso de dados, por este motivo não se deu importância ao processo de Interação Homem-Máquina (IHM), utilizando-se equipamentos de entrada e saída pouco intuitivos. Um exemplo clássico, consiste na leitura de dados/comandos em cartões perfurados, introduzida na máquina de processar dados de Herman Hollerith em 1890, sendo utilizada também no dispositivo que é considerado o primeiro computador eletro-mecânico, o qual foi construído por Konrad Zuze em 1936 (FILHO, 2007).

Com a evolução e disseminação dos computadores pessoais a demanda por metodologias específicas para IHM teve um aumento significativo, visto que os usuários passaram a necessitar de interfaces que maximizassem sua produtividade, sendo mais intuitivas e/ou de fácil manipulação (PREECE et al., 1994).

Nessa busca de uma interface de operação ideal, que satisfaça completamente os atuais requisitos de usabilidade e naturalidade de operação, surgiram diversas tecnologias, dentre elas, pode-se destacar a Realidade Virtual e a Realidade Aumentada (TORI; KIRNER; SISCOOTTO, 2006).

A Realidade Virtual (RV) é comumente definida como uma “interface avançada do usuário”, sendo caracterizada pela imersão e interação do usuário em ambientes virtuais tridimensionais em tempo real (TORI; KIRNER, 2006b).

A Realidade Aumentada (RA) pode ser definida como o enriquecimento do ambiente real com objetos virtuais, através de algum dispositivo tecnológico, em tempo real (TORI; KIRNER, 2006a).

De uma forma geral, tanto em aplicações de RV quanto de RA, o usuário necessita de realizar tarefas simples, como manipulação, criação e deleção de objetos virtuais. As quais são, muitas vezes, realizadas de formas não naturais, levando o utilizador

do ambiente a uma perda na sensação da imersão (BUCHMANN et al., 2004). Dessa forma, a investigação de metodologias para interação com estes ambientes parece ser um passo evolutivo na busca por interfaces mais próximas do usuários, sendo, por isso foco de vários trabalhos.

1.2 Motivação

Considerando o surgimento de novas tecnologias, como a RV e RA, a interação entre humanos e computadores ainda é tarefa difícil, principalmente, quando se considera o projeto de sistemas computacionais interativos, que sejam eficazes, eficientes, de fácil utilização e, além disso, possuam uma usabilidade agradável (DIX; FINLAY; ABOWD, 2004).

No sentido de tornar o acesso ou manipulação de informações digitais mais intuitivo, diversos trabalhos têm sido desenvolvidos. Em muitos destes, pode-se encontrar referências a uma modalidade de interação chamada *“natural interaction”*. Valli, por exemplo, define em (VALLI, 2008) este conceito como sendo uma forma de permitir a manipulação de um ambiente, em que a necessidade de aprendizado para a interação seja mínima, permitindo que a metáfora utilizada para interação seja tão simples de aprender que esta possa ser utilizada por qualquer um, através de um mínimo de treinamento.

Passando ao contexto da Realidade Aumentada, objetivo deste trabalho, Buchmann et al. (BUCHMANN et al., 2004) definem que as técnicas de interação para Realidade Aumentada devem ser tão intuitivas quanto possível, para que assim sejam bem aceitas pelos usuários finais. Portanto, devemos buscar meios de interação em que o usuário possa manipular o ambiente de RA de forma natural.

Considerando que as mãos são nosso meio de interação com os objetos reais a nossa volta, interfaces com RA deveriam também permitir esse tipo de artifício, que poderia ajudar muito na adaptação de atitudes já conhecidas na vivência diária para a interação com um objeto virtual (BUCHMANN et al., 2004). Tendo isso em vista, a motivação central deste trabalho consiste em contribuir para a criação de aplicações de Realidade Aumentada que tenham como característica uma metodologia de interação natural, ou seja, que permita que o usuário desempenhe suas funções através de uma interface mais intuitiva.

1.3 Objetivo

Neste trabalho pretende-se investigar técnicas de interação em Sistemas de Realidade Aumentada com o objetivo de desenvolver uma aplicação que permita a manipulação de objetos virtuais, através de gestos e do posicionamento da mão no espaço tridimensional, possibilitando assim efetuar operações de translação, rotação e escala sem a necessidade de acoplar qualquer aparato no corpo do usuário e que, além disso, não necessite de etapas de calibração ou de grandes restrições no ambiente de trabalho.

Para isso, foram definidas as seguintes metas:

- Pesquisar aplicações que realizem interação com ambientes de RA através de gestos com as mãos;
- Estudar as metodologias para reconhecimento de gestos e/ou mãos no espaço;
- Pesquisar sobre aparatos de hardware disponíveis no mercado que possam ser úteis neste tipo de aplicação;
- Pesquisar acerca de bibliotecas e/ou softwares que possibilitem metodologias mais concisas de processamento de imagens;
- Projetar o software a ser desenvolvido levando em conta algumas restrições de projeto, como a necessidade de não utilizar nenhum equipamento acoplado ao corpo do usuário, ser robusto em relação a utilização em ambientes não controlados e com variação de iluminação, suportar a utilização por pessoas de diferentes biotipos e ser compatível com sistemas operacionais Windows e Linux;
- Implementar o protótipo respeitando o requisito de tempo real de aplicações de RA;
- Testar o sistema desenvolvido em condições adversas de iluminação e ambiente de trabalho e com pessoas de características físicas distintas.

1.4 Organização da dissertação

Visando descrever o trabalho realizado, esta dissertação foi dividida da seguinte forma:

No Capítulo 1, são apresentadas as considerações iniciais sobre interfaces humano-computador, motivação, objetivos e a estrutura do trabalho.

No Capítulo 2, são tratados os fundamentos de Realidade Aumentada, assim como o campo de aplicação desse tipo de sistema. São discutidas ainda as tecnologias utilizadas para a visualização destes ambientes, assim como as formas de interação e as bibliotecas utilizadas.

O Capítulo 3 aborda o estado da arte no âmbito da interação em aplicações de RA, discutindo características, vantagens e desvantagens de sistemas com estas características.

No Capítulo 4, é discutida a análise e especificação do sistema através da apresentação de diagramas UML e da arquitetura do sistema.

O Capítulo 5 conta com detalhes e implementação, permitindo o entendimento das ferramentas de software e hardware utilizadas no desenvolvimento. Além disso, é discutido o ambiente de desenvolvimento e a metodologia para interação com o usuário.

No Capítulo 6 são tratados os resultados obtidos, tais como utilização em diferentes ambientes com fundos simples e complexos, variação na iluminação e biotipo do usuário.

Por fim, o Capítulo 7 discute as conclusões e trabalhos futuros.

2 *Fundamentos de Realidade Aumentada*

2.1 Introdução

O desenvolvimento das técnicas de multimídia, alavancado pelo rápido avanço da capacidade de processamento e disseminação dos computadores, proporcionou a integração de ambientes virtuais interativos com *streams* de vídeo, em tempo real, tornando viável o avanço de técnicas de Realidade Aumentada (RA) tanto em plataformas mais robustas quanto em outras com baixas taxas de processamento e memória disponível (TORI; KIRNER; SISCOOTTO, 2006).

Segundo (KIRNER; SISCOOTTO, 2007) a Realidade Aumentada pode ser definida de várias formas, dentre estas, pode-se destacar:

- É o enriquecimento do ambiente real com objetos virtuais, a partir da utilização de ferramentas computacionais que permitam essa execução em tempo real (KIRNER; SISCOOTTO, 2007);
- Consiste na melhoria do mundo real a partir da adição de textos, imagens e objetos virtuais com o auxílio do computador (INSLEY, 2003);
- É, de acordo com (AZUMA et al., 2001), um sistema que acrescenta ao mundo real objetos virtuais, que parecem coexistir no mesmo espaço e possui as seguintes propriedades:
 - faz a união de objetos reais e virtuais no ambiente real;
 - é executado em tempo real;
 - deve alinhar (inter-relacionar) objetos reais e virtuais.

Dessa forma é possível observar que essencialmente, uma aplicação de RA deve unir elementos virtuais a o ambiente real, satisfazendo o requisito de realizar essas

operações em tempo real. Adicionalmente, pode-se utilizar dispositivos específicos para a exibição do ambiente misturado como HMDs (*Head Mounted Displays*), não esquecendo da possibilidade de utilizar outros sentidos, como audição, tato e cheiro (AZUMA et al., 2001).

No sentido de prover ao leitor os conceitos básicos utilizados no desenvolvimento do protótipo, bem como das possibilidades de aplicação de sistemas de RA, as sessões a seguir discorrem acerca de diversas definições relativas à RA, que consistem desde campos de aplicações de RA até tecnologias utilizadas para a criação de ambientes aumentados.

2.2 Campos de Aplicação da Realidade Aumentada

O avanço das tecnologias de processamento de vídeo em tempo real e dos sistemas de computação gráfica, aliado a o desenvolvimento de novos sistemas de visualização, como os HMDs, convergiram para a possibilidade de criação de ambientes de Realidade Aumentada mais robustos. Com isso, passou-se a criar soluções para diversas áreas do conhecimento, logo, esta sessão tem como objetivo destacar algumas dessas aplicações.

2.2.1 Medicina

Como as tecnologias que utilizam imagens já são bastante incorporadas no campo da medicina, a utilização de técnicas de Realidade Aumentada não foi surpresa, visto que, a maioria das aplicações médicas trabalham com algum tipo de procedimento guiado por imagens, possibilitando, por exemplo, visualizar a anatomia interna do paciente.

Um exemplo de uma aplicação de RA utilizada em um sistema de cirurgia assistida por computador é abordado no trabalho mostrado em (PALOC et al., 2004). Neste, os autores propõem um sistema de RA para a visualização do fígado do paciente como o objetivo de auxiliar na cirurgia hepática, focando a necessidade de prover um desconforto mínimo para o cirurgião e o paciente. A Figura 2.1 mostra a visualização da RA para o referido trabalho. Nela pode-se observar a imagem real do paciente sendo enriquecida com o fígado virtual.

Existem ainda outras aplicações utilizadas na reabilitação de pacientes, como o



Figura 2.1: Aplicação de Realidade Aumentada na medicina proposta por (PALOC et al., 2004).

trabalho de (NOGUEIRA, 2007) que discorre sobre a utilização de técnicas de Realidade Virtual e Aumentada para a simulação de próteses de membros superiores, a qual objetiva o treinamento do paciente amputado com uma prótese virtual antes da inserção de prótese real, objetivando diminuir o impacto da adaptação do mesmo com o dispositivo que será acoplado em seu corpo. A Figura 2.2 mostra a simulação de um treinamento de flexão de braço com este sistema. Observa-se que a utilização de RA permite que o paciente visualize uma prótese virtual, que poderá ser comandada por sinais elétricos captados nos músculos do usuário.



Figura 2.2: Simulação de próteses de membros superiores utilizando RA (NOGUEIRA, 2007).

2.2.2 Entretenimento e Publicidade

Tecnologias de RA têm sido cada vez mais utilizadas em aplicações de entretenimento e campanhas publicitárias. Dentre essas aplicações podemos destacar os jogos, sejam eles educativos ou não, que fascinam pela utilização de técnicas 3D e interfaces mais intuitivas.

Um exemplo de aplicação consiste no “Jogo de Palavras” mostrado no trabalho de (ZORZAL et al., 2006). Neste jogo o objetivo é formar palavras. Para tanto, o sistema resgata a imagem relacionada a palavra e mostra o modelo 3D para o usuário, uma imagem exemplificando o funcionamento deste sistema é mostrada na Figura 2.3.



Figura 2.3: Resultados obtidos na utilização da aplicação “Jogo de Palavras” (ZORZAL et al., 2006).

Aplicações em publicidade são muito comuns e podem ser encontradas em jogos de futebol e automobilismo, nos quais são posicionados marcadores em locais estratégicos onde são, durante a partida, mostrados conteúdos de propaganda, como slogans. Nesse ponto, existem ainda campanhas de montadoras de automóveis que tem criado aplicações de RA agregadas ao lançamento de novos modelos, como é o caso do exemplo mostrado na Figura 2.4, que corresponde a uma campanha realizada pela BMW, na qual pode-se, por meio da utilização de uma câmera e de um marcador, visualizar os detalhes do novo modelo, bem como controlá-lo no ambiente real.



Figura 2.4: Aplicação de RA - Campanha BMW (BMW, 2010).

Com a evolução das aplicações de RA, a utilização desse tipo de campanhas se expandiu de forma massiva para as páginas web. Dentre essas aplicações, pode-se citar a pioneira campanha da Portuguesa de Desportos, um time do estado de São Paulo, que lançou seu terceiro uniforme com um recurso inédito entre as equipes do Brasil: um marcador fiducial presente na manga esquerda da camisa (Figura 2.5),

permite o acesso a uma promoção do site “Tudo sobre a Lusa”, mantido pela equipe de futebol.



Figura 2.5: Aplicação de RA - Campanha Portuguesa (Portuguesa de Desportos, 2010).

2.2.3 Aplicações Militares

A Realidade Aumentada também vem sendo bastante utilizada em aplicações militares. Dentre estas, um exemplo clássico consiste na apresentação de informações no para-brisa de aviões ou no visor dos capacetes.

Existem também aplicações para posicionamento de tropas e de auxílio em operações militares, como é o caso do trabalho em (LIVINGSTON et al., 2002), que utiliza o sistema BARS (*Battlefield Augmented Reality System*), o qual consiste de um computador, que é carregado em uma mochila, um sistema para rede sem fio e um HMD rastreado em tempo real com o auxílio de um GPS.

Com este aparato (LIVINGSTON et al., 2002) propõe a utilização de RA em operações militares em terreno urbano, visto que este ambiente apresenta condições bastante desafiadoras para soldados por ser extremamente complexo e permitir oclusão de objetos e/ou inimigos. A Figura 2.6 mostra um exemplo de protocolo de exibição de objetos ocultos, o bloco em vermelho indica a posição de algum elemento de interesse que esteja oculto, o qual poderia ser constituído por informações de túneis, minas terrestres e, até mesmo, a localização de um batalhão aliado, já que o sistema é colaborativo e, portanto, pode-se ter a visualização de outros soldados com o mesmo equipamento.



Figura 2.6: Ilustração do protocolo para visualização de objetos oclusos (LIVINGSTON et al., 2002).

2.2.4 Manutenção e Reparo de Equipamentos

Quando um técnico em manutenção se depara com uma série de peças e equipamentos que não são familiares a ele, o procedimento a se realizar seria buscar uma série de manuais. Neste caso, não é difícil imaginar que os manuais podem não estar acessíveis ou em boas condições e mais, tal leitura pode levar um tempo prejudicial aos utilizadores do equipamento.

Neste ponto, aplicações de Realidade Aumentada podem ser extremamente úteis, visto que, é possível mostrar desde o procedimento a se realizar até anotações, como nome das peças e dicas operacionais. O sistema de RA poderia atuar de forma a mostrar ao usuário onde encaixar uma determinada peça e até mesmo status da máquina em um cenário de testes.

A Figura 2.7 mostra um exemplo de manutenção auxiliada por RA, observa-se que o sistema provê ao usuário informações sobre o tipo de ferramenta a ser utilizada e o local da atuação, bem como informações textuais como o nome do equipamento.

Observa-se então, que a área de manutenção e até mesmo montagem de equipamentos consiste em um campo em que a RA pode ser bastante explorada, visto que, com o auxílio desta tecnologia, além de permitir uma execução mais rápida de tarefas já discriminadas no manual, ainda pode-se contar com os diversos benefícios da utilização de objetos virtuais, como a visualização de partes escondidas e de características complexas de equipamento em questão (SCHWALD; LAVAL, 2003).



Figura 2.7: Processo de manutenção auxiliado por RA (HENDERSON; FEINER, 2009).

2.3 Tecnologias para Visualização de Ambientes de Realidade Aumentada

De forma geral, os sistemas para visualização de ambientes de RA atuam na coleta de informações do mundo real e na tecnologia utilizada para a exibição dos elementos virtuais no ambiente real.

Estes dois pontos são muito importantes, visto que, o grau de imersão do usuário é diretamente dependente de tecnologia de exibição e indiretamente dependente do artifício utilizado para a coleta de informações (MCDONALD, 2003). Isso acontece, em primeiro lugar, porque a forma de visualização não deve provocar perturbações entre o ambiente misturado que é mostrado para o usuário e o ambiente real, sob pena da diminuição da imersão e, em se tratando da coleta de dados, uma aquisição lenta ou imprecisa, pode causar imperfeições e/ou atrasos, os quais viriam a interferir negativamente no nível de imersão (MCDONALD, 2003).

Dando enfoque a estes aspectos as sessões a seguir mostram duas classes destes tipos de dispositivos, que são ordenados com base no nível de imersão do usuário.

2.3.1 Visualização Baseada em Monitor

Neste tipo de sistema um monitor é utilizado para mostrar a cena com elementos virtuais. Uma câmera captura uma sequência de vídeo da cena real enquanto sua posição e orientação estão sendo monitoradas. Com base nessas informações, os objetos virtuais são unidos, na posição correta, ao vídeo capturado. A Figura 2.8

ilustra o processo realizado. Observa-se que a partir da captura das imagens do mundo real e do posicionamento câmera, o sistema gráfico gera os objetos virtuais aplicando as transformações de escala, posição e rotação adequadas. Logo, estes objetos são unidos à imagem capturada gerando uma sequência aumentada que será mostrada para o usuário.

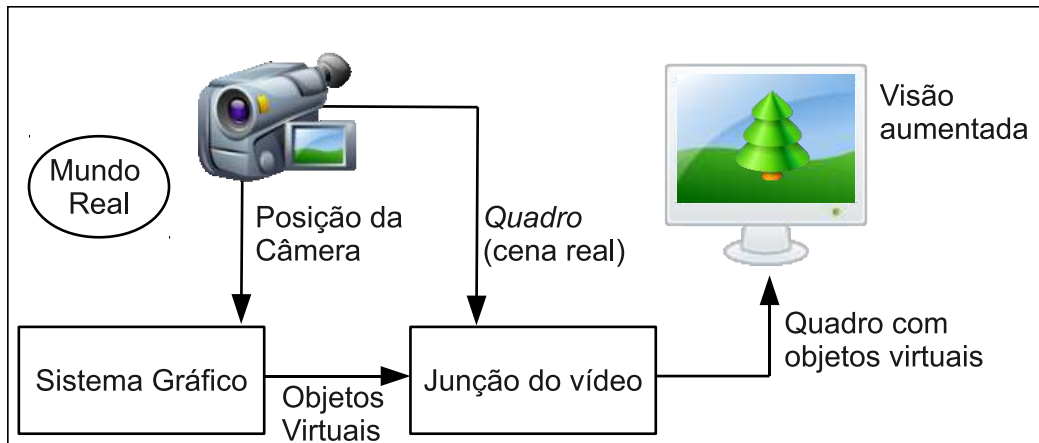


Figura 2.8: Sistema de Realidade Aumentada baseada em monitor, adaptado de (VALLINO, 1998).

Uma variação deste tipo de tecnologia consiste na configuração “espelho”, na qual a câmera e o monitor são orientadas para o usuário. Assim, este visualiza o sistema de RA como se estivesse vendo uma reflexão de sua imagem misturada com os objetos virtuais, conforme mostra a Figura 2.9.

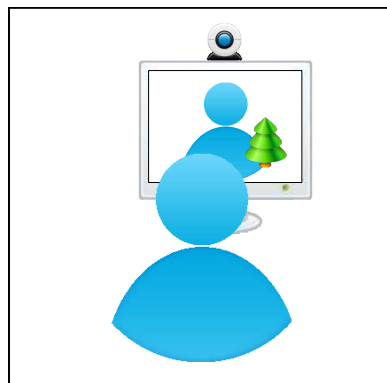


Figura 2.9: Exemplo de configuração espelho para sistemas de RA.

Apesar da utilização de monitores ser uma solução de baixo custo e, portanto, mais acessível, este tipo de sistema torna o usuário um observador de fora da cena, prejudicando a imersão.

Uma forma de diminuir essa perda, seria realizar o monitoramento da cabeça ou

posição do usuário e a exibição das imagens aumentadas diretamente para o usuário. Considerando este tipo de necessidade, vieram os *Head Mounted Displays*, que serão abordados em seguida.

2.3.2 Visualização com *Head Mounted Displays*

Devido à necessidade de aumentar a imersão do usuário outras tecnologias de visualização precisaram ser desenvolvidas e, como os HMDs já eram largamente utilizados em ambientes de Realidade Virtual eles acabaram por ser incorporados a aplicações de RA.

No entanto, diferentemente dos HMDs utilizados em sistemas de RV, que fornecem ao usuário um isolamento visual do ambiente real, os HMDs para RA possuem como requisito permitir a visualização do ambiente real. Com isso, estes equipamentos foram divididos em duas categorias chamadas “*video see-through*” (VST) e “*optical see-through*” (OST), que implementam diferentes formas de visualização do ambiente real.

Os HMDs da categoria VST possuem basicamente a mesma arquitetura dos sistemas baseados em vídeo, mas, como se deseja uma maior imersão, o *display* e a câmera estão acoplados próximos aos olhos do observador, como pode ser visto na Figura 2.10. A título de exemplo, a Figura 2.11 mostra um HMD com a tecnologia “*video see-through*”, nota-se que este conta com um rastreador, utilizado para determinar a posição da cabeça do usuário, duas câmeras para captura de imagens do mundo real e dois monitores responsáveis por mostrar as imagens misturadas com elementos virtuais.

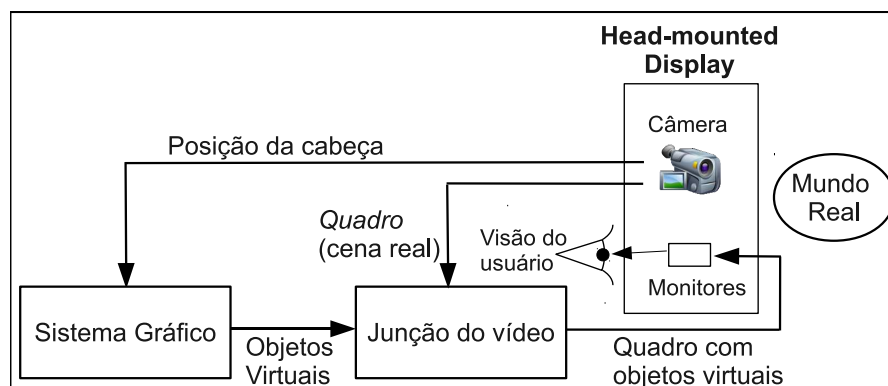


Figura 2.10: Esquema de HMD que utiliza a tecnologia “*video see-through*”, adaptado de (VALLINO, 1998).

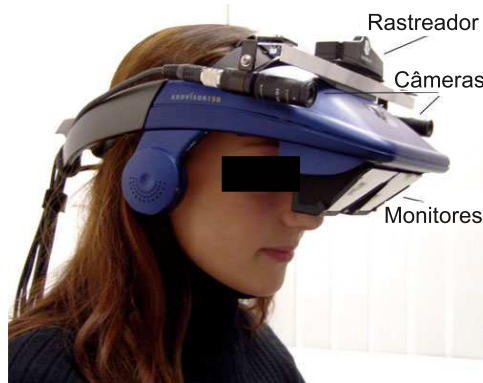


Figura 2.11: Exemplo de HMD com a tecnologia “*video see-through*”, adaptado de (BROLL et al., 2004).

Assim como nos sistemas baseados em monitor, os HMDs que utilizam a tecnologia “*video see-through*” precisam utilizar um artifício para unir dois canais de vídeo distintos: o capturado do mundo real e o que possui os elementos virtuais. Esta operação consiste em renderizar o elemento virtual quadro a quadro na imagem do mundo real mediante algum método de posicionamento. Isto se torna uma tarefa trivial a partir do momento em que bibliotecas de desenvolvimento adequadas são corretamente utilizadas.

Já os HMDs da categoria OST utilizam a cena real vista pelo usuário, como pode ser visualizado no diagrama mostrado na Figura 2.12. Dessa forma, o mundo real e os objetos virtuais são unidos através de combinadores óticos situados próximos aos olhos do usuário. A Figura 2.13 mostra um capacete o qual utiliza a tecnologia OST. Observa-se que as lentes para as quais o usuário olha são translúcidas, permitindo a visualização do mundo real e a projeção dos objetos virtuais na mesma.

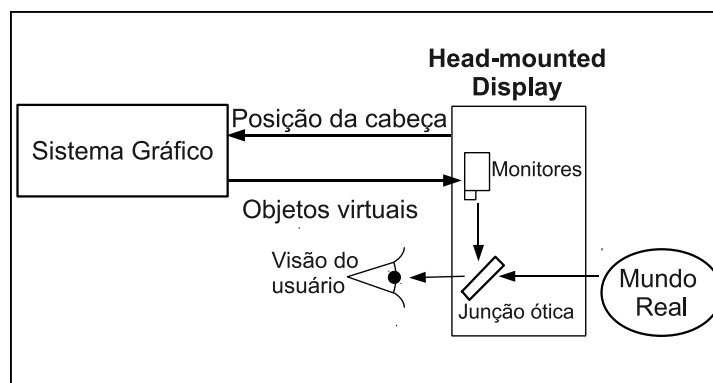


Figura 2.12: Esquema de HMD que utiliza a tecnologia “*optical see-through*”, adaptado de (VALLINO, 1998).

O trabalho mostrado em (AZUMA, 1997) cita diversas vantagens e desvantagens



Figura 2.13: Exemplo de HMD a tecnologia “*optical see-through*” (NVIS Inc., 2010).

das categorias de HMDs mostrados. No entanto, a escolha entre os tipos de tecnologia depende dos requisitos da aplicação, bem como dos custos, segurança e flexibilidade oferecidos.

2.4 Interação em Ambientes de Realidade Aumentada

Interação consiste na capacidade do usuário atuar no ambiente de forma a promover alterações e reações às suas ações e, em se tratando de aplicações de RA, pode trazer grandes ganhos de performance à medida que o usuário interage de forma mais intuitiva com o sistema (BUCHMANN et al., 2004).

Dessa forma, o trabalho abordado em (BROLL et al., 2005) provê uma classificação das técnicas de interação para ambientes de RA, as quais são subdivididas em:

- Espacial (*Spatial interaction*): tem como base a mudança das propriedades espaciais (rotação, translação e escala) de objetos físicos, geralmente, esse tipo de interação é realizada através de dispositivos apontadores, gestos de apontamento dinâmicos ou interfaces tangíveis ao usuário¹.

Um exemplo de interação espacial é mostrado na Figura 2.14, na qual a imagem do mundo real é enriquecida com linhas que representam cortes que poderiam ser realizados em uma intervenção cirúrgica. Com a utilização de um dispositivo que consiste em uma interface tangível, já que a interação com o computador é realizada com um objeto físico que aproxima o usuário do procedimento cirúrgico

¹Segundo (ISHII; ULLMER, 1997) as interfaces tangíveis ao usuário (TUIs - *Tangible User Interfaces*) consistem em um novo tipo de Interface Homem Máquina (IHM) em que o usuário não interage com o sistema computacional através de mouse e teclado, mas sim através de objetos físicos comuns ao mundo real, tornando a computação verdadeiramente ubíqua e invisível.

real.

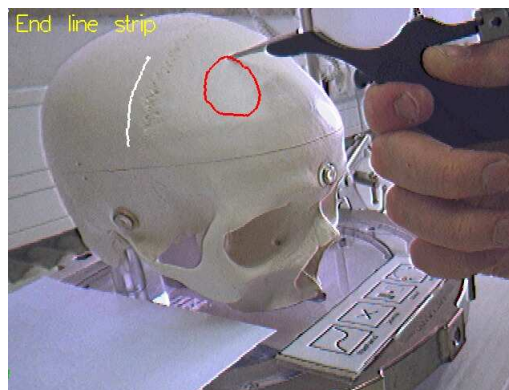


Figura 2.14: Exemplo de interação espacial em ambiente de RA (FISCHER; BARTZ; STRASSER, 2005).

- Baseada em comandos (*Command-based interaction*): é baseado em composições simples de entradas relacionadas com uma funcionalidade específica. Os casos mais comuns deste tipo de aplicação consistem em gestos estáticos e comandos de voz.

A aplicação desenvolvida em (SEO et al., 2008) (Figura 2.15), consiste em um exemplo deste tipo de interface. Nesta, a mão do usuário é utilizada para posicionar o objeto virtual no mundo real e para comandar o objeto, sendo que a mão aberta indica que a flor virtual deve estar aberta e fechada, que a mesma deve se fechar.



(a) Reação ao o gesto “mão aberta”.



(b) Reação ao gesto “mão fechada”.

Figura 2.15: Exemplo de interação baseada em comandos (SEO et al., 2008).

- Por controle virtual (*Virtual control interaction*): este tipo de interação é baseado na utilização de *widgets* 3D que são relacionados a uma função pré definida. A Figura 2.16 mostra um exemplo desse tipo de interface, que consiste em um

conjunto de widgets que representam os comando de um sistema de desenvolvimento de maquetes eletrônicas mostrado em (BROLL et al., 2004).

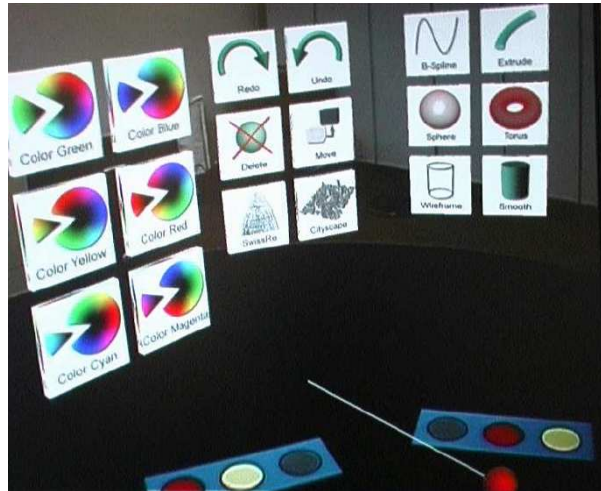


Figura 2.16: Exemplo de interação por controle virtual (BROLL et al., 2004).

- Por controle Físico (*Physical control interaction*): é baseado em ferramentas físicas ou painéis de controle que podem ser utilizados para controlar não somente objetos físicos como também objetos virtuais. Sua grande vantagem é a possibilidade de integração com ferramentas físicas na interface do usuário.

Tendo como base esta classificação, pode-se visualizar diversas modalidades e metodologias possíveis para a interação em ambientes de RA.

Considerando que a interação entre o usuário e o ambiente RA é realizada através da definição de uma metodologia de HCI (*Human-Computer Interaction*), é possível estender a visão da manipulação de uma aplicação de RA para um horizonte mais amplo.

Neste aspecto, pode-se encontrar diversos trabalhos que fazem menção à modalidade “*natural interaction*”, dentre estes, o trabalho de Valli em (VALLI, 2008) define este conceito como sendo uma forma de permitir a manipulação de um ambiente, em que a necessidade de aprendizado para a interação seja mínima, ou seja, a metáfora utilizada para interação deve ser tão simples que qualquer um conseguiria utilizar, necessitando de um mínimo de treinamento.

No mesmo âmbito, os trabalhos em (KRAHNSTOEVEER et al., 2002) e (YIN; DAVIS, 2010) apresentam aplicações que utilizam o conceito de interação natural. O trabalho mostrado em (KRAHNSTOEVEER et al., 2002), integra dados de vídeo e áudio, realizando o reconhecimento de gestos e de voz, que possibilitou a interação com

os protótipos desenvolvidos. Em (YIN; DAVIS, 2010) é apresentada uma aplicação intitulada “USAR” (*Urban Search and Rescue*), a qual permite o usuário o controle de um mapa 3D através da utilização de algumas classes de gestos.

Portanto, levando em conta as possibilidades da atuação em um ambiente de RA com uma metodologia que exija uma menor carga cognitiva e que permita uma maior imersão do usuário, através da realização de gestos e da manipulação direta (com as mãos), observa-se que este possa ser um bom caminho para a criação de aplicações de RA mais interativas e de fácil utilização nas diversas áreas do conhecimento.

2.5 Bibliotecas para Desenvolvimento de Aplicações de Realidade Aumentada

Em se tratando do desenvolvimento de aplicações de Realidade Aumentada, pode-se encontrar diversas bibliotecas que auxiliam no desenvolvimento deste tipo de ambiente. Por este motivo, esta seção tem como objetivo contemplar alguma das mais conhecidas dessas ferramentas.

2.5.1 ARToolKit

O ARToolKit (KATO et al., 2010) é uma biblioteca de software escrita em linguagem C, que foi desenvolvida inicialmente pelo Dr. Hirokazu Kato, na Universidade de Osaka e, desde então, passou a ser mantido pelo *Human Interface Technology Laboratory* (HIT Lab). Um exemplo clássico de utilização da biblioteca, consiste na imagem mostrada na Figura 2.17. Observa-se que é utilizado um marcador para posicionar o objeto virtual no ambiente real. Outro aspecto interessante da figura consiste na utilização do HMD, artifício este que permite que a experiência seja mais intuitiva e agradável para a usuária.

Esta biblioteca está disponível para utilização não comercial através da licença *GNU General Public License*² e também pode ser utilizada para fins comerciais através de contratos com a empresa *ARToolWorks Inc.*

Dentre suas principais características da versão não comercial estão o tracking para posicionamento e orientação de uma única câmera, a utilização de um marcador

²GNU GPL ou simplesmente GPL, é a designação da licença para software livre idealizada por Richard Stallman no final da década de 1980.



Figura 2.17: Exemplo de aplicação de Realidade Aumentada utilizando ARToolKit (KATO et al., 2010).

com bordas pretas, ser rápido para aplicações de RA em tempo real e possuir versões para as plataformas SGI IRIX, Linux, MacOS e Windows.

2.5.2 ARTag

Desenvolvido pelo *National Research Council of Canada*, a biblioteca ARTag assim como o ARToolKit utiliza marcadores fiduciais para posicionar o objeto virtual. No entanto, a identificação do marcador é realizada de forma diferente, visto que este é identificado de acordo com seu relacionamento com códigos digitais compostos de 0's e 1's.

Por este motivo, o marcador utilizado nesta biblioteca, além de possuir um retângulo de borda preta, conta com uma malha de dimensão 6x6 composta por quadrados brancos e pretos, representando os códigos digitais. A Figura 2.18 mostra objetos carregados com o ARTag.

Por ser inspirado no ARToolKit, os desenvolvedores do ARTag dedicaram certo tempo do desenvolvimento de sua biblioteca na correção de problemas identificados, entre estes pode-se destacar três, que são bastante comuns e foram corrigidos:

- *Inter-marker confusion*: ocorre quando o marcador é encontrado, mas é confundido com outro;
- *False positive*: acontece quando é detectado um marcador quando não há nenhum presente na cena;

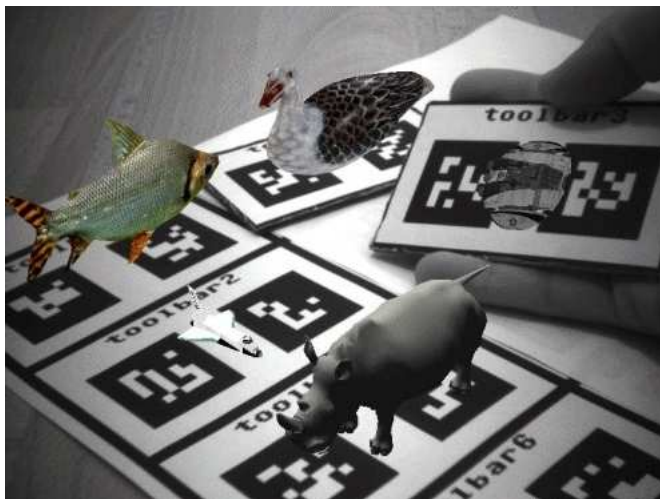


Figura 2.18: Exemplos de aplicações de Realidade Aumentada utilizando ARTag (FIALA, 2010).

- *False negative*: acontece quando existe um marcador no ambiente, mas ele não é detectado.

Além da correção de alguns problemas, o ARTag permite que o marcador seja reconhecido mesmo que uma parte dele esteja oclusa. Isso acontece devido à metodologia de detecção, que utiliza uma estratégia em que o código binário do marcador identificado na imagem é extraído, gerando quatro combinações de uma sequência de 36 bits, simulando as quatro posições em que o marcador poderia estar submetido devido sua rotação. Essa sequência binária possui encapsulado um ID de 10 bits que identifica o marcador e 26 bits de redundância, fazendo com que haja uma boa redução nas chances de detecção e identificação falsas e, além disso, garantindo unicamente uma das quatro possíveis rotações (FIALA, 2004).

2.5.3 FLARToolkit

O lançamento do *ActionScript 3.0* (AS3), juntamente com o ambiente de desenvolvimento *Flex 2.0* e o *Flash Player 9.0*, pela Adobe em 2006, abriu caminho para a evolução das aplicações RIA (*Rich Internet Applications* - Aplicações Ricas em Internet). O desenvolvimento dessas aplicações possibilitou a criação de diversas ferramentas de apoio para a criação de aplicações de Realidade Virtual (RV) para a *web*, suportadas pelo *Flash Player*. Como passo seguinte, a extensão destas tecnologias deram condições para o desenvolvimento de soluções de RA para *web*, com a utilização do FLARToolkit (SILVA et al., 2009).

O FLARToolkit é um conjunto de classes desenvolvidas em AS3, que juntamente com *engines* 3D desenvolvidas para o ambientes Flex 2.0, como o *Papervision3D* (WINDER; TONDEUR, 2009), possibilita o desenvolvimento de aplicações de Realidade Aumentada para web.

Para executar aplicações utilizando o FLARToolkit o cliente deve instalar no navegador o *Flash Player* versão 9 ou superior, e permitir o acesso do *plugin* à *webcam*. Da mesma forma que as bibliotecas mencionadas anteriormente, o FLARToolkit faz uso de marcadores para o posicionamento dos objetos virtuais.

2.6 Considerações Finais

Os tópicos abordados neste capítulo são muito importantes para o pleno entendimento do conceito e visualização das diversas possibilidades de aplicações para ambientes de RA permitindo, então, uma reflexão acerca da interação nestes ambientes.

Logo, o objetivo deste capítulo não é ensinar Realidade Aumentada, metodologias ou tecnologias de desenvolvimento de aplicações, mas prover ao leitor os conceitos básicos necessários para o entendimento do protótipo apresentado nesta dissertação.

Nesse sentido, o próximo capítulo apresenta o estado da arte por meio de trabalhos relacionados a esta proposta.

3 *Estado da Arte*

3.1 Introdução

Com o intuito de conhecer novas tecnologias que colaborem com o desenvolvimento do sistema proposto foi realizado um estudo do estado da arte em relação a aplicações de interação com ambientes de Realidade Aumentada. Portanto, neste capítulo, são apresentados alguns trabalhos que foram considerados de maior relevância em relação aos conceitos propostos e/ou métodos abordados.

Observa-se que o estudo destes trabalhos é de suma importância, visto que se inteirar com iniciativas semelhantes permite ao pesquisador ampliar sua visão e conhecimentos acerca do tema pesquisado, além de possibilitar que o mesmo mantenha o foco em problemas ainda não solucionados, auxiliando na definição das metas da pesquisa.

Dessa forma, ao final deste capítulo, será apresentado um quadro comparativo das tecnologias apresentadas.

3.2 Metodologia para a Análise

Para a construção deste capítulo e do trabalho em si, foram pesquisadas publicações que explorassem a interação em ambientes de RA através de gestos, considerando este tipo de interface como tangível ao usuário.

Tendo isto em vista, optou-se por dar foco às seguintes funcionalidades no momento da análise dos trabalhos relacionados:

- Hardware necessário para o funcionamento;
- Metodologia para a detecção da pele humana;

- Utilização ou não de gestos para atuação no ambiente de RA;
- Aparato utilizado para a interação (luvas, mão nua, marcadores fiduciais, etc);
- Número de graus de liberdade na manipulação do objeto virtual;
- Dependência do ambiente de trabalho.

3.3 Trabalhos Relacionados

3.3.1 Finger Tracking for Interaction in Augmented Environments

Desenvolvido pelo *Interactive Media Systems Group* da Universidade Tecnológica de Viena, o trabalho intitulado “*Finger Tracking for Interaction in Augmented Environments*” (DORFMÜLLER-ULHAAS; SCHMALSTIEG, 2001) tem o objetivo de mostrar um sistema de rastreamento de dedos que permite a interação mediante gestos que, segundo os autores, é simples, barata, rápida, precisa e robusta em relação a oclusão dos marcadores.

A detecção dos gestos é realizada através de um sistema com duas câmeras, fontes de luz infra-vermelha e filtros especiais para detecção deste espectro de luminosidade posicionados em uma área de trabalho pré definida. O usuário, por sua vez, deve fazer uso de uma luva que conta com um exoesqueleto formado por esferas retrorreflexivas e separadores de tamanhos conhecidos, que é mostrada na Figura 3.1.



Figura 3.1: Luva com marcadores retrorreflexivos (DORFMÜLLER-ULHAAS; SCHMALSTIEG, 2001).

Um dos requisitos desta aplicação é permitir que a calibração das câmeras seja transparente ao usuário. Desta forma, é realizada uma calibração adaptativa através da detecção de um ponto adquirido por cada uma das câmeras. Assim, a partir

deste processo, que é detalhado em (DORFMÜLLER; WIRTH, 1998), são estimadas as distâncias focais, a translação, rotação e profundidade de uma câmera em relação à outra.

A segmentação do dedo que contém os marcadores é realizada através de limiarização, ou seja, *pixels* que possuem valores RGB entre uma faixa de valores pré-definida são classificados como área de interesse, permitindo a detecção dos marcadores. Com a determinação da posição das esferas retrorreflexivas, o sistema de segmentação determina o centro de gravidade para cada marcador que, a partir deste estágio, será o ponto de referência deste no espaço tridimensional.

A partir da localização das esferas no espaço 3D, o sistema faz a conexão dessas esferas através da determinação do menor caminho de ligação entre estas. Como o nível de ruído é alto, é utilizado um filtro de Kalman¹ para extrair o estado cinemático do ruído e também auxiliar na predição das posições futuras do marcador.

Ao fim do trabalho os autores mostram um experimento no qual o usuário manipula peças virtuais em tabuleiro de xadrez real (Figura 3.2), para isso, o sistema de rastreamento foi acoplado via *sockets* com o sistema de RA *Studierstube*.



Figura 3.2: Experimento com Realidade Aumentada mostrado em (DORFMÜLLER-ULHAAS; SCHMALSTIEG, 2001).

Segundo os autores, o experimento foi realizado em um ambiente computacional com processador Athlon 800 MHz, câmeras PULNix TM-560 PAL e uma placa de aquisição ELTEC PcEye2, proporcionando a captação de 25 frames por segundo e

¹O filtro de Kalman consiste em um conjunto de equações matemáticas que fornecem meios para estimar o estado de um processo. Sendo muito poderoso em vários aspectos, como a estimação de estados do passado, presente e futuro, podendo obter bons resultados até mesmo quando a natureza do sistema modelado não é conhecida com precisão (WELCH; BISHOP, 1995).

uma latência entre 50 e 100 milissegundos.

Com este trabalho pôde-se observar que esse tipo de interação é mais intuitiva e fácil de aprender, visto que os movimentos são semelhantes aos métodos usuais de manipulação de objetos, no entanto, existem problemas quanto a oclusão das esferas retrorreflexivas, que devem estar voltadas para as câmeras. Dessa forma, como trabalhos futuros, existe a pretensão de implementar inserção da manipulação com as duas mãos e a detecção do contorno das mãos sem restrições de fundo.

3.3.2 Assembly Design and Evaluation based on Bare-Hand Interaction in an Augmented Reality Environment

Neste trabalho, os autores (WANG et al., 2009) propõe um ambiente de RA para montagem de protótipos e avaliação dos mesmos utilizando as mãos como interface homem máquina.

Baseado no ARToolKit (KATO et al., 2010), o protótipo desenvolvido provê uma contribuição muito interessante no que se diz respeito a interação em ambientes de RA, a qual consiste na utilização da mão nua, ou seja, sem nenhum tipo de aparato para manipulação dos objetos virtuais. A Figura 3.3 mostra a configuração da plataforma de montagem relatada. Observa-se que o ambiente é composto por um sistema de visualização que utiliza um HMD, uma câmera para captação das imagens, marcadores fiduciais para o posicionamento inicial das peças e das mãos do usuário como ferramenta de manipulação das partes virtuais.

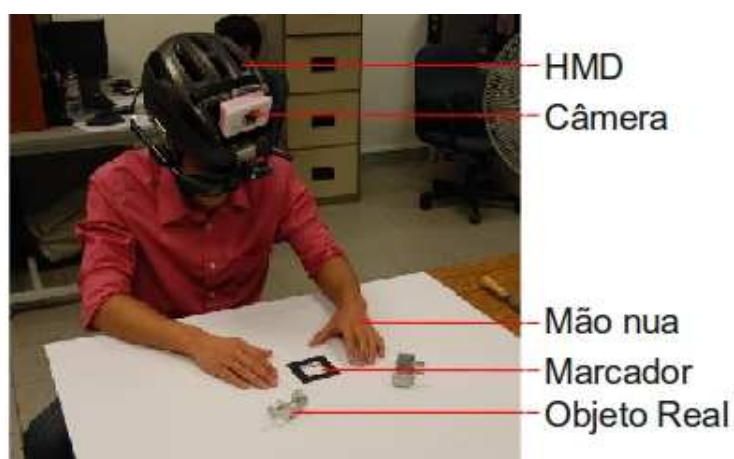


Figura 3.3: Plataforma de montagem da aplicação de RA com interação através das mãos (Adaptada de (WANG et al., 2009)).

Como a determinação dos parâmetros da distribuição de cores da pele do usuário

é muito difícil devido à variação dos iluminação do ambiente, foi utilizada uma rede neural RCE (*Restricted Coulomb Energy*) para a realização da classificação da mão humana o que, segundo os autores, foi uma solução simples e eficaz.

Adicionalmente, cada pixel dos quadros capturados foi convertido para o espaço de cores CIE $L^*a^*b^*$, que foi definido pela CIE (*Comission International de l'Éclairage*) em 1976, no qual L^* representa a luminância e a^* e b^* os canais de cromaticidade. A utilização deste espaço de cores é vantajoso à medida que modificações das condições de iluminação provocam sutis translações ao longo do eixo L^* e, segundo (YIN; GUO; XIE, 2001), apresenta melhores resultados em relação à qualidade da resposta e consumo de tempo para este tipo de aplicação, quando comparado com os espaços de cores RGB (*Red, Green, Blue*) e HSI (*Hue, Saturation, Intensity*).

A partir da separação da mão, é realizada a determinação da ponta dos dedos do usuário através de um algoritmo baseado em curvatura, que determina os vales e picos da imagem. Na Figura 3.4 é mostrado um exemplo de segmentação da mão e detecção das pontas dos dedos, observa-se que o resultado obtido satisfaz os requisitos deste tipo de aplicação, no entanto, não se tem dados sobre tempo de execução.



Figura 3.4: Segmentação da mão e determinação das pontas dos dedos (WANG et al., 2009).

Com a implementação do algoritmo de segmentação, os autores mostram um experimento que consiste na montagem de algumas peças da carenagem de uma motocicleta, a Figura 3.5 mostra um exemplo de interação da mão do usuário com uma das peças.

Observa-se que a interação é realizada no espaço 2D, já que se utiliza somente uma câmera. No entanto, apesar da restrição de movimentos, este tipo de montagem possui grandes vantagens, seja na redução de custos em relação a uma prototipagem

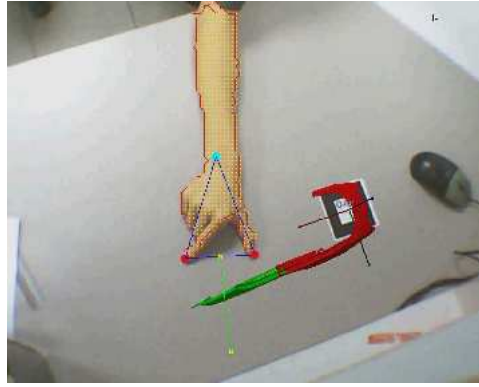


Figura 3.5: Interação com o ambiente de RA (WANG et al., 2009).

comum, no preço do hardware, que é mínimo ou na facilidade da manipulação com as mãos em relação à utilização de mouse e teclado. Como trabalhos futuros, os autores sugerem a expansão desta técnica para o espaço tridimensional, o tratamento de oclusões da mão e a melhoria na robustez do algoritmo de segmentação.

3.3.3 Manipulation of Virtual Objects in Marker-less AR System by Fingertip Tracking and Hand Gesture Recognition

Lee e Chun (LEE; CHUN, 2009) apresentam um método para manipular objetos virtuais em um sistema de RA sem marcador com o auxílio do modelo de coordenadas 3D desenvolvido.

O método proposto é dividido em três etapas principais, mostrados na Figura 3.6. O primeiro estágio consiste em efetuar a segmentação da mão do usuário, logo, determina-se também a posição das pontas dos dedos e do centro da mão, que são as entradas para o algoritmo da próxima fase.

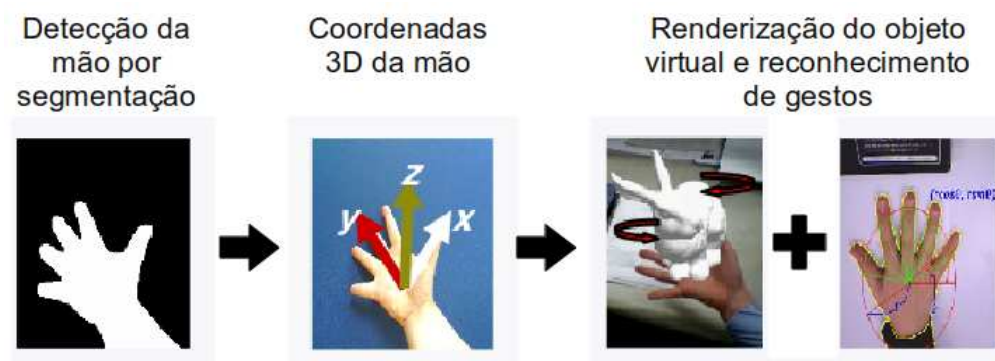


Figura 3.6: Fases do sistema proposto em (LEE; CHUN, 2009) (adaptado de (LEE; CHUN, 2009)).

Na segunda fase, a posição da câmera é estimada, permitindo que o sistema determine as coordenadas da mão. Com essas coordenadas, passa-se à fase final, na qual os objetos virtuais são renderizados no centro das coordenadas da mão e podem ser controlados através de gestos pré-definidos.

Para simplificar a tarefa de detectar a mão em ambientes complexos, foi utilizado o espaço de cores YC_bC_r , no qual a componente Y se refere a luminância e as componentes C_b e C_r às diferenças entre as cores azul e vermelho em relação a uma referência padrão. Como as condições de luminosidade não são interessantes para este tipo de aplicação, os autores optaram por descartar a componente Y , utilizando em seu modelo somente as componentes que carregam as informações de cores.

Dessa forma, para realizar a classificação, foi definida uma faixa de valores responsável por classificar se o pixel é ou não da mão (ou de outra parte da pele) do usuário. Com a definição da área que contem a mão do usuário, o sistema utiliza operadores morfológicos, obtendo um resultado refinado, pronto para a próxima etapa de processamento.

Feita a extração da mão, um método baseado na característica de curvatura da região da mão é utilizado, logo, o objetivo é encontrar os picos dessa imagem, que provavelmente coincidirão com as pontas dos dedos.

Com a determinação da posição dos dedos, parte-se para a determinação da posição da câmera, para isso, é utilizado um procedimento de calibração em que um padrão quadriculado é associado com uma imagem da mão esticada. Dessa forma, o sistema de coordenadas estimado é movido para o centro da mão de acordo com as posições relativas das pontas dos dedos em relação à origem do plano quadriculado, gerando o sistema de coordenadas da mão do usuário (Figura 3.7).

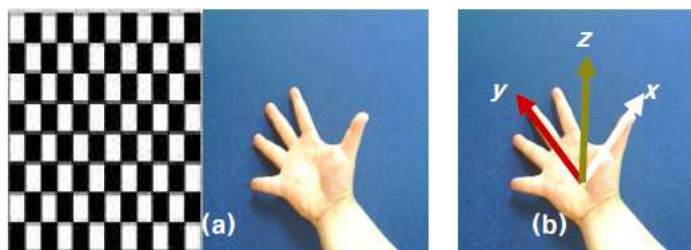


Figura 3.7: Correlação entre a imagem da mão e o padrão quadriculado (a). Sistema de coordenadas da mão do usuário (b) (LEE; CHUN, 2009).

A manipulação dos objetos virtuais no ambiente de RA é realizada a partir da detecção dos gestos realizados. Cada gesto é associado a um padrão de entrada e,

em tempo de execução, é realizado um reconhecimento baseado na similaridade do gesto com os padrões pré-gravados.

No experimento realizado, o sistema proposto, foi implementado em linguagem C++ em ambiente *WindowsTM* com auxílio das bibliotecas MFC (*Microsoft Foundation Classes*) e OpenGL, neste foram realizadas interações com objetos virtuais utilizando três gestos, os quais podem ser visualizados na Figura 3.8 e tem como função efetuar as operações rotação, escala e translação nos objetos virtuais.



Figura 3.8: Gestos cadastrados no experimento de (LEE; CHUN, 2009) (rotação, escala e translação, respectivamente).

De acordo com as diversas posições dos dedos os objetos virtuais foram modificados como esperado e os autores não relataram problemas referentes à iluminação, no entanto, observa-se que os experimentos mostrados aparentam ser sido realizados em um ambiente com fundo e iluminação controlados. Como trabalho futuro, há a intenção de expandir a proposta para interação em ambientes 3D.

3.3.4 FingARtips: Gesture Based Direct Manipulation in Augmented Reality

O trabalho abordado em (BUCHMANN et al., 2004) apresenta uma técnica para interação com gestos em ambientes de Realidade Aumentada aplicada a um sistema de planejamento urbano. Para a exibição do mapa virtual, que consiste na área de trabalho desta aplicação, é utilizado um aparato de 115 por 60 cm coberto com 24 marcadores retangulares com 10 cm em cada lado, como mostrado na Figura 3.9.

A interação com o ambiente é realizada mediante uma luva que possui três marcadores localizados na parte de cima da mão e nos dedos polegar e indicador, formando um tipo de garra, conforme pode ser observado na Figura 3.10. Complementarmente, o sistema de visualização conta com um HMD com resolução de 640x480 pixels e as imagens são capturadas com uma *webcam* comum (ver Figura 3.9).

Com a montagem de todo o ambiente de *hardware*, a próxima preocupação con-



Figura 3.9: Ambiente de trabalho da aplicação utilizada em (BUCHMANN et al., 2004).



Figura 3.10: Luva para interação proposta em (BUCHMANN et al., 2004).

siste no sistema de rastreamento, para isso, os autores utilizaram o ARToolkit (KATO et al., 2010), que é responsável por calcular a posição e orientação da câmera em relação aos marcadores fiduciais. Neste ponto, o rastreamento feito pela biblioteca é dividido em dois cenários: monitoramento da posição da cabeça do usuário e detecção da posição da mão, os quais são responsáveis, respectivamente, por posicionar o mapa virtual e proporcionar a interação.

O posicionamento do ambiente é realizado com a detecção de múltiplos marcadores (24) que estão posicionados na mesa, observa-se que essa quantidade de marcadores garante o funcionamento do sistema mesmo com a oclusão causada pelo movimento do usuário.

Tendo a possibilidade de localizar a mão do usuário no ambiente virtual, surge a necessidade de efetuar a identificação do gesto realizado, para isso, os autores utilizaram um modelo simplificado da mão, no qual o gesto formado pelos movimentos do polegar e indicador são determinados em relação ao marcador posicionado na mão, como pode-se observar na Figura 3.11. Com isso, são convencionados quatro tipos de interação possíveis: agarrar, apontar, navegar e pressionar.

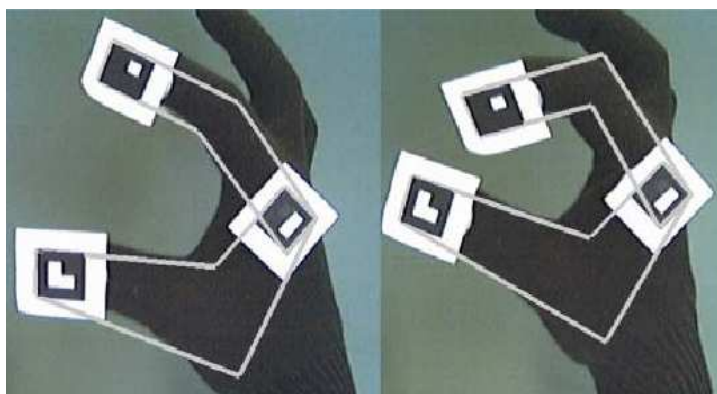


Figura 3.11: Modelo simplificado da mão (BUCHMANN et al., 2004).

Para sanar o problema de oclusão da mão por objetos virtuais, foi criada uma representação da garra formada pelos três marcadores colocados na luva (Figura 3.12). Esta solução, apesar de satisfazer a necessidade prover a visualização da ferramenta de manipulação de objetos virtuais, remove a mão do usuário, o que não é considerado desejável, já que se deseja manter a imersão do usuário no ambiente de RA.

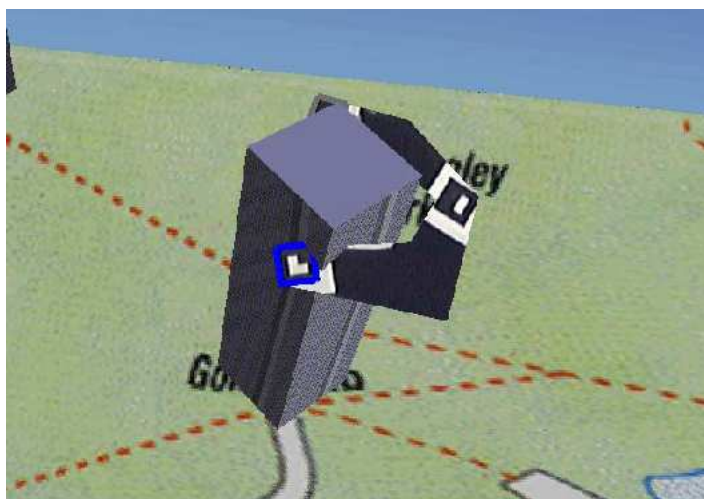


Figura 3.12: Solução para oclusão da mão por marcadores de (BUCHMANN et al., 2004).

Apesar dos resultados preliminares obtidos, os autores também ressaltam que a aplicação possui sérios problemas de usabilidade, dentre eles, a impossibilidade de

rotacionar a mão, já que os marcadores devem estar no campo de captura da câmera localizada na cabeça do usuário, além disso, foram relatados problemas em relação à robustez e precisão do sistema, que são agravados pela má iluminação do ambiente de trabalho.

Como trabalhos futuros há a intenção de melhorar o método proposto, permitindo a identificação de mais gestos e a criação de um cenário para a manipulação com as duas mãos.

3.3.5 Opportunistic Tangible User Interfaces for Augmented Reality

O trabalho de Henderson e Feiner (HENDERSON; FEINER, 2010) apresenta o conceito de *Opportunistic Controls* (OC) que consiste em uma interface tangível ao usuário que faz proveito dos métodos usuais de interação do ser humano, como gestos e toque, dessa forma uma interface OC permite ao usuário interagir com um ambiente de RA através do toque em locais pré-determinados de uma superfície de trabalho.

Com a utilização deste tipo de interface, um mecânico, por exemplo, poderia utilizar a exibição de documentos virtuais que lhe auxiliariam no procedimento a se realizar e até mesmo na visualização de características do objeto da manutenção, como partes físicas ocultas ou dados do funcionamento do equipamento, economizando tempo com a eliminação da busca em manuais e outros meios convencionais.

A título de exemplo, a Figura 3.13 mostra a atuação do usuário para a seleção de um determinado tipo de manutenção, observa-se que são apresentados diversos botões com possíveis cenários de defeito (*serviceable* (reparado), *leaking* (vazamento), *damaged* (danificado), *missing* (perdido)). É possível visualizar também o *feedback* visual que é dado ao usuário quando um dos *widgets* é selecionado.

O reconhecimento dos gestos do usuário é efetuado óticamente com uma câmera posicionada de forma a captar os *widgets* da interface. Logo uma *thread* de execução examina cada imagem verificando se partes da mão do usuário (tipicamente os dedos) estão contidas no espaço 2D que contém o *widget*. Dessa forma, caso seja encontrado um evento de toque, é acionada a ação correspondente ao elemento selecionado.

A visualização do ambiente aumentado é realizado através das imagens captadas por outra câmera que é acoplada ao HMD utilizado pelo usuário, logo, o posicionamento dos botões no equipamento é realizado através da determinação da posição

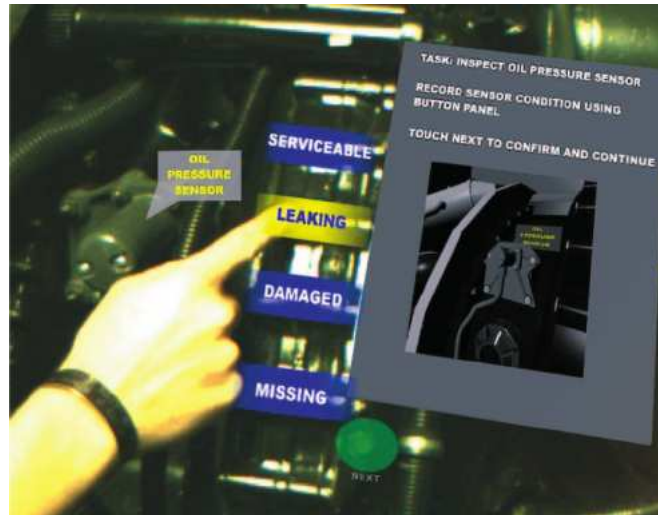


Figura 3.13: Exemplo de utilização de um OC (HENDERSON; FEINER, 2010).

da câmera em relação aos marcadores fiduciais, com o auxílio do sistema ARTag (FIALA, 2010).

Com a junção da imagem capturada e a atuação correspondente ao gesto realizado, pode-se notar que não há a preocupação com a oclusão dos objetos virtuais, ou seja, quando o usuário posiciona a mão para interagir com o *widget*, como acontece na Figura 3.13, este fica posicionado acima do dedo do usuário.

O protótipo implementado possui uma característica bastante diferente dos sistemas convencionais: cada câmera é ligada em um computador, desta forma, a atuação detectada no computador que monitora os gestos é enviada através da conexão em rede local para o sistema que capta e gera as imagens aumentadas vistas pelo usuário.

Neste trabalho, os autores relatam também os testes realizados com voluntários que reagiram bem à utilização do sistema, no entanto, são enumeradas diversas limitações do sistema, dentre elas podemos destacar:

- detecção de marcadores fiduciais: os marcadores são elementos que não fazem parte do ambiente real e poderiam ser substituídos pelo reconhecimento do equipamento através de suas características (forma, textura, etc.);
- algoritmo de segmentação: é altamente dependente da iluminação do ambiente, logo, o sistema necessita de funcionar em condições de iluminação ideais, o que não é viável para todos os tipos de ambientes, reduzindo assim a possibilidade de utilização do sistema fora do laboratório;

- detecção de gestos em múltiplos *widgets*: todos os botões mostrados nas imagens aumentadas podem ser acionados ao mesmo tempo, esta característica poderia facilitar a utilização multi-toque do sistema, no entanto, os autores relatam a necessidade de melhorar o algoritmo de atuação para este caso.

Como trabalhos futuros, foi relatada a intenção de efetuar algumas modificações no design da aplicação, como a posição dos botões, além da melhoria no método de segmentação que adicionaria a detecção de características dos equipamentos, evitando a utilização dos marcadores.

3.3.6 Using Haar-like Feature Classifiers for Hand Tracking in Tablet Augmented Reality

Fernandes e Fernández (FERNANDES; FERNÁNDEZ, 2010), apresentam um método para interação em RA que utiliza classificadores Haar (VIOLA; JONES, 2001) para o reconhecimento de mãos no ambiente de trabalho e dos gestos realizados por elas.

A aplicação abordada, permite ao usuário realizar translação, escala e rotação de objetos virtuais no espaço bidimensional utilizando as mãos sem nenhum aparato. O ambiente de trabalho, mostrado na Figura 3.14, consiste em um plano com a matriz de marcadores do ARTag (FIALA, 2010), que tem como função delimitar a área de atuação da mão e auxiliar na detecção de colisões.

Dessa forma, a interação é realizada com gestos em duas dimensões. O gesto que permite a movimentação do objeto virtual é realizado colocando a mão espalmada sob o elemento, como pode ser visto na Figura 3.14, que será selecionado para a realização do posicionamento. Já a rotação e o redimensionamento são realizados conforme mostrado na Figura 3.15, logo, para rotacionar o objeto, o usuário deve mostrar os dedos indicadores de ambas as mãos e rotacionar as mãos no sentido horário ou anti-horário formando um círculo. O redimensionamento é realizado através do afastamento dos dedos indicadores, o qual é traduzido na operação de escala nos três eixos cartesianos.

Um ponto importante, consiste que os autores consideram que as mãos do usuário deverão permanecer próxima ao plano com os marcadores restringindo a área de interação.

A performance do classificador Haar, que detecta os gestos, foi considerada pe-

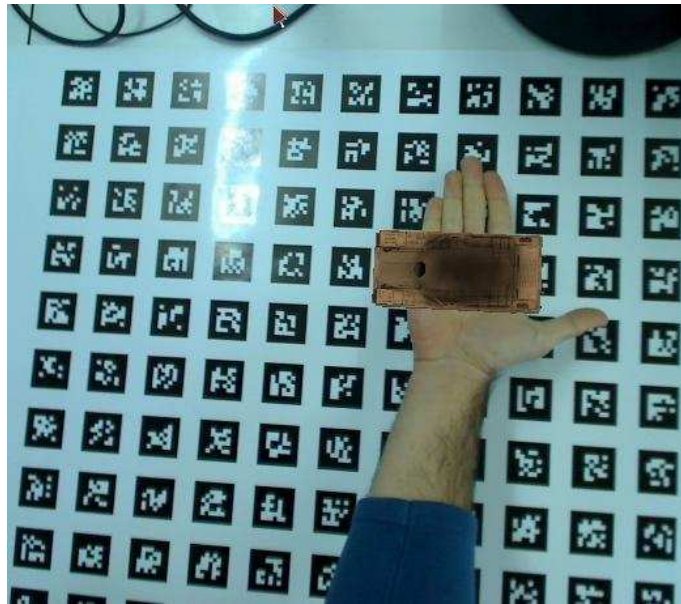


Figura 3.14: Ambiente de trabalho de (FERNANDES; FERNÁNDEZ, 2010) e movimentação de objeto virtual.

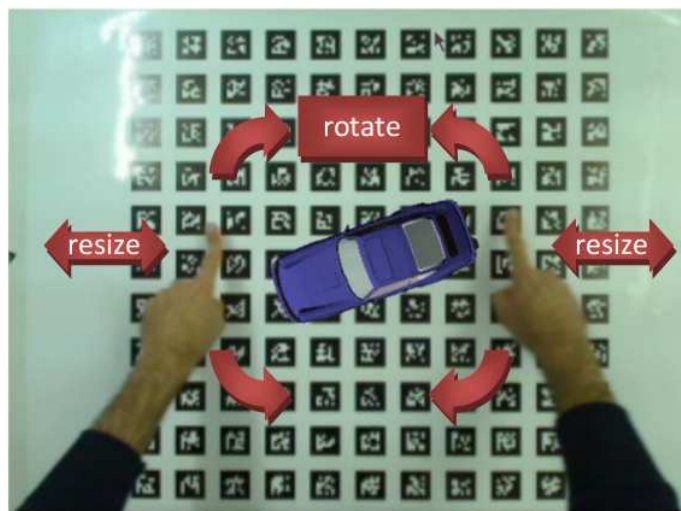


Figura 3.15: Operações de rotação e escala mostradas em (FERNANDES; FERNÁNDEZ, 2010).

quena para mãos de pessoas que não participaram do treinamento deste, alcançou uma taxas de 63% e 83% de detecção para os gestos de posicionamento e rotação/escala, respectivamente.

Como trabalhos futuros, os autores de (FERNANDES; FERNÁNDEZ, 2010) citam a necessidade de efetuar testes formais com usuários visando a comparação entre métodos formais de interação em planos deste tipo.

3.4 Considerações Finais

A partir da análise dos trabalhos relacionados, pode-se perceber que a maioria dos autores preza por utilizar um ambiente com o mínimo de *hardware* possível, minimizando os custos e a complexidade da montagem. No entanto, a utilização de componentes adicionais, como câmeras com capacidade de captar somente luz infravermelha pode contribuir na facilidade de detecção do objeto utilizado para a interação. Tendo em vista este ponto, a Tabela 3.1 mostra a comparação do *hardware* utilizado para a captação das imagens do ambiente real e na detecção da mão do usuário, observa-se que o dispositivo de visualização não foi levado em conta, visto que, em todos os trabalhos pode-se utilizar um HMD para este fim.

A Tabela 3.1 aborda também o espaço de cores e as bibliotecas de RA utilizadas. Unindo estas duas características, pode-se notar que os trabalhos os quais utilizam as bibliotecas de RA sem alterações fazem uso do esquema de cores RGB, visto que, este padrão possui bons resultados em relação à detecção de marcadores fiduciais.

Em contrapartida, existem iniciativas de buscar espaços de cores que auxiliem no processo de detecção de objetos mais complexos, como é o caso dos trabalhos de (LEE; CHUN, 2009) e (WANG et al., 2009), os quais realizaram implementações do sistema de segmentação buscando prover uma menor sensibilidade à variação de luz, mediante a utilização de modelos que tenham a característica de possibilitar a atenuação ou eliminação desta componente. Considerando ainda o espaço de cores, é importante notar que o trabalho de (DORFMÜLLER-ULHAAS; SCHMALSTIEG, 2001), é um caso a parte, já que a utilização do espaço RGB se fez suficiente, devido à captação do espectro de luz infravermelha.

Passando à comparação dos parâmetros definidos para a avaliação dos trabalhos correlatos, tem-se a Tabela 3.2, que relaciona as seguintes características para confrontação:

- Detecção da mão nua: refere-se à não utilização de luvas e outros tipos de marcações na mão do usuário, determinando a necessidade da detecção da pele do ser humano;
- Compatível com múltiplas cores de pele: tem o objetivo de elucidar a facilidade de adaptação do sistema ao usuário, já que o desejável é que o ambiente esteja operante sem a necessidade de exaustivas etapas de calibração;

Tabela 3.1: Comparação entre o *hardware* necessário, o espaço de cores e a *biblioteca* de RA utilizada nos trabalhos relacionados.

Publicações avaliadas	Hardware necessário	Espaço de cores	Biblioteca de RA
Finger Tracking for Interaction in Augmented Environments	• fonte de luz infra-vermelha (IR) • 2 câmeras com filtro IR • luva com marcadores retrorreflexivos	RGB	Studierstube
Assembly Design and Evaluation based on Bare-Hand Interaction in an Augmented Reality Environment	• 1 câmera comum	CIE L*a*b	Adaptação do ARToolKit
Manipulation of Virtual Objects in Marker-less AR System by Fingertip Tracking and Hand Gesture Recognition	• 1 câmera comum	YCbCr	Desenvolvido para o projeto (OpenGL e C++)
FingARTips: Gesture Based Direct Manipulation in Augmented Reality	• 1 câmera comum • Plataforma com marcadores • luva com marcadores fiduciais	RGB	ARToolKit
Opportunistic Tangible User Interfaces for Augmented Reality	• 2 câmeras comuns	RGB	ARTag
Using Haar-like Feature Classifiers for Hand Tracking in Tabletop Augmented Reality	• 1 câmera comum • plataforma com marcadores • algoritmo de Viola-Jones para segmentação da mão	RGB	ARTag

- Suporta variações de luz: faz referência a facilidade do sistema de se adaptar a variações de luminosidade, facilitando a utilização em ambientes não controlados;
- Determinação do gesto realizado: analisa se o algoritmo implementado faz distinção de gestos ou se somente detecta a mão em uma posição pré-determinada;
- Independe do ambiente de trabalho: o observa a dependência da aplicação em

relação a um ambiente livre de fundos complexos e com iluminação ideal.

Com o estudo comparativo realizado, entende-se que foi possível adquirir uma visão global do estado da arte de aplicações que utilizam interação através de gestos em ambientes de Realidade Aumentada. Neste aspecto, a Tabela 3.2 mostra a necessidade de investigar técnicas que possam melhorar a detecção de diversos tipos de pele e outras texturas de interesse, proporcionando também resistência a variação de luz em ambientes que não estejam preparados para este fim.

Tabela 3.2: Comparação das características analisadas nos trabalhos relacionados.

Trabalhos Relacionados	Detecção da mão nua	Compatível com múltiplas cores de pele	Suporta variações de luz	Determinação do gesto realizado	Independência do ambiente de trabalho
Finger Tracking for Interaction in Augmented Environments	✗	✗	✓	✓	✗
Assembly Design and Evaluation based on Bare-Hand Interaction in an Augmented Reality Environment	✓	✗	✓	✓	✗
Manipulation of Virtual Objects in Marker-less AR System by Fingertip Tracking and Hand Gesture Recognition	✓	✓	✗	✓	✗
FingARtips: Gesture Based Direct Manipulation in Augmented Reality	✗	✗	✗	✓	✗
Opportunistic Tangible User Interfaces for Augmented Reality	✓	✗	✗	✗	✓
Using Haar-like Feature Classifiers for Hand Tracking in Tabletop Augmented Reality	✓	✓	✗	✓	✗

4 *Análise e Especificação do Sistema*

4.1 Introdução

Com base nos estudos relacionados nos capítulos anteriores e nos fundamentos de Engenharia de Software, elaborou-se o projeto do sistema. Neste capítulo são abordados os pontos principais deste projeto, através da análise de requisitos, da apresentação de alguns diagramas comportamentais UML (*Unified Modeling Language*) e da discussão da arquitetura implementada.

4.2 Especificação do Sistema

Visando a compreensão das funcionalidades e relações entre os módulos do sistema desenvolvido, esta seção tem o objetivo de elucidar a determinação dos requisitos do sistema bem como alguns diagramas UML, tidos como importantes para o entendimento do funcionamento global do aplicativo desenvolvido.

4.2.1 Requisitos do Sistema

Segundo (SOMMERVILLE, 2007) os requisitos de um sistema consistem na descrição dos serviços a serem disponibilizados por este, bem como suas restrições operacionais. Desta forma, os requisitos representam as necessidades do público alvo do sistema.

Os requisitos de um sistema podem ser classificados em requisitos funcionais e requisitos não funcionais. Sommerville (SOMMERVILLE, 2007) os define da seguinte forma:

- **Requisitos funcionais:** consiste na declaração dos serviços que o sistema deve fornecer ao usuário, como será o comportamento mediante de entradas específicas e o comportamento em certas situações;
- **Requisitos não funcionais:** contempla as restrições sobre os serviços ou funções oferecidos pelo sistema.

A partir destas definições, foram elaborados os requisitos do sistema abordado neste trabalho, que consistem em:

- Requisitos funcionais:
 - Reconhecimento da mão nua;
 - Compatibilidade com múltiplas cores de pele;
 - Suportar variações de luz;
 - Minimizar a dependência do ambiente de trabalho;
 - Permitir a determinação da posição/gesto no espaço tridimensional;
 - Realizar interação/seleção em menus;
 - Não necessitar de exaustivas etapas de calibração e/ou treinamento.
- Requisitos não funcionais:
 - Ser compatível com sistemas operacionais Linux e Windows;
 - Operar com a taxa mínima de 15 FPS (*Frames per Second*).

4.2.2 Diagrama de Caso de Uso do Sistema

Um diagrama de caso de uso mostra conceitualmente o conjunto de funções que o sistema deve executar para atender aos requisitos do usuário (LIMA, 2010), representando, por meio de uma linguagem simples e visual, o comportamento do sistema através da perspectiva do usuário (GILLEANES, 2009).

Dessa forma, para facilitar o entendimento dos requisitos do sistema, foi desenvolvido o diagrama mostrado na Figura 4.1. Observa-se que é considerado somente um papel, o do usuário que atua com o ambiente de RA, através da interação com menus, criação e manipulação de objetos 3D.

Visando um melhor entendimento dos casos de uso mostrados na Figura 4.1, as tabelas 4.1, 4.2 e 4.3 apresentam a descrição dos casos de uso principais, é importante informar que os casos de uso mais simples, como “Selecionar Objeto” não foram aqui detalhados.

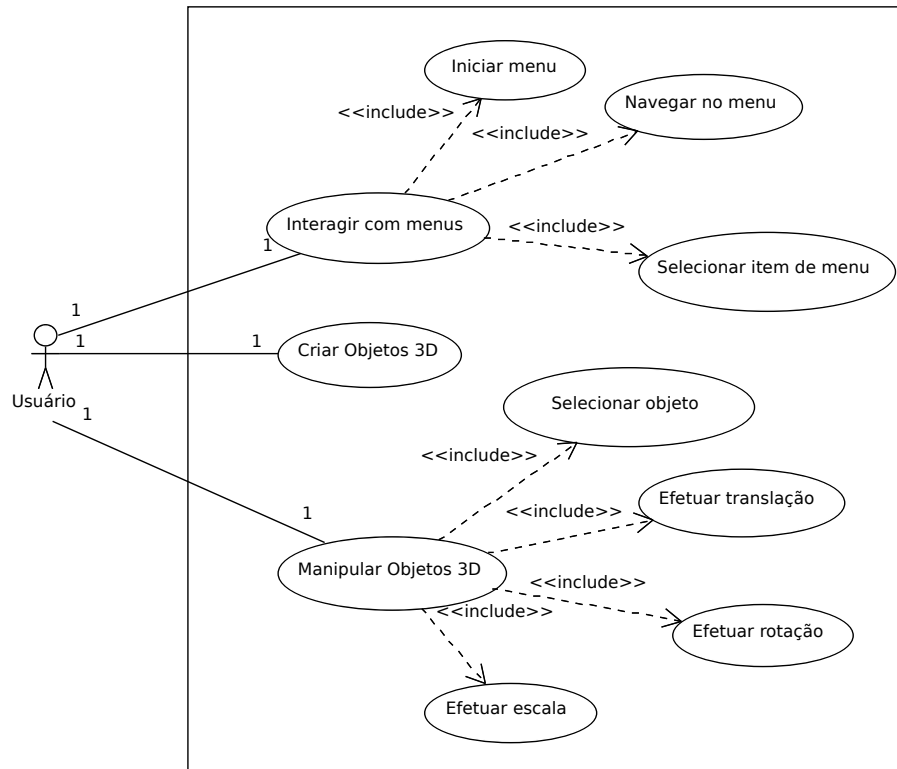


Figura 4.1: Diagrama de caso de uso para o sistema de interação com RA.

Tabela 4.1: Descrição do caso de uso “Interagir com Menus”.

<i>Caso de Uso: Interagir com Menus</i>
Resumo: Inicialização, navegação e seleção em menus. Atores envolvidos: Usuário Descrição: O usuário efetua um movimento com as mãos para inicializar o menu. Através do movimento da mão é realizada a navegação pelos itens do menu. A seleção é realizada através de outro gesto. Exceções: Caso a posição da mão seja perdida, o gesto de inicialização deve ser realizado novamente. Objetivo: Prover a possibilidade da manipulação de menus 2D.

A partir da definição do diagrama de caso de uso, pôde-se desenvolver a visão dinâmica do sistema, a qual será descrita através dos diagramas de atividades mostrados na próxima subseção.

Tabela 4.2: Descrição do caso de uso “Criar Objetos 3D”.

<i>Caso de Uso: Criar Objetos 3D</i>
Resumo: Criação de objeto 3D na cena. Atores envolvidos: Usuário Descrição: O usuário efetua a criação do objeto através de um menu e posiciona no espaço tridimensional. Exceções: Caso a posição da mão seja perdida, o gesto de inicialização deve ser realizado novamente. Objetivo: Criar e posicionar um objeto virtual.

Tabela 4.3: Descrição do caso de uso “Manipular Objetos 3D”.

<i>Caso de Uso: Manipular Objetos 3D</i>
Descrição: Efetuar operações de rotação, translação e escala no objeto virtual selecionado. Atores envolvidos: Usuário Descrição: O usuário seleciona o objeto, e através de um menu escolhe o tipo de operação que deseja realizar, através do movimento da mão, efetua a operação escolhida. Exceções: Caso a posição da mão seja perdida, o gesto de inicialização deve ser realizado novamente. Objetivo: Modificar a posição, ângulos de rotação ou escala do objeto virtual selecionado.

4.2.3 Diagrama de Atividade do Sistema

Um dos diagramas de compõem a visão dinâmica da UML consiste no diagrama de atividades, o qual permite modelar o comportamento do sistema através dos caminhos lógicos que um processo pode seguir. Isso é realizado através de uma estrutura baseada no conceito de Redes de Petri¹, que permite analisar de forma gráfica o comportamento dinâmico de sistemas (LIMA, 2010).

Dessa forma, com a utilização destes conceitos, foram desenvolvidos os diagramas correspondentes às principais atividades do sistema, as quais consistem na interação com o ambiente e a renderização do mesmo. Observa-se que essas atividades ocorrem de forma paralela, visto que o usuário necessita de uma resposta visual para cada ação realizada, com um mínimo de *delay*, para que a imersão não fique prejudicada.

Interação com o Ambiente de RA

Um dos módulos mais importantes da implementação, consiste na definição do fluxo de execução para o processamento da interação no ambiente de RA. A metodologia utilizada para a interação pode ser visualizada na Figura 4.2.

Observa-se que, primeiramente, o sistema fica aguardando o início da sessão de interação, que é inicializada através do gesto “*wave*”, que será descrito posteriormente.

Após o início da sessão, tem-se a atualização da posição da mão e a verificação se algum gesto está sendo realizado.

Se ocorrer algum gesto, este é identificado e caso esteja prevista alguma ação para ele, são lançados os eventos que mapeiam o gesto para o bloco “Atuar no ambiente”, o qual tem a função de tratar o evento e efetuar as modificações necessárias no ambiente de trabalho.

Terminada a atuação, o fluxo de execução retorna para a verificação de posição ou realização de gestos.

É importante lembrar que o processo de interação atua em conjunto com o de renderização, sendo dependente deste para existir. Portanto o processo de interação

¹Redes de Petri (RdP) foram criadas a partir tese de doutorado de Carl Adam Petri, intitulada *Kommunikation mit Automaten* (Comunicação com Autômatos), apresentada à Universidade de Bonn em 1962. Desde o princípio, RdP objetivaram a modelagem de sistemas com componentes concorrentes.

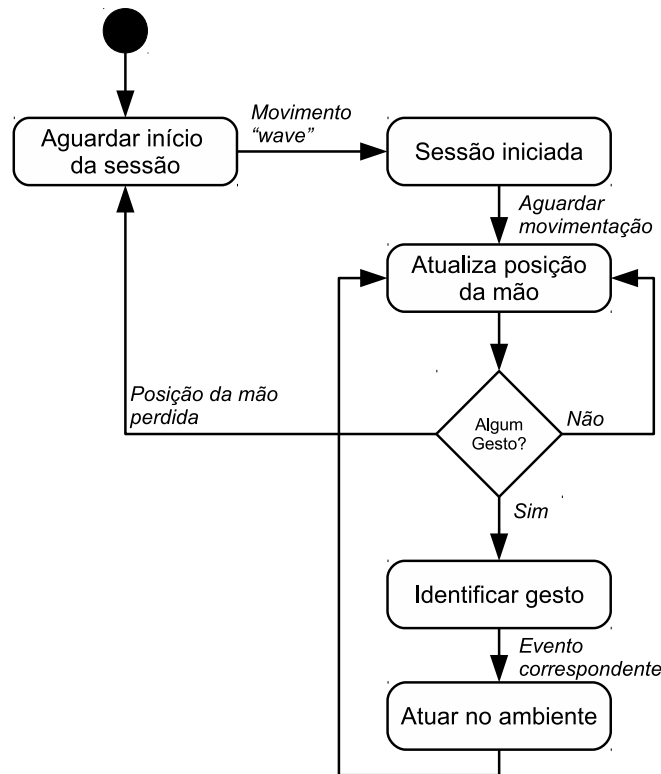


Figura 4.2: Diagrama de atividade para a interação.

é finalizado, em qualquer ponto, de acordo com o término processo de renderização, que será abordado a seguir.

Renderização do Ambiente

A renderização do ambiente é feita em paralelo com processo de interação. Desta forma, à medida que o usuário atua no sistema, e os componentes visuais são modificados, ocorre a atualização do *feedback* visual recebido pelo usuário.

A Figura 4.3 ilustra a execução do processo de renderização. Pode-se verificar que diferentemente da interação, esta etapa é iniciada juntamente com o programa, através do carregamento de elementos gráficos da aplicação como os menus e objetos virtuais.

Com o carregamento inicializa-se um *loop* de execução que tem o objetivo de capturar a imagem RGB do ambiente real, que será utilizada para atualizar a imagem do usuário, atualizar a posição e estados dos elementos gráficos contidos na tela e, por fim, renderizar a nova cena e exibir na tela do computador.

Observa-se que a modificação dos elementos gráficos é realizada através de procedimentos executados na atividade de interação, sendo que, sua renderização é ex-

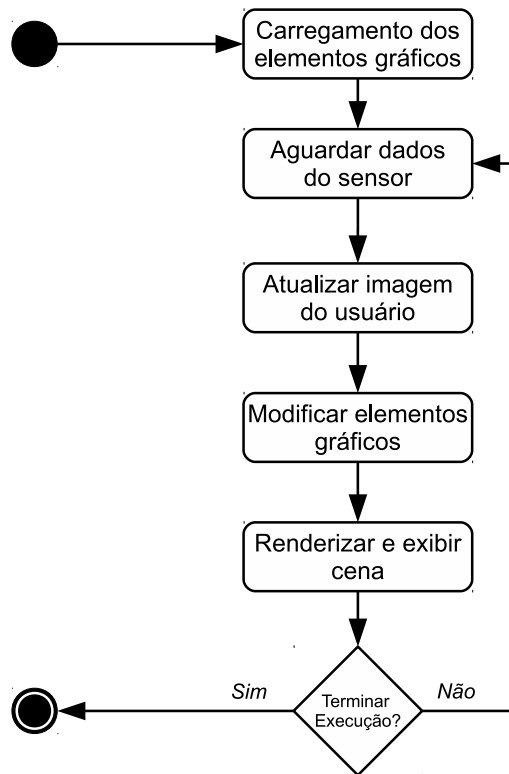


Figura 4.3: Diagrama de atividade para a renderização.

plicitada através do bloco “Modificar elementos gráficos”. Além disso, o término da execução do processo de renderização provoca o término das operações de interação.

4.2.4 Arquitetura do Sistema

Com o objetivo de ilustrar o funcionamento global do sistema a Figura 4.4 mostra um desenho da arquitetura do software desenvolvido. Observa-se que o fluxo de execução, inicia-se com a captura da imagem do usuário, passa pelo processamento e atuação de acordo com o conteúdo recebido e, por fim, é renderizada a imagem correspondente à nova cena. Além disso, nota-se que existem dois blocos principais:

- **Captura e Detecção:** recebe dados (imagem RGB e mapa de profundidade) de um sensor sendo responsável por processar essas informações, determinar se algum gesto está sendo realizado ou apenas determinar a posição da mão no espaço. Neste bloco será realizado o estabelecimento da conexão com o dispositivo, reconhecimento dos gestos e geração de eventos.
- **Gerenciamento de Interação:** é responsável por processar os eventos gerados pelo bloco “Captura e Detecção”, tratar e efetuar a interação (caso o evento seja

deste tipo) e, por fim, efetuar a renderização do novo quadro a ser exibido na interface de visualização do usuário, neste caso o monitor.

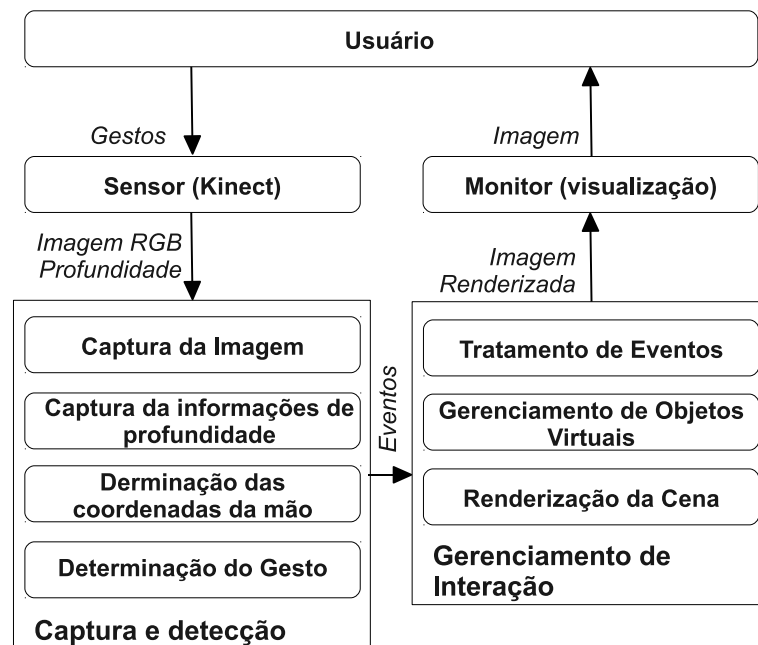


Figura 4.4: Arquitetura do sistema.

Como pode-se observar, a comunicação entre os blocos é realizada através de eventos, os quais podem ser dos seguintes tipos:

- Novo quadro: iniciado quando uma nova imagem RGB é recebida completamente do sensor;
- Novo mapa de profundidade: iniciado quando o mapa de profundidade é recebido completamente do sensor;
- Início da sessão: ocorre quando a mão do usuário começa a ser detectada pelo sistema;
- Atualização da posição da mão: é chamado quando tem-se uma nova posição para mão do usuário;
- Fim da sessão: acontece quando a mão do usuário é perdida;
- Gesto realizado: chamado quando um dos gestos utilizados no sistema é realizado pelo usuário.

Logo, a partir dos eventos mostrados, o sistema tem a capacidade de gerar uma resposta para a visualização do usuário.

4.3 Considerações Finais

A realização dos estudos de engenharia de software permitiram um melhor entendimento do problema fazendo com que o desenvolvimento do sistema fosse bem direcionado, tornando este melhor documentado e, conseqüentemente, mais preciso em relação às capacidades necessárias para o atendimento dos requisitos propostos.

Com o objetivo de permitir o entendimento do desenvolvimento do sistema, o próximo capítulo aborda os detalhes de implementação deste.

5 *Detalhes de Implementação*

5.1 Introdução

Neste capítulo são mostrados os detalhes mais relevantes na implementação do sistema. Para tal, serão apresentados as principais tecnologias de apoio, o ambiente de desenvolvimento e aspectos de desenvolvimento.

5.2 Tecnologias de Apoio

O desenvolvimento da aplicação tratada neste trabalho demandou a utilização de um *hardware* específico e alguns módulos de *software*. Esta seção tem o objetivo de apresentar estas ferramentas, de modo a facilitar o entendimento global do sistema.

5.2.1 Hardware

A ideia de utilizar uma câmera para interagir com um computador já rendeu o desenvolvimento de diversos *softwares*, como o CameraMouse (CAMERAMOUSE, 2011) e o Nouse (NOUSE, 2011). Estes sistemas, têm o objetivo de permitir com que pessoas com restrições corporais, como uma lesão na coluna, por exemplo, utilizem o computador através da movimentação da face. No entanto, esse tipo de interação pode vir a ser cansativo e não natural para uma pessoa com total capacidade física (GELB; SUBRAMANIAN; TAN, 2010).

Por esse motivo, o interesse pela utilização de gestos corporais, mais especificamente dos realizados com as mãos, cresceu largamente. Esse indicativo pode ser visualizado a partir do crescimento do número de artigos publicados nessa área. Grande parte destes, abordam metodologias para detecção e reconhecimento das mãos, visando uma interação mais precisa e eficaz com o ambiente computacional.

Diversos trabalhos abordam o reconhecimento da mão sem aparatos, dentre eles podemos citar os trabalhos em (CHOI; SEO; PARK, 2009; LOPES, 2005; TRUYENQUE, 2005; O'HAGAN; ZELINSKY; ROUGEAUX, 2002; YIN; GUO; XIE, 2001; YUAN et al., 2008). Estes trabalhos, utilizam métodos diversos para realizar detecção das mãos, cada um com seus pontos positivos e negativos. No entanto, a maioria deles sofre com o mesmo problema: exigem um alto processamento computacional (BRADSKI, 1998). Isso acontece porque a imagem é considerada como uma coleção de pixels com valores de cores e/ou intensidade luminosa, logo, o processamento de cada pixel consome muito tempo e pode ser impreciso devido às variações de cor e luminosidade do ambiente (BRADSKI, 1998), demandando uma alta carga de processamento.

A imprecisão gerada pela variação de cores também prejudica a necessidade de compatibilidade com múltiplas cores de pele e independência de luminosidade. Por este motivo, diversas técnicas de visão computacional têm sido desenvolvidas e aprimoradas. Por exemplo, Viola e Jones (VIOLA; JONES, 2001) introduziram um método para detectar objetos de forma precisa e rápida chamado *Haar Classifiers*. Para realizar a classificação do objeto, é utilizado um algoritmo de aprendizado para selecionar um certo número de características que são combinadas em um classificador muito eficiente.

Paralelamente, o avanço dos dispositivos periféricos, trouxe ao mercado novas possibilidades em relação ao reconhecimento de gestos. Uma das recentes novidades, é o reconhecimento através dos dados relacionados com a profundidade do ambiente (GELB; SUBRAMANIAN; TAN, 2010).

Logo, o surgimento e popularização de sensores, que permitem ao computador possuir dados da profundidade do ambiente, trás um novo universo de possibilidades, já que além da facilitação do reconhecimento de gestos, pode-se também realizar a interação no espaço tridimensional. Algo que era possível somente com a utilização de equipamentos como luvas com a posição controlada por sensores ou combinações de duas ou mais câmeras (BENKO; WILSON, 2009).

Portanto, a posição das mãos pode ser extraída com o auxílio de um sensor de profundidade, popularmente tratado como *depth camera*, o qual mede o tempo de alcance de uma fonte de luz Infra Vermelha (IR - *InfraRed*) pulsada, determinando assim a distância de cada pixel contido em sua visualização (GELB; SUBRAMANIAN; TAN, 2010).

Por ser um dispositivo caro, as câmeras com recursos para medição de profundidade possuem atualmente usos limitados. No entanto, os custos vem caído rapidamente. O primeiro sensor desse tipo, dedicado para a venda em larga escala, é o sensor Kinect (MICROSOFT, 2011b), desenvolvido para ser um controle para o console XBox da Microsoft®.

Levando em consideração os recursos e possibilidades da utilização de um sensor de profundidade, adotou-se o dispositivo Microsoft® Kinect, o qual diferentemente de outros sensores, pode ser encontrado facilmente no mercado brasileiro com um preço que pode ser considerado acessível em comparação com outros equipamentos.

O Kinect é o resultado da aquisição pela Microsoft® da tecnologia 3DV para reconhecimento baseado em câmera, apesar de ser desenvolvido para o console XBox, da Microsoft®, ele utiliza o padrão USB (*Universal Serial Bus*) como interface de entrada e saída de dados.

Logo, para realizar a utilização do Kinect em um computador comum, bastou a construção de um *driver*, que conseguisse interpretar e solicitar dados através de sua interface USB.

Dessa forma, a comunidade de desenvolvedores interessada nesse tipo de dispositivo, passou a contribuir para o desenvolvimento dos drivers para o Kinect, dando origem ao projeto OpenKinect (OPENKINECT, 2011), que visa a criação de bibliotecas e aplicações de código aberto que utilizem o potencial do referido dispositivo.

Além da disponibilização da biblioteca *opensource* do grupo OpenKinect, a empresa PrimeSense, que é a companhia responsável pelo *hardware* do Kinect, em conjunto com o grupo Willow Garage¹ e a empresa Side-Kick², liberaram a versão oficial para sistemas operacionais Windows e Linux (Ubuntu 10.10 ou superior) dos *drivers* *opensource* para o Kinect em meados de Dezembro de 2010. Portanto, quem já estava trabalhando com a tecnologia tem a possibilidade de desenvolver suas aplicações com a aprovação e suporte do fabricante.

Os drivers fornecidos permitem acesso às informações provindas dos sensores de áudio, vídeo e profundidade do Kinect através de um conjunto de APIs conhecidas como OpenNI (*Open Natural Interaction*) e do *middleware* NITE da empresa PrimeSense, os quais serão discutidos posteriormente.

¹O grupo Willow Garage desenvolve soluções de *hardware* e software para aplicações pessoais de robótica.

²Side-kick é uma empresa desenvolvedora de jogos que utilizam gestos e movimentos como interface.

Desta forma, juntamente com o dispositivo Kinect, optou-se pela utilização da API OpenNI e do *middleware* NITE, devido sua maturidade e os recursos já disponíveis.

5.2.2 Software

As ferramentas de software discriminados aqui, são compostas basicamente de APIs e bibliotecas utilizadas para o estabelecimento de comunicação com o dispositivo Kinect, reconhecimento de gestos e renderização 3D. É importante ressaltar, que o trabalho foi desenvolvido de forma a utilizar ferramentas de software que fossem compatíveis tanto com sistemas operacionais Windows, quanto Linux, visando a portabilidade da aplicação desenvolvida para estes SOs.

OpenNI

OpenNI é um framework multiplataforma com suporte a múltiplas linguagens de programação que define APIs para a escrita de aplicações as quais utilizam interação natural, ou seja, fazem uso da voz ou gestos para a interação com sistemas computacionais. Seu principal propósito é construir uma API padrão para comunicação com sensores de vídeo e áudio, e com *middleware* que realize percepção de visão e audição (OPENNI, 2011).

Uma característica importante, consiste no fato de que as APIs fornecidas pelo OpenNI permitem ao desenvolvedor realizar o escaneamento de cenas tridimensionais do mundo real, através dos tipos de dados gerados a partir da entrada do sensor (o Kinect, por exemplo), podendo obter a representação de um corpo, a localização de uma mão ou até uma matriz de pixels representando um mapa de profundidades da cena. Além disso, todas essas operações, podem ser realizadas de forma independente do sensor a ser utilizado e até mesmo do *middleware*, facilitando muito a reutilização do código.

A Figura 5.1 mostra uma visualização em três camadas do conceito utilizado pelo OpenNI, cada camada tem o seguinte significado:

- **Aplicação:** representa o software que implementa aplicações que fazem uso de gestos e voz para a interação;
- **OpenNI:** representa o OpenNI, provendo interfaces de comunicação com os sensores e os componentes de *middleware*, que analisam dados dos sensores;

- **Dispositivo:** representa o dispositivo de *hardware* que captura dados de áudio e vídeo da cena.

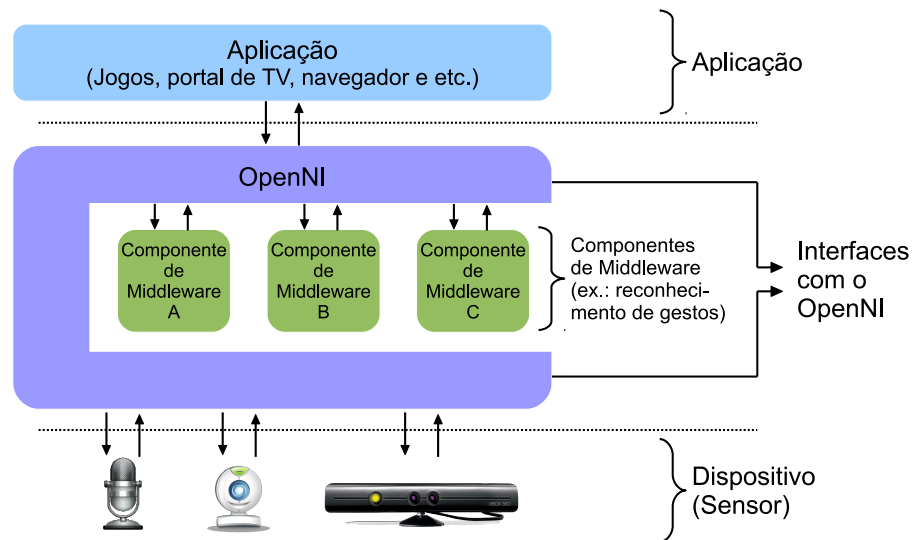


Figura 5.1: Abstração das camadas do framework OpenNI (adaptado de (OPENNI, 2011)).

Com o estudo das características do OpenNI, pôde-se concluir que ele atenderia às necessidades do projeto e que, portanto, seria utilizado no desenvolvimento do sistema, no papel de oferecer suporte à utilização do *middleware* NITE, da PrimeSense, abordado a seguir.

PrimeSense NITE

A utilização do dispositivo Kinect, demandou a adoção de ferramentas de desenvolvimento que tornassem possível a utilização de seus recursos. O *middleware* NITE, da PrimeSense, tem como função prover uma interface de alto nível para a comunicação com dispositivo, tendo como grande vantagem a possibilidade de sua utilização com outros dispositivos que possuam recursos similares, sem a necessidade de grandes modificações na aplicação (PRIMESENSE, 2011b).

O *middleware* NITE inclui métodos para processamento de dados de vídeo, inclusive com o recurso do mapeamento da profundidade da cena, contendo uma infraestrutura de algoritmos capaz de realizar identificação de pessoas, características de objetos em geral e reconhecimento de gestos, permitindo ao desenvolvedor criar aplicações controladas por gestos, encapsulando diversos algoritmos de processamento de imagens que são utilizados largamente em aplicações de visão computacional.

Além disso, por fazer parte da arquitetura de desenvolvimento de aplicações da PrimeSense, o *middleware* possui alta eficiência em termos de performance, robustez e taxas de detecção (PRIMESENSE, 2011a).

Em adição, através de uma metodologia orientada a eventos, o NITE possui a capacidade de capturar gestos pré-definidos, como uma ação de clique, em que o usuário deve aproximar a mão da interface de captura, e até mesmo gestos definidos e construídos pelo desenvolvedor.

Para efetuar a comunicação com o dispositivo Kinect e a criação de eventos em geral, o *middleware*, fornece funções integradas com as APIs do OpenNI.

As características descritas para o *middleware* em questão foram consideradas de suma importância, já que, além de desejar uma aplicação com boas taxas de reconhecimento e recursos avançados, não se pode esquecer que uma aplicação de RA ainda necessita de renderizar gráficos, que podem ser complexos. Portanto, apesar da detecção e reconhecimento de gestos constituírem uma parte importante do processo execução, estes não podem ser ineficientes a ponto de prejudicar a resposta visual (renderização dos gráficos), dada ao usuário.

OGRE3D

Visando uma maior portabilidade e eficiência, a aplicação mostrada neste trabalho foi desenvolvida em linguagem C++. Dessa forma, na fase de projeto foi necessário definir como os gráficos da aplicação seriam gerados.

A partir da escolha da linguagem C++, a renderização dos gráficos poderia ser realizada através da utilização das bibliotecas OpenGL (OPENGL, 2011) ou DirectX (MICROSOFT, 2011a). Como se deseja a portabilidade entre sistemas operacionais Windows e Linux, inicialmente, pode-se descartar a biblioteca DirectX já que ela não possui compatibilidade nativa com sistemas Linux.

Pensando que o desenvolvimento utilizando OpenGL exige uma programação de baixo nível, podendo demandar uma baixa taxa de produtividade, utilizou-se a estratégia de utilizar um motor de renderização, que pudesse abstrair as necessidades de renderização em baixo nível, provendo APIs que tornassem o desenvolvimento mais ágil e eficaz.

Dessa forma, optou-se pela utilização do Engine OGRE (*Object-Oriented Graphics Rendering Engine*) (OGRE3D, 2011) que é um motor de renderização orientado a ce-

nas e escrito em C++. A biblioteca OGRE abstrai os detalhes da utilização de bibliotecas como DirectX e OpenGL, provendo classes de alto nível para a renderização de cenas e objetos 3D. Outro ponto de interesse, consiste no fato da biblioteca ser gratuita e *opensource*, possuindo uma vasta e ativa comunidade (KERGER, 2010).

Um dos pontos interessantes da biblioteca OGRE é a possibilidade de programar utilizando o paradigma orientado a objetos operações que, caso fossem utilizadas diretamente bibliotecas como o OpenGL e o Direct3D, deveriam ser realizadas de forma procedural, em que se deve se preocupar com todo o fluxo de renderização, que vai desde a criação da geometria, passagens de informações para diversas APIs de renderização até a chegada da informação de renderização no processador (JUNKER, 2006).

Logo, a metodologia de geração do ambiente 3D baseada na orientação de objetos permite que não seja necessário tratar com a geometria, suas características e renderização. Cada forma passa a ser tratada como um objeto, ou seja uma instância de uma classe, que compõe o cena. Portanto, não é necessário efetuar chamadas a APIs de renderização, basta manipular os objetos da cena de acordo com seus métodos, o que consiste em uma forma mais intuitiva e produtiva de trabalhar com ambientes 3D (JUNKER, 2006).

Pode-se observar que o sistema de renderização da OGRE3D abstrai as necessidades das APIs 3D, tornando mais produtivo o desenvolvimento e portátil, à medida que a aplicação desenvolvida pode trabalhar com a API 3D disponível na máquina. Isso permite que, por exemplo, seja utilizada a mesma aplicação compilada para Windows com renderização através da biblioteca Direct3D e compilada para Linux, utilizando OpenGL (JUNKER, 2006).

Além disso, a biblioteca OGRE, é focada na utilização de padrões de projeto que permitem uma maior extensibilidade e usabilidade (JUNKER, 2006). Por isso, existem diversos módulos que podem ser facilmente adicionados ao ambiente de desenvolvimento com o intuito de prover, por exemplo, o suporte a simulações de física.

Apesar dos diversos recursos oferecidos, neste trabalho o único módulo adicional ao OGRE utilizado nas interfaces foi a biblioteca para geração de menus e interfaces CEGUI (*Crazy Eddies GUI System*). Essa biblioteca gratuita disponibiliza a funcionalidades como janelamento e widgets para APIs gráficas ou motores de renderização, sendo orientada a objetos e escrita em C++ (TURNER, 2011).

5.3 Ambiente de Desenvolvimento

Em tempo de projeto, foi determinado que a aplicação desenvolvida neste trabalho deveria ser compatível com sistemas Windows e Linux. Dessa forma, todos os módulos de *software* utilizados devem possuir versões compatíveis com estes sistemas operacionais.

Considerando que o OpenNI e o NITE possuem suporte a Windows e Linux, através da programação em linguagem C++, foi necessário escolher uma metodologia de renderização dos elementos virtuais. Dessa forma, o motor de renderização OGRE (OGRE3D, 2011) foi escolhido, devido suas características, dentre elas a possibilidade do desenvolvimento em linguagem C++.

Portanto, tem-se a garantia que todos os elementos de *software* utilizados para o desenvolvimento são compatíveis tanto para SOs Windows, quanto para o Linux, demandando uma fase de instalação de pré-requisitos e compilação da aplicação.

Apesar da compatibilidade das bibliotecas utilizadas com SOs Windows e Linux, a compilação em cada ambiente exige a instalação de bibliotecas específicas para o SO e arquitetura da máquina, estes pré-requisitos consistem em:

- PrimeSense Nite - versão 1.3 ou superior;
- OpenNI - versão do repositório de 10 de janeiro de 2011, ou superior;
- Ogre3D - versão 1.7 ou superior;
- CEGUI - versão 0.7.5 ou superior.

É importante informar que na maioria dos casos, é necessário efetuar a compilação das bibliotecas informadas, as quais possuem seus próprios pré-requisitos que devem ser respeitados e também instalados.

Quanto ao *hardware*, pode-se considerar que a configuração apesar de possuir algumas características especiais (como uma placa de vídeo dedicada), não fica longe dos equipamentos que são largamente utilizados no Brasil. Os equipamentos utilizados para desenvolvimento e teste foram:

- Notebook: Processador Intel®Core™2 Duo T6670 @ 2.20GHz, memória RAM: 4GB, placa de vídeo integrada;

- PC: Placa mãe Asus P5K PremiumTM, processador Intel[®]CoreTM 2 Duo E4700 @ 2.60GHz, memória RAM: 4GB, placa de vídeo: ATITM Radeon HD 3450.

5.3.1 Interface Gráfica

Um dos pontos principais da criação da interface gráfica, consistiu na forma da visualização do ambiente de RA por parte do usuário, pensando nisso, foi convencionado que o sistema utilizaria como forma de visualização do ambiente aumentado a configuração baseada em monitor (Figura 5.2), em que um monitor é utilizado para mostrar a cena com elementos virtuais. Neste caso, a câmera e o monitor são orientadas para o usuário. Assim, este visualiza o sistema de RA como se estivesse vendo uma reflexão de sua imagem misturada com os objetos virtuais.



Figura 5.2: Exemplo de configuração baseada em monitor para o sistema de RA proposto.

A partir desta definição, observa-se que o usuário deverá interagir com o sistema como se estivesse olhando para um espelho, que possui a imagem misturada (mundo real + objetos virtuais).

Partindo dessa premissa, e considerando que a interface com o usuário seria renderizada através da utilização do motor de renderização OGRE3D. A cena que compõe o ambiente de RA foi construída com os seguintes componentes:

- Câmera: é o objeto responsável por gerar a visualização da cena, devido a escolhas de projeto, a câmera fica fixa em uma posição do espaço;
- Painéis: são componentes que são renderizados em uma camada superior a todos os objetos visíveis da cena, sendo utilizados para mostrar informações

como a taxa de quadros por segundo e a existência de usuários na frente do sensor;

- Menus: também renderizados em uma camada superior a de todos os objetos da cena, porém, criados a partir da biblioteca CEGUI. Têm a função de permitir a criação de objetos e a execução de certas funções no sistema;
- Imagem do ambiente real: consiste em um objeto 2D da cena que faz o mapeamento da imagem RGB capturada do sensor, sendo posicionada na primeira camada da renderização e atualizada a cada novo quadro recebido;
- Objetos 3D: formas, primitivas e modelos 3D adicionados à cena em tempo de execução.

As Figuras 5.3 e 5.4 mostram a constituição da interface gráfica.



Figura 5.3: Elementos da interface gráfica (painéis e objeto 3D).

Na Figura 5.3, pode-se observar que, em camadas superiores à imagem RGB do sensor (mundo real), são adicionados os elementos da cena, sendo que:

1. Consiste em um painel para visualização da taxa de quadros por segundo (atual, média melhor e pior), número de triângulos renderizados na cena e o número de elementos que compõe a cena;
2. Consiste em um painel para visualização dos usuários que podem interagir com o sistema, neste caso, temos dois usuários, representados pelas cores vermelho e laranja;

3. Modelo 3D criado e posicionado na cena.

Na Figura 5.4, é possível ter a visualização de um menu, identificado com o marcador contendo o número quatro.



Figura 5.4: Elementos da interface gráfica (menu).

Para o desenvolvimento da interface mostrada, foi necessário efetuar a integração dos módulos de renderização e de captura de movimentos e imagens do sistema. Dessa forma, a cada quadro ou gesto realizado, os elementos gráficos são modificados de acordo com os eventos lançados a partir do módulo de comunicação com o sensor e reconhecimento de gestos, que serão abordados a seguir.

5.4 Comunicação com o Sensor e Detecção de Gestos

A obtenção dos dados do sensor e reconhecimento dos usuários e gestos realizados são uma parte de suma importância do sistema, visto que esta é a base para todos os outros processos.

Levando em conta as características do OpenNI, qualquer dispositivo de rastreamento 3D poderia ser conectado à aplicação, portanto, a inicialização deste envolve uma fase de determinação de suas capacidades. Para isso, inicialmente, é necessário definir um arquivo XML (*Extensible Markup Language*) que determina quais recursos serão utilizados pela aplicação. Este arquivo pode ser observado no fragmento de código 5.1.

Código 5.1: Arquivo XML de configuração da aplicação.

```

<OpenNI>
2     <ProductionNodes>
        <Node type="Depth">
4            <Configuration>
                <MapOutputMode xRes="640" yRes="480" FPS="
6                    30" />
                <Mirror on="true" />
            </Configuration>
8        </Node>
        <Node type="Image">
10            <Configuration>
                <MapOutputMode xRes="640" yRes="480" FPS="
12                    30" />
                <Mirror on="true" />
            </Configuration>
14        </Node>
        <Node type="Gesture" />
16        <Node type="Hands" />
        <Node type="User" />
18    </ProductionNodes>
</OpenNI>

```

Observa-se que a configuração mostrada anteriormente (Código 5.1), é composta por uma TAG “*ProductionNodes*” e uma série de TAGs “*Nodes*”, os quais compõe os serviços que serão utilizados na aplicação.

Os *Production Nodes* são definidos como um conjunto de serviços necessários para a construção de aplicações que utilizem interação natural (OPENNI, 2011). A biblioteca OpenNI conta com os seguintes *Production Nodes*:

- *Device*: representa um dispositivo físico (um sensor de profundidade, uma câmera RGB, etc.). É utilizado para habilitar a possibilidade de realizar configurações no dispositivo;
- *Depth Generator*: este nó gera o mapa de profundidade da cena;
- *Image Generator*: gera o mapa de imagem em formato RGB;
- *IR Generator*: provê a imagem da cena a partir da visualização do sensor Infravermelho do dispositivo;
- *Audio Generator*: gera uma stream de audio do ambiente;

- *Gestures Alert Generator*: cria chamadas para a aplicação quando determinados gestos são realizados;
- *Scene Analyzer*: realiza a análise da cena, incluindo a separação do fundo da mesma, sua principal saída é o mapa de profundidade;
- *Hand Point Generator*: dá suporte à detecção e rastreamento da mão;
- *User Generator*: gera uma representação do corpo humano, ou parte dele na cena 3D.

Cada *Production Node* pode conter suas respectivas configurações. Por exemplo, no arquivo XML mostrado os nós “*Depth*” e “*Image*”, os quais são responsáveis por prover o mapa de profundidade e a imagem RGB da cena, estipula-se parâmetros para resolução da imagem capturada, taxa de quadros por segundo.

Para a aplicação abordada neste trabalho, também são definidos os nós “*Gesture*”, “*Hands*” e “*User*”, que são responsáveis por prover a captura de gestos, da posição da mão e do usuário, respectivamente.

O Código 5.2 mostra como o arquivo XML descrito é carregado no programa e como a inicialização dos nós é realizada. Logo, a partir da inicialização, os nós descritos no XML são validados através da macro “*VALIDATE_GENERATOR*”, a qual verifica se o tipo de nó requerido está disponível para o dispositivo (neste caso o Kinect) conectado ao sistema. Caso algum dos nós não esteja disponível, é mostrada uma mensagem de erro e a aplicação é finalizada.

Código 5.2: Fragmento de código responsável pela inicialização do sensor.

```
// Inicializando o OpenNI a partir do XML
2 XnStatus rc = XN_STATUS_OK;
  rc = m_Context.InitFromXmlFile("TrackingConfig.xml");
4 CHECK_RC(rc, "InitFromXml");

6 // Verificando se os nos do OpenNI estao disponiveis
  VALIDATE_GENERATOR(XN_NODE_TYPE_DEPTH, "Depth", m_DepthGenerator);
8 VALIDATE_GENERATOR(XN_NODE_TYPE_IMAGE, "Image", m_ImageGenerator);
  VALIDATE_GENERATOR(XN_NODE_TYPE_USER, "User", m_UserGenerator);
10 VALIDATE_GENERATOR(XN_NODE_TYPE_HANDS, "Gesture", m_GestureGenerator);
  VALIDATE_GENERATOR(XN_NODE_TYPE_HANDS, "Hands", m_HandsGenerator);
12

  // Inicializando o middleware NITE
14 m_pSessionManager = new XnVSessionManager;
```

```

rc = m_pSessionManager->Initialize (&m_Context, "Wave", "RaiseHand");
16 [...]

```

Após o início do módulos do OpenNI, necessita-se de inicializar o *middleware* NITE, este processo é mostrado na linha 15 do Código 5.2, é criada uma chamada de função para o início da sessão, especificando como sinal para tal o movimento “wave”.

Com a confirmação da possibilidade da utilização dos recursos (profundidade da cena, imagem RGB, reconhecimento de usuários, determinação da posição da mão e reconhecimento de gestos) cada um dos respectivos objetos se torna um canal de comunicação que possibilita o acesso e manipulação dos dados do sensor.

Dessa forma, a atualização da imagem RGB, mostrada no fundo da cena, é efetuada através do objeto *m_ImageGenerator*, que fornece o mapa RGB da imagem capturada pelo Kinect. Da mesma forma, os procedimentos para utilização do posição da mão, utilização do mapa de profundidade e detecção de usuários são efetuados com o auxílio dos métodos contidos nos respectivos objetos.

O reconhecimento dos gestos e atualização da posição da mão é efetuado através no mecanismo de eventos da classe *XnVSessionManager*. Com o início da sessão é possível monitorar a ocorrência de ações do usuário através de classe relacionadas ao tipo de ação desejada e a *callbacks* definidos para tratar esses eventos.

O Código 5.3 exemplifica a definição de dois *listeners* para ações distintas: monitorar a posição da mão e aguardar um gesto circular. Partindo do pré-suposto que o usuário pode realizar qualquer um dos dois gestos, é necessário garantir que a aplicação monitore a ocorrência de ambos simultaneamente, isso é realizado através do distribuidor de eventos, mostrado na linha 9. Este objeto é uma instância da classe *XnVBroadcaster* e tem a função de distribuir para todas as instâncias atreladas a ele as informações passadas pelo objeto da sessão.

Código 5.3: Exemplo de objetos para monitoramento de gestos.

```

// Inicializando o middleware NITE
2 m_pSessionManager = new XnVSessionManager;
rc = m_pSessionManager->Initialize (&m_Context, "Wave", "RaiseHand");
4 [...]
m_pSessionManager->RegisterSession (this, SessionStarting, SessionEnding,
FocusProgress);
6 [...]

```

```

8 // Emissor de eventos principal
   broadcaster_main = new XnVBroadcaster;
10 m_pSessionManager->AddListener(broadcaster_main);

12 // Objeto para verificacao da posição da mão
   m_pointControl = new XnVPointControl;
14
   // Callback para atualizacao da posicao da mao
16 m_pointControl->RegisterPointUpdate(this, &OnPointUpdate);

18 // Adicionando o novo objeto ao emissor de eventos
   broadcaster_main->AddListener(m_pointControl);
20
   [...]
22
   // Objeto para verificacao do gesto circulo
24 circleDetector_createMainMenu = new XnVCircleDetector;

26 // Callback para atuacao na ocorrencia do gesto circulo
   circleDetector_createMainMenu->RegisterCircle(this, &CircleCB);
28
   // Adicionando o novo objeto ao emissor de eventos
30 broadcaster_main->AddListener(circleDetector_createMainMenu);

```

Considerando ainda o Código 5.3, para monitorar uma determinada ação, é necessário criar instâncias das classes relacionadas aos gestos a serem monitorados (linhas 13 e 24), adicionar os respectivos *callbacks* (linhas 16 e 24) e, por fim, informar ao distribuidor de eventos que os objetos criados estão aguardando informações (linhas 19 e 30).

Os *callbacks* por sua vez tem a função de tratar o evento recebido e efetuar as ações necessárias. O Código 5.4 tem o objetivo de mostrar o tratamento dado quando se tem um evento de nova posição da mão. Observa-se que, através dos métodos disponibilizados nas APIs do NITE e das classes do sistema desenvolvido, este *callback* efetua a modificação da posição de um elemento da cena de acordo com a atualização da movimentação da mão.

Código 5.4: Exemplo de *callback* para processamento da atualização da posição da mão.

```

void XN_CALLBACK_TYPE OnPointUpdate(const XnVHandPointContext* pContext,
    void* cxt)
2 {

```

```
4      // Objeto gerenciador dos Gestos
      HandController* This = (HandController*)cxt;

6      // Adquirindo os pontos x, y e z da mao
      XnPoint3D ptProjective(pContext->ptPosition);

8      // Definindo a nova posicao do objeto

10     Ogre::SceneNode* node = This->getSceneManager()->getSceneNode(This
        ->nodeToMove);
        node->setPosition(ptProjective.X, ptProjective.Y, ptProjective.Z);

12 }
```

Além dos gestos mostrados anteriormente, o *middleware* NITE suporta a criação de gestos através de programação em C++ e a utilização de formas de controle pré-definidas, são elas:

- *Push Detector*: verifica se ocorreu um movimento de aproximação e afastamento do sensor;
- *Swipe Detector*: apura se ocorreu um leve movimento para qualquer direção sucedido de uma parada;
- *Steady Detector*: é ativado se a mão ficar parada;
- *Wave Detector*: verifica se a mão efetuou quatro trocas de direção dentro de um espaço de tempo pré-determinado;
- *CircleDetector*: aguarda que o usuário efetue um movimento circular completo com a mão;
- *SelectableSlider1D*: implementa um controle deslizante unidimensional (x ou y ou z);
- *SelectableSlider2D*: implementa um controle deslizante bidimensional (x-y);

Através do mecanismo de reconhecimento da posição da mão e dos métodos de controle contidos no *middleware* NITE, foi desenvolvida a metodologia de interação com o ambiente de RA, que será abordado na próxima seção.

5.5 Interação com o Usuário

O captura da posição da mão é um pré-requisito para a realização de gestos. Logo, a operação de inicialização da posição da mão deve ser efetuada toda vez que o usuário iniciar o sistema. Caso a posição de sua mão seja perdida, seja por uma oclusão ou pela ausência de um usuário no campo de visão do sensor, deve-se repetir o gesto de inicialização.

Essa operação de inicialização, é feita através de um movimento chamado “wave”, que requer que o usuário efetue, horizontalmente, pelo menos quatro mudanças de direção. A Figura 5.5, ilustra este procedimento.

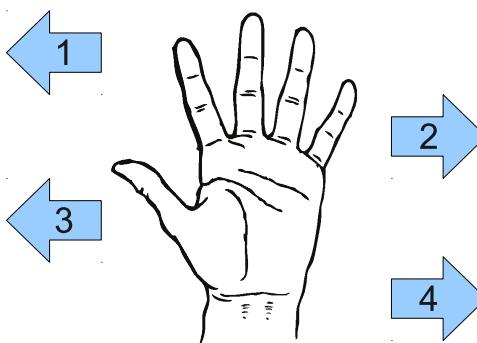


Figura 5.5: Movimentação para inicialização da posição da mão (wave).

É importante observar, que este gesto foi utilizado, em primeiro lugar, devido sua complexidade, visto que, em uma situação normal, dificilmente o usuário irá realizar este gesto involuntariamente e, em segundo lugar, pela analogia que se pode realizar ao imaginar que ao balançar a mão, o usuário estaria “chamando a atenção” do sistema, facilitando assim o aprendizado deste gesto.

Quando o gesto de inicialização é efetuado de forma válida, é adicionada na cena uma esfera vermelha, que acompanhará a mão do usuário, mostrando a posição da mesma no espaço, como é mostrado na Figura 5.6.

5.5.1 Navegação em Menus

Considerando que aplicações de propósito geral necessitam constantemente de realizar ajustes, acessar opções, ou até mesmo objetos à cena, a definição de uma boa metologia para acesso a menus é de grande importância. Logo, para a navegação em menus, foram convencionados três gestos/movimentos:



Figura 5.6: Indicação da detecção iniciada.

- entrada no menu;
- movimentação entre as opções do menu;
- seleção de um item de menu.

Considerando que um gesto só pode ser reconhecido se o movimento de inicialização foi executado, a entrada no menu é efetuada através da realização de um círculo completo com a mão que está sendo mapeada, conforme exemplificado na Figura 5.7.

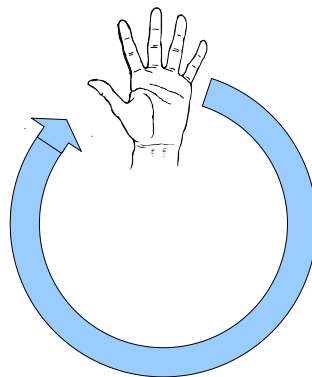


Figura 5.7: Movimentação para inicialização de menus (círculo).

Com a visualização do menu, o usuário poderá navegar através dos itens movimentando a mão horizontalmente, como se estivesse apontando para o item, como é mostrado na Figura 5.8.

Por fim, para selecionar um item, basta aproximar e afastar a mão da tela, como se estivesse efetuando um clique.

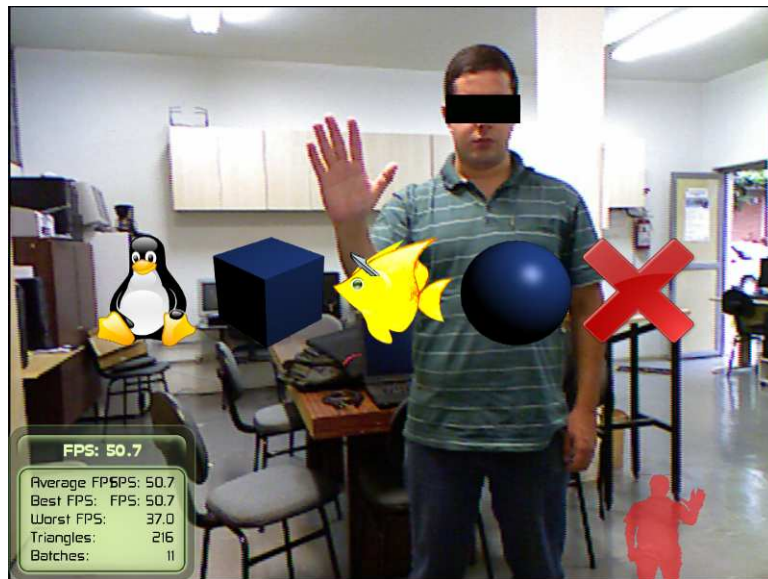


Figura 5.8: Navegação no menu.

5.5.2 Seleção de Objetos Virtuais

No desenvolvimento do projeto, convencionou-se que para selecionar um objeto virtual na cena, o usuário deve manter a mão sobre o mesmo por pelo menos três segundos. Dessa forma, o objeto será marcado através da exibição das arestas de um cubo que contém o objeto em questão e será exibido um menu de ações (escala, translação, rotação, apagar ou desistir) para o objeto selecionado (Figura 5.9).

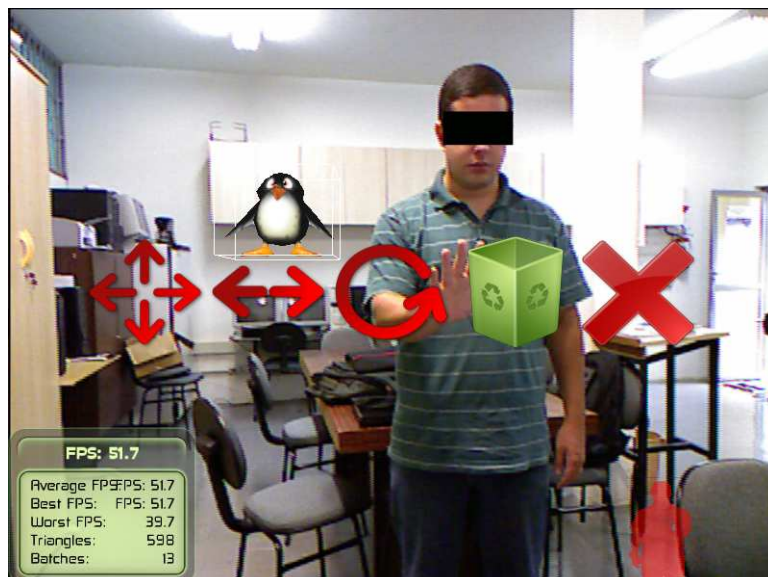


Figura 5.9: Objeto selecionado e menu de operações.

5.5.3 Translação, Rotação e Escala

A partir da seleção do objeto, o usuário pode escolher qual transformação deseja realizar sobre o mesmo, através do menu mostrado na Figura 5.9.

Caso a opção selecionada seja a translação, o objeto acompanhará a posição da mão no espaço, assumindo a cada momento uma posição nova (Figura 5.10). A posição definitiva será atribuída quando a mão do usuário permanecer parada por cinco segundos, conforme convenção de projeto.



Figura 5.10: Procedimento de translação.

Se a opção for rotação a transformação será executada através da movimentação vertical, horizontal e em profundidade da mão, sendo efetuada nos eixos X, Y e Z, conforme a variação da posição da mão ao longos dos eixos cartesianos. A Figura 5.11 apresenta quadros de uma operação de rotação.



Figura 5.11: Procedimento de rotação.

Caso a opção seja escala, será efetuada a escala nos três eixos, proporcionalmente, mediante o movimento horizontal da mão, como pode ser observado na Figura 5.12.



(a) Escala inicial.



(b) Escala intermediária.



(c) Objeto fixado com a nova escala.

Figura 5.12: Procedimento de escala.

5.6 Considerações Finais

Este capítulo apresentou os detalhes de implementação do sistema desenvolvido, mostrando algumas escolhas de projeto, como bibliotecas e o ambiente de trabalho.

Abordou-se também aspectos de desenvolvimento da interface gráfica da aplicação, bem como os procedimentos utilizados para detecção de gestos, comunicação com o sensor Kinect e interação com o usuário.

Com a finalidade de condensar a contribuição deste trabalho, o próximo capítulo trata dos resultados obtidos.

6 *Análise dos Resultados*

6.1 Introdução

Este capítulo tem o objetivo de mostrar os resultados obtidos com o desenvolvimento do sistema abordado neste trabalho.

Para tal, discute-se a forma com a qual foram atendidas as necessidades mostradas na Tabela 3.2, que são:

- Detecção da mão nua;
- Compatibilidade com múltiplas cores de pele;
- Suporte a variações de luz;
- Determinação do gesto realizado;
- Independência do ambiente de trabalho.

Adicionalmente, para cumprir requisitos de uma aplicação de Realidade Aumentada, podemos elencar outras necessidades gerais:

- Atuação em objetos virtuais em tempo real;
- Geração de um *feedback* visual para as operações realizadas.

Visando facilitar o entendimento, a tratativa dada a esses problemas será mostrada e discutida com o auxílio de imagens e resultados de testes realizados.

6.2 Escolha dos Módulos do Sistema

Uma das grandes dificuldades da aplicação proposta neste trabalho consiste na detecção das mãos e gestos de uma maneira rápida, eficaz, com boa taxa de acertos

independente de variações de luz e do ambiente de trabalho, atendendo, ainda, o requisito de execução tempo real, necessário para garantir a imersão do usuário no ambiente de RA.

É importante lembrar que a preocupação com o tempo se torna um ponto chave à medida que além da detecção e reconhecimento, a execução do programa também envolve a renderização dos gráficos da cena.

Estes aspectos, exigiram a investigação de métodos e possibilidades existentes os quais pudessem solucionar esse problema de forma eficiente e que cumprisse os objetivos propostos. Dessa forma, foi efetuado um estudo acerca das ferramentas de *hardware* e *software* que poderiam servir de apoio no desenvolvimento deste trabalho, abordado no Capítulo 5.

Neste sentido, observa-se que a escolha do dispositivo Kinect (MICROSOFT, 2011b), juntamente com o *framework* OpenNI (OPENNI, 2011) e o *middleware* NITE (PRIME-SENSE, 2011b), teve uma contribuição muito importante no que diz respeito à qualidade da detecção e compatibilidade com múltiplas cores de pele. Isto deve às características, tanto de *hardware*, quanto de *software* destes módulos.

Como relatado nos capítulos anteriores, o processamento de imagens RGB é trabalhoso e instável, à medida que a cor dos pixels da imagem varia com a mudança das condições de luminosidade da cena e que o ambiente se torna complexo, ou seja, composto por elementos que podem se confundir com o objeto que se deseja detectar.

Levando isso em conta, a utilização do Kinect trás grandes possibilidades. Sua capacidade de capturar informações do ambiente real através de um sensor infravermelho, permite extrair tanto dados sem a interferência da luz, quanto informações sobre a profundidade do cenário de trabalho, os quais consistem em informações muito ricas da cena e que podem ser utilizadas em aplicações de detecção de características do objeto.

O *framework* OpenNI define APIs para aplicações que fazem uso de gestos para a interação. Provendo métodos para a conexão de dispositivos de rastreamento tridimensional com aplicações que promovem interação via gestos. Logo, sua utilização provê uma forma padronizada para o desenvolvimento, tornando o sistema a ser desenvolvido capaz de se conectar com diversos tipos de *hardware* com as mesmas capacidades, ou seja, possibilita a substituição do Kinect por outro dispositivo semelhante caso necessário.

A utilização das APIs do OpenNI cria a necessidade da utilização ou desenvolvimento de um *middleware* que tenha a inteligência necessária para efetuar a comunicação, processamento dos dados e lançamento de eventos para a aplicação. Neste ponto, optou-se por utilizar o *middleware* NITE, o qual possui métodos padronizados para acesso ao dispositivo, além da implementação de metodologias de reconhecimento de características de objetos largamente investigadas em aplicações de visão computacional, largamente testadas e de alta eficiência.

Portanto, observa-se que estes componentes do sistema, consistem em partes fundamentais, fornecendo dados para os módulos de tratamento de eventos e renderização, responsável por atuar no ambiente de RA e gerar o *feedback* visual para o usuário.

6.3 Detecção da Mão Nua e Compatibilidade com Múltiplas Cores de Pele

A utilização das ferramentas de *hardware* e *software* descritas anteriormente, possibilitou o reconhecimento da mão sem aparatos e a compatibilidade com diversas cores de pele.

Um ponto importante da arquitetura, consiste no *middleware* NITE que trás consigo a implementação de diversos métodos de visão computacional, entre eles a possibilidade de reconhecimento de objetos de acordo com suas características.



Figura 6.1: Exemplo de reconhecimento prévio de usuários.

Dessa forma, quando uma pessoa se apresenta na frente do dispositivo, já é inicializado um processo, que marca a área ocupada por essa pessoa com um certo identificador (Figura 6.1). Com um gesto de inicialização, o sistema adquire as coordenadas da mão da pessoa que está atuando e passa a seguir sua posição no am-

biente tridimensional, já que a capacidade de reconhecimento de profundidade está disponível.

Com o objetivo de efetuar testes de qualidade de reconhecimento, este foi testado com pessoas de cores de pele distintas (amarela, branca, negra e parda), a título de exemplo a Figura 6.2 mostra dois voluntários de sexo e cor de pele distintos.



(a) Voluntário do sexo masculino com pele negra.



(b) Voluntário do sexo feminino com pele branca.

Figura 6.2: Amostras de testes com pessoas de cores de peles distintas.

É importante observar que as imagens da Figura 6.2 exibem círculos vermelhos mostrando um *feedback* visual para a posição da mão do usuário no ambiente, isso significa que o sistema conseguiu reconhecer a mão de um usuário com o tipo de pele relacionado. Nota-se que o círculo vermelho não está exatamente no centro da mão do usuário e que a distância para o centro pode variar conforme a posição do mesmo. Isso acontece porque a posição da mão capturada nos dados de profundidade da cena não é, necessariamente, igual à posição vista na representação RGB. Além disso, é necessário levar em conta que o círculo é desenhado do espaço tridimensional e transportado para o espaço bidimensional, podendo também causar distorções nesta distância.

Nestes testes, foi avaliada a possibilidade de interação e as taxas de acertos, obtendo-se taxas semelhantes para todas as cores de peles testadas.

6.4 Suporte a Variações de Luz e Independência do Ambiente de Trabalho

A utilização de um dispositivo de entrada que proporciona a captura de dados da cena através de um sensor de infra-vermelho, trás a vantagem da independência, por

parte do módulo de reconhecimento, da intensidade da luz no ambiente de trabalho.

Além disso, a possibilidade de gerar um mapeamento de profundidade da cena, trás a opção de eliminar partes indesejáveis desta, trabalhando somente em faixas de profundidade interessantes para a aplicação.

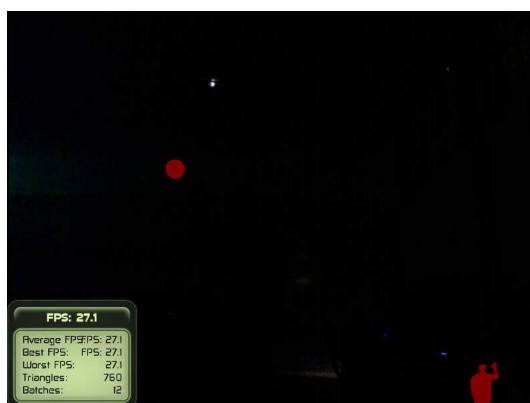
Dessa forma, a utilização de um sensor com as capacidades do Kinect, ou seja, mapeamento tridimensional da cena através de um sensor infra-vermelho, reduz de forma significativa a dependência da luminosidade e do ambiente de trabalho. A Figura 6.3 exibe algumas amostras da utilização da aplicação com diferentes taxas de luminosidade e, em adição, a Figura 6.4 mostra a utilização em ambientes complexos.



(a) Ambiente com luz natural.



(b) Ambiente com luz clara.



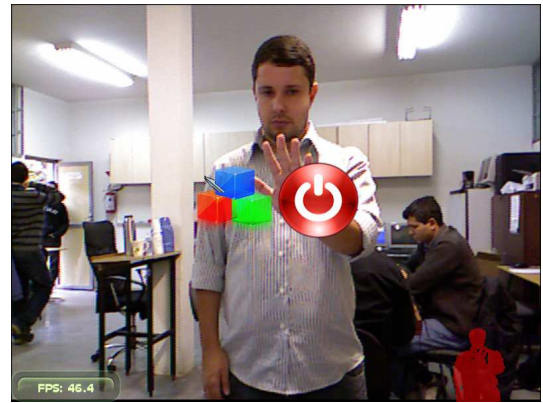
(c) Ambiente escuro.

Figura 6.3: Amostras de testes com variação de luminosidade.

Observa-se que a interação não foi prejudicada de forma perceptível nas condições apresentadas, no entanto, um aspecto que se deve levar em conta consiste no fato de que, à medida que o ambiente é escurecido, a imagem RGB utilizada para formar o *feedback* visual para o usuário é prejudicada, sendo, portanto, necessário definir um nível mínimo de luminosidade, o qual não foi aplicado neste trabalho, visto que, a intenção deste era realizar a interação nestas condições.



(a) Ambiente complexo com uma pessoa na cena.



(b) Ambiente complexo com mais de uma pessoa na cena.

Figura 6.4: Amostras de testes em ambientes complexos.

6.5 Determinação do Gesto Realizado

A utilização do *framework* OpenNI acrescido no *middleware* NITE trouxe formas para reconhecimento de gestos integradas com as capacidades discutidas anteriormente.

Para isso, estes módulos de *software*, possuem mecanismos de eventos a partir dos quais é possível detectar gestos pré estabelecidos nestas APIs, conforme, mostrado no Capítulo 5.

Logo, a aplicação desenvolvida neste trabalho, utiliza o reconhecimento dos seguintes gestos:

- *Wave*: inicialização da posição da mão;
- *Círculo*: chamada do menu principal;
- *Clique*: seleção de elementos de menu.

Associada a esses gestos a posição da mão no espaço, permitiu a interação com os objetos virtuais.

6.6 Atuação no Ambiente de RA e Visualização da Cena Aumentada

Como a aplicação em questão consiste em um ambiente de Realidade Aumentada, é de suma importância fornecer um *feedback* visual, em tempo real, para o usuário. Neste sentido, foi desenvolvida a integração dos eventos lançados pelo OpenNI com uma camada de processamento da ação e renderização, responsável por modificar e renderizar os objetos virtuais e exibir essas alterações no dispositivo utilizado para visualização, que neste caso consiste em uma tela de projeção ou um monitor funcionando como um espelho contendo a imagem aumentada (Figura 6.5).



Figura 6.5: Usuário utilizando o sistema.

Para tal, foi utilizado o ambiente de renderização do Ogre3D, acrescido de modificações para a exibição da imagem RGB capturada pelo Kinect e o tratamento de eventos tanto de gestos, como de colisão (seleção dos objetos).

Em termos de qualidade de renderização e taxa de quadros por segundo, pode-se considerar os resultados mais que satisfatórios, visto que, todo o conjunto, desde a captura dos dados do dispositivo até a atuação, não acusou taxas menores que 25 FPS.

6.7 Considerações Finais

Afim de captar dados acerca da qualidade do reconhecimento efetuado nas condições propostas, e da satisfação com a interação com o ambiente de RA, foram avaliados os seguintes itens para cada usuário que testou o sistema:

- Reconhecimento da Pele:
 - Níveis de reconhecimento:
 - * Não reconheceu a pele;
 - * Muitas perdas de posição da mão;
 - * Poucas perdas de posição da mão;
 - * Nenhuma perda de posição da mão.
- Satisfação do usuário:
 - Níveis de satisfação:
 - * Insatisfeito;
 - * Pouco satisfeito;
 - * Parcialmente satisfeito;
 - * Satisfeito;
 - * Muito satisfeito.

Os níveis de reconhecimentos foram aferidos segundo a utilização no ambiente com pouca iluminação e iluminação normal e separados pelas cores dos usuários. A quantidade de usuários e suas respectivas cores de pele estão relacionados na Tabela 6.1.

Tabela 6.1: Quantidade de pessoas que participaram dos testes e suas respectivas cores de pele.

Cor da Pele	Quantidade
Amarela	2
Branca	5
Negra	2
Parda	2

Os gráficos mostrados nas Figuras 6.6 e 6.7, mostram, respectivamente, as considerações sobre o reconhecimento em ambientes com iluminação normal e baixa.

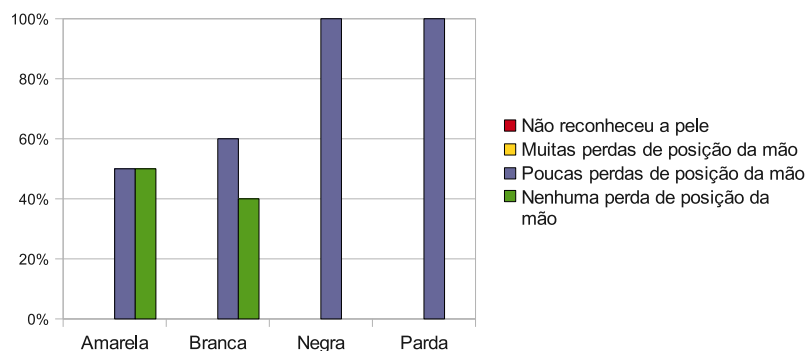


Figura 6.6: Níveis de reconhecimento - ambiente claro.

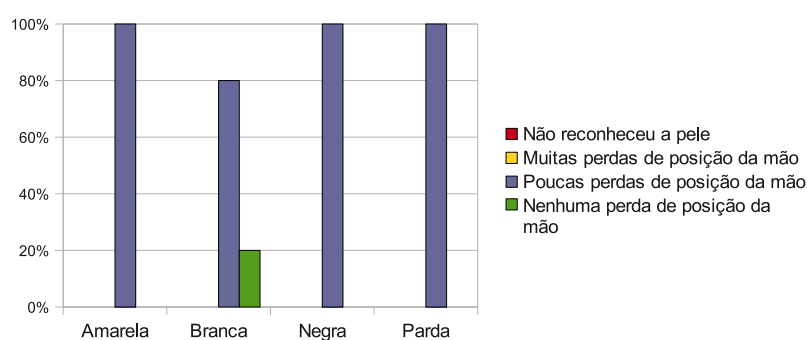


Figura 6.7: Níveis de reconhecimento - ambiente escuro.

Observa-se que a diferença entre os dados é mínima, devido à utilização dos módulos descritos anteriormente.

A Figura 6.8, mostra a satisfação dos usuários em relação à interação promovida. É importante ressaltar a importância de realizar testes de usabilidade mais completos, no entanto, estes não eram objetivo deste trabalho.

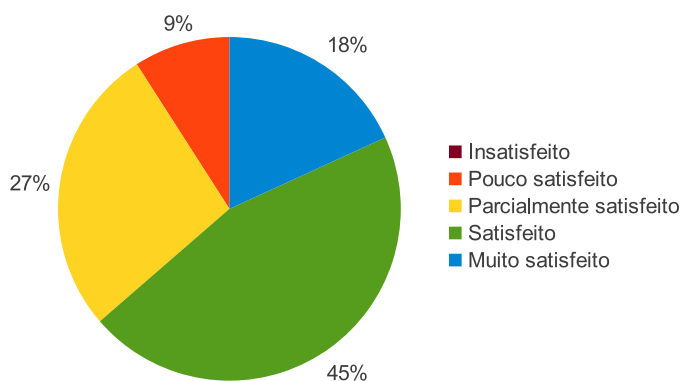


Figura 6.8: Nível de satisfação do usuário.

Por fim, pode-se observar a contribuição deste trabalho, através da comparação com as pesquisas relacionadas no estudo do estado da arte, mostrada na Tabela 6.2.

Tabela 6.2: Comparação das características analisadas nos trabalhos relacionados.

Trabalhos Relacionados	Deteção da mão nua	Compatível com múltiplas cores de pele	Suporta variações de luz	Determinação do gesto realizado	Independente do ambiente de trabalho
Finger Tracking for Interaction in Augmented Environments	✗	✗	✓	✓	✗
Assembly Design and Evaluation based on Bare-Hand Interaction in an Augmented Reality Environment	✓	✗	✓	✓	✗
Manipulation of Virtual Objects in Marker-less AR System by Fingertip Tracking and Hand Gesture Recognition	✓	✓	✗	✓	✗
FingARtips: Gesture Based Direct Manipulation in Augmented Reality	✗	✗	✗	✓	✗
Opportunistic Tangible User Interfaces for Augmented Reality	✓	✗	✗	✗	✓
Manipulação de Objetos 3D em Aplicações de Realidade Aumentada por Meio da Movimentação da Mão	✓	✓	✓	✓	✓

7 *Conclusões e Trabalhos Futuros*

7.1 Introdução

Este capítulo tem como objetivo relatar os principais pontos estudados nesta dissertação, apresentar possíveis trabalhos futuros e avaliar a contribuição científica deste trabalho.

7.2 Conclusões

O objetivo deste trabalho consistiu em investigar formas de interação natural e utilizá-las em uma aplicação de Realidade Aumentada.

A partir do estudo do estado da arte, percebeu-se que a utilização da mãos, através do posicionamento e gestos, consiste em uma interface intuitiva e cognitivamente vantajosa para utilização. Logo, determinou-se alguns pontos principais de trabalho, como o reconhecimento da mão nua (sem aparatos) em ambientes não controlados e suportando diversas cores de pele.

Desta forma, este trabalho, apresentou o desenvolvimento de uma aplicação que provê ao usuário a possibilidade de interagir com elementos virtuais, através da Realidade Aumentada, utilizando as mãos, sem a necessidade de utilizar qualquer aparato acoplado ao corpo. Tendo também como características, a baixa dependência da luminosidade e do ambiente de trabalho para o reconhecimento da mão do usuário e gestos realizados.

Para tal, foi utilizado o dispositivo Kinect, juntamente com o *framework* OpenNI e o *middleware* NITE, os quais consistem em módulos de *software* de uso específico para aplicações com interação via gestos. A exibição da imagem aumentada foi realizada através da implementação de um módulo de tratamento de eventos, atuação e renderização, que utilizou como motor de renderização a biblioteca Ogre3D.

A aplicação foi desenvolvida em linguagem c++, com bibliotecas compatíveis para sistemas operacionais Windows e Linux, visando a possibilidade de utilização nestes sistemas, necessitando para isso da compilação da aplicação no SO em questão.

Quanto ao desempenho, o conjunto apresentou boas taxas de quadros por segundo, que não foram menores que 25 FPS. Isso se deve a um conjunto de fatores, que vão desde a linguagem escolhida, até a eficiência e capacidade dos módulos de *software/hardware* implementados e utilizados.

Sendo assim, considerou-se que os objetivos traçados foram cumpridos de forma satisfatória.

7.3 Trabalhos Futuros

Considerando que a aplicação desenvolvida neste trabalho, apesar de simplificada, mostra um grande universo de possibilidades, no que diz respeito ao desenvolvimento da interação em aplicações de RA, como trabalhos futuros, pretende-se aplicar os conhecimentos e recursos desenvolvidos em aplicações específicas que venham atender a demandas sociais e/ou científicas.

Neste âmbito, pode-se trabalhar no sentido de transformar os pontos chave da aplicação gerada neste trabalho em uma biblioteca, que poderia ser utilizada por pesquisadores interessados em desenvolver aplicações de RA com as características apresentadas neste trabalho.

Além disso, torna-se necessária a realização de testes de usabilidade detalhados, visando aprimorar os conceitos de interação natural aplicados à Realidade Aumentada.

Por fim, considera-se de suma importância para a sensação de imersão, melhorar a forma de visualização do ambiente aumentado, para tal, pode-se efetuar a integração do sistema com um HMD (*Head Mounted Display*).

Referências bibliográficas

- AZUMA, R. et al. Recent advances in augmented reality. *IEEE Computer Graphics and Applications*, IEEE Computer Society, p. 34–47, 2001.
- AZUMA, R. T. A survey of augmented reality. *Presence-Teleoperators and Virtual Environments*, Citeseer, 6, n. 4, p. 355–385, 1997.
- BENKO, H.; WILSON, A. *DepthTouch: Using depth-sensing camera to enable freehand interactions on and above the interactive surface*. [S.l.], 2009.
- BMW. *BMW UK Web Site*. Jan 2010. Disponível em: <<http://www.bmw.co.uk/>>.
- BRADSKI, G. R. Computer vision face tracking for use in a perceptual user interface. *Intel Technology Journal*, Citeseer, 1998.
- BROLL, W. et al. ARTHUR: A collaborative augmented environment for architectural design and urban planning. *Journal of Virtual Reality and Broadcasting*, 1, n. 1, p. 1–10, 2004.
- BROLL, W. et al. An infrastructure for realizing custom-tailored augmented reality user interfaces. *IEEE transactions on visualization and computer graphics*, Citeseer, 11, n. 6, p. 722–733, 2005.
- BUCHMANN, V. et al. FingARtips: gesture based direct manipulation in Augmented Reality. In: *GRAPHITE '04: Proceedings of the 2nd international conference on Computer graphics and interactive techniques in Australasia and South East Asia*. New York, NY, USA: ACM, 2004. p. 212–221. ISBN 1-58113-883-0.
- CAMERAMOUSE. *Camera Mouse Web Site*. Jan 2011. Disponível em: <<http://cameramouse.org/>>.
- CHOI, J.; SEO, B. K.; PARK, J. I. Robust hand detection for augmented reality interface. p. 319–321, 2009.
- DIX, A.; FINLAY, J.; ABOWD, G. D. *Human-computer interaction*. [S.l.]: Prentice hall, 2004. ISBN 0130461091.
- DORFMÜLLER, K.; WIRTH, H. Real-Time Hand and Head Tracking for Virtual Environments Using Infrared Beacons. In: *CAPTECH '98: Proceedings of the International Workshop on Modelling and Motion Capture Techniques for Virtual Environments*. London, UK: Springer-Verlag, 1998. p. 113–127. ISBN 3-540-65353-8.
- DORFMÜLLER-ULHAAS, K.; SCHMALSTIEG, D. Finger Tracking for Interaction in Augmented Environments. In: *ISAR '01: Proceedings of the IEEE and ACM International Symposium on Augmented Reality (ISAR'01)*. Washington, DC, USA: IEEE Computer Society, 2001. p. 55. ISBN 0-7695-1375-1.

- FERNANDES, B.; FERNÁNDEZ, J. Using Haar-like Feature Classifiers for Hand Tracking in Tabletop Augmented Reality. *XII Symposium on Virtual and Augmented Reality*, p. 6–13, 2010.
- FIALA, M. Artag, an improved marker system based on artoolkit. *NRC Canada, Publication Number: NRC, 47419*, 2004.
- FIALA, M. *Documentação do ARTag*. Jan 2010. Disponível em: <<http://www.artag.net/>>.
- FILHO, C. F. *História da Computação: O caminho do pensamento e da tecnologia*. [S.l.]: EDIPUCRS, 2007.
- FISCHER, J.; BARTZ, D.; STRASSER, W. Intuitive and Lightweight User Interaction for Medical Augmented Reality. In: *Proceedings of Vision, Modeling and Visualization (VMV)*. [S.l.: s.n.], 2005. p. 375–382.
- GELB, D.; SUBRAMANIAN, A.; TAN, K. H. *Augmented Reality for Immersive Remote Collaboration*. [S.l.], 2010.
- GILLEANES, T. *UML - Uma Abordagem Prática 2*. [S.l.]: Novatec, 2009.
- HENDERSON, S.; FEINER, S. Opportunistic Tangible User Interfaces for Augmented Reality. *IEEE Transactions on Visualization and Computer Graphics*, IEEE Educational Activities Department, Piscataway, NJ, USA, 16, n. 1, p. 4–16, 2010. ISSN 1077-2626.
- HENDERSON, S. J.; FEINER, S. Evaluating the benefits of augmented reality for task localization in maintenance of an armored personnel carrier turret. In: *ISMAR '09: Proceedings of the 2009 8th IEEE International Symposium on Mixed and Augmented Reality*. Washington, DC, USA: IEEE Computer Society, 2009. p. 135–144. ISBN 978-1-4244-5390-0.
- INSLEY, S. Obstacles to general purpose Augmented Reality. *ECE 399H, Information Security & Cryptography, Oregon, EUA, dezembro de*, 2003.
- ISHII, H.; ULLMER, B. Tangible Bits: Towards Seamless Interfaces between People, Bits and Atoms. In: *CHI '97: Proceedings of the SIGCHI conference on Human factors in computing systems*. New York, NY, USA: ACM, 1997. p. 234–241. ISBN 0-89791-802-9.
- JUNKER, G. *Pro OGRE 3D programming*. [S.l.]: Apress, 2006. ISBN 1590597109.
- KATO, H. et al. *Documentação do ARToolKit*. Jan 2010. Disponível em: <<http://www.hitl.washington.edu/artoolkit/>>.
- KERGER, F. *Ogre 3D 1.7 - Create real-time 3D applications using Ogre 3D*. [S.l.]: Packt, 2010. ISBN 978-1-849512-48-0.
- KIRNER, C.; SISCOOTTO, R. A. Fundamentos de Realidade Virtual e Aumentada. *Realidade Virtual e Aumentada: Conceitos, Projeto e Aplicações*, p. 9–21, 2007.

KRAHNSTOEVEER, N. et al. A Real-Time Framework for Natural Multimodal Interaction with Large Screen Displays. In: *Proceedings of the 4th IEEE International Conference on Multimodal Interfaces*. Washington, DC, USA: IEEE Computer Society, 2002. (ICMI '02), p. 349. ISBN 0-7695-1834-6.

LEE, B.; CHUN, J. Manipulation of virtual objects in marker-less AR system by fingertip tracking and hand gesture recognition. In: *ICIS '09: Proceedings of the 2nd International Conference on Interaction Sciences*. New York, NY, USA: ACM, 2009. p. 1110–1115. ISBN 978-1-60558-710-3.

LIMA, A. da S. *UML 2.0 - Do Requisito à Solução*. 4. [S.l.]: Érica, 2010.

LIVINGSTON, M. A. et al. An augmented reality system for military operations in urban terrain. 2002, 2002.

LOPES, E. C. Determinando a Posição e a Orientação da Mão Através de Imagens de Vídeo. 2005.

MCDONALD, C. *Hand interaction in augmented reality*. Tese (Doutorado) — Carleton University, 2003.

MICROSOFT. *DirectX Web Site*. Jan 2011. Disponível em: <<http://msdn.microsoft.com/en-us/directx/default>>.

MICROSOFT. *Microsoft Kinect Web Site*. Jan 2011. Disponível em: <<http://www.xbox.com/en-US/kinect>>.

NOGUEIRA, K. L. *O Uso de Técnicas de Realidade Virtual e Aumentada na Simulação de Prótese de Membros Superiores*. Dissertação (Mestrado) — Universidade Federal de Uberlândia, 2007.

NOUSE. *Nouse Web Site*. Jan 2011. Disponível em: <<http://www.nouse.ca/>>.

NVIS Inc. *NVIS Inc. Web Site*. Jan 2010. Disponível em: <<http://nvisinc.com>>.

OGRE3D. *Ogre3D Web Site*. Jan 2011. Disponível em: <<http://www.ogre3d.org/>>.

O'HAGAN, R. G.; ZELINSKY, A.; ROUGEAUX, S. Visual gesture interfaces for virtual environments. *Interacting with Computers*, Elsevier, 14, n. 3, p. 231–250, 2002.

OPENGL. *OpenGL Web Site*. Jan 2011. Disponível em: <<http://www.opengl.org/>>.

OPENKINECT. *OpenKinect Comunity Site*. Jan 2011. Disponível em: <<http://openkinect.org/>>.

OPENNI. *OpenNI User Guide*. [S.l.], Jan 2011.

PALOC, C. et al. Computer-Aided Surgery based on Auto-Stereoscopic Augmented Reality. In: *IV '04: Proceedings of the Information Visualisation, Eighth International Conference*. Washington, DC, USA: IEEE Computer Society, 2004. p. 189–193. ISBN 0-7695-2177-0.

Portuguesa de Desportos. *Portuguesa de Desportos Web Site*. Jan 2010. Disponível em: <<http://www.portuguesa.com.br>>.

- PREECE, J. et al. *Human-Computer Interaction*. [S.l.]: Addison-Wesley, England, 1994.
- PRIMESENSE. *NITE Controls 1.3 - Programmer's Guide*. [S.l.], Jan 2011.
- PRIMESENSE. *PrimeSense Nite Web Site*. Jan 2011. Disponível em: <<http://www.primesense.com/?p=515>>.
- SCHWALD, B.; LAVAL, B. D. An augmented reality system for training and assistance to maintenance in the industrial context. *Journal of WSCG*, Citeseer, 11, n. 1, p. 425–432, 2003.
- SEO, B.-K. et al. One-Handed Interaction with Augmented Virtual Objects on Mobile Devices. In: *VRCAI '08: Proceedings of The 7th ACM SIGGRAPH International Conference on Virtual-Reality Continuum and Its Applications in Industry*. New York, NY, USA: ACM, 2008. p. 1–6. ISBN 978-1-60558-335-8.
- SILVA, M. D. et al. Uma proposta de middleware para distribuição de Realidade Aumentada pela web como ferramenta de apoio para educação a distância. *WRVA 2009*, 2009.
- SOMMERVILLE, I. *Engenharia de Software, 8a edição*. [S.l.]: Pearson Addison-Wesley, São Paulo, Brasil, 2007.
- TORI, R.; KIRNER, C. Fundamentos de Realidade Aumentada. In: _____. [S.l.: s.n.], 2006. (Livro do pré-simpósio - VIII Symposium on Virtual Relity), cap. 1.
- TORI, R.; KIRNER, C. Fundamentos de Realidade Virtual. In: _____. [S.l.: s.n.], 2006. (Livro do pré-simpósio - VIII Symposium on Virtual Relity), cap. 1.
- TORI, R.; KIRNER, C.; SISCOOTTO, R. *Fundamentos e Tecnologia de Realidade Virtual e Aumentada*. [S.l.]: SBC, 2006.
- TRUYENQUE, M. A. Q. Uma Aplicação de Visão Computacional que Utiliza Gestos da Mão para Interagir com o Computador. 2005.
- TURNER, P. D. *CEGUI Web Site*. Jan 2011. Disponível em: <<http://www.cegui.org.uk/>>.
- VALLI, A. The Design of Natural Interaction. In: _____. Hingham, MA, USA: Kluwer Academic Publishers, 2008. 38, n. 3, p. 295–305. ISSN 1380-7501.
- VALLINO, J. *Interactive Augmented Reality*. Tese (Doutorado) — Department of Computer Science, University of Rochester, Abril 1998.
- VIOLA, P.; JONES, M. Rapid Object Detection using a Boosted Cascade of Simple Features. *Computer Vision and Pattern Recognition, IEEE Computer Society Conference on*, IEEE Computer Society, Los Alamitos, CA, USA, 1, p. 511, 2001. ISSN 1063-6919.
- WANG, Z. et al. Assembly Design and Evaluation Based on Bare-Hand Interaction in an Augmented Reality Environment. In: UGAIL, H. et al. (Ed.). *CW*. [S.l.]: IEEE Computer Society, 2009. p. 21–28. ISBN 978-0-7695-3791-7.

- WELCH, G.; BISHOP, G. An introduction to the Kalman filter. *University of North Carolina at Chapel Hill, Chapel Hill, NC*, Citeseer, 1995.
- WINDER, J.; TONDEUR, P. *Papervision3D Essentials*. [S.l.]: Packt Publishing, 2009.
- YIN, X.; GUO, D.; XIE, M. Hand image segmentation using color and RCE neural network. *Robotics and Autonomous Systems*, Elsevier, 34, n. 4, p. 235–250, 2001.
- YIN, Y.; DAVIS, R. Toward natural interaction in the real world: real-time gesture recognition. In: *International Conference on Multimodal Interfaces and the Workshop on Machine Learning for Multimodal Interaction*. New York, NY, USA: ACM, 2010. (ICMI-MLMI '10), p. 15–1. ISBN 978-1-4503-0414-6.
- YUAN, M. et al. Robust hand tracking using a simple color classification technique. In: *VRCAI '08: Proceedings of The 7th ACM SIGGRAPH International Conference on Virtual-Reality Continuum and Its Applications in Industry*. New York, NY, USA: ACM, 2008. p. 1–5. ISBN 978-1-60558-335-8.
- ZORZAL, E. R. et al. Realidade Aumentada Aplicada em Jogos Educacionais. V *Workshop de Educação em Computação e Informática do Estado de Minas Gerais*, 2006.