

UNIVERSIDADE ESTADUAL DE CAMPINAS  
FACULDADE DE ENGENHARIA MECÂNICA  
DEPARTAMENTO DE PROJETO MECÂNICO

Projeto e Desenvolvimento de um Sistema  
de Geração Automática de Trajetória  
para Manipuladores

Autor: **Joselito Menezes da Cruz**

Orientador: **Prof. Dr. João Maurício Rosário**

10/93

ESTE EXEMPLAR CORRESPONDE A REDAÇÃO FINAL DA  
TESE DEFENDIDA POR Joselito Menezes  
da Cruz E APROVADA PELA  
COMISSÃO JULGADORA EM 08/10/93.

João Maurício Rosário  
ORIENTADOR

UNIVERSIDADE ESTADUAL DE CAMPINAS  
FACULDADE DE ENGENHARIA MECÂNICA  
DEPARTAMENTO DE PROJETO MECÂNICO

Projeto e Desenvolvimento de um Sistema  
de Geração Automática de Trajetória  
para Manipuladores

Autor: **Joselito Menezes da Cruz**

Orientador: **Prof. Dr. João Maurício Rosário**

Curso: Engenharia Mecânica

Área de Concentração: Projeto Mecânico

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

Campinas, 1993  
S.P. - Brasil

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

C889p Cruz, Joselito Menezes da  
Projeto e desenvolvimento de um sistema de geração automática de trajetória para manipuladores / Joselito Menezes da Cruz.--Campinas, SP: [s.n.], 1993.

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

1. Robótica. 2. Automação. 3. Interface de usuário (Sistema de computador). 4. Plataforma continental. I. Rosário, João Maurício. II. Universidade Estadual de Campinas. Faculdade de Engenharia Mecânica. III. Título.

UNIVERSIDADE ESTADUAL DE CAMPINAS  
FACULDADE DE ENGENHARIA MECÂNICA  
DEPARTAMENTO DE PROJETO MECÂNICO

## Tese de Mestrado

### Projeto e Desenvolvimento de um Sistema de Geração Automática de Trajetória para Manipuladores

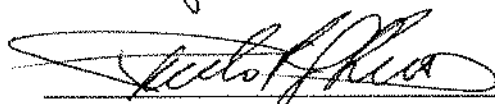
Autor: Joselito Menezes da Cruz

Orientador: Prof. Dr. João Maurício Rosário

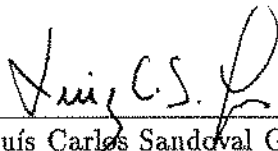
Aprovado por:



Prof. Dr. João Maurício Rosário - Presidente



Prof. Dr. Paulo Roberto Gardel Kurka



Prof. Dr. Luís Carlos Sandoval Góes

Campinas-SP, 08 de outubro de 1993

## AGRADECIMENTOS

- Ao amigo e professor João Maurício Rosário, pelo incentivo e acompanhamento durante todas as etapas do trabalho.
- Ao amigo Josemar Tavares pelo constante estímulo e interesse demonstrados durante o trabalho.
- Ao amigo Luiz Antônio, pelo companheirismo e pelas discussões elucidativas.
- Aos amigos Fançony e Coelho pelo apoio e sugestões.
- Ao colegas da república: Rogério, Fábio, Oswaldo, Rober, Michael e Fred pela convivência quase sempre tranquila.
- À Coordenação de Aperfeiçoamento de Pessoal de Ensino Superior (CAPES), pelo apoio financeiro para a realização deste trabalho.
- Aos técnicos do Laboratório de Projeto mecânico (DPM-FEM) e do Centro de Tecnologia (CT-UNICAMP) pelos serviços prestados na realização da parte experimental.
- Ao DPM, pela estrutura cedida para o desenvolvimento deste trabalho.
- Ao CENPES-PETROBRÁS pela oportunidade e apoio para a realização deste trabalho de pesquisa.
- A todos que, direta ou indiretamente, contribuíram para a conclusão deste trabalho.

*Aos meus pais pelo amor e carinho que sempre me dedicaram e pela formação moral que me deram.*

*Aos meus irmãos pelo companheirismo e espírito incentivador.*

## RESUMO

O objetivo principal deste trabalho é o desenvolvimento de um sistema de geração automática de trajetória para manipuladores, com o intuito de viabilizar a automatização de tarefas submarinas. A motivação para sua realização se deve a um programa de cooperação científica envolvendo a Universidade Estadual de Campinas, a Petrobrás e o Instituto de Pesquisas Tecnológicas de Geesthacht (GKSS) da Alemanha, no qual se pretende utilizar dispositivos robóticos para auxiliar na prospecção de petróleo em plataformas marítimas.

O desenvolvimento de um ambiente submarino automatizado utilizando robôs pressupõe a existência de um sistema capaz de controlar e monitorar as tarefas a serem realizadas. O sistema proposto por este trabalho visa suprir a um telemanipulador as necessidades de controle inerentes a ambientes hostis, pois conta com diversas facilidades tais como módulos de programação de tarefas *on-line* e *off-line*, possibilidade de visualização cartesiana em tempo-real e animação gráfica.

Inicialmente são introduzidos conceitos sobre robôs e manipuladores e a formulação matemática para trabalhar com eles, em seguida é apresentada a estrutura do sistema e a estratégia utilizada para resolver o problema proposto. O sistema foi implementado em um microcomputador compatível com a linha IBM-PC-AT

## ABSTRACT

The main objective of this work is the development of an automatic trajectory generation system with the intention of to turn viable the automation of underwater tasks. The motivation to its execution is due to a scientific cooperation program between State University of Campinas (UNICAMP), Petrobras and Geesthacht Institute of Technology (GKSS) from Germany, in which one intends to use robotics devices to assist in exploitation of oil on off-shore platforms.

The development of an automatic underwater environment using robots presupposes the existence of a system able to control and monitoring the tasks to be carried out. The proposed system by this work aims to supply a telemanipulator the control needs inherent to hostile environments, since it has a number of facilities such as modes of on-line and off-line task programming, possibility of cartesian real time visualization and graphic animation.

Firstly, the concepts of robotics and the mathematical formulation to handle with it are introduced, next, the structure of the system and the strategy used to solve the problem is presented. The system was implemented in a microcomputer compatible with the standard IBM-PC.



# Prólogo

A utilização de robôs industriais adaptados para a realização de tarefas submarinas automatizadas tem sido feita para possibilitar reparos, inspeções e manobras em águas intermediárias e profundas, nas quais seria altamente oneroso e perigoso o envio de um homem para a execução do trabalho.

Com esse objetivo, iniciou-se um projeto de cooperação envolvendo a Petrobrás, o Instituto de Pesquisas Tecnológicas de Geesthacht (GKSS) da Alemanha e a Universidade de Campinas, Unicamp, visando o desenvolvimento de uma célula automatizada de operações que permitisse a geração automática de tarefas a partir do conhecimento do ambiente de atuação, utilizando fundamentalmente robôs e telemanipuladores.

Dentro desse programa de cooperação, o GKSS cedeu o robô Manutec r3 e a Petrobrás, o manipulador submarino Kraft, para que fosse desenvolvida na Unicamp toda a infra-estrutura de apoio, a nível de construção mecânica, software e hardware, necessária para o desenvolvimento do projeto. Estes robôs tiveram que sofrer algumas alterações em relação às suas configurações originais para que se adaptassem ao tipo de tarefa às quais seriam submetidos.

Algumas das etapas básicas do projeto foram:

- Construção Mecânica (Base móvel, Maquetes para testes, ferramentas dedicadas);
- Desenvolvimento de um software de programação *off-line* do sistema robótico;

- Desenvolvimento de uma interface Homem-Máquina amigável com o manipulador Kraft;

A interface Homem-Máquina seria uma plataforma para se atingir dois outros objetivos maiores:

- Desenvolvimento de um sistema de Identificação de parâmetros e medida de acurácia;
- O desenvolvimento de uma linguagem de programação textual para o manipulador;
- O desenvolvimento de um sistema de geração automática de trajetória.

Este trabalho descreve o desenvolvimento da interface de comunicação com o manipulador, e o procedimento utilizado para a geração automática de trajetórias. Com esta finalidade, o trabalho é subdividido em quatro capítulos, conforme descritos a seguir:

O capítulo 1 faz uma introdução aos objetivos do programa de cooperação Unicamp/Petrobrás/GKSS e apresenta os sistemas robóticos disponíveis para o desenvolvimento do projeto, bem como a estrutura mecânica construída de forma a dar suporte a esses dispositivos.

O capítulo 2 descreve o sistema Kraft e apresenta de forma sucinta o desenvolvimento da interface de comunicação entre o manipulador e um microcomputador, assim como o software de controle para esta interface.

No capítulo 3 faz-se o desenvolvimento da modelagem geométrica do robô Manutec e do manipulador Kraft e é discutido alguns métodos numéricos para a solução do problema da modelagem cinemática de robôs.

O capítulo 4 trata diretamente do problema de geração de trajetórias, abordando principalmente os métodos de geração no espaço de juntas e no espaço cartesiano.

O sistema proposto no trabalho é validado através de sua implementação no manipulador Kraft e dos testes realizados no Centro de Pesquisas da Petrobrás (CENPES). O capítulo 5 descreve a metodologia utilizada para a execução desses testes e para a validação do sistema.

Finalmente, são apresentados os comentários finais, as perspectivas e as conclusões do trabalho.

# Conteúdo

<b>1</b>	<b>Posicionamento do Problema</b>	<b>16</b>
1.1	Introdução . . . . .	16
1.2	Utilização de Robôs e Manipuladores em Ambiente Submarino . . . . .	17
1.3	Descrição dos Dispositivos Robóticos Utilizados . . . . .	18
1.4	Estrutura Mecânica Construída . . . . .	24
1.4.1	Base Móvel . . . . .	24
1.5	Programação Off-line . . . . .	25
1.6	Conclusões . . . . .	28
<b>2</b>	<b>Interface Homem-Máquina</b>	<b>29</b>
2.1	Introdução . . . . .	29
2.2	Interface com o Robô Manutec R3 . . . . .	30
2.3	Interface com o Manipulador KRAFT . . . . .	30
2.4	O Hardware . . . . .	36
2.5	O Software . . . . .	39
2.6	Conclusões . . . . .	42
<b>3</b>	<b>Modelagem do Manipulador Kraft e do Robô Manutec r3</b>	<b>43</b>

3.1	Introdução . . . . .	43
3.2	Posicionamento do Problema . . . . .	44
3.3	Modelo Geométrico . . . . .	44
3.3.1	Transformação Homogênea . . . . .	46
3.3.2	Descrição da Matriz de Orientação Através de Ângulos . . . . .	47
3.3.2.1	Ângulos de Euler . . . . .	48
3.3.2.2	Ângulos Roll, Pitch, Yaw . . . . .	50
3.3.3	Parâmetros de Denavit-Hartenberg (D.H.) . . . . .	52
3.4	Modelo Geométrico do Manutec r3 . . . . .	54
3.5	Modelo Geométrico do Kraft . . . . .	56
3.6	Problema Inverso . . . . .	57
3.6.1	Modelagem Cinemática Inversa . . . . .	59
3.7	Conclusões . . . . .	62
<b>4</b>	<b>Geração de Trajetórias</b>	<b>63</b>
4.1	Introdução . . . . .	63
4.2	Controle de Trajetória . . . . .	64
4.3	Trajetória no Espaço de Juntas . . . . .	65
4.3.1	Lei de Movimento . . . . .	67
4.3.2	Gravação do Caminho Contínuo . . . . .	69
4.3.3	Gravação Ponto-a-Ponto . . . . .	72
4.3.3.1	Interpolação . . . . .	73
4.3.3.2	Filtragem . . . . .	75
4.4	Trajetória no Espaço Cartesiano . . . . .	80
4.4.1	Estratégia Implementada . . . . .	81

4.5	Conclusões . . . . .	85
<b>5</b>	<b>Implementação e Validação do Sistema</b>	<b>86</b>
5.1	Introdução . . . . .	86
5.2	Teste do Hardware de Comunicação . . . . .	86
5.2.1	Funcionamento das Placas A/D e D/A . . . . .	87
5.2.2	Chaveamento dos Modos de Operação . . . . .	88
5.3	Teste do Módulo de Geração de Trajetória . . . . .	88
5.3.1	Obtenção das Trajetórias . . . . .	88
5.3.2	Validação do Sistema . . . . .	91
5.3.2.1	Procedimento . . . . .	92
	<b>APÊNDICE A</b>	<b>96</b>
	<b>APÊNDICE B</b>	<b>102</b>
.1	Características do Manipulador Kraft . . . . .	103
.1.1	Especificações Gerais . . . . .	103
.1.2	Efetuator . . . . .	103
.1.3	Movimento das Juntas . . . . .	103
.2	Características do Robô Manutec r3 . . . . .	104
.2.1	Especificações Gerais . . . . .	104
.2.2	Velocidades . . . . .	104
.2.3	Acelerações . . . . .	105
.2.4	Movimento das Juntas . . . . .	105
.3	Características da Placa A/D Utilizada . . . . .	106
.4	Características da Placa D/A Utilizada . . . . .	107

5	Esquema Eletrônico da Placa ICMK .....	108
APÊNDICE C	.....	108

# Lista de Figuras

1.1	<i>Manipulador Kraft - Slave.</i> . . . . .	19
1.2	<i>Manipulador Kraft em operação na Unicamp.</i> . . . . .	20
1.3	<i>Manipulador Kraft - Master.</i> . . . . .	21
1.4	<i>Robô Manutec r3 em operação na Unicamp.</i> . . . . .	22
1.5	<i>Mock-up construído na Unicamp para testes.</i> . . . . .	25
1.6	<i>OCTOS-1000<sup>TM</sup>. Plataforma na qual o sistema irá atuar.</i> . . . . .	25
1.7	<i>Robô Manutec r3 operando sobre a Base Móvel.</i> . . . . .	26
1.8	<i>Estrutura da Programação off-line.</i> . . . . .	27
2.1	<i>Esquema de Programação off-line no robô Manutec.</i> . . . . .	31
2.2	<i>Sistema Master-Slave Kraft Grips.</i> . . . . .	32
2.3	<i>Configuração inicial do sistema.</i> . . . . .	33
2.4	<i>Configuração atual do sistema.</i> . . . . .	34
2.5	<i>Modo Monitoramento.</i> . . . . .	35
2.6	<i>Modo Controle.</i> . . . . .	36
2.7	<i>Modo Simulação de Vôo.</i> . . . . .	37
2.8	<i>Esquema do Software de Controle.</i> . . . . .	40
3.1	<i>Diagrama de blocos de um controlador de trajetória.</i> . . . . .	44

3.2	Vetor posição e orientação da garra. . . . .	47
3.3	Definição dos Ângulos de Euler. . . . .	49
3.4	Definição da função ATAN2. . . . .	50
3.5	Definição dos Ângulos Roll, Pitch e Yaw. . . . .	51
3.6	Parâmetros de Denavit-Hartenberg. . . . .	52
3.7	Múltiplas Soluções para o Modelo Geométrico Inverso . . . . .	58
4.1	Diagrama de blocos de um controlador de trajetória. . . . .	64
4.2	Possíveis caminhos suaves de $\theta_0$ a $\theta_f$ . . . . .	66
4.3	Trajatória linear no espaço de juntas. . . . .	67
4.4	Trajatória de um robô no plano $xy$ . Os set-points são indicados por pontos ao longo da linha. . . . .	68
4.5	Perfil de posição linear com extremidades parabólicas. . . . .	69
4.6	Perfil de velocidade trapezoidal: Necessário para movimento suave. . . . .	70
4.7	Uso de um gravador multicanal para gravar os sinais dos potenciômetros das juntas de um robô. . . . .	71
4.8	Trajatória de uma junta $i$ após o processo de interpolação. . . . .	76
4.9	Filtro FIR do tipo Janela Triangular. . . . .	77
4.10	(a)Exemplo de filtro triangular. (b)Exemplo de um sinal a ser filtrado. . . . .	78
4.11	Processo de convolução discreta. . . . .	79
4.12	Movimento em Linha Reta. . . . .	82
4.13	Processo de inversão do Jacobiano. . . . .	83
5.1	Ligação do sistema para teste das placas A/D e D/A. . . . .	87
5.2	Estrutura do sistema de geração de trajetória ponto-a-ponto. . . . .	90
5.3	Trajatória interpolada e filtrada. . . . .	91



5.4	<i>Sincronização do movimento das juntas 2 e 3.</i>	92
5.5	<i>Esquema da bancada para teste de verificação do comportamento do manipulador ao executar uma trajetória interpolada e uma filtrada.</i>	93
6	<i>Circuito eletrônico da placa ICMK</i>	108

## Lista de Tabelas

1.1	<i>Manutec × Kraft.</i> . . . . .	23
3.1	<i>Definição dos Parâmetros de Denavit-Hartenberg - 2 links adjacentes.</i> . . . .	53
3.2	<i>Robô Manutec r3 - Parâmetros de Denavit-Hartenberg.</i> . . . . .	54
3.3	<i>Manipulador Submarino Kraft - Parâmetros de D.H.</i> . . . . .	56
5.1	<i>Trajectoria utilizada para os testes.</i> . . . . .	89

# Capítulo 1

## Posicionamento do Problema

### 1.1 Introdução

A utilização de robôs em ambientes de difícil acesso ao homem tem se tornado cada vez mais objeto de interesse dos pesquisadores em todo o mundo.

Diversas companhias de renome estão injetando recursos vultuosos em pesquisas nessa área, que já é uma realidade crescente. A NASA desenvolveu um telemanipulador com a finalidade específica de auxiliar no posicionamento de dispositivos espaciais em órbita, tais como, satélites, telescópios, antenas etc., eliminando assim a necessidade de um homem arriscar sua vida para executar a mesma tarefa. Também, robôs com a finalidade de explorar solos de outros planetas tem sido desenvolvidos.

Uma outra aplicação cada vez mais frequente é a utilização dessas máquinas para inspeção e manutenção do interior de reatores nucleares.

A necessidade de prospecção de petróleo em águas intermediárias e profundas, e o alto custo e risco da utilização de mergulhadores para a execução de serviços de manutenção, inspeção e reparo propiciou a realização de um trabalho de investigação na área de automação com o objetivo de automatizar as tarefas através de dispositivos robóticos.

No decorrer deste capítulo serão descritas as principais etapas do programa de pesquisa citado com o objetivo de motivar o leitor e despertá-lo para a necessidade do desenvolvimento de interfaces que permitam a realização dos objetivos propostos neste trabalho.

## 1.2 Utilização de Robôs e Manipuladores em Ambiente Submarino

Para a execução de tarefas submarinas automatizadas, é necessário um sistema dedicado, controlado remotamente para observação de operações, inspeções, reparos e manobras em equipamentos e instalações. Os robôs industriais normalmente substituem o homem na execução de tarefas manuais bem-estruturadas, repetitivas e as vezes bastante tediosas. A utilização de robôs tele-operados, por outro lado, além de viabilizar a automação de tarefas menos corriqueiras, possibilita maior interação do homem com o ambiente submarino, aumentando suas capacidades manuais e perceptivas e extendendo-as a um ambiente remoto, hostil e perigoso, antes inacessível a ele.

A adaptação de robôs para trabalho submarino tem tornado possível se conceber e planejar tarefas de inspeção, reparo e manutenção (IRM) em locais antes considerados impossíveis, portanto, está se abrindo um novo mercado de oportunidades e um novo campo para a adaptação de robôs industriais [Rosário,91].

A Universidade Estadual de Campinas, Unicamp, em um programa de cooperação científica envolvendo o Centro de Pesquisas de Geesthacht (GKSS) na Alemanha e a Petrobrás (através de seu centro de pesquisas - CENPES) está realizando um projeto conjunto de pesquisa e desenvolvimento na área de tecnologia submarina. Um interesse especial do projeto é a adaptação de manipuladores submarinos para trabalhos automatizados no fundo do mar, a profundidades que variam de 500 a 2000m. Esse programa pode ser subdividido em três partes principais:

- **Instalação de um Ambiente Submarino:** Construção de um ambiente submarino apropriado para aplicações robóticas utilizando um robô industrial adaptado para o trabalho submarino [Aust,90] e sua unidade de controle, assim como um

manipulador submarino (ambos serão descritos posteriormente). O projeto e desenvolvimento dessas instalações foram realizadas tendo em vista a utilização desses dois robôs e a implementação do sistema em águas profundas brasileiras. Como resultado disso, foi fechado um acordo para a realização de um programa experimental, e então decidiu-se pela construção de dois *Mock-ups* (maquetes), um para ser instalado na Unicamp e outro no GKSS. As tarefas a serem automatizadas também foram especificadas.

- **Adaptação Robô/Manipulador:** Construção de uma base móvel sobre a qual o sistema está colocado e que é capaz de se movimentar por sobre trilhos existentes no *mock-up* e construção de ferramentas dedicadas para uso dos robôs;
- **Programação Off-line:** A programação *off-line* consiste no desenvolvimento de um programa computacional para simulação do cenário completo de atuação do robô e sua base móvel (o sistema completo de manobra, as ferramentas dedicadas utilizadas, o ambiente de trabalho, etc.).
- **Tarefas Automatizadas:** Desenvolvimento do controle e de interfaces homem-máquina voltadas para aplicações de automação de tarefas submarinas utilizando robôs e manipuladores.

### 1.3 Descrição dos Dispositivos Robóticos Utilizados

O Kraft é um sistema robótico tele-operado de propósitos gerais projetado para executar tarefas em ambiente submarino ou em outros ambientes hostis. Um robô tele-operado é aquele em que os movimentos do manipulador (*slave*) são comandados por um operador a partir de um dispositivo de controle. No caso, esse dispositivo é um *minimaster*, ou simplesmente *master*, que é um modelo do manipulador em escala reduzida.

Ele apresenta as vantagens de já estar condicionado ao ambiente submarino e adaptado a operações de telepresença. Sua estrutura mecânica é constituída de titânio e seu acionamento é hidráulico, proporcionando-lhe assim uma alta capacidade de carga (ele é capaz de suportar até 34 Kg com o braço totalmente estendido [Kraft,85]). Um inconveniente, é a necessidade da utilização de uma unidade de hidráulica de potência. A

figura 1.1 mostra a unidade *slave* do manipulador Kraft e a figura 1.2 mostra uma foto desse manipulador durante uma operação simulada.

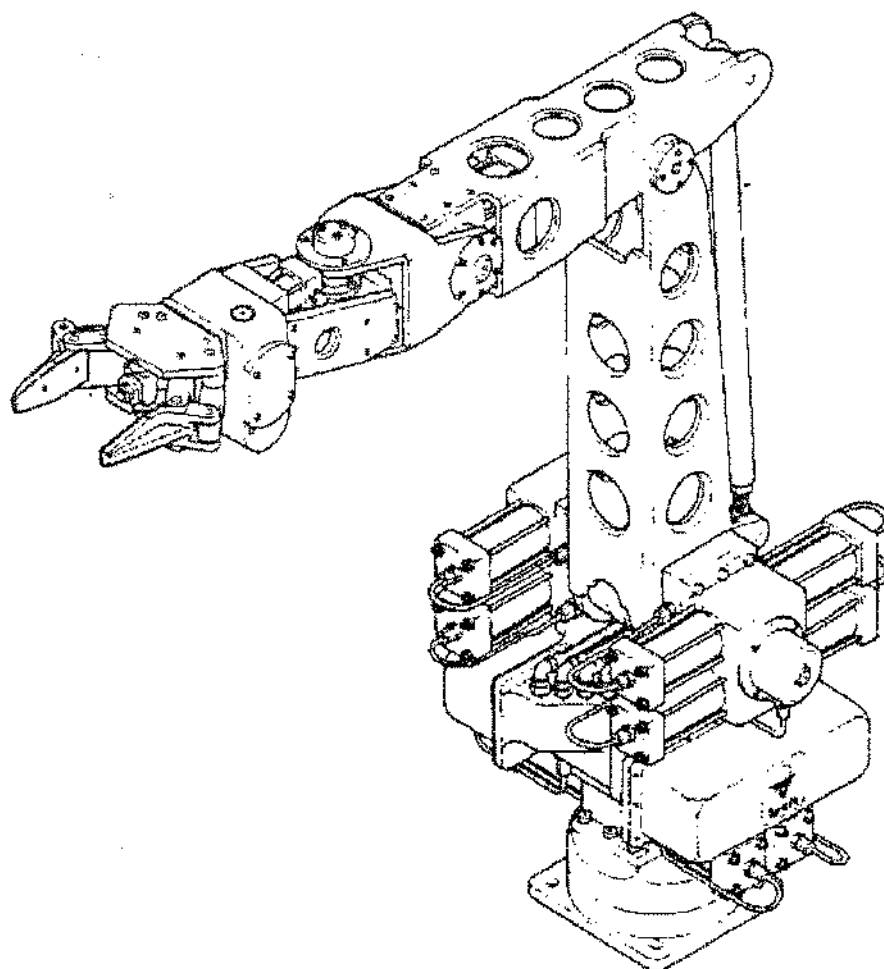
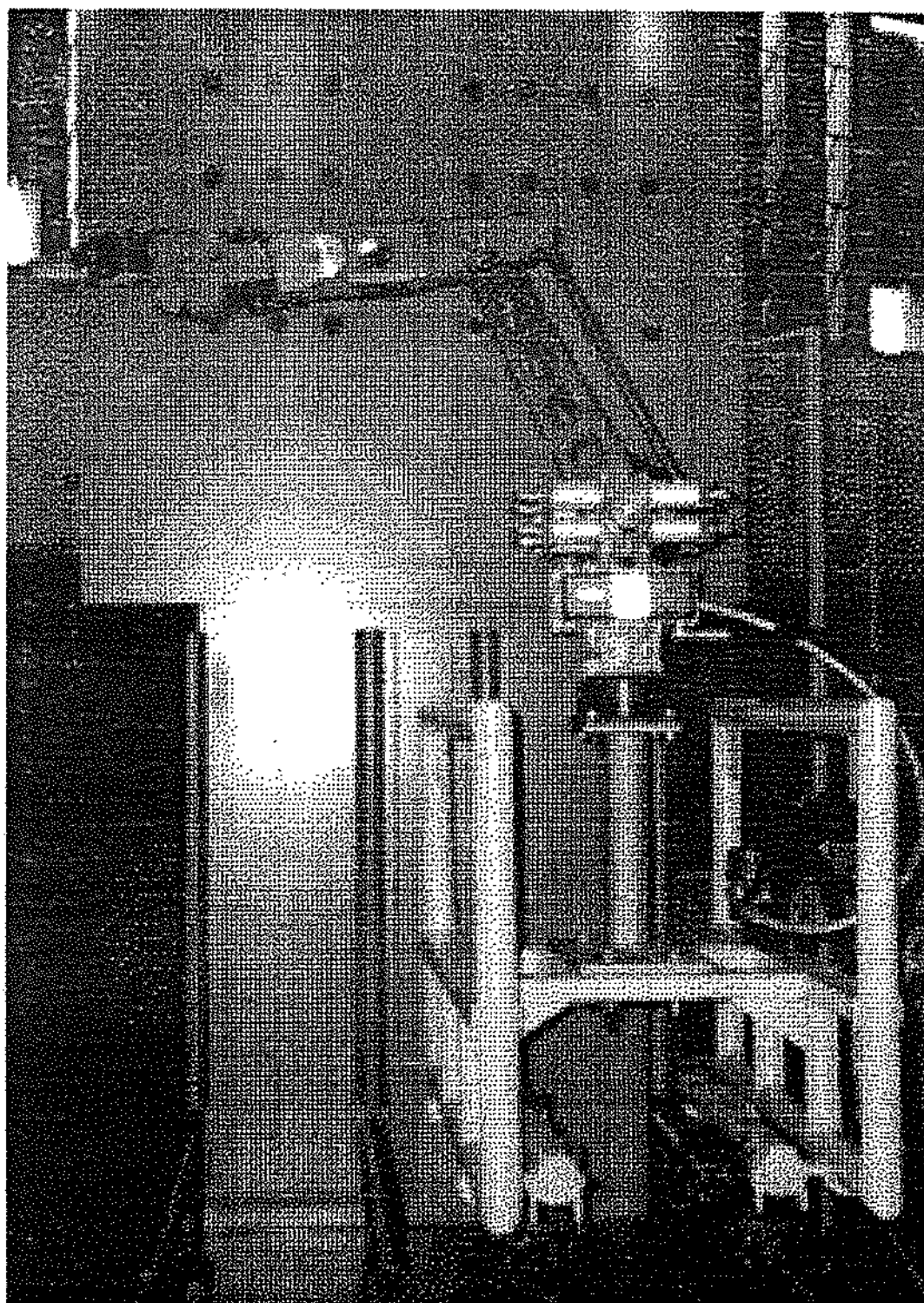


Figura 1.1: Manipulador Kraft - Slave.

O *master* é uma cópia reduzida do *slave*. Por movimentar seus *links*, o operador controla o movimento do *slave*. Tanto um quanto o outro, possuem potenciômetros de precisão de tal forma que o sistema de controle (KMC 9100) consegue acessar a informação angular proveniente de cada junta. Dois contra-pêso são utilizados para contrabalançar o *master* na elevação do ombro e no cotovelo (fig. 1.3). Mais detalhes sobre esses dispositivos serão fornecidos no próximo capítulo.

O Manutec r3 é um robô industrial fabricado pela Siemens e adaptado para



*Figura 1.2: Manipulador Kraft em operação na Unicamp.*

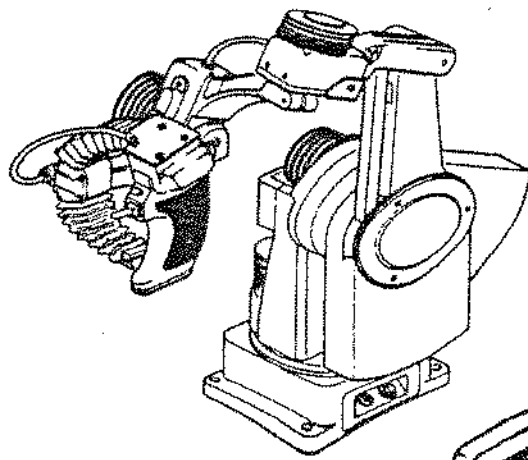


Figura 1.3: Manipulador Kraft - Master.

as condições submarinas pelo GKSS. Ao contrário do que se poderia supor, o Manutec r3, por ser um robô em escala industrial tem custo de implantação semelhante ao do manipulador Kraft, mesmo tendo características de performance melhores e também uma vasta quantidade de facilidades.

Como dito anteriormente, o projeto pretendia utilizar o robô Manutec ou o manipulador Kraft no sistema de intervenção submarina, dependendo da performance de cada um deles quando executando tarefas pré-definidas. Essa performance está diretamente ligada às características construtivas dos robôs, de forma que, a priori, o Manutec parecia levar uma certa vantagem visto que por possuir acionamento elétrico (Motores Brushless) ele é capaz de conseguir melhor precisão e repetibilidade, que é talvez, o principal atrativo quando se deseja automatizar tarefas. A figura 1.4 mostra esse robô executando uma operação.

Se isso não bastasse, o Manutec ainda contava com uma unidade de controle bastante sofisticada (Sirotec RCM), capaz de fazê-lo executar linhas retas em qualquer direção, interpolações circulares, mostrar a posição cartesiana da garra e muito mais, ao passo que ao Kraft restavam como características positivas sua maior capacidade de carga, o melhor comportamento de seus acionadores quando submetidos a ambientes submarinos e expostos a altas pressões e o fato de suas partes mecânicas já serem apropriadas para



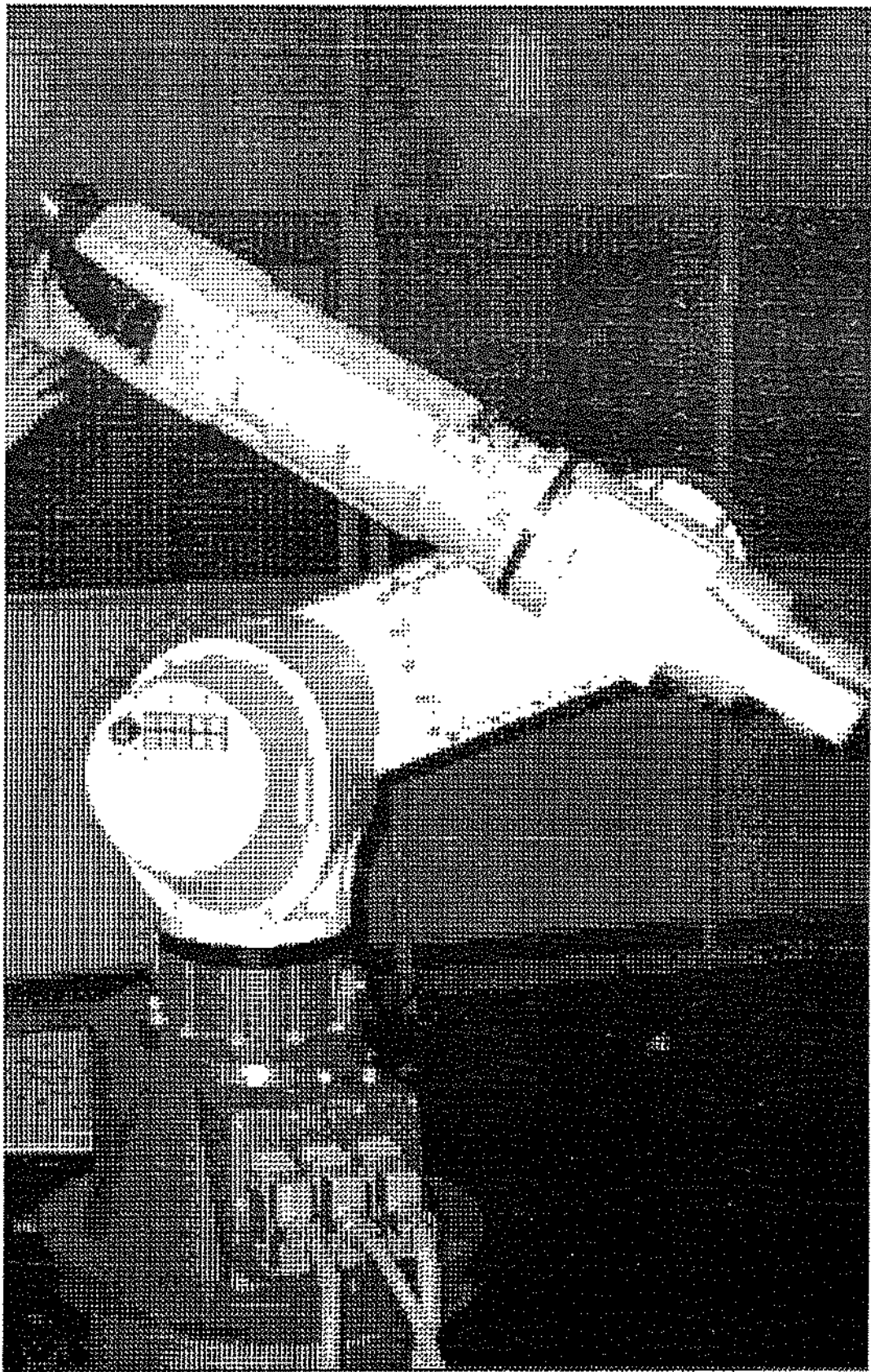


Figura 1.4: Robô Manutec r3 em operação na Unicamp.

o trabalho submarino. Porém, a necessidade de um operador para acioná-lo, a ausência de informações do sistema e a impossibilidade de programá-lo através de uma linguagem textual faziam do manipulador Kraft o menos indicado para o serviço. Contudo, era precoce qualquer prognóstico, uma vez que o sistema Kraft permitia grande acesso ao seu interior.

A tabela 1.1 faz um resumo comparativo das características do manipulador Kraft e do robô Manutec r3.

CARACTERÍSTICAS CONSTRUTIVAS		
	Manutec r3	Kraft
Acionamento	Elétrico (Brushless)	Hidráulico
Sensores de junta	Encoders óticos	Potenciômetros de precisão
Programação	Linguagem Textual	Via Master/Slave
Controlador	Sofisticado	Simple
Informações do Sistema	Muita	Pouca
Custo	Alto	Alto

Tabela 1.1: *Manutec × Kraft.*

Com base nas características mostradas na figura 1.1, pode-se citar as seguintes vantagens do Manutec com relação ao Kraft:

- Melhor precisão e repetibilidade;
- Exibição da posição instantânea da garra;
- Programação através de linguagem textual;
- Possibilidade de programação nos modos ponto-a-ponto, linear e circular;
- Aprendizagem via teclas;
- Sensores de junta mais precisos.

Obviamente, o manipulador Kraft também tem seus pontos fortes, que são citados abaixo comparativamente com o robô Manutec:

- Maior capacidade de carga;
- Melhor comportamento de seus acionadores quando submetidos ao ambiente submarino e expostos a altas pressões;
- Maior robustez;
- Partes mecânica já adaptada para funcionamento na água.

Considerou-se então a possibilidade de se tentar obter as características positivas de ambos os robôs e decidiu-se pelo desenvolvimento de um sistema de automação utilizando uma interface de comunicação com um microcomputador que tornasse possível a operação do manipulador de forma semelhante à do robô.

## 1.4 Estrutura Mecânica Construída

Para a realização dos testes de performance com os dois robôs, foi construída na Unicamp duas maquetes em escala-real (*Mock-up* - figura 1.5) de um cenário submarino típico (*Template-Manifold*) que permitirá a execução desses testes a seco na Unicamp e em águas rasas no GKSS (Alemanha) [Saramago,93].

Essas maquetes simulam parte de uma estrutura projetada pela Petrobrás, o OCTOS-1000<sup>TM</sup><sup>1</sup>, e serão instaladas no GKSS e na Unicamp para um programa conjunto de testes. A figura 1.6 apresenta um desenho descrevendo essa estrutura.

### 1.4.1 Base Móvel

O robô deverá trabalhar sobre uma base móvel e se deslocar através de trilhos solidários ao OCTOS-1000<sup>TM</sup>, até se posicionar em frente aos painéis colocados nas extremidades dos braços do mesmo. A figura 1.7 mostra o robô operando sobre sua base móvel. Após a fixação automática do conjunto, será possível a realização de tarefas automatizadas.

---

<sup>1</sup>OCTOS porque tem oito braços e 1000 porque pretende-se trabalhar a profundidades próximas de 1000m.

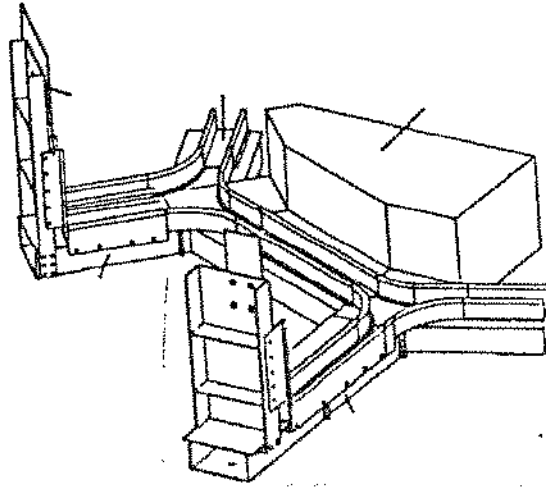


Figura 1.5: *Mock-up* construído na Unicamp para testes.

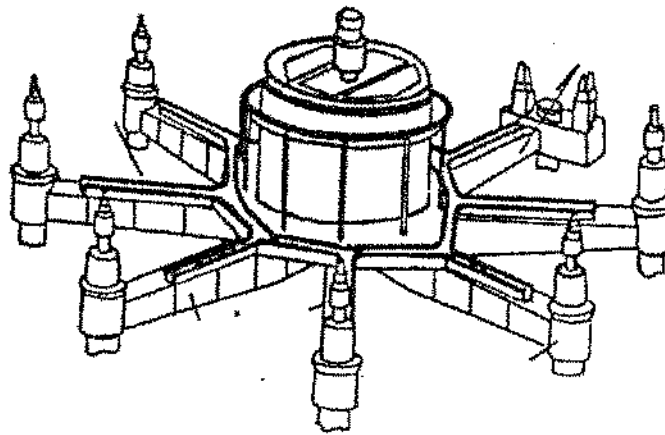


Figura 1.6: *OCTOS-1000™*. Plataforma na qual o sistema irá atuar.

## 1.5 Programação Off-line

O sistema desenvolvido é capaz de suportar dois tipos de programação de trajetória:

- On-line; (Utilizando o manipulador)

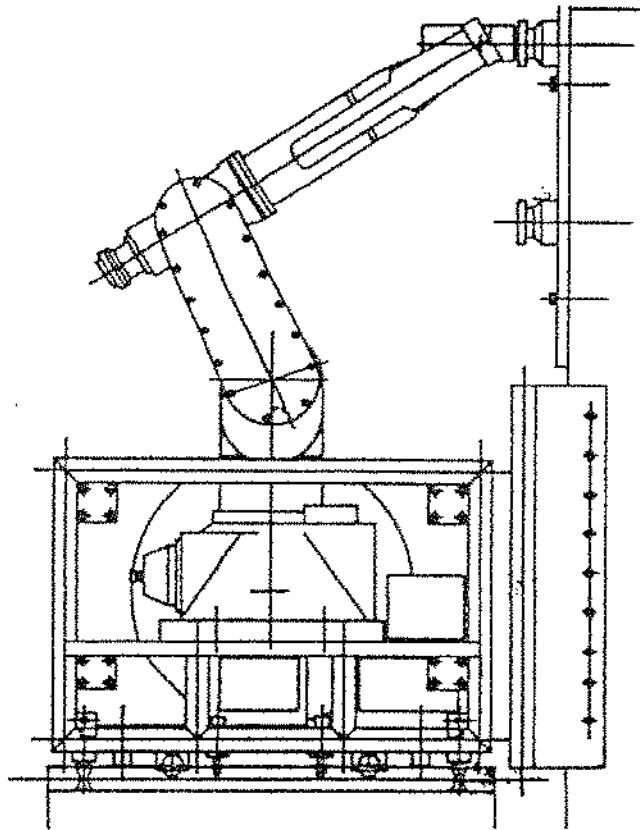


Figura 1.7: Robô Manutec r3 operando sobre a Base Móvel.

- Off-line. (Utilizando um software de simulação *off-line*)

Na programação *off-line*, utiliza-se um software de simulação e visualização que contém o modelo geométrico do robô, permitindo assim, a geração automática de trajetórias. Esse módulo foi idealizado para ser utilizado como uma ferramenta de suporte à geração de trajetória para o manipulador. Para a sua implementação foi necessário desenvolver também o modelo cinemático para o manipulador Kraft. Com um sistema de programação modular em CAD obtém-se trajetórias que podem ser enviadas diretamente ao controlador do manipulador. Este software apresenta as seguintes características:

- Utilização de material de baixo custo (computador);
- Fácil interação com o operador;

- Estrutura modular;
- Simulação e visualização de tarefas;
- Software completo de geração automática de trajetórias;
- Transmissão de dados do programa de visualização através das interface com o robô e com o manipulador via linha serial RS-232;
- Padronização dos arquivos de dados.

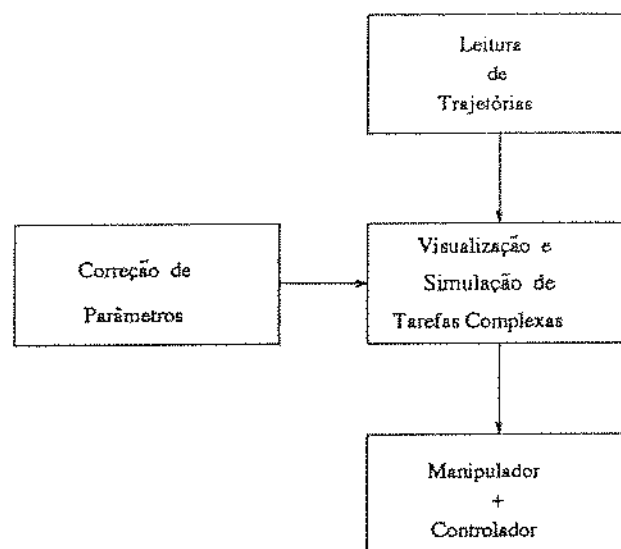


Figura 1.8: Estrutura da Programação off-line.

O pacote computacional de programação *off-line*, denominado SIMULA [Rosário,91] e mostrado na figura 1.8 consiste de um programa modular com as seguintes características:

- Criação de ferramentas e ambientes:

Um programa utilitário para a criação de arquivos contendo diferentes ferramentas e ambientes típicos. Uma imagem gráfica do cenário completo pode ser realizada;

- Simulação:

Um programa para simulação e visualização de um cenário completo de atuação, o qual contém o robô, a base móvel, ferramentas dedicadas, acessórios etc.

- Geração de Trajetórias:

Um programa utilitário para geração de arquivos de trajetórias compatíveis com o controlador do robô ou do manipulador. Além disso, esses arquivos sofrem uma “filtragem” levando-se em conta condições de velocidade e aceleração das juntas. Este módulo permite também a execução de testes de colisão;

## 1.6 Conclusões

Neste capítulo foi apresentado os principais objetivos para a automação de tarefas submarinas utilizando-se dispositivos robóticos. Ao mesmo tempo, apresentamos de forma sucinta, a infra-estrutura disponível para o desenvolvimento do projeto proposto.

Tendo em vista a necessidade da execução de testes tornou-se imperativo o projeto e a construção de uma interface homem-máquina entre o manipulador utilizado e um microcomputador.

O capítulo seguinte irá ater-se ao problema do desenvolvimento de um hardware e um software dedicados, capazes de tornar possível a automação do manipulador Kraft.

## Capítulo 2

# Interface Homem-Máquina

### 2.1 Introdução

No capítulo anterior foram apresentadas as principais linhas de desenvolvimento necessárias para a automação de tarefas em ambientes submarinos utilizando o robô Manutec e o manipulador Kraft num ambiente típico. O robô Manutec possui um alto grau de automação embarcada, pois é um sistema especialmente construído para trabalhar em linhas automatizadas como por exemplo, em aplicações automobilísticas. Entretanto, o manipulador Kraft, que foi concebido para trabalhar baseado em telepresença, não necessita para essas operações de um grau de complexidade em automação elevado.

Neste trabalho, estaremos preocupados em viabilizar o manipulador Kraft para as operações tradicionais de telepresença, bem como a realização de tarefas automatizadas a partir de trajetórias pré-planejadas.

Para tornar factíveis as implementações desejadas no sentido de aumentar seu nível de automação, como por exemplo um módulo de geração de trajetórias ou um módulo de programação *off-line*, era de fundamental importância a construção de uma interface de comunicação entre o seu controlador e um microcomputador.

Este capítulo mostra o desenvolvimento dessa interface de comunicação, que tem



como objetivo imediato, proporcionar ao operador uma melhor interação com o sistema, por fornecer um série de informações úteis, tais como, os sinais on-line das juntas do manipulador, a posição espacial de seu elemento terminal, o modo de operação corrente e principalmente possibilitar a gravação e o envio trajetórias.

Uma vez implementada, essa interface juntamente com os módulos citados no capítulo anterior dariam ao manipulador, no mínimo as mesmas condições do robô, ficando como diferença entre eles apenas suas características construtivas.

## 2.2 Interface com o Robô Manutec R3

A interface do robô Manutec (RCM) permite três formas de comando: Via Painel de Operação, *Teach-in-Box* e *off-line*. A programação *off-line* é realizada através de um software de comunicação entre duas interfaces RS-232 existentes no controlador RCM do robô e num microcomputador compatível com a linha IBM-PC-AT, tornando-se, assim possível a geração de trajetórias a partir do programa (*off-line*) de simulação e visualização descrito no capítulo anterior. Considerando que este software foi desenvolvido pela SIEMENS, e é dedicado ao controlador RCM, não há a disponibilidade de informações sobre protocolos de comunicação.

Um dos inconvenientes desse método de geração de trajetória cujo esquema é mostrado na figura 2.1 é a sua característica *off-line*, que deverá ser suprimida pela versão a ser desenvolvida para o manipulador Kraft.

## 2.3 Interface com o Manipulador KRAFT

Como visto na seção anterior, o robô Manutec R3 conta com uma unidade de controle (Sirotec RCM) através da qual pode-se programar, operar ou monitorar tarefas. O manipulador Kraft, cuja concepção industrial era apenas para funcionamento no modo master-slave (teleoperação), não possuía os mesmos recursos de interfaceamento homem-máquina, não informando por exemplo, o posicionamento e orientação da garra, nem os ângulos das juntas. Limitava-se apenas à execução de uma bateria de *self-tests* durante

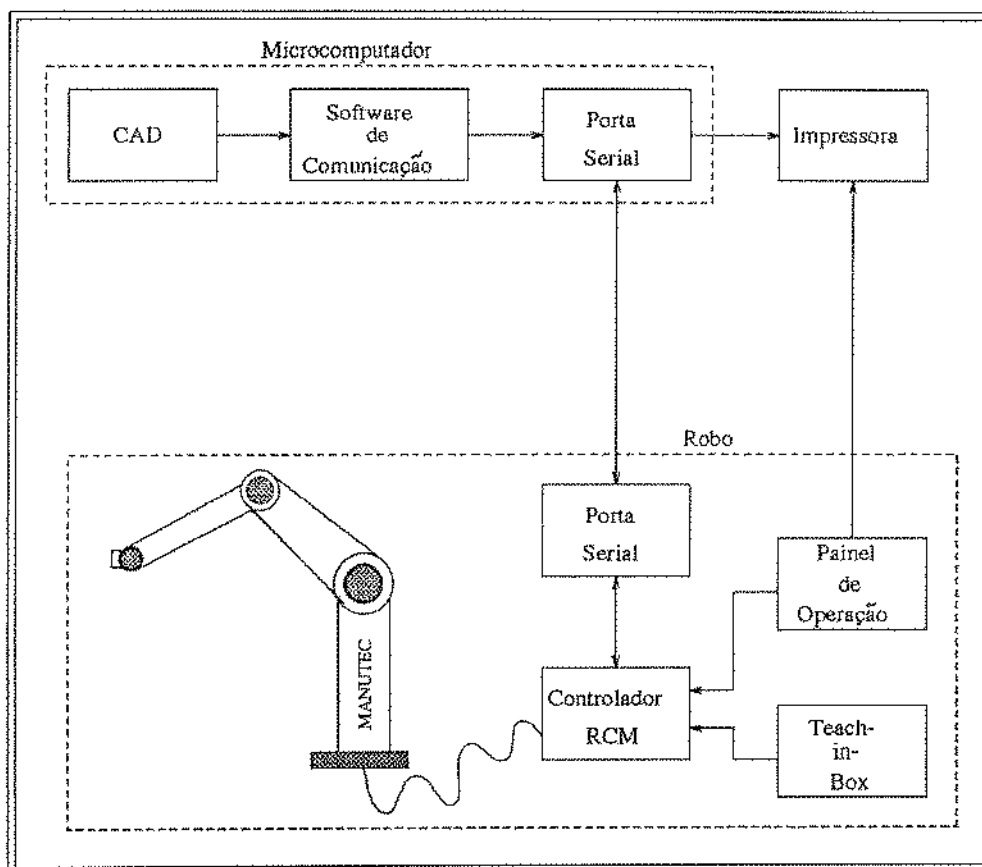


Figura 2.1: Esquema de Programação off-line no robô Manutec.

o processo de inicialização do sistema, com a intenção de detectar possíveis problemas a nível de hardware, como por exemplo, conexões mal-feitas ou soltas, níveis de tensão fora da faixa especificada e outros.

Na configuração inicial o sistema era composto pelas seguintes unidades:

- Slave
- Master
- KMC 9100

Esses componentes são mostrados na figura 2.2 e descritos abaixo [Kraft,85]:

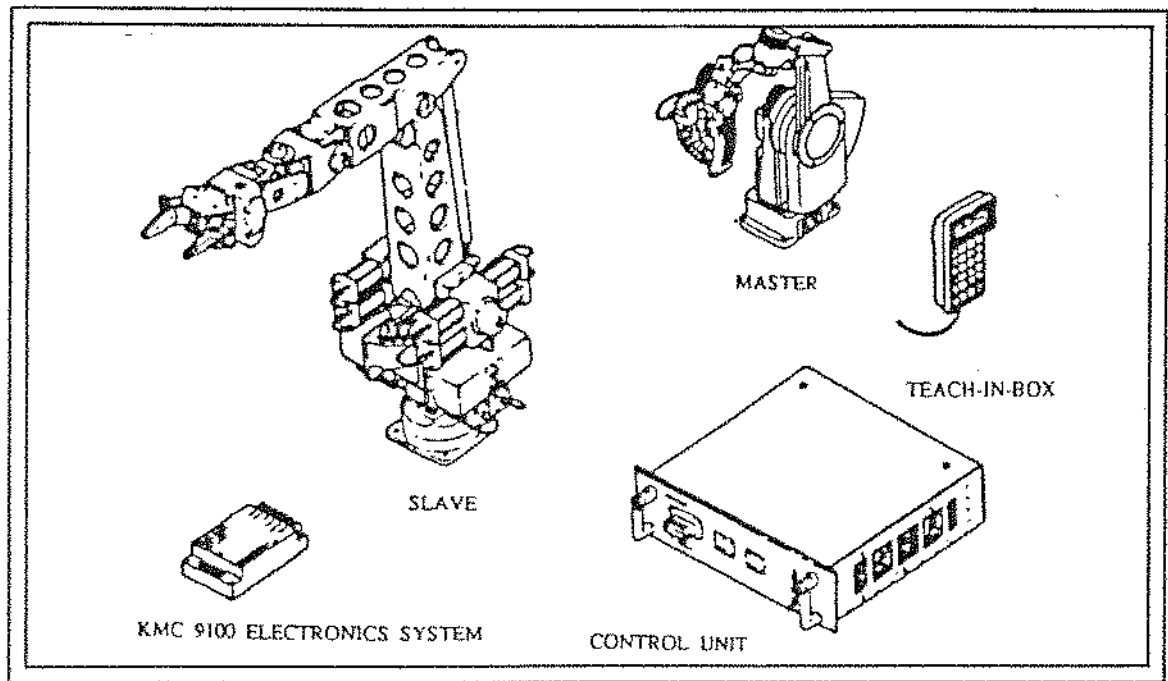


Figura 2.2: Sistema Master-Slave Kraft Grips.

O Slave é o manipulador propriamente dito, o elemento que, de fato, executa as tarefas. Ele é um braço eletrohidráulico com seis graus de liberdade mais a função de abrir/fechar a garra. O acionamento das juntas e da garra são realizados a partir de atuadores hidráulicos, acionados por servo-válvulas, num total de sete. Além dessas válvulas para controle do atuador, existem ainda uma válvula de redução de pressão e uma válvula solenóide. Seis potenciômetros de precisão acoplados aos atuadores do slave fornecem a informação de posição de cada junta para uma unidade de controle eletrônico.

O Master cinematicamente é similar ao *slave* tendo três "links" responsáveis pelo posicionamento e três movimentos para orientação da garra. Em sua extremidade existe uma empunhadura, que permite o operador manuseá-lo para executar uma determinada tarefa. Assim como no slave, existem potenciômetros de precisão que provêm a informação de posição e orientação do master. Existe também, em sua base, um conjunto de leds que informam o acionamento de algum sinal digital.

O KMC 9100 é um sistema de controle baseado em um microprocessador e projetado para prover o controle eletrônico e as funções de telemetria necessárias para a

operação do manipulador. ele é composto basicamente de três componentes [Kraft,85]:

- O módulo de controle do sistema do master (MSC), que corresponde à principal unidade de controle do manipulador, é responsável por receber os sinais dos potenciômetros do master e enviá-los em forma serial para a unidade Driver Remoto (RSD). Esta unidade executa uma série de *self-tests* tais como teste de RAM, PROM, processador, timer, co-processador matemático, interrupção, comunicação, alimentação e outros;
- O módulo driver remoto (RSD) localizado próximo do slave. O RSD interfacia diretamente com o slave e remotamente com o master através do MSC;
- um terminal de mão (HT) que provê entrada de dados e um display. Esta é a interface do operador com o KMC 9100 para operar suas capacidades não controláveis através do master. Todos os ajustes do sistema, com exceção do controle de força são feitos via HT, sem a necessidade de se acessar o chassis da eletrônica Kraft (KMC).

A figura 2.3 mostra como era feita a comunicação entre o master e o slave. Sem a presença de um microcomputador, os recursos para interação humana eram bastante limitados.

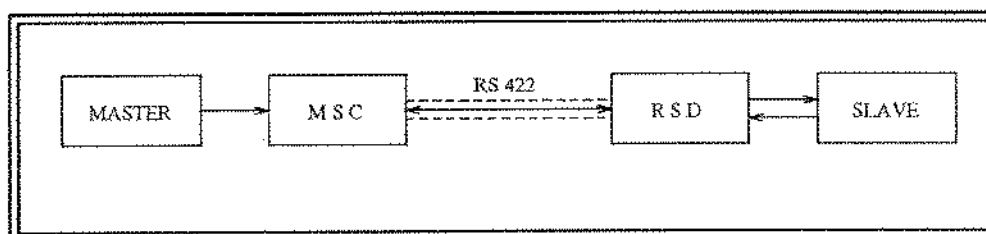


Figura 2.3: Configuração inicial do sistema.

Observe que seu modo básico de operação consistia em fazer o slave repetir o movimento executado pelo operador através do master, não permitindo, por exemplo, o acionamento de uma única junta por vez. Foi então necessário tornar o sistema Kraft mais amigável e menos susceptível aos movimentos imprecisos do operador.

A interface homem-máquina desenvolvida para o manipulador consiste de um microcomputador tipo IBM-PC-AT, um hardware eletrônico que faz a comunicação entre o manipulador e o micro e um algoritmo de controle e monitoramento. Ela é capaz de pôr o sistema em três diferentes modos de funcionamento (figura 2.4). O operador pode escolher uma das seguintes opções:

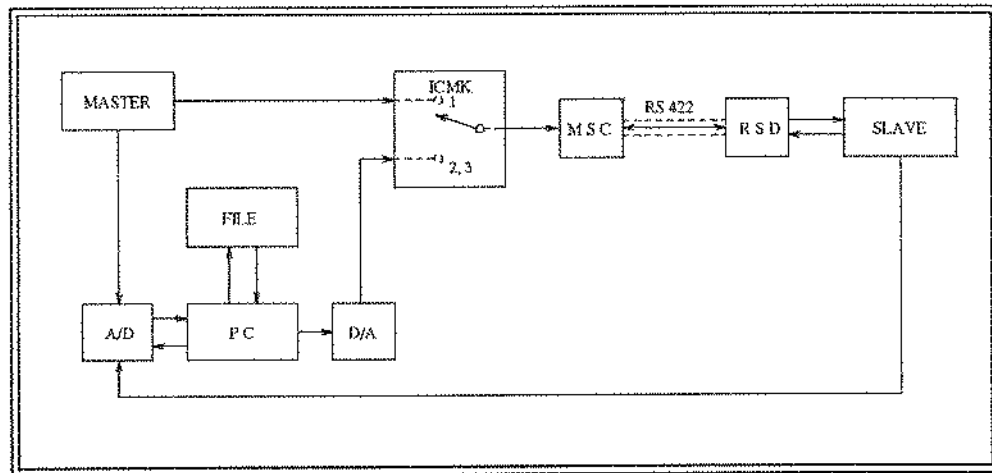


Figura 2.4: Configuração atual do sistema.

- MODO 1 - *Monitoramento*: Modo de controle remoto do manipulador. O operador controla o slave através do uso do master;
- MODO 2 - *Controle*: Modo automático. O operador controla o slave através do teclado ou permite um controle diretamente através do micro por executar um arquivo de pontos para uma tarefa pré-programada;
- MODO 3 - *Simulação de Vôo*: O operador, no momento em que está trabalhando com o manipulador através do master, é submetido a uma simulação do movimento do R.O.V. ( Remote Operated Vehicle). Isto permitirá acostumá-lo às condições adversas inerentes às operações submarinas.

Estando o sistema endereçado para *monitoramento*, os sinais analógicos e digitais do master (potenciômetros e chaves) são lidos pela unidade MSC. Caso algum sinal digital esteja ativo, este enviará comandos para o acionamento dos leds correspondentes.

Os sinais analógicos das juntas são convertidos para a forma serial e enviados sob protocolo RS-422 para o RSD, onde são reconvertidos para a forma paralela, e submetidos a uma estratégia de controle que decide então como agir para controlar o movimento do slave. A comunicação física entre o MSC e o RSD é feita através da utilização de par trançado, mas o sistema também suporta a utilização de cabo coaxial e fibra ótica. A figura 2.5 mostra a interface desenvolvida chaveando o sistema no modo *monitoramento*.

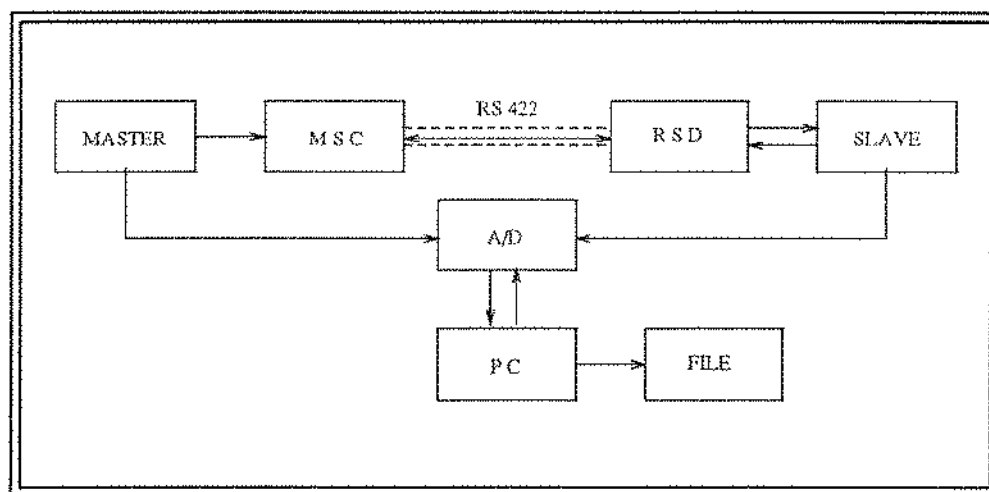


Figura 2.5: *Modo Monitoramento*.

Nesse modo o computador tem disponível a todo momento, através do conversor A/D, os sinais do master e do slave, de modo que se pode fazer visualização gráfica dos sinais de ambos, geração de trajetória, exibição da posição cartesiana e dos valores angulares das juntas, animação gráfica etc.

Se o sistema estiver endereçado para *controle* ou *simulação de voo*, o master não mais estará ligado ao MSC, mas somente ao microcomputador. No modo *controle* os sinais do master não estarão mais sendo lidos — de fato, neste modo, o master não estará sendo utilizado —. O micro assume a sua função e passa a controlar o slave através da utilização do teclado, onde pode-se acessar individualmente qualquer junta do manipulador, ou através da leitura de um arquivo de pontos. Esses sinais de controle são levados via conversor D/A para o MSC, que dá continuidade ao processo. Neste modo, pode-se fazer também visualização gráfica, exibição das coordenadas cartesianas da garra e das coordenadas articulares, animação gráfica e envio de trajetória. A figura 2.6 mostra

a ligação do sistema no modo *controle*.

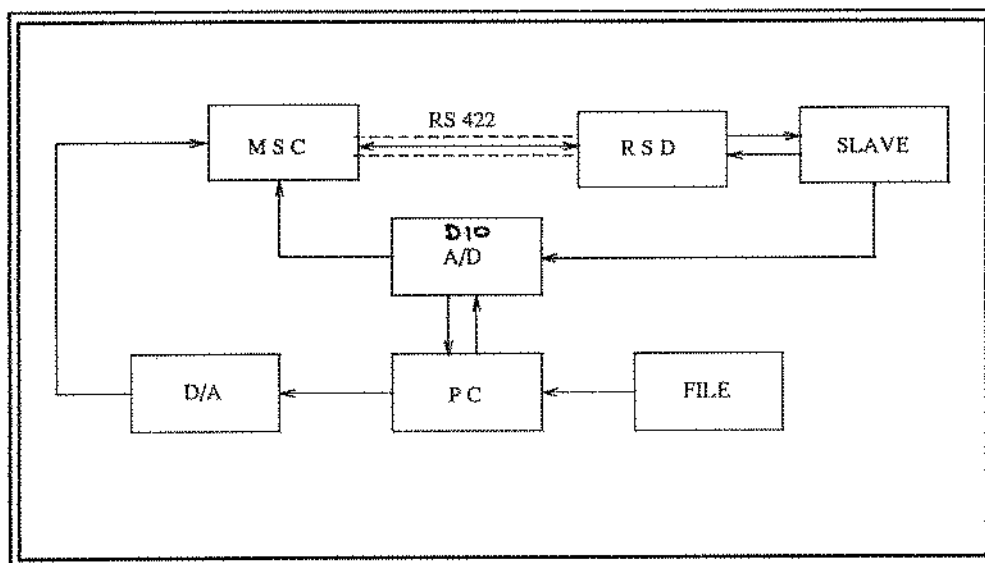


Figura 2.6: *Modo Controle*.

Durante a execução de tarefas submarinas, o manipulador é conduzido por um veículo. Esse veículo faz alguns movimentos devido ao movimento da água. É muito difícil para o operador levar o slave à posição desejada. O modo *simulação de vôo* tem como objetivo, acostumar o operador com as intempéries do ambiente real de trabalho do equipamento, pois ele simula as forças externas devido às correntes submarinas agindo sobre o slave. O que se faz é adicionar ao sinal enviado pelo master, um arquivo de pontos senoidais de baixa frequência, na intenção de fazer com que o operador tenha maior dificuldade para controlar o manipulador. Todos os recursos disponíveis nos modos anteriores, com exceção da geração de trajetória, também estão presentes neste modo. A figura 2.7 mostra o sistema sendo chaveado no modo *simulação de vôo*.

## 2.4 O Hardware

Para que o micro tivesse acesso aos sinais das juntas do manipulador Kraft, e de forma a tornar possível a operação do sistema com os recursos de monitoramento, controle e simulação de vôo, era necessário adicionar um hardware ao sistema original.

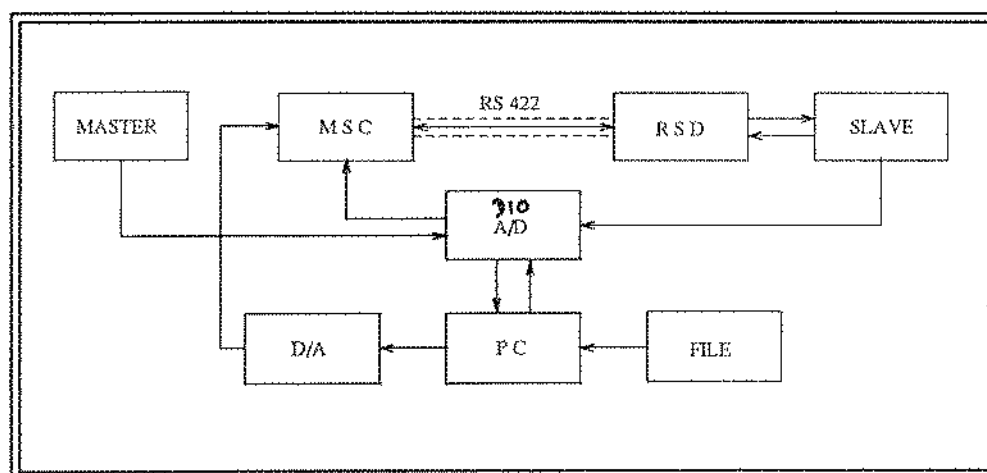


Figura 2.7: Modo Simulação de Voo.

Basicamente os três elementos instalados foram:

- Uma placa de conversão analógico para digital (CAD 12/36);
- Uma placa de conversão digital para analógico (CDA 12/08);
- Uma placa multiplexadora analógica e digital (ICMK).

A conversão A/D era necessária para que o micro pudesse “ler” os sinais analógicos do master e do slave, e a conversão D/A, para possibilitar o envio dos sinais de volta ao sistema. Algumas entradas e saídas digitais também foram necessárias para o tratamento dos sinais digitais provenientes dos botões de controle do Kraft.

A interface de chaveamento de modos (ICMK)<sup>1</sup>, como o próprio nome já diz, tem a finalidade de colocar o sistema num dos três modos de funcionamento descritos na seção anterior. De forma mais sucinta, o que o ICMK faz é interligar ou o master ou o micro ao SMC. Eletronicamente, esse dispositivo é um conjunto de multiplexadores analógicos e digitais controlados através do micro. O esquema eletrônico e mais detalhes sobre essa placa encontram-se no apêndice B.

<sup>1</sup>Esta interface foi projetada e construída na Unicamp especificamente para o propósito de viabilizar a automação do manipulador Kraft.



A placa CAD 12/36 [Lynx,89] é uma interface de expansão que permite integrar o uso de microcomputadores compatíveis com a família IBM PC (XT e AT) ao meio ambiente externo. Suas características lhe dão a capacidade de interfacear o micro a diversos sinais, sejam eles analógicos ou digitais, de baixa ou alta frequência.

Essa placa possui basicamente os seguintes recursos:

- Conversor analógico-digital (A/D) com 12 bits de resolução, para a leitura dos sinais analógicos;
- Entradas e saídas digitais que permitem a leitura e o acionamento de variáveis digitais;
- Base de tempo interna e contadores que permitem temporizar as operações do sistema;
- Expansão para conexão de subsistemas, geralmente conversores D/A, que podem ser usados para produzir sinais analógicos de estímulo, controle, set-point ou geração de formas de onda.

A CDA 12/08 é uma placa de expansão para microcomputadores compatíveis com IBM PC (XT e AT) que possui as seguintes funções [Lynx,91]:

- Até oito saídas independentes de conversão Digital-Analógica com resolução de 12 bits;
- Suporte a interrupções;
- Três contadores programáveis para geração de interrupção e contagem de eventos;
- Oscilador a cristal de 2MHz para uso dos contadores.

No apêndice B encontram-se informações mais detalhadas a respeito dessas duas placas.

## 2.5 O Software

Um algoritmo para controlar o sistema teve que ser desenvolvido e implementado. Este software visava fornecer ao usuário o maior número de recursos possíveis, exigindo dele o mínimo de esforço para operá-lo. Visto que o sistema deveria controlar e monitorar diversas interfaces, a linguagem escolhida para o desenvolvimento do programa foi C++, por se caracterizar como um linguagem de nível médio, isto é, ela possui tanto os recursos das linguagens de alto nível como Pascal ou Basic como os recursos e funcionalidades do Assembly [Schildt,90][Ker,78][Kelley,89]. A figura 2.8 mostra as principais funções do software de controle do sistema, e elas são descritas abaixo:

- **Teste de Hardware:** Quando essa função é selecionada, o software executa uma série de *self-tests* nos dispositivos que constituem o hardware. Toda vez que um *selftest* falhar, o sistema vai indicar onde ocorreu a falha. A última versão dessa função executa basicamente três tipos de teste:
  1. Teste dos conversores A/D e D/A;
  2. Teste das entradas e saídas paralelas;
  3. Teste de chaveamento dos modos de operação.
- **Calibração:** Essa função estabelece uma relação angular entre os potenciômetros do master e os potenciômetros do slave, de tal forma que a posição estabelecida para o master resulte na posição desejada para o slave.
- **Operação:** Nessa função o operador escolhe um dos seguintes três modos de operação:
  - **Controle:** Essa função permite o operador escolher uma das possibilidades abaixo:
    1. Enviar uma trajetória previamente gerada para o slave;
    2. Comandar o slave através do teclado do micro
  - **Monitoramento:** Essa função executa a aquisição dos sinais das juntas do master e do slave e em tempo-real plota esses sinais na tela de diferentes formas:

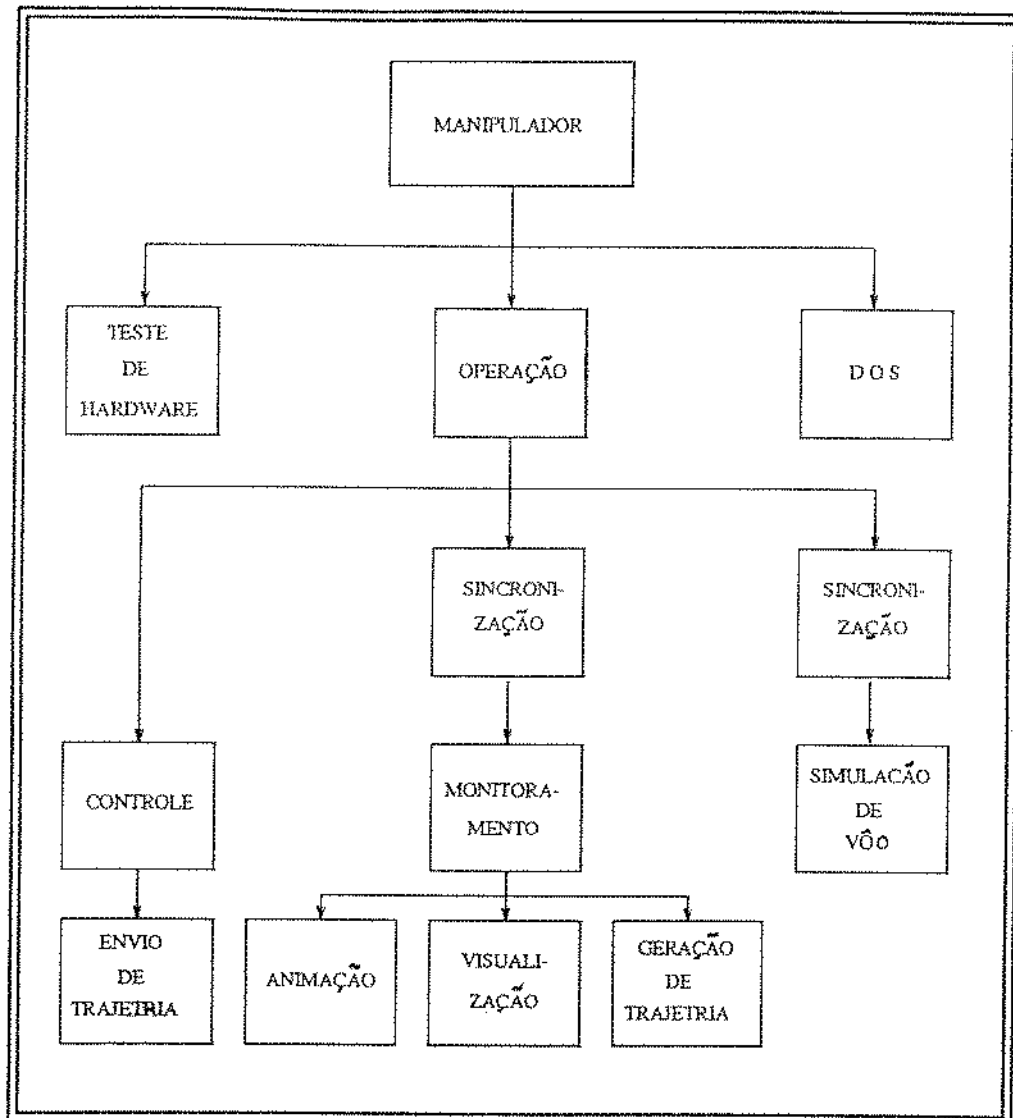


Figura 2.8: Esquema do Software de Controle.

1. Gráfico do deslocamento, velocidade e aceleração de cada junta;
2. Animação gráfica 3D do master e do slave;
3. Exibição da posição cartesiana real da ferramenta (através do uso do modelo geométrico direto do manipulador);
4. Status dos sinais digitais do sistema.

Essa função também tem implementado o sistema de geração de trajetória, que é o motivo fim deste trabalho.

- **Simulação de Vôo:** Essa função chaveia o sistema no modo 3, que é responsável por simular o movimento do veículo (R.O.V.) durante a execução de uma tarefa. Seu objetivo é acostumar o operador com as condições reais de trabalho.
- **Sincronização:** Em um manipulador do tipo master/slave há a necessidade inerente ao sistema de se levar o master e o slave a uma posição correspondente antes que a potência hidráulica seja aplicada. Caso contrário o manipulador, que faz parte de uma malha de controle de posição, teria que executar um movimento abrupto, possivelmente em todas as juntas, de forma a corrigir o erro de posição e entrar em sincronismo com o master. Visto que a trajetória executada para essa correção de posicionamento é desconhecida, o movimento poderia causar danos ao equipamento existente dentro do volume de trabalho e ao próprio manipulador.

A necessidade dessa função se deve por exemplo ao fato que quando o operador deseja mudar do modo *monitoramento* para o modo *controle*, no qual o slave passa a ser comandado a partir do teclado do micro, é necessário que este (o micro) saiba exatamente que sinais enviar. Para resolver esse problema, uma estratégia bastante simples e eficiente foi adotada. O micro lê os sinais das juntas do slave (disponíveis a todo momento), converte esses sinais para os correspondentes do master e envia estes últimos de volta ao slave. Neste ponto, o micro passa a funcionar como master.

Para fazer o caminho inverso, isto é, voltar do modo controle para o modo monitoramento, o sistema perde o sincronismo, visto que o slave sofreu modificações em seu posicionamento através da utilização do teclado do micro, enquanto que o master permaneceu parado. Uma tela mostrando setas direcionais para cada junta do manipulador ajuda o operador a conseguir novamente o sincronismo entre os dois.

- **DOS:** Essa opção sai do programa de controle e retorna ao sistema operacional.

O software foi desenvolvido de forma a ser bastante amigável. Todas as funções aparecem na tela na forma de menus do tipo *pop-up*. A seleção é feita através das teclas de movimento do cursor. Existem algumas rotinas internas às quais o operador não tem acesso, que estão relacionadas com características básicas, tais como a inicialização das placas de conversão A/D e D/A, o envio de dados para algum dispositivo e assim por diante.

## 2.6 Conclusões

Neste capítulo foi abordado o projeto e implementação de uma interface de comunicação entre um microcomputador e um manipulador submarino, necessária para tornar possível a automação de tarefas a partir desse sistema. Agora, pode-se pensar no desenvolvimento de softwares de aplicação e apoio para otimizar as tarefas a serem realizadas.

Porém, para a realização dessas facilidades, é necessário não somente o desenvolvimento dessa interface como também o conhecimento dos modelos geométrico e cinemático inverso. O capítulo seguinte será dedicado ao problema de modelagem do manipulador Kraft e do robô Manutec.

## Capítulo 3

# Modelagem do Manipulador Kraft e do Robô Manutec r3

### 3.1 Introdução

A automação de tarefas utilizando sistemas robotizados exige a elaboração de um modelo geométrico que permitirá a evolução desse sistema dentro de seu volume de trabalho com a precisão exigida.

Na maioria das aplicações industriais, a realização de tarefas está relacionada com o tipo de ferramenta utilizada, a partir de um sistema de coordenadas cartesianas fixo à base do robô. Em aplicações do tipo CAD ou controle em relação a um sistema de referência solidário à ferramenta de trabalho, necessitamos do conhecimento das velocidades e deslocamentos angulares da cada junta.

Para isso torna-se necessário a medida dessas grandezas em sistemas de coordenadas locais a cada junta, e a partir de transformações geométricas elementares efetuar o controle de posição e orientação do elemento terminal através do envio de informações aos atuadores elétricos, pneumáticos ou hidráulicos das juntas (modelo geométrico e análise cinemática).

Este capítulo tem como objetivo a realização de uma metodologia para análise geométrica e cinemática do robô industrial Manutec r3 e do manipulador submarino Kraft, visando o cumprimento dos objetivos propostos no início do trabalho.

## 3.2 Posicionamento do Problema

Para a realização de um sistema de controle de trajetória, é necessário o conhecimento antecipado da solução do problema de transposição direta, ou seja, uma relação que forneça a posição e orientação espacial da garra para uma dada configuração de juntas, assim como a solução do problema inverso, que gera os ângulos de junta correspondentes a uma determinada configuração espacial.

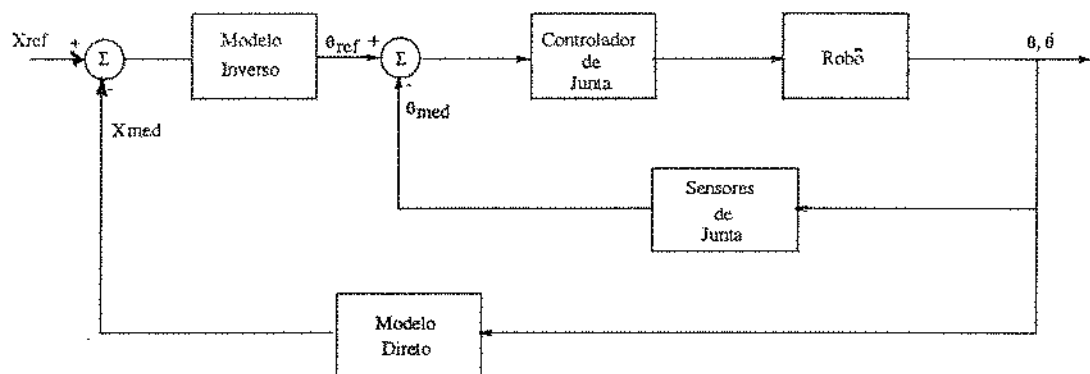


Figura 3.1: Diagrama de blocos de um controlador de trajetória.

Observando a figura 3.1, que mostra o esquema simplificado de uma malha de controle de trajetória no espaço cartesiano, pode-se perceber a necessidade dos modelos direto e inverso do robô.

## 3.3 Modelo Geométrico

Dois sistemas de coordenadas são utilizados para a descrição da posição e orientação de um manipulador no interior de seu volume de trabalho: coordenadas angulares e coordenadas cartesianas de cada *link*.

A modelagem geométrica modela o robô estaticamente, não levando em consideração fatores como tempo e velocidade. Para aplicações de controle, no entanto, são necessários modelos onde a relação destes fatores com o comportamento do robô sejam levados em consideração.

A modelagem cinemática trata do movimento das juntas independentemente das forças que o causam, referindo-se a todas as propriedades baseadas na geometria e no tempo. A descrição cinemática de um manipulador consiste no estabelecimento de uma relação matemática entre as coordenadas angulares e as coordenadas cartesianas de cada *link*.

O principal objetivo de um manipulador é trabalhar com seu elemento terminal (ou dispositivo porta-ferramentas), ou seja, o elemento terminal é a parte do manipulador que se interfaceia fisicamente com o ambiente de operação. Para executar uma dada tarefa, o manipulador deve saber onde o objeto a ser manipulado está localizado e qual a orientação que o elemento terminal deve ter em relação a este objeto. Para este propósito torna-se necessário o estabelecimento de um modelo completo para o manipulador.

O modelo geométrico direto consiste numa relação onde se obtém, a partir do conhecimento das variáveis articulares das juntas, a posição e orientação de cada *link*. Consequentemente a descrição da posição e orientação do efetuador em relação ao referencial da base será dada por:

$$\bar{X} = F(\bar{\Theta}) \quad (3.1)$$



onde:  $\bar{\Theta} = [\theta_1, \theta_2, \dots, \theta_n]^t$  é o vetor posição angular das juntas;

$\bar{X} = [x, y, z, \psi, \theta, \phi]^t$  é o vetor posição/orientação da ferramenta.

Trajetórias podem ser planejadas no sistema de coordenadas articulares ou em termos da posição e orientação da ferramenta. Como os acionadores requerem como sinal de referência uma coordenada angular, quando uma trajetória é planejada em termos de posição e orientação da ferramenta, a mesma deverá sofrer uma transformação cinemática inversa de coordenadas do tipo:

$$\bar{\Theta} = F^{-1}(\bar{X}) \quad (3.2)$$

Devido à característica fortemente não-linear da equação acima, sua solução pode ser difícil através de métodos analíticos, e além disso, quando encontrada, pode não ser única. Portanto, há a necessidade da utilização de algoritmos que escolham uma única solução da equação 3.1, em função dos limites físicos do robô (ranges de  $\theta_i$ ) e configurações geométricas ótimas (mais estáveis).

### 3.3.1 Transformação Homogênea

Matrizes de transformação homogênea permitem a passagem entre dois sistemas de coordenadas a partir da definição da posição e orientação entre esses dois sistemas. Elas podem ser expressas em matrizes 4 x 4 da seguinte forma:

$$T_{i-1,i} = \begin{bmatrix} n_x & s_x & a_x & \vdots & p_x \\ n_y & s_y & a_y & \vdots & p_y \\ n_z & s_z & a_z & \vdots & p_z \\ \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & \vdots & 1 \end{bmatrix} = \begin{bmatrix} & & \vdots & \\ & \bar{O}(t) & \vdots & \bar{P}(t) \\ & & \vdots & \\ \dots & \dots & \dots & \dots \\ & 0 & \vdots & 1 \end{bmatrix} \quad (3.3)$$

onde:

$T_{i-1,i}$  é a matriz de transformação de coordenadas do sistema  $i - 1$  com relação ao sistema  $i$ .

$\tilde{O}(t)$  é a matriz 3x3 dos cossenos diretores do sistema de coordenadas  $i$  expresso em relação ao sistema de coordenadas  $i - 1$ .

$\tilde{P}(t)$  é o vetor 3x1 do sistema de coordenadas  $i$  expresso em relação ao sistema de coordenadas  $i - 1$ .

As componentes que expressam a posição são descritas pelo vetor posição  $\tilde{P}(t)$  e a orientação pelos três versores ortonormais  $\tilde{n}(t)$ ,  $\tilde{s}(t)$  e  $\tilde{a}(t)$ , também chamados de vetores *normal*, *slide* e *approach*. Todos esses vetores são definidos em relação ao sistema de coordenadas da base (figura 3.2).

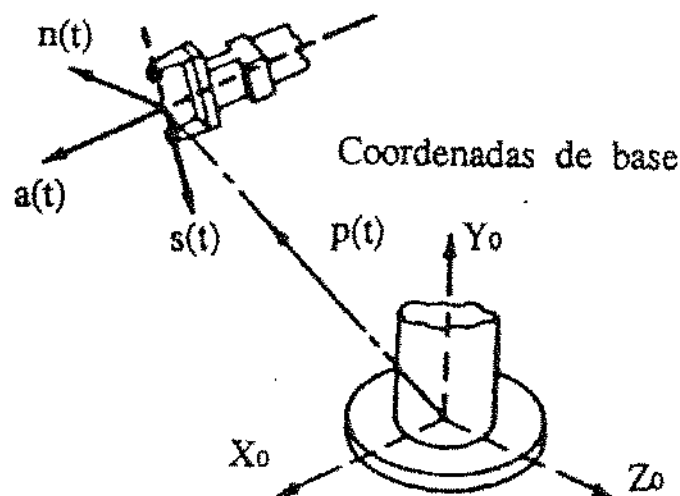


Figura 3.2: Vetor posição e orientação da garra.

### 3.3.2 Descrição da Matriz de Orientação Através de Ângulos

Para a implementação de algoritmos de controle, é essencial descrevermos o posicionamento do elemento terminal do manipulador num instante  $t$  em coordenadas

cartesianas expressas em relação à sua base.

O uso dos termos  $n$ ,  $s$  e  $a$ , como vetores de direção para descrever a orientação do punho, é, as vezes, difícil. Para comodidade dos cálculos torna-se necessário expressarmos a matriz de orientação, composta pelos componentes dos vetores  $\bar{n}$ ,  $\bar{s}$  e  $\bar{a}$  em termos de três ângulos  $A$ ,  $B$ ,  $C$ . Isto implicará que o posicionamento do elemento terminal do manipulador será descrito por um vetor final de seis dimensões representado por:  $[p_x, p_y, p_z, A, B, C]^t$ . Essa representação torna mais simples a utilização da informação de posição e orientação do manipulador para efeito de controle cartesiano.

### 3.3.2.1 Ângulos de Euler

- Obtenção da Matriz de Euler a partir da definição de 3 ângulos

A matriz de orientação pode ser definida em termos dos ângulos de Euler em relação ao sistema de coordenadas da base. Dessa forma, basta efetuarmos três rotações sucessivas em relação aos eixos  $Z(\psi)$ ,  $Y(\theta)$  e  $Z(\phi)$  (veja figura 3.3), obtendo-se uma matriz correspondente a essa transformação de coordenadas, ou seja:

$$EULER(\psi, \theta, \phi) = ROT(z, \psi) \cdot ROT(y, \theta) \cdot ROT(z, \phi)$$

$$EULER(\psi, \theta, \phi) = \begin{bmatrix} c\psi c\phi - s\psi c\theta s\phi & -c\psi s\phi - s\psi c\theta c\phi & s\psi s\theta \\ s\psi c\phi + c\psi c\theta s\phi & -s\psi s\phi + c\psi c\theta c\phi & -c\psi s\theta \\ s\theta s\phi & s\theta c\phi & c\theta \end{bmatrix} \quad (3.4)$$

- Problema Inverso - Obtenção de 3 ângulos de Euler a partir da matriz de orientação

Como foi mostrado anteriormente, o modelo geométrico direto de um robô fornecerá um vetor posição e uma matriz de orientação de sua garra expresso em relação

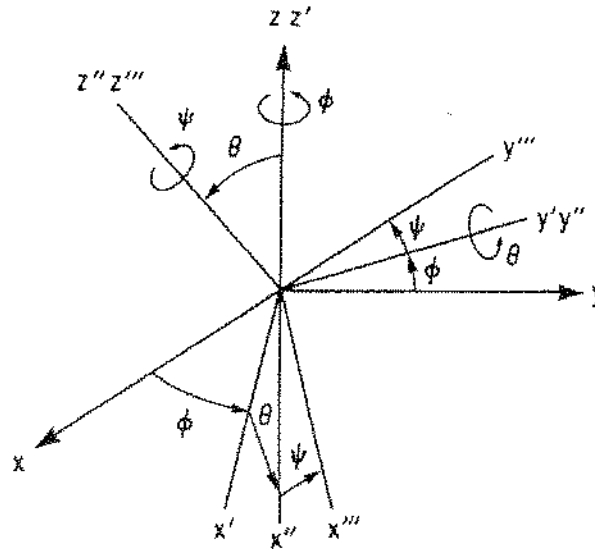


Figura 3.3: Definição dos Ângulos de Euler.

ao sistema de coordenadas da base. A matriz de orientação poderá ser expressa através dos três ângulos de Euler ( $\psi, \theta, \phi$ ). Para que isto seja possível, é necessário definirmos a função ATAN2, que tem por finalidade garantir a unicidade de solução. Comparando-se a matriz de Euler (equação 3.4) com a matriz de orientação final (equação. 3.3) obtemos os três ângulos de Euler:

$$\psi = ATAN2 \left[ \begin{array}{c} a_x \\ -a_y \end{array} \right]$$

$$\theta = ATAN2 \left[ \begin{array}{c} s\psi a_x - c\psi a_y \\ a_z \end{array} \right] \quad (3.5)$$

$$\phi = ATAN2 \left[ \begin{array}{c} -c\psi a_x - s\psi a_y \\ c\psi n_x + s\psi n_y \end{array} \right]$$

- Definição da função ATAN2

$ATAN2(x, y)$  calcula  $\tan^{-1}(y/x)$ , mas usa tanto o sinal de  $x$  como o de  $y$  para determinar o quadrante no qual o ângulo resultante está. A figura 3.4 resume esta definição.

$$\theta = ATAN2 = \left[ \frac{x}{y} \right] = \begin{cases} 0 \leq \theta \leq 90, & \text{com } +x, +y \\ 90 \leq \theta \leq 180, & \text{com } -x, +y \\ -180 \leq \theta \leq -90, & \text{com } -x, -y \\ -90 \leq \theta \leq 0, & \text{com } +x, -y \end{cases}$$

Figura 3.4: Definição da função  $ATAN2$ .

### 3.3.2.2 Ângulos Roll, Pitch, Yaw

Uma outra técnica para descrever a orientação do robô a partir de três ângulos, envolve o uso da representação *ROLL*, *PITCH* e *YAW* do punho. Estes são os três graus de liberdade possíveis, associados ao movimento do punho, ou, mais precisamente, os ângulos associados a esses graus de liberdade. São respectivamente, os ângulos de rotação do punho ao redor dos eixos X, Y e Z da conexão orgão terminal ao punho.

Alguns robôs industriais, por exemplo o robô Manutec r3, utilizam os ângulos Roll, Pitch, Yaw para expressar a orientação do elemento terminal em relação às coordenadas da base. Para isto basta efetuar três rotações sucessivas em relação aos eixos X(R), Y(P) e Z(Y), obtendo-se uma matriz correspondente a essa transformação de coordenadas, ou seja:

- Obtenção da Matriz RPY a partir da definição de 3 ângulos

$$RPY(\phi, \theta, \psi) = ROT(z, \phi) \cdot ROT(y, \theta) \cdot ROT(z, \psi)$$

$$RPY(\phi, \theta, \psi) = \begin{bmatrix} c\phi c\theta & c\phi s\theta s\psi - s\phi c\psi & c\phi s\theta c\psi + s\phi s\psi \\ s\phi c\theta & s\phi s\theta s\psi + c\phi c\psi & s\phi s\theta c\psi - c\phi s\psi \\ -s\theta & c\theta s\psi & c\theta c\psi \end{bmatrix} \quad (3.6)$$

A figura 3.5 mostra o processo para se obter os ângulos Roll, Pitch e Yaw.

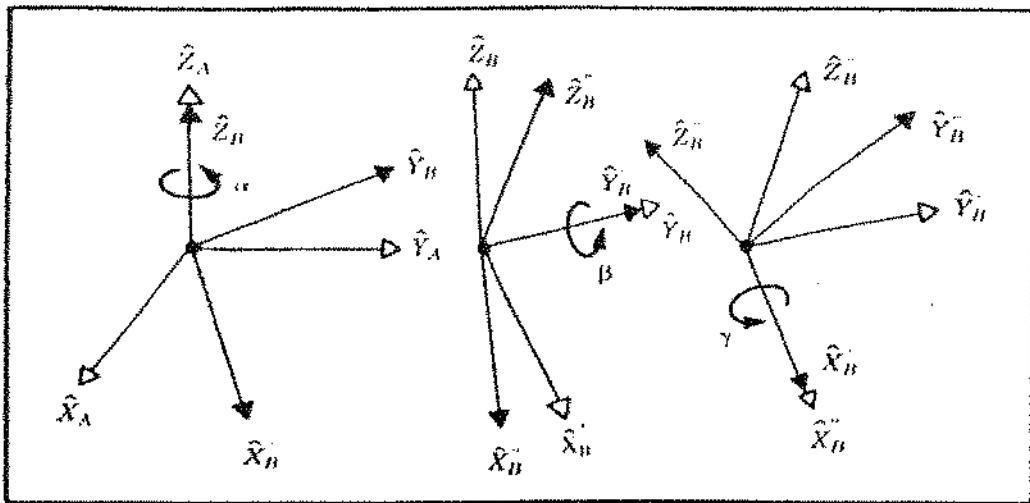


Figura 3.5: Definição dos Ângulos Roll, Pitch e Yaw.

- Problema Inverso - Obtenção dos 3 ângulos Roll, Pitch, Yaw a partir da matriz de orientação

$$\phi = ATAN2 \left[ \frac{n_y}{n_x} \right]$$

$$\theta = ATAN2 \left[ \frac{-n_z}{c\phi n_x + s\phi n_y} \right] \quad (3.7)$$

$$\psi = ATAN2 \left[ \frac{s\phi a_x - c\phi a_y}{-s\phi s_x + c\phi s_y} \right]$$

É importante observar que essa relação deve ser sempre biunívoca. Para o robô Manutec r3, os ângulos  $\phi$ ,  $\theta$  e  $\psi$  são denominados, respectivamente, A, B e C. A figura 3.5 mostra

### 3.3.3 Parâmetros de Denavit-Hartenberg (D.H.)

A partir do Modelo geométrico direto podemos determinar as coordenadas de posição e orientação de cada *link*, e conseqüentemente a posição e orientação do elemento terminal do robô/manipulador em relação ao sistema de coordenadas da base.

Para cada *link* de um robô/manipulador será fixado um sistema de coordenadas composto de três versores ortogonais. Estas coordenadas são chamadas de “link-coordinates”, e sua posição e orientação são definidas por matrizes de transformação homogêneas.

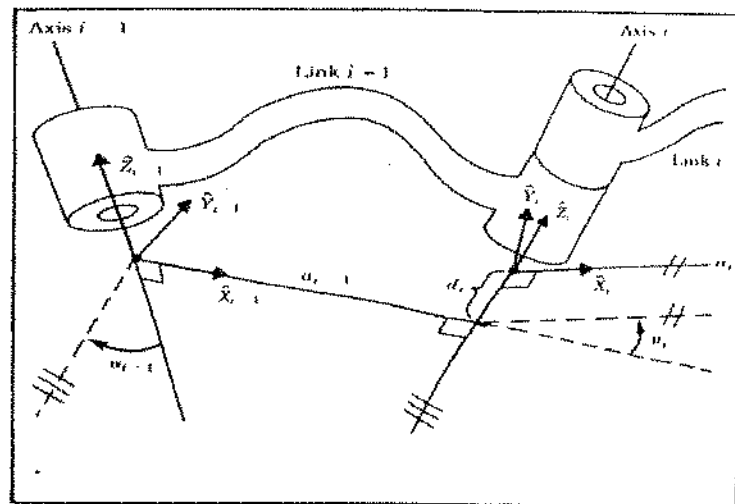


Figura 3.6: Parâmetros de Denavit-Hartenberg.

A sistemática estabelecida por Denavit e Hartenberg (ver figura 3.6), permite descrever a posição sucessiva das coordenadas de cada *link* através da definição de quatro parâmetros ( $a$ ,  $\alpha$ ,  $d$  e  $\theta$ ) [Denavit,55]. A tabela 3.1 sumariza a definição desses parâmetros.

No caso do robô/manipulador em estudo qualquer junta  $i$  é rotacional, conseqüentemente  $\theta_i$  é uma variável de junta e  $d_i$ ,  $a_i$ , e  $\alpha_i$  são constantes.

PARÂMETROS DE DENAVIT-HARTENBERG
$a_i =$ A distância entre $Z_i$ e $Z_{i+1}$ medida ao longo de $X_i$
$\alpha_i =$ O ângulo entre $Z_i$ e $Z_{i+1}$ medido sobre $X_i$
$d_i =$ A distância entre $X_{i-1}$ e $X_i$ medida ao longo de $Z_i$
$\theta_i =$ O ângulo entre $X_{i-1}$ e $X_i$ medido sobre $Z_i$

Tabela 3.1: Definição dos Parâmetros de Denavit-Hartenberg - 2 links adjacentes.

As coordenadas de posição e orientação de um dado *link*  $i$  em relação às coordenadas do *link*  $i - 1$  podem ser descritas como:

$$A_{i-1,i} = \begin{bmatrix} c\theta_i & -s\theta_i c\alpha_i & s\theta_i s\alpha_i & a_i c\theta_i \\ s\theta_i & c\theta_i c\alpha_i & -c\theta_i s\alpha_i & a_i s\theta_i \\ 0 & s\alpha_i & c\alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (3.8)$$

As coordenadas de posição e orientação de um *link*  $m$  em relação ao sistema de coordenadas fixo a base do robô será:

$$A_{0,m} = A_{0,1} \cdot A_{1,2} \cdots A_{m-2,m-1} \cdot A_{m-1,m} \quad (3.9)$$

onde:  $m$  é número de graus de liberdade do manipulador.

O robô/manipulador deverá possuir uma ferramenta presa ao último *link*, e a mesma fará parte de seu elemento terminal. Esta ferramenta possuirá um sistema de referência em relação às coordenadas deste *link*, que será designado como WKS (notação Manutec r3).

A posição e orientação descrita em relação às coordenadas da base do manipulador serão designadas como TOOL.

WKS representa uma matriz constante, função da geometria da ferramenta utilizada, enquanto TOOL uma função das variáveis articulares do robô ou manipulador,



ou seja:

$$TOOL = A_{0,m}.WKS \tag{3.10}$$

### 3.4 Modelo Geométrico do Manutec r3

O robô Manutec r3 possui seis juntas rotacionais. Seus parâmetros de Denavit-Hartenberg (D.H.) são apresentados na tabela 3.2. Utilizando estes valores na eq. 3.8, podemos obter as matrizes  $A_{i-1}$  apresentadas no apêndice C.

JUNTA	$\theta$ (gr)	d (mm)	$\alpha$ (gr)	a (mm)	range (gr)	REPRESENTACAO
1	$\theta_1$	665.0	-90.0	0.0	-165	
2	$\theta_2$	0.0	0.0	500.0	-20/+220	
3	$\theta_3$	0.0	90.0	0.0	-225/+45	
4	$\theta_4$	730.0	-90.0	0.0	-190	
5	$\theta_5$	0.0	90.0	0.0	-120	
6	$\theta_6$	100.0	0.0	0.0	-265	

Tabela 3.2: Robô Manutec r3 - Parâmetros de Denavit-Hartenberg.

Obs: Os parâmetros geométricos  $a$  e  $d$  da tabela 3.2 são aproximados e deverão ser estimados através de um algoritmo de identificação [Campos,93].

$$T_6(\theta_1, \theta_2, \dots, \theta_6) =$$

$$A_{0,6} = A_{0,1}.A_{1,2}.A_{2,3}.A_{3,4}.A_{4,5}.A_{5,6}$$

$$= \begin{bmatrix} & & & \vdots & & \\ \bar{n} & \bar{s} & \bar{a} & \vdots & \bar{p} & \\ & & & \vdots & & \\ \dots & \dots & \dots & \dots & \dots & \\ & 0 & & \vdots & & 1 \end{bmatrix}$$

• Orientação

$$n_x = [(C1C23C4 - S1S4).C5 - C1S23S5].C6 + [-C1C23S4 - S1C4].S6$$

$$n_y = [(S1C23C4 + C1S4).C5 - S1S23S5].C6 + [-S1C23S4 + C1C4].S6$$

$$n_z = [-S23C4C5 - C23S5].C6 + S23S4S6$$

$$s_x = -[(C1C23C4 - S1S4).C5 - C1S23S5].S6 + [-C1C23S4 - S1C4].C6$$

$$s_y = -[(S1C23C4 + C1S4).C5 - S1S23S5].S6 + [-S1C23S4 + C1C4].C6$$

$$s_z = [S23C4C5 + C23S5].S6 + S23S4C6$$

$$a_x = (C1C23C4 - S1S4).S5 + C1S23C5$$

$$a_y = (S1C23C4 + C1S4).S5 + S1S23C5$$

$$a_z = -S23C4S5 + C23C5$$

• Posição

$$p_x = [(C4S5d6).C23 + (C5d6 + d4).S23 + a2C2].C1 - S1S4S5d6$$

$$p_y = [(C4S5d6).C23 + (C5d6 + d4).S23 + a2C2].S1 + C1S4S5d6$$

$$p_z = -(C4S5d6).S23 + (C5d6 + d4).C23 - a2S2 + d1$$

Onde:

$$C_i = \cos(\theta_i) \quad i = 1, 2, \dots, 6$$

$$S_i = \sin(\theta_i) \quad i = 1, 2, \dots, 6$$

$$C_{ij} = \cos(\theta_i + \theta_j) \quad i, j = 1, 2, \dots, 6$$

$$S_{ij} = \sin(\theta_i + \theta_j) \quad i, j = 1, 2, \dots, 6$$

Como não dispunhamos das equações do modelo geométrico do robô Manutec r3, uma análise do modelo implementado para o mesmo foi efetuada a partir da comparação dos resultados obtidos na implementação do seu modelo geométrico com os resultados lidos no controlador SIROTEC do robô e os resultados foram considerados satisfatórios.

### 3.5 Modelo Geométrico do Kraft

O manipulador Kraft também possui seis juntas rotacionais. Os parâmetros de Denavit-Hartenberg (D.H.) para este manipulador são apresentados na tabela 3.3. Utilizando estes valores na eq. 3.8, podemos obter as matrizes de passagem  $A_{i-1}$ , que são apresentadas na apêndice C.

JUNTA	$\theta$ (gr)	$d$ (mm)	$\alpha$ (gr)	$a$ (mm)	range (gr)	REPRESENTACAO
1	$\theta_1$	$d_1$	-90.0	0.0	-90	
2	$\theta_2$	0.0	0.0	$a_2$	0/+120	
3	$\theta_3$	0.0	0.0	$a_3$	-20/-130	
4	$\theta_4$	730.0	-90.0	$a_4$	-42/+58	
5	$\theta_5$	0.0	90.0	0.0	-52.5	
6	$\theta_6$	$d_6$	0.0	0.0	0/+360	

Tabela 3.3: Manipulador Submarino Kraft - Parâmetros de D.H.

$$T_6(\theta_1, \theta_2, \dots, \theta_6) =$$

$$A_{0,6} = A_{0,1} \cdot A_{1,2} \cdot A_{2,3} \cdot A_{3,4} \cdot A_{4,5} \cdot A_{5,6}$$

$$= \begin{bmatrix} \vdots & & & & \\ \bar{n} & \bar{s} & \bar{a} & \vdots & \bar{p} \\ & & & \vdots & \\ \dots & \dots & \dots & \dots & \dots \\ & 0 & & \vdots & 1 \end{bmatrix}$$

- Orientação

$$n_x = -C1.(C234S5C6 + S234S6) - S1C5C6$$

$$n_y = -S1.(C234S5C6 + S234S6) + C1C5C6$$

$$n_z = -S234S5C6 + C234S6$$

$$s_x = C1.(C234S5S6 - S234C6) + S1C5S6$$

$$s_y = S1.(C234S5S6 - S234C6) - C1C5S6$$

$$s_z = S234S5S6 + C234C6$$

$$a_x = C1C234C5 - S1S5$$

$$a_y = S1C234C5 + C1S5$$

$$a_z = S234C5$$

- Posição

$$p_x = d6.(C1C234C5 - S1S5) + C1.(a4C234 + a3C23 + a2C2)$$

$$p_y = d6.(S1C234C5 + C1S5) + S1.(a4C234 + a3C23 + a2C2)$$

$$p_z = d6.S234C5 + a4S234 + a3S23 + a2S2 + d1$$

Uma análise do modelo geométrico direto implementado para o manipulador Kraft foi realizada. Os parâmetros geométricos utilizados foram estimados a partir dos desenhos básicos do manipulador e foram calculados com mais rigor a partir da utilização de técnicas de identificação de parâmetros [Campos,93].

### 3.6 Problema Inverso

A modelagem inversa de manipuladores refere-se a técnicas para obtenção dos valores angulares das juntas de maneira a se obter uma posição e uma orientação específica do elemento terminal do robô.

Os robôs e manipuladores, em geral, são comandados através de posições angulares de suas juntas, enquanto que os objetos a serem manipulados são normalmente

expressos em termos de coordenadas cartesianas. Quando se deseja conhecer os valores das variáveis de junta, dado a posição espacial da ferramenta de um robô, deve-se estabelecer o chamado modelo inverso do mesmo.

A solução do problema de modelagem inversa pode ser dividida em basicamente duas classes: Solução através de formas fechadas ou analíticas (modelo geométrico inverso) e soluções numéricas (modelo cinemático inverso). Dentre as técnicas que utilizam uma forma fechada estão o Método da Transformada Inversa [Paul,81] e os métodos geométricos [Lee,84]. Esses métodos tentam encontrar fórmulas algébricas ou geométricas que relacionem as variáveis cartesianas com as variáveis angulares do robô. Eles têm bastante aplicabilidade para robôs com até quatro graus de liberdade. Dentre os métodos numéricos, ou iterativos, destacam-se o de Gauss, o de Greville e o de Miss [Miss,92].

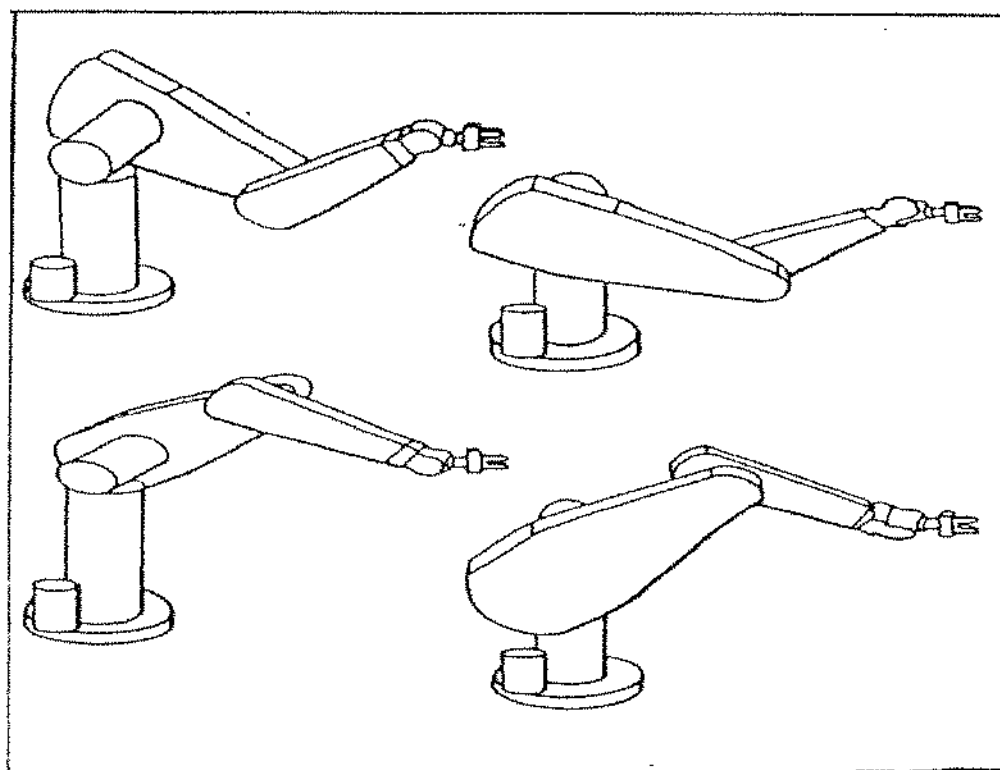


Figura 3.7: Múltiplas Soluções para o Modelo Geométrico Inverso

Embora os métodos numéricos sejam geralmente mais lentos, devido a sua natureza iterativa, que a solução correspondente em forma fechada, eles apresentam vantagens

sobre este último, visto que esses métodos são genéricos, isto é, eles podem ser aplicados a qualquer tipo de robô. Além disso, eles convergem sempre para a posição desejada e apresentam uma solução única.

Outro fato importante é que os pontos gerados em cada iteração de um método numérico podem ser diretamente enviados para o controlador do robô como um ponto de referência, agilizando, assim, o processo de movimento cartesiano e abrindo a possibilidade de se implementar testes de colisão em tempo-real.

Os métodos analíticos, não raro, conduzem a soluções múltiplas (ver figura 3.7), impondo ao sistema o dilema de escolher a melhor configuração.

Além disso, Craig mostra que quase todo sistema com juntas revolutas ou prismáticas tendo um total de seis graus de liberdade possui apenas a solução numérica para a modelagem inversa [Craig,89]. Apenas em alguns casos especiais, o problema da modelagem inversa para robôs com seis graus de liberdade pode ser resolvido analiticamente. Pieper [Pieper,68] apresentou uma solução analítica para robôs com seis graus de liberdade cujas três primeiras juntas são revolutas ou prismáticas e cujos eixos das três últimas se interceptam em algum ponto.

No caso em questão, o Manutec e o Kraft não satisfazem essa condição, além disso, o sistema desenvolvido, que será descrito no capítulo 4, pretende ser genérico, utilizando, dessa forma, a abordagem numérica.

### 3.6.1 Modelagem Cinemática Inversa

Como visto anteriormente, a cada movimento definido no espaço articular do robô, corresponde um movimento no espaço operacional ou cartesiano. Essa correspondência foi estabelecida pela equação 3.1 mostrada abaixo:

$$\tilde{X} = F(\tilde{\Theta}) \quad (3.11)$$

onde:

$$\begin{aligned}\bar{\Theta} &= (\theta_1, \theta_2, \dots, \theta_n). \\ \bar{X} &= (x, y, z, \psi, \theta, \phi),\end{aligned}$$

de forma que:

$$\begin{aligned}x &= f_1(\theta_1, \theta_2, \theta_3, \theta_4, \theta_5, \theta_6) \\ y &= f_2(\theta_1, \theta_2, \theta_3, \theta_4, \theta_5, \theta_6) \\ &\vdots \\ \phi &= f_6(\theta_1, \theta_2, \theta_3, \theta_4, \theta_5, \theta_6)\end{aligned}$$

A linearização dessas equações em torno de um ponto de equilíbrio pode ser obtida através do cálculo das derivadas parciais do funcional dado pela equação 3.11. Temos assim:

$$\frac{d\bar{X}}{dt} = \frac{\partial F(\theta)}{\partial \theta} \frac{d\bar{\theta}}{dt} \quad (3.12)$$

ou,

$$\frac{d\bar{X}}{dt} = \frac{\partial \bar{X}}{\partial \bar{\theta}} \frac{d\bar{\theta}}{dt} \quad (3.13)$$

Ao conjunto de elementos  $\frac{\partial \bar{X}}{\partial \bar{\theta}}$ , damos o nome de Jacobiano e podemos escrever, então, em forma matricial:

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \\ \dot{\psi} \\ \dot{\theta} \\ \dot{\phi} \end{bmatrix} = \begin{bmatrix} J_{11} & J_{12} & J_{13} & J_{14} & J_{15} & J_{16} \\ J_{21} & J_{22} & J_{23} & J_{24} & J_{25} & J_{26} \\ J_{31} & J_{32} & J_{33} & J_{34} & J_{35} & J_{36} \\ J_{41} & J_{42} & J_{43} & J_{44} & J_{45} & J_{46} \\ J_{51} & J_{52} & J_{53} & J_{54} & J_{55} & J_{56} \\ J_{61} & J_{62} & J_{63} & J_{64} & J_{65} & J_{66} \end{bmatrix} \begin{bmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \\ \dot{\theta}_3 \\ \dot{\theta}_4 \\ \dot{\theta}_5 \\ \dot{\theta}_6 \end{bmatrix} \quad (3.14)$$

onde:

$$J_{11} = \frac{\partial \bar{x}}{\partial \theta_1}, \quad J_{12} = \frac{\partial \bar{x}}{\partial \theta_2} \quad \dots \quad J_{16} = \frac{\partial \bar{x}}{\partial \theta_6}$$

$$J_{21} = \frac{\partial \bar{y}}{\partial \theta_1}, \quad J_{22} = \frac{\partial \bar{y}}{\partial \theta_2} \quad \dots \quad J_{26} = \frac{\partial \bar{y}}{\partial \theta_6}$$

⋮

$$J_{61} = \frac{\partial \bar{\phi}}{\partial \theta_1}, \quad J_{62} = \frac{\partial \bar{\phi}}{\partial \theta_2} \quad \dots \quad J_{66} = \frac{\partial \bar{\phi}}{\partial \theta_6}$$

Eliminando  $dt$  na equação 3.13, podemos escrever finalmente, para pequenos deslocamentos:

$$\Delta X_{n \times 1} = J_{n \times m} \Delta \theta_{m \times 1}$$

onde:

$\Delta X$  = incremento de posicionamento espacial;

$\Delta \theta$  = incremento angular das juntas;

$J$  = Jacobiano;

$n$  = número de variáveis cartesianas;

$m$  = número de graus de liberdade do robô.

De forma análoga, todo movimento definido no espaço cartesiano pode ser mapeado no espaço de coordenadas generalizadas (ângulos das juntas), e obtido através da inversão do modelo geométrico.

$$\Delta \bar{\theta} = J^{-1} \Delta \bar{X} \quad (3.15)$$

A obtenção de uma solução  $\Delta \bar{\theta}$  não é trivial. Como  $J$  é um funcional obtido pela linearização da equação 3.11 (não linear), não se pode garantir a existência e/ou



a unicidade de uma função  $J^{-1}$ . O cálculo da inversa da matriz jacobiana é complexo e demanda recursos sofisticados para sua execução em tempo real. Ressalta-se ainda que a sistemática apresentada permite o cálculo dos deslocamentos angulares necessários para o controle das juntas a partir de um algoritmo de inversão numérica. Conforme afirmado anteriormente, os métodos numéricos apresentam vantagens no que diz respeito a convergência e possibilidades de testes intermediários, tais como colisão, fim-de-curso, etc.

Se a programação do robô é feita através de aprendizagem, não é necessária a obtenção do modelo cinemático inverso, porque quando o operador conduz o órgão terminal, os pontos de referência correspondentes são registrados pelos sensores. No entanto, quando se deseja que o robô se desloque de um ponto a outro seguindo uma linha reta, por exemplo, torna-se necessário o cálculo de  $J^{-1}$ .

Num trabalho de dissertação apresentado nesta instituição [Fayan,92] foi feito um estudo sobre métodos de inversão numérica. Os resultados obtidos nesse trabalho serão utilizados na implementação do software geral para realização do sistema proposto.

### 3.7 Conclusões

Este capítulo apresentou o desenvolvimento de uma metodologia matemática necessária para o trabalho com sistemas robóticos. Foi discutido o problema da transposição direta e inversa. Os parâmetros das juntas e *links* de um robô foram definidos e uma matriz de transformação homogênea  $4 \times 4$  foi introduzida para descrever a localização de um *link* com relação a um sistema de coordenadas fixo.

O capítulo seguinte irá se ater ao problema específico da geração de trajetória e apresentará as implementações feitas no software desenvolvido para monitorar e otimizar o controle do manipulador Kraft.

# Capítulo 4

## Geração de Trajetórias

### 4.1 Introdução

De posse dos modelos geométrico e cinemático do manipulador discutidos no capítulo anterior, estamos aptos agora a focar o problema da geração de trajetória. Este capítulo aborda as principais características das técnicas de planejamento de trajetórias implementadas.

Em geral, trajetória refere-se à evolução no tempo da posição, da velocidade e aceleração de cada grau de liberdade do manipulador, ou seja, a geração de incrementos angulares de juntas necessários para que o mesmo realize de maneira adequada uma dada tarefa. Ao mesmo tempo, procedimentos devem ser adotados para que os limites de velocidade e aceleração de cada junta não sejam extrapolados e para que não haja colisões.

Esse tópico trata do problema de como desejamos especificar a trajetória ou o caminho através do espaço. De forma a tornar a descrição do movimento do manipulador mais fácil para o usuário, este não deve ser obrigado a escrever uma função complicada do espaço e do tempo para especificar a tarefa. Ao invés disso, o sistema deve permitir a capacidade de se especificar a trajetória com a simples descrição do movimento desejado,

e ele próprio cuidar dos detalhes. Por exemplo, o usuário pode descrever apenas a posição final desejada e a orientação da garra, e deixar que o sistema decida sobre a forma exata da trajetória para alcançar o objetivo, a duração, o perfil de velocidade e outros detalhes.

As técnicas de geração de trajetórias normalmente “interpolam” ou “aproximam” o caminho desejado por uma classe de funções polinomiais e geram uma sequência de pontos (set points) para o controle do manipulador. Tais trajetórias podem ser especificadas tanto em coordenadas de juntas como em coordenadas cartesianas, contudo, usualmente elas são especificadas em coordenadas cartesianas, porque o elemento mais importante para visualização é a ferramenta, e o referencial mais adequado é aquele colocado na garra. Se a coordenada de junta é desejada, então usamos a cinemática inversa para fazer a conversão necessária.

## 4.2 Controle de Trajetória

O problema do controle de trajetória mostrado de forma simplificada na figura 3.1 é agora tratado com maior profundidade. A figura 4.1 mostra um esquema detalhado da estrutura de controle de trajetórias proposta neste trabalho.

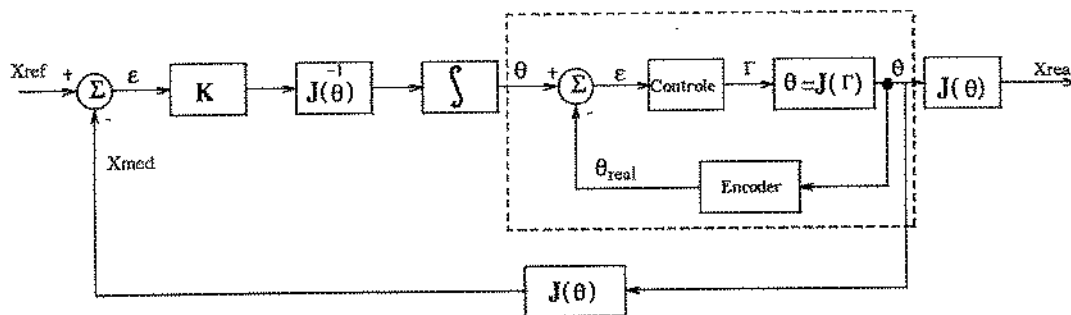


Figura 4.1: Diagrama de blocos de um controlador de trajetória.

onde:

$J_e J^{-1}$  são as matrizes de transposições direta e inversa respectivamente;

$X_{ref}$  é a posição cartesiana desejada para a garra do robô, fornecida a partir do algoritmo de geração de trajetória;

$X_{med}$  é a posição cartesiana real da garra do robô, obtida a partir do modelo geométrico do manipulador.

Através da comparação de  $X_{ref}$  e  $X_{med}$  um sinal de erro é gerado e amplificado. A seguir este sinal é transposto em termos de erros de coordenadas articulares ( $\delta\theta$ ) e integrado, para posteriormente ser utilizado como sinal de referência para o controle das juntas do robô. O bloco que representa o robô, relaciona os torques  $\Gamma$ , necessários para o acionamento das juntas, com os ângulos  $\theta$ .

A porção enclausurada representa o controle de trajetória a nível de juntas, mostrando assim que o sistema suporta a concepção de trajetórias tanto no espaço articular como no espaço cartesiano

As estratégias de controle de trajetória implementadas no sistema foram as seguintes:

- Controle a nível de juntas:
  - Caminho contínuo;
  - Ponto-a-ponto;
- Controle cartesiano.

Esses dois tipos básicos de geração de trajetória serão enfocados a seguir.

### 4.3 Trajetória no Espaço de Juntas

Nesta seção iremos discutir os métodos de geração de trajetória que se baseiam nos ângulos das juntas do robô.

Já vimos anteriormente que a descrição de uma trajetória pode ser feita tanto no espaço articular como no espaço cartesiano. Quando ela é especificada em termos de

funções dos ângulos das juntas, dizemos que essa trajetória está no espaço de juntas ou espaço articular.

Geralmente, com base nessa abordagem, uma trajetória é especificada em termos da posição e orientação de partida e da posição e orientação desejada. Cada ponto da trajetória planejada é “convertido” em um conjunto de ângulos de juntas através da cinemática inversa. Então, é encontrada, para cada uma das juntas uma função suave que passe pelo ponto de partida, pelos pontos de passagem e pelo ponto final da trajetória.

No caso do sistema Kraft essa “conversão” é desnecessária. Uma outra forma de se fazer a aquisição dos pontos de passagem é possível de ser realizada, visto que o manipulador pode ser programado por aprendizagem, isto é, ele é colocado em posições nas quais se deseja que seu elemento terminal passe e o valor angular de cada junta é gravado pelo microcomputador. O fato de seus sensores de junta serem do tipo absoluto (potenciômetros) e não incrementais (encoders) [Souza,92] também colaborou para a não necessidade do uso da cinemática inversa para esse tipo de trajetória.

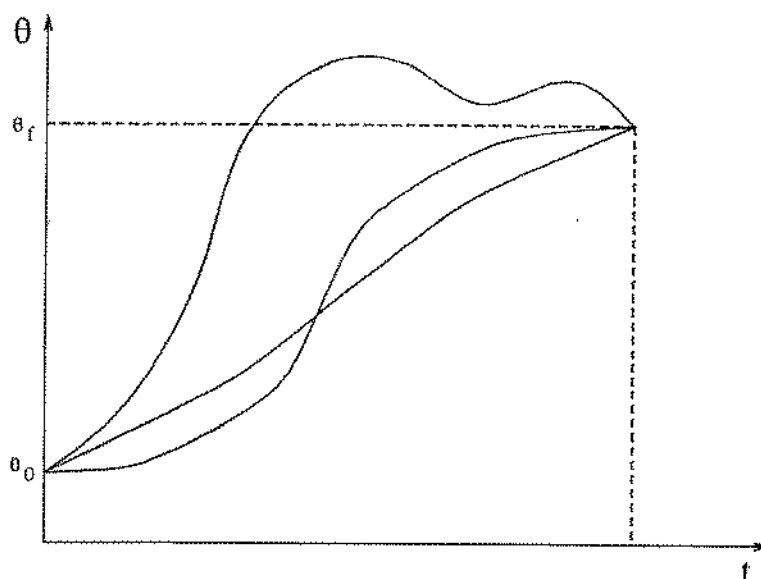


Figura 4.2: Possíveis caminhos suaves de  $\theta_0$  a  $\theta_f$ .

Voltando ao caso genérico, é óbvio que existem diversos caminhos suaves que levam o referencial da ferramenta de  $\theta_0$  a  $\theta_f$  em um intervalo de tempo  $\Delta t = t_f - t_0$ . A figura 4.2 mostra algumas soluções possíveis.

### 4.3.1 Lei de Movimento

A função de interpolação suave que deve passar por todos os pontos especificados do caminho, pode ser simplesmente uma reta no espaço de juntas. Cada ponto da trajetória é interligado com o seguinte através de uma linha reta. Note que embora o movimento de cada junta seja linear, a garra, em geral, não se move numa linha reta no espaço cartesiano. A figura 4.3 mostra como seria o movimento de uma junta hipotética executando uma trajetória deste tipo.  $P_i$  e  $P_f$  são respectivamente os pontos inicial e final da trajetória e  $P_{p1}$  e  $P_{p2}$  são pontos de passagem.

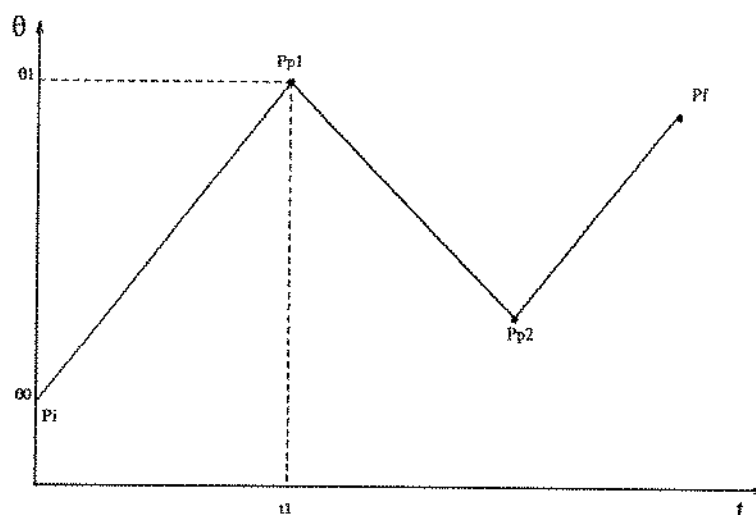


Figura 4.3: Trajetória linear no espaço de juntas.

Uma interpolação linear simples, no entanto, iria provocar no início da trajetória do mecanismo, uma aceleração muito grande [Snyder,85], que possivelmente nenhum manipulador convencional é capaz de fornecer. Mas o pior é que este tentaria alcançar tal variação de velocidade. Os robôs e manipuladores, e particularmente os modelos hidráulicos são muito potentes e capazes de altas acelerações, o que poderia tornar tal experiência bastante desagradável.

Para que não haja, então, uma variação muito brusca da velocidade, deve-se colocar muitos pontos na parte inicial da trajetória (fase de aceleração). A seguir, coloca-se pontos mais espaçados (fase de velocidade constante) e no final volta-se a colocar bastante pontos (fase de desaceleração). A figura 4.4 mostra tal interpolação no plano  $xy$ .

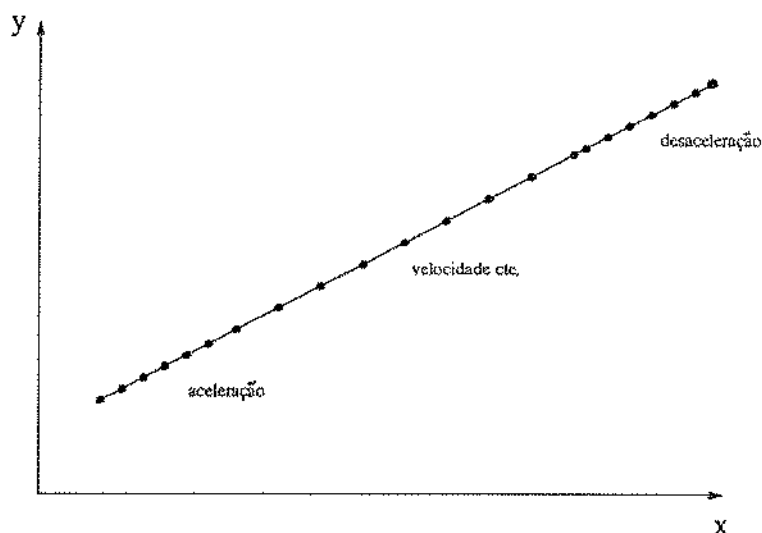


Figura 4.4: Trajetória de um robô no plano  $xy$ . Os set-points são indicados por pontos ao longo da linha.

Se plotarmos a interpolação linear de uma junta como mostrada pela figura 4.4 em um gráfico de deslocamento angular versus tempo, teremos uma curva como a mostrada na figura 4.5.

Essa figura mostra que a trajetória executada é formada por um trecho linear (velocidade constante) e por trechos parabólicos nas extremidades (aceleração e desaceleração). Isto fisicamente significa que a junta parte com velocidade zero e lentamente vai atingindo um valor constante. No final da trajetória ela novamente decresce até zero.

Na prática, para a realização de uma tarefa, o manipulador executa diversas trajetórias concatenadas, e essa técnica de suavizar as extremidades permite a execução de trajetórias de maneira uniforme, não necessitando de grandes sofisticções nos controladores.

Com o propósito de descobrir o perfil de velocidade para cada junta, de maneira a obter o perfil de deslocamento desejado, integramos a curva da figura 4.5, obtendo um

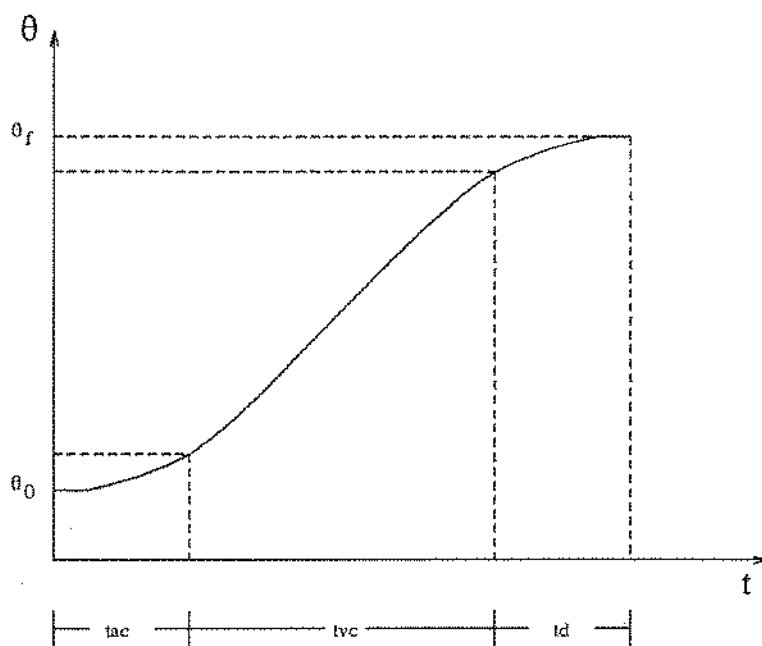


Figura 4.5: Perfil de posição linear com extremidades parabólicas.

perfil trapezoidal como mostrado na figura 4.6. Através desse perfil, pode-se observar que a aceleração e a desaceleração possuem valores constantes.

Na figura qualitativa 4.6 o valor da velocidade  $v_d$  é, a priori a máxima velocidade de deslocamento da junta,  $t_{ac}$  é o tempo de aceleração,  $t_{vc}$  é o tempo de velocidade constante e  $t_d$  é o tempo de desaceleração. Veremos, mais adiante, que o valor da velocidade de deslocamento  $v_d$  para uma junta específica depende e é proporcional ao deslocamento dessa junta, isso porque, é desejável que todas as juntas partam ao mesmo tempo e cheguem ao objetivo ao mesmo tempo.

### 4.3.2 Gravação do Caminho Contínuo

Consideremos o caso mais simples de controle de trajetória, no qual o robô é levado a executar um movimento através do controle do operador, e o sistema simplesmente grava a trajetória para posteriormente executá-la.

Considere, por exemplo, uma aplicação de pintura usando spray. Um canhão de spray é instalado na garra do robô, então, um pintor experiente segura o braço do



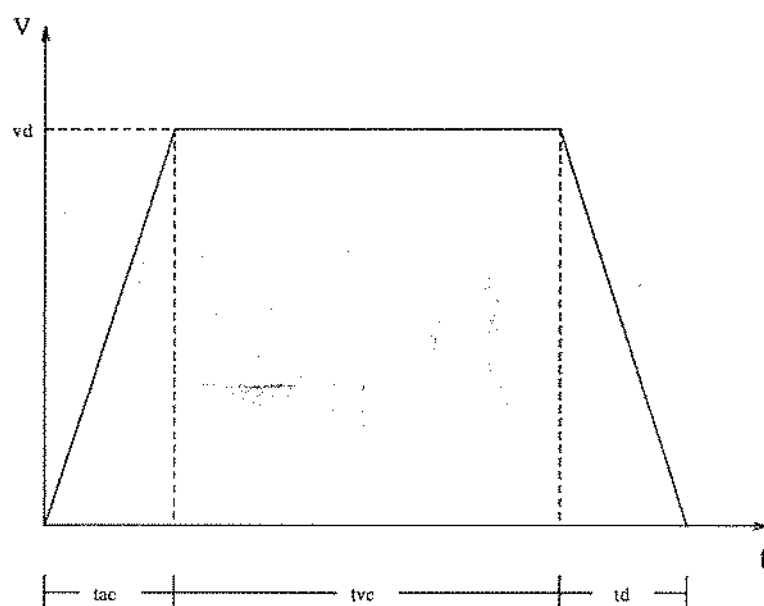


Figura 4.6: Perfil de velocidade trapezoidal: *Necessário para movimento suave.*

robô e o guia através da sequência de pintura. Cada uma das juntas é equipada com potenciômetros que provêem os sinais analógicos de posição. Um gravador analógico multicanal está ligado ao robô, com um canal conectado a cada potenciômetro (Fig. 4.7). Dessa forma, cada movimento executado pelo operador é gravado como um conjunto de posições instantâneas das juntas.

Para reproduzir o movimento, as saídas do gravador são alimentadas com o arquivo de pontos das juntas e o robô repetirá a sequência que foi gravada.

O manipulador Kraft funciona de forma semelhante, só que ao invés do operador manusear diretamente o seu braço, ele opera através do master, conforme apresentado no capítulo 2.

A simplicidade dessa técnica de gravação e reprodução, além do fato de que temos a garantia que o caminho resultante é fisicamente realizável, a torna bastante atraente.

Existem alguns problemas com essa técnica, é claro. O mais óbvio é o fato de que todas as imperfeições cometidas pelo operador durante o processo de aprendizagem são gravadas e repetidas pelo manipulador. Os pontos a serem atingidos devem estar em

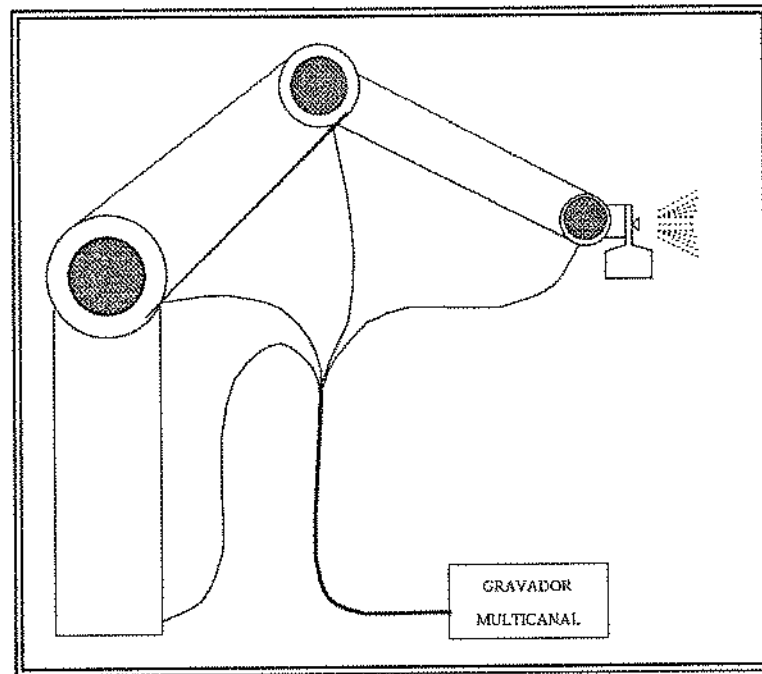


Figura 4.7: *Uso de um gravador multicanal para gravar os sinais dos potenciômetros das juntas de um robô.*

posições muito bem definidas o tempo todo. Isso não é problema para o caso de pintura com spray, porque a área atingida pelo spray é relativamente grande e pequenos erros de posicionamento são insignificantes. No caso de soldagem, no entanto, a acurácia é crítica e a execução precisa da tarefa pode ser excessivamente cara ou impossível. Um outro inconveniente deste método é a necessidade de muita memória para armazenar os arquivos gerados, pois o ciclo de movimento é dividido em centenas ou até milhares de pontos individuais, muito próximos uns dos outros, ao longo da trajetória. Obviamente, o número de pontos de um arquivo depende do tamanho da trajetória, da taxa de aquisição dos pontos e da velocidade com a qual o operador guia o manipulador, mas tipicamente esse valor é muito alto e desnecessário.

Para as aplicações nas quais este método é apropriado, ele provê um meio atrativo de se conseguir o controle de um caminho contínuo. Para aquelas em que ele não é bem apropriado, outras técnicas mais sofisticadas são requeridas.

### 4.3.3 Gravação Ponto-a-Ponto

A técnica de gravação do caminho contínuo, implementada em nosso sistema, se mostrou inadequada para as aplicações de automação submarina, uma vez que ela não proporciona a acurácia necessária para o desenvolvimento de tarefas num ambiente hostil, onde o manipulador está exposto a diversas forças externas.

Outras estratégias de controle de trajetória estão disponíveis, porém, aquelas que produzem um caminho bastante suave e preciso, normalmente requerem grande esforço computacional e conseqüentemente, são mais lentas. O controle de trajetória no espaço cartesiano é um exemplo disso.

Um método simples, mas que apresenta resultados satisfatórios é a técnica de gravação ponto-a-ponto. Snyder [Snyder,85] define as seguintes características desejáveis no desenvolvimento de estratégias de controle de trajetória, e atribui ao controle ponto-a-ponto a capacidade de atender aos itens 1 e 3.

1. Continuidade da posição, velocidade e aceleração;
2. Controle preciso do movimento;
3. Tempo de percurso e características de velocidade e aceleração impostas.

Essa técnica consiste em gravar alguns pontos da trajetória e em seguida interligar esses pontos através de um algoritmo de interpolação linear no espaço articular. Posteriormente é feito um tratamento de filtragem da trajetória gerada de forma a se obter um perfil de posição parabólico nas extremidades de cada intervalo.

Dentre as vantagens desse método com relação ao anterior podemos citar:

- Maior rapidez na execução da tarefa;
- Maior precisão de movimentos;
- Trajetória livre de imperfeições;
- Arquivo de pontos pequeno.

Como desvantagem, há apenas o fato de que o esforço computacional é levemente superior.

As seções seguintes irão focalizar com mais detalhes as etapas de interpolação e filtragem do sistema de geração de trajetória ponto-a-ponto implementado no software de controle.

#### 4.3.3.1 Interpolação

No procedimento para interpolação dos pontos gravados pelo operador, deve-se levar em consideração alguns critérios que podem afetar o desenvolvimento adequado da trajetória. Esses critérios dizem respeito principalmente a algumas restrições do sistema, como a aceleração e a velocidade máximas que cada junta pode fornecer, ou a base de tempo escolhida para amostragem dos sinais interpolados. Também são parâmetros importantes e que devem ser levados em consideração, o número de pontos a serem interpolados entre as extremidades do caminho e o deslocamento angular de cada junta.

Todas as variáveis citadas acima estão, de certa forma, associadas, de maneira que um tratamento apropriado com algumas delas pode ser suficiente para a solução do problema de geração de trajetória no espaço de juntas.

Quando um manipulador se move de um ponto no espaço de trabalho para outro, em geral, todas as suas juntas sofrem um deslocamento, dado pela diferença entre a posição angular final e a posição angular inicial. Esse deslocamento, então deve ser interpolado e posteriormente filtrado, de forma que o manipulador possa se movimentar com o máximo de velocidade e suavidade.

A velocidade máxima desenvolvida por cada junta durante um trecho de trajetória é proporcional ao seu deslocamento e será calculada de forma que todas elas partam e cheguem ao mesmo tempo na posição desejada (condição de sincronização). Se levarmos em consideração apenas o critério de máxima velocidade, fica fácil perceber que cada junta irá gastar um tempo diferente para executar o seu trecho de trajetória, uma vez que elas terão que percorrer distâncias diferentes a velocidades diferentes.

$$t_i = \frac{|\theta_{i_f} - \theta_{i_i}|}{\omega_{max_i}} \quad (4.1)$$

A equação 4.1 mostra o cálculo do tempo gasto por cada junta ( $t_i$ ), onde  $\theta_{i_f}$  e  $\theta_{i_i}$  são as posições angulares final e inicial da junta  $i$  respectivamente e  $\omega_{max_i}$  é a velocidade máxima que a junta  $i$  pode desenvolver<sup>1</sup>.

A questão é que se desconsiderarmos a premissa de movimento sincronizado das juntas, não conseguiremos obter uma trajetória suave. Portanto, devemos tomar como base o tempo gasto pelo manipulador para executar um trecho de trajetória respeitando a condição de sincronização. A equação 4.2 nos fornece a relação desejada.

$$t_c = \max \left( \frac{|\theta_{i_f} - \theta_{i_i}|}{\omega_{max_i}} \right) = \max(t_i) \quad (4.2)$$

onde  $t_c$  é o tempo comum gasto pelas juntas para a execução da trajetória da maneira desejada. Percebe-se que a junta que gastaria o maior tempo caso todas desenvolvessem suas velocidades máximas, é que determina a velocidade das demais [Groover,84], ou seja, essa junta irá desenvolver sua  $\omega_{max}$  enquanto as outras não. Elas têm que girar numa velocidade menor para que todas possam chegar ao mesmo tempo.

Uma vez obtido  $t_c$ , precisamos agora determinar o número de pontos  $n$  a serem interpolados no segmento de trajetória, ou seja, devemos dividir  $t_c$  em  $n$  intervalos de tempo  $\Delta t$ ,

$$n = \frac{t_c}{\Delta t} = t_c * f_{am} \quad (4.3)$$

onde :

$\Delta t$  é a base de tempo escolhida para amostragem dos sinais;

$f_{am}$  é a frequência de amostragem dos sinais.

---

<sup>1</sup>No apêndice A existe uma tabela contendo as velocidades máximas de cada junta do manipulador Kraft.

A escolha da base de tempo para amostragem dos sinais deve levar em consideração o tempo gasto pelo microcomputador para processar as instruções do algoritmo de controle que estão entre dois comandos de envio de pontos, e também o incremento angular, isto é, o incremento de tensão aplicado nos atuadores do manipulador.

Para o sistema Kraft, a base de tempo escolhida foi  $\Delta t = 10\text{ms}$ . Essa taxa de amostragem mostrou-se adequada porque resultou em movimentos contínuos. Quando a frequência de amostragem escolhida é menor que a adequada, o manipulador se move de forma visualmente discreta, através de degraus de tensão claramente perceptíveis. Esse movimento provoca uma série de oscilações no sistema e prejudica a precisão do manipulador ao final da trajetória. No caso da frequência de amostragem ser maior que a adequada, o manipulador se movimenta segundo uma curva contínua de posição, velocidade e aceleração, porém, o tempo gasto para a execução de uma tarefa se torna bastante alto. Deseja-se uma solução de compromisso entre velocidade e continuidade do movimento.

Voltando ao processo de interpolação, podemos agora calcular o incremento de posição  $\Delta\theta_i$  a ser enviado para cada junta.

$$\Delta\theta = \frac{\theta_{i_f} - \theta_{i_i}}{n} \quad (4.4)$$

#### 4.3.3.2 Filtragem

Como foi dito na seção 4.3, para que não haja grande variação de velocidade durante a execução de uma trajetória, e conseqüentemente, para que a acurácia do manipulador seja maior, é necessário que seja feito um alisamento nas extremidades de cada segmento linear que forma um caminho no espaço articular. Após o processo de interpolação, tem-se uma trajetória para uma junta  $i$  semelhante à mostrada na figura 4.8

A literatura propõe a utilização de polinômios cúbicos que obedeçam a pelo menos quatro restrições evidentes para a obtenção de um caminho suave. São elas:

- $\theta(t_0) = \theta_0$
- $\theta(t_f) = \theta_f$

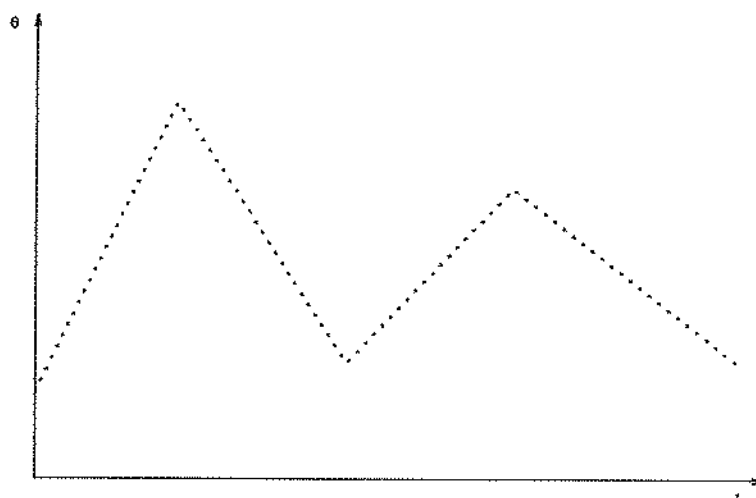


Figura 4.8: Trajetória de uma junta  $i$  após o processo de interpolação.

As duas restrições adicionais são necessárias para que a função  $\theta(t)$  seja contínua em velocidade.

- $\dot{\theta}(t_0) = 0$
- $\dot{\theta}(t_f) = 0$

No sistema desenvolvido, no entanto, utilizou-se um filtro FIR (Finite Impulsive Response) do tipo Janela Triangular para alisar o sinal interpolado. Este tipo de procedimento é frequentemente utilizado em análise de sinais em frequência, podendo ser aplicado à geração de trajetórias no tempo. A teoria de projeto de filtros afirma que esse tipo de filtro (FIR) é quase sempre restrito a implementações discretas no tempo e que o método mais simples de se projetar filtros FIR é o método das janelas [Oppenheim,89].

#### • Cálculo da função Janela Triangular

O processo de filtragem faz uma convolução do sinal interpolado com uma janela de duração finita. O algoritmo gera um vetor contendo a trajetória interpolada e um vetor contendo o filtro triangular e em seguida faz a convolução entre os dois vetores. Tal convolução para sinais discretos no tempo pode ser definida pela seguinte expressão:

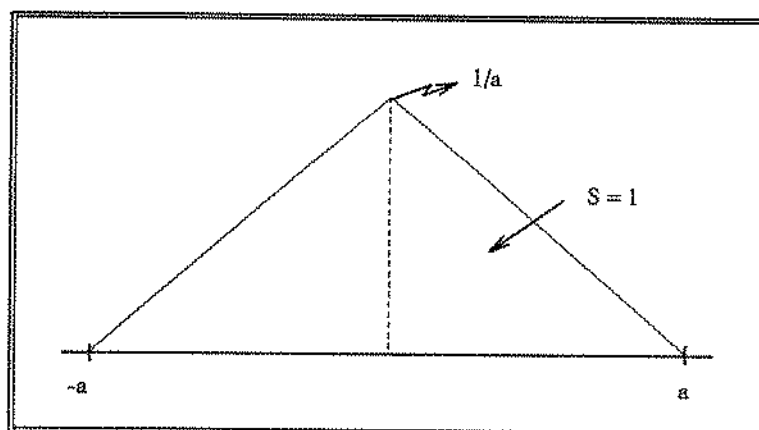


Figura 4.9: Filtro FIR do tipo Janela Triangular.

$$Y(n) = \sum_{k=-\infty}^{\infty} X[k] * W[n - k], \quad (4.5)$$

onde:

$X$  é o vetor que contém o sinal a ser filtrado

$W$  é o vetor que contém a janela triangular

$Y$  é o vetor que contém o sinal filtrado

$k$  é o número de amostras do filtro

$n$  é o número de amostras do vetor resultante

O exemplo abaixo ilustra melhor o processo de convolução discreta:

Suponhamos que o filtro tenha base  $2a = 10$ , portanto, para que sua área  $S$  seja unitária, sua altura deve valer  $h = 1/a = 0.2$ . Para exemplificarmos o procedimento utilizado, suponhamos que a janela contenha 11 amostras, como mostrado na figura 4.10(a) e que o sinal a ser filtrado seja o da figura 4.10(b).

Para executar a convolução, inicialmente rebate-se o sinal a ser convoluido com a janela, depois, faz-se o sinal “passar” através do filtro, ou vice-versa e a partir daí procede-se o cálculo dos elementos do vetor resultante da seguinte forma: O primeiro elemento de  $y$  é o resultado do produto da primeira amostra do filtro pela última amostra do sinal. O segundo, é o produto da segunda amostra do filtro pela última do sinal somado



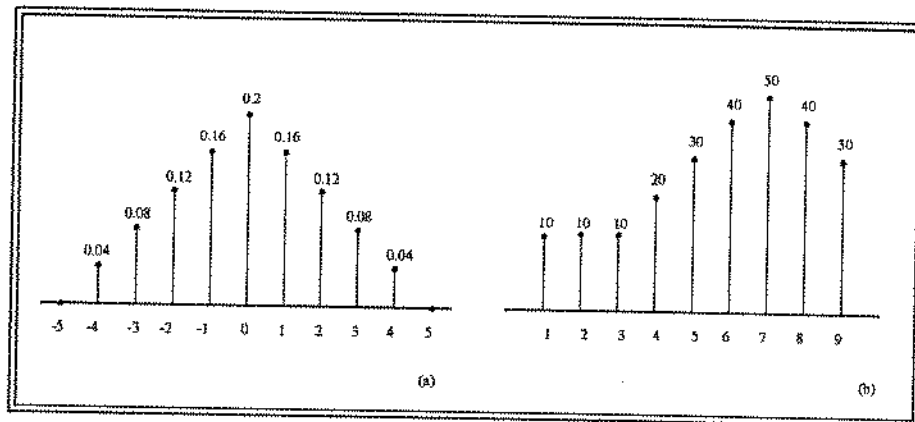


Figura 4.10: (a) Exemplo de filtro triangular. (b) Exemplo de um sinal a ser filtrado.

ao produto da primeira amostra do filtro pela penúltima do sinal, e assim até que todo o sinal “passe” pelo filtro. O número total de pontos do vetor  $Y$  corresponde à soma do número de pontos da janela com o número de pontos do sinal, menos um. A figura 4.11 sintetiza esse processo para o exemplo descrito anteriormente.

$$y(1) = 0 * 10 = 0$$

$$y(2) = 0 * 10 + 0.04 * 10 = 0.4$$

$$y(3) = 0 * 10 + 0.04 * 10 + 0.08 * 10 = 1.2$$

$$y(4) = 0 * 20 + 0.04 * 10 + 0.08 * 10 + 0.12 * 10 = 2.4$$

⋮

$$y(18) = 0.04 * 30 + 0 * 40 = 1.2$$

$$y(19) = 0 * 30 = 0$$

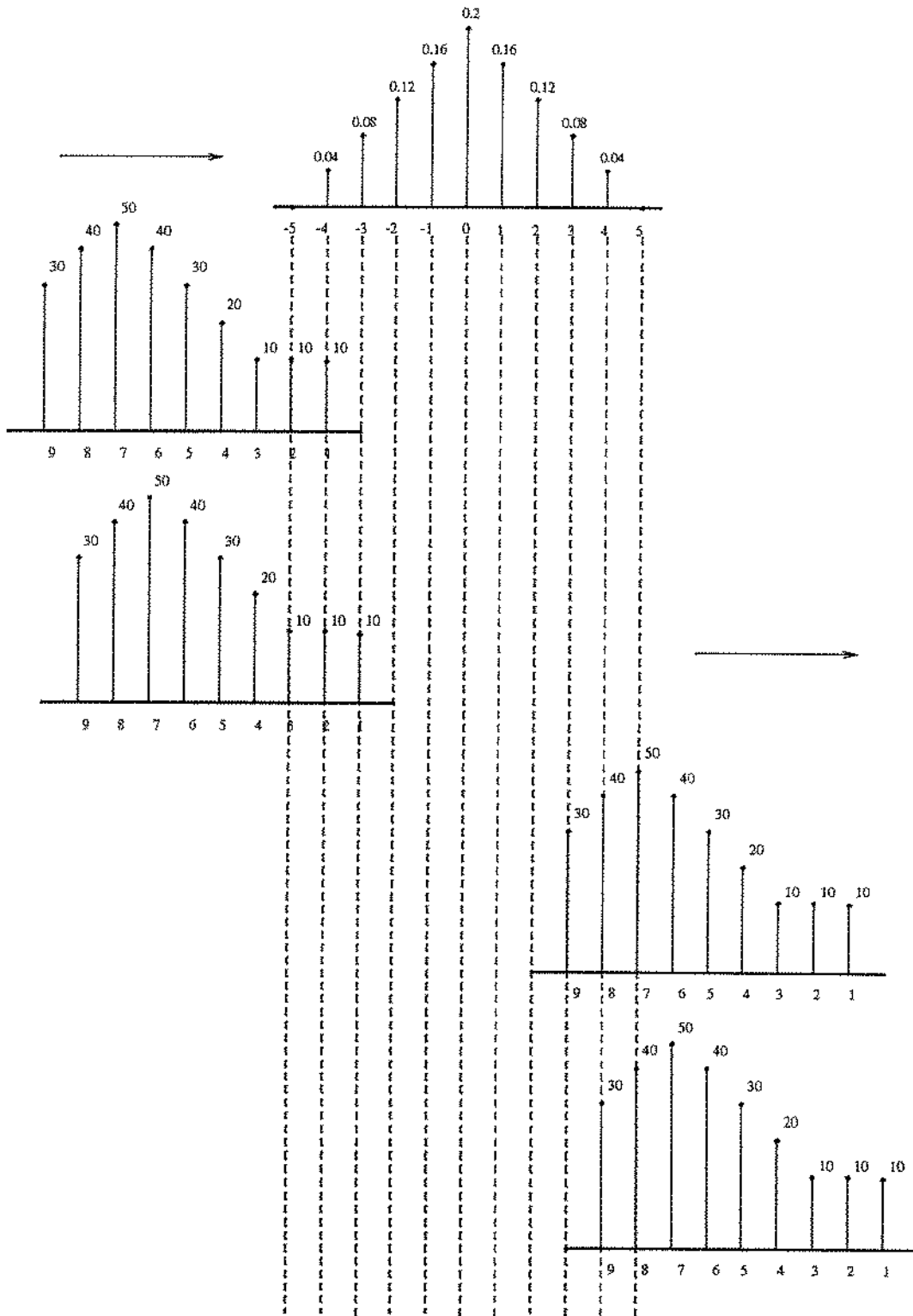


Figura 4.11: Processo de convolução discreta.

Com esse procedimento, obtém-se uma curva sem descontinuidades. Essa forma de gerar trajetórias, além de simples é bastante útil, pois apesar de não ter o controle sobre a trajetória real, existem vantagens em se trabalhar no espaço articular. A seguir temos uma relação de motivos que nos levou a implementar também esse modo de geração de trajetória:

1. A trajetória é planejada diretamente em termos das variáveis que serão controladas durante o movimento;
2. O planejamento da trajetória pode ser feito em tempo-real;
3. As trajetórias no espaço de juntas são mais fáceis de serem planejadas;
4. Como existe uma correspondência unívoca entre o espaço de juntas e o espaço cartesiano (o inverso é falso), não ocorre o problema de configurações singulares<sup>2</sup> no mecanismo;
5. Esse tipo de trajetória é a mais apropriada para movimentos de aproximação.

No apêndice A encontra-se a listagem da rotina que executa o processo de interpolação e filtragem de trajetórias gravadas no modo ponto-a-ponto.

## 4.4 Trajetória no Espaço Cartesiano

Para cumprir com sucesso parte dos objetivos propostos no início deste trabalho, foi necessário desenvolver um módulo de geração cartesiana de trajetória. A diferença básica entre uma trajetória gerada no espaço de juntas e outra gerada no espaço cartesiano é que no caso da primeira, o manipulador sai de um ponto conhecido (ponto de partida) e chega a outro ponto conhecido (ponto de parada), porém, a forma espacial do caminho excursionado pelo garra do manipulador é desconhecida, e está associada à cinemática

---

<sup>2</sup>*Configurações Singulares* ou *Singularidades* é o nome dado às configurações de um robô nas quais não se consegue resolver o problema da cinemática inversa. Na próxima seção esse tópico será abordado com mais detalhes

particular daquele manipulador. Neste modo estamos preocupados com a evolução da trajetória em relação à garra do manipulador.

Existem tarefas que exigem um movimento mais refinado, em que o referencial da ferramenta deve seguir um caminho específico, tal como uma reta ou um arco de círculo, e que além disso, esse caminho seja executado com bastante precisão. Em geral, numa tarefa o movimento de aproximação é feito no espaço articular e o movimento de manipulação no espaço cartesiano.

O controle cartesiano é a técnica mais genérica e precisa disponível [Fu,87]. No entanto, ele tem uma série de desvantagens, a começar pelo fato de que ele envolve uma validação contínua dos pontos de referência do manipulador e sua subsequente transformação para o espaço das juntas. Isso acarreta um grande número de cálculos a serem executados em tempo real. O cálculo do movimento no espaço articular, que é simplesmente uma interpolação linear entre dois pontos, consome cerca de 5 a 10% [Paul,81] do tempo de computação necessário para a geração de movimento cartesiano<sup>3</sup>.

Outra desvantagem desse tipo de movimento é que os métodos numéricos utilizados para a solução do problema, nem sempre convergem para o ponto desejado, embora isso seja raro. Quando isso ocorre, o sistema pára e faz-se necessário posicionar o robô em uma posição vizinha para que o processo possa continuar.

Porém a grande vantagem do movimento cartesiano quando comparado com o movimento articular é que a trajetória executada entre dois pontos é muito bem definida. A figura 4.12 mostra um robô executando uma tarefa que necessita de um movimento em linha reta.

#### 4.4.1 Estratégia Implementada

A geração cartesiana de trajetória implica diretamente na solução da cinemática inversa. Sabe-se do modelo geométrico direto, visto no capítulo 3, que uma transformação do espaço de juntas para o espaço cartesiano é dado, para pequenos deslocamentos, por:

---

<sup>3</sup>O termo Controle Cartesiano especificamente exclui a geração *off-line* de trajetória descrita no cap.1 e requer ao invés disso, que o cálculo dos pontos do caminho, assim como as transformações para o espaço de juntas sejam feitos em tempo real, à medida em que o robô se movimenta.

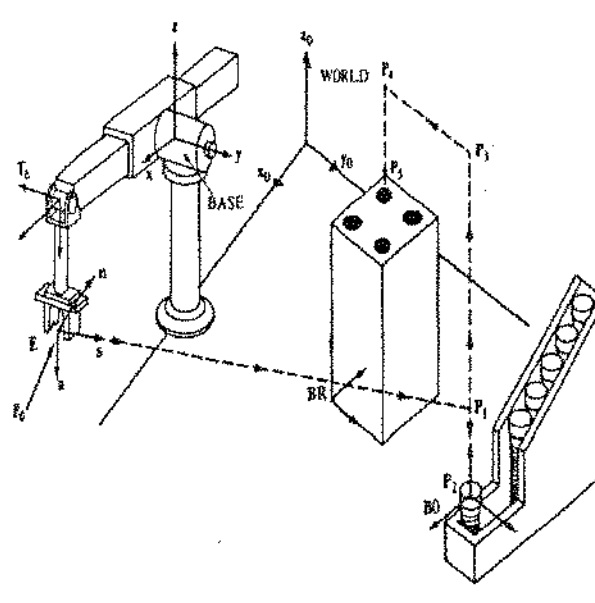


Figura 4.12: Movimento em Linha Reta.

$$\Delta X_{n \times 1} = J_{n \times m} \Delta \theta_{m \times 1} \tag{4.6}$$

É conhecido também, do modelo cinemático inverso que quando se deseja obter os valores angulares correspondentes a uma determinada configuração do manipulador, a relação usada é a seguinte:

$$\Delta \theta_{m \times 1} = J_{n \times m}^{-1} \Delta X_{n \times 1} \tag{4.7}$$

A aplicação iterativa da equação 4.7 foi a forma escolhida para resolver a cinemática inversa. A figura 4.13 mostra a evolução do processo. Inicialmente calcula-se o valor  $\Delta x$ , que representa o deslocamento total especificado, isto é, a distância entre o ponto inicial ( $P_i$ ) e o ponto final ( $P_f$ ). A aplicação desse  $\Delta x$  na equação 4.7 gera um vetor  $\Delta \theta$  que não representa o valor angular de cada junta para o movimento desejado, pois como explicado anteriormente, o modelo só é válido para pequenos deslocamentos. Este  $\Delta \theta$  é utilizado na equação 4.6, que por sua vez gera um ponto intermediário ( $P_1$ ) no espaço cartesiano, então, um novo  $\Delta x$  é calculado e o ciclo se repete até que o  $\Delta \theta$  seja menor que uma tolerância estipulada, e nesse momento, considera-se que o processo

convergiu.

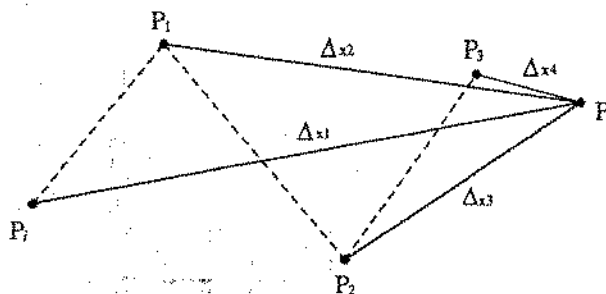


Figura 4.13: Processo de inversão do Jacobiano.

O algoritmo pára quando  $\Delta x$  assume um valor menor que um erro de convergência, no caso, fixado em 0.01mm ou se o número de iterações ultrapassar o valor estipulado para aquele movimento. Nesse caso diz-se que o sistema não convergiu.

O valor das variáveis articulares no ponto objetivo ( $P_f$ ) é encontrado a partir da integração dos  $\Delta\theta_i$ . Conseqüentemente:

$$\theta_f = \theta_0 + \sum_{i=1}^n (\Delta\theta_i), \quad (4.8)$$

onde:

- $\theta_f$  é o vetor das variáveis articulares relativas ao ponto objetivo  $P_f$
- $\theta_0$  é o vetor das variáveis articulares relativas a origem do deslocamento
- $\Delta\theta_i$  é o vetor de variação das variáveis articulares no incremento  $i$
- $n$  é o número de incrementos

O grande dilema do problema cinemático é a inversão do *Jacobiano*. Além do processo numérico ser mais lento, algumas situações desconfortáveis podem ocorrer.

Se numa determinada configuração o determinante do Jacobiano é nulo, diz-se que esta é uma configuração singular. Nessas configurações a inversa do Jacobiano não pode ser definida, portanto essa é uma situação que deve ser evitada. Outro problema ocorre quando a matriz jacobiana não é quadrada, provocando também a impossibilidade de obtenção de sua inversa.

Uma tarefa qualquer definida no espaço operacional ou cartesiano de um robô, exige no mínimo 6 graus de liberdade para sua execução [Ferreira,87]. Denotando  $m$  como o número de graus de liberdade necessários para a execução de uma tarefa e  $n$  o número de graus de liberdade do robô, as seguintes afirmações são válidas:

- Se  $n < m$ , então não existe solução para o sistema;
- Se  $n = m$ , então dois casos podem ocorrer:
  1. Existe um número finito de soluções fora das configurações singulares
  2. Existe um número infinito de soluções nas configurações singulares [Gorla,84]
- Se  $n > m$ , então o robô é redundante e existe um número infinito de soluções. Para a obtenção de uma solução única, introduz-se restrições adequadas ao espaço operacional, ou usa-se um método que minimize algum critério, por exemplo a energia cinética do movimento.

Há ainda o problema da matriz  $J$  ser mal-condicionada, o que pode levar o sistema a resultados indesejados e também o problema de não convergência do método utilizado.

Para enfrentar problemas dessa natureza, lança-se mão de artifícios como a pseudo-inversa e a inversa generalizada, que são ferramentas utilizadas por alguns métodos numéricos para inversão do Jacobiano.

Diversos métodos tem sido propostos na literatura com esse fim, porém, a escolha de um algoritmo específico deve levar em consideração o tipo de aplicação para a qual este será utilizado. Determinadas aplicações necessitam de uma convergência rápida, não importando muito os pontos intermediários gerados pelo algoritmo. Um exemplo de aplicação desse tipo é a identificação de parâmetros de um manipulador.

Outras aplicações, tais como, as de soldagem, exigem que o manipulador se mova ao longo de uma linha reta, e para isso, é interessante que o algoritmo seja capaz de gerar uma trajetória bem comportada. Fayan [Fayan,92] [Dias,93] estudou o comportamento de alguns algoritmos numéricos para inversão da matriz jacobiana e obteve resultados

bastante interessantes, que foram aproveitados para o desenvolvimento do módulo de geração cartesiana de trajetória.

esses resultados também foram aproveitados em um estudo paralelo que está sendo realizado na Unicamp no sentido de projetar e implementar um ambiente e uma linguagem de programação de robôs. A automatização do manipulador Kraft, também neste caso, é o objetivo do projeto [Coutinho,93].

## 4.5 Conclusões

Este capítulo abordou as três formas de geração de trajetória que o sistema proposto é capaz de realizar. Foi apresentado também a estratégia para interpolação e filtragem de trajetórias ponto-a-ponto e o processo de geração cartesiana de trajetórias.

O capítulo seguinte irá mostrar a metodologia utilizada para a execução de testes e validação do sistema. Alguns resultados experimentais também serão apresentados.



## Capítulo 5

# Implementação e Validação do Sistema

### 5.1 Introdução

Este capítulo tem como finalidade apresentar a metodologia e o aparato experimental utilizado para a execução de testes do sistema proposto e desenvolvido. Basicamente, as unidades testadas foram:

- O hardware de comunicação entre o microcomputador e o controlador do manipulador Kraft (ICMK) e as placas de conversão A/D e D/A;
- O módulo de Geração de Trajetória, através da gravação ponto-a-ponto.

### 5.2 Teste do Hardware de Comunicação

Os testes executados no software CMK e na interface de hardware ICMK estão, de certa forma, integrados, visto que para testar convenientemente o ICMK, é necessário que o software desenvolvido (CMK) esteja em funcionamento.

Uma facilidade do CMK que se mostrou bastante útil, foi o módulo chamado *Teste de Hardware*. Este módulo permite testar não somente a interface ICMK como também as placas de conversão A/D e D/A e os sinais digitais.

### 5.2.1 Funcionamento das Placas A/D e D/A

Para verificarmos o funcionamento adequado das placas A/D e D/A, a montagem mostrada na fig. 5.1 é realizada<sup>1</sup>.

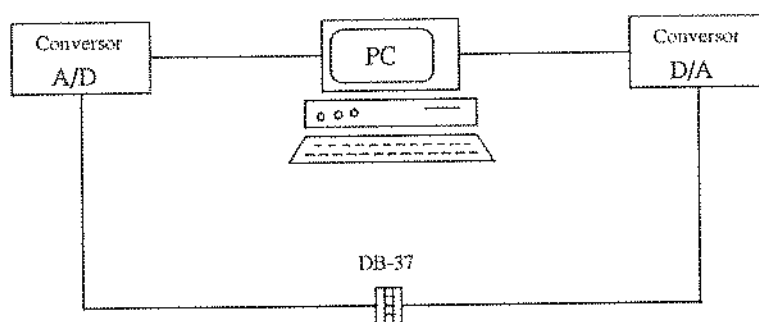


Figura 5.1: Ligação do sistema para teste das placas A/D e D/A.

Inicialmente o PC é colocado no modo *Teste de Hardware*, então aparece uma mensagem para que as placas de conversão A/D e D/A sejam interligadas. A saída da placa D/A, de conector DB-37 é ligada à entrada (também de conector DB-37) da placa A/D, e inicia-se um processo de geração/conversão/comparação.

O microcomputador envia sinais na forma digital para a placa D/A que os converte para a forma analógica e que por sua vez, através da placa A/D são reconvertidos para digital e enviados de volta ao microcomputador, que faz a comparação entre os valores enviado e recebido e emite uma tabela de erros para cada canal medido. Se um desses valores de erro ultrapassa um determinado valor de tolerância, a conversão é considerada incorreta e uma mensagem para que seja feita uma verificação nas placas é enviada. As entradas e saídas digitais podem ser testadas de forma semelhante.

Os conectores foram montados de forma a permitir a execução desse teste, que por ser um módulo permanente no sistema, se caracteriza como uma inestimável ajuda

<sup>1</sup>Esse teste foi executado quando da implementação do sistema no manipulador Kraft

ao operador na detecção e eliminação de possíveis problemas de mal-funcionamento do mesmo.

### 5.2.2 Chaveamento dos Modos de Operação

Com o sistema montado na forma normal, o microcomputador também é capaz de verificar se o chaveamento entre os três modos de operação (*Monitoramento, Controle e Simulação de Vôo*) estão sendo realizados corretamente. Este envia um sinal de controle para que o ICMK chaveie o sistema num dos três modos e a seguir, envia um trem de pulsos de frequência baixa de forma a fazer piscar o led do referido modo de operação.

Um outro teste que foi realizado consistiu em substituir o *master* por um gerador de funções e o *slave* por um osciloscópio, com o objetivo de injetar sinais conhecidos no sistema de forma a possibilitar a verificação do nível de ruído introduzido pelo conjunto PC/interface ICMK, além de observações acerca da taxa de amostragem dos sinais e das conversões A/D e D/A.

## 5.3 Teste do Módulo de Geração de Trajetória

Para a validação do sistema, optou-se pela avaliação do seu comportamento durante a execução de uma trajetória criada e filtrada no modo ponto-a-ponto.

### 5.3.1 Obtenção das Trajetórias

O procedimento adotado para a obtenção de trajetórias suaves e posterior realização do teste final do sistema, utilizando essas trajetórias está descrito abaixo:

1. Gerar uma trajetória ponto-a-ponto (.PTO), por exemplo, via programação *off-line*;
2. A partir dessa trajetória ponto-a-ponto realizar uma interpolação linear, como função da taxa de amostragem dos sinais, gerando assim uma trajetória interpolada (.INT);

3. Submeter a trajetória interpolada à janela triangular<sup>2</sup> de forma a obter uma trajetória final filtrada (.FIL);
4. Com o auxílio de softwares de visualização gráfica, verificar se a trajetória filtrada corresponde ao que a literatura considera como ideal para um melhor comportamento dos controladores das juntas do manipulador (fig. 4.5);
5. A partir de uma bancada de testes, verificar a resposta do controlador a essa trajetória filtrada.

Para exemplificarmos os procedimentos adotados, uma trajetória foi escolhida para movimentar a junta 1 de uma excursão de 60 graus. A decisão de movimentar apenas uma junta deu-se pelo fato de simplificar a análise dos resultados.

Com esse objetivo, a trajetória ponto-a-ponto escolhida é definida pela tabela 5.1:

PONTO	POSICÃO ANGULAR (graus)
1	0
2	30
3	0
4	-30
5	0

Tabela 5.1: Trajetória utilizada para os testes.

A figura 5.2 sintetiza todo o procedimento para geração da trajetória final, desde o arquivo ponto-a-ponto até o arquivo filtrado. Um módulo de testes de colisão poderia ser implementado no software de controle, tornando, assim o sistema mais completo e sofisticado. A trajetória interpolada foi submetida ao tratamento de filtragem e um

<sup>2</sup>Uma janela retangular também foi implementada no sistema. Esse filtro tem uma característica de suavização melhor que o triangular

arquivo de pontos (.FIL) foi gerado, resultando na curva da figura 5.3, que mostra um perfil típico das trajetórias geradas através da utilização desse processo.

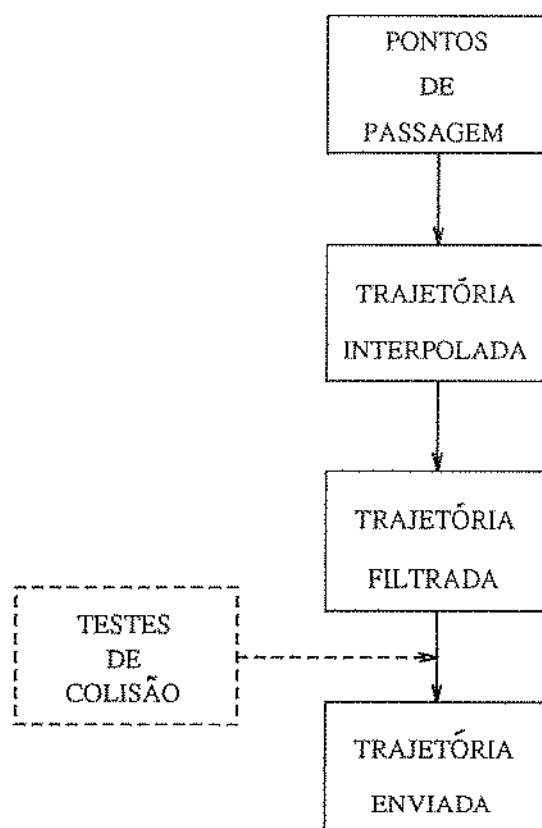


Figura 5.2: Estrutura do sistema de geração de trajetória ponto-a-ponto.

Essa curva filtrada foi comparada com aquelas que a literatura considera como ideais para a obtenção de um movimento suave (fig. 4.5). Portanto, chegou-se à conclusão que o sistema de gravação ponto-a-ponto atingiu seus objetivos plenamente.

De forma a obtermos uma melhor avaliação das características do controle, é proposta uma análise de sincronização das juntas, obtida a partir da curva gerada e apresentada na figura 5.4. Se durante o processo de geração da trajetória tomou-se o cuidado de se garantir a sincronização do movimento das juntas e se o controle à trajetória imposta for eficiente, o perfil dessa curva deverá se aproximar de uma reta.

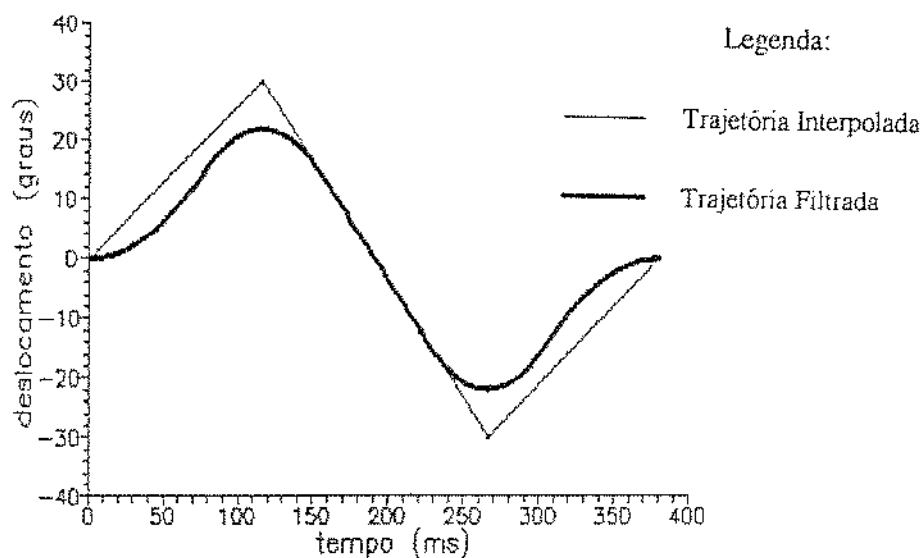


Figura 5.3: Trajetória interpolada e filtrada.

### 5.3.2 Validação do Sistema

Visando uma melhor avaliação do software desenvolvido, decidiu-se submeter o manipulador Kraft às trajetórias interpoladas e filtradas dos passos anteriores de maneira a medir o quão fielmente o manipulador responde a cada uma dessas trajetórias. Esse teste, que será realizado no Centro de Pesquisas da Petrobrás (CENPES) no Rio de Janeiro, tem como objetivo justificar o esforço de implementação do sistema de filtragem.

Para a sua realização será necessário a seguinte bancada:

- Dois microcomputadores compatíveis com a linha IBM-PC-AT;
- A interface ICMK;
- O manipulador Kraft.

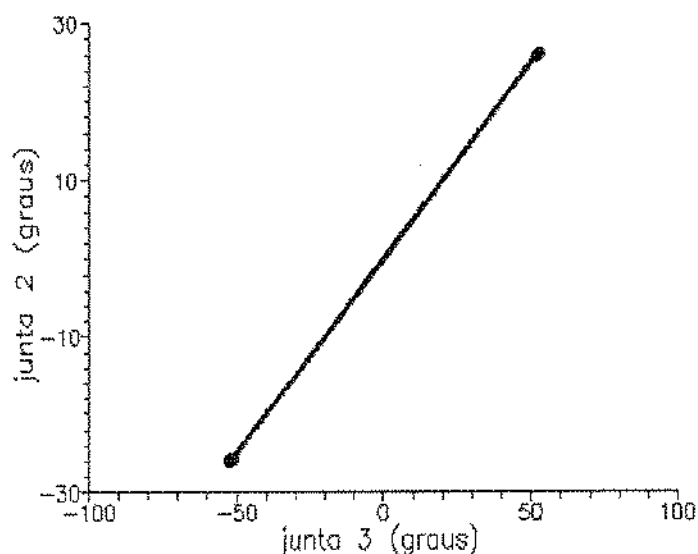


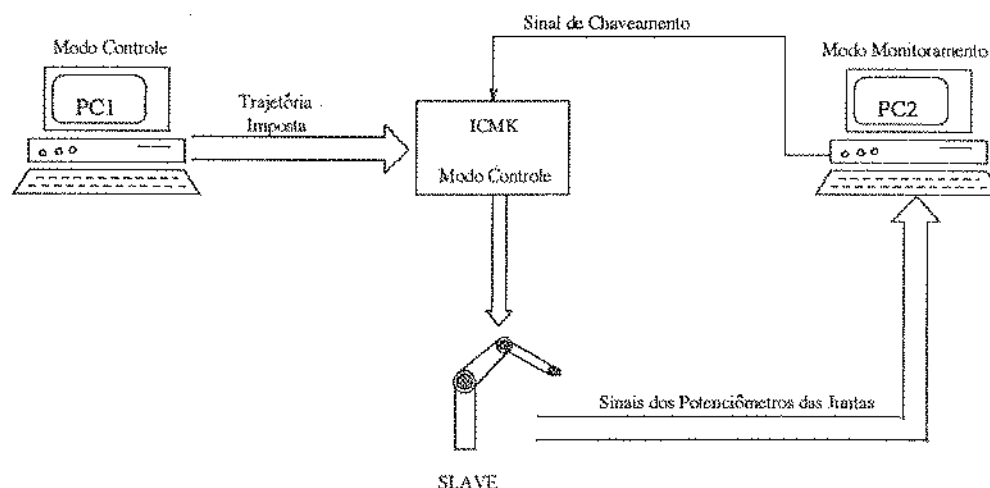
Figura 5.4: Sincronização do movimento das juntas 2 e 3.

### 5.3.2.1 Procedimento

Para a validação dos algoritmos implementados, uma estrutura como a mostrada na figura 5.5 será montada. Ambos os computadores estarão executando o software CMK, estando o PC1 no modo *controle* (enviando uma trajetória) e o PC2 no modo *monitoramento* (lendo e gravando os sinais do slave). A idéia é comparar os sinais enviados pelo PC1 com os sinais realizados pelo slave e lidos pelo PC2, de forma a avaliar se, de fato, os controladores das juntas respondem melhor ao sinal filtrado.

Para essa montagem teremos que instalar a placa D/A no PC1 e a placa A/D no PC2. Quando o programa CMK está setado em um modo de operação, ele chaveia automaticamente a interface ICMK para aquele modo, através de um sinal digital enviado pela placa A/D.

Visto que a interface ICMK deverá estar chaveando o sistema no modo *controle*



**Figura 5.5:** Esquema da bancada para teste de verificação do comportamento do manipulador ao executar uma trajetória interpolada e uma filtrada.

(único modo que possibilita o envio de trajetórias), e que o sinal digital para chaveamento da interface provém da placa A/D, que estará instalada no PC2, que por sua vez estará setado no modo *monitoramento* (necessário para gravar o arquivo de pontos), uma alteração no programa CMK que estará rodando no PC2 terá que ser feita de forma que quando este microcomputador é posto para monitorar, ele seta o ICMK não mais para *monitoramento*, mas sim, para *controle*.

Com os arquivos resultantes desse experimento, calcularemos o sinal de erro entre a trajetória enviada e a executada. Esperamos concluir com esse resultado que o sistema corresponde de forma satisfatória às nossas expectativas, ou seja, que as trajetórias impostas sejam capazes de suavizar o movimento do manipulador, aumentando sua precisão na execução de tarefas automatizadas.



## Comentários Finais e Conclusões

Este trabalho apresentou o desenvolvimento de um sistema de geração automática de trajetórias voltado para a utilização em manipuladores tele-operados. Algumas características desse sistema são considerados a seguir:

- Característica de movimento suave com respeito a velocidade e a aceleração das juntas;
- Possibilidade de programação on-line ou off-line;
- Dois modos de geração de trajetória on-line: Geração no espaço cartesiano e no espaço interpolado de juntas

Para a tornar possível a implementação deste sistema, foi desenvolvida e apresentada uma interface de comunicação entre o manipulador Kraft e um microcomputador compatível com a linha IBM-PC-AT. Também foi feito todo um trabalho de determinação do modelo geométrico dos robôs.

Mesmo na impossibilidade de se testar o sistema completo instalado no manipulador Kraft, considera-se os resultados obtidos satisfatórios, visto que o sistema alcançou a totalidade dos objetivos inicialmente propostos.

As próximas etapas deste trabalho devem estar voltadas para a utilização de métodos diferenciados para a realização da filtragem dos arquivos de pontos da trajetória e para a provável possibilidade de se verificar a resposta do manipulador a arquivos tratados e não tratados.

Também é motivo de estudos, a viabilidade do desenvolvimento de um sistema de tratamento de colisões em tempo-real durante a execução do algoritmo de modelagem

inversa, para ser implantado no mesmo manipulador, dando a este um grau ainda maior de segurança.

Finalmente, a partir do sistema proposto neste trabalho, será possível a utilização de um manipulador submarino, inicialmente projetado para o controle do tipo master/slave, em um ambiente hostil executando tarefas automatizadas de uma forma simples e eficiente.

# Apêndice A

## Listagem da Rotina de Interpolação e Filtragem

```
programa filtro;

uses graph , crt ;

type
  pt    = array[1..3 ] of real ;
  filtre_t = record
    a : integer ;
    k : array[-50..50] of real ;
  end ;

var
  t_int : array[-100..700] of pt ;
  t_fil : array[-50..300] of pt ;
  i ,j,errorcode , no_pt : integer ;
  nb_total : integer ;
  filtre : filtre_t ;
  tecla : char ;

const
```

```
t_obj : array[1..7] of pt = ((10 , 10 , 10 ) ,  
                             (15 , 200 , 10 ) ,  
                             (70 , 80 , 10 ) ,  
                             (70 , 80 , 10 ) ,  
                             (300 , 500 , 10 ) ,  
                             (450 , 50 , 10 ) ,  
                             (600 , 430 , 10 ) );
```

```
device: integer = 4 ;  
mode   : integer = 1 ;  
vmax   : real    = 10 ;  
stab   : integer = 0 ;
```

```
procedure drw_obj ;
```

```
begin  
  for i := 1 to 3 do  
    begin  
      setcolor ( 2 ) ;  
      line (round(t_obj [ i ] [1]) , round(t_obj [ i ] [2]) ,  
            round(t_obj [ i+1 ] [1]) , round(t_obj [ i+1 ] [2] )) ;  
    end;  
end ;
```

```
procedure putx ( x , y , cor : integer ) ;
```

```
var  
  i , j : integer ;  
begin  
  setcolor ( cor ) ;  
  y := y * 55 div 100 ;  
  line ( x - 1 , y , x + 1 , y ) ;  
  line ( x , y - 1 , x , y + 1 ) ;
```

```
end ;
```

```
procedure putxx ( x , y , cor : integer ) ;
```

```
var
```

```
    i , j : integer ;
```

```
begin
```

```
    setcolor ( cor ) ;
```

```
    y := y * 55 div 100 ;
```

```
    line ( x - 2 , y , x + 2 , y ) ;
```

```
    line ( x , y - 2 , x , y + 2 ) ;
```

```
end ;
```

```
procedure interpola ( pd , pf : pt ) ;
```

```
var
```

```
    delta ,
```

```
    inc : pt ;
```

```
    dmax : real ;
```

```
    i , j , nb , aux : integer ;
```

```
begin
```

```
    dmax := 0 ;
```

```
    for i := 1 to 3 do
```

```
        begin
```

```
            delta [ i ] := pf [ i ] - pd [ i ] ;
```

```
            if abs ( delta [ i ] ) > abs ( dmax ) then dmax := delta [ i ] ;
```

```
        end ;
```

```
    nb := round ( abs(dmax) / vmax ) - 1 ;
```

```
    for i := 1 to 3 do inc [ i ] := delta [ i ] / nb ;
```

```
    no_pt := no_pt + 1 ;
```

```
    t_int [ no_pt ] := pd ;
```

```
    putxx ( round ( t_int [ no_pt ] [ 1 ] )
```

```
        , round ( t_int [ no_pt ] [ 2 ] ) , 15 ) ;
```

```

aux := stab;
if pd[1]=pf[1] then
  if pd[2]=pf[2] then
    if pd[3]=pf[3] then  stab := 2*filtre.a;
if stab > 0 then
  begin
  for j := 1 to stab do
    begin
      no_pt := no_pt + 1 ;
      t_int[no_pt]:=t_int[no_pt-1];
    end ;
  end ;
stab := aux;
for j := 1 to nb - 1 do
  begin
    no_pt := no_pt + 1 ;
    for i := 1 to 3 do t_int [ no_pt ][ i ] := t_int [ no_pt - 1 ][ i
      inc [ i ] ;
    putx ( round (t_int [ no_pt ][ 1 ] )
      , round (t_int [ no_pt ][ 2 ] ) , 6 ) ;
  end ;

end ;

```

```

procedure traj_int ;
var
  i : integer ;
begin .
  no_pt := 0 ;
  t_int [ no_pt ] := t_obj [ 1 ] ;
  for i := 1 to 6 do interpola ( t_obj [ i ] , t_obj [ i + 1 ] ) ;
  no_pt := no_pt + 1 ;

```

```

t_int [ no_pt ] := t_obj [ 7 ] ;
putxx ( round ( t_int [ no_pt ] [ 1 ] )
      , round ( t_int [ no_pt ] [ 2 ] ) , 15 ) ;
for i := 1 to 40 do
  begin
    t_int [ no_pt + i ] := t_int [ no_pt ] ;
  end ;
nb_total := no_pt + 40 ;

end ;

procedure get_filtre ;
var
  g , pente , ar : real ;
  i : integer ;
begin
  with filtre do
    begin
      write ( ' a = ' ) ; read ( a ) ;
      ar := a ;
      pente := 1 / ar / ar ;
      for i := - 50 to 50 do k[i] := 0 ;
      for i := -a to 0 do k[i] := pente * ( i + ar ) ;
      for i := 1 to a do k[i] := - pente * ( i - ar ) ;
    end ;
  end ;
end ;

procedure traj_filtre ;
var
  i , j , l : integer ;
begin
  for i := -100 to 0 do t_int [ i ] := t_int [ 1 ] ;

```

```

for i := nb_total+1 to nb_total + 50 do t_int [ i ] := t_int [ nb_tot

for i := -filtre.a to nb_total do
  begin
    for j := 1 to 3 do
      begin
        t_fil [ i ][ j ] := 0 ;
        for l := - filtre.a to filtre.a do
          begin
            t_fil [ i ][ j ] := t_fil [ i ][ j ] +
                                t_int [ i + 1 ][j] * filtre.k[l] ;
          end ;
        end ;
        putx ( round ( t_fil [ i ][ 1 ] ) , round (t_fil [ i ][ 2 ] ) , 13
        end ;
      end ;
    end ;
  end ;
end ;

```

```

begin
  directvideo := false ;
  device := detect ;
  initgraph ( device , mode , ' ' ) ;
  errorcode := graphresult;

  repeat
    gotoxy ( 2 , 19 ) ;
    writeln ( ' ' );
    writeln ( ' ' );
    writeln ( ' ' );
    writeln ( ' ' );
    writeln ( ' ' );
    write ( ' ' );
    gotoxy ( 2 , 19 ) ;
  repeat

```



```
write ( 'Vmax = ' ) ; read ( vmax ) ;
get_filtre ;
write ( ' estabilizacao = ' ) ; read ( stab ) ;
tecla := ' ' ;
traj_int ;
traj_filtre ;

if ( tecla = 'c' ) or ( tecla = 'C' ) then cleardevice ;
writeln ;
write ( ' T : para terminar , C para apagar , uma outra tecla para co.
writeln ;
tecla := readkey ;
until ( tecla = 't' ) or ( tecla = 'T' ) ;

closegraph ;

end .
```

# Apêndice B

## Características Técnicas

### .1 Características do Manipulador Kraft

#### .1.1 Especificações Gerais

Máximo Alcance	130 cm
Capacidade de Carga com o Braço Estendido	34 kg
Graus de Liberdade de Movimento	Seis (+ abrir/fechar garra)

#### .1.2 Efetuador

Padrão	Garra Paralela
Abertura da Garra	10 cm
Força de Fechamento da Garra	90 kg
Método de Controle	Controle de Força Proporcional

#### .1.3 Movimento das Juntas

eixos	Excursão(gr)	Velocidade (gr/s)
-------	--------------	-------------------

Shoulder Azimuth	180	80
Shoulder Elevation	120	65
Elbow Pivot	110	50
Wrist Pitch	100	100
Wrist Yaw	105	115
Wrist Rotate		
Modo 1 (master)	180	200
Modo 2 (Contínuo)	-	210

## .2 Características do Robô Manutec r3

### .2.1 Especificações Gerais

Acionamento	Elétrico
Alimentação	220 V, 3 fases, 50 - 60 Hz
Graus de Liberdade de Movimento	Seis
Capacidade de Carga	34 kg
Momento Nominal sobre o eixo 6	22 N.m
Momento de Inércia sobre o eixo 6	0,94 kg.m <sup>2</sup>
Repetibilidade	± 0,1 mm
Pêso	920 kg

### .2.2 Velocidades

Ponto a Ponto	5.9 m/s
Linear	1.5 m/s

### .2.3 Acelerações

Ponto a Ponto:

Máx. Centrífuga

10 m/s<sup>2</sup>

Máx. Tangencial

9.8 m/s<sup>2</sup>

Linear: Máx. em caminho contínuo

6 m/s<sup>2</sup>

### .2.4 Movimento das Juntas

eixos	Excursão(gr)	Velocidade (rpm)
1	± 165	33
2	± 110	16
3	± 195	33
4	± 190	60
5	± 120	66
6	± 265	46

### .3 Características da Placa A/D Utilizada

Canais de Entrada Analógica	
Simples	16
Diferenciais Multiplexadas	8
Canais de Saída Analógica	até 4
Canais de Entrada Digital	16
Canais de Saída Digital	16
Resolução do Conversor A/D	12 bits (4096 níveis)
Contadores Programáveis/Temporizadores	3 (16 bits)
Oscilador a Cristal	2 MHz

A placa conta ainda com as seguintes características:

- Sequência de leitura e ganhos programáveis através de memória de canais;
- Aquisição em “Burst” propiciada por buffer (FIFO) de 16 posições;
- Suporte para DMA (Direct Memory Access: Acesso Direto à Memória), permitindo a velocidade máxima de coleta de sinais independente da UCP do microcomputador;
- Possibilidade de, com o auxílio de circuitos externos, realizar amostragem simultânea de até 16 canais;
- Suporte a interrupções.

## 4 Características da Placa D/A Utilizada

Canais de Saída Analógica	8
Faixas de Saída Configuráveis	
Bipolares	$\pm 10,0$ V $\pm 5,0$ V $\pm 2,5$ V
Unipolares	0 a 10,0 V 0 a 5,0 V
Mínima Carga de Saída	10 KOhm
Resolução	12 bits (4096 níveis)
Tipos de Representação Decimal	Complemento de Dois Binário com Off-set
Tipos de Atualização das Saídas	Individual por Canal Simultânea Comandada pela UCP Simultânea comandada pelo Contador
Tempo de conversão	4 $\mu$ s (máximo)
Contadores Programáveis	3
Oscilador a Cristal	2 MHz
Endereçamento Ajustável	de 0200h a 03F8h
Espaço de Endereçamento	8 Posições
Interrupções Seleccionáveis	IRQ2, IRQ3, IRQ4, IRQ5

## 5 Esquema Eletrônico da Placa ICMK

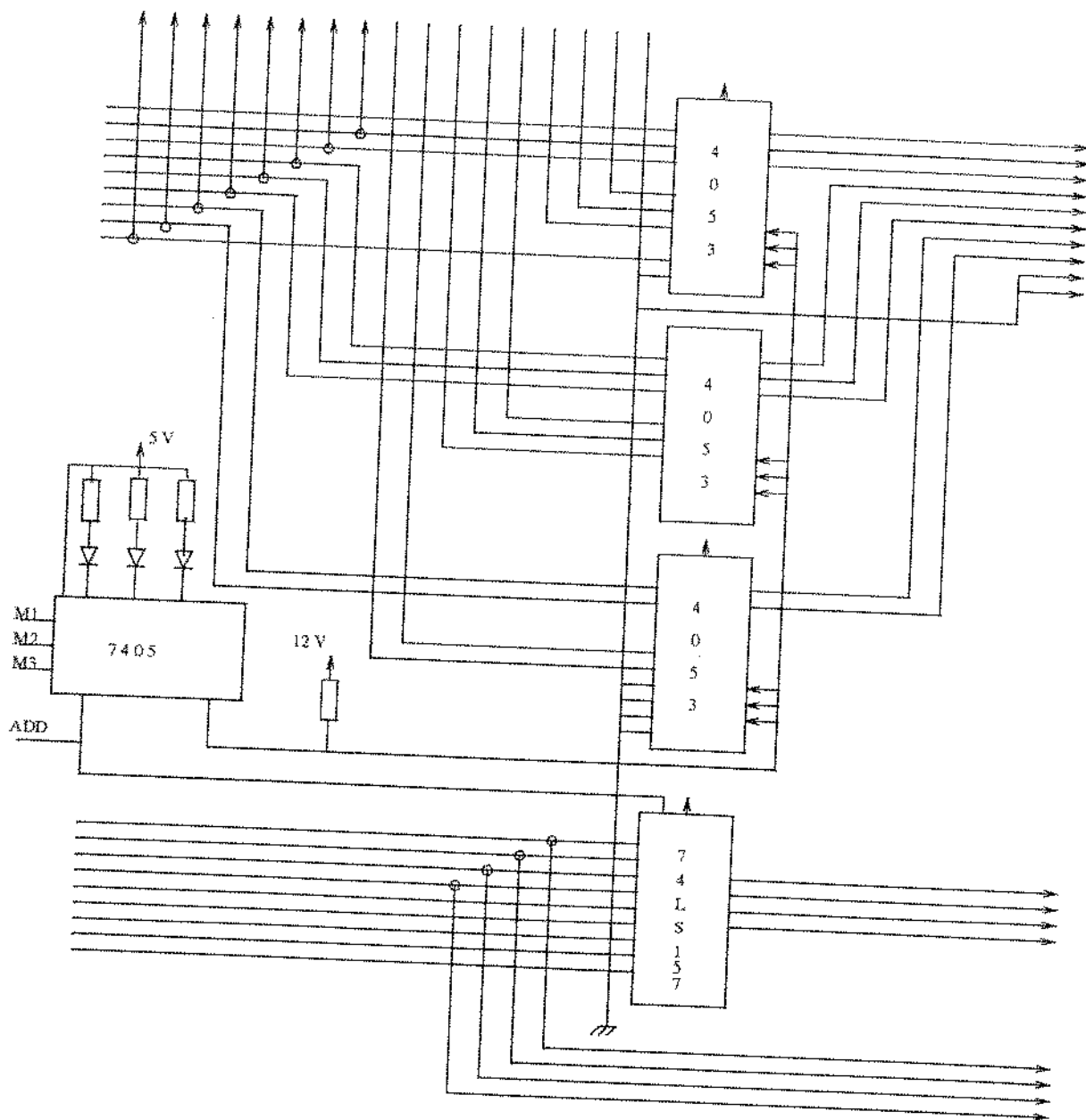


Figura .6: Circuito eletrônico da placa ICMK

# Apêndice C

## Manipulador Submarino Kraft - Matrizes de Passagem $A_{i-1,i}$

Definindo  $c\theta_i = \cos(\theta_i)$  e  $s\theta_i = \sin(\theta_i)$ , temos:

$$A_{0,1} = \begin{bmatrix} c\theta_1 & 0 & s\theta_1 & 0 \\ s\theta_1 & 0 & -c\theta_1 & 0 \\ 0 & 1 & 0 & d_1 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$A_{1,2} = \begin{bmatrix} c\theta_2 & -s\theta_2 & 0 & a_2.c\theta_2 \\ s\theta_2 & c\theta_2 & 0 & a_2.s\theta_2 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$A_{2,3} = \begin{bmatrix} c\theta_3 & -s\theta_3 & 0 & a_3.c\theta_3 \\ s\theta_3 & c\theta_3 & 0 & a_3.s\theta_3 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$A_{3,4} = \begin{bmatrix} c\theta_4 & 0 & -s\theta_4 & a_4.c\theta_4 \\ s\theta_4 & 0 & c\theta_4 & a_4.s\theta_4 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$A_{4,5} = \begin{bmatrix} -s\theta_5 & 0 & c\theta_5 & 0 \\ c\theta_5 & 0 & s\theta_5 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$A_{5,6} = \begin{bmatrix} c\theta_6 & -s\theta_6 & 0 & 0 \\ s\theta_6 & c\theta_6 & 0 & 0 \\ 0 & 0 & 1 & d_6 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$



# Bibliografia

- [Aust,90] Aust, E.; Boke, M.; Gustmann, M.; Hahn, G.; Niemann, H.R.; Schultheiss, G.F.: "Development and Testing of a Freely Programmable Robot for Work in Deep Water", SPE 1990 - Latin American Petroleum Conference - IV Congresso Brasileiro de Petroleo - Rio Centro, Rio de Janeiro, BRAZIL, PETROBRÁS,IBP,SPE, pp.01-18, 1990.
- [Campos,93] Campos, R., "Implementação de um Mecanismo de Calibração e Identificação de Parâmetros para Robôs".,Tese de Mestrado, Faculdade de Engenharia Mecânica-Unicamp, Campinas, SP, 1993
- [Coutinho,93] Coutinho, L. A. F., "Um Ambiente Integrado de Desenvolvimento de Software Aplicado a Robótica".,Tese de Mestrado, Faculdade de Engenharia Mecânica-Unicamp, Campinas, SP, 1993
- [Craig,89] Craig, J. J., "Introduction to Robotics : Mechanics and Control," Addison-Wesley, 2ed., 1989.
- [Denavit,55] Denavit, J., Hartenberg, R. S., "A Kinematic Notation for Lower-Pair Mechanisms Based on Matrices," in *Journal of Applied Mechanics*, pp. 215-221, Jun,1955.
- [Dias,93] Dias, C. H., "Implementação Experimental de um Supervisor de Controle para Robôs Industriais".,Tese de Mestrado, Faculdade de Engenharia Mecânica-Unicamp, Campinas, SP, 1992
- [Fayan,92] Fayan, B. L., "Estudo e Especificação de um Supervisor de Controle para um Robô Industrial".,Tese de Mestrado, Faculdade de Engenharia Mecânica-Unicamp,

Campinas, SP, 1992

- [Ferreira,87] Ferreira, E. P., "Robótica", II Escola Brasileiro-Argentina de Informática, Buenos Aires, Argentina, 1987
- [Fu,87] Fu, K. S., Gonzalez, R. C., Lee, C. S. G. "Robotics Control, Sensing, Vision, and Intelligence," McGraw-Hill,inc., 1987.
- [Gorla,84] Gorla, B.,Renaud, M., "Modèles des Robots Manipulateurs: Application à Leur Command", Cepadues Editions, Toulouse, 1984.
- [Groover,84] Gruver, W. A., "Industrial Robot Programming Languages: A Comparative Evaluation," in *IEEE Transactions on System Man and Cybernetics*, Vol. SMC-14, n. 4, pp. 565-570, 1984.
- [Kelley,89] Kelly-Bootle, S., "Mastering Turbo C," Sybex, Inc., 1989.
- [Ker,78] Kernighan, B., Ritchie, D., "The C Programming Language," Prentice-Hall, 1978.
- [Kraft,85] Kraft, "Underwater Manipulator System", 1985;
- [Lee,84] Lee, C. S. G., and Ziegler, M., "A Geometric Approach in Solving the Inverse Kinematics of PUMA Robots", in *IEEE Trans. Aerospace and Electronic Systems*, vol. AES-20, No. 6, pp. 695-706, 1984.
- [Lynx,89] Lynx Tecnologia Eletrônica Ltda, "CAD 12/36 Manual do Usuário e de Referência", 1989;
- [Lynx,91] Lynx Tecnologia Eletrônica Ltda, "CDA 12/08 Manual do Usuário e de Referência", 1991;
- [Miss,92] Miss, R. W., Schultheiss, G. F. "The Design of an Algorithm For the Control of Redundant Kinematic Chains", 5th ICAR, Pisa, 1992.
- [Niemann,92] Niemann, H. R.; Aust, Gustmann, M.; E.; Hahn, G.: "A Six DOF Robot Allows Diverless Intervention", GKSS 92/E/44, GERMANY, 1992.

- [Oppenheim,89] Oppenheim, A. V.;Schaffer R. W.; “Discrete-time Signal Processing”, Prentice-Hall International inc.; 1989
- [Paul,81] Paul, R. P., “Robot Manipulator: Mathematics, Programming and Control”, MIT Press, Cambridge, Mass, 1981.
- [Pieper,68] Pieper, D. L., “The Kinematics of Manipulators under Computer Control”, *Artificial Intelligence Project Memo No. 72.*, Computer Science Department, Stanford University, Palo Alto, CA, 1968.
- [Rosário,91] Rosario, J.M.; Weber, H.I.; Morooka, C.: “Development and Control of Advanced Robots for Underwater Tasks”, 5th ICAR, Pisa, Italy, pp. 1792-1795, 1991.
- [Saramago,93] Saramago, M. A. P., “Projeto e Desenvolvimento de um Sistema de Calibração e Medida de Precisão para Robôs Industriais”. Tese de Mestrado, Faculdade de Engenharia Mecânica-Unicamp, Campinas, SP, 1993
- [Schildt,90] Schildt, H., “C The Complete Reference,” McGraw-Hill, Inc., 1990.
- [Snyder,85] Snyder, W. E., “Industrial Robots: Computer Interfacing and Control”, Prentice Hall, Inc., 1985.
- [Souza,92] Souza, J. P., “Procedimento Automático para Aquisição e Tratamento do Movimento de um Robô”. Tese de Mestrado, Faculdade de Engenharia Mecânica-Unicamp, Campinas, SP, 1992