UNIVERSIDADE METODISTA DE PIRACICABA
FACULDADE DE CIÊNCIAS EXATAS E DA NATUREZA
MESTRADO EM CIÊNCIA DA COMPUTAÇÃO

ARQUITETURA DE HARDWARE/SOFTWARE PARA INTEGRAÇÃO MODULAR DE INTERFACES MULTIMODAIS EM SISTEMAS DE REALIDADE AUMENTADA

CARLOS EDUARDO RIBEIRO

ORIENTADOR: PROF. DR. MARCIO MERINO FERNANDES

PIRACICABA, SP
2008

UNIVERSIDADE METODISTA DE PIRACICABA

FACULDADE DE CIÊNCIAS EXATAS E DA NATUREZA

MESTRADO EM CIÊNCIA DA COMPUTAÇÃO

ARQUITETURA DE HARDWARE/SOFTWARE PARA INTEGRAÇÃO MODULAR DE INTERFACES MULTIMODAIS EM SISTEMAS DE REALIDADE AUMENTADA

CARLOS EDUARDO RIBEIRO

ORIENTADOR: PROF. DR. MARCIO MERINO FERNANDES

Dissertação apresentada ao Mestrado em Ciência da Computação, da Faculdade de Ciências Exatas e da Natureza, da Universidade Metodista de Piracicaba – UNIMEP, como requisito para obtenção do Título de Mestre em Ciência da Computação.

PIRACICABA, SP
2008

ARQUITETURA DE HARDWARE/SOFTWARE PARA INTEGRAÇÃO MODULAR DE INTERFACES MULTIMODAIS EM SISTEMAS DE REALIDADE AUMENTADA

AUTOR: CARLOS EDUARDO RIBEIRO

ORIENTADOR: PROF. DR. MARCIO MERINO FERNANDES

Trabalho de qualificação apresentado em 27 de Fevereiro de 2008, pela Banca Examinadora constituída dos Professores:

_____

Prof. Dr. Marcio Merino Fernandes
UNIMEP

_____

Prof. Dr. Claudio Kirner
UNIMEP

_____

Prof. Dr. Jorge Luiz e Silva
USP (ICMC-São Carlos).

A

Minha esposa Juliana, meu anjo de toda hora, todo o dia, que me apoiou e serviu de companhia em todos os momentos.

Aos anjinhos desta família que enfeitam nossas vidas: Vinícius e Isadora.

AGRADECIMENTOS

Ao meu orientador Prof. Dr. Márcio Merino Fernandes, que tornou possível a realização deste trabalho.

A todos que de alguma forma, contribuíram para a sua construção.

“Bom mesmo é ir à luta com determinação, abraçar a vida e viver com paixão, perder com e viver com ousadia. Pois o triunfo pertence a quem se atreve, e a vida é muito bela para ser insignificante.”

Charles Chaplin

(1889 – 1977)

RESUMO

O avanço tecnológico e o crescimento da indústria fizeram com que a Realidade Virtual se tornasse viável em diferentes áreas de aplicações, tornando o seu acesso possível, até mesmo para usuários individuais. Com a evolução de conceitos e tecnologias da área, algumas variações surgiram recentemente, notadamente a Realidade Aumentada. A disponibilização de mundos virtuais a um público cada vez maior impulsiona o desenvolvimento de novos mecanismos de interatividade. Um requisito desejável é que, sempre que possíveis, esses novos mecanismos sejam implementados, utilizando-se tecnologias (hardware e software) convencionais, de modo a viabilizar o seu uso a um público mais amplo. Assim, o surgimento de novos dispositivos de interface, tais como aqueles que possibilitam o reconhecimento de fala, viabilizou novas possibilidades de interatividade, tornando-se assim uma tecnologia importante e emergente para o desenvolvimento de sistemas de Realidade Aumentada. O objetivo deste trabalho é propor e desenvolver uma arquitetura de hardware e software que facilite a integração de dispositivos de interação multimodais para sistemas de Realidade Aumentada, em particular aqueles baseados no software ARToolKit.

PALAVRAS-CHAVE: Realidade Virtual, Realidade Aumentada, Reconhecimento de Voz, Interfaces Multimodais, ARToolKit.

ARCHITECTURE OF HARDWARE/SOFTWARE FOR INTEGRATION OF MODULE MULTIMODAL INTERFACES IN AUGMENTED REALITY SYSTEM

ABSTRACT

The technological advancements and growth of the industry have made Virtual Reality to become practicable in different application areas, making it possible to be accessed even by individual users. With the development of new concepts and technologies in the area, some changes have emerged recently, notably the one named Augmented Reality. The availability of virtual worlds to a larger number of users drives the development of new mechanisms for interaction. A desirable requirement of those mechanisms is the possibility of implementing them using standard technology (hardware and software), in order to facilitate its use to a wider public. Thus, the emergence of new interface devices, such as those that allow speech recognition, open up new opportunities for interactivity, becoming an important technology for the development of Augmented Reality systems. The objective of this study is to propose and develop a hardware and software architecture aiming to facilitate the integration of devices for multimodal interaction into Augmented Reality systems, in particular those based the ARToolKit software.

KEYWORDS: Virtual Reality, Augmented Reality, speech recognition, Multimodal Interfaces, ARToolKit.

SUMÁRIO

LISTA DE FIGURAS

LISTA DE ABREVIATURAS E SIGLAS

2D	Duas Dimensões
3D	Três Dimensões
6DOF	Six Degrees Of Freedom
ABEL	Advanced Boolean Equation Language
API	Application Programming Interface
ASCII	American Standard Code for Information Interchange
ASIC	Application Specific Integrad Circuit
BOOM	Binocular Omni-Orientation Monitor
CFG	Context-Free Gramar
CI	Circuito Integrado
CLP	Circuitos Lógicos Programáveis
CMOS	Complementary Metal Oxide Semiconductor
CORBA	Common Object Request Broker Architeture
CPU	Central Processing Unit
DLL	Dinamic Link Libraries
DLP	Dispositivos Lógicos Programáveis
FPGA	Field Programmable Gate Array
GDI	Graphics Device Interface
GLP	General Public License
GLU	OpenGL Utility Library
GND	Ground
HCPLDs	High Complex Programmable Logic Devices
HDL	Hardware Description Languege
HTML	HyperText Markup Language
JVM	Java Virtual Machine
L&H	Lernout & de Hauspie®
LOD	Level os Details
LTP1	Line Print Terminal1
OpenGL	Open Graphics Library
OpenSG	Open Source Scenegraph
RA	Realidade Aumentada
RAD	Rapid Apllication Development
RGB	Red Green Blue
RMI	Remote Method Invocation
ROM	Read Only Memory
RPC	Remote Procedure Call
RV	Realidade Virtual
SAPI	Speech Application Programming Interface
SPLDs	Simple Programmable Logic Device
TCP/IP	Transmission Control Protocol/Internet Protocol
TTS	Text-to-Speech
USB	Universal Serial Bus
VB	Visual Basic
VHDL	Very High Speed Integrated Circuit Hardware Descripton Language
VRML	Virtual Reality Modeling Language

LISTA DE TABELAS

1. INTRODUÇÃO

O avanço tecnológico e o crescimento da indústria fizeram com que a Realidade Virtual (RV) se tornasse viáveis, não somente para grandes empresas ou instituições, mas também para usuários individuais e pequenas empresas, permitindo desta forma a criação de hardware e software de baixo custo para aplicações em diferentes tipos de sistemas. No âmbito empresarial, podemos destacar várias aplicações bem sucedidas de RV: sistemas para treinamento de controle de aeronaves, treinamento e simulação de cirurgias na área médica, desenvolvimento e experimentos com produtos voltados às áreas de fisioterapia e biomecânica, projeto e produção de peças e equipamentos industriais, etc.

Para usuários individuais e pequenas empresas, encontramos aplicações nas áreas de treinamento, entretenimento e principalmente os jogos eletrônicos. Com a disponibilização de mundos virtuais na Internet, os conceitos de navegabilidade e interatividade nos mesmos foram incorporados, permitindo interações até então não realizadas, como passeios em ambientes virtuais, manipulações de robôs, etc.

A interação entre usuários e um mundo virtual pode ocorrer através de equipamentos convencionais como teclado e mouse, ou equipamentos especializados para esse fim. O surgimento de novos dispositivos de interação, tais como capacetes, câmeras, etc., contribuíram para o crescimento da Realidade Aumentada (RA), um conceito relacionado à RV que permite a inserção de elementos virtuais em mundos reais, os quais podem ser manipulados pelos usuários.

O amadurecimento da tecnologia para o reconhecimento da fala viabiliza novas possibilidades de interatividade, tornando-se assim uma tecnologia importante e emergente para o desenvolvimento de sistemas de RA.

A partir do momento em que os mecanismos de reconhecimento de voz passaram a identificar fonemas, sílabas e palavras e gerar instruções que passaram a ser executadas por computadores ou equipamentos eletro-eletrônicos, viabiliza-se um novo conceito de interatividade entre homem e máquina. O reconhecimento de fala associado a outros dispositivos de interação constitui um sistema multimodal de interação, permitindo o surgimento de novas e interessantes formas de aplicações utilizando RV e RA. As interfaces multimodais representam o conjunto de modalidades possíveis para a entrada e saída de informações entre usuários e sistemas.

1.1 OBJETIVOS E JUSTIFICATIVA DO TRABALHO

O objetivo deste trabalho é desenvolver uma *arquitetura de hardware e software que facilite a integração de dispositivos de interação multimodais para sistemas de Realidade Aumentada*. Em termos de hardware, a arquitetura se apoiará em computadores pessoais, redes e dispositivos de hardware dedicado, estes últimos usados para a implementação de dispositivos de interface. Em termos de software, a arquitetura fará uso de sistemas operacionais, software de rede, sistemas de reconhecimento de voz, e softwares para RA comumente empregados, neste caso em particular o ARToolKit [ARToolKit,2006]. Para a aplicação dos conceitos e soluções propostos neste trabalho, será desenvolvido um protótipo, o qual será chamado de **ARToolKit Multimodal**.

Os sistemas de computação atuais são baseados em hardware produzido em larga escala, tipicamente computadores pessoais ou estações de trabalho utilizando os sistemas operacionais MS-Windows® ou Linux. Porém, diversos outros sistemas não se adaptam bem a essas plataformas, em virtude de particularidades como tamanho, consumo de energia, portabilidade, etc., exigindo a utilização de dispositivos de hardware dedicado, ou ainda os chamados sistemas embarcados. Um exemplo é o reconhecimento de sinais em centrais de atendimento telefônico, onde um Circuito Integrado (CI) é

responsável por toda a operação lógica do sistema. Em alguns casos, podemos encontrar ainda a co-design de sistemas compostos por subsistemas baseados em hardware e em software. Com base neste conceito, este trabalho visa propor uma arquitetura englobando essas duas possibilidades, dado a provável heterogeneidade nas alternativas para o projeto e implementação de interfaces multimodais. Desse modo, a arquitetura proposta pode ser dividida em dois subsistemas: um oferecendo suporte aos *dispositivos de interação baseados em software*, e outro àqueles *baseados em hardware*.

O subsistema para dispositivos baseados em software deverá se apoiar em componentes de interface já existentes, cabendo ao projetista do sistema preocupar-se apenas com a implementação do mecanismo de interação em si. Este trabalho utilizará um sistema de software para reconhecimento de voz, o qual apresenta características funcionais interessantes para um sistema de RA. Além disso, esse sistema ilustra de maneira adequada os principais aspectos da arquitetura proposta, bem como a integração de outros mecanismos de interação baseados em software.

Nos últimos anos, tem-se observado a proliferação de sistemas e dispositivos para a implementação de interfaces baseadas no reconhecimento de voz. Porém, a utilização desses sistemas ainda apresentam desafios importantes, destacando-se entre eles: a necessidade de processamento em tempo real, exigência de baixo custo, necessidade de traduzir o sinal acústico, levando-se em consideração diferentes parâmetros como volume, sotaques de fala dos seres humanos, ambigüidade no significado de palavras, taxas de erro na interpretação, etc. Apesar de já existir há vários anos, só recentemente a tecnologia relacionada ao uso da voz tornou-se disponível para o público e programadores em geral. Como exemplo, na plataforma MS-Windows®, encontramos uma API (*Application Programming Interface*) capaz de realizar o reconhecimento de voz e cooperar com outros aplicativos, denominada SAPI (*Speech Application Programming Interface*).

O subsistema para dispositivos baseados em hardware deverá assumir o uso de dispositivos de interação não padronizados, desde os mais simples como: teclado, mouse e joysticks, até outros mais complexos e customizados, implementados utilizando tecnologias de circuitos integrados ou FPGAs (*Field Programmable Gate Array*).

Dessa forma, os subsistemas acima mencionados oferecerão o suporte necessário para o desenvolvimento de uma arquitetura, possuindo características de *portabilidade*, e também capacidade de *extensão*, no que diz respeito à utilização de novos dispositivos de interface. Neste caso, pretende-se oferecer uma *plataforma para a execução de sistemas de RA que possa acomodar dispositivos de interface adicionai, sem mudanças ou adaptações substanciais*. Como objetivo deste trabalho destaca-se também a construção de um *protótipo da arquitetura a ser proposta*, o qual servirá como prova de conceito, e para a aplicação de alguns testes de usabilidade.

Como justificativa deste trabalho, a arquitetura a ser proposta visa expandir as possibilidades de projeto e implementação de sistemas multimodais para RA, hoje baseados em software na sua grande maioria, e hardware proprietário em alguns poucos casos.

1.2 ESTRUTURA DO DOCUMENTO

Os capítulos que fazem parte deste documento refletem em grande parte a evolução dos estudos e atividades que se fizeram necessários para o desenvolvimento deste trabalho de mestrado.

No capítulo 02, será apresentado um resumo da revisão bibliográfica efetuada, conceituando-se os tópicos relevantes aos conceitos teóricos relacionados ao projeto. Em linhas gerais, a revisão bibliográfica abordou os seguintes tópicos:

- Realidade Virtual: Descrição de conceitos, tipos de ambientes e formas de interação e liberdade de movimento.
- Realidade Aumentada: Definição de realidade e virtualidade aumentada e interação.
- ARToolKit: fundamentos do software, interação com usuário, marcadores, calibração, estrutura e OpenGL.
- Interfaces Multimodais: Dispositivos e sensores.
- Sistemas Embarcados: Definição de sistemas, circuitos lógicos programáveis (CLP, DLP e FPGA), arquitetura e programação HDL.
- Sistemas Distribuídos: Conceitos, TCP/IP e *Sockets*.

O Capítulo 03 apresenta as considerações e proposta inicial para o modelo conceitual da arquitetura a ser desenvolvida, baseada em uma interface genérica entre os diversos dispositivos de interação, e o sistema ARToolkit.

No Capítulo 04 serão apresentados detalhes de implementação de uma série de protótipos, que podem ser vistos como instâncias da arquitetura proposta no Capítulo 3. Cada um desses protótipos ilustra a utilização de um tipo de interface, conforme se segue:

- Interface de voz baseada em software
- Interface de toque baseada em hardware
- Interface com joystick baseada em software
- Interface com mouse baseada em software

O Capítulo 05 apresenta uma descrição da arquitetura, integrando os subsistemas de hardware e software, ilustrado pelos protótipos desenvolvidos no Capítulo 04, a qual constitui o ARToolkit Multimodal.

O Capítulo 06 apresenta algumas possíveis aplicações práticas com o ARToolKit Multimodal, as quais podem servir como base para o desenvolvimento de aplicações mais elaboradas.

Finalmente, as conclusões deste trabalho, bem como possíveis melhoramentos que poderão ser feitos no futuro.

2. REVISÃO BIBLIOGRÁFICA

A revisão bibliográfica que se segue engloba diferentes tópicos, tais como: Realidade Virtual, Realidade Aumentada, ARToolKit e OpenGL no que diz respeito as tecnologias de software. Diferentes interfaces e suas relações serão representadas através do conceito multimodal, e entre estas interfaces os dispositivos dedicados apoiados em conceitos como: Sistemas Embarcados, Circuitos Lógicos Programáveis, com enfoque na Tecnologia FPGA, o qual representa uma boa alternativa para o desenvolvimento de sistemas embarcados e por último Sistemas Distribuídos, abordando os conceitos de Sockets e mapeamento IP (*Internet Protocol*), que de certa forma são pertinentes à execução deste projeto, incluindo aspectos teóricos e práticos.

2.1 REALIDADE VIRTUAL

A Realidade Virtual já tem muito tempo de história, começando em 1950, com um cineasta concebendo o primeiro equipamento que permitia a imersão no mundo Virtual tridimensional, logo depois em 1960, o surgimento do capacete e em 1980 a sua consolidação. Portanto, Realidade Virtual é uma convergência de tecnologias para a geração de ambientes tridimensionais interativos em tempo real, permitindo que o usuário por meio de recursos computacionais tenha uma representação interativa e imersiva [KIRNER, 2006].

Com a Realidade Virtual as ações dos usuários que eram representadas por um aperto de botão, acionamento de menus e outras ações ora provenientes do mundo 2D (duas dimensões), agora são representadas por ações tridimensionais, tais como: girar uma engrenagem, acionar um interruptor, girar uma chave, etc. Além de permitir que ambientes reais sejam representados.

Estas interações podem ter um preço. Por exemplo: a utilização de equipamentos especiais, como: capacetes, luvas, mouse 3D (três dimensões), entre outros, podem gerar um desconforto ao usuário no que diz respeito ao aspecto econômico ou tecnológico.

A interatividade entre Realidade Virtual e dispositivo pode ser analisada da seguinte forma:

- Imersão: quando o usuário faz parte do ambiente, sendo transportado para o ambiente da aplicação, utilizando dispositivos multisensoriais, e não-imersiva, quando o usuário participa do ambiente através de uma janela (monitor).
- Interação: modificação do mundo Virtual por meio de ações do usuário.
- Envolvimento: identifica o grau de relacionamento do usuário com o sistema, podendo ser passivo ou ativo. A participação na atividade proposta pelo sistema, pode ser uma exploração dirigida, onde o usuário escolhe o caminho e os pontos de observação, ou interativa, apresentando objetos virtuais que respondem e ou reagem às ações do usuário.

Sendo a navegação a forma mais simples de interação, a posição do usuário (*viewpoint*) pode determinar algumas técnicas [NETO,2002]:

- *Point-and-fly*: o usuário movimenta o guia Virtual utilizando qualquer dispositivo de navegação.
- *Eyeball-in-hand*: um dispositivo realiza uma leitura a partir de um ponto de observação do usuário, desta forma, carregando as informações. Um exemplo e o BOOM (*Binocular Omni-Orientation Monitor*), onde um braço mecânico e um display são movimentados pelo usuário, a figura 01, ilustra o modelo.

Figura 01 – BOOM [BEIER,2006]

- *Scene-in-hand*: é o oposto ao anterior, onde a movimentação ocorre no mundo Virtual fixando o ponto de observação do usuário.

Em alguns casos, são implementadas técnicas especiais, como a leitura labial, reconhecimento de voz, movimentos da cabeça ou dos olhos ou até mesmo uma mistura de diferentes técnicas, criando assim dispositivos multimodais, os quais serão detalhados posteriormente.

Toda essa liberdade oferecida pela Realidade Virtual, no que diz respeito à movimentação e navegação em ambientes virtuais, é caracterizada pelo conceito 6DOF (*Degrees Of Freedom*), seis graus de liberdade, permitindo que os movimentos sejam realizados em qualquer direção, especificados pelos eixos x,y e z, representando os movimentos de translação e rotação. A figura 02 mostra como os eixos são representados, podendo aumentar ou diminuir, além da rotação no eixo (direita e esquerda), desta forma apresentando dois tipos de movimentos para cada eixo.

A Realidade Virtual [KIRNER, 2006], é uma interface de usuário avançada, que permite a visualização e o movimento em ambiente virtual tridimensional em tempo real e a possibilidade de interação com esse ambiente, podendo ser aplicados outros sentidos como tato e audição.

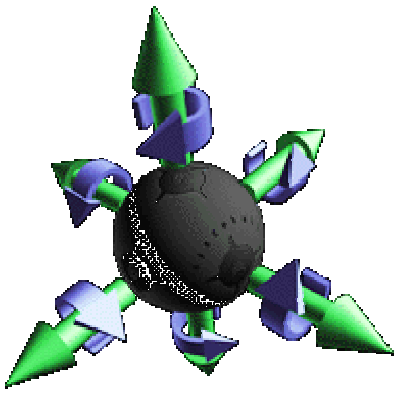


Figura 02 – 6DOF
[MINDFLUZ,2006]

Outros detalhes da Realidade Virtual, a serem considerados [KIRNER, 2006]:

- O nível de detalhamento dos objetos representados em uma cena de acordo com a sua distância, ou seja, quando mais longe o objeto está de seu observador menor é o número de detalhes do objeto, definido como LOD (*Level of Details*).
- A iluminação é realizada de maneira bem simples, garantindo um bom grau de realismo.
- A colisão, que permite implementar o conceito de objetos sólidos na aplicação Virtual, evitando que os usuários ou objetos atravessem outros objetos ou ambientes.

2.2 REALIDADE AUMENTADA

De forma inversa da Realidade Virtual, a Realidade Aumentada transporta o ambiente virtual para o espaço físico do usuário [KIRNER, 2006], através da utilização de interfaces comuns ou multimodais. Em um contexto mais amplo temos a realidade misturada que implica na inserção de objetos virtuais no mundo real, possuindo duas particularidades a realidade aumentada e a virtualidade aumentada, sendo:

- Realidade Aumentada: manipulam objetos virtuais em mundo real, possuindo alto desempenho em renderização de imagem, interação em tempo real e calibração, priorizando a interação com o usuário.

- Virtualidade Aumentada: manipulam objetos reais em mundo virtual, possuindo um mecanismo para combinar o mundo real com o virtual, enfatizando a qualidade de imagem e a interação do usuário.

A figura 03 a seguir ilustra o ambiente da Realidade Misturada.

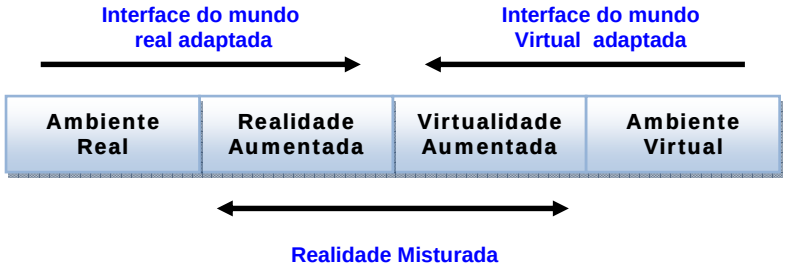


Figura 03 – Realidade Misturada [KIRNER, 2006]

Essa combinação da realidade aumentada permite que ambientes virtuais sejam suplementados, através da incorporação de objetos virtuais, podendo até mesmo ser manipulados, conforme visualizado na figura 4.

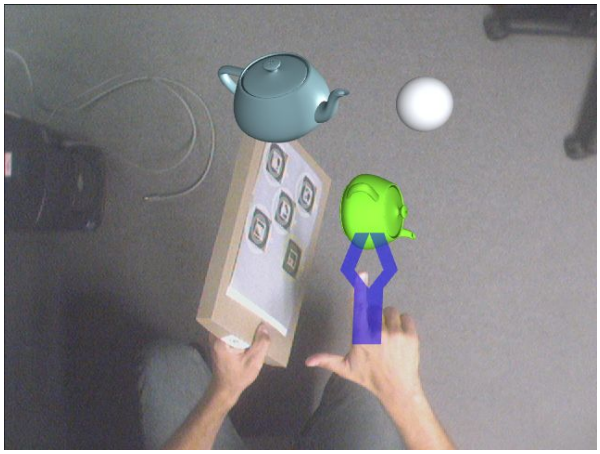


Figura 04 – Interação com objetos virtuais [KOLSCH,2006]

As definições de visualização imersiva e não-imersiva, podem ser analisadas da seguinte forma:

- Imersiva: quando o usuário vê o mundo misturado, apontando diretamente os olhos, neste caso utilizando um dispositivo, como por exemplo, os óculos 3D, visualizado na figura 05.

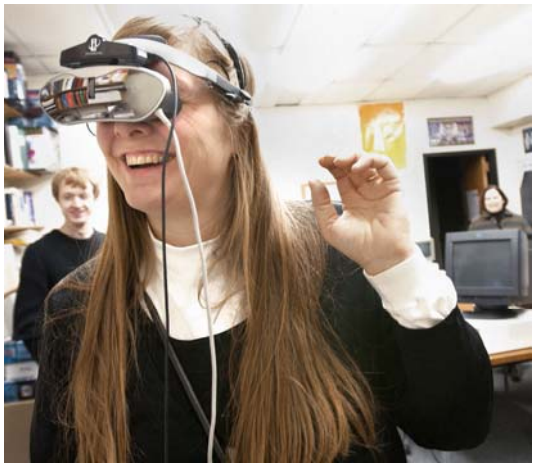


Figura 05 – Visualização imersiva [CLARKSON,2006]

- Não-imersiva: a visão do mundo ocorre através de um sistema de projeção, através de um monitor ou projetor como mostra a figura 06.

Figura 06 – Visualização não imersiva [MACHADO,2007]

2.3 ARTOOLKIT (*AUGMENTED REALITY TOOLKIT*)

Trata-se de uma ferramenta utilizada para desenvolvimento de aplicações para Realidade Aumentada, constituída por uma biblioteca em linguagem "C", de livre uso para aplicação não-comercial e distribuída "*open-source*" sob licença GPL (*General Public License*). O ARToolKit foi desenvolvido primeiramente pelo Dr. Hirokazu Kato da universidade de Osaka, Japão, e conta com o suporte das universidades e *Washington* (EUA) e *University of Canterbury (New Zealand)* [ARToolKit,2006]**.** O download da aplicação pode ser feito a partir de [ARToolKit,2006].

Através de técnicas de visão computacional, o ARToolKit é capaz de calcular a posição da câmera em relação a marcadores, permitindo desta forma que objetos virtuais sejam sobrepostos nos marcadores [CONSULARO,2006]. Seu funcionamento é baseado nos seguintes princípios, figura 07.

- A imagem de vídeo é capturada.
- A ferramenta encontra todos os quadriláteros e verifica qual padrão se encontra dentro deles, comparando-o com padrões pré-definidos.
- Propriedades da forma do quadrilátero serão usadas para determinar a posição da câmera em relação ao marcador físico.
- Determinada as coordenadas, os objetos virtuais são desenhados sobre as marcas reais.

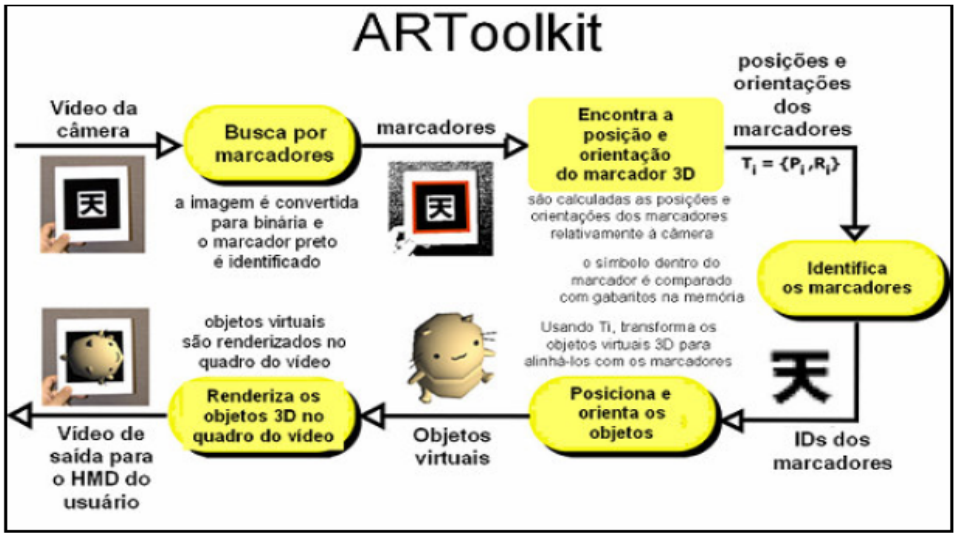


Figura 07 – ARToolKit [ARToolKit,2006]

A sobreposição dos objetos podem ser definidas à partir do OpenGL (*Open Graphics Library*), ou através do VRML (*Virtual Reality Modeling Language*) . Na figura 08, podemos observar os marcadores dispostos sobre a mesa e objetos virtuais flutuando sobre os marcadores.

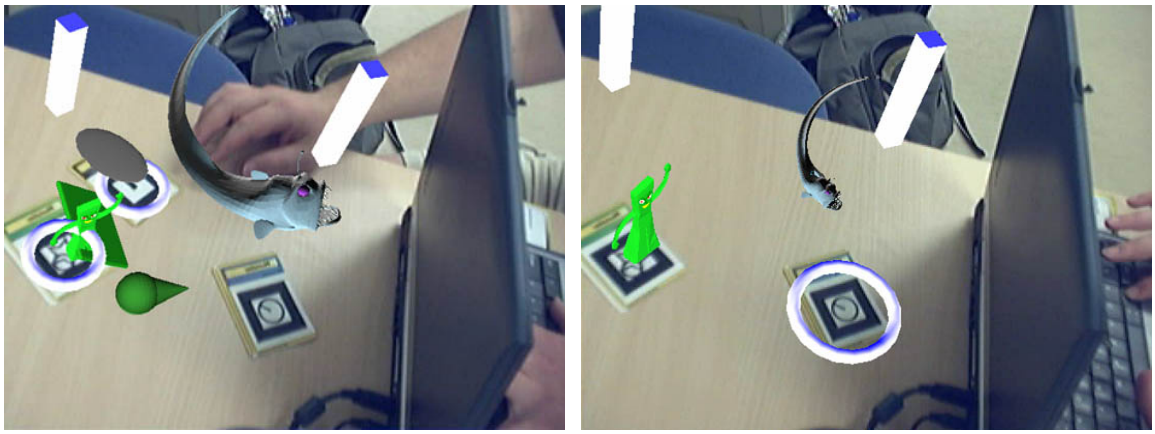


Figura 08 – Sobreposição de marcadores [GRONBCEK,2006]

É importante que os cálculos gerados pelo ARToolKit, sejam bem próximos da relação Mundo Real x Mundo Virtual, para isso a câmera deve estar devidamente calibrada. Este processo é obtido através dos aplicativos que acompanham o ARToolKit, realizando o ajuste necessário a câmera, em relação a distância focal, ponto central da imagem e distorção da lente [CARDOSO,2004], ilustrados em execução nas figuras 09 e 10.

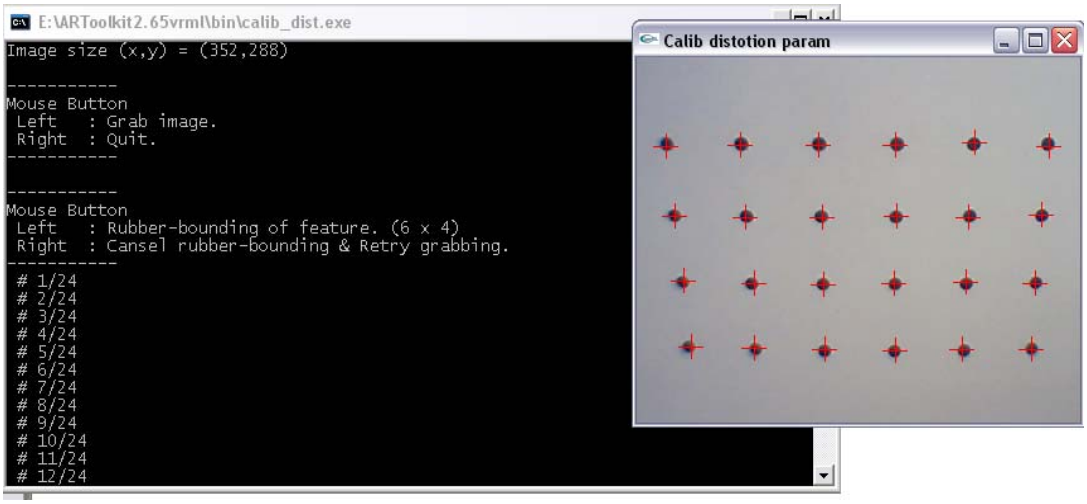


Figura 09 – Parâmetros de calibração

A figura 09 representa uma das etapas do processo de calibração do ponto central da câmera e a distorção da lente. Através de uma grade de pontos 6x4 separados uniformemente e disposta na frente da câmera, nota-se a irregularidade entre o espaçamento dos pontos. Essa irregularidade é utilizada para o cálculo da distorção da lente.

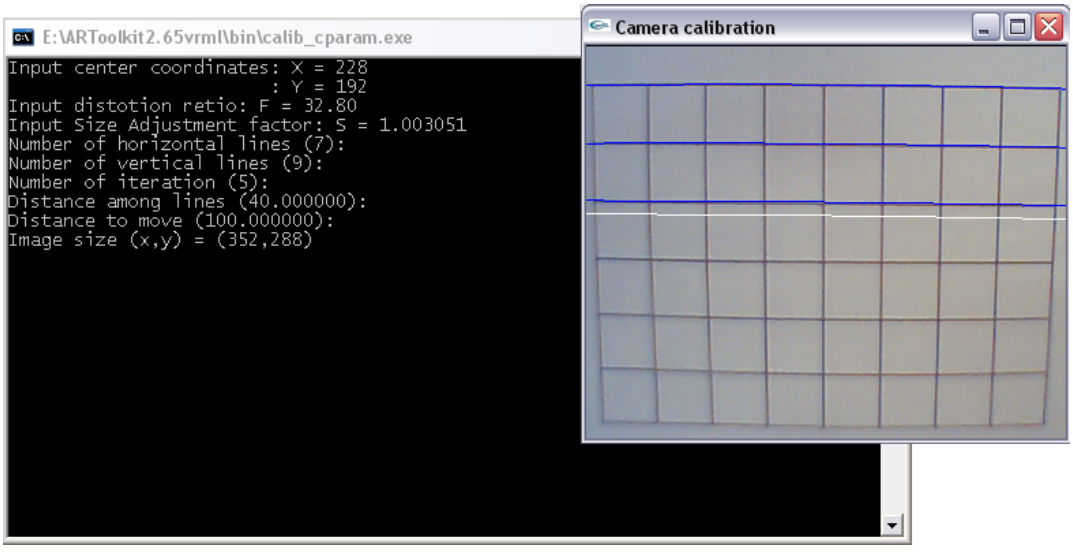


Figura 10 - Calibração

Para a calibração da distância focal da câmera, é utilizada uma grade de 7x9 linhas. Na figura 10, podemos verificar as linhas marcadas durante o

processo, estas informações serão gravadas e usadas posteriormente como parâmetros para a câmera.

2.3.1 OPENGL (*OPEN GRAPHICS LIBRARY*)

Com a disponibilidade de diferentes tipos de hardware, podemos aproveitar a capacidade que o hardware oferece no desenvolvimento de aplicações gráficas, a *OpenGL* (*Open Graphics Library*) é uma biblioteca que possibilita a modelagem 2D e 3D de maneira rápida e portátil. Desta forma, a *OpenGL* caracteriza-se como uma poderosa API (*Application Programming Interface*), podendo ser utilizada em diferente plataformas, como por exemplo a Linguagem “C”.

Sua portabilidade permite através de chamadas as funções da própria biblioteca ou da biblioteca GLU (*OpenGL Utility Library*) que acompanha o pacote, a criação de cenas 2D ou 3D. Dentro do ARToolKit, podemos encontrar implementações OpenGL para geração de imagens, um exemplo clássico é o cubo renderizado no estado default da aplicação, como podemos verificar na figura 11.

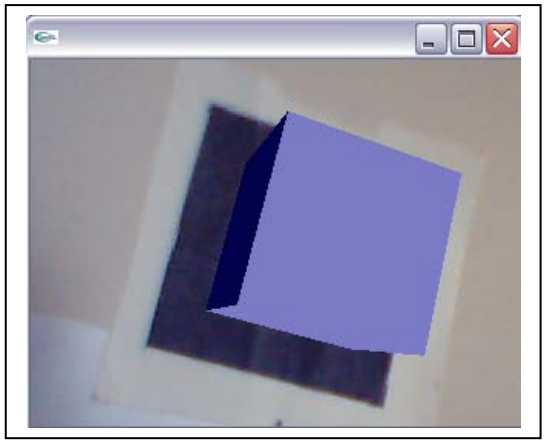


Figura 11 – Geração de Imagem 3D com OpenGL

2.4 INTERFACES MULTIMODAIS

Segundo Oviatt, interfaces multimodais podem ser definidas como a fusão de diferentes dispositivos de entrada dentro de uma modalidade [OVIATT,2003]. No ingresso ao mundo Virtual pode ser realizado por intermédio de diferentes dispositivos de entrada, os quais podem ser destacados.

- Som (voz) – O som poder ser tratado como uma fonte de comandos. Interpretado por um mecanismos de hardware ou software, o som em um caso mais específico a fala pode gerar comandos para uma determinada aplicação.
- Mouse 3D – O mouse apresenta o “degrees-of-freedom” (seis graus de liberdade), que permite que o usuário movimente objetos e o mouse livremente pelos eixos x, y e z, bem como movimentar a câmera nesses eixos, figura 12.

Figura 12 – Mouse 3D [UFRGS,2007]

- Luvas (glove) – reconhece os movimentos das mãos para manipulação ou movimentação no ambiente Virtual. Alguns equipamentos possuem o sistema de force feedback, que permite reações táteis.
- Sensores biológicos – reconhece sinais através de impulsos elétricos emitidos pelos músculos.

Figura 13
Sensor [UFRGS,2007]

- Dispositivos de trajetória – rastreamento dos movimentos de objetos. Sua operação é baseada na diferença de posição ou orientação em relação a um ponto, figura 13.
- Dispositivo de locomoção – os quais simulam a mobilidade humana (andar ou correr).

Em outros casos, além dos dispositivos convencionais como teclado, mouse, webcam, etc., podemos realizar a fusão de dois ou mais para a criação de dispositivos multimodais.

A utilização de interfaces multimodais pode aumentar a interação homem-máquina, tornando-se desta forma uma preferência para o usuário [OVIATT,2003]. Sharon Oviatt, em um de seus trabalhos, relata a utilização de interfaces multimodais que envolvem a fala associada a outros dispositivos como caneta óptica e leitura labial, relacionando alguns elementos importantes:

- O uso de interfaces multimodais é mais exato, pois permite um grande volume de informações, contudo esse volume não deverá interferir no processamento, deixando a aplicação lenta se comparada com um sistema unimodal.

- A fala humana (reconhecimento de voz) pode perder o grau de interação, quando levado em consideração os diferentes tipos de linguagens, sotaques, ambigüidades de palavras e diferentes timbres de voz (adulto e criança). Neste caso, o ruído do ambiente também poderá interferir no resultado.

- Movimento labial pode apresentar falha, quando comparado com os seus respectivos sons.

Interfaces multimodais podem ser utilizadas para a navegação em ambientes virtuais, usando a fala combinada com outros dispositivos, convencionais ou não. Conforme Frank e outros [ALTHOFF, 2003], uma proposta é um sistema utilizando uma gramática livre de contexto (CFG - *Context-Free Grammar*), o qual está representado na figura 14.

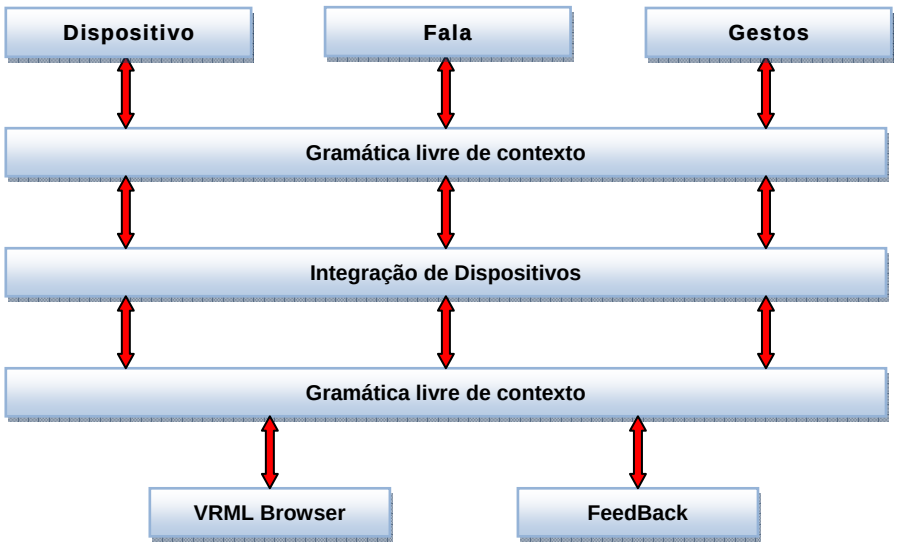


Figura 14 – Proposta interface multimodal [ALTHOFF, 2003],

A utilização de interfaces multimodais é mais representativa desde o usuário iniciante aos mais experientes dando preferência a fala [ALTHOFF, 2003].

Conforme [IRAWATI,2006], as interações multimodais também tem sido utilizadas para manipulações em ambientes de Realidade Aumentada, usando dispositivos como: voz, gesto, luvas, etc., e que relativamente existem poucas aplicações que utilizam-se dispositivos multimodais. A proposta multimodal apresentada por Sylvia Irawati e outros, consiste na elaboração de uma interface multimodal utilizando o Microsoft Speech API 5.1 para o reconhecimento de voz apoiado em um sistema de Realidade Aumentada no caso o ARToolKit, cujos comandos utilizados para o reconhecimento de voz encontram-se armazenados em uma base de dados. Agregados a este conceito introduziu-se os gestos como forma de entrada na interação.

2.5 SISTEMAS EMBARCADOS

Sistemas embarcados, tradução do termo *Embedded Systems*, podem ser definidos em linhas gerais como um sistema específico constituído por um processador programável. Ao contrário dos microcomputadores, o sistema

embarcado é destinado a uma tarefa específica, além de apresentar baixo consumo de energia e custo de produção [BONATO, 2004].

A Microsoft®, por exemplo, disponibiliza os sistemas operacionais Windows XP® *Embedded* e Windows CE.NET®, para serem aplicados em produtos eletrônicos de consumo, gateways, quiosques, dispositivos móveis e portáteis, pontos de vendas.

Um sistema embarcado possui como característica: a execução de um programa repetitivamente [BARROS,2006], baixo consumo de energia, são pequenos e rápidos, ainda atuam com o ambiente reagindo a suas mudanças e permitem que dados sejam computados em tempo real.

Sua estrutura [ZELENOVSKY,2006] é determinada pelo tipo de aplicação levando-se em consideração os dados de entrada/saída, tempo de resposta. A figura 15 representa a estrutura típica de um sistema embarcado.

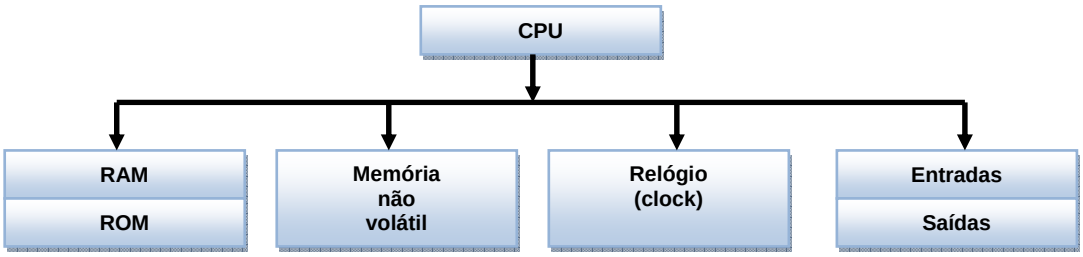


Figura 15 – Arquitetura sistema embarcado [ZELENOVSKY,2006]

O sistema embarcado utiliza-se de diferentes tipos de CPU (*Central Processing Unit*):

Simples: 08 bits: 8085, Z-80
16 bits: 8086, 8088, 80186, 80188, 80286, 68000, 68010.
Médias: compatíveis com a família 386 e 486.
Sofisticadas: compatíveis com a família Pentium.

Na arquitetura de sistema embarcado, figura 16, podemos encontrar hardware e software no mesmo dispositivo, como é o caso da plataforma A7E10-R2 da empresa CNX Tecnologia em Informática, que possibilita conexões em Ethernet, utilizando protocolos TCP/IP(*Transmission Control*

Protocol/Internet Protocol), podendo ser aplicados como: conversores de protocolos, sistemas distribuídos, integração de equipamentos, terminais de venda e dispositivos de controle.

Figura 16 - Sistema embarcado [BORESTE,2006]

2.5.1 CIRCUITOS LÓGICOS PROGRAMÁVEIS

Sempre pensamos em implementações imaginando o software, mas alguns sistemas mais complexos ou com alguma particularidade em especial podem exigir um nível de implementação em hardware, ou seja, realizar a programação diretamente no hardware. A utilização de ferramentas de programação permite o desenvolvimento de projetos de controladores digitais, utilizando para isso um dispositivo programável, conhecido como DLP (Dispositivo Lógico Programável).

Esses controladores caracterizam-se em executar um algoritmo de controle gravado em uma memória ROM (*Read Only Memory*), permitindo a comunicação com dispositivos externos para troca de informações. A figura 17 mostra a estrutura básica de um CLP (Circuitos Lógicos Programáveis), constituído de:

- Dispositivo de entrada/saída.
- Memória.
- CPU (*Central Processing Unit*).
- Circuitos de controle e comunicação.
- Fonte de alimentação.

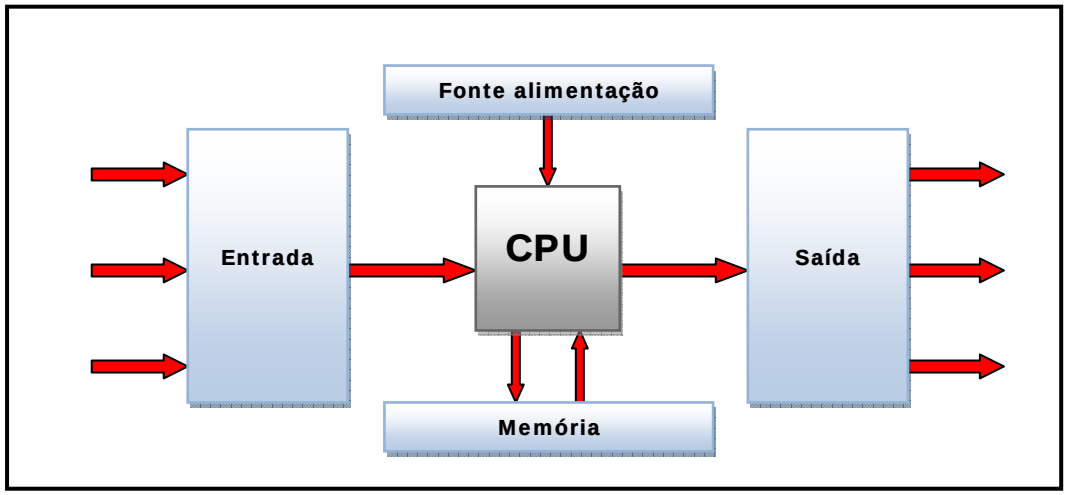


Figura 17 - Estrutura Básica CLP [COSTA,2006]

O funcionamento de um CLP é determinado pelo programa residente, o qual executa uma série de aplicações de forma seqüencial e repetitiva, este ciclo recebe a denominação de tempo de varredura [COSTA,2006]. Durante o processo de varredura, as etapas são realizadas conforme mostra o fluxo na figura 18.

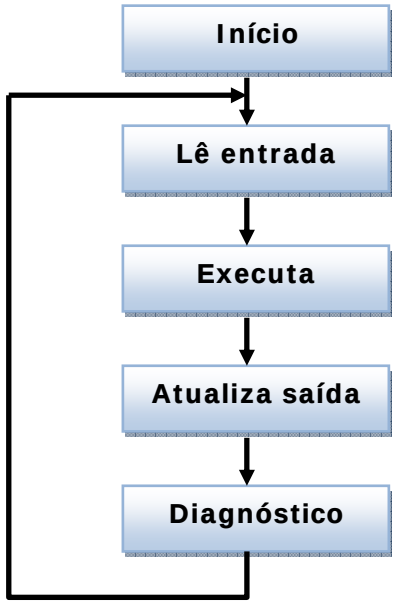


Figura 18 – Fluxo do scan time [COSTA,2006]

A realização do diagnóstico no final de cada ciclo tem por finalidade verificar se ocorreu alguma falha nos componentes internos.

Os dispositivos lógicos programáveis são circuitos integrados que podem ser configurados pelo próprio usuário, tornando assim a integração fácil em diferentes projetos.

Diferentes do CLPs, os DLPs possuem um algoritmo de controle que dispensa a necessidade de ciclos de busca e execução de instruções. Os DLPs podem ser classificados, conforme tabela 01.

Dispositivos	Complexidade	Portas	Tecnologia
Simple Programmable Logic Devic	simples	até 600	CMOS
High Complex Programmable Logic Devices	alta	> 600	CMOS

Tabela 01 – Dispositivos lógicos programáveis

A tecnologia SPLDs *(Simple Programmable Logic Devices*) não será abordada neste trabalho, visto que o foco principal é a tecnologia HCPLDs (*High Complex Programmable Logic Devices*), englobando o FPGAs (*Field Programmable Gate Array*). Os HCPLDs em virtude de sua capacidade lógica, possui as seguintes vantagens:

- Permitir a programação e reprogramação pelo usuário;
- Menor consumo de energia;
- As linguagens permitem testes de depuração;

2.5.2 TECNOLOGIA FPGA

Um FPGA (*Field Programmable Gate Array*), é um HCPLD que contem um grande número de células lógicas configuráveis em um único circuito integrado. Não possuem planos OR e AND [COSTA,2006], pois consistem em uma grande matriz bidimensional constituída por blocos de entrada e saída e chaves de interconexão, a figura 19, mostra o arranjo dos blocos.

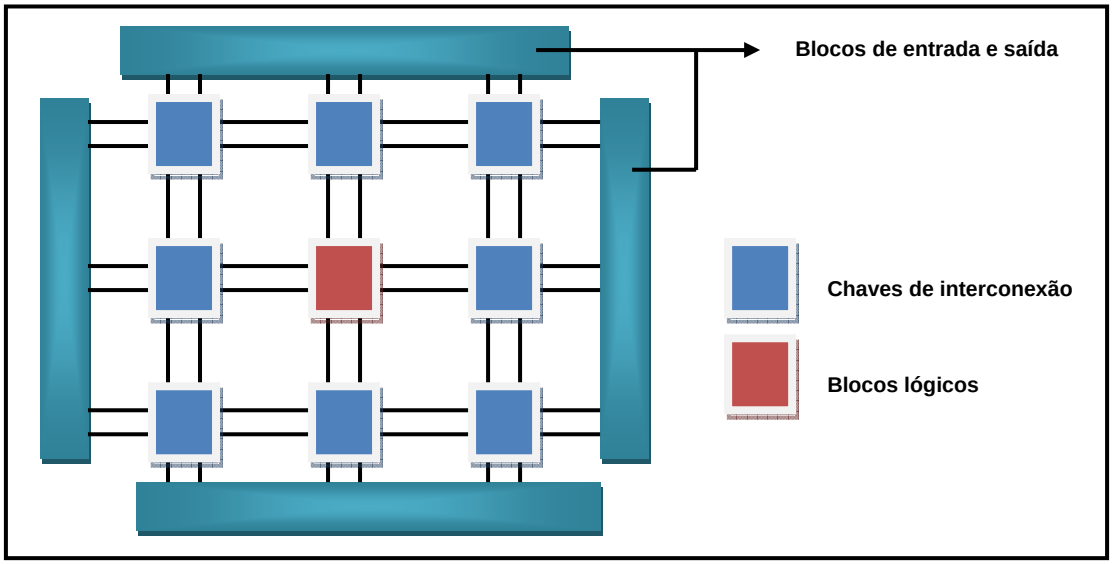


Figura 19 – Arranjo dos blocos de um FPGA [COSTA,2006]

No interior dos blocos são realizadas as implementações lógicas, através de células de armazenamento, o que por sua vez armazena os valores 0 e 1. Uma característica do FPGA é a sua granularidade, ou seja, é a menor unidade configurável, desta forma de acordo com a sua arquitetura podem ser classificados em grande, médio ou pequeno.

Desmembrando a sua arquitetura, podemos encontrar outros elementos visualizados na figura 20.

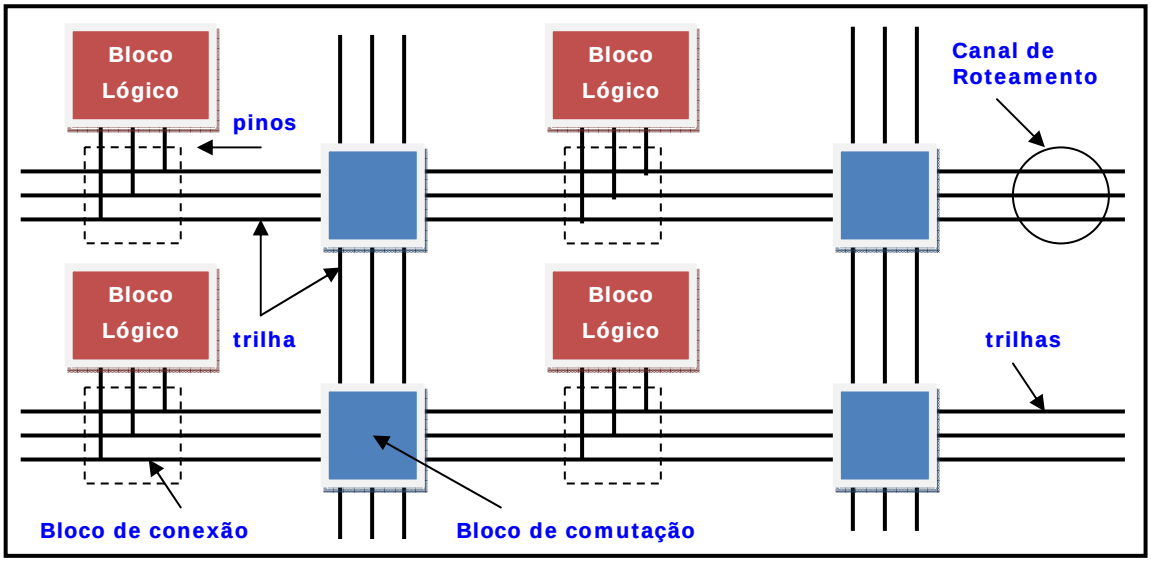


Figura 20 – Arquitetura de um FPGA [COSTA,2006]

Um FPGA deve ser analisado como uma seqüência de blocos [BARROS,2006], um bloco representa a implementação da lógica ou pedaço da lógica total, portanto, é necessário que novos blocos sejam interligados de tal forma a finalizar a implementação. Essa interligação é denominada de roteamento, mas tem que ser na medida para que os caminhos sejam suficientes, assim não gerando problemas de espaço.

Figura 21 - Altera MAX7000 CPLD [UNIPG,2007]

Outro bloco é responsável pela interconexão dos blocos de lógica e por último o controle da entrada e saída de dados. A figura 21 mostra um dispositivo do fabricante Altera, modelo MAX7000 CPLD.

O FPGA representa uma boa alternativa para o desenvolvimento de sistemas embarcados, pois possui uma boa performance se comparados com sistema ASIC (*Application Specific Integrated Circuit*) [BONATO, 2004].

2.5.3 HDL (HARDWARE DESCRIPTION LANGUAGE)

São Linguagens de descrição de hardware utilizadas para o desenvolvimento de dispositivos de hardware, tais como: FPGA, ASIC, etc. Entre as diferentes linguagens, podemos destacar:

ABEL – ***A****dvanced* ***B****oolean* ***E****quation* ***L****anguage*, destina a programação de dispositivos SPLD.

VHDL – ***VH****SIC* ***D****escription* ***L****anguage (****V****ery* ***H****igh* ***S****peed* ***I****ntegrated* ***C****ircuit Hardware Descripton Language*), destinadas a programação mais complexa como os FPGAs.

VERILOG – Permite a implementação de projetos analógicos, digitais ou até mesmo híbridos, caracterizando em uma linguagem modular de programação, podendo ser empregada em FPGAs.

2.6 SISTEMAS DISTRIBUÍDOS

Nos primeiros sistemas todo o hardware estava localizado em um único compartimento [DAVIS,1991]. Hoje, devido à grande demanda de informações e a necessidade de maiores velocidades de processamento, hardware e software foram distribuídos, tornando-se uma coleção de processadores que comunicam entre si por meio de redes de comunicação, atendendo um grande número de usuários que outrora eram concorrentes de um único computador. Ganhou-se desta forma a possibilidade das tarefas serem executadas separadamente em um ou mais computadores, cuja troca de processador pode ocorrer gerando um equilíbrio de tarefas executadas [SILBERSCHATZ,2000].

A criação de um sistema distribuído apresenta quatro vantagens em sua constituição [SILBERSCHATZ,2000]:

- Compartilhamento de recursos: fornecem compartilhamento de arquivos, dispositivos de hardware, processamento de informações e outras operações.
- Velocidade: a carga de processamento é distribuída para sistemas menos carregados.
- Confiabilidade: se um dos sistemas falharem o outro assume automaticamente.
- Comunicação: a troca de informações pode ser realizada em grandes distâncias.

Os sistemas distribuídos apresentam alguns mecanismos que tornam a comunicação eficaz.

- *Sockets*: a comunicação é estabelecida usando um par de *sockets*, um para cada processo, definido como uma das extremidades do canal de comunicação [TANENBAUM,2003], em uma arquitetura cliente/servidor. Neste caso, cria-se um estado de espera em que uma porta específica verifica e aceita o pedido do cliente

estabelecendo a comunicação ou não, conforme podemos verificar na figura 22.

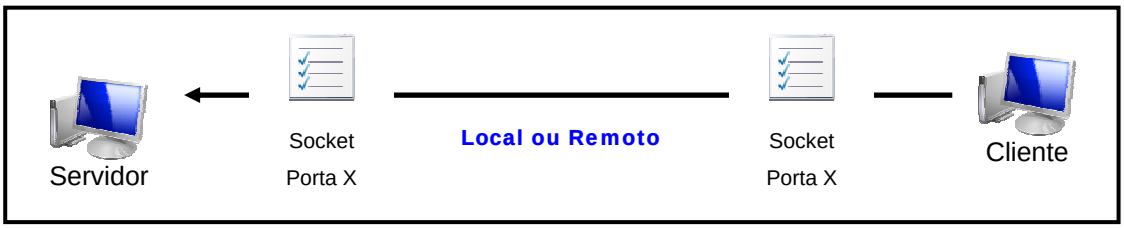


Figura 22 – *Socket*

- **RPC** – *Remote Procedure Call*: permite que uma *thread* chame um determinado procedimento ou função em outro computador conectado a uma rede, sua estrutura possibilita a execução de um procedimento remoto semelhante ao local.
- **RMI** – *Remote Method Invocation*: possibilita que *threads* acionem métodos de um determinado objeto remoto, os quais residem em uma JVM (*Java Virtual Machine)*, diferente da qual a *thread* está sendo executada.
- **CORBA** – *Common Object Request Broker Architeture*: é uma aplicação semelhante ao RMI, mas possibilita a execução de outro código, permitindo a comunicação de servidores heterogêneos.

A arquitetura de um sistema distribuído é representada pela especificação de seus elementos, cuja arquitetura de software pode ser representada em quatro partes [COULOURIS,2001]: aplicação do serviços, *middleware*, sistema operacional, computador e rede. Na figura 23, podemos analisar esta representação.

Figura 23 – Arquitetura de software de um sistema distribuído [COULOURIS,2001]

A plataforma identifica o tipo de hardware e o sistema operacional a ser utilizado, tais como: Intel X86/Windows, Intel X86/Linux, PowerPc/McOS entre outros. Na parte intermediária, o *middleware* representa o modelo de aplicação para os programadores, como por exemplo, o CORBA e o RMI, e por fim o aplicativo ou serviço que realizará o uso da rede.

Um sistema distribuído além de estabelecer e gerenciar a comunicação deve fornecer mecanismos para o gerenciamento de processos e tratar falhas que por ventura venham ocorrer.

É importante ressaltar que as aplicações em rede de forma colaborativa para Realidade Virtual devem levar em conta alguns fatores que possam criar problemas no processo de interatividade [TANENBAUM,2003], tais como:

- Latência cliente/servidor.
- Latência da rede (física).
- Tipo de estratégia de colaboração.
- Renderização.
- Interação entre os mecanismos.

2.6.1 - IP (*INTERNET PROTOCOL*)

O IP (*Internet Protocol*), de forma mais ampla pode ser considerado como um número representado por um conjunto de 04 octetos, conforme figura 24, os quais identificam um computador (ou máquina) em uma rede pública ou privada.

Representação: 200.18.243.37

Decimal	**200**	**18**	**243**	**37**
Hexadecimal	**110010000**	**00010010**	**11110011**	**00100101**

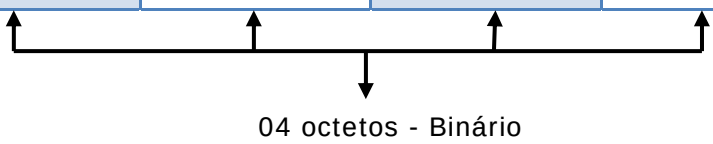


Figura 24 – Representação do número binário

Dividido em diferentes classes, o IP poderá identificar a rede e a máquina desta rede, entre as classes existem IPs reservados, dos quais serão destacados:

- **Endereço 127.0.0.0/8 (*localhost*)** – refere-se à indicação do local de onde está sendo executado o sistema, ou seja, a máquina com que se está trabalhando, considera-se um dispositivo de *loopback*, para que aplicações possam trocar informações consigo mesma.
- **Endereço 192.168.0.0** – utilizado para realizar mapeamento de rede, constituindo uma intranet.

A figura 25, mostra um rede local mapeada através do protocolo IP e uma máquina executando um loopback.

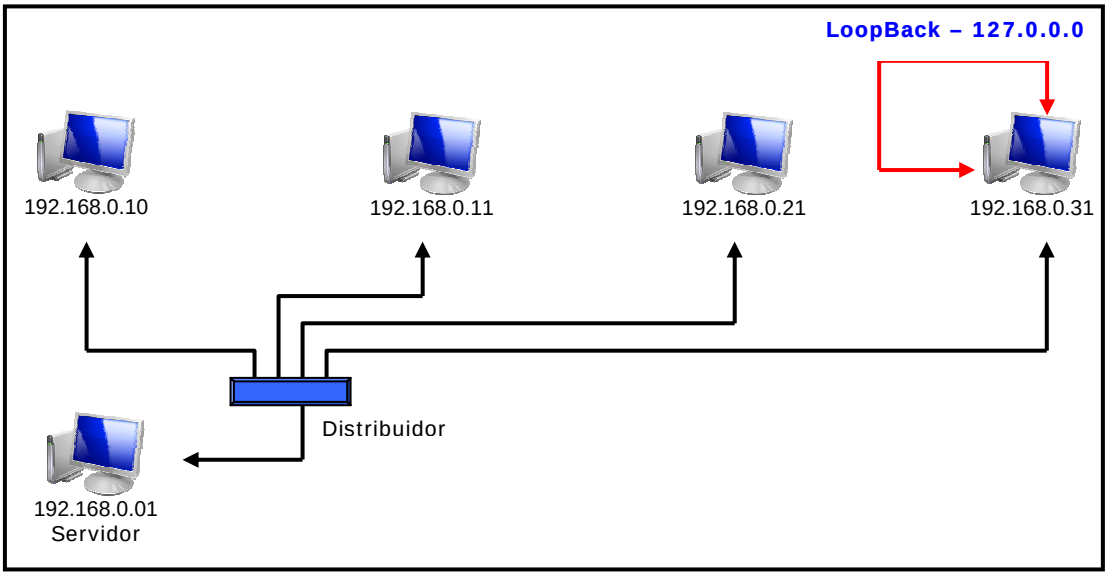


Figura 25 – Rede mapeada sobre o protocolo IP

Visto este recurso, podemos realizar a implementação do módulo denominado cliente/servidor, aplicando-se a qualquer programa que ofereça um serviço que possa ser alcançado através de uma determinada rede [COMER, 1998], uma requisição de serviço é realizada e o servidor por sua vez o executa e retorna o resultado ao seu solicitante, caso este seja necessário.

Além disso, podemos realizar a implementação de serviços de *sockets*, podendo ler ou escrever informações no formato cliente/servidor.

2.6.2 *SOCKET*

Os *sockets* comportam-se da mesma forma que arquivos, desta forma podemos realizar as operações de *ler* e *escrever.* Desta forma, uma aplicação cria uma conexão com o destino, podendo usar a chamada *escrever* para enviar um fluxo de dados, o aplicativo na outra ponta usa *ler* para recebê-los.

Em linhas mais específicas ***write(mensagem)*** escreve-se a mensagem a ser enviada e, do outro lado, ***read(mensagem)*** realiza a leitura da mensagem; além disso podemos especificar ou não o endereço no *socket*, na figura 26, apresenta a sistema de envio e recebimento de um *socket*.

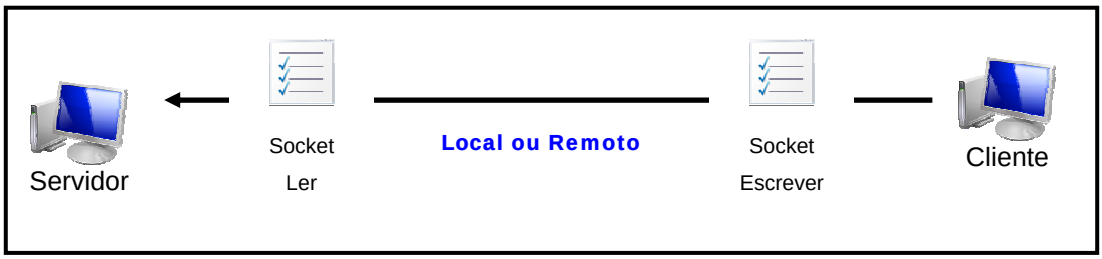


Figura 26 – Leitura e escrita de um socket

3. ARQUITETURA PARA INTEGRAÇÃO DE INTERFACES MULTIMODAIS

Os tópicos a seguir apresentam considerações e propostas iniciais para o desenvolvimento da arquitetura de um sistema de RA, possuindo interfaces multimodais implementadas em sistemas baseados em hardware e software. A arquitetura proposta terá como base o ARToolKit para sistema de RA, a qual será denominado de ARToolKit-Multimodal.

3.1 ARQUITETURA GENÉRICA – MODELO CONCEITUAL

Conforme citado anteriormente, o trabalho consiste na criação de um sistema de interação para RA que permita a utilização de interfaces multimodais baseadas em dados, voz, e outras formas, permitindo sua configuração e adaptação em diferentes sistemas, conforme suas necessidades. Neste sentido, para efeito de discussão será considerada nos demais subitens a implementação de diferentes interfaces.

A estrutura genérica da arquitetura está dividida em dois módulos, os quais foram denominados de módulo cliente e módulo servidor. Sua implementação tem como objetivo realizar o reconhecimento dos sinais enviados pelas interfaces utilizadas pelo usuário, através de comando de voz, joystick, mouse ou outro dispositivo de hardware implementado para esta finalidade. A figura 27 apresenta um diagrama dos principais elementos que compõem a estrutura da arquitetura proposta neste trabalho. A seguir, são descritos cada um desses elementos.

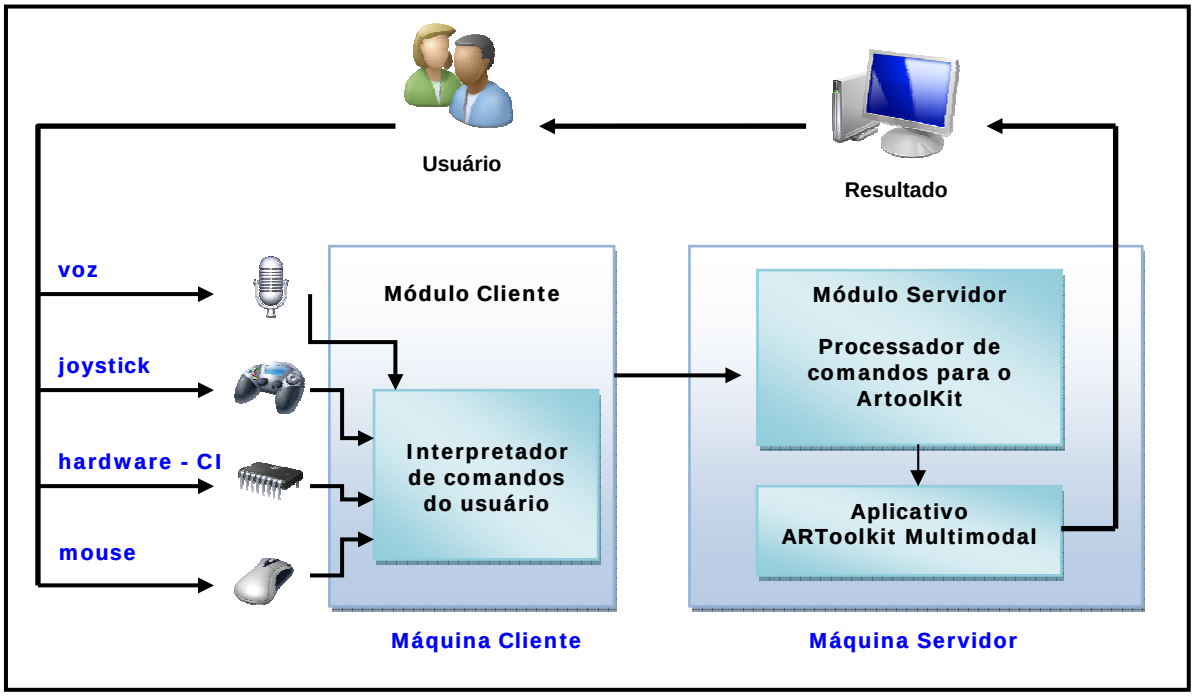


Figura 27 – Interface baseada em Software

3.1.1 MÓDULO CLIENTE

Consiste em reconhecer os sinais dos diferentes dispositivos, tratando os sinais e enviando a mensagem ao módulo servidor. Esta mensagem por sua vez poderá ser emitida para a própria máquina cliente caso o usuário não esteja em rede, ou através da rede usando *sockets*. Cada dispositivo de interação possui características particulares que deverão ser observadas:

- **Voz:** Todo comando de voz será tratado por um reconhecedor de voz já existente, o qual apenas interpretará os comandos fornecidos por uma lista de palavras (comandos), já pré-definida no módulo cliente.
- **Joystick:** Os comandos provenientes do joystick serão capturados através da saída USB (*Universal Serial Bus*). Cada alavanca do joystick, denominada de *stick* irá possuir representações distintas no envio da mensagem, desta forma

as coordenadas X e Y serão representadas pelo *stick* esquerdo e a coordenada Z, representando o espaço 3D, será indicado pelo *stick* direito.

- **Hardware – CI:** Um equipamento de hardware desenvolvido especificamente para testar a aplicação multimodal, utilizará a porta paralela para envio de dados ao módulo cliente.
- **Mouse:** A utilização do mouse será intuitiva na parte da aplicação, habilitando ou desabilitando os componentes de interação bem como participando da escolha de elementos de interação.

As opções de implementação mencionadas anteriormente serão detalhadas durante o desenvolvimento deste trabalho, mostrando suas características em particular, implementações e testes realizados. A figura 28 mostra as interfaces de cada dispositivo em relação ao módulo cliente.

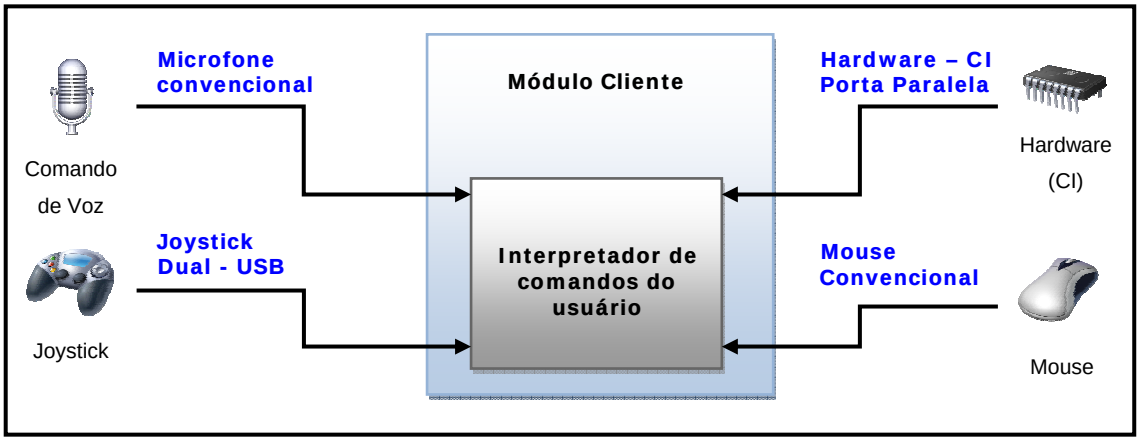


Figura 28 – Representação do Módulo Cliente

3.1.2 MÓDULO SERVIDOR

Consiste em receber e tratar as mensagens enviadas pelo módulo cliente, gerando comandos padronizados ao aplicativo ARToolKit Multimodal. As implementações realizadas neste trabalho permitem que outros dispositivos de interação possam ser incorporados à aplicação, desde que sigam as especificações das mensagens enviadas ao aplicativo ARToolKit Multimodal.

O módulo servidor deverá ser executado em uma máquina devidamente instalada e configurada em rede, cujo endereço deverá ser informado ao módulo cliente para que as mensagens sejam direcionadas, fazendo necessário a instalação do ARToolKit Multimodal. A comunicação entre o módulo cliente e o aplicativo ARToolKit será realizada através de um **arquivo de texto**, gerado na base do aplicativo, denominado de “comando.dat” (figura 29). O processador de comandos realizará a análise das mensagens, resolvendo conflitos entre dois ou mais dispositivos ou mensagens ambíguas que venham a interferir na execução do sistema.

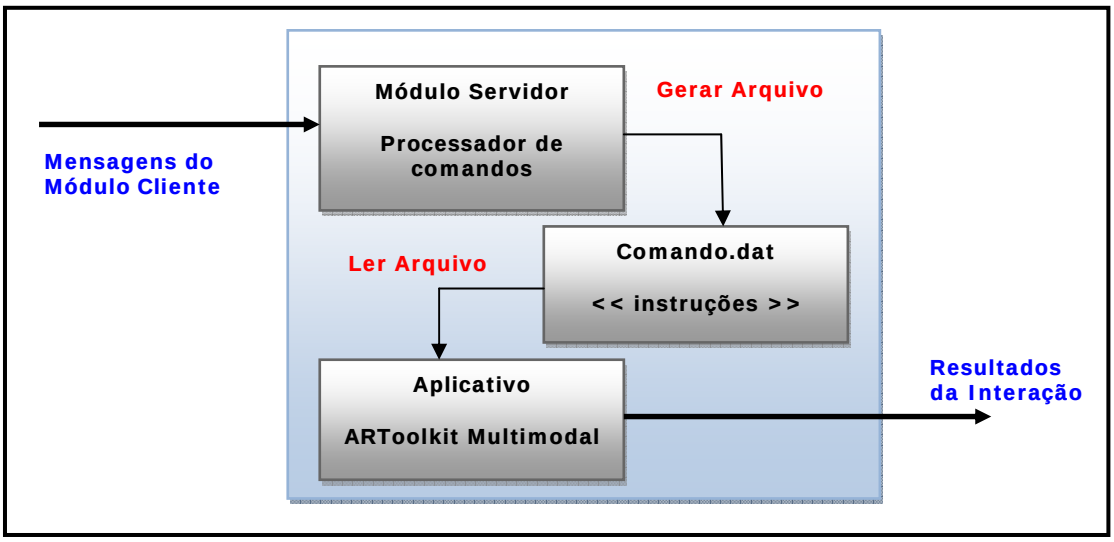


Figura 29 – Interação Módulo Servidor e ARToolKit Multimodal

O conteúdo do arquivo “comando.dat”, possui informações padronizas que irão determinar qual o tipo de interação que será realizada, procedimento este que será detalhado nos próximos capítulos.

A interação entre o módulo cliente e o módulo servidor (figura 30), será realizada através de *sockets*, indiferente se os módulos forem instalados na mesma máquina ou não. Assim, as mensagens serão enviadas como *localhost*, (IP 127.0.0.1) ou em outra instalada em rede, usando o nome desta máquina na rede mapeada sobre IP.

O uso do protocolo IP visa padronizar a arquitetura, independente da configuração de hardware do sistema.

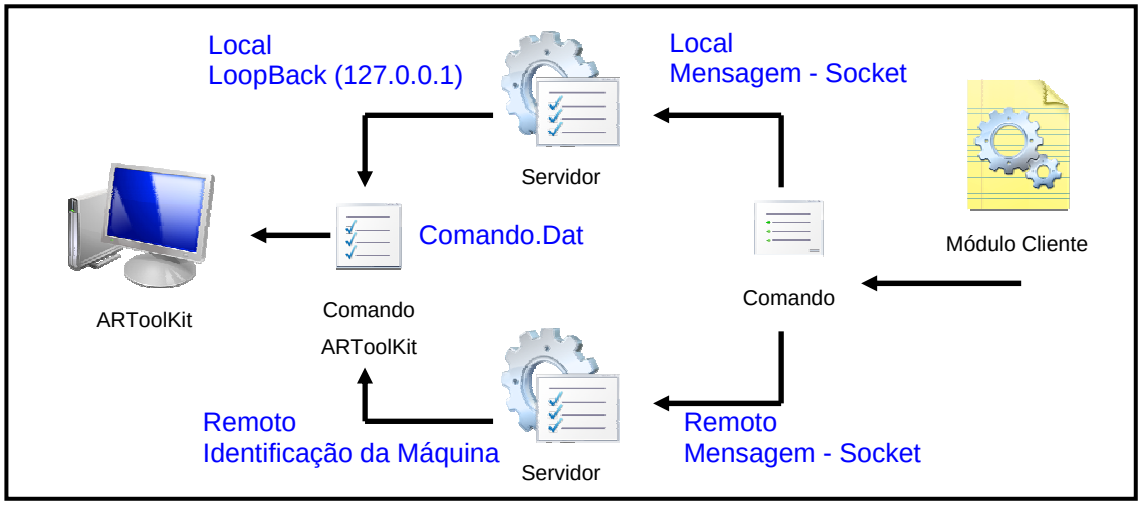


Figura 30 – Módulo de rede e/ou localhost

4. IMPLEMENTAÇÃO DE PROTÓTIPOS UTILIZANDO O ARTOOLKIT

Neste capítulo, será apresentada a descrição detalhada da implementação de alguns protótipos, os quais constituem instâncias da arquitetura proposta, cada um deles exercitando o uso de um tipo particular de dispositivos de interface. O objetivo da implementação destes protótipos é validar a arquitetura proposta, bem como realizar alguns testes antes de efetuar a integração de todos esses dispositivos em um único sistema integrado.

4.1 PROTÓTIPO 01 – INTERFACE DE VOZ BASEADA EM SOFTWARE

O protótipo 01 consiste em uma interface de voz baseada em software que tem como objetivo principal realizar a captura de um comando emitido por voz, através de um receptor (microfone), interpretar a palavra, codificar em comandos e enviar mensagens via *socket* ao ARToolKit, conforme apresentado na figura 27, anteriormente.

4.1.1 – SISTEMA DE SOFTWARE PARA RECONHECIMENTO DE VOZ

Para a arquitetura de software para reconhecimento de voz, utilizou-se como hardware receptor um microfone comum, a base da instalação padrão do Windows XP®, utilizando o dispositivo padrão *SoundMAX Digital Áudio*, com taxa de volume e velocidade de voz no nível normal.

4.1.1.1 – AMBIENTE DE DESENVOLVIMENTO

Por ser um aplicativo da categoria RAD (*Rapid Application Development)*, utilizou-se o VB (Microsoft Visual Basic 6.0®), pela facilidade no desenvolvimento da interface e por possuir o mesmo núcleo de compilação do Visual C++, permitindo o uso de funções adicionais fornecidas por uma API

(*Application Programming Interface)*, geralmente desenvolvidas para programadores em linguagem “C”. A API Win32 é um bom exemplo, a qual consiste de funções, estruturas e mensagens que podem ser acessadas por aplicativos da família Windows.

4.1.1.2 - API Win32

A API Win32 do sistema operacional Windows® está dividida em 04 categorias funcionais:

- Gerenciamento do Windows (User): Todas as entradas e saídas básicas passam por essa camada, incluindo teclado e mouse e todo processamento das mensagens do aplicativo.
- Interface de Dispositivos Gráficos (GDI - *Graphics Device Interface*): Fornece funções para trabalhar com os dispositivos gráficos aceitos no sistema como monitor e impressora.
- Serviços de Sistema (Kernel): Permite trazer recursos do sistema operacional.
- Multimídia (MMSystem): Permite reproduzir áudio como música e vídeo digital

Estas categorias são implementadas em forma de DLLs (*Dinamic Link Libraries*), as quais representam bibliotecas dinâmicas podendo ser chamadas por qualquer linguagem.

4.1.1.3 – Microsoft Agent®

Microsoft Agent® [MICROSOFT,2007], é uma tecnologia da Microsoft® que permite recursos de geração e reconhecimento de voz, representando o núcleo do sistema de reconhecimento de voz. Outros recursos como personagens e vozes é necessário a instalação de pacotes adicionais, como por exemplo, componentes relacionados à língua que são representados por DLL’s que sustentam o diálogo, janela, *tooltips* e textos de bolões, os quais

deverão ser adicionados ao motor principal do agente. Todos os pacotes estão disponíveis a partir de [Microsoft, 2007].

- **Pacote MsAgent** - Permite incorporação dos recursos de voz em aplicações Microsoft®, representado o núcleo da aplicação.
- **Pacote AgtX0416** - Disponibilizadas no formato de bibliotecas - DLL´s, oferecem recursos de geração de fala a partir de strings pré-definidas TTS (*Text-to-Speech*), o pacote especificado é o Português (Brasil).
- **Pacote lhttsptb** - Permite aumentar a potencialidade dos motores de saída (TTS), permitidos somente para aplicações Microsoft® e web pages. Estes motores são desenvolvidos por L&H (Lernout & de Hauspie®), denominados sob a sigla TTS3000, para diferentes idiomas. O pacote especificado é o motor para o português.
- **Pacote spchapi** - O SAPI 4.0, destinado ao Windows XP® faz-se necessário caso seja utilizados motores de entrada utilizando o idioma em inglês.
- **Pacote actcnc** - Este pacote por sua vez potencializa os motores de entrada para reconhecimento de voz, baseado em um único motor no idioma inglês.

4.1.1.4 – Componentes Visual Basic®

Após a instalação do **Pacote spchapi**, tornam-se disponíveis alguns componentes, como é mostrado na tabela 02.

Componente	Objeto	.dll
Microsoft Direct Speech Recognition	directSR	Xlisten.dll
Microsoft Direct Text-to-Speech	directSS	Xvoice.dll
Microsoft Voice Commands	Vcommand	Xcommand.dll
Microsoft Voice Dictation	Vdict	Vdict.dll
Microsoft Voice text	TexttoSpeech	Vtext.dll

Tabela 02 – Componentes

- **Microsoft Voice Commands -** Este componente é responsável pela captura e interpretação dos comandos. Para incluí-lo no projeto de software, basta acionar o Menu Project → Components (CTRL+T), disponibilizando a janela de componentes, após feita a seleção, confirme o processo para que o componente fique disponível, figura 31.

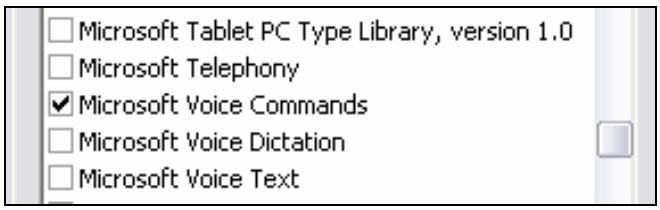


Figura 31 – Inclusão de Componente – Voice Commands

Na seqüência, podemos observar a inclusão do componente na barra de componentes, figura 32.

Figura 32 – Disposição do controle

- **Winsock** – Este componente é responsável pela implementação de *socket* na aplicação, da mesma forma que o componente anterior o mesmo deverá ser incluído no projeto, conforme mostra a figura 33.

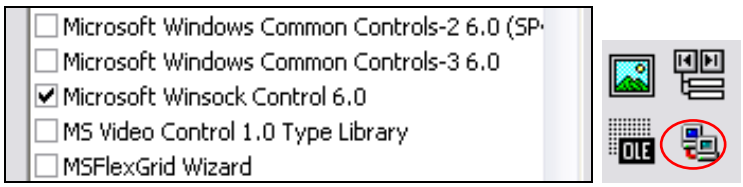


Figura 33 – Componente Winsock

4.1.1.5 – Widgets

Caracteriza-se como Widgets qualquer interface com o usuário, podendo ser representada por uma janela, caixa com botões, menu de opção, rótulos, caixa de texto ou qualquer outro componente, representando trechos de programação, popularmente encontrados em web sites.

4.1.2 ACOPLAMENTO DA INTERFACE

O acoplamento da interface de reconhecimento de voz envolve os dois módulos: servidor e cliente. A aplicação tem como finalidade reconhecer o comando de voz, interpretar o comando recebido e enviá-lo ao servidor, que por sua vez processa as mensagens recebidas e envia a mensagem ao ARToolkit.

4.1.2.1 MÓDULO SERVIDOR

O módulo servidor, quando ativado, tem como finalidade monitorar a porta de entrada. Neste projeto, a porta 1202 foi definida como padrão. Desta forma, mensagem enviada pelo aplicativo é recebida e interpretada em uma instrução padronizada para o aplicativo ARToolkit, conforme a figura 34.

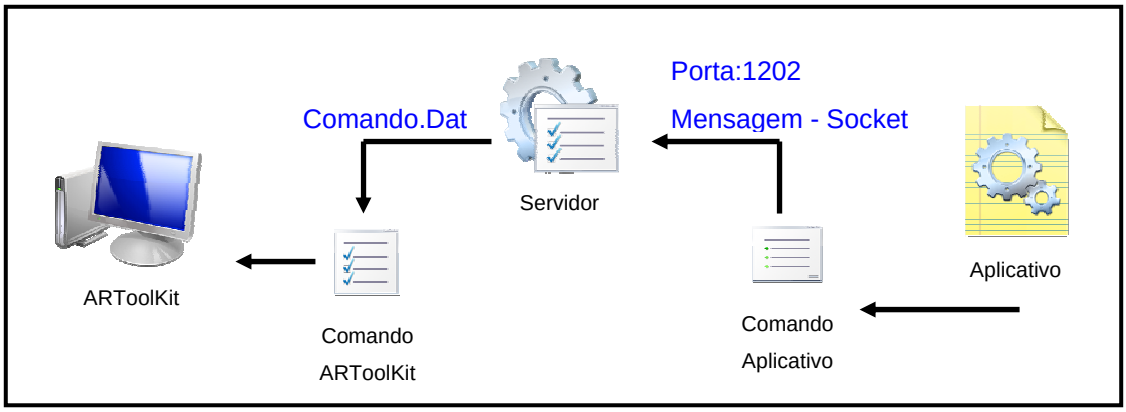


Figura 34 – Funcionamento do Módulo Servidor

O módulo servidor gera um arquivo denominado comando.dat, que será gravado dentro do diretório “\bin\” do aplicativo ARToolKit, que por sua vez será lido através de uma implementação realizada no arquivo ARVideo.cpp.

Conforme mostrada na figura 35, podemos verificar a funcionalidade do módulo servidor, o qual identifica o nome da máquina servidor (1), data da execução (2), hora inicial da execução (3), tempo percorrido da execução (4) e instruções (5).

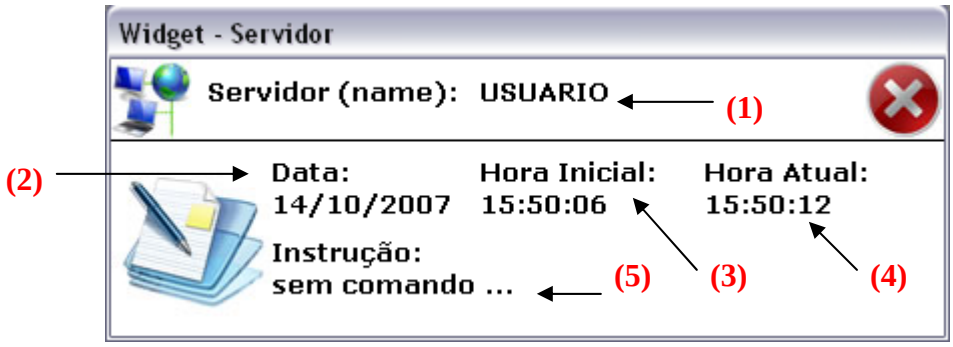


Figura 35 – Widget - Servidor

As instruções apresentadas pelo servidor permitirão ao usuário verificar a procedência da mensagem, ou seja, qual foi o meio utilizado de interação. Estas instruções se aplicam no nível de desenvolvimento da aplicação e não durante o uso da aplicação de RA.

4.1.2.2 – MÓDULO CLIENTE

No módulo cliente, o usuário poderá escolher diferentes meios de interações, mas neste protótipo somente a interação via comando de voz será abordada, as demais interações serão tratadas posteriormente.

A função principal do módulo cliente é reconhecer o comando de voz realizado pelo cliente e enviar esta mensagem ao módulo servidor, através de um *socket*, assim como no módulo servidor a porta 1202 é definida como padrão. A figura 36 mostra o nível de interatividade através do comando de voz.

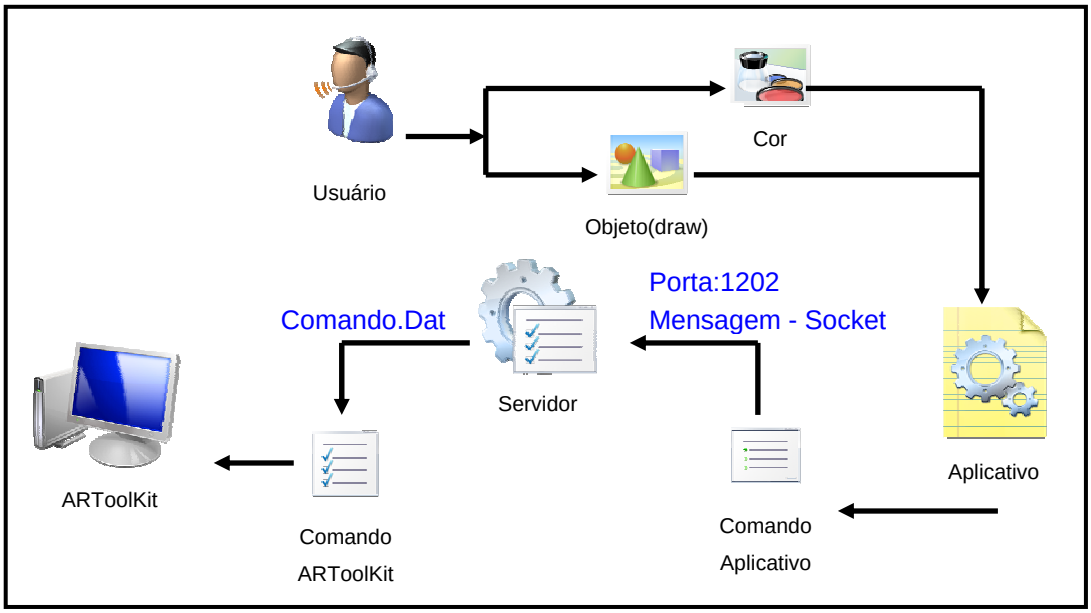


Figura 36 – Funcionamento do Módulo Cliente

A interação poderá ocorrer através da escolha de um objeto a ser visualizado pelo aplicativo ou sua cor. Para cada objeto, foi associado uma determinada palavra chave, a lista abaixo representa as palavras e quais os seus respectivos objetos:

- **CUBE** – Um cubo sólido.
- **LINE** – Um círculo, apenas no formato de linhas.
- **TRI** – Um triângulo sólido.
- **FORM** – Uma chaleira, apenas no formato de linhas.

Em relação às cores, foram utilizadas cores que compõe o sistema RGB (**Red**, **Green** e **Blue**), desta forma, para representar o sistema às palavras chaves possuem os mesmos nomes.

Conforme observado na figura 37, podemos verificar a funcionalidade do módulo cliente, onde a disposição de botões de controle no sistema liga/desliga permite ativar os módulos de interação com o usuário. No módulo cliente, encontramos a conexão com o servidor de rede (1), a interação de fala (2) e a interação de voz (3), as demais interações serão vistas posteriormente.

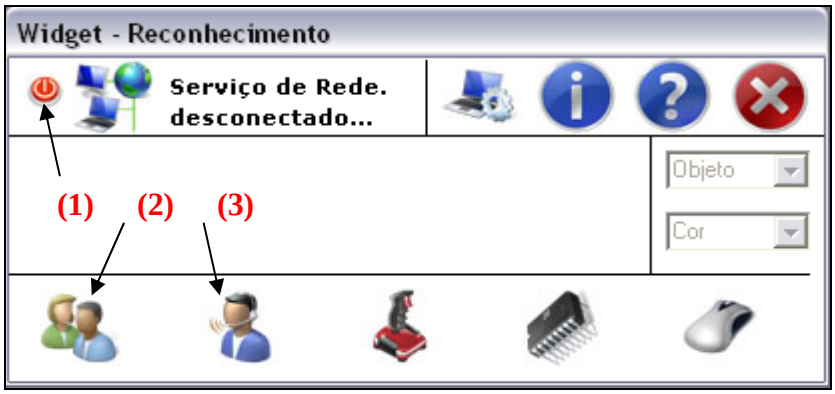


Figura 37 – Widget – Reconhecimento (Cliente)

4.1.3 INTERAÇÃO COM ARTOOLKIT

A interação com o ARToolKit, conforme mencionado anteriormente, será realizada através da leitura de um documento chamado comando.dat, gravado dentro do diretório “\bin” do aplicativo. Este arquivo trata-se de um arquivo de texto que irá conter informações sobre qual procedimento deverá ser executado pelo ARVideo.EXE.

A função de leitura foi implementada no arquivo ARVideo.cpp, na função *draw().* Desta forma o arquivo comando.dat é aberto, os caracteres de controles são lidos e na seqüência é realizada uma cadeia de comparações para definir qual procedimento a aplicação deverá realizar, o trecho de código abaixo ilustra a estrutura dos comandos para leitura e interpretação de comandos.

```
char caracter;
if((arquivo = fopen("comando.dat","r")) != NULL){
do  {
     caracter = getc(arquivo);
    // seqüência de  instruções para interação com o aplicativo
     }
} while(caracter != EOF);
fclose(arquivo);
```

Para execução deste protótipo, foi definido dois tipos de interação, desta forma, o usuário poderá escolher entre diferentes objetos apresentados

pelo módulo cliente e as cores que estes objetos deverão assumir no momento da apresentação pelo aplicativo ARToolKit.

4.1.4 EXECUÇÃO E TESTES

Ativado o aplicativo ARVideo, os módulos cliente e servidor o usuário deverá conectar-se ao serviço de rede (durante uma execução local ou remota) e “ligar” a interação desejada, neste caso, de voz. O módulo cliente deverá executar somente as palavras pré-definidas, outras palavras serão desprezadas.

Na figura 38, podemos verificar os componentes acionados e a relação entre o aplicativo, o módulo servidor e o módulo cliente. No módulo servidor, podemos verificar o nome da máquina servidor (1), as informações de status do servidor (2) e em instrução o reconhecimento do comando (3). No módulo cliente, o botão de serviço de rede conectado (4), e botão de reconhecimento ativo (5), disponibilizando o menu de opções ao usuário (6), e no fundo o aplicativo ARVideo em execução (7).

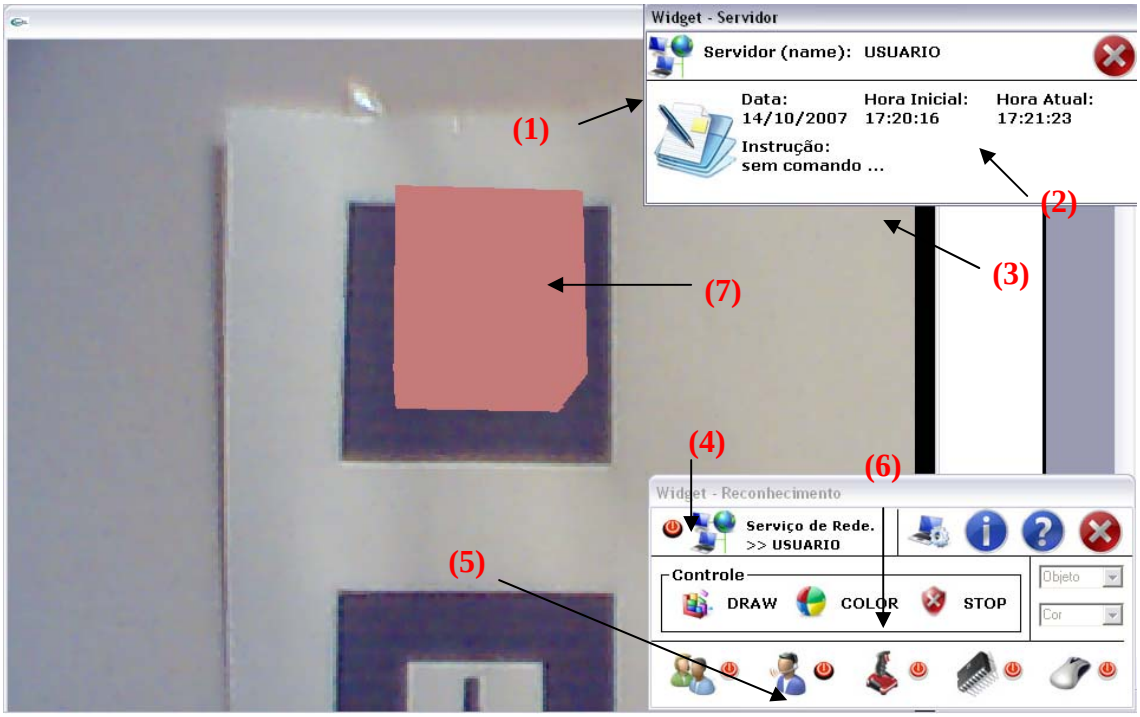


Figura 38 – Interação com Voz

Os *widgets* visualizados na figura 38 estão sobre a aplicação apenas para visualização, os mesmos estarão sendo executados atrás da aplicação principal.

No exemplo da figura 39, através de um comando de voz, foi acionado o comando *DRAW* no módulo cliente, fornecendo as figuras pré-definidas e suas respectivas palavras chaves (1), no módulo servidor podemos verificar o reconhecimento do comando através da instrução (2), e por fim o ARVideo (3), respondendo a solicitação. As ações relacionadas ao comando de voz dizem respeito à troca dos elementos visualizados sobre o marcador.

O marcador utilizado na aplicação não será modificado no momento das interações, desta forma, os novos objetos renderizados ou modificados utilizarão o mesmo marcador.

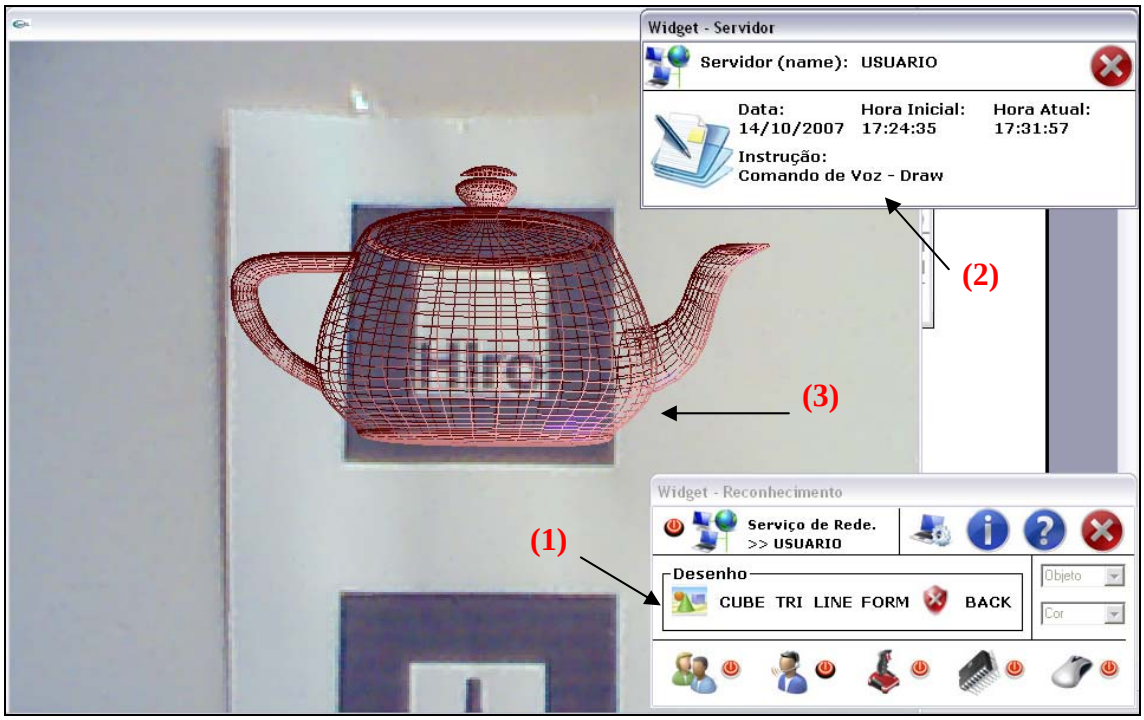


Figura 39 – Interação com Voz - Comando

4.2 – PROTÓTIPO 02 – INTERFACE DE TOQUE BASEADA EM HARDWARE

O protótipo 02 consiste em um dispositivo de hardware que tem como objetivo principal *simular a captura de gestos*, neste caso o toque em um receptor de contato, que enviará uma mensagem ao aplicativo através da porta paralela, conforme apresentado anteriormente na figura 27.

Tendo em vista os estudos preliminares apontarem como solução para criação de um dispositivo de hardware, a utilização de FPGA, o custo para aquisição do chip, kit de leitura e gravação e software de simulação direcionou o trabalho a buscar medidas alternativas. Desta forma, buscaram-se dispositivos que poderiam representar a criação de um dispositivo de hardware, oferecendo o mesmo resultado final de uma interface implementada com FPGAs, porém a um custo mais baixo.

4.2.1 – DISPOSITIVO DE HARDWARE PARA IMPLEMENTAÇÃO DA INTERFACE

A implementação do dispositivo de hardware está baseada em componentes simples e de baixo custo operacional, tais como: CI, chave de pressão e diodos. A simplicidade do dispositivo permite a interação com o aplicativo através da porta paralela, que receberá a mensagem e posteriormente através do módulo cliente será enviada ao módulo servidor via socket.

4.2.1.1 – CI – CIRCUITO INTEGRADO

Para a elaboração do protótipo, foi utilizado o CI SN74HC541N da Texas Instrumentos® [Datasheet,2007], o qual possui entrada e saídas em lados opostos e as entradas por sua vez são controladas por dois estados, desta forma permitindo a inserção de componentes de contato. Além disso, sua base de alimentação está de 2V até 6V, valores que podem ser facilmente

obtidos pela porta paralela (5V). O diagrama representado pela figura 40 mostra a estrutura do CI.

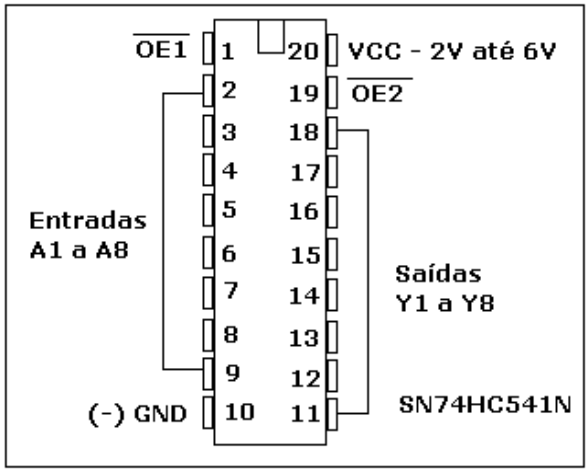


Figura 40 – CI - SN74HC541N

Como mostra a figura 41, podemos verificar a estrutura lógica do CI, onde uma porta AND com inversor de sinal em ambas as entradas e uma porta NOT. Os sinais são representados por **L** (*low*) e **H** (*high*), desta forma teremos o sinal de retorno nos canais Y1 até Y8 (ver figura 41), quando as entradas 01 e 19, receberam cargas baixas (L) e os canais 02 até 09, receberem cargas altas.

Entrada			Saída
01	19	2	18
L	L	L	L
L	**L**	**H**	**H**
H	X	X	Z
X	H	X	Z

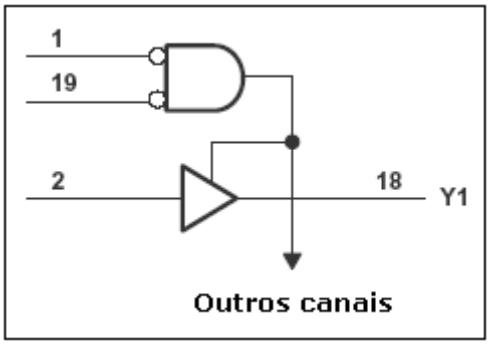


Figura 41 – Estrutura Lógica do CI.

Para agilizar o processo de construção do protótipo de hardware a estrutura foi montada sobre uma *protoboard,* modelo pront-o-labor- PL 551M, alimentada diretamente pela saída paralela em 5V.

No exemplo da figura 42, será apresentado um diagrama do sistema implementado na *protoboard*.

- (cinco) capacitores de 10nf
- (cinco) transistores de 470Ω
- (cinco) chaves de pressão
- (um) – CI - SN74HC541N

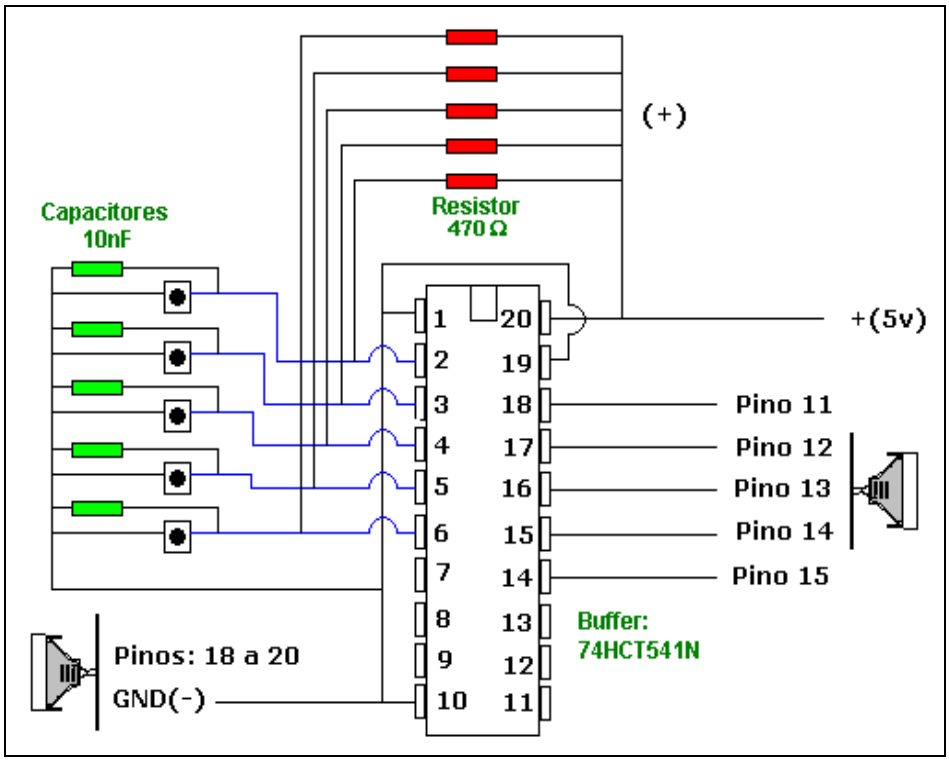


Figura 42 – Esquema do Circuito

No protótipo 02, foram utilizadas apenas três chaves de pressão para interação com a aplicação.

4.2.1.2 – SAÍDA PARALELA

Como interface de entrada no equipamento foi utilizada a saída paralela do micro, a LTP1 (*Line Print Terminal1*), representada pelos

endereços em decimais 888 para envio e 889 para leitura, os quais serão utilizados posteriormente pelo software para envio e captura de sinais.

O protótipo utiliza os canais de entrada e saída da LTP1, para transmitir e receber dados, desta forma os pinos 02 até 09, representados por D0 a D7 respectivamente permitem fornecer a energia necessária para o funcionamento da *protoboard* e os pinos 10,11,12, 13 e 15 receberão os sinais de retorno e por fim os pinos 18 a 25, identificados como GND negativo (*Ground*).

Entende-se como nível lógico 1, quando a tensão elétrica no pino está acima de 3.1V e até 5V. A figura 43 representa os pinos utilizados na elaboração do projeto.

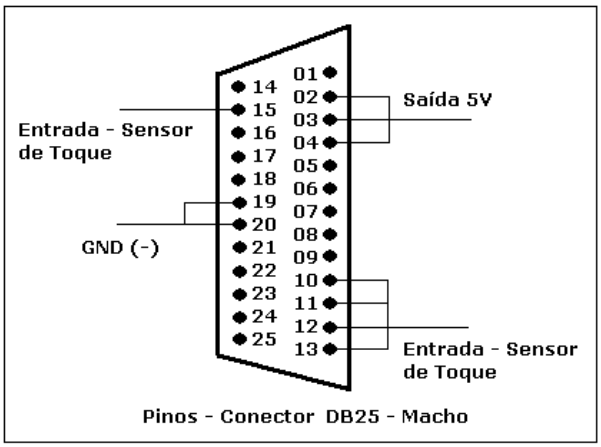


Figura 43 – Pinos do conector

O conector utilizado é o DB25M que é um conector envolvido por um envoltório no formato de “D”, cuja letra é usada como prefixo em seguida a identificação da quantidade de pinos pelo envoltório (A=15 pinos, B=25 pinos, C=37 pinos, D=50 pinos, E=9 pinos), na sequência a quantidade real de pinos e por último a identificação “M” (macho) ou “F” (fêmea).

4.2.1.3 – UTILIZANDO A PORTA PARALELA COM O WINDOWS XP®

O Windows XP, NT e 2000 diferentes das versões anteriores, implementaram uma política de segurança que impedem que os usuários

trabalhem diretamente com a porta paralela. Desta forma, para que o dispositivo funcione corretamente a sistema operacional deverá estar com a porta paralela desbloqueada.

Para realizar este processo, podemos utilizar de diferentes programas disponibilizados na Internet para realizar este desbloqueio, neste projeto uma das soluções abordadas foi a utilização de um software chamado UserPort, desenvolvido por Tomas Franzon, seu aplicativo é freeware e seu download poderá ser realizado em diferentes sites [ALCÂNTARA,2007].

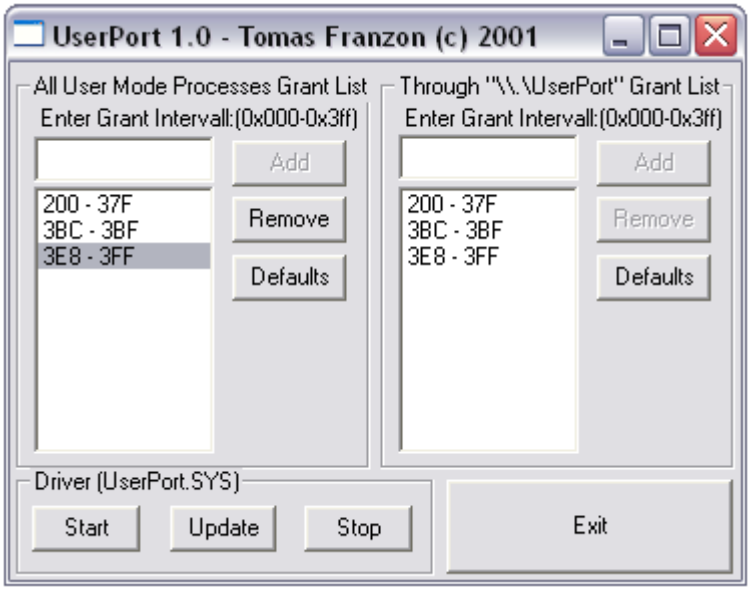


Figura 44 – UserPort de Tomas Franzon© 2001

Conforme figura 44, podemos verificar que a utilização do software é bem simples, basta indicar o número da porta a ser desbloqueada e clicar em Start, o procedimento inverso deverá ser realizado para interromper o serviço, escolha a porta e clique em Stop. Acompanha o software um arquivo denominado userport.sys, o qual deverá ser gravado no diretório “c:/windows/system32/drivers/” no caso do Windows XP® ou no diretório “c:/winnt/system32/drivers” no caso do Windows NT®.

4.2.2 INTERFACE DE GESTOS BASEADA EM HARDWARE

A interface de hardware visa simular a captura de gestos, no caso representado por uma interface háptica, referente ao toque. A princípio dois estados foram implementandos no protótipo para testes, a opção de escolha de objetos e mudança de cores.

No exemplo da figura 45, podemos perceber a forma de interação do usuário com o mecanismo de hardware.

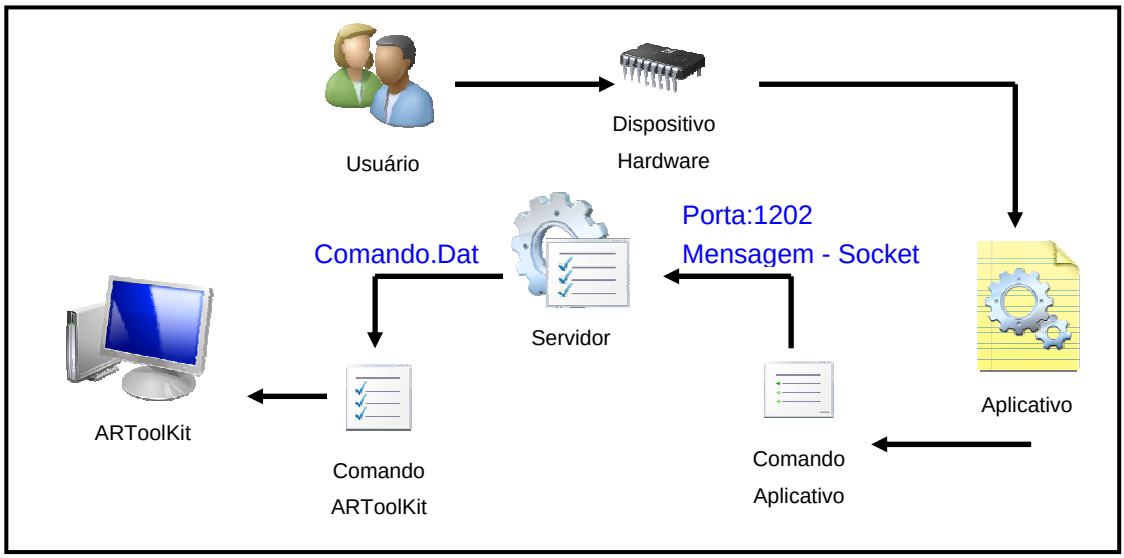


Figura 45 – Interação do Dispositivo de Hardware

Os gestos (toques) serão reconhecidos pelos toques nas chaves de pressão, figura 46, cada toque representa um estado de interação com o aplicativo. Em uma implementação mais realista, isso seria substituído por um sistema que realmente interpretasse gestos, como aquele descrito em [BONATO, 2006].

Figura 46 – Chave de Pressão

De acordo com o esquema do circuito, visualizado anteriormente na figura 42, o mesmo agora representado com os componentes eletrônicos exemplificado na figura 47, nota-se a organização do dispositivo de hardware.

(1) – GND(-), conexão nos pinos 19 e 20 da porta paralela;
(2) – Capacitores;
(3) – Botões de pressão, os quais estão ligados nas saídas 11,12 e 13 da porta paralela;
(4) – Cabo paralelo, com extremidade ligada ao conector BD25M.
(5) – Alimentação (5V+), realizada pelos pinos 2, 3 e 4. A utilização de mais de uma saída é a garantia que em nenhum momento o circuito fique sem alimentação.
(6) – CI - SN74HC541N
(7) - Transistor de 470Ω
(8) Protoboard – Pront -O-Labor PL 551M

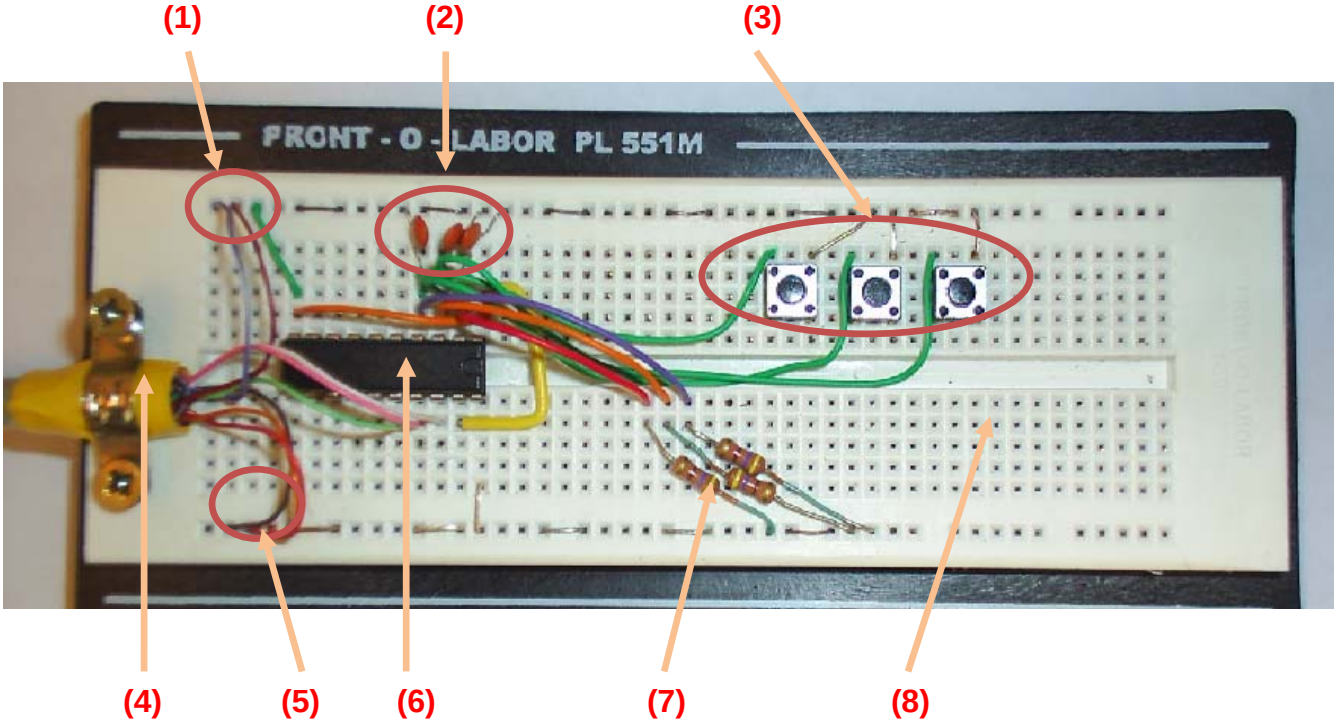


Figura 47 – Implementação do Protótipo 02

Podemos verificar nas laterais da *protoboard* algumas ligações que se fizeram necessárias para que o sinal (+) positivo e (-) negativo percorresse toda a estrutura, incluindo a alimentação no CI, conforme visto na figura 48.

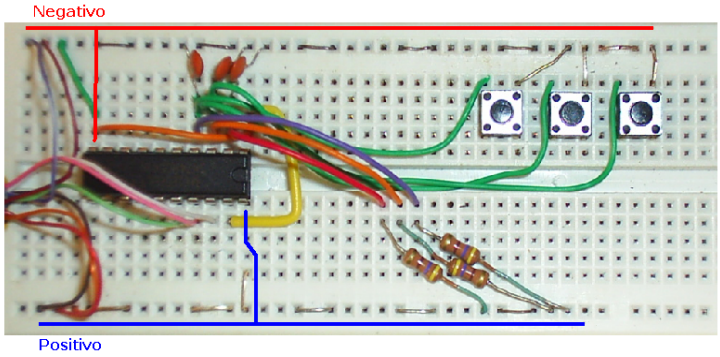


Figura 48 – Alimentação da *protoboard*

4.2.3 INTERAÇÃO COM ARTOOLKIT

Da mesma forma que a interface baseada em software, a interface baseada em hardware irá enviar através de porta paralela um número binário, que convertido em decimal irá representar um botão. O módulo cliente trata os códigos e os enviam para o módulo servidor via *socket*, que os padroniza e envia ao aplicativo.

4.2.4 INTERAÇÃO EM EXECUÇÃO

Com as portas paralelas devidamente liberadas e o dispositivo de hardware devidamente conectado a interação é realizada, seguindo os mesmos princípios da interface de voz.

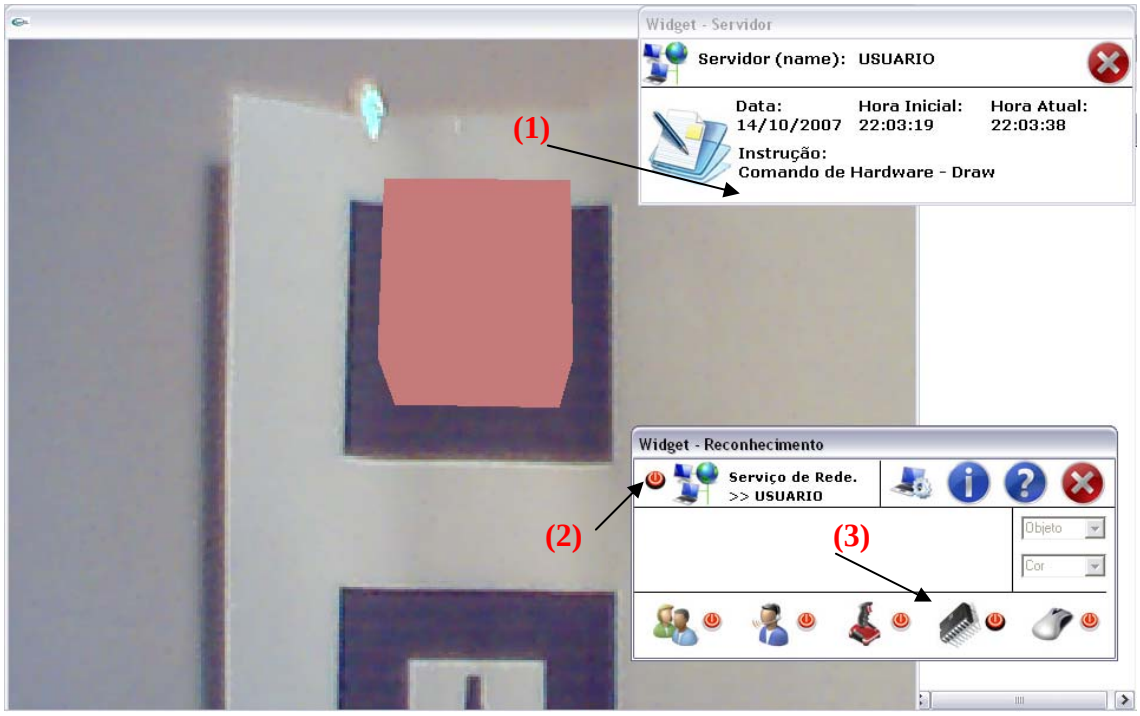


Figura 49 – Interação com Hardware

Conforme figura 49, encontramos o módulo servidor ativo e a instrução do dispositivo de hardware (1), sendo reconhecida. No lado cliente o serviço de rede ativo (2) e a interação como o dispositivo de hardware ativo (3).

4.3 PROTÓTIPO 03 – INTERFACE COM JOYSTICK BASEADA EM SOFTWARE

Para a Interação com o joystick, optou-se por um modelo dual, para que desta forma cada controle representasse um tipo de coordenada, desta forma, um *stick* (esquerdo) representa as coordenadas X e Y e outro *stick* (direito) representa a coordenada Z. A figura 50 mostra a representação dos controles e as suas coordenadas.

Figura 50 – Joystick e as suas coordenadas

4.3.1 – ARQUITETURA DE SOFTWARE PARA RECONHECIMENTO DO JOYSTICK

A arquitetura de software baseada em reconhecimento do joystick possui vários aspectos similares aos que foram apresentados no protótipo 01; desta forma, serão abordados somente os elementos particulares a esta arquitetura, como por exemplo os recursos do DirectX®.

O DirectX® é uma interface de software desenvolvida pela Microsoft® que tem acesso direto a alguns mecanismos, como vídeo para composição de imagem e dispositivos de controle, neste caso o joystick. Desta forma, os valores provenientes do joystick, via entrada USB, serão tratados pelas funcionalidades do DirectX, neste caso o DirectInput, que tem por finalidade permitir detectar as instruções de teclado, mouse e joystick. Como teclado e mouse já são mais comuns, a maioria das linguagens de programações já possui recursos para fazer a leitura destes dispositivos diferente do joystick que não é um dispositivo padrão.

Algumas características relevantes a serem observadas quando trabalhamos com o joystick.

- **Eixos** – os controles são preparados para trabalhar com dois eixos X e Y, mas seu conceito poderá ser utilizado para trabalhar com outros eixos.
- **Range** – é uma faixa de atuação dos eixos, geralmente utiliza-se o valor de 10.000 para cada eixo, sendo 5.000 o seu estado inerte, ou seja, a sua posição central.
- **Zona morta** – é uma porcentagem do eixo que aponta para a posição central, ou seja, se estabelecido uma margem de 10%, sobre um valor de 10.000, tudo que estiver entre 4.000 e 6.000 será considerado como 5.000.
- **Zona de saturação** – idem a zona morta, mas agora atua nas extremidades dos eixos.

4.3.2 – EXECUÇÃO E TESTES

Assim como nas especificações anteriores, a interação multimodal deverá seguir os mesmos procedimentos, onde o usuário deverá realizar a configuração da rede e posteriormente selecionar qual forma de interação deseja utilizar.

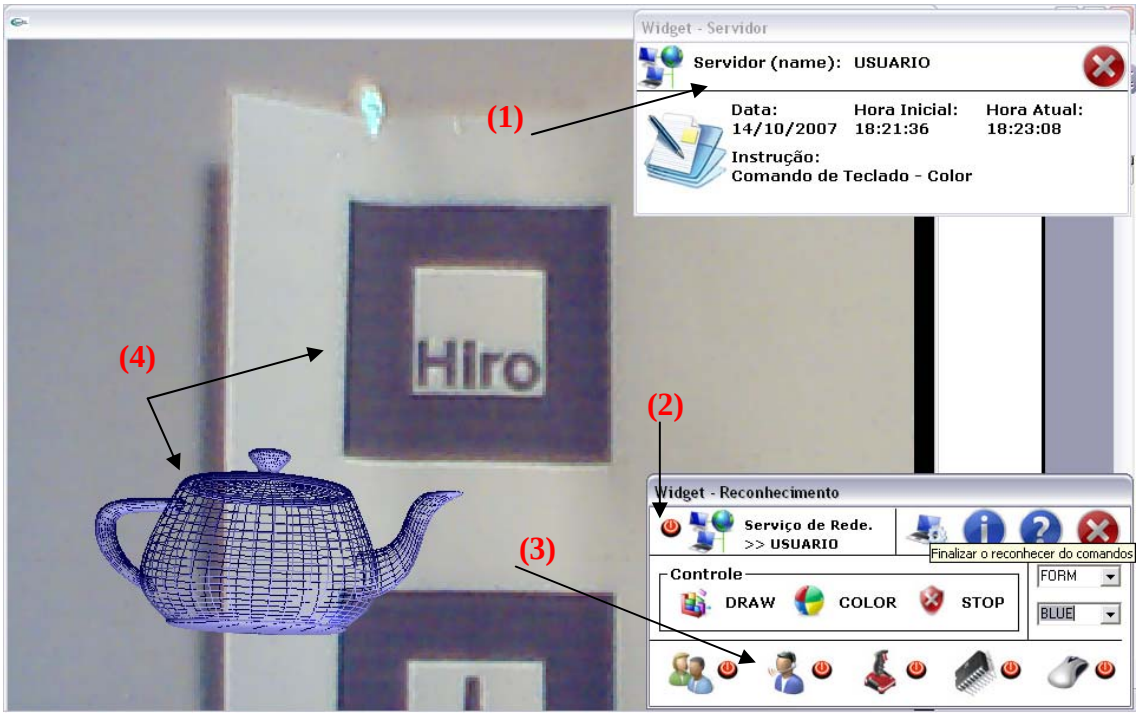


Figura 51 – Interação Multimodal

A figura 51 mostra o módulo servidor (1) ativo, o módulo cliente está com o serviço de rede ativo (2), os botões de interação estão todos ativos (3), no fundo podemos perceber um objeto deslocado de sua posição central (4), como resultado da interação com o joystick. Sua função é deslocar o objeto renderizado sobre o marcador, movimentando-o sobre os eixos x, y e z. No exemplo da figura 50, visualizada anteriormente, podemos identificar qual dos *sticks* representa os eixos de deslocamento da imagem.

4.4 PROTÓTIPO 04 – INTERFACE COM MOUSE

Similar ao protótipo 01, a interface com mouse baseada em software irá realizar a interação com o mouse de forma bastante simples. Com o dispositivo

ativado o usuário terá duas caixas de opções (figura 52), podendo realizar dois tipos de interação: selecionar o tipo de objeto a ser utilizado, e definir a sua cor.

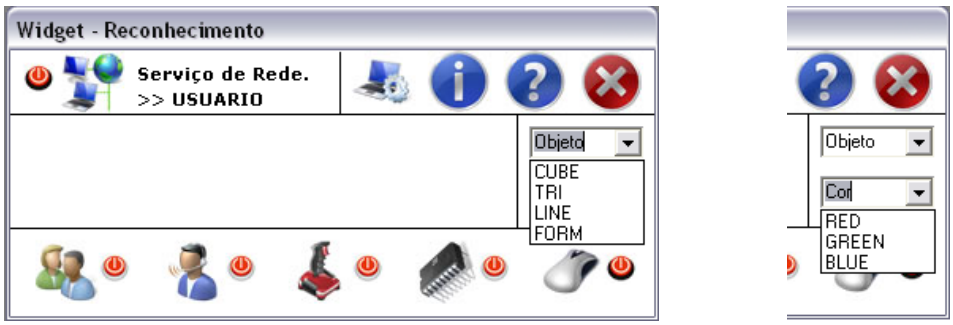


Figura 52 – Interação com Mouse

4.4.1 – EXECUÇÃO E TESTES

A interação com este tipo de interface deverá seguir os mesmos procedimentos descritos anteriormente, onde o usuário deverá ativar o serviço de rede e posteriormente selecionar qual forma de interação deseja utilizar.

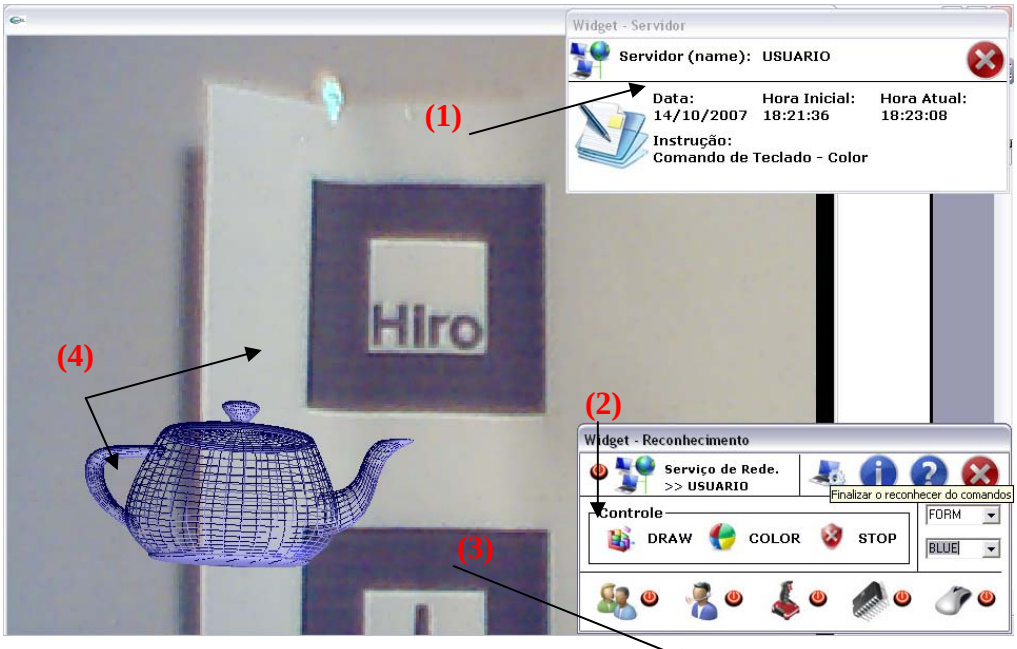


Figura 53 – Interação Multimodal

Na Figura 53, o módulo servidor (1) está ativo, o módulo cliente está com o serviço de rede ativo (2), os botões de interação estão todos ativos (3), observando os dados da combo, observamos que o objeto selecionado é **FORM** e a cor **BLUE**.

5. ARTOOLKIT MULTIMODAL

Uma interface multimodal tem como objetivo receber sinais de diferentes dispositivos, como um reconhecedor de comandos de voz, mouse, joystick e de hardware customizado. Além disso, o tratamento das informações recebidas deverá ser efetuado, de modo a gerar mensagens correspondentes, a serem enviadas via *socket* até o servidor. Este por sua vez emitirá mensagens padronizadas ao aplicativo ARToolKit, que deverá interpretá-las e executar as ações correspondentes. Esse conjunto constitui-se no **ARToolKit Multimodal**.

A figura 54 mostra a interação multimodal do usuário, dispositivos e a forma de interação. O usuário poderá selecionar um ou mais dentre vários dispositivos de interação com o ARToolKit Multimodal: comando de voz, joystick, mouse e hardware.

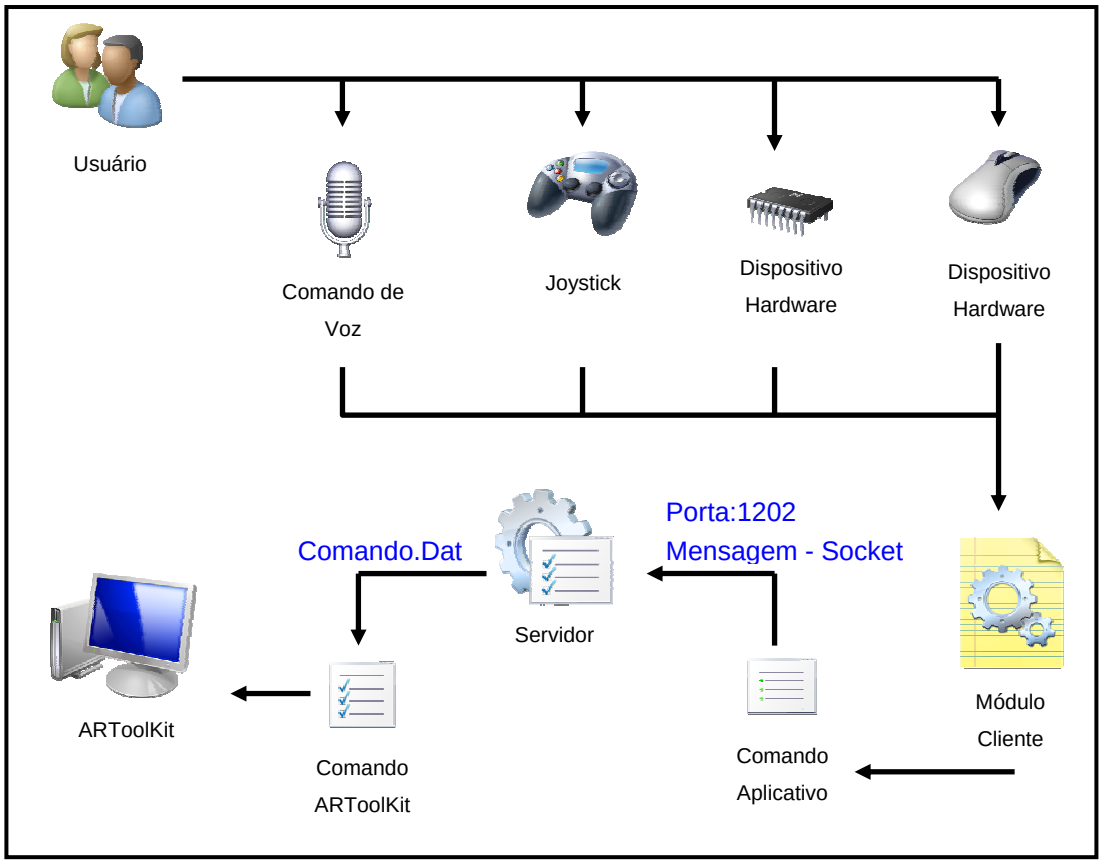


Figura 54 – Arquitetura do ARToolKit Multimodal

A partir do momento que uma interação é executada, o módulo cliente reconhece o sinal, e usando a porta 1202, envia uma mensagem ao módulo servidor através de *socket*, gerando o arquivo de comandos para o ARToolKit Multimodal, que realiza a leitura do arquivo e posteriormente executa o comando.

O ARToolKit Multimodal é composto pelas estruturas apresentadas nos protótipos anteriores, em um único módulo, mantendo todas as características de suas partes individuais.

5.1 INTERAÇÃO EM EXECUÇÃO

Assim como especificado anteriormente, a interação multimodal deverá seguir os mesmos procedimentos, onde o usuário deverá ativar o serviço de rede e posteriormente selecionar qual forma de interação deseja utilizar.

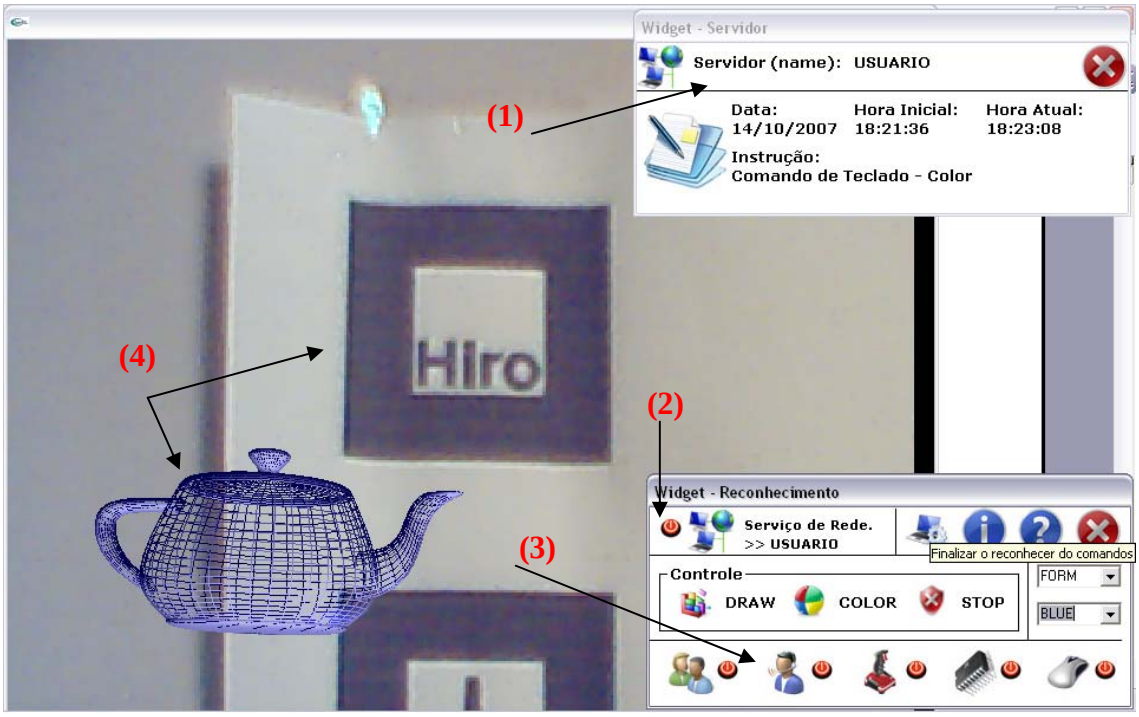


Figura 55 – Interação Multimodal

Na Figura 55, o módulo servidor (1) está ativo, o módulo cliente está com o serviço de rede ativo (2), os botões de interação estão todos ativos (3).

Desta forma o usuário poderá escolher entre os botões as diversas interações fornecidas pela arquitetura.

5.1.1 GERADOR DE COMANDOS

Como mencionado nos itens anteriores, no módulo cliente as opções da interfaces: comando de voz, joystick, mouse e hardware (CI), envia uma mensagem ao módulo servidor que gera um arquivo de texto, o qual será lido pelo ARToolKit Multimodal. A possibilidade da integração de diferentes recursos, bem como a associação futura a outros aplicativos é determinada pela **padronização das mensagens** de recepção tratadas pelo módulo cliente e o módulo servidor.

5.1.1.1 PROTOCOLO DE COMUNICAÇÃO DO MÓDULO CLIENTE

No módulo cliente, criou-se uma convenção de siglas a serem enviadas via *socket* ao módulo servidor, estas siglas sempre com três letras, representam a junção de expressões de comandos, conforme tabela 03.

Dispositivo	Entrada	Mensagem
Mouse	Clique nas caixas combos para: - definição de cor - definição de objeto	**K**eyboard **C**olor **R**ed (KCR) **K**eyboard **C**olor **G**reen (KCG) **K**eyboard **C**olor **B**lue (KCB) **K**eyboard **D**raw Object **A** (KDA) - Cube **K**eyboard **D**raw Object **B** (KDB) - Tri **K**eyboard **D**raw Object **C** (KDC) - Line **K**eyboard **D**raw Object **D** (KDD) – Form **K**eyboard **D**raw Object **X** (KDX) – Nenhum
Dispositivo	**Entrada**	**Mensagem**
Comando de Voz		**V**oice **C**olor **R**ed (VCR) **V**oice **C**olor **G**reen (VCG)

Dispositivo	Entrada	Mensagem
Comando de Voz	Reconhecimento de comandos através de uma listagem pré definida pela aplicação	**V**oice **C**olor **B**lue (VCB) **V**oice **D**raw Object **A** (VDA) - Cube **V**oice **D**raw Object **B** (VDB) - Tri **V**oice **D**raw Object **C** (VDC) - Line **V**oice **D**raw Object **D** (VDD) – Form **V**oice **D**raw Object **X** (VDX) – Nenhum
Dispositivo	**Entrada**	**Mensagem**
Hardware	Números binários recebidos através da porta paralela	**H**ardware **D**raw Object **A** (HDA) - Cube **H**ardware **D**raw Object **B** (HDB) - Tri **H**ardware **D**raw Object **C** (HDC) - Line **H**ardware **D**raw Object **D** (HDD) – Form **H**ardware **D**raw Object **X** (HDX) – Nenhum
Dispositivo	**Entrada**	**Mensagem**
Joystick	Movimento dos Stiks, via USB.	**J**oystick **X P**ositive (JXP) **J**oystick **X N**egative (JXN) **J**oystick **Y P**ositive (JXY) **J**oystick **Y N**egative (JXY) **J**oystick **Z P**ositive (JXZ) **J**oystick **Z N**egative (JXZ)

Tabela 03 – Mensagem do Módulo Cliente

Desta forma outra aplicação que siga o padrão acima poderá enviar mensagens ao módulo servidor, que por sua vez identifica pela seqüência de caracteres as ações a serem realizadas, identificando a primeira letra o dispositivo de origem (K – keyboard (mouse), V - voz, H – hardware e J - joystick), a segunda letra a ação que deverá ser tomada (C - cor e D – draw), e a terceira letra que parâmetro se refere a ação. No caso do joystick P – positivo e N – negativo, representa qual é o deslocamento para as coordenadas X, Y e Z, dentro do espaço 3D. A troca de mensagem mencionada anteriormente encontra-se ilustrada na figura 56.

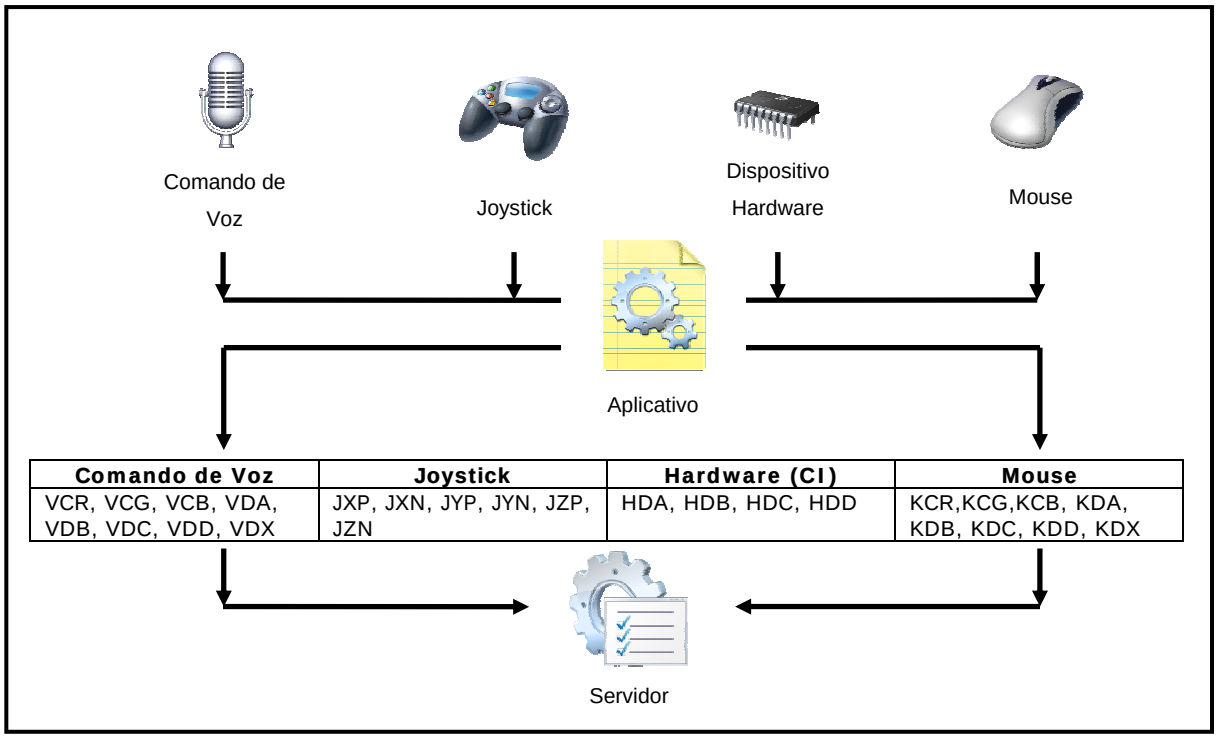


Figura 56 – Troca de mensagens - Interação x Módulo Cliente

5.1.1.2 PROTOCOLO DE COMUNICAÇÃO DO SERVIDOR

No módulo servidor, as mensagens recebidas são verificadas e, de acordo com as letras, determina-se a mensagem de interação a ser fornecida ao usuário via *widge*t e qual ação executada e o parâmetro será executado; neste momento, o teste de colisão é realizado para verificar se nenhuma mensagem estará em conflito ou exista ambigüidade.

A partir deste processo, o arquivo “comando.dat”, conforme ilustrado figura 57, será gerado na pasta raiz da aplicação para a leitura do ARToolKit Multimodal.

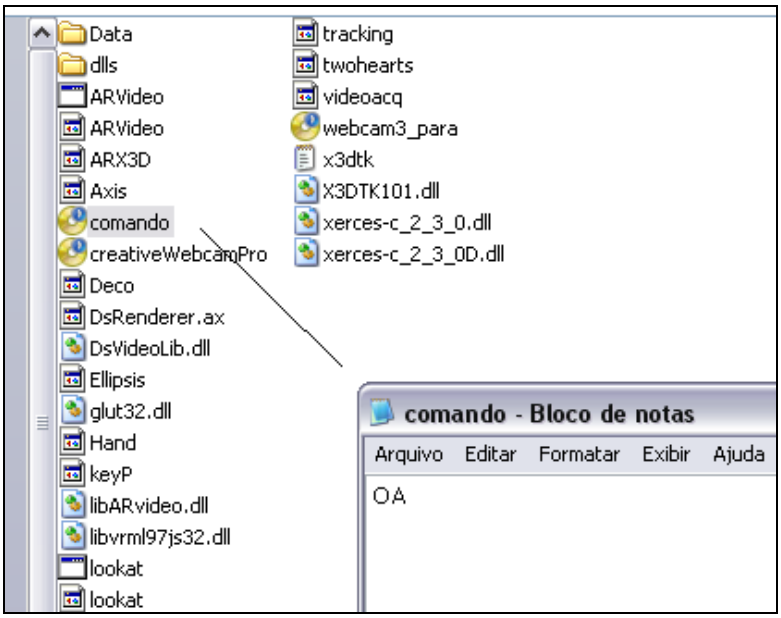


Figura 57 – Gerando o arquivo comando.dat

Da mesma forma que as mensagens provenientes do módulo cliente são padronizadas, o módulo servidor também padroniza as suas mensagens, conforme tabela 04.

Dispositivo	Ação	Mensagem
Comando de Voz Mouse Hardware(CI)	Renderização de Imagem	**O**bject **A** (OA) - Cube **O**bject **B** (OB) - Tri **O**bject **C** (OC) - Line **O**bject **D** (OD) – Form
Dispositivo	**Ação**	**Mensagem**
Comando de Voz Mouse Hardware(CI)	Colorir imagem	**P**rint **R**ed (PR) **P**rint **G**reen (PG) **P**rint **B**lue (PB)
Dispositivo	**Ação**	**Mensagem**
Joystick	Movimentar o objeto	**J**oystick **X P**ositive (JXP) **J**oystick **X N**egative (JXN) **J**oystick **Y P**ositive (JXY) **J**oystick **Y N**egative (JXY) **J**oystick **Z P**ositive (JXZ) **J**oystick **Z N**egative (JXZ)

Tabela 04 – Mensagem do módulo Servidor

De acordo com a tabela as mensagens geradas pelo módulo servidor identificam:

- **O** – *Object*
- **P** – *Printer*

Para desenhar e colorir os objetos renderizados pelo aplicativo, as instruções de joystick permanecem inalteradas, figura 58.

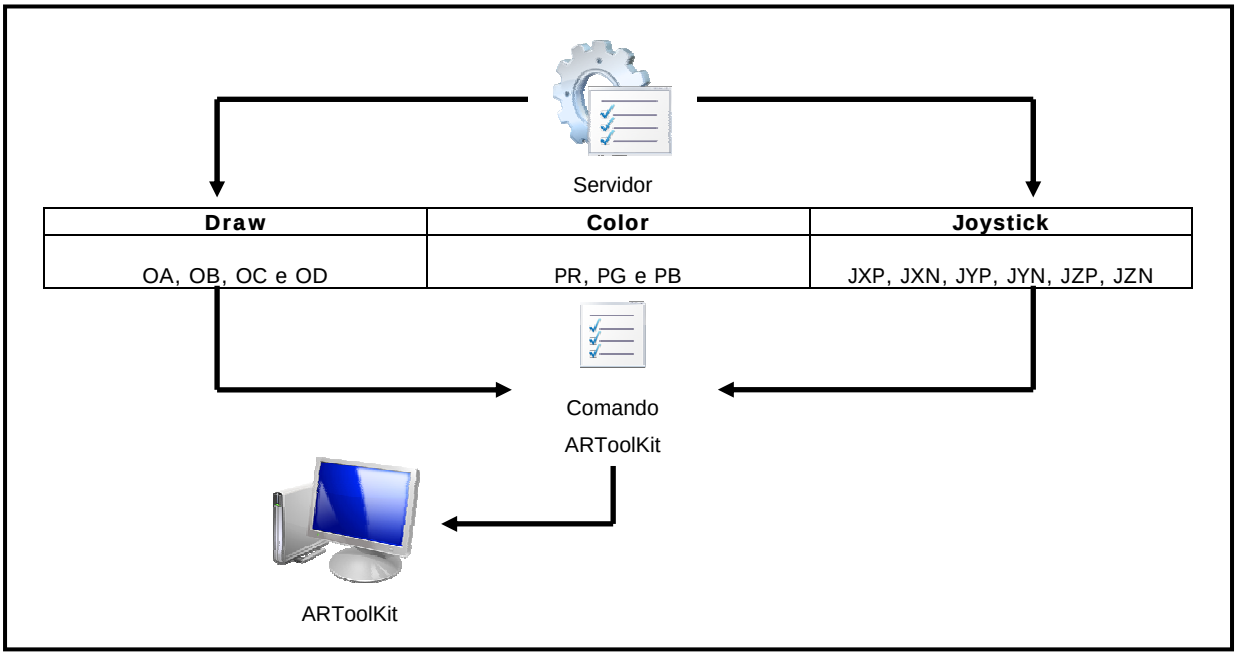


Figura 58 – Estrutura dos comandos

Com a segmentação e a padronização das mensagens, outros aplicativos poderão realizar de interação com o modelo ARToolKit Multimodal apresentado neste trabalho.

Seguindo os parâmetros apresentadas nos tópicos anteriores, as figuras 59 e 60 representam a aplicação em seu estado original e onde outros aplicativos poderão realizar a intervenção no que diz respeito a interação multimodal.

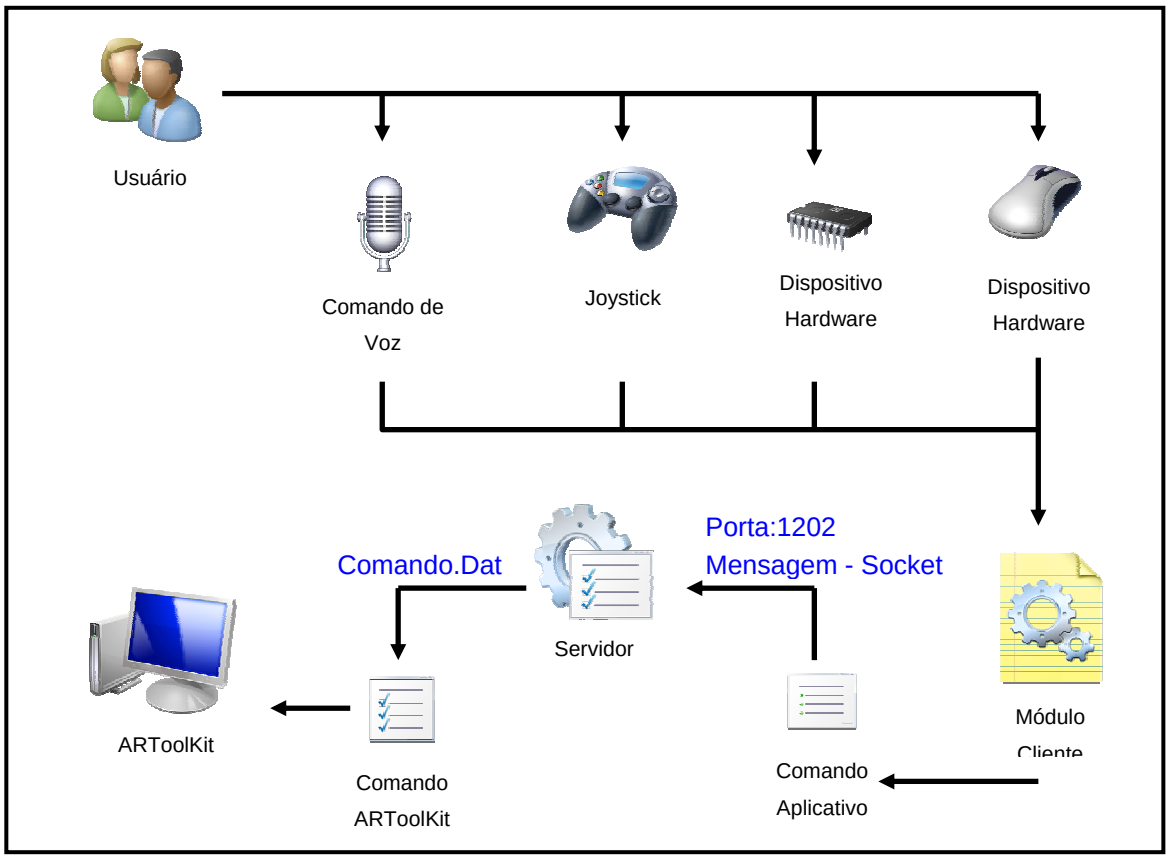


Figura 59 – Aplicação ARToolKit Multimodal

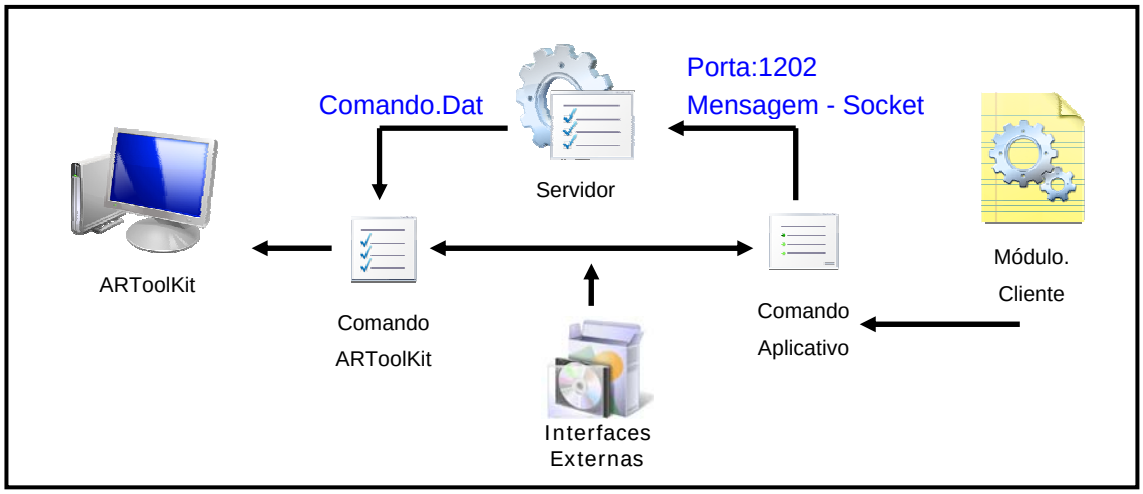


Figura 60 – Aplicação ARToolKit Multimodal com Interfaces Externos

5.2 Detalhes de Implementação

Neste capítulo, serão apresentados alguns detalhes de implementação realizadas para a configuração do ARToolKit Multimodal, os quais poderão ser utilizadas como referências para o desenvolvimento de novos projetos similares.

5.2.1 Módulo de Leitura

Conforme mencionado no capítulo 05, a estrutura de envio de mensagem ao servidor é realizada através de um documento de texto que possui mensagens padronizadas. A codificação ora implementada foi realizada no arquivo ARVideo.cpp, na função *draw(),* a qual tem por finalidade a renderização dos objetos.

O código abaixo representa a parte função da leitura do arquivo:

```
…
CHAR CARACTER;
IF((ARQUIVO = FOPEN("COMANDO.DAT","R")) != NULL){
DO {
        CARACTER = GETC(ARQUIVO);
        // INSTRUÇÕES
} WHILE(CARACTER != EOF);
FCLOSE(ARQUIVO);
```

As instruções especificadas dentro do loop serão executadas de acordo com os caracteres recebidos, desta forma teremos:

Objetos: Serão gerados, a partir do reconhecimento do código 79, letra “O”, indicando qual objeto será posteriormente renderizados, conforme código abaixo:

```
// sequência de controle para geração de Objetos
if(caracter==79){
   caracter = getc(arquivo);
      if(caracter >=65 && caracter <=68) {
            obj=caracter;
            break;}}
```

Os objetos estão representados pelas letras A, B, C e D, conforme seus respectivos códigos 65, 66, 67 e 68, representado pela tabela ASCII (*American Standard Code for Information Interchange*).

Todos os objetos possuem um tamanho pré-definido, representado pelo valor 50 (cinqüenta), armazenada na variável <tamanho>, a qual posteriormente será utilizada para modificações na escala do objeto.

```
// define variável do tamanho do Objeto
double      tamanho = 50;
```

A relação entre o código, objeto e seus respectivos código fonte poderá ser verificado através dos itens especificados abaixo:

Cubo: Letra ”A” – Código 65 – Código fonte, representação gráfica figura 61.

```
// este código desenha um cubo
if(obj==65)
      glutSolidCube(tamanho);
```

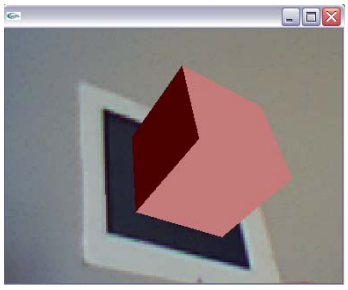


Figura 61 – CUB – Codificação *OpenGL*

TRI (triângulo): Letra ”B” – Código 66 – Código fonte, representação gráfica figura 62.

```
//este código desenha um triangulo
if(obj==66){
        glBegin(GL_TRIANGLES);
                glVertex2d(-tamanho,-tamanho);
                glVertex2d(tamanho,-tamanho);
                glVertex2d(0,tamanho);
        glEnd();}
```

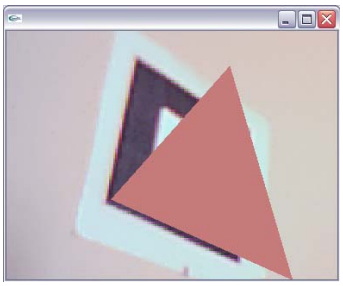


Figura 62 – TRI - Triângulo – Codificação *OpenGL*

LINE (círculo): Letra ”C” – Código 67 – Código fonte, representação gráfica figura 63.

```
//este código desenha um circulo
if(obj==67){
        glBegin(GL_LINE_LOOP);
```

```
int i;
double angulo;
for (i=0;i<=50;i++)      {
    angulo= 2 * 3.14 * i / tamanho;
    glVertex2f(tamanho*cos(angulo), tamanho*sin(angulo));
}
glEnd();}
```

Figura 63 – LINE – Codificação *OpenGL*

FORM: Letra ”D” – Código 68 – Código fonte, representação gráfica figura 64.

```
// este código desenha um pote
if(obj==68)
        glutWireTeapot(tamanho);
```

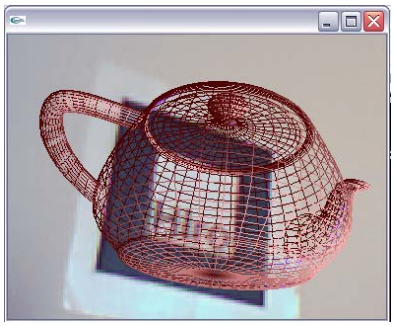


Figura 64 – FORM – Codificação *OpenGL*

5.2.2 MODIFICANDO AS CORES

A modificação das cores segue o mesmo princípio da leitura dos objetos, onde três varáveis: cr1, cr2 e cr3 deverão representar respectivamente o padrão RGB (*Red, Green, Blue*), no código abaixo a cr1 (vermelho), já está definida como default.

```
// inclusão das variáveis para modificação de cor.
float cr1,cr2,cr3;
cr1=1.0;
cr2=0.0;
cr3=0.0;
```

Para cada elemento de cor poderá ser atribuídos valores entre 0 (zero) e 1 (um), podendo realizar combinações para geração de novas cores. Neste projeto somente as cores básicas foram utilizadas, a figura 65 abaixo representa os três padrões de cores utilizados.

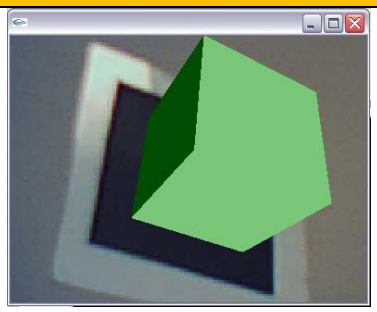
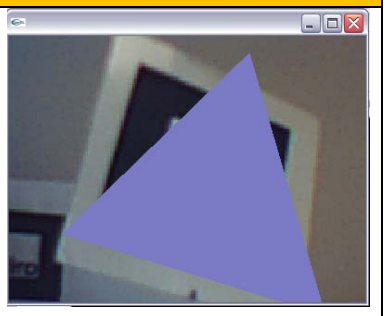
Red	Green	Blue
		

Figura 65 - Padrões de Cores

Os valores referentes as cores são transferidos como parâmetros para a instrução *OpenGL*, como segue:

```
GLfloat   mat_ambient[]     = {cr1, cr2, cr3, 1.0};
```

5.2.3 MOVIMENTANDO OBJETOS

A movimentação de um objeto é baseada sobre os três eixos que compõe o espaço 3D, onde cada eixo poderá ser acrescido de uma unidade positiva ou negativa conforme a utilização do joystick sobre o eixo especificado. O código abaixo descreve, o processo de identificação dos eixos através dos valores 74, 88, 89 e 90, os quais representam as letras J, X, Y e Z. Conforme mencionado anteriormente a letra “J” identifica o meio de interação, neste caso o joystick.

```
// sequência de controle para comando de voz
                if(caracter==74){
                   caracter = getc(arquivo);
                      if(caracter==88){
                                caracter = getc(arquivo);
                                if(caracter==80){
                                        eixox=eixox+1;}
                                if(caracter==78){
                                        eixox=eixox-1;}
                                break;
                                }
                      if(caracter==89){
                                caracter = getc(arquivo);
                                if(caracter==80){
                                        eixoy=eixoy-1;}
                                if(caracter==78){
                                        eixoy=eixoy+1;}
                                break;
                                }
                      if(caracter==90){
                                caracter = getc(arquivo);
```

```
if(caracter==80){
        eixoz=eixoz+1;}
if(caracter==78){
        eixoz=eixoz-1;}
break;
}
```

Após os eixos receberam as informações pertinente a movimentação do Joystick, os valores são transferidos no formato de parâmetros para a instrução *OpenGL*, como segue:

```
glTranslatef( eixox, eixoy, eixoz);
```

5.2.4 INVERTENDO A IMAGEM

É característica do ARToolKit em realizar a visualização do objeto no forma invertida, para que se tenha um aproximação maior do mundo real, utilizou-se um procedimento desenvolvido pelo Prof. Dr. Nivaldi Calonego Junior (UNIMEP), para realizar a inversão da imagem, apresentada baixo:

```
// A Instrução abaixo permite realizar a inversão da imagem
// Prof.Dr.Nivaldi (Julho/2006)
        ARUint8         x;
        for (j=0; j<(ysize/2); j++)
        for (int i=0; i<4*xsize; i++) {
                x = dataPtr[i + j*4*xsize];
                dataPtr[i + j*4*xsize] = dataPtr[i+4*(ysize-j-1)*xsize];
                dataPtr[i+4*(ysize-j-1)*xsize] = x;
        }
        // Fim da instrução
```

Este procedimento está incorporada a função *void mainLoop()* do arquivo ARVideo.cpp.

5.2.5 Reconhecendo um Sinal do Hardware

A interação com o dispositivo de hardware ocorre através de toque do usuário nos botões de pressão, cujo sinal de retorno é tratado pelo módulo cliente e a mensagem padronizada é enviada ao módulo servidor. Como o dispositivo é alimentado diretamente pela porta paralela, a opção de ativação no módulo cliente (*widget*) realiza o envio dos sinais (5V) as saídas correspondentes.

Os códigos apresentados anteriormente serão reutilizados para o dispositivo de hardware.

5.3 Os módulos cliente/servidor para ARToolKit Multimodal

Nesta seção, serão descritos os layouts dos módulos cliente/servidor que fazem parte da aplicação ARToolKit Multimodal.

5.3.1 Módulo Servidor

No módulo servidor podemos observar as características de conexão e as mensagens que serão tratadas durante a interação, figura 66.

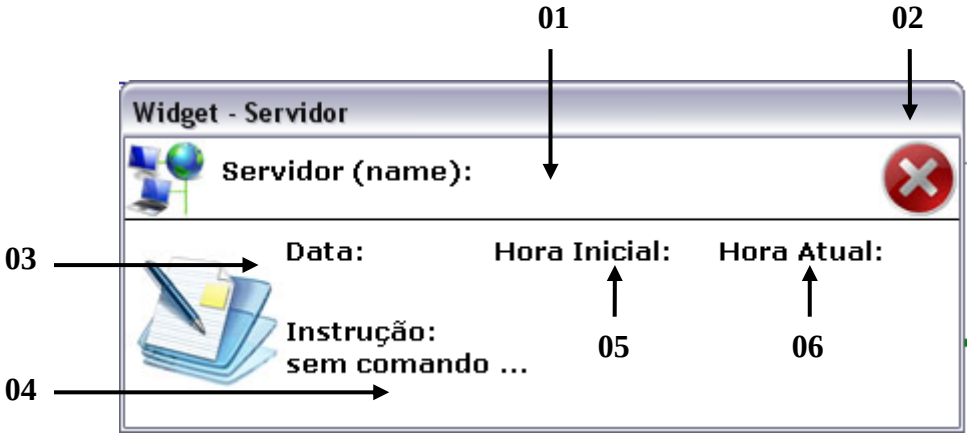


Figura 66 – Detalhes Módulo Servidor

Descrições:

01 – Identifica o nome da máquina onde o módulo servidor está instalado.

02 – Finaliza o módulo servidor.

03 – Informa a data de conexão.

04 – Identifica qual o tipo de interação foi utilizada.

05 – Hora inicial da aplicação, registro o início da execução do módulo.

06 – Tempo real.

5.3.2 MÓDULO CLIENTE

No módulo cliente encontraremos todos os recursos relacionados aos modelos de interações que o usuário poderá ter acesso.

Na figura 67, podemos verificar todos os recursos disponíveis neste módulo.

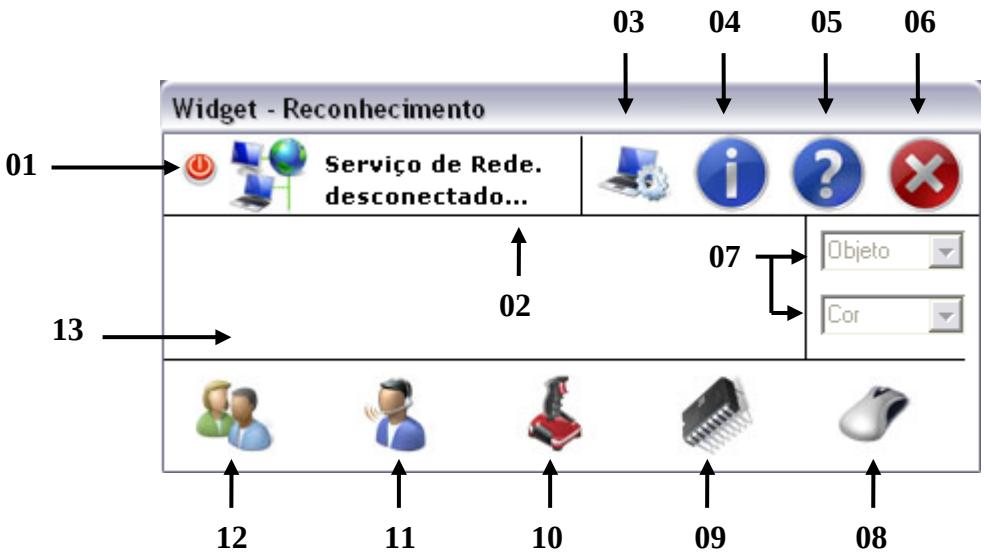


Figura 67 – Detalhes Módulo Cliente

Descrições:

01 – Botão para conexão com o módulo servidor, antes de ser ativado o nome da máquina onde o servidor estiver instalado deverá ser configurada no botão ferramentas.

02 – Após a conexão será disponibilizado o nome da máquina onde o módulo servidor foi instalado.

03 – Nesta opção deverá ser informado o nome da máquina onde o módulo servidor for instalado, como default o nome da máquina local é carregado, figura 68. Na opção, encontramos dois botões: “gravar e sair” e “não gravar e sair”.

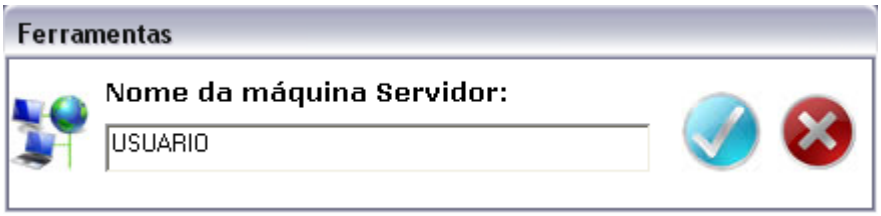


Figura 68 – Definição do nome do servidor

04 – Uma janela será apresentada, figura 69, mostrando informações sobre o módulo.

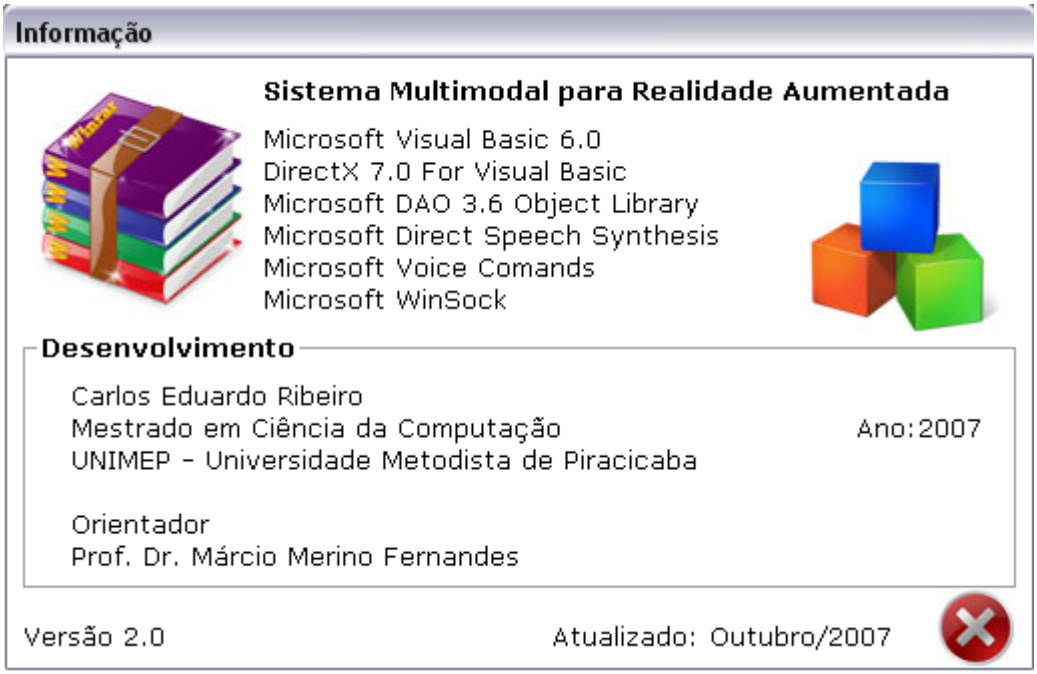


Figura 69 – Informações sobre o sistema

05 – Tela de help do sistema, no formato HTML (*HyperText Markup Language*), explica as opções do módulo cliente, figura 70.

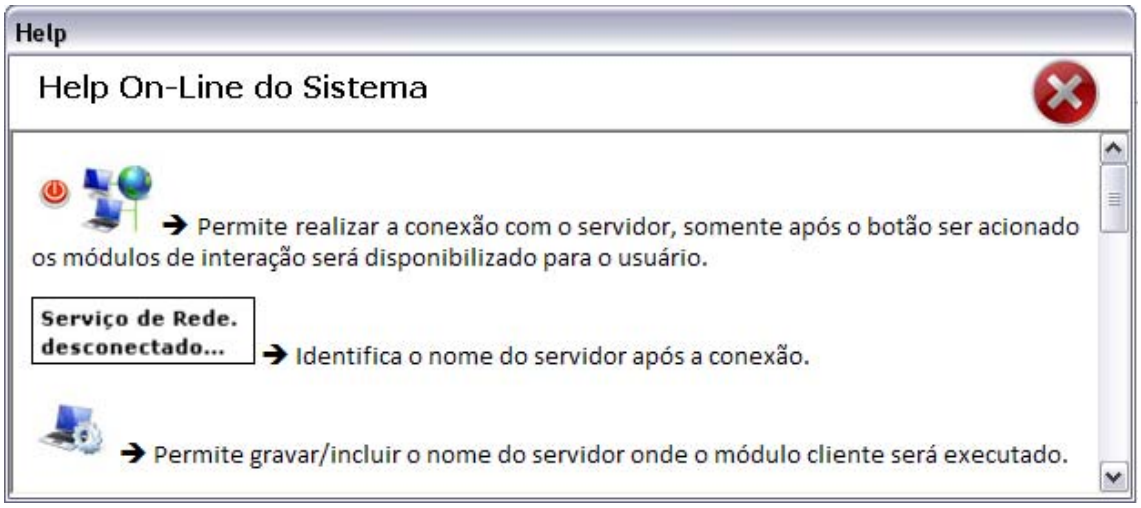


Figura 70 – Help do módulo cliente

06 - Finalização do módulo cliente.

07 - Opções de interação com o mouse.

08 - Ativa/desativa interação com mouse.

09 - Ativa/desativa interação com hardware.

10 - Ativa/desativa interação com joystick.

11 - Ativa/desativa interação com voz.

12 - Ativa/desativa interação recurso de fala.

13 - Área de visualização referente às opções para interação com voz, com as opções de desenho (*draw*) e pintura (*color*), figura 71.

Figura 71 – Opções de comando – Voz

6. APLICAÇÕES PRÁTICAS COM O ARTOOLKIT MULTIMODAL

Neste capítulo serão apresentados algumas aplicações práticas já realizadas com o ARToolKit Multimodal, e outras sugestões que poderão ser desenvolvidas à partir destas aplicações.

6.1 ARTOOLKIT MULTIMODAL EM SALA DE AULA

Como teste prático, o ARToolKit Multimodal, foi levado para uma sala de aula, para uma apresentação de fundamentos de RV e RA. Esta aula prática utilizou-se de apenas duas modalidades de interação: reconhecimento de comandos de voz e joystick (figura 72).

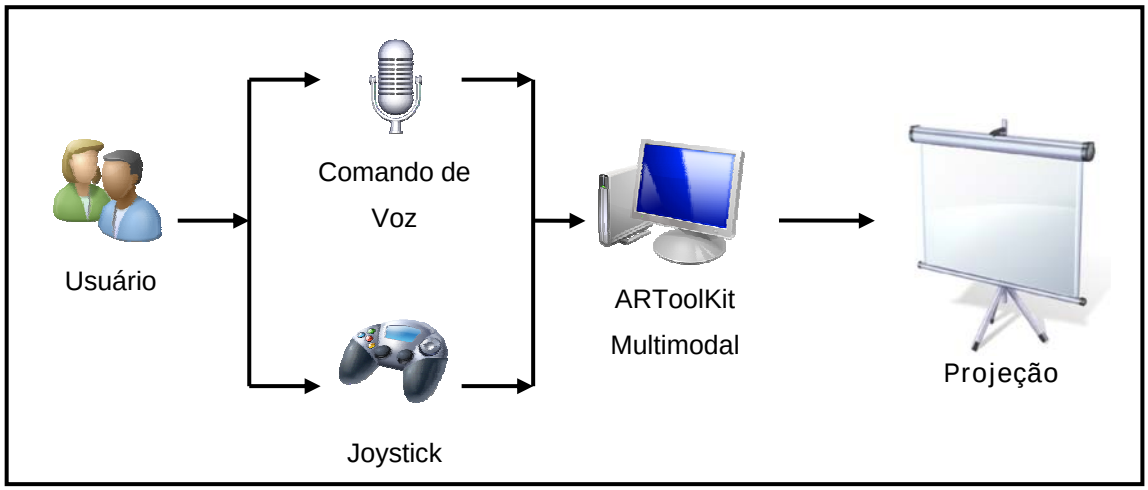


Figura 72 – Teste prático com ARToolKit Multimodal

No primeiro teste o comando de voz e o joystick, permaneceram ativos, observando-se:

- Durante a apresentação os objetos eram iniciados automaticamente sem que o comando para tal finalidade fosse acionado através do comando de voz.
- O som ambiente passou a interferir na aplicação ativando objetos.
- O joystick permaneceu funcionando sem qualquer problema.

Em um segundo teste, foi inserido um pré-comando que através da voz disponibilizaria os demais comandos, os quais após a execução eram desabilitados, aguardando que fossem ativados novamente. Esses comandos foram estruturados na programação em formato de árvore (figura 73).

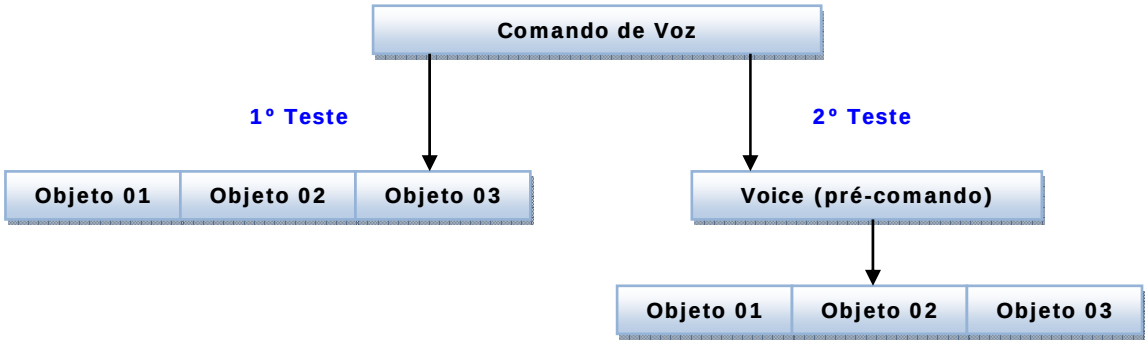


Figura 73 – Estruturação de comandos

6.2. APLICAÇÕES EM VRML

O módulo cliente/servidor do ARToolKit Multimodal, poderá ser utilizado para interagir com aplicações VRML, utilizando diferentes dispositivos de interação.

6.2.1. ANIMAÇÃO DE OBJETOS

Um exemplo hipotético é a utilização dos recursos de reconhecimento de voz para manipulação de objetos em VRML, renderizados pelo navegador, podendo mudar a forma do objeto, dar-lhe movimentos entre outras funcionalidades.

O modelo resume-se na utilização de uma prototipação em VRML, ou seja, os objetos poderão ser expandidos à partir de arquivos externos. No exemplo da figura 74, podemos verificar a interação com o arquivo externo e a aplicação VRML.

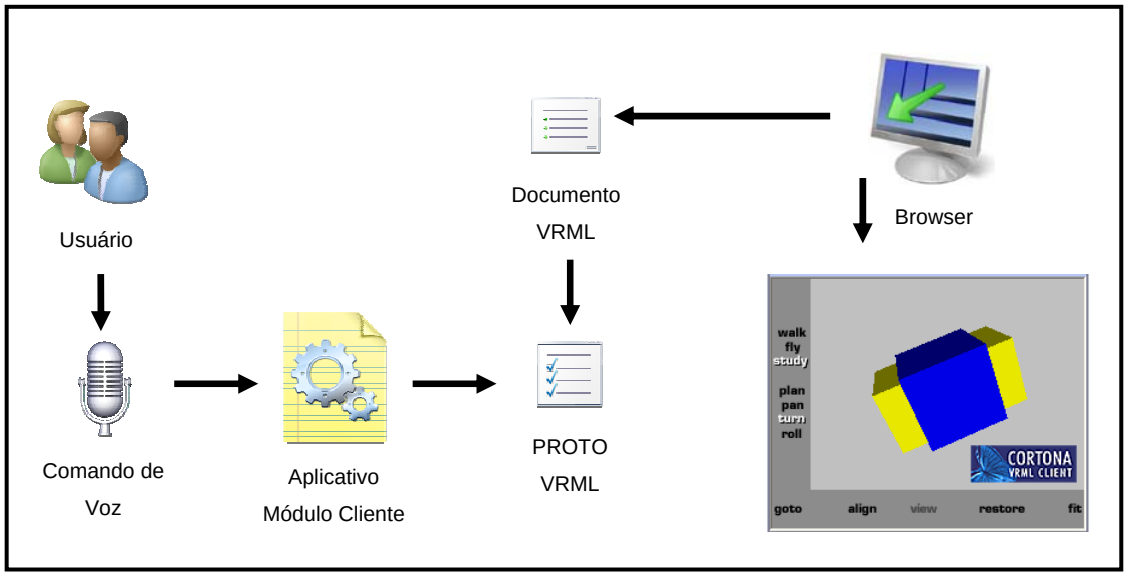


Figura 74 – Interação com Objetos VRML

6.2.2. GAMES

Usando o mesmo princípio do item anterior, associado a outro dispositivo de interação, como por exemplo, um protótipo de hardware ou o joystick, seria possível utilizar o ARToolKit Multimodal na área de games.

Um modelo básico poderá ser desenvolvido, utilizando os recursos de reconhecimento de voz, joystick e hardware, cujos sinais serão tratados pelo módulo cliente, possibilitando que objetos sejam movimentados em cena, permitindo adicionar outros efeitos que caracterizam a aplicação de um game, tais como som, objetos animados, placar, etc.

A figura 75 apresenta um protótipo de um jogo, utilizando os recursos da linguagem VRML, cujo cenário é uma nave espacial em movimento na captura de esferas, visualizada na figura 76.

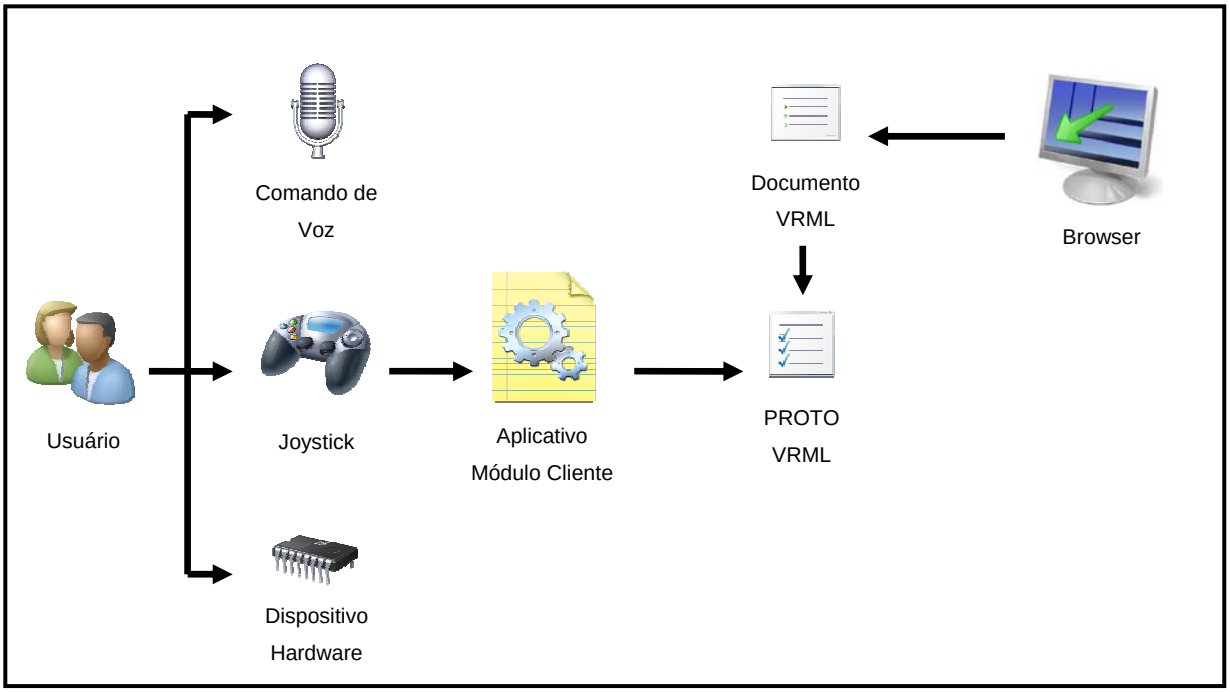


Figura 75 – Protótipo do Modelo

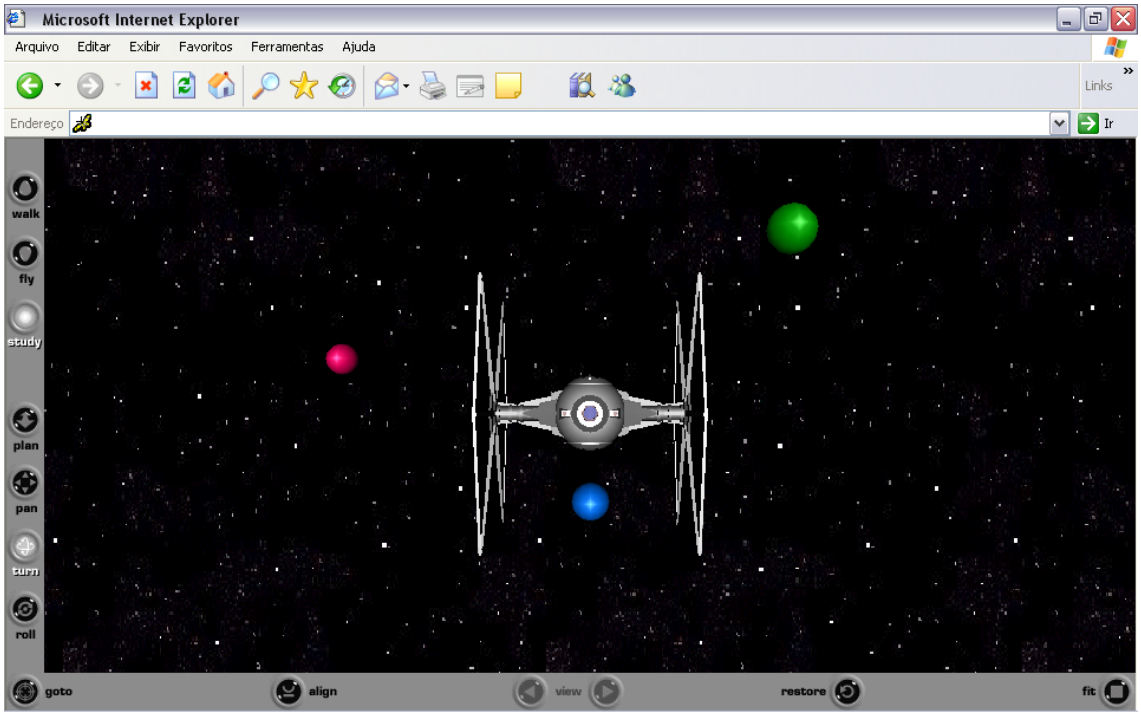


Figura 76 – Imagem do Jogo

O nave, visualizada na figura 76, é um código exemplo disponibilizado após a instalação do *OpenSG* 1.6 (*Open Source Scenegraph*), [OpenSG,2007].

6.3. PROTÓTIPO – RECONHECEDOR DE MOVIMENTOS

O objetivo do aplicativo é realizar a interação com o usuário através do reconhecimento de movimento ou deslocamento de objetos com interação em um ambiente de RA, detectado por sensores a laser. Na figura 77, mostra a exemplificação do projeto.

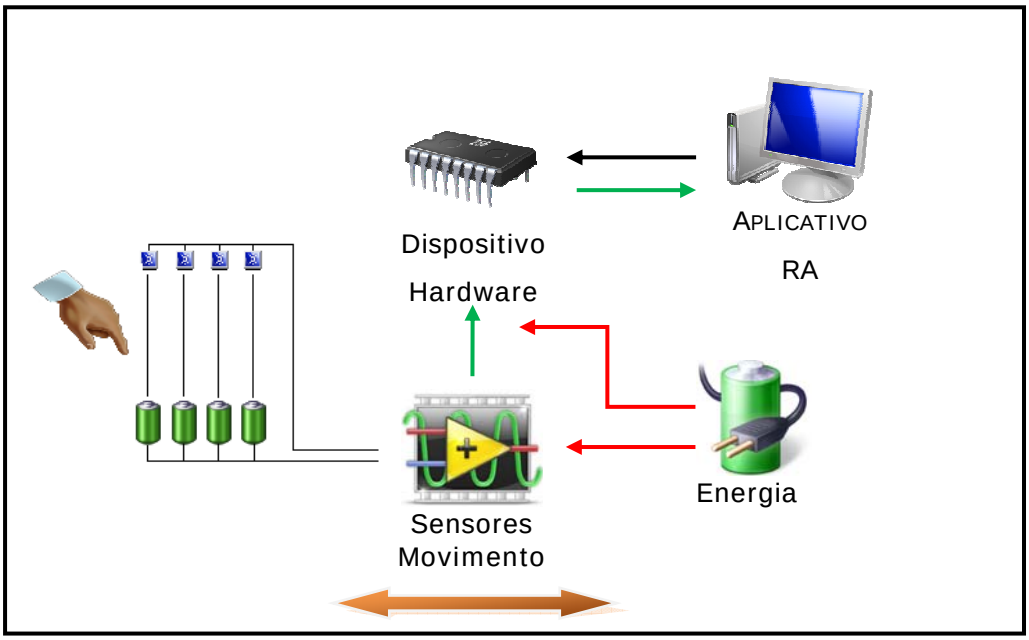


Figura 77 – Reconhecedor de movimentos

O dispositivo de hardware utilizado para o controle dos sensores segue as mesmas características outrora apresentadas, a modificação realizada diz respeito a troca dos botões de pressão para fotos-receptores. Para que a iluminação ambiente não realizasse nenhuma interferência nos sensores, dois lasers (doméstico) de diodo, figura 78 foram utilizados para manter os sinais ativos.

Figura 78 – Fotos-receptores e laser doméstico de diodo

Para garantir uma alimentação correta dos dispositivos de hardware e a integridade da porta paralela do equipamento, utilizou uma fonte de alimentação externa de 6V, garantindo a alimentação do foto-receptor, CI e de um par de lazeres.

A *protoboard* mantém as características utilizadas neste projeto apenas o sensor de pressão foi substituído por foto-receptor, a figura 79 mostra a estrutura da *protoboard* para a maquete e em detalhe a troca dos botões de pressão por foto-receptores.

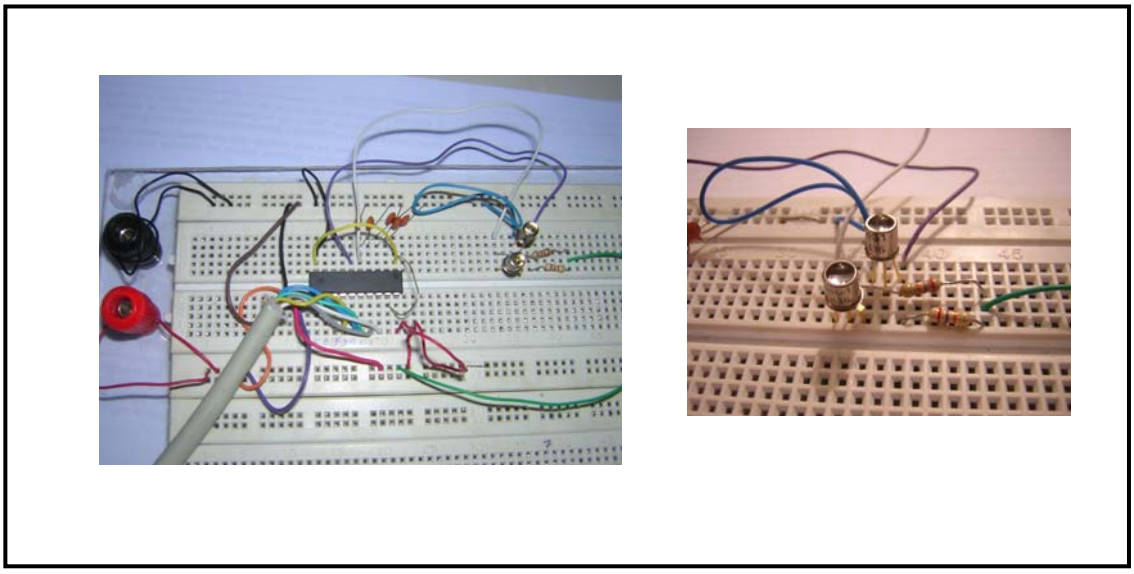


Figura 79 – *Protoboard* – Reconhecedor de Movimento

CONCLUSÃO

Este trabalho abordou questões teóricas e práticas relacionadas com a utilização de interfaces multimodais em sistemas de realidade aumentada. Como decorrência do trabalho, foi proposta uma arquitetura genérica visando padronizar e facilitar a incorporação de variados dispositivos de interface no sistema ARToolkit. Esse conjunto, denominado ARToolKit Multimodal, foi utilizado para o desenvolvimento de algumas aplicações práticas, utilizando realidade aumentada com diferentes dispositivos de entrada, que acredita-se ser uma tendência na área.

Em decorrência do trabalho desenvolvido, chegou-se a algumas conclusões que podem ser úteis para a evolução deste trabalho, ou outros relacionados. Uma delas é que o reconhecimento de voz, por ser uma forma de interação bastante natural e eficiente, deve ser melhor explorada em sistemas avançados de computação. Apesar de não ser perfeito, o nível de evolução dessa tecnologia já permite a sua utilização em sistemas que exigem um grau de robustez elevado.

Outra conclusão é que as interfaces multimodais, nas suas mais variadas formas e configurações, não combinam necessariamente com todo tipo de aplicação. Ou seja, é importante que se leve em consideração qual é o objetivo principal da aplicação, e quais os meios mais eficientes e interessantes para que esta aplicação seja viável operacionalmente.

Estas considerações foram baseadas nos diferentes protótipos que foram implementados para testes dos conceitos ora explorados durante o desenvolvimento deste trabalho. Das observações efetuadas durante o processo de desenvolvimento e testes realizados, deve ser considerado que:

- A aceitação de uma interface multimodal é dependente das preferências pessoais de quem a está utilizando, reportando para uma ou outra de acordo com a sua facilidade de adaptação;
- A utilização de termos na língua inglesa para interfaces de reconhecimento de voz, e a relação de comandos associados a estes termos, podem gerar desconforto aos usuários que não possuem familiaridade com o idioma Inglês;

- Muitas vezes fatores externos podem influenciar diretamente na interface de reconhecimento de voz, como ruídos externos e a fonética.
- Para usuários inexperientes, muitas vezes a utilização de múltiplos recursos de interação pode gerar desconforto e dificuldade no uso do sistema;

Em relação às técnicas de interação abordadas neste trabalho, vários aspectos positivos poderão ser considerados, como por exemplo, a interface de voz que poderá ser direcionada para:

- Desenvolvimento de jogos com interação de joystick e reconhecimento de voz, onde o jogador possui as mãos “ocupadas” no controle do joystick, e comandos podem ser acionados por voz, salvo a performance do jogo.
- A utilização de aplicações com reconhecimento de voz por pessoas com necessidades especiais;
- A utilização de diferentes recursos de acordo com a aplicação pode ser utilizada em diferentes equipamentos (hardware) que não possuem todos os níveis de interação. Dessa forma, o usuário poderá escolher entre as interações aquelas que possuem disponibilidade no hardware.

Finalmente em relação aos protótipos desenvolvidos neste trabalho, convém especificar que os mesmos foram destinados para prova dos conceitos e abrir caminho para que outras pesquisas sejam criadas, explorando os novas possibilidades oferecidas pelo uso de interfaces multimodais em sistemas de realidade aumentada.

REFERÊNCIAS BIBLIOGRÁFICAS

[ALCÂNTARA,2007] – ALCÂNTARA, Roberto. **Acesso à Porta Paralela nos Windows XP / NT/ 2000**. Disponível em: http://www2.eletronica.org/hack-s-dicas/acesso-a-porta-paralela-nos-windows-xp-nt-2000/. Acesso em: 10 set. 2007.

[ALTHOFF, e outros, 2003] – ALTHOFF, F.; MCGLAUN, G.;SCHULLER,B.; MORGUET. P.;LANG. M. **Using Multimodal Interaction to Navigate in Arbitrary Virtual VRML Worlds**. ACM, Orlando, Dec. 2003.

[ARToolKit,2006] – ARToolKit**, ARToolKit Home Page.** Disponível em: http://www.hitl.washington.edu/artoolkit/. Acesso em: 20 nov. 2006.

[ARToolKit,2006] – CONSULARO, L.A.; CALONEGO Jr, N.; DAINESE, C.A.; GARBIN, T. R.; KIRNER, C.; TRINDADE, J.; FIOLHAIS, C. - **ARToolKit: Aspectos Técnicos e Aplicações Educacionais.** In: CARDOSO, A.; LAMOUNIER Jr, E. editores. Realidade Virtual: Uma Abordagem Prática. Livro dos Minicursos do SVR2004, SBC, São Paulo, 2004, p. 141-183. Disponível em: http://www.ckirner.com/download/capitulos/ARToolKit-CAP6-2004.pdf Acesso em: 15 mar. 2007.

[BARROS,2006] – BARROS, E. N. S. **Curso de Arquitetura de Sistemas Embarcados**. Disponível em: http://www.cin.ufpe.br/~ensb/courses/public_html /index.htm. Acesso em: 22 nov. 2006.

[BEIER,2006] - BEIER K.P., **Virtual Reality: A Short Introduction**. Disponível em: http://www-vrl.umich.edu/intro/index.html. Acesso em: 20 nov. 2006

[BONATO, 2004] – BONATO,V.; MENOTTI,R.; SIMÕES,E.; FERNANDES, M.M.; MARQUES,E. Teaching **Embedded Systems with FPGAs Throughout a Computer Science Course.** In: ISCA 2004 31 st International Symposium on Computer Architecture, 2004, Munich, Germany. Workshop on Computer Architecture Education, 2004. v. 1. p. 8-14.

[BONATO, 2006] – Bonato, Vanderlei; Fernandes, Marcio Merino, Marque, Eduardo. A smart câmera with gesture recognition and SLAM capabilities for mobile robots. International Journal of Electronics, v.93, p.385-401, 2006.

[BORESTE,2006] – Boreste, **Conectividade Inteligente**. Disponível em: http://www.boreste.com/pd_ship.htm. Acessado em: 20. nov. 2006.

[CARDOSO;, 2004] – Cardoso, A.; Lamounier Jr, E. editores - **Realidade Virtual: Uma Abordagem Prática. Livro dos Minicursos do SVR2004**, SBC, São Paulo, 2004. Cap. 06 – Disponível em http://www.ckirner.com/ download/capitulos/ARToolKit-CAP6-2004.pdf. Acesso em: 04 maio 2007.

[CLARKSON,2006] – Clarkson University, def convention, **Virtual Reality Wheelchair Becoming A Reality Thanks To Clarkson Students**. Disponível em http://www.clarkson.edu/news/photos/virtualreality.jpg. Acesso em: 20 nov. 2006.

[COMER, 1998] – COMER, Douglas E. **Interligação em Rede com TCP/IP**. Rio de Janeiro,RJ: Campus, 1998. cap 13,19 e 20.

[CONSULARO,2006] – CONSULARO, L. A.; COELHO, R. C.; FERNANDES, M. M. **Tutorial sobre ARToolKit.** Disponível em: http://www.consularo.com /ARToolKit/. Acesso em: 22 nov. 2006.

[COSTA,2006] – Costa. C. **Projetando Controladores Digitais com FPGA**. São Paulo, SP: Novatec, 2006. cap. 01 - 04.

[COULOURIS,2001] – COULOURIS, G.; DOLLIMORE, J.; KINDBERG, T. **Distributed Systems – Concepts and Design**. Addison-Wesley, 2001. cap 02.

[Datasheet,2007] – Datasheet, **DatasheetCatalog**. Disponível em: http://www.ortodoxism.ro/datasheets2/f/0cdu2ajr763f58gr41r5p9ciz3wy.pdf. Acesso em: 10 out. 2007.

[DAVIS,1991] – DAVIS. W.S. **Sistemas Operacionais – Uma visão Sistemática**. Rio de Janeiro, RJ: Campus, 1991.

[FAZOLO ,2007] – Fazolo, Fazolo Componentes Eletrônicos Ltda. Disponível em: http://www.fazolo.com.br/webapp/interface/site.php?exc=produto.get.166. 30/. Acesso em: 10 out. 2007.

[GRONBCEK,2006] – GRONBCEK, Kaj. Munsters, Disponível em: http://ekot.dk/projekter/AR/. Acesso em: 20 nov. 2006.

[IRAWATI,2006] - Sylvia Irawati, Scott Green, Mark Billinghurst, Andreas Dünser, Heedong Ko: **An Evaluation of an Augmented Reality Multimodal Interface Using Speech and Paddle Gestures**. ICAT 2006: 272-283

[KIRNER, 2006] – TORI, R.; KIRNER, C.; SISCOUTTO, R. **Fundamentos e Tecnologia de Realidade Virtual e Aumentada.** Belém-PA, Livro do Pré-Simpósio, VIII Symposium on Virtual Reality. Editora SBC – Sociedade Brasileira de Computação. 2006. cap 01 – 02.

[KOLSCH,2006] - The HandVu, vision-based hand gesture interface. Disponível em: http://www.movesinstitute.org/~kolsch/HandVu/HandVu.html. Acesso em: 20. nov. 2006.

[MACHADO,2007] – Machado, Liliane dos Santos; Pesquisa e Desenvolvimento de Sistemas de Realidade Virtual para Treinamento em Oncologia Pediátrica. Disponível em: http://www.di.ufpb.br/liliane/rvmed_p.html. Acesso em: 20 set. 2007.

[MESQUITA,2006] – MESQUITA. L. **Projetos de Sistemas Digitais com VHDL**: Dispositivos Lógicos Programáveis CPLD – FPGA. Disponível em: http://dee.feg.unesp.br/Disciplinas/. Acesso em: 18 dez. 2006.

[MICROSOFT, 2007] – Microsoft, **Microsoft Agent download page for end-users**. Disponível em: http://www.microsoft.com/msagent/downloads/. Acesso em: 30 abr. 2007.

[MICROSOFT, 2007] – Microsoft, **Microsoft Agent.** Disponível em: http://www.microsoft.com/msagent/. Acesso em: 13 mar. 2007.

[MINDFLUZ,2006] – Mindfluz, DISCONTINUED PRODUCT. Disponível em: http://www.mindflux.com.au/products/spacetec/sorb360.html. Acesso em: 20 nov. 2006.

[NETO,2002] – NETTO, ANTONIO. V.; MACHADO, LILIANE. S.; OLIVEIRA, MARIA. C. F. **Realidade Virtual – Fundamentos e Aplicações.** Florianópolis, SC**:** Visual Books, 2002. cap 01 – 02.

[OpenSG,2007] – OpenSG, **OpenSG - Open Source Scenegraph**. Disponível em: http://www.opensg.org/download.EN.html. Acesso em: 10 Dez 2007.

[OVIATT,2003] – OVIATT, S.L.; **Advances in Robust Multimodal Interface Design**. IEEE Computer Society Press Los Alamitos, CA, USA, Sept/Oct. 2003. Pages 62-68.

[SILBERSCHATZ,2000] – SILBERSCHATZ. A.; GALVIN. P.; GAGNE. G. **Sistemas Operacionais – Conceitos e Aplicações**. Rio de Janeiro, RJ: Campus, 2000.

[TANENBAUM,2003] – TANENBAUM. A. S. **Redes de Computadores**. Rio de Janeiro, RJ: Campus, 2003.

[UFRGS,2007] UFRGS **-** Universidade Federal do Rio Grande do Sul. **Equipamentos** - Disponível em: http://www.pgie.ufrgs.br/siterv/equipamentos .htm. Acesso em: 05 maio 2007.

[UNIPG,2007] – UNIPG - Universita' Degli Studi di PeruGia. **Programmable logic device** Disponível em: http://www.unipg.it/~clingmat/labs/ClimaticRoom /altera/altera.jpg Acessado em: 30 abr. 2007.

[ZELENOVSKY,2006] – ZELENOVSKY, R.; MENDONÇA, A. **Introdução aos Sistemas Embutidos.** Disponível em: http://www2.eletronica. org/artigos/eletronica-digital/introducao-aos-sistemas-embutidos/. Acesso em: 10 nov. 2006.