ÊSTE EXEMPLAR CORRESPONDE A REDAÇÃO FINAL DA TESE DEFENDIDA POR Alvaro Joffre Uribe Quevedo E APROVADA PELA COMISSÃO JULGADORA EM 01/12/2008

João Mauricio Rosário
ORIENTADOR

UNIVERSIDADE ESTADUAL DE CAMPINAS
FACULDADE DE ENGENHARIA MECÂNICA
COMISSÃO DE PÓS-GRADUAÇÃO EM ENGENHARIA MECÂNICA

Manipulação de Objetos em Ambiente Virtual utilizando uma Garra Antropomórfica acoplada a um Robô Industrial

Autor: **Alvaro Joffre Uribe Quevedo**
Orientador: **Prof. Dr. João Mauricio Rosário**

105/2008

UNIVERSIDADE ESTADUAL DE CAMPINAS
FACULDADE DE ENGENHARIA MECÂNICA
COMISSÃO DE PÓS-GRADUAÇÃO EM ENGENHARIA MECÂNICA
DEPARTAMENTO DE PROJETO MECÂNICO

Manipulação de Objetos em Ambiente Virtual utilizando uma Garra Antropomórfica acoplada a um Robô Industrial

Autor: **Alvaro Joffre Uribe Quevedo**
Orientador: **Prof. Dr. João Mauricio Rosário**

Curso: Engenharia Mecânica
Área de Concentração: Mecânica dos Sólidos

Dissertação de mestrado acadêmico apresentada à comissão de Pós-Graduação da Faculdade de Engenharia Mecânica, como requisito para a obtenção do título de Mestre em Engenharia Mecânica.

Campinas, 2008
S.P. – Brasil

FICHA CATALOGRÁFICA ELABORADA PELA
BIBLIOTECA DA ÁREA DE ENGENHARIA E ARQUITETURA - BAE - UNICAMP

Ur33m

Uribe Quevedo, Alvaro Joffre
Manipulação de objetos em ambiente virtual utilizando uma garra antropomórfica acoplada a um robô industrial / Alvaro Joffre Uribe Quevedo. --Campinas, SP: [s.n.], 2008.

Orientador: João Maurício Rosário.
Dissertação de Mestrado - Universidade Estadual de Campinas, Faculdade de Engenharia Mecânica.

1. Realidade virtual. 2. Robótica. 3. Mecatrônica. I. Rosario, João Maurício. II. Universidade Estadual de Campinas. Faculdade de Engenharia Mecânica. III. Título.

Título em Inglês: Virtual environment object manipulation, using and anthropomorphical gripper with an industrial robot
Palavras-chave em Inglês: Virtual Reality, Robotics, Mecatronics
Área de concentração: Mecânica dos Sólidos e Projeto Mecânico
Titulação: Mestre em Engenharia Mecânica
Banca examinadora: Jean-Paul Frachet, Antônio Batocchio
Data da defesa: 01/12/2008
Programa de Pós Graduação: Engenharia Mecânica

UNIVERSIDADE ESTADUAL DE CAMPINAS
FACULDADE DE ENGENHARIA MECÂNICA
COMISSÃO DE PÓS-GRADUAÇÃO EM ENGENHARIA MECÂNICA
DEPARTAMENTO DE PROJETO MECÂNICO

DISSERTAÇÃO DE MESTRADO

Manipulação de Objetos em Ambiente Virtual utilizando uma Garra Antropomórfica acoplada a um Robô Industrial

Autor: Alvaro Joffre Uribe Quevedo
Orientador: Prof. Dr. João Mauricio Rosário

A Banca Examinadora composta pelos membros abaixo aprovou esta Tese:

Prof. Dr. João Maurício Rosário, Presidente
Universidade Estadual de Campinas

Prof. Dr. Jean-Paul Frachet
ISMEP – SUPMECA, Paris, France

Antônio Batocchio
Universidade Estadual de Campinas

Campinas, 01 de Dezembro de 2008

Dedicatória:

A minha família e todos aqueles que formam parte do circulo da minha vida.

Agradecimentos

A conclusão deste trabalho foi possível graças às seguintes pessoas que estiveram comigo durante esta etapa da minha vida:

Aos meus pais e amigos por o apoio e cuidado prestado desde a distância em tudo momento e pelo incentivo e motivação para continuar com meu crescimento pessoal e profissional.

Ao meu orientador quem acreditou no meu potencial, conhecimentos e experiência para me guiar através deste processo.

Ao pessoal do administrativo do Laboratório de Automação e Robótica LAIR, as secretárias da Pós-graduação e do Departamento de Projeto Mecânico e outros que de alguma prestaram sua colaboração durante este processo.

Adicionalmente agradeço ao CNPQ pelo financiamento prestado durante o desenvolvimento desta dissertação de mestrado.

A boa memória não é aquela que lembra tudo,
é aquela que só esquece o trivial.
Fray Juan Marquez

Resumo

URIBE QUEVEDO, Alvaro Joffre*, Manipulação de Objetos em Ambiente Virtual Utilizando uma Garra Antropomórfica acoplada a um Robô Industrial*.; Faculdade de Engenharia Mecânica, Universidade Estadual de Campinas, 2008. 111 p. Dissertação (Mestrado)

Nesta dissertação de mestrado, são utilizados conceitos de Realidade Virtual para desenvolver um aplicativo de baixo custo e modular, que permita planejar e executar a preensão de objetos num ambiente virtual através do uso de um dispositivo mecatrônico de tipo mão antropomórfica que é a ferramenta de um dispositivo robótico industrial, constituindo assim uma célula robótica cooperativo para o planejamento e realização de tarefas de pick-and-place.

A metodologia proposta para o desenvolvimento desta dissertação é fundamentada nos componentes de um sistema de realidade virtual, enfatizando na interface de usuário e na informação de entrada e saída. A realimentação para o usuário está composta por objetos em três dimensões que representam o dispositivo antropomórfico virtual e a execução da forma de preensão selecionada pelo usuário, e por um arquivo contendo essas informações permitindo assim sua interpretação pelo dispositivo real que realizará a tarefa de preensão desejada através da interação com o robô industrial em uma trajetória planejada para agarrar e liberar objetos baseados nas características antropométricas do dispositivo virtualizado.

Palavras Chave

Realidade virtual, Robótica, Mecatrônica.

Abstract

URIBE QUEVEDO, Alvaro Joffre, *Virtual Environment Object Manipulation Using An Anthropomorphical Gripper Attached To An Industrial Robot*.; Faculdade de Engenharia Mecânica, Universidade Estadual de Campinas, 2008. 111 p. Dissertação (Mestrado)

Virtual Reality concepts are used in this Master's dissertation in order to develop a low-cost and modular application, for planning and executing a desired grasp over some defined objects through a virtual environment with a hand based anthropomorphic mechatronic device attached to an industrial robot as a tool, the integration of the previous components conforms a robotics workcell for picking and placing tasks.

The proposed methodology for developing this dissertation is based upon the components of a virtual reality system, such as the graphics user interface, input and output parameters. The feedback obtained from the developed application is divided in its 3D environmental feed with a virtual device representing the real one performing the chosen grasp, and a file containing that particular information formatted to be executed within a planned trajectory for grabbing and releasing some objects based on the anthropometric characteristics virtualized mechanical hand.

Keywords

Virtual Reality, Robotics, Mecatronics.

Índice

Lista de Figuras

Lista de Tabelas

Lista de Algoritmos

Nomenclatura

Abreviações

3D - Três dimensões
DH - Denavit-Hartenberg
E/S – Entradas e saídas
RV - Realidade Virtual
***.obj** - Extensão de arquivo para o OBJ
***.wrl** - Extensão de arquivo para o VRML

Siglas

API - Application Program Interface
CAD - Computer Aided Design
CAE - Computer Aided Engineering
CAM - Computer Aided Manufacturing
CCD - Charge Coupled Device
CRT - Cathode Ray Tube
GUI - Graphics User Interface
HMD - Head Mounted Display
HTML - Hypertext Markup Language
IDE - Integrated Development Environment
IO - Input, Output
kbps - kilobits per second
LCD - Liquid Crystal Display
MEMS - Micro Electro Mechanical System
OBJ - 3D object file extension
RGB - Red Green Blue
TCP/IP - Transmission Control Protocol/Internet Protocol
usb - Universal Serial Port
VRML - Virtual Reality Modeling Language

Capitulo 1
Introdução

Os avanços tecnológicos e a pesquisa fundamental têm permitido a criação de dispositivos baseados nos membros de um ser humano que atendam as suas principais funcionalidades. Estes são desenvolvidos para aplicações específicas e por isso não são construídos em grandes quantidades o que faz que cada um tenha diferentes propriedades para realizar tarefas segundo seu desenho. Dentre esses sistemas, podemos destacar os dispositivos do tipo antropomórfico baseados na mão humana os quais podem se beneficiar da realidade virtual e suas aplicações na robótica como são o planejamento de trajetórias, programação *offline*, e disponibilização de um ambiente virtual semelhante ao real para realizar praticas quando planta física não esteja disponível.

1.1. Apresentação do problema

Tendo em conta as aplicações da Realidade Virtual na robótica, tomou se uma célula robótica composta por um robô industrial de 6 graus de liberdade e um dispositivo antropomórfico baseado em mão humana com cinco dedos e 6 graus de liberdade. No nível operacional uma das limitantes do sistema é a dependência do hardware (componentes eletrônicos, mecânicos ou os dois dispositivos) para realizar práticas ou treinamento gerando assim uma restrição no uso da célula.

A utilização dos conceitos de Realidade Virtual representa um avanço na solução deste problema, permitindo disponibilizar parcial ou totalmente a célula robótica em um ambiente virtual, fornecendo informação compatível para a execução de tarefas nos componentes reais, permitindo ainda a integração dos sistemas e acréscimo de capacidade e disponibilidade de uso da célula.

1.2. Objetivos do trabalho

Desenvolver um protótipo de aplicação baseado em software para operar uma garra (gripper) antropomórfica em um ambiente virtual através da preensão de geometrias básicas e execução destas em uma planta física composta por um robô industrial e um dispositivo antropomórfico. Conseqüentemente, os seguintes objetivos deverão ser atingidos no final desse projeto de pesquisa.

- Analisar o dispositivo antropomórfico através de um modelo cinemático.
- Criar um ambiente virtual com objetos 3D criados em *software* de CAD e modelagem 3D.
- Desenvolver um protótipo experimental de *software* de baixo custo para realizar preensões de tipo esféricas, planar e cilíndrica.
- Validar experimentalmente o protótipo da aplicação através do envio de informação para a integração com a planta física composta pelo robô industrial e dispositivo antropomórfico.

1.3. Abordagem do problema

O desenvolvimento desse trabalho de pesquisa é realizado através da abordagem de três componentes principais:

a) Estudo da cinemática dos dispositivos a serem integrados no ambiente virtual,
b) Implementação de um ambiente virtual para operação colaborativa desses dispositivos,
c) Realização de testes experimentais através de célula automatizada composta por um robô industrial e uma garra de tipo antropomórfico (Fig. 1.1).

Para a abordagem destes três componentes, serão utilizadas informações provenientes do funcionamento do dispositivo antropomórfico, propriedades da ferramenta para o desenvolvimento da aplicação de manipulação de objetos num ambiente virtual e características da célula robótica para finalmente realizar a integração do ambiente virtual com o real.

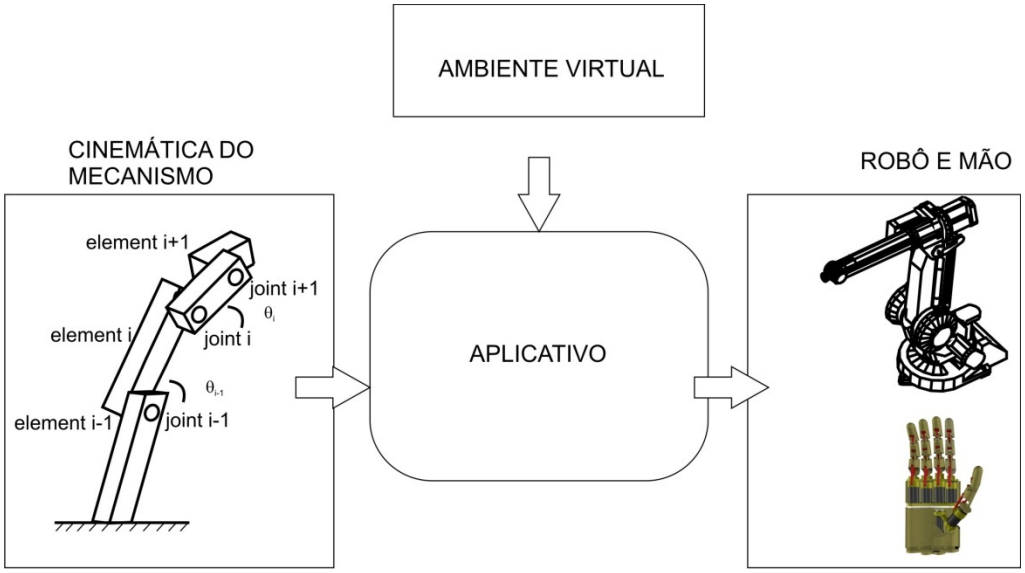


Fig. 1.1: Elementos para abordar do problema.

Para atingirmos os objetivos delineados anteriormente, esta dissertação de mestrado foi desenvolvida a partir dos seguintes capítulos:

O capitulo I apresenta a introdução, apresentação do problema, os objetivos e a abordagem proposta para o desenvolvimento da dissertação.

O capitulo II apresenta a revisão de conceitos básicos na área de Realidade Virtual, permitindo assim contextualizar o leitor ao problema em estudo.

O capitulo III apresenta a revisão bibliográfica, o qual foi abordada em ordem cronológica e seguindo os trabalhos mais representativos que integram a robótica serial (tipo robô industrial e antropomórfico baseados em mãos) com realidade virtual.

O capítulo IV apresenta a metodologia proposta para o desenvolvimento deste projeto de pesquisa, onde as seções estão divididas na análise da cinemática da mão humana, estudo e definição das tecnologias a utilizar para a criação do aplicativo, edição do ambiente 3D e interface de usuário e finalmente a configuração da saída para o envio de informação para a planta física através das ferramentas escolhidas.

O capítulo V sintetiza os principais resultados obtido, após a realização de testes em ambiente virtual, enfatizando desde a compatibilidade com arquivos VRML criados desde diferentes softwares de projeto. As interfaces de saídas foram testadas: a navegação e o arquivo de texto plano para o envio de informação para a planta física.

Finalmente, no último capítulo são apresentadas as conclusões do trabalho, obedecendo à metodologia proposta inicialmente. Também são apresentadas algumas propostas para trabalhos futuros baseadas na informação e conhecimento adquirido durante o desenvolvimento deste trabalho.

Capitulo 2
Conceitos Básicos

Para contextualizar ao leitor através da leitura deste documento, alguns conceitos básicos são apresentados neste capítulo. Para a área de realidade virtual, aspectos históricos, noções básicas e áreas de aplicação aplicações são abordadas, já que o dispositivo virtual do aplicativo proposto é de tipo mão antropomórfica, as características básicas sobre a mão humana como graus de liberdade e movimentação são descritos.

2.1. Realidade Virtual

A realidade virtual permite aos usuários interatuar com ambientes virtuais através de uma interface computacional em tempo real. O principal objetivo da realidade virtual é criar uma simulação, na qual as imagens criadas em computador são utilizadas para obter um ambiente realista que responde às entradas do usuário.

Para obter-se sucesso no desenvolvimento de um sistema utilizando o conceito de realidade virtual torna-se imprescindível o grau de interatividade por parte do usuário e ambiente de imersão. Estas duas propriedades interagem entre si, e no caso de ocorrência de falha de alguma delas, a outra também será afetada.

As imagens geradas através do computador podem ser utilizadas de forma completa ou parcial, de acordo com a forma o nome da aplicação se altera, conforme pode ser observado na

Fig. 2.1. Conseqüentemente, no caso de um ambiente ser completamente 3D, será designado de realidade virtual, enquanto se tivermos um ambiente virtual com objetos reais, é virtualidade aumentada, e se tiver um ambiente real com objetos virtuais, é realidade aumentada, como pode ser observado na Fig. 2.1.

Fig. 2.1: Do ambiente virtual para o real.

A história da realidade virtual começou em 1962 quando Morton Heiligh inventou o Sensorama, cujo principio de funcionamento era baseado num vídeo 3D que foi obtido com a filmagem de duas câmaras uma junto a outra, som estéreo, aromas, efeito de vento e uma cadeira móvel, entretanto nessa época a idéia teve pouca acolhida. Este trabalho abriu o caminho para o desenvolvimento dos HMD e Heilig realizou os primeiros projetos, mas foi Ivan Sutherland quem trabalhando com CRT programou o primeiro HMD, descobrindo as possibilidades de utilizar imagens geradas através de computador para alimentar aos CRT. Nessa ocasião, Sutherland afirmou que num futuro o sentido do tato seria necessário para que os usuários tomassem objetos virtuais. Em 1971 Frederick Brooks Jr. e colegas da Universidade do Norte de Carolina simularem forças no momento de colisão de um braço robótico, trabalho que serviu como referencia para a tecnologia háptica atual.

Nos anos 70 e 80 o maior desenvolvimento da realidade virtual foi apresentado pela necessidade de redução de custos dos simuladores de vôos semelhantes, pois quando o modelo de uma aeronave era descontinuado, o simulador também era descontinuado, assim um simulador implementado com imagens geradas por computador permitiu o desenvolvimento de um sistema mais econômico.

Conforme a tecnologia de computação foi avançando a realidade virtual começou a alcançar novas aplicações em diferentes áreas como:

- **Engenharia**
  - Software de CAD, CAM e CAE;
  - Tele-operação (utilização de robôs em ambientes perigosos);
  - Simulação (método dos elementos finitos);
  - Treinamento para utilização de maquinas, equipamentos e ferramentas dedicadas;
  - Prototipagem rápida
- **Medicina**
  - Planejamento de cirurgias
  - Diagnostico de imagens através de reconstrução 3D
  - Tele-cirurgia
  - Treinamento
- **Entretenimento**
  - Vídeo games
  - Filmes
- **Arquitetura**
  - Avaliação de desenhos
  - Impacto no projeto na área real
  - Visitas (tour) em Ambientes virtuais
- **Publicidade**
- **Educação**
  - Aplicações interativas
  - Ensino a distancia
- **Militar**
  - Treinamento
  - Planejamento
- **Aviação**
  - Simuladores de vôo
  - Simuladores de controle e trafego aéreo
  - Treinamento para manutenção de equipamentos

De acordo com o desenvolvimento e as necessidades de cada aplicação foram criados diferentes dispositivos para interagir com os ambientes virtuais, conforme pode ser observado na Fig. 2.2.

Normalmente numa aplicação podemos utilizar diferentes dispositivos ao mesmo tempo, e eles podem ser utilizados classificados da seguinte forma:

.

- **Entradas**
  - *Mouse*
  - *Joystick*
  - Baseados na mão
    - Exoesqueleto

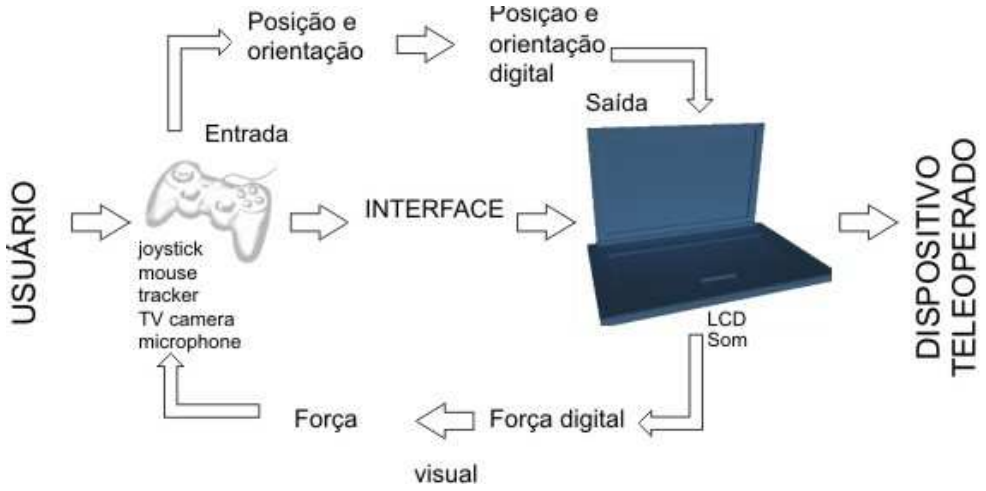


Fig. 2.2: Dispositivos de um sistema de realidade virtual.

- **Saídas**
  - Visuais
    - Projetores
    - LCD
    - HMD
    - *Shutter glasses*
    - Anaglifo
    - *Óculos Polarizados*
  - Auditivos
  - Dispositivos com realimentação de força (*Force Feedback)*

2.2. Tele-operação

A área da realidade virtual de forte interesse para o desenvolvimento deste trabalho de pesquisa é a tele-operação robótica, a qual é definida como as técnicas utilizadas para manipular objetos e efetuar operações a distancia, são diferentes da tele-presença, pois na segunda o usuário está imerso no ambiente de trabalho com ajuda de dispositivos visuais da realidade virtual. Existem dois tipos de tele-operação:

- Independente: onde o dispositivo robótico tele-operado devera realizar uma tarefa bem conhecida e definida.
- Dependente: onde o sucesso da tarefa está relacionado, entre outros fatores, à experiência, habilidade e idade do usuário.

Nos sistemas supervisores a relação homem-máquina é feita de forma que a pessoa só participa do processo quando se requere fazer correção de uma situação imprevista. O nível de interação do operador depende da autonomia do sistema que pode ir desde somente supervisão ao controle manual.

Conforme pode ser observado na
Fig. **2.3**, os sistemas de tele-operados são compostos de:

- Operador ou tele-operador: Pessoa que faz a operação do dispositivo remoto seja de forma continua ou intermitente.
- Dispositivo tele-operado: dispositivo que executa uma determinada tarefa num ambiente remoto.
- Interface: Conjunto de dispositivos que permitem ao operador interatuar com o sistema, seja através de *software* ou *hardware*.
- Controle e canais de telecomunicação.
- Sensores: Dispositivos para tomar a informação da zona remota como local.

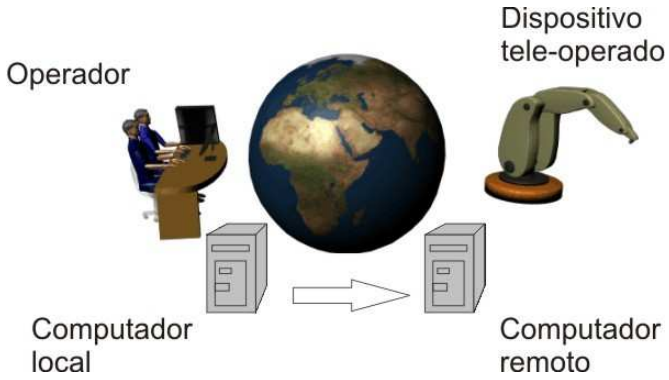


Fig. 2.3: Dispositivos integrante de um sistema tele-operado.

A pesquisa sobre tele-operação começou com a manipulação de materiais radioativos. O primeiro robô tele-operado mecanicamente foi construído em 1948 (GRANT et al., 2004) onde um manipulador controlado por uma pessoa enviava os movimentos que eram feitos pelo outro manipulador, mas foram uns anos depois que os manipuladores tiverem acionamento elétrico.

Nas aplicações atuais de tele-operação têm permitido muitos avanços, principalmente ao alto grau de desenvolvimento de tecnologia dos sistemas telecomunicações, automação industrial e robótica, eletrônica, dispositivos hápticos (BURDEA, 2003) (KARTOUN; STERN; EDAN, 2004) e realidade virtual, que permitem maior imersão, precisão e controle sobre o robô tele-operado.

Esta tecnologia já é muito utilizada nas mais diversas aplicações, desde na indústria nuclear, como também para outros setores começarem a ser objetivo desta tecnologia, sobretudo onde as tarefas a serem desenvolvidas não possam ser realizadas através de pessoas, seja por segurança ou por serem realizadas em ambientes perigosos para a saúde humana.

Algumas aplicações de sistemas tele-operados podem ser encontradas nas seguintes áreas da Aeronáutica, Espaço, Nuclear, Militares, Submarino, Terrestre, Medicina (SRIVSAN M. A., 1997) (JADHAV, 2004) e Engenharia (SEMERE; KITAGAWA; OKAMURA, 2004).

2.3. Descrição da mão humana

A mão humana é composta de 26 ossos (Fig. 2.5), dois grupos de músculos (os intrínseco e os extrínsecos), os quais em conjunto geram 20 graus de liberdade (FIELD; PALASTANGA; SOAMES, 2001), criando uma ferramenta multi-propósito capaz de desenvolver tarefas que precisem desde uma boa precisão até força. A Fig. 2.4 apresenta os movimentos que podem ser efetuados pelos dedos e sua classificação.

- **Flexão e extensão:** Movimento realizado pelo dedo através da rotação da sua falange proximal sobre a cabeça do osso metacarpal para ir da posição de mão aberta à fechada. No caso do dedo indicador, a rotação não excede os 90 graus e esta aumenta só um pouco até o dedo mínimo. Dependendo de cada pessoa, para o caso do movimento de extensão a rotação, a mesma pode ir até aproximadamente 50 graus em forma ativa (movimento realizado através dos músculos) e até 90 graus de forma passiva (movimento realizado sem a utilização dos músculos)(Fig. 2.6 a).

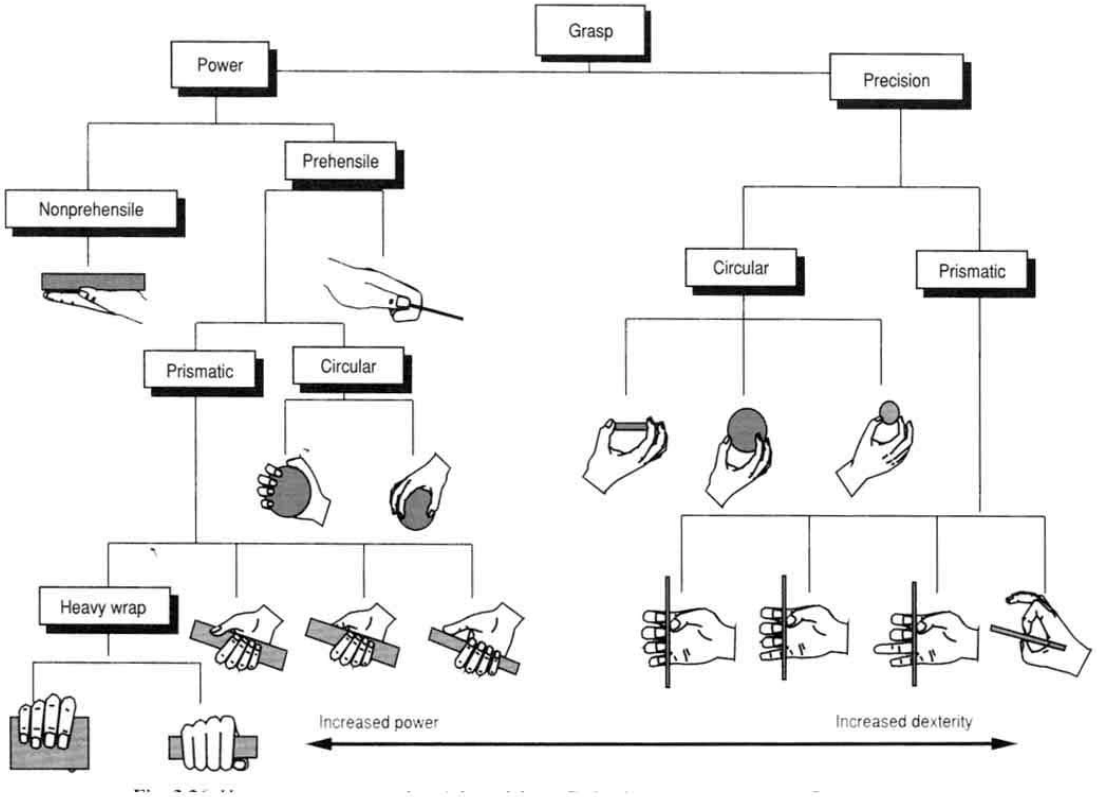


Fig. 2.4: Formas de preensão de uma mão humana. (BURDEA, 2003)

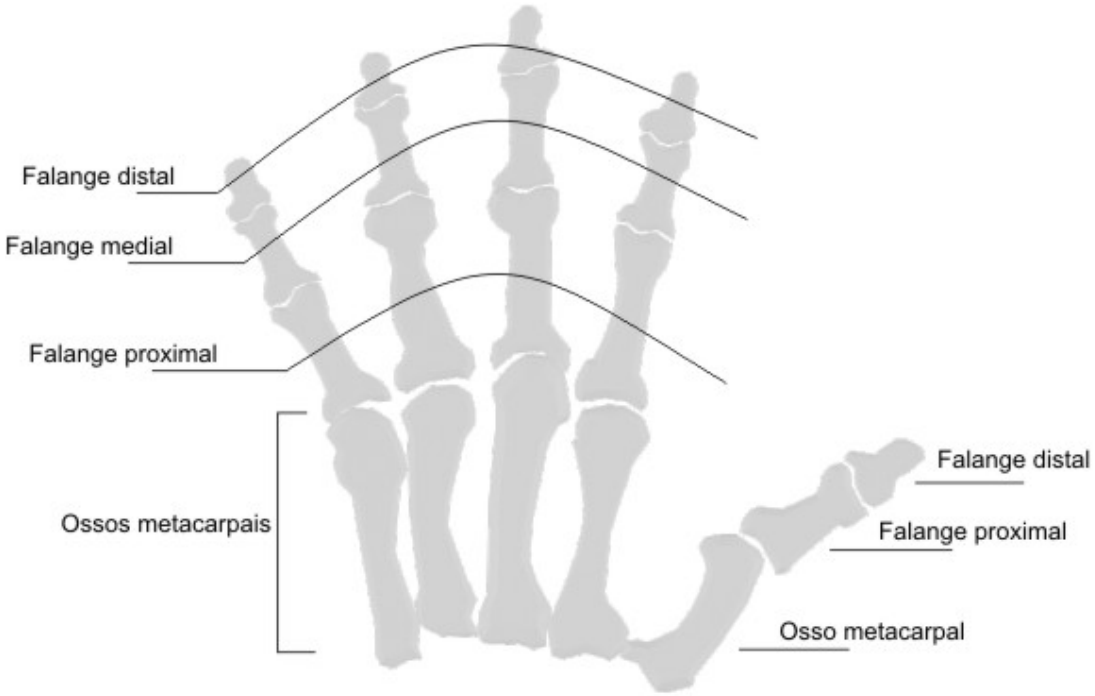
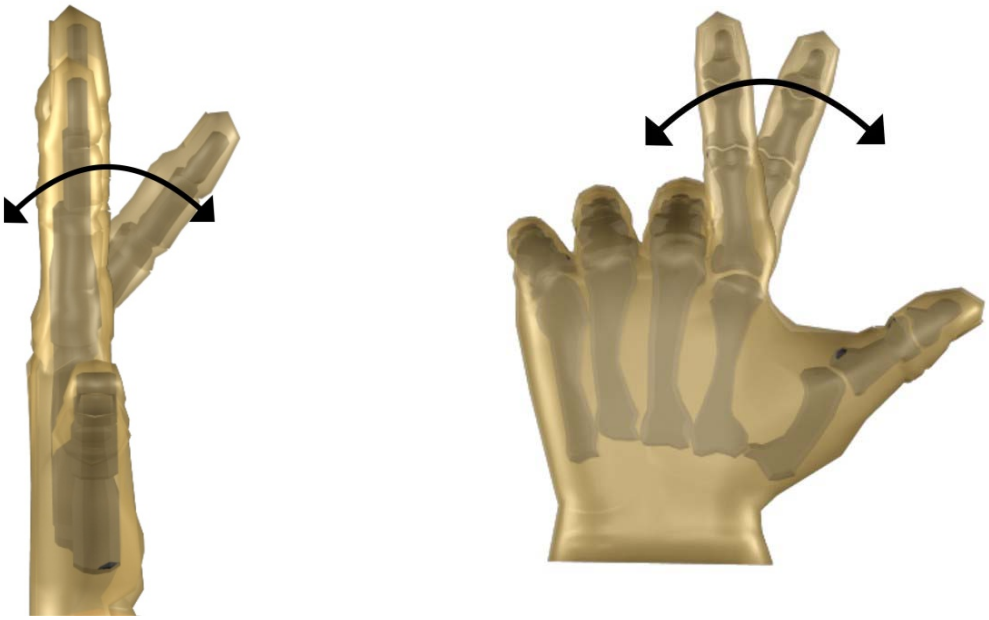


Fig. 2.5: Ossos da mão humana.

- **Abdução e adução:** Neste movimento a rotação sobre o osso metacarpal é realizada de forma que os dedos efetuem um movimento de rotação na direção do dedo médio e na direção contrária, as rotações são de até 30 graus para cada dedo em sua posição estendida, e na posição fechada não é possível mais do que 10 graus (Fig. 2.6 b).



a) Flexão e extensão (esquerda). b) Abdução e adução (direita).

Fig. 2.6 Movimentos do dedo.

Após o levantamento das características da mão, direcionando estas para preensão de dispositivos ou objetos, as mesmas adquirem a propriedade de antropomorfismo, conseqüentemente as mãos antropomórficas procuram imitar as funcionalidades de uma mão humana.

Capitulo 3
Revisão da Literatura

A utilização cada vez mais freqüente de ambientes virtuais em Automação e Robótica tem permitido embarcar funcionalidades para ajudar o desenvolvimento de dispositivos automatizados. No momento que estas ferramentas direcionadas a Realidade Virtual começaram a serem disponibilizadas e acessíveis para as mais diversas aplicações, despertando um forte interesse de pesquisadores de todo o seu potencial, e os primeiros passos foram tomados para aproveitar esta nova tecnologia, na qual as aplicações desenvolvidas podem ser classificadas como *offline* e *online*.

3.1. Realidade virtual e robótica

Dentro desse contexto a utilização de conceitos de Realidade Virtual pode ser considerada de grande diversidade, em função das necessidades específicas para cada aplicação. Inicialmente, o padrão VRML2.0 ou 97 foi estudado por (HIRUKAWA et al., 1997) para o desenvolvimento de um protótipo de sistema de tele-operação, sendo observado nesse estudo algumas limitações que não permitiam ao VRML ser uma solução completa e independente para uma aplicação virtual pelas limitações de interatividade e da cinemática. O problema foi abordado fazendo uso de software que deverá ser utilizado para complementar o VRML. Neste caso o uso da linguagem Java para desenvolvimento de interface remota com o usuário permite implementar a

modelagem cinemática associada à aplicação e o HTML, conforme podemos observar através da configuração apresentada na Fig. 3.1.

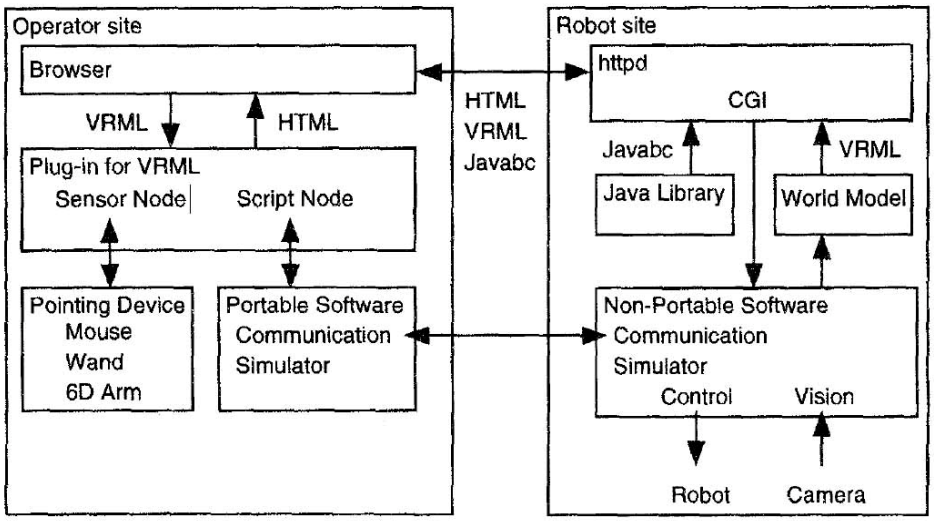
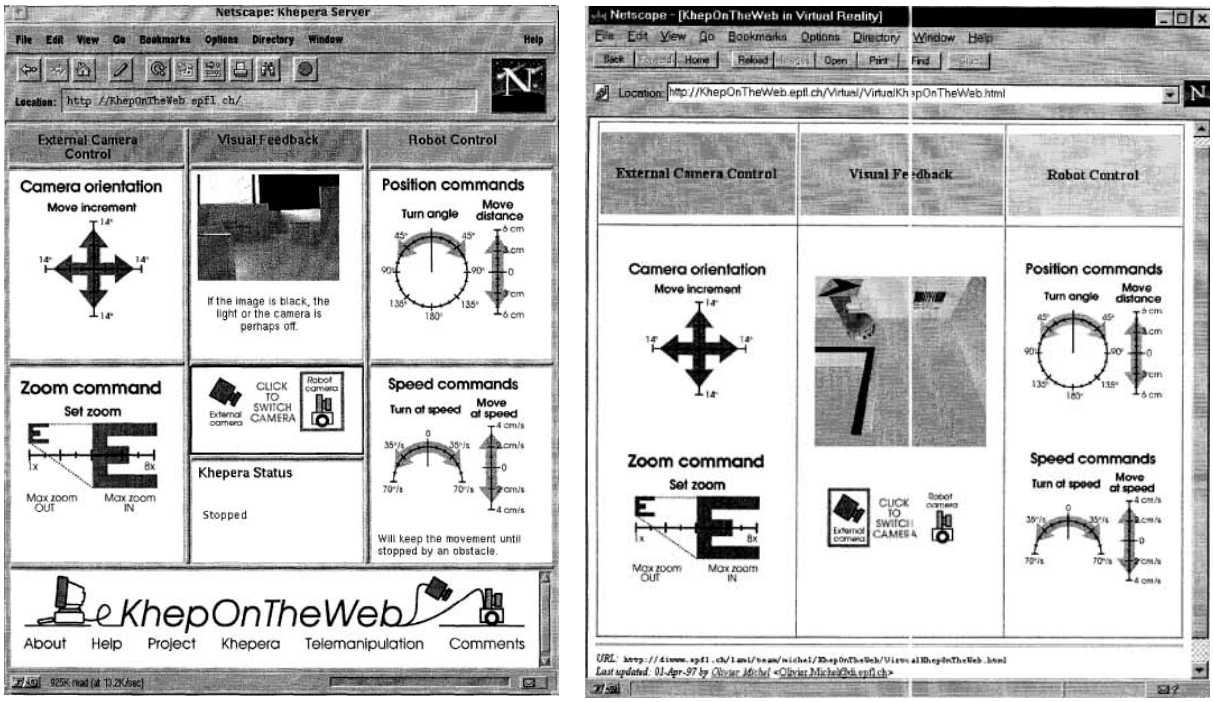


Fig. 3.1: Configuração do sistema. HIRUKAWA et al., 1997.

No campo da robótica móvel (MICHEL; SAUCY; MONDADA, 1997) faz uma proposta levando em consideração os altos custos envolvidos no desenvolvimento de plataformas físicas para aplicações experimentais associadas, o fato de que nem todo usuário tenha acesso a mesma, justificam a utilização cada vez mais freqüentemente dos conceitos de Realidade Virtual para supervisão e controle de dispositivos robóticos móveis. Como o ambiente de trabalho é um labirinto, o VRML foi escolhido em função de sua grande flexibilidade, permitindo assim a obtenção de um modelo 3D próximo ao modelo real.

Ao mesmo tempo, a configuração do comportamento do sensor de tempo de VRML permitirá obtermos uma simulação próxima a uma aplicação implementada em tempo real. Com relação aos sensores de posição, os mesmos não foram utilizados, pois os mesmos são relativos à câmera ativa do VRML, assim direcionou-se a implementação de uma interface para o usuário do robô móvel contendo um sistema de detecção de posicionamento que utiliza as suas coordenadas de posição e considera os valores de posicionamento não permitidos ao mesmo. Nesse sistema foi utilizada a linguagem Java para enviar eventos entre os controladores e o ambiente virtual, ambos inseridos em HTML (Fig. 3.2), tendo como principal resultado a

possibilidade de disponibilização através da internet, uma aplicação de operação de um robô móvel.



a) Aplicação real **b) Aplicação Virtual**

Fig. 3.2: Interface para o controle remoto. MICHEL; SAUCY; MONDADA, 1997.

Outro campo de dispositivos robóticos, o desenvolvimento de uma tele-operação micro-manipulada para dispositivos MEMS foi desenvolvida por (ALEX; VIKRAMADITYA; NELSON, 1998), foram utilizadas as mesmas ferramentas disponibilizadas para implementação de um ambiente virtual e aplicadas a uma estação de micro-montagem. Entretanto, para esta aplicação foi adicionado um espaço de trabalho que permitisse a interatividade do usuário com o modelo real, recebendo uma realimentação em ambiente virtual (Fig. 3.3). Neste caso a linguagem Java não foi apenas utilizada para enviar eventos ao VRML, como também para implementação de uma interface de comunicação com o micro-ambiente.

Conhecidas as vantagens da Realidade Virtual em relação a robótica, (AFSHARI; PAYANDEH, 1999) foram estudados três formas nas quais o VRML e linguagem Java poderiam ser utilizadas para desenvolvimento de um software para o treinamento, planejamento de

trajetórias e tele-operação. Assim a proposta foi direcionada para tele-operação e utilização da internet, onde uma análise sobre diferentes tecnologias para visualização de mundos virtuais foi realizada, enfatizando a necessidade de termos um sistema que não dependa de uma determinada plataforma, e conseqüentemente, a melhor solução encontrada foi à utilização da linguagem Java. O sistema físico era composto de um manipulador serial com quatro graus de liberdade projetado inteiramente em linguagem Java (Fig. 3.4).

Fig. 3.3: Interface supervisora para micro-manipulação. ALEX; VIKRAMADITYA; NELSON, 1998.

Da mesma forma que o trabalho apresentado anteriormente (BELOUSOV; TAN; CLAPWORTHY, 1999) propôs a realização de tele-operação através da utilização da *internet*, onde procurou-se solucionar um dos principais inconvenientes apresentados na utilização da *web*, a realimentação da manipulação do robô realizada através de câmeras, associada a uma largura de banda muito baixa. A solução proposta neste trabalho foi a escolha da linguagem Java, de modo a não ter problemas de compatibilidade através da *web*. O manipulador utilizado foi um robô PUMA com seis graus de liberdade com modelo implementado em Java, juntamente com seus controladores e *applets* para *internet* (Fig. 3.5). Devido às limitações de velocidade para

esta largura de banda, a geometria do robô foi trabalhada de forma especial, de modo a obter 11 quadros de imagem por segundo durante a utilização dessa ferramenta. Os experimentos realizados permitiram avaliarmos a versatilidade desta ferramenta para realização de tarefas nos modos *online* e *off-line*.

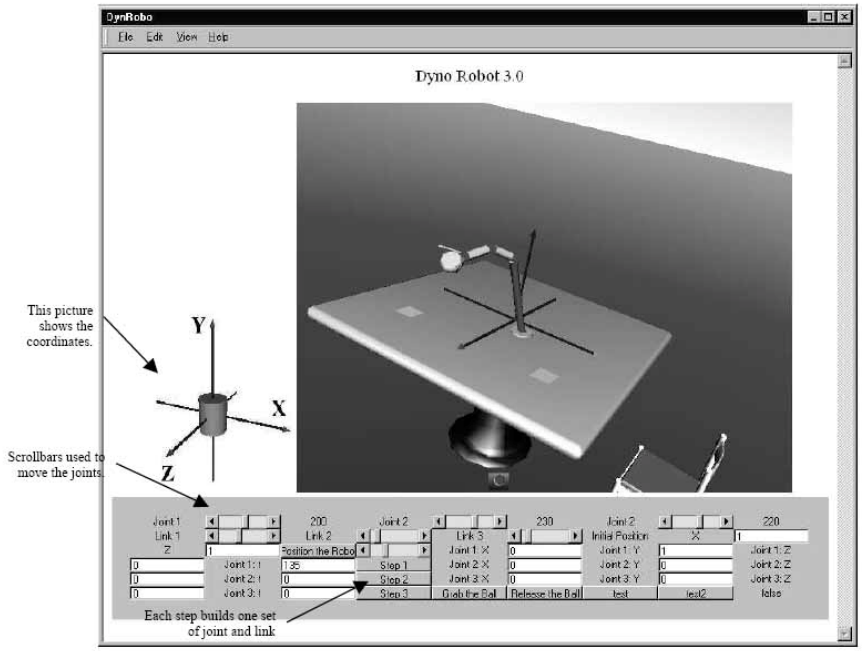


Fig. 3.4: Interface Dyno robot. AFSHARI; PAYANDEH, 1999.

Continuando com a procura de melhores soluções, a implementação de uma aplicação para evitar os problemas com a largura de banda em tele-operadas que fazem a utilização do vídeo, (BELOUSOV; CHELLALI; CLAPWORTHY, 2001) escolhe um grupo de tecnologias para utilizar conceitos de Realidade Virtual aplicados a tele-robótica. Esta arquitetura foi desenvolvida em três etapas:

- Robô: Dois tipos de robôs com seis graus de liberdade foram utilizados para esta aplicação: o robô PUMA 650 e o robô CRS A465.
- Servidor: Equipado com uma câmera.
- Cliente: Equipe com uma luva digital (*data-glove*) com seis graus de liberdade que permitia o controle destes manipuladores.

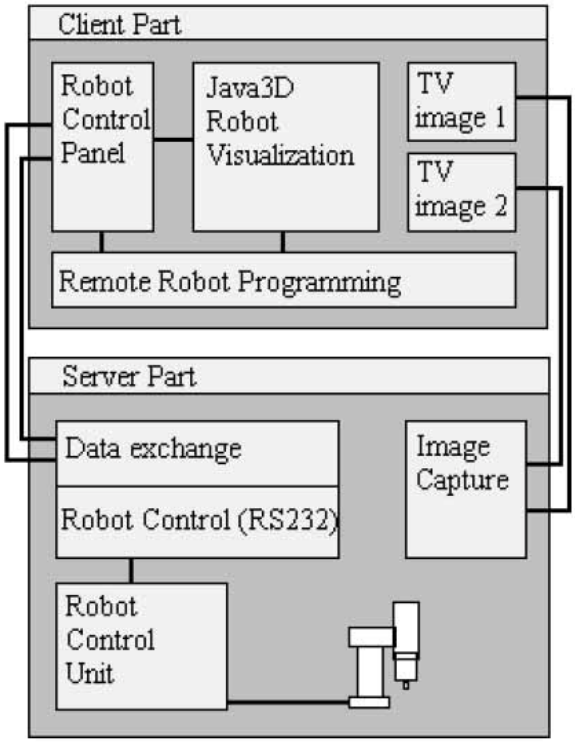


Fig. 3.5: Arquitetura do sistema. BELOUSOV; CHELLALI; CLAPWORTHY, 2001.

Para o funcionamento independente deste aplicativo, foi utilizado o sistema operacional Java3D, que permite a execução de um programa ou uma aplicação para ser inserida num HTML. Para as interfaces de comunicação foram utilizadas as portas RS232 e o protocolo TCP/IP. Para dar realismo ao ambiente 3D, uma câmera virtual foi modelada de forma a ficar semelhante ao sistema real, e através de realidade aumentada as cenas puderam ser comparadas. Após o desenvolvimento dos ambientes virtuais foram realizados testes que mostraram as inúmeras vantagens da utilização da Realidade Virtual para o aproveitamento da largura de banda.

Com o crescimento no uso dos ambientes gráficos através do computador e seu impacto na manufatura, um sistema robusto e escalonável foi proposto por (HAAGE; NILSSON, 1999). O sistema descrito na Fig. 3.6, apresenta a utilização de uma plataforma implementada em Java, que após ser comparada com outras tecnologias como C++, apresentou inúmeras vantagens tais como: uma grande flexibilidade de manipulação da memória, muito importante numa aplicação

dessa natureza, que necessita de execução em tempo próximo ao real, e ainda as características gráficas que poderão ser ajustadas aos objetivos do projeto.

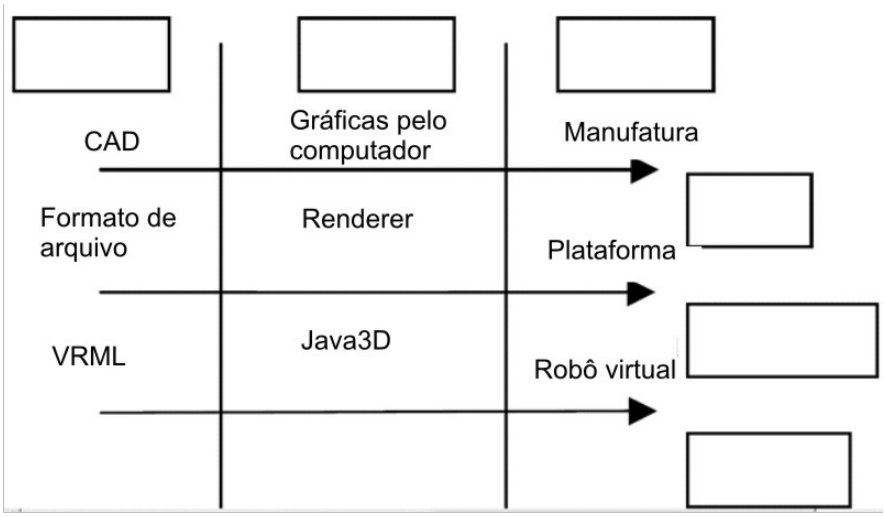


Fig. 3.6: Sistema descrito em etapas. HAAGE; NILSSON, 1999.

É importante destacar que este tipo de solução, aproveitou que os softwares de CAD permitissem exportar as geometrias como arquivos VRML, e ainda que os modelos 3D já estivessem prontos, e ainda o fato que naquele momento de estudo, não eram disponíveis formas de carregamento de VRML no ambiente Java e por isso a necessidade de desenvolvimento de um programa para realizar este tipo de operação. Testes realizados encapsulando o VRML em Visual Basic e Java não apresentaram diferenças de comportamento permitindo a obtenção de uma solução escalonável.

Superada a barreira da dependência de software implementado em Java, um sistema para visualizar um robô foi desenvolvido por (SPECK; KLAEREN, 1999), utilizando Java como parte essencial com o sistema de supervisão e controle. O software proposto utiliza as linguagens Java1.1 e o Java3D se comunicando diretamente com uma planta industrial de controle, permitindo assim a obtenção de uma implementação com maior realismo. Através destas características foi possível o desenvolvimento de aplicações a serem utilizadas de modo local, na intranet ou diretamente na *internet* direcionadas ao treinamento em educação e pesquisa, sistemas colaborativos entre os ambientes virtuais e reais, e programação *offline*.

Através de estudos comparativos realizados com programas disponíveis no mercado, observa-se que estas aplicações são totalmente implementadas numa máquina e posteriormente esta informação é transferida para o robô, permitindo assim que a ferramenta desenvolvida apresente a vantagem de trabalhar diretamente sobre o controle do manipulador, tendo como resultado um comportamento mais próximo do real. Para o estabelecimento da comunicação do ambiente virtual com o real, uma arquitetura de camadas é trabalhada de forma a obtermos as seguintes camadas: dispositivo e simulação, comunicação, aplicações de tarefas e interface do usuário. Na época que as versões da linguagem Java foram comparadas, com a linguagem Java3D ainda se encontrava em fase de desenvolvimento, estando disponível somente em determinados sistemas operativos, direcionado a resultados de compatibilidade melhores na utilização do Java1.1, mesmo que as capacidades visuais e de realismo fossem bem melhores em relação ao Java3D.

Enquanto que a maioria dos estudos e pesquisa abrange a interoperabilidade e realismo da ferramenta, (SONG; KABER, 2000) com o objetivo de analisar desde a perspectiva da implementação do sistema até fatores humanos, foram desenvolvidos trabalhos do projeto de uma interface para tele-operação baseada na *web*. Através da análise de algumas ferramentas disponíveis para tele-robótica, encontramos uma possibilidade de integração entre o operador e o sistema tele-robótico, sendo para isto importante que os sistemas utilizados levem em consideração as capacidades de processamento da informação do ser humano. Conseqüentemente, a partir da tele-operação realizada através de realimentação de vídeo, notou-se que a mesma poderia apresentar alguns inconvenientes devido às limitações da sua largura de banda, comprometendo assim o sistema.

Dentre as diversas soluções propostas, podemos contemplar o aumento do range de quadros por segundo, que poderão ser alcançados através da diminuição da resolução de vídeo. Para o protótipo desenvolvido (Fig. 3.7), um robô PUMA foi utilizado, e dentre os aspectos mais importantes a serem observados nesta aplicação, encontram-se a possibilidade de observação através de vários pontos de vista, a implementação de janelas com os sistemas coordenados e de botões para movimentação das juntas. Os principais objetivos deste protótipo foram não somente

a apresentação de inúmeras possibilidades para a representação do projeto, como também a apresentação de diferentes vantagens a nível operacional e econômico deste tipo de ferramenta.

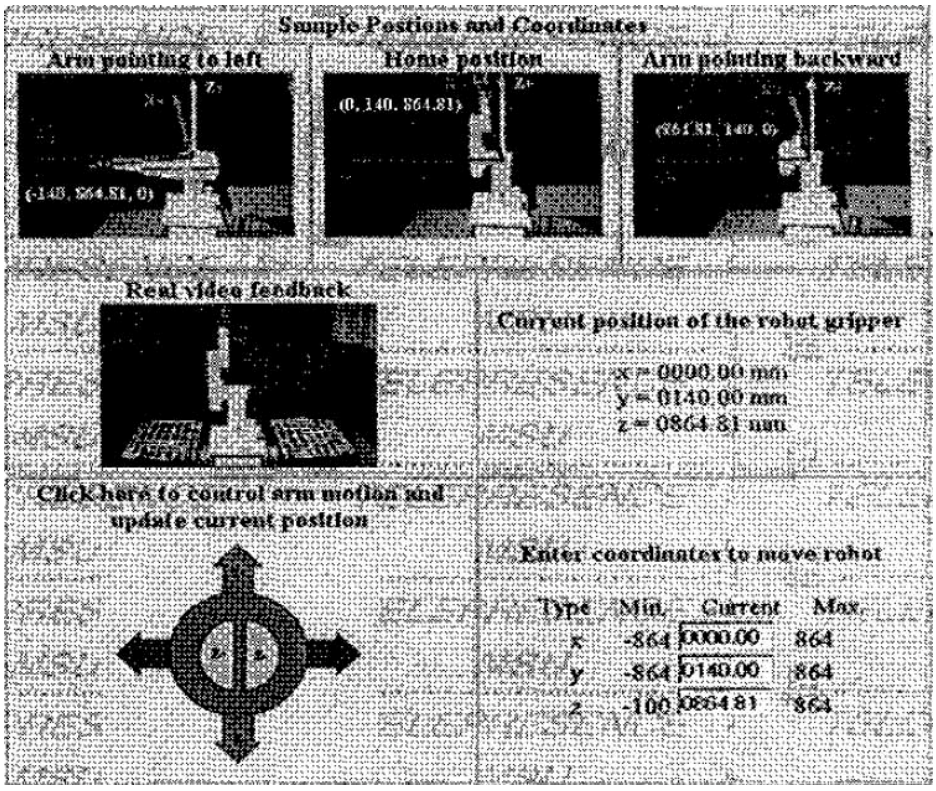


Fig. 3.7: Protótipo de aplicação. SONG; KABER, 2000.

3.2. Ensino

Ante as necessidades apresentadas anteriormente, (SAFARIC et al., 2001), fundamenta sua proposta de ambiente virtual para robótica, considerando o fato de que nem todas as instituições de ensino e pesquisa possuem recursos e condições econômicas de adquirir equipamentos para experimentos na área de robótica. Os primeiros testes foram realizados utilizando um servo-motor de corrente contínua, onde a aplicação principal encontra-se num *applet* de Java e a interface com o usuário escrita em linguagem HTML, permitindo assim configuração e controle deste dispositivo.

Com o objetivo de implementação de um Laboratório Virtual de Robótica, o VRML foi analisado, onde sua principal desvantagem foi que utilizando o mesmo, não se poderiam ser detectadas colisões entre objetos, o qual é um requisito indispensável na hora de evitar a possibilidade das mesmas numa aplicação em ambiente real. Duas formas de solucionar o problema encontram-se disponíveis, uma através de Java e outra utilizando bibliotecas desenvolvidas em C++. Para testar esta plataforma (Fig. 3.8), uma série de atividades foi definida para serem desenvolvidas pelos estudantes.

A partir da definição de uma ferramenta, os testes mostraram que para usuários dentro de um campus, um laboratório virtual poderia ser utilizado sem grandes problemas a partir de computadores clientes. Entretanto os usuários domésticos poderiam encontrar alguns problemas na utilização em tempo real, devido às limitações da largura de banda (Máxima de 28.8 kbps). Para esta situação um procedimento que foi adotado, de modo a conseguir um desempenho aceitável, foi a implementação de uma cópia do ambiente virtual nas máquinas utilizadas pelas equipes que estivessem fora da rede local do laboratório. A experiência adquirida pelos estudantes foi avaliada através da implementação de um laboratório virtual para posteriormente ser levado a uma plataforma real.

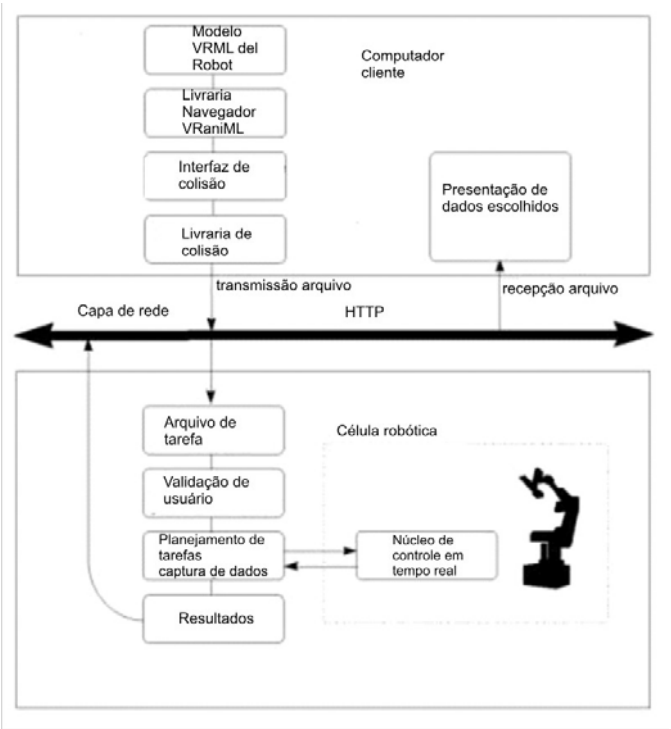


Fig. 3.8: Interface cliente - servidor. SAFARIC et al., 2001.

Considerado como um dos pioneiros em aplicações para educação, (FREIRE et al., 2004) desenvolveu um ambiente multimídia para o ensino de sistemas robóticos procurando uma solução mais econômica. O propósito desta ferramenta foi apresentar um robô aos estudantes de engenharia sem entrar nos detalhes no seu funcionamento interno. Dentre as opções disponíveis que permitissem uma representação gráfica com baixo custo envolvido e que ao mesmo tempo fossem independentes do sistema operativo, somente o VRML e Java atendiam tais exigências. Entretanto o primeiro depende da estabilidade dos *plugins*, enquanto que o segundo apresentou muitas vantagens que fizeram dele a linguagem escolhida para o desenvolvimento desta ferramenta, pois permitia o funcionamento independente da plataforma, com uma boa manipulação de memória já que a maquina virtual administrava todos os recursos necessários.

Esta aplicação disponibiliza quatro ferramentas para o usuário (Fig. 3.9). A primeira é direcionada a simulação do robô, a qual permite interagir, movimentar e observar o comportamento do manipulador com outros objetos do cenário, disponibilizando ainda para o usuário uma janela com apresentações e informações complementares do sistema. A segunda consta de uma apresentação dinâmica que permite o acesso as informações de áudio e vídeo como forma de ajuda no processo de aprendizagem. Finalmente, a interação com o robô real é realizada através de uma porta paralela do computador e o envio da informação através de Java.

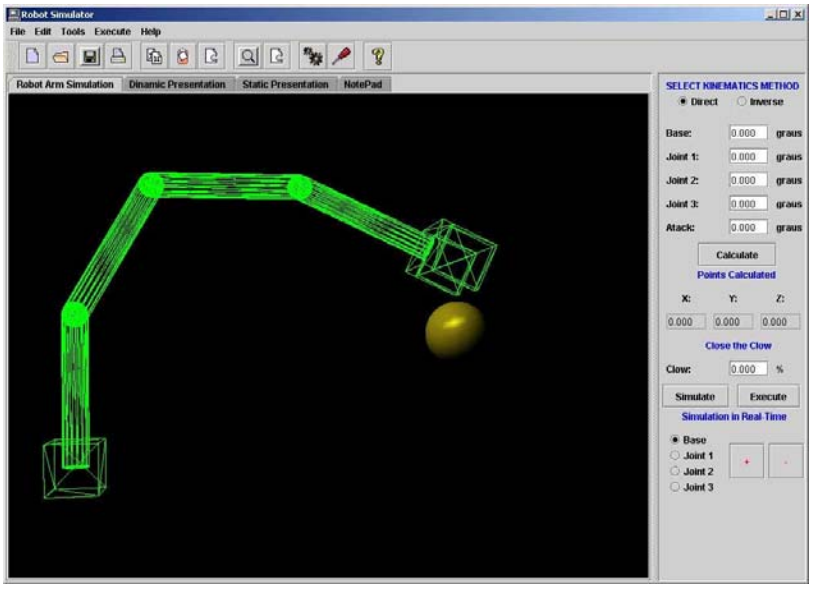
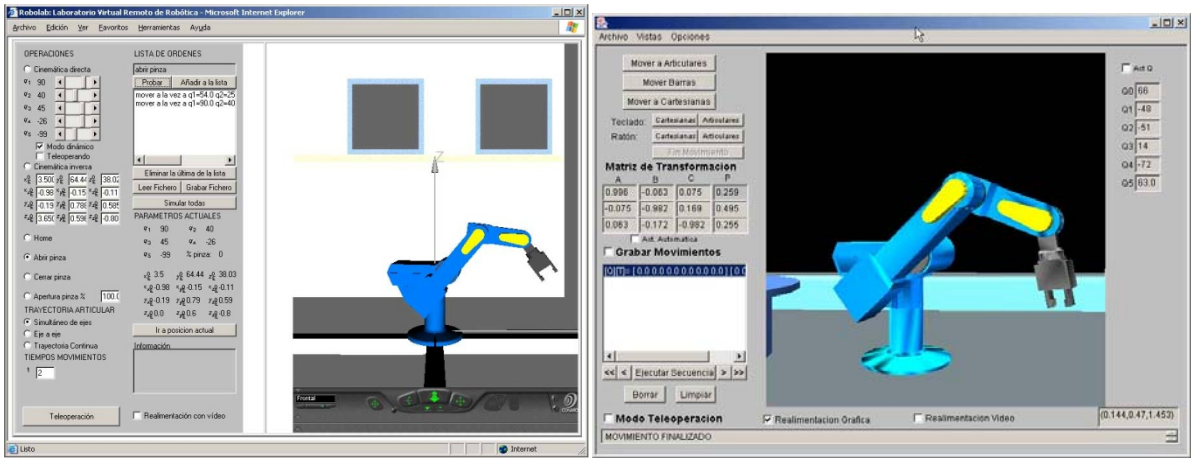


Fig. 3.9: Exemplo de aplicação de simulação. FREIRE et al., 2004.

A partir das inúmeras possibilidades oferecidas através da utilização das ferramentas virtuais no processo de educação, (HERIAS et al., 2004) estabeleceu-se um laboratório virtual, estudando o seu impacto junto aos estudantes. O software ROBOLAB desenvolvido teve como principal finalidade a prática dos comandos de movimento de um robô industrial no processo de aprendizagem de sua cinemática. A arquitetura proposta para o sistema apresenta o manipulador com o seu sistema de controle, um servidor para vídeo e outro para *web* e usuários clientes. Este software necessita apenas de uma máquina virtual em Java e de um *plugin* para VRML.
O aplicativo em Java permite a definição das coordenadas de trabalho do manipulador, a visualização de suas trajetórias e o simulador calcula a cinemática direta e inversa. Após a realização dos comandos para o manipulador, a informação é enviada ao servidor do robô, permitindo assim a realização de uma simulação no controlador e avaliação do seu código para depois enviar à planta real. Duas formas podem ser escolhidas para assistir a execução: a primeira através de câmeras e segunda através das linhas de comando amostrando a seqüência de avanço de comandos. Para melhorar o sistema, o visor de VRML foi removido e trocado por Java3D, obtendo-se assim uma interface e aplicativo unificado e criando a possibilidade de detecção de colisões e inclusão de objetos no espaço de trabalho (interface na Fig. 3.10).

Para avaliar o impacto desta ferramenta, foram implementados diversos experimentos de laboratório obedecendo a objetivos específicos, através de experimentos diferentes implementados em grupos de estudantes, e realizando questionários para os estudantes e professores envolvidos. Os resultados de avaliação dos diferentes grupos mostram que este software foram muito bem aceitos (cerca de 80%), sendo considerado como uma ferramenta complementar ao processo de ensino, e não para substituição de um professor.

(YANG et al., 2004) desenvolveu uma aplicação baseada na *internet* para facilitar o estudo do robô CRS A465. Este sistema apresentou dois componentes principais: o controlador do manipulador em seu ambiente 3D e a planta real. Através da utilização desta ferramenta o usuário pode aprender conceitos de cinemática direta e inversa, modelagem dinâmica e planejamento de trajetórias.



a) **VRML.** **b) Java3D**

Fig. 3.10: Interface ROBLAB para o controle remoto. HERIAS et al., 2004.

Para o primeiro subsistema, o OpenGL foi utilizado para gerar esta interface, que possui controladores no ambiente virtual para a cinemática e colisão, permitindo assim, que o estudante possa realizar a manipulação sem riscos de avarias e danos para o dispositivo robótico real. No momento, que o estudante possuir completo domínio do manipulador, ele estará capacitado para utilizar o segundo sistema que consta de um sistema servidor e cliente. Neste caso o controle do robô é realizado a partir de câmeras dispostas no ambiente de trabalho, que enviam imagens para que o usuário possa comparar o que ele executou *off-line* com as ações realizadas pela planta. Para disponibilizar esta ferramenta na *internet* foi utilizado um protocolo TCP/IP e realizada a conexão do cliente com o servidor.

Conhecendo as inúmeras vantagens e aplicações da realidade virtual para o ensino de robôs (HERIAS et al., 2005) desenvolveu um sistema flexível que permitia o uso de vários manipuladores diferentes e a adição de novos modelos para serem estudados, permitindo também adicionar novos elementos no espaço de trabalho sem modificarmos a interface do usuário. A estrutura funcional era composta de um *applet* de Java na máquina cliente que permitia manipularmos a simulação e tele-operação que se comunicava com a segunda parte: o servidor principal, onde os arquivos e a informação e armazenada.

Nesta aplicação foi utilizado o protocolo HTTP para realizar a comunicação entre o cliente e o servidor, através de duas formas de apresentar a realimentação: através de câmeras *web* ou

através do modelo 3D com equipe cliente. A validação foi realizada entre estudantes, apresentando uma grande aceitação pela maioria deles. Esta aplicação não utilizou o controle através de um robô, somente enviando os comandos para serem executados. A partir da informação disponível têm-se uma matriz de transformação, os valores das juntas, e dados escolhidos pelo usuário. Os dispositivos de entrada reconhecidos foram o teclado, *mouse* e *joysticks* utilizados com e sem realimentação de força.

3.3. Ambientes não adaptados para o homem

Considerando as implicações de realização de tarefas em ambientes perigosos e hostis com equipamentos complexos e de alto custo, (SAFARIC et al., 2003) propôs a utilização de ambientes virtuais como ferramenta de treinamento, planejamento e posteriormente para execução *off-line*, sem necessidade de implementar os sistemas físicos. O equipamento necessário consta de um simulador que possa ser utilizado através da *internet* (Fig. 3.11), representando assim, uma alternativa às soluções que envolvem altos custos existentes no mercado.

O VRML, implementado em Java, permite a visualização dos objetos virtuais, e geração do tratamento de colisões, enquanto que a solução estudada com C++ limita a interoperabilidade da ferramenta. Conseqüentemente, o controle das juntas de um robô foi realizado através do de *Matlab- Simulink*® permitindo deixar disponível o servidor de *internet*, e através disto garantir o acesso à plataforma real, se não estiver *online* o planejamento fica no estado de espera para ser executado. Uma vez que o operador adapte-se à programação *offline*, os comandos podem ser executados e visualizados através de câmeras, proporcionando um enriquecimento da experiência, oferecendo assim, uma solução econômica e adequada para a aprendizagem.

Dentro das necessidades de utilizarmos os conceitos de Realidade Virtual para ambientes não adaptados para um ser humano (ZHANG et al., 2003) desenvolveram um sistema tele-operado para utilização em águas profundas, composto de um sistema de controle do robô, um

console de tele-operação, e um sistema de processamento de imagens para um manipulador submarino o qual consta de dois braços robóticos com sete articulações (Fig. 3.12).

O sistema supervisão e controle eram compostos de câmeras CCD no ambiente de trabalho, que permitiam a obtenção das coordenadas dos objetos para seu posicionamento no ambiente virtual. O ambiente virtual foi desenvolvido utilizando Java3D e as interfaces de comunicação através do TCP/IP. Dois modos de operação foram implementados, o primeiro associado ao operador, permitindo a sua interação com o modelo 3D para o planejamento de trajetórias, e o outro de modo automático, onde a ferramenta funciona com base na informação recebida pelas câmeras para efetuar as tarefas. Após a realização dos testes e a implementação de algoritmos para correções de atrasos devido à utilização *online*, os resultados obtidos mostraram a não necessidade que essas correções fossem realizadas.

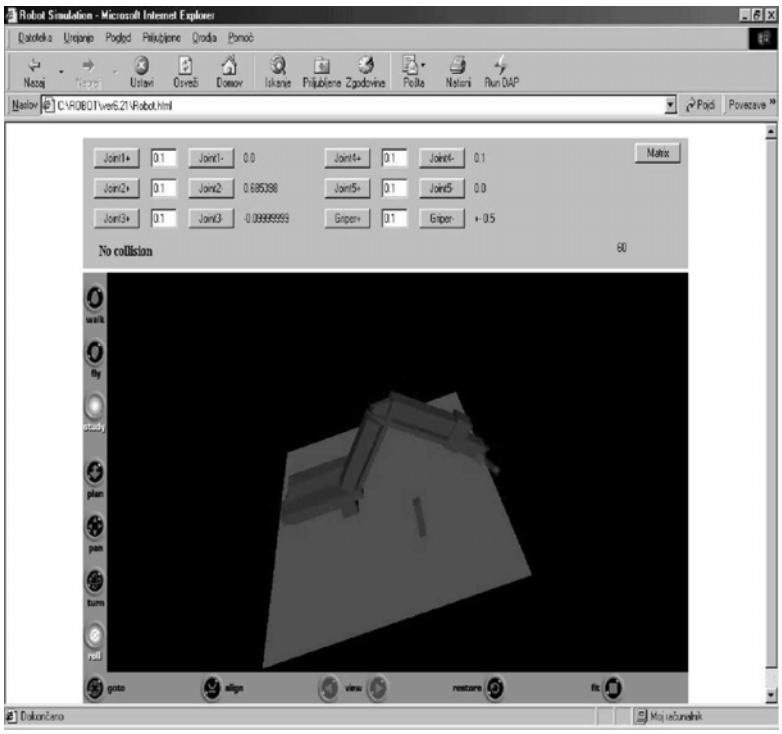
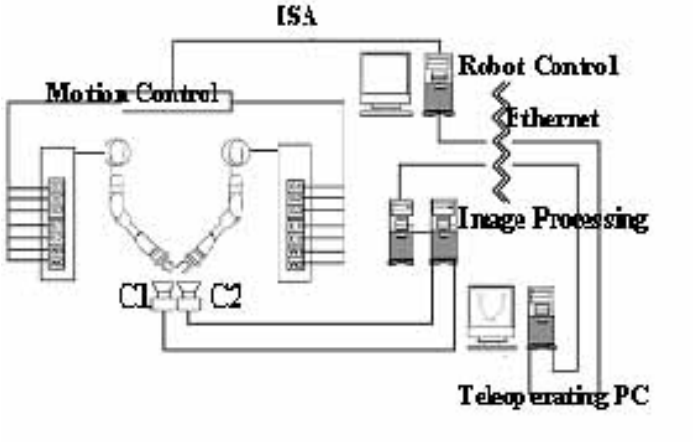


Fig. 3.11: Simulação em ambiente não adaptado ao homem. SAFARIC et al., 2003.

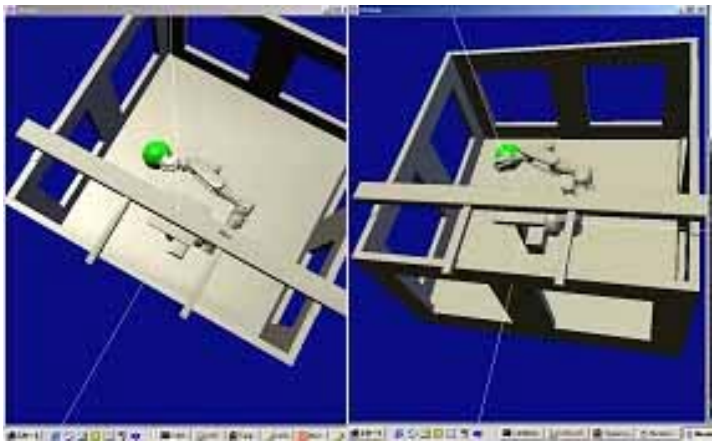
Até o presente momento, os trabalhos desenvolvidos enfatizaram apenas aplicações utilizando robôs seriais, (ALENCASTRE-MIRANDA; GOMEZ; RUDOMIN, 2003) propôs uma tele-operação de robôs em ambientes multi-usuário (Fig. 3.13). Os quatro componentes

principais de um sistema tele-operado podem ser alterados devido às novas tecnologias. Conseqüentemente, os componentes têm evoluído da seguinte forma:

a) O dispositivo mestre (*master*) poderá ser alterado de dispositivos mecânicos ou elétricos para dispositivos de entrada como *mouse* e *joystick* enquanto os escravos (*slave*) poderão ser virtuais ou reais.
b) As interfaces de comunicações poderão não só ser baseadas através de cabeamento físico, como também de sistemas *wireless*,
c) O sistema de supervisão poderá ser realizado através de *webcam* e não através de circuito fechado de televisão.



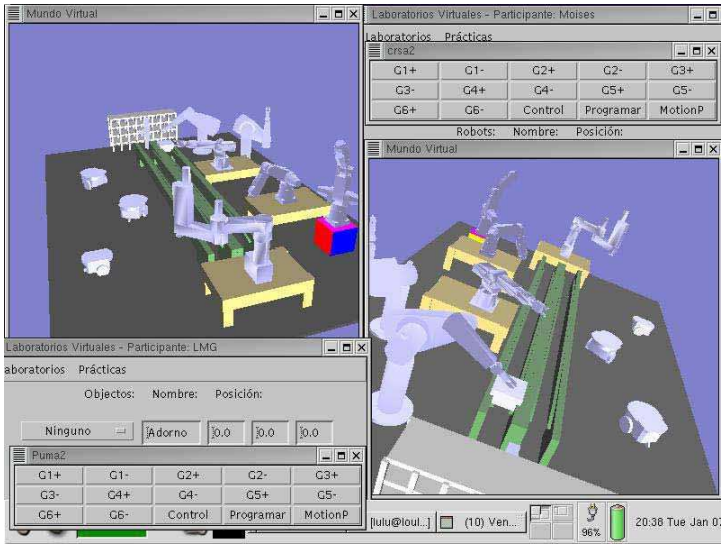
a) Configuração do sistema.



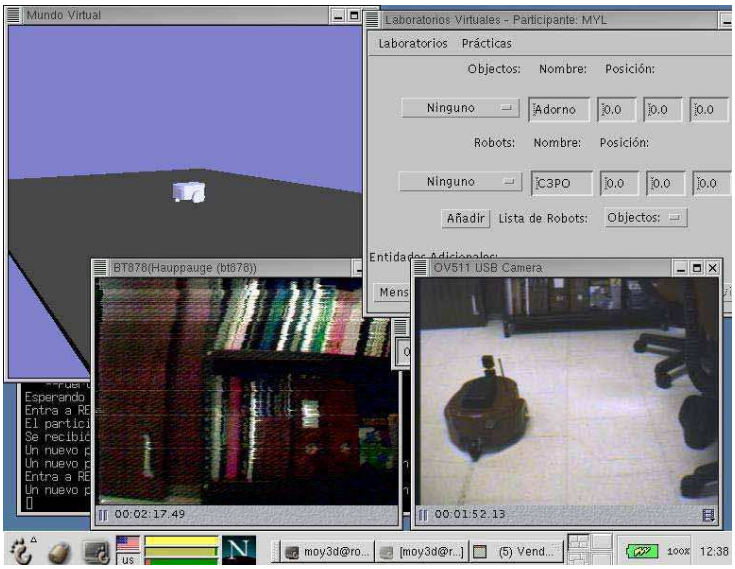
b) Modelo de controle automatizado.

Fig. 3.12: Sistema para manipulação de robô em águas profundas. ZHANG et al., 2003.

Como toda plataforma de tele-operação o robô deve seguir os comandos do operador, e a proposta procura terem vários manipuladores ao mesmo tempo em que vários operadores fazendo uso deles permitindo utilizar uma planta real ou virtual. A arquitetura da ferramenta é planejada através dos seguintes módulos: comunicação para objetos locais e remotos, informações para se ter um sistema flexível, entidades reais para tele-operar robôs reais, sistema de supervisão e controle, e o dispositivo mestre.



a) Tele-operação com vários robôs



b) Tele-operação em tempo real

Fig. 3.13: Implementação de Testes. ALENCASTRE-MIRANDA; GOMEZ; RUDOMIN, 2003.

O MySQL foi escolhido para Banco de Dados, por ser uma solução de código aberto, enquanto as geometrias foram importadas utilizando arquivos OBJ e VRML2.0, para realizar a comunicação o C++ e as Java Native Interfaces foram utilizados. Os dois dispositivos de entrada foram o *Sidewinder Joystick* e *Wheel*, e também para o sistema supervisório considerando a necessidade de fazer *streaming* de vídeo o *Java Media Framework* permitindo assim, a captura utilizando *web*cams.

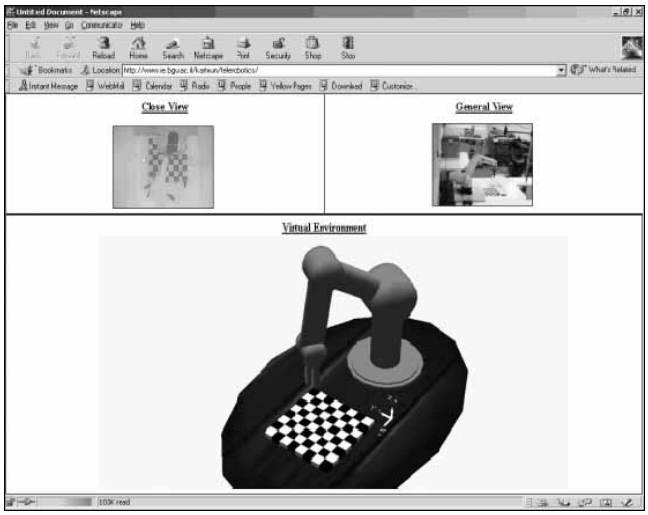
Através do projeto desta plataforma, sete robôs puderam ser disponibilizados num ambiente virtual, sendo realizados testes até com onze manipuladores, operados através de dois usuários utilizando de sistemas de supervisão e controle independentes. Esses testes permitiram verificar as vantagens do sistema desenvolvido, pois ao ser implementado através de orientação a objetos, sua modularidade permitiu a adição de novos elementos neste ambiente, mostrando ainda que devido principalmente a sua portabilidade, o mesmo poderia ser utilizado por uma grande numero de utilizadores (ALENCASTRE-MIRANDA; GOMEZ; RUDOMIN, 2003).

(KARTOUN; STERN; EDAN, 2004) propõe um sistema tele-robótico através de uma interface virtual utilizando o método dos mínimos quadrados para definição da matriz de transformação de *mapping* do ambiente virtual para o real. Nesta plataforma o usuário realizou um planejamento através de um objeto virtual e fazer o controle do real através de sua apresentação virtual em tempo real, para provar a aplicação o cenário escolhido foi a manipulação de elementos perigosos como explosivos químicos ou biológicos.

Conseqüentemente, a partir do planejamento da trajetória e após carregamento da mesma no controlador do robô, o principal objetivo desta manipulação foi a obtenção do conteúdo de um reservatório sem riscos ao operador. A arquitetura do sistema foi adaptada ao usuário e a interface de controle desenvolvida através de um cenário 3D que amostra o robô real, um tabuleiro de xadrez e um sistema coordenado (Fig. 3.14).

O Ambiente virtual foi modelado através de *3D Studio Max* e *Alice*, e a comunicação é realizada através do protocolo TCP/IP. A realimentação sensorial foi conseguida pelas câmeras *web*, para a calibração do sistema a cinemática inversa permite um cálculo fácil e rápido. Para a

realização de testes de precisão foi calculada uma matriz de transformação e durante a realização de testes comparativos entre o sistema virtual em relação ao real foram observados erros da ordem de 3 mm.



a) Ambiente para *internet*

b) Ambiente real

Fig. 3.14: Testes realizados. KARTOUN; STERN; EDAN, 2004.

3.4. Sistemas de preensão através de mãos antropomórficas

Muito antes do nascimento dos primeiros dispositivos robóticos baseados na mão humana, a mesma já tinha sido objeto de diversos estudos devido as suas características fisiológicas. (NAPIER, 1956) faz uma analise dos movimentos e formas de preensão de diferentes objetos.

Considerando a interação entre os músculos, ligamentos e ossos, a classificação em dois grupos de preensão foi realizada em função da forma de preensão pela mão e dedos de um determinado objeto.

Para avaliar as capacidades de preensão da mão, um modelo cinemático foi proposto por (BUCHHOLZ; ARMSTRONG, 1992), onde a geometria dos membros da mão foram aproximados por elipsóides e as articulações idealizadas por linhas que fazem a união das juntas. Os algoritmos para realizar as preensões são baseados na classificação feita por (NAPIER, 1956), e os testes foram realizados a partir de geometrias circulares e as dimensões do protótipo obtidos a partir de pesquisa antropométrica. Dentre as principais limitações deste modelo, encontra-se o fato que somente foram considerados os ângulos de extensão e flexão, geometrias circulares e problemas com a movimentação do polegar.

3.4.1. Modelagem de mãos

Através de aprofundado trabalho de revisão bibliográfica, com o objetivo de estudar e implementar mecanismos semelhantes a partes de um corpo humano, as próteses de mão ocupam um lugar importante, como base do desenvolvimento de um dispositivo antropomórfico. Conseqüentemente, (DRAGULESCU; UNGUREANU; STANCIU, 2005) implementou o modelo para o movimento de um dedo. O modelo cinemático foi baseado em uma cadeia serial de quatro barras, com o método de Denavit-Hartenberg se conseguem as matrizes e parâmetros das articulações, as limitações do sistema estão determinadas pelos ângulos máximos e mínimos que permitem as juntas da mão real. Para visualizar o movimento, os algoritmos foram implementados em *Matlab* ®, considerando ainda que as massas sejam muito pequenas, podendo assim, serem desprezadas na analise dinâmica do problema em estudo.

(DRAGULESCU et al., 2007) desenvolveu também um modelo anatômico de uma mão humana, acrescentando uma analise no espaço de trabalho, considerando o mesmo método proposto anteriormente, agora utilizando os cinco dedos de uma mão (Fig. 3.15), assim mesmo as medidas e limites de rotação para cada um. O modelo cinemático obtido permite que através

de um aplicativo desenvolvido em *Matlab* ®, a trajetória para cada dedo e seu espaço de trabalho possa ser visualizado.

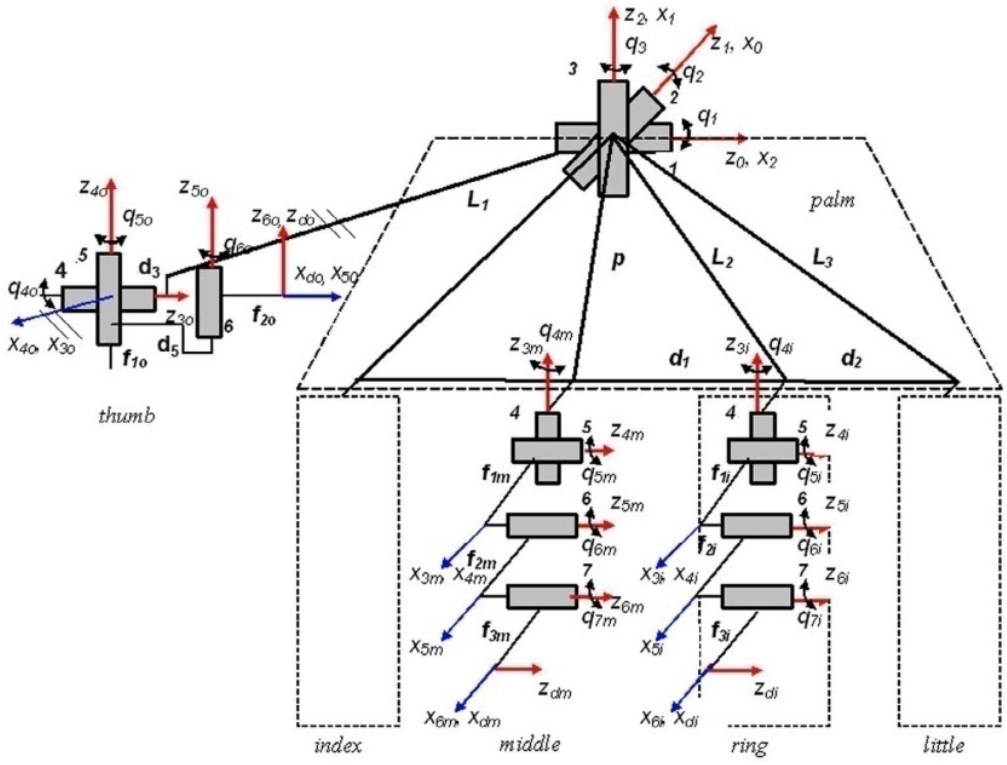
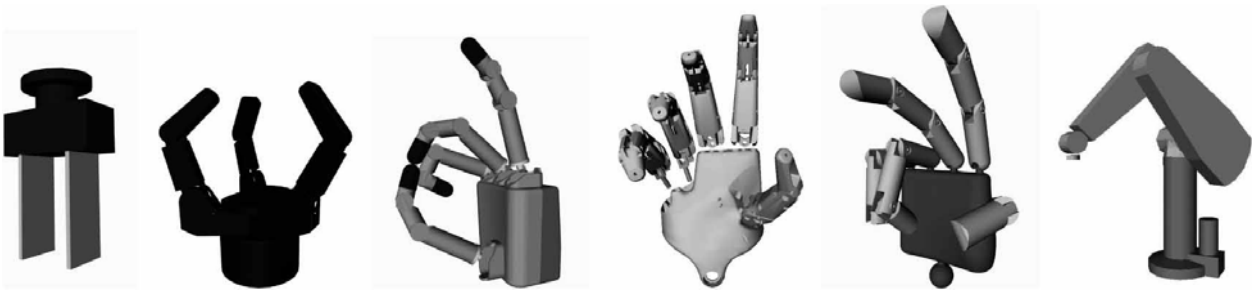


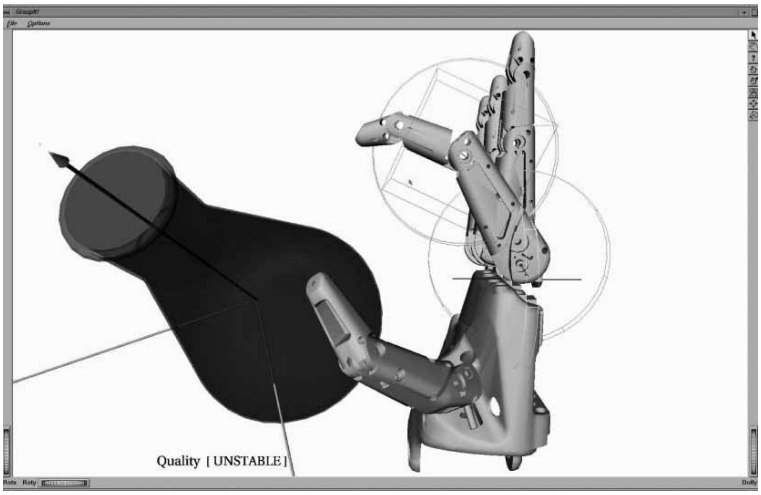
Fig. 3.15: Modelagem de uma mão antropomórfica. DRAGULESCU et al., 2007.

3.4.2. *Grippers* e mãos virtuais

Outro tipo de aplicação freqüente dentro robótica são a utilização de dispositivos baseados em partes do corpo humano, e no caso particular das mãos. (MILLER et al., 2005) desenvolve procedimentos para visualização e simulação da preensão de diferentes *grippers* existentes (Fig. 3.16). Duas ferramentas foram utilizadas, o *Grasp it!* e *Matlab-Simulink*®.



a) Alguns tipos de garras testados



b) Exemplo de teste de preensão.

Fig. 3.16: Software Grasp it!. MILLER et al., 2005.

O software aplicativo *Graspt it!* foi desenvolvido para ser utilizado em diferentes mãos mecânicas, permitindo a detecção de colisões e geração de movimentos de preensão para diferentes tipos de objetos, simulação dinâmica e algoritmos de controle permitem uma melhor execução das tarefas.

Dentro os principais elementos de *Graspt it!* (Fig. 3.16), têm-se formas geométricas básicas e elementos dinâmicos, onde a principal diferença é que alguns destes, só podem ser utilizados como obstáculos, enquanto outros fazem parte do *gripper*. Os robôs são localizados no espaço através dos parâmetros de Denavit-Hartenberg, onde o numero de dispositivos podem variar, permitindo assim a criação de plataformas de trabalho utilizando diferentes robôs, permitindo ainda a fácil adição de novos manipuladores importados através de software de CAD.

Este programa permite ainda, uma interação com *Matlab* ® de forma que através de *sockets* TCP, poderão ser realizadas a simulação dinâmica, retornando posteriormente à informação para este aplicativo. A detecção de colisões permite que os objetos nas cenas não sejam interpostos uns aos outros. Para solucionar o planejamento das preensões, o problema foi abordado através de posições pré-definidas, entretanto, isto não garantia que o planejamento de preensão seja ótimo para cada *gripper*.

As aplicações de mãos virtuais podem também ser utilizadas em aplicações médicas. (KOLLREIDER et al., 2007) apresenta um dispositivo robótico para reabilitação de pacientes com apoplexia, onde o modelo físico é otimizado através do uso de um modelo virtual (Fig. 3.17), permitindo assim, o estudo do comportamento do dispositivo.

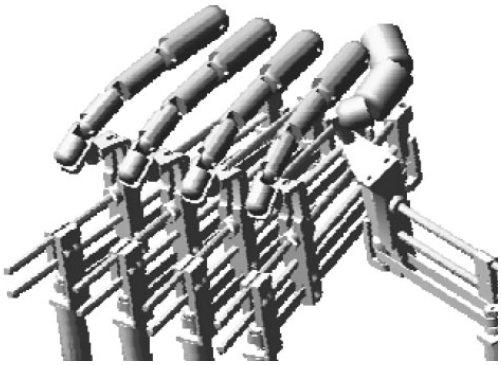
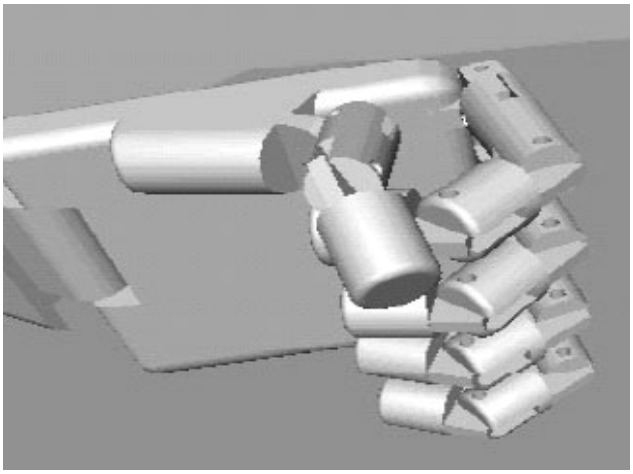
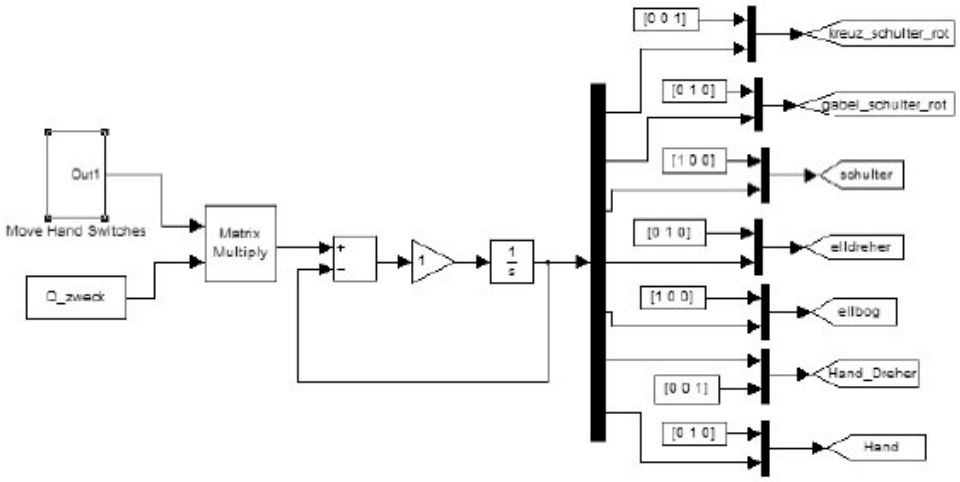


Fig. 3.17: Dispositivo para reabilitação de pacientes com apoplexia. KOLLREIDER et al., 2007.

Devido ao crescimento e desenvolvimento de *grippers* do tipo mão humana, uma ferramenta virtual para mãos virtuais 3D e VRML foi desenvolvida por (TSEPKOVSKIY et al., 2008), considerada de grande importância para a implementação e monitoramento de próteses e robôs (Fig. 2.18). Para os modelos 3D do dispositivo o SolidWorks® foi utilizado, já que o mesmo permite exportar arquivos no formato VRML para serem utilizados com o *Virtual Reality Toolbox* de *Matlab-Simulink*® para implementar, a arquitetura de controle e realizar a simulação do comportamento dinâmico do dispositivo.



a) Gripper na posição de preensão esférica.



b) Representação em Matlab do modelo de movimentação da mão.

Fig. 3.18: Software para projeto e simulação de grippers. TSEPKOVSKIY et al., 2008.

3.5 Comentários finais do capitulo

O trabalho de revisão bibliográfica apresentada neste capitulo desta dissertação abrange a documentação pesquisada em recentes fontes de informação (Conferencias, Proceedings e revistas internacionais indexadas) durante o período de 1956 ate 2008. A Tabela 3.1 apresenta uma classificação das principais fontes pesquisadas e correspondente fator de impacto.

Journal	Clase	ISSN	Citações	Factor de impacto	Artigos citados
Journal Biomechanics	-	-	-	2.542	-
Journal of robotics systems	-	0741-2223	534	0.380	51
Journal of Bone and Joint Surgery	-	219355	-	2.444	-
Industrial Robot: An International Journal	B	0143-991x	-	0.278	-
IEEE Robotics and Automation Magazine	-	1070-9932	280	0.826	34
IEEE Transactions on Robotics and Automation	A	1042-296X	3909	2.126	110
ASME Journal of Applied Mechanics	A	0021-8936	-	0.651	-
Artificial Intelligence in Engineering	C	0954-1810	-	1.295	-
Autonomous Robots	A	0929-5593	516	1.309	27
Integrated Computer-Aided Engineering	C	1069-2509	53	0.148	26
IEEE Transactions on Biomedical Engineering	A	0018-9294	6423	1.815	267

Tabela 3.1 Síntese das principais revistas indexadas na área e respectivo fator de impacto.

A base de informações consultada nesse trabalho de pesquisa permite observar a importância da Realidade Virtual nas diferentes áreas da engenharia, mostrando que esse assunto vem sendo cada vez mais explorado por pesquisadores. Podemos também destacar um significativo aumento na sua utilização nestes últimos anos, não somente para os robôs convencionais, como também para os mais recentes avanços da engenharia, através do desenvolvimento de dispositivos robóticos baseados em formas humanas.

No próximo capitulo deste trabalho de pesquisa serão apresentados elementos para a criação de um ambiente virtual, com ênfase na modelagem de dispositivos antropomórficos do tipo *gripper*, e ambientes de operação para os mesmos, baseados nos ambientes propostos neste capitulo de revisão bibliográfica.

Capitulo 4
Desenvolvimento do Ambiente Virtual

Neste capitulo serão apresentados elementos para o desenvolvimento de um ambiente virtual, com ênfase na modelagem de dispositivos antropomórficos do tipo *gripper*, e ambientes de operação para os mesmos, baseados nos ambientes propostos no capitulo apresentado anteriormente.

Para a criação de um ambiente virtual, o primeiro procedimento a ser realizado é a concepção do dispositivo antropomórfico, no objeto de estudo desse trabalho: uma mão humana (Fig. 4.1). Os conhecimentos prévios das dimensões permitem determinar os pontos de rotação e deslocamento no espaço de trabalho com respeito à origem desejada.

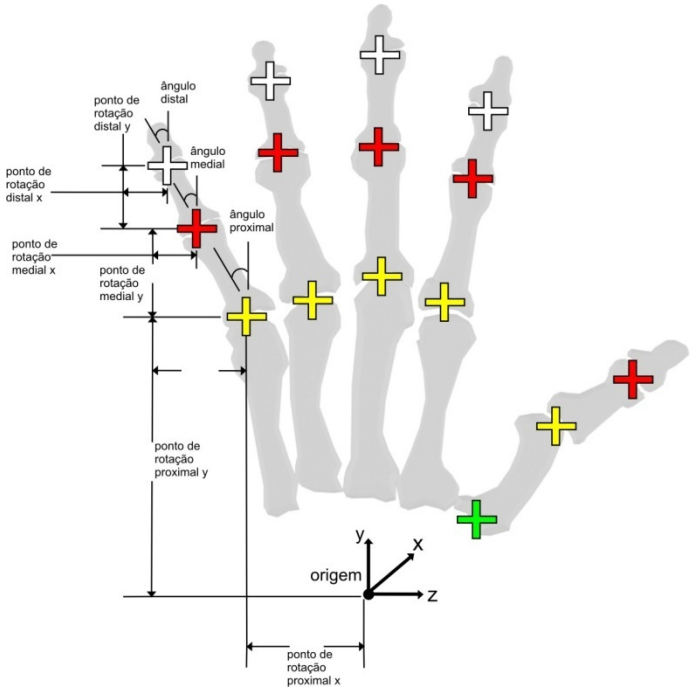


Fig. 4.1: Articulações dos ossos da mão humana para um dedo.

4.1. Modelagem do dispositivo antropomórfico

Inicialmente, para entender o funcionamento cinemático de um dispositivo baseado nas características da mão humana, realizou-se a modelagem de uma mão antropomórfica, sendo suficiente a consideração de somente um dedo. O comportamento cinemático do dedo é igual ao de um mecanismo serial de três barras.

Dos métodos disponíveis para analisar o dedo, tem-se o método geométrico e o DH, a maior diferença entre estes dois é que o primeiro tem uma maior velocidade de computação já que as equações estão prontas em contraste com as operações matriciais onde operações com zeros fazem que os cálculos tomem maior tempo, como para o analise requerido não é necessária uma rapidez computacional, o método de DH é utilizado já que através de seus parâmetros consegue se adicionar ou apagar graus de liberdade e obter as posições das barras devido aos ângulos. O método de DH consiste no estabelecimento dos parâmetros e respectivas matrizes de transformação associadas ao dispositivo em estudo, permitindo assim, o calculo das equações de um dedo a partir de seus movimentos de rotação angular.

4.1.1. Parâmetros Denavit-Hartenberg

A partir das características geométricas do modelo 3D, os eixos coordenados devem ser definidos para determinação dos parâmetros de DH. Para obtermos uma relação correta desses parâmetros, os seguintes passos devem ser realizados:

De um modo geral quando definimos a posição e a orientação do sistema de coordenadas i em relação ao sistema i-1, os seguintes parâmetros de Denavit-Hartemberg deverão ser especificados:

- a_i : Distância ao longo do eixo x_i desde a interseção do eixo z_{i-1} com o eixo x_i até a origem do sistema i no caso das articulações de rotação. No caso das articulações prismáticas é a distancia entre os eixos z_{i-1} e z_i.

- d_i : Distância ao longo do eixo z_{i-1} desde a origem do *i-1* sistema até a interseção do eixo z_{i-1} com o eixo x_i, é um parâmetro variável para as articulações prismáticas.
- θ_i : Ângulo entre os eixos x_{i-1} e x_i medido num plano perpendicular ao eixo z_{i-1}, fazendo uso da regra da mão direita, é um parâmetro variável para as articulações de rotação.
- α_i : Ângulo de separação do eixo z_{i-1} e o eixo z_i medido num plano perpendicular ao eixo x_i fazendo-se a utilização da regra da mão direita.

Os anteriores parâmetros poder ser vistos em detalhe na Fig. 4.2.

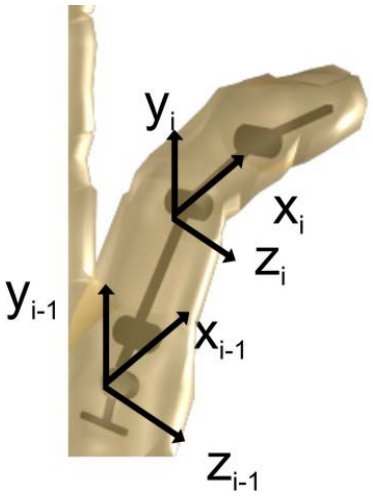


Fig. 4.2: Eixos para definir os parâmetros DH.

A partir das definições dos sistemas de referência associados a cada grau de liberdade é realizada uma relação dos ângulos de cada falange, associando cada um ao respectivo parâmetro DH definido anteriormente, como é apresentado na Tabela 4.1

A partir da identificação dos parâmetros associados aos dedos, e considerando estes parâmetros iguais para todos os dedos (Fig. 4.3), inclusive para o polegar, o sistema em estudo é idealizado e constituído de cinco mecanismos seriais unidos a uma base comum (Fig. 4.4), neste caso a palma da mão.

Tabela 4.1: Ângulos para cada dedo.

Falange	Proximal	Medial	Distal
Extensão, ângulo de flexão	θ_p	θ_m	θ_d
Abdução, ângulo de adducção	α_p	0	0

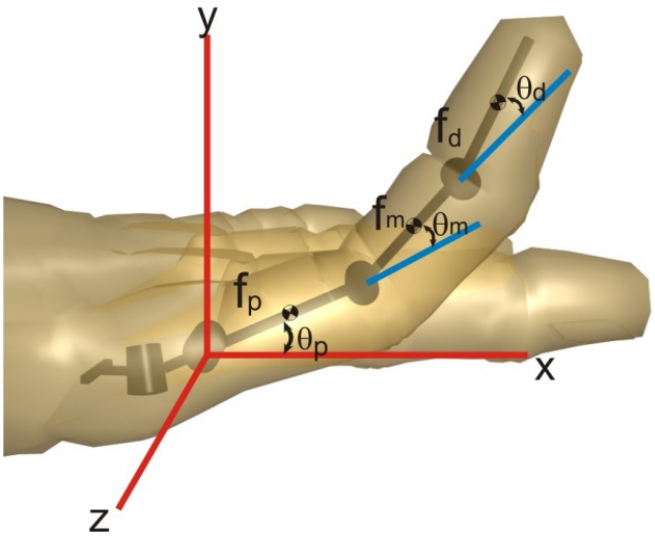


Fig. 4.3: Parâmetros para o dedo.

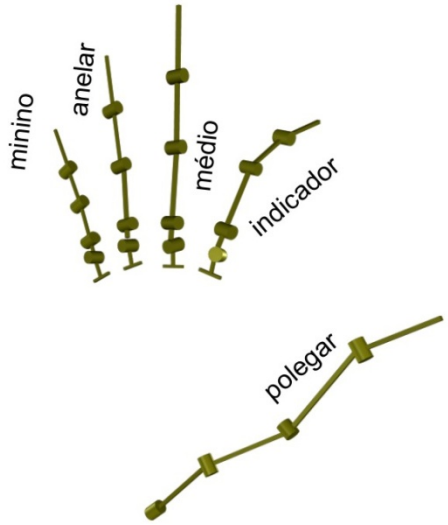


Fig. 4.4: Idealização de uma mão como um conjunto de mecanismos seriais anexo a mão.

Para o estabelecimento da tabela de DH (Fig. 4.3) as rotações de abdução e adução não são consideradas, e as mesmas não são necessárias para a realização de testes de preensão, os quais através dessa restrição ainda abordam a maioria de objetos que uma mão pode tomar, sendo assim os mais comuns de tipo esférico e prismático.

Devido ao uso da sistemática de DH, no momento de precisar as rotações restringidas, estas podem ser adicionadas sem problema nenhum pela escalabilidade do método. Assim, os parâmetros DH considerados para o dedo em estudo são descritos na Tabela 4.2. Essa informação permite a criação de matrizes de transformação para cada mecanismo serial representadas genericamente através da equação 4.1 relacionada com a Fig. 4.5:

Tabela 4.2: Parâmetros DH.

Articulação/Parâmetro DH	α_i	a_i	θ_i	d_i
1	0	f_p	θ_p	0
2	0	f_m	θ_m	0
3	0	f_d	θ_d	0

$$\boldsymbol{T_{i-1}^{i}} = \begin{pmatrix} \boldsymbol{\cos(\theta_i)} & \boldsymbol{(-\sin(\theta_i))\cos(\alpha_i)} & \boldsymbol{\sin(\theta_i)\sin(\alpha_i)} & \boldsymbol{a_i\cos(\theta_i)} \\ \boldsymbol{\sin(\theta_i)} & \boldsymbol{\cos(\theta_i)\cos(\alpha_i)} & \boldsymbol{(-\cos(\theta_i))\sin(\alpha_i)} & \boldsymbol{a_i\sin(\theta_i)} \\ \boldsymbol{0} & \boldsymbol{\sin(\alpha_i)} & \boldsymbol{\cos(\alpha_i)} & \boldsymbol{d_i} \\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{1} \end{pmatrix} \quad \textbf{4.1}$$

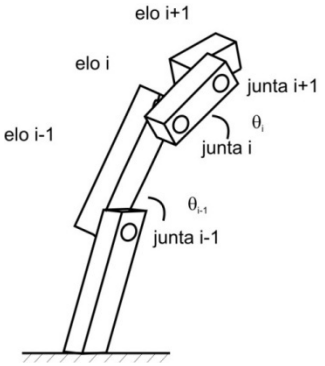


Fig. 4.5: Sistema de numeração considerado na obtenção das matrizes de transformação de coordenadas.

A matriz final de transformação estará constituída pela multiplicação das matrizes de cada junta, como é mostrado na Equação 4.2, onde se obtém uma matriz associada a cada articulação do dedo, sendo possível determinarmos a rotação na posição ponta dos dedos.

$$\boldsymbol{T^3 = T_0^1 T_1^2 T_2^3} \quad \textbf{4.2}$$

4.1.2. Validação da cinemática

Para avaliar o método escolhido, o mesmo foi implementado de acordo com o diagrama de fluxo apresentado na Fig. 4.6, onde os parâmetros estão baseados nas características antropomórficas do dispositivo como as dimensões das partes associadas com as falanges e os ângulos de flexão ou extensão de cada falange de forma independente como as entradas. O Algoritmo 1 implementado apresenta detalhadamente o diagrama da Fig. 4.6 e os resultados gráficos obtidos são apresentados na Fig. 4.7.

A obtenção da posição e orientação de cada falange assim como da ponta do dedo através das matrizes de transformação com base nos parâmetros DH definidos e os ângulos de entrada escolhidos pelo usuário, podemos constatar que é suficiente a alteração dos parâmetros de entrada para habilitar ou desabilitar qualquer grau de liberdade de forma a obter movimentos para cada dedo similares aos de uma mão.

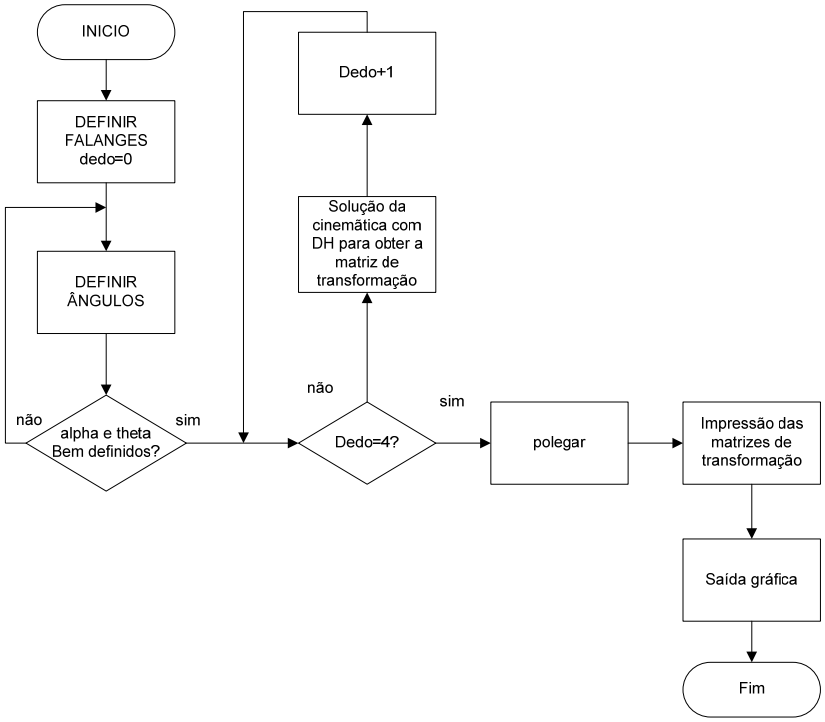
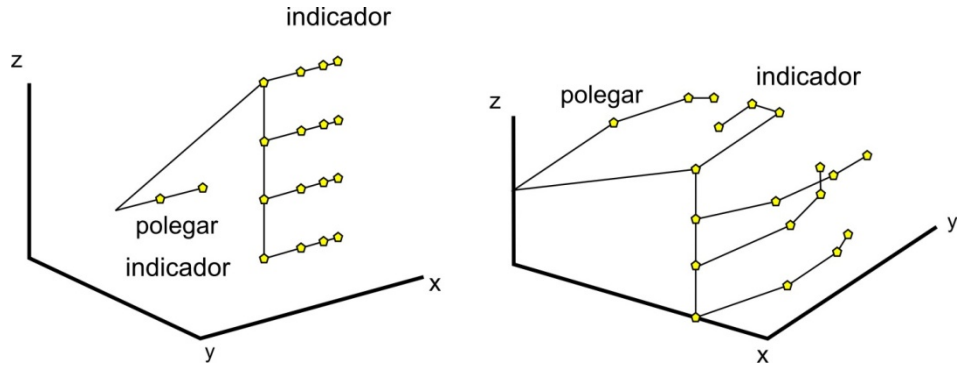


Fig. 4.6: Diagrama Blocos do código implementado.

input : Ângulos de rotação para cada falange θ_p, θ_d e θ_m
output: Gráfica em 3D das posições dos dedos

1.1 *Dimensões das falanges estão fixas as medidas de uma mão humana*;
1.2 **foreach** *ângulo* θ *de cada falange* **do** assinar valor não maior a 90 graus;
1.3 *armazenamento em* $\theta_p\,\theta_d\,\theta_m$;
1.4 *define-se a matriz de transformação*;
1.5 **for** $i \leftarrow 0$ **to** 2 **do**
1.6 | *calculo de matriz de transformação para articulação i*;
1.7 | *armazenamento da matriz*;
1.8 **end**
1.9 *visualização das matrizes*;
1.10 *multiplicação das matrizes*;
1.11 *transformação e rotação das falanges para levar a gráfica*;

Algoritmo 1: Implementação dos parâmetros de DH.



a) Mão Aberta **b) Diferentes ângulos de entrada**

Fig. 4.7: Resultados gráficos da posição da mão a partir da implementação do método.

4.2. Projeto de um ambiente virtual

Para a criação dos componentes de um ambiente virtual existem várias ferramentas para desenho assistido por computador disponíveis, neste aspecto nossa delimitação de objetivos permite estabelecer certas características e assim escolher a mais apropriada. Na procura de soluções para obter um ambiente virtual com geometrias criadas nesses software de CAD, existem opções como VRML e Java3D.

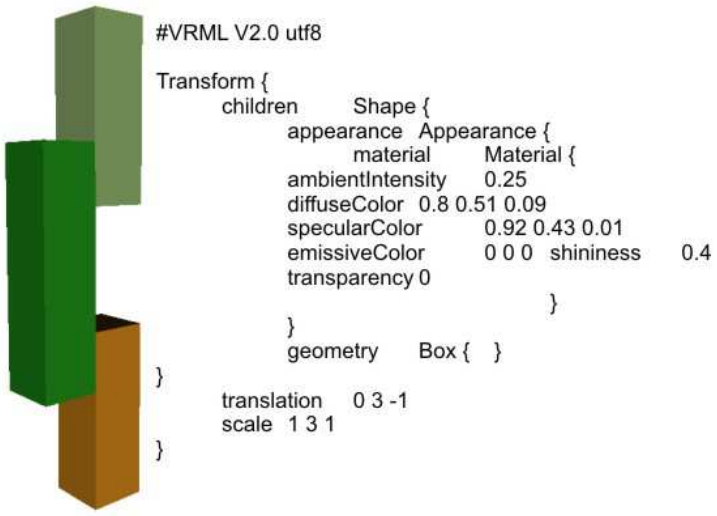
4.2.1. VRML

O VRML, que significa Linguagem de Modelagem de Realidade Virtual foi criado com o objetivo de levar objetos em três dimensões para a internet e brindar a possibilidade de interagir com eles. O arquivo tem o formato de texto, o padrão que ainda é utilizado é o *VRML 2.0* e sua codificação é *utf8*, a extensão do arquivo é **.wrl* e pode ser editado através de *software* de edição de texto. Dentre as principais vantagens do VRML estão a capacidade de reprodução de um conteúdo multimídia como fotografias, vídeos (somente no formato mpeg) e áudio (somente no formato wav), dentro dos ambientes virtuais. A estrutura do VRML é composta por nós, os quais podem ser classificados como:

i) *Shape* para a utilização de formas geométricas, tais como cubos, esferas, cilindros ou cones e texto,
ii) *Appearence* para determinar os materiais e nível de textura,
iii) *Anchor* para o uso de *hiperlinks*,
iv) *Navigator* para configurar o tipo de vista e a velocidade de navegação do avatar,
v) *Sound*, *lights*, *time sensor* e *touch sensor* entre outros.

Uma maior informação desses tipos de padrões pode ser encontrada no endereço WEB *http://www.web3d.org/x3d/specifications/vrml/*.

Esta linguagem além de permitir a criação, edição e navegação de ambientes 3D, possibilita também a edição e navegação com objetos exportados a VRML criados com *software* de CAD ou modelagem 3D. A Fig. 4.8 apresenta um exemplo de um mecanismo de três barras implementado a partir de formas geométricas disponíveis em VRML. Para fazer uso dos ambientes virtuais gerados com VRML é imprescindível a utilização de um *plugin*, o qual permite a visualização, navegação e interação através de um navegador para internet.



```
#VRML V2.0 utf8

Transform {
        children        Shape {
                appearance  Appearance {
                        material        Material {
                ambientIntensity      0.25
                diffuseColor  0.8 0.51 0.09
                specularColor         0.92 0.43 0.01
                emissiveColor         0 0 0  shininess      0.4
                transparency 0
                                                }
                }
                geometry      Box {   }
}
        translation     0 3 -1
        scale  1 3 1
}
```

Fig. 4.8: Mecanismo implementado através de cubos em VRML e código associado.

Considerando que a ultima atualização do padrão VRML foi realizada em 1997, funcionalidades adicionais não têm sido implementadas, fazendo de esta opção uma ferramenta não escalável, por isso para acrescentar as propriedades de compatibilidade, independência de *plugin* e processamento de informação, software adicional se precisa para complementar esta linguajem.

Alguns softwares, tais como o *Virtual Reality ToolBox* de Matlab®, no qual o modelo VRML é conectado com o *SimMechanics* para simular seu funcionamento (TSEPKOVSKIY ET AL., 2008), e outras como as aplicações baseadas em Java que permitem ganhar a escalabilidade que esta linguagem oferece (HERIAS et AL., 2004)(HERIAS et AL., 2005).

4.2.2. JAVA 3D

Java3D é um API que pode ser utilizado com a linguagem de programação Java de *SunMicrosystems*, que permite obter as vantagens da programação orientada aos objetos, e adicionalmente pode ser executado sobre diferentes sistemas operativos, considerando que os programas são executados numa máquina virtual de Java, garantindo assim, a sua portabilidade em outras aplicações.

O Java3D permite implementar programas para criar e visualizar telas gráficas em três dimensões. As aplicações desenvolvidas em Java podem ser utilizadas como *standalone* ou *applets* para paginas *web*.

4.2.2.1 – Principais características

O universo virtual é criado a partir de um *scenegraph* (Fig. 4.10), onde os grafos são compostos através de nós e arcos, e em geral obedece a hierarquia de relacionamento do tipo pai-filho. A cena é construída com nodos relacionados de forma pai-filho, constituindo assim, uma estrutura similar a um arvore, conforme é mostrado na Fig. 4.9.

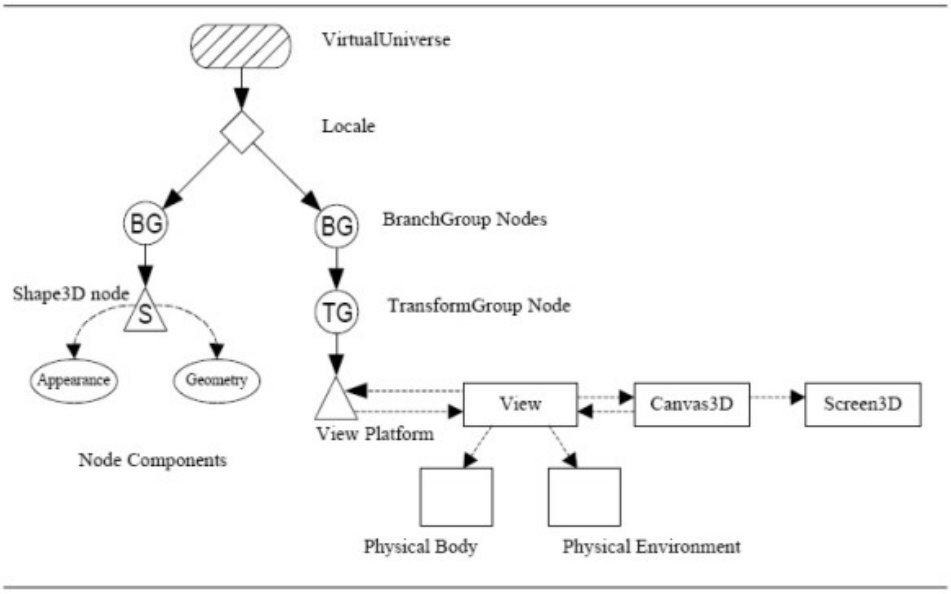


Fig. 4.9: Exemplo de hierarquia utilizando um *scenegraph* em (Getting started with Java3D).

4.2.2.2 – Classes utilizadas no aplicativo

Durante a implementação de um aplicativo são utilizadas as classes apresentadas a seguir:

- *BranchGrou*p
- *Canvas3D*: permite o *render* dos objetos na cena.
- *Transform3D*: o objeto transformado é representado através de uma matriz 4x4, que pode representar movimentos de translação e/ou rotação ou um fator de escala

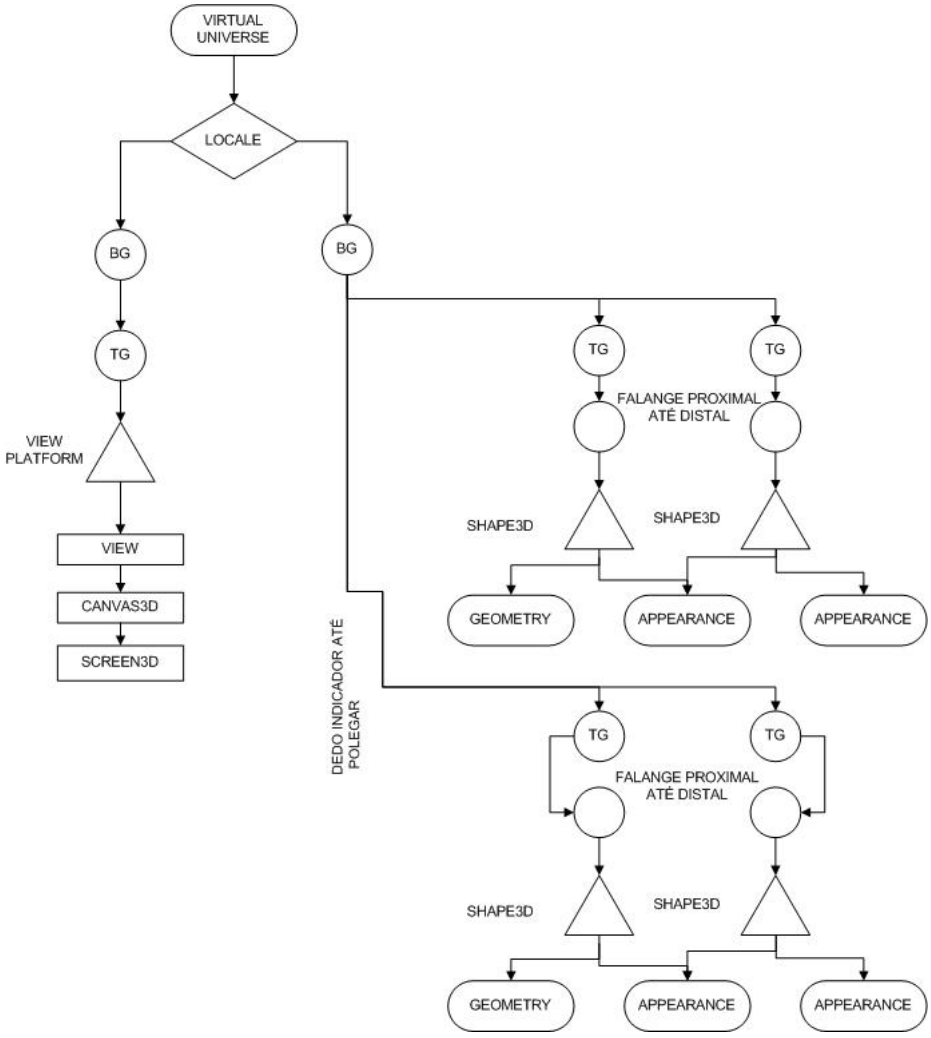


Fig. 4.10: *Scenegraph* para o aplicativo proposto.

4.2.2.3 – Nós

Para visualizarmos os objetos no espaço 3D, os atributos visuais devem ser configurados, como é apresentado na Fig. 4.11 que apresenta algumas subclasses para estabelecer esses parâmetros. Para fazer um objeto visível no Canvas3D, temos que definir o seu nó de *appearance*, a *light* e o *background*.

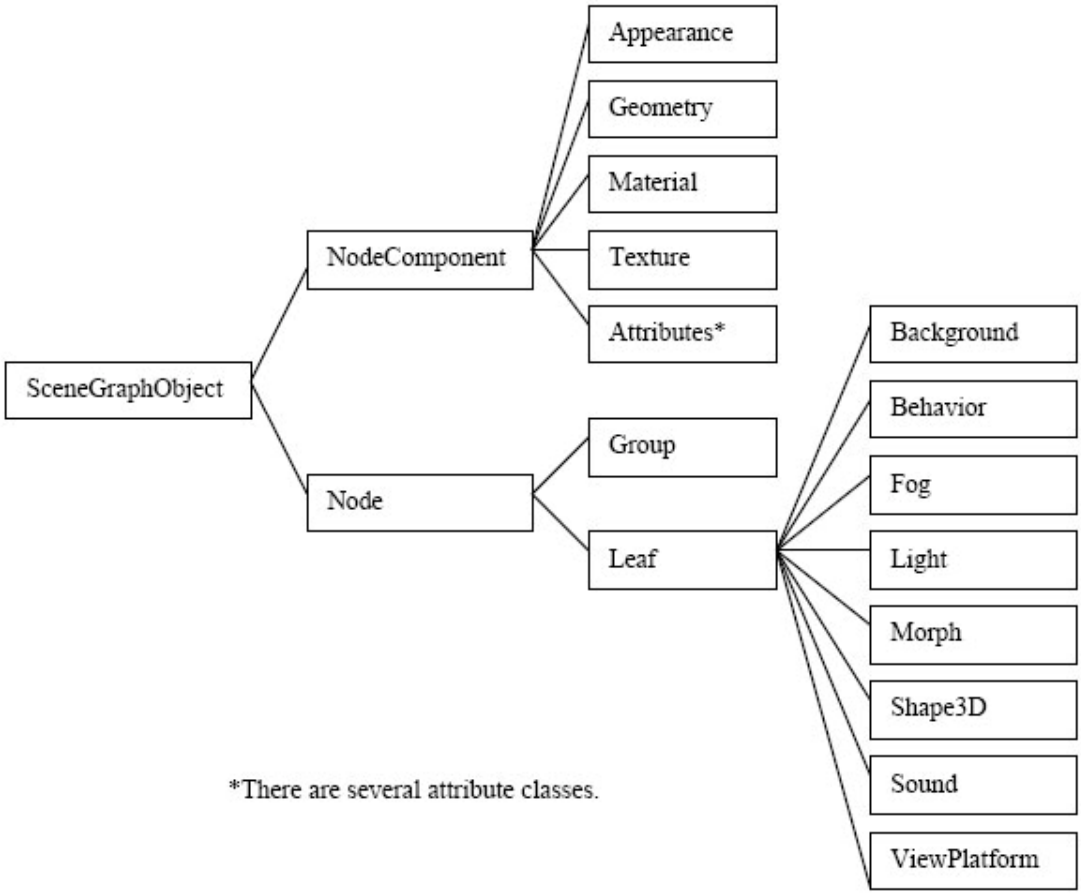


Fig. 4.11: Subclasses para definição de atributos visuais, (Getting started with Java3D).

Desta forma são definidos os atributos visuais dos objetos virtuais os quais podem ser observadas na Fig. 4.12, onde um ou mais materiais podem ser criados para cada elemento e as propriedades do VRML podem ser utilizadas, o código de implementação é apresentado no Algoritmo 2.

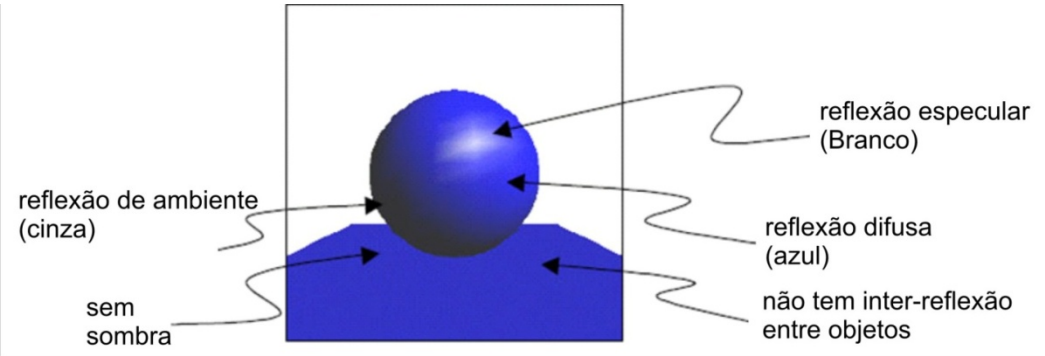


Fig. 4.12: Visualização das características dos nós *appearance*, (Getting started with Java3D).

```
2.1 Appearance app = new Appearance();
2.2 Color3f color = new Color3f(0.815f,0.815f,0.0f);
2.3 Material mat1 = new Material();
2.4 mat1.setDiffuseColor(color);
2.5 mat1.setSpecularColor(1.0f,1.0f,1.0f);
2.6 mat1.setShininess(64.0f);
2.7 mat1.setAmbientColor(new Color3f(0.0f,0.0f,0.0f));
2.8 app.setMaterial(mat1);
```

Algoritmo 2: Nó *Appearance*

As cores são definidas no formato RGB e seus valores devem ser flutuantes, a Tabela 4.3 algumas combinações básicas para obtenção de diferentes tipos de cores.

A definição das cores não é suficiente para garantir que os objetos sejam visíveis, por isto o ambiente precisa ter iluminação como no mundo real, e existem diferentes tipos, conforme mostra a fig. 4.13:

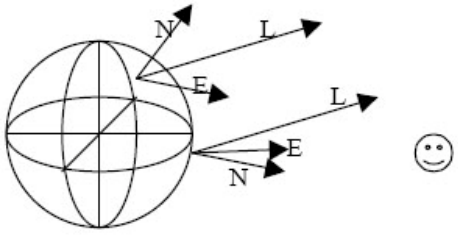
i) *Directional light* : necessita dos valores em *x*, *y* e *z* para especificar a direção da luz;
ii) Point *light* : necessita dos valores em *x*, *y* e *z* para especificar a posição pois ilumina em todas as direções.
iii) *Spot light :* os parâmetros de configuração são a posição, atenuação e o angulo de abertura da luz.

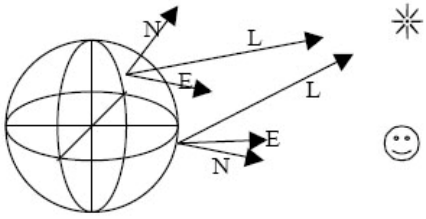
Estes tipos devem ser especificados dentro de uma geometria limitante neste caso uma esfera, de forma que a luz somente devera iluminar no interior desta esfera. As cores de luz são definidas através das componentes RGB.

Para o ambiente virtual do aplicativo, dois tipos de iluminação são escolhidos, uma luz de ambiente e outra direcional (algoritmo 3). A primeira emite a iluminação do ambiente, enquanto a segunda permite tornar visíveis as geometrias da mão. Ao mesmo tempo o fundo do ambiente é definido e também suas cores são especificadas através do padrão RBG.

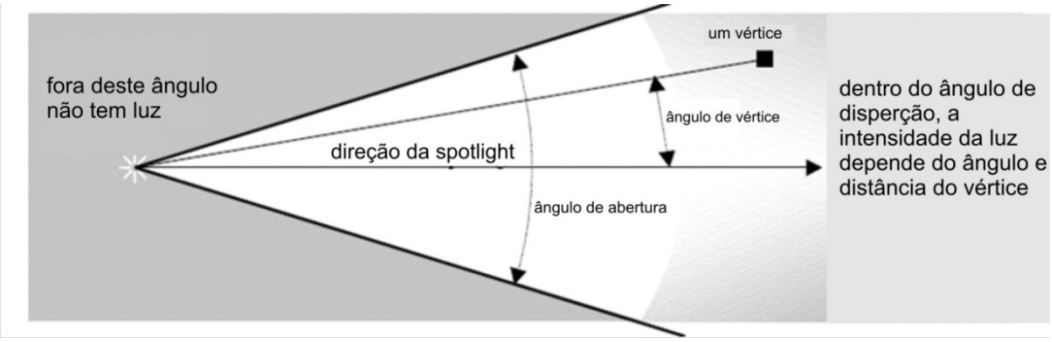
Tabela 4.3: Valores de algumas cores para aparência

Cor	Vermelho (Red)	Verde (Green)	Azul (Blue)
Preto	0	0	0
Verde	0	0.5	0
Prata	0.75	0.75	0.75
Lime	0	1	0
Cinza	0.5	0.5	0.5
Olive	0.5	0	0.5
Branco	1	1	1
Amarelo	1	1	0
Navy	0	0	0.5
Vermelho	1	0	0
Azul	0	0	1





a) Directional light b) Point light



i. Spot Light

Fig. 4.13 Tipos de iluminação e parâmetros de configuração. (Getting started with Java3D).

```
3.1  TransformGroup tgLight = new TransformGroup();
3.2  BoundingSphere bounds = new BoundingSphere(new Point3d(), Double.MAXVALUE);
3.3  AmbientLight lightA = new AmbientLight();
3.4  lightA.setInfluencingBounds(bounds);
3.5  tgLight.addChild(lightA);
3.6  DirectionalLight lightD2 = new DirectionalLight();
3.7  lightD2.setInfluencingBounds(bounds);
3.8  lightD2.setDirection(new Vector3f(-1.0f, -1.0f, -1.0f));
3.9  lightD2.setColor(new Color3f(0.9f, 0.9f, 0.9f));
3.10 tgLight.addChild(lightD2);
3.11 Background bg = new Background(new Color3f(0.1f,0.1f,0.2f));
3.12 bg.setApplicationBounds(bounds);
3.13 tgLight.addChild(bg);
3.14 return tgLight;
```

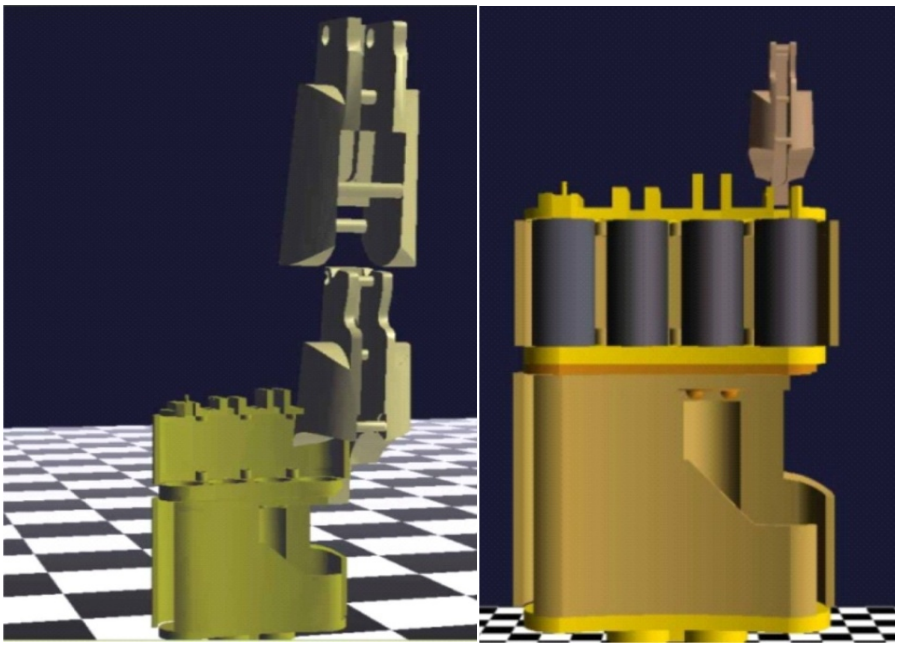
Algoritmo 3: Iluminação e fundo de ambiente 3D.

4.2.3. Arquivos compatíveis com a linguagem Java

A linguagem Java3D é compatível com arquivos de tipo **.obj* e **.wrl*, os quais são utilizados através de um *software* de CAD e modelagem 3D. Esta compatibilidade permite que o usuário não fique dependendo das geometrias primitivas (esfera, cubo, cilindro e cone) que oferece esta linguagem, tornando possível exportar os desenhos implementados desde e fazer uso deles no ambiente Java. Para conseguir importar as geometrias implementadas em Java é necessário utilizar o pacote *j3d-vrml97.jar*. Testes realizados entre arquivos tipo **.obj* e **.wrl*, apresentaram vantagens como a conservação de escalas, posições e em alguns casos as geometrias.

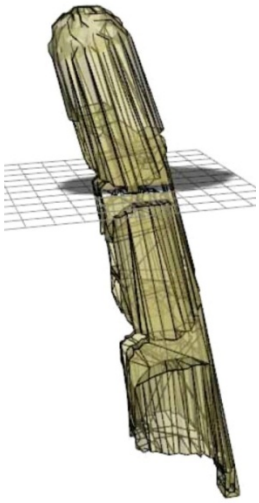
A Fig. 4.14 a, permite visualizarmos uma correta importação dos arquivos feitos em CAD utilizando o VRML, na Fig. 4.14 b, foi utilizado o mesmo procedimento mas utilizando arquivos de tipo OBJ, sendo que a medida que um novo arquivo era importado a relação de escala foi perdida e cada objeto importado, ficava de maior tamanho e sua posição original alterada, em

algumas ocasiões a informação das superfícies foi perdida durante o processo de exportação como se apresenta na Fig. 4.14 c.



a) Importação em Java de arquivo OBJ

b) Importação em Java de arquivo OBJ de forma correta



c) Importação errada de arquivo de CAD para OBJ com perda de informações.

Fig. 4.14: Resultados de importação e exportação de alguns arquivos em Java.

4.2.4. Edição de geometrias

Para o projeto de peças mecânicas, um *software* especializado é utilizado, permitindo assim a reconstrução de partes, no caso de não ter os planos dos modelos originais a reconstrução acarreará perdas de informação, qualidade, precisão e tempo.

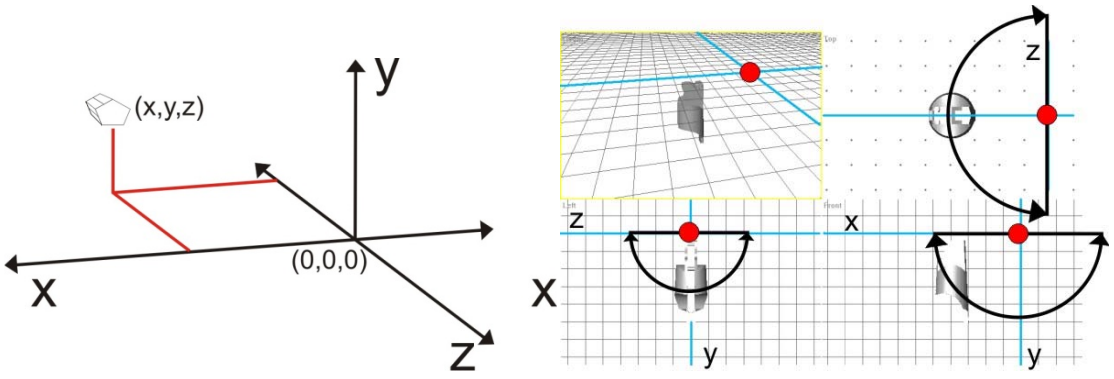
A partir da definição do VRML, algumas considerações deverão ser realizadas devido às características do mesmo. O primeiro é garantir que a exportação seja realizada para o padrão 2.0, pois a mesma apresenta um melhor suporte para algumas operações de geometrias (extrusão, triangulação, etc.). Caso o software permita uma triangulação das peças, podemos obter uma melhor geometria após a sua exportação, e por ultimo deve-se revisar a escala e unidades, isto para garantir que as peças conserva as proporções em relação ao modelo original.

O ambiente virtual VRML não apresenta um padrão definido de unidades, conseqüentemente, se a unidade de medida é escolhida de forma errada um objeto cujo comprimento é de 1 m pode aparecer como de 1 cm.

Após a exportação do arquivo VRML, o passo seguinte é a edição da localização no espaço virtual, pois a mesma pode ser alterada, dependente do programa no qual os modelos foram implementados. Inicialmente, o ambiente considera que as peças exportadas terão um deslocamento e um movimento de rotação em relação aos eixos (*x, y, z*), isto é normal, pois a peça faz parte de um conjunto e conserva suas relações de localização com o mesmo.

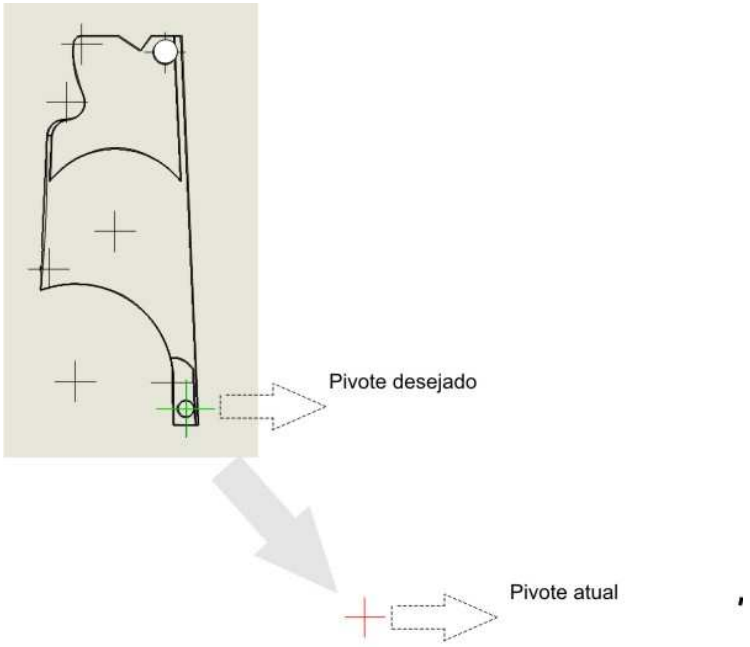
Se estas partes são introduzidas em Java3D o conjunto vai ficar aparentemente bem montado, mas o problema é que os origens de cada elemento vão ficar nas coordenadas (0,0,0) do universo VRML de forma que, se for necessário efetuar uma rotação ou movimentação de um determinado componente, este vai realizar uma ação em relação ao seu deslocamento inicial, pois o ponto de origem fica sempre na origem do VRML (Fig. 4.15a c). Para se obter uma articulação no lugar correto, é preciso conhecer todas as medidas e pontos de rotação de cada elemento do dispositivo (Fig. 4.15b), e isto pode ser realizado de duas maneiras distintas, através

da edição do código VRML ou com uma ferramenta de proprietária dedicada (*authoring*) de VRML.



a) Deslocamento da geometria exportada para VRML.

b) Articulação de rotação de uma peça sem edição.



c) Movimento de rotação desejada para a articulação atual.

Fig. 4.15: Deslocamento, rotação e posição ideal da articulação para cada peça.

4.3. Implementação do modelo em Java3D

Para a implementação de um ambiente virtual tomou-se como base o *scenegraph* da Fig. 4.10, estabelecendo-se uma hierarquia de geometrias (Fig. 4.16). Os valores das articulações são determinados conforme as dimensões dos elementos referenciados do modelo 3D que o precede. Modelos 3D baseados nos ossos de uma mão humana foram implementados em *software* especializado de modelagem 3D e animação.

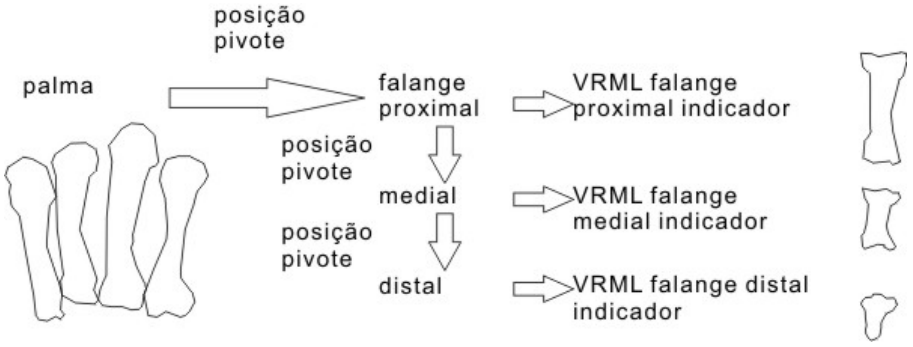


Fig. 4.16: Hierarquia das geometrias implementada em Java.

Um passo importante a ser considerado são os sistemas de coordenadas das peças, pois os mesmos podem variar dependendo do software no qual foram implementados, onde se pode não obter uma origem padrão no ambiente virtual em Java, já que no momento de importar as partes elas permanecem com os eixos em Java, e ao mesmo tempo, herdam as características do elemento do qual são filhos. Assim se uma peça efetua um movimento de rotação num determinado eixo, os filhos também terão essa mesma rotação. O Algoritmo 4 apresenta a criação das transformações para rotação e deslocamento da geometria e do articulação, seguido se aplica o deslocamento em *x y z* correspondente, as rotações se aplicam de forma individual, por exemplo, primeiro *x y z* ou em qualquer ordem se tiver, depois se devem habilitar a escritura e leitura das transformações, finalmente se aplica a hierarquia proposta, neste caso a geometria n (falange proximal) e a geometria n+1 (falange medial) junto com o articulação da geometria n+1 são usadas obtendo que o articulação n+1 e filho da geometria n, e a geometria n+1 é filha do seu articulação.

```
4.1  //Transformação para rotar e trasladar pivotes;
4.2  Transform3D rotaçao geometria n = new Transform3D();
4.3  Transform3D traslaçao geometria n = new Transform3D();
4.4  Transform3D traslaçao pivote geometria n = new Transform3D();
4.5  //translação;
4.6  traslaçao pivote geometria n.set(new Vector3f(x,y,z));
4.7  traslaçao pivote geometria n.rotX(angulo);
4.8  //Se aplica a translação e rotação do pivote a geometria;
4.9  traslaçao pivote geometria n.mul(traslaçao geometria n);
4.10 //Termino de rotação e deslocamento de pivote;
4.11 //Activação das transformações para a geometria;
4.12 tg geometria n = new TransformGroup();
4.13 tg geometria n.setCapability(TransformGroup.ALLOW TRANSFORM READ);
4.14 tg geometria n.setCapability(TransformGroup.ALLOW TRANSFORM WRITE);
4.15 //Fim da activação das transformações para a geometria;
4.16 //Criando uma geometria filhia da outra respeito a seu pivote;
4.17 tg geometria n+1.addChild( geometria n+1) );
4.18 tg geometria n.addChild(pivote tg n+1);
4.19 pivote tg n+1.setTransform(traslacao pivote geometria n+1);
4.20 pivote tg n+1.addChild(tg geometria n+1);
```

Algoritmo 4: Deslocamento e rotação de articulações.

4.4. Desenvolvimento do aplicativo

Normalmente, os modelos 3D são suficientes para se ter um ambiente 3D completo, onde para poder interagir com o mesmo é necessário o desenvolvimento de um código que permita a integração do 3D com o usuário e a planta física. Considerando o tipo de operação realizada com a aplicação, as entradas e saídas do sistema são determinadas da seguinte forma:

- Entradas
    - Ângulo de rotação para cada dedo.
    - Forma de preensão: esférica, planar ou cilíndrica.
- Saídas
    - Movimento de cada dedo no ambiente 3D.
    - Preensão no ambiente 3D.
    - Arquivo com informação de preensão para a planta física.

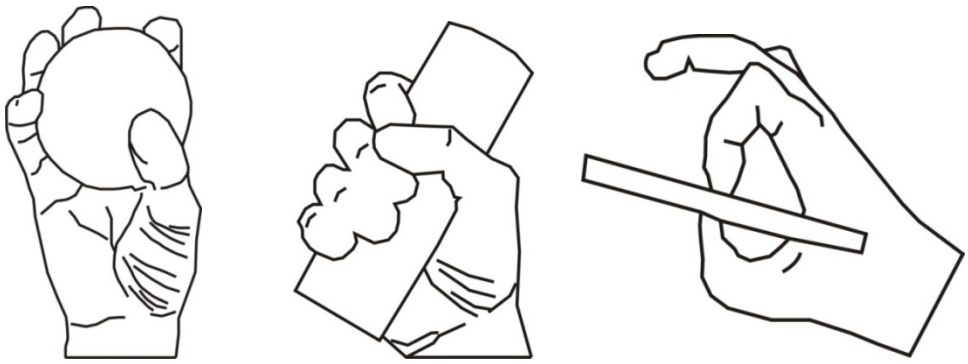
- Vínculo a uma janela de execução de tarefas.

A partir da especificação dos parâmetros definidos anteriormente, são estabelecidos os dispositivos de entrada e saída, onde neste caso serão os componentes de um computador padrão, como é o teclado, monitor, mouse e portas padrão USB. Já que os anteriores dispositivos estão suportados com as bibliotecas em Java, plug-ins ou software adicional não é necessário tanto para o desenvolvimento quanto para o usuário.

4.4.1. Definição da operação do dispositivo virtual e navegação

Assim como a mão humana, através de movimentos individuais de cada dedo pode ser gerado um tipo de preensão para preensão de um determinado objeto ou executar algum tipo de tarefa, onde o dispositivo antropomórfico virtual vai operar de acordo com o modelo implementado pelo usuário.

Os testes iniciais foram realizados com uma estrutura de ossos o comportamento similar ao de uma mão humana. Desta forma cada dedo terá a capacidade de estar em sua posição de flexão ou extensão, e posteriormente devem-se especificar os movimentos a serem realizados pelos mesmos em função das formas de objetos e capacidade de preensão dos mesmos, já que se procura uma solução genérica, considerando as capacidades da mão. Conseqüentemente, foram especificados três tipos de elementos mais comuns para preensão de objetos: esféricos, planos e cilíndricos (Fig. 4.17).



a) Esférico b) Cilíndrico c) Plano

Fig. 4.17: Principais formas de preensão de objetos através de uma mão humana.

Para conseguir uma rotação semelhante à da mão humana, uma relação dos ângulos é estabelecida de forma que as falanges mediais e distais realizarão um movimento de rotação quando a falange proximal efetuar um movimento de rotação, e de forma semelhante acontecendo com o polegar. Para obtenção das pré-formas de preensão, deverão ser definidas as rotações de cada elemento, assim como uma posição de *home* para retornar à posição corresponde à mão aberta.

Para garantir que usuário possa navegar no espaço 3D, as interações através do teclado e mouse deverão ser configuradas, conforme é apresentado no Algoritmo 5.

```
5.1  //Deslocamento da câmera;
5.2  t3d = new Transform3D();
5.3  vec = new Vector3d(0, 10, 30);
5.4  t3d.setTranslation(vec);
5.5  viewtrans.setTransform(t3d);
5.6  //Espaço de navegação;
5.7  BoundingSphere bounds = new BoundingSphere(new Point3d(), 100.0);
5.8  //Rotação pelo mouse;
5.9  MouseRotate rotator = new MouseRotate(MouseBehavior.INVERTINPUT);
5.10 rotator.setTransformGroup(viewtrans);
5.11 rotator.setSchedulingBounds(bounds);
5.12 rotator.setFactor(0.007);
5.13 objRoot.addChild(rotator);
5.14 //Translação pelo mouse;
5.15 MouseTranslate translator = new MouseTranslate(MouseBehavior.INVERTINPUT
5.16 translator.setTransformGroup(viewtrans);
5.17 translator.setSchedulingBounds(bounds);
5.18 translator.setFactor(0.007);
5.19 objRoot.addChild(translator);
5.20 //zoom pelo mouse;
5.21 MouseZoom zoomer = new MouseZoom(MouseBehavior.INVERTINPUT);
5.22 zoomer.setTransformGroup(viewtrans);
5.23 zoomer.setSchedulingBounds(bounds);
5.24 zoomer.setFactor(0.007);
5.25 objRoot.addChild(zoomer);
5.26 //Navegação pelo teclado, zoom e rotação respeito o eixo y;
5.27 KeyNavigatorBehavior keyNavigator = new KeyNavigatorBehavior(viewtrans);
5.28 keyNavigator.setSchedulingBounds(bounds);
5.29 objRoot.addChild(keyNavigator);
5.30 return objRoot;
```

Algoritmo 5: Deslocamento da Câmera e navegação através do teclado e mouse.

4.4.2. Diagramação da aplicação

Para representar o processo que devera ser desenvolvido como estrutura da aplicação, foi implementado um diagrama de fluxo de informações (Fig. 4.18) como base prévia para a escritura do código.

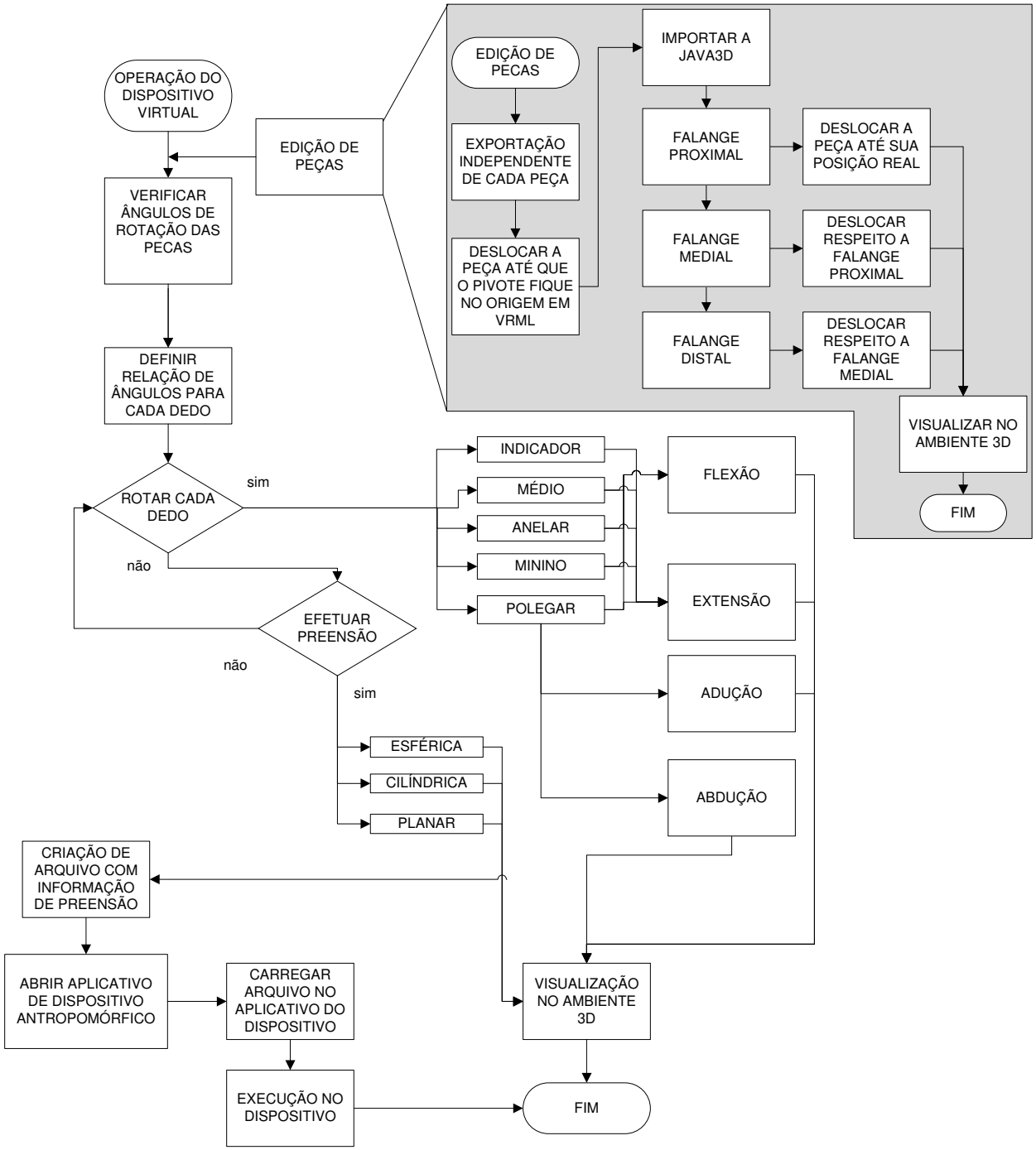


Fig. 4.18: Diagrama de fluxos de informações.

4.4.3. Interface com o usuário

Para o desenvolvimento da interface com o usuário (GUI), procurou-se manter o foco principal no ambiente virtual (*Canvas3D* denominado em Java) com o objetivo de que o usuário possa visualizar em detalhes o dispositivo. No lado esquerdo e no direito encontram-se os botões que permitem à operação individual de cada dedo, as pré-formas de preensão estabelecidas com seu retorno a posição *home*, assim como o botão para abrir o aplicativo para execução de tarefas de preensão e trajetória do robô industrial, como parte do funcionamento *online* e etapa de experimentação como é apresentado na Fig. 4.19.

Fig. 4.19: Proposta de interface com o usuário.

4.5. Instrumentação virtual

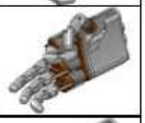
Para realizar a comunicação e envio de dados ao dispositivo real, ferramentas de instrumentação virtual são utilizadas já que através de seus componentes a operação pode ser

feita. O sistema está composto pelo aplicativo para gerenciamento da informação de preensão e IO, duas placas de aquisição de dados e os respectivos circuitos de condicionamento de sinal.

4.5.1. Informação de preensão

Como característica para o planejamento de tarefas, um arquivo é criado tendo a informação do respectivo tipo de preensão, baseado nas pré-formas escolhidas (objetos esféricos, cilíndricos e planar de tamanho fixo). Já que a planta física está associada ao dispositivo antropomórfico MUC-I (mão antropomórfica implementada no LAR-UNICAMP) (AVILES et al., 2007) (AVILES et al., 2008), o arquivo a ser implementado deverá estar em concordância com o padrão requerido pelo software para a movimentação da mão. A tabela de preensões está baseada em valores numéricos equivalentes com o tempo de rotação do motor de cada mecanismo responsável pela realização dos movimentos de flexão e extensão (valores de ângulos normalizados), conforme é apresentado na base de dados para o dispositivo (Tabela 4.4).

Tabela 4.4: Base de dados de preensões do dispositivo MUC-I.

Nível de oposição		Ângulos Normalizados						Posição dos dedos na preensão
		M1	M2	M3	M4	M5	M6	
1	Grande	0.5	0.5	1	1	1	1	
6	Médio	0.7	0.5	1	1	1	1	
20	Pequeno	0.7	0.5	1	0	0	0	
14	Pequeno	0	0.7	0	1	1	1	
11	Pequeno	0	0.7	1	0	0	0	
9	Médio	0.5	0.5	1	1	0	0	

M_i onde i é o número de motor do dispositivo antropomórfico MUC I.

A relação dos motores com os respectivos dedos está apresentada a continuação:

- Motor 1 (M1): abdução e adução do polegar.
- Motor 2 (M2): flexão e extensão do polegar.
- Motor 3 (M3): flexão e extensão do indicador.
- Motor 4 (M4): flexão e extensão do médio.
- Motor 5 (M5): flexão e extensão do anelar.
- Motor 6 (M6): flexão e extensão do mínimo.

Através desta tabela podemos observar que somente a informação dos três tipos de preensões escolhidas foi considerada, de modo que o arquivo implementado só possui os valores numéricos respectivos de cada movimento e este será gerado no momento de utilização do botão correspondente a aplicação.

Para realizar a comunicação com a planta física e o aplicativo foi implementado um arquivo de texto no formato ASCII para determinar qual vai ser a forma de preensão a ser realizado pelo dispositivo antropomórfico. Após o robô industrial enviar uma informação correspondente a trajetória de aproximação do objeto e realização da operação de preensão por parte da mão antropomórfica implementada no elemento terminal do robô.

Para a criação do arquivo, tomou-se como base o padrão feito baseado na Tabela 4.4, para assim movimentar cada motor do dispositivo MUC (Fig. 4.20), e conseguir a execução da preensão desejada. Assim, tem-se o uso de dois bits para gerar as rotações no sentido horário e antihorário, as quais vão ser visualizadas na MUC com os dedos fazendo a flexão ou extensão, e para polegar adicionalmente a abdução e adução.

Tomando a informação das preensões geram-se as seguintes linhas de dados para ser executadas nos motores do dispositivo antropomórfico com o objetivo de efetuar os três tipos de preensão escolhidos e a posição inicial na qual os dedos estão na posição estendida. O arquivo de texto não vai só conter uma linha de informação, vai conter no mínimo dois, uma para o

acionamento dos motores e execução da preensão desejada, neste caso o dispositivo tomaria o objeto relacionado com a geometria de preensão, e a outra linha faz a execução do movimento antagônica de forma que o objeto é deixado livre e os motores giram para voltar às articulações a sua posição estendida. Os valores à ser enviados aos motores podem ser normalizados para representar as posições do dedo desde seu estado completamente estendido (0) até o completamente flexionado (1). Desta forma a Tabela 4.5 apresenta a informação a ser inclusa no arquivo de texto para cada preensão.

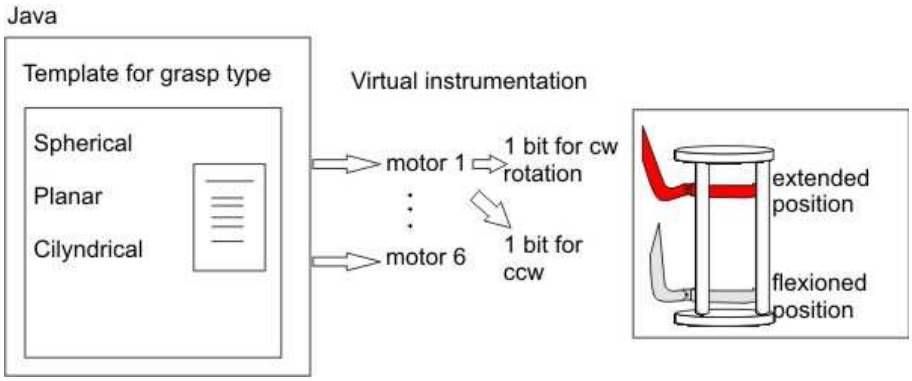
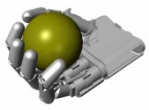
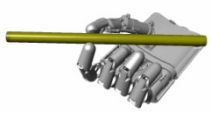
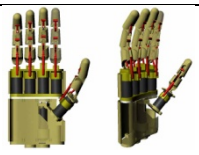


Fig. 4.20: Bits para rotação do motor no dispositivo MUC

Tabela 4.5: Informação de preensão de exemplo para gerar desde o aplicativo virtual.

Preensão	Ângulos normalizados						Representação virtual esperada
	M1	M2	M3	M4	M5	M6	
Esférica	0.5	0.5	1	0.8	0.8	1	
Planar	1	0.8	1	0/1	0/1	0/1	
Cilíndrica	1	0.8	0.7	0.7	0.7	0.7	
Mão aberta	0	0	0	0	0	0	

4.5.2. Gerenciamento de dados

Para o gerenciamento de dados tomou-se como base o aplicativo próprio da MUC (Fig. 4.21), para adequar ele às necessidades deste projeto, as entradas e saídas são analisadas obtendo:

- Entradas:
  - Arquivo de texto com a informação de rotação para cada motor.
  - Sinal do controlador do robô industrial para execução de um tipo de preensão.
- Saídas:
  - Envio de sinais aos motores.
  - Sinal para o controlador do robô industrial para execução de trajetória.

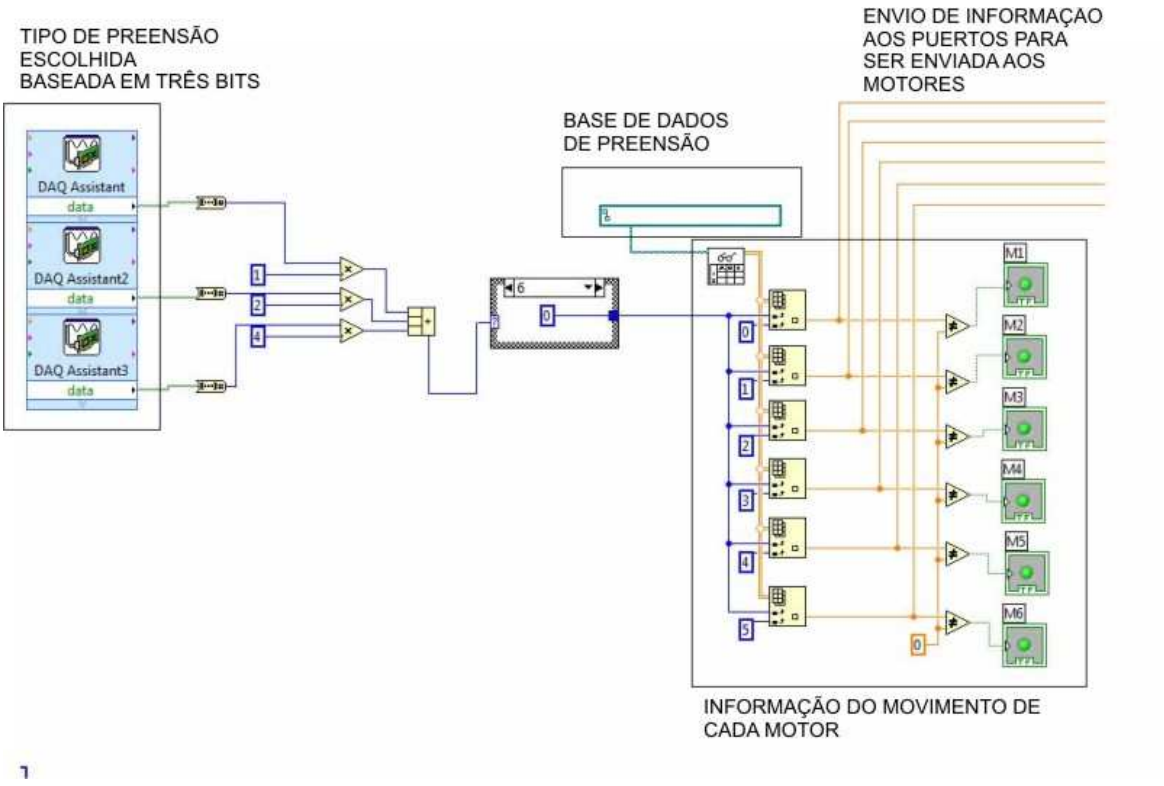


Fig. 4.21: Estrutura de instrumentação virtual da MUC.

Para o caso deste projeto, a informação de preensão não vai ter sua origem no aplicativo do robô senão do arquivo de preensão gerado através do aplicativo virtual, mudando as entradas e saídas como na Fig. 4.22, e o robô vai aguardar um sinal do aplicativo enviado pelo usuário para executar a trajetória, uma vez no ponto de agarre do objeto outro sinal para fazer a preensão escolhida, após a execução da trajetória de liberação de objeto e com a mão na posição aberta um ultima sinal faz o manipulador robótico ir para sua posição de inicio.

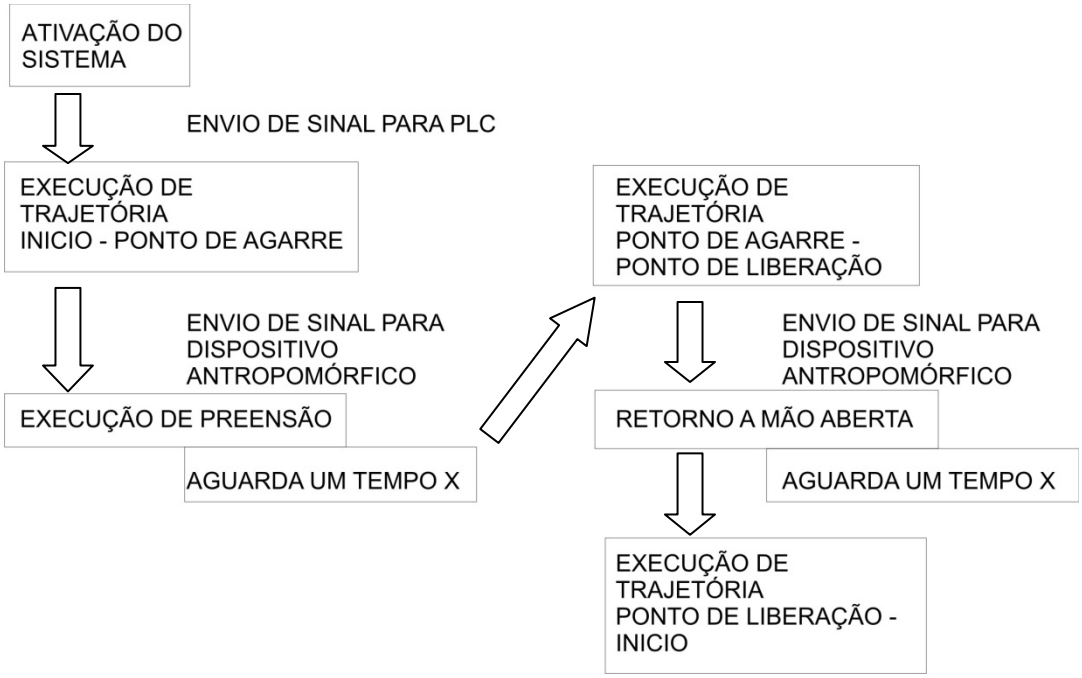


Fig. 4.22: Diagrama do aplicativo para execução na planta física

4.6. Características da MUC para o agarre de objetos

Dois aspectos da MUC limitam a range de objetos que podem ser tomados por este dispositivo, um deles o material com o qual são construídos os dedos e o outro as geometrias da sua falange distal.

Os dedos estão feitos de dois materiais diferentes, do indicador para o mínimo de teflon e o polegar de ABS, devido ás características mecânicas de cada um deles o atrito é muito baixo dificultando na preensão de objetos feitos de materiais com propriedades similares.

Dois tipos de geometrias diferentes têm os dedos da MUC, uma é a mesma para os dedos desde o indicador até o mínimo, e outra para o polegar (Fig. 4.23). No caso da geometria cilíndrica e esférica na ponta a área de contato é mínima sem importar o objeto, para o polegar a falange tem uma face plana o qual incrementa a área de contacto.

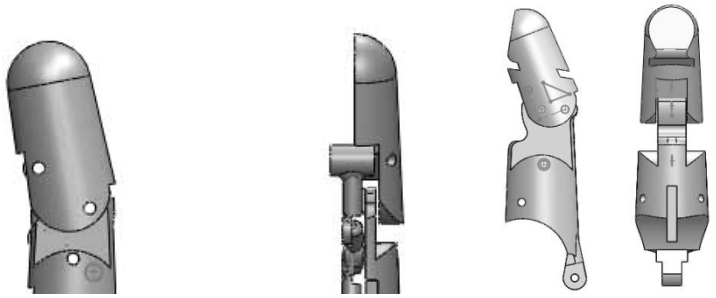


Fig. 4.23: Falanges distais do dedo indicador e o polegar, vista frontal e lateral.

4.7 Comentários finais do capitulo

Neste capítulo foram apresentados os principais componentes para o desenvolvimento de um ambiente virtual, enfatizando conceitos referentes a modelagem de dispositivos antropomórficos do tipo *gripper*, e ambientes de operação para os mesmos. Também foram abordadas características de preensão referentes ao dispositivo físico disponível para os testes. No próximo capitulo será apresentado os principais resultados obtidos durante o processo de implementação experimental da aplicação proposta neste capitulo e testes realizados para sua validação.

Capitulo 5
Validação Experimental e Resultados

Neste capitulo serão apresentados os principais resultados obtidos durante o processo de implementação experimental da aplicação em estudo, obedecendo à metodologia proposta no capitulo anterior deste trabalho. Para validação foi realizado uma série de testes, permitindo assim uma avaliação completa do sistema desenvolvido. Os resultados obtidos são organizados e apresentados conforme as áreas estabelecidas para abordar o problema.

5.1. Plataforma experimental implementada

Para validação e testes do sistema desenvolvido, foi implementado uma plataforma experimental no Laboratório de Automação Integrada e Robótica da Faculdade de Engenharia Mecânica da UNICAMP subdividida em componentes locais e remotos.

5.1.1. Componentes

Para a parte local tem-se um computador, no qual vai ser executado o *software* de manipulação virtual de objetos e o aplicativo para enviar a informação de preensão para o dispositivo antropomórfico. Os componentes remotos são compostos pelo *hardware* como o robô, o dispositivo antropomórfico e um computador para gerenciar e controlar informações de entrada e saídas digitais (I/O), permitindo assim uma interação entre os diferentes componentes

da arquitetura proposta. A Fig. 5.1 apresenta a arquitetura de integração proposta para o *hardware* envolvido nesta plataforma experimental.

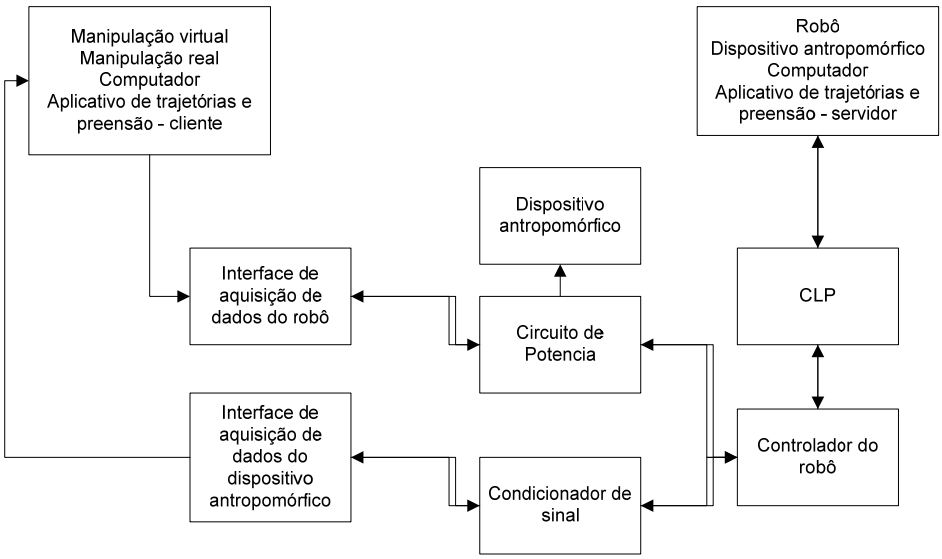


Fig. 5.1: Arquitetura proposta para a Plataforma experimental.

Para os componentes locais e remotos de *software* foram desenvolvidos três pacotes computacionais, considerando que do ponto de vista remoto duas aplicações foram necessários:

i) Desenvolvimento de aplicativo para a manipulação virtual que cria um arquivo com a informação de preensão escolhida pelo usuário,
ii) Desenvolvimento de um aplicativo para o envio dos comandos e execução da trajetória do robô e dispositivo antropomórfico de preensão.

As comunicações entre as diferentes etapas foram realizadas da seguinte forma (Fig. 5.2)

- Unidirecional: Desde o aplicativo de manipulação 3D para o aplicativo de envio de informação de preensão para o dispositivo antropomórfico.
- Bidirecional: Desde o aplicativo de preensão o qual envia um sinal para execução de trajetória e forma de preensão e com o aplicativo de geração de trajetórias do robô, que aguarda um sinal do dispositivo para a execução de determinada operação do robô e posterior retorno do robô para a posição *HOME*.

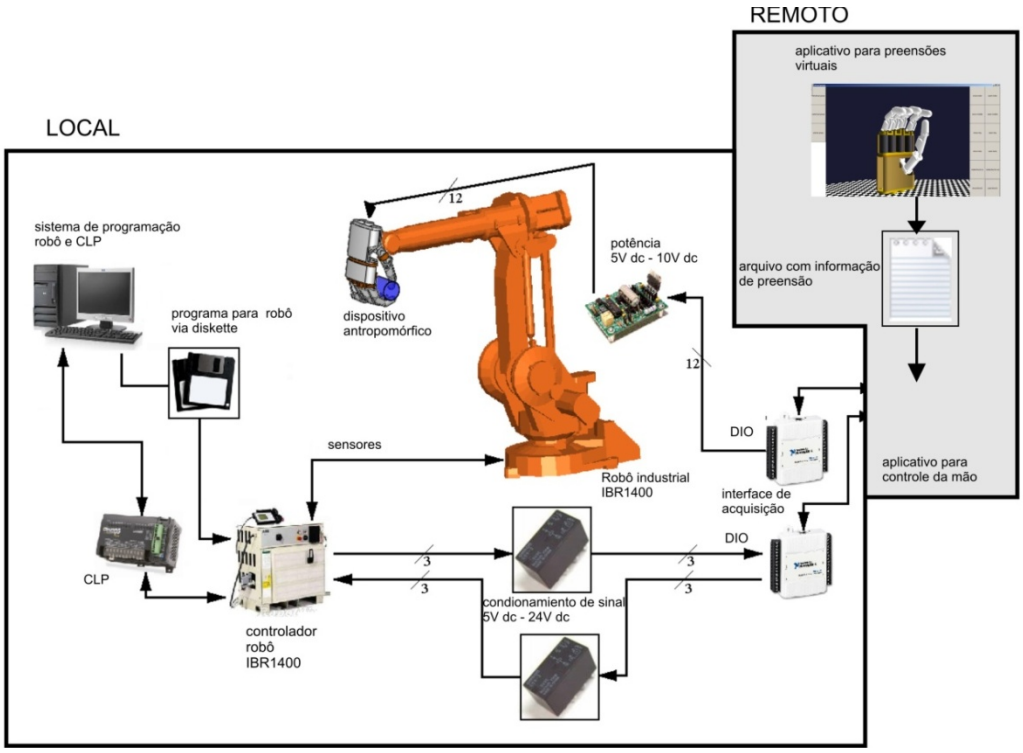


Fig. 5.2: Esquema de comunicação entre os componentes da planta de testes.

A configuração de cada elemento é feita baseada nas necessidades do projeto, assim para a interpretação e envio de informação para o dispositivo antropomórfico desde o arquivo de preensão é feita a través do aplicativo proposto em Labview (Apêndice E), o dispositivo que administra as tarefas junto com os sinais de entrada e saída da mão entre o robô industrial e a mão mecânica e uma CLP configurada como se apresenta no Apêndice D. Finalmente para execução de trajetórias o robô industrial ABB IBR1400 é configurado como na Fig.5.3 e sua programação se apresenta no Apêndice C.

Fig. 5.3: Espaço de trabalho e trajetória da mão para tomar e liberar objetos.

5.1.2. Configuração do espaço de testes

Fisicamente o espaço de testes está composto de uma mesa através da qual se disponibilizam os objetos representativos das três geométricas escolhidas para ser tomadas pelo dispositivo físico (Fig. 5.3), desta forma obtêm-se duas zonas de interesse sobre a mesa, uma para pegar o objeto e outra para soltar. O estabelecimento das coordenadas destas duas zonas é importante já que permite realizar a programação da trajetória para o robô. Da Fig. 5.3, tem-se que o cubo vermelho é a zona de preensão e a amarela a de liberação, as linhas pretas representam a trajetória que a mão vai seguir.

5.1.3. Seleção de objetos baseados nas características da MUC

Devido a que objetos com baixo atrito podem não ser agarrados pelos dedos da MUC, materiais que contrastem essa característica são utilizados, desta forma foi selecionada a espuma de poliuretano, já que não só permite obter as geometrias desejadas senão também um alto atrito e como ela se deforma quando tiver pressão facilita o agarre (Fig. 5.4).

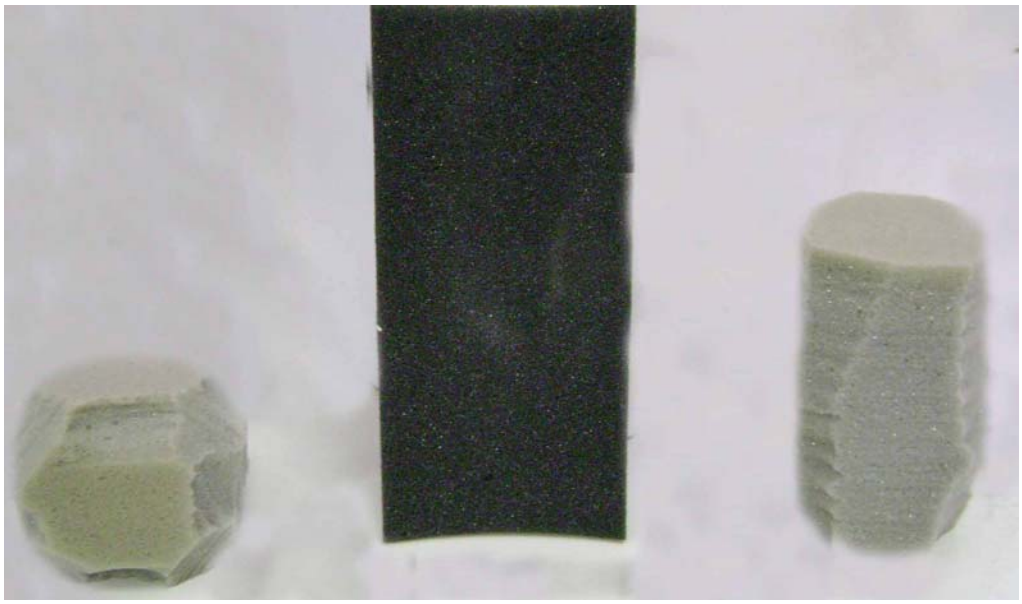


Fig. 5.4: Objetos em espuma para preensão.

5.2. Resultados experimentais

5.2.1. Geometrias 3D

Para realizar a validação do ambiente virtual, geometrias 3D foram implementadas a partir de um software de CAD e modelagem 3D, e posteriormente utilizadas obedecendo a um processo de edição e determinação dos eixos coordenados (Fig. 5.5). É importante observar que no caso de não editarmos as peças ou não realizar de forma correta as relações dos eixos de coordenadas, podem-se obter problemas na visualização, pois as mesmas poderão apresentar uma orientação, e conseqüentemente realizarem uma rotação de forma errada.

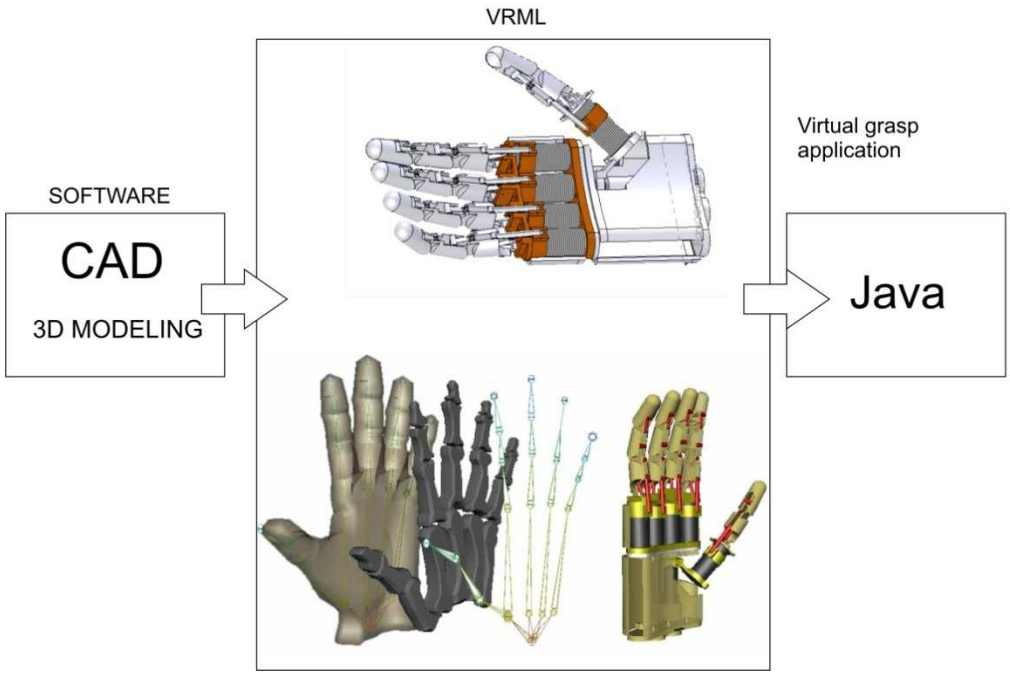


Fig. 5.5: Processo de edição dos arquivos 3D.

Após a implementação dos modelos propostos anteriormente, para as geometrias geradas através de software de modelagem 3D, foi implementado o ambiente virtual apresentado na Fig. 5.7, enquanto que para a geometria gerada a partir de CAD, obteve-se o ambiente apresentado na Fig. 5.9.

5.2.2. Aplicativo do ambiente virtual

O primeiro teste realizado no aplicativo foi de tipo visual e funcional sobre a interface de usuário apresentada na Fig. 5.6. Três principais zonas dividem o espaço de trabalho, a primeira contem as pré-formas de preensão e o retorno para a posição inicial do dispositivo virtual, a zona dois contém o entorno virtual no qual o usuário através do mouse e teclado pode realizar navegação fazendo zoom, pan e rotação, e finalmente na ultima zona estão os botões que permitem movimentar cada dedo do indicador ao mínimo (cada um com a opção de abrir ou fechar o dedo na direção da palma da mão) em seu movimento de flexão e extensão e para o caso do polegar também abdução e adução. A interface é intuitiva e o espaço 3D permite fazer uma boa navegação e detalhe.

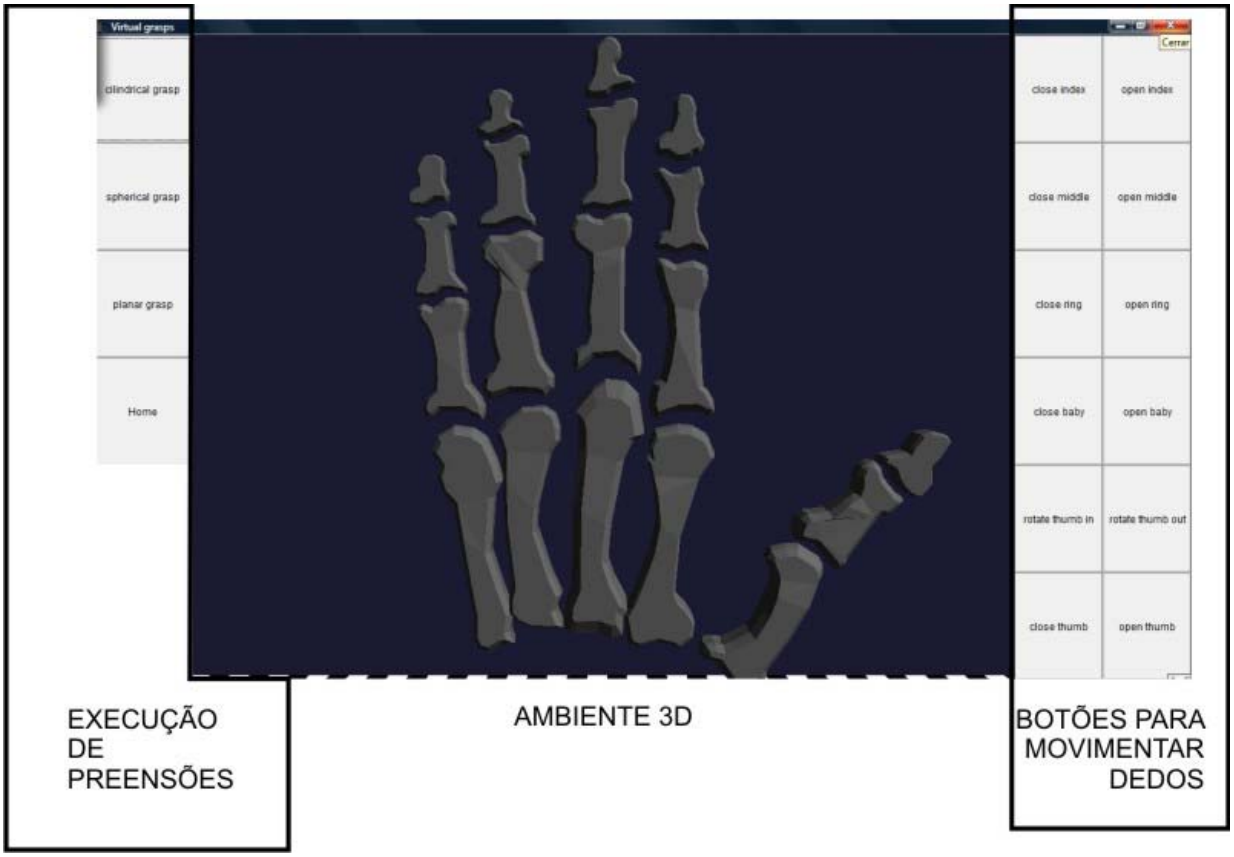


Fig. 5.6: Interface de usuário.

Para conseguir a movimentação dos dedos da mão antropomórfica virtual, os botões implementados na área de movimentação permitem operar os mesmos separadamente, ou seja, rotar um a um de forma independente permitindo gerar vários tipos de preensão através da flexão e extensão das falanges (Fig. 5.7).

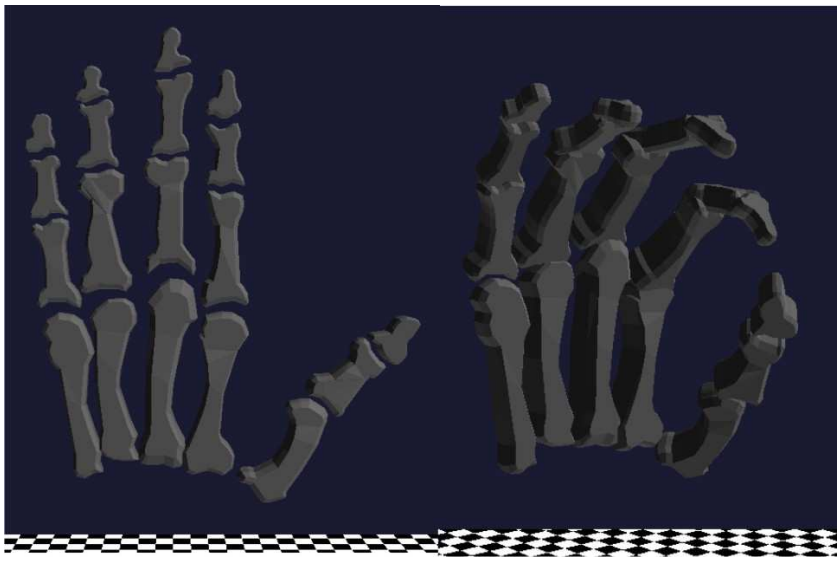
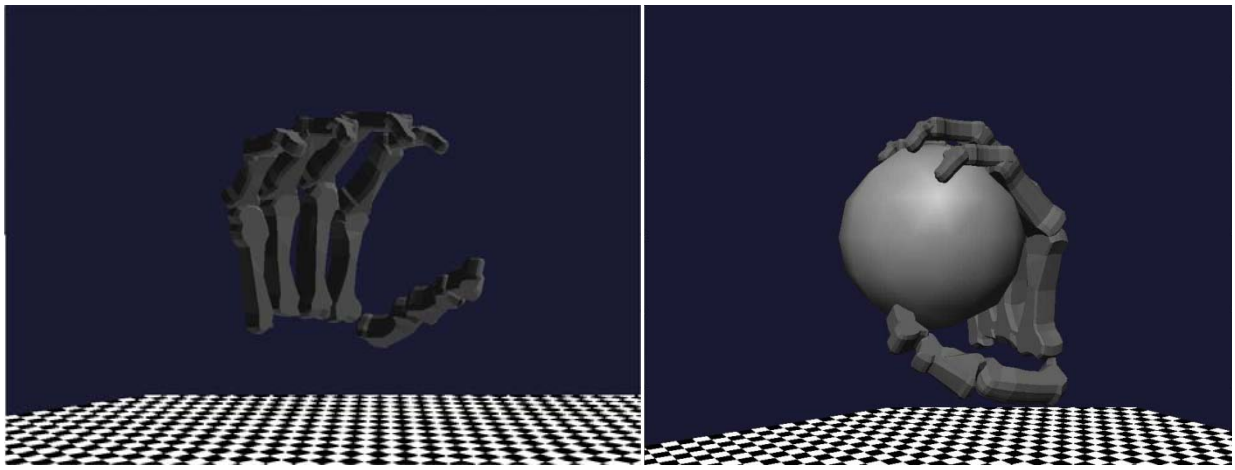


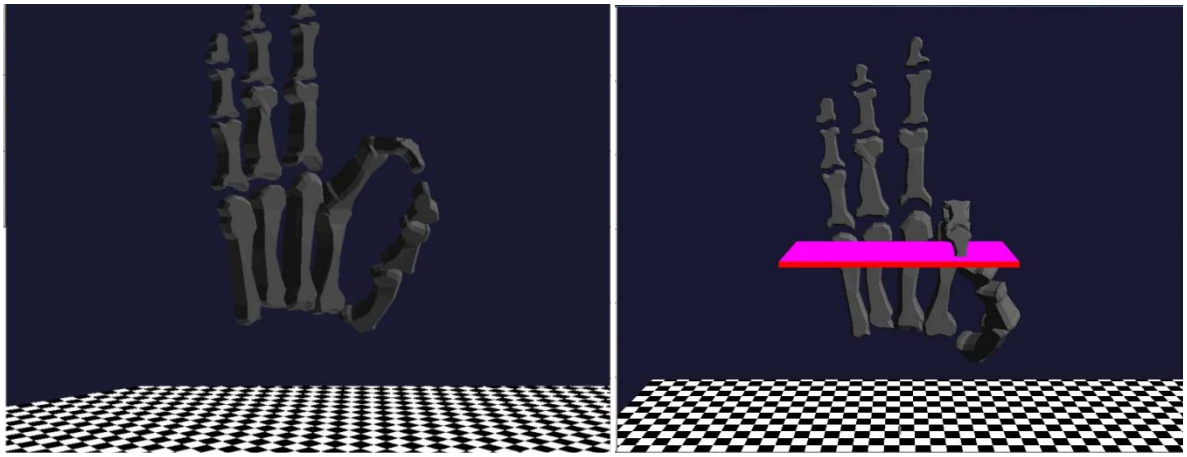
Fig. 5.7: Representação dos ossos de uma mão 3D em Java3D e movimentos dos dedos.

Para gerar as três formas de preensão propostas na aplicação experimental, três botões foram etiquetados como cilíndrico esférico e planar na área de execução de preensões que movimentam os dedos para posições pré-definidas. O botão de *home* permite retornar as geometrias para a posição correspondente a mão totalmente aberta. A Fig. 5.8 apresenta os resultados obtidos após de utilizar os botões de preensão.

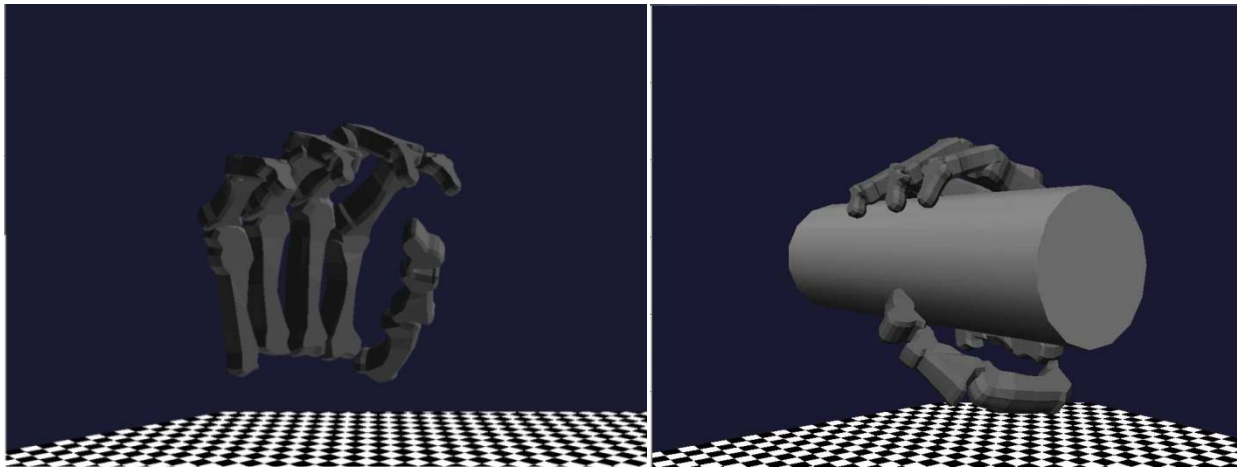
Considerando que o procedimento para carregar os arquivos VRML implementado é genérico, e a única diferença se encontra nas medidas, sistemas de referencia das articulações das juntas e o nível de detalhamento do modelo, como esta aplicação se concentra na visualização dos dedos e as capacidades para preensão dos diferentes objetos, muito dos componentes do dispositivo não necessitam ser implementado, podendo até ser apagados, isto permite aperfeiçoar o arquivo e melhorar sua navegação, pois o numero de objetos a serem implementados nesta aplicação são bem menores.



a) Esférico



b) Planar



c) Cilíndrico.

Fig. 5.8: Formas de preensão da garra virtual.

Para testar o comportamento da aplicação foram tomados os arquivos CAD do sistema de preensão antropomórfico - *MUC-I*, que faz parte de um projeto de tese de doutoramento na área de sistemas antropomórficos de preensão em desenvolvimento na UNICAMP.

Considerando a simplificação do modelo de dispositivo e sua configuração em linguagem Java para operar de forma semelhante a um sistema real, foram testados os movimentos, e os principais resultados obtidos são apresentados na Fig. 5.9.

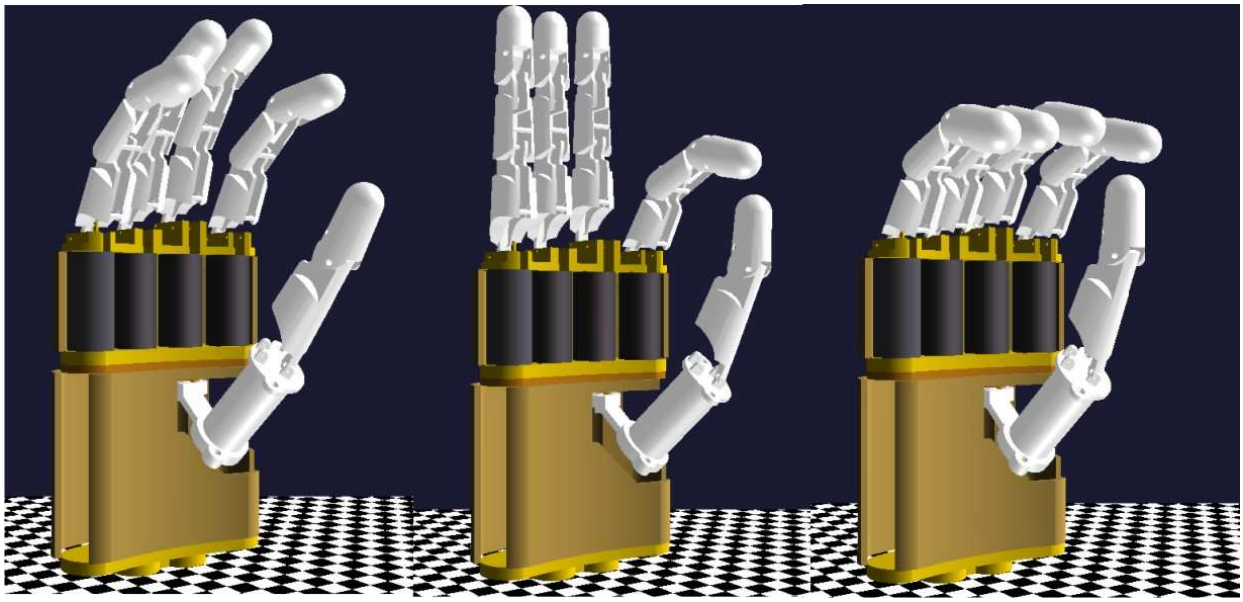


Fig. 5.9: Testes de preensão realizados através de um modelo 3D baseado num dispositivo antropomórfico real.

5.2.3. Testes realizados com o robô e dispositivo antropomórfico

Após configurar fisicamente os sistemas testou se a célula de trabalho obtendo como resultado o movimento de cada dedo conforme foi especificado no arquivo de dados. As geometrias de espuma facilitam a preensão por seu atrito e capacidade de permitir um pouco de deformação, as Fig. 5.10, Fig. 5.11 e Fig. 5.12 apresentam o agarre de um cilindro poligonal, um retângulo, e uma esfera poligonal. Devido ao controle em laço aberto da MUC os rangos de

movimento não podem exceder certos limites com o objetivo de evitar algum travamento mecânico.

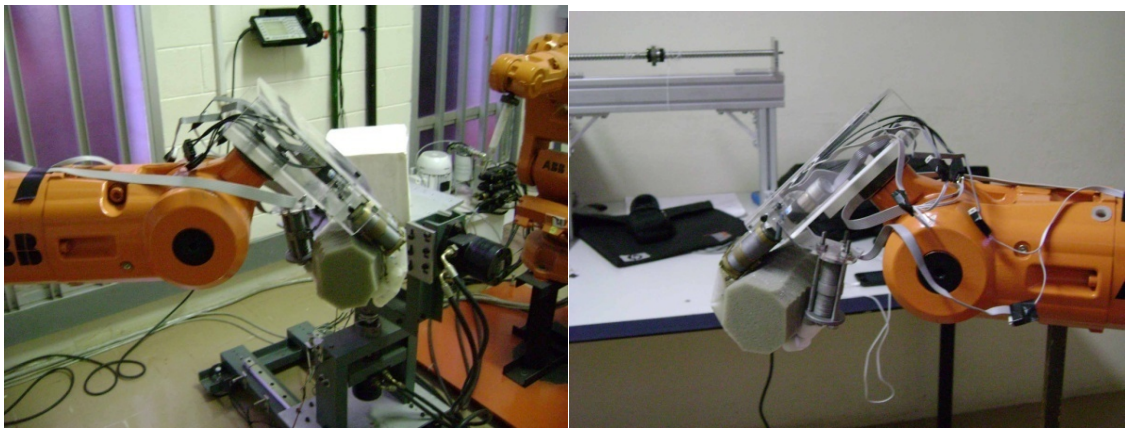


Fig. 5.10: Preensão de um objeto cilíndrico.

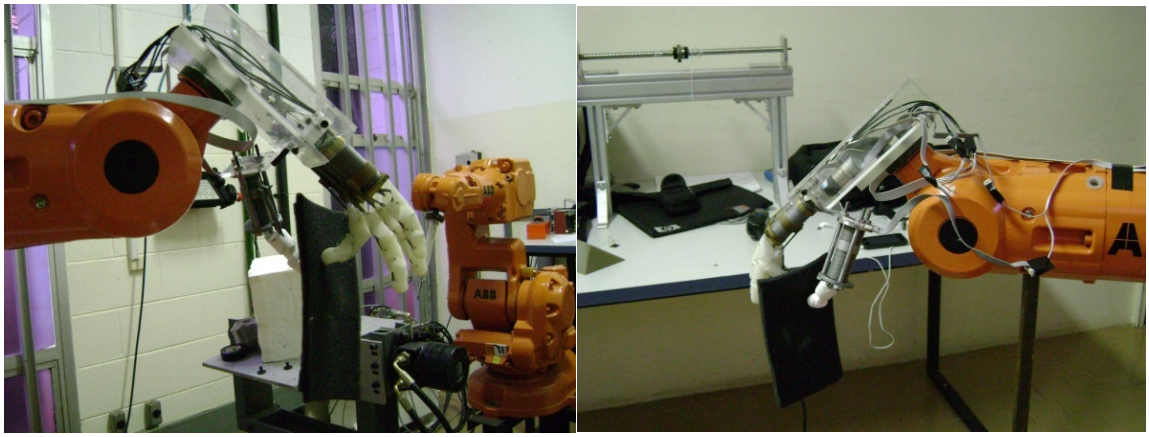


Fig. 5.11: Preensão de um objeto planar.

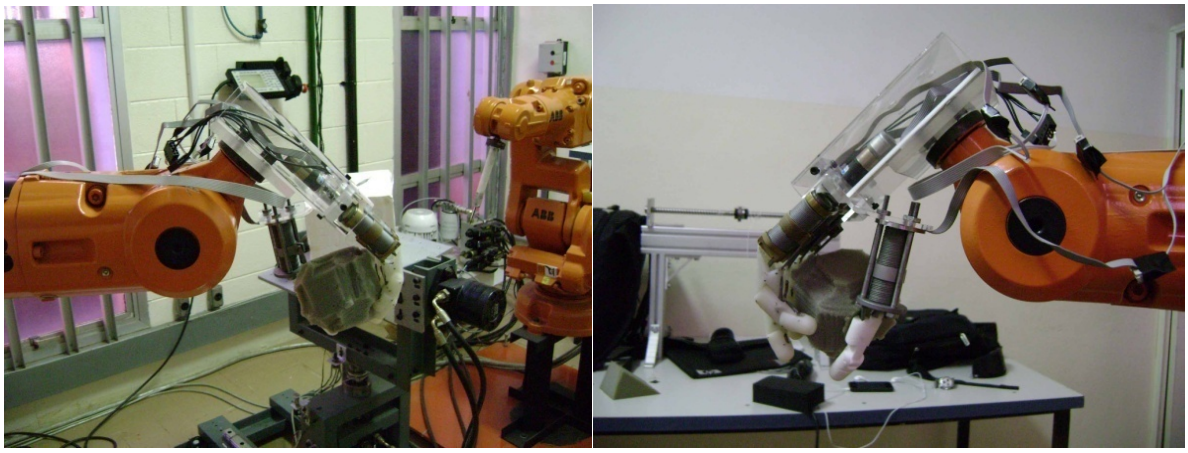


Fig. 5.12: Preensão de um objeto esférico.

5.3 Comentários finais do capitulo

Neste capitulo foram apresentados os principais resultados obtidos na implementação experimental da aplicação em estudo, onde um dispositivo antropomórfico de preensão foi colocado no elemento terminal de um robô industrial. Os resultados obtidos permitiram a validação deste trabalho de pesquisa, abrindo perspectivas futuras na área de dispositivos hápticos e aplicações de Realidade Virtual em Automação.

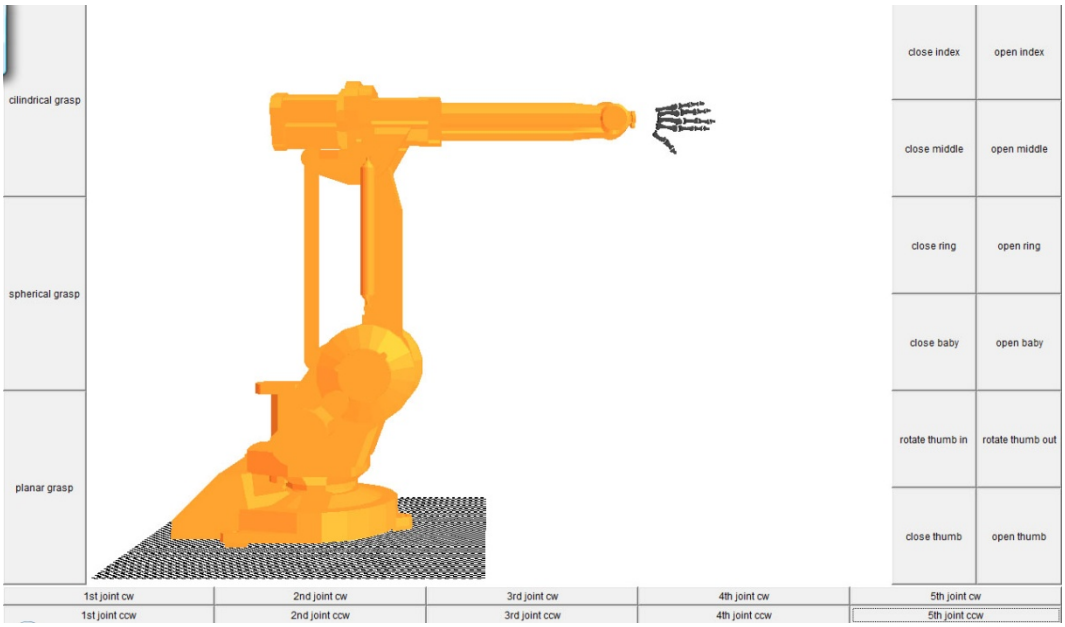
Capitulo 6
Conclusões Finais e sugestões para trabalhos futuros

Com os grandes avanços e pesquisa no desenvolvimento de dispositivos mecatrônicos antropomórficos baseados em mãos, geram-se necessidades que podem ter solução através dos conceitos de realidade virtual, permitindo assim a implementação de um aplicativo para realizar em um ambiente 3D, o planejamento de tarefas de preensão de objetos.

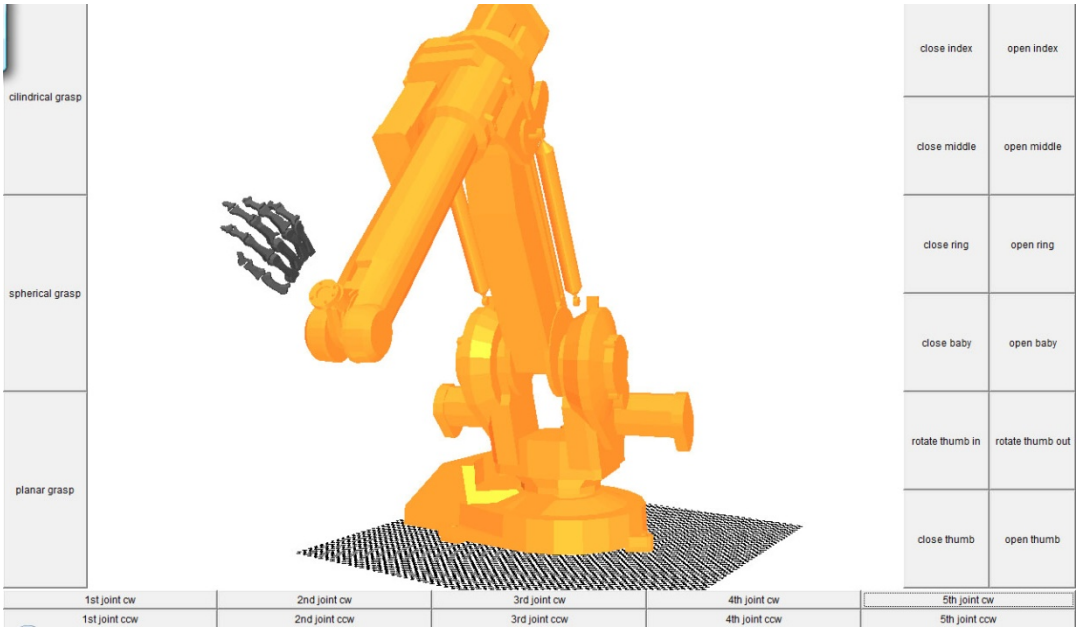
Esta dissertação de Mestrado apresentou o desenvolvimento de uma ferramenta para realizar manipulação de objetos em um ambiente virtual utilizando como plataforma de testes uma mão antropomórfica acoplada a um robô industrial. Ela permite visualizar três tipos de preensões pré-definidas ou gerar qualquer uma através do movimento independente de cada dedo do dispositivo antropomórfico no espaço virtual, característica útil para o entendimento do funcionamento móvel do dispositivo real e para ter acesso a ele quando não estiver disponível fisicamente para testes. Para aproveitar as práticas feitas no ambiente virtual, um arquivo com a informação de preensão é salvado para poder ser enviado à plataforma real permitindo realizar um planejamento de tarefas, desta forma acrescentando o uso da planta física através da realidade virtual.

A principal vantagem da metodologia proposta neste trabalho de pesquisa é a possibilidade de expandirmos a utilização deste aplicativo para obter uma solução mais robusta. Para comprovar esta característica, as geometrias CAD correspondente ao robô ABB-1400 foram obtidas em formato VRML na pagina WEB do fabricante do robô ABB. Após realizar o mesmo procedimento para a edição de peças para o dispositivo antropomórfico, as mesmas foram

colocadas dentro do aplicativo configurando cada junta para realizar as respectivas rotações e assim obter um movimento similar ao do robô industrial utilizado (Fig. 6.1).



a) Posição inicial do robô.



b) Movimentação de algumas juntas.

Fig. 6.1: Visualização 3D do Robô ABB em VRML.

A principal contribuição deste trabalho está na criação do protótipo de aplicação para visualização de diferentes tipos de preensão permitindo o uso do dispositivo antropomórfico virtual como se for o real, criando informação dos três diferentes tipos de preensão escolhidas, gerando independência do dispositivo físico caso não esteja disponível. O uso de ferramentas de baixo custo e uma metodologia que permite ser aplicado a outros dispositivos mecatrônicos, através da "virtualização" de dispositivos reais permite garantir o acesso destes ao usuário, para assim familiarizar-se e realizar tarefas *offline*.

6.1. Conclusões e contribuições de trabalho

Através do desenvolvimento das principais etapas delineadas nesse projeto de pesquisa, onde todos os objetivos inicialmente previstos foram atingidos, pode-se concluir:

- O modelo cinemático de um dispositivo antropomórfico idealizado através de um conjunto de mecanismos seriais permite analisar de forma escalonada dispositivos cujos graus de liberdade variem, de forma que o uso da sistemática de DH permite adicionar ou eliminar certas rotações de abdução, adução, flexão ou extensão para cada extremidade e devido à simplicidade do mecanismo oferecendo resultados satisfatórios.
- Para conseguir deslocar ou efetuar movimentos rotacionais de geometria 3D dentro do ambiente Java, não é necessário implementar o modelo cinemático direto, nem a forma geométrica através de DH, isto se deve a que o ambiente Java3D internamente utiliza matrizes de transformação para esta finalidade.
- Das soluções disponíveis como ferramentas para o desenvolvimento de um aplicativo de baixo custo, portável e escalonado, a linguagem de Java e Java3D possui todos os requisitos para cumprir essas necessidades já que sua compatibilidade com arquivos de geometria 3D e *rendering* é boa através de diferentes plataformas e equipes de computo.
- Dos formatos 3D mais utilizados entre os aplicativos de CAD e modelagem 3D que podem ser utilizados sem perder o nível de detalhamento na geometria e relações de tamanho dentro do Java e após a realização de testes de validação, o VRML é a opção que se mostrou mais adequada como formato de arquivo 3D utilizado.

- Para obter um nível de realismo no ambiente 3D, este depende dos modelos utilizados e da de edição realizado pelo usuário, sendo importante possuir as especificações de projeto e esquemas contendo detalhamento de medidas e acoplamentos, e que o projeto final implementado obedeça todas as especificações inicialmente estabelecidas.
- O uso do aplicativo e a criação de um arquivo contendo as informações de preensão permitem a sua utilização quando o dispositivo antropomórfico não estiver disponível, e no caso que esteja pronto, o arquivo é enviado para realizar a tarefa escolhida.
- A representação virtual não só do modelo do dispositivo senão das preensões permite ao usuário olhar o que acontece e planejar tarefas para enviar à planta real, já sabendo que esperar dela.
- As preensões escolhidas e o material dos objetos permitem realizar os agarres de forma exitosa já que estão conforme as capacidades do dispositivo antropomórfico.
- A integração do aplicativo virtual com um robô industrial e garra antropomórfica MUC, e dispositivos automatizados (CLP, interfaces E/S, potência) permitem ter uma célula integrada para realização de operações de *pick-and-place,* utilizando o conceito de plataforma colaborativa.
- As restrições estabelecidas na célula robótica composta pelo dispositivo antropomórfico MUC I e o robô industrial IBR1400, o aplicativo desenvolvido permite obter dependência da planta física caso algum componente não permita seu uso através do ambiente virtual.
- Os conceitos e metologia apresentados neste trabalho poderão ser utilizados facilmente em outras células de manufatura integrada para que através da RV possam acrescentar sua disponibilidade como ferramentas de treinamento ou ensino.

6.2. Trabalhos futuros

Considerando que os principais componentes de um sistema de realidade virtual, possam ser utilizados em grande diversidade de aplicações, este fato deverá ser levado em conta no desenvolvimento de trabalhos futuros, devido ás características do aplicativo, elas permitem seu crescimento em nível de ambiente virtual, interface, dispositivos de entrada e saída, e gerenciamento de informações, acrescentado a funcionalidade, interação e imersão do usuário.

A programação em linguagem Java permite acrescentar as propriedades do aplicativo devido a possibilidade de poder se comunicar através de *sockets* com outros aplicativos. A sua linguagem é orientada aos objetos, a qual permite otimizar o código, e conseqüentemente melhorar a sua execução.

No componente correspondente ao aplicativo pode-se procurar o envio da informação em tempo real para o dispositivo antropomórfico e a partir de um sistema de supervisão e controle, o usuário poderia visualizar a execução das tarefas.

Nos componentes de entrada e saída, poderíamos contemplar o desenvolvimento de um dispositivo háptico, com capacidade de manipular objetos em ambientes perigosos para homem, neste caso a realimentação de força seria indispensável para interação com o usuário final.

Virtualizar outras células robóticas ou automatizadas fazendo uso da arquitetura desenvolvida para execução de tarefas remotas, praticas *offline*, e planejamento de tarefas.

Referências Bibliográficas

AFSHARI, K. E.; PAYANDEH, S. Toward. implementation of java/vrml environment for planning, training and tele-operation of robotic. systems. In: Americas Conference on Information Systems-AMCIS. [S.l.: s.n.], 1999.

ALENCASTRE-MIRANDA, M.; GOMEZ, L. M. noz; RUDOMIN, I. Teleoperating robots in multiuser virtual environments. In: Proceedings of the 4th Mexican International Conference on Computer Science. [S.l.: s.n.], 2003.

ALEX, J.; VIKRAMADITYA, B.; NELSON, B. J. Teleoperated micromanipulation within a vrml environment using java. In: Intelligent Robots and Systems, 1998. Proceedings., 1998 IEEE/RSJ International Conference on. [S.l.: s.n.], 1998.

AVILES, O. et al. Development of a prehension system for robotics gripper application. In: 23rd ISPE International Conference on CAD/CAM, ROBOTICS & Factories of the Future, CARS & FOF 07. [S.l.: s.n.], 2007.

AVILES, O. F. et al. Five fingered antrhopomorphic hand design: Muc-1. In: 7th Brazilian Conference on Dynamics, Control and Applications. [S.l.: s.n.], 2008.

BELOUSOV, I. R.; CHELLALI, R.; CLAPWORTHY, G. J. Virtual reality tools for internet robotics. In: Distributed Simulation and Real-Time Applications, 2004. DS-RT 2004. Eighth IEEE International Symposium on. [S.l.: s.n.], 2001.

BELOUSOV, I. R.; TAN, J. C.; CLAPWORTHY, G. J. Teleoperation and java3d visualization of a robot manipulator over the world wide web. In: Third International Conference on Information Visualisation (IV'99). [S.l.: s.n.], 1999.

BUCHHOLZ, B.; ARMSTRONG, T. A kinematic model of the human hand to evaluate its prehensile capabilities. Journal Biomechanics, v. 25, p. 149–162, 1992.

BURDEA, P. C. G. C. Virtual Reality Technology. [S.l.]: Wiley - IEEE, 2003.

DIAS, C. Implementação de um Supervisor de Controle para Robôs Industriais. Dissertacão (Mestrado) — UNICAMP, 1993.

DRAGULESCU, D. et al. 3d active workspace of human hand anatomical model. BioMedical Engineering OnLine, 2007.

DRAGULESCU, D.; UNGUREANU, L.; STANCIU, A. Modeling the motion of the human middle finger. Proceedings. Workshop on Human Motion, 2005.

FIELD, D.; PALASTANGA, N.; SOAMES, R. Anatomia y movimiento humano. [S.l.: s.n.], 2001.

FREIRE, J. C. et al. A multimedia environment for supporting the teaching of robotics systems. Americas Conference on Information Systems-AMCIS, 2004.

GETTING Started with the Java 3D API. [S.l.]: Sun microsystems.

GRANT, D. et al. Haptics interfaces and devices. Sensor Review, v. 24, p. 19, 2004.

HAAGE, M.; NILSSON, K. On the scalability of visualization in manufacturing. In: Proceedings. ETFA '99. 1999 7th IEEE International Conference on Emerging Technologies and Factory Automation. [S.l.: s.n.], 1999.

HERIAS, F. A. C. et al. Laboratorio virtual remoto para robotica y evaluacion de su impacto en la docencia. Revista Iberoamericana de Automatica e Informatica, 2004.

HERIAS, F. A. C. et al. Flexible system for simulating and tele-operating robots through the internet. Journal of Robotic Systems, 2005.

HIRUKAWA, H. et al. A prototype of standard teleoperation systems on an enhanced vrml. In: Proceedings of the 1997 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS '97. [S.l.: s.n.], 1997.

JADHAV, C. A low-cost Framework for individualized interactive Telerehabilitation. Dissertação (Mestrado) — Graduate School of The State University of New York at Buffalo, 2004.

JUHASZ, T. Behavioral robot simulation for production systems. Production Systems and Information Engineering, v. 2, p. 121–129, 2004.

KARTOUN, U.; STERN, H.; EDAN, Y. Virtual reality telerobotic system. e-ENGDET 2004 4th international conference on e-engineering & digital enterprise technology, p. 10, 2004.

KOLLREIDER, A. et al. Robotic hand/finger rehabilitation for apoplexy patients. European Symposium Technical Aids for Rehabilitation TAR 2007, 2007.

MICHEL, O.; SAUCY, P.; MONDADA, F. Khepontheweb : An experimental demonstrator in telerobotics and virtual reality. Virtual Systems and MultiMedia. VSMM'97. Proceedings, 1997.

MILLER, A. et al. From robotic hands to human hands: a visualization and simulation engine for grasping research. Industrial Robot: An International Journal, 2005.

NAPIER, J. The prehensile movements of the human hand. THE JOURNAL OF BONE AND JOINT SURGERY, 1956.

SAFARIC, R. et al. Virtual environment for telerobotics. Integrated Computer-Aided Engineering, v. 8, p. 95–104, 2001.

SAFARIC, R. et al. Control of robot arm with virtual environment via the internet. Proceedings of the IEEE, v. 91, 2003.

SEMERE, W.; KITAGAWA, M.; OKAMURA, A. Teleoperation with sensor/actuator asymmetry: Task performance with partial force feedback. p. 7, 2004.

SONG, D.; KABER, D. B. Web-based interface design for teleoperation. In: Proceedings of the IEA 2000/HFES 2000 Congress. [S.l.: s.n.], 2000.

SPECK, A.; KLAEREN, H. Robosim: Java 3d robot visualization. Industrial Electronics Society. IECON'99 Proceedings, 1999.

SRIVSAN M. A., B.-C. Haptics in virtual enviroments: taxonomy, research status and challenges. Computer & Grpahics, v. 21, p. 12, 1997.

TSEPKOVSKIY, Y. et al. Development of a 3d and vrml virtual hand models for different mechanical gripper. Journal of the University of Chemical Technology and Metallurgy, 2008.

YANG, X. et al. Internet-based teleoperation of a robot manipulator for education. In: HAVE 2004. Proceedings. The 3rd IEEE International Workshop on Haptic, Audio and Visual Environments and Their Applications. [S.l.: s.n.], 2004.

ZHANG, P. et al. A teleoperating system for underwater manipulator to practice a time limit task. Intelligent Control. 2003 IEEE International Symposium on, 2003.

Apêndice A
Publicações

Oscar F Avilés S, João Mauricio Rosário, Álvaro Uribe, Paola Niño, Ricardo Gutierrez‡.,"Antrhopomorphic Grippers - Modelling, Analysis and Implementation", in Recent advances in Control Systems, Robotics and Automation- Third edition ISBN: 978-88-901928-6-9 , Aceito para publicação em 2009.

Oscar F Avilés S†,‡, João Mauricio Rosário‡, Álvaro Uribe‡, Paola Niño†, Ricardo Gutierrez‡.,"Antrhopomorphic Grippers - Modelling, Analysis and Implementation", in International Journal of Factory Automation, Robotics and Soft Computing ISSN 1828 - 6984. Aceito para publicação em 2009.

Alvaro Uribe Quevedo, João Mauricio Rosário, Oscar Avilés Sánchez, Paola Niño Suarez.,"Virtual Environment for Visualization and Movement Control of an Anthropomorphic Gripper",, in Recent advances in Control Systems,Robotics and Automation- Third edition ISBN: 978-88-901928-6-9. Aceito para publicação em 2009

Alvaro Uribe Quevedo, João Mauricio Rosário, Oscar Avilés Sánchez, Paola Niño Suarez.,"Virtual Environment for Visualization and Movement Control of an Anthropomorphic Gripper", in International Journal of Factory Automation, Robotics and Soft Computing ISSN 1828 - 6984. Aceito para publicação em 2009.

Oscar F Avilés S, João Mauricio Rosário, Álvaro Joffre and Fabian Lara. “Five Fingered Antrhopomorphic Hand Design: Muc- 1”, 7th Brazilian Conference on Dynamics, Controls and Applications, DINCON 2008, UNESP Campus Presidente Prudente.

Oscar F Avilés S, João Mauricio Rosário, Álvaro Uribe, Paola Niño, “Diseño De Un Sistema Mecatronico Antropomórfico De Cinco Dedos”, XIII Congreso Latinoamericano de Control Automático | VI Congreso Venezolano de Automatización y Control, Venezuela 2008.

Apêndice B

Características técnicas do aplicativo desenvolvido

1. IDE: NetBeans IDE 6.1
2. API: Java 3D 1.3
3. Pacote para suporte VRML: j3d-vrml97.jar
4. Linguagem de programação: Java
5. Pacotes:

```
com.sun.j3d.utils.behaviors.keyboard.KeyNavigatorBehavior;
com.sun.j3d.utils.behaviors.mouse.MouseBehavior;
com.sun.j3d.utils.behaviors.mouse.MouseRotate;
com.sun.j3d.utils.behaviors.mouse.MouseTranslate;
com.sun.j3d.utils.behaviors.mouse.MouseZoom;
java.applet.*;
java.awt.*;
com.sun.j3d.utils.applet.MainFrame;
javax.media.j3d.*;
java.awt.event.*;
com.sun.j3d.utils.universe.*;
javax.vecmath.*;
org.jdesktop.j3d.loaders.vrml97.VrmlLoader;
java.util.Vector;
com.sun.j3d.loaders.IncorrectFormatException;
com.sun.j3d.loaders.ParsingErrorException;
java.io.*;
javax.swing.*;
```

Apêndice C

Programa implementado no Robô ABB

```
%%%
 VERSION:1
 LANGUAGE:ENGLISH
%%%

MODULE AJUQ
 CONST jointtarget pos0:=[[-30,0.04,21.36,90.63,-88.07,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
 CONST robtarget p20:=[[903.96,-598.65,848.5],[0.634192,0.713654,-0.044255,-0.294177],[-1,1,-
1,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
 CONST robtarget p10:=[[919.97,-607.9,1203.49],[0.682234,0.689732,-0.172569,-0.170432],[-
1,1,0,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
 CONST robtarget p2:=[[709.28,292.65,584.2],[0.923869,-1.4E-05,0.382709,-1.5E-05],[0,0,-
1,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
 PERS tooldata mao:=[TRUE,[[227.918,9.23821,-10.9193],[1,0,0,0]],[5,[85,0,65],[1,0,0,0],0.01,0.01,0.01]];
 CONST robtarget ini:=[[955.01,554.78,1194.96],[0.706952,0.000148,0.707262,-0.000163],[0,-
1,0,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
 CONST robtarget p1:=[[1013.81,12.87,819.88],[0.806993,-0.000294,0.590561,0.000937],[0,0,-
1,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];

 !
 !###########################################################################################
 PROC inicial()
   MoveAbsJ [[120,0,0,0,-45,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]]\NoEOffs,v500,z50,mao;
 ENDPROC

 !
 !###########################################################################################
 !
 PROC Pos_1()
   MoveAbsJ [[0,0,0,0,-45,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]]\NoEOffs,v100,z50,mao;
 ENDPROC

 !
 !###########################################################################################
 !
 PROC Peg_obj1()
   MoveAbsJ [[0,0,0,0,-45,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]]\NoEOffs,v300,z50,mao;
   !   MoveJ p1,v100,z50,mao;
!   MoveJ p1,v100,z50,mao;
   MoveL Offs(p2,216,-100,400),v200,z0,mao;
   MoveL Offs(p2,216,-100,300),v200,z0,mao;
   MoveL Offs(p2,216,-100,60),v100,z0,mao;
   WaitTime 1;
   Set DO10_1;
```

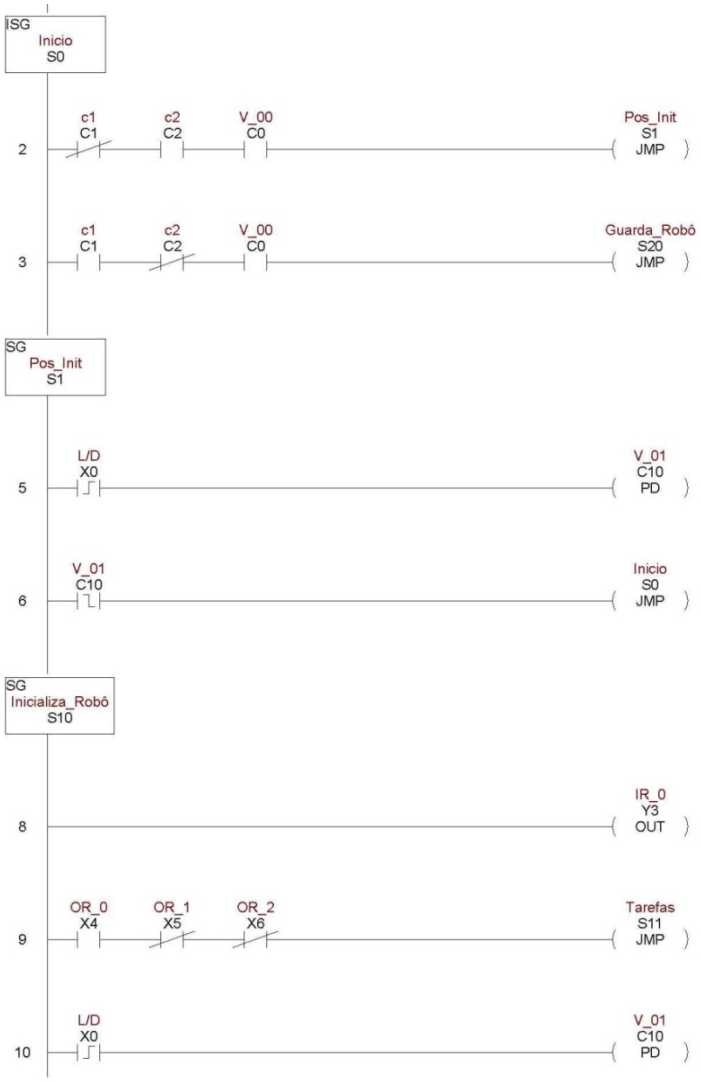
```
   Set DO10_2;
!   Set DO10_3;
   !
   WaitUntil DI10_1=1;
   !
   Reset DO10_1;
   Reset DO10_2;
!   Reset DO10_3;
   !
   MoveL Offs(p2,216,-100,300),v200,z0,mao;
!   MoveAbsJ [[0,0,0,0,-45,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]]\NoEOffs,v300,z50,mao;
   !
   !
 ENDPROC

 !
 !
 !#############################################################################################
 !
 PROC LEVA_OBJ1()
!   MoveAbsJ [[0,0,0,0,-45,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]]\NoEOffs,v300,z50,mao;
   MoveL Offs(p2,216,-400,300),v100,z0,mao;
   MoveL Offs(p2,216,-400,60),v100,z0,mao;
   WaitTime 1;
!   Set DO10_1;
!   Set DO10_2;
   Set DO10_3;
   !
   WaitUntil DI10_1=1;
   !
!   Reset DO10_1;
!   Reset DO10_2;
   Reset DO10_3;
   !
   MoveL Offs(p2,216,-400,300),v200,z0,mao;
   MoveAbsJ [[0,0,0,0,-45,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]]\NoEOffs,v300,z50,mao;
   !
!
 ENDPROC

 !#############################################################################################
 PROC APERTO_MAO()
   !
   MoveAbsJ [[-40,0,0,0,-45,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]]\NoEOffs,v300,z50,mao;
   MoveL p10,v200,z50,mao;
   MoveL p20,v200,z50,mao;
   WaitTime 1;
   !
   Set DO10_1;
!   Set DO10_2;
!   Set DO10_3;
   !
   WaitUntil DI10_1=1;
   !
   Reset DO10_1;
!   Reset DO10_2;
!   Reset DO10_3;
   !
   !
   MoveL Offs(p20,0,0,20),v200,z10,mao;
   MoveL Offs(p20,0,0,-20),v200,z10,mao;
   MoveL Offs(p20,0,0,20),v200,z10,mao;
   MoveL Offs(p20,0,0,-20),v200,z10,mao;
   MoveL Offs(p20,0,0,20),v200,z10,mao;
   MoveL Offs(p20,0,0,-20),v200,z10,mao;
   MoveL Offs(p20,0,0,20),v200,z10,mao;
   MoveL Offs(p20,0,0,-20),v200,z10,mao;
   MoveL Offs(p20,0,0,20),v200,z10,mao;
   MoveL Offs(p20,0,0,-20),v200,z10,mao;
   MoveL Offs(p20,0,0,20),v200,z10,mao;
```

```
   MoveL Offs(p20,0,0,0),v200,z10,mao;
   WaitTime 1;
!   Set DO10_1;
!   Set DO10_2;
   Set DO10_3;
   WaitTime 2;
!   Reset DO10_1;
!   Reset DO10_2;
   Reset DO10_3;
!
 ENDPROC

 !
 !*********************************************************************************************
 !
 PROC TRAZ_OBJ1()
   MoveAbsJ [[30,0,0,0,-45,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]]\NoEOffs,v300,z50,mao;
 ENDPROC

 !
 !*********************************************************************************************
 !
 ! INICIO DO PROGRAMA PRINCIPAL
 !_____________________________________________________________________________________________
 !
 !*********************************************************************************************
 !_____________________________________________________________________________________________
 PROC main()
!   MoveL p2,v100,z50,mao;
!   MoveL p10,v200,z50,mao;
!   MoveL p20,v200,z50,mao;
   IF DI10_3=1 AND DI10_4=1 AND DI10_5=0 THEN
     !
     inicial;
     !
   ENDIF
   !-----------------------------------------------------------------------------------------
   IF DI10_3=1 AND DI10_4=0 AND DI10_5=0 THEN
     !
     Peg_obj1;
     !
   ENDIF
   !-----------------------------------------------------------------------------------------
   IF DI10_3=0 AND DI10_4=1 AND DI10_5=0 THEN
     !
     LEVA_OBJ1;
     !
   ENDIF
   !-----------------------------------------------------------------------------------------
   IF DI10_3=1 AND DI10_4=1 AND DI10_5=1 THEN
     !
     APERTO_MAO;
     !
   ENDIF
 ENDPROC
ENDMODULE
```

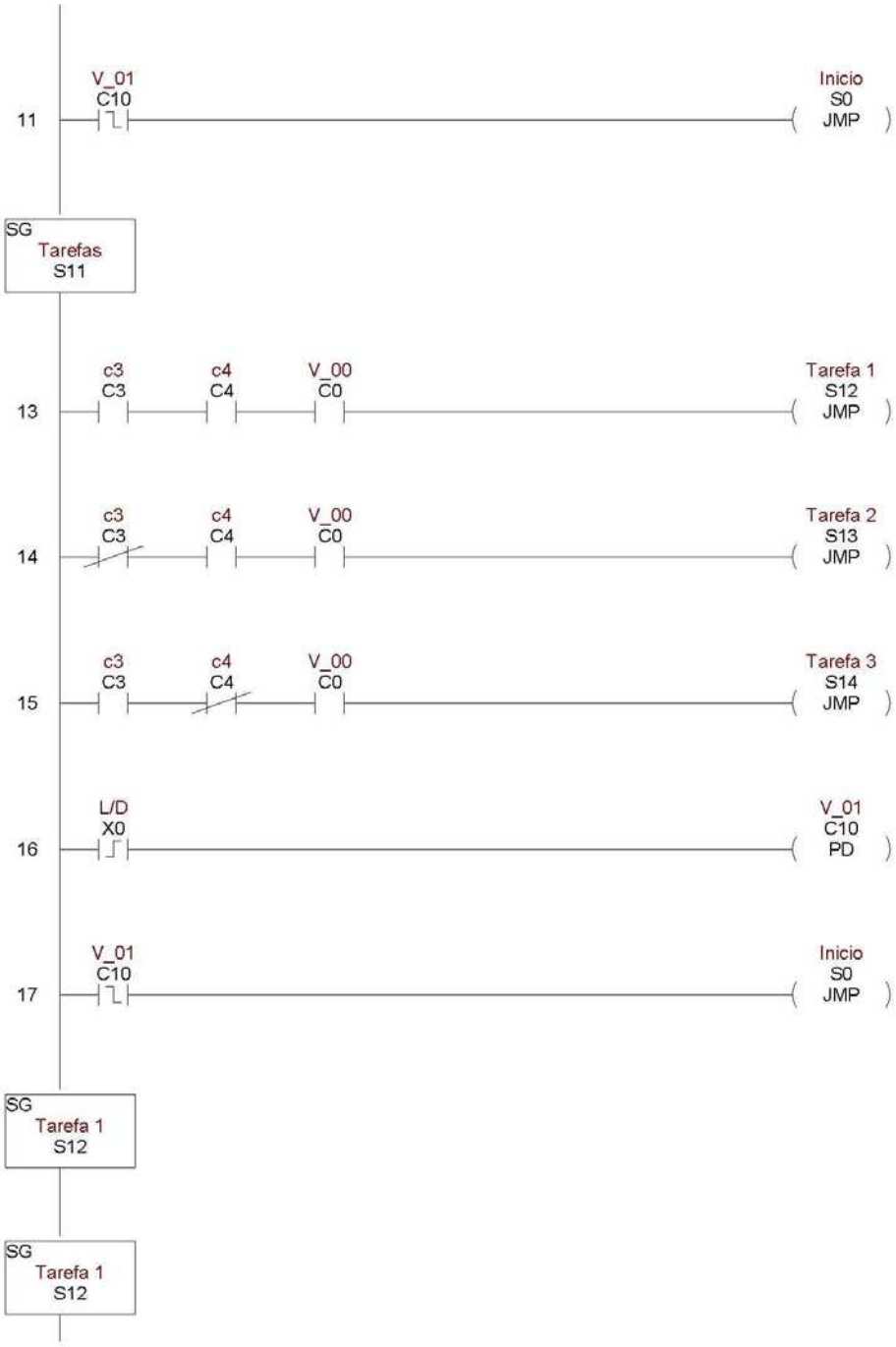
Apêndice D

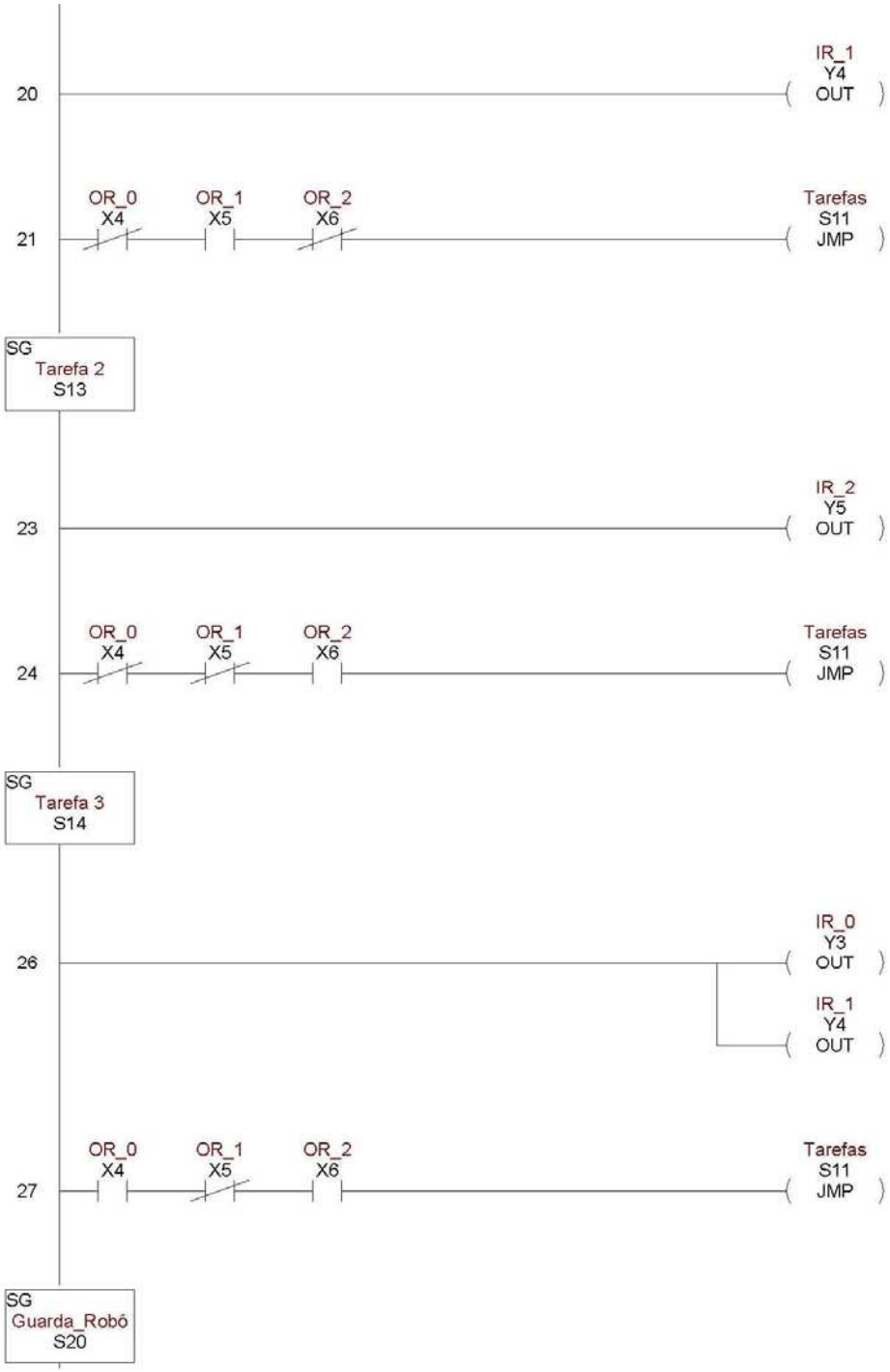
Programa Implementado na CLP KOYO

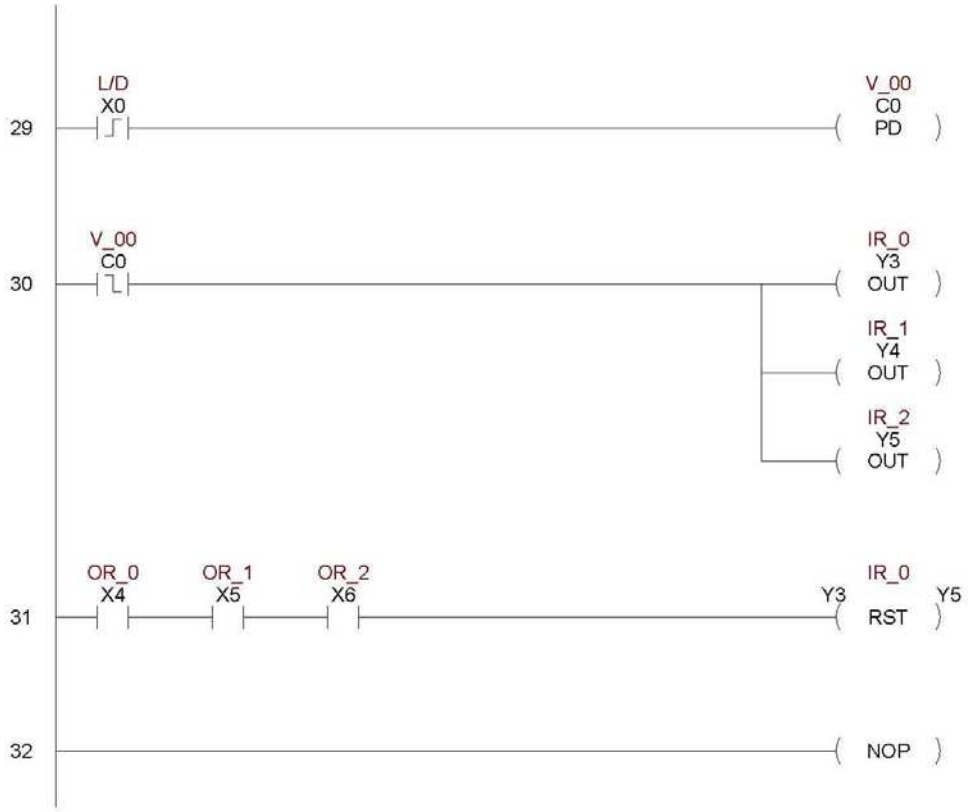
Representação por ladder



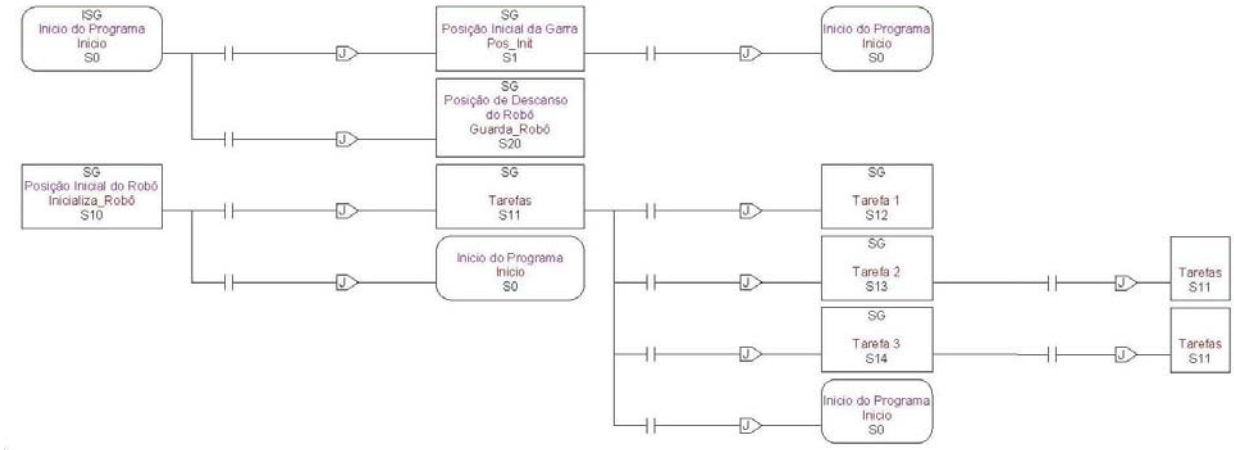
.


11
V_01
C10
Inicio
S0
JMP
SG
Tarefas
S11
13
c3
C3
c4
C4
V_00
C0
Tarefa 1
S12
JMP
14
c3
C3
c4
C4
V_00
C0
Tarefa 2
S13
JMP
15
c3
C3
c4
C4
V_00
C0
Tarefa 3
S14
JMP
16
L/D
X0
V_01
C10
PD
17
V_01
C10
Inicio
S0
JMP
SG
Tarefa 1
S12
SG
Tarefa 1
S12


20
IR_1
Y4
OUT
21
OR_0
X4
OR_1
X5
OR_2
X6
Tarefas
S11
JMP
SG
Tarefa 2
S13
23
IR_2
Y5
OUT
24
OR_0
X4
OR_1
X5
OR_2
X6
Tarefas
S11
JMP
SG
Tarefa 3
S14
26
IR_0
Y3
OUT
IR_1
Y4
OUT
27
OR_0
X4
OR_1
X5
OR_2
X6
Tarefas
S11
JMP
SG
Guarda_Robô
S20



Representação por estágios



Apêndice E

Programa implementado em LABVIEW