Faculdades Adamantinenses Integradas (FAI)

www.fai.com.br

AFFONSO, Elaine Parra; OLIVEIRA. Eliane Vendramini. Um suporte a interação remota para sistemas de realidade aumentada. Omnia Exatas, v.3, n.1, p.54-71, 2010.

UM SUPORTE A INTERAÇÃO REMOTA PARA SISTEMAS DE REALIDADE AUMENTADA

SUPPORT ON THE REMOTE INTERACTION FOR AUGMENTED REALITY SYSTEM

Elaine Parra Affonso
Mestre em Ciências da Computação – UNIVEM/Marília
Professora da FATEC – Presidente Prudente

Eliane Vendramini de Oliveira
Mestre em Engenharia Elétrica – UNESP/ Ilha Solteira
Professora da FAI e Professora da FATEC – Presidente Prudente

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: Realidade Aumentada, Interação, Comunicação, ARToolkit

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.

Key-words: Augmented Reality, Interaction, Communication, ARToolkit.

INTRODUÇÃO

Atualmente o computador tornou-se uma interface entre as pessoas, permitindo que estas possam se comunicar, interagir e compartilhar informações. Esta interface pode ser melhorada através do emprego de técnicas de Realidade Virtual (RV) e Realidade Aumentada (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, sendo denominados de Ambientes Virtuais Distribuídos (AVDs) (ZINGHAL, 1999).

Considerando, portanto, este contexto, o presente artigo descreve um suporte à comunicação entre os processos participantes de um Ambiente Virtual Distribuído 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 do sistema a partir de uma comunicação *multicast*.

O suporte utiliza a biblioteca ARToolkit (KATO, 2003), a qual provê funções para que programadores desenvolvam aplicações de RA. O ARToolkit tem se tornado viável no desenvolvimento de aplicações de RA, pois apresenta a vantagem do seu código estar aberto, gratuito e disponível para diversas plataformas de sistemas operacionais e possibilitar a utilização de diversos dispositivos de RA.

A comunicação entre os participantes ocorreu através dos serviços de *sockets multicast*. O modelo *multicast* é indicado quando existe a necessidade de transmitir dados para vários *hosts* simultaneamente. O uso da 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 aos enviar as informações. A comunicação *multicast* é muito usada em aplicações como videoconferência, simulação de interação distribuída, jogos em rede, ensino e treinamento à distância (MAUFER, 1998).

MATERIAIS E MÉTODOS

Os sistemas de RA vêm possibilitando experiências como a sensação de presença, envolvendo os usuários a interagir com objetos virtuais e permitindo também a colaboração nesses ambientes. Assim, esses ambientes podem oferecer aos participantes maiores informações sensitivas, onde a principal característica das aplicações de RA é criar um ambiente onde as informações do mundo virtual são utilizadas para incrementar o cenário real.

Utilizando um livro real para transpor usuários entre a realidade e a virtualidade, o projeto *MagicBook* (BILLINGHURST et al., 2003), utiliza a RA para visualização de objetos virtuais combinados com o mundo real e com tecnologia para suportar a colaboração, permitindo que vários usuários compartilhem o mesmo ambiente virtual. Nesse projeto os participantes podem virar as páginas do livro, olhar as figuras e ler os textos sem qualquer tecnologia adicional, mas quando utilizam dispositivos de RA, podem ver os objetos virtuais de vários pontos de vistas.

Com o objetivo de ensinar matemática e geometria, o projeto *Construct3D* (KAUFMANN; SCHMALSTIEG,2003) utiliza o sistema de RA colaborativa móvel *Studierstube (SCHMALSTIEG* et al*.,1996)* para possibilitar a colaboração e a interação entre professores e estudantes. Criou-se uma solução híbrida de hardware para completar diversas interações entre professores e estudantes no cenário virtual, como a sala de aula aumentada. Neste sistema foi utilizado 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. Os estudantes movem e giram um marcador e o objeto é exibido na tela de projeção. O *Construct3D* possui também a solução de aula híbrida distribuída; neste tipo de configuração os estudantes são equipados com computadores pessoais que trabalham com RV, assistindo o processo de construção da cena virtual na tela do seu micro. Esse sistema foi construído usando uma câmera para rastrear as posições e um marcador. 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. Esse projeto apresenta o diferencial de permitir a coordenação de atividades para realização do trabalho, onde possui um módulo que permite a um participante, no caso o professor, ficar responsável por garantir que as tarefas dos demais participantes sejam realizadas na ordem e no tempo correto.

Existem também, pesquisas com RA para colaboração local, como o *Shared Space* (BILLINGHURST,2000), que por meio de um ambiente físico com vários marcadores, e com a utilização de técnicas de visão computacional, permite a visualização das imagens virtuais alinhadas com os objetos físicos. Nesse projeto os usuários podem ver os demais suportando uma comunicação face a face, permitindo que vários usuários no mesmo local trabalhem simultaneamente em ambos os mundos, real e virtual. Desde que todos os usuários compartilhem o mesmo banco de dados de objetos virtuais, eles vêem os mesmos objetos anexados aos

marcadores de todos os seus pontos de vista. O usuário pode mover e mostrar os cartões para outros participantes, bem como passar ou pedir os objetos da mesma maneira que acontece com objetos reais.

Por meio de técnicas de navegação e sistema de anotação, o projeto *Outdoor Collaborative Augmented Reality* (OCAR) (REITMAYR, 2001), utiliza a RA para gerar anotações em determinados lugares físicos, possibilitando compartilhar informações com outros usuários. Esse sistema inclui a computação móvel, para permitir que os usuários alcancem e manipulem as informações independentes do momento e de sua posição. Desta forma, o usuário pode navegar em uma cidade, ao invés de utilizar mapas para conhecer o lugar, possibilitando utilizar o sistema para conhecer seu destino, combinando momentaneamente cenas do mundo real e virtual. 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 pode selecionar um conjunto de ícones para visualizar informações através de imagens e textos, por exemplo, pode ver datas interessantes sobre edifícios e lugares turísticos que eles está visitando. 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.

Os projetos analisados permitem que tanto os objetos, quanto o ambiente, sejam compartilhados e as mudanças ocorridas são visíveis a todos os participantes, criando, assim, um ambiente flexível e a possibilidade de ocorrem múltiplas visões das atividades que estão sendo realizadas.

Em todos os ambientes, os participantes envolvidos utilizam meios como gestos, som e textos para poderem se comunicar e interagirem entre si.

Todos os projetos analisados utilizam a biblioteca ARToolkit para calcular o ponto de vista real da câmera e sua orientação em relação aos marcadores. A ferramenta se destaca para construção de aplicações de RA por possuir bibliotecas de rastreamento e seu código fonte ser flexível para ser utilizado em diversas plataformas de sistemas operacionais ou ser adaptado para resolver as especificidades da aplicação do desenvolvedor. Apresenta como requisitos básicos de hardware, uma câmera de vídeo, um dispositivo de aquisição de vídeo com seus respectivos *drivers*, e fornece suporte para RA com visão direta por vídeo ou visão direta óptica.

O desenvolvimento de aplicações de RA com ARToolkit requerem que o desenvolvedor escreva as aplicações e treine as rotinas de processamento de imagens sobre os marcadores do mundo real que serão usadas na aplicação.O Artoolkit está presente no desenvolvimento de diversas aplicações acadêmicas, industriais e sistemas colaborativos.

Um Suporte à Interação Remota para Sistemas de Realidade Aumentada

O trabalho desenvolvido, como citado anteriormente visa fornecer suporte à comunicação e a interação entre os participantes de uma aplicação de RA distribuída, realizando o gerenciamento de acesso aos objetos virtuais por meio de controle de bloqueios e utilizando os serviços de transporte de sockets UDP *multicast.*
Foi estruturado de forma modular, cujos componentes são discutidos a seguir.

Estrutura do Suporte

Conforme ilustra a figura 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 Realidade Aumentada Distribuída.

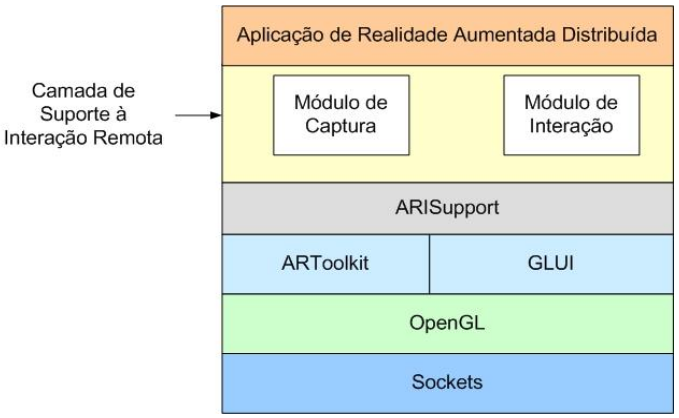


Figura 1: Estrutura em Camadas

A Camada de Suporte à Interação Remota foi estruturada como uma biblioteca dividida em dois módulos integrados: Módulo de Captura e Módulo de Interação.
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 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 (RADEMACHER, 2005).

Utilizou-se da biblioteca ARISupport (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 *sockets* UDP, utilizando o modelo de comunicação *multicast*.

Módulo de Captura

Oferece o gerenciamento e 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.
Esse módulo possui funções para configuração das propriedades do vídeo e da câmera, captura da imagem real, escolha de diferentes marcadores, cálculo da matriz transformação contendo as informações de posicionamento e orientação.

Módulo de Interação

Esse módulo realiza o controle e gerenciamento da interação dos objetos no ambiente virtual, como também a renderização dos objetos virtuais. Permite que os participantes envolvidos na aplicação executem ações como selecionar e manipular objetos virtuais, conseguindo dessa maneira alterar a posição e a orientação dos objetos. A interação pode acontecer por meio de marcadores, interface de controle 2D e pinças virtuais.

O Módulo de Interação também implementa um controle de bloqueio para permitir o gerenciamento de acesso à interação dos objetos virtuais, contendo uma função para determinar qual máquina terá o controle de interação sobre os objetos virtuais e uma função para realizar o pedido de controle para realizar a interação, permitindo que os usuários manipulem os objetos ou apenas visualizem.

Interação com Marcadores

O uso de marcadores é uma das formas de conseguir interação em sistemas de RA, pois através 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 por marcadores, são usadas as funções do Módulo de Captura para detecção dos marcadores e o próprio cenário real com as ações realizadas pelo usuário.

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

O usuário poderá rotacionar e transladar os marcadores e conseqüentemente interagir com o objeto virtual, pois este realizará os mesmo movimentos do marcador, como se estivesse preso ao outro (figura 2).

Figura 2: Interação com Marcadores

Dessa forma, a interação depende apenas 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 mãos.

A interação por meio de marcadores apresenta algumas limitações. Os objetos virtuais só podem ser exibidos quando os marcadores estiverem visíveis. Portanto, se os usuários cobrirem parte do padrão com suas mãos ou outros objetos, os objetos virtuais desaparecerão. Há limites também na inclinação dos marcadores.

Caso o usuário incline muito o marcador e relação ao eixo da câmera, o padrão inscrito no marcador pode se tornar irreconhecível pelo programa. 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 (AZUMA et al., 2001).

Existem ainda problemas relativos a limitações de espaço. Quanto maior for o tamanho físico do marcador, maior será à distância da qual será detectado e, portanto, maior será o volume do espaço de interação no qual o usuário poderá ser rastreado (AZUMA, 1997)

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

Interação por Interface de Controle 2D

Esse método apresenta uma função para criação da interface gráfica, responsável pela criação de controles de escala, rotação, botões, caixa de seleção e inserção da captura da imagem dentro do formulário da interface.

O processo de interação foi desenvolvido utilizando-se a biblioteca GLUI (*Graphic Library Interface*), que permite a construção de controles como botões, menus, caixas de seleção, entre outros (RADEMACHER, 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 chaleira, por meio de 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 à 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 (figura 3).

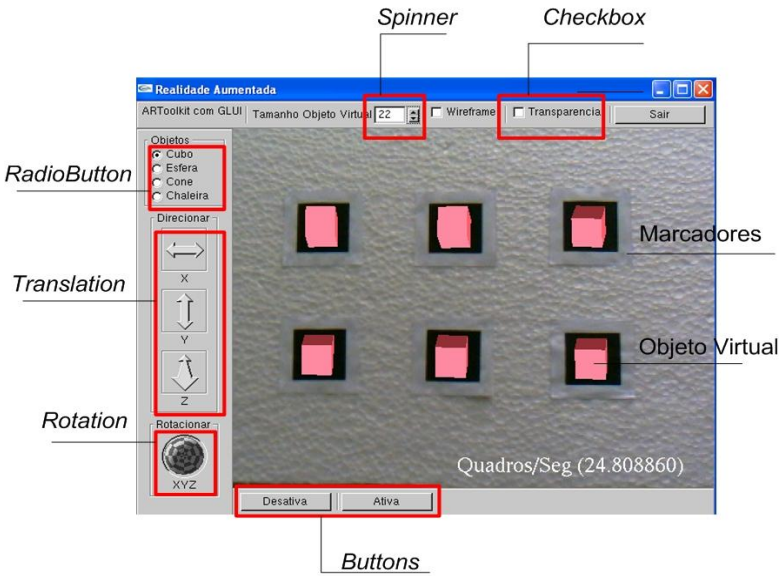


Figura 3: Controle de Interface 2D

Interação por meio de Pinças Virtuais

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

Na interação por meio de Pinças Virtuais utilizou-se uma função de detecção 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).

A interação foi realizada através de dois marcadores acoplados aos dedos, os quais representam objetos de interação e um marcador que representa um objeto fixo.
Esses marcadores foram sobrepostos por esferas virtuais, onde após a colisão entre as duas esferas, a esfera fixa passa a ter o mesmo comportamento herdado das ações dos marcadores que simulam os dedos do usuário (figura 4 (a) (b)).

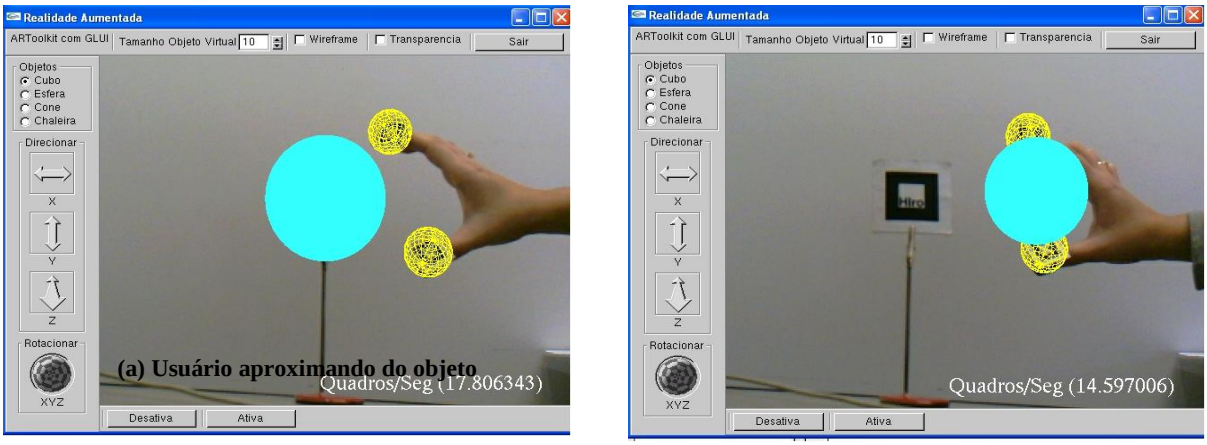


Figura 4: Interação por meio de Pinças Virtuais

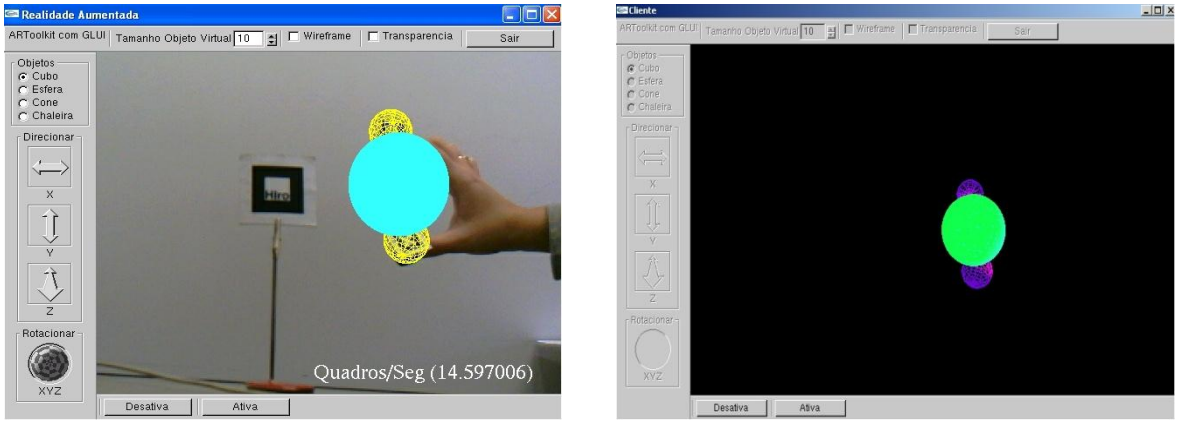
Implementação da Aplicação de Realidade Aumentada Distribuída

A fim de validar os serviços oferecidos pela Camada de Suporte, desenvolveu-se uma aplicação de RA distribuída.

Esta aplicação utiliza o modelo cliente-servidor, em que 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.

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

A aplicação servidora utiliza uma câmera para captura da imagem real e poderá realizar interação por meio de marcadores, interface 2D e pinças virtuais. Após capturar a imagem e detectar a posição dos marcadores envia o posicionamento e orientação dos objetos virtuais aos participantes do sistema, como também as modificações feitas nos mesmos (figura 5).



(a) Usuário movendo objeto no servidor **(b) Cliente visualizando objeto**

Figura 5: Aplicação de Realidade Aumentada Distribuída

A aplicação servidora tem a função de controlar a permissão de acesso para interação dos objetos virtuais (controle de bloqueios) e receber as interações realizadas a aplicação cliente.

A aplicação cliente permite receber as informações da aplicação servidora e também pode solicitar o controle de acesso para interagir com os objetos virtuais (figura 6).

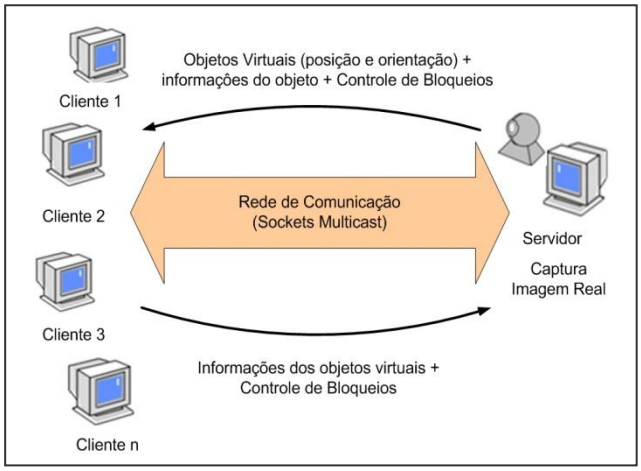


Figura 6: Arquitetura da Comunicação Cliente-Servidor

Aplicação Servidora

A aplicação servidora armazena a matriz transformação (informações de posicionamento e orientação do marcador) em uma estrutura de dados (figura 7) e a envia para o cliente, incluindo a identificação do marcador, a visibilidade dos objetos e as informações referentes à interação com o uso da interface de controle 2D, e logo após renderiza o objeto virtual através da API OpenGL. Através dessa estrutura de dados é possível enviar os dados armazenados à aplicação remota, possibilitando a visualização e a interação dos objetos virtuais por todos os participantes da aplicação.

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

As principais ações da aplicação servidora são:

a) Configurar a janela de exibição e associação dos padrões utilizados com os marcadores;
b) Estabelecer a conexão da aplicação com serviços de sockets UDP *multicast*;
c) Calcular e exibir a da taxa de frames por segundo;
d) Criar a interface de controle 2D, com botões e demais controles;
e) Capturar a imagem, detectar os marcadores e obter as respectivas matrizes de transformação. Estas informações são passadas aos clientes através da estrutura de dados;
f) Receber as informações de interação do cliente;
g) Realizar o controle de bloqueio;
h) Fechar a conexão com sockets *multicast*;
i) Encerrar a aplicação.

```
typedef struct {
  int frame;
  int objnum;
  infmarca marca[NUM_MAX];
  int objt2;
  int  esc;
  float PosXYZ[3];
  float Mat_Rotacao[16];
  float  DistTotal[2];
  double AuxA [3],AuxB[3],AuxC[3];
  int wire;
  int transp;
  int nmarca;
  int control;
  int pedido;
  int ident;
  int situacao;
  int acesso;
  clock_t tempo;
} infoframe;
infoframe b;
```

```
typedef struct {
int id;
float trans[3][4];
int objvisivel;
int infmarc;
} infmarca;
```

Figura 7: Estrutura de dados no Servidor

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.

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

Foi criada para essa aplicação, uma função para ajustar a taxa de exibição (frames por segundo) dos objetos de acordo com a captura executada pelo servidor, denominada de Evento_Tempo.

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.

No Suporte, a rotina Evento_Tempo possui uma função para realizar a transformação de matrizes vindas da aplicação servidora. Essa função foi implementada para tratar essas informações vindas do servidor, pois estas serão utilizadas para alterar a posição dos objetos na aplicação cliente. Como o OpenGL utiliza para criação do sistema de visualização um vetor de 16 posições (matriz 4x4) é necessário transformar a matriz 3x4 (vinda da aplicação servidora) em matriz 4x4, obter a sua transposta e armazena-la em um vetor de 16 posições.

A rotina Evento_Tempo também possui uma função 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 movimentação dos objetos virtuais na aplicação cliente, foi 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. Essa estrutura contém informações da matriz transformação, o número do frame, a visibilidade do objeto e outras.

Para permitir a interação dos objetos na aplicação cliente foram utilizadas técnicas de interação por meio de interface de controle 2D, presentes no Módulo de Interação da Camada de Suporte à Interação Remota; com isso foi necessário criar uma nova estrutura de dados (figura 8) para armazenar as interações do cliente sobre os objetos virtuais. Nessa estrutura serão armazenados as informações dos objetos e o controle de bloqueio.

```
typedef struct {
   int control;
   int esc;
   int Objt2;
   int wire;
   int transp;
   float Mat_Rotacao[16];
   float ObjtoPosXYZ[3];
   int ident;
   int pedido;
   int tipo;
   int situacao;
   clock_t tempo;
} info;
info l;
```

Figura 8: Estrutura de dados no Cliente

As ações da aplicação cliente são:

a) Estabelecer a conexão da aplicação através de serviços sockets UDP *multicast*;
b) Criar a janela de Interface 2D, com botões e demais controles;
c) Executar a função Evento Tempo, recebendo as matrizes de transformação e informações de interação enviadas pela aplicação servidora;
d) Enviar as interações para o servidor através da estrutura de dados;
e) Realizar o controle de bloqueios;
f) Fecha a conexão do sockets UDP *multicast*;
g) Encerra a aplicação.

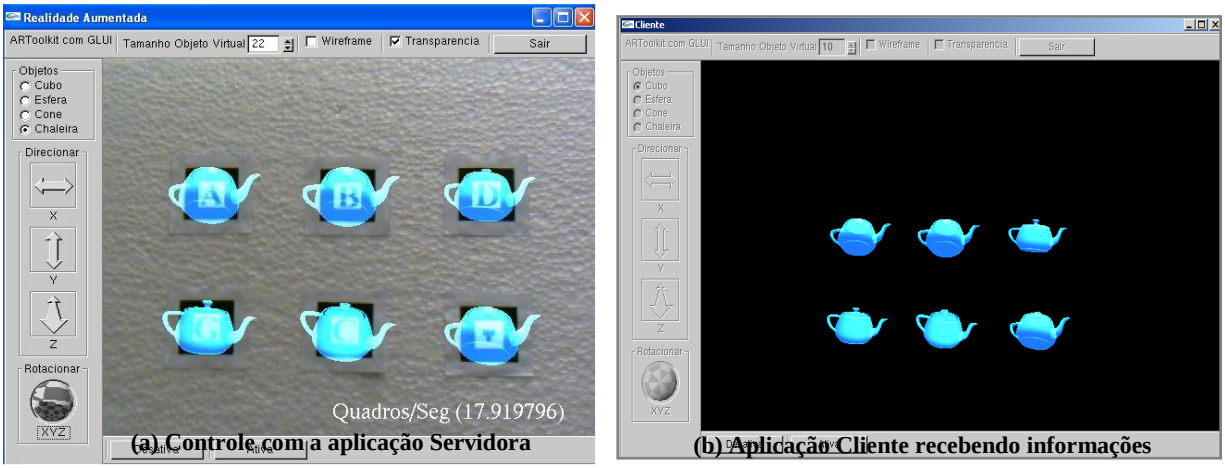
Controle de Bloqueios

O sistema desenvolvido permite que tanto a aplicação servidora quanto à 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, criou-se uma função Controle de Bloqueio, presente no Módulo de Interação, na Camada de Suporte à Interação Remota.

Cada aplicação possui botões de controle na sua interface gráfica, 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 de Interface Gráfica, presentes no Módulo de Interação.

Conforme ilustra a figura 9, 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) Controle com a aplicação Servidora (b) Aplicação Cliente recebendo informações

Figura 9: Janela das aplicações servidora e cliente

No inicio 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, a interface gráfica da aplicação cliente só permanece ativada os botões “Ativa” e “Desativa” para que ele possa realizar o pedido de interação, ficando as janelas dos controles desativada.

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. Caso aconteça de vários clientes pedirem permissão simultaneamente, o servidor libera o controle por ordem de chegada de pedido.

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.

RESULTADOS E DISCUSSÃO

Para realização dos experimentos foi utilizado um laboratório composto de nove microcomputadores PC (*Personal Computer*), com processador AMD *Athlon* 1500 + 1.33 GHz e um *notebook*, processador *Pentium* IV, sendo 1 servidor (*notebook*) e 9 clientes (PC) interligados em uma rede através de um switch , 16 portas, 10/100 Mbits. A figura 10 ilustra a disposição dos equipamentos utilizados.

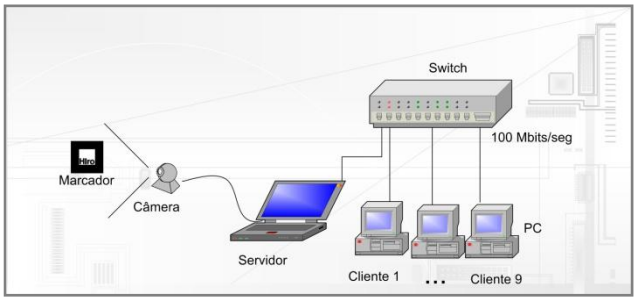


Figura 10: Disposição dos equipamentos

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 11).

Figura 11: Ambiente Experimental para validação dos testes

Os testes realizados concentraram-se na observação da latência 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 12 (a)), medindo 4 cm, a uma distância de 4cm entre eles para que não acontecesse oclusão dos marcadores.

Para realizar a interação por meio de pinças virtuais foram usados dois marcadores de 2 cm para serem os dedais (figura 12 (b)) e um marcador de 4 cm para ser o marcador fixo (figura 12 (c)).

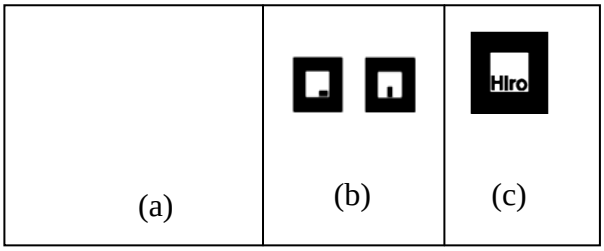


Figura12: Marcadores utilizados para teste

Latência da Comunicação

A latência foi obtida 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, subtrai desse valor o recebido na mensagem do cliente e divide pela função CLOCKS_PER_SEC.

Para realização da análise, as mensagens de solicitação de tempo foram enviadas 500 e 1000 vezes para os computadores (ou nós) participantes do sistema. As estimativas foram realizadas para nove nós clientes, utilizando seis marcadores estáticos (figura 13 e figura 14).

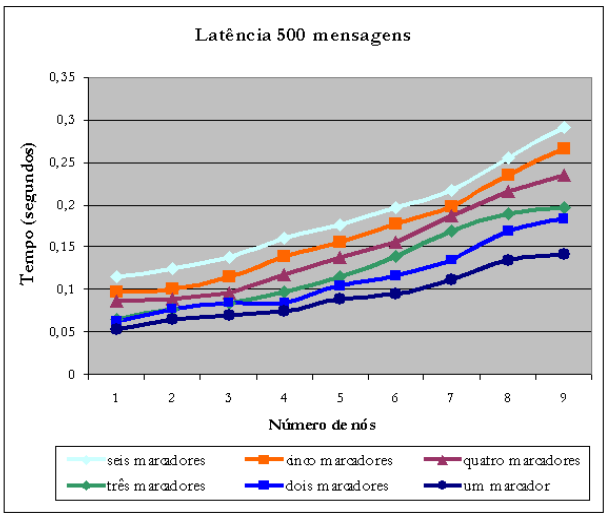


Figura 13: Latência para 500 mensagens

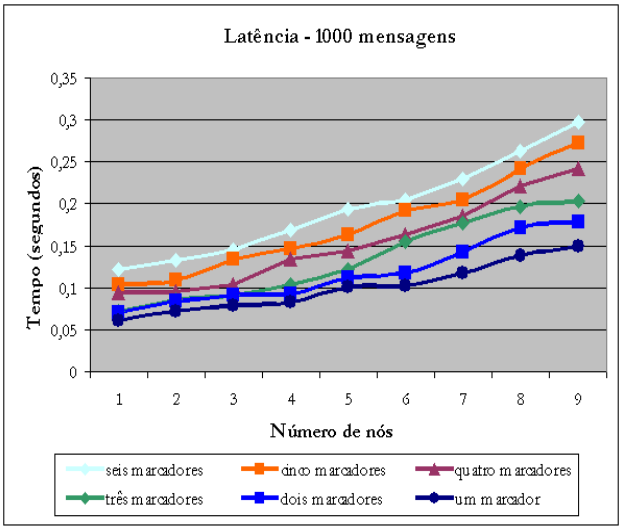


Figura 14: Latência para 1000 mensagens

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 significativamente 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.

Análise da Geração de Imagens

Em relação à geração das imagens foi analisada a taxa de frames 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, 2006).

Para essa análise, foram realizadas sessões de testes, como a taxa de frames obtida no servidor e no cliente utilizando marcadores estáticos; marcadores em movimento; interação 2D com cliente; interação 2D com o servidor; cliente e servidor interagindo simultaneamente. As mensagens foram enviadas 500 vezes aos participantes do sistema em cada sessão realizada e, posteriormente, foi calculada a média dos resultados obtidos (figura 15 e figura 16).

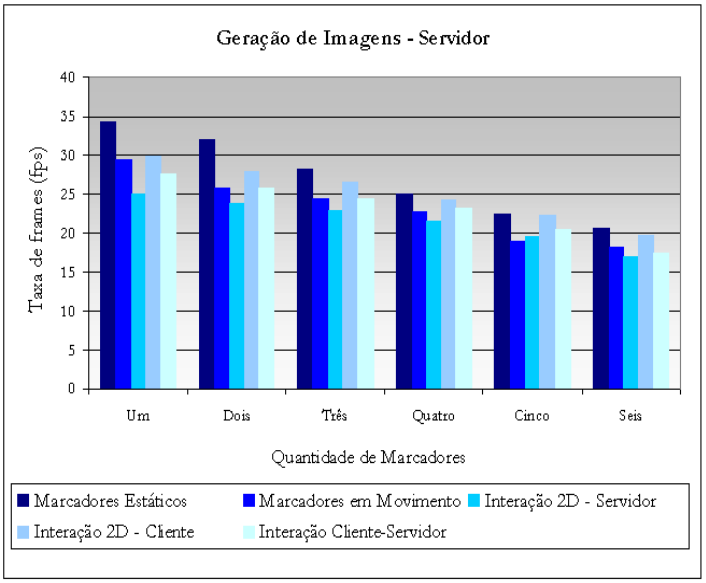


Figura 15: Geração de Imagens – taxa de frames – Comparação I

Observando os testes realizados para medir a taxa de frames por segundo na aplicação servidora (figura 15), 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, pois é necessário para cada marcador adicionado o reconhecimento do padrão, o cálculo da matriz transformação, a verificação de quais marcadores estão visíveis, gerando então um processamento maior e causando uma menor taxa de frames.

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

A forma mais comum de criar imagens gráficas tridimensionais por computador baseia-se no uso dos polígonos. A taxa de frames do ponto de vista gráfico depende da complexibilidade do mundo virtual, iluminação, sombreamento e texturas (DURLACH; MAVOR, 1995). 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. 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 objeto virtual "cubo" apresenta menos polígonos que um objeto virtual "chaleira", com isso, esse objeto apresenta maior facilidade de renderização, resultando em uma maior taxa de frames por segundo.

Analisando os resultados obtidos com os testes medidos na máquina cliente (figura 16), 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 aumentando a taxa de frames por segundo.

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

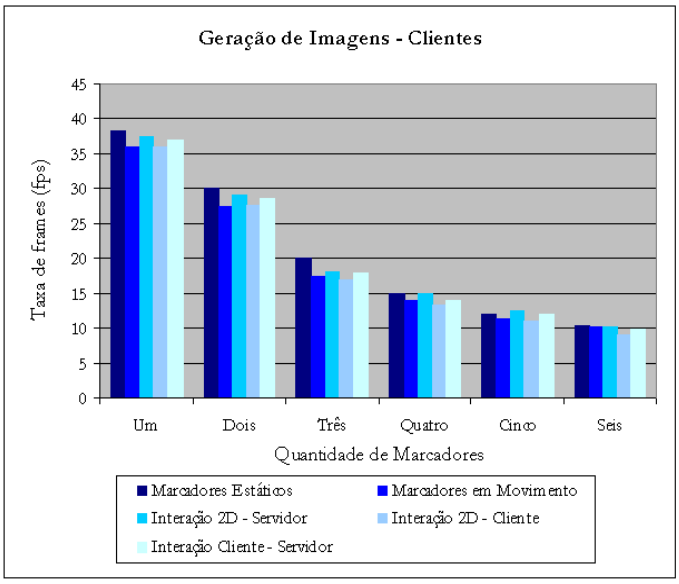


Figura 16: Geração de Imagens – taxa de frames – Comparação II

O teste com clientes e servidor interagindo simultaneamente foi realizado com 10 participantes e observou-se que não houve dificuldades no manuseio do sistema quanto ao uso por vários usuários. A interface gráfica utilizada ofereceu flexibilidade ao usuário, permitindo interagir com os objetos de uma maneira simples.

Porém, não pertence ao escopo deste trabalho, a análise da facilidade de uso da aplicação por parte dos participantes.

CONCLUSÕES

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 os objetos virtuais, transmitindo suas ações para todos os participantes da aplicação.

No protótipo implementado desenvolveu-se a interação de objetos por meio de interação direta de usuário com controles 2D, pinças virtuais e marcadores. 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.

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 em nível da aplicação de realidade virtual distribuída.

- Adição de câmeras nas aplicações clientes: 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 com marcadores e pela pinça virtual.
- Uso de Técnicas de Dead-reckoning: o dead-reckoning é uma técnica usada em AVDs, em que a idéia principal é transmitir pacotes apenas quando ocorrem determinadas atualizações. 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: Seria interessante usar na Aplicação de RA distribuída funções para carregamento de imagens, 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 área específica, por exemplo, ensino a distância, medicina, treinamento entre outros.

REFERÊNCIAS BIBLIOGRÁFICAS

AZUMA, R. A Survey of Augmented Reality. In: **Teleoperators and Virtual Environments 6, 4** (August 1997), p.355-385.

AZUMA, R. BAILLOT, Y. BEHRINGER, R. FEINER, S. JULIER, S. MACINTYRE, B. Recent advances in Augmented Reality. In **Presence IEEE Computer Graphics and Applications**, v. 21, no 6, 2001, p. 34-47.

BILLINGHURST, M. KATO, H. POUPYREV, I. The MagicBook – Moving Seamlessly between Reality and Virtuality. In: **IEEE Computer Graphics and Applications**, May 2001, p. 2 - 4.

BILLINGHURST, M. KATO, H. POUPYREW, I. MAY, R. Mixing Realities in Shared Space: In **Multimedia and Expo ICME 2000**. 2000 IEEE Internation Conferece, New York, volume 3, 2000, 1641-1644.

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

KATO, H. BILLINGHURST, POUPYREV, I. **Manual ARToolkit –** version 2.33, 2003

KAUFMANN, H SCHMALSTIEG D. Mathematics and Geometry Education with Collaborative Augmented Reality. In **Computer & Graphics**, vol. 27, no.3, p. 339-345, 2003.

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

LOPES, F. **O estudo e a implementação de interfaces para utilização em sistemas de realidade aumentada**. Dissertação de (Mestrado em Ciência da Computação) – Universidade Eurípides Soares da Rocha – UNIVEM, Marília, 2005.

MAUFER. THOMAS. **Deploying IP *Multicast* in the Enterprise**. Prentice Hall, 1998.

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

REITMAYR, G. SCHMALSTIEG, D. Mobile Collaborative Reality. In **Proceedings IEEE and ACM Internacional Symposium**, New York, NY, 2001, p.114-123.

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

ZINGHAL, S. ZYDA, M. **Networked Vitual Environments: Design and Implementation**. New York: Addison-Wesley, 1999.