FUNDAÇÃO DE ENSINO “EURÍPIDES SOARES DA ROCHA”
CENTRO UNIVERSITÁRIO “EURÍPIDES DE MARÍLIA” - UNIVEM
PROGRAMA DE MESTRADO EM CIÊNCIA DA COMPUTAÇÃO

ELAINE PARRA AFFONSO

UM SUPORTE À INTERAÇÃO REMOTA PARA SISTEMAS DE REALIDADE AUMENTADA

Marília
2006

ELAINE PARRA AFFONSO

UM SUPORTE À INTERAÇÃO REMOTA PARA SISTEMAS DE REALIDADE AUMENTADA

Dissertação apresentada ao Programa de Mestrado do Centro Universitário de Eurípides de Marília, mantido pela Fundação de Ensino Eurípides Soares da Rocha, para obtenção do Título de Mestre em Ciência de Computação.

Orientador:
Prof. Dr. Antonio Carlos Sementille

Marília
2006

AFFONSO, Elaine Parra
Um Suporte à Interação Remota para Sistemas de Realidade Aumentada / Elaine Parra Affonso; orientador: Antonio Carlos Sementille. Marília, SP: [s.n], 2006.
130 f.

Dissertação (Mestrado em Ciência da Computação) – Centro Universitário Eurípides de Marília, Fundação de Ensino Eurípides Soares da Rocha.

1. Realidade Virtual 2. Realidade Aumentada

CDD: 006

ELAINE PARRA AFFONSO

UM SUPORTE À INTERAÇÃO REMOTA PARA SISTEMAS DE REALIDADE AUMENTADA

Banca examinadora da dissertação apresentada ao Programa de Mestrado da UNIVEM, /F.E.E.S.R., para obtenção do Título de Mestre em Ciência da Computação.

A Comissão Julgadora:

Prof. Dr. Antonio Carlos Sementille

Prof. Dr. Ildeberto Aparecido Rodello

Prof. Dr. Aparecido Nilceu Marana

Marília, 25 de Outubro de 2006.

Dedico este trabalho aos meus pais, Osvaldo e Iolanda, ao meu namorado Edgar, e ao meu irmão Clayton, que não pouparam esforços para que durante esse período de estudo eu fosse uma pessoa feliz.

AGRADECIMENTOS

A Deus, que, com sua infinita bondade, enviou-me coragem e iluminação para atingir meus objetivos.

Aos meus pais, Osvaldo e Iolanda, pelo incentivo incondicional demonstrado nos momentos mais difíceis e sempre me oferecendo apoio incansável.

Ao meu namorado Edgar, pela imensa compreensão e prova diária de amor, obrigada pela confiança depositada em mim.

Ao meu irmão Clayton, que mesmo distante sempre colaborou com sua amizade, estando de alguma forma presente em minha vida.

Ao meu orientador, Prof. Dr. Antonio Carlos Sementille, pela sua orientação, compreensão e paciência para comigo nos momentos difíceis, e colaboração durante todo o desenvolvimento desse trabalho.

À minha amiga Cristiane Lucy, pela amizade e companhia nos estudos durante o percorrer desse mestrado.

Às minhas amigas, Elisandra e Cristiane Fernandes, pelo incentivo e compreensão de minha ausência nos diversos momentos.

Aos meus tios Milton e Luzia pelo acolhimento em sua casa durante o período do mestrado.

Aos meus ex-patrões, Regina Pontelli e Gilmar Pitteri, pelas dispensas nas horas de trabalho e apoio ao ingresso no mestrado.

Ao Centro Paula Souza que possibilitou que esse sonho virasse uma realidade.

Aos meus colegas de trabalho que integram o corpo docente da ETE Amim Jundi e da Faculdade da Alta Paulista (FADAP/FAP), pelo companheirismo e apoio.

Aos professores Remo e Fátima pelos ensinamentos transmitidos nas suas disciplinas específicas.

A todos vocês que me ajudaram e acreditaram em mim, o meu sincero agradecimento.

Grandes realizações não são feitas por impulso,
mas por uma soma de pequenas realizações

Vicent Van Gogh

Affonso, Elaine Parra. **Um Suporte à Interação Remota para Sistemas de Realidade Aumentada**. 2006. 130 f. Dissertação (Mestrado em Ciências da Computação) – Centro Universitário "Eurípides de Marília", Fundação de Ensino Eurípides Soares da Rocha, Marília, 2006.

RESUMO

A representação do imaginário e a reprodução do real tornaram-se mais fáceis de ser obtidas através de aplicações de Realidade Virtual. Essas aplicações permitem ao usuário a imersão, a navegação e a interação em tempo real no ambiente tridimensional gerado por computador, chamado de Ambiente Virtual. Tais ambientes permitem que os participantes se encontrem para interagir dinamicamente com outros participantes, realizando o desenvolvimento de tarefas em conjunto e que, com uso de técnicas de Realidade Aumentada, consigam combinar simultaneamente cenas reais e virtuais, possibilitando também a interação sobre objetos virtuais. É nesse contexto que se situa este trabalho, que apresenta um estudo da Realidade Aumentada Colaborativa bem como o desenvolvimento de um suporte que permita a interação remota de objetos virtuais.

Palavras-Chaves: Ambiente Virtual Colaborativo, Realidade Aumentada, Comunicação e Interação.

Affonso, Elaine Parra. **Um Suporte à Interação Remota para Sistemas de Realidade Aumentada**. 2006. 130 f. Dissertação (Mestrado em Ciências da Computação) – Centro Universitário "Eurípides de Marília", Fundação de Ensino Eurípides Soares da Rocha, Marília, 2006.

ABSTRACT

The representation of the imaginary and the reproduction of the real have become easier to be obtained through applications of Virtual Reality. These applications allow the user the immersion, the navigation and the interaction in real time in the tridimensional environment generated by computer, called virtual environment. Such environments allow that the participants meet to interact dynamically with other participants, accomplishing, the development of tasks in groups, that with the use of Augmented Reality, combine simultaneously real and virtual scenes, giving also possibilities to the interaction on virtual objects. This piece of works is placed in this context. It introduces a study of Collaborative Augmented Reality, as well the development of a support that allows the remote interaction of virtual objects.

KEYWORDS: Collaborative Virtual Environment, Augmented Reality, Collaboration, Communication and Interaction

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

2D	Bidimensional
3D	Tridimensional
ARISUPPORT	*Augmented Reality Interaction Support*
ARTOOLKIT	*Augmented Reality Toolkit*
AV	Ambiente Virtual
AVC	Ambiente Virtual Colaborativo
AVD	Ambiente Virtual Distribuído
AVI	Áudio Vídeo *Interleave*
BMP	*Bitmap*
CAVE	*Cave Automatic Virtual Environment*
CSCW	*Computer Supported Cooperative Work*
DIV	*Distributed Open Invetor*
DWARF	*Distributed Wearable Augmented Reality Framework*
GHz	*Gigahertz*
GLUI	*Graphic Library User Interface*
HHD	*Handheld Augmented Reality Display*
HMD	*Head Mounted Display*
ID	Identificação
IP	*Internet Protocol*
JPEG	*Joint Photographic Experts Library*
OCAR	*Outdoor Collaborative Augmented Reality*
OPENGL	*Open Graphic Library*
PC	*Personal Computer*
PIP	*Personal Interaction Panel*
RA	Realidade Aumentada
RV	Realidade Virtual
TCP	*Transmission Control Protocol*
TTL	*Time To Live*
UDP	*User Datagram Protocol*

USB	*Universal Serial Bus*
VA	Virtualidade Aumentada
VRML	*Virtual Reality Modeling Language*
WTK	*World Toolkit*
XML	*Extensible Markup Language*

SUMÁRIO

INTRODUÇÃO

A evolução nas tecnologias de hardware, juntamente com o avanço nas capacidades gráficas, contribuiu para o desenvolvimento de uma nova tecnologia, a Realidade Virtual (RV), que surgiu como uma nova forma de interface homem-máquina, possibilitando a navegação, imersão e interação em ambientes tridimensionais.

A RV vem ganhando novos vertentes como, por exemplo, a Realidade Aumentada (RA), que consiste em inverter a busca por ambientes virtuais, cujos limites seriam a própria realidade, enriquecendo-a com recursos virtuais, abrindo novas fronteiras, limitadas apenas pela imaginação. É como se a "Realidade Virtual" estivesse evoluindo para uma "Virtualidade Real" (KIRNER; TORI, 2004).

Enquanto a RA caracteriza-se pelo predomínio do mundo real sobre o virtual, na Virtualidade Aumentada (VA) ocorre o predomínio do virtual sobre o real .

O aumento do poder computacional, seguido de novas aplicações de RV, de constantes avanços nas tecnologias de redes e da possibilidade de colocar um ou mais usuários dentro de um ambiente tridimensional, contribuiu bastante para o desenvolvimento dos Ambientes Virtuais Distribuídos (AVDs). Esses sistemas permitem aos usuários compartilharem um mesmo espaço virtual e interagirem em tempo real, sendo que esses podem estar localizados em lugares diferentes.

Dentro desse contexto, em AVDs, os usuários podem compartilhar um mesmo espaço dimensional virtual, onde poderão se auxiliar na execução de uma determinada tarefa, baseando-se nos princípios de Trabalho Cooperativo Suportado por Computador (CSCW – *Computer Supported Cooperative Work*) e será classificado como um Ambiente Virtual Colaborativo (AVC) (WICHERT, 2002).

Os AVCs estão intimamente ligados aos AVDs, desta forma estão diretamente relacionados a rede de computadores e utiliza tecnologia de realidade distribuída para suportar

trabalho em grupo, assim, de acordo com Singhal e Zyda (1999), para o desenvolvimento deste tipo de ambiente é necessário haver uma escolha adequada do modelo de comunicação usado na aplicação, como também estabelecer um compromisso entre a latência de comunicação, largura de banda e confiabilidade do sistema .

Os AVCs permitem a interação entre participantes, possibilitando também a navegação e a imersão. Pode-se conseguir a interação em AVC utilizando tecnologia de RA para criação de aplicações locais ou remotas, de modo que uma das características da RA é a interação com os objetos virtuais, uma vez que os humanos sabem como interagir com objetos reais, como controlá-los e manipulá-los (BILLINGHURST; KATO, 1999).

Nesse contexto, o principal objetivo desta dissertação é o estudo e a implementação de um sistema que possibilite mecanismos para um suporte à comunicação entre os processos participantes de um AVD e a interação com objetos virtuais controlados a partir de técnicas de RA.

Desta forma os participantes remotos poderão interagir com objetos virtuais, transmitindo suas ações a outros participantes da aplicação por meio de comunicação *multicast* conseguindo assim melhorar a escalabilidade do ambiente distribuído e criando uma aplicação de RA distribuída.

Desenvolveu-se, então, um sistema distribuído de RA, denominado “Um Suporte a Interação Remota para Sistemas de Realidade Aumentada”.

Com o objetivo de proporcionar um melhor entendimento, este trabalho está organizado em seis capítulos, conforme a descrição a seguir:

No Capítulo 1 é realizada uma conceituação e caracterização dos AVCs, sendo discutidas a sua definição, a relação com Trabalho Colaborativo Suportado por Computador e a interação em AVCs.

O Capítulo 2 apresenta um levantamento sobre os principais sistemas de AVCs com utilização da tecnologia de RA encontrados na literatura, os quais são analisados em relação às suas características, como formas de comunicação, dispositivos utilizados, interação, entre outras e, ao final, uma comparação entre os mesmos.

Na seqüência, o Capítulo 3 mostra as considerações de projeto para desenvolvimento de uma aplicação de AVC voltado à utilização de RA, incluindo protocolos a serem utilizados, modelos de comunicação, enfatizando a importância do modelo *multicast* para aplicações distribuídas e ferramentas para a criação de RA.

O Capítulo 4 apresenta uma visão geral da arquitetura proposta, bem como as funcionalidades oferecidas, e a implementação da aplicação de RA distribuída.

O ambiente de trabalho, os principais recursos de *hardware* e *software* utilizados, bem como os testes realizados para cada interação e a análise dos resultados obtidos, são descritos no Capítulo 5.

Por fim são apresentadas as conclusões, contribuições, pesquisas futuras e as referências que foram utilizadas neste trabalho.

CAPÍTULO 1 - AMBIENTES VIRTUAIS COLABORATIVOS

Os AVCs se destacam por oferecer um espaço digital no qual os participantes podem compartilhar informações, permitir novas formas de interação múltiplas e o desenvolvimento de tarefas em conjunto de maneira coordenada.

Com relação aos AVCs serão analisadas suas principais características, incluindo seus componentes básicos, formas de interação, suas necessidades e dificuldades. Será dado um enfoque aos ambientes de trabalho cooperativo suportado por computador (CSCW) que tem se tornado cada vez mais importantes nesse tipo de atividade colaborativa.

1.1 Ambientes Virtuais (AV)

O AV pode ser considerado um cenário tridimensional onde os usuários de um sistema de RV podem navegar e interagir dinamicamente, de forma que os cenários modificam-se em tempo real à medida que vai ocorrendo a interação com o ambiente. Um AV pode ser projetado para simular tanto um ambiente imaginário quanto um ambiente real (REBELO; PINHO, 2004).

O AV pode ser representado de várias formas, através de prédios, terrenos, carros e pode não ter nenhuma referência física, ser um modelo abstrato. Para ter uma boa visualização é necessário levar em consideração alguns atributos como cores, texturas, iluminação, características dinâmicas, atributos acústicos e outros (KIRNER; TORI, 2004).

São também importantes em um AV questões como a entrada de um usuário e a saída do sistema. Os sinais de entrada do usuário consistem no rastreamento da posição e orientação de sua cabeça e mãos. Os sinais de saída representam a reação, aspectos visuais e sonoros, atuando paralelo com o mundo virtual (KIRNER; TORI, 2004).

Também denominados de “mundos virtuais”, esses ambientes são implementados através de ferramentas especiais, e nesse contexto inclui-se recursos como VRML (*Virtual Reality Modeling Language*), JAVA 3D, *WorldToolkit* (WTK) e outras.

1.1.1 Ambientes Virtuais Distribuídos

Com a evolução das formas de comunicação remota, surgiu uma extensão de ambientes virtuais para atender as novas necessidades dos sistemas como, por exemplo, o Ambiente Virtual Distribuído.

AVDs são sistemas onde os usuários podem compartilhar um mesmo espaço virtual e interagir com outros usuários em tempo real, sendo que estes podem estar localizados em lugares diferentes. Os AVDs são caracterizados como um ambiente onde os usuários dispersos geograficamente têm como objetivo o compartilhamento dos recursos computacionais em tempo real, usando um suporte de redes de computadores para melhorar o desempenho coletivo por meio da troca de informações (SINGHAL; ZYDA, 1999).

1.1.2 Ambientes Virtuais Colaborativos

Um AVC é um sistema distribuído que oferece imagens tridimensionais e cenário digital. Dentro desses cenários, os participantes podem compartilhar e interagir a informação com outros participantes (CHURCHILL *et al.,* 2001).

O AVC é usado para descrever especificamente sistemas que possuem interface gráfica tridimensional para colaboração entre múltiplos usuários. Nestes sistemas um mesmo ambiente gráfico tridimensional é exibido por dois ou mais computadores, cada um deles representando a visão de um usuário imerso no ambiente. Dentro desse ambiente cada usuário

é representado para seus parceiros na forma de um avatar[1] e pode navegar e interagir com objetos do ambiente além de comunicar-se com seus parceiros (CHARÃO; KOPPER, 2004; PINHO, 2002).

Um AVC visa simular um espaço real. Esse ambiente digital permite que haja comunicação e interação entre seus participantes em um ambiente compartilhado interconectado através de redes de computadores e envolve o uso da tecnologia de realidade distribuída para suportar trabalho em grupo. Para oferecer aos usuários uma sensação de realismo, os ambientes virtuais possuem gráficos 3D e som, além de outras características desejáveis. Os usuários podem jogar, trabalhar, se tocar, conversar, compartilhar experiências, mesmo estando separados geograficamente.

O uso de computadores para melhorar a colaboração entre as pessoas tem aumentado. As ferramentas colaborativas tornam-se mais comuns que a interface homem-computador, surgindo um caminho para interface humano-humano mediado por computador. As pessoas, quando olham para uma tela de projeção ou se reúnem ao redor de uma mesa com um monitor, podem recorrer a objetos reais ou usar a comunicação natural para interagir entre si. (BILLINGHURST; KATO, 2002)

Os AVCs estão intimamente ligados aos AVDs, já que os AVCs utilizam os mesmos princípios, apenas orientando o enfoque e criando características especiais para o trabalho cooperativo. Dessa maneira, o sistema deve, a todo o momento, manter uma noção de espaço, tempo e presença compartilhada. Os usuários deverão enxergar o espaço e outros usuários em tempo real, aumentando assim o realismo e a imersão do ambiente. Além disso, é necessário que, de alguma forma, os participantes possam manipular, clicar e mover os objetos contidos no ambiente. Além da visualização, os sistemas devem ter alguma forma de comunicação

[1] Representação de um personagem na cena tridimensional

entre os participantes, como gestos, texto, voz, etc. Esta comunicação adiciona a cena à sensação de realismo.

A essência de um AVC é que os usuários são representados explicitamente um ao outro dentro de um espaço compartilhado. Os participantes devem mover-se livremente dentro desse espaço, encontrando-se com outros usuários e também objetos e informações de interesse comum.

Gouveia (2000) apresenta duas condições para que um sistema seja considerado um AVC: a existência de um acesso simultâneo a um sistema de RV e o suporte explícito das necessidades dos participantes que pretendam trabalhar em conjunto.

Os participantes, representados na forma de avatares, são livres para navegar no espaço, encontrar outros participantes se comunicarem com eles, manipular dados e objetos.

Os AVC são vistos de duas formas pelos pesquisadores e desenvolvedores (PINHO, 2002):

- Em termos de suporte, os AVCs suportam a comunicação espacial de forma natural, suporta autonomia e grande número de participantes.
- Em termos de computação, e em particular redes de computadores, os AVCs realizam conexão dinâmica, apresentam confiabilidade, consistência e escalabilidade.

1.2 Diferença entre Espaço e Lugar Compartilhado

Um espaço é o volume físico (ou digital) que pode ser uma casa, na qual os eventos podem levar a um lugar. Embora o espaço físico seja o mesmo, existem vários entendimentos de como o espaço pode ser utilizado. Dependendo da ocasião, um espaço físico pode apresentar vários lugares ao mesmo tempo. Por exemplo, uma sala de esportes pode ser um

lugar para um jogador, um lugar para um ganhador de um prêmio, um lugar para uma aula de dança (CHURCHILL *et al.,* 2001).

Existem muitos meios de interagir e comportar-se no espaço, dependendo da ocasião ou ação em que o espaço vai ser usado.

Um espaço somente torna-se um lugar quando uma ação compreendida é planejada ou está em andamento ou está associada ao modo de vida e experiência vivida.

A CAVE (*Cave Automatic Virtual Environment*) é um espaço físico do mundo real que atua como um lugar no qual o usuário pode conhecer espaços e lugares virtuais. A CAVE ilustra como o espaço e o lugar podem ser combinados ao mesmo tempo (CHURCHILL *et al.,* 2001).

Na CAVE, os usuários devem transitar ambos os espaços, o virtual e o real, e podem compartilhar as experiências com outros usuários, ao contrário dos AVCs baseados em *Head-Mounted Displays* (HMDs), no qual a interface do usuário o isola do mundo real.

Os AVCs baseados em textos podem criar espaços (salas) que têm somente um espaço textual, mas que com certeza são lugares onde as pessoas se reúnem para conversar, trabalhar ou jogar.

1.3 Trabalho Colaborativo

O Trabalho Colaborativo acontece em um contexto cooperativo que é caracterizado por participantes humanos e seus objetivos. Nesse contexto cooperativo o número de participantes pode variar, e cada participante pode estar envolvido em vários contextos cooperativos e possivelmente de forma simultânea (HOFTE, 1998).

Segundo Hofte (1998) em um contexto colaborativo, os participantes trabalham para alcançar uma meta, compartilhar uma meta, negociar, e nesse trabalho vários humanos estão

comprometidos. Para executar essas atividades, os participantes usam duas habilidades humanas fundamentais: a comunicação direta com outros participantes e a manipulação de objetos. A manipulação compartilhada de um objeto pode ser observada por outros participantes, constituindo assim uma forma indireta de comunicação entre os participantes.

Trabalho Colaborativo é aquele em que várias pessoas, local ou remotamente distribuídas, cooperam para a realização de uma mesma tarefa. Esse tipo de atividade tem se tornado cada vez mais importante e reflete a evolução das organizações. Ultimamente têm sido realizados diversos trabalhos na área que hoje recebe o nome de Trabalho Cooperativo Suportado por Computador, do inglês *Computer Supported Cooperative Work* (CSCW) (PINHO, 2002; MOECKEL, 2000).

O CSCW é um campo de pesquisa para estudar o uso de computadores para um trabalho em grupo (WICHERT, 2002).

CSCW pode ser definido como a combinação de um grupo de pessoas que utilizam a tecnologia de redes, associada a hardware, software, técnicas e serviços.

Segundo Moeckel (2000) existem algumas controvérsias em relação às definições de CSCW. Por exemplo, nem todas as aplicações desenvolvidas nessa área objetivam especificamente a realização de trabalho; algumas atividades envolvem interação social, lazer ou educação.

O controle em um sistema de CSCW deve ser sofisticado e é necessário levar em conta o papel de cada membro do grupo, para que se possam estabelecer diferentes formas de acesso. A atribuição de papéis aos usuários é uma importante característica dos sistemas de suporte ao trabalho em grupo.

CSCWs devem facilitar a colaboração entre os indivíduos, permitindo que o acesso aos dados ocorra independente da localização dos usuários, de modo que gerenciem o controle de acesso quando vários participantes tentam modificar os mesmos dados ao mesmo

tempo e garantam que as informações usadas no trabalho cooperativo sejam disseminadas entre a equipe.

Segundo Moeckel (2000), a colaboração, a troca de informação, a capacidade de comunicação, o respeito às diferenças individuais e o exercício da negociação são requisitos importantes para o trabalho cooperativo.

Com o trabalho colaborativo podem-se produzir melhores resultados do que com trabalhos individuais. Colaborando, os participantes do grupo têm a possibilidade de identificar facilmente as falhas em seu raciocínio e, juntos, podem buscar idéias e informações.

1.4 Características dos AVCs

Os AVCs apresentam características bastante interessantes no que diz respeito à comunicação, cooperação, coordenação e percepção (CHURCHILL *et al.,* 2001; FUKS *et al.*, 2002).

- **Comunicação:** Através da comunicação as pessoas compartilham idéias, discutem, negociam e tomam decisões. As informações são transmitidas através de um canal de percepção criado no espaço compartilhado onde ocorre a conversação. Por exemplo, em uma conversa face-a-face, as informações são transmitidas através do som, dos gestos e das expressões dos indivíduos, entre outros. É importante em um trabalho cooperativo ter certeza de que o participante não apenas recebeu a informação, mas também teve entendimento da mesma. Sem um entendimento compartilhado os participantes terão dificuldade em se coordenar de modo a somar seus esforços para a conclusão das tarefas.

- **Coordenação:** Para garantir a realização do trabalho colaborativo através da soma dos trabalhos individuais, é necessária a coordenação das atividades. Esta coordenação organiza o grupo para evitar que esforços de comunicação e cooperação sejam perdidos e que as tarefas sejam realizadas na ordem correta, no tempo correto e cumprindo as restrições e objetivos.
- **Cooperação:** Cooperação é a operação conjunta dos membros do grupo no espaço compartilhado. Em um espaço virtual os participantes cooperam produzindo, manipulando e organizando informações.
- **Percepção:** Perceber é adquirir informação do que está acontecendo e o que as outras pessoas estão fazendo, mesmo sem se comunicar diretamente com elas. A percepção torna-se central para a comunicação, coordenação e cooperação de um grupo de trabalho. Dessa maneira, os indivíduos percebem as mudanças causadas no ambiente pelas ações dos participantes e redirecionam as suas atitudes. Nos Ambientes Virtuais, o suporte à percepção fica menos claro, pois os meios de transmitir as informações aos seres humanos ficam restritos. Estações de trabalho típicas são limitadas a fornecer informações em uma tela com apenas duas dimensões e, em alguns casos, através de caixas de som (FUKS *et al*, 2002). Alguns exemplos de informações de percepção que podem ser providas são: o objetivo comum, o papel de cada participante dentro do contexto, como proceder, qual o impacto das ações realizadas, quem está por perto, o que as outras pessoas estão fazendo, a localização, a origem dos objetos de cooperação.
- **Contexto Compartilhado:** O contexto compartilhado é crucial para ações colaborativas e pode ocorrer de várias maneiras. Podem compartilhar o conhecimento, tanto os conhecimentos atuais como os passados, compartilhar o ambiente e compartilhar objetos. Quando os objetos são manipulados, não só eles se

tornam assunto de comunicação entre os usuários, mas também o meio de comunicação, por exemplo, se um usuário manipula um objeto, as mudanças do mesmo devem ser visíveis a outros em um processo de visualização.

- **Flexibilidade e Vários Pontos de Vista:** O desenvolvimento das atividades em um AVCs requer freqüentemente o uso de representações múltiplas, cada uma com ponto de vista diferente. As visões múltiplas em alguns casos podem ser da própria tarefa do usuário ou da tarefa de outro. No que se refere às múltiplas representações dos dados, um ambiente gráfico tridimensional oferece para cada usuário sua visão particular do cenário e dos demais usuários. Entretanto, todos vêem os dados do ambiente com o mesmo nível de detalhamento, variando apenas o ponto de vista de cada um.

Outra característica essencial dos ambientes colaborativos é a possibilidade do usuário passar de tarefas individuais a colaborativas sem muitas dificuldades e sem causar problemas para os demais usuários. Para isso é preciso que os indivíduos no ambiente colaborativo possam compartilhar um mesmo entendimento sobre as tarefas a serem realizadas e quais os objetivos de cada uma delas.

Em relação à presença de colaboradores em AVCs é observada a personificação ou a incorporação de personagens por meio de avatares. Essa personificação é representada, em geral, por um modelo tridimensional cuja aparência representa de alguma forma a aparência do usuário ou sua função no ambiente.

1.5 Componentes Básicos de um AVC

Como os AVCs herdam algumas características dos AVDs, quando implementados utilizam alguns componentes básicos como *displays* gráficos, dispositivos de controle e comunicação, sistema de processamento e rede de dados (SINGHAL; ZYDA, 1999).

- ***Displays* Gráficos:** Os *displays* oferecem ao usuário uma imagem 3D do ambiente. A geração de imagens é geralmente feita com a utilização de uma API (*Application Programming Interface*) gráfica, como a API OpenGL (*Open Graphics Library*). Para aumentar a sensação de imersão e ocorrer uma disseminação de imagens, alguns sistemas utilizam dispositivos que envolvem completamente o usuário, bloqueando toda sua visão de fora do ambiente virtual, por exemplo, através do uso de aparelhos complexos como os HMDs (*Head Mounted Displays*).
- **Dispositivos de Controle e Comunicação:** Os dispositivos de controle e comunicação possibilitam ao usuário se locomover e se comunicar com o ambiente. O usuário precisa ser capaz de mover, clicar e manipular objetos com outros participantes do AV, e isso é possível através do uso de vários dispositivos de entrada, como o uso do mouse, teclado, *joystick*, luvas, sensores, CAVE.
- **Sistemas de Processamento:** O sistema de processamento é responsável pelo gerenciamento de todo o ambiente. O processador recebe eventos dos dispositivos de entrada do usuário e calcula como essas entradas podem mudar a localização do usuário e dos objetos dentro do ambiente. O processador determina como e quando notificar outros participantes dessas mudanças. Similarmente, ele recebe informações providas de outros participantes, como suas localizações e comportamentos dentro do ambiente virtual.
- **Rede de Comunicação:** Os participantes de um ambiente virtual colaborativo confiam em uma rede de computadores para trocar informações. Por exemplo, suas atualizações na rede ou a visualização da localização correta de outros participantes. Similarmente, caso o usuário movimente um objeto no ambiente, os outros participantes precisam ser notificados qual a real posição do objeto. A rede também é

usada para sincronizar o estado compartilhado do ambiente e também deve suportar comunicação por texto, áudio e vídeo entre os participantes.

1.6 Interação em Ambientes Virtuais Colaborativos

A interação em ambientes virtuais é desenvolvida com a intenção de oferecer operações de seleção, manipulação e navegação no ambiente tridimensional (3D). Os participantes da aplicação realizam ações que são concretizadas por meio de técnicas de interação. Essas técnicas correspondem a métodos por meio dos quais o usuário especifica comando e dados para o sistema operacional, conseguindo dessa forma a execução das tarefas. O sistema responde ao usuário através de dispositivos de saída que estimulam os sentidos do usuário (tato, visão e audição). Assim, os estímulos são processados e assimilados pelo cérebro (NEDEL; OST 2004; REBELO; PINHO, 2004), conseguindo atingir umas das características fundamentais em um AVC, a percepção citada na Seção 1.4.

Método de seleção consiste em definir, dentre os objetos de um ambiente tridimensional, quais deles devem sofrer ação do usuário.

Por manipulação entende-se que é a tarefa de mudar algum parâmetro de um objeto tridimensional previamente selecionado que acarrete mudança de suas propriedades geométricas (tamanho, posição ou orientação) ou não.

A navegação, por sua vez, é o processo de mudar a posição do observador do ambiente tridimensional (PINHO, 2000). A navegação possibilita a visualização de vários pontos de vista, como observar sua orientação e posição, conseguindo dessa forma explorar uma parte específica do ambiente (SZALAVÁRI; GERVAUTZ, 1993).

As técnicas mais comuns em AVCs são aquelas que permitem ao usuário tocar virtualmente o objeto e utilizar os movimentos de seu corpo (mão, braços, cabeças, etc.) para realizar mudanças na posição e/ou orientação desse objeto (REBELO; PINHO, 2004).

Para realizar essas técnicas é necessário que o sistema de RV possua funções de suporte ao rastreamento das mãos e da cabeça do usuário, o reconhecimento de gestos e detecção do apontamento de um objeto. Dessa maneira, o sistema tem que ter a capacidade de realizar um mapeamento natural e intuitivo entre a ação do usuário no mundo real e ação resultante no mundo virtual (REBELO; PINHO, 2004).

De acordo com Pinho (2002), os ambientes virtuais apresentam algumas dificuldades em relação à interação como, por exemplo, a seleção de objetos é complicada se comparada com a manipulação de objetos no mundo real e nos AVCs a tecnologia de voz ainda é precária, de modo que a comunicação com os outros participantes do ambiente através de voz é fundamental no processo de interação.

A manipulação simultânea sobre um mesmo objeto é considerada uma ação complicada em AVCs.

O processo de interação em um AV é considerado como sendo um sistema contínuo onde cada ação do usuário deve ser respondida com uma ação do controlador do sistema de ambiente virtual (PINHO, 2000).

Os sistemas contínuos que fazem uma interface entre seres humanos e máquinas possuem um dispositivo de entrada que tem o objetivo de capturar os movimentos dos usuários, uma função de transferência que é responsável por mapear os movimentos dos participantes sobre os elementos controlados pelo sistema e um dispositivo de saída que mostra os efeitos das ações do usuário (PINHO, 2000).

O usuário aplica estímulos motores sobre dispositivos de entrada e através de dispositivos de saída recebe respostas capazes de impressionar um ou mais de seus sentidos.

Podem ser definidas três categorias em relação à interação em Ambientes Virtuais: (MINE, 1995).

- **Interação Direta de Usuário:** inclui técnicas interativas que utilizam o corpo do usuário para realizar a interação no ambiente (mãos, braços, gestos, direção do olhar, apontamento, etc.), conseguindo dessa maneira atuar diretamente sobre o objeto através do toque virtual.
- **Interação Através de Controles Físicos:** inclui o uso de botões, pedais, *joysticks* e outros para interagir com o mundo virtual, conseguindo assim aumentar no participante a sensação de presença dentro do ambiente, pois pode permitir algum tipo de sensação tátil.
- **Controles Virtuais:** Qualquer coisa que se imagine, um teclado, um mouse, pode ser implementado como um controle virtual. Essa categoria inclui a possibilidade de se representar visualmente um dispositivo físico, embora essa flexibilidade apresente algumas desvantagens como a falta de retorno sensorial e a dificuldade de interação com o objeto virtual.

1.7 Necessidades e Dificuldades de um AVC

O desenvolvedor de um AV deve prever quais informações de percepção são importantes, como elas podem ser capturadas ou geradas, onde elementos de percepção são necessários, de que forma apresentá-los e como dar aos indivíduos o controle sobre eles (FUKS *et al*, 2002).

É importante que haja um controle para que o fluxo de informações no ambiente não seja maior do que a capacidade do indivíduo de processá-la e assimilá-la.

Existem falhas com a maioria das atuais tecnologias colaboradoras, especialmente quando usadas para interagir com algum conteúdo no espaço. Em colaboração face a face, as pessoas usam a fala, gesticulam, usam o olhar e dão sugestões tentando se comunicar de forma mais clara possível. Porém, em muitos casos, o mundo real ou os objetos da realidade exercem um papel vital, particularmente em projeto e tarefas colaborativas no espaço. Os objetos reais também são uma fonte de informação, eles participam das atividades colaboradoras, criam referências para comunicação e uma interação dinâmica, especialmente em participações com vários usuários (BILLINGHURST; KATO, 2002).

A principal dificuldade em um AVC em relação à interação é que diferentemente do que ocorre com uma metáfora de mesa, utilizada em diversos ambientes de janelas gráficas, onde a mesa e o mouse oferecem restrições e suporte físicos para o movimento, as técnicas de interação tridimensionais dificilmente darão ao usuário o retorno tátil e/ ou comportamental igual como oferece um objeto real (REBELO; PINHO, 2004).

Igualmente acontecem nos AVDs , os AVCs apresentam uma série de desafios em seu desenvolvimento. Entre esses desafios, podem ser citados os problemas relacionados ao gerenciamento de recursos de rede, às aplicações gráficas em tempo real e às aplicações multiusuários.

Um dos desafios enfrentados pelos desenvolvedores de AVC tem sido de alocar o tempo disponível de processador entre as tarefas requeridas para suportar a presença do usuário dentro desses ambientes.

CAPÍTULO 2 - AMBIENTES VIRTUAIS COLABORATIVOS CONSTRUÍDOS COM RECURSOS DE REALIDADE AUMENTADA

Para o desenvolvimento de um ambiente virtual colaborativo de RA é necessário que ocorra uma análise dos principais ambientes já existentes, verificar suas aplicações, suas características e os dispositivos utilizados, podendo dessa maneira caracterizá-los e avaliá-los. Neste capítulo será dado um enfoque na RA, analisando suas propriedades e situando-a no contexto da RV.

2.1 Realidade Aumentada

Com o objetivo de compreender o conceito de RA, é interessante situá-la no contexto da RV.

RV é uma interface avançada para aplicações computacionais onde o usuário pode navegar e interagir, em tempo real, em um ambiente tridimensional gerado por computador, usando dispositivos multisensoriais (KIRNER; PROVIDELO, 2004).

Segundo Milgram e Kishino (1994), o termo RV estava sendo aplicado em uma série de ambientes e nem sempre esses ambientes possibilitavam uma imersão total. Ele criou um *Virtuality Continuum* (Contínuo de Virtualidade) (Figura 2.1), em que os extremos são o "Ambiente Real" e o "Ambiente Virtual". Entre um extremo e outro ele denominou de Realidade Misturada, onde fica situada a RA caracterizando-se pelo predomínio do mundo real sobre o virtual, e a VA onde ocorre o predomínio do virtual sobre o real.

Esse enquadramento permite justificar a clara separação entre RA e RV e dessa forma situar a Realidade Misturada dentro do contexto ambiente real e ambiente virtual.

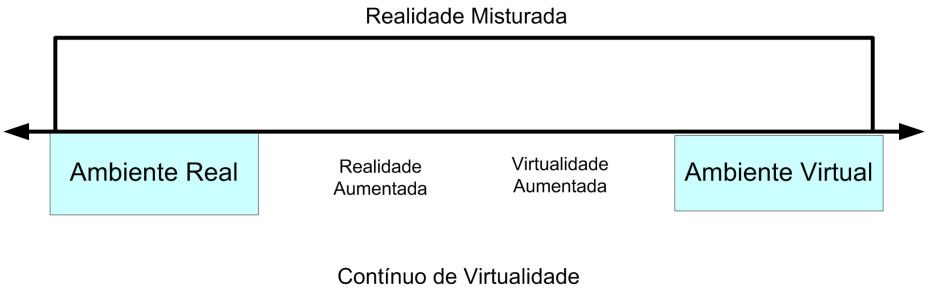


Figura 2.1: Representação do *Virtuality Continuum* (MILGRAM; KISHINO, 1994)

Assim, a VA pode ser definida como a particularização da realidade misturada que ocorre quando o ambiente principal é virtual ou há predominância do virtual. VA também pode ser definida como o enriquecimento do AV com elementos reais pré-capturados ou capturados em tempo real. Permite a inserção de avatares humanóides realistas no mundo virtual, possibilitando melhores condições para trocar idéias, trabalhar em conjunto e interagir de várias maneiras (KIRNER; TORI, 2004).

Contrariamente, a RA permite o enriquecimento do ambiente real com elementos virtuais (Figura 2.2) usando algum dispositivo tecnológico e funcionando em tempo real.

Figura 2.2: Realidade Aumentada

De acordo com definição de Azuma *et al*. (2001), a RA é um sistema que suplementa o mundo real com objetos virtuais gerados por computador, parecendo coexistir no mesmo espaço e apresentando as seguintes propriedades: combina objetos reais e virtuais no ambiente

real; executa interativamente em tempo real; alinha objetos reais e virtuais entre si e aplica-se a todos os sentidos, incluindo audição, tato, força e cheiro.

De acordo com Reitmayr (2004), a RA está baseada em uma interface que procura fundir a interface humano-computador com o mundo real, pois busca estabelecer uma interface mais natural, oferecendo propriedades e informações abstratas do mundo real ou associando fenômenos encontrados no mundo real, como espaço e tempo. A idéia principal da RA é sobrepor ou combinar as sensações geradas pelo computador com estímulos gerados pelo mundo real. Uma parte importante de qualquer interface do usuário de RA é a interação 3D, pois humanos sabem como interagir com objetos reais, como controlá-los e manipulá-los.

De acordo com Azuma (1997), para que a RA permita a combinação do mundo real com o virtual, a interação em tempo real e o alinhamento tridimensional do real e virtual, são necessários que alguns problemas sejam resolvidos como, por exemplo: rastreamento de objetos reais, alinhamento e calibração das sobreposições na cena virtual combinada e a interação.

2.2 Realidade Aumentada Colaborativa

No ambiente da computação observam-se inovações e em particular como a RA pode ser usada em trabalhos colaborativos. O espaço compartilhado pode integrar várias tecnologias de interfaces como, por exemplo, a RA, o trabalho colaborativo, interface tridimensional e técnicas de visão de computadores por rastreadores e posicionamento (BILLINGHURST; KATO, 2002).

A RA colaborativa utiliza o apoio de sistemas distribuídos para suportar múltiplos usuários, permitindo uma experiência em comum com os outros participantes do ambiente (BILLINGHURST; KATO, 1999).

Um AVC, utilizando RA, pode usar um HMD com uma câmera e como método de posicionamento da imagem são usados marcadores para associar os objetos às placas.

A interface de um espaço compartilhado permite trabalhar vários usuários simultaneamente no mundo real e virtual. Desde que todos os usuários compartilhem o mesmo banco de dados de objetos virtuais, eles poderão ver os mesmos objetos associados nos marcadores. De todos os pontos de vista, os usuários podem selecionar e mostrar os cartões para outros participantes ou passar e pedir objetos da mesma maneira que acontece no mundo real (BILLINGHURST *et al*., 2000).

A RA colaborativa é a combinação de CSCW com a RA, na qual o aumento do ambiente real de um usuário acontece com as ações de outro usuário (WICHERT, 2002).

As tecnologias de RA têm o potencial para habilitar uma comunicação mais natural. Na colaboração a RA pode combinar os mundos físicos e virtuais de maneira que os objetos reais interajam com o conteúdo tridimensional e possibilitem um compartilhamento de informações com um aumento de compreensão (BILLINGHURST *et al*., 2000).

A RA pode ser usada para melhorar o trabalho físico compartilhado e criar uma interface tridimensional para CSCW. Uma das primeiras interfaces que mostraram o potencial da RA para colaboração face a face foi o *Studierstube*. Nesse projeto os participantes usavam HMD que permitia a colaboração em um cenário tridimensional e a observação de um modelo 3D sobrepostos no mundo real (BILLINGHURST *et al*., 2000).

Os pesquisadores de *Studierstube* identificaram cinco características nos ambientes de RA colaborativa (BILLINGHURST *et al*. 2000).

- **Virtualidade**: os objetos que não existem no mundo real podem ser visualizados;
- **Aumento:** os objetos reais podem ser aumentados com anotações virtuais;
- **Cooperação**: os usuários podem ver uns aos outros e cooperar naturalmente;

- **Independência:** os usuários podem controlar seu ponto de vista independentemente;
- **Individualidade:** as informações mostradas podem ser diferentes para cada participante.

A interface de RA colaborativa pode produzir comportamentos de comunicação que são mais semelhantes na colaboração face a face do que colaboração baseada em tela de projeção. Isso porque quando as pessoas colaboram em uma mesa, elas podem ver os objetos ao mesmo tempo. Porém, quando os usuários estão colaborando em frente a uma tela, o espaço de trabalho é o espaço da tela e pode ser separado da comunicação interpessoal. A interface baseada em tela pode introduzir descontinuidade que faz com que os colaboradores exibam comportamentos de comunicação diferentes.

Esta forma de colaboração tem se mostrado bastante interessante, pois permite que a colaboração seja feita da mesma maneira como é feita normalmente entre as pessoas em um ambiente real, acrescida das informações fornecidas pelo ambiente virtual. Esses sistemas têm sido usados em aplicações como jogos, visualização cientifica, modelagem de ambientes e objetos e educação, entre outros.

2.3 Aplicações de Realidade Aumentada

Os sistemas de RA possibilitam experiências como a sensação de presença, envolvendo interação e formas de colaboração. Assim, esses sistemas podem oferecer aos participantes maiores informações sensitivas, onde a principal característica das aplicações de RA é criar um ambiente onde as informações do mundo real são utilizadas para incrementar o cenário virtual.

2.3.1 *MagicBook*

O *MagicBook* é uma interface de RA que utiliza um livro real para transpor usuários entre a realidade e a virtualidade. Suporta a colaboração, permitindo que vários usuários compartilhem o mesmo ambiente virtual (BILLINGHURST *et al*., 2001).

Os participantes podem virar as páginas do livro, olhar as figuras e ler os textos sem qualquer tecnologia adicional (Figura 2.3 (a)), mas quando utilizam dispositivos de RA, podem ver os objetos virtuais de vários pontos de vista alinhados a marcadores (Figura 2.3 (b)) e são representados na cena por um avatar; Portanto podem colaborar e visualizar uns aos outros momentaneamente (Figura 2.3 (c)).

Figura 2.3: Utilização do MagicBook. (a) Realidade, (b) Realidade Aumentada, (c) Realidade Virtual Imersiva (BILLINGHURST et al., 2001)

Utiliza a ferramenta ARToolkit[2] para rastrear a posição da cabeça do usuário e os marcadores. Para criação dos objetos virtual nas páginas do livro utiliza as ferramentas VRML e OpenGL.

O *MagicBook* consiste na utilização de um dispositivo *Handheld Augmented Reality Display* (HHD), uma estação gráfica e um livro. O usuário utiliza esses equipamentos para gerar a visualização da cena virtual individual. O HHD é um cabo com um *SONY Glasstron*

[2] Conjunto de bibliotecas desenvolvidas para aplicações de Realidade Aumentada

PLM-A35 colocado no topo, um rastreador no fundo, uma pequena câmera de vídeo na frente do *Glasstron Display* .

Todos usuários possuem independência para visualizar o conteúdo e interagir facilmente com o modelo virtual.

2.3.2 - *Construct3D*

O projeto *Construct3D* é uma aplicação projetada para ensinar matemática e geometria através de técnicas de RA. Foi baseado no sistema de RA colaborativa móvel *Studierstube*[3] (SCHMALSTIEG, 1996) e utiliza a RA para possibilitar colaboração e interação face a face entre professores e estudantes (Figura 2.4).

(a) (b)

Figura 2.4: Estudantes trabalhando com o Construct3D. (a) O estudante insere uma esfera dentro do cone. (b) Exemplo simples de álgebra de vetor. (KAUFMANN, 2003)

Operações no sistema como carregar, apagar e desfazer são realizadas através de um Painel de Interação Pessoal (PIP - *Personal Interaction Panel*).

O PIP é formado por uma prancheta e um apontador. O usuário utiliza HMD transparente para visualizar os objetos 3D e o PIP para selecionar um objeto e colocá-lo na prancheta para visualizá-lo, imitando uma mesa sobre a qual se coloca um objeto de estudo. O

[3] Ambiente para desenvolvimento de trabalhos de RA móvel colaborativa (SCHMALSTIEG, 1996).

PIP permite a integração direta de interface de elementos 2D como botões, lâminas e a integração 3D (SZALAVÁRI; GERVAUTZ, 1997).

O sistema permite a seleção de visibilidade por usuário e por camada. Por exemplo, no modo independente, todo estudante pode ver os elementos construídos por ele; no modo colaborador, toda a informação é visível a todos os participantes e no modo professor um participante controla as camadas através do PIP e executa toda a construção.

Criou-se uma solução híbrida de hardware para completar diversas interações entre professores e estudantes no cenário virtual, como as soluções abaixo:

- **Sala de aula aumentada:** Consiste em utilizar um notebook, um HMD com câmera e uma luva. Os usuários podem mostrar interesse nos objetos virtuais, desde que estejam usando equipamentos ligados em rede local sem fio para possibilitar a comunicação (Figura 2.5 (a)). Além disso, há uma mesa pequena, servindo como um lugar para colaboração entre dois usuários (Figura 2.5 (b)). Na mesa há um marcador, que é identificado por uma câmera que fica sobre a cabeça do usuário. Os estudantes movem e giram o marcador e o objeto é exibido na tela de projeção.

(a) (b)

Figura 2.5: Ambientes de Realidade Aumentada. (a) Demonstração do equipamento de realidade aumentada.(b) Sala de aula aumentada. Dois usuários interagindo na construção enquanto um terceiro usuário inspeciona um modelo na tela de projeção (KAUFMANN, 2003)

- **Salas de aulas com telas de projeção:** Com a ajuda de um computador com uma câmera de vídeo e uma tela de projeção é possível exibir a construção da cena

virtual, que pode ser compartilhada por um grupo de usuários, mostrando imagens estereoscópicas com auxílio de óculos estereoscópicos (Figura 2.6) (KAUFMANN, 2003).

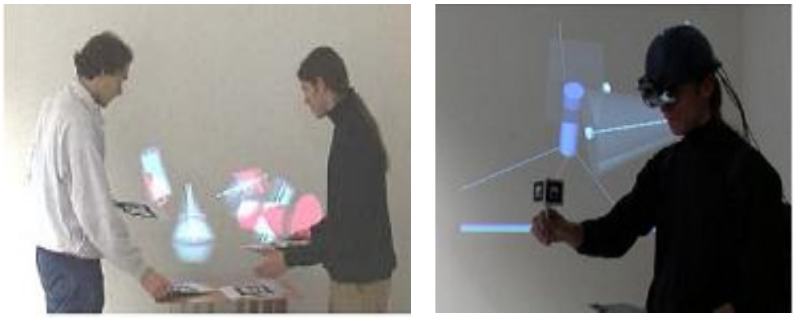


Figura 2.6: Interação através de tela de projeção. (KAUFMANN, 2003)

- **Sala de aula híbrida distribuída**: Neste tipo de configuração os estudantes são equipados com computadores pessoais que trabalham com RV, assistindo o processo de construção na tela do seu micro (Figura 2.7). Foi construído um sistema de RV usando uma câmera para rastrear as posições e um cartão gráfico. Os estudantes podem escolher pontos de vista individuais e manipular cópias dos mesmos objetos construídos. O professor também pode escolher e mostrar seu ponto de vista.

Figura 2.7: Utilização de RA através do monitor (KAUFMANN, 2003)

A implementação do *Construct3D* foi desenvolvida na linguagem C++ e utilizou a plataforma do *Studierstube*, baseada na biblioteca *Open Inventor* e na biblioteca OpenGL para renderizar as cenas gráficas.

2.3.3 - *Shared Space*

O projeto *Shared Space* foi desenvolvido com o objetivo de melhorar a interação entre humanos e computadores. Em particular, foi investigado como a RA pode ser usada para desenvolver ambientes colaborativos face a face, para permitir a combinação de objetos do mundo virtual com o mundo real utilizou a biblioteca ARToolkit (BILLINGHURST *et al*., 2000)

No ambiente físico há vários cartões de marcadores com um quadrado de borda e no meio um símbolo. Quando o usuário olha para os marcadores, são usadas técnicas de visão de computadores para calcular a posição e a orientação da cabeça do usuário em relação aos marcadores, e as imagens virtuais são alinhadas com os objetos físicos (BILLINGHURST *et al*., 2000).

Nesse projeto os usuários podem ver outros usuários, a expressão facial, gestos e linguagem do corpo suportando uma comunicação face a face. Assim a interface do *Shared Space* permite que vários usuários no mesmo local trabalhem simultaneamente em ambos os mundos, real e virtual (Figura 2.8).

Figura 2.8: Ambiente Colaborativo Shared Space (BILLINGHURST et al., 2000)

Desde que todos os usuários compartilhem o mesmo banco de dados de objetos virtuais, eles vêem os mesmos objetos anexados nos marcadores de todos os seus pontos de

vista. O usuário pode mover e mostrar os cartões para outros participantes ou passar ou pedir os objetos da mesma maneira que acontece com objetos reais.

Vários marcadores podem ser rastreados simultaneamente, então a posição relativa dos objetos no marcador pode ser usada para ativar a interação dos objetos virtuais. Por exemplo, colocando um cartão com um disco voador e no próximo cartão um alienígena, é possível animar o alienista voando junto com o disco voador (Figura 2.9).

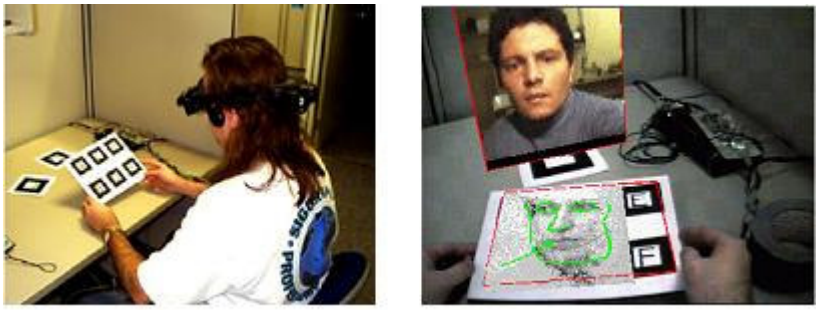
Figura 2.9: Interação Espacial no *Shared Space*: Usuários criando animação de objetos virtuais (neste caso o alienista e o disco voador) trabalhando com dois marcadores juntos. (BILLINGHURST et al., 2000)

2.3.4 - *Real World Teleconferencing*

Este projeto descreve uma aplicação de RA que é representada por imagens de vídeo ao vivo ou imagens de avatares anexadas a objetos e podem ser livremente posicionadas pelo usuário.

O usuário utiliza uma interface de RA para realização da aplicação, como um HMD e uma câmera pequena. O usuário de RA utiliza um conjunto de marcadores onde cada colaborador remoto tem seu nome gravado no marcador. Para iniciar a comunicação, o usuário olha para o cartão que representa o colaborador remoto. Técnicas de visão de computadores são usadas para identificar o usuário específico (usando o nome do usuário no

cartão) e assim é visualizado um avatar virtual 3D do usuário remoto. As técnicas de visão são usadas para calcular a posição e a orientação relativa dos cartões e então as imagens são precisamente alinhadas com os marcadores. (Figura 2.10) (BILLINGHURST; KATO, 1999).



(a) (b)

Figura 2.10: *Real World Teleconferencing* (a) Utilizando uma interface de RA .(b) Representação de usuário remoto utilizando RA (BILLINGHURST; KATO, 1999)

2.3.5 *Outdoor Collaborative Augmented Reality* (OCAR)

Este projeto apresenta como objetivo a possibilidade de dois ou mais usuários interagirem em suas tarefas, como navegação e sistema de anotação ao ar livre utilizando técnicas de RA.

Foi empregado o uso de RA para realçar a percepção dos usuários em relação ao mundo real juntamente com informações geradas pelo computador, incluindo a computação móvel para permitir que os usuários alcancem e manipulem as informações independentes do momento e de sua posição.

OCAR (REITMAYR; SCHMALSTIEG, 2003) consiste em permitir que o usuário navegue em uma cidade, ao invés de utilizar mapas para conhecer o lugar, pode utilizar-se do sistema para conhecer seu destino, combinando momentaneamente cenas do mundo real e virtual (Figura 2.11)

Figura 2.11: Usuário caminhando em lugar histórico em Viena (REITMAYR; SCHMALSTIEG, 2003)

Foi desenvolvida uma aplicação turística para a cidade de Viena, e ela oferece uma ajuda para realizar a navegação, pois permite que o usuário se dirija a um local destino e utilize um browser de informações que exibe detalhes do local especificado.

O usuário pode alternar entre modos diferentes de aplicação como, por exemplo, navegação, informação e anotação.

No modo navegação o usuário escolhe um local e o sistema calcula o caminho mais curto através de possíveis rotas de rede, este modo é interativo e reage aos movimentos do usuário.

No modo informação o usuário pode selecionar um conjunto de ícones para visualizar informações através de imagens e textos, por exemplo, pode ver datas interessantes sobre os edifícios e lugares turísticos que ele está visitando (Figura 2.12).

Figura 2.12: Modo Informação (REITMAYR; SCHMALSTIEG, 2003)

E no modo anotação, o usuário pode anotar no ambiente informações que desejar através de ícones virtuais, com diferentes cores e pode compartilhar com outros usuários, incluindo o nome do usuário que os criou.

Para interação com o ambiente, a aplicação utiliza dispositivos como um *Wacom Tablet e Pen*[4] associados a uma câmera e marcadores.

O sistema baseia-se no ambiente *Studierstube* (SCHMALSTIEG *et al.*, 1996), permitindo uma aplicação multiusuária e utilização de vários dispositivos. Para calcular a posição e orientação do usuário utilizou-se técnicas do ARToolkit.

O projeto utiliza um *notebook* com um processador de 2GHZ e sistema operacional *Windows XP*. Para habilitar a comunicação com outras unidades móveis é utilizada uma rede local sem fio. Todo equipamento é montado nas costas do usuário, incluindo um HMD, um sensor de orientação e uma câmera.

2.3.6 - Comparação das Principais Características de AVC de Realidade Aumentada

Observou-se nos projetos analisados, que todos apresentam uma característica crucial para ambientes colaborativos, o contexto compartilhado. Tanto os objetos quanto o ambiente são compartilhados e as mudanças ocorridas são visíveis a todos os participantes, caracterizando assim a flexibilidade do ambiente e a possibilidade de ocorrerem múltiplas visões das atividades que estão sendo realizadas.

[4] Técnica de interação em que uma plataforma fixa (mesa ou painel de parede) ou móvel (prancheta) controla a posição da janela onde deve ser exibidas opções de menus (REBELO; PINHO, 2004).

Todos os ambientes analisados apresentam as principais características encontradas nos AVCs. Em todos os ambientes os participantes envolvidos utilizam meios como gestos, som e textos para poderem se comunicar e interagirem no ambiente.

No ambiente *MagicBook* e no *Shared Space* os usuários podem estar totalmente imersos na mesma cena e são representados através de um personagem virtual.

A coordenação de atividades para realização do trabalho colaborativo é observada no ambiente *Construct3D*, onde a camada professor é a responsável para garantir que as tarefas dos participantes sejam realizadas na ordem e no tempo corretos. Considere por exemplo um professor que trabalha e assiste na construção com estudantes. A construção inteira é visível a todos os usuários. Se o professor deseja que os estudantes pratiquem a mesma construção novamente, ele troca o modo enquanto a aplicação ainda está ocorrendo. Agora cada estudante pode ver apenas os elementos que ele construiu, sem ser influenciado pelo trabalho dos professores ou outros estudantes da mesma categoria.

A cooperação é observada em todos os ambientes, pois os participantes produzem, manipulam e organizam informações.

No projeto *Shared Space*, a cooperação é observada na possibilidade dos participantes trocarem e visualizarem os cartões de todos os pontos de vista, e no projeto OCAR os participantes realizam anotações virtuais sobre os objetos, permitindo que outros usuários compartilhem a informação.

Todos os projetos utilizam algoritmos baseados em técnicas de visão computacional calcular o ponto de vista real da câmera, como a posição e a orientação de um marcador em tempo real.

Apenas o ambiente *Construct3D* permite que o participante tenha um modo independente de visualizar as tarefas construídas por ele, pois este tem a possibilidade de alternar entre tarefas individuais e colaborativas sem muitas dificuldades.

O ambiente *MagicBook* apresenta dificuldades em relação à percepção, devido que nem sempre os participantes estão atentos no avatar do outro participante, dificultando dessa forma a colaboração entre eles.

A Tabela I classifica alguns AVCs com RA, destacando suas características mais relevantes.

Tabela I: Principais características dos AVCs levantados

	MagicBook	**Construct3D**	**Shared Space**	**Real World Teleconferencing**	**OCAR**
Comunicação	*Broadcas*t	*Multicast*	-	*Multicast*	-
Tecnologia	ARToolkit, VRML, OpenGL	*Studierstube,* Artoolkit	ARToolkit, OpenGL	ARToolkit	*Studierstube,* XML, ARToolkit
Dispositivos	HHD	HMD e PIP	HMD	HMD	HMD e PIP
Formas de Colaboração	Visão, fala e gestos	Visão, fala e gestos	Visão, fala e gestos	Visão, fala, gestos e textos	Visão, fala e textos
Interação	Navegação, orientação no espaço, presença e imersão.	Navegação, orientação no espaço, presença e imersão.	Navegação, orientação no espaço, presença e imersão.	Navegação, orientação no espaço, presença e imersão.	Navegação, orientação no espaço, presença e imersão.
Tipo de Colaboração	Local/Remota	Local/Remota /Remota Móvel	Local	Remota	Remota/Móvel
Coordenação de Atividades	Não	Sim	Não	Não	Não
Área de Aplicação	Educação	Educação	Entretenimento	Vídeo Conferência	Turismo

De acordo com o levantamento das aplicações, observou-se que o *Construct3D* é o projeto que mais apresenta características de colaboração. Através de sua solução híbrida de hardware, possibilita que no seu ambiente ocorra a comunicação, a coordenação, a consistência de indivíduos e a visualização da cena através de vários pontos de vista, características analisadas na Seção 1.4.

Em seguida, o projeto OCAR herda algumas características de colaboração, pois possibilita alternar entre diferentes modos de aplicação, conseguindo assim, através do modo de anotação, compartilhar informações com outros usuários. Inclui-se também nesse projeto o uso RA móvel.

No geral, todos os projetos concentram seus objetivos no compartilhamento de informações e técnicas de RA para combinar o mundo real e virtual, permitindo ao usuário interagir nesse ambiente e realizar a colaboração entre os participantes.

Através da análise dos ambientes citados, observa-se que AVCs utilizam a tecnologia de AVD para suportar o trabalho em grupo; desta forma é preciso compreender e verificar algumas considerações desses ambientes para permitir a colaboração entre os participantes, incluindo um estudo dos protocolos, recursos de camada de *middleware* e as linguagens que são utilizadas para o desenvolvimento de aplicações de RA.

CAPÍTULO 3 – CONSIDERAÇÕES DE PROJETO PARA AMBIENTE VIRTUAL COLABORATIVO UTILIZANDO REALIDADE AUMENTADA

O desenvolvimento de uma aplicação de RA Colaborativa segue os mesmos desafios encontrados nos AVCs, portanto depende do objetivo, dos recursos disponíveis para o desenvolvimento, do número de usuários que irão utilizá-los, do nível de interatividade desejado, entre outros.

Este Capítulo apresenta algumas características para criação desse ambiente, seus protocolos, largura de banda, heterogeneidade e recursos para permitir a comunicação de dados.

Posteriormente serão detalhadas algumas linguagens para criação de aplicações utilizando RA.

3.1 Características de um Projeto de AVC

Em um projeto de AVC existem diversas características de comunicação em rede que devem ser consideradas, como a latência, largura de banda, confiabilidade e os protocolos de comunicação (Figura 3.1) (SINGHAL; ZYDA, 1999).

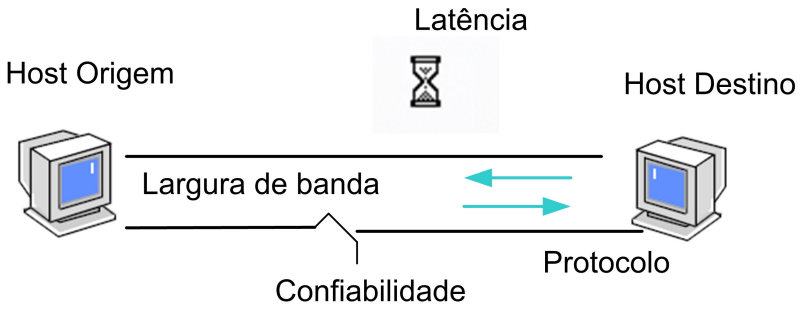


Figura 3.1: Características de uma rede de comunicação (SINGHAL; ZYDA, 1999)

a) Largura de Banda: A largura de banda corresponde à taxa na qual a rede pode entregar dados ao *host* de destino. A largura de banda disponível é determinada pelo tipo de meio usado para transportar os dados e é também limitada pelo hardware utilizado para transmitir esses dados. É a razão entre o número e bits transmitidos e o tempo necessário para a transferência, isto é, a largura de banda mede quantos bits podem ser transportados por unidade de tempo. Os AVCs requerem largura de banda para suportar vários usuários, recursos audiovisuais, troca de primitivas gráficas tridimensionais e modelos em tempo real.

b) Confiabilidade: É a medida que reflete a quantidade de dados que se perde no trajeto do *host* origem ao *host* destino. Confiabilidade significa que o sistema pode assumir que dados enviados são sempre recebidos corretamente, mesmo que periodicamente haja a necessidade de reenviar a informação.

c) Latência: É o tempo requerido pela rede para transferir um bit de dado de um ponto da rede a outro. A latência da rede corresponde ao atraso (*delay*) na transferência de dados. Ela mede quantos segundos um byte de dados leva para trafegar do *host* origem ao *host* destino. Um dos desafios dos AVC é manter uma latência mínima na comunicação, pois a latência tem grande influência sobre o realismo da simulação. Por isso, deve ser escolhido um modelo que garanta tempo de resposta rápido, de forma a simular uma interação de maneira mais realista, próxima a do mundo real. Esta necessidade sugere protocolos de comunicação menos confiáveis (por exemplo, o UDP – *User Datagram Protocol*) e uma estrutura de comunicação em que a perda de mensagens não se torne tão importante, pois pode ser compensada por mensagens subseqüentes.

d) Protocolos: Os protocolos de rede descrevem um conjunto de regras definidas que possibilitam a comunicação entre as aplicações. Protocolo é a linguagem usada

pelos dispositivos de rede de modo que eles consigam se entender, isto é, trocar informações entre si (TORRES, 2001). A seguir são descritos alguns protocolos utilizados na construção de AVCs :

- **IP (*Internet Protocol*):** é um protocolo não orientado à conexão, isto é, não verifica se o datagrama chegou ou não ao destino. A principal função do IP é o roteamento, ou seja, adicionar mecanismos para que o datagrama chegue o mais rapidamente possível ao seu destino.
- **TCP (*Transmission Control Protocol*):** O IP não garante que as mensagens sejam recebidas na ordem correta, nem se elas serão recebidas. O TCP recebe os datagramas IP e trata de colocá-los em ordem e verificar se todos chegaram corretamente. É um protocolo de transporte orientado à conexão, pois ao receber um pacote de dados, o protocolo envia uma mensagem de confirmação de recebimento máquina transmissora, chamada de *acknowledge*. Se o transmissor não receber uma confirmação de recebimento dentro de um determinado tempo, o pacote é retransmitido. O problema da retransmissão é o tempo que é perdido. O transmissor tem de esperar o recebimento de uma mensagem de confirmação para então enviar um novo pacote, tornado o mecanismo de envio de dados extremamente lento (TORRES, 2001).
- **UDP (*User Data Protocol*):** é um protocolo de transporte não orientado à conexão, isto é, ao contrário do TCP, ele não verifica se o pacote de dados chegou ou não ao seu destino. A vantagem para aplicações usarem o UDP em vez do TCP é que a transmissão de dados fica mais rápida. Primeiro, o tamanho do pacote de dados a ser transmitido fica menor, já que o cabeçalho UDP é bem menor que o cabeçalho TCP e, segundo, no UDP não existe mecanismo de verificação de chegada do pacote, acelerando o envio de pacotes.

Segundo Pinho (2002), o protocolo a ser usado em um AVC depende do tipo de mensagem a ser transmitida. Mensagens do tipo atualizações de estado, que informam, por exemplo, a posição atual de um objeto, não necessita ser transmitidas com protocolos confiáveis como TCP. Nesse tipo de mensagem cada novo dado recebido deve se sobrepor aos dados anteriores, pois a única informação relevante é o estado atual do objeto. Assim, se por um erro no protocolo, um dado for incorretamente recebido, este pode ser descartado sem a necessidade de um reenvio, pois a chegada de um novo dado irá corrigir o problema detectado. Em casos como esses, é possível usar, por exemplo, protocolos como UDP que não garante confiabilidade, porém é mais rápido que outros como TCP.

Outros fatores que também influenciam no desenvolvimento de um AVC e que tem muito em comum com AVDs são (SINGHAL; ZYDA, 1999):

- **Heterogeneidade:** Algumas vezes, na utilização de um AVC, os usuários não possuem acesso a alguns equipamentos. Por exemplo, enquanto alguns participantes estão usando uma estação gráfica com teclado e mouse conectados ao telefone, outros podem estar usando um HMD com *datagloves* conectados a um multiprocessador e ligados através da rede local. Surge também a heterogeneidade nas redes, pois diferentes usuários podem estar conectados em AVCs usando redes diferentes. Conseqüentemente alguns usuários são capazes de receber mais informações do que outros. A heterogeneidade surge também quando o assunto é *display* gráfico, capacidade computacional e capacidade de áudio.
- **Sistema de Tempo Real:** Um AVC precisa suportar interações em tempo real com o usuário. O desenvolvimento do software deverá executar detecção e processamento rápido das ações do usuário. A demora no processamento pode causar lentidão e por conseqüência distração na interação do usuário. A geração de imagens

deverá acontecer em taxa fixa, por exemplo, 30 quadros por segundo e a demora no processamento pode reduzir a qualidade de imersão do participante no ambiente.

- **Gerência de Falhas**: Quando o sistema é distribuído, poderá ocorrer a possibilidade de um ou mais *hosts* conectados falhar ou simplesmente desligar ao mesmo tempo. Quando ocorre uma falha ou existe a sobrecarga em um nó, é necessário que os processos desse nó com problemas sejam repassados para outra CPU, e o nó inativo deverá ficar suspenso até que o problema seja solucionado.
- **Escalabilidade:** Pode ser definida como número de entidades que podem simultaneamente participar do ambiente. As entidades podem ser um humano controlado, um terreno (associados a aspectos como pedras, árvores e prédios) e até mesmo objetos lógicos como o estado de tempo atual ou um grupo de objetos. A escalabilidade depende de vários fatores, incluindo a capacidade da rede, capacidade do processador, velocidade de renderização e a velocidade de servidores compartilhados.

Além das características e fatores acima relatados, é necessário a escolha de um modelo de comunicação para a construção de um AVCs ou um sistema de RA distribuída, pois esses modelos também apresentam uma influência significativa no desempenho da aplicação. Entre os principais modelos de comunicação tem-se o *unicast*, o *broadcast* e o *multicast* (MACEDONIA, ZYDA, 1997).

A comunicação *unicast* estabelece a comunicação entre dois *hosts*. Um host envia mensagem para um *host* específico.

A grande vantagem deste esquema de comunicação é a facilidade para sincronização de atualização das informações. Um dos problemas é que para um número elevado de usuários este esquema irá resultar em um número muito alto de conexões, uma vez que é necessário a conexão de cada nó com os demais nós da rede. Outro problema é o grande

número de mensagens geradas. Uma representação deste modelo de comunicação é mostrada na Figura 3.2.

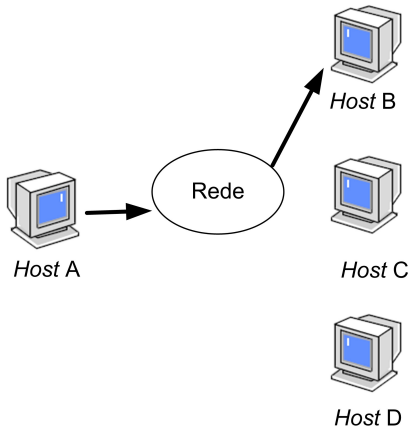


Figura 3.2: Modelo de Comunicação *Unicast*

Na comunicação *broadcast* o dado é enviado para todos os *hosts* conectados á rede. Nesse modelo de comunicação todos os usuários de um mesmo ambiente devem "aceitar" em estabelecer a comunicação por meio de um canal comum (Figura 3.3).

A maior vantagem deste modelo de comunicação é a diminuição das mensagens geradas, pois a cada atualização na aplicação resulta em somente uma mensagem.

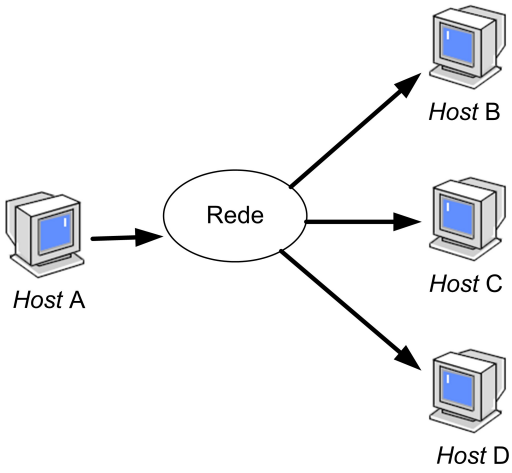


Figura 3.3: Modelo de Comunicação *Broadcast*

A comunicação *multicast* é o tipo de transmissão caracterizada pelo envio dos dados para um conjunto determinado de elementos da rede de comunicação ou de computadores (DANTAS, 2002).

Este serviço permite transmitir cada pacote para um grupo especial *multicast.* Cada grupo *multicast* possui um endereço especial de destino (endereço IP, classe D da Internet, endereços livres de 224.0.0.0 a 239.255.255.255), sendo assim, o pacote é entregue apenas aqueles hosts que participam daquele grupo *multicast* (Figura 3.4). Com *multicast,* diferentes grupos podem se associar a diferentes endereços *multicast* (MAUFER, 1998).

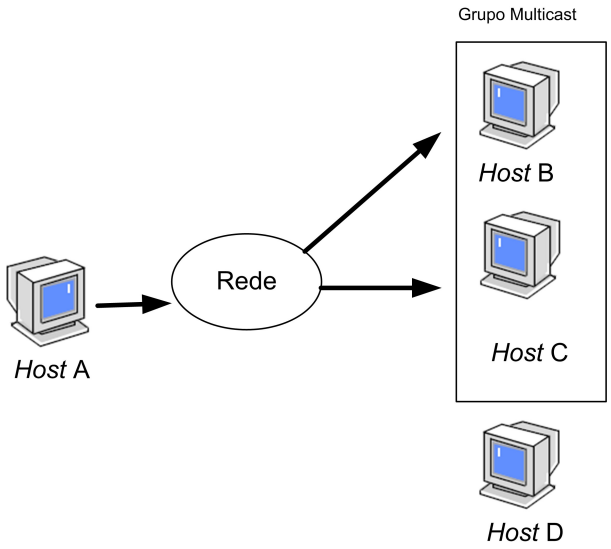


Figura 3.4: Modelo de Comunicação Multicast

Quando um processo envia um pacote para um endereço da classe D, é feita uma tentativa de entregá-lo a todos os membros do grupo endereçado, mas não há qualquer garantia de que isso realmente acontecerá. É provável que alguns membros não obtenham o pacote (TANENBAUM, 2003).

Este modelo permite comunicação 1:N (um-para-muitos) e N:M (muitos-para-muitos) onde os dados são enviados para um grupo indeterminado de máquinas simultaneamente.

Este tipo de comunicação é indicado quando existe a necessidade de transmitir dados para vários hosts simultaneamente (mas não necessariamente a todos os *hosts*). A comunicação *multicast* é muito usada em aplicações como videoconferência, compartilhamento de *whiteboard*, simulação de interação distribuída, jogos em redes, ensino e treinamento à distância, distribuição de dados em tempo real (MAUFER, 1997).

O uso de comunicação *multicast* tem a vantagem de permitir que a aplicação sirva a vários usuários sem sobrecarregar a rede, como também minimiza os serviços da aplicação servidora ao enviar as informações. Inclui também a possibilidade das aplicações estarem disponíveis a um grande número de participantes, pois a aplicação que utiliza multicast pode transmitir seus dados tanto para poucos como a muitos usuários.

Portanto, a comunicação *multicast* é comumente utilizada em aplicações de RA distribuídas por limitar o número de participantes em uma mesma recepção de mensagem, eliminando, dessa forma, as trocas de informações irrelevantes. Como as mensagens são enviadas para um grupo de usuários simultaneamente há a economia de largura de banda, pois a quantidade de mensagens é reduzida. Sem o uso do *multicast* a aplicação teria que enviar *n* mensagens aos *n* participantes do grupo.

Alguns parâmetros usados como métricas para aplicações *multicast* incluem (MAUFER, 1997):

- Número de mensagens enviadas simultaneamente para os grupos;
- Tamanho do grupo envolvido (número de participantes, extensão geográfica do grupo);
- A largura de banda usada durante o processo;
- O nível de interação (humana) entre os membros do grupo.

Um requisito básico a todos os *hosts* que utiliza comunicação *multicast* para enviar e receber dados são o uso do protocolo IGMP (*Internet Group Management Protocol*) que

representa uma parte do espaço do endereçamento IP. O protocolo IGMP é usado por *hosts* para informar quem participa (ou pretende participar) de um determinado grupo.

3.2 Camada *Middleware*

Recentes avanços em redes de computadores viabilizaram o surgimento de ambientes computacionais cujos componentes encontram-se fisicamente distribuídos (COSTA; KON 2002).

A demanda por aplicações distribuídas tem crescido continuamente desde o surgimento das mesmas. Os projetos de AVCs estão diretamente ligados a redes de computadores e suas aplicações devem aproveitar a estrutura oferecida por estas para desenvolver seus projetos.

As aplicações distribuídas, em geral são executadas em ambientes heterogêneos, constituídos por computadores que, em muitos casos, apresentam arquiteturas e sistemas operacionais diferentes. Uma das soluções utilizadas para amenizar os problemas decorrentes dessa heterogeneidade foi interpor entre aplicações e sistemas operacionais uma terceira camada de software. Essa terceira camada, denominada *middleware*, permite que desenvolvedores de aplicações distribuídas possam dispor de uma interface de programação uniforme.

Como exemplo de recursos utilizados na camada *middleware*, tem-se CORBA (*Common Object Request Broker Architecture*), JAVA RMI (*Java Remote Method Invocation*), RPC (*Remote Procedure Call*) e outras.

Porém para as aplicações de RA distribuídas, a utilização da camada *middleware* só se justifica se o atraso decorrente for aceitável, ou seja, não afetar em demasia a taxa de frames por segundo e de interação.

Muitas vezes é dada preferência à utilização dos recursos de comunicação do sistema operacional, como por exemplo, os *sockets* UDP.

Sockets é um mecanismo utilizado para definir uma comunicação fim a fim entre dois *hosts* em uma rede. Os sockets podem representar muitos tipos diferentes de canais, incluindo comunicação segura com um único *host* destino, comunicação incerta com um único *host* destino, comunicação incerta com múltiplos *hosts* destino (SINGHAL; ZYDA, 1999).

Os *sockets* identificam várias informações, incluindo cinco aspectos sobre canais de comunicação (SINGHAL; ZYDA, 1999).

- **Protocolo:** como os sistemas operacionais trocam dados em uma aplicação, o protocolo sugere em nível de confiança se o sistema operacional de destino enviará pacotes de confirmação e se deveriam ser retransmitidos se não recebeu confirmação.
- **Host destino:** O endereço do *host* de destino envia pacotes através do *socket*. Em alguns casos, o endereço do destino não é armazenado com o *socket*, no qual a aplicação tem que especificar um endereço sempre que envia um pacote.
- **ID da aplicação destino ou porta:** Este número identifica o *socket* apropriado no *host* destino. Para cada protocolo, cada *socket* do *host* é associado a um inteiro de 16-bit diferentes com o sistema operacional. Especificando o protocolo junto com esse número de porta em cada pacote, assegura que o *host* origem pode entregar o pacote à aplicação correta (*host* destino).
- **Host de origem:** Este endereço identifica qual *host* está enviando os dados.
- **ID da aplicação local ou porta:** um inteiro de 16 bits identifica qual aplicação está enviando dados através do *sockets*. Através da inclusão desse número de porta junto com o endereço do *host* origem no pacote, o *host* origem assegura que o *host* destino poderá mandar de volta pacotes de resposta à aplicação.

Quando uma aplicação envia dados para um *socket*, o sistema operacional tem informação suficiente para configurar a saída e enviar os dados (com o uso de protocolos), para quem vai enviar os dados (endereço de *host* destino e porta da aplicação destino) e como identificar o remetente (*host* origem e porta da aplicação origem) (SINGHAL; ZYDA, 1999).

De acordo com Dias (2003), os *sockets* podem ser classificados como:

- ***Stream Socket*** - Oferece seqüênciamento e fluxo bidirecional. Este *socket* transmite dados sobre uma base confiável e com capacidade de transmissão de dados expressos. *Stream Socket* trata a comunicação como seqüência contínua de caracteres e utiliza o protocolo TCP.
- ***Datagram Socket*** - Suporta fluxo de dados bidirecional, mas não oferece um serviço confiável como *Stream Socket*. Mensagens duplicadas e em ordem diferente (não seqüenciadas) são problemas que podem aparecer nesse tipo de *socket*, que precisa ler mensagens inteiras de uma vez e utiliza o UDP, que é pouco confiável e orientado à mensagem.

Os programas baseados em *sockets* são geralmente aplicações Cliente-Servidor. O Servidor espera pedidos de seus clientes, processa-os e retorna os resultados ao Cliente. O Cliente estabelece uma conexão com o Servidor, conectando-se a uma porta do Servidor na máquina onde este está sendo executado. É preciso saber qual o número da porta de uma aplicação antes de se iniciar um processo de conexão (SEIXAS, 2005).

Os serviços de *sockets* permitem a execução em ambientes heterogêneos envolvendo diferentes plataformas, como *Windows* e *Unix.*

Sockets apresentam uma grande aceitação no mercado e grande parte dos protocolos utilizados na Internet (HTTP, NNTP E SMTP) utilizam essa biblioteca.

Quando um *socket* é criado, o programa tem que especificar o endereço do domínio e o tipo de s*ocket* para que a conexão possa ser estabelecida. Dois processos podem se comunicar somente se seus sockets são do mesmo tipo e no mesmo domínio.

A utilização de s*ockets* representa uma boa solução para a escrita de programas Cliente-Servidor em ambientes que suportem a arquitetura TCP/IP. Com a utilização de *sockets*, a rede é vista como se fosse um arquivo, permitindo que se utilize, para a comunicação, primitivas simples como *read/write* ou *sendto/recvfrom.*

3.3 Ferramentas para Criação de Aplicações de RA

Conforme afirma Reitmayr (2004), desenvolver aplicações de RA ainda é uma tarefa desafiadora, até mesmo depois de alguns anos de pesquisa e demonstrações de aplicações construídas. As dificuldades surgem diretamente das propriedades básicas citadas por Azuma (1997) como: associar o real e o virtual, interação em tempo real e registros 3D[5].

Vários grupos de pesquisa têm desenvolvido uma série de aplicações para RA, que permite a criação de softwares para explorar as possíveis interfaces da RA.

3.3.1 *Coterie*

O sistema *Coterie* foi baseado no *Modula 3* e suporta cenas gráficas baseadas em programação gráfica e dispositivos de rastreamento. Foi base para os projetos Máquina de *Turing* e *Mars*. Depois ocorreu uma extensão para permitir aplicações de RA distribuída (REITMAYR, 2004).

---

[5] Consiste em alinhar objetos virtuais aos marcadores (AZUMA, 1999).

O objetivo principal do *Coterie* é oferecer aplicações em sistemas distribuídos e também compartilhamento de memória. As aplicações são construídas usando múltiplas *threads* que se comunicam com objetos compartilhados (PIEKARSKI; THOMAS, 2001).

O sistema possui uma biblioteca 3D integrada e camada de rastreadores que permite o usuário implementar e compilar ou interpretar programas para trabalhar com RA ou ambientes de RV.

Nessa aplicação as estruturas de dados complexos, como grafos de cena podem ser compartilhados facilmente.

3.3.2 *Tinmith*

O sistema *Tinmith* é uma arquitetura de software para aplicações de RA móveis, desenvolvida pela Universidade do Sul da Austrália. O fluxo de dados é o conceito dessa arquitetura, pois os dados de um sensor chegam ao sistema e é processado por uma série de camadas e então é feita a exibição. O *Tinmith* define todas essas camadas de processo e oferece uma solução completa para construir ambientes virtuais (PIEKARSKI; THOMAS, 2001).

Foi desenvolvido para aplicações orientada a objetos e para suportar hierarquia de grafos de cena. Implementado em C++, caracteriza um suporte para sistemas de informação e armazenamento de arquivo. Suporta renderização e pode ser composto de uma estrutura de grafo de cena para criar modelos geométricos.

É um software livre e permite trabalhar com rastreadores, sistema operacional *Linux*, ferramentas e bibliotecas GNU, compiladores C/C++, renderização com OpenGL (BRUGGE *et al*, 2002).

O *Tinmith* tem apresentado aplicações nas áreas militares, em jogos e pesquisas acadêmicas e seus principais objetivos são o aumento do desempenho de velocidade e da eficiência das aplicações de RA. (BRUGGE *et al*, 2002).

3.3.3 DWARF

O DWARF (*Distributed Wearable Augmented Framework*) foi desenvolvido pela Universidade Técnica de Munique, é baseada no conceito de trabalhar com *framework* para serviços colaborativos distribuídos. Um serviço é uma parte do software que é executado em um computador fixo ou móvel e permite uma determinada função, como um rastreador óptico. Os serviços podem ser conectados para usar as funções de outros serviços, formando assim uma rede de fluxo de dados.

O DWARF contém serviços para rastreamento de posição, renderização 3D, entrada e saída *multimodal* (BAUER *et al*, 2003).

Os *framework* podem ser facilmente estendidos para adicionar novos serviços ou melhorar os já existentes.

Os serviços são realizados como processos individuais ou *threads* e podem ser executados em computadores locais ou móveis conectados através de redes de computadores.

Para modelar o que um serviço pode oferecer a outro serviço, o *DWARF* usa o conceito de “necessidades” e “habilidades”. A combinação da habilidade de um serviço com a necessidade de outro conduz a uma conexão entre os serviços. Essa configuração é feita pelo gerente de serviços distribuídos (BAUER *et al*, 2003).

Habilidades descrevem a funcionalidade que um serviço oferece, como dados da posição de marcadores ópticos. O serviço pode ter várias habilidades como, por exemplo, um rastreador óptico que pode rastrear vários marcadores simultaneamente.

Necessidades descrevem a funcionalidade requerida de outros serviços. Por exemplo, um rastreador óptico precisa de uma seqüência e descrição de marcadores que deverão ser detectados. Precisa também do tipo, e somente habilidades do mesmo tipo podem satisfazer o que precisa.

Na camada *middleware* o recurso CORBA administra os serviços. Cada nó de rede do sistema DWARF tem um gerente de serviço, não há nenhum componente central. Cada gerente de serviço controla os serviços locais do nó e mantém descrições deles. As aplicações são criadas configurando os serviços disponíveis em rede que conectam as suas necessidades e habilidades.

Atualmente DWARF suporta as linguagens C++, Java e *Phyton* e executa nos sistemas operacionais *Linux*, *Windows NT*, 2000 e XP, e Mac OSX. (KLINKER, 2005 ; BRUGGE *et al*, 2002).

DWARF tem mostrado aplicações na área de pesquisa acadêmica e indústria (BRUGGE *et al*, 2002).

3.3.4 ARToolkit – *Augmented Reality ToolKit*

Um dos recursos de software, muito usado no desenvolvimento de aplicações de RA, é o ARToolkit. Além de ter seu código aberto e ser gratuito, esse software está disponível para uso em diversas plataformas de sistemas operacionais.

De acordo com Kato *et al.* (1999), o Artoolkit é uma biblioteca em linguagem C que permite aos programadores desenvolver aplicações de RA.

Atualmente, o *ARToolkit* executa nas plataformas SGI Irix, PC *Linux*, PC *Windows* 95/98/NT/2000/XP e Mac OSX. Existem versões para cada uma dessas plataformas.

A versão atual do ARToolkit oferece suporte para RA com visão direta por vídeo ou visão direta óptica. A RA com visão direta por vídeo é aquela cujas imagens virtuais são sobrepostas às imagens de vídeo adquiridas do mundo real. A outra alternativa é a RA por visão direta óptica, na qual os objetos virtuais são sobrepostos diretamente às imagens do mundo real percebidas pelo usuário. A RA por visão direta requer um dispositivo HMD e exige também um procedimento de calibração da câmera que adquire imagens do mundo real.

O pacote inclui bibliotecas de rastreamento e disponibiliza o código fonte completo, tornando possível o transporte do código para diversas plataformas ou adaptá-lo para resolver as especificidades de suas aplicações.

Os requisitos básicos de hardware necessários para desenvolver e executar aplicações do ARToolkit são uma câmera de vídeo e uma interface ou um dispositivo de aquisição de vídeo com seus respectivos drives. Em um *PC-Window* a captura de vídeo pode se dar por uma câmera USB, por um dispositivo de aquisição de vídeo ou por placas gráficas com alguma entrada de vídeo.

O ARToolkit usa técnica de visão computacional (captar a informação relativa à cena) para calcular o ponto de vista real da câmera e sua orientação em relação aos marcadores (KATO *et al.*, 1999).

A imagem de vídeo capturada é transformada em uma imagem binária (branca e preta - *thresholded)* . Depois, o ARToolkit encontra todos os quadrados na imagem binária, muitos dos quais não correspondem a marcadores de referência. Para cada quadrado, o desenho padrão é capturado e comparado com alguns gabaritos pré-treinados. Se houver alguma similaridade, então o ARToolkit considera que encontrou um dos marcadores de referência. O ARToolkit usa então o tamanho conhecido do quadrado e a orientação do padrão encontrado para calcular a posição real da câmera em relação a posição real do marcador. Uma matriz 3X4 conterá as coordenadas reais da câmera em relação ao marcador. Esta matriz é usada para

calcular a posição das coordenadas da câmera virtual. Se as coordenadas virtuais e reais da câmera forem iguais, o objeto 3D pode ser desenhado precisamente sobre o marcador real (Figura 3.5) (KATO *et al.*, 1999).

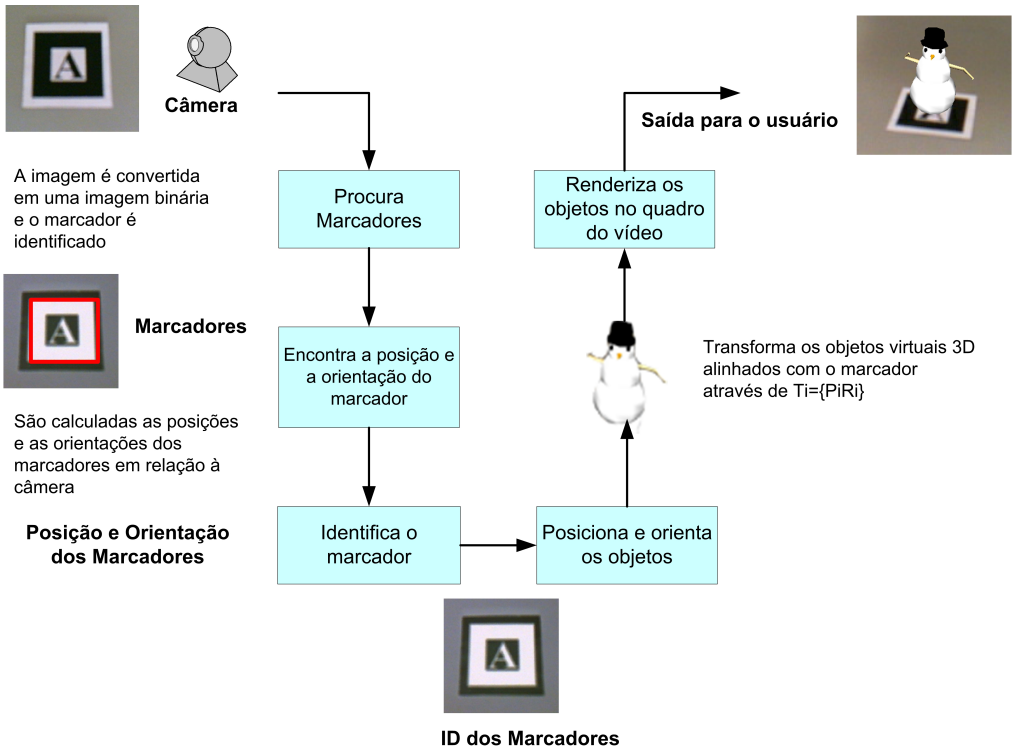


Figura 3.5: Técnicas de Rastreamento e Registros

Para que o objeto virtual apareça, toda a borda preta do quadrado que envolve o desenho padrão deve estar enquadrada pela imagem da câmera, assim como o próprio desenho padrão.

O desenvolvimento de aplicações de RA com ARToolkit requer que o desenvolvedor escreva a aplicação e treine as rotinas de processamento de imagens sobre os marcadores do mundo real que serão usadas na aplicação.

Para escrever aplicações com ARToolkit, Kato *et al.* (1999) descrevem as seguintes etapas (Figura 3.6):

- Inicialização do caminho dos parâmetros de vídeo;
- Leitura dos arquivos que contém os padrões dos marcadores;
- Captura do quadro de vídeo;
- Detecção dos marcadores e reconhecimento dos padrões;
- Cálculo da matriz transformação, contendo as informações de posicionamento e orientação da câmera em relação ao marcador;
- Desenho do objeto virtual;
- Encerramento da entrada do vídeo;

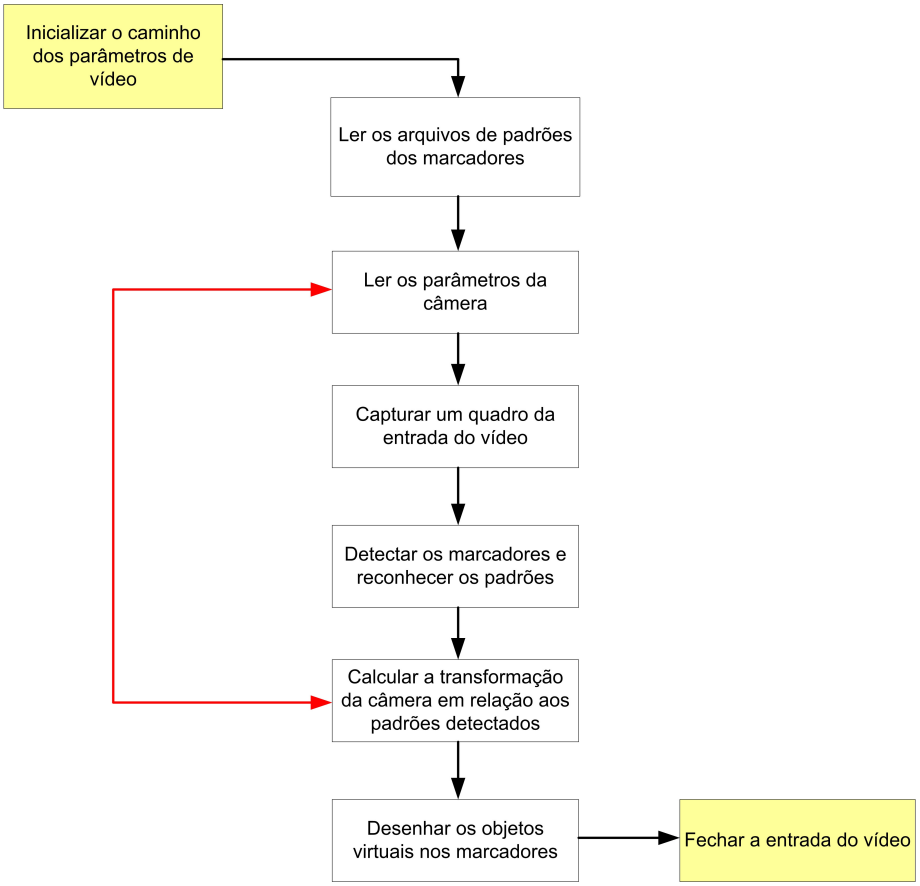


Figura 3.6: Etapas para o desenvolvimento de uma aplicação utilizando o ARToolkit

Além desses passos, a aplicação pode precisar responder ao mouse, ao teclado ou a outros eventos específicos da aplicação.

O ARToolkit está presente no desenvolvimento de diversas aplicações acadêmicas, industriais e sistemas colaborativos (BRUGGE *et al*., 2002).

3.3.5 *Studierstube – Environment for Collaboration in Augmented Reality*

Studierstube é um ambiente para desenvolvimento de aplicações colaborativas de RA. Esse ambiente foi desenvolvido pelo Instituto de Computação Gráfica da Universidade de Tecnologia de Viena. O ambiente *Studierstube* baseia-se na utilização de uma coleção de classes de C++ construídas com as melhores ferramentas gráficas da linguagem *Open Inventor* (BAUER *et al*., 2003).

O rico ambiente gráfico do *Open Inventor* permite rápida prototipagem de novas aplicações. O *Open Inventor* é um grafo de cena orientado a objetos que armazena as informações e a interação de objetos ativos.

Para rastrear a localização de dados o *Studiestube* utiliza a biblioteca *Open Tracker*. O maior objetivo é de separar a transformação exigida na manipulação de dados individuais e construir uma rede de fluxo de dados para operações genéricas**.**

O *Studierstube* suporta execução distribuída e oferece uma biblioteca gráfica compartilhada chamada *Distributed Open Invetor* (DIV) usando um protocolo *multicast* seguro.

Segundo Moeckel (2000), esse projeto da universidade de Vienna trata de um ambiente para trabalho colaborativo voltado a aplicações de visualização cientifica. A idéia é que o usuário possa interagir com um experimento e, ao mesmo tempo, visualizar os demais colaboradores, podendo assim conversar, apontar e gesticular a fim de discutir o fenômeno observado. O principal objetivo é buscar formas naturais e intuitivas de interação, sem o uso de dispositivos caros para visualização, rastreamento e interação.

O projeto *Studierstube* tenta provar como usar a interação 3D em um ambiente de trabalho onde várias tarefas são realizadas simultaneamente. O *Studierstube* possibilita suportar a produtividade, tipicamente associada à metáfora *desktop*, como a colaboração, tipicamente associada à aplicação de trabalho colaborativo suportado por computador.

Esse projeto utiliza a RA para gerar as imagens do mundo real. A RA utiliza dispositivos como HMDs ou telas de projeção para combinar objetos gráficos do computador com visões do usuário do mundo real, permitindo que múltiplos usuários compartilhem o ambiente virtual, sendo capaz de utilizar trabalho colaborativo suportado por computador.

Através de equipamentos como HMDs o usuário consegue visualizar objetos tridimensionais e escolher o ponto de vista para essa visualização. Observa-se na Figura 3.7, que os usuários possuem independência para a utilização do objeto compartilhado (HESINA, 2001).

Figura 3.7: Utilizando HMD para observar objetos virtuais (HESINA, 2001)

Ao contrário dos sistemas distribuídos de RV e jogos de redes que são baseados na metáfora egocêntrica do mundo virtual, a RA colaborativa permite vários usuários co-localizados experimentarem um trabalho virtual colaborativo através de HMDs ou de ambientes de projeção. No ambiente de trabalho virtual é possível ocorrer a comunicação e a interação tanto com objetos reais como virtuais (SCHAMLSTIEG, 2001).

O *Studierstube* tem a intenção de tentar acomodar as exigências dos padrões de redes de computadores, baseado em um sistema distribuído de RV. Usou-se para isso um grafo de cena compartilhado distribuído para esconder os detalhes de transmissão da rede ao programador da aplicação. Uma grande contribuição é a unificação de toda aplicação, sendo gráfica ou não, é representada no grafo de cena através de nós de aplicação. Os nós de aplicação são distribuídos pelo mesmo mecanismo que os nós em grafo de cena convencional. Um outro aspecto de contribuição observado é a migração de um nó de aplicação que permite uma administração do grupo de trabalho de forma dinâmica.

Nesse projeto, o usuário utiliza um PIP e um HMD transparente para poder interagir com o experimento e ao mesmo tempo visualizar os demais colaboradores.

O *Studierstube* tem mostrado suas características em diversas aplicações médicas, educação e pesquisas acadêmicas.

3.3.6 Comparação entre as Ferramentas Analisadas

O objetivo principal das ferramentas analisadas é oferecimento de recursos para criação de aplicações de RA, permitindo principalmente, a navegação, a visualização das informações, a interação em tempo real e o alinhamento tridimensional do real com o virtual.

Algumas ferramentas contribuem para a criação de aplicações que permitem participação simultânea de vários participantes, proporcionando a realização de trabalhos colaborativos e realizados em ambientes locais ou remotos.

A Tabela II sintetiza algumas características das ferramentas estudadas.

Tabela II: Comparação entre as ferramentas de RA

	Coterie	*Tinmith*	DWARF	ARToolkit	*Studierstube*
Linguagem	Modula 3	C++	Java, C++, *Phyton*	C, Java	C++
Tipo de Comunicação	Local/Remota	Local/Remota	Local/Remota	Local/Remota	Local/Remota
Sistema Operacional	*Unix, Windows* NT e 95, SGI *Irix*, *Solaris*, *Sun OS*	*Unix, Linux*	*Linux, Windows, Mac OSX*.	*Linux, Windows, Mac OSX SGI Irix*,	*Windows, SGI Irix*
Descrição do Mundo	Grafo de Cena	Grafo de Cena	Grafo de Cena	_	Grafo de Cena
Renderização	-	OpenGL	VRML	VRML/OpenGL	-
Tipo de Projeto	Pesquisa Acadêmica	Pesquisa Acadêmica e Militar	Pesquisa Acadêmica e Industrial	Pesquisa Acadêmica e Industrial	Pesquisa Acadêmica
Categoria	*Freeware*	Código Reservado	Código Reservado	*Freeware*	*Freeware*
Aplicações	RA Móvel e Médica	RA Móvel e jogos.	Produção e Manutenção.	Educação Entretenimento, Colaboração e outras	Educação, Médica. Colaborativa

Projeto como ARToolkit pode ser utilizado para calcular a posição e a orientação da câmera em relação aos marcadores. Devido essas funções, ferramentas como *Studierstube* e o *Tinimith* utilizam as bibliotecas do Artoolkit para realizar rastreamento de dispositivos.

O *Studierstube* focaliza suas aplicações na utilização de interação 3D e execução de várias tarefas simultaneamente, permitindo também aplicações colaborativas.

Tanto *Coterie* quanto *ARToolkit*, *Studierstube* e DWARF apresentam heterogeneidade em relação às plataformas de sistemas operacionais, ao contrário de *Tinmith* que centraliza suas aplicações na plataforma *Unix* e *Linux*.

Projetos como ARToolkit e *Tinmith* utilizam OpenGL para descrever seus objetos no mundo real, e assim serem capazes de renderizar objetos em sistema de RA. O uso de OpenGL apresenta algumas desvantagens por ser uma biblioteca de nível baixo para renderização 3D, de modo que ARToolkit pode utilizar a linguagem VRML para criação das cenas.

De acordo com as ferramentas analisadas, a escolha da ferramenta para criação da aplicação vai depender em grande parte dos objetivos requeridos, dos dispositivos e das características do sistema.

Um dos recursos de software que tem se tornado viável no desenvolvimento de aplicações para RA é o ARToolkit, pois apresenta a vantagem do seu código estar aberto e disponível para diversas plataformas de sistema operacional e possibilitar a utilização de diversos dispositivos de RA .

CAPÍTULO 4 – SUPORTE A INTERAÇÃO REMOTA PARA SISTEMAS DE REALIDADE AUMENTADA

Atualmente o computador tornou-se uma interface entre as pessoas, permitindo que estas possam se comunicar, interagir e compartilhar informações. O crescimento constante da informação incentiva a necessidade dos participantes desenvolverem tarefas em conjunto e compartilharem seus resultados para atingir um fim comum.

Conforme visto no Capítulo 3, os AVCs de RA têm o objetivo de prover um ambiente compartilhado e possibilitar aos usuários se beneficiarem de várias tecnologias como, por exemplo, a RA, o trabalho colaborativo e a interface tridimensional. Através do uso dessas tecnologias, o ambiente possibilitará aos participantes compartilhar seus conhecimentos, buscar idéias, informações e interagir uns com os outros.

Uma vantagem na utilização da tecnologia de RA é que esta oferece a possibilidade de criar ambientes, sem a necessidade de dispositivos de alto custo, como luvas, capacetes e CAVEs utilizados em aplicações de RV, tornando-se uma opção de tecnologia mais acessível, além de permitir uma interação mais próxima da realidade dos usuários.

Porém, dificilmente o participante terá o mesmo retorno tátil quando manipula objetos virtuais, se comparados com o mundo real; esta é uma das dificuldades encontradas em relação à interação, conforme analisado na Sub-Seção 1.7.

O uso da RA também apresenta alguns desafios tecnológicos citados por AZUMA (1997), como rastreamento de objetos reais e a calibração das sobreposições no ambiente combinado.

Nesse contexto, este Capítulo apresenta um suporte à interação remota para sistemas de RA distribuída, descrevendo a metodologia elaborada para a utilização de técnicas de RA,

as formas de interação nos objetos virtuais, o gerenciamento do controle de bloqueios e o modelo de comunicação com *sockets* UDP *multicast*.

4.1 Objetivos Principais

São objetivos do presente trabalho:

- O estudo e a implementação de mecanismos que forneçam suporte à comunicação entre os processos participantes de um AVC e a interação com objetos virtuais controlados a partir de técnicas de RA;
- Estudar recursos e funções da biblioteca ARToolkit, bem como bibliotecas relacionadas, como OpenGL e GLUI;
- Implementação de um protótipo para a validação dos mecanismos estruturados;
- Análise do desempenho do ambiente implementado.

Existem outros aspectos importantes relativos aos AVCs, tais como gerenciamento de nível de detalhe da cena virtual, tratamento de colisões, dispositivos de força para realizar interação, etc. Porém, a implementação dessas facilidades está fora do escopo do presente trabalho.

4.2 Estruturação do Suporte

Conforme mostra a Figura 4.1, os serviços e facilidades desenvolvidos encontram-se em uma camada denominada Camada de Suporte à Interação Remota, fornecendo uma interface definida para a camada superior, ou seja, o nível da Aplicação de RA Distribuída.

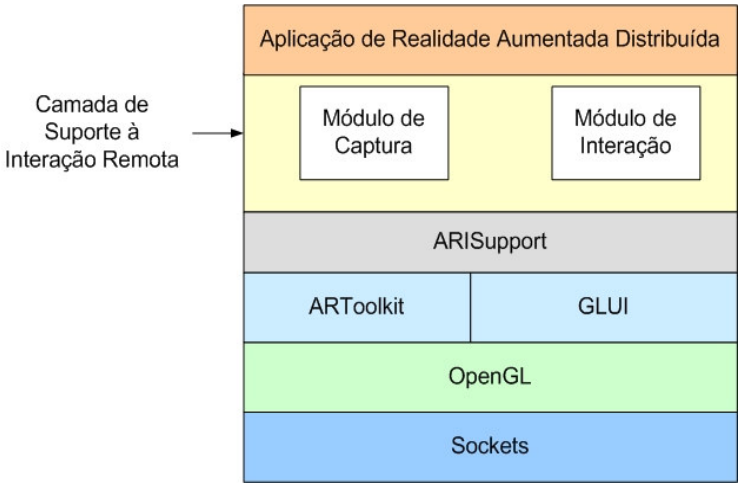


Figura 4.1: Diagrama da Arquitetura do Projeto

O suporte foi estruturado como uma biblioteca dividida em dois módulos integrados:

- **Módulo de Captura:** Oferece o gerenciamento e o controle da captura da cena real e a detecção dos marcadores. Para isso foram utilizadas técnicas de visão computacional através das funções da biblioteca ARToolkit, permitindo, dessa forma, ao participante sobrepor objetos virtuais aos marcadores.
- **Módulo de Interação:** Módulo que realiza o controle e o gerenciamento da interação dos objetos no ambiente virtual. De acordo com a Sub-Seção 1.6, a interação em AVCs deve permitir que os participantes envolvidos executem ações como selecionar e manipular objetos virtuais, conseguindo dessa maneira alterar a posição e a orientação dos objeto, bem como a navegação pela cena tridimensional.

Os módulos de Interação e de Captura estão relacionados entre si, pois, para que ocorra a interação dos participantes com os objetos virtuais, é necessário que o sistema de RA possua funções de suporte ao rastreamento das ações do usuário como, por exemplo, o movimento dos marcadores, utilizando técnicas de visão computacional incluídas na biblioteca ARToolkit.

A geração dos objetos virtuais foi feita a partir da utilização da API OpenGL, responsável pelo cálculo das coordenadas virtuais da câmera e desenho das imagens virtuais.

A interação do usuário com os objetos virtuais, como também a interface gráfica da aplicação, foi realizada por intermédio das funções da biblioteca GLUI (RADEMARCHER, 2005).

Utilizou-se da biblioteca ARISupport (*Augmented Reality Support*) (LOPES,2005) para realizar a interação por meio de pinças virtuais.

E, finalmente, para realizar a comunicação remota das aplicações, implementou-se uma estrutura baseada nos serviços de *socket*s, utilizando o modelo de comunicação *multicast,* conforme citado no Capítulo 3.

Como mostra a Figura 4.1, a camada superior é o ambiente desenvolvido e foi implementado pela contribuição das camadas que estão hierarquicamente subordinadas a ela. A Aplicação RA Distribuída baseia-se no modelo Cliente-Servidor e será discutida em detalhes na Sub-Seção 4.4.1.

4.3 Implementação da Camada de Suporte e Interação

Os componentes providos por essa camada e suas funcionalidades são apresentados a seguir:

4.3.1 Módulo de Captura

Este módulo é composto de várias funções como, por exemplo, configuração das propriedades do vídeo, captura da imagem, escolha de marcadores, ajuste da taxa de exibição, transformação de matrizes que serão explicadas a seguir:

Função para configuração das propriedades do vídeo

- **Função Início:** Responsável em definir os parâmetros para o dispositivo de vídeo e para ler os parâmetros iniciais para uma aplicação ARToolkit, como as

características da câmera que está sendo utilizada e os padrões que serão reconhecidos nos marcadores.

Funções para realizar eventos de vídeo

A função Evento_Vídeo é a rotina que realiza as etapas de captura de vídeo, detecção de marcadores e reconhecimento de seus padrões, cálculo das matrizes transformações em relação aos padrões detectados e a escolha dos diferentes marcadores utilizados na aplicação. Estão presentes na rotina Evento_Vídeo as seguintes funções:

- **Função para capturar vídeo:** O quadro de vídeo é capturado usando a função *arVideoGetImage.*
- **Função para detectar marcadores***:* Utiliza a função *arDetectMarker* para detectar os padrões corretos nos marcadores. Essa função possui informações sobre o marcador (visibilidade do marcador, matriz transformação) e o número de marcadores detectados. A Figura 4.2 ilustra o código para detectar marcadores.

```
if (arDetectMarker(dataPtr, 100, &InfoMarca, &NMarca) < 0)
   {
      Limpar();
      exit(0);
   }
```

Figura 4.2: Função para detecção de marcadores

- **Função para calcular matriz transformação:** Através da função *arGetTransMat* é encontrada a posição real da câmera e sua orientação em relação ao marcador obtendo a matriz transformação.

Nessa rotina (Evento_Vídeo) a função para detectar marcadores compara os padrões dos marcadores com padrões pré-definidos. Esses padrões são carregados em tempo de execução e ficam armazenados em um diretório. Nesse diretório, um arquivo especifica quais

marcadores devem ser reconhecidos e especifica os padrões que serão associados a cada marcador.

Esse arquivo começa com número de marcadores que serão identificados pela aplicação, seguidos de uma estrutura de dados para cada marcador, como o nome do padrão utilizado no programa de RA para associação do marcador, o tamanho em milímetros do marcador e as coordenadas X Y referentes ao centro do marcador (Figura 4.3).

```
#Número de Marcadores
9
#Marca 1
MarcaA
dedal1
20
00
#Marca 2
MarcaB
dedal2
20
00
#Marca 3
MarcaB
Patt.hiro
40
00
#Marca 4
MarcaD
patt.c
40
0 0
#Marca 5
MarcaE
patt.d
40
0 0
#Marca 6
MarcaF
patt.a
40
0 0
#Marca 7
MarcaG
patt.b
40
0 0
#Marca 8
MarcaH
retang
40
0 0
#Marca 9
MarcaI
patt.g
40
0 0
```

Figura 4.3: Estrutura de padrão do marcador no arquivo Marcadores

Para criação de novos padrões de reconhecimento, utilizou-se um programa chamado mk_patt, incluso no pacote de instalação do ARToolkit. Esse programa cria esses arquivos de padrões pré-definidos.

- **Função para opção de marcadores na aplicação servidora:** Responsável pela escolha de diferentes marcadores utilizados na aplicação e chamada da função para renderização dos objetos virtuais alinhadas no marcador, de acordo com padrão definido. (Figura 4.4).

```
if (strcmp(DObjt[k].Nome,"MarcaD")==0) /* Desenha objeto na marca D, referente a padrão patt.b*/
{
    RenderizaObjeto3D(0);
}
if (strcmp(DObjt[k].Nome,"MarcaE")==0) /* Desenha objeto na marca E, referente a padrão patt.d */
{
   RenderizaObjeto3D(0);
}
```

Figura 4.4: Função para opção de marcadores

4.3.2 Módulo de Interação

A interação em AV é desenvolvida com a intenção de prover ações como, por exemplo, seleção ou manipulação. Esse módulo oferece o controle e o gerenciamento da interação dos objetos no AV. A interação pode acontecer por meio por meio de marcadores, interface de controle 2D e pinças virtuais.

Interação por meio de marcadores

O uso de marcadores é uma das formas de conseguir interação em sistema de RA, pois por meio de marcadores é possível definir coordenadas espaciais e de orientação dos objetos a partir do ponto de vista do usuário, além de identificar alterações de posicionamento.

Quando se utiliza interação com marcadores, são usadas as funções do Módulo de Captura para detecção dos marcadores e o próprio cenário real com as ações realizadas pelo usuário.

Em ambientes de RA, o mundo real é “aumentado” com informações que não estão presentes na cena capturada, por meio de adição de objetos virtuais, e o usuário passa a ser um elemento participativo no cenário em que imagens reais são combinadas com virtuais para criar uma percepção aumentada (AZUMA, 2001).

O usuário poderá rotacionar e transladar os marcadores e conseqüentemente interagir com o objeto virtual, pois este realizará os mesmos movimentos do marcador, como se um estivesse preso ao outro (Figura 4.5).

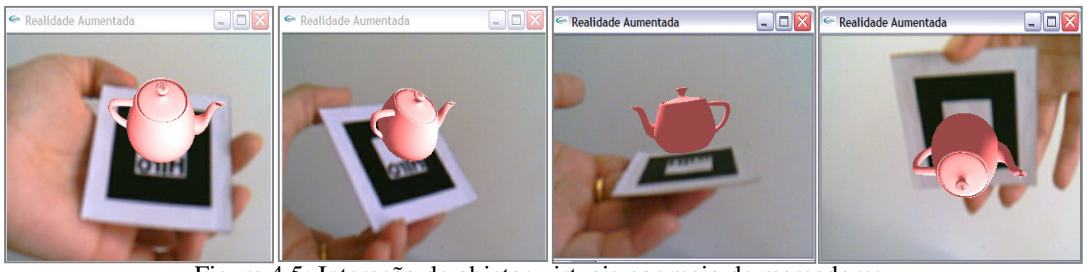


Figura 4.5: Interação de objetos virtuais por meio de marcadores

Dessa forma, a interação depende do cenário real, do objeto virtual e do marcador. Quando a aplicação é executada, a imagem capturada pela câmera aparece na tela do monitor. No momento em que a câmera captura a imagem do marcador, as funções da biblioteca ARToolkit permitem que ocorra o posicionamento do objeto virtual sobre o marcador no cenário real, combinando as imagens. Quando o usuário movimenta o marcador, o objeto virtual acompanha este movimento, permitindo sua manipulação com as mãos.

A interação por meio de marcadores apresenta algumas limitações. Os objetos virtuais só podem ser exibidos quando os marcadores estiverem visíveis. Essa dependência do rastreamento limita o tamanho dos objetos virtuais e seus movimentos. Portanto, se os usuários cobrirem parte do padrão com suas mãos ou outros objetos, os objetos virtuais desaparecerão (Figura 4.6). Há limites também na inclinação dos marcadores. Caso o usuário incline muito o marcador em relação ao eixo óptico da câmera, o padrão inscrito no marcador pode se tornar irreconhecível pelo programa. Conforme os marcadores se inclinam, os padrões do centro serão menos visíveis e o reconhecimento tornar-se-á menos confiável (AZUMA, 2001).

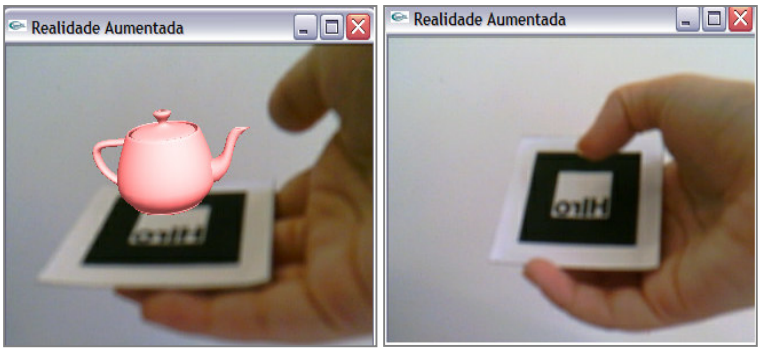


Figura 4.6: Marcador sendo encoberto durante a interação na aplicação

Os resultados do rastreamento do marcador também são afetados por condições de iluminação. O ambiente com muita luz ou falta de iluminação pode tornar difícil a tarefa de reconhecimento dos marcadores.

Existem ainda problemas relativos a limitações de espaço. Quanto maior for o tamanho físico do marcador, maior será a distância da qual poderá ser detectado e, portanto, maior será o volume do espaço de interação no qual o usuário poderá ser rastreado (AZUMA, 2001).

Por isso, existe a necessidade de planejar os tamanhos adequados dos marcadores, a iluminação do ambiente e verificar o estado dos marcadores, pois os marcadores estáticos causam menos problemas de reconhecimento do que aqueles que são movimentados.

Interface de Controle 2D

Esse método representa a categoria de interação através de controles 2D.

A seguir são explicadas as funções responsáveis para criação desse tipo de controle e gerenciamento das ações do usuário em relação aos controles.

a) **Função para criação da Interface Gráfica:** Responsável pela criação da interface gráfica da aplicação, como controles de escala, rotação, botões, caixa de seleção e inserção da captura da imagem dentro do formulário da interface (Figura 4.7).

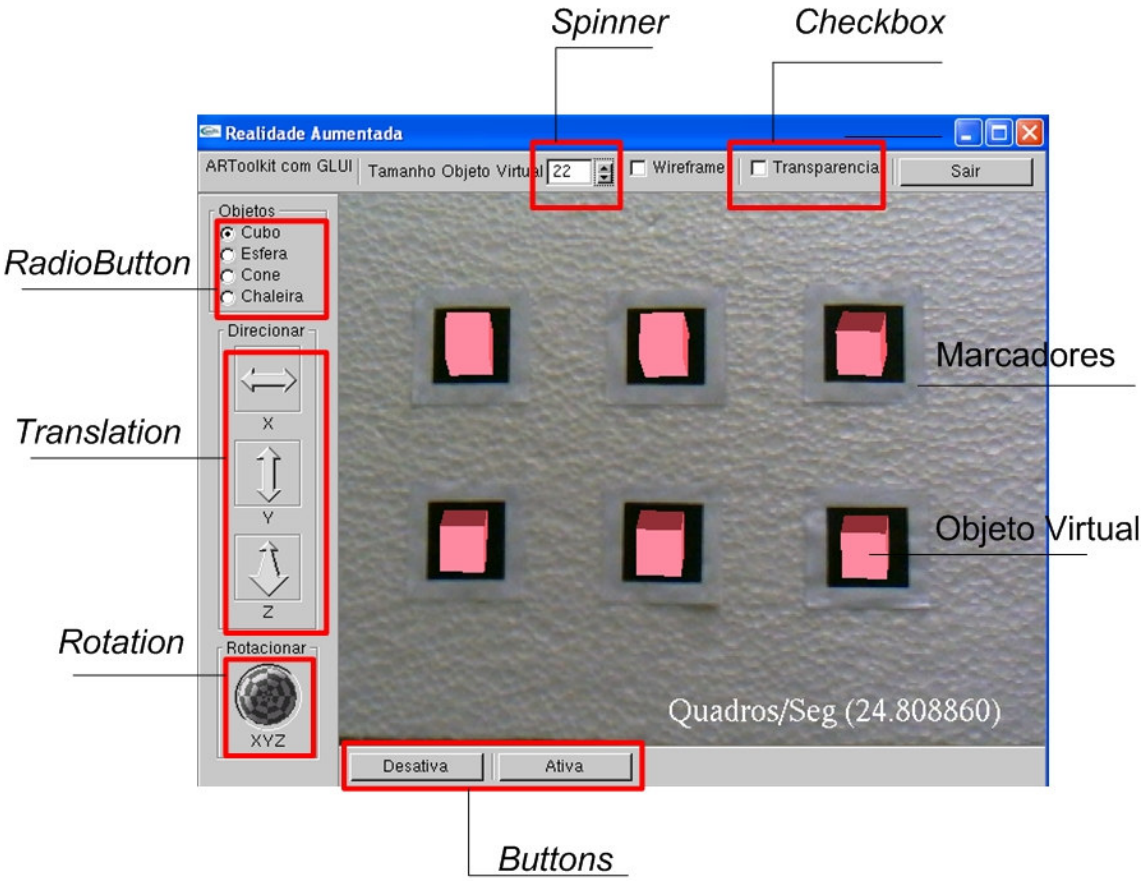


Figura 4.7: Interface de Controle 2D

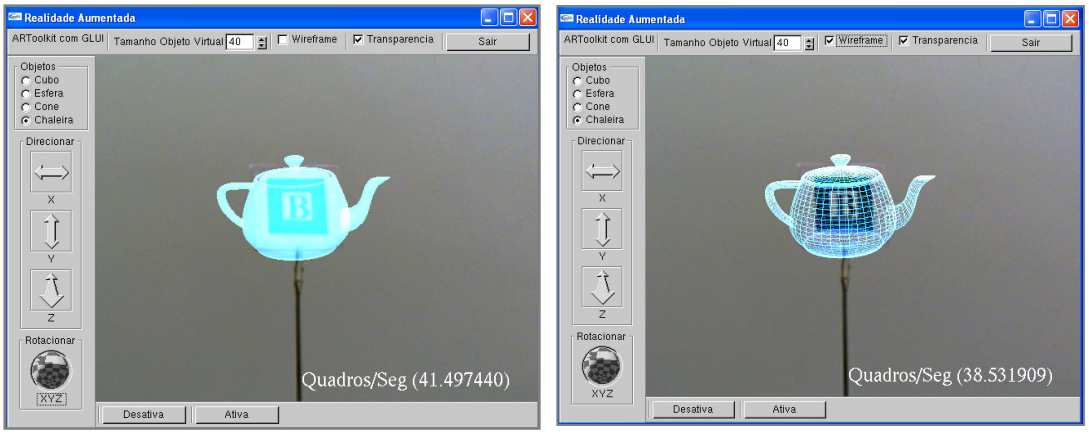
O processo de interação foi desenvolvido utilizando-se a biblioteca GLUI (*Graphic Library User Interface*), que permite a construção de controles como botões, menus, caixas de seleção, entre outros (RADEMARCHER, 2005). A interação é feita através de comandos realizados a partir dessa interface gráfica sobre o objeto virtual. A imagem capturada pela câmera é projetada dentro da tela central. Á esquerda, no topo e abaixo, foi criado um painel onde os controles são desenhados.

Os controles fornecem ao usuário ações como a mudança da forma dos objetos virtuais como, por exemplo, um cubo, uma esfera, um cone ou uma chaleira, por meio de uma caixa de seleção. Os controles podem afetar apenas um marcador ou até seis marcadores. Outros controles podem direcionar o objeto no eixo XYZ e rotacioná-lo, ou seja, mudar a

orientação e a posição do objeto virtual. Também permitem a alteração da escala dos objetos e a mudança do seu formato para *wireframe*, bem como a aplicação de transparência.

Os controles são explicados a seguir:

- **Caixa de Seleção para Tipo de Objeto Virtual:** Criou-se uma caixa de seleção (*radiobutton*) que permite ao usuário escolher qual o tipo de objeto virtual que será desenhado sobre o marcador em tempo real
- **Controle de Translação:** De acordo com os três eixos coordenados (XYZ), foram implementados três controles (*translation*) para realizar o deslocamento do objeto virtual em relação ao marcador.
- **Controle de Rotação:** A partir desse controle (*rotation*) é possível rotacionar o objeto virtual que está ativo no momento.
- **Controle de Escala do Objeto Virtual:** Criou-se um controle (*spinner*) que permite a alteração da escala do objeto.
- **Controle Função Transparência:** Foi adicionado na interface um controle (*checkbox*) que permite introduzir efeitos de transparências sobre o objeto. Um exemplo do resultado da utilização dessa função é demonstrado na Figura 4.8 (a), na qual se consegue ver o marcador sobreposto por uma chaleira virtual.
- **Controle de Exibição em *wireframe*:** Implementou-se um controle (*checkbox*) que habilita a eliminação das superfícies sólidas dos objetos 3D desenvolvidos em OpenGL (Figura 4.8 (b)).
- **Botões para controle de bloqueio dos objetos:** Com o uso do componente *buttons*, implementaram-se botões que permitem aos usuários interagir com os objetos virtuais.



(a) Transparência (b) Transparência e *Wireframe*

Figura 4.8: Uso dos controles de Interface 2D

b) Função para gerenciar o uso dos controles e renderizar os objetos virtuais: Essa função foi implementada para criar a renderização dos objetos de acordo com o uso dos controles da interface gráfica, por exemplo, alteração do tipo de objeto, escala dos objetos, posicionamento etc. A Figura 4.9 ilustra trecho do código para renderizar os objetos virtuais.

```
if (Objt==0) //testa o tipo do objeto do radiogroup
{
        if ((wire==1) &  (transp==1)) //testa estado do objeto e aplicação de transparência
        {
                glEnable (GL_BLEND);
                glDepthMask (GL_FALSE);
                glBlendFunc (GL_SRC_ALPHA, GL_ONE);
                glColor3f(1.9f, 0.6, 0.7f);
                glutWireCube(esc);
                glDepthMask (GL_TRUE);
                glDisable (GL_BLEND);
        }
        if ((wire==1) &  (transp==0)) //testa estado do objeto e aplicação de transparência
        {

                glColor3f(1.9f, 0.6, 0.7f);
                glutWireCube(esc);

        }
```

Figura 4.9: Verificação de controles e renderização

Interação por meio de Pinças Virtuais

Esse método permite ao usuário interagir diretamente com os objetos virtuais utilizando pinças virtuais. Para conseguir realizar a interação utilizou-se da função abaixo presente na biblioteca ARISupport (LOPES, 2005).

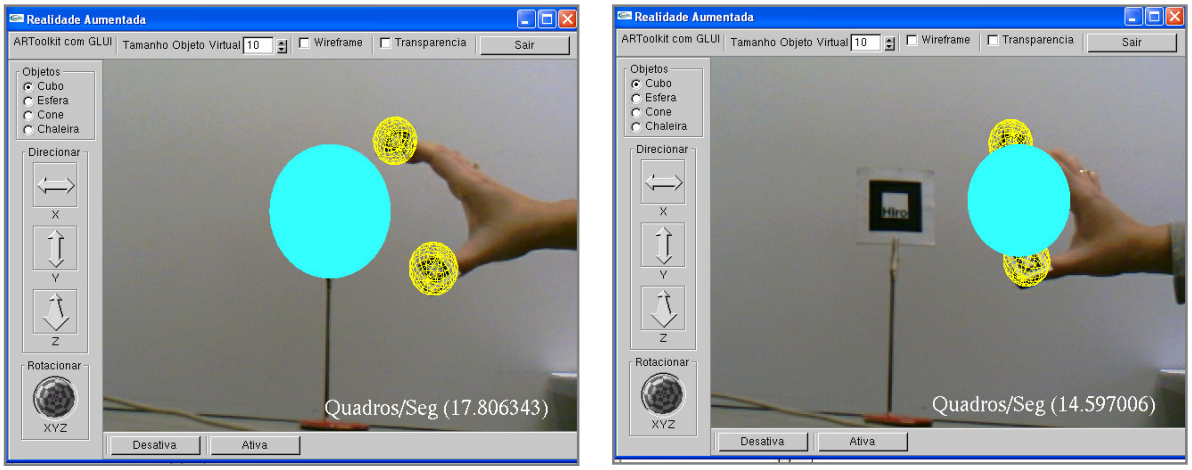
- **Função de Colisão entre esferas:** Nessa função, a interação é representada por meio do método de detecção de colisão, cuja função é inscrever cada objeto, ou parte deste, em uma esfera e verificar se esta intercepta outra. Com o uso desse método é apenas necessário verificar se a distância entre os centros das duas esferas é menor do que a soma dos raios de ambas, o que indica a ocorrência da colisão (LOPES, 2005).

As formas de interação com as esferas foram feitas por meio de dois marcadores acoplados aos dedos, os quais representam objetos de interação e um marcador que representa um objeto fixo, como ilustra a Figura 4.10.

(a) Marcadores acoplados aos dedos (b) Marcador fixo

Figura 4.10: Objetos de Interação

Esses marcadores (Figura 4.11(a)) foram sobrepostos por esferas virtuais, onde após a colisão entre as duas esferas (Figura 4.11(b)), a esfera fixa passa a ter o mesmo comportamento herdado das ações dos marcadores que simulam os dedos do usuário.



(a) Marcadores com as esferas sobrepostas (b) Objeto virtual sendo movido

Figura 4.11: Interação por meio de Pinças Virtuais

Essa aplicação necessita de planejamento em relação à luz ambiente, pois caso a iluminação não esteja adequada, aumenta a dificuldade de manipulação do objeto.

O módulo de interação também possui funções para permitir o gerenciamento de acesso à interação dos objetos virtuais, permitindo que os usuários participem da aplicação ou apenas a visualize, conforme é explicado na seção a seguir.

Função para Controle de Bloqueio

Para que a interação entre as aplicações ocorra de forma correta e não permita que uma aplicação interfira na outra, implementaram-se funções para controlar o acesso aos controles disponíveis nas janelas GLUI e com isso possibilitar que cada usuário manipule seu objeto separadamente.

- **Função para verificar pedido de controle**: Essa função faz o controle de bloqueios para determinar qual máquina Cliente terá o controle de interação sobre os objetos virtuais. A Figura 4.12 apresenta o algoritmo utilizado.

```
Início da Função
{
se aplicação cliente não faz pedido de interação
 {
          se aplicação servidora enviou permissão de pedido de interação
          {
                    se aplicação cliente não tem interação
                    {
                              aplicação servidora envia liberado para pedido de interação
                              desativa interface de interação na aplicação servidora
                    }
          se aplicação servidora pede permissão de interação
          {
                   se aplicação cliente não tem interação
                   {
                    aplicação servidora envia negado para pedido de interação
                    ativa interface de interação na aplicação servidora
                   }
          }
         se aplicação servidora tem acesso de pedido bloqueado
          {
                    se aplicação cliente liberou interação de objetos
                    {
                              aplicação servidora envia pedido de bloqueio negado
                              ativa interface no servidor
                    }
          }
 }

 se aplicação cliente faz pedido de interação
 {
          se aplicação servidora permite aplicação cliente fazer pedido de interação
          {
              aplicação servidora envia acesso de pedido bloqueado
              aplicação servidora recebe a identificação do usuário
              aplicação servidora envia controle de interação para a identificação que pediu bloqueio
          }
 }
Fim da Função
}
```

Figura 4.12: Pseudocódigo da Função para verificar pedido de controle

- **Função para realizar pedido de controle:** Realiza o pedido de controle para realizar a interação sobre os objetos virtuais, utilizando a interface 2D construída com a biblioteca GLUI, conforme ilustra a Figura 4.13.

```
Início da Função
{
se aplicação servidora enviou acesso liberado para fazer pedido
{
   se aplicação cliente aciona botão para fazer pedido para realizar interação
     {
          aplicação cliente envia pedido
          aplicação cliente envia sua identificação
      }
se  aplicação servidora envia pedido liberado
     {
        se identificação que aplicação servidora enviou for igual a identificação local da máquina
          {
             ativa interface gráfica do cliente
             aplicação cliente envia que controle de interação está com ele
           }
          senão
          se  identificação que aplicação servidora enviou for diferente da identificação local da máquina
          {
                    aplicação cliente envia que não quer mais pedido de interação
           }
se aplicação cliente desativa controle e acesso de pedido ao aplicação servidora estiver negado
{
          desativa interface gráfica de aplicação cliente
          aplicação cliente envia que não quer mais pedido de interação
          controle de interação não pertence mais aplicação cliente

}
}
Fim da Função
```

Figura 4.13: Pseudocódigo da Função para realizar pedido de controle

4.4 Implementação da Aplicação de RA Distribuída

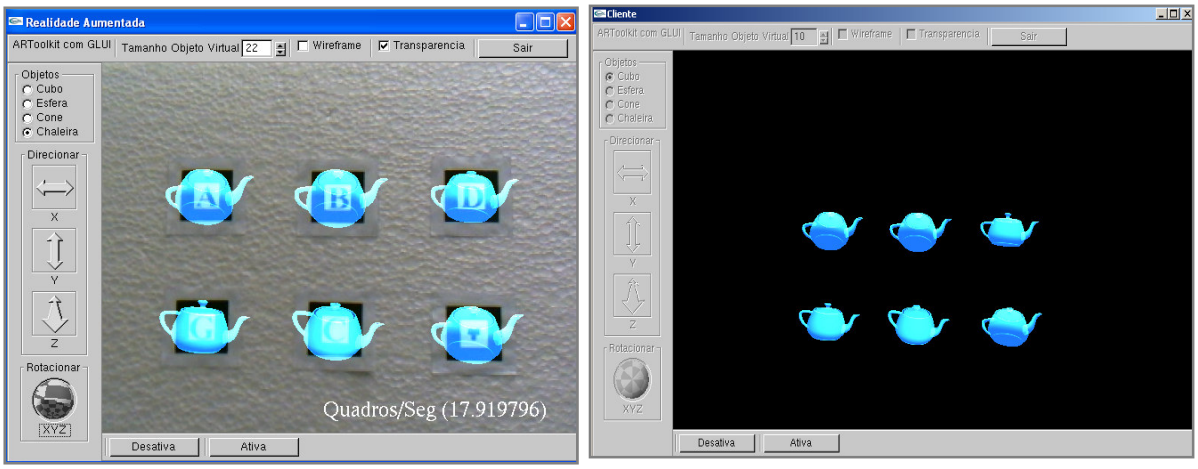
O sistema consiste no uso de técnicas de RA para combinação de imagens reais e virtuais, juntamente com a interação remota de objetos virtuais.

A aplicação de RA Distribuída utiliza os módulos da camada de interação remota, descrita na Seção 4.3, que permite ao usuário manipular os objetos virtuais por meio de marcadores, interface de controle 2D ou por meio de pinças virtuais e transmitir as interações realizadas para outros usuários da aplicação.

As interações sobre os objetos virtuais são transmitidas para todos os participantes do sistema, possibilitando que qualquer usuário possa visualizar a aplicação, e até mesmo interagir com os objetos visualizados.

Apenas o participante que possui a câmera para captura de imagens reais poderá realizar interação por meio de marcadores e pinças virtuais.

Conforme ilustra a Figura 4.14, os objetos virtuais são visualizados em tempo real por todos os participantes da aplicação, possibilitando a estes a interação sobre os objetos.



(a) Aplicação Servidora (b) Aplicação Cliente

Figura 4.14: Aplicação de Realidade Aumentada Distribuída

4.4.1 Estrutura da Aplicação

A aplicação de RA distribuída utiliza o modelo Cliente/Servidor, em que as máquinas clientes e servidora são conectadas entre si por uma rede baseada na arquitetura TCP/IP. A comunicação ocorre através dos serviços de *sockets* UDP *multicast*. Dessa maneira a informação é enviada para um endereço de grupo e todas as estações do grupo a recebem.

A aplicação servidora realiza a captura de imagens reais através de uma câmera digital e envia o posicionamento e orientação dos objetos virtuais aos participantes do sistema, como também as modificações feitas nos objetos. Tem a função de controlar a

permissão de acesso aos objetos virtuais (controle de bloqueios) e receber as interações realizadas da aplicação cliente.

A aplicação cliente permite receber as informações vindas da aplicação servidora e também possibilita ao participante solicitar o controle de acesso para interagir com o objeto virtual (Figura 4.15).

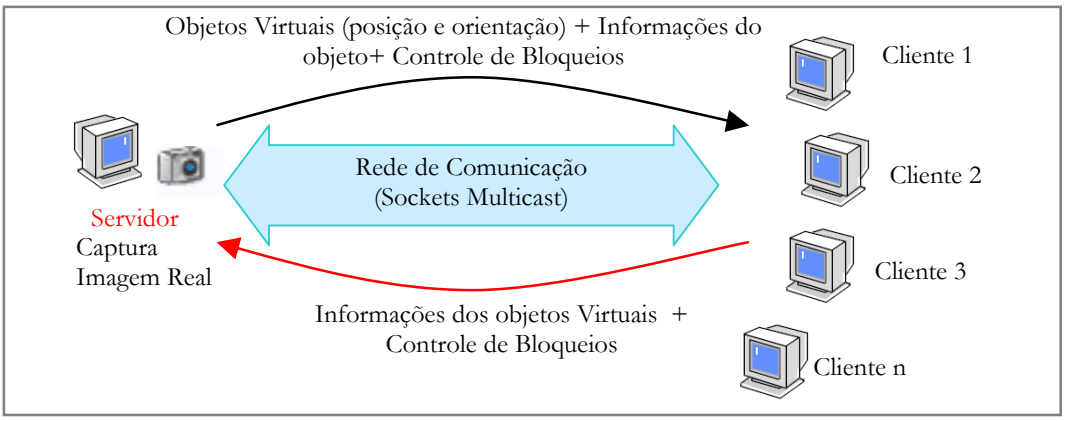


Figura 4.15: Arquitetura da Comunicação Cliente – Servidor

Comunicação

Para ocorrer a interação remota, utilizou-se serviço de *socket*s, que consiste em uma biblioteca de funções do *Windows* (*Winsock* API). A biblioteca *Winsock* permite escrever aplicações com protocolos compatíveis de qualquer fabricante.

Esse serviço apresenta a utilização de DATAGRAM *SOCKET*S (Seção 3.2) ou modo orientado à datagrama, que funciona sobre o protocolo UDP. O UDP é um protocolo de transporte que oferece um serviço de comunicação não orientada à conexão e sem garantia de entrega. O DATAGRAM *SOCKET*S é mais simples do que a comunicação à conexão TCP (STREAM *SOCKET*S), porque o Cliente não precisa aguardar a aceitação da conexão do Servidor, e por isso apresenta maior rapidez na troca de mensagens se comparado com o

modo orientado à conexão, tornando-se muito importante, já que o realismo nas aplicações de RA é um fator de extrema importância.

O suporte utiliza a comunicação *multicast* para envio dos dados. Essa forma de comunicação é semelhante à forma de enviar por *broadcast*; a diferença é que os dados devem ser enviados para um endereço de grupo *multicast*.

As funções utilizadas são: *socket*( ) para criar uma conexão fim-a-fim, *bind*( ) para associar uma identificação ao *socket*, *sendto*( ) para enviar os dados (origem) e o *recvfrom*( ) para receber os dados (destino) e a *close*( ) encerrar a conexão. A Figura 4.16 mostra o fluxo de controle de *sockets* UDP, em uma aplicação Cliente-Servidor.

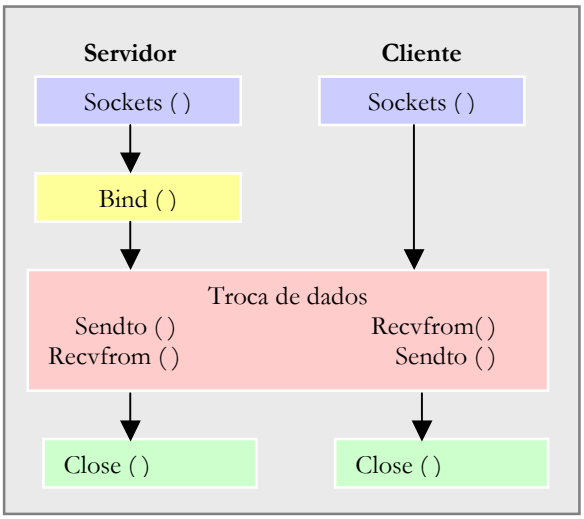


Figura 4.16: Utilização de Sockets UDP

O primeiro procedimento na utilização de *socket*s, utilizando a API *Winsock* é a sua inicialização, que se dará com a criação do objeto *WSADATA*, que consiste em uma estrutura contendo informações sobre a implementação de *socket*s para *Windows*, conforme mostra a Figura 4.17.

```
WSADATA wsaData;
```
Figura 4.17: Declaração da estrutura WSADATA

Abaixo são explanadas as funções para realizar a comunicação entre as aplicações:

- **Função para conexão**: O início da aplicação ocorrerá com a chamada da função *WSAStartup*, que inicializa o uso da biblioteca *WS2_32.lib*. A *WSAStartup* possui o parâmetro *MAKEWORD (2,1)* que solicita a versão do *WinSock* no sistema e fixa a versão passada como a mais alta suportada pelo *Windows Sockets* (Figura 4.18).

```
rc=WSAStartup (MAKEWORD(2,1), &wsaData);
if (rc!=0)
{
printf("Erro na inicialização do Winsock: %d\n", rc);
return 1;
}
else
{
printf ("Winsock inicializado!\n");
}
```

Figura 4.18: Inicialização do Winsock

Posteriormente é chamada a função *socket*, que é a responsável pela criação do *socket*, especificação da família de protocolos a ser usada com o *socket*, o tipo de comunicação (SOCK_STREAM ou SOCK_DGRAM) e a especificação de um tipo de protocolo de transporte particular usado com o *socket*, a Figura 4.19 ilustra a utilização da função *socket*.

```
s=socket (AF_INET, SOCK_DGRAM,0);
  if(s==INVALID_SOCKET)
       {
          printf("Erro na criação do socket:%d\n", WSAGetLastError( ));
          return 1;
       }
      else
       {
          printf("Criado socket UDP!\n");
       }
```

Figura 4.19: Criação de um Socket

Conforme ilustra a Figura 4.20, na aplicação cliente faz-se necessário a criação de uma estrutura para referenciar os elementos do endereço do *socket* (*destAddr.sin*), onde consta a família do endereço (AF_INET), o número da porta (1234) utilizada para enviar dados, e o

endereço IP *multicast* para enviar os dados (234.5.6.7). Esse endereço deve ser classe D (224.0.0.0 a 239.255.255.255). O endereço "224.0.0.1" é reservado, agrupando todos os *socket*s IP *multicast*.

```
destAddr.sin_family=AF_INET; /*usa endereço da familia de protocolos da Internet*/
destAddr.sin_port=htons(1234); /* Porta utilizada para enviar dados*/
destAddr.sin_addr.s_addr=inet_addr("234.5.6.7"); /* endereço IP Multicast para enviar dados*/

stIpMreq.imr_multiaddr.s_addr=inet_addr("234.5.6.7");
stIpMreq.imr_interface.s_addr=INADDR_ANY;
```

Figura 4.20: Associação dos elementos de endereço para a estrutura *multicast* Cliente

Na aplicação servidora, como demonstra a Figura 4.21, utiliza-se também a estrutura destAddr.sin, modificando apenas o destAddr.sin_addr.s_addr, fazendo com que o valor seja automaticamente igual ao endereço IP da máquina em que o processo é executado (INADDR_ANY).

```
destAddr.sin_family=AF_INET;
destAddr.sin_port=htons(1234);
destAddr.sin_addr.s_addr=htonl(INADDR_ANY);
```

Figura 4.21: Associação dos elementos de endereço para a estrutura *multicast* Servidor

Para o Servidor aceitar uma solicitação do Cliente será necessário associar o endereço local com a *socket* por meio da função *bind* (Figura 4.22).

```
rc=bind(s,(SOCKADDR*)&destAddr, destAddrLen);
   if (rc==SOCKET_ERROR)
     {
          printf("Erro no Bind:%d\n", WSAGetLastError( ));
          return 1;
     }
     else
         {
          printf("Socket na porta 1234 inicializado\n");
         }
```

Figura 4.22: Uso da função bind

Para recepção de dados por *multicast*, a aplicação precisa subscrever o *socket* ao endereço *multicast*, ou seja, adicionando a um ou mais grupos. A subscrição é registrada,

chamando a função *setsockopt* ( ) com a opção do IP _ ADD_MEMBERSHIP do nível IPPROTO_IP (Figura 4.23 (a)). É possível associar um valor de tempo limite (TTL – Time To Live) para as rotas a serem percorridas, que pode ser controlado usando a opção IP_MULTICAST_TTL, O valor do TTL pode ser especificado, invocando a função *setsockopt*() antes de enviar a mensagem (Figura 4.23 (b)). A chamada *setsockopt* (s, SOL_*SOCKET*, SO_REUSEADDR, (CHAR*)&one, sizeof(one)) permite que a máquina local use a mesma porta para receber os dados por diferentes endereços de grupos *multicast* (Figura 4.23 (c)).

```
rc=setsockopt(s,IPPROTO_IP, IP_ADD_MEMBERSHIP, (char *) &stIpMreq, stIpMreqLen);
    if (rc==SOCKET_ERROR)
       {
          printf("Erro no Set:%d\n", WSAGetLastError());
          return 1;
        }
      else
       {
          printf("Aguardando Conexão da Aplicação 1!!\n");
       }
```

(a) Associação do endereço IP *multicast* ao grupo.

```
unsigned char ttl=31;
setsockopt (s, IPPROTO_IP, IP_MULTICAST_TTL, (CHAR*)&ttl, sizeof(ttl));
```

(b) Tempo para o datagrama percorrer um percurso *Time To Live*(TTL).

```
int one=1;
setsockopt (s, SOL_SOCKET, SO_REUSEADDR, (CHAR*)&one, sizeof(one));
```

(c) Reutilização de portas.

Figura 4.23: Demonstração da função setsockopt

- **Função para enviar dados:** Utilizou-se a função *sendto*( ) para possibilitar a comunicação por meio de *socket*s de datagramas, conforme ilustra a Figura 4.24.

```
if (sendto(s, (char*) &l,  sizeof(info), 0,(SOCKADDR*)&destAddr,sizeof(SOCKADDR)) < 0)
{
        printf ("Erro ao enviar estrutura");
        exit(1);
 }
```

Figura 4.24: Envio de mensagens

- **Função para receber dados**: Para receber as informações utilizou-se a função *recvfrom*( ), demonstrada na Figura 4.25.

```
if ((numbytes=recvfrom(s, (char*)&b, sizeof(infoframe) , 0,(SOCKADDR*)&destAddr, &destAddrLen)) == -1)
{
        printf(" Erro ao receber estrutura");
        exit(1);
}
```

Figura 4.25: Recebimento de mensagens

- **Função para fechar o sockets***:* Quando um processo acaba de usar um socket, ele chama o recurso *closesocket* que possui a seguinte forma (Figura 4.26):

```
Closesocket(s);
```

Figura 4.26: Demonstração de comando para fechar um *Sockets*

Aplicação Servidora

A aplicação servidora utiliza a biblioteca ARToolkit para calcular o ponto de vista real da câmera em relação a um marcador no mundo real. Após capturar a imagem são realizados os testes de similaridade com marcadores e transformações da posição da câmera.

Essa aplicação armazena a matriz transformação (matriz 3X4) em uma estrutura de dados e a envia para o Cliente, incluindo a identificação do marcador, a visibilidade dos objetos e as informações referentes à interação com o uso do controle 2D (janela GLUI), e logo após renderiza o objeto virtual através da API OpenGL.

A aplicação servidora faz o controle do acesso aos objetos virtuais para permitir que os participantes do sistema interajam com os objetos.

A Figura 4.27 ilustra a aplicação servidora enviando as informações do objeto virtual a um grupo *multicast*.

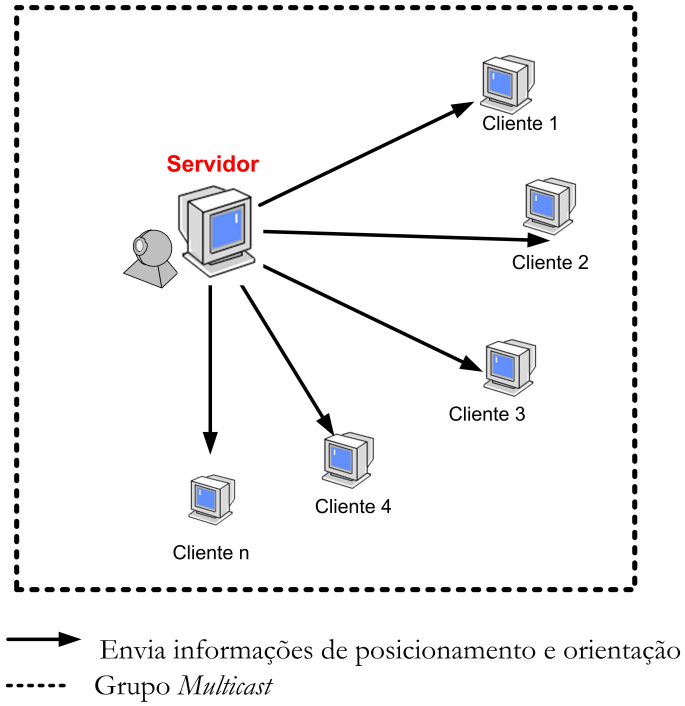


Figura 4.27: Aplicação de Realidade Aumentada Distribuída controle com o Servidor

É usada uma estrutura de dados para permitir o armazenamento das informações da manipulação dos marcadores e suas interações. A Figura 4.28 ilustra o formato do pacote enviado pela aplicação servidora. Dessa forma, é possível enviar os dados armazenados à aplicação remota, possibilitando a visualização e interação dos objetos virtuais por todos os participantes da aplicação.

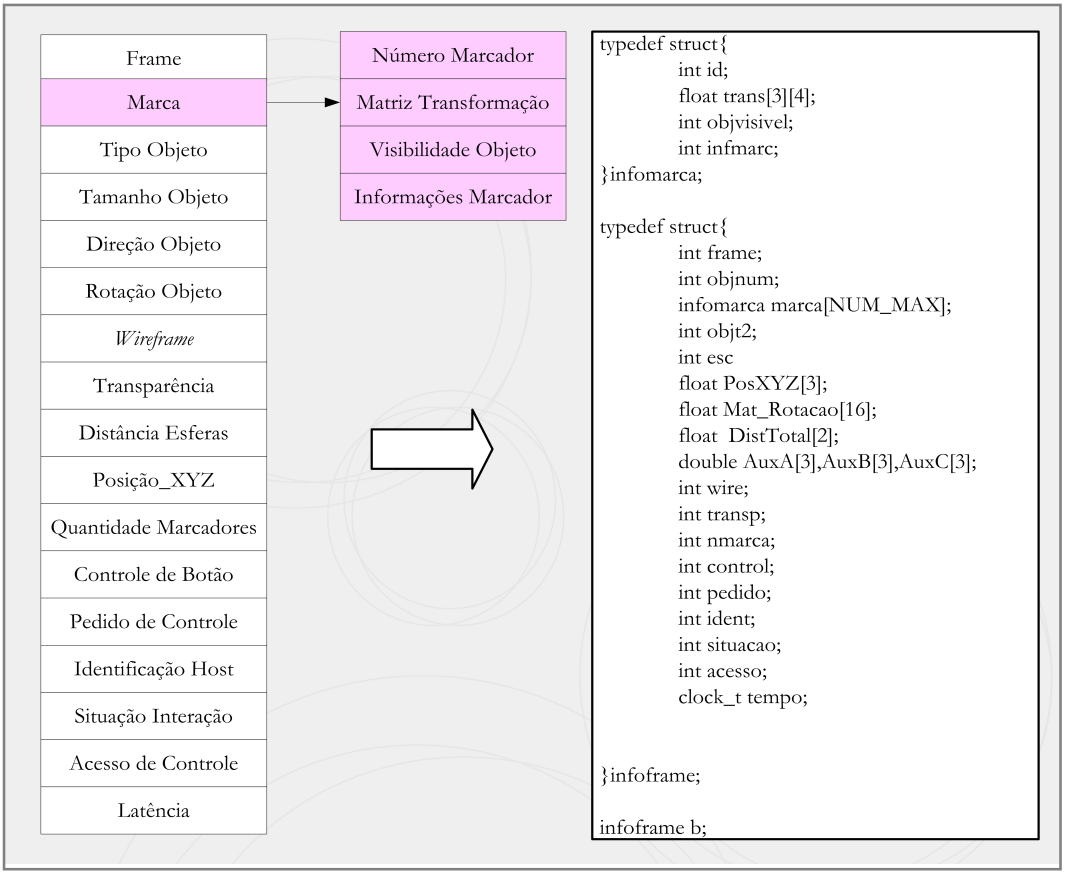


Figura 4.28: Formato da mensagem enviada pela aplicação servidora

a) Diagrama de funcionamento da aplicação servidora

O diagrama de funcionamento da aplicação servidora, representado pela Figura 4.29, mostra as funções responsáveis pela inserção da imagem real ao ambiente virtual, o controle de bloqueio e a renderização dos objetos. As principais ações da aplicação servidora são:

1. Configuração do tamanho da janela, associação dos padrões utilizados com os marcadores.
2. Conexão da aplicação com interface *socket*s UDP *multicast*.
3. Cálculo e exibição da taxa de quadros/segundos.
4. Criação da Interface gráfica 2D, com botões e demais controles.
5. Laço principal para captura da imagem real e matriz de transformação.
   - Envia a matriz transformação e interações com objetos virtuais para os clientes.
   - Recebe as interações realizadas pelos clientes sobre os objetos virtuais.

- Testa o controle de bloqueio para verificar qual participante terá o controle sobre o objeto virtual.
- Se o controle da interação estiver com a aplicação servidora, ela realiza a verificação de quais marcadores estão visíveis e chama função para renderização dos objetos virtuais, de acordo com a interação realizada pelo usuário através de sua interface 2D. Caso o controle da interação esteja com o Cliente, é realizada a verificação de quais marcadores estão sendo capturados pela aplicação servidora e posteriormente é chamada à função para renderização dos objetos, de acordo com a interação realizada pelo Cliente.

6. Fecha a conexão de *sockets* multicast.
7. Encerra a aplicação.

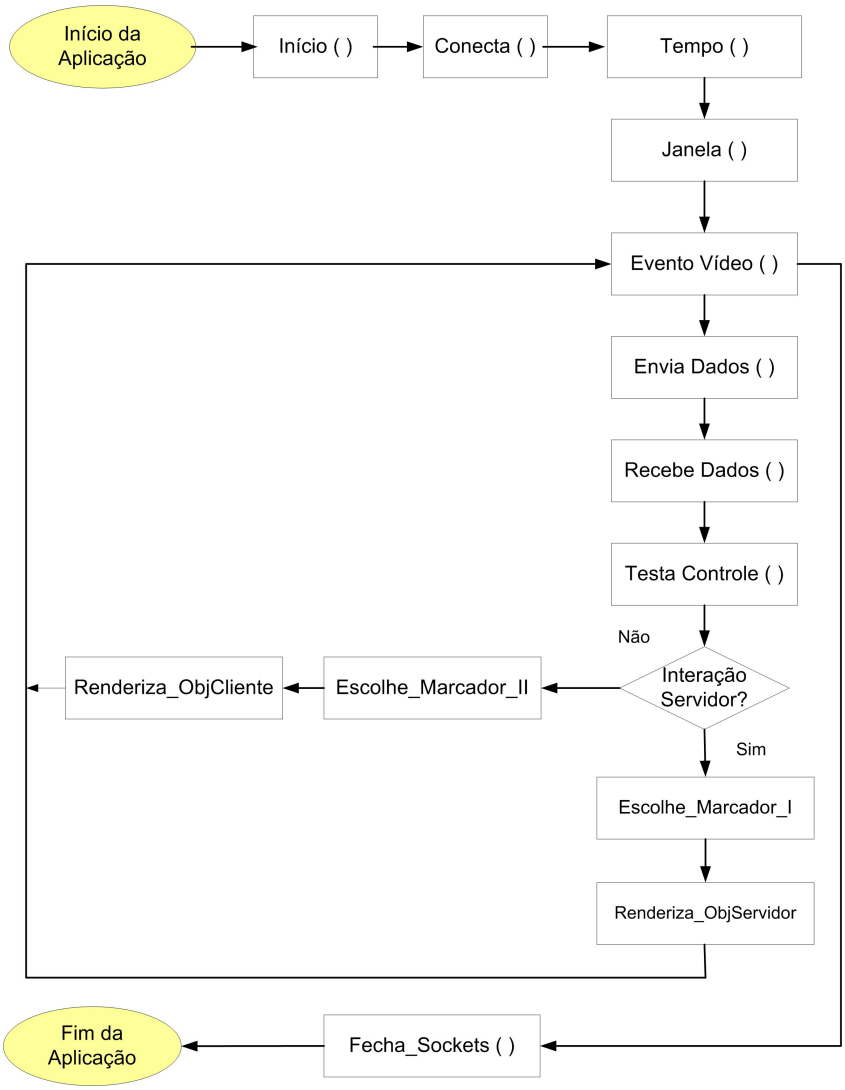


Figura 4.29: Diagrama da Aplicação Servidora

Aplicação Cliente

A aplicação cliente foi implementada para receber as informações do objeto virtual provenientes da aplicação servidora e usá-las para a exibição do seu objeto virtual para uma posterior interação (Figura 4.30).

Como a aplicação cliente não utiliza os recursos da biblioteca ARToolkit, esta foi implementada utilizando apenas os recursos da API OpenGL e GLUI, juntamente com os recursos de *socket*s UDP *multicast* para a comunicação remota.

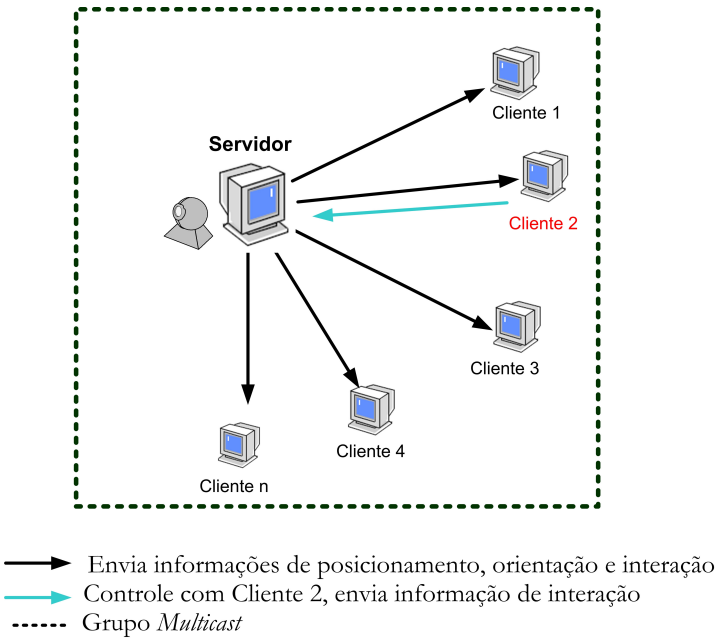


Figura 4.30: Aplicação de Realidade Aumentada Distribuída controle com o Cliente

a) Função para ajustar a taxa de exibição dos objetos

A aplicação cliente apresenta a função Evento_Tempo para ajustar a taxa de exibição (quadros por segundo) dos objetos de acordo com a captura executada pelo Servidor.

Essa função recebe as transformações de posição e orientação da câmera vinda da aplicação servidora e utiliza-as para a exibição da movimentação do seu objeto virtual para uma posterior interação.

Para isso, utilizou-se a função *glutTimerFunc* da API OpenGL que possibilita uma função *callback*. Esta função exige como parâmetros, o nome da função que deve ser chamada e o tempo que ela deve esperar antes de chamar a função novamente. Assim é possível permitir que as transformações de posição e orientação recebidas da aplicação servidora sejam atualizadas e sincronizadas em tempo real, gerando o movimento do objeto na aplicação cliente.

São funções dessa rotina:

- **Função para Transformação de Matrizes:** Essa função recebe da aplicação servidora as transformações de posição e orientação dos objetos virtuais, que são representados por uma matriz 3X4. Ela foi implementada para tratar essas informações vindas do Servidor, pois estas serão utilizadas para alterar posicionamento dos objetos na aplicação cliente. Como o OpenGL utiliza para a criação do sistema de visualização um vetor de 16 posições (matriz 4X4) é necessário transformar a matriz 3X4 (vinda da aplicação servidora) em uma matriz 4X4, obter a sua transposta e armazená-la em um vetor de 16 posições.
- **Função para opção de marcadores na aplicação cliente:** Responsável pela verificação de diferentes marcadores utilizados na aplicação servidora e por realizar a chamada da função para renderização dos objetos virtuais.

Para conseguir a realização do movimento dos objetos virtuais na aplicação cliente, é necessário utilizar a matriz transformação recebida da aplicação servidora. Dessa forma, a aplicação cliente possui a mesma estrutura de dados utilizada na aplicação servidora para receber as informações. Esta estrutura contém a matriz transformação, número do frame, visibilidade do objeto, etc.

Para exibição dos objetos, faz-se necessário o uso da função Transformação de Matrizes, juntamente com a função Evento Tempo para sincronizar o tempo de renderização, com isso é possível a API OpenGL exibir corretamente os objetos na aplicação cliente.

Para permitir a interação dos objetos na aplicação cliente, foram utilizadas técnicas de interação por meio de interface de controle 2D (biblioteca GLUI); com isso foi necessário criar uma nova estrutura para armazenar as interações do Cliente sobre os objetos virtuais. Nessa estrutura serão armazenadas as informações dos objetos, controle de bloqueio e as interações (Figura 4.31).

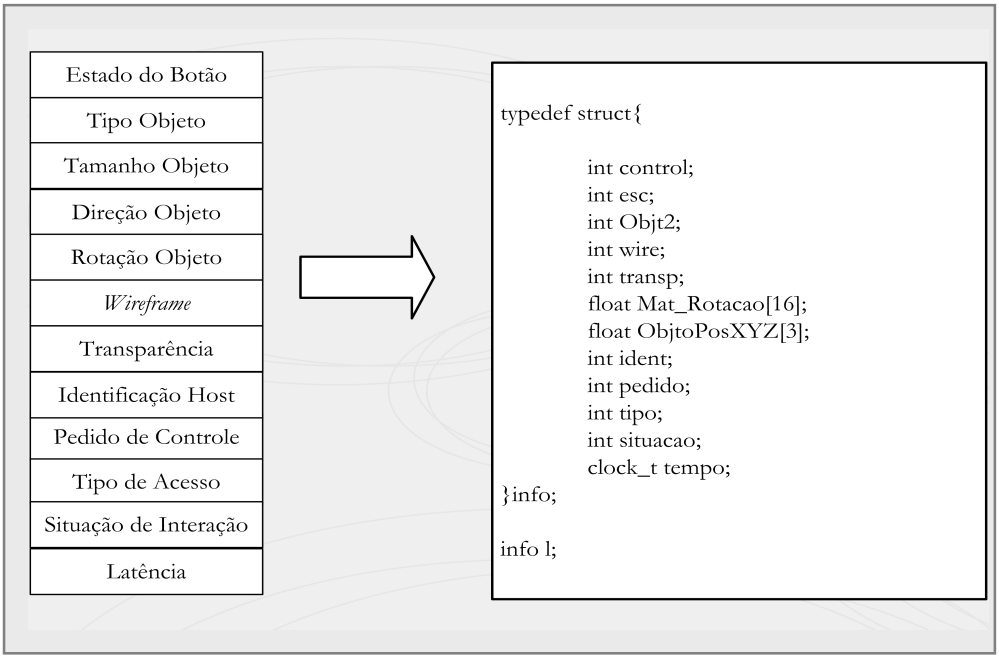


Figura 4.31: Formato da mensagem enviada pela aplicação cliente

b) Diagrama de funcionamento da aplicação cliente

A Figura 4.32 apresenta o diagrama de funcionamento da aplicação cliente, a qual é composta das seguintes etapas:

1. Conexão da aplicação através de serviços sockets UDP multicast
2. Criação da janela de interface 2D, com botões e demais controles.
3. Execução da função *Evento Tempo*, responsável por sincronizar a taxa de quadros por segundo.

- Recebe a matriz transformação e interação enviada pela aplicação servidora.
- Envia interação realizada sobre objeto virtual para aplicação servidora.
- Testa o controle de bloqueio.
- Se o controle de interação estiver com o Servidor, a aplicação cliente transforma a matriz, verifica os marcadores que estão visíveis no Servidor e posteriormente realiza a renderização dos objetos de acordo com as informações recebidas da aplicação servidora.
- Caso o controle seja do Cliente, são feitas as mesmas operações do passo anterior, com exceção de que, agora, a manipulação do objeto virtual é feita pelo Cliente.

4. Fecha a conexão do *socket*s.
5. Encerra aplicação.

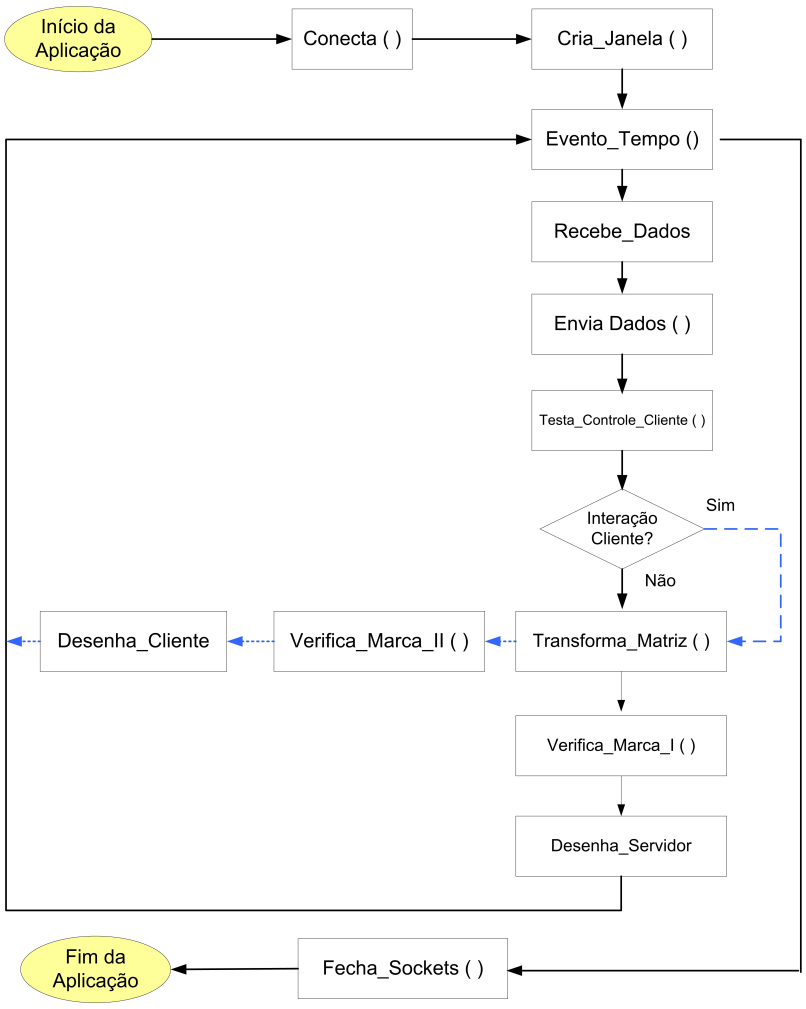


Figura 4.32: Diagrama da Aplicação Cliente

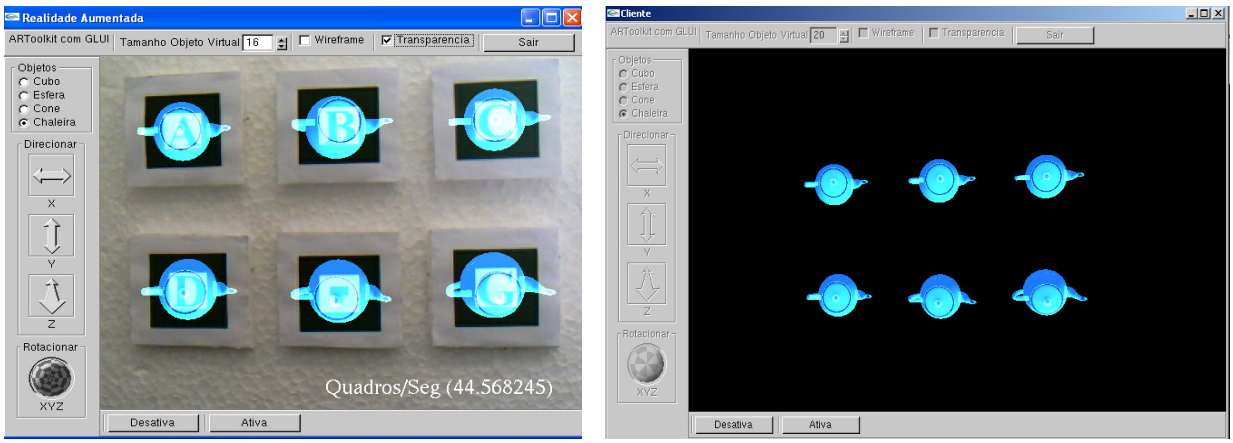
Controle de Bloqueio

O sistema desenvolvido permite que tanto a aplicação servidora quanto a aplicação cliente realizem interações sobre os objetos virtuais. Para que essas interações aconteçam de forma organizada, possibilitando que uma aplicação não interfira na execução da outra, utilizou a função "Controle de Bloqueio", presente no Módulo de Interação descrito no item 4.3.2.4.

Cada aplicação possui na sua interface gráfica botões de controle, um denominado "Ativa", responsável em pedir o controle de interação e um chamado "Desativa", para desativar o controle de interação.

Esses botões foram criados a partir da função "Criação da Interface Gráfica" presentes no módulo de interação item 4.3.2.2.

Conforme ilustra a Figura 4.33 (a) e (b), ambas as aplicações possuem botões para permitir o controle de interação sobre os objetos virtuais, estando nesse caso o controle com a aplicação servidora.



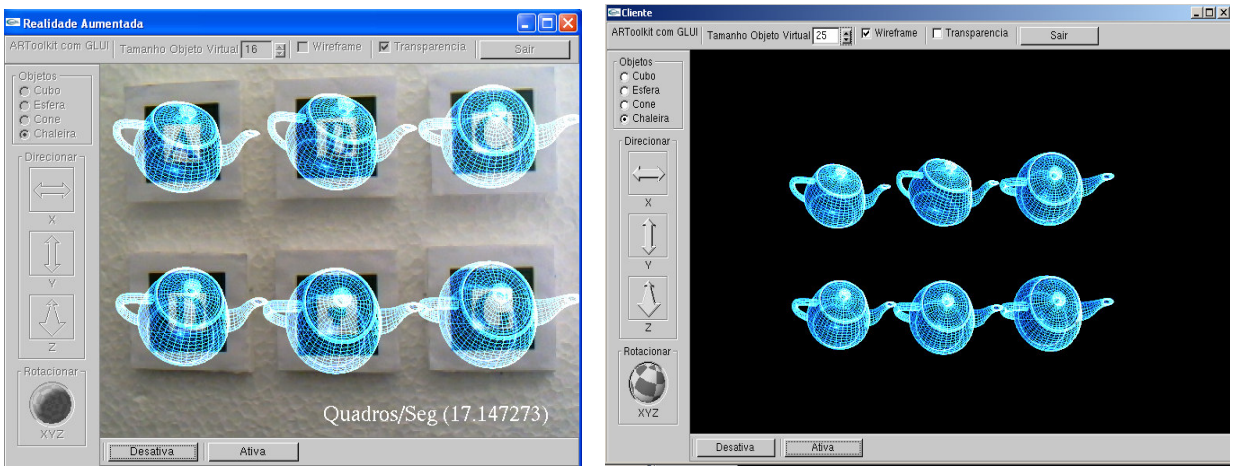
(a) Aplicação Servidora realizando interação (b) Aplicação Cliente recebendo as informações

Figura 4.33: Controle de Bloqueio com a aplicação servidora

No início da execução do sistema, a aplicação servidora tem a preferência de começar a interação dos objetos virtuais, pois a função de captura de imagens reais está sob seu controle. Dessa maneira a aplicação servidora mantém bloqueado o pedido de controle de interação dos objetos para os clientes.

Quando o Servidor não desejar mais possuir o controle de interação, ele libera o sistema para a aplicação cliente realizar pedido de acesso ao controle de interação.

No momento em que a aplicação servidora recebe pedido de um Cliente, ela verifica a identificação de quem pediu o controle e libera a interação para esse Cliente (Figura 4.44 (a) (b)). Caso aconteça de vários clientes pedirem permissão simultaneamente, o Servidor libera o controle por ordem de chegada de pedido.



(a) Servidor recebendo interação do Cliente (b) Cliente realizando interação sobre o objeto virtual

Figura 4.34: Controle de Bloqueio com a aplicação cliente

No instante em que o Cliente decide interromper a interação com os objetos, ele libera o controle de interação para domínio do Servidor. Caso aconteça de nenhum Cliente realizar pedido de controle de interação, a aplicação servidora pode retomar o controle novamente.

CAPÍTULO 5 - TESTE DA APLICAÇÃO E ANÁLISE DE RESULTADOS OBTIDOS

Neste Capítulo é apresentada uma análise da aplicação descrita no capítulo anterior. Esta análise tem como objetivo demonstrar a viabilidade da abordagem utilizada nessa dissertação e dar uma idéia do desempenho da aplicação.

5.1 O Ambiente Experimental

Para realização dos experimentos foi utilizado um dos laboratórios da Escola Técnica Estadual Amim Jundi, composto de 9 microcomputadores PC (Personal Computer) e um *notebook* , sendo 1 Servidor (*notebook*) e 9 Clientes (PC) em uma rede interligada através de um *switch Encore*, 16 portas, 10/100 Mbits. A Figura 5.1 ilustra a disposição dos equipamentos utilizados.

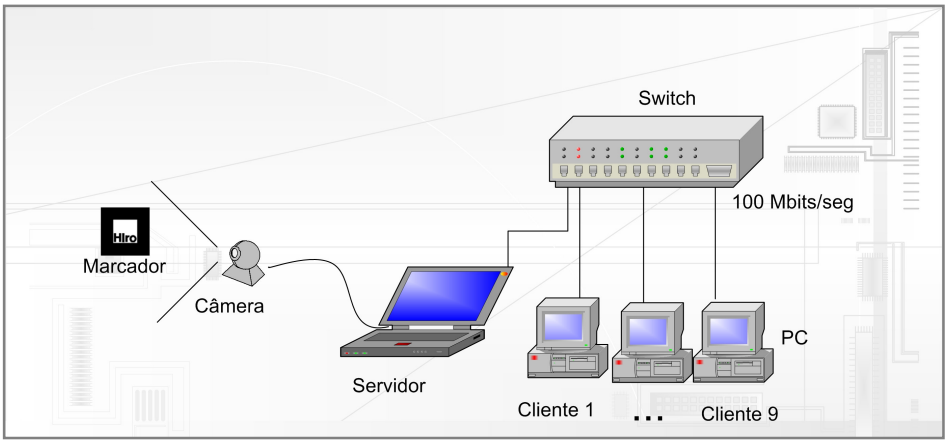


Figura 5.1: Disposição dos equipamentos

O ambiente experimental foi composto por uma mesa de tamanho 40 cm por 60 cm, uma câmera digital e um painel de isopor para fixação dos marcadores. A câmera foi posicionada a uma distância de 80 cm do painel. (Figura 5.2).

Figura 5.2: Ambiente Experimental para validação dos testes

As aplicações clientes possuíam cópias idênticas do protótipo.

A) Recursos de Hardware

Nos experimentos foram utilizados os seguintes recursos de hardware:

- Nove microcomputadores utilizados para serem os clientes da aplicação, com a seguinte configuração de *hardware:* processador AMD *Athlon* 1500 + 1.33 GHz, com placa de vídeo modelo NVIDIA GeForce 4MX 4000, 256 *megabytes* de memória RAM (*Random Access Memory*) e placa de rede *Ethernet* 10/100 Mbps.
- Para o Servidor utilizou-se um *notebook Toshiba* com a seguinte configuração de *hardware*: processador *Pentium* IV, placa vídeo modelo ATI *Mobility Radeon* 9000 IGP com 64 *megabytes* de memória, otimizada para *Direct*3D e OpenGL, 512 *megabytes* de memória RAM e placa de rede *Ethernet* 10/100 Mbits.
- Câmera digital, tipo *Webcam*, da marca *Creative*, modelo PC *Cam* 880, de resolução de 3.1 *megapixels*.

B) Recursos de Software

Os principais recursos de *software* no desenvolvimento das aplicações foram os seguintes:

- Biblioteca Gráfica OpenGL (*Silicon Grafics*) versão 2.0, responsável pelas funções de geração dos objetos 3D.
- Biblioteca Gráfica GLUI (RADEMARCHER, 2005) versão 2.0, usada no desenvolvimento das interfaces gráficas.
- Biblioteca de Realidade Aumentada ARToolkit (KATO e BILLINGHURST, 1999) versão 2.52, responsável pelo oferecimento das funções de RA.
- Biblioteca de Multimídia *DirectX* (*Microsoft)*, versão 9.0, que realiza o suporte aos dispositivos de vídeo do Sistema Operacional.
- Biblioteca *Winsock* API (*Microsoft*) versão 2.0, que permite a comunicação remota entre os participantes da aplicação.
- Visual C++ (*Microsoft*) versão 6.0.
- Sistema Operacional *Microsoft Windows XP Professional* (*Microsoft*) versão 2002.

5.2 Testes Realizados e Resultados Obtidos

Os testes realizados concentraram-se na observação da latência (atraso) da comunicação, como fator determinante na escalabilidade do sistema, bem como o desempenho da geração dos objetos 3D.

Para essa avaliação foram utilizados seis marcadores adicionados ao painel de teste (Figura 5.2(a)), medindo 4 cm, a uma distância de 4 cm entre eles para que não acontecesse oclusão dos marcadores. O painel foi colocado a uma distância de 80 cm à frente da câmera de vídeo.

Para realizar a interação por meio de pinças virtuais foram usados dois marcadores de 2 cm para serem os dedais e um marcador de 4 cm para ser o marcador fixo (Figura 5.3 (b) (c)).

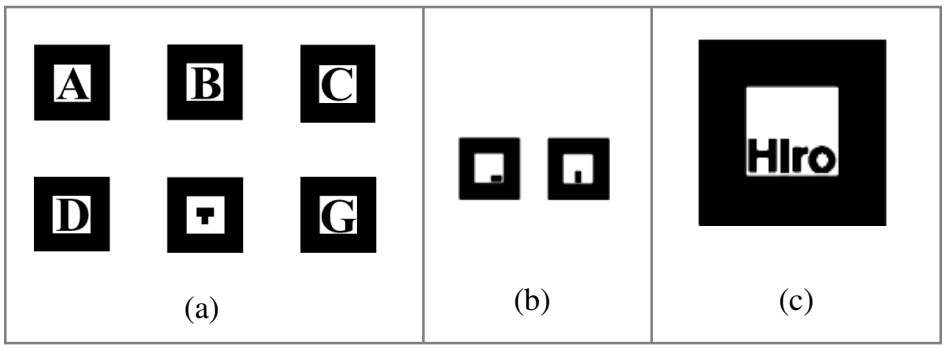


Figura 5.3: Marcadores utilizados para teste. (a) Marcadores para objetos da Interface 2D. (b) Marcadores usados como dedais (Interação com Pinças Virtuais). (c) Marcador para esfera fixa (Interação com Pinças Virtuais)

5.2.1 Latência da Comunicação

É interessante justificar a maneira como a latência foi obtida: o processo é iniciado quando a aplicação servidora envia uma mensagem com o tempo atual tomado a partir do *timer* da máquina servidora para aplicação cliente, via *multicast*. A aplicação cliente, ao receber o tempo, retorna o mesmo valor à aplicação servidora, através de uma nova mensagem. A aplicação servidora, ao receber a mensagem, faz uma nova tomada do tempo e subtrai desse valor o recebido na mensagem do Cliente.

A latência foi verificada nas diferentes sessões abaixo:

- **Sessão 1: Transmissão de 100, 500 e 1000 pacotes de mensagens pela rede:** Para a realização dessa análise, as mensagens de solicitação de tempo foram enviadas 100, 500 e 1.000 vezes para os computadores (ou nós) participantes do sistema. As estimativas foram realizadas para nove nós clientes, utilizando um a seis marcadores estáticos (Figura 5.4 - 5.5 - 5.6).

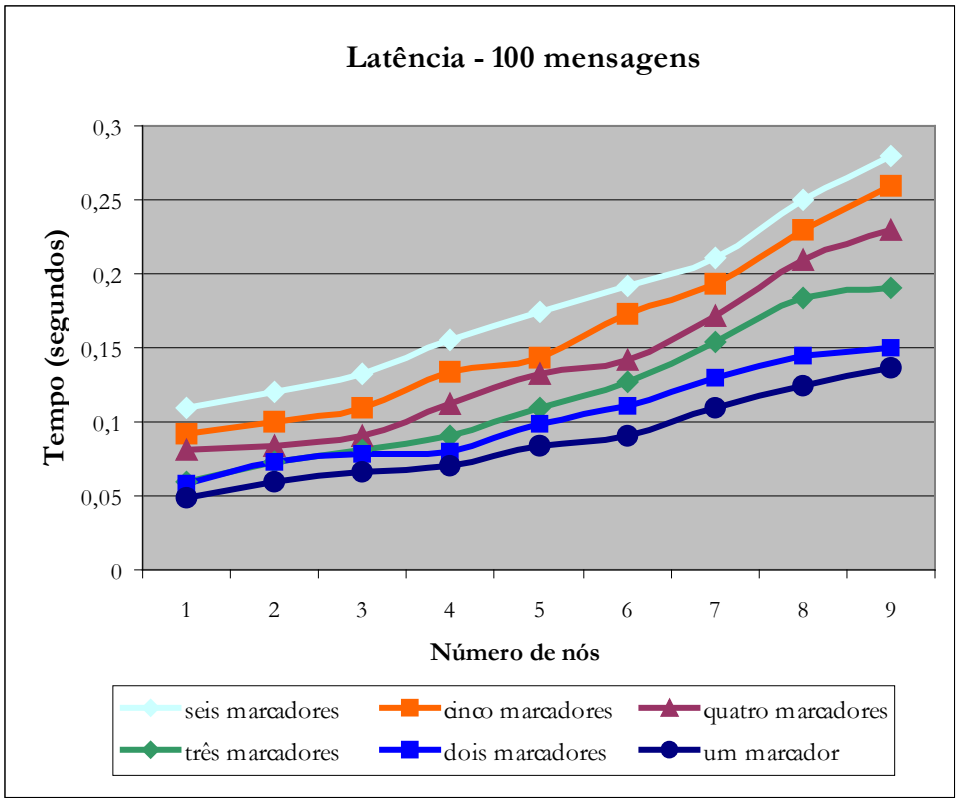


Figura 5.4: Latência para 100 mensagens

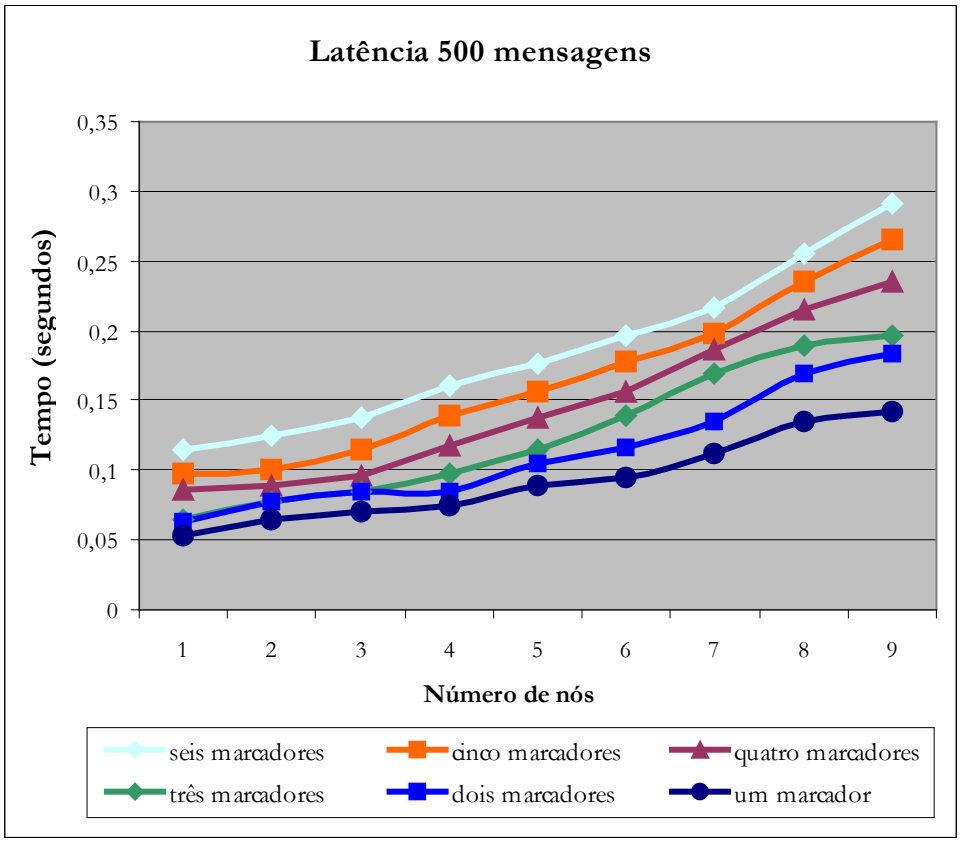


Figura 5.5: Latência para 500 mensagens

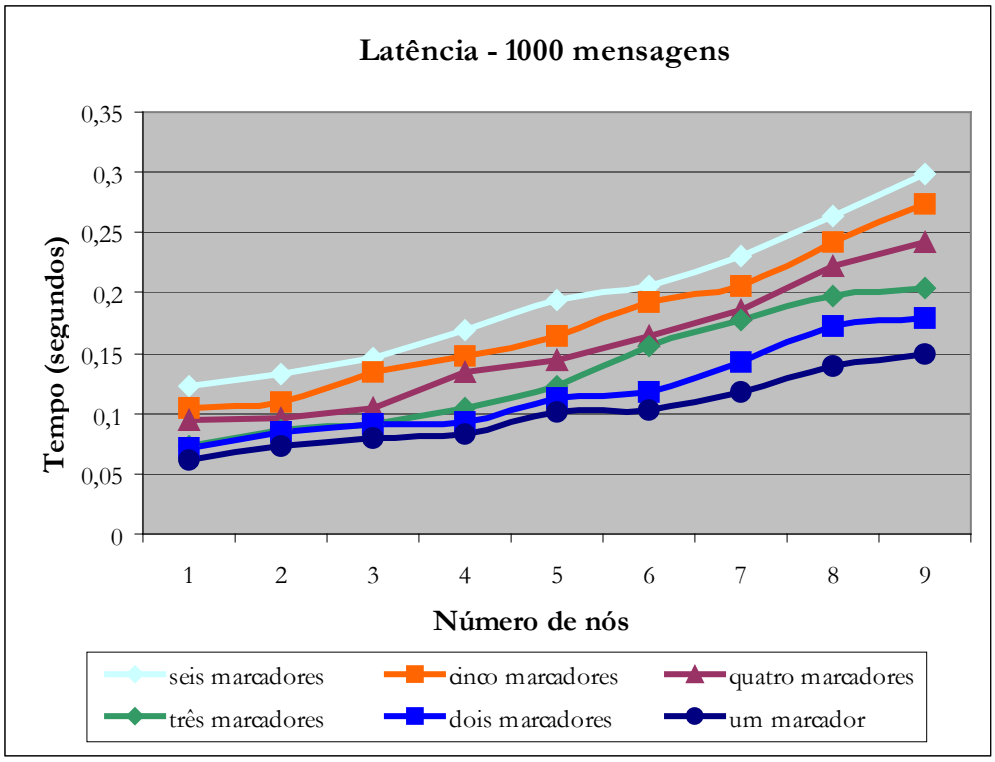


Figura 5.6: Latência para 1000 mensagens

Analisando os testes feitos para a verificação da latência da rede, observou-se que a quantidade de marcadores adicionados à cena faz com que ocorra um maior tempo de atraso na entrega das mensagens, isto ocorre porque aumenta o tamanho da estrutura da mensagem a ser transmitida para a aplicação cliente.

Devido ao uso da comunicação *multicast,* o número de nós que participou da aplicação não influenciou a latência, pois com o uso deste modelo a replicação dos pacotes é de responsabilidade da rede e não da aplicação. Como as mensagens são enviadas simultaneamente para um grupo de usuários, acontece uma economia de largura de banda. Caso a aplicação tivesse usado a comunicação *unicast*, ela deveria enviar n mensagens aos n participantes do grupo, ocasionado um maior número de mensagens e com isso um aumento na latência da rede.

A latência também não sofre influência quando são realizadas interações sobre os objetos virtuais, pois embora tenham ocorrido alterações nos objetos, como posição, rotação, escala e outros, o tamanho da estrutura da mensagem enviada continua o mesmo.

5.2.2 Análise da Geração de Imagens

Em relação à geração das imagens foi analisada a taxa de frames (quadros) por segundo (fps).

O conceito de animação é proveniente de uma sucessão rápida de quadros, como acontece em um filme. Na geração de cenas visuais, o sistema requer altas taxas de quadros, o ideal é de 20 quadros por segundos ou mais, sendo o mínimo aceitável na ordem de 8 a 10 quadros por segundos, para manter a ilusão de movimento (KIRNER; PINHO, 2004).

Para essa análise, foram realizadas sessões de testes, nas quais as mensagens foram enviadas 500 vezes aos participantes do sistema em cada sessão realizada e, posteriormente, foi calculada a média dos resultados obtidos.

- **Sessão 1:** Geração de Imagens – Taxa de frames obtida no Servidor, utilizando marcadores estáticos (Figura 5.7).

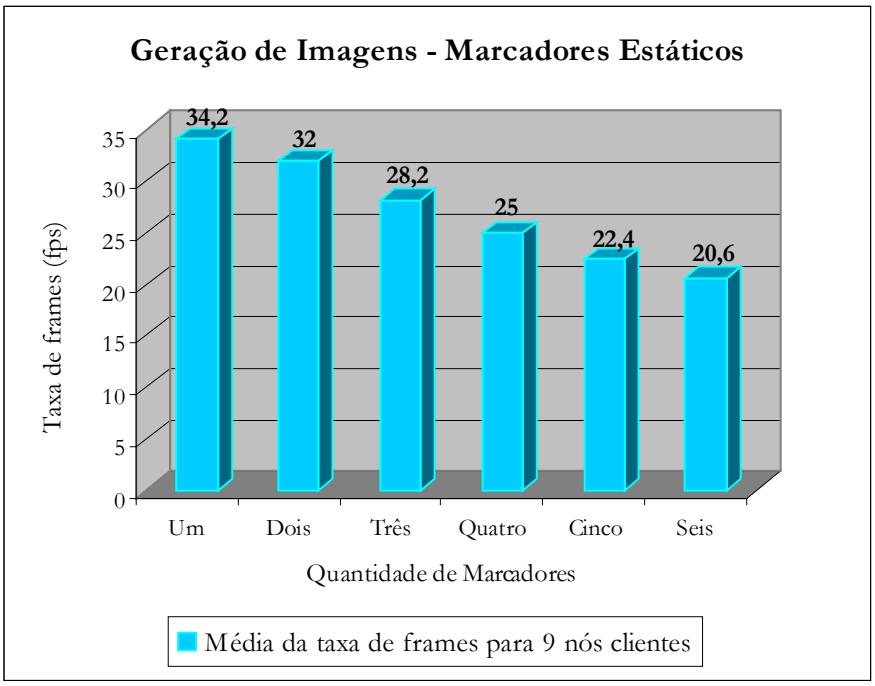


Figura 5.7: Geração de Imagens: Marcadores Estáticos

Observa-se na Figura 5.7 que com relação à adição de seis marcadores, a diminuição da taxa de frames por segundo foi de 39.8% e a cada marcador adicionado à cena ocorre uma diminuição em média de 8%.

A Tabela III abaixo apresenta os desvios-padrão para as médias de taxa de frames com relação ao número de marcadores adicionados à cena. Esses valores representam testes obtidos com marcadores estáticos.

Tabela III: Desvio Padrão – Marcadores Estáticos

Número de Marcadores	Média da taxa de frames (fps)	Desvio Padrão
Um	34.2	6.0
Dois	32.0	6.0
Três	28.2	5.5
Quatro	25.0	3.0
Cinco	22.4	2.6
Seis	20.6	2.4

- **Sessão 2:** Geração de Imagens – Taxa de frames calculada no Servidor, realizando movimento dos marcadores (Figura 5.8).

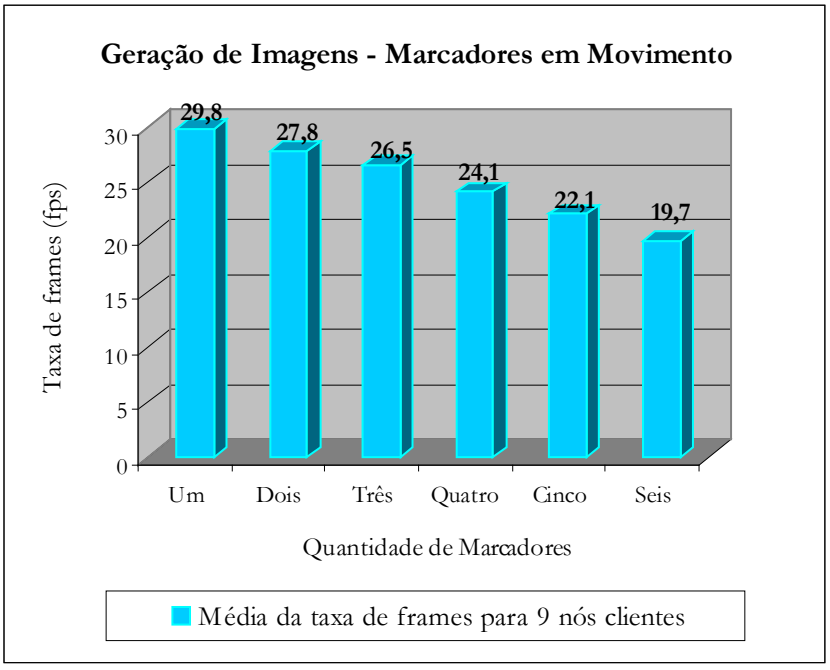


Figura 5.8: Geração de Imagens: Marcadores em Movimento

A cada marcador adicionado à cena a taxa de frames por segundo diminui em média 7.8% e quando comparado à adição de seis marcadores a diminuição na taxa de frames por segundo foi de 33.8% (Figura 5.8).

Os desvios-padrão para as médias de taxa de frames e quantidades de marcadores adicionados estão apresentados na Tabela IV. Esses valores foram obtidos com os marcadores em movimento.

Tabela IV: Desvio Padrão - Marcadores em Movimento

Número de Marcadores	Média da taxa de frames (fps)	Desvio padrão
Um	29.8	6.0
Dois	27.8	5.2
Três	26.5	5.7
Quatro	24.1	4.0
Cinco	22.1	2.7
Seis	19.7	2.8

- **Sessão 3:** Geração de Imagens – Taxa de frames (fps) calculada no Servidor por meio de interação com a interface gráfica 2D (Figura 5.9).

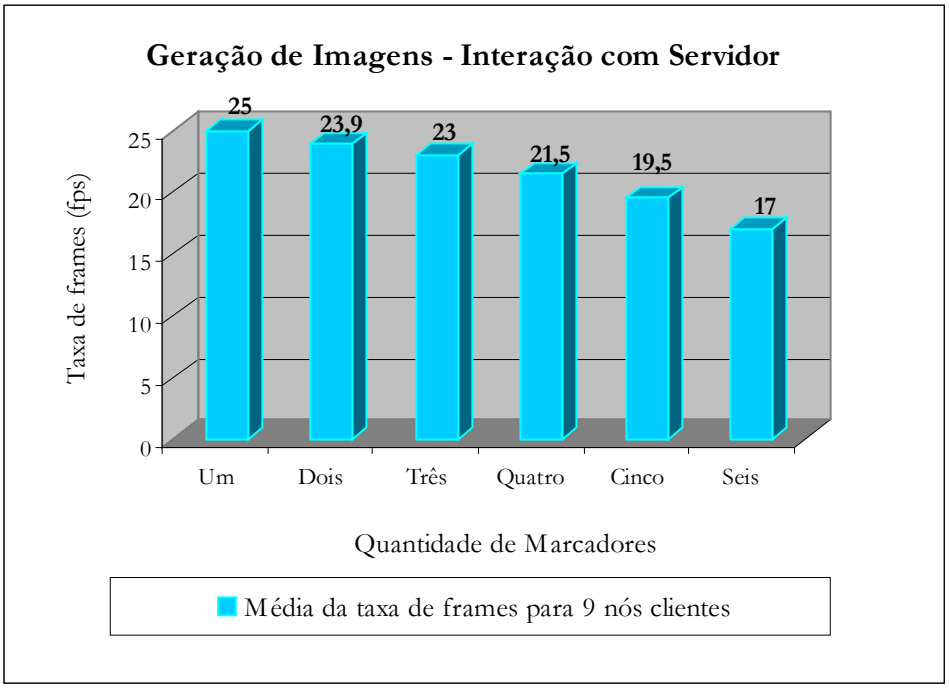


Figura 5.9: Geração de Imagens: Interação 2D Servidor

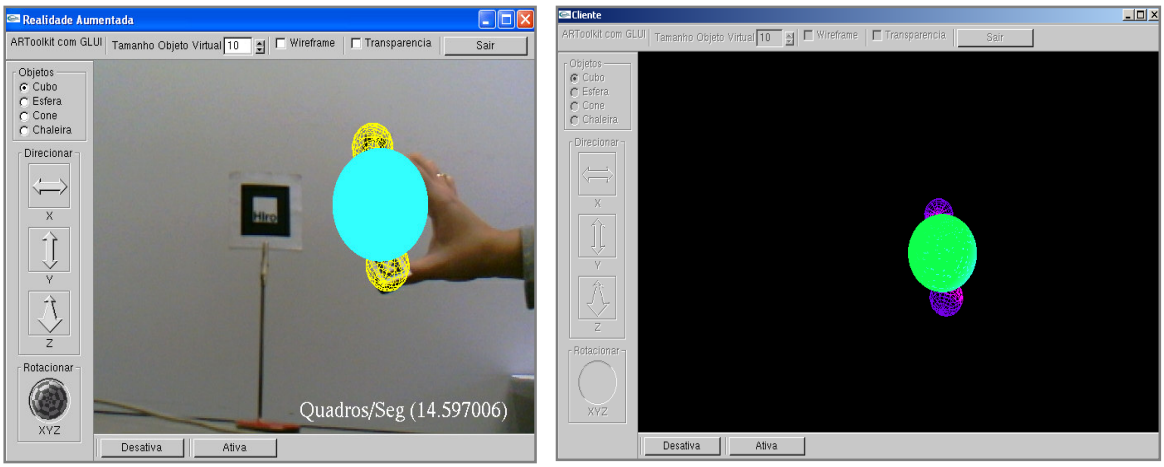
A Figura 5.9 ilustra que quando a aplicação utiliza seis marcadores na cena virtual acontece uma diminuição 32% e a cada marcador adicionado à cena essa taxa diminui em média 6.1%.

A Tabela V representa os desvios-padrão para as médias de taxa de frames para as quantidades de marcadores adicionadas. Esses valores foram obtidos pelo teste realizado com interação realizada pelo Servidor através de interface gráfica 2D.

Tabela V: Desvio Padrão - Interação com Interface 2D – Servidor

Número de Marcadores	Média da taxa de frames (fps)	Desvio Padrão
Um	25.0	7.0
Dois	23.9	7.2
Três	23.0	7.2
Quatro	21.5	6.6
Cinco	19.5	4.5
Seis	17.0	4.4

- **Sessão 4**: Geração de Imagens utilizando pinças virtuais: A Figura 5.10 ilustra a movimentação do objeto através da interação com pinças virtuais



(a) usuário movendo esfera no Servidor (b) usuário Cliente visualizando a esfera

Figura 5.10: Interação por Pinças Virtuais

Na Tabela VI encontra-se os resultados que foram obtidos pelo teste realizado com interação através de pinças virtuais, ilustrando o desvio padrão para as médias de taxa de frames para nove nós clientes, utilizando três marcadores adicionadas à cena.

Tabela VI: Desvio Padrão - Interação Pinças Virtuais

Número de Marcadores	Média da taxa de frames (fps)	Desvio Padrão
Três	19.5	5.0

- **Sessão 5:** Geração de Imagens – Taxa de frames calculada no Servidor por meio de interação com a interface gráfica 2D realizada pelo Cliente. O participante da aplicação cliente realizou a interação sobre o objeto virtual utilizando os controles da janela GLUI, onde mudou o tipo de objeto, alterou a escala, a direção, a rotação, aplicou transparência e mudou o estado para *wireframe.* A Figura 5.11 demonstra os resultados obtidos.

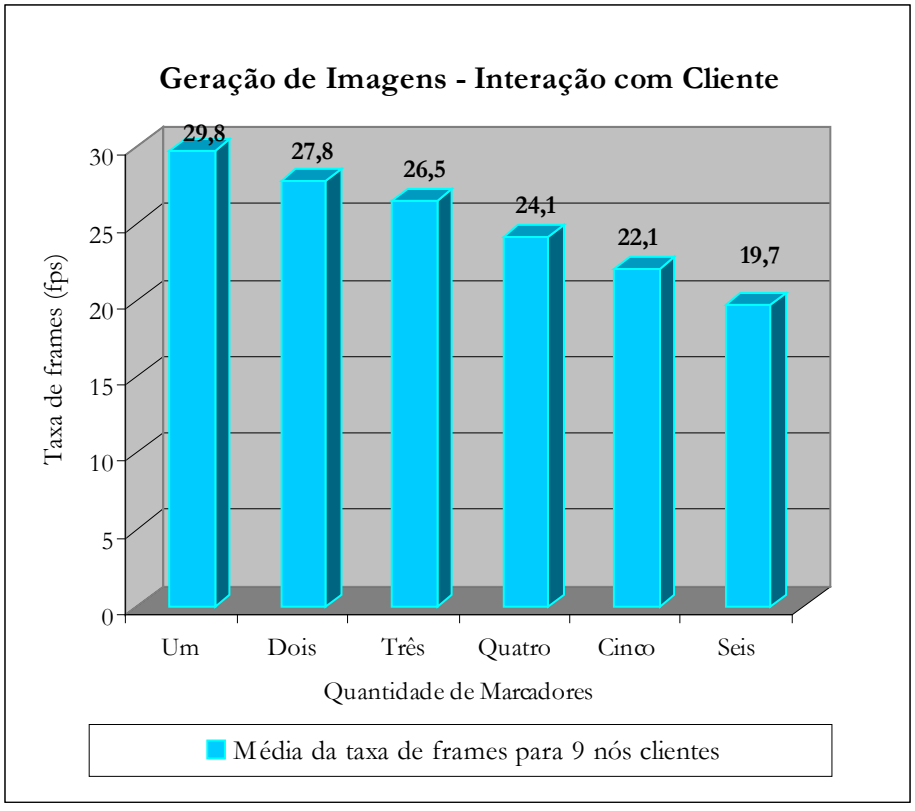


Figura 5.11: Geração de Imagens: Interação 2D - Cliente

Na Figura 5.11 nota-se que a cada marcador adicionado à cena acontece uma diminuição média na taxa de frames de 6.5%. Em relação à taxa inicial de 29.8 frames/segundo ocorre uma diminuição de 33.8% com seis marcadores se comparado apenas com um.

A Tabela VII apresenta os desvio padrão para as médias de taxa de frames e quantidades de marcadores adicionadas para nove nós clientes. Esses valores foram obtidos pela interação sendo realizada por um Cliente através de interface 2D.

Tabela VII: Desvio Padrão - Interação com Interface 2D – Cliente

Número de Marcadores	Média da taxa de frames (fps)	Desvio Padrão
Um	29.8	7.0
Dois	27.8	6.6
Três	26.5	6.8
Quatro	24.1	5.4
Cinco	22.1	4.0
Seis	19.7	3.5

- **Sessão 6:** Geração de Imagens – Taxa de frames calculada no Servidor, através de interação por meio de marcadores e interação com interface 2D, juntamente com as interações realizadas pelo Cliente através da interface 2D. Esse teste foi realizado por dez participantes interagindo com a aplicação. Um participante ficou com o controle do Servidor e interagiu com a aplicação por meio dos marcadores, e também pôde interagir utilizando os controle da Interface 2D. Os noves participantes restantes interagiram com a aplicação utilizando interface de controle 2D. A Figura 5.12 ilustra o resultado obtido em relação a taxa de frames por segundo.

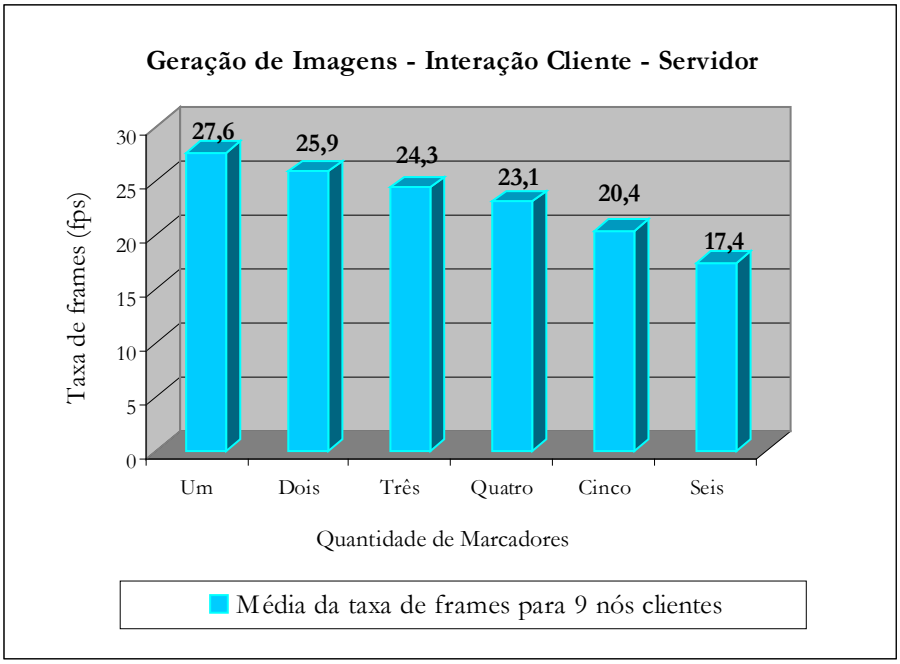


Figura 5.12: Geração de Imagens: Interação Cliente-Servidor

A cada marcador adicionado à cena observou-se que ocorre em média uma diminuição na taxa de frames de 7.2%. Comparando a taxa inicial de 27.6 frames/segundo, relata-se em uma diminuição de 36.9% na taxa de frames utilizando seis marcadores (Figura 5.12).

A Tabela VIII abaixo representa os desvios-padrão para as médias de taxa de frames e marcadores utilizados. Esses valores foram obtidos através de testes com a interação sendo realizada pelo Servidor juntamente com o Cliente.

Tabela VIII: Desvio Padrão - Interação Cliente - Servidor

Número de Marcadores	Média da taxa de frames (fps)	Desvio Padrão
Um	27.6	7.4
Dois	25.9	7.3
Três	24.3	7.3
Quatro	23.1	6.0
Cinco	20.4	5.1
Seis	17.4	5.0

Comparação das Interações usadas nos testes

a) Medições realizadas na máquina servidora

Observando os testes realizados para medir a taxa de frames por segundo na aplicação servidora, notou-se que, quando aumenta-se o número de marcadores na cena, a taxa de frames diminui; isso acontece devido ao fato de ocorrer um maior processamento para detecção de novos marcadores.

A taxa de frames também é influenciada pela situação dos marcadores (estáticos ou em movimento). Quando o usuário realiza ações sobre o marcador, observa-se que diminui a taxa de frames, devido às dificuldades conseguir o reconhecimento dos marcadores.

A forma mais comum de criar imagens gráficas tridimensionais por computador baseia-se no uso dos polígonos (DURLACH; MAVOR, 1995). A taxa de frames do ponto de vista gráfico depende da complexibilidade do mundo virtual, iluminação, sombreamento e texturas. Na geração de cenas visuais, o sistema requer altas taxas de frames por segundo (ideal é de 20 quadros por segundo) e tempo de resposta rápida para manter a ilusão de cenas realistas próximas ao mundo real (KIRNER, 1998). Portanto, observa-se que, quando os usuários realizaram interações através do controle de interface 2D, a escolha dos tipos de objetos influenciou a taxa de frames, por exemplo, um “cubo” apresenta menos polígonos que uma “chaleira”, com isso, esse objeto apresenta maior facilidade de renderização, resultando em uma maior taxa de frames por segundo.

O teste realizado com as pinças virtuais foi o que apresentou taxa de frames/segundos menor, isto ocorreu por causa da necessidade de um maior processamento para calcular a detecção de colisão.

A Figura 5.13 ilustra a comparação de todos os testes realizados.

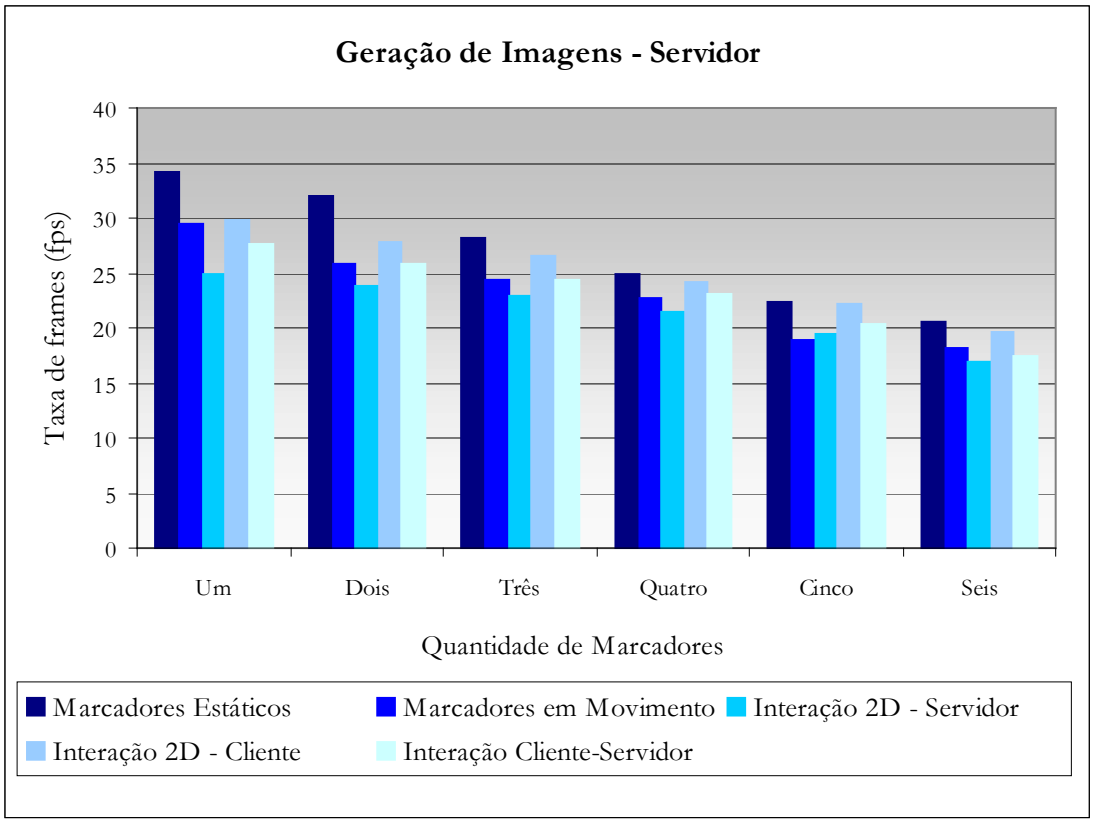


Figura 5.13: Geração de Imagens – Taxa de frames – Comparação I

a) Medição realizada na máquina cliente

Analisando os resultados obtidos com os testes medidos na máquina Cliente, observou-se que o aumento da quantidade de marcadores utilizados na aplicação servidora, quando esta captura o mundo real, prejudica a taxa de frames nessa aplicação, pois dependendo da quantidade de marcadores capturada pela câmera, a aplicação cliente deverá renderizar a mesma quantidade de objetos, com isso diminuindo a taxa de frames por segundo.

Na aplicação cliente o número de polígonos de um objeto virtual também pode influenciar a qualidade de uma aplicação, resultando em valor menor de taxa de frames.

A Figura 5.14 demonstra a comparação dos testes de interação realizados na aplicação cliente.

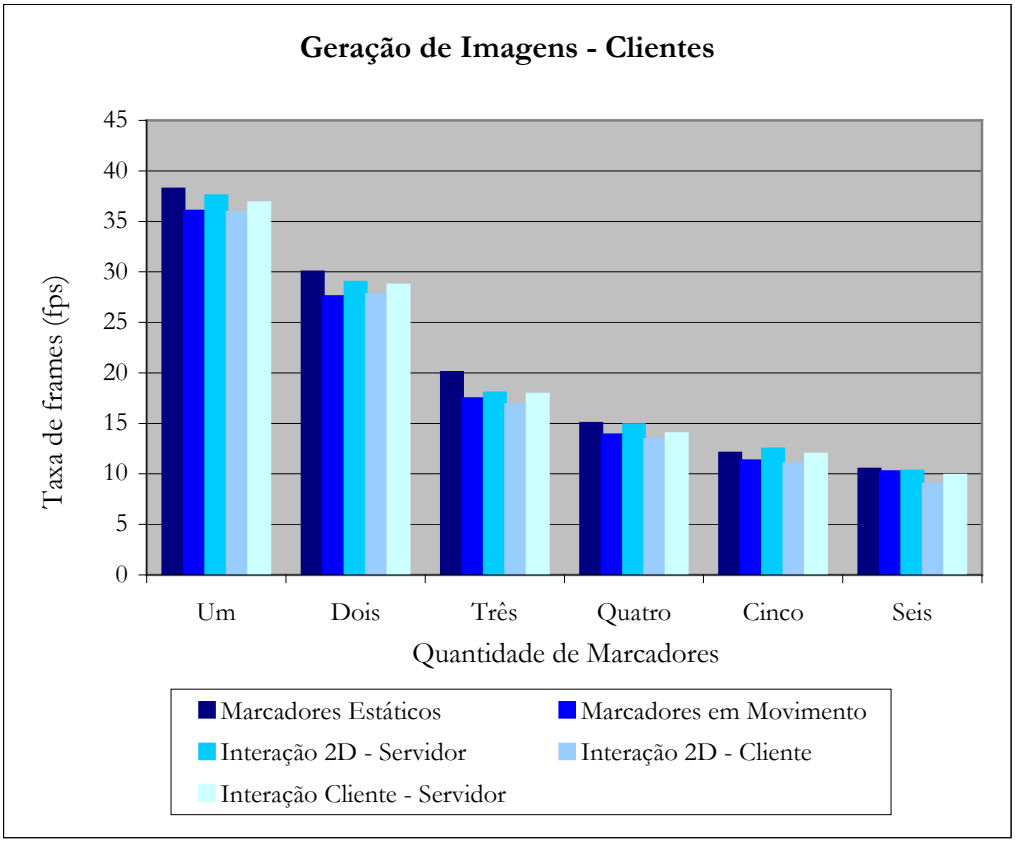


Figura 5.14: Geração de Imagens – Taxa de frames – Comparação II

Quanto ao uso do sistema por vários usuários notou-se que não houve dificuldades no manuseio do sistema. A interface gráfica utilizada ofereceu flexibilidade ao usuário, permitindo interagir com os objetos de uma maneira simples. O usuário que manipulou os marcadores utilizou as técnicas de RA e assim comprova que uma parte importante de qualquer interface do usuário de RA é a interação com objetos virtuais, uma vez que os humanos sabem como interagir com objetos reais, como controlá-los e manipulá-los, podendo oferecer ao usuário maiores informações sensitivas facilitando a reflexão sobre a situação.

Porém, esta fora do escopo deste trabalho analisar a facilidade de uso da aplicação por parte dos participantes da aplicação, sendo que o objetivo desta dissertação é prover mecanismos para interação remota, como o gerenciamento o controle de bloqueio da aplicação, a transmissão de informações através do modelo de comunicação multicast e interação sobre os objetos virtuais.

CONCLUSÃO

Atualmente o computador tornou-se uma interface entre as pessoas, permitindo que estas possam se comunicar, interagir e compartilhar informações. Esta interface pode ser muito melhorada através do emprego de técnicas de RV e RA.

Os ambientes criados a partir da utilização destas técnicas podem permitir que usuários dispersos geograficamente compartilhem recursos computacionais em tempo real, usando um suporte de redes de computadores para melhorar o desempenho coletivo por meio de troca de informações.

A RA distribuída possibilita um compartilhamento de informações com um maior aumento de compreensão e permite a interação de objetos virtuais entre os participantes do sistema, possibilitando operações de seleção, manipulação e navegação no ambiente 3D.

Esse capítulo finaliza o trabalho, apresentando as contribuições e os trabalhos futuros sobre interação remota para sistemas de RA.

a) Contribuição do trabalho

A principal contribuição desse trabalho consiste em apresentar um suporte à interação remota de objetos virtuais por meio de transmissão multicast, possibilitando dessa forma que vários usuários interajam sobre o objeto virtuais, transmitindo suas ações para todos os participantes da aplicação.

No levantamento efetuado nesse trabalho, verificou-se que, até o momento, são poucas as aplicações desenvolvidas com a RA remota para grupo multicast; utilizam-se muito aplicações com comunicação *unicast*.

No protótipo implementado desenvolveu-se a interação de objetos por meio de interação direta de usuário e controles virtuais. Essa aplicação foi realizada com o uso da biblioteca *ARToolkit*, OpenGL, GLUI e linguagem de programação C, sendo que também poderia ser utilizado para renderização dos objetos a linguagem VRML, além de poder ser executada nos microcomputadores atualmente acessíveis à maioria dos usuários.

O protótipo consiste em uma contribuição por ser totalmente funcional, podendo ser aplicada em qualquer área como ferramenta de ensino, treinamento e pesquisa. O processo de interação com os objetos virtuais e sua transmissão em tempo real aos demais participantes aumenta seu envolvimento e facilita a cooperação.

b) Trabalhos futuros

A seguir são explanadas as pesquisas que poderão dar continuidade a esse trabalho. Estas pesquisas referem-se tanto ao aperfeiçoamento no nível de interação em sistemas de RA quanto aos melhoramentos que poderão ser feitos a nível da aplicação de realidade virtual distribuída.

- **Transmissão da Imagem Real para todos os participantes**

  No protótipo a captura da imagem real ficou apenas para a aplicação servidora, e esta tem a possibilidade de visualizar a imagem real; os demais participantes visualizam apenas o objeto gerado virtualmente, assim como as interações realizadas sobres ele. Poderá ser implementada uma rotina para envio da imagem real para todos os participantes da aplicação, podendo esses ter a alternativa de visualizar ou não a imagem real transmitida pelo Servidor.

  Deve-se ressaltar que esta rotina poderá ter um impacto sobre a latência, pois aumentará quantidade de mensagens enviadas.

- **Adição de Câmeras nas aplicações clientes**

A RA consiste em uma combinação de mundo real com mundo virtual, em que a interação com o objeto virtual torna-se muito semelhante se comparado quando um usuário manipula um objeto no mundo real. Poderia implementar na aplicação cliente as funções de captura de imagens reais por meio de técnicas de RA, assim todos os participantes poderiam usufruir da técnica de interação.

- **Uso de técnicas *Dead-reckoning***

De acordo com Singhal e Zyda (1999), o *Dead-Reckoning* é uma técnica usada em AVDs, em que a idéia principal é transmitir pacotes apenas quando ocorrem determinadas atualizações. Enquanto o sistema não recebe uma nova atualização, cada *host* utiliza as informações baseado-se nos dados recebidos no último pacote de atualização. Quando um novo pacote chega, o *host* atualiza e aplica as novas informações. Dessa forma essa técnica pode ser incluída na aplicação desenvolvida para se melhorar o desempenho a partir da minimização da latência de comunicação, realizando o envio da orientação e rotação de objetos, assim como suas alterações de escala, tipo de objeto, apenas quando ocorrem alterações.

- **Funções para carregamento de imagens**

Na aplicação de RA Distribuída os objetos renderizados são formas básicas produzidas pelo OpenGL. Seria interessante usar na Aplicação de RA distribuída funções para carregamento de imagens que já se encontram disponíveis no ARISUPPORT (LOPES, 2005), permitindo assim carregar imagens no formato BMP (Bitmap) e JPEG (*Joint Photografic Experts Group*) como textura de superfícies e adição de texturas aos objetos virtuais em formatos diferentes. Como também

realizar a importação de modelos 3D desenvolvidos em ambientes gráficos profissionais, como modelos desenvolvidos no 3d Studio e funções de importação de vídeos e manipulação dos quadros em formato AVI (*Áudio Vídeo Interlave*).

- **Criação de uma aplicação específica**

A aplicação provê mecanismos para interação remota, mas não foca nenhuma aplicação específica, assim pode-se direcionar este trabalho para algum tipo de aplicação como, por exemplo, ensino a distância, medicina, treinamento entre outros.

- **Análise da usabilidade pelos participantes da aplicação**

A aplicação permite a interação dos participantes sobre os objetos virtuais. Seria interessante analisar como os participantes reagem ao uso da RA, controles GLUI, comunicação remota, conseguindo analisar a facilidade do uso da aplicação pelos participantes do sistema.

REFERÊNCIAS

AZUMA, R. T**. A Survey of Augmented Reality**. In: Teleoperators and Virtual Environments 6, p. 355-385, Agosto 1997.

AZUMA, R.T; BAILLOT, Y; BEHRINGER, R; FEINER, S; JULIER, S; Macintyre, B**. Recent advances in Augmented Reality.** In: Presence: IEEE computer Graphics and Applications, v.21, n.6, p.34-47, 2001.

BAUER, M.**;** HILLIGES A.; MACWILLIAMS A.; SANDOR C.; WAGNER G.; KLINKER G.; NEWMAN J.; REITMAYR G.; PINTARIC T.; FAHMY T.; SCHALSTIEG D. **Integrating Studierstube and DWARF.** In Presence**:** International Workshop on Software Technology for Augmented Reality Systems. Tokyo, Japão, Outubro, 2003 .

BILLINGHURST, M.; KATO, H. **Real World Teleconferencing**. In: Conference on Human Factors in Computing (CHI 99), ACM Press, Nova York, p. 194-195, 1999.

BILLINGHURST, M.; KATO, H.; POUPYREV, I.; MAY R. **Mixing Realities in Shared Space:** An Augmented Reality Interface for Collaborative Computing**.** In**:** ICME, p. 1641-1644**,** 2000.

BILLINGHURST, M.; KATO, H.; POUPYREV, I. **The MagicBook: A Transitional AR Interface.** In**:** Computers and Graphics. p. 745-753, 2001.

BILLINGHURST, M.; KATO, H.; POUPYREV, I. **The MagicBook —Moving Seamlessly between Reality and Virtuality.** In: IEEE Computer Graphics and Applications, May/June, p. 6-8, 2001.

BILLINGHURST, M.; KATO, H. **Collaborative Augmented Reality (2002).** Disponível em: <http://www.hitl.washington.edu/publications/r-98-36/ 2005>. Acessado em Julho de 2004.

BRUGGE, B.; MACWILLIAMS, A.; REICHER, T. **Study on Software Architectures for Augmented Reality Systems.** Publically as Technical Report TUM –I0410 – 2004.

CHARÃO, M.; KOPPER, R. **Desenvolvimento de Aplicações de Realidade Virtual**. Mini-curso SRV. In: VII Symposium on Virtual Reality, São Paulo, SP, 2004.

CHURCHILL, E.F.; SNOWDON, D. N.; MUNRO, A. J. **Collaborative Virtual Environments**. Springer, 312 p. 2001.

COSTA, F.; KON F. **Novas Tecnologias de Middleware: Rumo à Flexibilização e ao Dinamismo.** Simpósio Brasileiro de Redes de Computadores. p. 1- 61. SBC. Búzios, 2002.

DANTAS, M. **Tecnologias de Redes de Computação e Computadores**. Excel Books, 328p. 2002.

DIAS, B. C. **Análise da tecnologia WAP via estudo de caso em jogos distribuídos e interativos**. Trabalho de Conclusão de Curso (Ciências da Computação) Universidade Federal de Lavras, 2003.

DURLACH, N. I; MAVOR, A.S. - **Virtual Reality: Scientific and Technological Challenges**. National Academy Press, Washington, DC, 1995

FUKS, H.; RAPOSO, A.; GEROSA, M.A. **Engenharia de Groupware: Desenvolvimento de Aplicações Colaborativas. In XXI Jornada de Atualização em Informática.** Anais do XXII Congresso da Sociedade Brasileira de Computação, V2, Cap.3, ISNB 85-88442-8, p. 89-128, 2002.

GOUVEIA, L. M. B. **Ambientes Virtuais Colaborativos: A procura de formas alternativas de interação,** 2000. In: Revista politécnica. Edições da cooperativa de ensinopolitécnico, Porto: Dezembro - 2000.

HESINA, G. **The Distributed Collaborative Augmented Reality,** 2001. Disponível em: <http://www.cg.tuwien.ac.at/research/theses/hesina/hesina_thesis.pdf>. Acessado em Janeiro de 2005.

HOFTE, H. T. **Working Apart Together, Foundations for Component Groupware**. In: Telematica Instituut Fundamental Research Series, vol. 001. Netherlands, 1998.

KATO, H.; BILLINGHURST, M.; BLANDING, R.; MAY, R. **Manual ARToolkit – PC version 2.11,** 1999. Disponível em: www.hitl.washington.edu . Acessado em Julho de 2004.

KIRNER, C.; PROVIDELO C. **Realidade Aumentada: Conceitos e Ambientes de Hardware**. Mini-curso SVR 2004. In: VII Symposium on Virtual Reality, São Paulo, SP, 2004.

KIRNER, C.; TORI R. **Introdução à Realidade Virtual, Realidade Misturada e Hiper-realidade**. Realidade Virtual Conceitos e Tendências - Pré Simpósio. In: VII Symposium on Virtual Reality, São Paulo, SP, 2004.

KIRNER, C. **Sistemas de Realidade Virtual.** Disponível em: http://www.dc.ufscar.br/~grv/. Acessado em Janeiro de 2006.

KLINKER, G. **Augmented Reality Research Group**. Disponível em <http://wwwbruegge.in.tum.de/DWARF/WebHome>. Acessado em Março de 2005.

KAUFMANN, H**. Collaborative Augmented Reality in Education.** In: Imagina 2003 conference, Feb. 3rd, 2003.

KAUFMANN, H**. Geometry Education with Augmented Reality,** 2004. Dissertação de PhD. Vienna University of Technology, Março 2004.

KAUFMANN, H.; SCHMALSTIEG D**. Mathematics And Geometry Education With Collaborative Augmented Reality.** In: Computers & Graphics, Vol. 27, No. 3, p. 339-345, 2003.

LOPES, L. F. **O Estudo e a Implementação de Interface para Utilização em Sistemas de Realidade Aumentada**. 2005. 105 f. Dissertação (Mestrado em Ciência da Computação). Universidade Eurípides Soares da Rocha – UNIVEM, Marília, 2005.

MACEDONIA, M. R.; ZYDA, M. J. **A taxonomy for networked virtual Environments**. In: IEEE Multimedia, vol. 4 nº.1, p. 48-56, Jan.- Mar. 1996.

MAUFER, T. **Deploying IP Multicast in the Enterprise**. Prentice Hall, 1998. 275p.

MILGRAM, P.; KISHINO F. **A taxonomy of mixed reality visual displays**. In: IEICE Transactions on Information Systems, Vol. E77-D, No.12 December 1994.

MINE, M. R. **Virtual Environment Interaction Techniques,** 1995. In: UNC Chapel Hill Computer Science Technical Report. TR95-018, 1995.

MOECKEL, A. **Modelagem de Processos de Desenvolvimento em ambiente de engenharia simultânea: implementações com as tecnologias workflow e bscw**, 2000. 109f. Dissertação (Mestrado em Tecnologia). Centro Federal de Educação Tecnológica do Paraná, 2000.

NEDEL, L.; OST, C. L. **Usando Modelagem Formal para Especificar Interação em Ambientes Virtuais: Por que?** 2004. In: VI Symposium on Virtual Reality, Ribeirão Preto - SP, 2004.

PIEKARSKI, W.; THOMAS, B. **Tinmith-evo5 A Software Architecture for Supporting Research Into Outdoor Augmented Reality Environments**, 2001. In: Technical Report, University of South Austrália, Adelaide, AS, Report No. CIS-02-001.

PINHO, M. S**. Interação em Ambientes Tridimensionais.** Tutorial WRV 2000. Minicurso. In: WORKSHOP DE REALIDADE VIRTUAL, 2000, Gramado, RS. Disponível em http://www.inf.pucrs.br/~pinho/3DInteraction/. Acessado em Fevereiro de 2005.

PINHO, M. S**. Manipulação Simultânea de Objetos em Ambientes Virtuais Imersivos.** 2002. Tese de Doutorado (Ciências da Computação). Universidade Federal do Rio Grande do Sul. Porto Alegre, 2002.

RADEMACHER, P. GLUI User Interface Library. Versão 2.1. Disponível em: http://www.cs.unc.edu/~rademach/glui/. Acesso em outubro 2005.

REBELO, I. B.; PINHO, M. **Interação em Ambientes Virtuais Imersivos**. Realidade Virtual - Conceitos e Tendências - Pré Simpósio. In: VII Symposium on Virtual Reality, São Paulo, SP, 2004.

REITMAYR, G**. On Software Design for Augmented Reality,** 2004. 183 p. Tese de Doutorado (Ciência da Computação). Vienna University of Technology, 2005.

REITMAYR, G.; SCHMALSTIEG, D. **Collaborative Augmented Reality for Outdoor Navigation and Information Browsing.** In: Proc. Symposium Location Based Services and TeleCartography , Geowissenschaftliche Mitteilungen Nr. 66, p. 53-62, 2005

REITMAYR, G.**;** SCHMALSTIEG, D. **Mobile Collaborative Augmented Reality,** 2001.ISAR 2001, p. 114, IEEE and ACM International Symposium on Augmented Reality (ISAR'01), 2001.

SCHMALSTIEG, D.; GERD, Hesina. **Distributed Applications for Collaborative Augmented Reality**. In: Proc. IEEE VR 2002, p. 59 – 66, Orlando, Florida, USA, Março 2002.

SCHMALSTIEG, D.; FUHRMANN A.; SZALAVÁRI, Z.; GERVAUTZ, M. **Studierstube**

An Environment for Collaboration in Augmented Reality. In: Proc.of Collaborative Virtual Environments Workshop'96, Nottingham, UK, Setembro 1996.

SEIXAS, C. **Comunicação através da API Sockets sobre TCP/IP,** 2005. Disponível em <http://www.cpdee.ufmg.br/~seixas/PaginaSDA/Download/DownloadFiles/TcpIp.PDF> . Acessado em Janeiro de 2005.

SINGHAL, S.; ZYDA, M. **Networked Virtual Environments: Design and Implementation**. New York: Addison Wesley, 1999. 331p.

SZALAVÁRI, Z.; GERVAUTZ, M. **The personal Interaction Panel – A Two-Handed Interface for Augmented Reality**. In: Proceedings of EUROGRAPHICS´97. Volume 16, 1997, number 3.

TANENBAUM, A. S. **Redes de Computadores**. Elseveir, 2003. 945p.

TORRES, G. **Redes de Computadores**. Excel Books, 2001. 663p.

WICHERT, R. **Collaborative Gaming in a mobile augmented reality environment**. 2002. In: EUROGRAPHICS - Ibero-American Symposium in Computer Graphics – SIACG 2002. Proceedings, Guimares, p. 31-37. Portugal, 2002.