UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ
DEPARTAMENTO ACADÊMICO DE ELETROTÉCNICA
CURSO DE ENGENHARIA INDUSTRIAL ELÉTRICA/AUTOMAÇÃO

ANDRÉ FILIPE ROOS
ANDRÉ HENNING SANTOS
RODRIGO VALÉRIO ESPINOZA

PROJETO E IMPLEMENTAÇÃO DE UM SISTEMA DE GERAÇÃO DE TRAJETÓRIAS PARA O ROBÔ HANDLER DA UTFPR

TRABALHO DE CONCLUSÃO DE CURSO

CURITIBA

2011

ANDRÉ FILIPE ROOS
ANDRÉ HENNING SANTOS
RODRIGO VALÉRIO ESPINOZA

PROJETO E IMPLEMENTAÇÃO DE UM SISTEMA DE GERAÇÃO DE TRAJETÓRIAS PARA O ROBÔ HANDLER DA UTFPR

Trabalho de Conclusão de Curso apresentado ao Departamento Acadêmico de Eletrotécnica como requisito parcial para obtenção do grau de Engenheiro no Curso de Engenharia Industrial Elétrica/Automação da Universidade Tecnológica Federal do Paraná.

Orientador: Prof. Dr. Winderson Eugenio dos Santos

CURITIBA

2011

Dedicamos este trabalho aos nossos pais, por sempre prezarem por uma educação de qualidade, tanto na escola quanto em casa, e pela compreensão demonstrada durante as incontáveis reuniões noturno-matutinas dedicadas à engenharia.

AGRADECIMENTOS

Nossos sinceros agradecimentos aos familiares, namoradas e amigos, que foram sempre muito solícitos em seu apoio incondicional durante esta extensa fase de nossas vidas. Gostaríamos de agradecer, também, aos colaboradores deste trabalho, em especial o orientador Prof. Dr. Winderson Eugenio dos Santos, por nos nortear durante o desenvolvimento do trabalho e ao aluno Cristian Valle por dedicar seu tempo ao preparo do computador RT. Agradecemos à Prof. Rosângela Winter, pelas despedidas regulares, precisamente às 23 horas, incentivando a continuidade das tarefas e ao Prof. Dr. Antonio Carlos Pinho, pelos comentários sempre bem-humorados na sala vizinha ao nosso laboratório. Agradecemos, por fim, ao caro Djalma por prover uma alimentação imediata e de qualidade no intervalo das reuniões.

“O truque da filosofia é começar por algo tão simples que ninguém ache digno de nota e terminar por algo tão complexo que ninguém entenda” (Bertrand Russell)

“Para mim, é muito melhor compreender o universo como ele realmente é do que persistir no engano, por mais satisfatório e tranquilizador que possa parecer” (Carl Sagan)

“Equipado com seus cinco sentidos, o ser-humano explora o Universo ao seu redor, e chama essa aventura de Ciência” (Edwin Powell Hubble)

RESUMO

ROOS, André F; SANTOS, André H; ESPINOZA, Rodrigo V. Projeto e Implementação de um Sistema de Geração de Trajetórias para o Robô Handler da UTFPR. 117 f. Trabalho de Conclusão de Curso – Curso de Engenharia Industrial Elétrica/Automação, Universidade Tecnológica Federal do Paraná. Curitiba, 2011.

Este trabalho consiste no projeto e implementação de um sistema de geração de trajetórias e um aplicativo de interação homem-máquina para um robô manipulador de cinco graus de liberdade de propriedade da Univesidade Tecnológica Federal do Paraná. Este robô, apelidado de Handler, está na UTFPR há muitos anos e por algum tempo permaneceu em desuso. Nos últimos anos, participou de um minucioso processo de restauração por um grupo de Tecnologia em Automação Industrial, que implementou o controle individual dos motores por saídas digitais rápidas de controladores lógicos programáveis (CLPs). Nesta nova etapa, procedeu-se com uma modelagem matemática que possibilitou o equacionamento da cinemática direta e cinemática inversa do robô. Substituíram-se os CLPs pela placa de entrada/saída digital NI PCI-6601 da National Instruments, instalada em um computador rodando o sistema operacional de tempo real LabVIEW® RT. Este sistema é adequado para processos de tempo crítico como acionamento de motores de passo pois garante a execução de tarefas de modo determinístico. Um computador mestre executa um aplicativo de controle desenvolvido em LabVIEW® com interface gráfica, por meio da qual o usuário pode observar e simular movimentos do robô com um modelo tridimensional, gravar sequências de pontos e comandá-lo remotamente via protocolo Ethernet, monitorando uma diversidade de dados relevantes. Agora o Handler conta com um algoritmo de cinemática e um *hardware* de alto desempenho, de forma que pode processar, armazenar e executar movimentos mais sofisticados. O aplicativo, desenvolvido em conformidade com padrões de projeto de engenharia de *software*, possibilita a reintegração do robô em sala de aula como uma nova ferramenta para o ensino da robótica na universidade.

Palavras-chave: Robótica, Cinemática Inversa, Geração de Trajetórias, LabVIEW RT

ABSTRACT

ROOS, André F; SANTOS, André H; ESPINOZA, Rodrigo V. Design and Implementation of a Trajectory Generation System for the Robot Handler of UTFPR. 117 f. Trabalho de Conclusão de Curso – Curso de Engenharia Industrial Elétrica/Automação, Universidade Tecnológica Federal do Paraná. Curitiba, 2011.

This work consists in the design and implementation of a trajectory generation system and an human-machine interaction application for a five degrees of freedom robot manipulator owned by Universidade Tecnológica Federal do Paraná. This robot, dubbed Handler, has been in UTFPR for many years and for some time remained in disuse. In recent years, it participated in a meticulous retrofitting work done by students of the Industrial Automation Technology Degree Program, which implemented the control of individual stepper motors using programmable logic controllers (PLC) fast digital input/output ports. In this new stage, the project dealt with a mathematical model that lead to the robot's direct kinematics and inverse kinematics solution. The PLCs have been replaced by the timing and digital I/O board NI PCI-6601 manufactured by National Instruments, installed on a computer running the LabVIEW® RT real-time operating system. This system is suitable for time-critical processes like stepper motor driving as it ensures the execution of tasks deterministically. A master computer runs a control application developed in LabVIEW® with a graphical user interface, through which the user can observe and simulate the robot motion with the assistance of a three-dimensional model, as well as record sequences of points and control it remotely via the Ethernet protocol, monitoring several relevant data. Now Handler has a kinematic algorithm and a high-performance hardware, so it can process, store and run more sophisticated movements. The application, developed meeting software engineering design patterns, enables the reintegration of the robot in classes as a new tool for teaching robotics at the university.

Keywords: Robotics, Inverse Kinematics, Trajectory Generation, LabVIEW RT

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE SIGLAS

DAELT	Departamento Acadêmico de Eletrotécnica
UTFPR	Universidade Tecnológica Federal do Paraná
TCC	Trabalho de Conclusão de Curso
CLP	Controlador Lógico Programável
SOTR	Sistema Operacional de Tempo Real
E/S	Entrada/Saída
NI	National Instruments Corporation
SEMAP	Setor de Manutenção e Patrimônio
DAMEC	Departamento Acadêmico de Mecânica
VI	*Virtual Instrument* - Instrumento Virtual
RT	*Real Time* - Tempo Real
P	Junta Prismática
R	Junta Rotacional
GDL	Grau(s) de Liberdade
LED	*Light Emitter Diode* - Diodo Emissor de Luz
CPU	*Central Processing Unit* - Unidade Central de Processamento
E/S	Entrada/Saída
DH	Denavit-Hartenberg
GUI	*Graphic User Interface* - Interface Gráfica de Usuário)
PC	*Personal Computer* - Computador Pessoal
MEF	Máquina de Estados Finita
UML	*Unified Modeling Language* - Linguagem de Modelagem Unificada
RPY	*Roll Pitch Yaw* - Rolagem Arfagem Guinada
AC	*Alternate Current* - Corrente Alternada

RMS	*Root Mean Square* - Valor Eficaz
DC	*Direct Current* - Corrente Contínua
PWM	*Pulse Width Modulation* - Modulação por Largura de Pulso

LISTA DE SÍMBOLOS

®	Marca registrada
n	Número natural genérico
P_p	Potência primária da fonte de alimentação
P_c	Potência do sinal de controle
P_a	Potência do sinal de amplificação
P_{da}	Potência dissipada pelo amplificador
P_m	Potência mecânica do atuador
P_{ds}	Potência dissipada pelo atuador
P_u	Potência útil entregue pela transmissão
P_{dt}	Potência dissipada pela transmissão
f	Frequência
$\{i\}$	*Frame* genérico i
O_i	Origem do *frame* i
$\hat{\mathbf{x}}_i$	Vetor unitário $\mathbf{x}$ do *frame* $\{i\}$
${}^a\mathbf{p}_b$	Vetor-posição do *frame* $\{b\}$ descrito em relação ao *frame* $\{a\}$
${}^a p_x^b$	Componente escalar x do vetor-posição ${}^a\mathbf{p}_b$
${}^a\hat{\mathbf{x}}_b$	Vetor unitário $\mathbf{x}$ do *frame* $\{b\}$ descrito em relação ao *frame* $\{a\}$
${}^a R_b$	Matriz de rotação do *frame* $\{b\}$ em relação ao *frame* $\{a\}$
ψ	Ângulo de *Yaw*
θ	Ângulo de *Pitch*
ϕ	Ângulo de *Roll*
${}^a\mathbf{r}$	Vetor genérico $\mathbf{r}$ expresso em relação ao *frame* $\{a\}$
${}^a T_b$	Matriz de transformação homogênea 4x4 de $\{b\}$ para $\{a\}$
$\mathrm{Tr}(x, y, z)$	Matriz de translação homogênea

$\text{Rot}(x, \theta)$	Matriz de rotação homogênea para uma rotação de um ângulo θ em torno do eixo x
I_4	Matriz identidade 4×4
H	Matriz de transformação homogênea final genérica
$\hat{\mathbf{n}}$	Vetor unitário normal do efetuador
$\hat{\mathbf{o}}$	Vetor unitário de abertura do efetuador
$\hat{\mathbf{a}}$	Vetor unitário de aproximação do efetuador
$\mathbf{q}$	Vetor de variáveis de junta
a_i	Comprimento de elo
d_i	Deslocamento de junta
θ_i	Ângulo de junta
α_i	Ângulo de torção do elo
$\mathbb{R}^3$	Espaço vetorial tridimensional
$\mathbf{q}_{min}$	Vetor dos limites mínimos de variáveis de junta
$\mathbf{q}_{max}$	Vetor dos limites máximos de variáveis de junta
Q	Espaço de trabalho no espaço de juntas
W_a	Espaço de trabalho alcançável

SUMÁRIO

1 INTRODUÇÃO

1.1 TEMA

A busca por um sistema produtivo mais eficiente, flexível e de baixo custo passou, ao longo dos anos, a ditar o progresso industrial. A modernização fabril é consequência direta da adaptação às exigências do mercado (ROSÁRIO, 2009). Dentro deste palco surge o robô, abandonando o papel de protagonista em obras de ficção científica e assumindo fundamental importância para que uma grande parcela de indústrias seja tecnologicamente atualizada e competitiva.

As aplicações de robôs compreendem um amplo espectro do conhecimento humano, envolvendo a engenharia biomédica, militar, industrial, aeroespacial, etc. Dentre as várias preocupações atuais da área, Walter (1996) cita, por exemplo, a busca por robôs portadores de algoritmos de aprendizado rápido que aperfeiçoam sua coordenação motora a ponto de atingir a desempenho de sistemas biológicos. As projeções indicam que, em um futuro próximo, não serão incomuns cirurgias inteiramente executadas por robôs e sua presença massiva no ambiente domiciliar.

Dentro da esfera produtiva, estes dispositivos eletromecânicos são em geral denominados manipuladores ou robôs industriais. A norma ISO 8373:1994 delimita o conceito de robô industrial como "uma máquina manipuladora, com vários graus de liberdade, controlada automaticamente, reprogramável, multifuncional, que pode ter base fixa ou móvel, para utilização em aplicações de automação industrial" (SANTOS, 2001 apud INTERNATIONAL ORGANIZATION FOR STANDARDIZATION, 1994, p.1, tradução dos autores).

Inserido neste contexto o cenário paranaense, que vivencia nas últimas décadas uma forte transição de economia majoritariamente agrícola para industrial, nota-se uma carência de profissionais devidamente capacitados a atuar em áreas de tecnologia como a de controle e automação.

> O Estado do Paraná possui o segundo maior nível de automação industrial entre as regiões investigadas até o momento pela Pesquisa de Atividade Econômica Regional (Paer) sobretudo em termos de proporção de plantas automatizadas (44%)(WALENIA, 2006, p.12).

Com o intuito de atender a esse mercado de trabalho em pleno crescimento, o Departamento Acadêmico de Eletrotécnica (DAELT) da Universidade Tecnológica Federal do Paraná (UTFPR) criou o curso de Engenharia Industrial Elétrica com ênfase em Automação.

Embora no projeto do curso se preveja a oferta de uma disciplina optativa visando ao contato do corpo discente com a área de robótica, não há descrita a disponibilidade de nenhum dispositivo didático para a realização de aulas práticas. Entretanto, dentro dos materiais de propriedade da UTFPR encontra-se o Handler, um robô manipulador de cinco graus de liberdade que pode vir a auxiliar na prática do ensino. A Figura 1 apresenta o robô.

Figura 1: O robô Handler

No final de 2009 este robô participou de um detalhado processo de restauração oriundo de um Trabalho de Conclusão de Curso (TCC) do curso de Tecnologia em Automação Industrial (BICUDO; TURESSO; HALUC, 2010), do qual recebeu um novo sistema de acionamento e controle. O acionamento é composto de seis motores de passo e seis *drivers*

de potência comandados por três controladores lógicos programáveis (CLPs), cujo *software* envia comandos para cada junta individualmente.

A proposta é implementar um algoritmo de cinemática inversa que viabilize um controle de movimento mais sofisticado arcando, para tanto, com todas as modificações necessárias no sistema. O novo *software* deve, também, prover uma interface gráfica amigável e de fácil utilização para o operador.

Os atuais CLPs serão trocados por um computador com um sistema operacional de tempo real (SOTR), contendo uma placa de entrada/saída (E/S) para comando dos *drivers* de potência. Um segundo computador conterá a interface gráfica e comandará o primeiro, comunicando-se por meio de uma rede Ethernet. O desenvolvimento do *software* utilizará a plataforma LabVIEW® [1], que oferece um ambiente de programação inteiramente gráfico e instrumentos virtuais destinados a controle e monitoração remotos.

Assim sendo, com a execução do projeto surgem múltiplas possibilidades para o uso do Handler, de maneira que o DAELT terá um robô operacionalmente completo à sua disposição.

1.2 PROBLEMA E PREMISSAS

O manipulador Handler, não obstante tenha passado por um cauteloso processo de restauração, foi implementado com CLPs, que se mostram inapropriados à categoria de *hardware* dedicado ao controle de movimento em robôs.

A programação dos CLPs é tradicionalmente realizada em mnemônicos booleanos por ser a forma mais condizente com o modo com o qual os processadores operam (SILVEIRA; SANTOS, 2009). Esta técnica, porém, não é otimizada para a resolução de problemas de robótica, os quais necessitam de constantes manipulações matriciais, facilitadas por meio de paradigmas de programação de mais alto nível.

Na situação atual os CLPs estão subutilizados, haja vista que a maior parte das saídas normais se encontra ociosa e, em decorrência das particularidades no acionamento de motores de passo, apenas as saídas rápidas conectam-se aos *drivers* de potência.

Adicionalmente, a motivação para o enriquecimento do *software* com um algoritmo de cinemática inversa se esvaece com a não-modularidade do código e a falta de um levantamento preciso das dimensões mecânicas.

[1] LabVIEW é marca registrada da National Instruments Corporation (NI).

> O problema da cinemática inversa não é tão simples quanto o da direta. Sendo as equações cinemáticas não-lineares, sua solução não é sempre fácil ou mesmo possível. Também, levantam-se as questões acerca da existência de uma solução ou de múltiplas soluções (CRAIG, 1989, tradução dos autores).

Existe, pois, uma grande dificuldade atual na execução de trajetórias complexas, tanto no que tange ao posicionamento exato dos pontos desejados quanto na limitação pré-estabelecida da quantidade dos mesmos.

Dada a abordagem matemática exigente da área de robótica e a intrínseca dificuldade humana em simular imaginativamente o movimento tridimensional de mecanismos complexos, mostra-se valiosa a disponibilidade de dispositivos mecatrônicos facilmente reprogramáveis.

Por conseguinte, o robô necessita de modificações para que contemple integralmente as exigências práticas da robótica, acrescentando valor ao já mobilizado projeto e trazendo maior retorno à Instituição.

1.3 OBJETIVOS

1.3.1 OBJETIVO GERAL

Ampliar as potencialidades do Handler com o projeto e a implementação de um novo sistema de controle de movimento, por meio de um algoritmo de cinemática inversa, capaz de executar trajetórias programáveis.

1.3.2 OBJETIVOS ESPECÍFICOS

- Avaliar as características mecânicas, elétricas e computacionais do robô em seu estado atual;
- Realizar a troca dos CLPs pela placa NI PCI-6601[2], com a inclusão de um computador operando com um SOTR;
- Implementar um *software* de geração de trajetórias baseado em cinemática inversa;
- Elaborar uma interface gráfica, utilizando-se de instrumentação virtual, que possibilitará o controle e monitoração do robô;

[2]Trata-se de uma placa de E/S digital da NI.

- Analisar as vantagens decorrentes da substituição do sistema de controle e acionamento.

1.4 JUSTIFICATIVA

Os laboratórios de um curso de engenharia industrial auxiliam a capacitação do aluno, apresentando-o às possibilidades de aplicações dos conhecimentos teóricos aprendidos em sala de aula.

> [...] as aulas práticas no ambiente de laboratório podem despertar curiosidade e, consequentemente, o interesse do aluno [...] O uso deste ambiente também é positivo quando as experiências em laboratório estão situadas em um contexto histórico-tecnológico, relacionadas com o aprendizado do conteúdo de forma que o conhecimento empírico seja testado e argumentado, para enfim acontecer a construção de idéias (LEITE; SILVA; VAZ, 2005, p.3).

A atualização do Handler é imprescindível para lhe delegar tarefas mais úteis como, por exemplo, a de ferramenta robótica didática. Uma placa comandada por computador, comparativamente aos CLPs, acarretará melhorias visíveis na facilidade de operação, manutenção e compreensão do funcionamento do robô. Com a troca, os CLPs remanescentes se tornarão disponíveis para posterior uso em aulas ou projetos de automação que façam uso integral de suas potencialidades.

A substituição do controle promoverá um contato mais amigável entre o programador e o *software* do dispositivo, já que a linguagem do LabVIEW® suporta um paradigma de alto nível, visual e intuitivo. Abstrações podem atingir níveis de complexidade maiores, de forma a oferecer implementações consideravelmente mais rápidas, eficientes e compreensíveis. Pretende-se, com a elaboração do novo controle baseado em cinemática inversa, permitir uma maior facilidade de programação e de eventuais modificações no código.

O LabVIEW® viabiliza ainda a utilização de instrumentação virtual personalizada, que potencialmente "aplica operações matemáticas em tempo real para processamento, análises e controle envolvendo sinais de E/S que sejam *online* e/ou *offline*" (NATIONAL INSTRUMENTS, 2009).

Por fim, é de se considerar o valor histórico do robô e as carências a serem superadas para que recupere a utilidade da época em que foi construído e enviado à instituição. O empenho humano e financeiro empregado na sua restauração será recompensado à medida

que houver prosseguimento no projeto e suas futuras capacidades possam devolvê-lo às salas de aula.

1.5 PROCEDIMENTOS METODOLÓGICOS

Os procedimentos metodológicos e recursos do projeto estão detalhados dentro dos subitens seguintes. A ordem de apresentação desses procedimentos não corresponde necessariamente à sequência temporal de execução.

1.5.1 PESQUISA BIBLIOGRÁFICA

Pesquisa bibliográfica, na biblioteca da UTFPR e na internet, sobre tópicos de robótica: transformações, cinemática direta, cinemática inversa, geração de trajetória, sensores e atuadores. Adicionalmente, todos os integrantes do grupo cursaram as disciplinas de Robótica e de Instrumentação Virtual, que contemplam grande parte do conteúdo estudado.

1.5.2 AQUISIÇÃO DE DADOS

Consistiu na aquisição de dimensões pertinentes do robô (distância entre juntas, espaço de trabalho, comprimento dos elos, etc). O procedimento foi realizado no Setor de Manutenção e Patrimônio (SEMAP) da UTFPR, com ferramentas de metrologia mecânica provenientes do Departamento Acadêmico de Mecânica (DAMEC) e com o acompanhamento de um aluno graduando de Engenharia Mecânica.

1.5.3 ESTUDO DAS CAPACIDADES DO HANDLER

Averiguação geral das características físicas prévias do robô (conexões elétricas, estudo do sistema de atuação, controle, etc), seguindo indicações do orientador, na mesma sala onde se encontra o dispositivo atualmente (Q001).

1.5.4 PROJETO DO *HARDWARE*

Elaboração de um projeto do *hardware*, consultando-se manuais de operação e *datasheets* (*drivers* de potência, placa de E/S da NI, cabeamento). Atentou-se para o *layout* interno e para a elaboração de diagramas elétricos no *software* de captura

esquemática Proteus® [3].

A etapa foi encaminhada tanto na sala Q001 quanto fora da universidade.

1.5.5 SUBSTITUIÇÃO DE *HARDWARE*

Execução de modificações no *hardware* previstas pela etapa anterior, na mesma sala Q001, utilizando-se de ferramentas do DAELT e próprias, como alicates, chaves, multímetros e osciloscópio.

1.5.6 REVISÃO DE TRABALHOS ANTERIORES

Leitura detalhada do projeto dedicado à restauração do robô para uma visão mais abrangente quanto às ações da equipe anterior e do estado do robô.

Obteve-se, assim, diagramas de conexão elétrica, modelos de dispositivos empregados (sensores, motores) e estrutura do *software*, útil para o estudo das capacidades do Handler.

1.5.7 FAMILIARIZAÇÃO COM O LABVIEW®

Estudo e prática com o *software* LabVIEW® visando a obter uma perspectiva global das potencialidades do programa, por meio do menu *help*, livros, apostilas e videoaulas. Tópicos abordados foram: ambiente, painéis frontais, diagrama de blocos, VIs e subVIs, loops e estruturas, dados, gráficos, fórmulas, *debugging* e interfaceamento.

Fez-se uso de uma grande quantidade de apostilas e videoaulas gratuitas na internet e de uma cópia do LabVIEW® proprietária da UTFPR com licença de estudante disponibilizada ao grupo. Além disso, como já comentado, a equipe cursou, no primeiro semestre de 2011, a disciplina de Instrumentação Virtual, a qual abrange boa parte dos tópicos citados.

1.5.8 DESENVOLVIMENTO DO SOFTWARE

Projeto e implementação do *software*, baseados em bibliotecas já existentes. No código final coexistem rotinas de cinemática direta e inversa e de interface gráfica. Os recursos são o computador onde a cópia do LabVIEW® foi instalada e o cabeamento que interliga a máquina ao *hardware* do robô.

[3] Proteus é marca registrada da Labcenter Electronics Ltd.

1.5.9 ENSAIOS DE VALIDAÇÃO

Os ensaios de validação visam a avaliar as novas capacidades do robô por meio de experimentos. É a etapa sine qua non para a elaboração da conclusão, uma vez que as comparações com o estado anterior ditam o êxito ou não-êxito do trabalho.

1.5.10 REDAÇÃO DA MONOGRAFIA

Etapa realizada concomitantemente com o restante, em que se transmite ao documento as informações de relevância como grafos, diagramas, formulações e códigos. Segue a estrutura prevista na Seção 1.6.

1.6 ESTRUTURA DO TRABALHO

Este trabalho está estruturado em seis capítulos, dispostos conforme segue.

Capítulo 1: introdução ao projeto com o objetivo de contextualizar o leitor, apresentar a proposta de trabalho e despertar interesse no tema.

Capítulo 2: revisão bibliográfica sobre aspectos relevantes da teoria de robótica e instrumentação virtual necessários para a compreensão do desenvolvimento.

Capítulo 3: modelagem matemática do robô, resultando nas equações de cinemática direta e inversa.

Capítulo 4: apresentação do estado físico prévio do robô, desenvolvimento do projeto de *hardware* e sua instalação.

Capítulo 5: apresentação do estado prévio do *software* do robô e do desenvolvimento do novo *software* de geração de trajetórias com interface de operação em LabVIEW® .

Capítulo 6: realização de ensaios para avaliação do desempenho e validação do sistema de controle implementado.

Capítulo 7: conclusões e sugestões para trabalhos futuros.

Referências: listagem de todos os documentos que contribuíram para a redação da monografia.

2 FUNDAMENTAÇÃO TEÓRICA

Neste capítulo, discute-se uma série de aspectos teóricos fundamentais para o desenvolvimento do trabalho.

Um robô é um sistema integrado de elementos estruturais, sensoriais, de atuação e de controle. A Seção Elementos do Robô comenta-os com maior detalhamento, introduzindo a classificação e modelagem mecânica de manipuladores como uma série de elos conectados por meio de juntas.

O movimento resultante da estrutura robótica é fruto da composição dos movimentos elementares de cada elo em relação ao anterior. Portanto, a fim de se manipular um objeto no espaço, é necessário descrever a posição e orientação do efetuador. A Seção Transformações e Descrições Espaciais desenvolve uma abordagem sistemática, baseada em álgebra linear, para descrever a postura de corpos no espaço tridimensional em relação a qualquer sistema de coordenadas.

A Seção Cinemática lida com o estudo da geometria do movimento de um braço robótico com relação a um sistema de coordenadas de referência fixo sem levar em consideração as forças e momentos que causam o movimento. Detalha-se os dois problemas fundamentais em cinemática de manipuladores, referidos como o problema da cinemática direta e da cinemática inversa. No primeiro problema se busca a postura do efetuador dada uma configuração particular de juntas, utilizando a sistemática de Denavit e Hartenberg. No segundo problema, busca-se a configuração das juntas para uma postura do efetuador desejada.

A Seção Conceitos Básicos de LabVIEW® apresenta esta ferramenta, a terminologia e filosofia de programação, incluindo o módulo *Real Time* (RT). Por fim, a seção Conceitos de Engenharia de *Software* aborda modelos de ciclo de vida para o desenvolvimento de *software* e arquiteturas básicas em LabVIEW® que resolvem problemas recorrentes na programação de aplicações mais complexas.

2.1 ELEMENTOS DO ROBÔ

2.1.1 ESTRUTURA MECÂNICA

A estrutura mecânica de um robô manipulador é usualmente modelada como um mecanismo constituído de uma sequência de corpos rígidos (elos) conectados por articulações (juntas). Tal configuração recebe o nome de cadeia cinemática e é classificada em aberta ou fechada de acordo com a topologia de conexão entre elos. No caso de manipuladores de cadeia cinemática aberta, cada elo se conecta a apenas dois outros elos, de maneira que um extremo da cadeia é uma base fixa ou móvel e outro extremo um efetuador.

Os movimentos relativos entre elos da cadeia são providos por uma série de juntas. Não obstante haja seis juntas robóticas possíveis (FU; GONZALEZ, 1987), duas são predominantemente empregadas em robôs industriais:

- Juntas prismáticas (P). Criam movimento translacional relativo entre os elos;
- Juntas Rotacionais (R). Criam movimento rotacional relativo entre os elos e são "usualmente preferidas a juntas prismáticas pela compactabilidade e confiabilidade" (SICILIANO et al., 2009, p.4, tradução dos autores).

A Figura 2 apresenta um exemplo de cadeia cinemática aberta e a simbologia para cada elemento.

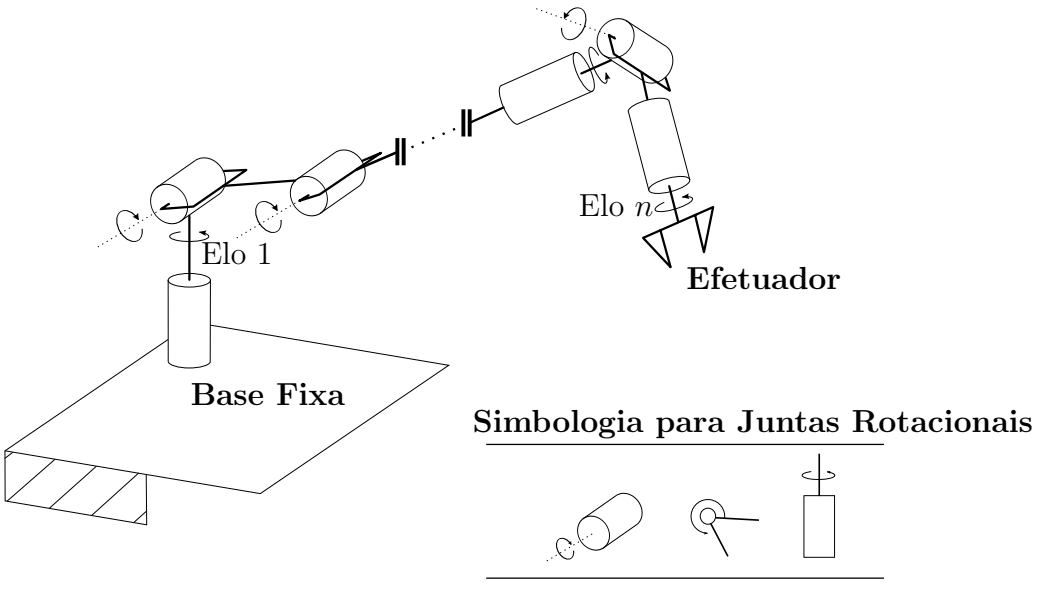

Figura 2: Exemplo de cadeia cinemática aberta

A maior parte dos robôs inseridos em plantas são montados em uma base presa ao chão (GROOVER et al., 1986). Em virtude da semelhança com o modelo anatômico humano, usualmente a estrutura de um manipulador é dividida em corpo, braço, punho e mão. O corpo é fixado na base e sustenta o braço. No final do braço fixa-se o punho, que orienta a mão.

A mão, tecnicamente conhecida por efetuador, é um dispositivo adicional montado na extremidade distal do braço que permite a um robô de propósitos gerais emprego em aplicações específicas (GROOVER et al., 1986). Efetuadores dividem-se em duas categorias: garras e ferramentas. Garras servem à função de captar objetos e segurá-los durante o ciclo de trabalho do robô, sendo limitadas às ações primitivas de abrir e fechar, úteis para transferência de materiais. Uma ferramenta é usada no caso de o robô executar operações específicas em uma peça de trabalho, como solda, pintura e furação.

2.1.2 GRAUS DE LIBERDADE

Segundo Santos (2001), "graus de liberdade é o número total de movimentos independentes que um dispositivo pode efetuar".

Baseando-se em Khatib (2007) pode-se elucidar esta definição de maneira mais precisa. Considere-se o manipulador da figura 2 com todas as juntas rotacionais removidas. Neste caso, a posição e orientação de cada elo no espaço tridimensional só seria completamente descrita mediante três parâmetros independentes para posição e três para orientação, ou seja, para os n elos, seriam necessários $6n$ parâmetros independentes. Reintroduzindo-se as juntas, cada uma estabelecerá cinco restrições para um elo, permitindo-o apenas um movimento independente. Desta forma n, igual a $6n - 5n$, é o número mínimo de parâmetros que descrevem completamente a posição e orientação dos corpos que compõem o robô. Estes parâmetros são conhecidos em mecânica como graus de liberdade (GDL). Assim, conclui-se que cada uma das juntas provê à estrutura um grau de liberdade, de modo que um robô de cadeia aberta terá tantos GDL quanto o número de juntas rotacionais/prismáticas.

Tipicamente, um manipulador deveria possuir pelo menos seis GDL - três para posição e três para orientação para alcançar todos os pontos no ambiente de trabalho com orientação arbitária (SPONG; HUTCHINSON; VIDYASAGAR, 2004). Em contrapartida, existem muitos robôs industriais com cinco ou menos GDL, úteis para tarefas específicas que não requerem seis GDL. Esta característica é favorável por

simplificar a análise matemática e não enquadrar o robô na categoria de redundante[1].

Os três primeiros GDL de um manipulador compõem a estrutura conhecida por braço e posicionam o punho no espaço tridimensional. A partir do punho, os GDL restantes são usados para estabelecer a orientação do efetuador (SCHILLING, 1990).

2.1.3 CLASSIFICAÇÃO

Manipuladores de cadeia aberta podem ser classificados focando-se apenas nas três primeiras juntas, a começar pela da base. Com juntas prismáticas ou rotacionais, há setenta e duas configurações diferentes possíveis, pois cada junta pode ser P ou R e eixos de junta podem ser paralelos, ortogonais ou perpendiculares (JAZAR, 2010).

A Tabela 1 sumariza as configurações mais usuais na indústria, de acordo com a combinação de juntas P e R para os três primeiros eixos do robô.

Tabela 1: Configurações usuais de robôs industriais

Robô	Eixo 1	Eixo 2	Eixo 3
Cartesiano	P	P	P
Cilíndrico	R	P	P
Esférico	R	R	P
SCARA	R	R	P
Antropomórfico	R	R	R

Fonte: Adaptado de Schilling (1990).

Os robôs antropomórficos (RRR) têm o primeiro eixo de junta ortogonal aos próximos dois. Por similaridade ao braço humano, a segunda junta é chamada de ombro e a terceira de cotovelo. A estrutura antropomórfica é a mais destra, pela presença exclusiva de juntas rotacionais. Todavia, "a correspondência entre os GDL e as variáveis de espaço cartesianas são perdidas, assim como a precisão de posicionamento do punho dentro do espaço de trabalho" (SICILIANO et al., 2009, p.8, tradução dos autores).

2.1.4 SISTEMA DE ATUAÇÃO

Para que o robô efetivamente se movimente, a potência da rede de distribuição deve fluir pelo chamado sistema de atuação até que seja convertida em movimento pelos atuadores. Em seguida, apresentam-se o diagrama de blocos deste sistema e sua explicação.

[1]Robôs cinematicamente redundantes têm mais de seis GDL e apresentam maior complexidade em

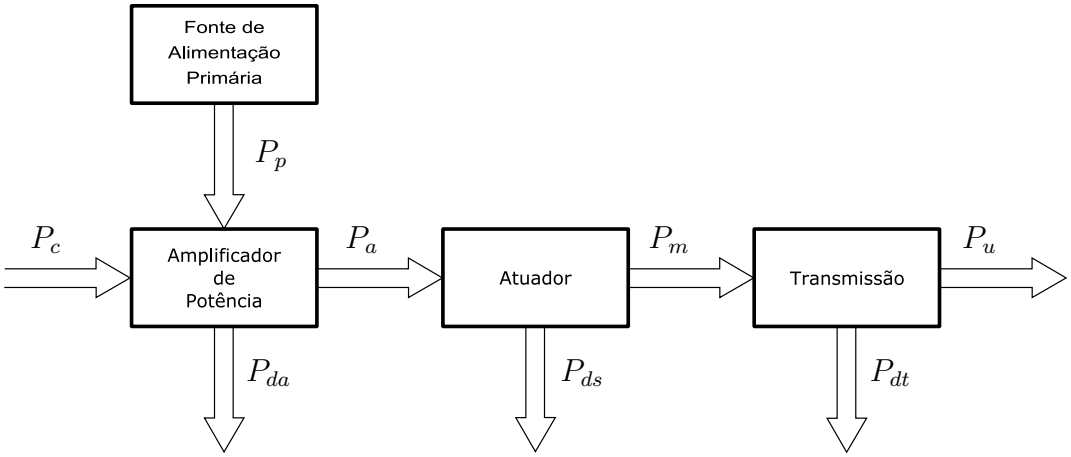

Figura 3: Diagramas de blocos do Sistema de Atuação
Fonte: Adaptado de Siciliano et al. (2009).

onde:

$$P_p = \text{potência primária}$$
$$P_c = \text{potência de controle}$$
$$P_a = \text{potência do amplificador}$$
$$P_m = \text{potência mecânica}$$
$$P_u = \text{potência útil}$$
$$P_{da} = \text{potência dissipada na amplificação}$$
$$P_{ds} = \text{potência dissipada pelo atuador}$$
$$P_{dt} = \text{potência dissipada na transmissão}$$

A rede de distribuição é a fonte primária de energia, fornecendo uma potência P_p ao estágio de amplificação de natureza idêntica à potência entregue ao atuador.

A potência primária P_p da rede e P_c do sinal de controle são entradas para o amplificador, que gera a saída P_a (potência elétrica amplificada de alimentação dos atuadores) e dissipa por perdas a potência P_{da}. O amplificador de potência tem a tarefa de condicionar o sinal de controle para níveis de tensão e corrente apropriados para os atuadores.

A próxima etapa do sistema de atuação são os atuadores, cuja entrada é o sinal amplificado de acionamento. Este bloco o converte na potência mecânica P_m e dissipa a

controle e programação.

potência P_{ds}.

A escolha do tipo de atuador determina o desempenho dinâmico do manipulador e, em muitos casos, sua gama de aplicações (GROOVER et al., 1986). Robôs industriais comerciais são majoritariamente atuados por sistemas hidráulicos, elétricos ou pneumáticos. Motores elétricos são interessantes se avaliadas as vantagens decorrentes da atuação elétrica, como custo, manutenção, precisão e eficiência energética; entretanto, como enfatiza Jazar (2010), não competem com a alta velocidade e capacidade de carga de atuadores hidráulicos.

Em se tratando de robôs atuados eletricamente, embora servomotores dominem o meio industrial,

> [...] motores de passo também não são raros. Estes atuadores são controlados por seqüências de excitação adequadas e seu princípio de funcionamento não requer medição da posição angular do eixo do motor. O comportamento dinâmico de motores de passo é muito influenciada pela carga, no entanto. Além disso, induzem vibração da estrutura mecânica do manipulador. Tais inconvenientes limitam o uso de motores de passo para o campo de micromanipuladores, para o qual uma implementação de baixo custo prevalece sobre a necessidade de alto desempenho dinâmico (SICILIANO et al., 2009, p.195, tradução dos autores).

O sistema de atuação tem como último estágio o bloco de transmissão, que recebe a potência mecânica P_m dos atuadores e a otimiza para as juntas do robô, devolvendo a potência P_u e dissipando por fricção P_{dt}. Embora existam casos adequados ao acionamento direto na junta (*direct-drive*), a exigência por altos torques e baixas velocidades torna conveniente um sistema de transmissão para otimizar a transferência de potência atuador-junta. O uso de transmissões pode produzir ganhos em desempenho estático e dinâmica ao se posicionar atuadores na base, reduzindo o peso do manipulador e o carregamento nas posturas que causam seu deslocamento.

2.1.5 SENSORES

Sensores são conversores de energia empregados na medição de grandezas físicas. Diferem de transdutores por converterem tipos diversos de energia *exclusivamente* em sinais elétricos (FRADEN, 2004). Em robótica, sensores são normalmente divididos em duas categorias:

- Sensores Proprioceptivos[2]: retornam informações sobre variáveis internas do sistema

[2]Também referenciados por "sensores internos" na literatura.

robótico como posição e velocidade;

- Sensores Exteroceptivos[3]: retornam informações que o robô interpreta para extrair características significativas do ambiente externo.

Esta classificação abriga uma série de subcategorias, como sensores táteis, de proximidade, de posição, de temperatura, de pressão e mesmo complexos sistemas de visão de máquina, para inspeção, reconhecimento e tarefas correlatas (GROOVER et al., 1986).

Em sistemas robóticos mais simples é comum que o único tipo de sensor empregado seja de proximidade, para limitação no percurso das juntas. Um tipo de dispositivo que serve a este propósito é a chave óptica, cujo acionamento é fruto da interposição de um objeto a um feixe luminoso de um transmissor em direção a um receptor, localizados fisicamente em lados opostos de uma região de interesse (BRAGA, 2011). O transmissor é normalmente um LED infravermelho e o receptor, por exemplo, um fototransistor. Quando uma junta atinge o limite de percurso, um pino em movimento solidário interrompe o feixe de luz e a saída do receptor muda de estado, indicando que o sistema de controle deve cessar o acionamento do atuador. A figura 4 esquematiza uma chave óptica.

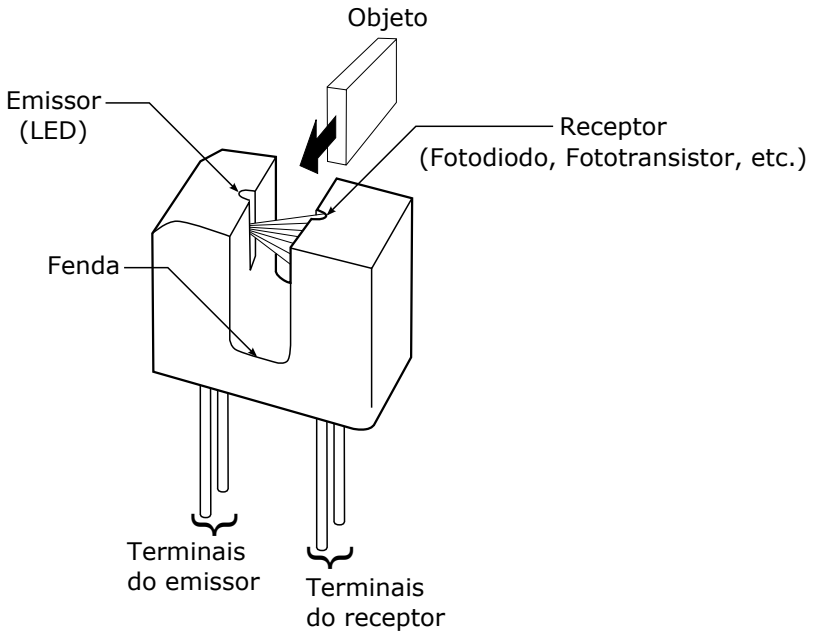

Figura 4: Estrutura típica de uma chave óptica
Fonte: Adaptado de Braga (2011).

[3]Também referenciados por "sensores externos" na literatura.

É importante não confundir a terminologia:

> Com efeito, um sensor de proximidade é uma versão limiar de um sensor de posição. Um sensor de posição é muitas vezes um dispositivo linear cujo sinal de saída representa uma distância até o objeto a partir de um certo ponto de referência. O sensor de proximidade, no entanto, é um dispositivo um pouco mais simples que gera o sinal de saída quando uma certa distância até o objeto torna-se relevante para uma indicação (FRADEN, 2004, p.253, tradução dos autores).

2.1.6 CONTROLE

De todos os blocos construtivos de um sistema robótico, o controlador é o mais complexo e tem o maior grau de variação entre fabricantes diferentes (REHG, 1985). É geralmente um computador dedicado com CPU, memória e dispositivos de E/S.

O tipo de controle utilizado para posicionar o efetuador permite a classificação de robôs em sistemas de malha fechada e sistemas de malha aberta. Sistemas de malha fechada empregam sensores para monitorar e retroalimentar continuamente variáveis como posição, velocidade e aceleração, traduzindo-as para comandos em tempos específicos sobre um ou mais atuadores. A Figura 5 é um exemplo de sistema de controle de movimento em malha fechada.

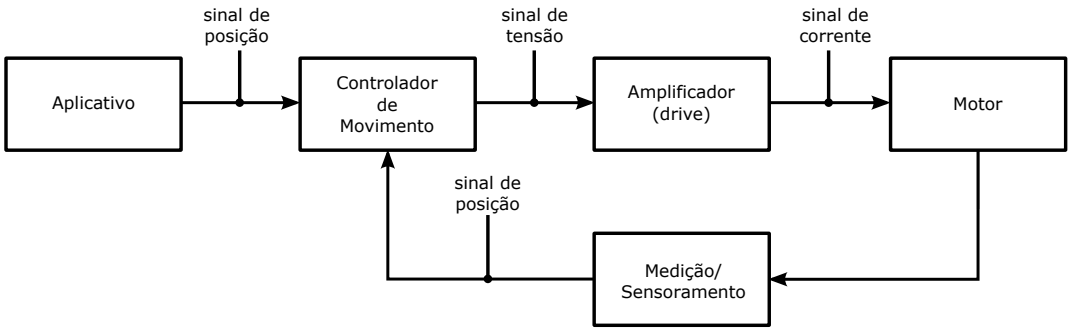

Figura 5: Controle de movimento em malha fechada
Fonte: Adaptado de National Instruments (2011).

O bloco de aplicação promove interação com o usuário, recebe dados e, a partir de algoritmos baseados na cinemática e/ou dinâmica do manipulador, determina perfis de movimento e transmite sinais compreensíveis ao controlador. As instruções geradas pelo aplicativo, eventualmente somadas a uma retroalimentação de posição, são processadas pelo controlador de movimento. Este, por sua vez, envia sinais de tensão para o *driver* de

potência que, suprindo as necessidades impostas, provê energia suficiente para acionar o motor, modificando os parâmetros mecânicos que definem a posição e orientação do robô.

Em sistemas de malha aberta, como a maioria dos atuados por motores de passo, não há retroalimentação. Neste caso, uma junta só cessará o movimento se atingir um limite mecânico ou se o controle cessar o comando depois de supor que se atingiu a posição desejada. A integridade do sistema mecânico e o tempo permitido para se alcançar uma posição garantem uma operação relativamente acurada (REHG, 1985). Tendo em vista que há uma relação direta entre os pulsos entregues a um motor de passo e os passos executados, é possível controlar posição, velocidade e aceleração angulares do eixo mesmo sem retroalimentação. De fato, um trem de pulsos com frequência f será diretamente proporcional à velocidade rotórica do motor.

Quanto à trajetória executada, pode-se distribuir os manipuladores em quatro categorias, em ordem crescente de sofisticação (GROOVER et al., 1986):

- Robôs de sequência limitada: juntas individuais podem apenas ser movidas para seus limites de percurso;
- Robôs com controle ponto-a-ponto: ciclos de movimento consistem em uma série de pontos desejados gravados em memória, entre os quais não há controle no caminho executado. O programador pode discretizar em vários pontos um caminho para obter maior controle sobre o traçado;
- Robôs com controle de caminho contínuo: especificam-se pontos de início e fim para um caminho e o controlador se encarrega de interpolar os pontos intermediários;
- Robôs inteligentes: alteração do ciclo de movimento programado por meio de decisões lógicas baseadas em informações sensoriais.

Outro ponto a ser levado em consideração é que, como em motores de passo a relação de ângulo por pulso executado é bastante direta, para que se controle satisfatoriamente sua velocidade e aceleração é necessário que os comandos sejam executados em tempos bem definidos.

2.2 TRANSFORMAÇÕES E DESCRIÇÕES ESPACIAIS

Na seção anterior, o manipulador foi modelado como uma sequência de corpos rígidos conectados. Em seguida, descrever-se-á matemáticamente a localização de cada

componente do robô no espaço tridimensional, assim como do ambiente de trabalho e de objetos relevantes dentro deste ambiente.

Em outras palavras, busca-se a postura de cada elemento do sistema robótico, que é completamente descrita pelos atributos posição e orientação.

2.2.1 POSTURA

Para se descrever a postura de um corpo no espaço "sempre se fixa um sistema de coordenadas, ou *frame*, rigidamente ao objeto" (CRAIG, 1989, p.5, tradução dos autores), de sorte que o *frame* se deslocará solidariamente a todos os pontos do corpo.

Matematicamente, um *frame* $\{i\}$ é uma entidade que consiste em uma origem O_i e uma tríade de vetores unitários mutuamente ortogonais $\hat{\mathbf{x}}_i$, $\hat{\mathbf{y}}_i$ e $\hat{\mathbf{z}}_i$, ou

$$\{i\} = (\hat{\mathbf{x}}_i, \hat{\mathbf{y}}_i, \hat{\mathbf{z}}_i, O_i) \qquad , \qquad \hat{\mathbf{x}}_i \text{ definindo eixo } x_i \tag{2.1}$$

A postura de um corpo será sempre expressa relativamente a outro corpo; desta forma, é invariavelmente equivalente à postura de um *frame* em relação a outro.

Em geral, estabelece-se também um *frame* universal imóvel e arbitrário ao qual todos os objetos podem ser referenciados. Cada *frame* pode então ser descrito em função do *frame* universal ou de qualquer outro, de acordo com a conveniência. A Figura 6 mostra a representação gráfica de dois *frames* quaisquer $\{a\}$ e $\{b\}$, que estão fixados em corpos que poderiam fazer parte de um sistema robótico.

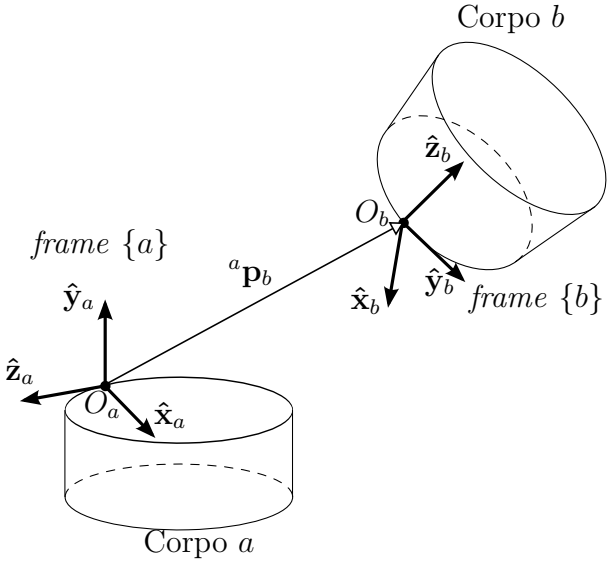

Figura 6: Fixação de *frames* em corpos rígidos

Em consonância com a figura ilustrada anteriormente, a posição da origem O_b de $\{b\}$ em relação a origem O_a do frame $\{a\}$ pode ser expressa pelo vetor-posição ${}^a\mathbf{p}_b$:

$$ {}^a\mathbf{p}_b = \begin{bmatrix} {}^a p_x^b \\ {}^a p_y^b \\ {}^a p_z^b \end{bmatrix} \tag{2.2} $$

As componentes escalares ${}^a p_x^b$, ${}^a p_y^b$ e ${}^a p_z^b$ são o vetor posição da origem O_b em relação a O_a de $\{a\}$.

À primeira vista, poder-se-ia orientar $\{b\}$ de infinitas maneiras mantendo a posição constante. Assim, a postura só estará integralmente definida com a inclusão do atributo orientação.

A orientação de $\{b\}$ em relação a $\{a\}$ pode ser descrita expressando-se os vetores unitários $(\hat{\mathbf{x}}_b, \hat{\mathbf{y}}_b, \hat{\mathbf{z}}_b)$ em termos de $(\hat{\mathbf{x}}_a, \hat{\mathbf{y}}_a, \hat{\mathbf{z}}_a)$, por meio dos vetores ${}^a\hat{\mathbf{x}}_b$, ${}^a\hat{\mathbf{y}}_b$ e ${}^a\hat{\mathbf{z}}_b$. É conveniente organizá-los como vetores-coluna da chamada matriz de rotação ${}^a R_b$, como segue:

$$ {}^a R_b = \begin{bmatrix} {}^a\hat{\mathbf{x}}_b & {}^a\hat{\mathbf{y}}_b & {}^a\hat{\mathbf{z}}_b \end{bmatrix} = \begin{bmatrix} \hat{\mathbf{x}}_b \cdot \hat{\mathbf{x}}_a & \hat{\mathbf{y}}_b \cdot \hat{\mathbf{x}}_a & \hat{\mathbf{z}}_b \cdot \hat{\mathbf{x}}_a \\ \hat{\mathbf{x}}_b \cdot \hat{\mathbf{y}}_a & \hat{\mathbf{y}}_b \cdot \hat{\mathbf{y}}_a & \hat{\mathbf{z}}_b \cdot \hat{\mathbf{y}}_a \\ \hat{\mathbf{x}}_b \cdot \hat{\mathbf{z}}_a & \hat{\mathbf{y}}_b \cdot \hat{\mathbf{z}}_a & \hat{\mathbf{z}}_b \cdot \hat{\mathbf{z}}_a \end{bmatrix} = \begin{bmatrix} n_x & o_x & a_x \\ n_y & o_y & a_y \\ n_z & o_z & a_z \end{bmatrix} \tag{2.3} $$

Esta matriz contêm nove elementos, mas apenas três parâmetros são necessários para descrever a orientação, como comenta Khatib e Siciliano (2008). Caso se deseje expressar $\{a\}$ em relação a $\{b\}$, basta calcular a matriz inversa de ${}^a R_b$, ou seja,

$$ {}^b R_a = {}^a R_b{}^{-1} \tag{2.4} $$

Como os vetores unitários de $\{a\}$ são mutualmente ortogonais e normais, assim como os vetores unitários de $\{b\}$ a matriz ${}^a R_b$ também é ortogonal e de determinante unitário positivo. Portanto

$$ {}^a R_b{}^{-1} = {}^a R_b{}^T \tag{2.5} $$

Afim de reduzir o número de parâmetros para descrever orientações, há uma série de representações alternativas em robótica. Dentre as mais comuns está a RPY (*Roll-Pitch-Yaw*), sumarizada em um vetor de três ângulos (ϕ, θ, ψ), chamados ângulos RPY. Estes ângulos representam uma sequência de três rotações em torno dos eixos fixos x,y z eixos fixos.

Sendo $\{a\}$ um *frame* fixo, quando se diz que a orientação de um *frame* $\{b\}$ rotacionado

a partir de $\{a\}$ é (ϕ, θ, ψ) significa que, para atingi-la, realizou-se a sequência de rotações

1. ψ em torno de $\hat{\mathbf{x}}_a$
2. θ em torno de $\hat{\mathbf{y}}_a$
3. ϕ em torno de $\hat{\mathbf{z}}_a$

como mostra a Figura 7.

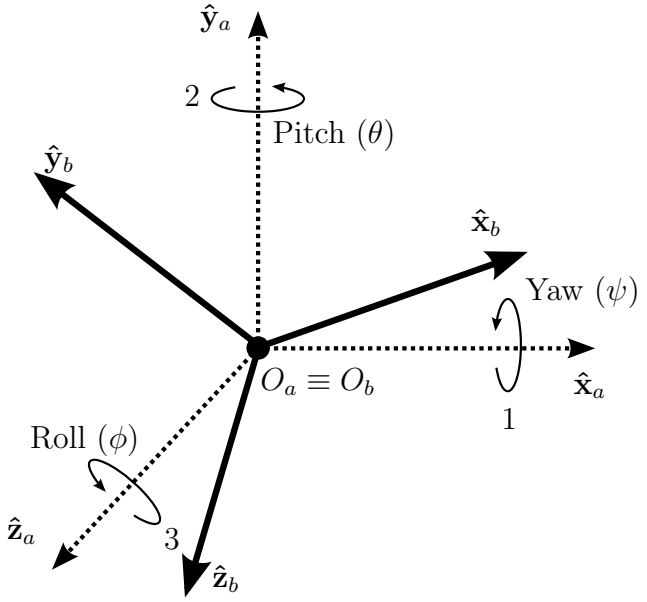

Figura 7: *frame* $\{b\}$ rotacionado na sequência ψ - θ - ϕ

2.2.2 MATRIZES DE TRASFORMAÇÃO HOMOGÊNEAS

Transformações homogêneas introduzem uma notação "compacta - embora computacionalmente ineficiente" (BARRIENTOS et al., 1997, p.14, tradução dos autores) - que combina o vetor-posição e a matriz de rotação em uma única entidade matemática. Conhecidas a posição e orientação de um *frame* $\{b\}$ em relação a um *frame* $\{a\}$, é possível transformar qualquer vetor ${}^b\mathbf{r}$, expresso relativamente a $\{b\}$, em ${}^a\mathbf{r}$, através da equação

$$ {}^a\mathbf{r} = {}^aR_b\,{}^b\mathbf{r} + {}^a\mathbf{p}_b \tag{2.6} $$

que pode ser reescrita como

$$ \begin{bmatrix} {}^a\mathbf{r} \\ 1 \end{bmatrix} = \begin{bmatrix} {}^aR_b & {}^a\mathbf{p}_b \\ \mathbf{0}^T & 1 \end{bmatrix} \begin{bmatrix} {}^b\mathbf{r} \\ 1 \end{bmatrix} \tag{2.7} $$

onde

$$ {}^{a}T_{b} = \begin{bmatrix} {}^{a}R_{b} & {}^{a}\mathbf{p}_{b} \\ \mathbf{0}^{T} & 1 \end{bmatrix} \tag{2.8} $$

${}^{a}T_{b}$ é a matriz de transformação homogênea 4x4 e $\begin{bmatrix} {}^{a}\mathbf{r} & 1 \end{bmatrix}^{T}$ e $\begin{bmatrix} {}^{b}\mathbf{r} & 1 \end{bmatrix}^{T}$ são as representações homogêneas dos vetores-posição ${}^{a}\mathbf{r}$ e ${}^{b}\mathbf{r}$, respectivamente.

As matriz de transformação homogênea admite três interpretações:

- É um descritor. ${}^{a}T_{b}$ descreve $\{b\}$ em relação a $\{a\}$ especificando as translações e rotações necessárias para atingi-lo.
- É um mapeador. Multiplicando-se um vetor descrito em relação a um *frame* por uma matriz de transformação, obtem-se o mesmo vetor descrito em relação a outro *frame*:

$$ \begin{bmatrix} {}^{a}\mathbf{r} \\ 1 \end{bmatrix} = {}^{a}T_{b} \begin{bmatrix} {}^{b}\mathbf{r} \\ 1 \end{bmatrix} \tag{2.9} $$

- É um operador. Translaciona e rotaciona um vetor ${}^{a}\mathbf{r}_{1}$ gerando um vetor ${}^{a}\mathbf{r}_{2}$ descrito em relação ao mesmo *frame*:

$$ \begin{bmatrix} {}^{a}\mathbf{r}_{2} \\ 1 \end{bmatrix} = T \begin{bmatrix} {}^{b}\mathbf{r}_{1} \\ 1 \end{bmatrix} \tag{2.10} $$

  onde T não descreve um *frame* em relação a outro, mas sim as translações e rotações sobre um vetor em um mesmo *frame*.

Como um mapeador, a matriz homogênea ${}^{a}T_{b}$ transforma vetores do *frame* $\{b\}$ para o *frame* $\{a\}$. Sua inversa, ${}^{a}T_{b}{}^{-1}$, transforma vetores do *frame* $\{a\}$ para o *frame* $\{b\}$, e não é simplesmente a transposta, pois não se trata de uma matriz ortogonal como é o caso das matrizes de rotação. De fato, ocorre que

$$ {}^{a}T_{b}{}^{-1} = {}^{b}T_{a} = \begin{bmatrix} {}^{a}R_{b} & -({}^{a}R_{b}{}^{T}){}^{a}\mathbf{p}_{b} \\ \mathbf{0}^{T} & 1 \end{bmatrix} \tag{2.11} $$

Como em Santos (2001), uma translação de (x, y, z) pode ser expressa pela matriz $\mathrm{Tr}(x, y, z)$:

$$ \mathrm{Tr}(x, y, z) = \begin{bmatrix} 1 & 0 & 0 & x \\ 0 & 1 & 0 & y \\ 0 & 0 & 1 & z \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.12} $$

Desenvolvendo-se a equação (2.3) para rotações puras de um ângulo θ em torno dos eixos $\hat{\mathbf{x}}_a$, $\hat{\mathbf{y}}_a$ e $\hat{\mathbf{z}}_a$ de um *frame* $\{a\}$, chega-se às matrizes homogêneas Rot() definidas a seguir:

$$\mathrm{Rot}(x,\theta) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\theta & -\operatorname{sen}\theta & 0 \\ 0 & \operatorname{sen}\theta & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.13}$$

$$\mathrm{Rot}(y,\theta) = \begin{bmatrix} \cos\theta & 0 & \operatorname{sen}\theta & 0 \\ 0 & 1 & 0 & 0 \\ -\operatorname{sen}\theta & 0 & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.14}$$

$$\mathrm{Rot}(z,\theta) = \begin{bmatrix} \cos\theta & -\operatorname{sen}\theta & 0 & 0 \\ \operatorname{sen}\theta & \cos\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.15}$$

onde $\mathrm{Rot}(x,\theta)$ é a matriz de rotação homogênea para uma rotação de um ângulo θ em torno do eixo x.

Pode-se compor transformações através da multiplicação matricial entre as matrizes de transformação homogêneas. Dados os *frames* $\{a\}$, $\{b\}$ e $\{c\}$, será sempre válido que

$$^aT_c = {}^aT_b\,{}^bT_c \tag{2.16}$$

mas se deve atentar para a ordem, já que a multiplicação matricial não é comutativa. De maneira geral, como preconiza Barrientos et al. (1997):

- Se o sistema fixo $\{a\}$ e o sistema transformado $\{b\}$ são coincidentes, a matriz de transformação homogênea será a matriz identidade I_4.
- Se $\{b\}$ é obtido mediante transformações ao longo dos eixos de um sistema fixo $\{a\}$, a matriz de transformação homogênea deverá pré-multiplicar as prévias.
- Se $\{b\}$ é obtido mediante transformações o longo dos eixos de um sistema móvel, a matriz de transformação homogênea deverá pós-multiplicar as prévias.

No caso de um manipulador de n elos, pode-se realizar a composição das matrizes homogêneas que descrevem cada *frame* a fim de se estabelecer uma matriz final $H = {}^0T_n$

que exprima a postura do efetuador ($\{n\}$) em relação à da base ($\{0\}$), na forma

$$ {}^0T_n = H = \begin{bmatrix} \hat{\mathbf{n}} & \hat{\mathbf{o}} & \hat{\mathbf{a}} & \mathbf{p} \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} n_x & o_x & a_x & p_x \\ n_y & o_y & a_y & p_y \\ n_z & o_z & a_z & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.17} $$

onde $\hat{\mathbf{n}}$, $\hat{\mathbf{o}}$ e $\hat{\mathbf{a}}$ são os vetores unitários do *frame* do efetuador e $\mathbf{p}$ é seu vetor-posição em relação à base. Fica implícito que são vetores de $\{n\}$ expressos em $\{0\}$.

A Figura 8 apresenta a convenção para a nomenclatura dos vetores unitários do *frame* do efetuador.

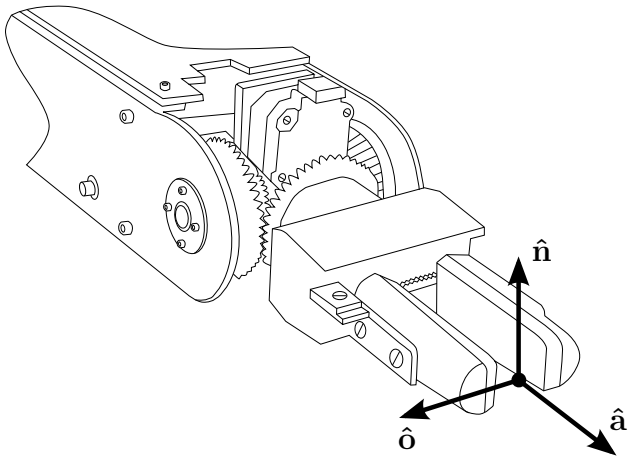

Figura 8: Convenção para o *frame* do efetuador

Utilizando-se ainda da composição de matrizes homogêneas, é possível construir a matriz de rotação resultante para a representação *Roll-Pitch-Yaw*:

$$ \begin{aligned} &\mathrm{RPY}(\phi,\theta,\psi) = \mathrm{Rot}(z,\phi)\,\mathrm{Rot}(y,\theta)\,\mathrm{Rot}(x,\psi) = \\ &= \begin{bmatrix} \cos\phi & -\operatorname{sen}\phi & 0 & 0 \\ \operatorname{sen}\phi & \cos\phi & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos\theta & 0 & \operatorname{sen}\theta & 0 \\ 0 & 1 & 0 & 0 \\ -\operatorname{sen}\theta & 0 & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\psi & -\operatorname{sen}\psi & 0 \\ 0 & \operatorname{sen}\psi & \cos\psi & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \\ &= \begin{bmatrix} \cos\phi\cos\theta & \cos\phi\operatorname{sen}\theta\operatorname{sen}\psi - \operatorname{sen}\phi\cos\psi & \cos\phi\operatorname{sen}\theta\cos\psi + \operatorname{sen}\phi\operatorname{sen}\psi & 0 \\ \operatorname{sen}\phi\cos\theta & \operatorname{sen}\phi\operatorname{sen}\theta\operatorname{sen}\psi + \cos\phi\cos\psi & \operatorname{sen}\phi\operatorname{sen}\theta\cos\psi - \cos\phi\operatorname{sen}\psi & 0 \\ -\operatorname{sen}\theta & \cos\theta\operatorname{sen}\psi & \cos\theta\cos\psi & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \end{aligned} \tag{2.18} $$

A sequência de rotação ocorre respectivamente nos eixos $x - y - z$ mas, tendo em

vista que as rotações ocorrem em torno de eixos fixos, as matrizes homogêneas Rot() são pré-multiplicadas a cada nova rotação. Vale comentar que pode-se compreender essas rotações como feitas ao longo dos eixos móveis $z - y - x$, esta notação é conhecida como Euler 321.

Com o objetivo de isolar cada ângulo, primeiramente se compara a matriz homogênea RPY com a equação (2.17), explicitando-se as relações

$$n_x = \cos\phi\cos\theta \tag{2.19}$$

$$n_y = \operatorname{sen}\phi\cos\theta \tag{2.20}$$

$$n_z = -\operatorname{sen}\theta \tag{2.21}$$

$$o_z = \cos\theta\operatorname{sen}\psi \tag{2.22}$$

$$a_z = \cos\theta\cos\psi \tag{2.23}$$

O ângulo ϕ é encontrado de forma direta, ao se dividir a equação (2.20) pela (2.19)

$$\frac{\operatorname{sen}\phi}{\cos\phi} = \frac{n_y}{n_x} \tag{2.24}$$

ou, reescrevendo em função de atan2,

$$\phi = \operatorname{atan2}\left(n_y, n_x\right) \tag{2.25}$$

Somando o quadrado das equações (2.22) e (2.23) segue que

$$(\cos\theta)^2 = {o_z}^2 + {a_z}^2 \tag{2.26}$$

Dividindo-se (2.21) pela raiz quadrada de (2.26)

$$\frac{-\operatorname{sen}\theta}{\cos\theta} = \frac{n_z}{\sqrt{{o_z}^2 + {a_z}^2}} \tag{2.27}$$

ou, reescrevendo em função de atan2,

$$\theta = \operatorname{atan2}\left(-n_z, \sqrt{{o_z}^2 + {a_z}^2}\right) \tag{2.28}$$

De forma direta, outra vez, ψ é resultante da divisão da equação (2.22) pela (2.23)

$$\frac{\operatorname{sen}\psi}{\cos\psi} = \frac{o_z}{a_z} \tag{2.29}$$

ou, reescrevendo em função de atan2,

$$\psi = \text{atan2}\left(o_z, a_z\right) \tag{2.30}$$

2.3 CINEMÁTICA

A cinemática constitui o equacionamento mais fundamental para a geração de trajetórias de um manipulador.

> Para projetar sistemas de controle é necessário que se realize a descrição matemática dos sistemas físicos - plantas - que vão permitir a aplicação de ferramentas matemáticas para prever a resposta de saída para uma entrada definida [...] Os modelos devem permitir que o projeto de controle seja aplicável a um sistema físico real.(OHNISHI; SABANOVIC, 2011, p.1, tradução dos autores)

Existem duas abordagens para a solução da cinemática de um manipulador, a numérica e a de forma fechada. Por um lado, "soluções de forma fechada são desejáveis, porque são mais rápidas do que as numéricas e identificam prontamente todas as soluções possíveis" (KHATIB; SICILIANO, 2008, p.28, tradução dos autores). Por outro, além de que ao se aumentar o número de elos, cálculos analíticos em robótica se tornam tarefas tediosas (JAZAR, 2010, p.485, tradução dos autores), as soluções de forma fechada podem tornar-se impossíveis, e portanto, a abordagem numérica se faz necessária.

Neste trabalho, o estudo da cinemática será dividido em duas partes. Primeiramente se tratará da cinemática direta, que expressa a postura do efetuador em função das variáveis de junta de um manipulador. Em seguida, a modelagem da cinemática inversa possibilitará, de forma complementar, definir as coordenadas de junta necessárias para se atingir uma postura desejada. A Figura 9 expressa as relações indicadas.

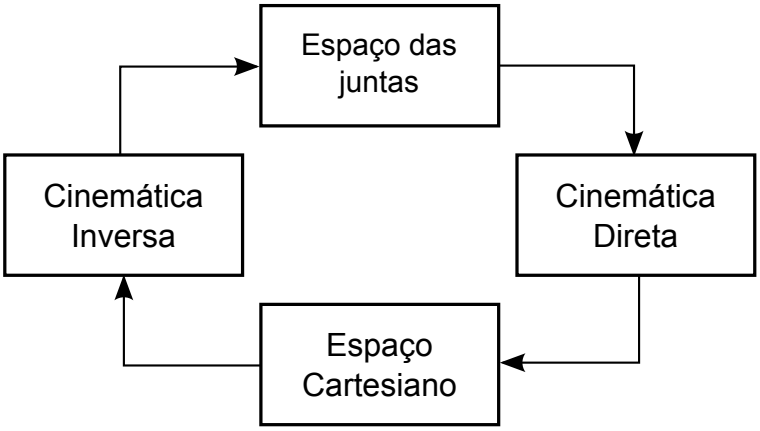

Figura 9: Relação entre cinemática direta e inversa
Fonte: Adaptado de Nagrath e Mittal (2007).

As subseções seguintes esclarecerão os conceitos apresentados.

2.3.1 CINEMÁTICA DIRETA

A mobilidade e destreza de um robô está diretamente associada à quantidade e à configuração das juntas. Para que seja possível prever com precisão seus movimentos, é necessário estabelecer uma função que relacione as coordenadas das juntas e o *frame* do efetuador. Para tanto, introduz-se o conceito de cinemática direta.

> Dito de modo mais formal, o problema da cinemática direta consiste em determinar a posição e orientação do efetuador final, dados os valores para as variáveis de junta do robô (SPONG; HUTCHINSON; VIDYASAGAR, 2004, p.61, tradução dos autores).

Esta relação pode ser expressada matematicamente como

$$(p_x, p_y, p_z, \phi, \theta, \psi) = f(\mathbf{q}) \tag{2.31}$$

onde p_x, p_y e p_z são as coordenadas da posição da origem do *frame* final e ϕ, θ e ψ são os ângulos RPY, que descrevem a orientação do mesmo. O argumento $\mathbf{q}$, por sua vez é o vetor de variáveis de junta, "que são ângulos de rotação, no caso de uma juntas de revolução ou deslocamentos no caso de juntas prismática" (SPONG; HUTCHINSON; VIDYASAGAR, 2004, p.62, tradução dos autores).

2.3.1.1 Notação de Denavit-Hartenberg

Existem diversos métodos para a alocação dos *frames* de um manipulador, que pode ser feita, de forma bastante arbitrária. É útil, porém, que se siga uma metodologia robusta como, por exemplo, o algoritmo de Denavit-Hartenberg (DH).

A universalidade algébrica do algoritmo de DH torna-o "[...] uma forma muito simples de modelar elos e juntas de robôs que pode ser utilizado para qualquer configuração, independentemente de sua sequência e complexidade" (NIKU, 2001, p.67, tradução dos autores).

A notação de DH, como relatado em Nagrath e Mittal (2007), estabelece a representação de um par junta-elo por apenas quatro parâmetros. As definições destes parâmetros, dadas por Santos (2001), são:

1. Comprimento de elo a_i
   Distância medida ao longo da normal comum entre os eixos das juntas.

$$a_i = \overline{O_i, (z_{i-1} \cap x_i)}\Big|_{x_i} \tag{2.32}$$

2. Deslocamento de junta d_i
   Traduz a distância entre elos ao longo do eixo da junta anterior.

$$d_i = \overline{O_i, (z_{i-1} \cap x_i)}\Big|_{z_{i-1}} \tag{2.33}$$

3. Ângulo de junta θ_i
   Ângulo entre o eixo de um elo e o eixo do elo seguinte.

$$\theta_i = \angle(x_i, x_{i-1})\Big|_{z_{i-1}} \tag{2.34}$$

4. Ângulo de torção do elo α_i
   Ângulo entre o eixo de uma junta e o eixo da junta seguinte.

$$\alpha_i = \angle(z_{i-1}, z_i,)\Big|_{x_i} \tag{2.35}$$

Sendo

- O_i - Ponto de origem do *frame* $\{i\}$.
- $z_i \cap x_i$ - Ponto de intersecção entre os eixos z_i e x_i.

- $\overline{(O_i, P_i)}\big|_{x_i}$ - Distância do ponto O_i ao ponto P_i medida ao longo do eixo x_i.
- $\angle(x_i, z_i,)\big|_{y_i}$ - Ângulo entre os eixos x_i e z_i medido na rotação em torno de y_i.

Uma importante observação é que, muito embora haja uma sistematização pelo algoritmo de DH, existem diversas possibilidades de alocação que ainda respeitam a notação escolhida. Em função desta flexibilidade, existe certa variação nas definições específicas do algoritmo dentro da literatura, que normalmente permitem um grande número de arbitrariedades. Fu e Gonzalez (1987), por exemplo, resume o processo específico de alocação dos *frames* em três regras simples:

1. O eixo $z_(i-1)$ é inserido ao longo do eixo de movimento da articulação i.
2. O eixo x_i é normal ao eixo $z_(i-1)$, e apontando para longe dele.
3. O eixo y_i completa o sistema de coordenadas seguindo a regra da mão direita.

Neste caso, percebe-se que a alocação da origem do *frame* 0 é arbitrária desde que esteja inserida ao longo do eixo z_0 e existem também várias possibilidades de alocação do eixo x_i.

2.3.1.2 Síntese da Matriz de Transformação Homogênea Final

Utilizando-se das propriedades de pré e pós-multiplicação descritas na Seção 2.2 e da equação (2.16), temos que a matriz de transformação homogênea final, ou seja, a equação que descreve a postura do *frame* do efetuador em relação ao *frame* da base do robô, é dada pelo produto das matrizes parciais:

$$ {}^{0}T_n = {}^{0}T_1 {}^{1}T_2 {}^{2}T_3 ... {}^{n-1}T_n \tag{2.36} $$

Percebe-se portanto que, para o desenvolvimento de tal equacionamento, é necessário se sintetizar primeiramente as matrizes de transformação homogêneas parciais. As equações (2.37) e (2.38) de Spong, Hutchinson e Vidyasagar (2004) contém, respectivamente, a definição e o desenvolvimento da matriz homogênea parcial genérica ${}^{i-1}T_i$, que relaciona duas juntas consecutivas em função dos quatro parâmetros de DH.

$$ {}^{i-1}T_i = \mathrm{Rot}(z, \theta_i)\,\mathrm{Tr}(l_i, 0, 0)\,\mathrm{Tr}(0, 0, d_i)\,\mathrm{Rot}(x, \alpha_i) \tag{2.37} $$

Assim,

$$
\begin{aligned}
{}^{i-1}T_i &= \begin{bmatrix} \cos\theta_i & -\operatorname{sen}\theta_i & 0 & 0 \\ \operatorname{sen}\theta_i & \cos\theta_i & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & l_i \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\alpha_i & -\operatorname{sen}\alpha_i & 0 \\ 0 & \operatorname{sen}\alpha_i & \cos\alpha_i & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \\
&= \begin{bmatrix} \cos\theta_i & -\operatorname{sen}\theta_i\cos\alpha_i & \operatorname{sen}\theta_i\operatorname{sen}\alpha_i & a_i\cos\theta_i \\ \operatorname{sen}\theta_i & \cos\theta_i\cos\alpha_i & -\cos\theta_i\operatorname{sen}\alpha_i & a_i\operatorname{sen}\theta_i \\ 0 & \operatorname{sen}\alpha_i & \cos\alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix}
\end{aligned} \tag{2.38}
$$

Uma vez sintetizadas as matrizes parciais, é possível substituí-las na equação (2.36), encontrando-se, por fim, a matriz de transformação homogênea final.

2.3.2 CINEMÁTICA INVERSA

A cinemática direta estabelece a relação entre os parâmetros do efetuador que descrevem sua posição e orientação a partir dos parâmetros das juntas. Contudo, em diversas em aplicações industriais um controle baseado exclusivamente em cinemática direta não será suficiente para a utilização do robô em um processo.

Em muitos casos, pretende-se controlar a posição e orientação do efetuador em trajetórias bem definidas. É neste contexto que surge o estudo da cinemática inversa, cujo objetivo é definir os parâmetros de junta de modo a obter uma determinada postura para o efetuador.

Matematicamente, pode-se definir a cinemática inversa para um manipulador de n juntas pela expressão

$$\mathbf{q} = f(p_x, p_y, p_z, \phi, \theta, \psi) \tag{2.39}$$

Em outras palavras, cinemática inversa é "a determinação de todas as configurações possíveis e viáveis das variáveis de junta, que permitam alcançar a posição e orientação especificada do efetuador do manipulador com relação à referência da base" (NAGRATH; MITTAL, 2007, p.113, tradução dos autores).

A resolução da cinemática inversa é, na maior parte dos casos, mais complexa que a da cinemática direta, pela existência de equações não-lineares e possibilidade de haver múltiplas soluções ou nenhuma para uma postura específica do efetuador.

Um dos processos utilizados para obter estas soluções é o método geométrico, em que se utiliza a inversão das matrizes de transformação parciais, partindo do pressuposto de que cada uma delas é função de uma única variável de junta, ou seja,

$$^{0}T_1 = f_1(q_1), \quad ^{1}T_2 = f_2(q_2), \quad ... \quad ,^{n-1}T_n = f_n(q_n) \tag{2.40}$$

Este método consiste em realizar manipulações na matriz de transformação homogênea final através de pré e pós-multiplicações de matrizes parciais, obtendo-se equações em função de um número menor de variáveis de juntas. Primeiramente, iguala-se a equação (2.36) à matriz de transformação homogênea final genérica H da equação (2.17):

$$^{0}T_n = {}^{0}T_1{}^{1}T_2{}^{2}T_3...^{n-1}T_n = H \tag{2.41}$$

Pode-se então realizar as manipulações algébricas, mantendo a veracidade da igualdade inicial. Neste ponto, intuições algébricas e geométricas tornam-se necessárias para encontrar manipulações que resultem em equações vantajosas (SICILIANO et al., 2009). Um exemplo de uma pré-multiplicação é

$$(^{0}T_1)^{-1} \cdot H = {}^{1}T_2{}^{2}T_3...^{n-1}T_n \tag{2.42}$$

Por fim, comparam-se os elementos das matrizes resultantes, de forma a obter as variáveis de junta isoladas. É interessante perceber que a quarta linha de cada matriz é sempre $(0, 0, 0, 1)$, ou seja, quatro das dezesseis equações geradas são triviais. Consequentemente, cada manipulação de matrizes resultará em doze equações não triviais, que podem se tornar úteis na solução da cinemática inversa.

Este método muitas vezes decompõe o problema espacial da cinemática, resultando em problemas planares Khatib e Siciliano (2008) que podem ser solucionados por geometria simples.

2.3.3 ESPAÇO DE TRABALHO

Espaço de trabalho é o lugar geométrico de todos os pontos em $\mathbb{R}^3$ que podem ser alcançados pelo *frame* do efetuador (SCHILLING, 1990). Para estabelecê-lo, é necessário levar em consideração a geometria do manipulador e limites mecânicos de juntas.

Como exposto em Siciliano et al. (2009), é relevante comentar que existe uma distinção entre o chamado espaço de trabalho alcançável (*reachable*) e destro (*dextrous*). O primeiro

engloba os pontos alcançáveis por *pelo menos* uma orientação do efetuador. O segundo é um subconjunto mais interessante na prática que compreende os pontos alcançáveis com orientação arbitrária.

Sendo $\mathbf{q}_{min}$ e $\mathbf{q}_{max}$ vetores denotando os limites de n juntas, o conjunto Q de todos os valores que variáveis de junta podem assumir é chamado de Espaço de Trabalho no Espaço de Juntas e tem a forma

$$Q = \{\mathbf{q} \in \mathbb{R}^n : \mathbf{q}_{min} \leq \mathbf{q} \leq \mathbf{q}_{max}\} \tag{2.43}$$

Com estas observações, o espaço de trabalho alcançável W_a é matematicamente o lugar geométrico de todos os pontos $p \in \mathbb{R}^3$ tangíveis ao efetuador, ou seja,

$$W_a = \{p(\mathbf{q}) \in \mathbb{R}^3 : \mathbf{q} \in Q\} \tag{2.44}$$

Em outras palavras, W_a é a imagem do Espaço de Trabalho no Espaço de Juntas Q sob a posição do efetuador $p(\mathbf{q})$ (SCHILLING, 1990). Convêm lembrar que a posição do efetuador é a quarta coluna da matriz de transformação homogênea final ${}^0T_n$.

Pode-se ainda gerar uma família de espaços de trabalho W_a com cada membro associado a uma orientação particular do efetuador.

2.4 CONCEITOS BÁSICOS DE LABVIEW®

LabVIEW (*Laboratory Virtual Instrument Engineering Workbench*) é um ambiente de desenvolvimento de propriedade da NI baseado em uma linguagem de programação gráfica chamada G. Sua natureza gráfica a torna ideal para aplicações como aquisição e análise de dados, automação e controle de instrumentos.

2.4.1 VIS

Simplificadamente, um instrumento virtual (VI - *Virtual Instrument*) é um elemento de programação do LabVIEW® , que consiste em um painel frontal, um diagrama de blocos e um ícone com conectores. O painel frontal é uma interface gráfica de usuário (GUI, na qual estão presentes controles (entrada de dados) e indicadores (saída de dados). O diagrama de blocos contém o código que executa o trabalho da VI, enquanto o ícone é uma representação visual com conectores para as entradas e saídas do programa. Múltiplas VIs podem ser usadas para criar aplicações em larga escala, invocando outras VIs (chamadas

subVIs), e não há limites para a profundidade desta hierarquia. A Figura 10 é um exemplo de painel frontal e diagrama de blocos de uma aplicação simples de monitoramento de temperatura.

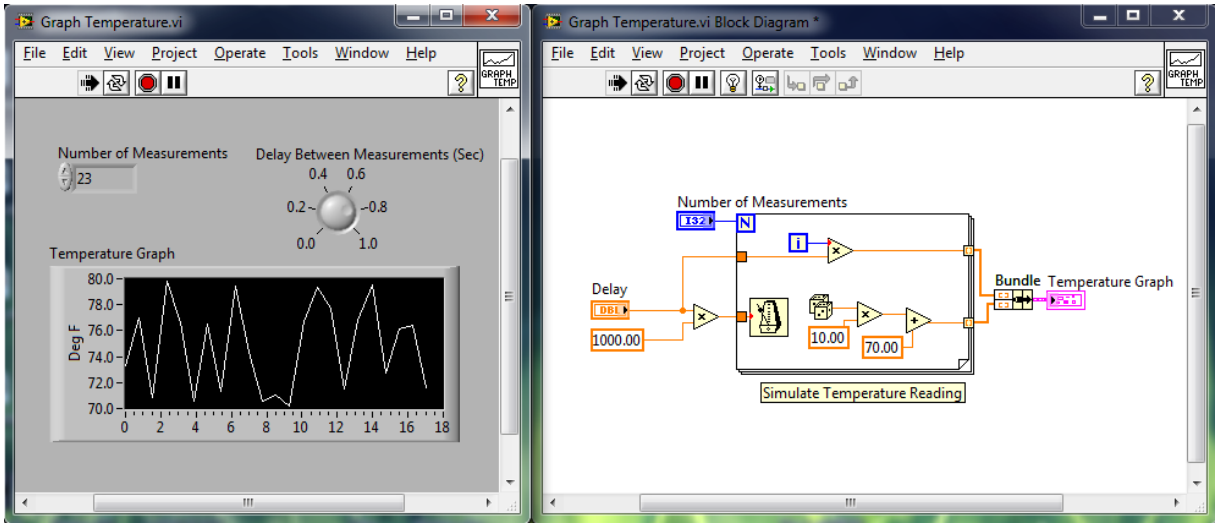

Figura 10: Exemplo de VI do LabVIEW®

A linguagem G, analogamente ao Java, não é uma linguagem interpretada; as VIs são compiladas em um código que a *engine* de execução do próprio LabVIEW® processa em tempo de execução (BITTER; MOHIUDDIN; NAWROCKI, 2007).

Programas LabVIEW® utilizam a extensão .vi. No entanto, várias VIs podem ser salvas em formato de biblioteca com a extensão .llb. Bibliotecas agrupam VIs para prover gerenciamento de arquivos.

2.4.2 ORIENTAÇÃO A FLUXO DE DADOS

Aplicações LabVIEW® são compostas por nós e fios e seguem um paradigma de programação orientada a fluxo de dados. Cada elemento em um diagrama que tem entrada ou saída é considerado um nó, que pode ser uma operação matemática simples ou mesmo uma subVI; os pontos de conexão entre os nós são fios. A coleção de nós e fios compõem o diagrama de fios que o compilador utilizará para execução. A ideia fundamental da orientação a fluxo de dados é que um nó só pode ser executado quando todas as suas entradas necessárias estão disponíveis.

É perfeitamente possível que vários nós recebam suas entradas aproximadamente ao mesmo tempo, de modo que tarefas paralelas sejam tratadas mais ou menos simultaneamente. Isso faz com que códigos multitarefa sejam muito facilmente

implementados. Contudo, semelhantemente a muitos sistemas operacionais *multithreading*, o LabVIEW® não dá quaisquer garantias sobre qual a ordem em que as operações paralelas irão ocorrer (BITTER; MOHIUDDIN; NAWROCKI, 2007).

2.4.3 VARIÁVEIS COMPARTILHADAS

Variáveis globais são um artifício utilizado quando dados precisam ser manipulados em várias VIs. A vantagem é que, uma vez definido, o dado pode ser lido ou escrito em VIs múltiplas. As variáveis locais são semelhantes às globais, mas só estão disponíveis na VI em que foram criadas.

Variáveis compartilhadas são um recurso recente introduzido no LabVIEW® 8.0; semelhantemente às globais, variáveis compartilhadas podem ser lidas e escritas por múltiplas VIs sem a conexão por fios. A diferença é que, com elas, os dados podem ser compartilhados não apenas entre VIs no mesmo computador, mas também através de uma rede.

2.4.4 LABVIEW® RT

O LabVIEW® RT é um conjunto de *hardware* e *software* da NI que permite a criação de sistemas de tempo real, ou seja, determinísticos e com temporização precisa (NATIONAL INSTRUMENTS, 2010b). O módulo RT permite que os usuários utilizem grande parte da paleta de funções básicas do LabVIEW® modificada para implementar sistemas de tempo real. Exemplos incluem controle de movimento, geração de sinais e aquisição de dados.

O Módulo RT também proporciona uma série de ferramentas novas de tempo real como o *Real-Time Execution Trace Toolkit*(análise e depuração de execução), *Real-Time FIFOs* (filas determinísticas), *loops* cronometrados e acesso de baixo nível a *hardware* de tempo real.

Há uma variedade de plataformas de *hardware* que se integram com os programas escritos com o Módulo RT, como a NI CRIO (entrada e saída reconfiguráveis) e a NI PXI (instrumentos com interface PCI). Além dos produtos da NI, o usuário pode converter a maioria dos sistemas *single-board*, computadores industriais e *Personal Computers* (PCs) *desktop* em sistemas de tempo real utilizando o sistema operacional LabVIEW® RT (NATIONAL INSTRUMENTS, 2010a).

2.5 CONCEITOS DE ENGENHARIA DE *SOFTWARE*

Como frisam Bitter, Mohiuddin e Nawrocki (2007), quando uma aplicação se torna grande deve haver uma série de considerações antes do início da codificação, como extensibilidade, flexibilidade, manutenção, reutilização de código e legibilidade. Tal imposição leva muitos desenvolvedores a aderir a um conjunto básico de princípios de desenvolvimento. Um deles, o modelo de ciclo de vida, descreve os passos a se seguir no desenvolvimento de *software*, desde as etapas conceituais até a manutenção e atualizações.

Há modelos de ciclo de vida amplamente utilizados em engenharia de *software*, como o *code-and-fix* e o cascata. O primeiro, muito presente em projetos pequenos, é baseado em codificação constante com correções de problemas apenas na medida em que aparecem. Em geral, este modelo leva a um retrabalho excessivo em arquiteturas mais complexas.

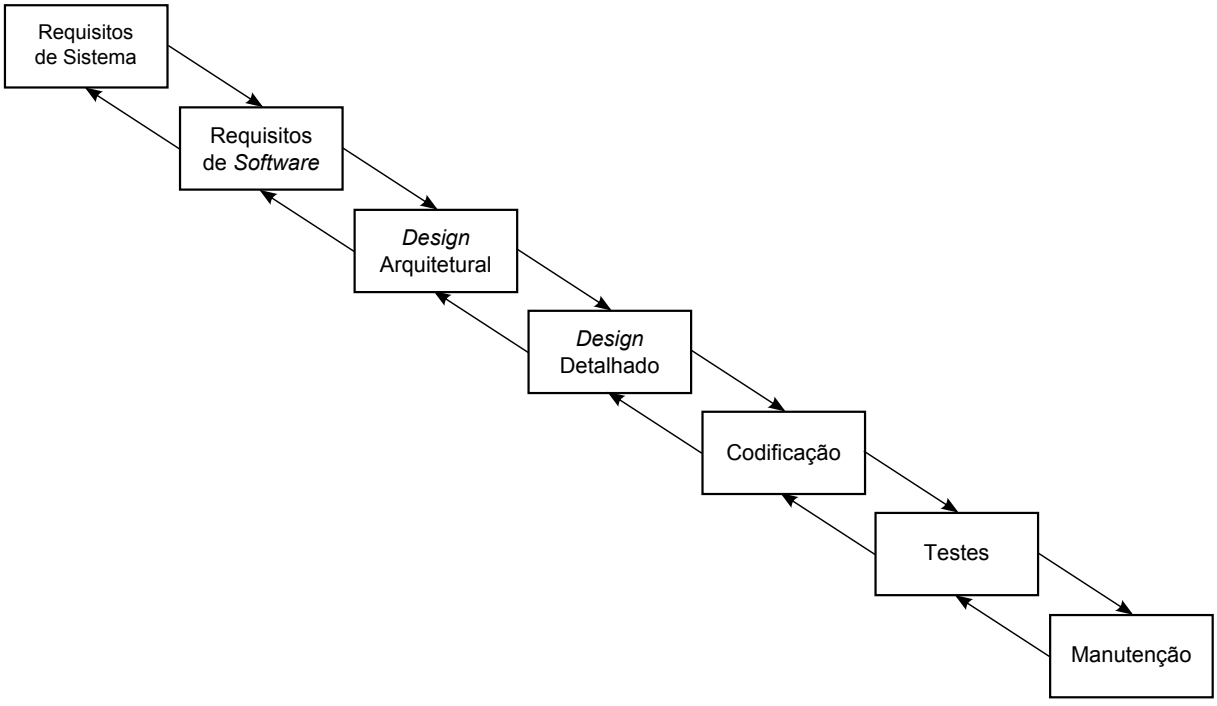

Figura 11: Modelo cascata

O modelo cascata é um clássico que, em sua configuração pura, consiste em várias fases sequenciais sem sobreposição, detalhadas em National Instruments (2003):

- Análise de requisitos de sistema: estabelecem-se os componentes para a construção do sistema, como requisitos de *hardware* e ferramentas de *software*;
- Análise de requisitos de *software*: avaliam-se as necessidades de capacidade e

funcionalidade da aplicação, como interface com o usuário, interação com outras aplicações, desempenho, etc;

- *Design* arquitetural: consiste em se decidir sobre a estrutura de alto nível da aplicação que atenderá aos requisitos. Descreve os componentes mais globais do *software* e suas interfaces;
- *Design* detalhado: descreve estes componentes em um nível que permita sua construção;
- Codificação: é a implementação propriamento dita do projeto;
- Testes: verifica-se o atendimento aos requisitos iniciais e a presença de falhas no código;
- Manutenção: resolvem-se problemas não observados no período de testes, solicitações de melhoria e atualizações.

O modelo cascata não proíbe retorno a uma fase anterior. Cada etapa exige revisão a fim de avaliar a possibilidade de transição para a próxima, e pode contemplar prototipação. No modelo em cascata puro não há sobreposição entre as fases, contudo tal medida pode se mostrar necessária para corrigir problemas de codificação ou falhas nos requisitos.

No caso de aplicações LabVIEW® , a fase de *design* arquitetural pode descrever a estrutura pela hierarquização das VIs e a modelagem comportamental do sistema. Como menciona Bitter, Mohiuddin e Nawrocki (2007), estruturalmente uma aplicação deve ser dividida em três níveis:

- Nível principal: consiste na interface do usuário e no executivo de teste;
- Nível de teste ou lógico: responsável pela execução de quaisquer atividades lógicas e de tomada de decisão;
- Nível de acionamento: executa todas as comunicações com dispositivos e outras aplicações.

Esta filosofia de camadas possibilita dividir as tarefas entre a equipe de desenvolvimento e organizar a hierarquia dos componentes do *software*. Na chamada abordagem *top-down*, adequada a aplicações de grande porte, toma-se a interface do usuário em primeiro lugar, como nível principal. O trabalho pode então continuar até o nível mais baixo, de acionamento.

Ainda no escopo do *design* arquitetural, há padrões de projeto, também muito conhecidos pelo termo original em inglês, *design patterns*, que descrevem soluções consagradas para problemas recorrentes no desenvolvimento de *software* (BLUME, 2007). Entre eles, vale citar a máquina de estados finita, o padrão mestre-escravo e o padrão produtor-consumidor.

2.5.1 MÁQUINA DE ESTADOS FINITA

A máquina de estados finita (MEF) pode ser usada para implementar algoritmos complexos de tomada de decisão representando-os por diagramas de transição de estados. As MEFs modelam aspectos dinâmicos de um sistema reativo, ou seja, um sistema que reage constantemente com o ambiente em resposta a eventos externos.

A UML (*Unified Modeling Language* - Linguagem de Modelagem Unificada) é uma linguagem visual utilizada para modelar sistemas computacionais por meio do paradigma de orientação a objetos que se tornou um padrão na indústria de engenharia de *software*. Dentre os diversos diagramas suportados pela versão atual da UML (2.0) as MEFs são um dos tipos de diagramas comportamentais (RUMBAUGH; JACOBSON; BOOCH, 2005).

Alguns conceitos importantes em uma MEF são estados, superestados, pseudoestados, eventos, transições, condições e ações. Um estado é uma condição ou situação estável de uma máquina, em que se executam atividades ou apenas se aguarda por um evento. Superestados são estados que contém outros estados em seu interior. Pseudoestados são uma abstração de estados com uma funcionalidade bem específica, como estado inicial e final. Um evento é um acontecimento relevante ocorrendo em um tempo bem definido, modelado como instantâneo, interno ou externo à máquina. Dentre os eventos possíveis estão o recebimento assíncrono de um sinal, chamada de rotinas ou métodos, tempo decorrido a partir da entrada no estado e mudança no valor de uma variável ou expressão. Uma transição é uma mudança de um estado para outro, podendo conter qualquer combinação de três elementos: eventos, verificação de uma condição (expressão lógica) e execução de ações.

A Figura 12 mostra um exemplo genérico de modelo comportamental de um sistema por MEF seguindo a notação UML 2.0. Para um maior detalhamento acerca da UML 2.0, consultar Rumbaugh, Jacobson e Booch (2005).

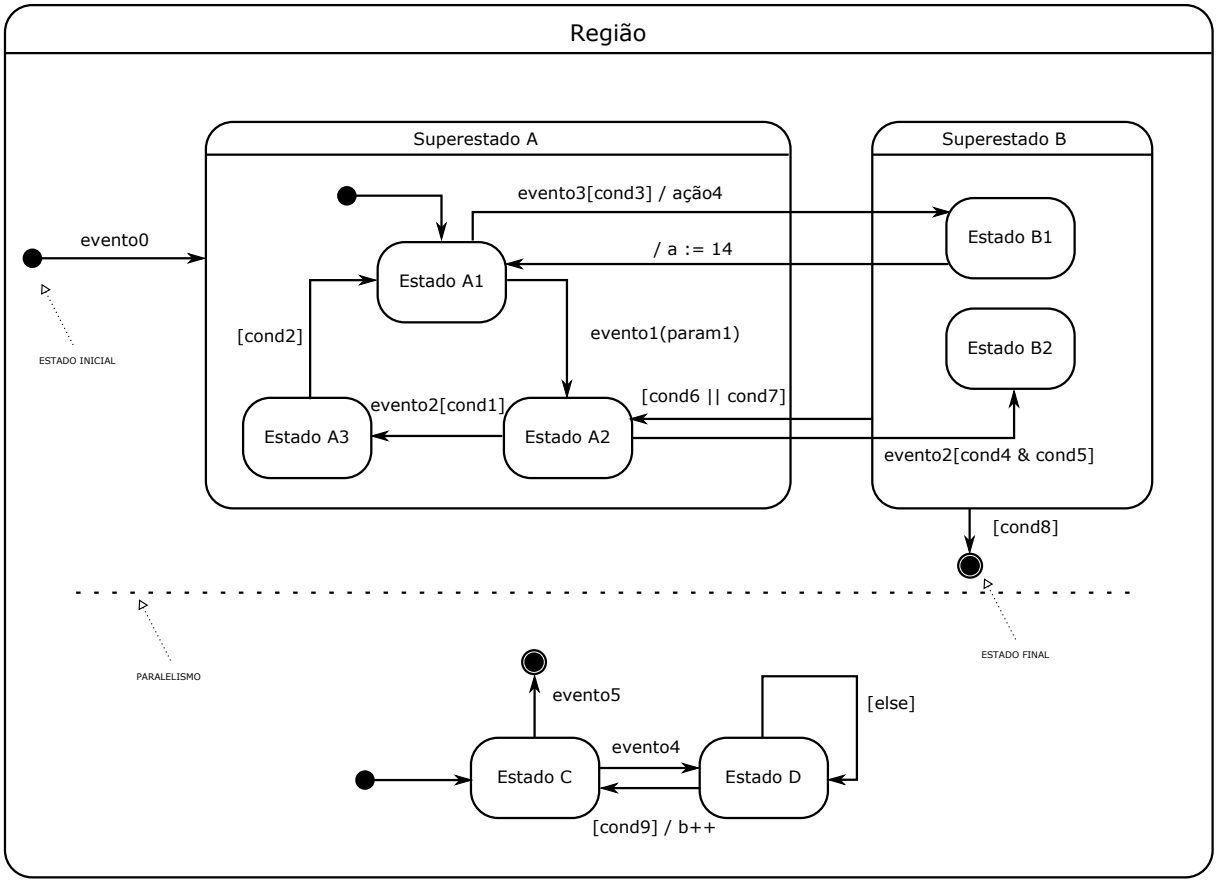

Figura 12: Exemplo genérico de MEF na notação UML 2.0

No exemplo, observa-se que há duas MEFs que executam independentemente separadas por uma linha tracejada e que a superior é composta por dois superestados, que podem ser considerados MEFs internas. As transições podem ser compostas pelo atendimento a um evento, como o decorrer de 5 s ou o acionamento de uma chave, pelo atendimento a uma condição lógica, como $[i = 0]$ e, ainda, caso esta condição seja satisfeita, uma ação pode ser executada, como a atribuição de um valor em uma variável ($/\ a := 14$). Qualquer um desses elementos pode compor sozinho uma transição, se necessário. Caso se deseje expressar todas as condições complementares, utiliza-se o símbolo $[else]$.

Em LabVIEW® , há uma série de possíveis implementações para MEFs, utilizando filas, notificadores, *loops* de sequência, etc., dependendo do nível de sofisticação necessário. Um exemplo é a máquina de estados convencional, conforme mostra a Figura 13.

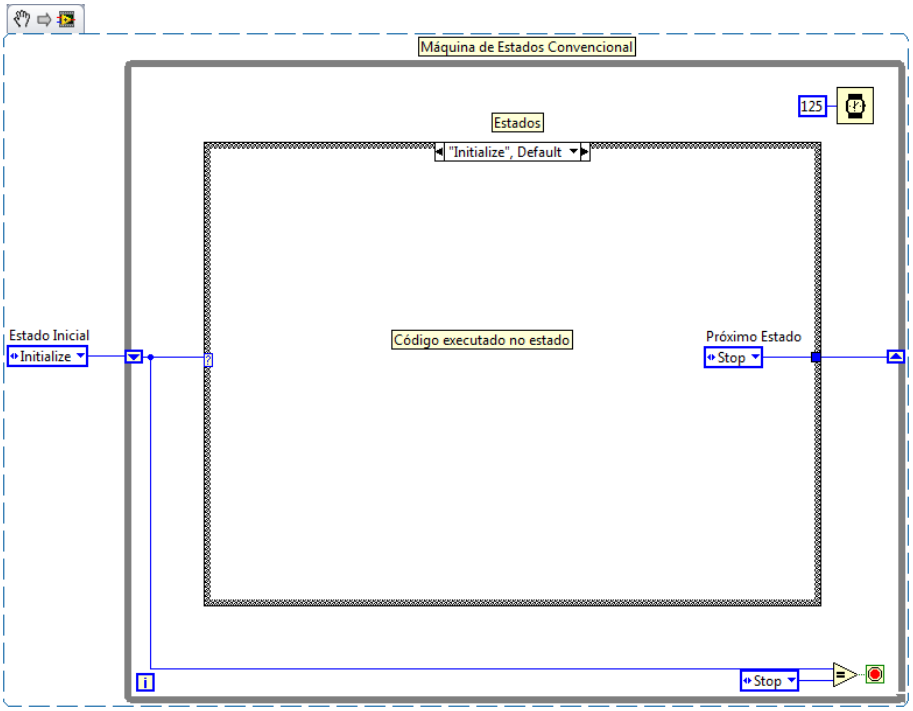

Figura 13: Exemplo de MEF no LabVIEW®

2.5.2 PADRÃO MESTRE-ESCRAVO

O padrão mestre/escravo é geralmente útil quando se têm dois ou mais processos que devem executar concorrentemente, mas em taxas diferentes. Dado que a comunicação de dados entre esses processos quebra a ideia de fluxo de dados, deve-se realizar a escrita e leitura a partir de conjuntos de dados disponíveis a nível global - ou seja, variáveis locais, globais, notificadores ou filas.

2.5.3 PADRÃO PRODUTOR-CONSUMIDOR

O padrão produtor/consumidor é uma subclasse do padrão mestre/escravo, onde a comunicação principal entre os *loops* é realizada por filas. Isto proporciona um efeito de acumulação; se o escritor ocasionalmente produzir informação mais rápido do que o leitor pode processá-la, dados não serão perdidos (BITTER; MOHIUDDIN; NAWROCKI, 2007). Isso pode ser especialmente útil para manipulação de eventos de GUI que levam um longo tempo para serem tratados. A Figura 14 apresenta uma possível implementação deste padrão em LabVIEW® .

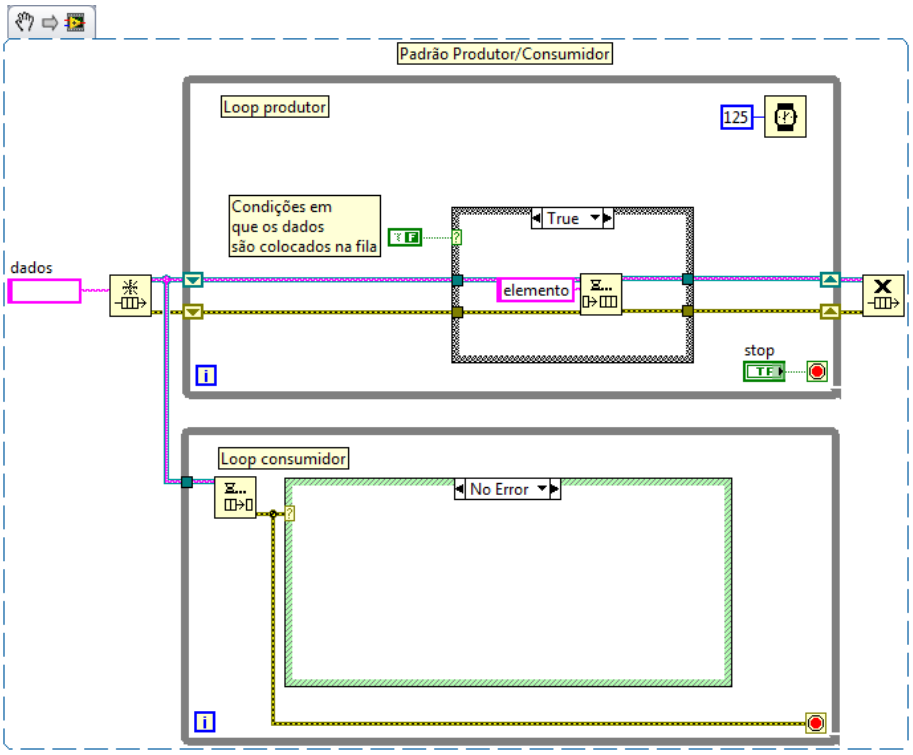

Figura 14: Padrão produtor/consumidor no LabVIEW®

3 MODELAGEM MATEMÁTICA

Este capítulo tem por objetivo apresentar o desenvolvimento da modelagem algébrica da cinemática inversa do robô Handler, passo intermediário para que seja possível se desenvolver o *software*. A programação efetiva do *software* em LabVIEW® será tratada no Capítulo 5.

A modelagem cinemática apresentada nas próximas subseções é baseada na proposta por Nagrath e Mittal (2007) e utiliza-se da notação de Denavit-Hartenberg já apresentada no Capítulo 2. Ao fim do presente capítulo, expõe-se uma análise comparativa entre a solução desenvolvida e a solução calculada por meio de um *software*, com o objetivo de validar o desenvolvimento.

3.1 ESTRUTURA E TIPOLOGIA DO MANIPULADOR HANDLER

O robô Handler é um manipulador de cadeia aberta com cinco elos interligados por meio de cinco juntas rotacionais, sendo o último elo uma garra (*gripper*). É classificado como robô antropomórfico (RRR), com o primeiro eixo de junta ortogonal aos próximos dois. A Figura 15 relaciona o manipulador com a representação simbólica de mecanismos equivalente, em vista perspectiva e lateral.

O robô Handler possui cinco GDL, portanto não é redundante. Seus três primeiros GDL compõem a estrutura conhecida por braço e posicionam o punho no espaço tridimensional. O punho do Handler é composto por duas juntas que garantem os movimentos de *pitch* e *roll*, não realizando, portanto, o movimento de *yaw*.

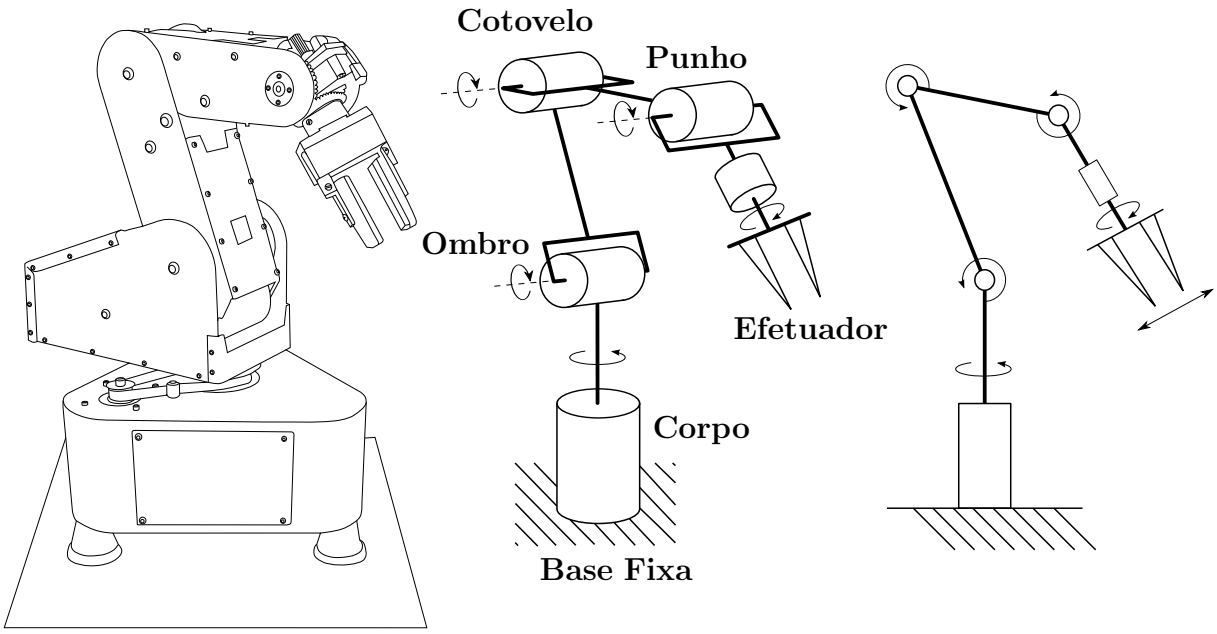

Figura 15: Representação simbólica para o Handler

3.2 ALOCAÇÃO DOS FRAMES

A notação de Denavit-Hartenberg (DH) tornou-se um padrão adotado pela maioria dos autores clássicos e será, portanto, a escolhida para a modelagem matemática, facilitando a busca por referências e validações. Em função da diferença das definições encontradas na literatura, como explicado no Capítulo 2, escolheu-se desenvolver um algoritmo que maximiza a quantidade de zeros nas matrizes de transformações parciais. Para tanto, a alocação do eixo x_i tornou-se mais limitada, visando a torná-lo paralelo ou coincidente a x_{i-1} sempre que possível.

O algoritmo exposto na Figura 16 foi baseado nos propostos por Nagrath e Mittal (2007), Fu e Gonzalez (1987) e Santos (2001). Toda situação não especificada pelo algoritmo para alocação de eixos, seja em relação a direção ou sentido, deve ser executada arbitrariamente.

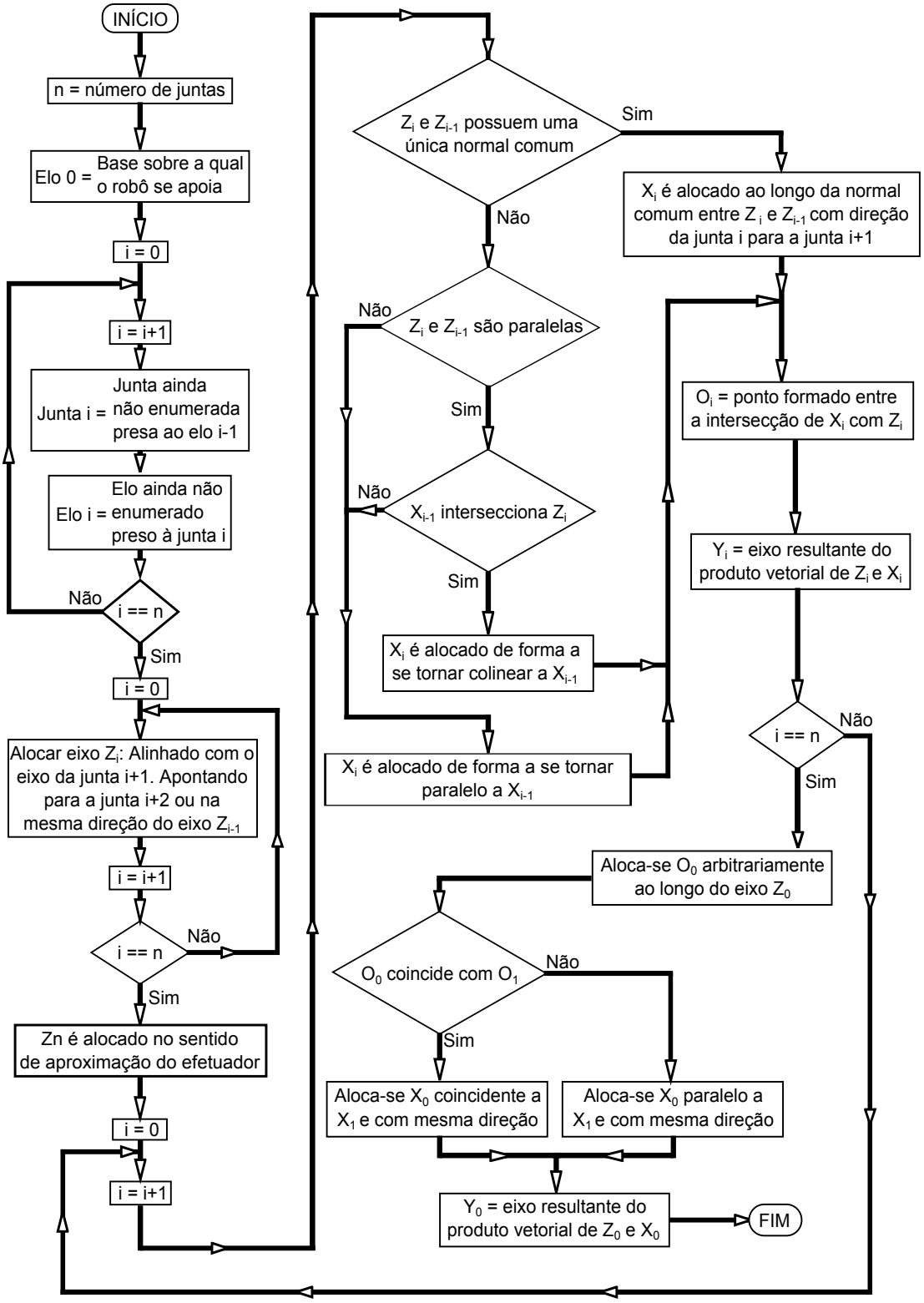

Figura 16: Fluxograma do algoritmo de DH executado

Como algumas arbitrariedades ainda são possíveis, justifica-se o detalhamento das escolhas realizadas ao se executar o algoritmo de DH apresentado na Figura 16.

O passo zero consiste na enumeração dos pares junta-elo do manipulador. Para o

robô Handler, inicia-se pela base (elo 0) e termina-se no efetuador (elo 5), resultando no exposto na Figura 17.

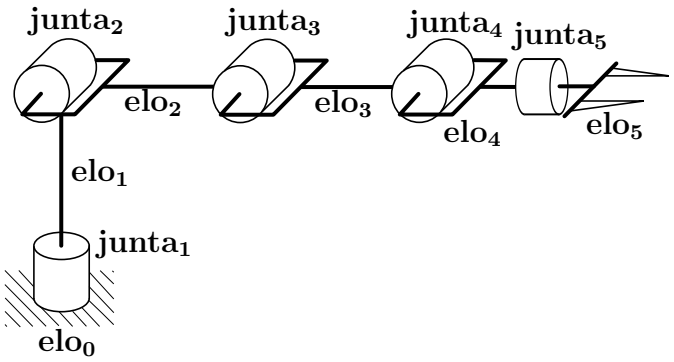

Figura 17: Numeração de Elos e Juntas

Enumerados os elos e as juntas, e lembrando que para o Handler $n = 5$, prossegue-se com os passos descritos abaixo:

1. Alocar z_i alinhado com o eixo da junta $i + 2$, para $i = 0$ até $i = n - 1$. z_n é alocado na direção de aproximação do efetuador

   - z_0 - Alocado na vertical, apontando para a junta 2.
   - z_1 - Arbitrariamente entrando no plano do papel.
   - z_2 - Alocado com o mesmo sentido de z_1.
   - z_3 - Alocado com o mesmo sentido de z_2.
   - z_4 - Alocado arbitrariamente na direção de aproximação do efetuador.
   - z_5 - Alocado na direção de aproximação do efetuador.

2. Alocar eixos restantes e a origem dos *frames* intermediários e do efetuador, ou seja, alocar x_i, O_i e y_i para i variando de 1 até n.

   - x_1 - Existe normal comum entre z_0 e z_1, logo, x_1 é alocado ao longo desta normal e aponta para a junta 2.
   - O_1 - Ponto de intersecção entre z_1 e x_1.
   - y_1 - Resultante do produto vetorial $z_1 \times x_1$
   - x_2 - z_1 e z_2 são eixos paralelos e x_1 intersecciona o eixo z_2, logo, x_2 é alocado colinear a x_1 e aponta no mesmo sentido.
   - O_2 - Ponto de intersecção entre z_2 e x_2.
   - y_2 - Resultante do produto vetorial $z_2 \times x_2$

- x_3 - z_2 e z_3 são eixos paralelos e x_2 intersecciona o eixo z_3, logo, x_2 é alocado colinear a x_2 e aponta no mesmo sentido.
- O_3 - Ponto de intersecção entre z_3 e x_3.
- y_3 - Resultante do produto vetorial $z_3 \times x_3$
- x_4 - Existe normal comum entre z_3 e z_4, logo, x_4 é alocado ao longo desta normal e aponta, arbitrariamente, para cima.
- O_4 - Ponto de intersecção entre z_4 e x_4.
- y_4 - Resultante do produto vetorial $z_4 \times x_4$
- x_5 - z_4 e z_5 são coincidentes. x_5 é alocado para cima, de modo a se tornar paralelo à x_4, apontando na mesma direção, e cortando z_5, arbitrariamente, no limite mecânico final do efetuador
- O_5 - Ponto de intersecção entre z_5 e x_5.
- y_5 - Resultante do produto vetorial $z_5 \times x_5$

3. Alocar O_0, x_0 e y_0

   - O_0 - É Alocado ao longo do eixo z_0, arbitrariamente. Escolheu-se o ponto onde o eixo z_0 intersecciona o plano da base sobre a qual o robô se apoia.
   - x_0 - Como O_0 não coincide com O_1, x_0 é alocado paralelamente e com mesmo sentido de x_1
   - y_0 - Resultante do produto vetorial $z_0 \times x_0$

A Figura 18 apresenta o resultado final da aplicação do algoritmo de DH para o Handler. O robô em posição inicial está posicionado à esquerda, e à direita representam-se os *frames* associados a cada junta.

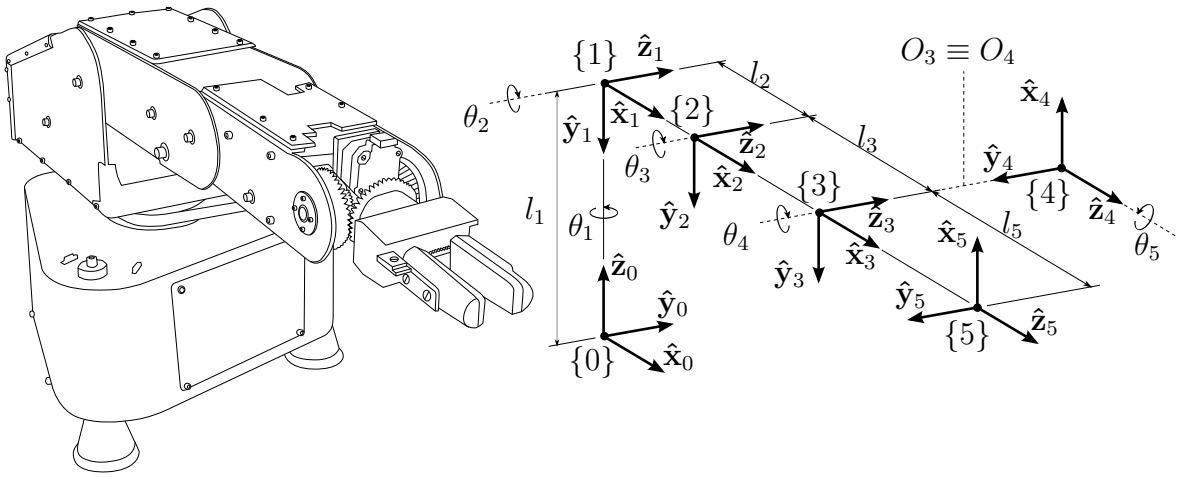

Figura 18: Handler com os eixos alocados

3.3 PARÂMETROS DE DENAVIT-HARTENBERG

Uma vez realizada a alocação dos *frames* utilizando-se do algoritmo de DH, é possível aquirir os quatro parâmetros que complementarão a descrição física do manipulador. Diferentemente da execução do algoritmo descrita na subseção anterior, os valores dos parâmetros de DH são únicos para determinada alocação de eixos, não dando espaço a arbitrariedades. Portanto, torna-se possível a aplicação direta das definições destes parâmetros, descritas pelas equações (2.32), (2.33), (2.34) e (2.35). A Tabela 2 apresenta os parâmetros DH do robô Handler.

Tabela 2: Parâmetros DH do robô Handler

Junta (i)	a_i	d_i	α_i	θ_i
1	0	l_1	$-90°$	θ_1
2	l_2	0	$0°$	θ_2
3	l_3	0	$0°$	θ_3
4	0	0	$-90°$	$\theta_4 - 90°$
5	0	l_5	$0°$	θ_5

3.4 SÍNTESE DA MATRIZ DE TRANSFORMAÇÃO HOMOGÊNEA FINAL

Aplicando-se a equação (2.36) para o caso do robô Handler, tem-se que a matriz de transformação homogênea final é o produto das matrizes homogêneas parciais:

$$^0T_5 = {}^0T_1\,{}^1T_2\,{}^2T_3\,{}^3T_4\,{}^4T_5 \tag{3.1}$$

O grafo da Figura 19 traduz a relação de transformação entre os *frames* do manipulador.

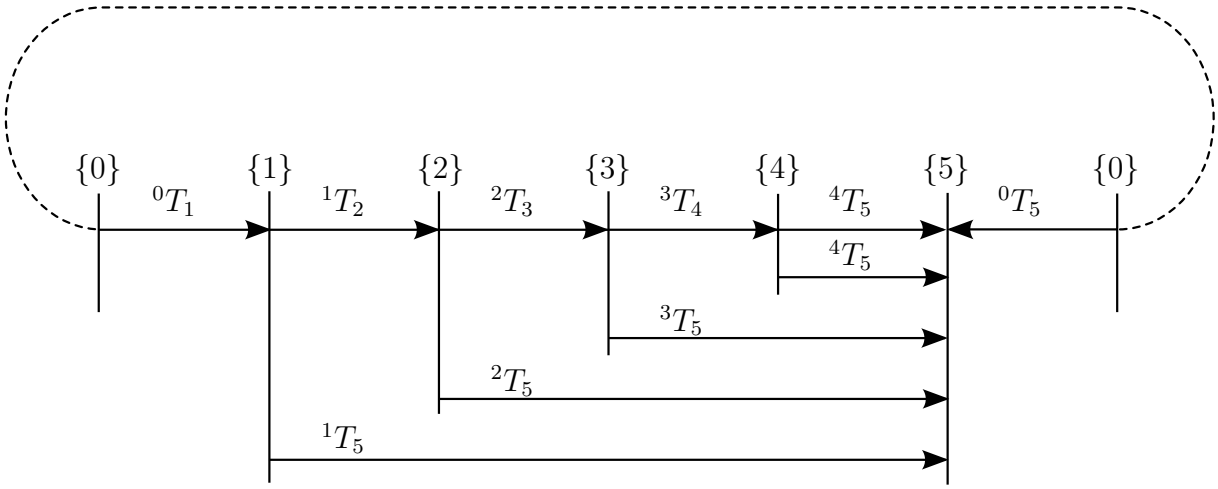

Figura 19: Grafo de transformações
Fonte: Adaptado de Paul (1986).

Como esperado, é necessário que se calcule primeiramente as matrizes de transformação homogêneas parciais, obtidas mediante a substituição dos parâmetros organizados na Tabela 2 na matriz de transformação parcial DH genérica da equação (2.38):

$$^0T_1 = \begin{bmatrix} \cos\theta_1 & 0 & -\operatorname{sen}\theta_1 & 0 \\ \operatorname{sen}\theta_1 & 0 & \cos\theta_1 & 0 \\ 0 & -1 & 0 & l_1 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.2}$$

$$^1T_2 = \begin{bmatrix} \cos\theta_2 & -\operatorname{sen}\theta_2 & 0 & \cos\theta_2 l_2 \\ \operatorname{sen}\theta_2 & \cos\theta_2 & 0 & l_2 \operatorname{sen}\theta_2 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.3}$$

$$^2T_3 = \begin{bmatrix} \cos\theta_3 & -\operatorname{sen}\theta_3 & 0 & \cos\theta_3 l_3 \\ \operatorname{sen}\theta_3 & \cos\theta_3 & 0 & l_3 \operatorname{sen}\theta_3 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.4}$$

$$
{}^{3}T_{4} = \begin{bmatrix} \operatorname{sen}\theta_4 & 0 & \cos\theta_4 & 0 \\ -\cos\theta_4 & 0 & \operatorname{sen}\theta_4 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.5}
$$

$$
{}^{4}T_{5} = \begin{bmatrix} \cos\theta_5 & -\operatorname{sen}\theta_5 & 0 & 0 \\ \operatorname{sen}\theta_5 & \cos\theta_5 & 0 & 0 \\ 0 & 0 & 1 & l_5 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.6}
$$

Executando-se a multiplicação matricial descrita na equação (3.1), tem-se que a matriz de transformação final é

$$
{}^{0}T_{5} = \begin{bmatrix} s_1 s_5 + c_1 c_5 s_{234} & c_5 s_1 - c_1 s_{234} s_5 & c_1 c_{234} & c_1(c_2 l_2 + c_{23} l_3 + c_{234} l_5) \\ c_5 s_1 s_{234} - c_1 s_5 & -c_1 c_5 - s_1 s_{234} s_5 & c_{234} s_1 & s_1(c_2 l_2 + c_{23} l_3 + c_{234} l_5) \\ c_{234} c_5 & -c_{234} s_5 & -s_{234} & l_1 - l_2 s_2 - l_3 s_{23} - l_5 s_{234} \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.7}
$$

onde

$$
\operatorname{sen}\theta_i = s_i \qquad \operatorname{sen}(\theta_i + \theta_j) = s_{ij} \qquad \operatorname{sen}(\theta_i + \theta_j + \theta_k) = s_{ijk}
$$

e

$$
\cos\theta_i = c_i \qquad \cos(\theta_i + \theta_j) = c_{ij} \qquad \cos(\theta_i + \theta_j + \theta_k) = c_{ijk}
$$

Esta é a notação a ser adotada, de agora em diante, para as funções trigonométricas $sen()$ e $cos()$.

3.5 SOLUÇÃO DA CINEMÁTICA DIRETA

Finalmente, pode-se solucionar a cinemática direta propriamente dita, ou seja, os valores de p_x, p_y, p_z, ϕ, θ e ψ (para RPY) em função dos ângulos de junta. Matematicamente, para o caso em estudo,

$$
(p_x, p_y, p_z, \phi, \theta, \psi) = f(\theta_1, \theta_2, \theta_3, \theta_4, \theta_5) \tag{3.8}
$$

Pela quarta coluna da matriz da equação (3.7), obtem-se as variáveis de posição do

efetuador p_x, p_y e p_z:

$$\boxed{p_x = c_1(c_2 l_2 + c_{23} l_3 + c_{234} l_5)} \tag{3.9}$$

$$\boxed{p_y = s_1(c_2 l_2 + c_{23} l_3 + c_{234} l_5)} \tag{3.10}$$

$$\boxed{p_z = l_1 - l_2 s_2 - l_3 s_{23} - l_5 s_{234}} \tag{3.11}$$

Comparando-se a matriz homogênea final da equação (3.7) com a equação (2.17) e substituindo seus elementos nas equações (2.25), (2.28) e (2.30), chega-se a

$$\boxed{\phi = \text{atan2}(c_5 s_1 s_{234} - c_1 s_5 \ , \ s_1 s_5 + c_1 c_5 s_{234})} \tag{3.12}$$

$$\boxed{\theta = \text{atan2}\left(-c_5 c_{234} \ , \ \sqrt{(c_{234} s_5)^2 + (s_{234})^2}\right)} \tag{3.13}$$

$$\boxed{\psi = \text{atan2}(c_{234} s_5 \ , \ s_{234})} \tag{3.14}$$

É importante perceber que estas equações possuem pontos de singularidade devido à descontinuidade da função atan2 no ponto $(0, 0)$. Será necessário, portanto, se realizar ajustes nestas condições, de modo a garantir a convergência do mesmo para valores corretos. Analisando-se as equações encontradas, temos que a situação de singularidade não ocorre para θ, e ocorre para ϕ e ψ quando, simultaneamente,

$$s_{234} = 0 \qquad \text{e} \qquad s_5 = 0 \tag{3.15}$$

A solução completa pode ser encontrada analisando-se a influência que a única variável de junta restante (q_1) exerce sobre cada ângulo nas condições descritas. Sabendo que esta junta possui o eixo coincidente a z_0, percebe-se que sua rotação afetará somente ϕ, uma vez que este é o ângulo a ser rotacionado em torno de z_0 na notação RPY. Temos então que ϕ e ψ são

$$\phi = \phi_0 + q_1 \tag{3.16}$$

$$\psi = \psi_0 \tag{3.17}$$

Onde ϕ_0 e ψ_0 são, respectivamente, ϕ e ψ calculados para o vetor $\mathbf{q}$ nulo. Ou seja, as equações para o ponto de singularidade de atan2 são

$$\boxed{\phi = \pi + q_1} \tag{3.18}$$

$$\boxed{\psi = 0} \tag{3.19}$$

3.6 SOLUÇÃO DA CINEMÁTICA INVERSA

A seção anterior tratou da resolução da cinemática direta para o robô Handler, estabelecendo a relação entre os parâmetros do efetuador que descrevem sua posição e orientação a partir dos cinco parâmetros das juntas.

Para a resolução da cinemática inversa, desenvolveu-se uma solução de forma fechada utilizando-se do método geométrico, já descrito na Subseção 2.3.2. Como descrito na equação (2.41), iguala-se a matriz de transformação homogênea final à matriz genérica H, ou seja,

$$ {}^0T_5 = {}^0T_1\,{}^1T_2\,{}^2T_3\,{}^3T_4\,{}^4T_5 = \begin{bmatrix} n_x & o_x & a_x & p_x \\ n_y & o_y & a_y & p_y \\ n_z & o_z & a_z & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.20} $$

Para a resolução da primeira e quinta variável de junta, rearranja-se equação (3.20), pré-multiplicando-se ambos os lados desta equação pela inversa da primeira matriz parcial:

$$ ({}^0T_1)^{-1} \cdot {}^0T_5 = {}^1T_5 = {}^1T_2\,{}^2T_3\,{}^3T_4\,{}^4T_5 \tag{3.21} $$

ou, de forma expandida,

$$ \begin{bmatrix} n_xc_1 + n_ys_1 & o_xc_1 + o_ys_1 & a_xc_1 + a_ys_1 & p_xc_1 + p_ys_1 \\ -n_z & -o_z & -a_z & l_1 - p_z \\ n_yc_1 - n_xs_1 & o_yc_1 - o_xs_1 & a_yc_1 - a_xs_1 & p_yc_1 - p_xs_1 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

$$ = \begin{bmatrix} s_{234}c_5 & -s_{234}s_5 & c_{234} & l_5c_{234} + l_3c_{23} + l_2c_2 \\ -c_{234}c_5 & c_{234}s_5 & s_{234} & l_5s_{234} + l_3s_{23} + l_2s_2 \\ -s_5 & -c_5 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.22} $$

Analisando-se o elemento $(3, 4)$ da equação (3.22) encontra-se a solução para θ_1:

$$ p_yc_1 - p_xs_1 = 0 \tag{3.23} $$

$$ \boxed{\theta_1 = \text{atan2}(p_y, p_x)} \tag{3.24} $$

De maneira semelhante ao procedimento anterior, θ_5 é obtido a partir da análise dos

elementos $(3,1)$ e $(3,2)$ da equação (3.22)

$$-s_5 = n_y c_1 - n_x s_1 \tag{3.25}$$

$$-c_5 = o_y c_1 - o_x s_1 \tag{3.26}$$

Logo:

$$\boxed{\theta_5 = \text{atan2}(n_x s_1 - n_y c_1, o_x s_1 - o_y c_1)} \tag{3.27}$$

A solução de θ_{234} auxiliará a continuidade do desenvolvimento, e é encontrada comparando os elementos $(1,3)$ e $(2,3)$ da equação (3.22):

$$c_{234} = a_x c_1 + a_y s_1 \tag{3.28}$$

$$s_{234} = -a_z \tag{3.29}$$

$$\boxed{\theta_{234} = \text{atan2}(-a_z\ ,\ a_x c_1 + a_y s_1)} \tag{3.30}$$

Neste momento, para encontrar as equações de θ_2, θ_3 e θ_4, torna-se possível desacoplar a parte da solução já encontrada da faltante, reduzindo o problema a um braço RRR que será solucionado por relações trigonométricas. Este desacoplamento é possível uma vez que a variação de θ_1 não influencia na distância entre o punho do robô e O_0, origem do sistema de referência da base. A Figura 20 mostra o novo problema.

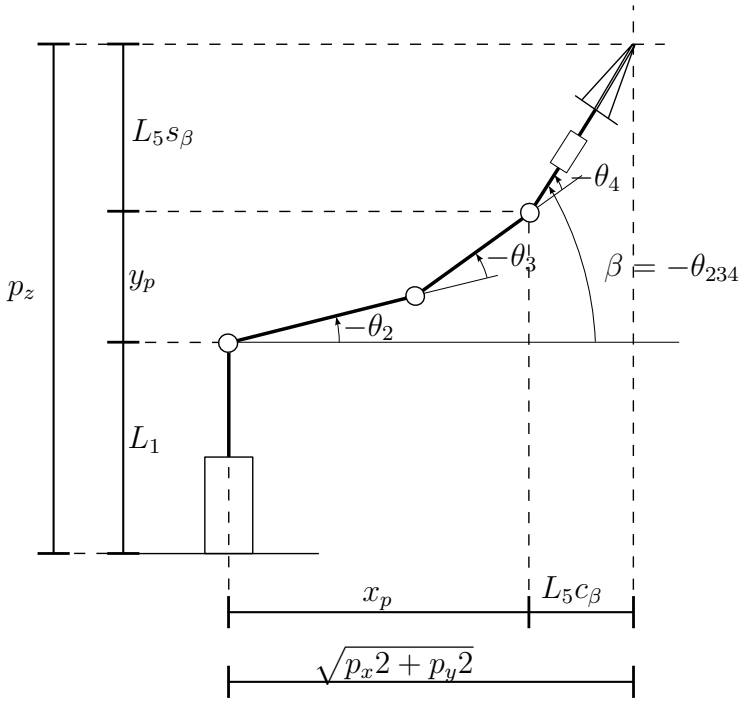

Figura 20: O problema reduzido(RRR)

Sendo x_p a projeção sobre o plano xy da distância da junta 2 até o punho, e y_p a mesma distância até o punho, porém projetada sobre o eixo z. p_x, p_y e p_z são as coordenadas da origem do *frame* da garra. Os ângulos θ_2, θ_3 e θ_4 estão com sinal negativo, pois a rotação está sendo realizada no sentido anti-horário da convenção definida na alocação dos eixos. Dando continuidade ao desenvolvimento, por observação tem-se que:

$$x_p = \sqrt{p_x^2 + p_y^2} - l_5 c_\beta \tag{3.31}$$

$$y_p = p_z - l_1 - l_5 s_\beta \tag{3.32}$$

Como θ_{234} já é conhecido, pode-se simplificar o problema ainda mais uma vez a um braço RR, conforme a Figura 21.

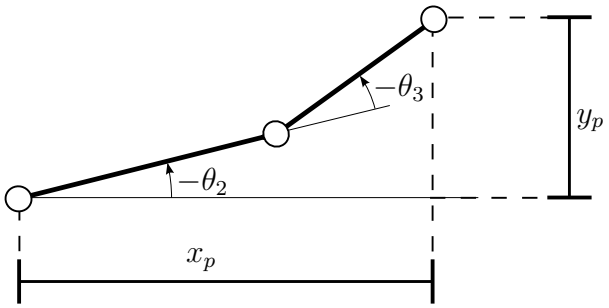

Figura 21: O problema reduzido novamente (RR)

Neste ponto as seguintes equações são definidas:

$$x_p = l_2 \cos(-\theta_2) + l_3 cos(-\theta_2 - \theta_3) \tag{3.33}$$

$$y_p = l_2 sen(-\theta_2) + l_3 sen(-\theta_2 - \theta_3) \tag{3.34}$$

e, utilizando-se das relações trigonométricas

$$cos(-\alpha) = cos(\alpha) \tag{3.35}$$

$$sen(-\alpha) = -sen(\alpha) \tag{3.36}$$

$$cos(\alpha + \gamma) = cos(\alpha)cos(\gamma) - sen(\alpha)sen(\gamma) \tag{3.37}$$

$$sen(\alpha + \gamma) = cos(\alpha)sen(\gamma) + sen(\alpha)cos(\gamma) \tag{3.38}$$

chega-se a

$$x_p = l_2c_2 + l_3c_2c_3 - l_3s_2s_3 \tag{3.39}$$

$$y_p = -l_2s_2 - l_3c_2s_3 - l_3s_2c_3 \tag{3.40}$$

elevando ambas as equações ao quadrado e somando-as,

$$x_p^2 = (l_2c_2)^2 + l_2l_3c_3(c_2)^2 - l_2l_3c_2s_2s_3 + (l_3c_2c_3)^2 - (l_3)^2c_2c_3s_2s_3 + (l_3s_2s_3)^2 \tag{3.41}$$

$$y_p^2 = (l_2s_2)^2 + l_2l_3c_3(s_2)^2 + l_2l_3c_2s_2s_3 + (l_3s_2c_3)^2 + (l_3)^2c_2c_3s_2s_3 + (l_3s_2s_3)^2 \tag{3.42}$$

$$x_p^2 + y_p^2 = l_2^2 + 2l_2l_3c_3 + l_3^2 \tag{3.43}$$

isola-se então θ_3

$$\boxed{\theta_3 = \pm \arccos \frac{x_p^2 + y_p^2 - l_2^2 - l_3^3}{2l_2l_3}} \tag{3.44}$$

Esta equação comprova a possibilidade, já comentada, da existência de múltiplas soluções da cinemática inversa. Percebe-se que existem, para uma mesma postura, duas respostas possíveis para θ_3. Objetivando-se encontrar a solução de θ_2, reescrevem-se as Equações 3.39 e 3.40, isolando c_2 e s_2:

$$c_2 = \frac{x_p + l_3s_2s_3}{l_2 + l_3c_3} \tag{3.45}$$

$$s_2 = \frac{-y_p - l_3c_2s_3}{l_2 + l_3c_3} \tag{3.46}$$

Substituindo, primeiramente, a Equação 3.46 na Equação 3.45 e desenvolvendo:

$$c_2 = \frac{x_p + l_3s_3\left(\dfrac{-y_p - l_3c_2s_3}{l_2 + l_3c_3}\right)}{l_2 + l_3c_3} \tag{3.47}$$

$$c_2 = \frac{x_p(l_2 + l_3c_3) - l_3s_3(y_p + l_3c_2s_3)}{(l_2 + l_3c_3)^2} \tag{3.48}$$

$$c_2 = \frac{x_p(l_2 + l_3c_3) - l_3s_3y_p - c_2(l_3s_3)^2}{(l_2^2 + 2l_2l_3c_3 + (l_3c_3)^2)} \tag{3.49}$$

$$c_2(l_2^2 + 2l_2l_3c_3 + (l_3c_3)^2 + (l_3s_3)^2) = x_p(l_2 + l_3c_3) - l_3s_3y_p \tag{3.50}$$

Por fim, comparando com a Equação 3.43, isola-se c_2:

$$c_2 = \frac{x_p(l_2 + l_3c_3) - l_3s_3y_p}{x_p^2 + y_p^2} \tag{3.51}$$

Retornando, agora, às Equações 3.46 e 3.45 e substituindo, desta vez, c_2 em s_2,

$$s_2 = \frac{-y_p - l_3 s_3 \left(\frac{x_p + l_3 s_2 s_3}{l_2 + l_3 c_3} \right)}{l_2 + l_3 c_3} \tag{3.52}$$

Realizando-se um desenvolvimento análogo ao demonstrado para c_2, tem-se que

$$s_2 = \frac{-y_p(l_2 + l_3 c_3) - l_3 s_3 (x_p + l_3 s_2 s_3)}{(l_2 + l_3 c_3)^2} \tag{3.53}$$

$$s_2 = \frac{-y_p(l_2 + l_3 c_3) - l_3 s_3 x_p - s_2 (l_3 s_3)^2}{(l_2^2 + 2 l_2 l_3 c_3 + (l_3 c_3)^2)} \tag{3.54}$$

$$s_2(l_2^2 + 2 l_2 l_3 c_3 + (l_3 c_3)^2 + (l_3 s_3)^2) = -y_p(l_2 + l_3 c_3) - l_3 s_3 x_p \tag{3.55}$$

onde se compara com a Equação 3.43 e se isola s_2, de forma que

$$s_2 = \frac{-y_p(l_2 + l_3 c_3) - l_3 s_3 x_p}{x_p^2 + y_p^2} \tag{3.56}$$

Dividindo a Equação 3.56 pela Equação 3.51:

$$\frac{s_2}{c_2} = \tan(\theta_2) = \frac{-y_p(l_2 + l_3 c_3) - l_3 s_3 x_p}{x_p(l_2 + l_3 c_3) - l_3 s_3 y_p} \tag{3.57}$$

de forma que esta equação da cinemática inversa será:

$$\boxed{\theta_2 = \text{atan2}(-y_p(l_2 + l_3 c_3) - l_3 s_3 x_p, x_p(l_2 + l_3 c_3) - l_3 s_3 y_p)} \tag{3.58}$$

Por fim, o valor da última variável, θ_4, é calculado em função das outras já encontradas:

$$\boxed{\theta_4 = \theta_{234} - \theta_3 - \theta_2} \tag{3.59}$$

3.7 VALIDAÇÃO DA CINEMÁTICA

A validação das equações de cinemática direta e inversa será auxiliada pelo *software* MatLab® [1]. A decisão de utilizá-lo é fundamentada na familiaridade dos integrantes com a linguagem de programação, na robustez dos resultados gerados e na possibilidade de se utilizar variáveis simbólicas ao longo deste processo.

O primeiro passo da validação é conferir o levantamento dos parâmetros DH e, consequentemente, a alocação de eixos. Esta verificação foi feita por meio de uma

[1] Matlab é marca registrada da Mathworks Inc.

ferramenta para o apoio ao estudo de robótica chamada *Robotics Toolbox*[2] de Peter Corke. Atualmente, esta ferramenta está na oitava edição e possui diversas funções numéricas úteis para a simulação de manipuladores.

O trecho de código exibido na Figura 22 atribui os parâmetros DH a cada par junta-elo do manipulador a ser modelado. A Figura 23 apresenta o resultado do modelo matemático, plotado em 3D pela *Toolbox*.

```
lst1=link([-pi/2 0 0 l1 0]);
lst2=link([0 l2 0 0 0]);
lst3=link([0 l3 pi/2 0 0]);
lst4=link([-pi/2 0 0 0 0 -pi/2]);
lst5=link([0 0 0 l5 0]);

handler = robot({lst1 lst2 lst3 lst4 lst5});
```

Figura 22: Trecho do código de Matlab®

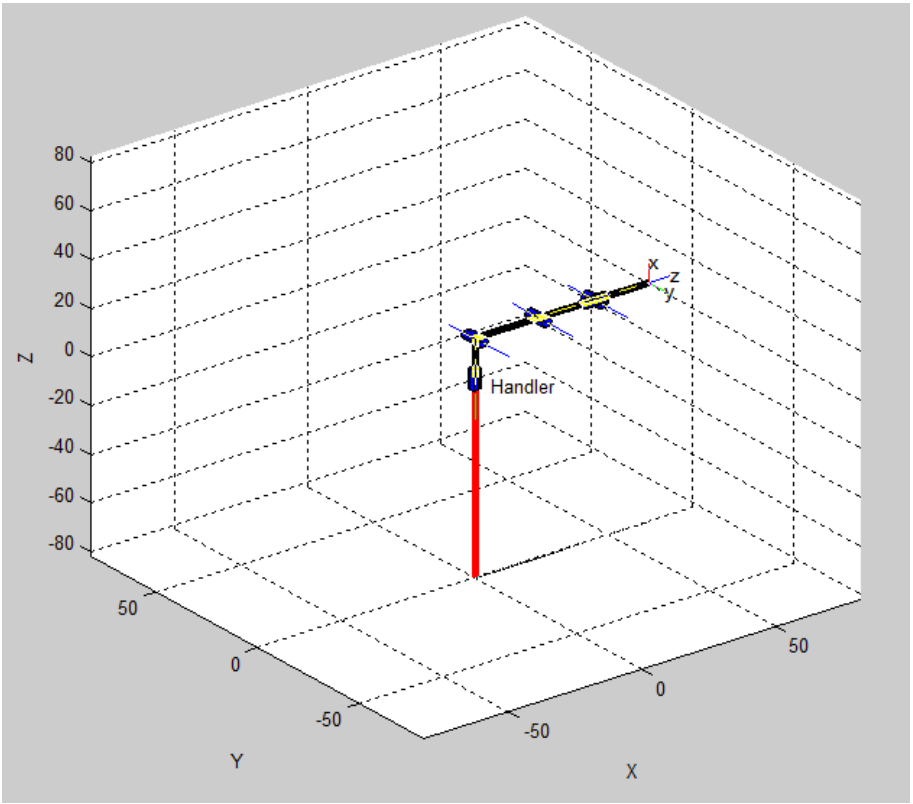

Figura 23: Modelo gerado pela *Robotics Toolbox*

Uma vez que o modelo vai ao encontro das expectativas, pode-se afirmar que a alocação dos *frames* e o levantamento dos parâmetros de DH foram executadas com

[2] *Toolbox* é um conjunto de *scripts* gerados para complementar as funcionalidades de um *software*.

êxito. Possuindo esta garantia, as multiplicações matriciais e manipulações algébricas realizadas a partir deste ponto do desenvolvimento foram cuidadosamente recalculadas no Matlab® por meio de variáveis simbólicas.

Um exemplo do produto resultante deste processo é a matriz de transformação homogênea final capturada diretamente da tela do Matlab® , conforme a Figura 24.

```
====== Matriz de transformação efetuador-base ====================
--------- T50 ----------

ans =

[ s1*s5 + c1*c5*s234,   c5*s1 - c1*s234*s5, c1*c234, c1*(c2*l2 + c23*l3 + c234*l5)]
[ c5*s1*s234 - c1*s5, - c1*c5 - s1*s234*s5, c234*s1, s1*(c2*l2 + c23*l3 + c234*l5)]
[            c234*c5,             -c234*s5,   -s234, l1 - l2*s2 - l3*s23 - l5*s234]
[                  0,                    0,       0,                            1]
```

Figura 24: Matriz gerada pelo Matlab®

Por fim, para conclusão do processo de validação da modelagem matemática, um programa, contendo 3 etapas, foi desenvolvido em Matlab® . As etapas são:

- Recebendo como parâmetro de entrada um vetor de variáveis de juntas qualquer, à escolha do usuário, este programa utiliza-se das equações da cinemática direta para, primeiramente, calcular a postura final do efetuador atingida;
- Em seguida essa postura serve como entrada das equação para cinemática inversa, retornando um vetor de variáveis de juntas que é uma das possíveis soluções da cinemática inversa e, em alguns casos, idêntica à entrada inicialmente;
- Devido às múltiplas soluções possíveis da etapa anterior, uma terceira etapa torna-se necessária. O processo da primeira etapa é refeito para o novo vetor de variáveis de junta, gerando uma nova postura final que é, no caso das equações estarem corretas, necessariamente igual à encontrada anteriormente.

As equações de cinemática direta e inversa exigem, também, que se obtenham os valores do comprimento dos elos do robô. Com a finalidade de se completar a validação,

os seguintes valores aproximados de l_1, l_2, l_3 e l_5 foram utilizados:

$$l_1 = 16,5cm$$
$$l_2 = 24,0cm$$
$$l_3 = 21,0cm$$
$$l_5 = 20,5cm$$

O exemplo abaixo visa a uma melhor compreensão do funcionamento do programa descrito. Definiu-se um vetor de variáveis de junta qualquer $\mathbf{q}_{arbitrario}$:

$$\mathbf{q}_{arbitario} = \begin{bmatrix} \frac{\pi}{3} \\ -\frac{\pi}{5} \\ -\frac{\pi}{4} \\ \frac{\pi}{3} \\ \pi \end{bmatrix} \tag{3.60}$$

Aplicou-se então $\mathbf{q}_{arbitario}$ como entrada do programa, que retornou as informações vistas na Figura 25.

```
q_entrada =

    1.0472   -0.6283   -0.7854    1.0472    3.1416


postura_efetuador =

   20.9200   36.2344   58.6948    1.0472    1.2043   -0.0000


q_saida =

    1.0472   -1.3585    0.7854    0.2066    3.1416


postura_efetuador2 =

   20.9200   36.2344   58.6948    1.0472    1.2043   -0.0000
```

Figura 25: Tela com o dados retornados

Como definido como uma possibilidade, o vetor de variáveis de junta retornado não foi idêntico ao vetor de entrada, porém, independentemente disso, a postura atingida por ambos foi idêntica. Validam-se assim as equações de cinemática encontradas.

4 *HARDWARE* DO ROBÔ HANDLER

O presente capítulo é dividido em duas seções. A primeira fornece uma visão geral do estado prévio do robô e a segunda classifica-o conforme conceitos de elementos do robô e espaço de trabalho, descrevendo o projeto e implementação de mudanças necessárias no *hardware*.

4.1 VISÃO GERAL DO SISTEMA ANTERIOR

Pode-se dividir o sistema robótico compreendido pelo Handler como a união de três unidades: controle, acionamento e atuadores. Conforme descrito no trabalho de Bicudo, Turesso e Haluc (2010), os atuadores do robô recebiam energia de duas fontes 12V em série. O outro conjunto formado por duas fontes do mesmo modelo em série alimentavam o sistema de comando e acionamento.

O conjunto de atuadores foi implementado com seis motores de passo para movimentação dos eixos e movimento de abertura e fechamento de garra. O movimento dos eixos dos motores era e é transmitido às juntas do manipulador por meio de polias e correias. Toda a estrutura mecânica funcionava adequadamente.

O controle era realizado por meio de três CLPs que recebiam sinais de sensores de fim de curso e gerenciavam o movimento dos motores através do envio de sinais digitais para a unidade de acionamento.

O condicionamento dos sinais de controle era realizado por seis *drivers* de potência da marca Autonics, que acionavam cada motor de passo individualmente.

4.2 ENGENHARIA DO NOVO *HARDWARE*

A principal modificação no *hardware* foi a substituição dos CLPs pela placa de controle NI PCI-6601, possibilitando a implementação do *software* descrita no Capítulo 5. As outras modificações apresentadas nesta seção são consequência direta da troca.

4.2.1 SISTEMA DE ATUAÇÃO

4.2.1.1 Fonte de Alimentação Primária

Retiraram-se duas das quatro fontes originais em virtude da remoção dos CLPs. A Figura 26 ilustra o novo esquema de alimentação do robô Handler.

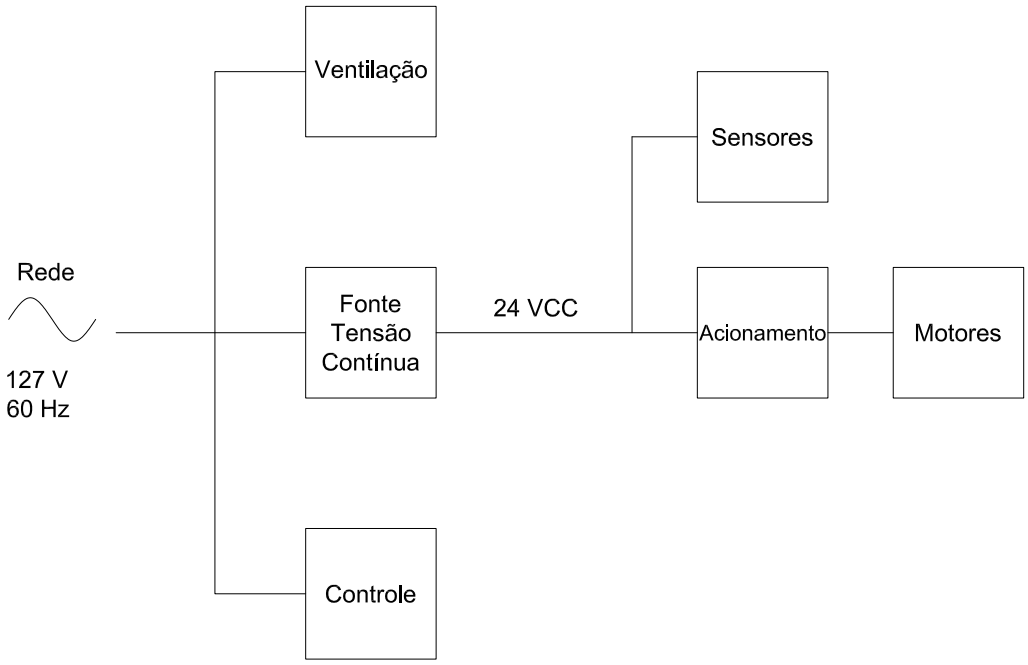

Figura 26: Diagrama unifilar de alimentação

Os motores e sensores recebem energia das duas fontes restantes em série, modelo TRIDONIC TALEXXconverter 0100 K240-2 12V, cujas principais especificações técnicas estão na Tabela 3.

Tabela 3: Dados técnicos da fonte de alimentação

Parâmetro	Valor
Range de Tensão de Entrada AC	100-264 V_{RMS}
Range de Tensão de Entrada DC	120-240 V_{DC}
Corrente de Entrada @ 230 V 50 Hz	0,6 A
Frequência	0/50/60 Hz
Eficiência	>80 %
Tensão de Saída	12 V_{DC}
Potência de Saída	10-100 W

Fonte: Adaptado de Bicudo, Turesso e Haluc (2010).

4.2.1.2 Atuadores

Nenhuma modificação foi necessária neste quesito. O robô Handler possui seis atuadores elétricos do tipo motor de passo, cuja convenção para numeração consta na Figura 27.

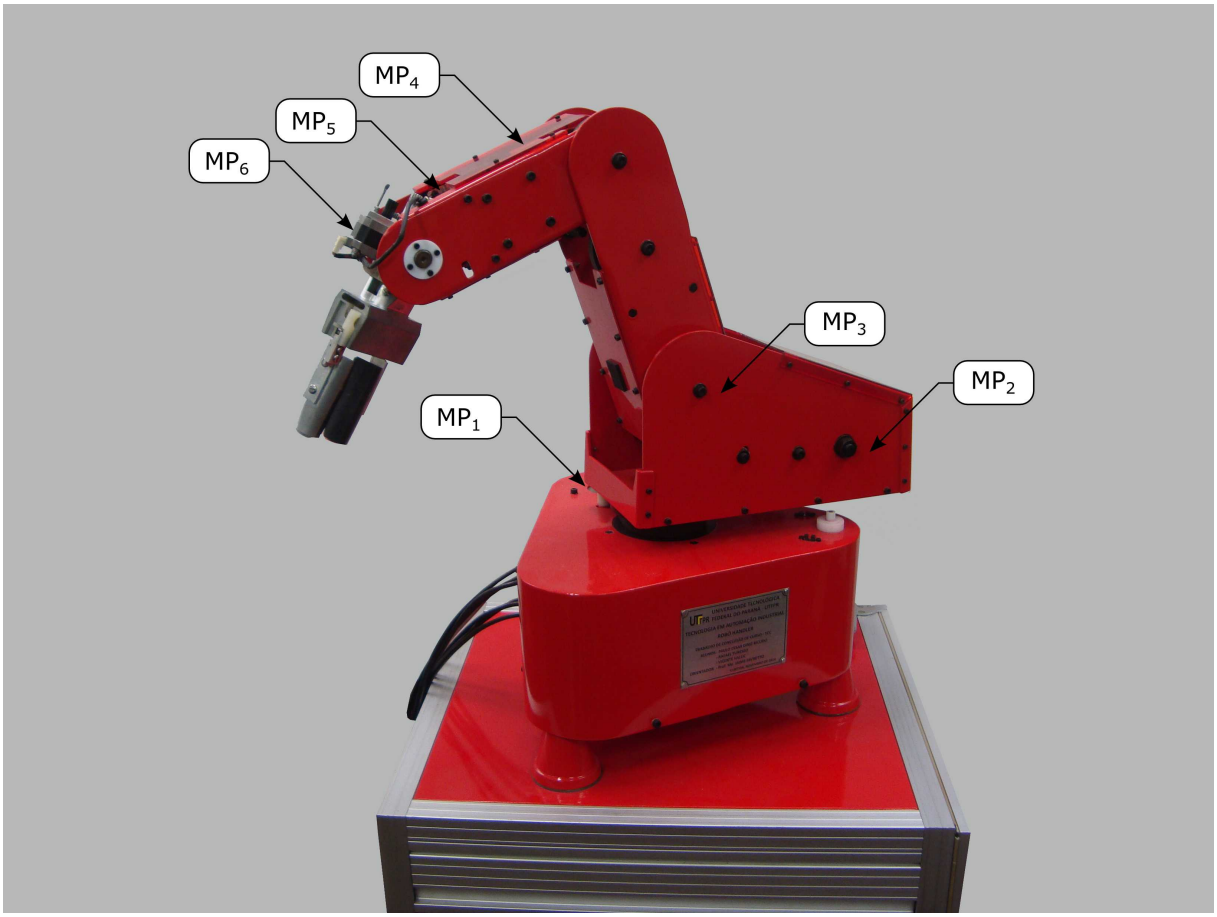

Figura 27: Identificação e localização dos atuadores

$MP_i,\ 1 \leq i \leq 3$: motor de passo que rotaciona a junta i

MP_4, MP_5 : motores de passo que comandam, simultaneamente, as juntas 4 e 5

MP_6 : motor de passo que comanda a garra

A tabela 4 reúne os dados técnicos mais relevantes de cada motor de passo presente no robô.

Tabela 4: Dados técnicos dos motores de passo do robô

Parâmetro	MP_1	MP_2	MP_3	MP_4	MP_5	MP_6
Marca	Autonics	Autonics	Syncro	Syncro	Syncro	Syncro
Modelo	A2K M243	A6K G264	841.001-4	851.111-7	851.211-8	851.111-7
N° de Fases	2	2	4	5	5	5
N° de Fios	6	6	6	10	10	10
N_{passos}	500	500	500	200	200	200
Tipo	Unipolar	Unipolar	Unipolar	Bipolar	Bipolar	Bipolar
Tensão	5 V_{DC}	5 V_{DC}	5 V_{DC}	2,12 V_{DC}	3,62 V_{DC}	2,12 V_{DC}
Corrente/Fase	1,2	2,0	1,0	1,25	1,25	1,25
Torque	0,206 Nm	0,570 Nm	0,400 Nm	1,100 Nm	2,200 Nm	1,100 Nm

Fonte: Adaptado de Bicudo, Turesso e Haluc (2010).

4.2.1.3 Amplificação de Potência

O condicionamento dos sinais de controle é realizado por seis *drivers* de potência da marca Autonics, que acionam cada motor de passo individualmente.

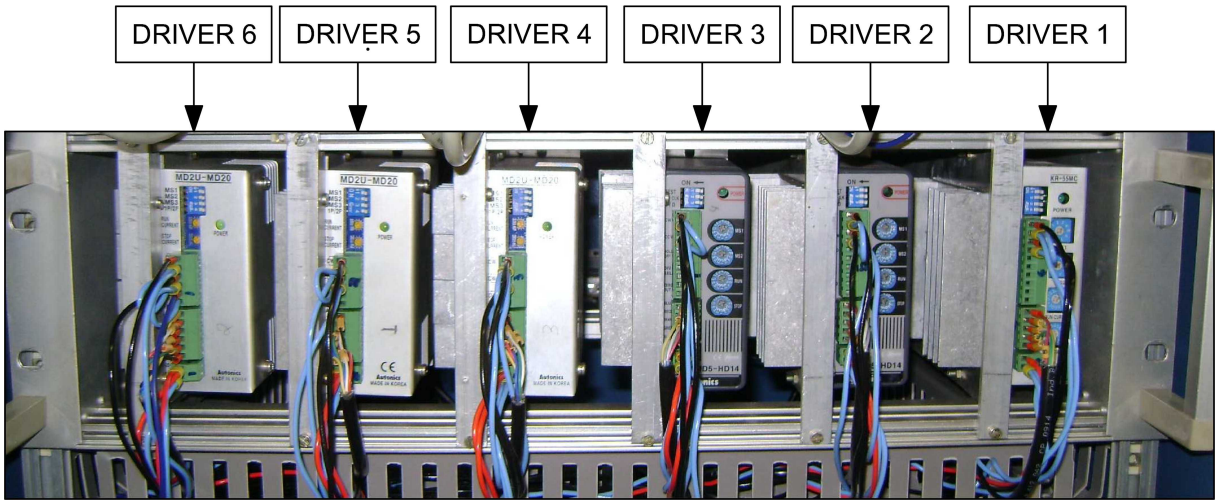

Figura 28: *Drivers* de potência.

Os modelos dos *drivers* são M2DU-MD20, MD5-HD14 e KR-55MC e suas principais especificações técnicas encontram-se dispostas na Tabela 5.

Tabela 5: Dados técnicos dos *drivers* de potência

Parâmetro	MD2U-MD20	MD5-HD14	KR-55MC
Numeração Atribuída	4 , 5 , 6	2 , 3	1
Alimentação	24-35 V_{DC} @ 3 A	24-35 V_{DC}@ 3 A	24-35 V_{DC} @ 3 A
Corrente Máx. de Acionamento	2,0 A/fase	1,4 A/fase	1,4 A/fase
Método de Acionamento	Unip., Microp.	Bip., Microp.	Bip., Microp.
Resolução Máx.	20 div.	250 div.	80 div.
N° de fases	2	5	5
Tensão de Pulso	[H] 4-8 V_{DC}	[H] 4-8 V_{DC}	[H] 4-8 V_{DC}
	[L] 0-0,5 V_{DC}	[L] 0-0,5 V_{DC}	[L] 0-0,5 V_{DC}
Frequência Máx. de Pulso	40 *kpps*	500 *kpps*	500 *kpps*

Fonte: Adaptado de Bicudo, Turesso e Haluc (2010).

4.2.1.4 Sistema de Transmissão

Até o dado momento, este trabalho limitou-se a descrever as características físicas e a modelagem matemática do robô separadamente, sem levar em consideração a relação existente entre elas. Também é importante recordar que a alocação de *frames* realizada no Capítulo 3 é executada por um método que admite certa arbitrariedade e portanto, justifica-se esclarecer a relação entre os movimentos de junta e os movimentos angulares dos motores do robô Handler.

Primeiramente, destaca-se que o movimento angular dos eixos dos motores não é o mesmo que o das juntas, sem exceção. Um sistema de transmissão composto por engrenagens e correias contribui para a definição da maneira com que todos os movimentos do manipulador se relacionam , inclusive suas características dinâmicas. A Tabela 6 relaciona os principais aspectos do sistema de transmissão do robô Handler.

Tabela 6: Sistema de transmissão

Junta	Meio de Transmissão	Motores Relacionados	Redução
1	Correia	$M1$	$3,011$
2	Correia e Engrenagens	$M2$	$18,587$
3	Correias e Engrenagens	$M3$	$8,840$
4	Correias e Engrenagens	$M4$, $M5$ (sentido contrário)	$3,015$
5	Correias e Engrenagens	$M4$, $M5$ (mesmo sentido)	$12,050$

Levando-se em consideração a disposição dos motores percebe-se que, para os movimentos do punho e abertura da garra, o acionamento independente dos motores não atuam independentemente no movimento de uma junta. De fato, o comando das juntas 4, 5 e a abertura da garra são controlados pelo acionamento conjunto dos motores

4, 5 e 6, diferentemente do que ocorre para as 3 primeiras juntas, onde existe esta independência. A figura 29 mostra a disposição dos motores do punho e os eixos de rotação alocados.

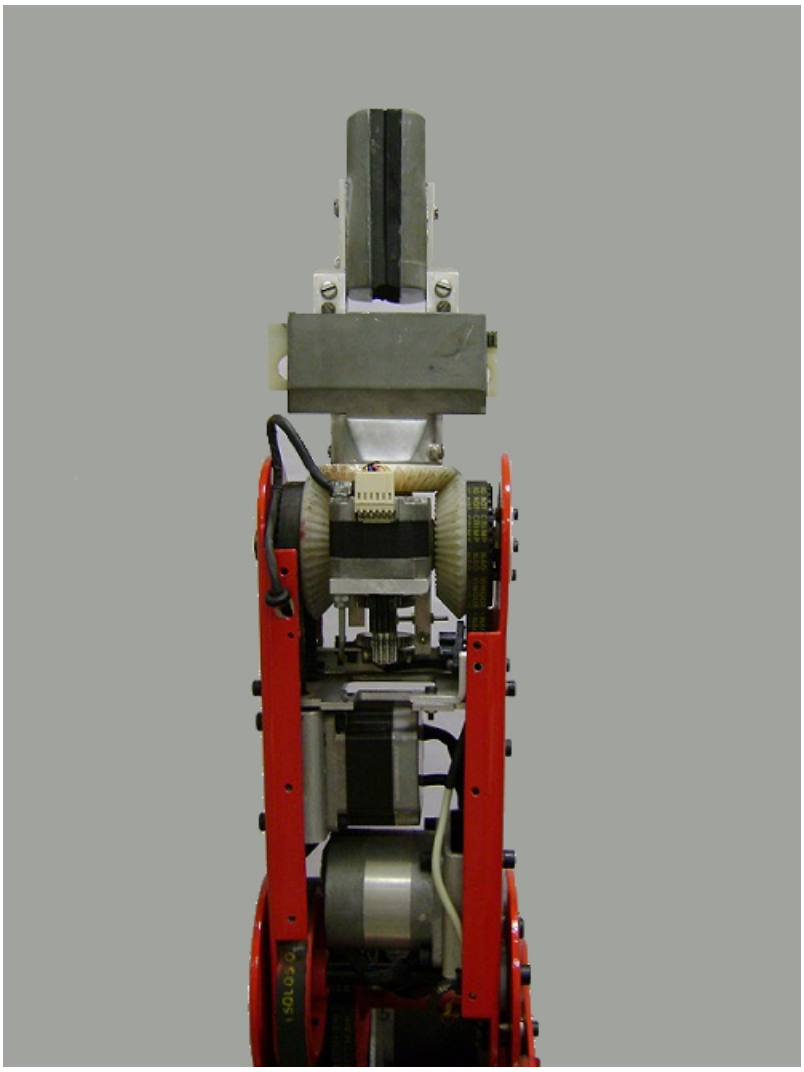

Figura 29: Fotografia do punho

Sendo np_i o número de pulsos entregues ao motor i, pa_{ij} a relação entre pulsos entregues ao motor i e movimento angular executado na junta j do manipulador e pl_6 a relação entre pulsos e movimento linear de abertura da garra, tem-se que:

$$q_1 = \frac{np_1}{pa_{11} \cdot pa_{11}} \tag{4.1}$$

$$q_2 = \frac{np_2}{pa_{22} \cdot pa_{22}} \tag{4.2}$$

$$q_3 = \frac{np_3}{pa_{33} \cdot pa_{33}} \tag{4.3}$$

$$q_4 = \frac{np_5 \cdot pa_{45} - np_4 \cdot pa_{55}}{pa_{55} \cdot pa_{44} + pa_{54} \cdot pa_{45}} \tag{4.4}$$

$$q_5 = -\frac{np_4 \cdot pa_{54} + np_5 \cdot pa_{44}}{pa_{55} \cdot pa_{44} + pa_{54} \cdot pa_{45}} \tag{4.5}$$

$$abertura_{garra} = \frac{np_6 - q_5 \cdot pl_6}{pa_{65}} \tag{4.6}$$

Por fim, a Tabela 7 resume a inter-relação entre os sistemas de atuação e transmissão.

Tabela 7: Relação sistema de atuação - transmissão

Driver	Motor	Passos	Resolução do Driver	Ângulo por pulso	Atua
1	1	500	2	0,36 °	Junta 1
2	2	500	1	0,72 °	Junta 2
3	3	500	1	0,72 °	Junta 3
4	4	200	4	0,45 °	Juntas 4 e 5 e Garra
5	5	200	4	0,45 °	Juntas 4 e 5 e Garra
6	6	200	1	1,80 °	Garra

4.2.2 SENSORES

O sistema de controle do robô Handler foi concebido em malha aberta, utilizando apenas chaves ópticas para reconhecer limites na rotação de juntas, oriundas da restauração prévia detalhada em Bicudo, Turesso e Haluc (2010) e escolhidas pela simplicidade e baixo custo. São utilizados seis sensores no manipulador; distribuídos conforme segue:

- Sensor 1 e Sensor 2: reconhecem os limites da junta 4;
- Sensor 3: reconhece os limites da junta 3;
- Sensor 4: reconhece os limites da junta 2;
- Sensor 5 e Sensor 6: reconhecem os limites da junta 1.

A alimentação e transmissão dos sinais dos sensores se dá via um cabo DB25 conectado em uma placa de interface com a unidade de controle. A principal função desta placa é disponibilizar tensão adequada à alimentação dos sensores e condicionar seus sinais a níveis compatíveis com a unidade de controle.

A princípio a placa de interface foi adaptada e utilizada durante a fase de familiarização com os sensores. Contudo, percebeu-se que a placa não era adequada, pois não havia nenhum tipo de proteção em caso de ligações errôneas.

Para contornar o problema, projetou-se outra placa de interface. A nova placa tem as funcionalidades da anterior com o acréscimo de componentes de proteção e circuitos de disparo para garantir comportamento digital na saída dos sensores. A Tabela 8 lista todos os componentes utilizados na confecção do novo hardware de interface dos sensores.

Tabela 8: Lista de componentes da placa dos sensores

Componente	Função	Quantidade
Diodo UF4007	Proteção	10
Resistor 22 $k\Omega$	Limitar a corrente	10
Resistor 270 Ω	Limitar a corrente	10
CI 74HC14	Disparo	2
CI LM7805	Regular tensão em 5 V	2
Capacitor 1 uF	Estabilizar a tensão de alimentação da placa	2
Capacitor 47 uF	Estabilizar a tensão de alimentação dos sensores	2
Conector macho 3 vias	Prevenir ligações erradas	10
Conector PCI 2 vias	Conexão entre placa e unidade de controle	10
Estanho 1 mm	Soldagem dos componentes	1 m
Placa de fenolite universal	Conexão entre todos os componentes	1

As Figuras 30 e 31 apresentam, respectivamente, a foto do resultado final da placa e o esquema seguido para elaboração da mesma.

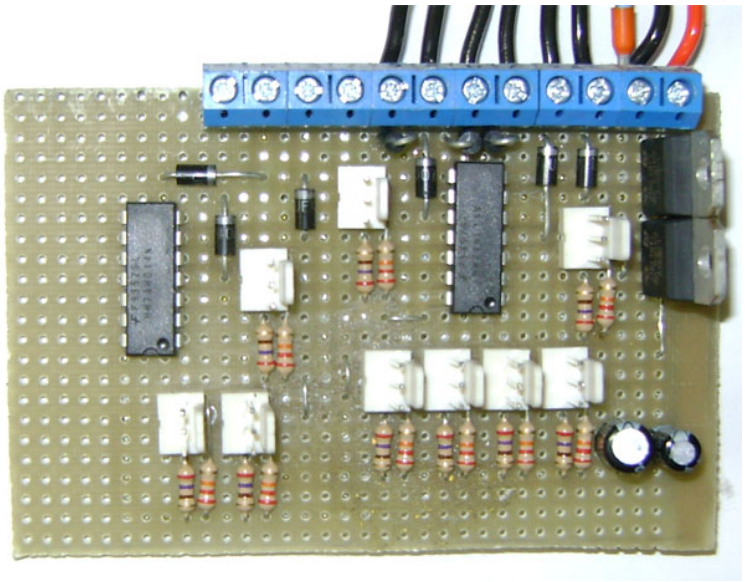

Figura 30: Foto da placa dos sensores

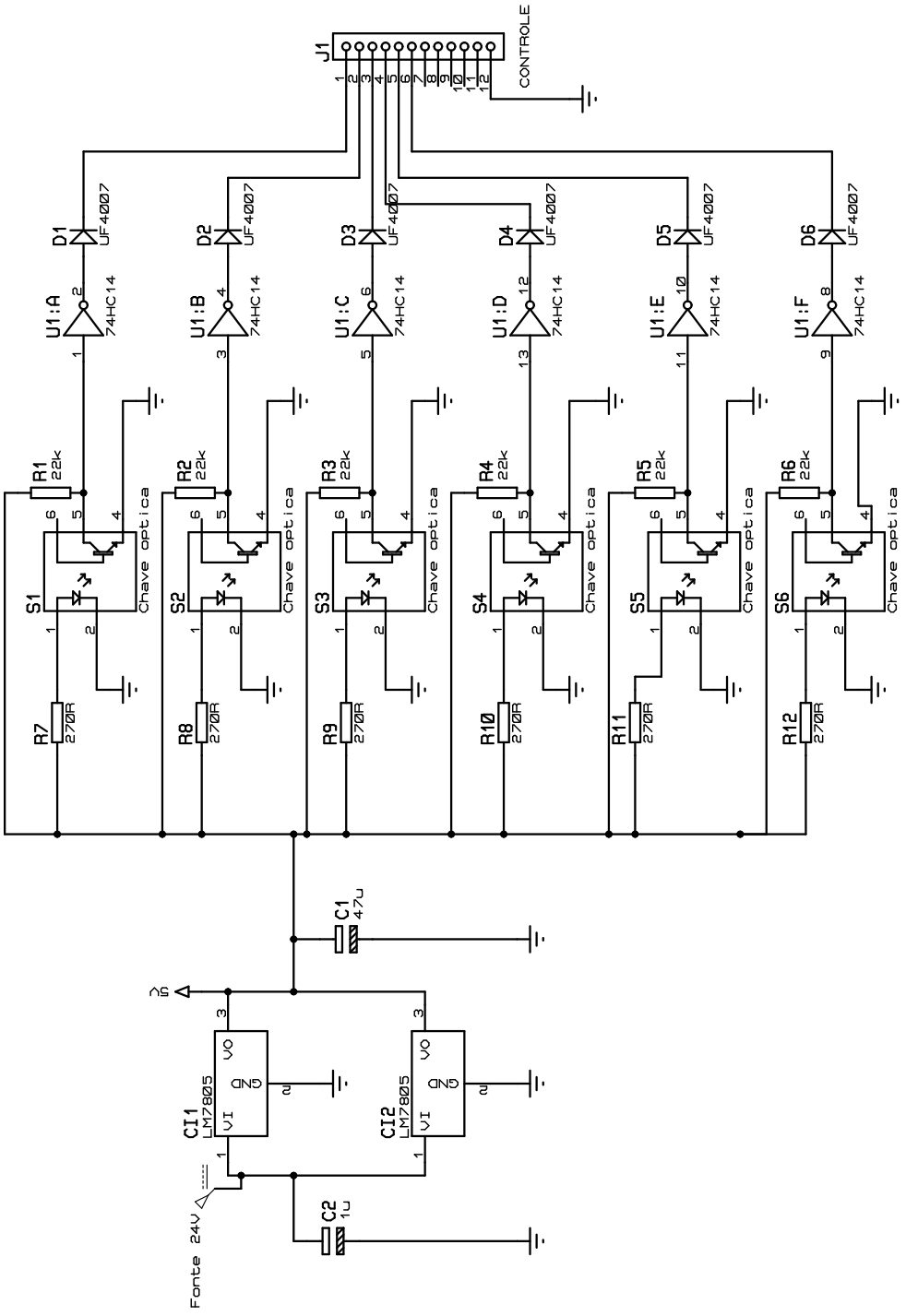

Figura 31: Esquemático da placa dos sensores

4.2.3 CONTROLE

4.2.3.1 Placa NI PCI-6601

A implementação do controle se faz por meio da placa NI PCI-6601.

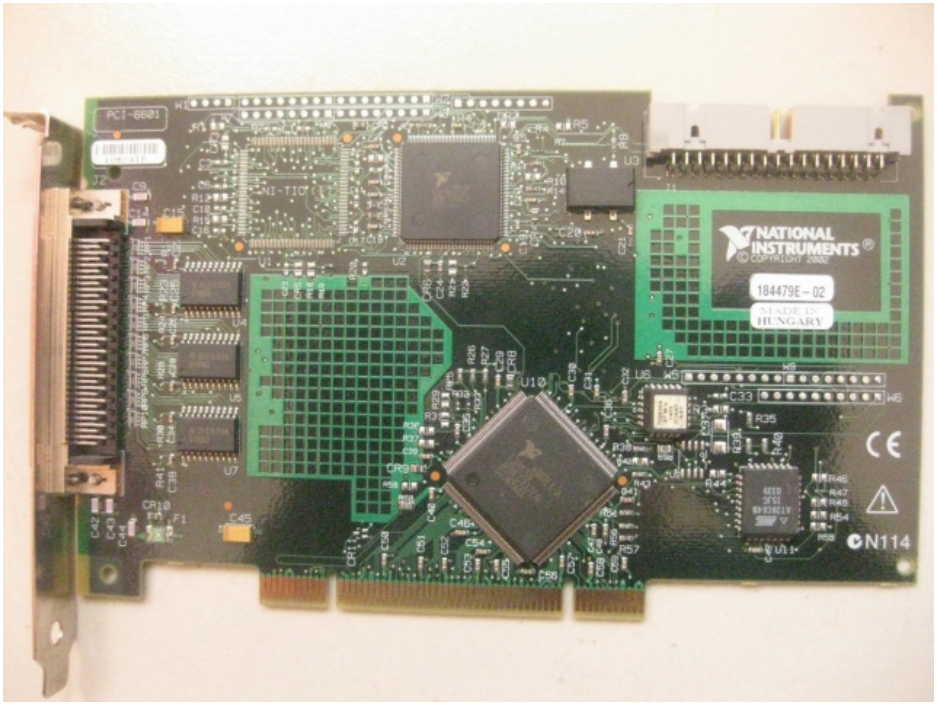

Figura 32: Placa NI PCI-6601

A Tabela 9 compara as principais características e funcionalidades pertinentes ao projeto.

Tabela 9: Características da NI PCI-6601

Característica	NI PCI-6601
Possuir processador digital de sinais	Não
Controle de trajetória em hardware	Não
Programação de alto nível	Não
Possibilidade de controlar seis motores de passo	Sim

A NI PCI-6601 funciona como um dispositivo de medição e geração de sinais digitais dedicado à aquisição de dados. Dentre seu leque de funcionalidades, destacam-se para o projeto os 4 contadores e 32 portas digitais de E/S. Portas digitais E/S podem ser utilizadas para leitura ou escrita de sinais de propósito geral; contadores são dedicados para tarefas que exigem temporização mais precisa na medição de frequência, contagem de eventos, medição de intervalo, geração de pulsos e PWM (*Pulse Width Modulation* - Modulação por Largura de Pulso). Outra opção consistem em se utilizar as portas de E/S em conjunto com contadores que geram uma base de tempo fixa, de forma a atingir frequências mais altas e estáveis.

Com o auxilio de um osciloscópio digital, verificou-se que as frequências máximas de saída dos contadores e portas E/S são respectivamente 1 MHz e 1 kHz.

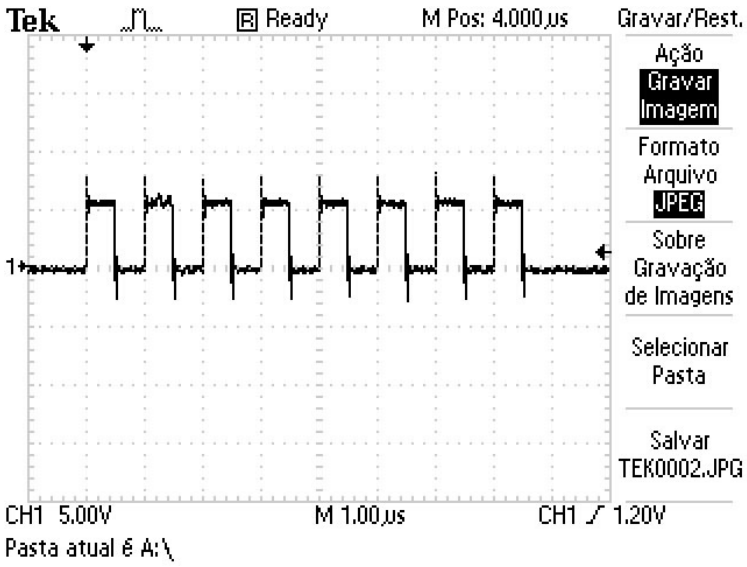

Figura 33: Frequência máxima de saída dos contadores

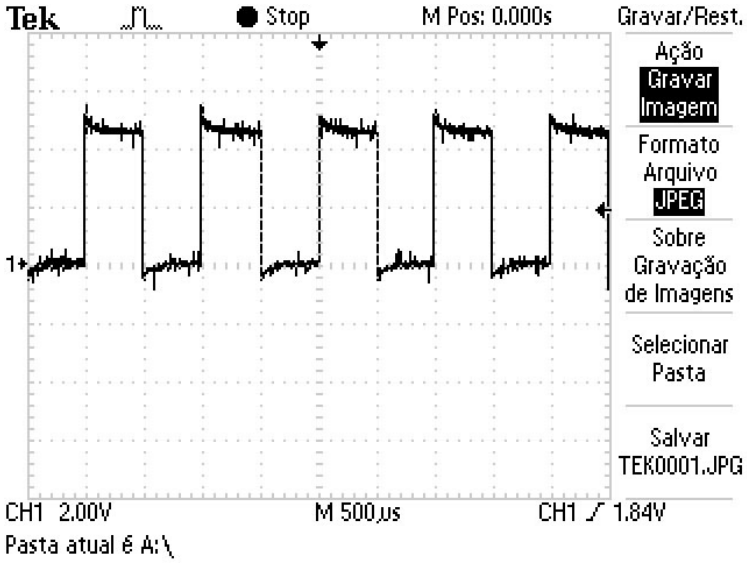

Figura 34: Frequência máxima de saída das portas E/S

A Figura 35 expõe o sistema de controle implementado no robô Handler.

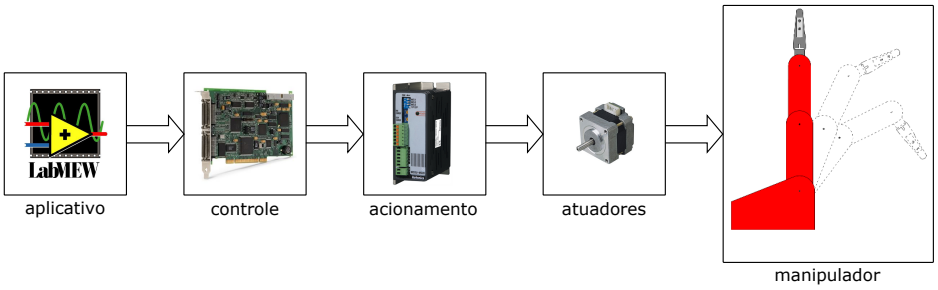

Figura 35: Sistema de controle a ser implementado

Há apenas quatro contadores disponíveis na placa, não suficientes para o acionamento individual de cada motor de passo. Além disso, tal escolha exigiria dois contadores por motor, um fornecendo a base de tempo e outro para desligá-lo após uma sequência pré-definida de pulsos. Desta forma, utilizaram-se as portas de E/S em conjunto com um contador apenas, fornecendo uma base de tempo comum para controle dos motores de passo. Esta combinação, embora mais lenta, é suficiente para implementá-lo com uma velocidade que atende às expectativas do projeto.

A Tabela 10 exibe a maneira com que as portas de E/S da placa foram implementadas.

Tabela 10: Portas E/S utilizadas

Pino	Nome	Configuração	Função
60	E/S 21	Saída	Pulsos para o motor 1
61	E/S 22	Saída	Pulsos para o motor 2
28	E/S 23	Saída	Pulsos para o motor 3
29	E/S 24	Saída	Pulsos para o motor 4
63	E/S 25	Saída	Pulsos para o motor 5
64	E/S 26	Saída	Pulsos para o motor 6
23	E/S 16	Saída	Sentido do motor 1
22	E/S 15	Saída	Sentido do motor 2
21	E/S 14	Saída	Sentido do motor 3
54	E/S 11	Saída	Sentido do motor 4
53	E/S 12	Saída	Sentido do motor 5
52	E/S 13	Saída	Sentido do motor 6
48	E/S 6	Entrada	Sensor fim de curso 1
47	E/S 5	Entrada	Sensor fim de curso 2
13	E/S 4	Entrada	Sensor fim de curso 3
12	E/S 3	Entrada	Sensor fim de curso 4
45	E/S 2	Entrada	Sensor fim de curso 5
44	E/S 1	Entrada	Sensor fim de curso 6
36	D GND	Entrada	Referência

4.2.3.2 Computador de Tempo Real

Um sistema operacional convencional, de propósito geral, não é adequado para processos de tempo crítico como acionamento de motores de passo pois, a qualquer momento, pode ocorrer atraso ou mesmo suspensão de um programa em execução. Em oposição a este tipo de sistema operacional, os SOTR garantem a execução de tarefas em tempos bem definidos, ou seja, deterministicamente (NATIONAL INSTRUMENTS, 2010c). Para a execução de movimentos do robô Handler, principalmente de trajetórias,

exige-se que os programas que gerenciam o sistema de atuação sejam executados com elevado sincronismo e alto grau de confiabilidade. Para tanto, a placa NI PCI-6601 está instalada em um computador rodando um SOTR da NI, o LabVIEW® RT.

A Figura 36 expõem de que maneira o computador de tempo real está arranjado no interior do armário de comandos do robô Handler.

Figura 36: Computador RT no armário de comandos

4.3 ESPAÇO DE TRABALHO

Para um esboço do espaço de trabalho do robô Handler, efetuaram-se medições de comprimento dos elos e limites de deslocamento angular de cada junta. Já que se trata de uma primeira aproximação com o objetivo de elucidar a proporcionalidade entre elementos da estrutura, não houve um comprometimento com precisão. A Tabela 11 relaciona as medições de comprimento de elo e limites de movimentação das juntas.

Tabela 11: Comprimentos de elo e limites de juntas

	Elo	Junta
1	165 mm	$-166,5\ ^\circ/+166,5\ ^\circ$
2	240 mm	$-41\ ^\circ/+37\ ^\circ$
3	210 mm	$-91\ ^\circ/+108\ ^\circ$
4	0 mm	$-79\ ^\circ/+80\ ^\circ$
5	205 mm	$\pm 360\ ^\circ$

Em seguida, desenhou-se por meio do *software* AutoCAD® [1] o primeiro elo nas posições q_{1min} e q_{1max}. Para cada uma destas posições, projetou-se o segundo elo nas posições q_{2min} e q_{2max}. Seguiu-se analogamente até que o elo do efetuador fosse representado em q_{5min} e q_{5max} para cada possibilidade. Por fim, traçou-se o lugar geométrico dos pontos alcançáveis, conforme a Figura 37.

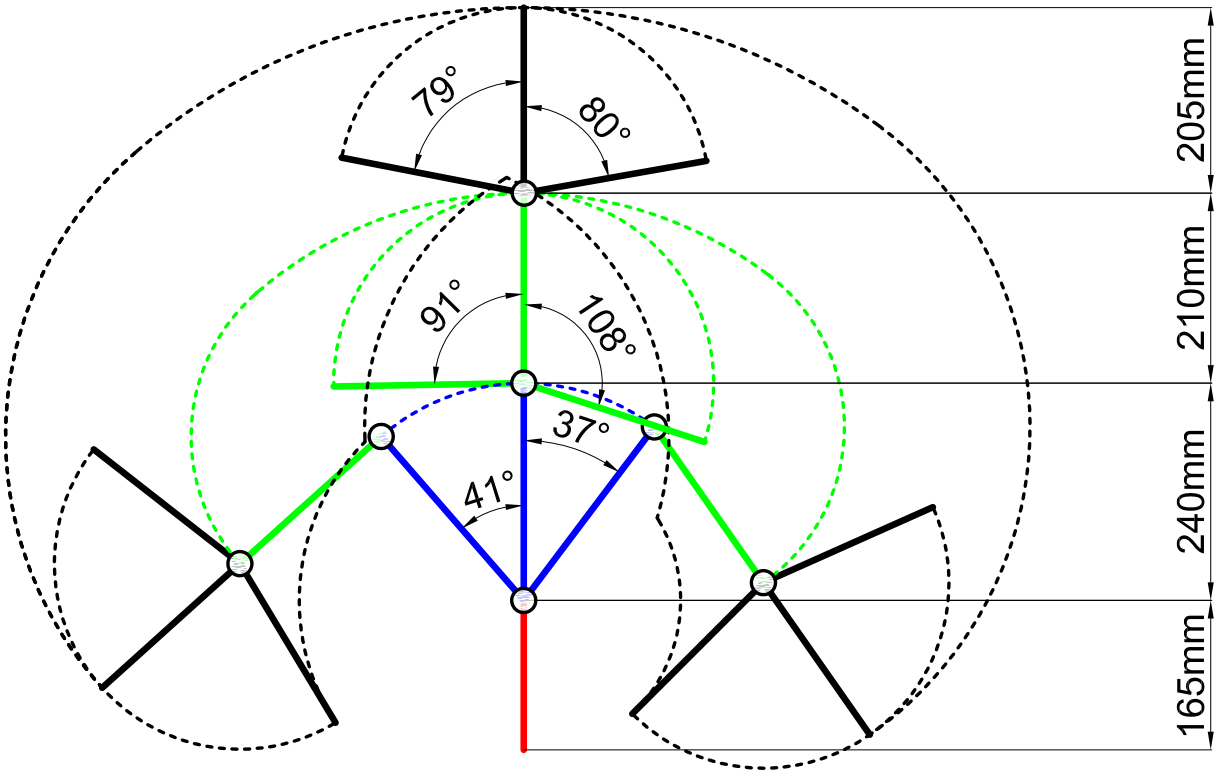

Figura 37: Projeção dos limites de junta

[1] AutoCAD é marca registrada da Autodesk Inc.

As Figura 38 apresenta as vistas superior e lateral do espaço de trabalho.

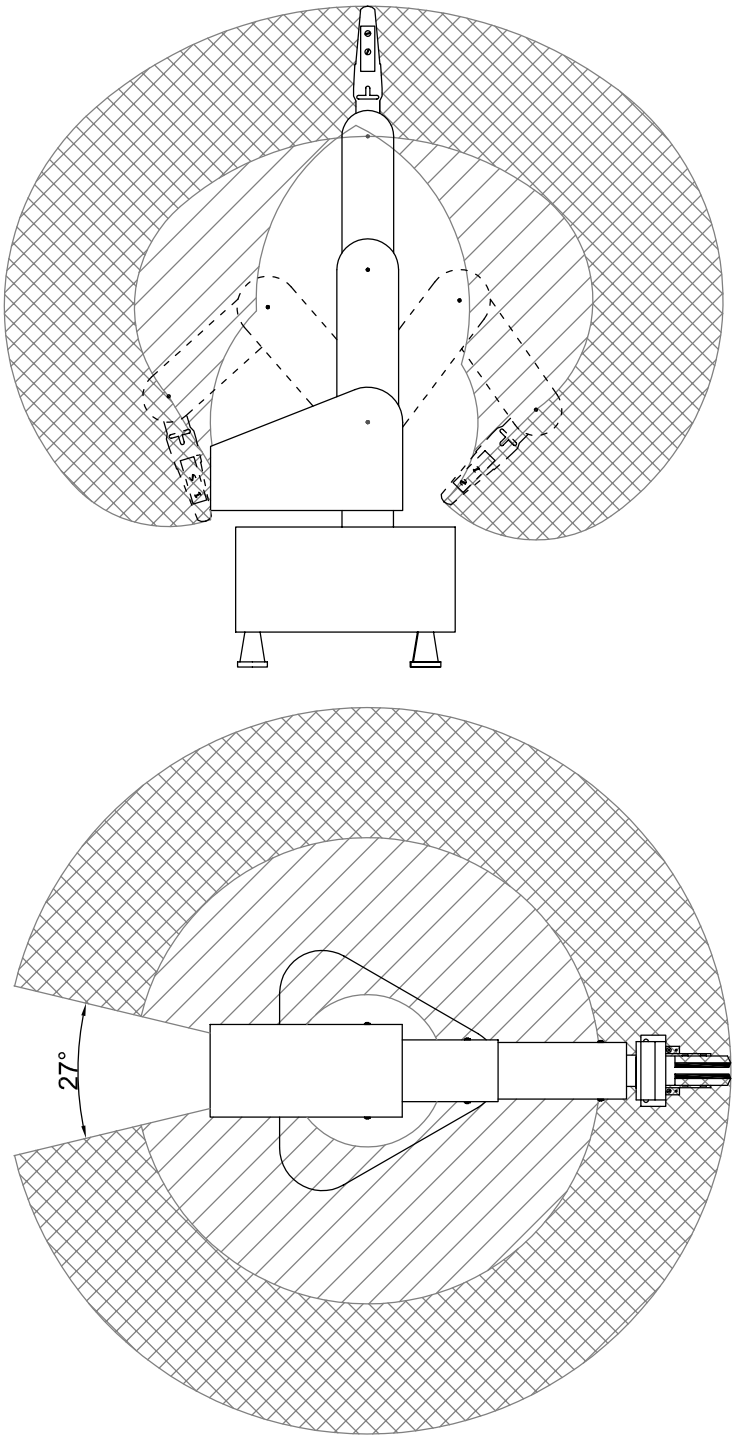

Figura 38: Espaço de trabalho do robô

A região com hachuras simples representa o espaço de trabalho alcançado por solução única e a região com hachuras duplas o espaço de trabalho alcançado por múltiplas soluções. Observa-se na vista lateral que não existe simetria do espaço de

trabalho. De fato, as juntas 2, 3 e 4 do robô possuem diferentes limites angulares para cada sentido de rotação. Uma importante consideração a ser feita é a existência de pontos do espaço de trabalho que interceptam a estrutura do robô. Tal análise serve de alerta para certos cuidados que deverão ser tomados durante o projeto do *software*, pois a negligência deste tipo de consideração certamente acarretará problemas funcionais.

Por fim, gerou-se um modelo tridimensional do espaço de trabalho para maior clareza, conforme a Figura 39.

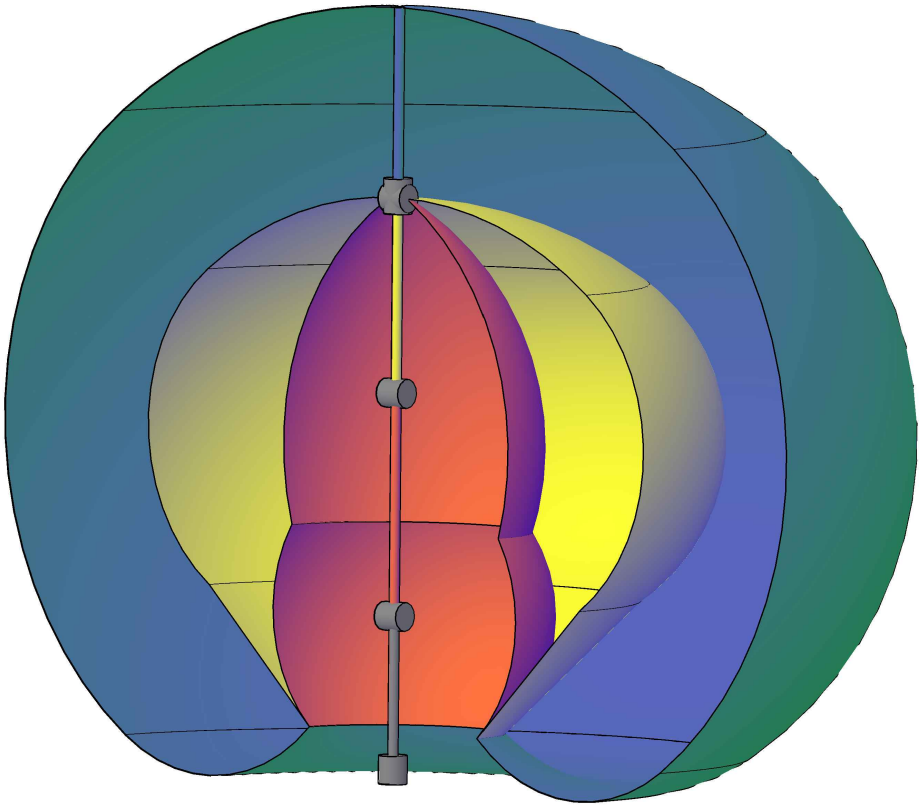

Figura 39: Modelo tridimensional do espaço de trabalho

A Figura 39 não exibe o espaço de trabalho completo do robô, mas uma seção de 180° com o intuito de delimitar os espaços de trabalho alcançável e destro abordados no Capítulo 2. O volume compreendido entre a casca vermelha e a azul é o espaço de trabalho alcançável e o volume descrito entre a casca amarela e a azul é o espaço de trabalho destro. No volume externo à casca azul e interior à casca vermelha localizam-se pontos que o robô é impossibilitado de alcançar.

5 *SOFTWARE* DO ROBÔ HANDLER

Este capítulo tem o objetivo de apresentar o desenvolvimento do aplicativo em LabVIEW® que possibilitará o controle do robô Handler por meio de uma interface gráfica.

5.1 VISÃO GERAL DO SISTEMA ANTERIOR

O trabalho de Bicudo, Turesso e Haluc (2010) implementou o controle do robô com a linguagem de programação gráfica diagrama *ladder*, que é padronizada para programação de CLPs e faz uma analogia com esquemas elétricos de bobinas e contatos para representar funções lógicas.

O fluxo de operação do sistema proposto naquele trabalho começava com um posicionamento adequado para o robô, a cargo do usuário. Então, acionava-se um conjunto de chaves seletoras que determinavam o modo de operação - ciclo manual, ciclo automático passo a passo ou ciclo automático contínuo. Podia-se ainda alterar a velocidade de execução do movimento e gravar posturas.

Uma botoeira de partida forçava o robô a realizar um movimento de busca de posição inicial para cada eixo, seguido de um estado de espera. Se o modo manual estivesse selecionado, um controle de videogame adaptado permitia que o usuário acionasse cada junta separadamente e gravasse até dez posturas para o manipulador. O *software* monitorava constantemente os sensores de fim de curso para evitar colisões.

O modo automático permitia a reprodução das posturas memorizadas, apenas uma vez ou continuamente. O procedimento para desativar o robô consistia em desligar a chave seletora de modo e aguardar que o robô buscasse a postura inicial. Os motores eram então desenergizados automaticamente.

Naquele projeto, conforme relatado por Bicudo, Turesso e Haluc (2010), encontrou-se diversas dificuldades na implementação com CLPs, tanto pelo baixo nível de abstração

da linguagem quanto pela limitação de memória, que permitiu apenas dez posturas pré-programadas.

5.2 ENGENHARIA DO NOVO *SOFTWARE*

Os tópicos seguintes irão descrever os aspectos relevantes do novo *software* desenvolvido seguindo as etapas especificadas no Capítulo 2. Muito embora o andamento do projeto não tenha sido linear e sequencial como propõe o método cascata, acredita-se que a explicação ficará mais clara e estruturada desta forma.

5.2.1 ANÁLISE DE REQUISITOS DE SISTEMA

Os tópicos seguintes delimitam algumas especificações de projeto que norteiam o desenvolvimento do *software*. Nos requisitos de sistema aspectos globais como a interface entre *hardware* e *software* são abordados.

- O sistema é composto pelo robô, um PC mestre (preferencialmente *laptop*) e um armário de comando com a placa NI PCI-6601, os *drivers* e um PC escravo. O PC mestre roda a aplicação principal, próxima ao usuário, sobre o sistema operacional Windows® . O PC escravo roda o *software* de baixo nível que controla o robô, sobre o SOTR LabVIEW® RT. Os dois computadores se comunicam pelo protocolo Ethernet.

- A placa NI PCI-6601 aciona os motores de passo individualmente e lê o sinal de saída das chaves ópticas. Estas tarefas são coordenadas por um sistema de tempo real. Um dos contadores disponíveis na placa fornece uma base de tempo precisa que, devidamente manipulada, gera frequências independentes para acionamento de cada motor, com um perfil de velocidade em rampa ou constante. As frequências são calculadas de modo que o movimento dos motores começa e termina ao mesmo tempo, independentemente do número de passos necessários a cada um. As chaves são lidas a uma taxa tão rápida quanto um período do trem de pulsos que aciona os motores.

- O robô deve ser capaz de realizar trajetórias ponto a ponto com uma precisão suficiente para que desenhe, por exemplo, uma senóide em um plano.

5.2.2 ANÁLISE DE REQUISITOS DE *SOFTWARE*

Nos requisitos de *software*, aspectos como a funcionalidade do sistema, descrita mediante um estudo de uso hipotético do *software*, e a interface de usuário fornecem subsídios para uma análise estrutural posterior.

5.2.2.1 Funcionalidade

Com o armário de comando desenergizado, o usuário garante que o robô está em uma posição inicial padronizada chamada Park. Ela foi definida visando a um desligamento suave do robô, ou seja, ao cessar sua alimentação o braço não deve cair e colidir consigo mesmo. A Figura 40 apresenta o Handler nessa posição.

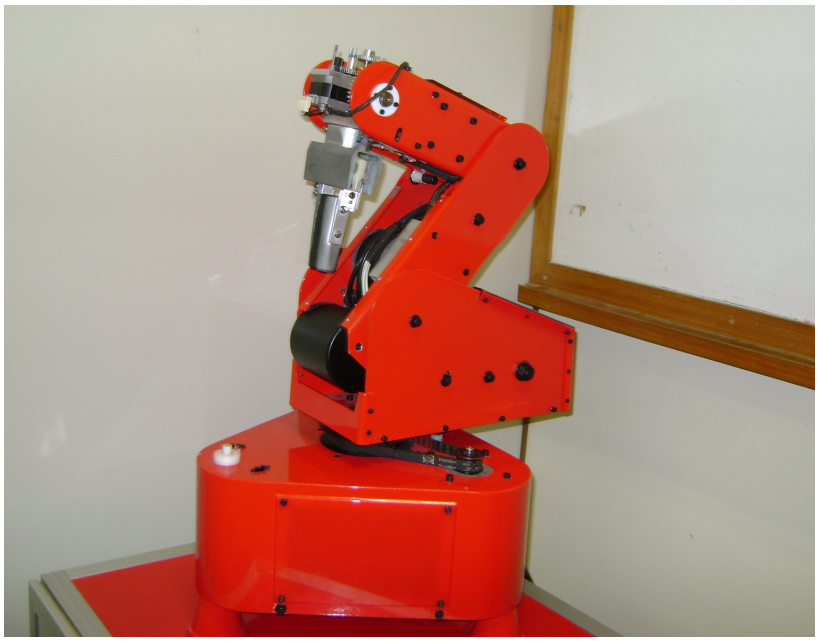

Figura 40: O robô handler na posição Park

Desarmando-se a botoeira de emergência, energiza-se o robô e o PC escravo. Imediatamente, o robô busca os fins de curso e move-se rumo à posição Home. A Figura 41 abaixo mostra o robô Handler nesta posição.

Figura 41: O robô handler na posição Home

Estabelecendo-se uma conexão entre os PCs, o aplicativo principal está pronto para executar. Dele, é possível comandar a qualquer momento: uma rotina Referência, para que o robô Handler encontre os fins de curso; Home, para que se mova para uma postura em que todos os ângulos de junta, com excessão da segunda, são nulos (tal medida contorna o problema de o robô não ser capaz de atingir a postura de referência adotada nos cálculos de cinemática); Park, para que se posicione de modo a evitar colisões e a ocupar menos espaço; e Demo, que executa movimentos de demonstração.

Há uma janela de ajuda, contendo instruções para o correto manuseio do *software* e uma janela com informações sobre este projeto.

Existem três modos para geração de caminhos:

- Geração por comando de junta. Neste modo, pode-se estipular valores de ângulo para cada junta individualmente, e uma porcentagem de abertura da garra. Ainda, há a opção de informar esta configuração em modo absoluto ou em modo relativo. O usuário pode, então, gravar e deletar quantas destas configurações ângulos de junta/abertura da garra/modo desejar, que serão exibidas em uma tabela.

- Geração por cinemática inversa. Neste modo, pode-se estipular uma postura (posição e orientação) desejada e uma porcentagem de abertura da garra, em

modo absoluto ou relativo. Como no modo anterior, pode-se gravar e deletar indefinidamente configurações postura/abertura da garra/modo, que serão também exibidas em tabela.

- Geração por controle manual. Neste modo, o usuário é capaz de comandar em tempo real todas as juntas e a garra.

Tanto no modo Geração por comando de junta quanto no modo Geração por cinemática inversa, o robô só executará a sequência memorizada se o usuário explicitamente solicitar o início da execução. Antes disto, é possível salvá-la em um arquivo .xls ou mesmo carregar um arquivo .xls com uma sequência previamente editada em planilha eletrônica.

Como artifício de retroalimentação, um modelo tridimensional constitui, simultaneamente à execução de qualquer movimento, uma representação virtual do robô. Os valores de postura e ângulos de junta podem ser monitorados em tempo real.

Há um seleção de modo simulado em que o aplicativo executa a sequência de configurações gravada e volta à postura atual em que o robô se encontra, sem acioná-lo. Tal ferramenta torna possível projetar cuidadosamente movimentos sem que o robô os execute.

5.2.2.2 Interface gráfica

Tentou-se atender à maior parte das recomendações de boa prática para o desenvolvimento de interfaces gráficas, extensamente comentadas por Blume (2007) e Ritter (2002). Os botões, abas, tabelas, controles deslizantes, controles numéricos, indicadores numéricos, quadros e caixas de seleção tipo *checkbox* e *radio* são exatamente os mesmos do sistema operacional que roda a aplicação. Como atualmente este sistema operacional é o Windows Seven® , a aplicação possui seu aspecto gráfico. Caso qualquer outro sistema operacional Windows seja escolhido, os elementos gráficos se adaptam a ele. Isto é vantajoso por flexibilizar o aplicativo e por ir de encontro a interfaces familiares a grande parte dos usuários de computador.

A GUI do aplicativo principal tem o aspecto apresentado na Figura 42. As outras abas de modo estão apresentadas na Figura 43.

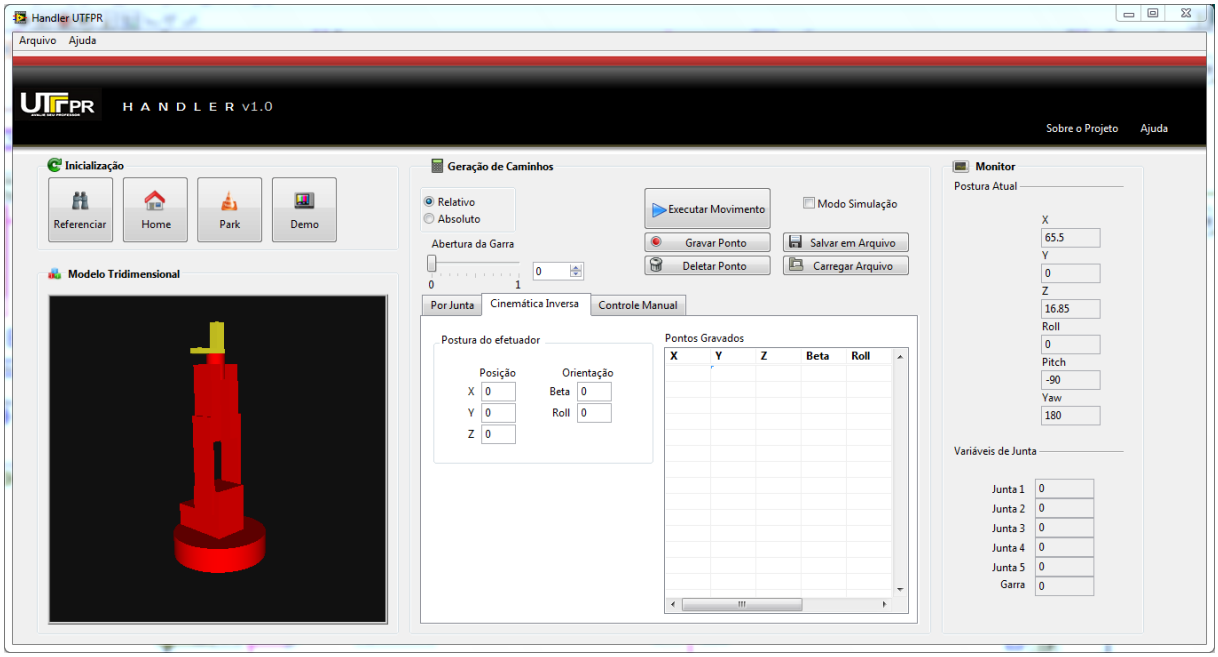

Figura 42: GUI do aplicativo principal

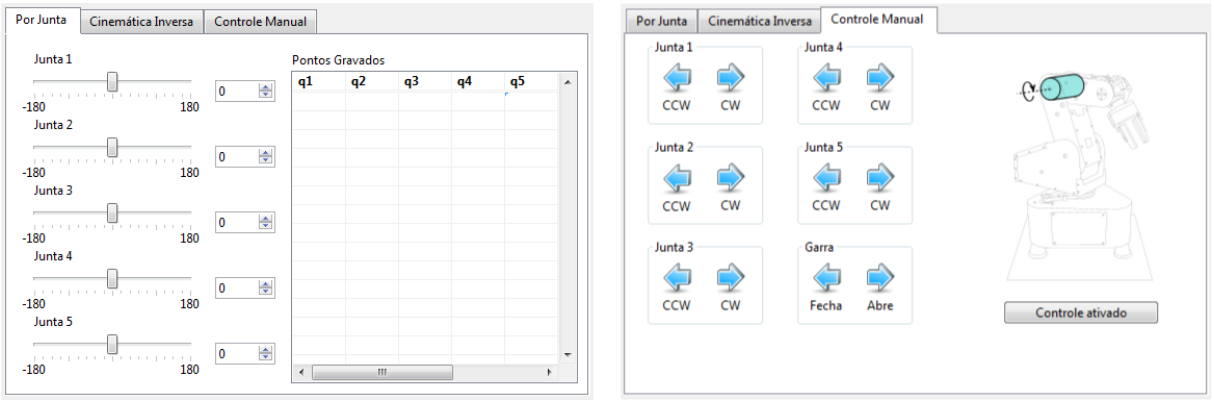

Figura 43: Abas restantes da GUI do aplicativo principal

A maior parte dos botões têm ícones para orientar rapidamente o usuário. O uso de abas permite reunir um grande conjunto de informações em um espaço reduzido; de fato, toda a interface é visível em apenas uma tela de um monitor com resolução igual ou superior a 1366x768 pixels.

Uma imagem na parte superior da tela faz mensão ao nome do projeto e contêm a logomarca da UTFPR. Mais ao lado direito, é possível abrir as caixas de diálogo Ajuda e Sobre o Projeto. Quadros com ícones agrupam controles de características similares, para segmentar logicamente a aplicação e facilitar a experiência do usuário. Há quatro quadros globais:

- Inicialização: contém os botões Referenciar, Home, Park e Demo.

- Geração de Caminhos: contêm as abas de modo de operação, seleção de abertura da garra, seleção de modo relativo e absoluto, seleção de modo Simulação e os botões Executar Movimento, Gravar Ponto, Deletar Ponto, Salvar Arquivo e Carregar Arquivo. Na aba Controle Manual, caso o cursor do *mouse* passe em cima de qualquer um dos botões de acionamento das juntas, uma figura animada ao lado direito exibe qual a junta correspondente no robô.
- Monitor: contém os indicadores de postura (posição e orientação RPY).
- Modelo Tridimensional: exibe o modelo tridimensional animado.

5.2.3 *DESIGN* ARQUITETURAL

Em consonância com o proposto no Capítulo 2, escolheu-se o padrão de projeto Máquina de Estados Finita (MEF). Há duas MEFs executando em paralelo: a principal (a partir de agora referenciada como MEF Principal), no PC mestre, comandando uma segunda (a partir de agora referenciada MEF RT), no PC escravo. A Figura 44 apresenta a máquina de estados completa para o *software*.

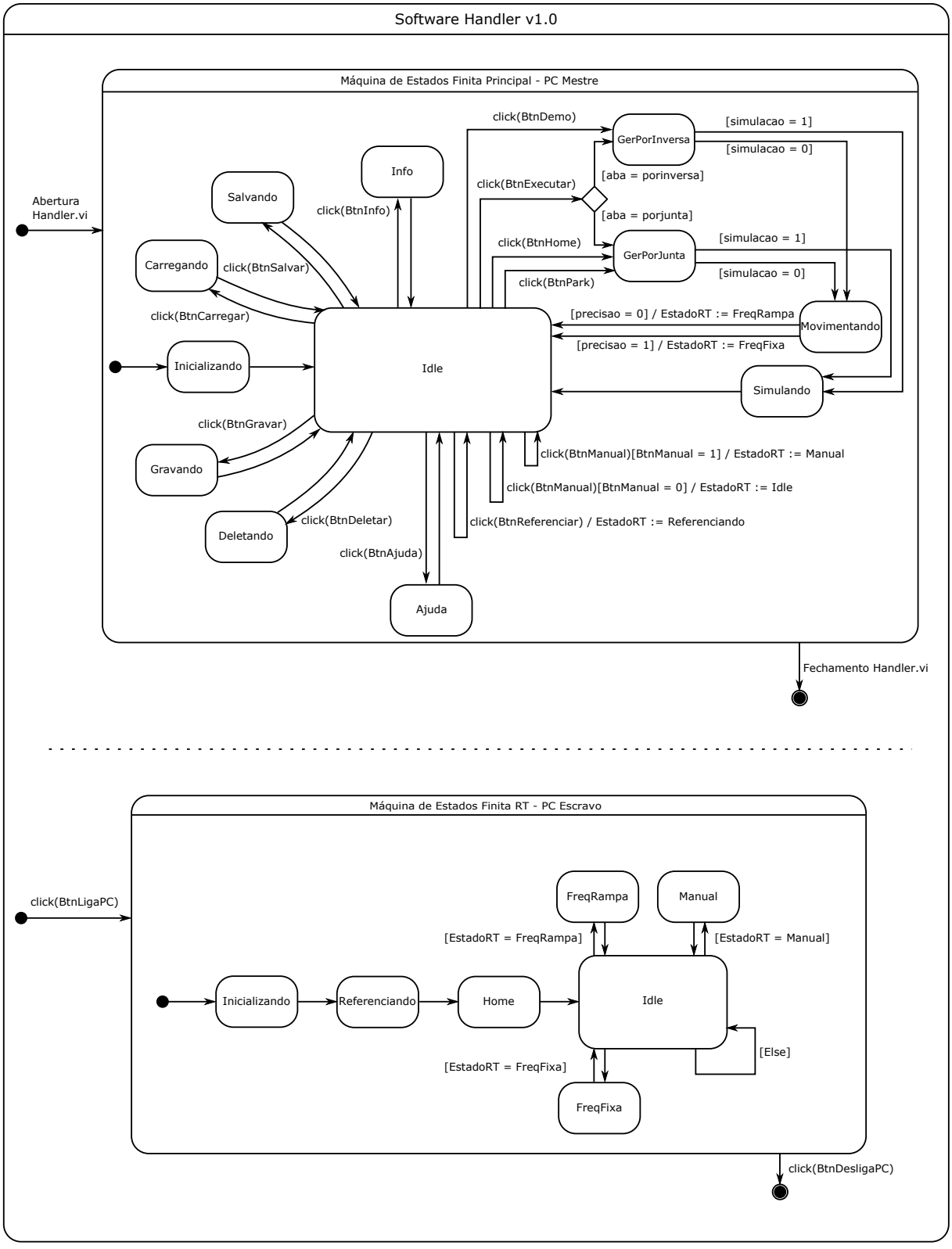

Figura 44: Máquina de estados finita do aplicativo

5.2.3.1 MEF Principal

A MEF principal é uma VI implementada em LabVIEW® utilizando-se de um padrão chamado Máquina de Estados por Filas Guiada por Eventos (do inglês *Event-Driven Queued State Machine*). Em tal arquitetura, cada estado pode enfileirar uma sequência de outros estados. A GUI é o próprio painel frontal desta VI.

O estado Inicializando é chamado apenas uma vez e inicializa as variáveis que são compartilhadas entre os estados.

O estado ocioso, chamado de Idle, espera por algum evento da interface de usuário comunicado pelo sistema operacional, como uma mudança em um valor de um botão e o passar do mouse por alguma região da tela. Enquanto não houver eventos o sistema fica em espera, poupando recursos para outras tarefas em paralelo. Como mostra a Figura 44, alguns eventos disparam um ou mais estados e outros eventos, citados abaixo, comandam a MEF RT:

- Mudança de valor no botão Ativa Controle Manual: este evento seta ou reseta a variável compartilhada que define o estado Manual da MEF RT.
- Mudança de valor no botão Referenciar: o próximo estado da MEF RT será Referenciando.
- Mudança de valor em qualquer botão da aba Controle Manual: envia o comando via variáveis compartilhadas para a MEF RT, que serão interpretados como ordem de movimento se ela estiver no estado Manual.

A programação guiada por eventos segue uma filosofia contrária ao chamado *polling*, em que se monitora constantemente o valor de uma variável até que mude, em um ciclo indefinido. O *polling* consome muito mais recursos computacionais, fato que guiou a escolha pelo monitoramento de eventos. Em LabVIEW® há uma estrutura chamada *Event Loop* com esta funcionalidade. A característica de haver uma espera por eventos da GUI apenas no estado ocioso implica que o usuário fica impossibilitado de interagir enquanto outro estado está ativo.

Os estados InfoProjeto e Ajuda exibem caixas de diálogo do tipo *popup* com informações, que podem ser fechadas ou minimizadas. O estado Gravando adiciona a configuração de variáveis de junta ou postura desejadas para o robô em uma fila de tamanho indefinido. Há uma fila independente para o modo Geração por comando de

junta e outra para o modo Geração por cinemática inversa. O estado Deletando remove uma configuração da fila. O estado Salvando grava uma fila em um arquivo .xls e o estado Carregando abre um arquivo .xls e extrai suas informações, colocando-as na fila.

O estado GerPorJunta retira todas as configurações da fila de Geração por comando de junta, contendo as variáveis de junta, modo e abertura da garra desejados. Este estado realiza uma série de cálculos e retorna variáveis de acionamento - o número de passos que cada motor deve executar (para atingir cada configuração), a frequência de acionamento de cada motor para que terminem a execução simultaneamente, o sentido de rotação dos motores, o vetor de variáveis de junta atual e o passo absoluto atual de cada motor.

O estado GerporInversa é análogo ao GerporJunta, mas toma como entrada a fila de configurações do modo Geração por cinemática inversa, com a postura desejada, o modo e a abertura da garra. Deste modo, é necessário que haja um cálculo de cinemática inversa para converter a postura em um vetor de variáveis de junta, que podem então ser processadas para retornar as variáveis como número de pulso, sentido e frequência para os motores.

O estado Simulando toma como entrada todas as variáveis de acionamento dos motores e decrementa o número de pulsos como se de fato estivessem sendo executados pelo *driver*. Retorna os passos absolutos atuais de cada motor para que o movimento seja projetado na tela pelo modelo tridimensional, mas sem alterar a postura em que o robô real se encontra. É um artifício que emula uma retroalimentação inexistente da posição dos motores de passo.

O estado Movimentando envia todas as variáveis de acionamento calculadas para a MEF RT e também simula uma retroalimentação, sincronizando a atualização do modelo tridimensional com o número de passos já executados pelo robô real.

A MEF Principal contêm VIs de processamento que efetuam cálculos, manipulação de arquivos, manipulação de erros, geração de gráficos, etc., e executa indefinidamente até que a aplicação seja fechada. Contudo, conforme especificado nos requisitos de *software*, há algumas funcionalidades que devem ser executadas continuamente, independentemente de qual estado está ativo, como a animação do modelo tridimensional e todos os indicadores do painel frontal. Portanto, estas rotinas executam em um *loop* paralelo atualizado a uma taxa de aproximadamente $60Hz$ (uma frequência que gera uma animação suficientemente confortável ao olho humano). Como a MEF produz dados muito mais rapidamente do que o *loop* de atualização da tela consegue absorver, optou-se por implementar um padrão produtor/consumidor entre os *loops*. A

MEF deposita em uma fila todos os dados relevantes para o painel frontal e o *loop* de atualização da tela os retira quando possível.

O modelo tridimensional, ampliado na Figura 45, foi confeccionado utilizando-se da paleta Graphics and Sound do LabVIEW® . Para tanto, cada parte do modelo - elos e juntas - foi modelada parametricamente e conectada de forma hierárquica, ou seja, uma parte $n+1$, filha de uma parte n, está acoplada mecanicamente a ela. Então, adiciona-se o conjunto a uma cena, que estará disponível a outras VIs em forma de referência. Um indicador desta referência gera uma tela com o modelo estático que, devidamente inserido em um *loop*, será animado. Pode-se ainda configurar a câmera, cor de fundo, iluminação e adiante. VIs de transformação espacial possibilitam que um vetor com ângulos de junta aplicados à referência do modelo atualizem sua postura em consonância com o efeito no robô real.

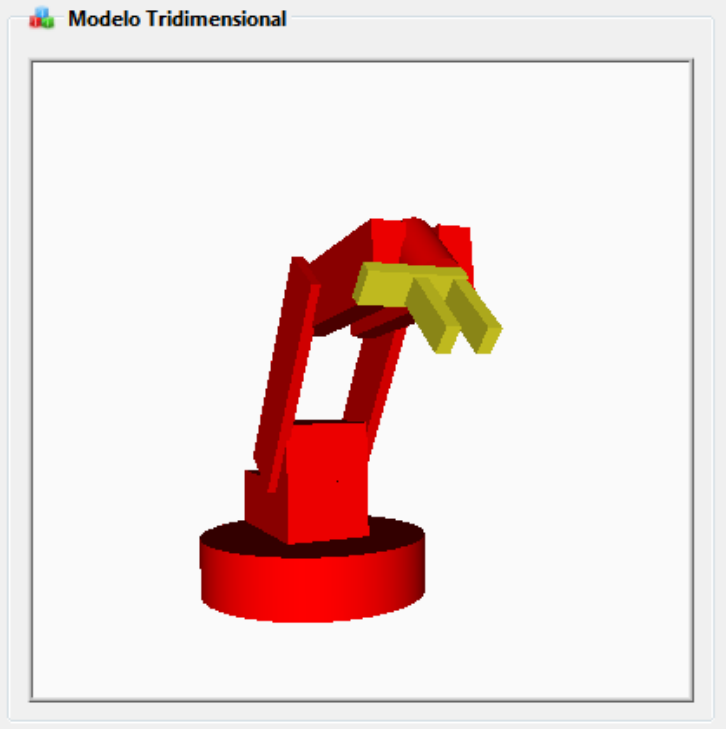

Figura 45: Aspecto do modelo tridimensional

O modelo 3D, já inteiramente modelado e conectado hierarquicamente, foi gravado em um arquivo .dat. Este arquivo é chamado na inicialização do *software* para a geração da cena. O modelo apresenta um aspecto visual simples pela crença de que não há necessidade e mesmo suporte (sem o uso de módulos especiais) para a geração de um modelo ricamente detalhado.

5.2.3.2 MEF RT

A MEF rodando no PC escravo (MEF RT) é uma VI implementada em LabVIEW® utilizando-se de outro padrão chamado Máquina de Estados Convencional (do inglês *Standard State Machine*), pois não há necessidade de sofisticações como estados enfileirados e monitoração de eventos. De fato, a MEF RT muda de estado apenas quando explicitamente comandado pela MEF Principal. A comunicação entre os PCs impossibilita que as VIs remotas se comuniquem pelo princípio orientado a fluxo de dados. Como as aplicações estão conectadas via Ethernet, há o desafio de compartilhar dados entre tarefas rodando em sistemas separados. Solucionou-se o problema com o recurso de variável compartilhada (*shared variable*), que implementa esta transferência de dados de maneira transparente ao programador.

A MEF RT contêm as VIs de baixo nível responsáveis pela leitura e escrita de dados na placa NI PCI-6601, que controla o robô. Assim, com o PC escravo energizado, a MEF RT roda indefinidamente, coletando todos os dados processados pela MEF principal e convertendo-os em comandos diretos ao robô.

O primeiro estado, Inicializando, movimenta todas as juntas do robô em um ângulo calibrado, pois sua postura Park logo após o sistema ter sido energizado pode ter ultrapassado o limite de fins de curso.

O estado Referenciando é chamado logo em seguida e realiza uma movimentação constante de todas as juntas até que se encontre todos os fins de curso.

O estado Home, na sequência, posiciona o robô com uma postura em que todos os ângulos de junta são zero, com excessão do segundo que é de 90 graus. Há então uma transição para o estado ocioso Idle.

O estado Idle executa em *polling* aguardando por um comando da MEF Principal, via variáveis compartilhadas, indicando que deve haver uma transição para um estado diferente.

O estado Manual é chamado toda vez que o usuário explicitamente o ativa, acionando o robô de acordo com comandos manuais, em frequência constante.

O estado FreqFixa é chamado quando existe uma fila de configurações desejadas para o robô em modo preciso, como a descrição de um caminho específico por vários pontos próximos. É o estado de acionamento para caminhos que exigem maior rapidez e precisão na transição entre pontos, como tarefas de desenho.

O estado FreqRampa é chamado quando existe uma fila de configurações desejadas para o robô em modo não preciso, ou seja, em caminhos ponto a ponto que podem ser executados com um perfil de velocidade em rampa, como o caso de tarefas de movimentação de materiais.

5.2.3.3 Camadas do *software*

A arquitetura descrita permite organizar o *software* em três camadas, consonantemente com o proposto no Capítulo 2. A Figura 46 ilustra as camadas e a Figura 47 fornece um detalhamento sobre as VIs que compõem cada camada.

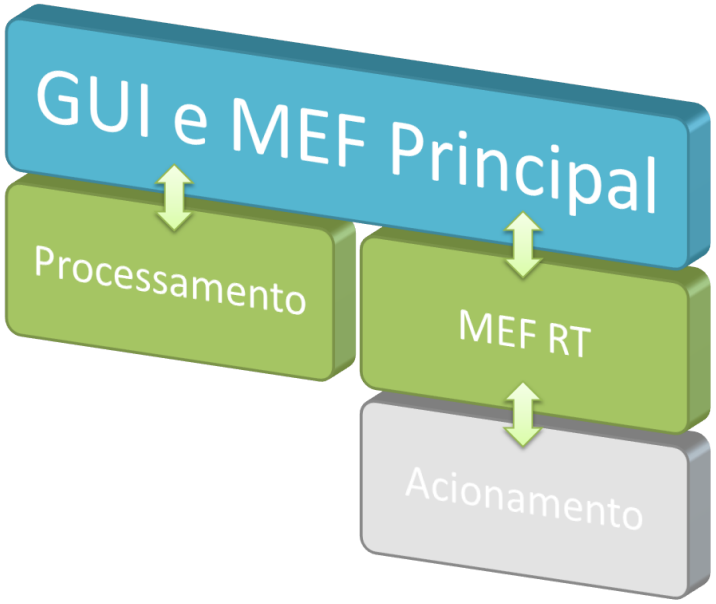

Figura 46: Estrutura do *software* em três camadas

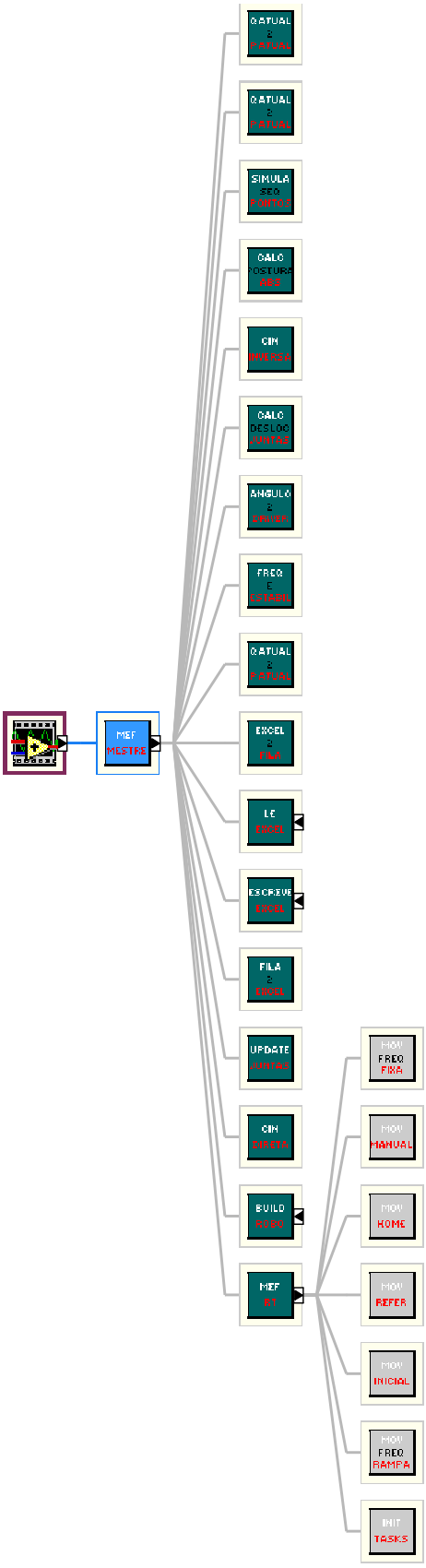

Figura 47: Hierarquia simplificada das VIs do projeto

A VI de camada 1 do *software* se chama **MEF Principal.vi** e contém a máquina de estados do PC Mestre e a GUI. As VIs de camada 2 são:

- **Vetor Q Atual para Passo Atual.vi**. Recebe um vetor de variáveis de junta atual do robô e retorna o passo absoluto atual de cada motor.
- **Passo Atual para Vetor Q Atual.vi**. Realiza a operação inversa da VI anterior.
- **Simula Movimento Frequência em Rampa.vi**. Recebe um vetor de *clusters* com o número de pulsos e parâmetros de perfil de velocidade em rampa para cada motor e simula o movimento, retornando por variáveis globais em tempo real o número de pulsos executados.
- **Calcula Postura Absoluta.vi**. Recebe um *cluster* com a postura do efetuador e porcentagem de abertura da garra do robô desejados em modo relativo ou absoluto e retorna a postura resultante em modo absoluto.
- **Calcula Deslocamento Juntas.vi**. Recebe um *cluster* de variáveis de junta e porcentagem de abertura da garra do robô desejados em modo relativo ou absoluto e retorna o deslocamento angular necessário.
- **Ângulo para Driver.vi**. Recebe um *cluster* com o deslocamento angular necessário para as juntas do robô e retorna o número de passos e sentido a serem enviados para o *driver* de cada um dos seis motores.
- **Perfil em Rampa.vi**. Recebe um *cluster* com o número de passos necessários para cada motor e retorna um perfil de velocidade em rampa.
- **Excel para Fila.vi**. Converte uma string 2-D proveniente do Excel em uma fila de posturas ou variáveis de junta a serem executadas pelo robô.
- **Fila para Excel.vi**.Recebe a fila com a postura ou vetor de variáveis de junta desejados para o robô Handler e retorna uma tabela de strings formatada para exibição ou gravação em planilha.
- **Lê Excel.vi**. Lê um arquivo Excel e retorna um array 2-D de strings com todos os valores contidos no arquivo.
- **Escreve Excel.vi**. Lê um array 2-D de strings e converte para um arquivo Excel.

- **Update Modelo 3D.vi**. Recebe um vetor com deslocamentos angulares para cada junta e atualiza o modelo tridimensional do robô com as transformações espaciais necessárias.
- **Constrói Modelo 3D.vi**. Recebe um *cluster* com as partes e hierarquia do modelo tridimensional do robô e retorna uma referência a uma cena 3D.
- **Cinemática Direta.vi**. Recebe as variáveis de junta desejadas para o robô e computa a postura (posição e orientação) resultante do efetuador.
- **Cinemática Inversa.vi**. Recebe a postura desejada para o efetuador do robô e retorna as variáveis de junta correspondentes para tal.
- **MEF RT.vi**. Contém a máquina de estados do PC Escravo.

As VIs de camada 3 são:

- **Inicializa Tasks.vi**. Inicializa todas as tarefas de I/O e retorna um cluster com as referências.
- **Movimento Frequência Fixa.vi**. Recebe um *cluster* de tarefas de I/O e gera trens de pulso em tempo real com frequência fixa para cada motor, no sentido especificado e sensível a sensores de fim de curso.
- **Movimento Frequência Rampa.vi**. Como a VI anterior, mas o acionamento é feito com um perfil de velocidade em rampa.
- **Movimento Inicial.vi**. Movimenta sutilmente o robô para garantir que sua postura inicial não ultrapassou nenhum fim de curso.
- **Movimento Referência.vi**. Realiza um movimento de procura pelos fins de curso do robô.
- **Movimento Home.vi**. Após a procura por fins de curso, posiciona o robô em uma postura Home calibrada.
- **Movimento Manual.vi**. Recebe comandos de junta manuais por variáveis compartilhadas e aciona os motores do robô em frequência fixa.

6 ENSAIOS, CALIBRAÇÕES E RESULTADOS

Alguns ensaios realizados neste capítulo foram executadas em paralelo com a programação do *software* e tiveram como objetivo realizar o levantamento de dados cruciais, referentes à operação do Handler. Estes dados possibilitaram o funcionamento do robô de acordo com o desejado e que, por meio de um processo de calibração, garantisse-se certa precisão em seus movimentos. Ao fim deste capítulo encontram-se alguns ensaios executados após as ações descritas no Capítulo 5 onde se aquirem os resultados da implementação final do *software*.

6.1 ATUAÇÃO DE FIM DE CURSO

Antes de qualquer tipo de ensaio, é necessário garantir que os fins de curso estejam sendo atuados corretamente caso o robô passe do seu limite de espaço de trabalho, podendo ocasionar uma colisão consigo mesmo ou atingir uma postura da qual não consiguirá sair devido à limitação de torque dos seus motores.

A lógica de proteção implementada consiste em cortar a geração de pulsos uma vez que os sensores de fim de curso sejam sensibilizados. A Figura 48 apresenta esta proteção em funcionamento, auxiliada por um *schimitt-trigger* (CI74HC14) que garante uma transição rápida entre os níveis lógicos alto e baixo.

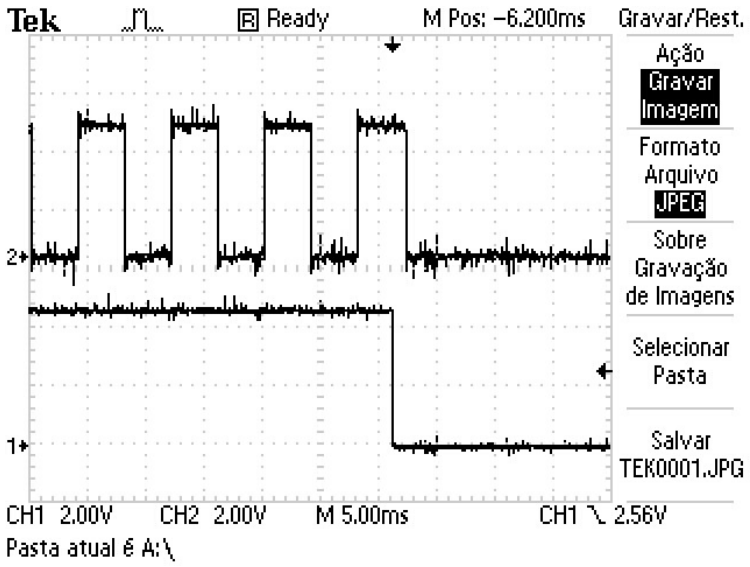

Figura 48: Sinal de pulsos (acima) e do sensor (abaixo)

6.2 RELAÇÃO DE PULSOS/GRAU E PULSOS/ABERTURA DA GARRA

Para a realização deste ensaio, programou-se uma VI que aciona cada junta e, paralelamente, conta a quantidade de pulsos enviados a cada motor. Esta VI cessa a geração de pulsos automaticamente, caso detecte a passagem pelo fim de curso, ou manualmente, caso seja comandado pelo usuário.

Estabelecendo-se uma frequência bastante baixa, de 15 Hz, as juntas 1, 2, 3 e 4 foram acionadas até atingir seus respectivos fins de curso em um sentido. Seus contadores foram então zerados e o motores voltaram a ser acionados, porém no sentido oposto, até que o fim de curso do outro extremo fosse sensibilizado e o movimento cessasse imediatamente.

Levando-se em consideração que a junta 5 (movimento de rolagem do punho) e a abertura da garra não possuem nenhuma indicação de fim de curso, o ensaio, para estes casos, foi diferenciado. Para a junta 5, executou-se uma rotação completa, acionando-se, a princípio, somente os motores 4 e 5. Como consequência o movimento acionado tornou-se composto por uma rolagem mais uma quantidade de abertura da garra. Em seguida foi realizada uma calibração empírica do motor 6, ajustando o número de pulsos a serem dados, de maneira a tornar a abertura da garra nula.

A tabela 12 contém os pulsos contados, as medidas dos limites, realizada no capítulo 4 e a relação procurada.

Tabela 12: Resultados do ensaio dos limites

Variável	Pulsos	Medida	Relação
Junta 1	2769	331°	8,36 pulsos/grau
Junta 2	1962	76°	25,81 pulsos/grau
Junta 3	2431	198°	12,27 pulsos/grau
Junta 4 (M4 e M5)	1032	154°	6,71 pulsos/grau
Junta 5 (M4 e M5)	2410	360°	6,69 pulsos/grau
Junta 5 (M6)	1426	360°	3,96 pulsos/grau
Abertura (M6)	1800	13 cm	138,46 pulsos/cm

6.3 MOVIMENTOS DE PRÉ E PÓS OPERAÇÃO DO ROBÔ

6.3.1 INICIALIZAÇÃO

No momento em que o robô é ligado, não há garantia da sua exata posição e, como o sistema de controle não é retroalimentado, assume-se que sua posição inicial é semelhante à posição calibrada para seu estado de Park, que pode ser visto na Figura 40 do Capítulo 5.

Deve-se, porém, considerar a possibilidade de que, durante seu desligamento, as juntas 2, 3 e 4 passem dos seus finais de curso. Portanto, antes de iniciar o seu processo de referenciação, ou busca por fins de curso, é necessário que as juntas sejam acionadas de maneira a garantir que não estejam situadas na região pós fim de curso.

Utilizando-se da VI de contagem de pulsos já descrita anteriormente, calibraram-se as seguintes quantidades de pulsos a serem executados nesse movimento de inicialização:

Tabela 13: Pulsos do movimento de inicialização

Variável	Pulsos na Inicialização	Sentido
Junta 2	500	positivo
Junta 3	100	negativo
Junta 4	260	negativo

6.3.2 HOME

Uma vez atingidas as condições especificadas na subseção anterior, executa-se o processo de referenciação, já descrito no Capítulo 5. Em sequência, antes de o controle ser passado ao usuário, o robô deve movimentar-se para a posição chamada Home.

Normalmente, estipula-se o vetor de ângulos de junta para a posição Home como nulo, porém, neste caso, a junta 2 não atinge a angulação de 0 graus e, portanto, este vetor foi

definido como

$$\mathbf{q}_{home} = \begin{bmatrix} 0 \\ -90^\circ \\ 0 \\ 0 \\ 0 \end{bmatrix} \tag{6.1}$$

Um benefício desta postura é que o robô mantém-se parado caso sua energia cesse abruptamente, ou seja, o braço permanece em 90° com o solo sem cair. Utilizando-se novamente da VI de contagem de pulsos e dos instrumentos de medição, a quantidade de pulsos necessários para atingir a posição *home* de cada junta foram adquiridas e estão descritas na tabela 14

Tabela 14: Pulsos do movimento de Home

Variável	Pulsos para *Home*	Sentido
Junta 1	1180	negativo
Junta 2	1112	positivo
Junta 3	1340	negativo
Junta 4	509	negativo

Utilizando-se desses valores e dos valores contidos na tabela 12, torna-se possível calcular os limites de cada junta, em ângulos absolutos da referência. Este ângulos estão apresentados na tabela 15.

Tabela 15: Ângulos limites das juntas em valores absolutos

Variável	Limite Mínimo	Limite Máximo
Junta 1	$-190,05^\circ$	$140,95^\circ$
Junta 2	$-133,07^\circ$	$-57,07^\circ$
Junta 3	$-88,85^\circ$	$109,15^\circ$
Junta 4	$-78,05^\circ$	$75,95^\circ$

6.3.3 PARK

O processo de calibração do *Park*, assim como os outros, foi realizado utilizando-se a VI de contagem de passos, até que ficasse na posição apresentada na Figura 40 do

Capítulo 5. O vetor de ângulos de junta calibrado, em valores absolutos, é

$$\mathbf{q}_{home} = \begin{bmatrix} 0 \\ -128^\circ \\ 107^\circ \\ 75^\circ \\ 0 \end{bmatrix} \tag{6.2}$$

6.4 CONSIDERAÇÕES DE ACIONAMENTO

Em implementações envolvendo motores de passo, deve-se atentar para a frequência máxima aceitável para acionamento. Pode-se dizer que esta frequência é atingida quando o número de passos comandados torna-se, por limitações mecânicas, maior que o número de passos executados pelo motor. Essa condição, na qual se "perdem passos", ocorre mais facilmente quando o motor é submetido a acelerações bruscas (do tipo degrau) e em sistemas mecânicos de alta inércia.

6.4.1 PERFIL DEGRAU DE ACIONAMENTO

Embora esta aceleração brusca acarrete maior limitação na velocidade do motor, seu sistema de controle é bastante simplificado se comparado aos que geram perfis de acionamento suaves. De fato, é comum que seja necessário utilizar algoritmos numéricos, devido a dificuldade envolvida nos cálculos que executam um perfil de movimento suave entre vários pontos de uma trajetória.

Visando à simplificação do sistema de controle, definiu-se um modo de operação do robô chamado Preciso, no qual a velocidade dos motores é acionada num perfil degrau de pequena amplitude. Não obstante não seja o objetivo deste trabalho otimizar as capacidades dinâmicas do robô, é interessante que se realize um ensaio para se definir uma velocidade razoável de operação na qual se garanta que passos não sejam "perdidos".

Para a realização deste ensaio os seguintes passos foram seguidos:

- Cada junta é colocada em um de seus finais de curso;
- As outras juntas são ajustadas para que a junta em teste encontre-se na sua posição de maior inércia, ou seja, a que exige maior esforço de seu(s) motor(es) para ser movida;

- Aciona-se, por meio de um degrau da velocidade a ser testada, o(s) motor(es) em um sentido, contando o número de passos;
- Movimenta-se a mesma junta, à uma frequência bastante baixa, no sentido contrário, contando novamente o número de passos; e
- Incrementa-se o valor da velocidade a ser testada e repete-se o processo até que se note diferença nos valores contados na ida e na volta.

A tabela 16 a seguir contém os resultados obtidos com a realização deste ensaio.

Tabela 16: Resultados do ensaio ao degrau

Variável	Frequência máxima(pulsos/segundo)
Junta 1	250
Junta 2	1000
Junta 3	1000
Junta 4	1000
Junta 5	1000
Abertura da garra	400

A Figura 49 foi adquirida por meio de um osciloscópio digital e apresenta um trem de pulsos para um perfil de aceleração do tipo degrau.

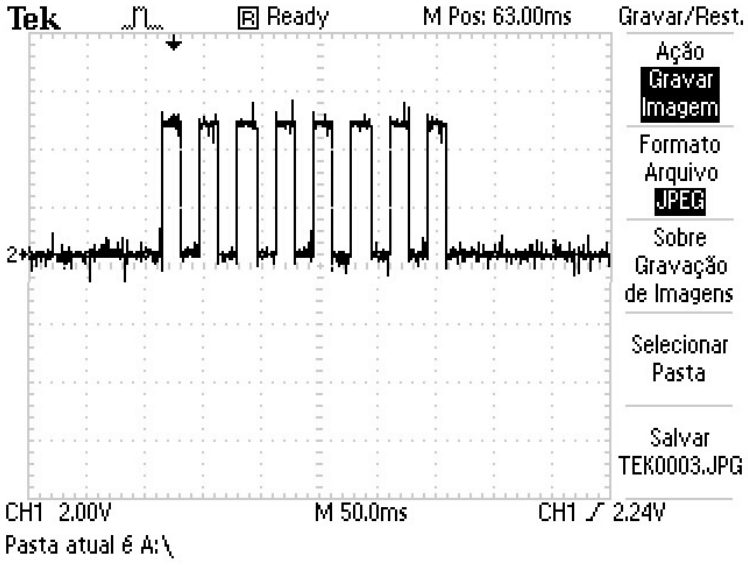

Figura 49: Exemplo de perfil de acionamento do tipo degrau

6.4.2 PERFIL RAMPA DE ACIONAMENTO

É um engano assumir que a geração de rampas, por ser um tipo de perfil bastante simples, seja de fácil implementação. Embora, por definição, uma rampa de velocidade

seja um perfil com derivada $\frac{d(v)}{dt}$ constante, deve-se levar consideração dois pontos fundamentais que aumentam a complexidade dos cálculos exigidos.

Primeiramente, a variável de controle para geração desse movimento não é a velocidade propriamente dita, ou ainda a frequência dos pulsos do acionamento. Esta variável é o período de tempo, contado por um contador com frequência de base fixa, a ser aguardado até o próximo pulso. Por essa variável ser proporcional ao inverso da frequência ($\frac{1}{f}$) perde-se a linearidade que existia no caso da velocidade, ou seja, a derivada $\frac{d(periodo)}{dt}$ não é uma constante.

Em segundo lugar, mesmo que se pudesse controlar a velocidade, deve-se atentar para o fato de que a cada atualização o tempo aguardado diminui. Portanto, a utilização de um incremento constante também não resultaria em uma rampa, mas sim num perfil exponencial. Em outras palavras, a derivada de uma variável em função do tempo não corresponde à variação a ser aplicada a cada contagem, uma vez que o tempo entre essas atualizações varia.

Dessa maneira, utilizou-se a equação a seguir, que relaciona o período anterior a ser contado c_{i-1} como base para se definir o próximo período a ser atualizado dinamicamente.

$$c_i = c_{i-1} - \frac{i-1}{2(i+1)} \tag{6.3}$$

i é um número natural que é incrementado a cada iteração, e c_0 é o primeiro valor do período a ser contado, que define também a aceleração da rampa. Para maiores informações sobre geração de perfis de rampa para motores de passo em tempo real, consultar Austin (2004).

A Figura 50 foi plotada no Matlab® e é resultado do uso da equação descrita acima, para aceleração e desaceleração, com um valor de c_0 inicial igual a 50000 para uma frequência base do contador de 100 kHz.

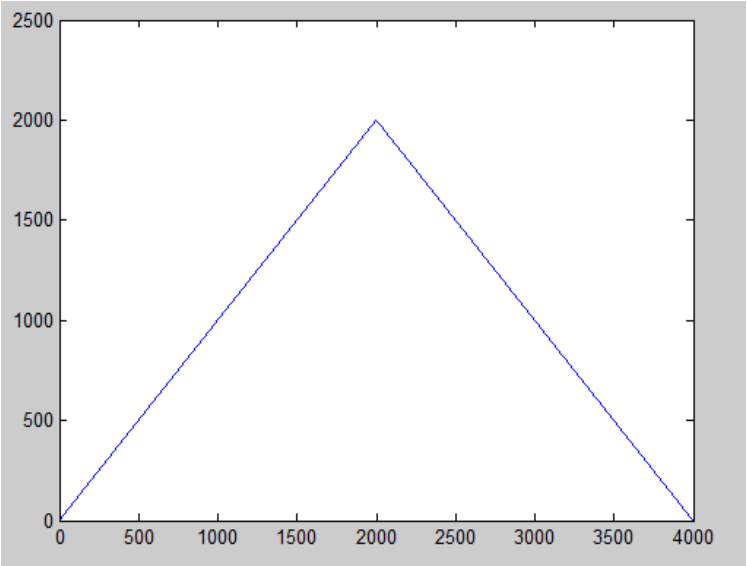

Figura 50: Perfil de acionamento do tipo rampa

Utilizando-se desses princípios realizou-se a implementação de uma VI de acionamento, que objetiva, por meio do uso de rampas, atingir velocidades maiores dos motores, sem que haja o problema de "perda" de contagem de passos. A figura 51 foi adquirida com um osciloscópio digital e apresenta a forma de onda do comando da VI implementada aos motores de passo.

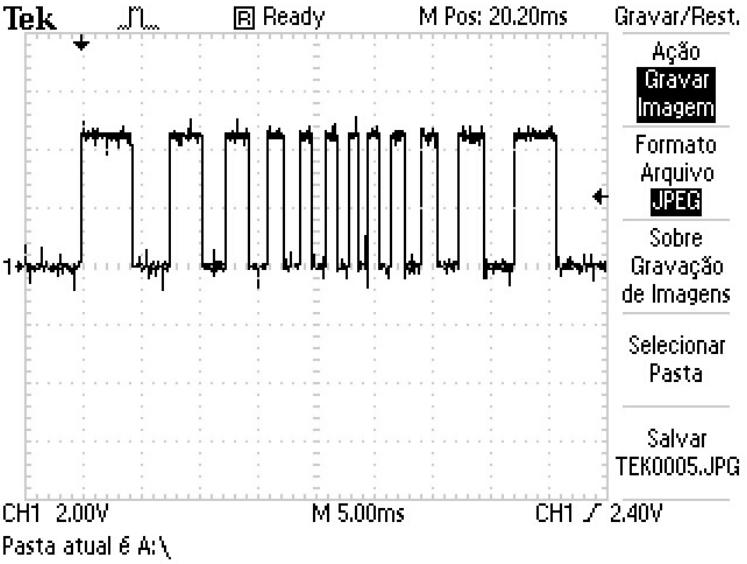

Figura 51: Pulsos enviados no acionamento em rampa

6.5 VALIDAÇÃO DA IMPLEMENTAÇÃO FINAL

Realizadas as calibrações, e as considerações de acionamento chega-se na etapa final deste projeto, na qual se realizará ensaios, para avaliar o desempenho da implementação como um todo, visando, principalmente, à execução de trajetórias.

6.5.1 CÁLCULO DAS FREQUÊNCIAS

Como mencionado na análise de requisitos de sistema do Capítulo 5, o movimento de todos os motores do robô devem iniciar e terminar ao mesmo tempo, independentemente do número de pulsos a serem executados em cada um. Por este motivo, programou-se uma VI que realiza o cálculo das frequências dos motores em função do número de pulsos a serem executados.

A Figura 52 apresenta a forma de onda de dois motores sendo acionados em um único movimento.

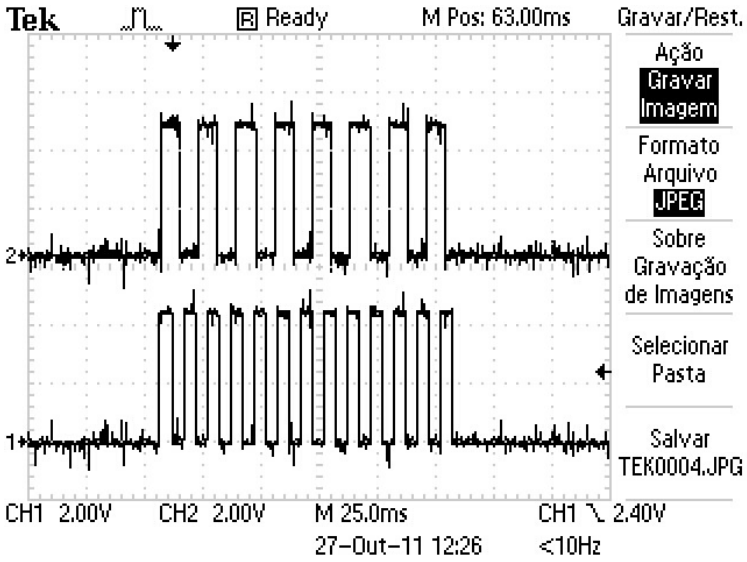

Figura 52: Início e término sincronizados de pulsos

Percebe-se que, embora um motor tenha recebido 8 pulsos e o outro 13, ambos concluíram o movimento ao mesmo tempo, validadndo a funcionalidade da VI responsável pelos cálculos.

6.5.2 TRAJETÓRIAS

Primeiramente, alguns pontos no espaço de trabalho do robô foram escolhidos aleatoriamente, para averiguar visualmente se os movimentos realizados correspondiam aos esperados. Em seguida, para uma medição mais precisa, diversas funções em matlab foram desenvolvidas para gerar trajetórias discretizada em uma série de pontos e salvá-las em arquivos .xls, como a que pode ser vista na figura 53.

	A	B	C	D	E	F	G
1	35	-6,28319	30	0	0	0	A
2	35	-6,15625	30,63296	0	0	0	A
3	35	-6,02932	31,25574	0	0	0	A
4	35	-5,90239	31,85831	0	0	0	A
5	35	-5,77545	32,43098	0	0	0	A
6	35	-5,64852	32,96454	0	0	0	A
7	35	-5,52159	33,4504	0	0	0	A
8	35	-5,39465	33,88073	0	0	0	A
9	35	-5,26772	34,24863	0	0	0	A
10	35	-5,14079	34,54816	0	0	0	A
11	35	-5,01385	34,77451	0	0	0	A
12	35	-4,88692	34,92404	0	0	0	A
13	35	-4,75999	34,99434	0	0	0	A
14	35	-4,63306	34,98427	0	0	0	A
15	35	-4,50612	34,89401	0	0	0	A
16	35	-4,37919	34,725	0	0	0	A
17	35	-4,25226	34,47997	0	0	0	A
18	35	-4,12532	34,16285	0	0	0	A
19	35	-3,99839	33,77875	0	0	0	A
20	35	-3,87146	33,33385	0	0	0	A
21	35	-3,74452	32,8353	0	0	0	A
22	35	-3,61759	32,29113	0	0	0	A
23	35	-3,49066	31,7101	0	0	0	A
24	35	-3,36373	31,10155	0	0	0	A
25	35	-3,23679	30,47528	0	0	0	A
26	35	-3,10986	29,84136	0	0	0	A
27	35	-2,98293	29,20999	0	0	0	A
28	35	-2,85599	28,59134	0	0	0	A
29	35	-2,72906	27,99535	0	0	0	A
30	35	-2,60213	27,43161	0	0	0	A
31	35	-2,47519	26,90921	0	0	0	A
32	35	-2,34826	26,43653	0	0	0	A

Plan1 Plan2 Plan3

Figura 53: Exemplo de tabela para trajetória senoidal

Prendendo-se uma caneta do tipo marcador à garra do robô e colocando-o de frente para um quadro, as trajetórias foram comandadas e seu comportamento observado. Após alguns poucos ajustes, para deixá-lo perfeitamente perpendicular ao quadro, os resultados das trajetórias comandadas finalmente puderam ser observados na forma de desenhos.

Gerou-se uma trajetória senoidal com dois períodos completos e 3 cm de amplitude. A Figura 54 apresenta o robô executando este traço.

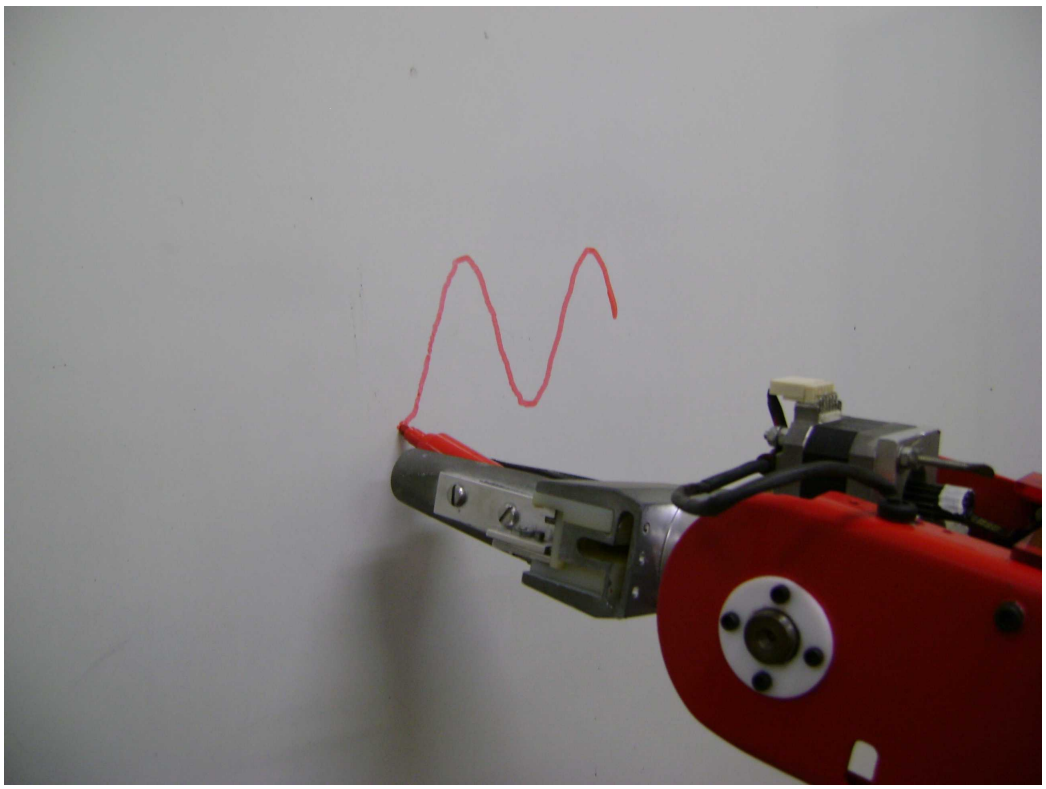

Figura 54: Execução de trajetória senoidal

Em seguida, comandou-se a trajetória de três senoides sobrepostas, com amplitudes de 3, 6 e 9 cm. A figura 55 apresenta a plotagem desta trajetória e a Figura 56 apresenta a resposta do manipulador.

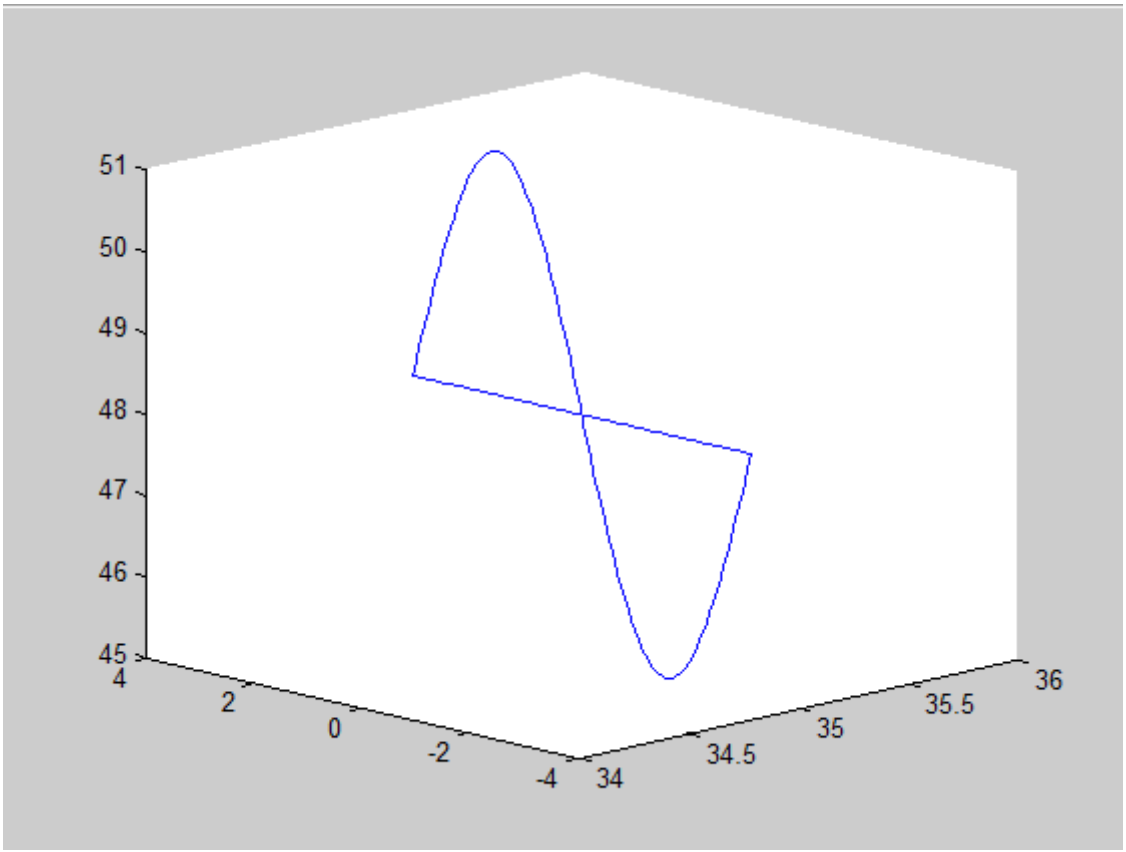

Figura 55: Plotagem da trajetória senoidal gerada

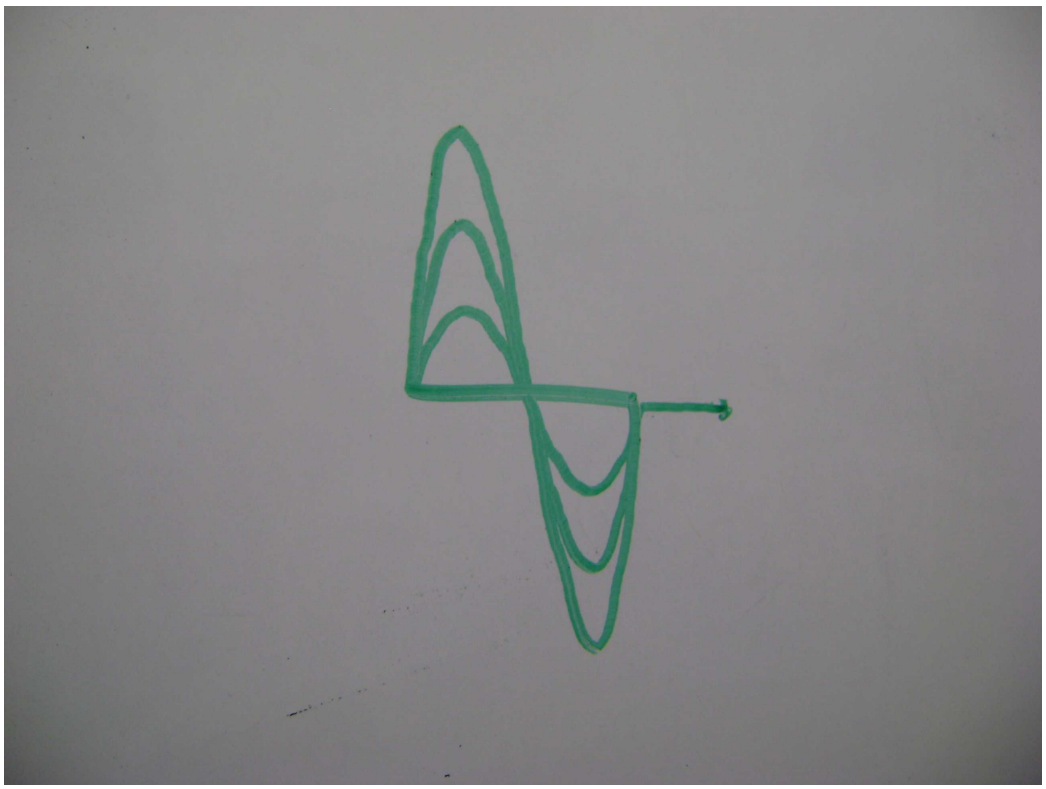

Figura 56: Execução de trajetórias senoidais sobrepostas

7 CONCLUSÃO

Este capítulo tem o objetivo de apresentar as considerações finais pertinentes deste trabalho, na tentativa de ressaltar sua importância, desafios, avaliações e possibilidades.

7.1 AVALIAÇÃO GERAL DO TRABALHO

Pode-se afirmar que, em relação ao seu estado prévio, houve uma evolução considerável nas possibilidades de utilização do robô. Todos os objetivos específicos delimitados na proposta foram atingidos e, por consequência, o Handler atualmente possui implementado um sistema de controle de movimento capaz de executar trajetórias complexas. A Tabela 17, ao final da seção, apresenta um resumo comparativo entre características do projeto prévio e do atual.

Para a UTFPR, o projeto torna possível que o robô seja devolvido às salas de aula, de forma que os cursos de Engenharia Elétrica e Engenharia Mecânica possuem à sua disposição uma ferramenta didática que certamente aprimorará as aulas práticas de disciplinas como Robótica e Instrumentação Virtual.

Para a equipe, a multidisciplinaridade deste trabalho ofereceu uma oportunidade de desenvolver as habilidades técnicas, de metodologia científica, de engenharia e de comunicação. Dentre os conhecimentos técnicos pode-se citar: a familiaridade adquirida com o ambiente LaTeX, extremamente útil para a elaboração de trabalhos acadêmicos; a experiência com o *software* LabVIEW® e MatLab® , que serão de grande valia para a bagagem profissional e a grande melhora na capacidade de gerenciar e executar projetos de grande porte, no que tange à divisão de tarefas, atendimento a prazos, modularização de subtarefas e interação social.

Como se pode observar pela tabela comparativa, uma série de características favoráveis foram agregadas com este trabalho, como aproveitamento de recursos, desenvolvimento matemático, aplicação visual, funcionalidades, capacidade de memória e flexibilidade.

Tabela 17: Parâmetros DH do robô Handler

Quesito	**Previamente**	**Atualmente**
Modelagem Matemática	Nenhuma	Cinemática Direta e Inversa
Arquitetura de *software*	Não documentada	MEF, Produtor-Consumidor
Controlador	CLPs	Placa NI PCI-6601 em um PC com LabVIEW® RT
Placa de sensores	Disparo de relés, dimensões 20cm x 10 cm	TTL com *Schimitt-Trigger*, dimensões 8 cm x 10 cm, proteção de corrente inversa
Alimentação	4 Fontes	2 Fontes
Interface de Usuário	Botoeiras e controle de videogame	Interface gráfica amigável com controle virtual
Gravação de pontos	Por botoeira, não visualizável	Virtual, por carregamento de arquivos .xls, visualizável
Quantidade de pontos armazenados	Dez	Milhões
Execução de trajetórias complexas	Não	Sim
Modos de operação	Manual, cíclico passo-a-passo e cíclico contínuo	Manual, por junta e por postura
Modos de acionamento dos motores	Degrau	Degrau e Rampa
Dificuldade de reprogramação	Enorme para o *software*, e trabalhosa para os pontos	Pequena, tanto para o *software* quanto para os pontos
Retroalimentação	Não existente	Simulada
Inicialização do robô	Manual	Automática

7.2 DESAFIOS ENCONTRADOS

No decorrer do projeto, uma série de decisões precisaram ser discutidas à medida que os problemas apareciam. Nas etapas iniciais, questionou-se a necessidade de um sistema de tempo real para acionamento do robô, e só após a comparação entre uma geração de trem de pulsos com um *loop* normal e um *loop* cronometrado ficou claro o ganho de qualidade e confiabilidade que ele proporciona. Deslocar a responsabilidade sobre o estabelecimento de uma base de tempo para o *hardware* RT aumentou o determinismo tanto no acionamento quanto na rapidez e robustez da leitura dos sensores.

Contudo, embora haja confiabilidade na transmissão, não há muito controle sobre os tempos envolvidos na comunicação entre os PCs via Ethernet, ou seja, ela não é determinística. Além disso, há o problema de que existem variáveis compartilhadas e, portanto, constitui-se uma região crítica na memória de um dos computadores, acessada

e alterada por ambos os lados. Desta forma é necessário que se estabeleçam mecanismos de exclusão mútua para que haja uma garantia de que os dois sistemas não acessem esta região ao mesmo tempo. Ainda dentro da comunicação, percebeu-se uma certa instabilidade na manutenção da conexão por diversas vezes.

A placa NI PCI-6601 não é voltada ao controle de movimento. Portanto, não há um conjunto de ferramentas prontas para acionamento de motores de passo, geração de trajetória, etc. Este entrave demandou o desenvolvimento das VIs em um nível extremamente baixo. Foi necessário criar todo o conjunto de VIs para o controle do robô, tarefa que não seria obrigatória caso houvesse acesso ao *hardware* específico da NI para os fins citados. Com ele, questões como sincronia entre início e fim de acionamento dos motores, perfis de velocidades com rampas de aceleração e desaceleração e interpolação de pontos seriam transparentes ao programador.

A multidisciplinaridade intrínseca do projeto exigiu uma pesquisa abrangente, que por vezes gerou dúvidas quanto à notação e métodos a serem escolhidos. A maior parte da documentação da NI é voltada para sistemas de aquisição de dados; nas aplicações de robótica, normalmente assume-se que o usuário dispõe do módulo *Motion Control*. Por isto, muitas das peculiaridades do trabalho não tinham cobertura suficiente e soluções híbridas foram implementadas. Dezenas de horas foram dedicadas em pesquisas em *sites* como o NI Discussion Foruns, NI Developer Zone, LabVIEW Wiki e o LAVA Forum, muito completos e frequentados por profissionais competentes e mesmo usuários com problemas semelhantes.

O robô Handler sofreu uma restauração muito cuidadosa e beneficente. Todavia, ainda há uma série de imperfeições no que diz respeito a folgas em engrenagens, entrelaçamento de cabos, carência de fins de curso em alguns motores, etc. A falta de experiência com mecânica e o não enquadramento deste tipo de tarefa no escopo do projeto reservou à equipe o direito de repassar estes problemas para grupos futuros. As medições parecem suficientes para um desempenho satisfatório, porém sentiu-se a ausência de um trabalho prévio de desmontagem e medição.

Adicionalmente, a concepção do trabalho anterior sobre o robô Handler era muito diferente. O que se aproveitou foi o manipulador mecânico propriamente dito, os *drivers* e o armário de comando. O restante, por incompatibilidade, teve de ser removido e reprojetado, como a placa de sensores. Estes empecilhos não ofuscam o fato de que o trabalho anterior forneceu uma estrutura muito útil e grande parte da fiação foi reaproveitada. Algumas funcionalidades antigas como botoeiras de reset, chaves

seletoras de ciclo, velocidade e gravação de pontos foram ignoradas por se acreditar na filosofia de intrumento virtual do LabVIEW® .

7.3 SUGESTÃO PARA TRABALHOS FUTUROS

Para trabalhos que visem a dar continuidade a este projeto, algumas sugestões pertinentes são:

- Geração de caminhos com interpolação linear e polinomial. De posse de algoritmos deste tipo, o *software* será capaz de, com apenas dois pontos fornecidos pelo usuário no modo de geração por cinemática inversa, interpolar uma sequência de pontos próximos com espaçamento definido que se configura em uma reta ou mesmo um polinômio de segundo grau. Esta funcionalidade aumentaria a suavidade dos traços e ampliaria as possibilidades para a construção de figuras geométricas mais elaboradas diretamente da interface, sem a necessidade de criação dos caminhos em *softwares* de terceiros.

- Executar uma calibração dimensional precisa do robô.

- Efetuação de reparos na integridade mecânica do robô ou mesmo um novo projeto mecânico de manipulador semelhante, possivelmente implementado com servomotores.

- Melhorias na aplicação com interface gráfica, como animação dos *frames* e plotagem de gráficos que representem a posição, velocidade e mesmo aceleração das juntas em função do tempo. Da mesma forma, a geração de um gráfico tridimensional que exiba uma prévia do caminho desejado facilitaria a projeção dos movimentos.

- Implementação de retroalimentação nos motores de passo e sensores adicionais nas juntas 2, 3, 5 e na garra. A instalação de um sensor de pressão na garra possibilita ao robô uma maior inteligência ao manipular objetos. Pode ser implementado um algoritmo que define limites para a pressão exercida sobre objetos, de forma que tarefas mais delicadas fossem viáveis.

- Avaliação e estudo da dinâmica do robô.

- Implementação de algoritmos de visão de máquina com as devidas modificações no *hardware*.

- Controle de Erros a nível de *software*. Não há um controle rígido sobre a validade das variáveis de junta desejadas (a não ser pelo bloqueio via fins de curso) e sobre a postura desejada. Sendo assim, o usuário deve tomar um certo cuidado em relação às configurações inseridas, pois mesmo com o uso da função de simulação não há uma garantia de que o robô real atinja os pontos alcançados pelo modelo tridimensional.

REFERÊNCIAS

AMERICAN MATHEMATICAL SOCIETY. Users guide for the amsmath package. Providence, 2002.

ASADA, Haruhiko; SLOTINE, Jean-Jacques E. **Robot Analysis and Control**. Nova Iorque: Wiley-Interscience, 1986.

AUSTIN, David. Generate stepper-motor speed profiles in real time. 2004. Disponível em: <http://www.eetimes.com/design/embedded/4006438/Generate-stepper-motor-speed-profiles-in-real-time>. Acesso em: 20 set. 2011.

BARRIENTOS, Antonio et al. **Fundamentos de Robótica**. 1. ed. Madri: McGraw-Hill, 1997.

BEKEY, George A. et al. **Robotics**: state of art and the future challenges. Londres: Imperial College Press, 2008.

BICUDO, Paulo C. D.; TURESSO, Rafael; HALUC, Vicente. **Restauração e Implementação do Sistema de Acionamento e Software de Geração de Trajetória de um Robô Didático**. 44 f. Trabalho de Conclusão de Curso (Graduação) - Curso Superior de Tecnologia em Automação Industrial — Universidade Tecnológica Federal do Paraná, Curitiba, 2010.

BISHOP, Robert H. **The Mechatronics Handbook**. 1. ed. Boca Raton: CRC Press, 2002.

BITTER, Rick; MOHIUDDIN, Taqi; NAWROCKI, Matt. **LabView Advanced Programming Techniques**. 2. ed. Boca Raton: CRC Press, 2007.

BLUME, P.A. **The LabVIEW Style Book**. Crawfordsville: Prentice Hall, 2007.

BRAGA, Newton C. Acopladores e chaves ópticas. 2011. Disponível em: <http://www.newtoncbraga.com.br/index.php/como-funciona/872-acopladores-e-chaves-opticas-art120.html>. Acesso em: 28 mai. 2011.

CRAIG, John J. **Introduction to Robotics**: mechanisms and control. 2. ed. Massachussetts: Addison-Wesley, 1989.

FRADEN, Jacob. **Handbook Of Modern Sensors**: Physics, designs, and applications. 3. ed. Nova Iorque: Springer-Verlag, 2004.

FU, K. S. R. C.; GONZALEZ, C. S. G. Lee. **Robotics**: Control, sensing, vision and intelligence. 1. ed. Nova Iorque: McGraw-Hill, 1987.

GREENBERG, H. J. A simplified introduction to LaTeX. Disponível em: <http://www.cudenver.edu/~hgreenbe/aboutme/pubrec.html>.

GROOVER, Mikell P. et al. **Industrial Robotics**: Technology, programming and applications. 1. ed. Nova Iorque: McGraw-Hill, 1986.

INTERNATIONAL FEDERATION OF ROBOTICS. Service robots: Getting successfully established. **Robotics Online**, set. 2010. Disponível em: <http://www.robotics.org/content-detail.cfm/Industrial-Robotics-Feature-Article/Service-Robots-%E2%80%93-Getting-Successfully-Established/content_id/2349>. Acesso em: 10 out. 2010.

INTERNATIONAL ORGANIZATION FOR STANDARDIZATION. Iso 8373:1994: Manipulating industrial robots - vocabulary. Geneva, 1994.

JAZAR, Reza N. **Theory of Applied Robotics**: Kinematics, dynamics and control. 2. ed. Londres: Springer, 2010.

KHATIB, Oussama. Introduction to robotics - lecture 2. Stanford University, 2007. Disponível em: <http://academicearth.org/lectures/spatial-descriptions>. Acesso em: 14 mar. 2011.

KHATIB, Oussama; SICILIANO, Bruno. **Springer Handbook of Robotics**. Berlim: Springer, 2008.

KURFESS, Thomas R. **Robotics and Automation Handbook**. Nova Iorque: CRC Press, 2005.

LEITE, A. C. S.; SILVA, P. A. B.; VAZ, A. C. R. A importância das aulas práticas para alunos jovens e adultos: uma abordagem investigativa sobre a percepção dos alunos do PROEF II. **Ensaio Pesquisa em Educação e Ciências**, v. 7, p. 3, 2005.

MORRIS, Alan S. **Measurement and Instrumentation Principles**. 1. ed. Oxford: Butterworth-Heinemann, 2001.

NAGRATH, I. J.; MITTAL, R. K. **Robotics and Control**. 6. ed. Nova Deli: Tata McGraw-Hill, 2007.

NATIONAL INSTRUMENTS. Labview development guidelines. Austin, 2003. Disponível em: <http://www.ni.com/pdf/manuals/321393d.pdf>. Acesso em: 23 jul. 2011.

NATIONAL INSTRUMENTS. Vantagens da utilização do labview em pesquisas acadêmicas. 2009. Disponível em: <http://zone.ni.com/devzone/cda/tut/p/id/10161>. Acesso em: 28 set. 2010.

NATIONAL INSTRUMENTS. Building a real-time system with ni hardware and software. 2010. Disponível em: <http://zone.ni.com/devzone/cda/tut/p/id/4040>. Acesso em: 19 jul. 2011.

NATIONAL INSTRUMENTS. Do i need a real-time system? 2010. Disponível em: <http://zone.ni.com/devzone/cda/tut/p/id/10342>. Acesso em: 19 jul. 2011.

NATIONAL INSTRUMENTS. What is a real-time operating system (rtos)? 2010. Disponível em: <http://zone.ni.com/devzone/cda/tut/p/id/3938>. Acesso em: 11 set. 2011.

NATIONAL INSTRUMENTS. Fundamentals of motion control. 2011. Disponível em: <http://zone.ni.com/devzone/cda/tut/p/id/3367#toc2>. Acesso em: 20 mar. 2011.

NEI. A indústria emprega mais robôs para ganhar flexibilidade e competitividade. fev. 2005. Disponível em: <http://www.nei.com.br/artigos/artigo.aspx?i=19>. Acesso em: 13 set. 2010.

NIKU, Saeed B. **Introduction to Robotics**: Analysis, systems, applications. Nova Jersey: Prentice Hall, 2001.

OHNISHI, Kouhei; SABANOVIC, Asif. **Motion Control Systems**. 1. ed. Noida: John Wiley and Sons, 2011.

PAUL, Richard P. **Robot Manipulators**: Mathematics, programming and control. 7. ed. Massachussetts: MIT Press, 1986.

REHG, James A. **Introduction to robotics**: a systems approach. 1. ed. Englewood Cliffs: Prentice-Hall,, 1985.

RITTER, David J. **LabVIEW GUI**: essential techniques. [S.l.]: McGraw-Hill, 2002.

ROMANO, Vitor F.; DUTRA, Max Suell. **Robótica Industrial**. São Paulo: Edgard Blücher, 2008.

ROSÁRIO, João Maurício. **Automação Industrial**. São Paulo: Baraúna SE, 2009.

RUMBAUGH, James; JACOBSON, Ivar; BOOCH, Grady. **The Unified Modeling Language Reference Manual**. Reading: Addison-Wesley, 2005.

SANTOS, Vitor M. F. **Robótica Industrial**. Aveiro: Departamento de Engenharia Mecânica da Universidade de Aveiro, 2001.

SCHILLING, Robert J. **Fundamentals of Robotics**: Analysis and control. 1. ed. Nova Jersey: Prentice-Hall, 1990.

SELIG, J. M. **Introductory Robotics**. Hertfordshire: Prentice Hall, 1992.

SHIMIZU, Heitor. **Robô, O Filho Pródigo**: seremos seus bichos de estimação? São Paulo: Mostarda, 2006.

SICILIANO, Bruno et al. **Robotics**: Modelling, planning and control. Londres: Springer-Verlag, 2009.

SILVEIRA, Paulo R.; SANTOS, Winderson E. **Automação e Controle Discreto**. São Paulo: Érica, 2009.

SIMAO, Jean. **Slides de Fundamentos de Programação 1**. Acesso em: 05 out. 2010. Disponível em: <http://www.pessoal.utfpr.edu.br/jeansimao/Fundamentos1/Algoritmos/Fundamentos1-SlidesA2-E-31072008.pdf>.

SPONG, Mark W.; HUTCHINSON, Seth; VIDYASAGAR, M. **Robot Dynamics and Control**. 2. ed. Nova Iorque: Wiley, 2004.

TSAI, Lung-Wen. **Robot analysis**: The mechanic of serial and parallel manipulators. 1. ed. Nova Iorque: John Wiley and Sons, 1999.

WALENIA, Paulo Sérgio.

Proposta de Implementação do Curso de Engenharia Industrial Elétrica Ênfase Automação — Universidade Tecnológica Federal do Paraná, Curitiba, 2006.

WALTER, Jörg. **Rapid learning in robotics**. Göttingen: Cuvillier, 1996.