

UNIVERSIDADE FEDERAL DE SANTA CATARINA  
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA MECÂNICA

**ESTUDO DE MÉTODOS PARA A SOLUÇÃO  
DA CINEMÁTICA INVERSA DE ROBÔS INDUSTRIAIS  
PARA IMPLEMENTAÇÃO COMPUTACIONAL**

DISSERTAÇÃO SUBMETIDA À UNIVERSIDADE FEDERAL DE SANTA CATARINA  
PARA A OBTENÇÃO DO GRAU DE MESTRE EM ENGENHARIA MECÂNICA.

**JORGE LUIZ ERTHAL**

FLORIANÓPOLIS, OUTUBRO DE 1992

**ESTUDO DE MÉTODOS PARA SOLUÇÃO DA CINEMÁTICA INVERSA  
DE ROBÔS INDUSTRIAIS PARA IMPLEMENTAÇÃO COMPUTACIONAL**

**JORGE LUIZ ERTHAL**

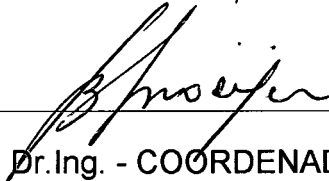
ESTA DISSERTAÇÃO FOI JULGADA PARA OBTENÇÃO DO TÍTULO DE

**MESTRE EM ENGENHARIA**

ESPECIALIDADE ENGENHARIA MECÂNICA E APROVADA EM SUA FORMA FINAL  
PELO PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA MECÂNICA.



Prof. **NELSON BACK**, Ph.D. - ORIENTADOR

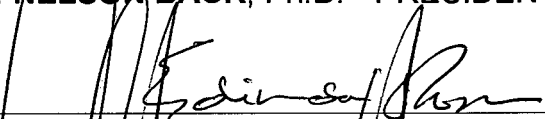


Prof. **BEREND SNOEIJER**, Dr. Ing. - COORDENADOR DO CURSO

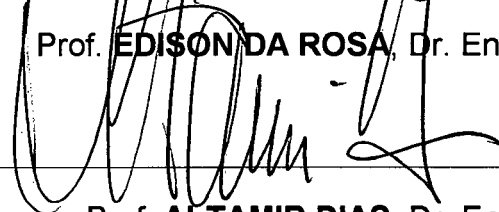
**BANCA EXAMINADORA**



Prof. **NELSON BACK**, Ph.D. - PRESIDENTE



Prof. **EDISON DA ROSA**, Dr. Eng. Mec.



Prof. **ALTAMIR DIAS**, Dr. Eng. Mec.

*À Tata, à Mãe  
e à Sueli,  
com muito carinho.*

## AGRADECIMENTOS

*"Amigo é coisa pra se guardar  
do lado esquerdo do peito,  
mesmo que o tempo e a distância digam não."*

*(Milton Nascimento)*

- Ao Professor Nelson Back, pela orientação e pelo exemplo de profissionalismo, competência, dedicação e humildade.
- Aos colegas do DAMEC, pelo apoio prestado durante o período de afastamento.
- Ao pessoal do Laboratório de Projeto, pelas discussões, pela amizade e pela convivência agradável durante estes anos. Em especial ao Fernando e ao Christian, pela "força".
- Aos meus amigos de "boteco", pelos momentos de descontração, passados, presentes e futuros.
- Ao CEFET-PR e CAPES, por proporcionarem as condições para a realização deste trabalho.
- A todos os que direta ou indiretamente contribuíram para a realização deste trabalho.

## ÍNDICE

RESUMO.....	viii
ABSTRACT.....	ix
<b>1 INTRODUÇÃO.....</b>	<b>1</b>
<b>2 CONCEITOS E DEFINIÇÕES SOBRE CINEMÁTICA DE ROBÔS.....</b>	<b>6</b>
2.1 Introdução.....	6
2.2 Definições e Características dos Robôs Industriais.....	6
2.3 Estrutura de um Sistema Robótico.....	10
2.3.1 Estrutura Mecânica.....	11
2.3.2 Sistemas de Controle.....	14
2.3.3 Unidades de Acionamento.....	15
2.4 Parâmetros Básicos.....	15
2.4.1 Espaço de Trabalho.....	16
2.5 Programação.....	21
2.6 Planejamento da Trajetória.....	21
<b>3 CINEMÁTICA DE ROBÔS.....</b>	<b>23</b>
3.1 Introdução.....	23
3.2 Modelagem Cinemática de um Robô.....	24
3.3 Cinemática Direta.....	29
3.4 Cinemática Inversa.....	33
3.5 Existência da Solução.....	35
3.6 Número de Soluções.....	36
<b>4 MÉTODOS PARA A OBTENÇÃO DA CINEMÁTICA INVERSA.....</b>	<b>39</b>
4.1 Introdução.....	39
4.2 Métodos de Solução da Cinemática Inversa.....	39

4.3	Métodos Analíticos.....	40
4.3.1	Método Geométrico.....	41
4.3.2	Métodos Algébricos.....	42
4.4	Métodos Numéricos.....	49
4.4.1	Método da Análise da Posição-Zero.....	49
4.4.2	Método de Newton-Raphson.....	49
4.4.3	Pseudo-Inversa.....	51
4.4.4	Pseudo-Inversa para Regiões Singulares.....	53
4.4.5	Método da Integração das Velocidades.....	53
4.4.6	Cinemática Inversa de Velocidade e Aceleração.....	54
4.5	Métodos Mistos.....	55
4.5.1	Solução para Pulso Não-Esférico.....	55
4.5.2	Manipulador com Sete Graus de Liberdade.....	56
4.6	Comparação entre os Métodos Apresentados.....	56
<b>5</b>	<b>DESCRIÇÃO DO PROGRAMA.....</b>	<b>60</b>
5.1	Introdução.....	60
5.2	Estrutura Global do Programa.....	61
5.2.1	Bloco "ROBÔ".....	62
5.2.2	Bloco "AMBIENTE".....	65
5.2.3	Bloco "TAREFA".....	66
5.2.4	Bloco "VISUALIZAÇÃO".....	71
<b>6</b>	<b>UTILIZAÇÃO DO PROGRAMA E TESTES.....</b>	<b>74</b>
6.1	Introdução.....	74
6.2	Utilização do Programa.....	74
6.3	Testes.....	86
<b>7</b>	<b>CONCLUSÕES E RECOMENDAÇÕES.....</b>	<b>104</b>
7.1	Conclusões.....	104
7.2	Recomendações.....	106

<b>REFERÊNCIA BIBLIOGRÁFICA.....</b>	<b>108</b>
<b>APÊNDICE 1 CÁLCULO DOS PARÂMETROS CINEMÁTICOS.....</b>	<b>117</b>
<b>APÊNDICE 2 REPRESENTAÇÃO GRÁFICA DA CADEIA CINEMÁTICA.....</b>	<b>130</b>
<b>APÊNDICE 3 ESTRUTURA DOS ARQUIVOS DE DADOS.....</b>	<b>138</b>
<b>APÊNDICE 4 CINEMÁTICA INVERSA SEGUNDO O MÉTODO DA ANÁLISE DA POSIÇÃO-ZERO.....</b>	<b>146</b>
<b>APÊNDICE 5 FUNÇÕES QUE CALCULAM A CINEMÁTICA INVERSA.....</b>	<b>155</b>

## RESUMO

Este trabalho apresenta um estudo comparativo de formulações analíticas e numéricas, utilizadas no cálculo da cinemática inversa de posição de manipuladores robóticos. Para implementação computacional, selecionou-se um método vetorial, baseado na Análise da Posição-Zero. O método consiste na minimização dos erros de posição e de orientação do efetuador, por meio da movimentação individual das juntas do robô. Tal movimentação é obtida utilizando o conceito de rotação de um vetor sobre um eixo que, no caso, corresponde ao eixo da junta.

Foram feitos testes para avaliar o comportamento do método quanto à convergência, precisão e estabilidade, em diferentes pontos do espaço de trabalho, inclusive em regimes singulares, onde muitos métodos apresentam problemas. Os resultados mostraram que o método baseado na análise da Posição-Zero apresenta bom desempenho mesmo nas regiões singulares.

É apresentada a proposta de um programa para análise cinemática de manipuladores robóticos utilizando o método testado. Foram desenvolvidas algumas rotinas de entrada de dados e representação gráfica que permitem analisar qualquer configuração de manipuladores robóticos.



## **ABSTRACT**

This work presents a comparative study about analytical and numerical methods used to solve the position inverse kinematics problem of robotic manipulators. The method based on the Zero-Position analysis was chosen to be implemented. It consists on the minimization of a norm by the individual movement of the joints. Such movement is obtained using the rotation of a vector around an axis which, in this case, corresponds to the joint axis.

Tests were accomplished to estimate the behaviour of the method at different points of the workspace, taking account the convergence, precision and stability. Singular regions were investigated. The results showed that this method presents a good performance even in the singular regions.

A proposal of a robot kinematics analysis program is presented using the tested method. Some data input and graphics routines were implemented in order to allow the analysis of any manipulator configuration.

# CAPÍTULO 1

## INTRODUÇÃO

A robótica é uma tecnologia relativamente nova, que vem sendo aplicada na automação da manufatura e está rapidamente substituindo máquinas para fins específicos, a chamada "automação rígida" (*hard automation*) [01]. O crescimento da robótica na indústria, tem-se justificado face às exigências de maior qualidade, produtividade e flexibilidade nos processos fabris. Além disso, a racionalização da área de trabalho, com relação ao espaço ocupado e às condições ambientes, fazem com que a utilização de robôs seja cada vez maior.

Com relação à pesquisa, a robótica possui a característica de ser multidisciplinar, abrangendo várias áreas de interesse. Entre elas, destacam-se a cinemática, a dinâmica, o planejamento de trajetórias, o controle do movimento, os sensores, as linguagens de programação e a inteligência artificial [02].

As ferramentas necessárias são trazidas de várias áreas "clássicas" [03]. A Engenharia Mecânica fornece metodologias para o estudo de máquinas em situações estáticas e dinâmicas. A Física fornece ferramentas para a descrição do movimento espacial e outros atributos ligados à modelagem do manipulador. A Teoria do Controle, contribui para o projeto e avaliação de algoritmos para a obtenção de movimentos e forças desejadas. A Engenharia Elétrica fornece técnicas para o projeto de sensores e interfaces. Finalmente, a Computação fornece a base para a programação de tarefas e as ferramentas necessárias para a simulação.

Até algum tempo atrás, as pesquisas se concentravam na área do controle, visando obter estratégias que fizessem com que os robôs existentes se

tornassem mais ágeis. Pouco se fez no melhoramento ou na alteração de estruturas mecânicas existentes, a ponto delas tornarem-se excessivamente robustas e pesadas para os requisitos de desempenho exigidos pelo controle [04].

Apesar da maioria dos robôs possuírem configuração antropomórfica, existe uma diferença sensível na habilidade de manipulação destes, que se situam bem abaixo da habilidade de um ser humano [01]. A relação entre a carga útil e o peso do robô está na faixa de 2% a 5%, enquanto que, para uma pessoa normal, ela fica entre 20% e 25%, cerca de dez vezes maior. Por isso, atualmente, os esforços estão concentrados na melhoria do desempenho global dos robôs, pela utilização de estruturas mecânicas e estratégias de controle otimizadas [01].

O projeto de um robô, quando executado adequadamente, deve levar a um melhor desempenho, menor peso e menor custo. Chega-se a trajetórias mais suaves e a uma maior precisão devido às menores deformações estáticas e dinâmicas. Os sistemas de controle são mais simples e há um aumento da agilidade do robô. Em seu livro, Rivin [04] salienta que somente um profundo entendimento do comportamento e das modificações potenciais dos sistemas mecânicos a serem controlados, permitiria o desenvolvimento de robôs de desempenho elevado. Nota-se, portanto, a necessidade de se investir na melhoria da estrutura mecânica dos robôs.

Outro problema existente, agora na fase de aplicação, é a seleção do robô mais apropriado e a sua instalação numa célula de trabalho. Cada tarefa possui suas peculiaridades tais como velocidades exigidas, extensão, ambiente onde será executada e precisão. Para cada tarefa existe um robô mais apropriado. À medida que o número de tarefas aumenta, a seleção torna-se cada vez mais complicada. A instalação e a programação de um robô, quando feita diretamente no chão de fábrica, leva de uma a duas semanas [05]. Em se tratando de uma linha de montagem, onde dezenas de robôs trabalham simultaneamente, o tempo de preparo aumenta bruscamente, podendo a programação no local, tornar-se impraticável.

A simulação em computadores vem auxiliando na solução destes problemas. Através dela, pode-se estudar um robô desde a sua fase de projeto até a sua instalação numa célula de trabalho. Para isto, vários programas tem sido desenvolvidos, particularmente para a seleção de robôs e projeto de células de trabalho robotizadas [05 a 23]. A área principal dos programas simuladores é a

programação *off-line*, que é a programação feita em um computador, e não no robô. A programação *off-line* permite, através da simulação num computador, selecionar, testar e programar um robô, antes mesmo que ele tenha sido comprado ou até projetado [05].

Através da simulação, além da definição do arranjo físico, pode-se obter, mediante rotinas matemáticas que modelam o comportamento físico, dados sobre o comportamento cinemático e dinâmico do robô, que auxiliam na definição da trajetória otimizada, em relação ao tempo ou em relação à energia consumida [06]. Deflexões estáticas e dinâmicas podem ser detectadas, definindo-se assim a precisão correspondente [07]. Problemas complexos de programação, envolvendo mais de um robô, podem ser solucionados [14]. Pode-se mapear o espaço de trabalho em termos da carga máxima transportável [23]. Na fase de projeto, pode-se estudar várias estruturas cinemáticas diferentes e os respectivos espaços de trabalho, visando a escolha da estrutura ideal para cada tipo de tarefa. Com base nos torques e forças atuantes na estrutura, pode-se avaliar o comportamento dinâmico, flexibilidade, estabilidade em diferentes condições de movimento, sem despendar muito tempo. Por fim, pode-se utilizar a simulação como uma forma bastante simples e barata de ensinar os conceitos principais sobre robótica. Todas estas aplicações são enriquecidas pela utilização de ferramentas de CAD, tão disseminadas hoje em dia.

Com estes exemplos, pode-se concluir que a modelagem e a simulação são ferramentas poderosas que podem e devem ser aplicadas na pesquisa, treinamento e mesmo no ensino em robótica. Contudo, a modelagem só é eficiente na medida em que ela possa representar os fenômenos da forma mais realista possível [17].

Dentro das áreas de estudo da robótica, a cinemática e a dinâmica formam o ponto de partida para o conhecimento das demais áreas. Na cinemática, estuda-se o movimento do robô sem levar em conta forças e torques que atuam sobre ele. Incluem-se neste item a análise de todas as propriedades do movimento baseadas na geometria e no tempo [03]. Dentro da cinemática, um tópico de destaque é a solução da cinemática inversa, pois dela depende a programação *off-line* de trajetórias e a validação das técnicas de controle que trabalham em tempo real.

Várias propostas de solução da cinemática inversa tem sido apresentadas, todas elas visando a obtenção da solução da forma mais rápida

possível e abrangendo a maior quantidade de robôs. Ocorre, no entanto, que os métodos de cálculo mais rápidos, não são genéricos. Além disso, as soluções específicas abrangem apenas uma classe de arranjos cinemáticos. Se, por um lado, esta solução da cinemática inversa é obtida de uma forma bastante rápida, por outro lado, devido a tal limitação, não se tem explorado outros arranjos cinemáticos que pudessem ter aplicação [01].

Tendo em vista os problemas apresentados e a carência de ferramentas computacionais para pesquisa e ensino da robótica, este trabalho tem o objetivo de apresentar um estudo sobre os métodos utilizados para a solução da cinemática inversa, e apontar, entre os métodos estudados, quais os que teriam condições de ser utilizados em um programa de simulação cinemática de robôs.

É apresentada, também, uma proposta de um simulador cinemático de robôs, dentro do qual está implementado um método numérico e outro analítico, para o cálculo da cinemática inversa.

O conteúdo do trabalho foi elaborado de modo a apresentar, inicialmente, alguns conceitos envolvendo robótica, antes de atacar o problema da cinemática inversa propriamente dito.

No Capítulo 2 estão apresentados os conceitos sobre robótica, que, de algum modo, estão relacionados com a cinemática de robôs.

No Capítulo 3, estão apresentadas algumas técnicas de modelagem cinemática de robôs, bem como as características gerais das cinemáticas direta e inversa.

O Capítulo 4 trata especificamente dos métodos de obtenção da cinemática inversa pesquisados. É feito um levantamento das características principais, vantagens e desvantagens, na tentativa de se chegar a um método adequado para ser utilizado em um simulador.

No Capítulo 5, é apresentada uma proposta de um programa simulador cinemático, no qual se incluiu o cálculo da cinemática inversa. É feita a descrição da estrutura geral do programa proposto e das funções nele implementadas.

A maneira de se utilizar o programa está apresentada no Capítulo 6. Neste capítulo também estão incluídos alguns exemplos e testes de desempenho do método de solução da cinemática inversa implementado.

No Capítulo 7 estão apresentadas as conclusões sobre o estudo e sobre o programa. Algumas recomendações para melhoria do programa e sugestões para trabalhos futuros envolvendo a cinemática são também apresentadas.

Para a modelagem cinemática, foram utilizados dois conjuntos de parâmetros diferentes: Denavit-Hartenberg e Posição-Zero. Tais conjuntos, na maioria dos casos, são de difícil obtenção, por se estar tratando de cadeias cinemáticas espaciais, com até seis graus de liberdade. O Apêndice 1 apresenta três procedimentos para a obtenção automática dos parâmetros.

O procedimento utilizado para a representação gráfica do robô está apresentado no Apêndice 2. Tal procedimento utiliza a representação *wireframe*, e baseia-se nos parâmetros de Denavit-Hartenberg.

O programa utiliza ou gera alguns arquivos de dados, cujas estruturas estão apresentadas no Apêndice 3.

Dentre os métodos de solução da cinemática inversa estudados, selecionou-se o método da análise da Posição-Zero, para implementação no programa. A formulação correspondente está apresentada no Apêndice 4 e a implementação, no Apêndice 5.

## **CAPÍTULO 2**

# **CONCEITOS BÁSICOS E DEFINIÇÕES SOBRE CINEMÁTICA DE ROBÔS**

### **2.1 INTRODUÇÃO**

Este Capítulo tem por objetivo introduzir os conceitos que envolvem a cinemática de robôs e apresentar alguns tópicos que possam ser objeto de pesquisas futuras, visando a otimização do projeto mecânico de robôs.

Para que se possa estudar a cinemática de robôs é necessário que se conheça a estrutura de um sistema robótico, suas características e interdependências. Neste Capítulo são apresentadas as principais características que definem a geometria, os componentes, as estruturas mecânicas existentes e os parâmetros que definem o desempenho de um robô. Algumas definições envolvendo programação e planejamento de trajetória também são apresentados visando mostrar a sua relação com a cinemática.

### **2.2 - DEFINIÇÕES E CARACTERÍSTICAS DOS ROBÔS INDUSTRIAIS**

A fim de evitar confusão entre manipuladores e robôs industriais, é apresentado na Figura 2.1, um fluxograma que situa os robôs dentro da classe dos manipuladores industriais [25].

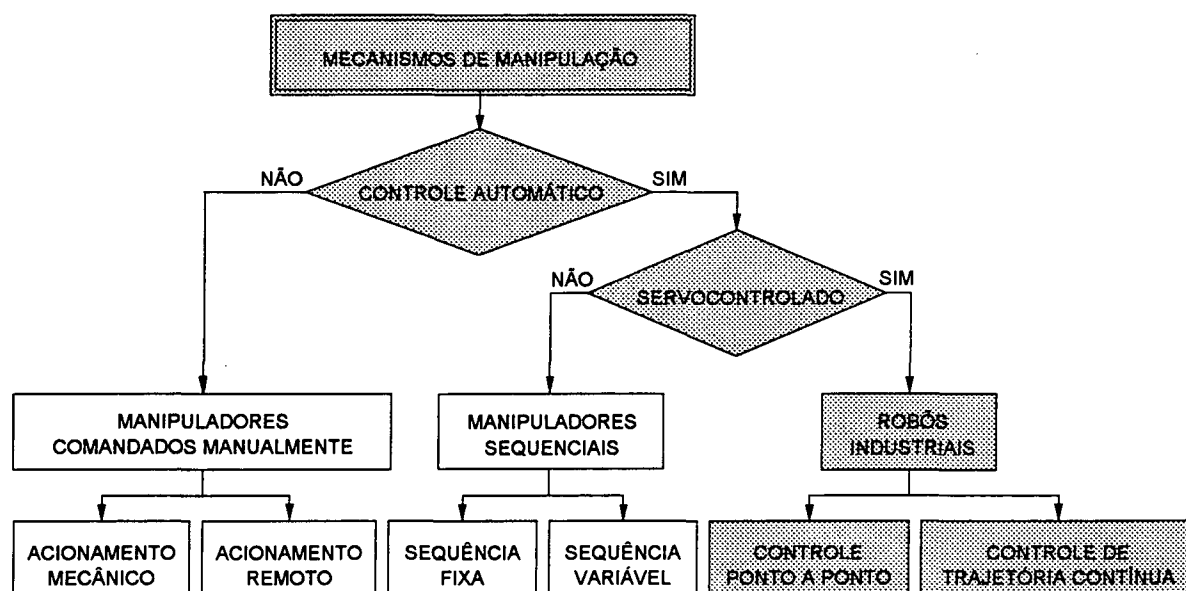


Figura 2.1 - Família dos manipuladores industriais [25].

Se comparados com o restante dos mecanismos de manipulação, nota-se que os robôs industriais possuem duas características que os diferem dos demais: controle automático e servocontrole. Tais características possibilitam que sejam reprogramados a qualquer instante, o que leva à seguinte definição:

***"Um robô industrial é um manipulador multifuncional, reprogramável, projetado para mover materiais, peças, ferramentas ou dispositivos, por meio de movimentos programáveis variados, a fim de desenvolver uma série de tarefas [25]."***

Tal definição leva em conta duas propriedades específicas exigidas de um robô: **versatilidade** e **auto-adaptação ao ambiente** [26]. A versatilidade, que depende da sua geometria e características mecânicas, diz respeito à capacidade de executar uma variedade de tarefas simples e à possibilidade de se ter uma estrutura geométrica que possa ser modificada, se necessário. A auto-adaptação está relacionada com o poder de decisão na execução de tarefas que não tenham sido completamente definidas em face à necessidade de mudanças imprevistas no ambiente. Esta propriedade depende da habilidade de "perceber" o ambiente (através



de sensores), da capacidade de analisar o ambiente e executar um plano de operações.

Sob o ponto de vista cinemático, um robô industrial pode ser definido como uma estrutura composta por vários corpos rígidos interconectados, os quais são acionados e controlados independentemente, de modo a assumir posições e velocidades previamente especificadas. A Figura 2.2 apresenta a estrutura de um robô industrial.

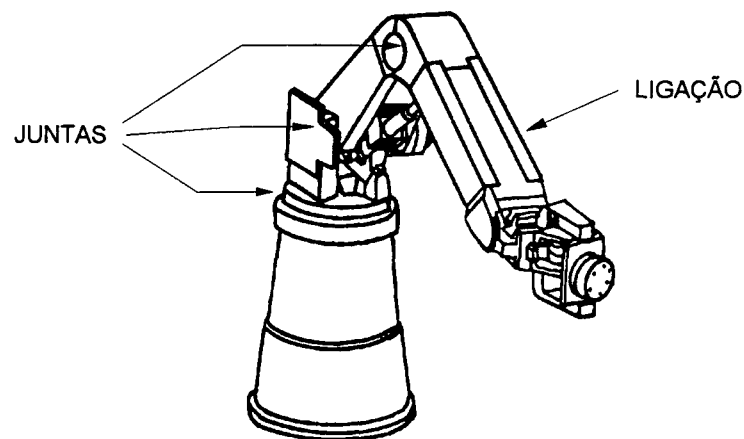


Figura 2.2 - Estrutura mecânica de um robô industrial [32].

Os corpos rígidos que formam a estrutura são denominados de **ligações** e os elementos que unem as ligações são as **juntas** [02]. As ligações e as juntas formam a **cadeia cinemática**.

A cadeia cinemática formada é classificada como **simples e aberta** [27]. Simples por não possuir ramificações (cada junta conecta apenas duas ligações) e aberta por existir pelo menos uma ligação conectada por apenas uma junta. A cadeia cinemática é montada sobre uma **base**, normalmente fixa. A extremidade livre suporta o **efetuador** que pode ser uma garra ou uma ferramenta. A função da cadeia cinemática é posicionar e orientar o efetuador.

Dois tipos de juntas são utilizados: juntas **rotativas** e juntas **prismáticas**. As juntas rotativas permitem o movimento de rotação em torno de um eixo enquanto que as prismáticas permitem o movimento de translação ao longo de um eixo conforme

mostra a figura 2.3. O fato de cada junta possuir apenas um grau de liberdade faz com que o número de juntas determine o **número de graus de liberdade** do robô.

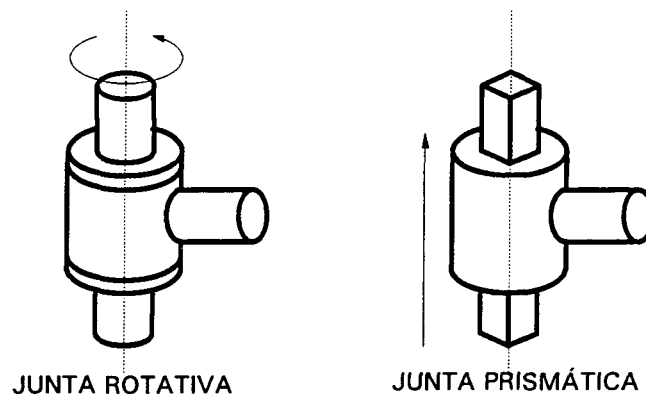


Figura 2.3 - Tipos de junta.

Para posicionar e orientar o efetuador no espaço tridimensional são necessários seis graus de liberdade (três para posicionar e três para orientar). Assim, para que um robô possa acessar pontos no espaço com qualquer orientação ele deve possuir pelo menos seis graus de liberdade. A Figura 2.2 apresenta um robô com seis graus de liberdade rotativos. Robôs com mais de seis graus de liberdade são denominados **redundantes** e são utilizados quando a presença de obstáculos no ambiente de trabalho dificulta a execução da tarefa.

A configuração da cadeia cinemática (posicionamento da cadeia no espaço) é identificada por um conjunto de variáveis denominadas de **coordenadas de juntas** ou **coordenadas generalizadas**. Cada coordenada corresponde a um grau de liberdade. As coordenadas de juntas formam um vetor  $\mathbf{q}$  na forma:

$$\mathbf{q} = [q_1 \quad q_2 \quad \dots \quad q_N]^T \quad (2.1)$$

onde  $N$  representa o número de graus de liberdade do robô. Este vetor define univocamente o posicionamento do efetuador.

O efetuador é representado no espaço externo por um vetor  $\mathbf{X}$ , denominado **coordenadas externas ou espaciais**, dado por:

$$\mathbf{X} = [x_1 \quad x_2 \quad \dots \quad x_M] \quad (2.2)$$

Nesta expressão,  $M$  representa o número de coordenadas necessárias para representar o efetuador no espaço externo. Estas coordenadas, também conhecidas por **número de graus de liberdade do efetuador**, dependem do sistema de referência adotado.

Tanto o tipo de sistema escolhido quanto a dimensão  $M$  dependem da tarefa e da estrutura do robô. No caso mais genérico  $M = 6$  quando, como já mencionado, se pode acessar um ponto com qualquer orientação. O vetor de coordenadas externas compreende duas parcelas: a de posição e a de orientação. Tem-se então:

$$\mathbf{X} = \begin{bmatrix} \mathbf{X}_I^T & \mathbf{X}_{II}^T \end{bmatrix} \quad (2.3)$$

onde  $\mathbf{X}_I$  especifica a posição e  $\mathbf{X}_{II}$  a orientação do manipulador no espaço tridimensional em relação a um sistema de coordenadas, normalmente fixo na base do robô. A posição é obtida de acordo com o sistema de coordenadas utilizado: cartesiano, esférico, ou cilíndrico. Dentre os três sistemas, o mais usual é o cartesiano por possibilitar uma melhor visualização do ambiente.

A orientação pode ser expressa por: ângulos de Euler ou parâmetros de Euler.

A definição dos parâmetros que representam a posição e a orientação pode ser encontrada nas referências [02], [03], [27], [28], [29], [30] e [31].

### 2.3 - ESTRUTURA DE UM SISTEMA ROBÓTICO

Um sistema robótico, representado na Figura 2.4, compreende a estrutura mecânica, o sistema de controle e o sistema de acionamento [32].

A **estrutura mecânica** é composta pelo manipulador propriamente dito. O **sistema de controle** coordena os movimentos de cada junta em função das

informações definidas no planejamento de trajetória. Finalmente, o **sistema de acionamento**, formado pelos órgãos mecânicos que produzem o movimento nas juntas (unidades de acionamento e unidades de força). Ele é o responsável pela produção do movimento nas juntas em função das informações transmitidas a eles pelo controle.

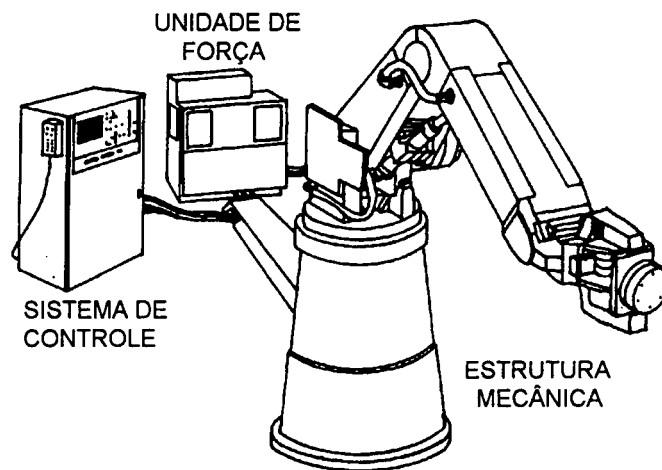


Figura 2.4 - Estrutura de um sistema robótico [32].

### 2.3.1 - ESTRUTURA MECÂNICA

Com base na maioria dos robôs industriais existentes, pode-se dividir a estrutura mecânica de um manipulador em: braço, pulso e efetuador [32].

O **braço**, **estrutura principal** ou **estrutura regional** contém os três primeiros graus de liberdade e é o responsável pelos movimentos mais amplos do efetuador. O **pulso** ou **estrutura orientacional** contém os graus de liberdade mais próximos do efetuador e é responsável pelos movimentos locais. O **efetuador** é o dispositivo que executa a tarefa. Pode ser um órgão ativo (bico de solda, pistola de pintura, esmeril ou furadeira) ou uma garra. O tipo e a forma do efetuador dependem da aplicação específica do robô.

### a) Braço

As características geométricas do braço permitem agrupar os robôs industriais em quatro estruturas: cartesiana, cilíndrica, esférica e articulada. As três primeiras estruturas foram denominadas fazendo-se uma analogia com os respectivos sistemas de coordenadas. A Figura 2.5 apresenta as quatro estruturas citadas.

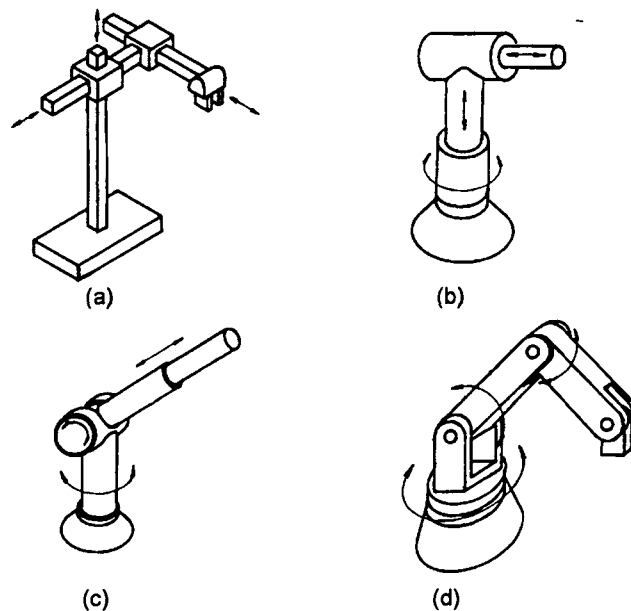


Figura 2.5 - Estruturas mecânicas do braço de robôs:  
a) cartesiana, b) cilíndrica, c) esférica e d) articulada.

Cada estrutura possui características próprias, que a qualifica para determinadas tarefas [04]. Robôs **cartesianos**, por exemplo, devido à rigidez de sua estrutura, são indicados para montagens de precisão e operações de inspeção. Os robôs **cilíndricos** são utilizados na alimentação de máquinas e no transporte de peças. As estruturas **esférica** e **articulada** possuem um alcance maior que as primeiras, podendo acessar pontos inferiores à base do robô. Devido à sua grande mobilidade, a estrutura articulada é a mais utilizada em operações de manipulação, solda e pintura. A presença das juntas rotativas, contudo, dificulta o controle do robô devido à presença de acoplamentos entre as juntas e à ocorrência de degenerações em certas regiões do espaço de trabalho.

## b) Pulso

A função do pulso é orientar o efetuador no espaço. Por isso deve possuir três juntas rotativas que permitam rotações ao longo dos três eixos cartesianos. Embora a maioria dos pulsos possuam três graus de liberdade, existem aplicações tais como soldagem, que necessitam de apenas dois eixos de rotação.

O pulso deve ser o mais compacto e leve possível para permitir que a carga transportada e o espaço de trabalho sejam os maiores possíveis.

Dois tipos de juntas rotativas são utilizados no pulso: **roll e bend** (fig.2.6).

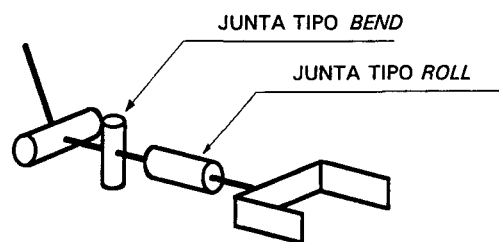


Figura 2.6 - Juntas rotativas *roll* e *bend* [34].

Juntas *roll* permitem faixas de orientação maiores por não sofrerem interferências mecânicas e por não produzirem mudanças na posição do efetuador, características inerentes das juntas *bend*. Por isso costuma-se escolher a última junta como sendo *roll* [04]. Além disso esta escolha facilita a operação em certos casos de manipulação de peças e montagem. A combinação destes dois tipos de junta rotativa possibilita a orientação do efetuador.

Rivlin [04] estudou as várias combinações possíveis de juntas *bend* e *roll* em pulsos com um, dois e três graus de liberdade. Pulsos com um e dois graus de liberdade são utilizados em sistemas robóticos específicos tais como aqueles utilizados em operações de manipulação (*pick and place*). Este é o caso dos robôs **SCARA** (1 grau de liberdade para orientação) e dos robôs de pintura (dois graus de liberdade para orientação) onde a última junta *roll* não é necessária devido à simetria axial do jato. Para pulsos com três graus de liberdade, as combinações com aplicação prática são as que possuem a última junta *roll*.

### c) Efetuador

O efetuador é o dispositivo, fixado no pulso do robô, que permite ao braço, de uso geral, executar tarefas específicas [33]. Do mesmo modo que um dispositivo de fixação de uma máquina operatriz, o efetuador pode ser padronizado ou projetado para um fim específico. De modo geral, os efetuadores podem ser classificados como **garras** ou **ferramentas**, conforme a sua finalidade.

As **garras** são utilizadas simplesmente para segurar e manusear objetos como no caso de carga e descarga de máquinas, colocação de peças em transportadores (ou paletes) ou ainda em processos de montagem. **Ferramentas** são efetuadores projetados para executar um trabalho sobre a peça. Tais trabalhos envolvem solda, pintura, furação, corte, além de outros processos.

Uma descrição dos vários tipos de efetuadores e suas aplicações foi feita por Andeen [35] e Ránky [36]. Este último descreve ainda os dispositivos para troca automática de efetuador (*automated robot hand changer* - ARHC). Tais dispositivos permitem um intercâmbio de ferramentas o que permite um menor número de robôs necessários numa operação de montagem.

As referências [37], [38] e [39] apresentam detalhes sobre projeto, construção e aplicação de garras, em função das tarefas a serem executadas.

### 2.3.2 SISTEMA DE CONTROLE

O sistema de controle fornece sinais às juntas para produzir o movimento desejado no efetuador. Estes sinais são obtidos por meio de sensores que podem fornecer informações das variáveis de posição, velocidade, aceleração e força.

O tipo de controle utilizado permite a classificação em robôs de seqüência limitada, robôs de repetição e robôs inteligentes. Os robôs de **seqüência limitada** utilizam chaves limitadoras de fim de curso. Isto faz com que o efetuador acesse um número reduzido de pontos no espaço, correspondentes aos limites das juntas. Nos robôs de **repetição**, uma série de pontos são previamente gravados na

memória do controle, através do posicionamento do efetuador. Estes pontos podem ser obtidos isoladamente ou continuamente, o que leva a classificar os robôs de repetição em de **controle ponto-a-ponto** e de **controle contínuo**. Os robôs **inteligentes**, além de gerarem a trajetória contínua, podem corrigi-la em resposta a sinais transmitidos por sensores.

### 2.3.3 UNIDADES DE ACIONAMENTO

As unidades de acionamento são as responsáveis pela movimentação das juntas. Elas podem ser de origem hidráulica, pneumática, elétrica ou mista, com ou sem transmissão mecânica. Desse modo, quanto à forma de acionamento, os robôs podem ser classificados em pneumáticos, hidráulicos e elétricos. A aplicação dos robôs **pneumáticos** está restrita a pequenos robôs de dois a quatro graus de liberdade, em tarefas de transporte de peças. O movimento é obtido através de atuadores lineares e rotativos. Os robôs **hidráulicos** possuem capacidade de carga elevada sendo, contudo, desaconselháveis para ambientes que devem ser limpos devido à possibilidade da ocorrência de vazamentos. O movimento pode ser feito por meio de atuadores rotativos ou por pistões hidráulicos. Os robôs **elétricos** não possuem a capacidade de carga e a agilidade dos hidráulicos, de mesmo porte, por terem a massa dos motores adicionada à das ligações. Pode-se ter tanto juntas rotativas quanto prismáticas (movidas através de polias ou cremalheiras). Estes robôs são os que apresentam melhores características de precisão e repetibilidade, o que os torna preferíveis em tarefas de montagem.

### 2.4 - PARÂMETROS BÁSICOS

Quando se trata da utilização ou mesmo do projeto de um robô, é muito importante observar os parâmetros básicos que caracterizam o seu comportamento e desempenho. Andeen [35] especifica quatro conjuntos de parâmetros básicos que



caracterizam um robô: parâmetros físicos ou funcionais, parâmetros operacionais, parâmetros de utilidade e parâmetros ambientais.

Os **parâmetros físicos ou funcionais** são aqueles relacionados com as características potenciais do robô e com o seu desempenho na execução de uma tarefa. São eles: espaço de trabalho, mobilidade, carga útil, agilidade, rigidez, precisão, resolução e repetibilidade. Estes parâmetros não podem ser representados por um simples número pois assumem valores diferentes em diferentes pontos do espaço.

Os **parâmetros operacionais** relacionam-se com o modo de controle e o tipo de programação empregados.

Os **parâmetros de utilidade** relacionam-se com a adequação do robô ao ambiente de trabalho num contexto de célula de fabricação. São eles: vida útil esperada, tempo médio entre falhas, tempo de reparo, espaço ocupado, peso, segurança, treinabilidade, custo e potência.

Os **parâmetros ambientais** relacionam-se com o tipo de ambiente onde o robô é instalado com respeito a temperatura, tipo de atmosfera e tipo de fonte de energia disponíveis.

Entre os parâmetros citados, um que merece destaque é o espaço de trabalho. O estudo cinemático do robô, particularmente a cinemática inversa, depende, em grande parte, do seu conhecimento. Além disso, existem parâmetros que estão intimamente ligados ao espaço de trabalho, tais como mobilidade, carga útil, agilidade, precisão e rigidez. Desse modo, em virtude da sua importância, são apresentados, no próximo item, alguns comentários sobre a influência da estrutura cinemática sobre o espaço de trabalho correspondente.

#### **2.4.1 - ESPAÇO DE TRABALHO**

O espaço de trabalho é o espaço, delimitado por uma superfície tridimensional, dentro do qual o robô pode operar [32]. A superfície que delimita o

espaço de trabalho é formada de elementos de superfície plana, esférica, toroidal e cilíndrica, dependendo do tipo das juntas e dos comprimentos das ligações [40].

Geralmente o diagrama do espaço de trabalho fornecido pelo fabricante, como o mostrado na figura 2.7, denominado de **espaço de trabalho nominal** [34], é obtido com base no pulso do robô. Para o usuário este diagrama não é de muita utilidade pois ele não considera o efetuador, cuja geometria tem grande influência sobre a forma final do espaço de trabalho. Assim, métodos de obtenção do espaço de trabalho tem sido desenvolvidos [41], [42], [43], alguns dos quais utilizando técnicas de computação gráfica cujos recursos tornam a representação bastante clara e detalhada [40].

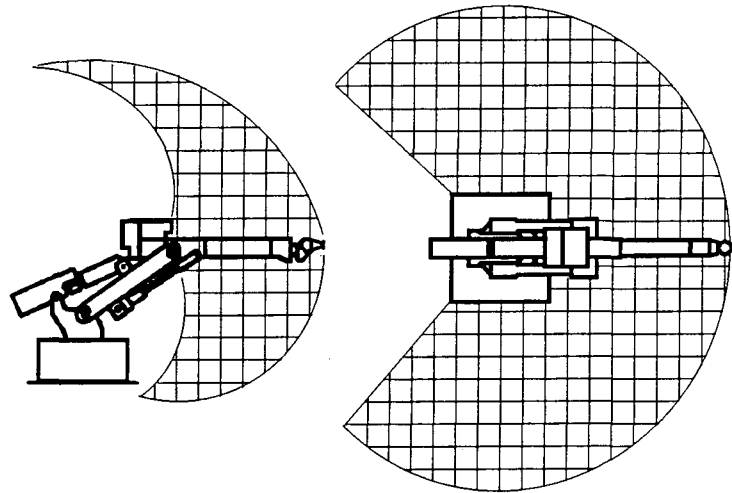


Figura 2.7. - Representação do espaço de trabalho nominal de um robô [32].

O conhecimento do espaço de trabalho, além de auxiliar na avaliação das diferentes características de desempenho do robô, constitui-se em uma importante bagagem para o entendimento da influência da geometria do robô no seu funcionamento [41]. Ele auxilia na escolha do robô mais adequado para a execução de determinada tarefa, bem como para o planejamento de trajetórias, quando feito em coordenadas espaciais pois, é através dele que se pode verificar se os pontos escolhidos estão ou não ao alcance do efetuador. Diferentes configurações do robô para uma mesma posição do efetuador no espaço e posições singulares [43] podem ser verificadas. Além disso, com base no espaço de trabalho, pode-se delimitar a área de operação, garantindo a segurança no ambiente de trabalho [32]. Parâmetros de

desempenho como carga útil, agilidade e precisão podem ser melhorados limitando-se a operação a regiões específicas do espaço de trabalho.

A seqüência das juntas e seus limites definem a forma do espaço de trabalho enquanto que os comprimentos das ligações definem o seu tamanho [32]. A alteração destes parâmetros podem modificar completamente o espaço de trabalho. Certas relações entre os comprimentos das ligações levam ao aparecimento de **vazios** e **zonas-mortas** [42]. Os vazios são regiões internas ao espaço de trabalho onde o efetuador não tem acesso. As zonas-mortas são regiões externas ao espaço de trabalho porém circundadas por ele. Zonas-mortas são encontradas em robôs que possuem ombros, como é o caso do robô PUMA [02]. A Figura 2.8 apresenta o espaço de trabalho de um robô onde ocorre a presença de vazios e zonas-mortas. Neste exemplo, o espaço de trabalho produzido tem a forma aproximada de um sólido toroidal vazado. O espaço interno do toróide é um vazio e a região ao redor do eixo da primeira junta é uma zona-morta.

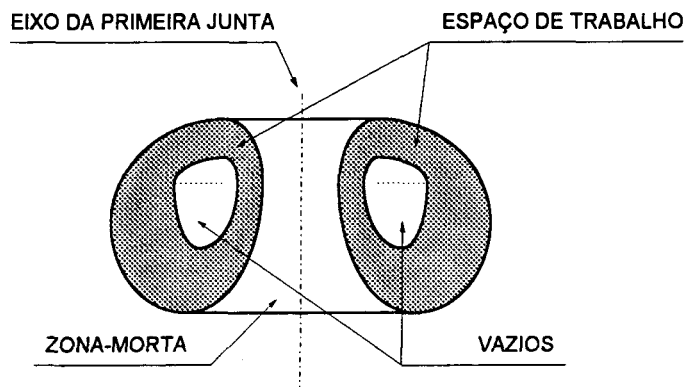


Figura 2.8 - Espaço de trabalho contendo vazios e zonas-mortas [42].

Estudos sobre o volume, forma e sub-regiões do espaço de trabalho tem sido feitos de modo a se conhecer melhor o comportamento do robô face às tarefas requeridas [44][45].

A maioria das características de desempenho de um robô depende da posição e da orientação do efetuador dentro do espaço de trabalho. Por isso é útil a caracterização da totalidade das posições e orientações que um robô pode assumir. Um conceito conveniente para este fim é o de diferentes regiões dentro do espaço de trabalho associadas às possíveis posições e orientações do efetuador.

Existem duas regiões distintas dentro do espaço de trabalho [41], conforme mostra a Figura 2.9: o espaço de trabalho **primário** e o **secundário**. Neste exemplo, tem-se um robô planar com três graus de liberdade com as ligações representadas pelos segmentos OA, AB e BC. Supõe-se que as três juntas podem girar livremente uma volta completa. Para este exemplo o espaço delimitado pelas circunferências **C1** e **C4** o espaço de trabalho secundário. Note-se que sobre as duas circunferências, só se tem uma orientação. Entre as circunferências **C2** e **C3**, tem-se o espaço de trabalho primário.

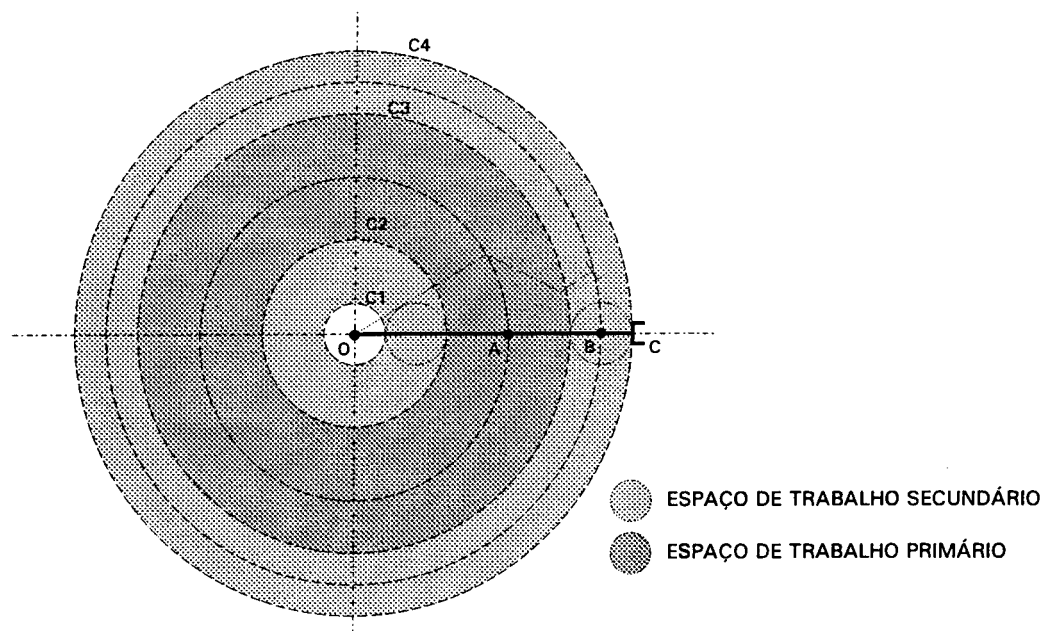


Figura 2.9 - Espaços de trabalho **primário** e **secundário**.

A região mais abrangente, denominada de espaço de trabalho secundário, engloba os pontos que podem ser acessados com pelo menos uma orientação. Dentro de espaço de trabalho secundário situa-se o espaço de trabalho primário ou destro, cujos pontos podem ser acessados com qualquer orientação. Esta divisão é importante para o estudo da cinemática inversa, que se comporta de maneira diferente para cada ponto do espaço de trabalho.

O espaço de trabalho é sensivelmente diminuído quando se leva em conta os limites de deslocamento das juntas e as restrições mecânicas do robô. Isto pode ser constatado no robô articulado da Figura 2.7, cujo espaço de trabalho, sem

levar em conta os limites das juntas, seria uma esfera de raio igual à soma dos comprimentos das ligações.

Os comprimentos das ligações também interferem na forma do espaço de trabalho [46]. A Figura 2.10 apresenta três manipuladores com diferentes relações entre os comprimentos das ligações e com os mesmos limites nas juntas. Pode-se notar que o espaço de trabalho é diferente para cada um deles.

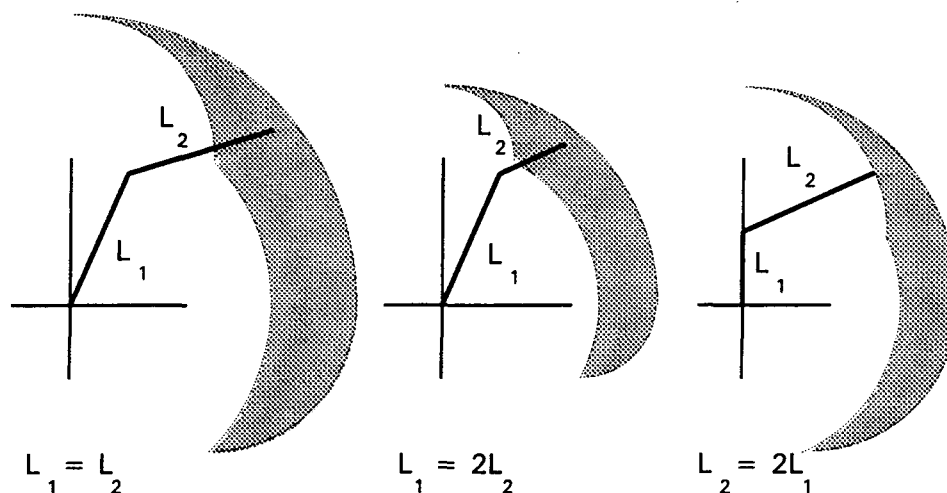


Figura 2.10 - Variação do espaço de trabalho com os comprimentos das ligações [46].

O número de graus de liberdade de um robô pode variar ao longo do seu espaço de trabalho. De acordo com a expressão (2.1), a cadeia cinemática do robô possui um número de graus de liberdade ( $N$ ), correspondente ao seu número de juntas e denominado de **número de graus de liberdade do robô**. O efetuador, por sua vez, conforme a expressão (2.2), possui um número de graus de liberdade ( $M$ ), denominado de **número graus de liberdade do efetuador**, que pode ser menor ou igual ao do robô. Se o número de graus de liberdade do robô for igual ao do efetuador ( $N = M$ ) para qualquer posição do efetuador, o robô é denominado **não-redundante** [34]. Se o número de graus de liberdade do efetuador for menor que o do robô ( $M < N$ ), dois casos podem existir. Se a condição é satisfeita em todo o espaço de trabalho o robô é dito **redundante**. Se a condição é satisfeita em apenas algumas configurações, o robô é definido como **localmente redundante** ou **degenerado**. Tais configurações, chamadas **singulares**, são problemáticas para o cálculo da cinemática inversa, conforme será apresentado no Capítulo 3.

## 2.5 PROGRAMAÇÃO

Entende-se por programação de um robô o conjunto de informações sobre trajetória, controle e sinais dos sensores que resultam em sinais ao manipulador para executar determinada tarefa [33]. Em outras palavras, consiste em ensinar ao robô o seu ciclo de trabalho.

Existem dois tipos de programação: por aprendizagem e textual. A programação **por aprendizagem**, iniciada na década de 60, consiste em mover o robô na seqüência de movimentos desejada enquanto as posições são gravadas na memória do controlador. Os parâmetros referentes às velocidades são definidos posteriormente, independentemente das posições. A programação **textual**, desenvolvida na década de 70, emprega uma linguagem de programação para estabelecer a seqüência de movimentos. Isto é feito através de um terminal de computador. Um dispositivo que permite o posicionamento manual do efetuador (*teach-in box*) é utilizado para definir os pontos fazendo com que a programação ainda dependa do robô para ser concluída.

Estudos tem sido feitos para tornar a programação totalmente independente (*off-line*) evitando dessa maneira a parada do robô na produção. A programação seria feita no computador e, somente após ter sido testada e validada, seria transferida para o robô. Para que isto seja possível, é necessário o cálculo das cinemáticas direta e inversa. O problema maior na programação *off-line* está na definição das posições a serem usadas no ciclo, razão pela qual ainda se utiliza o *teach-in box* nas linguagens textuais.

## 2.6 PLANEJAMENTO DA TRAJETÓRIA

A especificação de uma tarefa pode ser dividida em dois estágios: seleção dos pontos e seleção da trajetória entre estes pontos [27].

A **seleção das trajetórias entre os pontos** é feita por intermédio de polinômios interpoladores cujo grau deve ser escolhido de modo a garantir

continuidade de posição, velocidade e aceleração nos extremos dos intervalos [02][27][67]. Assim, a finalidade do planejamento de trajetória é controlar o percurso entre os pontos escolhidos de modo a produzir o deslocamento desejado de maneira suave e contínua.

A interpolação pode ser feita nas coordenadas externas ou nas coordenadas de juntas, dependendo de se desejar um percurso sobre um caminho bem definido ou não. Caso se faça a interpolação em coordenadas externas, é necessário o cálculo da cinemática inversa tanto para os pontos escolhidos como para os calculados na interpolação. Isto causa um aumento no volume de cálculo, além dos problemas decorrentes da presença de pontos singulares. Devido a estes inconvenientes, a maioria dos robôs suportam tanto o movimento cartesiano quanto o movimento em juntas. A preferência é dada ao movimento interpolado nas juntas sendo o movimento cartesiano utilizado quando estritamente necessário [03].

O planejamento de trajetórias pode ser feito *on-line* ou *off-line* [36]. O cálculo *on-line* permite mudanças durante o trajeto, o que é conveniente quando se está definindo uma trajetória na presença de obstáculos. Por outro lado este cálculo é mais demorado resultando em pontos calculados mais afastados uns dos outros. O cálculo *off-line* é feito quando se tem operações repetitivas, onde os dados são calculados apenas uma vez. Isto resulta em pontos mais próximos uns dos outros tornando a trajetória mais precisa.

## CAPÍTULO 3

### CINEMÁTICA DE ROBÔS

#### 3.1 - INTRODUÇÃO

A preocupação maior no controle de robôs está na obtenção de um algoritmo estável para coordenar o movimento das juntas de modo a permitir o posicionamento do efetuador com a precisão desejada. A dificuldade aumenta quando se define a tarefa em coordenadas cartesianas pois, o sistema de controle exige que a referência de entrada seja especificada em coordenadas de juntas [49].

As características mecânicas do robô influenciam diretamente na forma e no volume do seu espaço de trabalho. A relação entre os tipos e limites das juntas e o espaço de trabalho deve ser analisada já na fase de projeto [04] de modo a minimizar regiões singulares que restringem o movimento do robô.

Com respeito aos aspectos levantados pode-se concluir que tanto na fase de projeto quanto na de utilização do robô, torna-se necessário o conhecimento da relação entre as coordenadas das juntas e as do efetuador no espaço externo. Tal relação é estudada através da cinemática. A cinemática de robôs trata do estudo da geometria do movimento de um robô com relação a um sistema de referência fixo, sem levar em conta as forças ou os torques que o originam [02]. Ela trata, portanto, da descrição analítica do deslocamento espacial do robô como função do tempo.

Tem-se dois problemas a estudar [50]: a cinemática de posição e a diferencial. A **cinemática de posição** trata da relação entre as coordenadas de juntas



e posição do efetuador, enquanto que a **cinemática diferencial** trata da relação entre os deslocamentos infinitesimais das juntas e os do efetuador.

A relação entre coordenadas das juntas e do efetuador pode ser feita nos dois sentidos. Tem-se então as cinemáticas direta e inversa. A **cinemática direta** fornece a posição e a orientação do efetuador quando são fornecidas as coordenadas das juntas. A solução é sempre possível e é única. A **cinemática inversa** trata da obtenção das coordenadas de juntas correspondentes a uma dada posição e orientação do efetuador no espaço externo. Por se tratar da solução de um sistema de equações não-lineares, ela é mais complexa, podendo ter uma, várias ou nenhuma solução. Dependendo da configuração do robô, a solução pode ser obtida em forma de expressões analíticas explícitas (forma fechada) ou então através de métodos numéricos.

A fim de possibilitar o estudo cinemático de um robô é preciso que este possa ser representado matematicamente. Em outras palavras, é preciso definir um modelo matemático do robô de modo a reproduzir o seu comportamento da maneira mais realista possível. O item seguinte apresenta algumas técnicas utilizadas na modelagem cinemática de robôs.

### 3.2 - MODELAGEM CINEMÁTICA DE UM ROBÔ

A estrutura cinemática de um robô consiste de uma série de corpos rígidos (**ligações**) interconectadas por **juntas** e é denominada de **cadeia cinemática** [28].

A cadeia cinemática pode ser representada por um conjunto de sistemas de coordenadas cada qual localizado num ponto sobre cada ligação. A relação entre os sistemas de coordenadas é feita através de matrizes de transformação entre sistemas vizinhos.

A escolha do ponto de referência depende do que se deseja analisar. Para a análise dinâmica, pode-se escolher o centro de massa das ligações, aproveitando as direções principais de inércia [27]. Tal escolha, no entanto, não é

adequada para a análise cinemática pela grande quantidade de parâmetros não-nulos que aparecem, o que aumenta consideravelmente o volume de cálculo.

Para a análise cinemática, a localização ideal para o sistema de coordenadas situa-se sobre os eixos das juntas, com o eixo z na direção da junta. Esta posição permite a representação local da ligação além de facilitar a descrição do movimento que agora passa a ser representado por um deslocamento ou uma rotação sobre o eixo z do sistema local.

Definidos os sistemas de coordenadas em cada ligação, pode-se estudar a cadeia cinemática através da transformação de coordenadas de um sistema para outro. Assim a estrutura mecânica perde o seu significado, e o comportamento cinemático fica representado pelo conjunto de sistemas de coordenadas.

Tomando dois sistemas quaisquer  $i-1$  e  $i$  da cadeia cinemática, a posição do sistema  $i$  em relação ao  $i-1$  é dada pelo vetor  $r$ , que representa a origem do sistema  $i$ , como mostra a Figura 3.1.

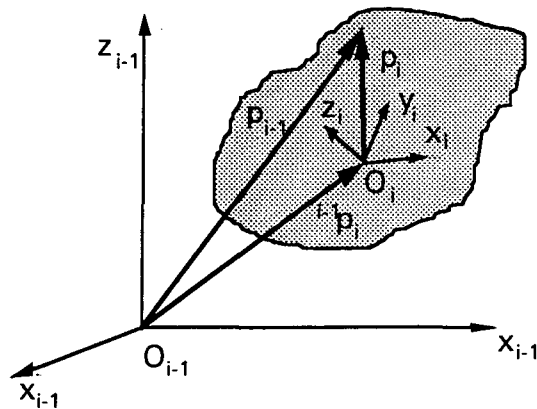


Figura 3.1 - Transformação entre sistemas vizinhos [39].

A orientação do sistema  $i$ , relativa ao  $i-1$ , é representada pela matriz de rotação  ${}^{i-1}R_i$ . Assim, a posição de um ponto  $P$ , representado no sistema  $i$  pelo vetor  $p_i$ , pode ser dada no sistema  $i-1$  por

$$p_{i-1} = {}^{i-1}p_i + {}^{i-1}R_i \cdot p_i \quad (3.1)$$

Desta forma, a ligação  $i$  pode ser completamente descrita em termos da equação (3.1).

A matriz de rotação  $R$  pode ser obtida de várias maneiras dependendo do tipo de coordenadas que se está utilizando para representar a orientação. As mais usuais são os **ângulos de Euler**, os **parâmetros de Euler**, os **parâmetros de Rodrigues** e os **co-senos diretores**. As relações entre tais coordenadas e os elementos da matriz de rotação podem ser encontradas nas referências [02], [03], [27] e [31].

Através da equação (3.1) a posição e a orientação do último sistema de coordenadas, fixado no efetuador, pode ser obtida em relação ao sistema inercial mediante um processo recursivo.

Com base nestes conceitos, várias técnicas de modelagem foram desenvolvidas. Entre elas destacam-se a que utiliza transformações homogêneas baseadas na notação de Denavit-Hartenberg [28], a vetorial [49],[51], e a da fórmula de Rodriguez [27].

A formulação utilizando **matrizes homogêneas** juntamente com a notação de **Denavit-Hartenberg** é a mais utilizada por permitir estudos tanto em cinemática quanto em dinâmica. Os sistemas de coordenadas das ligações são fixados ao longo das juntas de modo a facilitar a definição das rotações e translações. As origens dos sistemas das ligações são localizadas na normal comum entre as juntas que a limitam, conforme mostra a Figura 3.2.

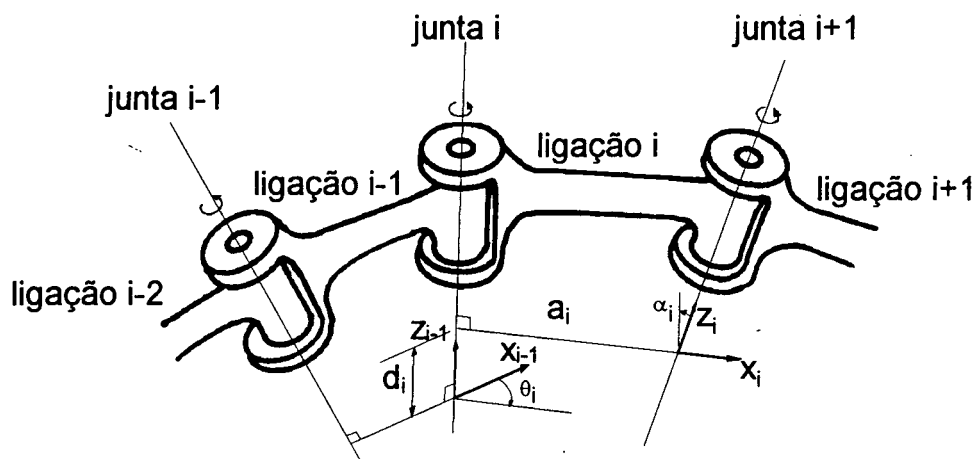


Figura 3.2 - Modelagem utilizando matrizes homogêneas e a notação de Denavit-Hartenberg.

A transformação do sistema  $i-1$  para o sistema  $i$  é composta por quatro transformações elementares [28] que, agrupadas numa mesma matriz, resultam em:

$${}^{i-1}\mathbf{A}_i = \begin{bmatrix} \cos \theta_i & -\sin \theta_i \cos \alpha_i & \sin \theta_i \sin \alpha_i & a_i \cdot \cos \theta_i \\ \sin \theta_i & \cos \theta_i \cos \alpha_i & -\cos \theta_i \sin \alpha_i & a_i \cdot \sin \theta_i \\ 0 & \sin \alpha_i & \cos \alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (3.2)$$

Na expressão (3.2),  $i$  representa o índice da junta  $i$ ,  $\theta_i$  o ângulo da junta  $i$ ,  $d_i$  o deslocamento da junta  $i$ ,  $a_i$  o comprimento da ligação  $i$  e  $\alpha_i$  a torção da ligação  $i$ . Os parâmetros  $\theta_i$ ,  $d_i$ ,  $a_i$  e  $\alpha_i$  são denominados de parâmetros de Denavit-Hartenberg da ligação  $i$ . As referências [27] e [28] descrevem com detalhes a obtenção dos parâmetros a partir da definição dos sistemas de coordenadas de cada ligação. Seguindo-se esta abordagem, a relação entre dois sistemas vizinhos fica definida por quatro parâmetros: dois descrevendo a ligação e dois a junta [03]. Três desses parâmetros são constantes e apenas um é variável ( $\theta_i$  ou  $d_i$ ), conforme a junta seja rotativa ou prismática.

Esta modelagem mostra-se bastante adequada para a análise de mecanismos com vários graus de liberdade por ser mais compacta. Ela também é utilizada em algumas técnicas de cinemática inversa e na análise dinâmica de manipuladores [32].

Na abordagem segundo a **fórmula de Rodrigues**, fixa-se os eixos dos sistemas de coordenadas locais, paralelamente aos eixos principais de inércia da ligação e sua origem no centro de massa, conforme mostra a Figura 3.3.

Para a modelagem dinâmica, tais sistemas facilitam a representação do tensor de inércia das ligações. Por outro lado, sob o ponto de vista cinemático, as suas localizações não são adequadas pois, a não coincidência com os eixos das juntas ou com a normal comum entre as juntas torna a modelagem mais complexa do que aquela utilizando os parâmetros de Denavit-Hartenberg [27]. Esta modelagem não utiliza matrizes homogêneas. Assim sendo, a posição e a orientação são representadas separadamente, conforme a equação (3.1).

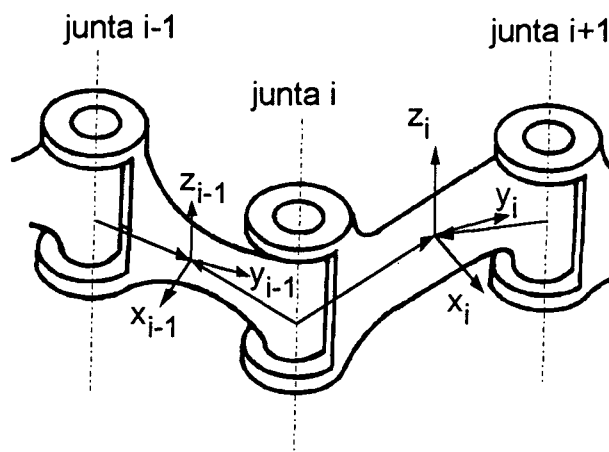


Figura 3.3 - Modelagem segundo a fórmula de Rodrigues.

Na modelagem pela **representação vetorial** são definidos dois conjuntos de vetores: um representando as ligações e outro representando a orientação das juntas. O posicionamento do efetuador é obtido, recursivamente, mediante a rotação e a adição dos vetores das ligações.

Uma abordagem vetorial é proposta por Gupta, baseada na **Posição-Zero** [52]. A cadeia cinemática é dada por dois conjuntos de vetores: **vetores de orientação**, unitários,  $u$ , representando a orientação das juntas, e **vetores de corpo**  $b$ , representando o comprimento das ligações. A representação destes vetores se encontra na figura 3.4.

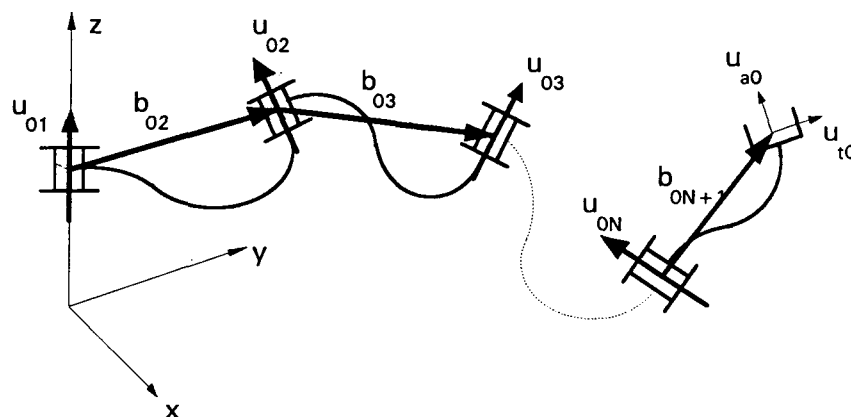


Figura 3.4 - Vetores da Posição-Zero.

Os vetores de corpo são definidos livremente pela distância entre dois pontos quaisquer, cada um deles pertencendo a uma junta. O efetuador é

representado por dois vetores unitários  $u_a$  e  $u_t$ . Este conjunto de vetores define completamente a cadeia cinemática. Tal representação consiste no "congelamento" da cadeia cinemática numa dada posição, denominada **Posição-Zero**, onde todas as variáveis de juntas são consideradas nulas [52]. Uma nova posição é obtida a partir desta, utilizando-se de matrizes de rotação sobre um vetor (em caso de juntas rotativas) ou de um vetor deslocamento ao longo da junta (em caso de juntas prismáticas). A posição e a orientação do efetuador são obtidas recursivamente pela soma dos vetores de corpo devidamente posicionados.

Com a abordagem vetorial se consegue um aumento considerável na velocidade do cálculo pois ela não executa multiplicações por 1 e por zero, comuns nos produtos das matrizes homogêneas.

Tanto os parâmetros de Denavit-Hartenberg quanto os vetores da Posição-Zero requerem uma manipulação trabalhosa de dados referentes às juntas e às ligações. Tais parâmetros são obtidos pela inspeção direta da cadeia cinemática que, dependendo da sua complexidade, podem ser de difícil obtenção. É o caso de robôs com seis graus de liberdade que possuem cadeias espaciais de difícil representação. A fim de facilitar a tarefa de determinar os parâmetros de Denavit-Hartenberg e da Posição-Zero, está apresentado no Apêndice 1 uma sistemática que permite calcular automaticamente os referidos parâmetros a partir da orientação de cada junta e um ponto sobre a mesma.

### 3.3 - CINEMÁTICA DIRETA

A cinemática direta determina qual a posição e orientação do efetuador, sendo conhecidas as posições das juntas do robô. Matematicamente, a cinemática direta é dada por:

$$\mathbf{X} = f(\mathbf{q}) \quad (3.3)$$

onde  $\mathbf{X}$  e  $\mathbf{q}$  são os vetores definidos nas expressões (2.1) e (2.2) e  $\mathbf{f}$  é uma função vetorial não-linear [27]. A equação (3.3) é conhecida como **equação cinemática do robô** [28].

As técnicas mais comumente utilizadas para a obtenção da equação cinemática, baseiam-se na geometria do robô ou em equações algébricas. Neste último caso, utiliza-se a notação de Denavit-Hartenberg com as matrizes homogêneas ou a notação vetorial. O procedimento geométrico [32] pode ser adequado para robôs planares com poucos graus de liberdade sendo, no entanto, específico para cada robô.

A estrutura da equação cinemática depende da abordagem que se está utilizando. No caso das matrizes homogêneas, obtém-se expressões explícitas da posição e orientação do efetuador em relação às coordenadas das juntas, enquanto que pela abordagem vetorial as relações são recursivas.

Utilizando-se a **notação de Denavit-Hartenberg**, considerando que os parâmetros já são conhecidos, o cálculo da cinemática direta é feito pelas transformações sucessivas entre os sistemas de coordenadas desde a base do robô até o efetuador. Isto é obtido pelas multiplicações sucessivas das matrizes  ${}^{i-1}\mathbf{A}_i$ , equação (3.2), fazendo  $i$  variar de 1 até  $N$ . Deste modo obtém-se a seguinte expressão:

$$\mathbf{T}_N = {}^0\mathbf{A}_1(q_1) \cdot {}^1\mathbf{A}_2(q_2) \cdot \dots \cdot {}^{i-1}\mathbf{A}_i(q_i) \cdot \dots \cdot {}^{N-1}\mathbf{A}_N(q_N) \quad (3.4)$$

onde a matriz (4x4)  $\mathbf{T}_N$  representa a posição e a orientação do efetuador no espaço. As três primeiras colunas representam a orientação dada pelos co-senos diretores dos eixos do sistema fixo no efetuador em relação ao sistema inercial. A quarta coluna representa a posição da origem do sistema de coordenadas fixo no efetuador em relação ao sistema inercial. Como em cada matriz  ${}^{i-1}\mathbf{A}_i$  apenas o parâmetro  $q_i$  é variável, a equação cinemática é função de apenas  $N$  variáveis.

Pela **abordagem vetorial**, segundo o método da Posição-Zero (Fig. 3.4), a posição e a orientação do efetuador são obtidas através da rotação dos vetores de corpo em torno dos vetores unitários, a partir de uma posição inicial da cadeia

cinemática, denominada de Posição-Zero. Matematicamente este procedimento baseia-se nas seguintes relações recursivas:

$$\mathbf{u}_k = \mathbf{R}_{k-1} \cdot \mathbf{u}_{0,k} \quad (3.5)$$

$$\mathbf{b}_{k+1} = \mathbf{R}_k \cdot \mathbf{b}_{0,k+1} \quad (3.6)$$

$$\mathbf{u}_a = \mathbf{R}_N \cdot \mathbf{u}_{a0} \quad (3.7)$$

$$\mathbf{u}_t = \mathbf{R}_N \cdot \mathbf{u}_{t0} \quad (3.8)$$

onde  $k$  varia de 1 a  $N$  e  $\mathbf{R}_k$  é uma matriz de rotação  $3 \times 3$  dada pela relação recursiva:

$$\mathbf{R}_0 = \mathbf{I} \quad (3.9)$$

$$\mathbf{R}_k = \mathbf{R}_{k-1} \quad (\text{junta prismática}) \quad (3.10)$$

$$\mathbf{R}_k = \mathbf{R}_{k-1} \cdot \mathbf{R}(\theta_k, \mathbf{u}_{0,k}) \quad (\text{junta rotativa}) \quad (3.11)$$

A matriz  $\mathbf{R}(\theta, \mathbf{u})$  representa uma rotação  $\theta$  sobre um vetor unitário  $\mathbf{u}$  [29].

O seu cálculo é apresentado a seguir:

$$\mathbf{R}(\theta, \mathbf{u}) = \begin{bmatrix} (u_x^2 - 1) \cdot v\theta + 1 & u_x \cdot u_y \cdot v\theta - u_z \cdot s\theta & u_x \cdot u_z \cdot v\theta + u_y \cdot s\theta \\ u_x \cdot u_y \cdot v\theta + u_z \cdot s\theta & (u_y^2 - 1) \cdot v\theta + 1 & u_y \cdot u_z \cdot v\theta - u_x \cdot s\theta \\ u_x \cdot u_z \cdot v\theta - u_y \cdot s\theta & u_y \cdot u_z \cdot v\theta + u_x \cdot s\theta & (u_z^2 - 1) \cdot v\theta + 1 \end{bmatrix} \quad (3.12)$$

Nesta expressão,  $u_x$ ,  $u_y$  e  $u_z$ , representam as coordenadas do vetor  $\mathbf{u}$ ;  $c\theta$  e  $s\theta$  representam o co-seno e o seno do ângulo  $\theta$ ; e  $v\theta = (1 - \cos\theta)$ .

Os vetores que fornecem a posição das juntas são calculados por:

$$\mathbf{P}_1 = \mathbf{0} \quad (3.13)$$

$$\mathbf{P}_{k+1} = \mathbf{P}_k + q_k \mathbf{u}_k \quad (\text{junta prismática}) \quad (3.14)$$

$$\mathbf{P}_{k+1} = \mathbf{P}_k + \mathbf{b}_{k+1} \quad (\text{junta rotativa}) \quad (3.15)$$

Na equação 3.14,  $q_k$  representa o deslocamento de translação da junta  $k$ .



A posição e a orientação do efetuador são dadas por:

$$\mathbf{P}_E = \mathbf{P}_{N+1} \quad (3.16)$$

$$\mathbf{R}_E = \mathbf{R}_N \quad (3.17)$$

Ang e Tourassis [49] utilizam a notação de Denavit-Hartenberg juntamente com a representação vetorial para a obtenção da cinemática direta. Os vetores, representando os sistemas em cada junta, são definidos seguindo as translações e rotações utilizadas segundo Denavit-Hartenberg. Assim, os vetores que fornecem a orientação do sistema  $i$  em relação ao  $i-1$  ficam definidos por:

$$\mathbf{x}_i = \mathbf{x}_{i-1} \cdot \cos \theta_i + \mathbf{y}_{i-1} \cdot \text{sen} \theta_i \quad (3.18)$$

$$\mathbf{z}_i = \mathbf{z}_{i-1} \cdot \cos \alpha_i + (\mathbf{x}_{i-1} \cdot \text{sen} \theta_i - \mathbf{y}_{i-1} \cdot \cos \theta_i) \cdot \text{sen} \alpha_i \quad (3.19)$$

$$\mathbf{y}_i = \mathbf{z}_i \times \mathbf{x}_i \quad (3.20)$$

e o vetor posição pela expressão:

$$\mathbf{P}_i = \mathbf{P}_{i-1} + dz_{i-1} + a_i \mathbf{x}_i \quad (3.21)$$

Com  $i$  variando de 1 a  $N$  obtém-se o posicionamento do efetuador em relação ao sistema inercial. Esta abordagem elimina a principal desvantagem das matrizes homogêneas que são as multiplicações por zero e por 1.

Diferenciando-se a equação (3.3), obtém-se a expressão para o cálculo da **cinemática direta diferencial**, que relaciona os deslocamentos infinitesimais das juntas com os do efetuador.

$$\delta \mathbf{X} = \frac{\partial f(\mathbf{q})}{\partial \mathbf{q}} \cdot \delta \mathbf{q} \quad (3.22)$$

Nesta equação  $\delta \mathbf{X}$  e  $\delta \mathbf{q}$  representam, respectivamente, os deslocamentos infinitesimais no efetuador e nas juntas. As derivadas parciais em relação às

coordenadas das juntas formam a **matriz Jacobiana** do robô,  $J(\mathbf{q})$ . Dividindo ambos os membros de (3.22) por  $\delta t$  chega-se a:

$$\dot{\mathbf{X}} = J(\mathbf{q}) \cdot \dot{\mathbf{q}} \quad (3.23)$$

onde  $\dot{\mathbf{X}}$  e  $\dot{\mathbf{q}}$  representam os vetores velocidade do efetuador e das juntas, respectivamente, sendo dados por:

$$\dot{\mathbf{X}} = [v_x \quad v_y \quad v_z \quad \omega_x \quad \omega_y \quad \omega_z]^T \quad (3.24)$$

$$\dot{\mathbf{q}} = [\dot{q}_1 \quad \dot{q}_2 \quad \dots \quad \dot{q}_N]^T \quad (3.25)$$

Neste caso, a matriz Jacobiana está estabelecendo o mapeamento entre as velocidades das juntas e do efetuador. Sua dimensão é  $6 \times N$ , sendo  $N$  o número de graus de liberdade do robô. As três primeiras linhas estão associadas às velocidades lineares e as três últimas às velocidades angulares. Cada vetor coluna representa a contribuição de cada junta para as velocidades lineares e angulares do efetuador [28].

O cálculo da matriz Jacobiana é importante para o controle do robô na solução da cinemática inversa e no tratamento de forças externas que agem no efetuador [27]. Esta matriz depende da configuração atual do robô o que faz com que seja necessário o seu cálculo a cada nova posição. Um procedimento eficiente para o seu cálculo é proposto por Asada [28], com base na contribuição de cada junta ao movimento do efetuador.

### 3.4 - CINEMÁTICA INVERSA

A **cinemática inversa de posição** trata da obtenção das coordenadas das juntas, dadas as coordenadas externas do efetuador. Matematicamente, ela se resume na solução da equação (3.3).

Duas situações exigem o uso da cinemática inversa. A primeira é no planejamento de trajetórias quando feita em coordenadas externas. Neste caso as trajetórias definidas no espaço externo devem ser convertidas em coordenadas de juntas para que se possa fazer o controle da posição e do movimento de cada junta. A segunda aplicação, mais recente, é na geração de trajetórias em tempo real ou seja, a trajetória é definida de acordo com sinais captados por sensores detectando a presença de obstáculos [26].

Para a cinemática direta a solução sempre existe e é única. A obtenção da cinemática inversa, contudo, é muito mais complexa por se tratar da solução de sistemas de equações não-lineares. A solução pode não existir e, se existir, ela pode não ser única, dependendo da geometria do robô e da configuração dentro do seu espaço de trabalho. Por se tratar de um sistema de equações não-lineares, deve-se preocupar com o método de solução, com a existência da solução e com o número de soluções possíveis.

A **cinemática inversa diferencial** trata da solução da equação (3.22), ou seja, da obtenção dos deslocamentos infinitesimais nas juntas correspondentes aos deslocamentos impostos ao efetuador. Matematicamente:

$$\delta \mathbf{q} = \mathbf{J}^{-1} \cdot \delta \mathbf{X} \quad (3.26)$$

o que resulta no problema da inversão da matriz Jacobiana. Tal inversão não ocorre nos chamados pontos singulares onde o Jacobiano é nulo.

As singularidades estão relacionadas com o espaço de trabalho do robô [03]. Existem **singularidades de contorno**, nos limites do espaço de trabalho e as **singularidades internas**, que ocorrem dentro do espaço de trabalho. Em ambas as situações, ocorre a perda de um grau de liberdade, ou seja, existe alguma direção na qual o efetuador não pode se deslocar.

A Figura 3.5-(a) mostra um exemplo de singularidade de contorno e a Figura 3.5-(b), de singularidade interna. No primeiro caso, o efetuador não pode se mover na direção indicada por estar na extensão máxima. No segundo caso, o efetuador não pode girar em torno do eixo, devido ao travamento da cadeia.

Na geração do movimento, as singularidades podem ser evitadas ou pela modificação da trajetória nas suas proximidades ou se fazendo a interpolação da trajetória em coordenadas de junta nestas regiões. Pode-se ainda, em casos onde existem infinitas soluções, adotar uma delas, seguindo algum critério [27]. Este último caso será discutido novamente no Capítulo 4.

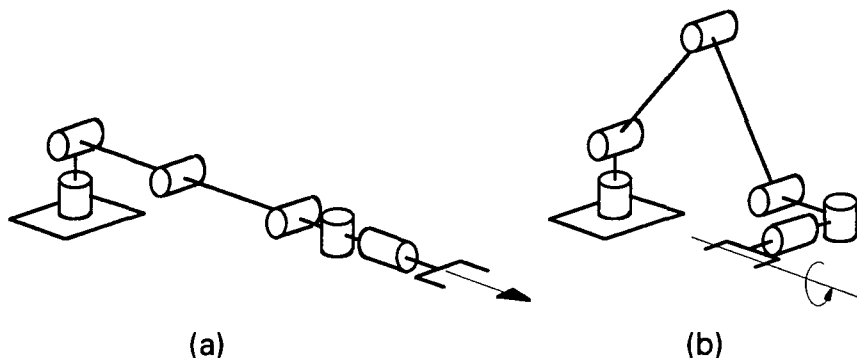


Figura 3.5 - Posições em que ocorre (a) singularidade de contorno e (b) singularidade interna [34].

### 3.5 - EXISTÊNCIA DA SOLUÇÃO

A condição de existência da solução para a cinemática inversa é que o determinante da matriz Jacobiana seja diferente de zero [50]. Quando o determinante é nulo, a configuração é dita **singular**. Para robôs com mais de seis graus de liberdade (redundantes), a configuração singular ocorre quando  $J$  não é de posto cheio.

Existem três razões possíveis para a ausência de solução [26]: a geométrica, a mecânica e a matemática.

A razão **geométrica** diz respeito a pontos que, devido a restrições mecânicas, são incompatíveis com a geometria do mecanismo (ponto fora do espaço de trabalho do robô). O número de graus de liberdade menor que 6 também contribui para a não existência da solução visto que nem todos os pontos do espaço tridimensional podem ser atingidos. A existência de uma ferramenta como efetuador também pode interferir na solução já que um aumento no comprimento do efetuador

provoca a diminuição do espaço de trabalho primário. Estas observações levam à conclusão de que a condição do ponto estar dentro do espaço de trabalho não garante a existência da solução, pois ela depende da região do espaço de trabalho onde se localiza o efetuador.

A razão **mecânica**, uma extensão da geométrica, envolve limites das juntas ou acoplamentos existentes entre as articulações e os atuadores.

A razão **matemática** diz respeito à solução de sistemas de equações não-lineares com número de incógnitas diferente do número de equações. Se a dimensão do espaço externo for menor que o número de graus de liberdade ( $M < N$ ), o robô é dito **redundante**, possibilitando um número infinito de soluções. Se a dimensão do espaço externo for igual ao número de graus de liberdade ( $M = N$ ) o número de soluções torna-se limitado, mas diferente de uma, devido às não-linearidades causadas pelas juntas rotativas. Quanto maior a presença destas juntas, maior é o número de soluções diferentes. Se a dimensão do espaço externo for maior do que o número de graus de liberdade ( $M > N$ ) a solução pertencerá a um subespaço do espaço tridimensional. É o que acontece com robôs com menos de seis graus de liberdade os quais não são capazes de varrer todo o espaço tridimensional com qualquer posição e orientação [28].

### 3.6 - NÚMERO DE SOLUÇÕES

A existência de soluções múltiplas se deve à presença de juntas rotativas na cadeia cinemática. O manipulador planar com 3 graus de liberdade da Figura 3.6 mostra duas soluções possíveis para a mesma posição do efetuador.

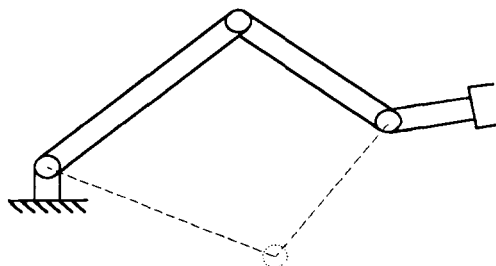


Figura 3.6 - Soluções possíveis para um manipulador com 3 graus de liberdade [03].

À medida que o número de juntas rotativas aumenta, também aumenta o número de soluções possíveis. O robô **PUMA** [03], por exemplo, é mostrado na Figura 3.7 em quatro posições que satisfazem a posição do efetuador no espaço.

O pulso *roll-bend-roll*, mostrado na Figura 3.8, possui duas soluções possíveis. Combinando-se as soluções do braço com as do pulso tem-se um total de 8 soluções diferentes para o mesmo ponto.

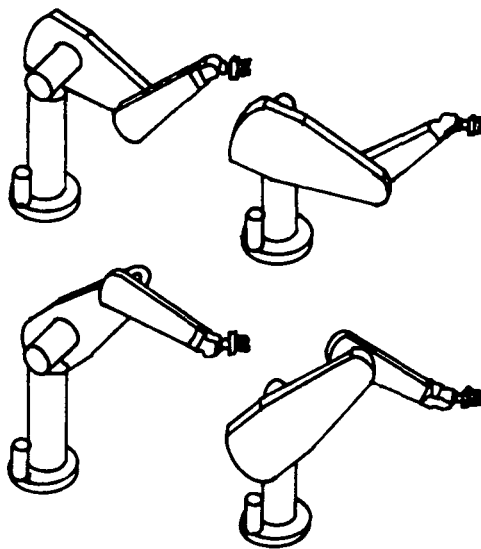


Figura 3.7 - Quatro soluções do PUMA 560 [03].

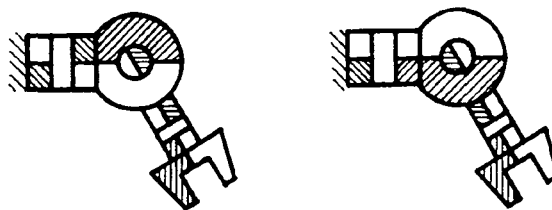


Figura 3.8 - Soluções múltiplas para o pulso [03].

Além do tipo e número de juntas, o número de soluções depende também dos parâmetros cinemáticos das ligações e dos limites das juntas. Voltando ao exemplo do robô PUMA, caso ele não possuísse o comprimento da segunda ligação ("ombro"), o número máximo de soluções cairia para 4. Levando em consideração ainda os limites das juntas, este número pode ainda ser menor.

Assim, um dado que deve ser levado em conta, na cinemática inversa é qual a solução que se deseja. Para os métodos de solução em forma fechada, isto pode ser feito pela mudança de sinais nas expressões matemáticas. Para o PUMA, por exemplo, Fu *et alli* [02] definiram constantes, valendo +1 e -1, relacionadas com o "braço" (ARM), "cotovelo" (ELBOW) e "pulso" (WRIST), que foram embutidas nas expressões. Assim, basta atribuir a estas constantes os valores +1 e -1 para se obter a configuração desejada. A combinação das três constantes levam às oito soluções possíveis para aquele robô. Já para os métodos numéricos, só se pode chegar a uma solução. Faz-se necessário então definir alguma posição inicial para que a solução desejada seja obtida. Normalmente os métodos numéricos fornecem a solução mais próxima da posição inicial fornecida.

## **CAPÍTULO 4**

### **MÉTODOS PARA A OBTENÇÃO DA CINEMÁTICA INVERSA**

#### **4.1 - INTRODUÇÃO.**

Conforme apresentado no capítulo anterior, a obtenção da cinemática inversa é complexa por se tratar de um sistema de equações não lineares. Várias abordagens tem sido desenvolvidas visando buscar um método ao mesmo tempo estável, geral e eficiente. A generalidade e a rapidez são as características mais desejadas num método de solução, apesar de que elas se contrapõem: métodos genéricos são mais lentos que os específicos. Fica então a dúvida na seleção do método mais apropriado.

Este capítulo tem a finalidade de fazer uma análise de alguns métodos de obtenção da cinemática inversa, tentando abordar suas características principais, vantagens e desvantagens. Com base neste estudo, foram escolhidos dois métodos para implementação, um analítico e outro numérico, visando compor o programa de simulação cinemática.

#### **4.2 - MÉTODOS DE SOLUÇÃO DA CINEMÁTICA INVERSA.**

Um manipulador é dito solucionável quando todos os conjuntos de variáveis de juntas, associados a um dado posicionamento do efetuador, puderem ser



obtidos numa forma explícita [03]. Neste sentido, pode-se agrupar os métodos de solução em duas categorias principais:

- métodos analíticos;
- métodos numéricos.

Os **métodos analíticos** ou em **forma fechada** são aqueles obtidos por meio de expressões analíticas de modo a não necessitar de cálculos iterativos. Estes métodos possibilitam a determinação de todas as soluções possíveis para robôs com até seis graus de liberdade. Eles podem ser subdivididos em algébricos e geométricos. Os **algébricos** baseiam-se na solução da equação cinemática, enquanto que, os **geométricos** baseiam-se em propriedades geométricas da estrutura, o que exige visão espacial da cadeia cinemática do manipulador. O fato de se ter expressões explícitas torna o método analítico bastante rápido, sendo a sua utilização muito eficiente no controle do robô, onde se exige cálculo em tempo real [53].

As soluções **numéricas** baseiam-se em cálculos iterativos e fornecem apenas uma entre as soluções possíveis. São, portanto, mais demoradas de se obter do que as soluções em forma fechada. Apesar disso, as soluções numéricas podem ser utilizadas de modo genérico pois, ao contrário da solução analítica, independem da estrutura cinemática do robô.

### 4.3 - MÉTODOS ANALÍTICOS.

Os métodos analíticos são aqueles que proporcionam soluções em forma fechada, ou seja, não se necessitam de iterações para se chegar à solução. Baseiam-se na geometria da cadeia cinemática (método geométrico), ou na equação cinemática (método algébrico).

A obtenção da solução analítica para um manipulador com 6 graus de liberdade está vinculada à condição de que três eixos adjacentes devem interceptar-se em um único ponto [27]. Isto faz com que o problema possa ser separado em dois.

Outro fato que auxilia na solução analítica é a presença de vários parâmetros de Denavit-Hartenberg nulos. Devido à importância de se ter a solução em forma fechada, muitos projetistas levaram em consideração a condição acima, de modo que quase a totalidade dos robôs existentes possuem solução analítica [03].

#### 4.3.1 - MÉTODO GEOMÉTRICO.

O método geométrico, como o próprio nome diz, baseia-se nas características geométricas da cadeia cinemática. Aproveitando características geométricas peculiares, procura-se decompor a estrutura em subconjuntos que possam ser tratados pela geometria plana.

Por meio deste método, pode-se obter facilmente expressões que indicam os valores limites do espaço de trabalho além de todas as soluções possíveis.

Este procedimento pode ser aplicado a qualquer robô, desde que, como já mencionado, ele seja solucionável.

Encontra-se na literatura pesquisada, a solução geométrica dos seguintes robôs: planar com três graus de liberdade [03], PUMA [02] e Cincinnati Milacron T [54], ambos com seis graus de liberdade.

Fu *et alli* [02], contornam o problema da multiplicidade de soluções, para o caso do robô PUMA, definindo constantes para se chegar a cada uma delas. Conforme apresentado no Capítulo 3 (ver Figuras 3.7 e 3.8), este robô apresenta um total de oito soluções distintas. Definiu-se as constantes OMBRO\_ESQUERDO, OMBRO\_DIREITO, COTOVELO\_BAIXO, COTOVELO\_ALTO, PULSO\_BAIXO e PULSO\_ALTO, cada uma delas com os valores +1 e -1. Assim, para se chegar a uma solução específica, basta escolher o conjunto de constantes correspondentes. Deve-se notar, entretanto, que uma vez escolhida a configuração, não se pode modificá-la, a menos que se mude o valor das constantes.

Pinto e Bevilacqua [72] propõem um método geométrico iterativo para robôs planares. O procedimento se baseia na correção do erro de posição do efetuador pela movimentação isolada de cada junta. Não existe correção de

orientação. O teste 6, apresentado no Capítulo 6, compara os resultados deste método com o da Análise da Posição-Zero, apresentado mais adiante.

Em seu trabalho, Hunt [54] salienta que muitas vezes se necessita de um maior conhecimento da cadeia cinemática para que se possa obter equações cinemáticas mais simples. Isto só é possível utilizando o método geométrico. Sua conclusão final é que um método genérico às vezes pode não ser adequado, por ser muito lento e por fornecer resultados que não condizem com a realidade sem que isto fique evidente para o projetista.

#### 4.3.2 - MÉTODOS ALGÉBRICOS.

A maioria dos métodos algébricos baseiam-se nas equações cinemáticas representadas pelas matrizes de transformação homogênea, conforme a equação (3.4), reproduzida a seguir.

$$\mathbf{T}_N = {}^0\mathbf{A}_1(q_1) \cdot {}^1\mathbf{A}_2(q_2) \cdot \dots \cdot {}^{i-1}\mathbf{A}_i(q_i) \cdot \dots \cdot {}^{N-1}\mathbf{A}_N(q_N) \quad (4.1)$$

O primeiro membro desta expressão,  $\mathbf{T}_N$ , representa os vetores de orientação e posição do efetuador, sendo completamente conhecido. O segundo, resulta da multiplicação de todas as matrizes de transformação, definidas no capítulo anterior pela expressão (3.2), sendo, portanto, função das coordenadas generalizadas  $\mathbf{q}$ .

O método algébrico mais conhecido é o proposto por Paul, descrito a seguir.

##### a) Método Proposto por Paul.

Por este método, as coordenadas das juntas são obtidas igualando-se os elementos correspondentes das matrizes  $\mathbf{T}_N$  e  ${}^0\mathbf{A}_N$ , representadas em (4.1). Busca-se,

assim, igualdades que, após algumas transformações, resultem na coordenada de junta procurada.

Caso não se consiga obter as relações desejadas da equação (4.1), o que frequentemente ocorre, pesquisa-se entre matrizes que relacionem sistemas intermediários. Matematicamente isto corresponde a se proceder pré e/ou pós multiplicações, em ambos os lados de (4.1), pelas inversas das matrizes  ${}^{i-1}\mathbf{A}_i$ , uma de cada vez, de modo a formar novos conjuntos de equações.

Para um robô com seis graus de liberdade, por exemplo, tem-se os seguintes conjuntos de equações matriciais disponíveis:

$$\begin{aligned}
 \mathbf{T}_6 &= ({}^0\mathbf{A}_1) \cdot ({}^1\mathbf{A}_2) \cdot ({}^2\mathbf{A}_3) \cdot ({}^3\mathbf{A}_4) \cdot ({}^4\mathbf{A}_5) \cdot ({}^5\mathbf{A}_6) \\
 ({}^0\mathbf{A}_1)^{-1} \cdot \mathbf{T}_6 &= ({}^1\mathbf{A}_2) \cdot ({}^2\mathbf{A}_3) \cdot ({}^3\mathbf{A}_4) \cdot ({}^4\mathbf{A}_5) \cdot ({}^5\mathbf{A}_6) \\
 ({}^1\mathbf{A}_2)^{-1} \cdot ({}^0\mathbf{A}_1)^{-1} \cdot \mathbf{T}_6 &= ({}^2\mathbf{A}_3) \cdot ({}^3\mathbf{A}_4) \cdot ({}^4\mathbf{A}_5) \cdot ({}^5\mathbf{A}_6) \\
 ({}^2\mathbf{A}_3)^{-1} \cdot ({}^1\mathbf{A}_2)^{-1} \cdot ({}^0\mathbf{A}_1)^{-1} \cdot \mathbf{T}_6 &= ({}^3\mathbf{A}_4) \cdot ({}^4\mathbf{A}_5) \cdot ({}^5\mathbf{A}_6) \\
 ({}^3\mathbf{A}_4)^{-1} \cdot ({}^2\mathbf{A}_3)^{-1} \cdot ({}^1\mathbf{A}_2)^{-1} \cdot ({}^0\mathbf{A}_1)^{-1} \cdot \mathbf{T}_6 &= ({}^4\mathbf{A}_5) \cdot ({}^5\mathbf{A}_6) \\
 ({}^4\mathbf{A}_5)^{-1} \cdot ({}^3\mathbf{A}_4)^{-1} \cdot ({}^2\mathbf{A}_3)^{-1} \cdot ({}^1\mathbf{A}_2)^{-1} \cdot ({}^0\mathbf{A}_1)^{-1} \cdot \mathbf{T}_6 &= ({}^5\mathbf{A}_6)
 \end{aligned} \tag{4.2}$$

Por inspeção, observa-se os elementos correspondentes em cada equação matricial, buscando-se aqueles que possam fornecer uma expressão que possibilite a solução para a junta a ser solucionada. Este procedimento se repete para outros elementos desta ou das outras equações matriciais, até que se chegue a todas as coordenadas das juntas.

Para o robô STANFORD [29], por exemplo, a segunda equação matricial está representada a seguir, mostrando-se apenas os elementos (3,4), que interessam para a obtenção de  $\theta_1$ .

$$\begin{bmatrix}
 x & x & x & x \\
 x & x & x & x \\
 x & x & x & -\text{sen}\theta_1 \cdot p_x + \text{cos}\theta_1 \cdot p_y \\
 0 & 0 & 0 & 1
 \end{bmatrix} = \begin{bmatrix}
 x & x & x & x \\
 x & x & x & x \\
 x & x & x & d_2 \\
 0 & 0 & 0 & 1
 \end{bmatrix} \tag{4.3}$$

Igualando-se os elementos (3,4) tem-se uma expressão com apenas  $\theta_1$  como variável:

$$-\text{sen}\theta_1 \cdot p_x + \text{cos}\theta_1 \cdot p_y = d_2 \quad (4.4)$$

Após algumas transformações trigonométricas chega-se ao valor de  $\theta_1$ :

$$\theta_1 = \text{atg}\left(\frac{p_y}{p_x}\right) - \text{atg}\left(\frac{d_2}{\sqrt{p_x^2 + p_y^2 - d_2^2}}\right) \quad (4.5)$$

Ao se igualar os elementos das matrizes, frequentemente se recai em expressões trigonométricas cujas soluções seguem um procedimento específico. Paul [29] e Ramos [55] apresentam algumas expressões típicas encontradas bem como seus procedimentos de resolução.

Como a maioria dos robôs industriais são solucionáveis [02], pode-se obter as suas cinemáticas inversas por este método.

Do mesmo modo que na abordagem geométrica, o conhecimento da cadeia cinemática auxilia sobremaneira na solução pelo método algébrico.

Encontra-se na literatura a solução algébrica para: robô PUMA [03][53], robô planar com três graus de liberdade [03], robô STANFORD [28][29], robô *artropóide* (3 graus de liberdade rotativos, sem pulso) [27], robô ELBOW (articulado com pulso *bend-bend-roll*) [29], robô UFU-6R-86 (articulado com pulso *roll-bend-roll*) [56], e os robôs K15 (5 graus de liberdade) e K16 (6 graus de liberdade) da Volkswagen [55].

Craig [03] apresenta o caso particular do robô YASUKAWA-MOTOMAN L-3, com cinco graus de liberdade, acionado indiretamente por meio de atuadores lineares e mecanismos de quatro barras. Esta característica torna a solução mais trabalhosa devido ao acoplamento das juntas ocasionado pelo acionamento simultâneo de duas ou mais juntas. Neste caso fez-se uma abordagem mista, algébrica e geométrica.

A cinemática inversa diferencial também pode ser obtida por métodos algébricos. Paul *et alli* [57] apresenta um método de obtenção da cinemática inversa

diferencial partindo da diferenciação da cinemática inversa de posição. Neste caso não há necessidade de se calcular a matriz Jacobiana e os pontos singulares são facilmente identificados. Como na cinemática inversa algébrica de posição, deve-se distender um certo esforço para se chegar às expressões finais.

### b) Método Utilizando os Vetores Screw.

Hunt [54][58] trata da cinemática inversa diferencial por meio dos vetores *Screw*.

Um vetor qualquer pode ser representado por um vetor *Screw*,  $\$$ , através das coordenadas dispostas conforme a expressão (4.6).

$$\$ = [L \ M \ N \ P \ Q \ R]^T \quad (4.6)$$

Nesta expressão, L, M, e N, representam as coordenadas do vetor em relação a um sistema de referência e P, Q e R, as componentes do momento que o vetor produz, em relação ao mesmo sistema. Tome-se como exemplo um vetor  $v$ , de módulo  $v$ , posicionado conforme a Figura 4.1.

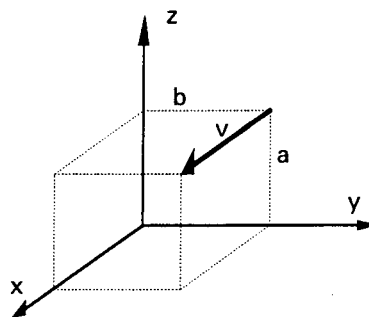


Figura 4.1 - Representação de um vetor pelas coordenadas *Screw*.

As componentes deste vetor, em relação ao sistema indicado, valem  $v_x = v$ ,  $v_y = 0$  e  $v_z = 0$ . Os momentos que este vetor produz em relação a cada eixo valem  $M_x = 0$ ,  $M_y = v \cdot a$  e  $M_z = -v \cdot b$ , obedecendo a regra da mão direita. Deste modo, as coordenadas *Screw* correspondentes ao vetor  $v$  são as seguintes:

$$\$_v = [v \ 0 \ 0 \ 0 \ v \cdot a \ -v \cdot b]^T \quad (4.7)$$

Se  $v$  for um vetor unitário, as coordenadas passam a ser:

$$\$_v = [1 \ 0 \ 0 \ 0 \ a \ -b]^T \quad (4.8)$$

Utilizando-se este conceito pode-se considerar que as colunas da matriz Jacobiana de um robô podem ser representadas por um vetor *Screw*. O sistema de referência pode ser qualquer [58]. A vantagem deste procedimento está em se obter mais rapidamente a matriz Jacobiana literal do robô e na liberdade de se utilizar qualquer sistema de referência. Pode-se escolher um sistema de forma a se conseguir com que vários elementos da matriz sejam nulos ou unitários o que facilita a obtenção literal da inversa da matriz Jacobiana e o seu determinante. Com isto se consegue agilizar a cinemática inversa de velocidades bem com a detecção dos pontos singulares. A Figura 4.2 representa a cadeia cinemática do robô PUMA, representada pelos vetores *Screw*.

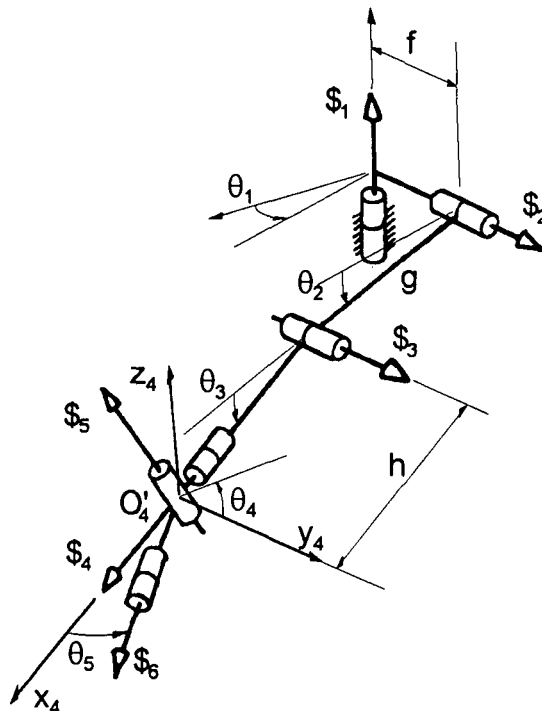


Figura 4.2 - Representação do robô PUMA utilizando vetores *Screw* [58].

Tomando-se como referência o sistema  $O_4'$ , as coordenadas e o momento do vetor correspondente à junta 3 valem, respectivamente  $[0 \ 1 \ 0]$  e  $[0 \ 0 \ -h]$ . Assim tem-se o vetor *Screw* correspondente:

$$\mathcal{S} = [0 \ 1 \ 0 \ 0 \ 0 \ -h]$$

Procedendo-se do mesmo modo para as demais juntas, tem-se a matriz Jacobiana do robô em relação ao sistema  $O_4'$ .

$$J = \begin{bmatrix} -s_{23} & 0 & 0 & 1 & 0 & c_5 \\ 0 & 1 & 1 & 0 & -s_4 & c_4 \cdot s_5 \\ c_{23} & 0 & 0 & 0 & c_4 & 0 \\ -f \cdot c_{23} & g \cdot s_3 & 0 & 0 & 0 & 0 \\ g \cdot c_2 + h \cdot c_2 & 0 & 0 & 0 & 0 & 0 \\ -f \cdot c_{23} & -(g \cdot c_3 + h) & -h & 0 & 0 & 0 \end{bmatrix} \quad (4.9)$$

O procedimento descrito acima foi aplicado aos robôs PUMA, CM-T<sup>3</sup>, esférico e articulado com cinco graus de liberdade. Para cada um deles, um sistema intermediário escolhido adequadamente foi utilizado como referência, produzindo expressões bastante simples para a matriz Jacobiana, sua inversa e seu determinante.

### c) Método Simbólico

Cheng et alli [59] desenvolveram um método simbólico para a obtenção das expressões da equação cinemática. As operações de soma e produto com funções trigonométricas são representadas pelos símbolos mostrados na Figura 4.3.

Com a utilização de tais operadores, a matriz de transformação  ${}^{i-1}A_i$ , dada pela equação (3.3), toma o aspecto apresentado na Figura 4.4.

Aplicando-se esta técnica para todas as juntas, chega-se à equação cinemática simbólica do robô, conforme mostra a Figura 4.5, para o robô PUMA.



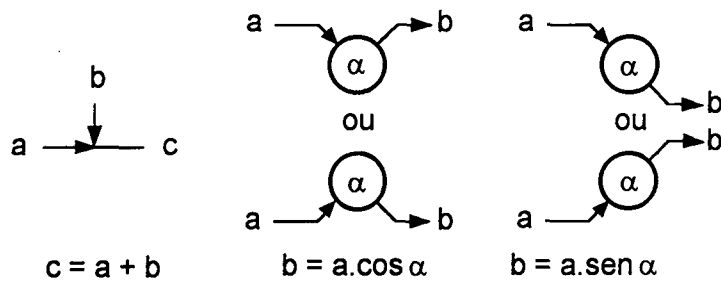


Figura 4.3 - Operadores simbólicos.

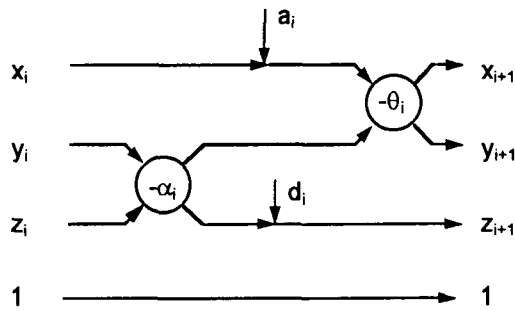


Figura 4.4 - Representação simbólica da matriz de transformação  $A_i^{-1}$  [59].

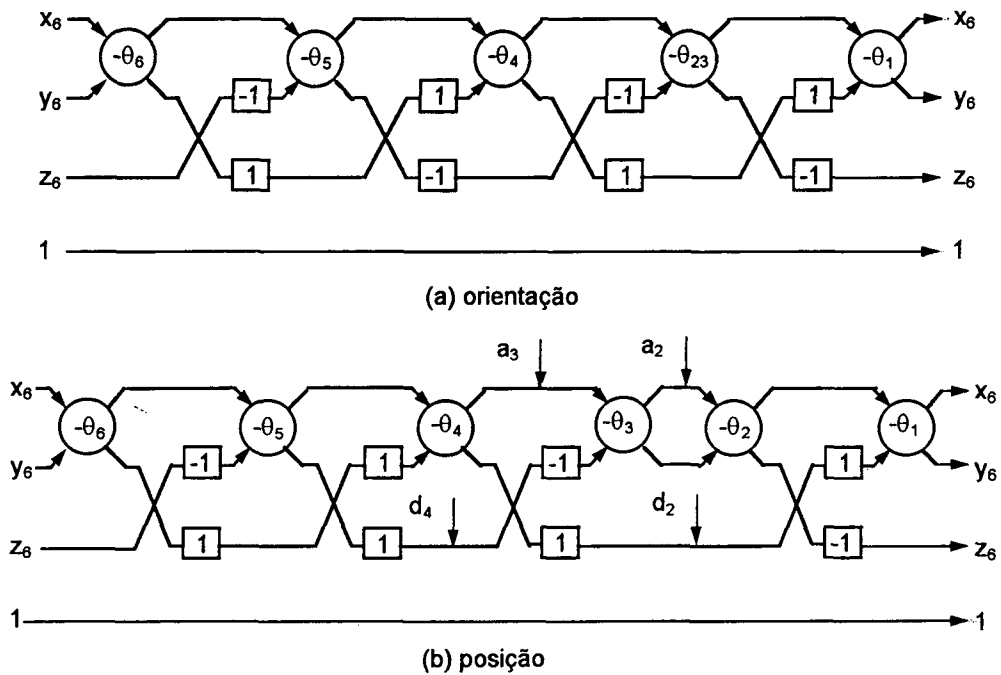


Figura 4.5 - Equação simbólica do robô PUMA: a) orientação, b) posição [59].

Por este procedimento foram obtidas as equações da cinemática inversa dos robôs PUMA, STANFORD e ELBOW.

A característica principal deste método vem da facilidade com que se procede a montagem das equações cinemáticas, evitando o produto de matrizes e as operações de multiplicação por zero. Além disso, o método permite visualizar graficamente a influência dos parâmetros das juntas na equação cinemática.

## **4.4 - MÉTODOS NUMÉRICOS**

### **4.4.1 - MÉTODO DA ANÁLISE DA POSIÇÃO-ZERO**

O procedimento apresentado por Kazerounian [51], baseia-se na idéia de modificar valores das juntas, uma de cada vez, de modo a minimizar o erro entre o posicionamento do efetuador, após a modificação da junta, e o posicionamento desejado.

A modelagem do manipulador segue o método da Análise da Posição-Zero, descrita no item 3.2. O posicionamento do efetuador (cinemática direta) para uma configuração qualquer, é obtido por meio de rotações sobre os vetores unitários  $u_k$  das juntas. Cada junta é movimentada, uma de cada vez, de modo a minimizar os erros de posição e orientação do efetuador, conforme mostra a equação (A4.11).

Apesar deste método não ser muito mais rápido que o de Newton-Raphson, ele possui a vantagem de ser estável, mesmo próximo das regiões singulares.

O Apêndice 4 apresenta o método com detalhes.

### **4.4.2 - MÉTODO DE NEWTON-RAPHSON**

Este método, comumente utilizado no problema da cinemática inversa de manipuladores não-redundantes e redundantes [27], baseia-se na expansão linear da função erro  $F(\mathbf{q})$ , definida por:

$$F(\mathbf{q}) = \mathbf{x}_e - f(\mathbf{q}) \quad (4.10)$$

onde  $f(\mathbf{q})$  é a função contínua e diferenciável que representa a cinemática direta do manipulador e  $\mathbf{x}_e$  é o vetor que representa o posicionamento desejado do manipulador.

Deseja-se obter um vetor  $\mathbf{q}^*$  que resulte em

$$F(\mathbf{q}^*) = \mathbf{x}_e - f(\mathbf{q}^*) = 0 \quad (4.11)$$

a partir de uma **solução aproximada**  $\mathbf{q}^k$ .

Expandindo  $F(\mathbf{q})$  em séries de Taylor e mantendo-se somente os dois primeiros termos, tem-se a seguinte expressão iterativa, que é a base do método:

$$\mathbf{q}^{k+1} = \mathbf{q}^k + \mathbf{J}^{-1}(\mathbf{q}^k) \cdot [\mathbf{x}_e - f(\mathbf{q}^k)] \quad (4.12)$$

O controle do processo iterativo é feito pela função erro (4.10), que deve resultar num valor menor que um valor  $\varepsilon$  previamente definido.

Este procedimento, na sua forma básica, pode não convergir caso a solução não esteja suficientemente próxima da configuração inicial. Neste caso, deve-se proceder o controle do incremento  $\mathbf{x}_e - f(\mathbf{q}^k)$  de modo que o procedimento linear convirja corretamente.

Whitney [60] propõe um algoritmo que, além de garantir a convergência para soluções relativamente próximas do valor inicial, acelera o cálculo tornando o procedimento mais eficiente. Neste caso, a expressão (4.12) passa a ser:

$$\mathbf{q}^{k+1} = \mathbf{q}^k + \mathbf{J}^{-1}(\mathbf{q}^k) \cdot s \cdot [\mathbf{x}_e - f(\mathbf{q}^k)] \quad (4.13)$$

Nesta expressão, o controle do passo "s", obtido segundo um outro processo iterativo, garante o menor erro na iteração.

Com este procedimento, tem-se uma diminuição significativa no número de iterações e no tempo de cálculo apesar de se ter um outro procedimento iterativo

para o cálculo de "s". Isto ocorre porque a frequência de cálculo de  $\mathbf{J}$  e  $\mathbf{J}^{-1}$ , que demandam tempo computacional elevado, é menor que no procedimento básico.

#### 4.4.3 - PSEUDO-INVERSA

Existem casos em que não são necessários seis graus de liberdade para executar determinadas tarefas. Um exemplo típico é o dos robôs utilizados em soldagem, onde cinco graus de liberdade são suficientes. Outro exemplo é o robô SCARA, com quatro graus de liberdade, utilizado para montagens de precisão sobre um plano. Nestes casos existe uma diferença entre o número de graus de liberdade do robô e o do espaço operacional. Além disso, tem-se as redundâncias decorrentes de pontos singulares, que também diminui o número de graus de liberdade do robô. Em ambas as situações apresentados, a matriz Jacobiana correspondente não é quadrada, tornando impossível sua inversão por métodos convencionais.

Para se tratar estes casos, utiliza-se o conceito da pseudo-inversa [61]. A relação diferencial dada por

$$\dot{\mathbf{X}} = \mathbf{J} \cdot \dot{\mathbf{q}} \quad (4.14)$$

pode ser considerada, matematicamente, como sendo o mapeamento entre o espaço de velocidades do efetuador ( $V^M$ ) e o espaço de velocidades das juntas ( $V^N$ ) [28]. A matriz Jacobiana é a transformação linear entre os dois espaços representada na figura 4.6.

O conceito da pseudo-inversa baseia-se na utilização de soluções pertencentes ao espaço nulo  $N(\mathbf{J})$  da transformação. Como existem infinitas soluções, utiliza-se critérios de otimização para se chegar a uma solução aproximada que satisfaça a condição de mínimo  $\delta\mathbf{q}$  entre todos os  $\delta\mathbf{q}$  que preenchem a condição

$$\min \|\delta\mathbf{X} - \mathbf{J} \cdot \delta\mathbf{q}\| \quad (4.15)$$

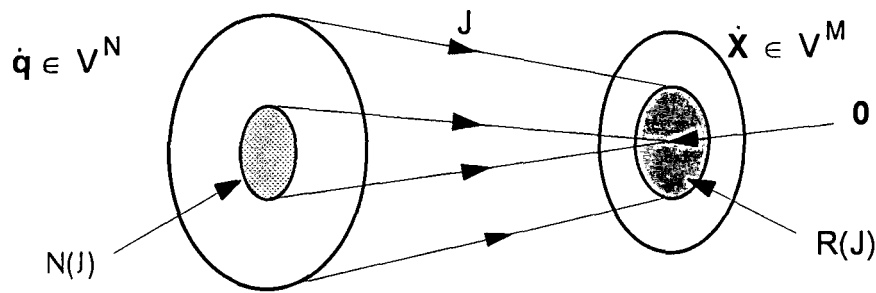


Figura 4.6 - Transformação linear entre as velocidades do efetuador e as das juntas.

No caso de **robôs com menos de seis graus de liberdade** ( $N < M$ ), o posto de  $\mathbf{J}$  não é o espaço manipulável total, apesar de ser cheio ( $\dim R(\mathbf{J}) = N$ ). O critério de otimização leva à projeção, em  $R(\mathbf{J})$ , do vetor do movimento. A solução é dada por:

$$\delta \mathbf{q} = (\mathbf{J}^T \cdot \mathbf{J})^{-1} \cdot \mathbf{J}^T \cdot \delta \mathbf{X} \quad (4.16)$$

Na **presença de singularidades**, a solução por mínimos quadrados é feita devido à diminuição do posto de  $\mathbf{J}$ . A solução, obtida entre as infinitas possíveis, devido à redundância local, é a seguinte:

$$\delta \mathbf{q} = \mathbf{J}^T \cdot (\mathbf{J} \cdot \mathbf{J}^T)^{-1} \cdot \delta \mathbf{X} \quad (4.17)$$

A utilização das pseudo-inversas é necessária para tornar o método de Newton-Raphson mais completo, ampliando a sua aplicação para qualquer manipulador com até seis graus de liberdade e permitindo que se possa operar nas proximidades de pontos singulares. Apesar disso, como será apresentado a seguir, não é um método estável em tais regiões.

Goldemberg *et alli* [30], utilizam o método de Newton-Raphson, juntamente com a pseudo-inversa, em manipuladores redundantes ( $N > 6$ ), com controle de passo e levando em conta os limites das juntas.

#### 4.4.4 - PSEUDO-INVERSA PARA REGIÕES SINGULARES

Apesar da pseudo-inversa apresentar uma solução nas proximidades de um ponto singular, ela se torna completamente instável à medida que se aproxima do ponto. Para superar tal problema, Nakamura e Hanafusa [50], propõem uma solução que pode ser considerada como uma extensão da pseudo-inversa.

Segundo os autores, a pseudo-inversa minimiza o erro mas não faz nenhum controle da magnitude da solução, podendo ela assumir valores que podem ultrapassar os limites físicos do robô. A proposta para superar este problema é modificar o procedimento de otimização, minimizando uma função que leva em conta tanto o erro quando a magnitude da solução. Utilizando este procedimento, pode-se contornar o problema de singularidade que ocorre quando se utiliza a inversa normal ou a pseudo-inversa. Dentre os métodos que se baseiam na expressão (4.14), este parece ser o mais estável.

Liu e Tsay [62], utilizaram este procedimento em um manipulador com cinco graus de liberdade. Baseado no conhecimento da geometria do robô, resolve-se a cinemática inversa de modo hierárquico, pela partição da matriz Jacobiana em submatrizes mais simples. Esta partição permite a obtenção literal das cinemáticas inversas de posição e diferencial. Assim, para pontos afastados das singularidades, a solução é algébrica e para pontos mais próximos, utiliza-se a inversa descrita neste item.

Como resultado, tem-se um aumento da eficiência computacional devido à partição do Jacobiano, aliado à estabilidade em regiões singulares. É necessário, contudo, que se conheça a geometria do manipulador, o que acarreta na perda da generalidade.

#### 4.4.5 - MÉTODO DA INTEGRAÇÃO DAS VELOCIDADES

Este método, apresentado por Gupta e Kazerooni [63], soluciona a cinemática inversa a partir das velocidades especificadas para o efetuador.

Utilizando a modelagem vetorial da Posição-Zero, o método calcula inicialmente as velocidades das juntas, utilizando a inversão da matriz Jacobiana.

Para a integração das velocidades, utilizou-se o método "preditor-corretor". O preditor escolhido foi o de Crane-Klopfenstein, e o corretor, o de Adams-Bashforth-Moulton. Tal escolha se deve ao fato deles trabalharem com passos relativamente grandes, o que, em integração numérica, melhora consideravelmente a eficiência do algoritmo.

Este método possui a vantagem de fornecer tanto a posição quanto as velocidades das juntas. Em se tratando, contudo, de um método que utiliza a inversão da matriz Jacobiana, exige-se cuidados nas proximidades dos pontos singulares, o que aumenta o tempo computacional. Mesmo assim, o tempo de cálculo pode ser de cinco a quinze vezes menor que o método de Newton-Raphson com controle de passo [63].

Tsai e Orin [64] utilizaram este procedimento, juntamente com a pseudo inversa, para a inclusão do algoritmo num micro processador específico para cálculo da cinemática inversa. A finalidade era a de se obter a solução em tempo real. No teste de um robô com seis graus de liberdade, chegou-se a uma frequência de cálculo de 2kHz com um erro dentro dos limites aceitáveis de repetibilidade dos robôs industriais [63].

#### **4.4.6 - CINEMÁTICA INVERSA DE VELOCIDADE E ACELERAÇÃO**

Ang e Tourassis [49], apresentam um método recursivo para o cálculo das cinemáticas diretas de posição, velocidade e aceleração, bem como das cinemáticas inversas de velocidade e aceleração. O procedimento utiliza as expressões (3.18) a (3.21) e suas derivadas em relação ao tempo.

Pelas fórmulas recursivas, com base na convenção de Denavit-Hartenberg, são obtidos os vetores dos sistemas de coordenadas de cada junta, representados no sistema de base.

O procedimento não leva em conta as singularidades pois, segundo os autores, pode-se evitá-las na fase de planejamento de trajetória.

No caso de se tratar de um robô com número de graus de liberdade maior ou menor que seis, utiliza-se a pseudo-inversa.

A cinemática inversa de aceleração é obtida pela solução do sistema

$$\ddot{\mathbf{X}} - \mathbf{J} \cdot \dot{\mathbf{q}} = \mathbf{J} \cdot \ddot{\mathbf{q}} \quad (4.18)$$

O método recursivo faz com que se elimine os cálculos envolvendo multiplicações por zero e um, comuns quando se utilizam as matrizes homogêneas, que demandam tempo computacional. A desvantagem principal do método é o não tratamento das singularidades.

## 4.5 - MÉTODOS MISTOS

Estão agrupados nesta categoria aqueles métodos em que parte da solução é algébrica e parte numérica. Dois exemplos de solução são apresentados a seguir. O primeiro visa solucionar o problema de um robô com pulso não esférico. O segundo, aplica o procedimento a um robô com sete graus de liberdade.

### 4.5.1 - SOLUÇÃO PARA PULSO NÃO-ESFÉRICO

Lumelsky [65] utiliza a técnica mista para a solução de um robô sem pulso esférico, no qual o acoplamento entre as juntas da estrutura principal e as do pulso inviabiliza a solução algébrica completa.

Modelada segundo Denavit-Hartenberg, a estrutura cinemática é dividida em duas partes (estrutura principal e pulso). Utilizando a técnica proposta por Paul, descrita no item 4.3.2 a), obtém-se a solução analítica para cada uma das partes. Um procedimento iterativo é, então, aplicado até que se chegue a uma precisão aceitável.

O método fornece posição exata e orientação dentro dos limites preestabelecidos. Não há controle próximo das singularidades.



Quanto à convergência o autor afirma que normalmente ela ocorre dentro de cinco iterações para um erro menor que 0,1.

O método perde em generalidade pois necessita da solução algébrica dos subconjuntos, o que inviabiliza sua utilização em procedimentos genéricos.

#### **4.5.2 - MANIPULADOR COM SETE GRAUS DE LIBERDADE**

Outra aplicação utilizando formulação mista é feita por Benati *et alli* [66], para um robô tipo "antropomórfico", com sete graus de liberdade. Este tipo de robô compõe-se de três juntas rotativas, sendo a primeira e a última esféricas. A estrutura se assemelha muito à de um braço humano.

Utilizando-se das matrizes de rotação sobre um vetor qualquer, estabelece-se algebricamente, expressões para as rotações em cada junta. Fixada uma posição e orientação para o efetuador, desenvolve-se um procedimento recursivo para adaptar o braço àquela posição do pulso. O processo recursivo é semelhante ao utilizado por Kazerounian [51] em seu algoritmo.

#### **4.6 - COMPARAÇÃO ENTRE OS MÉTODOS APRESENTADOS**

A principal conclusão a que se pode chegar, sobre a seleção do método mais adequado, é que ela está vinculada à aplicação. Foi observado, no Capítulo 3, que a cinemática inversa é utilizada no planejamento de trajetória, quando esta é feita em coordenadas espaciais, e na programação *off-line*, que é feita fora do ambiente de trabalho.

No primeiro caso, quando o planejamento é feito em tempo real, há necessidade de que o cálculo seja feito numa frequência bastante elevada, o que exige métodos rápidos. Na programação *off-line*, a necessidade maior é de se ter um método que seja genérico, para que se possa utilizá-lo em diferentes tipos de robôs. A mesma exigência se aplica no caso de análise cinemática em projetos de robôs, onde frequentemente se altera a sua geometria.

Independentemente da finalidade, as características principais que definem a eficiência de um método, são:

- velocidade,
- generalidade,
- estabilidade, e
- obtenção das soluções.

Os **métodos analíticos**, em geral, são rápidos, pois fornecem a solução em forma explícita, sem iterações. Tanto os métodos algébricos quanto os geométricos fornecem todas as soluções possíveis. Sua limitação está na restrição da solução aos robôs solucionáveis, como os de pulso esférico. Outra limitação reside na complexidade de se obter as expressões, específicas de cada robô. Tal característica não é adequada quando se deseja variar com frequência a sua geometria.

Entre os métodos analíticos, a solução geométrica é mais adequada quando se tem poucos graus de liberdade (até três), pela facilidade de visualização espacial. Para robôs com seis graus de liberdade, a solução algébrica é mais adequada, apesar de exigir intensa manipulação matemática. Tal dificuldade pode ser reduzida utilizando-se a representação simbólica [59], que substitui os produtos matriciais.

Quanto à estabilidade dos métodos analíticos, pode-se contornar o problema em pontos singulares atribuindo-se certos valores às juntas críticas, de modo a eliminar indeterminações. No entanto, podem ocorrer problemas nas vizinhanças das singularidades, onde a precisão utilizada pode não ser suficiente.

Em se tratando de generalidade, obrigatoriamente se recai em métodos numéricos. Como tais procedimentos são iterativos, tem-se um aumento no tempo computacional.

Os **métodos numéricos** estudados podem ser reunidos em três grupos:

- método da Análise da Posição-Zero,
- método de Newton-Raphson, e
- método da integração das velocidades.

O método da Análise da Posição-Zero, matematicamente apresenta-se bastante estável, mesmo em regiões próximas de pontos singulares. Nestas regiões, contudo, ele se torna mais demorado, necessitando de mais iterações. A estabilidade matemática é comprovada nos pontos singulares, quando o travamento do movimento não interfere no cálculo. Esta característica pode ser aproveitada para a obtenção dos limites do espaço de trabalho.

O método de Newton-Raphson, na sua configuração básica [27], apresenta problemas de estabilidade em regiões singulares e quando a solução estiver longe da configuração inicial. O controle do passo [60] diminui bastante o problema da configuração inicial. A utilização da pseudo-inversa [61], permite a aplicação do método em casos onde a matriz Jacobiana não é quadrada. Em regiões singulares, no entanto, ela continua instável. Neste caso, a extensão da pseudo-inversa [50] é utilizada com vantagem.

Apesar do método de Newton-Raphson melhorar bastante com as alterações descritas acima, ele se torna matematicamente mais complexo que o método da Posição-Zero.

O método da integração de velocidades [63] tem a complexidade similar ao de Newton-Raphson pois também envolve a inversão da matriz Jacobiana com suas dificuldades em regiões singulares. Contudo tem-se, neste caso, as cinemáticas inversas de velocidade e posição simultaneamente.

A cinemática inversa de velocidades e acelerações [49], tem sua importância por ser o único método a fornecer estes valores. Sua principal desvantagem está em não se levar em conta as singularidades.

Os métodos mistos, apesar de tratarem de robôs não solucionáveis, também são específicos para cada robô por utilizarem expressões deduzidas a partir de sua geometria.

Em se tratando de procurar um método ideal para a sua inclusão num programa de análise cinemática de robôs, buscou-se um método genérico, estável e com baixa complexidade computacional. O tempo de processamento não foi considerado importante por não se tratar de um problema de cálculo em tempo real. Assim, o método que se apresentou mais eficiente, foi o da Posição-Zero [51]. Isto não

significa que o método de Newton-Raphson, com suas melhorias, deva ser descartado. Como se verá nos testes, ele também se apresenta de forma satisfatória.

Na procura de um método completo, onde se obtenha a cinemática inversa de posição, velocidade e aceleração, pode-se pensar num método híbrido entre o da integração de velocidades [63] e a cinemática inversa de velocidade e aceleração [49].

Para efeito de uma análise mais aprofundada em situações particulares, optou-se pela implementação do método da análise da Posição-Zero e da solução analítica do robô PUMA. Uma breve descrição do programa é feita no capítulo seguinte. A título de comparação utilizou-se também o método de Newton-Raphson com controle de passo.

## **CAPÍTULO 5**

### **DESCRIÇÃO DO PROGRAMA**

#### **5.1 - INTRODUÇÃO**

Pela variedade de métodos apresentada no capítulo anterior, pode-se observar que existe uma preocupação muito grande na procura do método mais adequado para o cálculo da cinemática inversa e que ainda não se chegou no método ideal, que atendesse as necessidades de generalidade, eficiência, estabilidade e rapidez.. A fim de facilitar a pesquisa deste assunto, procurou-se implementar um programa que permitisse avaliar o desempenho do método da Análise da Posição-Zero e que possibilitasse, posteriormente, a inclusão nele de outros métodos existentes.

Este capítulo tem o objetivo de apresentar a proposta de um programa para a análise cinemática de robôs industriais, envolvendo principalmente a aplicação do método da Análise da Posição-Zero. A fim de facilitar a visualização do mecanismo cinemático, prevê-se um modo de representação gráfica.

Procurou-se desenvolver o programa de modo que possa ser utilizado tanto para fins didáticos quanto para pesquisa.

Por se tratar de uma proposta, algumas partes, que não dizem respeito à cinemática inversa, ainda não foram implementadas. Achou-se, entretanto, oportuna a apresentação da estrutura completa do programa, ficando como base para algum trabalho futuro.

### 5.2 - ESTRUTURA GLOBAL DO PROGRAMA

A Figura 5.1 apresenta a estrutura do programa com as suas subdivisões.

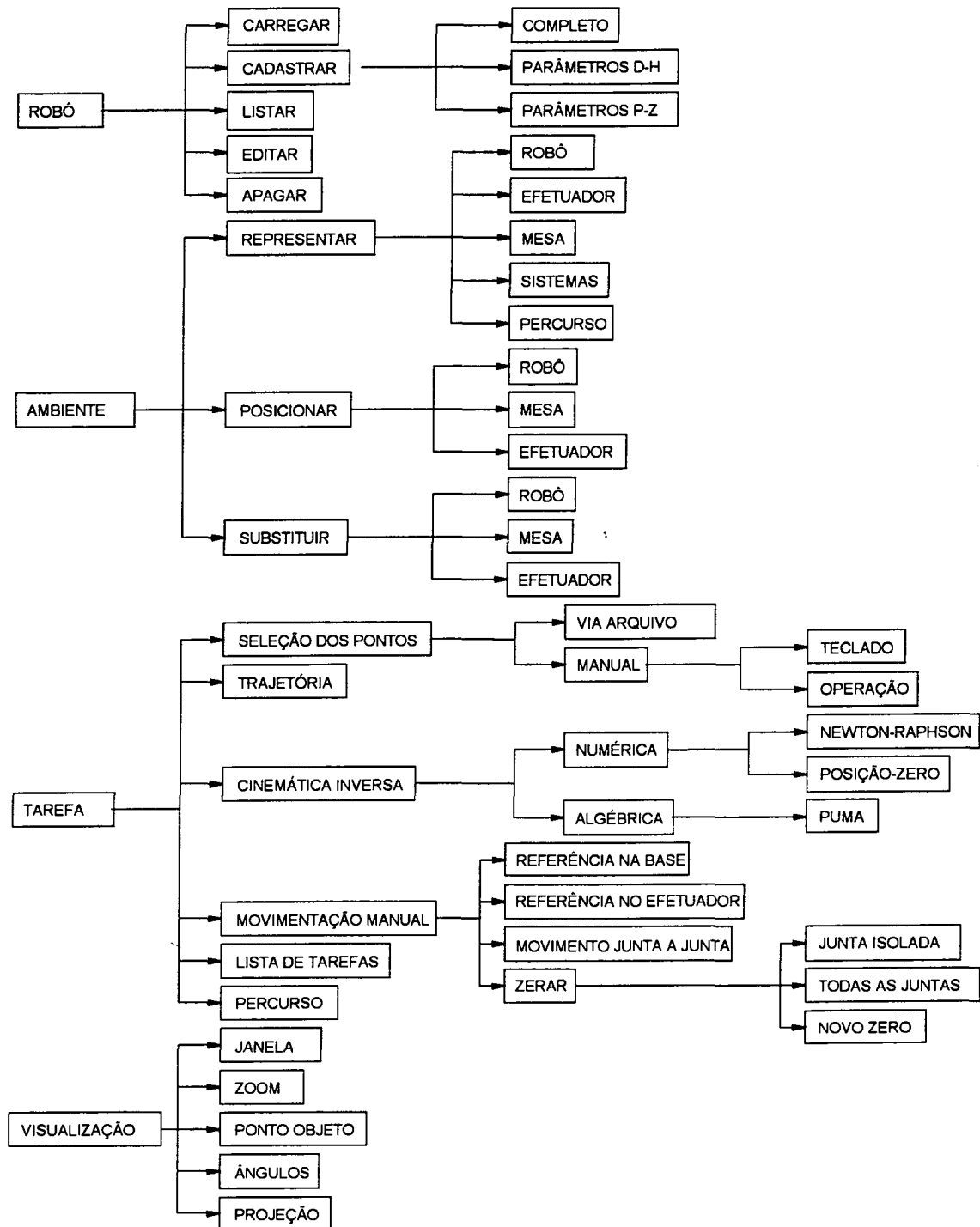


Figura 5.1 - Estrutura global do programa.

O programa foi estruturado de modo a formar um conjunto de funções, termo utilizado em linguagem C, para representar as sub-rotinas. Assim, cada opção do programa principal permite acesso à sua função correspondente que, por sua vez, tem acesso a outras funções que a formam. Por se tratar de um programa protótipo, onde várias modificações deverão ser feitas, a utilização deste tipo de estrutura é bastante apropriada pela clareza e facilidade de modificação..

Inicialmente implementado em uma estação gráfica **Intergraph IP-32**, o programa utiliza duas janelas. A primeira é utilizada para a comunicação com o usuário. A segunda janela apresenta graficamente a cadeia cinemática do robô e a sua movimentação.

O programa compõe-se de quatro blocos principais:

- Bloco 1 - **ROBÔ**
- Bloco 2 - **AMBIENTE**
- Bloco 3 - **TAREFA**
- Bloco 4 - **VISUALIZAÇÃO**

O primeiro bloco, **ROBÔ**, trata da criação e seleção do robô a ser analisado. O bloco **AMBIENTE** relaciona-se com a representação do ambiente de trabalho e das tarefas a serem executadas. O bloco **TAREFA** diz respeito aos percursos que o robô deve executar. Nele se inclui a cinemática inversa. O quarto e último bloco, **VISUALIZAÇÃO**, apresenta opções de mudança do modo de visualização gráfica do robô.

### 5.2.1 - BLOCO "ROBÔ"

O bloco **ROBÔ** é responsável pela seleção do robô a ser analisado e pela inclusão de novos robôs ainda não cadastrados no programa. Pode-se, ainda, criar uma estrutura cinemática diferente daquelas já existentes. Vale lembrar que são válidas somente cadeias cinemáticas abertas.

lisad

Existem cinco subdivisões neste bloco: **CARREGAR**, **CADASTRAR**, **LISTAR**, **EDITAR** e **APAGAR**.

#### a) **CARREGAR**

A opção **CARREGAR** seleciona o robô desejado entre aqueles que já estão cadastrados. Os parâmetros de cada robô estão armazenados em dois arquivos identificados pelo nome do robô e pelas extensões "**PAR**" e "**TXT**", cujas estruturas se encontram apresentadas no Apêndice 3. O arquivo com a extensão "**PAR**" fornece os dados numéricos necessários ao programa. O arquivo com extensão "**TXT**" apresenta as mesmas informações, só que em forma de relatório, o que facilita o entendimento dos dados numéricos.

No programa em linguagem C funções pertencentes à opção **CARREGAR**, são: `escolhe_arquivo()`, `le_parametros_robo()`, `imprime_parametros()` e `prepara_tela()`. A função `escolhe_arquivo()` lê os nomes dos robôs armazenados no arquivo `colecão.dat` e apresenta-os na tela para que a escolha possa ser efetuada. A leitura dos nomes é feita por meio da função `le_arquivo_colecao()`. A função `le_parametros_robo()` carrega todos os dados armazenados no arquivo com extensão **PAR**, correspondente ao robô escolhido.

Após a leitura dos dados, faz-se a conversão, de graus para radianos, dos ângulos das juntas ( $\acute{U}$ ), das torções das ligações ( $\acute{O}$ ) e dos ângulos de visualização, já que as funções trigonométricas da linguagem C utilizam tal unidade. O conteúdo da variável `flagrep` é alterado para indicar que um robô foi carregado, permitindo o acesso às demais opções do programa.

Todos os parâmetros lidos são apresentados na tela por meio da função `imprime_parametros()`.

A função `prepara_tela()` executa a preparação para a representação gráfica, detalhada no Apêndice 2. A sequência desta função é a seguinte:



## b) CADASTRAR

Tratando-se de um robô ainda não cadastrado, tem-se a opção **CADASTRAR** que permite a inclusão de novos robôs. Esta inclusão pode ser feita de três modos:

- pelos dados de posição e orientação das juntas,
- pelos parâmetros de Denavit-Hartenberg ou
- pelos vetores da Posição-Zero.

O cadastramento a partir dos dados de posição e orientação das juntas, opção **COMPLETO**, pode ser utilizado quando não se tem nenhuma informação sobre os parâmetros de Denavit-Hartenberg nem sobre os vetores da Posição-Zero do robô. Neste caso, propõe-se um procedimento para se calcular tais parâmetros. O Apêndice 1 apresenta detalhes do procedimento. A entrada das posições e orientações das juntas e do efetuador é feita pela função **entrada\_teclado()**. Também são introduzidos dados sobre os limites de posição, velocidade e aceleração das juntas. O cálculo dos parâmetros cinemáticos é feito na função **calcula\_parametros()**. Os resultados são apresentados na tela por meio da função **imprime\_parametros()**. Por fim os dados são gravados nos arquivos de extensão PAR e TXT, por meio da função **grava\_parametros\_roboto()**.

O cadastramento a partir dos **PARÂMETROS DE DENAVIT-HARTENBERG** inicia-se pela introdução destes parâmetros e dos limites das juntas ( posição, velocidade e aceleração ). A função correspondente é **entrada\_teclado\_DH()**. Após o término da entrada de dados, são calculados os vetores da Posição-Zero, por meio da função **calcula\_PZ\_de\_DH()**. Este procedimento é apresentado no Apêndice 1. Posteriormente os resultados são mostrados na tela ( função **imprime\_parametros()** ) e gravados nos arquivos do robô ( função **grava\_parametros()** ).

O cadastramento via **PARÂMETROS DA POSIÇÃO-ZERO** faz o procedimento contrário ao anterior. Introduce-se os vetores da Posição-Zero e os limites

das juntas ( função `entrada_teclado_PZ()` ) e calcula-se os parâmetros de Denavit-Hartenberg ( função `calcula_DH_de_PZ()` ). Do mesmo modo que na opção anterior, os resultados do cálculo são apresentados na tela e posteriormente gravados.

Estas três opções foram implementadas para permitir que se tenha os dois conjuntos de parâmetros, independentemente do modo como o robô foi cadastrado.

### c) LISTAR, EDITAR e APAGAR

A opção **LISTAR**, apresenta o nome de todos os robôs já cadastrados. A função que executa esta opção é `lista_arquivos()`. Ela faz a leitura dos nomes dos robôs armazenados no arquivo `colecao.dat`, por meio da função `le_arquivo_colecao()`, e apresenta-os na tela.

Qualquer modificação nos parâmetros do robô pode ser feita, a qualquer momento, por meio da opção **EDITAR**, que aumenta a agilidade do programa quando ainda não se tem uma cadeia cinemática perfeitamente definida. Esta opção não foi implementada.

Qualquer robô pode ser retirado da lista por meio da opção **APAGAR**. Isto é feito através da eliminação do nome do robô da lista constante no arquivo `colecao.dat`. Os arquivos referentes aos dados do robô, contudo, só são apagados via sistema operacional.

### 5.2.2 - BLOCO "AMBIENTE"

Este bloco permite definir o modo como o robô deve ser representado, bem como o posicionamento relativo entre robô, sistema auxiliar e efetuador. Pode-se também proceder a troca do robô e/ou do efetuador. Suas opções são: **REPRESENTAR**, **POSICIONAR** e **SUBSTITUIR**.

A opção **REPRESENTAR** define o que deve ser representado graficamente: robô, efetuador, mesa, sistemas de coordenadas e percurso do efetuador. A função correspondente é **representar\_ambiente()**. Nela, alteram-se os conteúdos das variáveis relativas aos elementos que devem ser representados, de modo que possam ser identificados dentro da função **desenha\_robô\_WF()**, que é a função responsável pela representação gráfica do robô na tela.

A opção **POSICIONAR** permite que se defina a posição relativa do robô, do sistema auxiliar de referência e do efetuador. Esta opção é utilizada quando não há nenhuma informação sobre o posicionamento dos três sistemas. Os posicionamentos são introduzidos por meio de três vetores: posição da origem, orientação do eixo  $x$  e orientação do eixo  $y$ . Estes dados irão compor a matriz correspondente do robô, do sistema auxiliar e do efetuador, e poderão ser gravados num arquivo de tarefa (extensão **AMB**). Esta opção não foi implementada

A opção **SUBSTITUIR** permite que se altere o posicionamento do robô, do sistema auxiliar e do efetuador, no caso em que eles já tenham sido definidos numa tarefa ou pela opção **POSICIONAR**. Os novos posicionamentos são introduzidos do mesmo modo que na opção anterior. Esta opção não foi implementada.

### 5.2.3 - BLOCO "TAREFA"

Este é o bloco mais importante do programa, pois é nele que está incluída a função relacionada com a cinemática inversa. As opções são as seguintes: **SELEÇÃO DOS PONTOS, TRAJETÓRIA, CINEMÁTICA INVERSA, MOVIMENTAÇÃO MANUAL, LISTA DE TAREFAS e PERCURSO.**

#### a) SELEÇÃO DOS PONTOS

Esta opção faz a seleção dos pontos que irão definir o percurso a ser executado. O percurso é composto por vários segmentos os quais podem ser retas, no

caso de interpolação cartesiana, ou uma curva desconhecida, no caso de interpolação nas juntas. A seleção pode ser feita: **VIA ARQUIVO** e **MANUAL**.

Pela seleção **VIA ARQUIVO**, os pontos são obtidos pela leitura de um arquivo com extensão **AMB** cujo formato se encontra no apêndice 3. A função que executa a leitura do arquivo é **le\_percurso()**.

A seleção **MANUAL** permite que os pontos sejam escolhidos de duas formas: via **TECLADO** e via **OPERAÇÃO**. A seleção manual por **TECLADO** define os pontos pela introdução dos vetores referentes ao posicionamento do robô, do sistema auxiliar e do efetuador, seguido da tarefa. Em outras palavras, todos os dados que formam o arquivo da tarefa ( extensão **AMB** ) são introduzidos manualmente. Os sistemas de coordenadas são definidos pelo vetor posição da origem e pelas orientações dos vetores **y** e **z**. O terceiro eixo é calculado pelo produto vetorial dos outros dois.

A função que permite a seleção via teclado é **le\_percurso\_teclado()**. Duas outras funções auxiliam na criação das matrizes homogêneas: **prod\_vet()**, que calcula o terceiro eixo do sistema e **monta\_matriz\_homo()**, que executa a montagem da matriz homogênea correspondente aos quatro vetores (**x**, **y**, **z** e **p**).

No final da função **le\_percurso\_teclado()**, faz-se a gravação em arquivo para preservar os dados inseridos. A gravação é feita num arquivo com extensão **AMB**, por meio da função **grava\_percurso()**.

A seleção por **OPERAÇÃO** define os pontos do percurso pela movimentação do efetuador até os limites de cada segmento. Cada ponto é armazenado na memória e posteriormente gravado num arquivo de tarefa. Esta opção não está implementada.

## b) TRAJETÓRIA

A opção **TRAJETÓRIA** seleciona o modo como se vai executar a trajetória, ou seja, a maneira como se fará a interpolação entre os pontos escolhidos do percurso [02][27][67]. Esta opção não foi implementada.

A título de aplicação da cinemática inversa, implementou-se o planejamento do percurso cartesiano proposto por Paul [68].

### c) CINEMÁTICA INVERSA

A opção **CINEMÁTICA INVERSA** permite que se selecione um procedimento **NUMÉRICO** ou **ALGÉBRICO** para a sua obtenção. Como já mencionado no Capítulo 4, o método algébrico é específico para cada robô. Assim, para que se possa utilizar esta opção, é necessário que a cinemática inversa correspondente esteja implementada. A única opção algébrica implementada é a do robô PUMA [02]. Para solução numérica, tem-se o método da **POSIÇÃO-ZERO**. A função **seleciona\_cin\_inv()** executa a seleção do método da cinemática inversa a ser utilizado. As funções **cin\_inv\_PZ()** e **cin\_inv\_PUMA()**, que calculam as cinemáticas inversas implementadas, estão apresentadas no Apêndice 5.

### d) MOVIMENTAÇÃO MANUAL

A **MOVIMENTAÇÃO MANUAL** permite que se movimente as juntas como se estivesse operando o robô manualmente. O movimento pode ser referenciado a um sistema de coordenadas situado na **BASE** ou no **EFETUADOR**. Ele pode ainda ser executado pelo deslocamento **JUNTA A JUNTA**.

Entre os três movimentos possíveis, o único implementado foi o último, executado na função **mov\_junta\_a\_junta()**. Nesta função, pede-se qual a junta a ser movimentada e qual a sua nova posição. Interpola-se, então, posições intermediárias e executa-se o desenho de cada uma delas, pela função **desenha\_robo\_WF()**, de modo a produzir a animação do movimento entre as duas posições.

Para facilitar o posicionamento do robô, tem-se a opção **ZERAR**, que pode ser executada para apenas uma junta (**JUNTA ISOLADA**), para todas elas

(**TODAS AS JUNTAS**), além de permitir que se defina uma nova posição para os zeros das juntas (**NOVO ZERO**).

#### e) **LISTA DE TAREFAS**

A opção **LISTA DE TAREFAS** apresenta o nome de todas as tarefas gravadas em arquivos com extensão **AMB**. A estrutura da função é similar àquela que apresenta a lista de robôs ( **lista\_arquivos()** ), na opção **ROBÔ - LISTAR**. Esta função não foi implementada.

#### f) **PERCURSO**

Após definidos os pontos limites de cada segmento, o método para o cálculo da cinemática inversa e o tipo de interpolação de trajetória, calcula-se, por meio da função **percurso()**, as coordenadas das juntas para cada ponto de cada segmento do percurso total. A figura 5.2 apresenta o fluxograma da função.

Na ETAPA 1, faz-se o planejamento dos percursos entre os pontos limites de cada segmento, que pode ser uma linha reta ( interpolação cartesiana ) ou uma curva qualquer ( interpolação nas juntas ). As referências [29] e [68] apresentam o procedimento com detalhes.

No caso da interpolação **cartesiana**, os pontos intermediários são calculados utilizando a rotação de um sistema de coordenadas em torno de um vetor [29]. Como as posições inicial e final do segmento são conhecidas, calcula-se o ângulo de rotação e o vetor em torno do qual se deve girar o sistema inicial para se chegar no final.

As orientações intermediárias são obtidas pela divisão do ângulo de rotação em tantas partes quantas forem os pontos intermediários. As posições intermediárias são obtidas pela divisão da distância entre os pontos inicial e final em tantas partes quantos forem os pontos intermediários.

Se a interpolação for feita **nas juntas**, basta que se tenha as matrizes correspondentes aos pontos inicial e final do segmento.

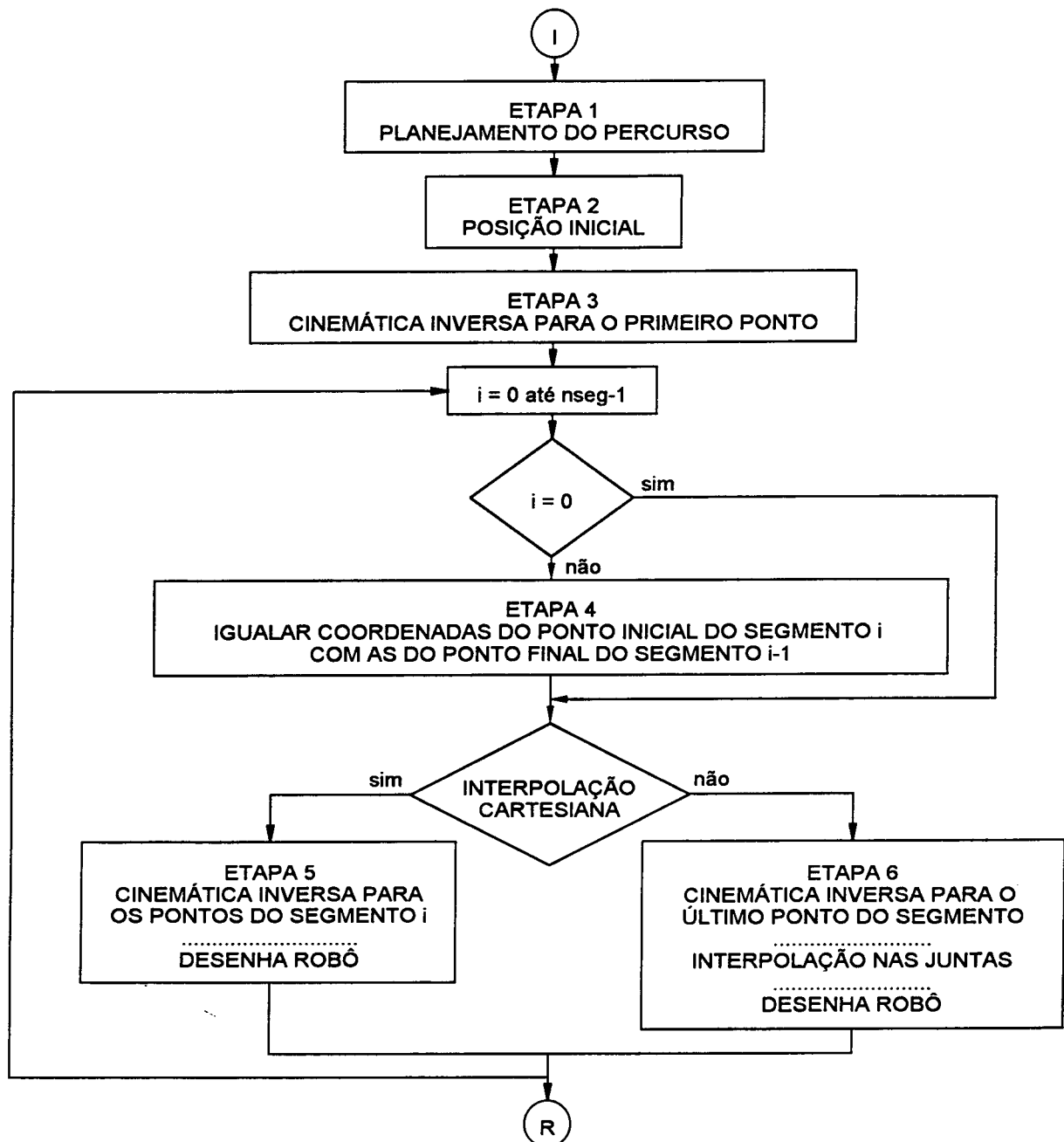


Figura 5.2: Descrição da função **percurso()**.

Na ETAPA 2, é definida a posição a partir da qual serão calculadas as coordenadas do primeiro ponto do percurso. Como foi observado no Capítulo 4, a

cinemática inversa numérica exige que se defina uma posição inicial. A função **posicao\_inicial()** define estas coordenadas.

A partir da posição inicial e da matriz de posicionamento do primeiro ponto, calcula-se a cinemática inversa para este ponto e faz-se a representação gráfica da configuração calculada (ETAPA 3).

Definidas as coordenadas das juntas correspondentes ao primeiro ponto do primeiro segmento, parte-se para as etapas de cálculo dos demais segmentos.

Na ETAPA 4, iguala-se as coordenadas das juntas do ponto inicial do segmento a ser calculado, às do último ponto do segmento anterior, por se tratar do mesmo ponto.

Se o segmento for uma reta, parte-se para a interpolação cartesiana (ETAPA 5). Neste caso, calcula-se a cinemática inversa para cada ponto intermediário do segmento. Caso se deseje a interpolação nas juntas, calcula-se as coordenadas das juntas correspondentes ao ponto final do segmento e procede-se à interpolação dos ângulos (ETAPA 6). Após cada cálculo das coordenadas das juntas faz-se a representação gráfica da configuração obtida.

Todas as coordenadas das juntas são armazenadas numa variável, de modo que se pode repetir o movimento tantas vezes quanto se queira. Estes valores podem também ser gravados num arquivo com extensão **RES**, para que possa ser utilizado posteriormente. A função **grava\_juntas\_percurso()** grava os resultados.

#### 5.2.4 - BLOCO "VISUALIZAÇÃO"

A representação gráfica, do modo como foi implementada, utilizando a representação *wire-frame*, pode gerar dúvidas. Para que se possa ter uma visualização adequada, torna-se necessária a mudança dos ângulos de visualização e do ponto visualizado (ponto objeto). Tais mudanças são possíveis neste bloco.

As opções do bloco VISUALIZAÇÃO são as seguintes: **JANELA**, **ZOOM**, **PONTO OBJETO**, **ÂNGULOS**, e **PROJEÇÃO**.



A opção **JANELA** permite alterar o seu tamanho nos casos em que a representação gráfica não aproveite a totalidade da área da tela ou quando a representação ultrapasse seus limites. A função correspondente é `altera_janela()`. A janela é definida a partir do ponto objeto que, normalmente, corresponde à origem do sistema da base do robô. Os valores correspondentes às margens direita e superior são positivos, enquanto que os das margens esquerda e inferior são negativos. Após inseridos os novos valores das margens, faz-se o recálculo da janela e da matriz de mapeamento e projeção ( funções `calcula_janela()` e `calcula_matriz_map_proj()` ) antes de se proceder o desenho do robô.

A opção **ZOOM** aproxima ou afasta o observador do ponto objeto. O efeito de aproximação e afastamento é obtido pela diminuição ou aumento do tamanho da janela do desenho. Isto se faz por meio da multiplicação destes valores por uma constante ( `zoom` ), que representa a percentagem de ampliação ou redução desejada. O valor 1 equivale ao tamanho original. Valores maiores que 1 afastam o desenho e os menores, aproximam. A função que executa tais mudanças é `altera_zoom()`.

A opção **PONTO OBJETO** altera a posição do ponto central de visualização do desenho. A partir deste ponto é que se define a janela do desenho e da tela. Normalmente o Ponto Objeto coincide com a base do robô. Esta opção altera o ponto objeto para o efetuador, para o sistema da mesa ou para qualquer outro ponto de interesse.

A mudança do ponto objeto é particularmente necessária no caso de se seleccionar os pontos de um percurso via operação do robô, podendo-se assim, obter o ponto com maior precisão. Esta função não foi implementada.

A visualização do desenho pode ser alterada pela opção **ÂNGULOS**. Tem-se dois ângulos de visualização: o ângulo **horizontal**, definindo uma rotação do desenho em torno de um eixo vertical, e o ângulo **vertical**, que define uma rotação em torno de um eixo horizontal. A função responsável pela alteração dos ângulos de visualização é `muda_visualizacao()`. A alteração visa obter uma melhor clareza na representação gráfica. Ela é feita lentamente, em intervalos, de modo a produzir uma

animação do movimento. Para cada imagem deve-se recalcular a matriz de visualização ( função `calcula_matriz_visu_obj()` ).

A opção **PROJEÇÃO** modifica a representação do desenho. A representação do robô pode ser feita de duas formas: **isométrica** ou **cônica exata** [24]. A seleção é feita por intermédio da função `altera_projecao()`. Para a projeção cônica exata, a localização do observador está pré-definida no arquivo de dados do robô.

## **CAPÍTULO 6**

### **UTILIZAÇÃO DO PROGRAMA E TESTES**

#### **6.1 - INTRODUÇÃO**

Este Capítulo tem por finalidade apresentar o procedimento de utilização das etapas implementadas no programa. Estão incluídos, ainda, os resultados de alguns testes relacionados com a cinemática inversa.

Apesar do método de Newton-Raphson não ter sido implementado no programa, ele também foi utilizado separadamente, num outro programa [70], para a avaliação do desempenho, juntamente com o método da Posição-Zero.

Alguns robôs foram cadastrados para serem utilizados nos testes e para se testar as funções que cadastram os robôs.

#### **6.2 - UTILIZAÇÃO DO PROGRAMA**

Neste item, são apresentados alguns procedimentos para a utilização das rotinas do programa. Com as funções já implementadas, pode-se cadastrar robôs, operá-los manualmente (junta a junta) e executar uma tarefa pré-definida, utilizando a cinemática inversa. Além disso, pode-se modificar a visualização do robô para facilitar a observação da tarefa. A seguir, são descritos os procedimentos de cadastramento, operação manual e execução de percurso.

### a) CADASTRAMENTO DE UM ROBÔ

Ao se iniciar o programa, tem-se o menu principal (Figura 6.1), composto dos quatro blocos, descritos no item 5.2.

```
=====
MENU PRINCIPAL
=====
1 - ROBÔ
2 - AMBIENTE
3 - TAREFA
4 - VISUALIZAÇÃO
5 - FIM
=====
OPÇÃO =>
```

Figura 6.1 - Menu principal.

O cadastramento é feito dentro da opção 1 - **ROBÔ**, cujo menu está apresentado na Figura 6.2.

```
=====
ROBÔ
=====
1 - CARREGAR
2 - CADASTRAR
3 - LISTAR
4 - EDITAR
5 - APAGAR
6 - VOLTAR
=====
OPÇÃO =>
```

Figura 6.2 - Menu **ROBÔ**

A opção 2 - **CADASTRAR** apresenta as alternativas apresentadas na Figura 6.3.

```
=====
ROBÔ - CADASTRAR
=====
1 - COMPLETO
2 - PARÂMETROS DE D-H
3 - PARÂMETROS DA P-Z
4 - VOLTAR
=====
OPÇÃO =>
```

Figura 6.3 - Opção **ROBÔ - CADASTRAR**

Considerando que não se conhece nenhum dos conjuntos de parâmetros cinemáticos, seleciona-se a opção 1 - **COMPLETO**. Nesta opção, introduz-se dados de posição e orientação das juntas e do efetuador. O programa, então, calcula os parâmetros de Denavit-Hartenberg e os vetores da Posição-Zero.

Tome-se como exemplo o robô STANFORD [29], apresentado na Figura 6.4. Os dados inseridos estão apresentados na Figura 6.5. A introdução das coordenadas de cada vetor é feita ao mesmo tempo, separadas por um espaço. Todos os vetores são representados no sistema situado na base do robô.

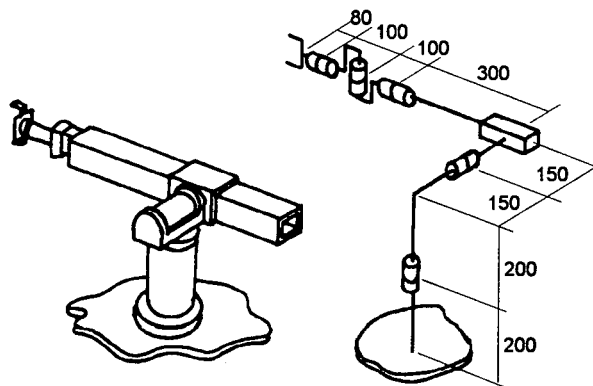


Figura 6.4 - Robô STANFORD.

Após a entrada dos dados, são definidos os parâmetros gráficos *default*, os quais podem ser posteriormente alterados e gravados.

A sequência de cálculo dos parâmetros cinemáticos se inicia pelo cálculo das origens dos sistemas de coordenadas, que são apresentados na tela para confirmação ou alteração. Para o exemplo em questão, as origens calculadas estão apresentadas na Figura 6.6.

Confirmadas as posições das origens, calculam-se os **vetores de corpo** da Posição-Zero e os eixos  $x_i$  e  $z_i$  dos sistemas de coordenadas de cada junta. Do mesmo modo que no caso das origens, os eixos  $x_i$  também são apresentados na tela para conferência. Para o robô **Stanford**, estes valores estão apresentados na Figura 6.7.

Confirmados os valores calculados, parte-se para o cálculo dos parâmetros de Denavit-Hartenberg. Após concluídos os cálculos, são apresentados,

na tela, os dados de entrada, os dois conjuntos de parâmetros cinemáticos calculados e os parâmetros gráficos. Todos os valores apresentados são gravados em arquivos com extensão **PAR** e **TXT**. No exemplo, são criados os arquivos **STANFORD.PAR** e **STANFORD.TXT**.

ENTRADA DE DADOS			
NOME DO ROBÔ		=>	<i>stanford</i>
NÚMERO DE GRAUS DE LIBERDADE		=>	6
DADOS SOBRE AS JUNTAS			
JUNTA 1	TIPO ( r OU p )	=>	<i>r</i>
	POSIÇÃO ( mm )	=>	<i>0 0 200</i>
	ORIENTAÇÃO ( - )	=>	<i>0 0 1</i>
JUNTA 2	TIPO ( r OU p )	=>	<i>r</i>
	POSIÇÃO ( mm )	=>	<i>150 0 400</i>
	ORIENTAÇÃO ( - )	=>	<i>1 0 0</i>
JUNTA 3	TIPO ( r OU p )	=>	<i>p</i>
	POSIÇÃO ( mm )	=>	<i>300 0 400</i>
	ORIENTAÇÃO ( - )	=>	<i>0 1 0</i>
JUNTA 4	TIPO ( r OU p )	=>	<i>r</i>
	POSIÇÃO ( mm )	=>	<i>300 300 400</i>
	ORIENTAÇÃO ( - )	=>	<i>0 1 0</i>
JUNTA 5	TIPO ( r OU p )	=>	<i>r</i>
	POSIÇÃO ( mm )	=>	<i>300 400 400</i>
	ORIENTAÇÃO ( - )	=>	<i>0 0 1</i>
JUNTA 6	TIPO ( r OU p )	=>	<i>r</i>
	POSIÇÃO ( mm )	=>	<i>300 500 400</i>
	ORIENTAÇÃO ( - )	=>	<i>0 1 0</i>
DADOS SOBRE O EFETUADOR			
	POSIÇÃO ( mm )	=>	<i>300 580 400</i>
	DIREÇÃO ( $u_a$ )	=>	<i>0 1 0</i>
	ORIENTAÇÃO ( $u_t$ )	=>	<i>0 0 -1</i>

Figura 6.5 - Entrada de dados para o robô STANFORD na opção **COMPLETO**.

sistema	origem			
0	[	0.000	0.000	0.000 ]
1	[	0.000	0.000	400.000 ]
2	[	300.000	0.000	400.000 ]
3	[	300.000	400.000	400.000 ]
4	[	300.000	400.000	400.000 ]
5	[	300.000	400.000	400.000 ]
6	[	300.000	580.000	400.000 ]

MODIFICAR ALGUMA POSIÇÃO ? (S)im ou (N)ão ? ==>

Figura 6.6 - Origens calculadas.

sistema	eixo x			
0	[	1.000	0.000	0.000 ]
1	[	0.000	1.000	0.000 ]
2	[	0.000	0.000	1.000 ]
3	[	0.000	0.000	1.000 ]
4	[	1.000	0.000	0.000 ]
5	[	1.000	0.000	0.000 ]
6	[	1.000	0.000	0.000 ]

MODIFICAR ALGUMA POSIÇÃO ? (S)im ou (N)ão ? ==>

Figura 6.7 - Eixos  $x_i$  calculados.

## b) OPERAÇÃO MANUAL

A operação manual executa movimentos isolados em cada junta do robô. O procedimento se inicia pela seleção do robô, que é feita pelas opções **ROBÔ - CARREGAR**. É apresentada, então, uma lista numerada dos robôs cadastrados. Faz-se a seleção pelo número correspondente ao robô desejado.

Feita a seleção, retorna-se ao menu principal e seleciona-se a opção **TAREFA**, cujo menu é apresentado na Figura 6.8. Selecionando-se a opção **4 - MOVIMENTAÇÃO MANUAL**, tem-se o menu apresentado na Figura 6.9.

A opção **3 ( JUNTA A JUNTA )** é então selecionada. Para cada movimento, pede-se o número da junta a ser movimentada e o valor da nova posição. O procedimento se repete até que se digite 0 (zero), que faz com que se retorne ao menu anterior.

TAREFA	
1 -	SELECIONAR PONTOS
2 -	TRAJETÓRIA
3 -	CINEMÁTICA INVERSA
4 -	MOVIMENTAÇÃO MANUAL
5 -	LISTAR TAREFAS
6 -	EXECUTAR PERCURSO
7 -	VOLTAR

OPÇÃO =>

Figura 6.8 - Menu **TAREFA**.

TAREFA - MOVIMENTAÇÃO MANUAL	
1 - REFERÊNCIA NA BASE	
2 - REFERÊNCIA NO EFETUADOR	
3 - JUNTA A JUNTA	
4 - ZERAR	
5 - VOLTAR	
OPÇÃO =>	

Figura 6.9 - Menu **MOVIMENTAÇÃO MANUAL**.

Selecione-se, como exemplo, o robô **artropoidal** [34], apresentado na Figura 6.10-a. Para a posição indicada, as coordenadas das juntas e o posicionamento do efetuador são também apresentados na tela. A Tabela 6.1 mostra os valores correspondentes ao posicionamento da Figura 6.10-a.

Tabela 6.1 - Posicionamento das juntas e do efetuador, correspondentes à configuração da Figura 6.10-(a).

JUNTA	VALOR	MATRIZ DE POSICIONAMENTO DA GARRA			
1	0				
2	0	0	0	1	450
3	-90	0	1	0	0
4	90	-1	0	0	-200
5	90	0	0	0	1
6	-90				

Movimentando-se as juntas 1, 2, 4 e 6, respectivamente, para as posições 90 , 45 , -45 e 0 , o robô assume a configuração da Figura 6.10-b, à qual correspondem os valores apresentados na Tabela 6.2.

Tabela 6.2 - Posicionamento correspondente à configuração da Figura 6.10-b.

JUNTA	VALOR	MATRIZ DE POSICIONAMENTO DA GARRA			
1	90				
2	45	-1	0	0	0
3	-90	0	1	0	424.2641
4	-45	0	0	-1	-50
5	90	0	0	0	1
6	0				



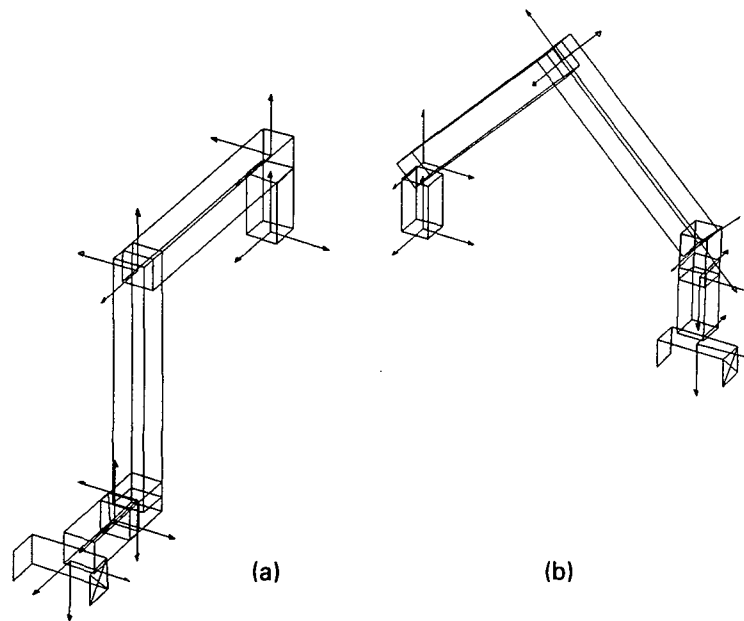


Figura 6.10 - Robô **ARTROPOIDAL** nas posições apresentadas nas Tabelas 6.1 e 6.2.

### c) EXECUÇÃO DE UM PERCURSO

O percurso a ser executado deve, inicialmente, ser introduzido pela opção **TAREFA - SELECIONAR PONTOS**. O menu correspondente é apresentado na Figura 6.11.

```
=====
TAREFA - SELECIONAR PONTOS
=====
1 - VIA ARQUIVO
2 - MANUAL
3 - VOLTAR
=====
OPÇÃO =>
```

Figura 6.11 - Menu **SELECIONAR PONTOS**.

Caso o percurso já esteja gravado, pode-se carregá-lo por meio da opção **1 - VIA ARQUIVO**. Do contrário, seleciona-se a opção **2 - MANUAL**, que permite a inclusão dos dados via teclado ou pelo posicionamento manual do robô. Esta última alternativa não foi implementada.

Tome-se como exemplo, uma tarefa para o robô **SCARA**, apresentada na Figura 6.12. Para inserir a tarefa, seleciona-se, a partir do menu principal, as opções **TAREFA**, **SELECIONAR PONTOS**, **MANUAL** e **TECLADO**, conforme a Figura 5.1. O programa, então, pede os dados apresentados na Figura 6.13.

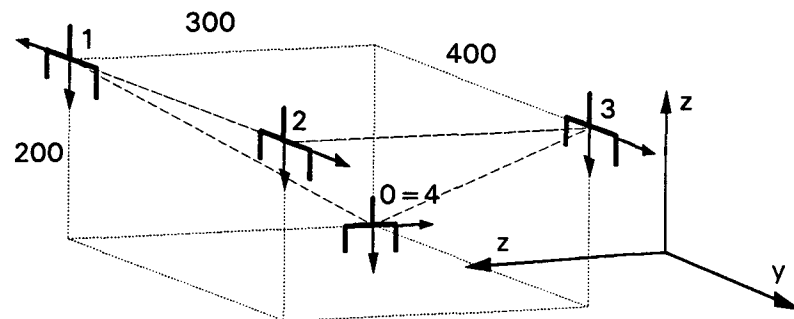


Figura 6.12 - Tarefa exemplo para o robô SCARA, armazenada no arquivo **SCARA.AMB**.

Concluída a entrada dos dados, a tarefa é apresentada na tela na forma de matrizes dos sistemas de coordenadas relativos ao robô, ao sistema auxiliar, ao efetuador e aos pontos que definem o percurso. Em seguida, a tarefa é gravada num arquivo com extensão **AMB**.

Definido o percurso, retorna-se ao menu **TAREFA** (Figura 6.8). Faz-se, então, a seleção da cinemática inversa a ser utilizada ( opção 3). Caso o método escolhido seja o da Posição-Zero, pede-se também a precisão desejada e a estratégia para a definição do peso  $c$ , utilizado no cálculo dos ângulos das juntas (equação A4.24 do Apêndice 4). Esta estratégia pode ser escolhida entre dez propostas, as quais estão apresentadas a seguir:

- Estratégia 1: Equação A4.30 para todas as juntas.
- Estratégia 2:  $c = 1$ , para a junta 1;  
Equação A4.30, para as juntas 2 a N-1;  
 $c = 0$ , para a junta N.
- Estratégia 3:  $c = 1$ , para as juntas 1 a 3;  
 $c = 0$ , para as juntas 4 a N;

- Estratégia 4:  $c = 1$ , para todas as juntas;
- Estratégia 5: Equação A4.30, para as juntas 1 a 3;  
 $c = 0$ , para as juntas 4 a N;

```

ENTRADA DO PERCURSO VIA TECLADO
DEFINIÇÃO DO PERCURSO
NOME DO PERCURSO:  scara

SISTEMAS DE REFERÊNCIA EM RELAÇÃO AO INERCIAL

ROBÔ
Eixo Y (Yx Yy Yz):      0 1 0
Eixo Z (Zx Zy Zz)      0 0 1
Posição (Px Py Pz):    -100 0 0

SISTEMA AUXILIAR
Eixo Y (Yx Yy Yz):      0 1 0
Eixo Z (Zx Zy Zz)      0 0 1
Posição (Px Py Pz):    0 0 600

FERRAMENTA
Eixo Y (Yx Yy Yz):      0 1 0
Eixo Z (Zx Zy Zz)      0 0 1
Posição (Px Py Pz):    0 0 0

PERCURSO:  scara
NÚMERO DE SEGMENTOS:      4
NÚMERO DE PONTOS POR SEGMENTO:  10

PONTO INICIAL DO SEGMENTO 0
Eixo Y (Yx Yy Yz):      -1 0 0
Eixo Z (Zx Zy Zz)      0 0 -1
Posição (Px Py Pz):    200 -200 0

PONTO INICIAL DO SEGMENTO 1
Eixo Y (Yx Yy Yz):      0 -1 0
Eixo Z (Zx Zy Zz)      0 0 -1
Posição (Px Py Pz):    500 -200 200

PONTO INICIAL DO SEGMENTO 2
Eixo Y (Yx Yy Yz):      0 1 0
Eixo Z (Zx Zy Zz)      0 0 -1
Posição (Px Py Pz):    500 200 200

PONTO INICIAL DO SEGMENTO 3
Eixo Y (Yx Yy Yz):      0 1 0
Eixo Z (Zx Zy Zz)      0 0 -1
Posição (Px Py Pz):    200 200 200

PONTO FINAL DO SEGMENTO 3
Eixo Y (Yx Yy Yz):      -1 0 0
Eixo Z (Zx Zy Zz)      0 0 -1
Posição (Px Py Pz):    200 -200 0

```

Figura 6.13 - Dados de entrada para a tarefa SCARA.AMB.

A partir da estratégia 6, propõem-se variações do peso segundo uma curva definida a qual pode ser uma reta, uma parábola do 2º grau ou uma parábola cúbica. As curvas, representadas na Figura 6.14, obedecem as equações (6.1) e (6.2).

$$c = \left[ \frac{j - N}{1 - N} \right]^n \quad (6.1)$$

$$c = \frac{j^n - N^n}{1 - N^n} \quad (6.2)$$

Nas expressões (6.1) e (6.2), N representa o número de graus de liberdade do robô, j a junta rotativa para a qual se vai calcular o ângulo e n é um expoente que assume os valores 1, 2 e 3, dependendo da curva que se deseje (reta, parábola do 2º grau ou parábola cúbica).

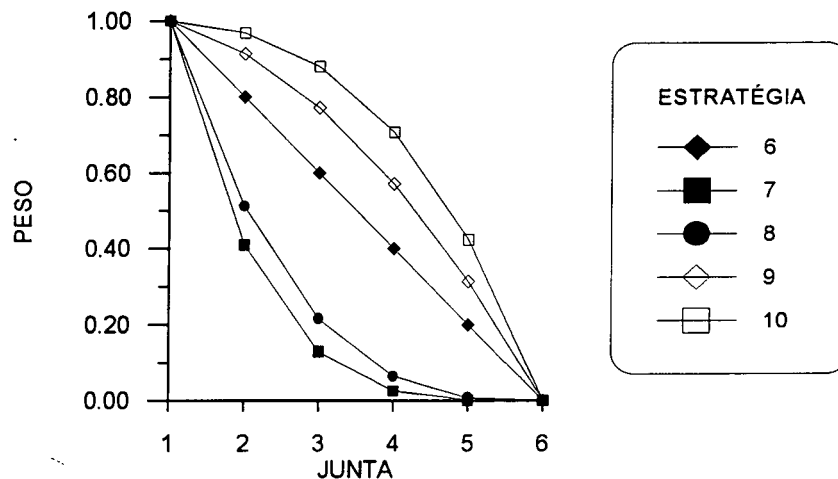


Figura 6.14 - Variação do peso segundo as equações (6.1) e (6.2), para as estratégias 6 a 10.

Deste modo, as estratégias 6 a 10 são as seguintes:

- Estratégia 6: Equação (6.1) e  $n = 1$  (interpolação linear).
- Estratégia 7: Equação (6.1) e  $n = 2$  (parábola do 2º grau).
- Estratégia 8: Equação (6.1) e  $n = 3$  (parábola do 3º grau).

- Estratégia 9: Equação (6.2) e  $n = 2$  (parábola do 2º grau).
- Estratégia 10: Equação (6.2) e  $n = 3$  (parábola do 3º grau).

Para o exemplo foi utilizada a **estratégia 5**, por ser uma das que forneceu melhores resultados, como se poderá observar nos testes apresentados mais adiante.

Selecionado o percurso e o método de cálculo da cinemática inversa, escolhe-se a opção **6 - EXECUTAR PERCURSO**. Nesta opção, inicialmente deve-se inserir os dados correspondentes à posição inicial, definindo-se assim, a configuração que o robô tomará em cada ponto do percurso. A Tabela 6.3 e Figura 6.15 mostram a posição inicial para a tarefa SCARA.AMB.

Tabela 6.3 - Coordenadas da posição inicial para o cálculo do percurso SCARA.AMB.

JUNTA	COORDENADA
1	$-90^\circ$
2	$90^\circ$
3	$0^\circ$
4	300 mm

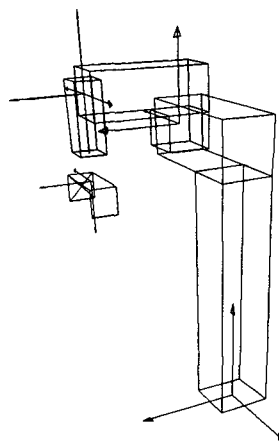


Figura 6.15 - Posição inicial do percurso SCARA.AMB.

Para cada início de segmento, pede-se o tipo de interpolação que se deseja (cartesiana ou nas juntas). No caso da tarefa SCARA.AMB, optou-se pela interpolação cartesiana em todos os segmentos, o que requer o cálculo da cinemática inversa para todos os pontos do percurso. A Figura 6.16 apresenta as posições extremas dos segmentos do percurso escolhido.

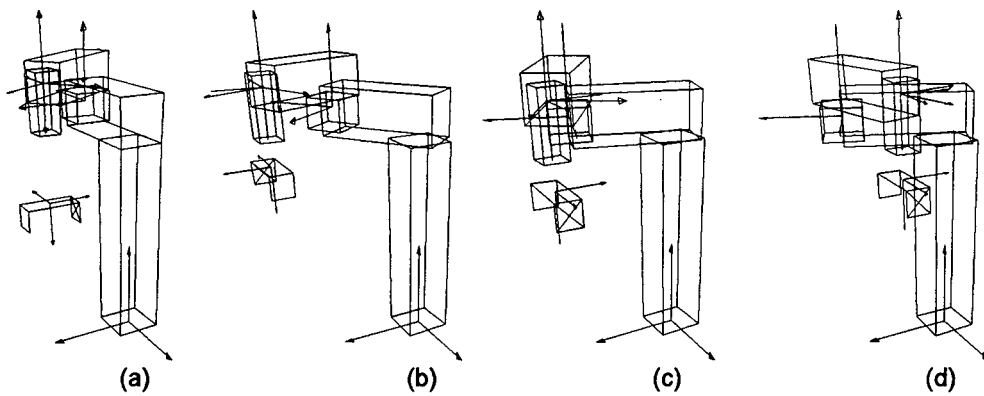


Figura 6.16 - Posições extremas de cada segmento do percurso SCARA.AMB.

- (a) Início do segmento 0 e final do segmento 3;
- (b) Início do segmento 1;
- (c) Início do segmento 2;
- (d) Início do segmento 3.

Após terminados os cálculos, pode-se gravar os resultados em um arquivo com extensão **RES**. Os resultados do percurso SCARA.AMB, estão apresentados, na forma de gráfico, na Figura 6.17.

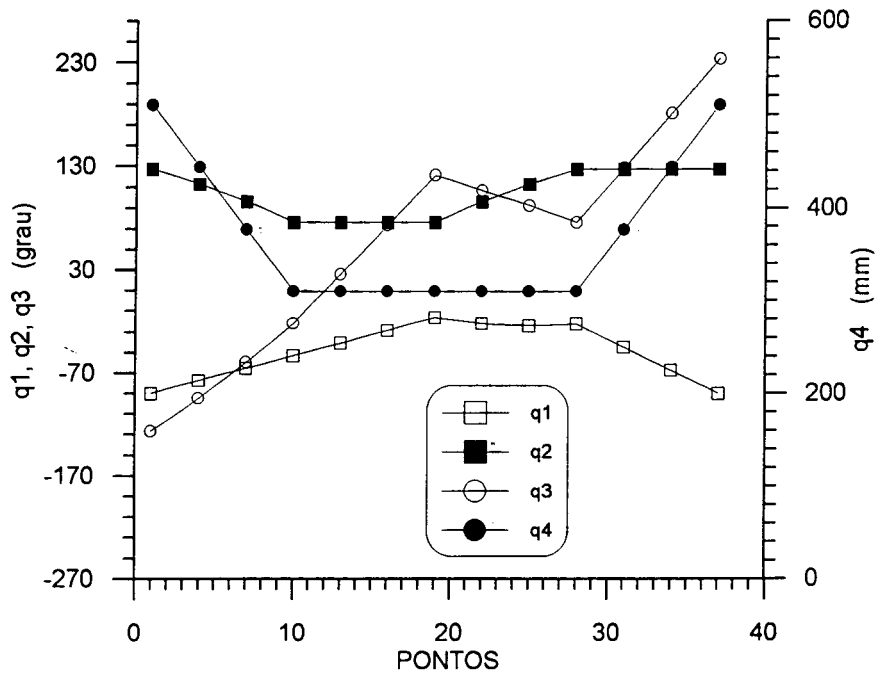


Figura 6.17 - Resultados do percurso SCARA.AMB, apresentado na Figura 6.12.

### 6.3 - TESTES

Este item apresenta uma série de cinco testes efetuados utilizando o método da Posição-Zero com o objetivo de verificar o seu comportamento ao longo do espaço de trabalho. Para fins comparativos, utilizou-se também o método de Newton-Raphson.

Os testes desenvolvidos são os seguintes:

- Teste 1 - Comportamento dos métodos em pontos diferentes do espaço de trabalho;
- Teste 2 - Comportamento dos métodos com a variação da precisão exigida;
- Teste 3 - Comportamento do método da Posição-Zero nas diferentes estratégias para a definição do peso;
- Teste 4 - Comprovação do método da Posição-Zero com um resultado extraído da literatura;
- Teste 5 - Comportamento de diferentes robôs na execução de uma mesma tarefa.
- Teste 6 - Comparação do método da Análise da Posição-Zero com o método geométrico iterativo [72], para manipulador planar não-redundante e redundante.

#### a) TESTE 1

O teste 1 visa mostrar o comportamento dos métodos da Posição-Zero e Newton-Raphson, em diferentes posicionamentos, dentro do espaço de trabalho. Para isto, selecionou-se o robô **ELBOW**, com seis graus de liberdade. Definiu-se seis pontos, devendo cada um dos quais ser atingido a partir de uma mesma posição inicial. Como mostra a Figura 6.18, o primeiro ponto (1), situa-se numa região, aparentemente sem problemas de singularidades. Os demais pontos foram escolhidos de modo a se aproximarem de um ponto singular, que, neste caso, corresponde à

máxima extensão do braço. O ponto de número seis situa-se fora do espaço de trabalho do robô.

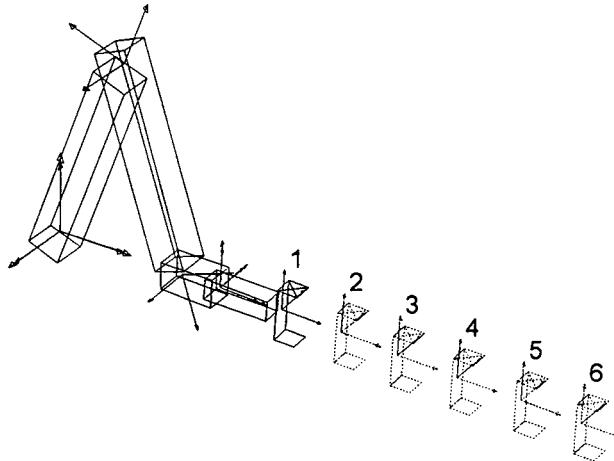


Figura 6.18 - Robô ELBOW. Posições utilizadas no teste 1.

A Tabela 6.4 apresenta o posicionamento do efetuador em cada ponto, por meio dos vetores  $\mathbf{u}_a$ ,  $\mathbf{u}_t$  e  $\mathbf{P}_H^G$ , respectivamente, vetor aproximação, vetor transversal e vetor posição do efetuador, representação que segue o proposto por Kazerounian [51]. A cinemática inversa, para cada um dos pontos, foi calculada a partir de uma mesma posição inicial, apresentada na Tabela 6.5. Os resultados do teste estão apresentados nas figuras 6.19 para a Posição-Zero e 6.20 para Newton-Raphson..

Tabela 6.4 - Pontos selecionados para o teste 1

PONTO	$\mathbf{u}_a$			$\mathbf{u}_t$			$\mathbf{P}_H^G$		
1	0	1	0	0	0	1	0	1200	0
2	0	1	0	0	0	1	0	1500	0
3	0	1	0	0	0	1	0	1800	0
4	0	1	0	0	0	1	0	2100	0
5	0	1	0	0	0	1	0	2400	0
6	0	1	0	0	0	1	0	2700	0

Tabela 6.5 - Coordenadas da posição inicial para o cálculo dos pontos do teste 1.

JUNTA	COORDENADA
1	$90^0$
2	$90^0$
3	$-90^0$
4	$90^0$
5	$90^0$
6	$0^0$



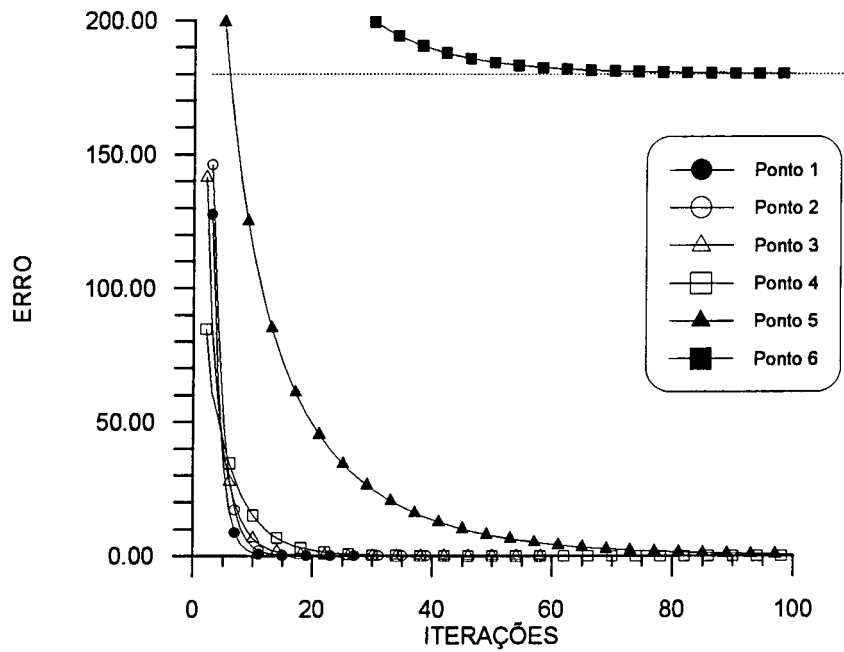


Figura 6.19 - Variação do erro com o número de iterações utilizando o método da **Posição-Zero**.

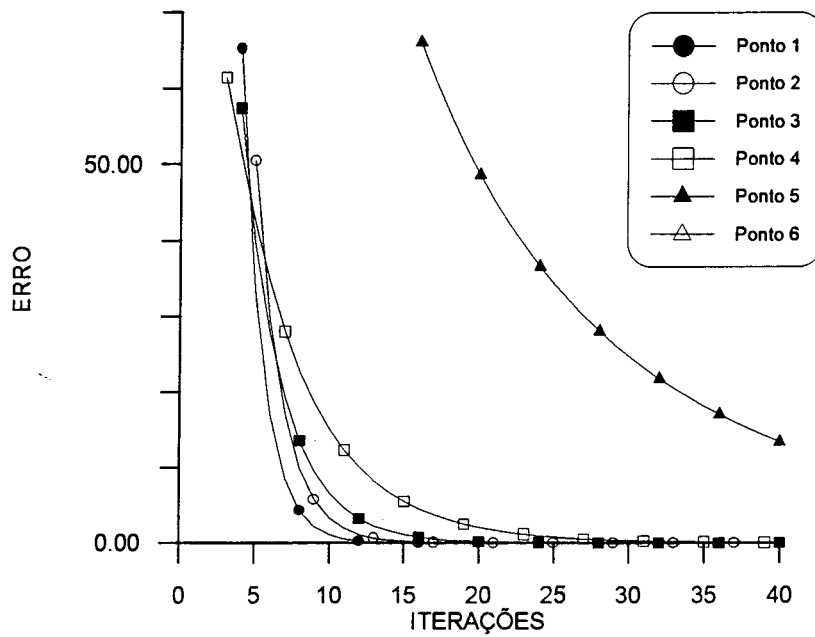


Figura 6.19-a - Ampliação da Figura 6.19.

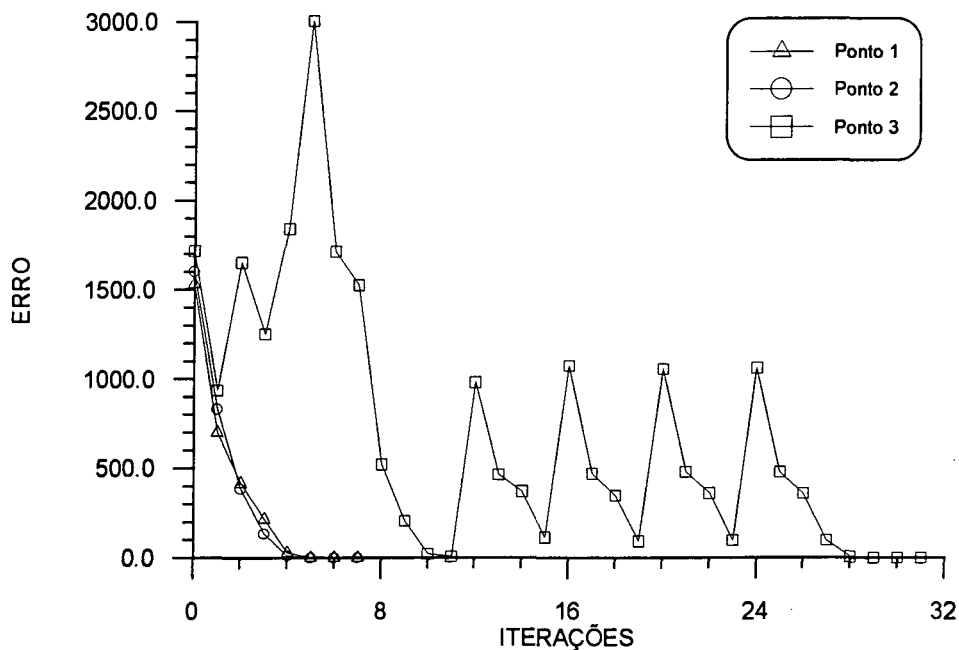


Figura 6.20 - Variação do erro com o número de iterações utilizando o método de **Newton-Raphson**.

As Figuras 6.19 e 6.20 apresentam a variação do erro em cada iteração, para cada método utilizado. A Figura 6.19-a apresenta uma ampliação da Figura 6.19 para se observar melhor o comportamento do método da Posição-Zero. Os erros são calculados pelas expressões (A4.12) e (A4.13), para o método da Posição Zero, e pela expressão (4.10) para o método de Newton-Raphson.

A Tabela 6.6 apresenta, como resultado, o número de iterações necessário para cada ponto. Para o método da Posição-Zero fixou-se um limite de cem iterações e para o método de Newton-Raphson, este limite era de sessenta iterações. Os resultados dos testes são apresentados nas Figuras 6.19, 6.19-a e 6.20 e na tabela 6.6.

Tabela 6.6 - Número de iterações necessárias para convergência em cada método.

PONTO	NÚMERO DE ITERAÇÕES	
	POSIÇÃO-ZERO	NEWTON-RAPHSON
1	29	7
2	40	7
3	59	31
4	N.C.	N.C.
5	N.C.	N.C.
6	N.C.	N.C.

Obs.: N.C. = não convergiu no número de iterações especificado.

Vale lembrar que não se está avaliando o aspecto quantitativo da variação dos erros, visto que as expressões fornecem resultados que não podem ser confrontados. Como já mencionado, o teste visa apenas mostrar como cada método se comporta na extensão do espaço de trabalho.

A Tabela 6.6 mostra, para os dois métodos, que apenas os três primeiros pontos convergiram dentro do número de iterações pré-definido.

Para o método da Posição-Zero, há um aumento considerável do número necessário de iterações, à medida que o ponto se aproxima da singularidade. Isto pode ser observado através dos pontos 1, 2 e 3. Os pontos 4 e 5 não convergiram dentro das 100 iterações pré-definidas para o método. Apesar disso, pode-se observar pela Figura 6.19, que a tendência era de convergir, pois, para os dois pontos, o erro estava tendendo a zero não ocorrendo problemas de instabilidade. Para o ponto 6, não há convergência pelo fato do ponto estar fora do espaço de trabalho. Nota-se, no entanto, que o método continua estável mesmo para este ponto. O fato do erro estabilizar no valor 180 confirma que o robô atingiu sua extensão máxima (2520 mm). O erro encontrado representa a distância entre o efetuador, na máxima extensão, e a posição desejada, que era de 2700 mm.

Para o método de Newton-Raphson, a Tabela 6.6 e a Figura 6.20 mostram uma rápida convergência, obtida em 7 iterações, para os dois primeiros pontos. O terceiro ponto apresentou uma certa instabilidade antes de convergir, necessitando de 31 iterações. Isto se deve ao fato da solução estar mais distante do ponto inicial do que os dois primeiros pontos. O quarto ponto não convergiu dentro das 60 iterações pré-definidas. A princípio imaginou-se que isto se devia ao fato de se estar aproximando de um ponto singular. Um novo cálculo, desta vez utilizando o ponto 3 como posição inicial, mostrou que o problema não era a proximidade do ponto singular mas sim, o fato da posição final se encontrar afastada demais da posição inicial. Após ser recalculada, a solução foi obtida em 6 iterações. Os pontos 5 e 6 não convergiram mesmo tomando o ponto 4 como posição inicial. Isto se deve à aproximação do ponto singular, no caso do ponto 5, e da presença da singularidade, no caso do ponto 6.

Pelos resultados apresentados no teste 1, pode-se constatar a sensibilidade do método de Newton-Raphson a pontos singulares e a soluções

distantes da posição inicial. Para o caso de pontos "bem comportados", o método apresentou-se bastante rápido. O método da Posição-Zero, por sua vez, apesar de ser mais demorado, apresenta-se estável, mesmo no caso de pontos fora do espaço de trabalho.

## b) TESTE 2

O segundo teste, avalia o número de iterações necessários para se obter a precisão exigida. O robô escolhido para este caso é o PUMA, com seis graus de liberdade. O robô deve posicionar o efetuador no ponto indicado na Tabela 6.7. A posição inicial está representada na Figura 6.21 e na Tabela 6.8.

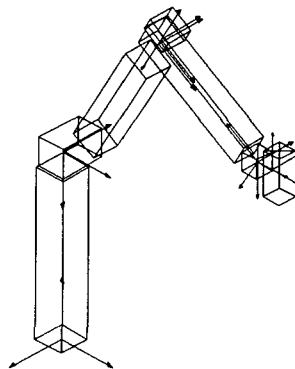


Figura 6.21 - Robô PUMA.

Tabela 6.7 - Ponto selecionado para o teste 2.

$u_a$	0	1	0
$u_t$	0	0	1
$P_H^G$	-149.09	700	370

Tabela 6.8 - Coordenadas da posição inicial para o cálculo do ponto no teste 2.

JUNTA	COORDENADA
1	$90^0$
2	$-60^0$
3	$180^0$
4	$0^0$
5	$-30^0$
6	$90^0$

A Tabela 6.9 apresenta os resultados para os dois métodos. Pode-se observar que, para o ponto escolhido, o método de Newton-Raphson converge rapidamente com um erro bastante pequeno. Trata-se, neste caso, de um ponto afastado de singularidades.

Tabela 6.9 - Variação do número de iterações com a precisão desejada.

PRECISÃO	NÚMERO DE ITERAÇÕES	
	POSIÇÃO-ZERO	NEWTON-RAPHSON
$10^{-1}$	18	4
$10^{-2}$	25	4
$10^{-3}$	31	4
$10^{-4}$	38	4
$10^{-5}$	44	5
$10^{-6}$	50	5
$10^{-7}$	56	5

### c) TESTE 3

O teste 3 avalia as alternativas de definição do peso utilizado no cálculo das coordenadas das juntas no método da Posição-Zero. Conforme apresentado no Apêndice 4, o peso representa um valor que permite corrigir mais a posição ou a orientação do efetuador, no cálculo de uma coordenada de junta. Nos casos extremos, o valor zero leva à correção somente da orientação, enquanto que, o valor 1 corrige apenas a posição.

Conforme comentado no Capítulo 2, geralmente a estrutura mecânica dos manipuladores é projetada de modo que as primeiras ligações possuem comprimentos bem maiores que os do pulso. Pode-se afirmar então que a função das primeiras juntas é de posicionar enquanto que as juntas do pulso servem para orientar. Como a definição do peso interfere diretamente na correção da posição e da orientação, definiu-se vários modos de se chegar ao valor do peso, ao que se denominou de **estratégias**. Dez estratégias foram definidas e já apresentadas no item 6.2 c).

O teste 3 consiste em posicionar o robô PUMA no ponto indicado na Tabela 6.10, a partir da mesma posição inicial do teste 2. Os resultados estão apresentados nas Figuras 6.22 e 6.23.

Tabela 6.10 - Ponto selecionado para o teste 3

$u_a$	0	1	0
$u_t$	0	0	1
$p_H^G$	-149.09	800	370

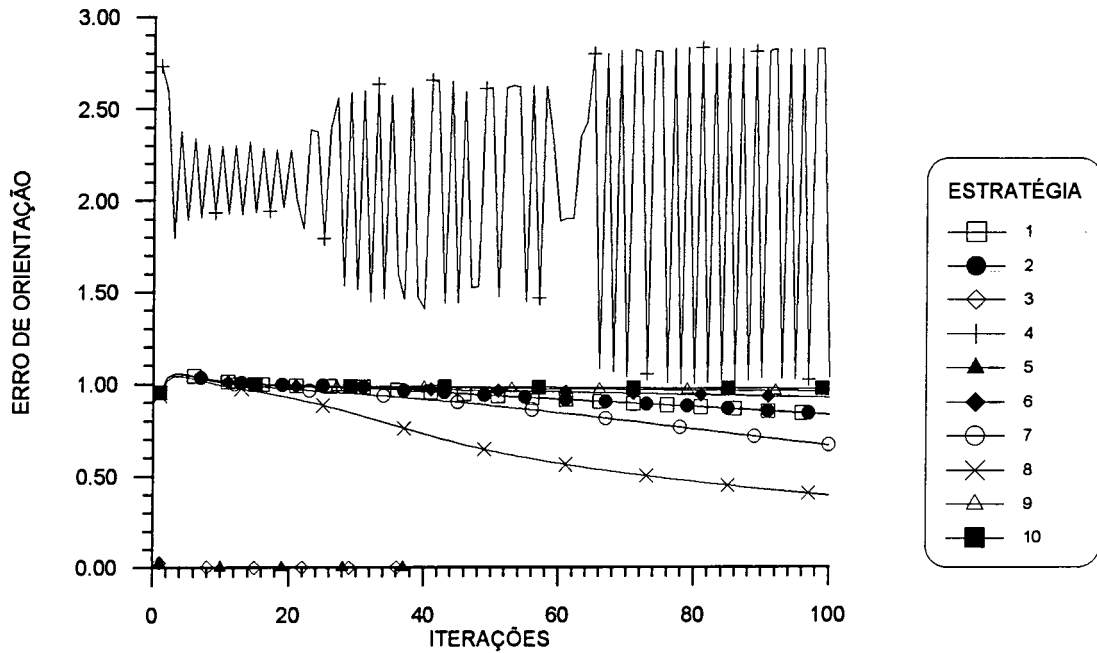


Figura 6.22 - Erro de orientação para as estratégias do teste 3.

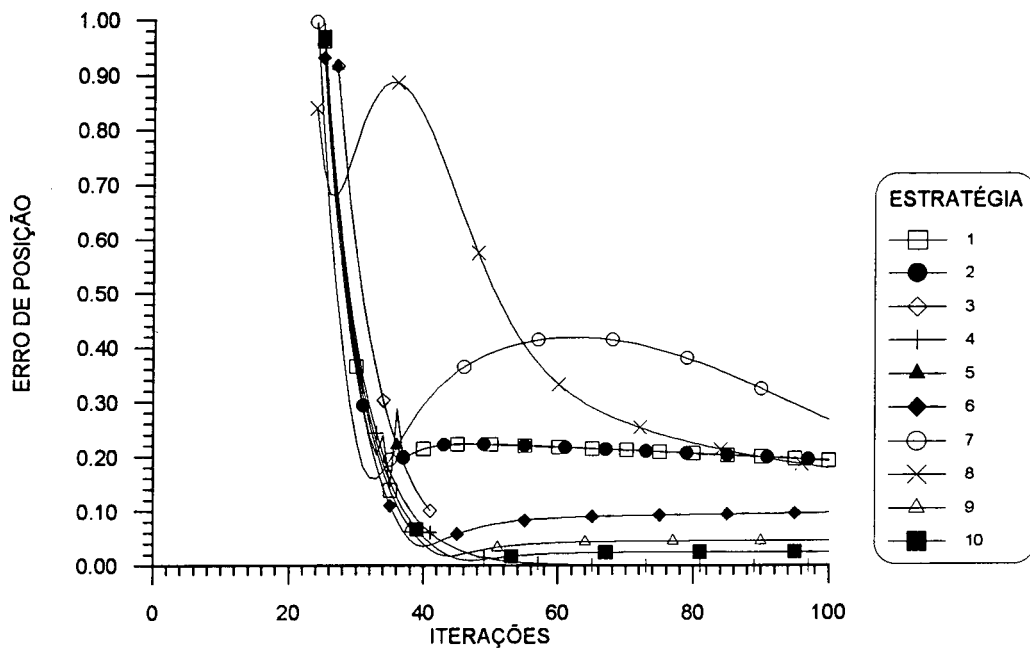


Figura 6.23 - Erro de posição para as estratégias do teste 3.

Observa-se que o erro de posição (Figura 6.22), não é sensivelmente afetado pelas diferentes estratégias. Já no erro de orientação (Figura 6.23), verificam-se sensíveis mudanças. As estratégias 1 e 2, representadas pela linha contínua, chegam a erros praticamente iguais. Isto se explica pela semelhança entre as duas estratégias. Note-se que, para este ponto, não ocorreu a convergência dentro das cem iterações. As estratégias 3 e 5 foram as que mais rapidamente convergiram (41 iterações). Isto se explica por elas aproveitarem completamente o fato das juntas do pulso corrigirem somente orientação. A estratégia 4 não é adequada, pois utiliza todas as juntas para corrigir somente a posição. Contudo ela foi a que mais rapidamente corrigiu o erro de posição (39 iterações). A não convergência se deve à não correção da orientação. Das estratégias 6 a 10, a que melhor se comportou foi a 8, apesar de convergir muito lentamente, ao ponto de não chegar ao resultado esperado dentro das cem iterações. Estas estratégias não são adequadas por não acelerarem a correção do erro mais grosseiro de posição ou de orientação, o que é possível na estratégia 5.

#### d) TESTE 4

Este teste visa confrontar os resultados obtidos pelo cálculo da cinemática inversa pela Posição-Zero, com os resultados da literatura [71]. A tarefa consiste em percorrer, com orientação constante, as bordas de uma chapa quadrada de 500 mm de lado posicionada a 2000 mm da origem do sistema de referência ao longo do sentido positivo do eixo  $x$ ., conforme apresentado na Figura 6.24.

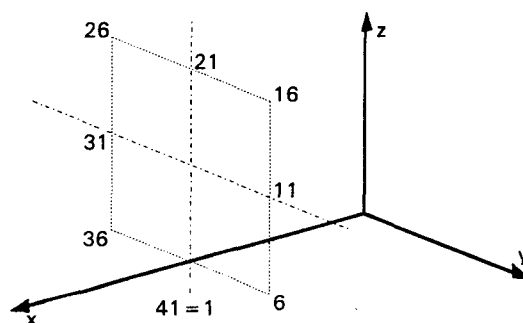


Figura 6.24 - Tarefa do teste 4.

O robô utilizado é o **T3R3 - Cincinati - Milacron**, apresentado na Figura 6.25. A Tabela 6.11 mostra a posição inicial do robô.

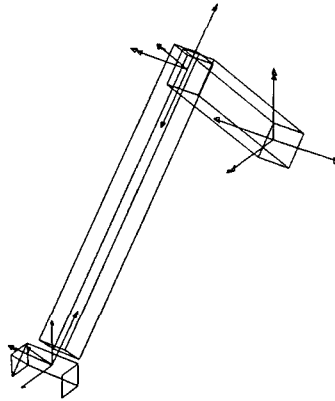


Figura 6.25 - Robô T3R3 - Cincinati - Milacron.

Tabela 6.11 - Coordenadas da posição inicial para o cálculo do percurso do teste 4.

JUNTA	COORDENADA
1	0 <sup>o</sup>
2	45 <sup>o</sup>
3	0 <sup>o</sup>
4	0 <sup>o</sup>
5	45 <sup>o</sup>
6	0 <sup>o</sup>

As Figuras 6.26-(a) e (b) apresentam, respectivamente, os resultados da literatura e os calculados. Pode-se observar, pela semelhança dos gráficos, que o método chega aos mesmos resultados que os apresentados na literatura. A diferença nos valores correspondentes à junta 3, se deve às diferentes referências utilizadas para medir a coordenada da junta 3, defasadas entre si de 270 graus.

#### e) TESTE 5

Este teste avalia o comportamento de diferentes robôs na execução de uma mesma tarefa. O objetivo é mostrar a importância da simulação na avaliação do desempenho cinemático de robôs. O teste consiste de duas etapas. Na primeira etapa os robôs perfazem o percurso numa determinada posição. Na segunda etapa, os robôs



são reposicionados de modo a se tentar perfazer o percurso de uma forma mais eficiente.

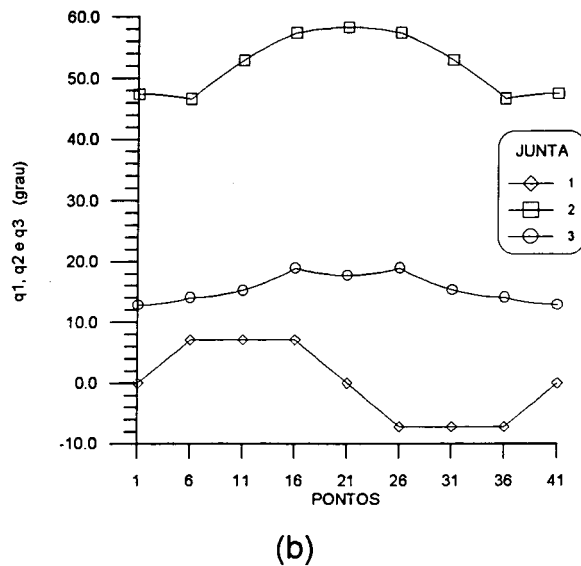
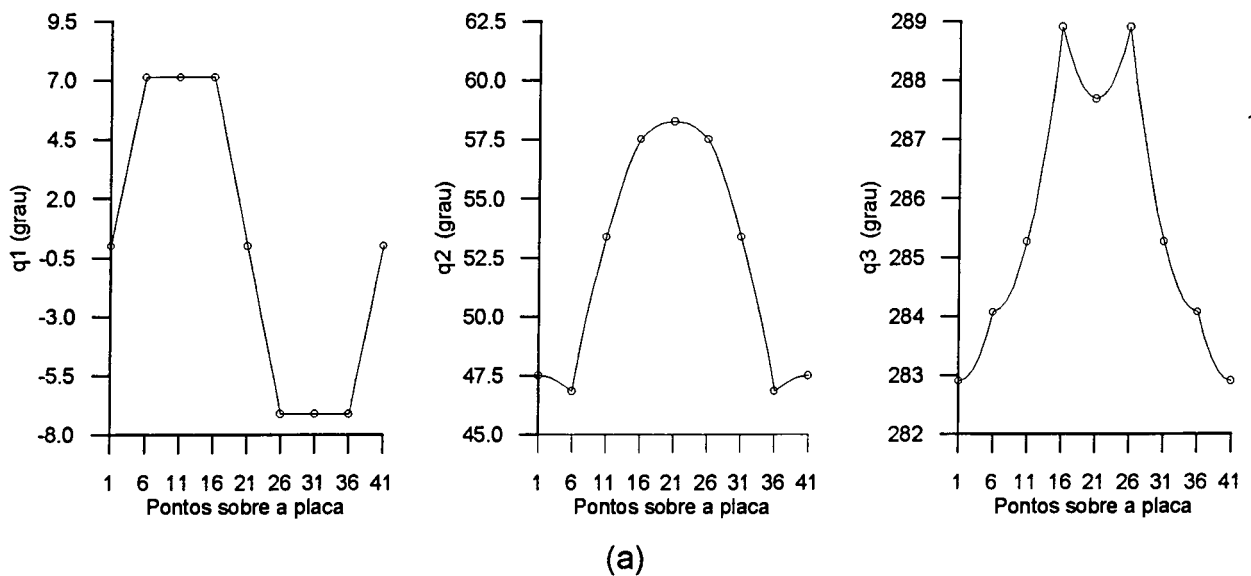


Figura 6.26 - Comparação dos resultados das três primeiras juntas no teste 4.

a) Dados obtidos da referência [71]; b) Dados calculados pelo programa.

A tarefa consiste de dois segmentos, conforme a Figura 6.27. Do ponto inicial, ponto 0, o efetuador desloca-se 500 mm em linha reta, na vertical, girando 90 graus até atingir o ponto 1. Em seguida, desloca-se 300 mm na horizontal, até o ponto 2, com orientação constante.

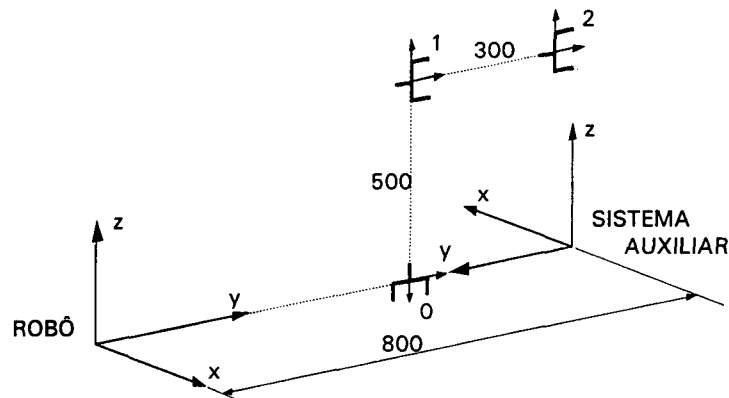


Figura 6.27 - Tarefa para o teste 5.

Os pontos foram definidos num sistema auxiliar para permitir o reposicionamento da tarefa. A Tabela 6.12 apresenta as coordenadas de cada ponto, bem como o posicionamento do sistema auxiliar em relação ao robô.

Tabela 6.12 - Posicionamento da tarefa para o teste 5.

REFERÊNCIA	OBJETO	EIXO Y			EIXO Z			POSIÇÃO		
SIST. INERCIAL	ROBÔ	0	1	0	0	0	1	0	0	0
	SIST. AUXILIAR	0	-1	0	0	0	1	0	800	0
	EFETUADOR	0	1	0	0	0	1	0	0	0
SIST. AUXILIAR	PONTO 0	0	-1	0	0	0	-1	0	300	0
	PONTO 1	0	0	1	0	-1	0	0	300	500
	PONTO 2	0	0	1	0	-1	0	0	0	500
NÚMERO DE SEGMENTOS: 2										
NÚMERO DE PONTOS POR SEGMENTO: 9										

Para a execução da tarefa, foram utilizados os robôs PUMA, ELBOW, STANFORD e CARTESIANO. As posições iniciais para o cálculo da cinemática inversa estão relacionadas na Tabela 6.13. O método utilizado foi o da Posição-Zero, com precisão 0.1 e estratégia 5.

Tabela 6.13 - Posições iniciais para o teste 5.

JUNTA	COORDENADAS			
	PUMA	ELBOW	STANFORD	CARTESIANO
1	90 <sup>0</sup>	90 <sup>0</sup>	90 <sup>0</sup>	1000 mm
2	-90 <sup>0</sup>	120 <sup>0</sup>	90 <sup>0</sup>	800 mm
3	180 <sup>0</sup>	-120 <sup>0</sup>	400 mm	650 mm
4	0 <sup>0</sup>	-90 <sup>0</sup>	90 <sup>0</sup>	0 <sup>0</sup>
5	45 <sup>0</sup>	90 <sup>0</sup>	0 <sup>0</sup>	0 <sup>0</sup>
6	90 <sup>0</sup>	0 <sup>0</sup>	0 <sup>0</sup>	180 <sup>0</sup>

Os resultados do percurso, para cada robô, na primeira etapa, estão apresentados na Figura 6.28.

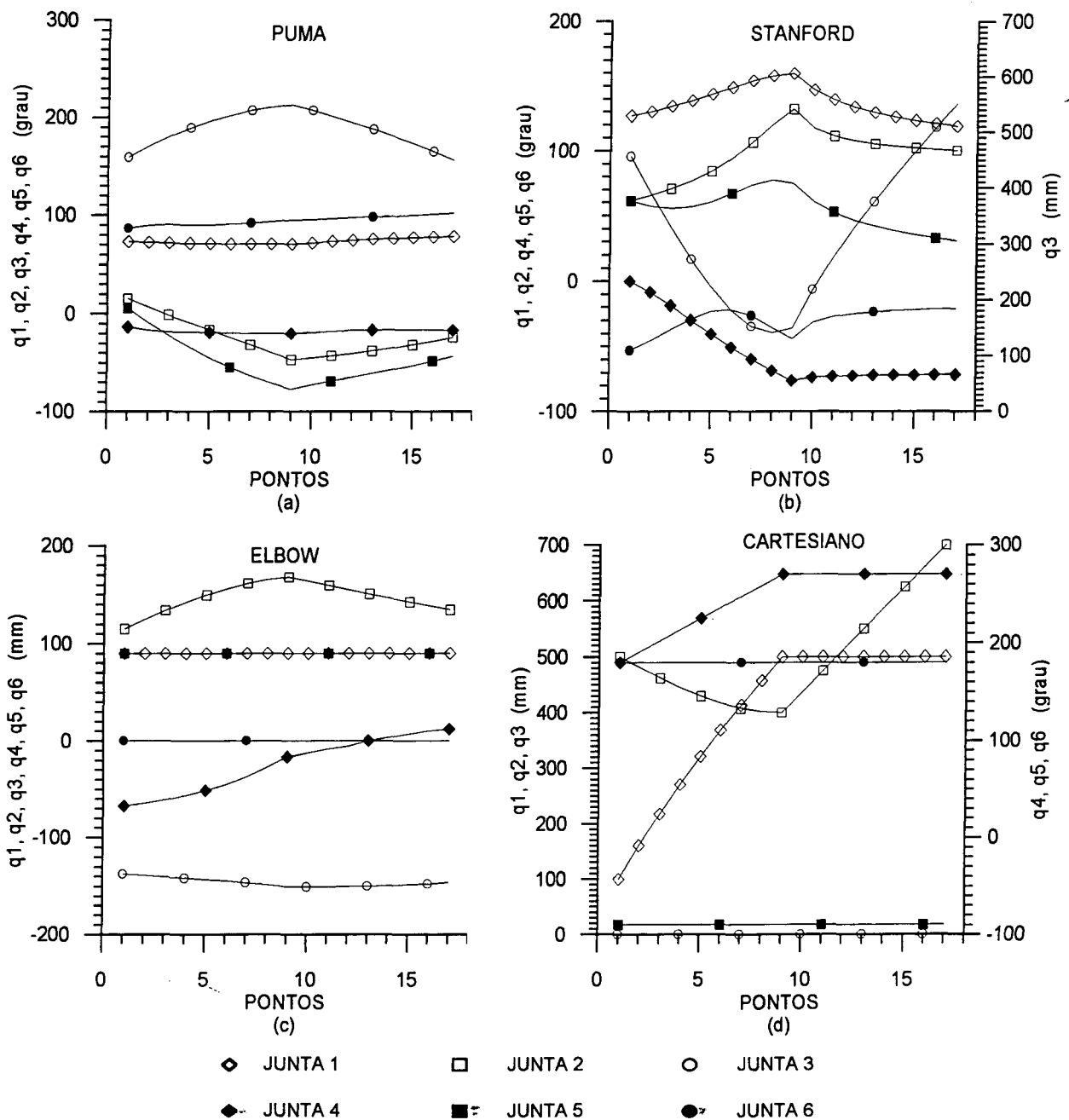


Figura 6.28 - Resultados da primeira etapa do teste 5.

(a) PUMA, (b) STANFORD,  
(c) ELBOW, (d) CARTESIANO.

Observa-se que há uma sensível diferença no comportamento das juntas, o que é natural devido às diferentes estruturas cinemáticas. Vale destacar o comportamento do robô CARTESIANO, na Figura 6.28-(d), cujo desacoplamento das juntas prismáticas faz com que o movimento seja efetuado utilizando-se um menor número de juntas. No segundo segmento do percurso para este robô, somente a junta 2 se movimenta.

A presença do "ombro" no robô PUMA força o movimento da junta 1, o que não ocorre no robô ELBOW, que não possui "ombro". Isto pode ser observado nas curvas correspondentes à junta 1 dos gráficos 6.28-(a) e 6.28-(c).

Quanto maior o comprimento do "ombro", maior é o deslocamento necessário na junta 1. Comparando-se o gráfico do PUMA (a) com o do STANFORD (b), observa-se que, no segundo robô, o deslocamento da primeira junta é maior. Isto se deve ao fato deste robô possuir um "ombro" (300 mm) maior que o do primeiro (149,09 mm).

Na **segunda etapa** do teste, a posição da tarefa é alterada, para os robôs PUMA, ELBOW e STANFORD visando buscar uma posição de trabalho mais adequada. A fim de eliminar a influência do "ombro" no movimento da junta 1 dos robôs PUMA e STANFORD, deslocou-se o sistema auxiliar para uma posição sobre a direção do braço. Para o robô ELBOW, afastou-se o sistema auxiliar, buscando uma posição onde as ligações pudessem trabalhar mais estendidas. A tarefa para o robô CARTESIANO foi deslocada para um ponto qualquer, mantendo a orientação do sistema original.

As novas posições do sistema auxiliar, para a segunda etapa do teste, estão apresentadas na Tabela 6.14. Os resultados obtidos encontram-se na Figura 6.29.

Tabela 6.14 - Posição do sistema auxiliar para a **segunda etapa** do teste 5.

ROBÔ	POSIÇÃO DO SISTEMA AUXILIAR		
	X	Y	Z
PUMA	-149.09	800	350
ELBOW	0	1800	0
STANFORD	300	800	200
CARTESIANO	100	200	150

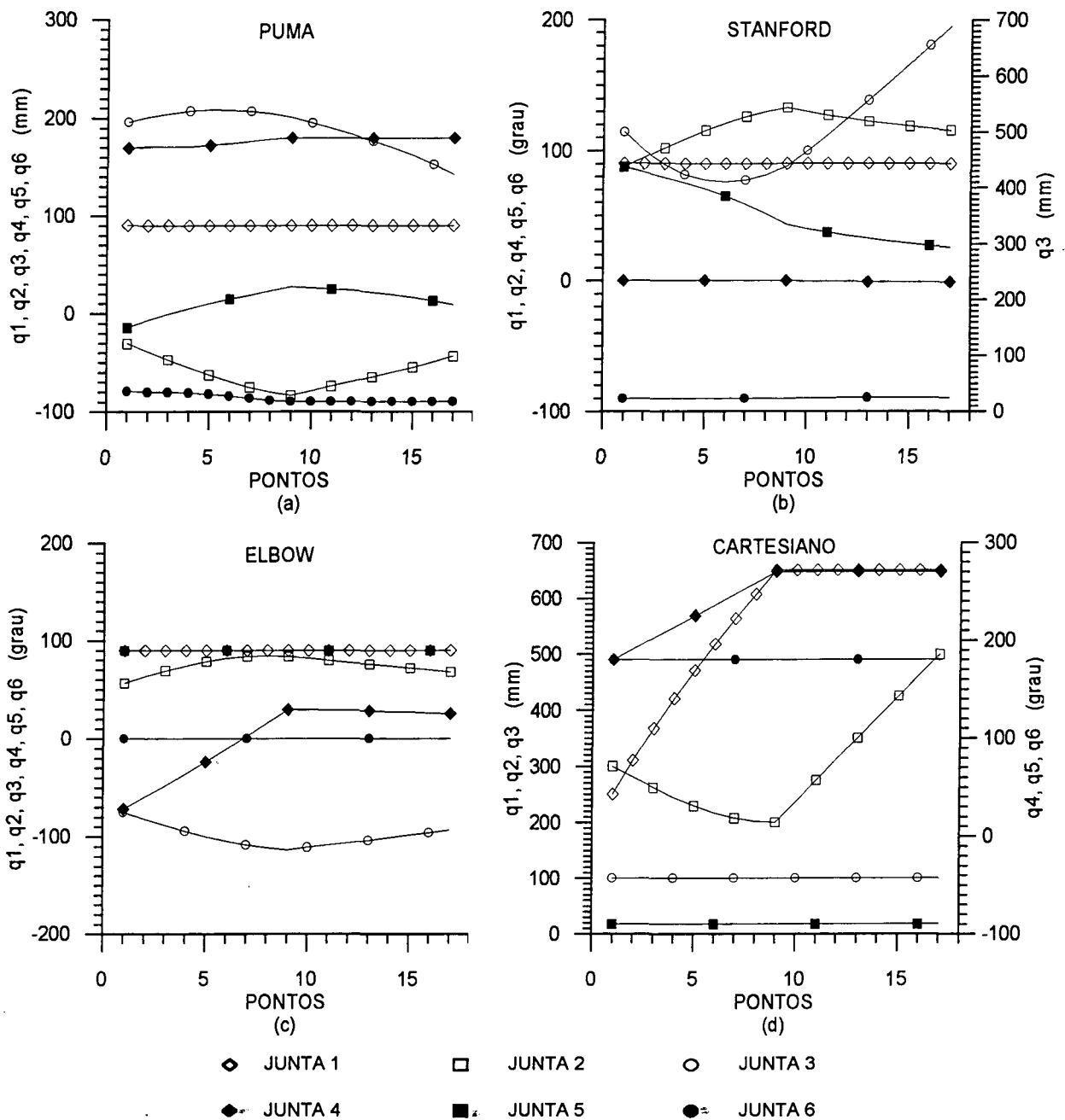


Figura 6.29 - Resultados da **segunda etapa** do teste 5, após o reposicionamento dos robôs. (a) PUMA, (b) STANFORD, (c) ELBOW, (d) CARTESIANO.

Para o robô PUMA, nota-se uma redução nos deslocamentos das juntas, principalmente na junta 1, que permanece imóvel. No robô STANFORD, a redução é ainda maior, comprovada pela não movimentação das juntas 1, 4 e 6, e pela redução dos deslocamentos da junta 3. Para o robô ELBOW, não ocorreram sensíveis alterações. Houve uma redução no deslocamento da junta 2 e um aumento do

deslocamento das juntas 3 e 4. As juntas 1, 5 e 6 permanecem imóveis, como na primeira etapa. Para o robô CARTESIANO, ocorre apenas uma translação na escala das juntas prismáticas, enquanto que, nas rotativas, o comportamento não muda de uma etapa para outra. Isto se explica pelo fato da tarefa ter sido modificada apenas na translação, mantendo-se a mesma orientação.

Embora este teste apresente resultados significativos, vale lembrar que a seleção de um robô ou a otimização de uma tarefa não deve se basear apenas nas características cinemáticas apresentadas. Devem ser avaliados os parâmetros dinâmicos, tais como torques e forças atuantes nas juntas, capacidade de carga, precisão, energia despendida e tempo de execução da tarefa.

#### f) TESTE 6

O Teste 6 compara os métodos da Posição-Zero com o método geométrico iterativo proposto por Pinto e Bevilacqua [72]. A finalidade do teste é verificar o comportamento do método da Posição-Zero em robôs com menos de seis graus de liberdade e robôs redundantes.

A tarefa utiliza os dois manipuladores planares apresentados na figura 6.30. O primeiro possui dois graus de liberdade e o segundo três. Trata-se, portanto, no segundo caso, de um robô **redundante**.

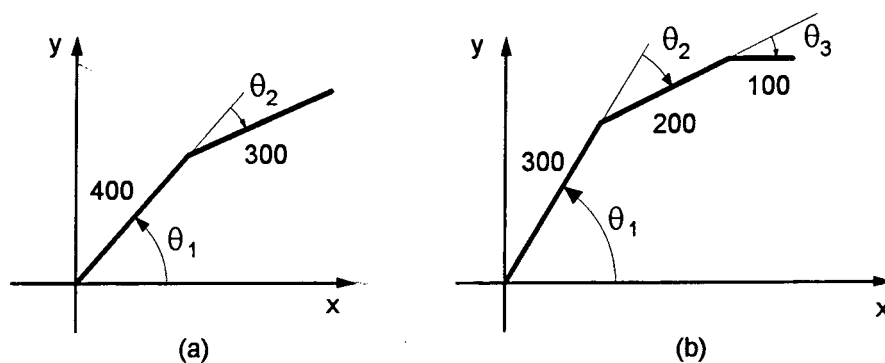


Figura 6.30 - Robôs planares utilizados no Teste 6:

(a) não-redundante (2 g.l.); (b) redundante (3 g.l.).

Cada um dos robôs devem atingir os pontos representados na Tabela 6.15. O número de iterações máximo foi limitado em 100 para os dois métodos.

Tabela 6.15 - Pontos alvo para os robôs no Teste 6.

Ponto	X mm	Y mm
1	-281	582
2	6	482
3	294	382
4	581	282
5	0	800

Como na referência [72] não há indicação da posição inicial, adotou-se os valores  $90^\circ$  e  $-90^\circ$  para o primeiro robô e  $90^\circ$ ,  $-90^\circ$  e  $90^\circ$  para o segundo. Os resultados para os ângulos das juntas, número de iterações e erro de posição estão apresentados nas Tabelas 6.16 e 6.17.

Tabela 6.16 - Resultados do Teste 6 para o robô com 2 graus de liberdade.

Ponto	Método Geométrico				Método da Posição-Zero			
	$\theta_1$ (rad)	$\theta_2$ (rad)	Iterações	Erro	$\theta_1$ (rad)	$\theta_2$ (rad)	Iterações	Erro
1	2,356	-0,79	100	0,835	2,3591	-0,7072	50	0,001
2	2,228	-1,64	10	0,001	2,2280	-1,6444	10	0,001
3	1,584	-1,64	11	0,001	1,5845	-1,6444	10	0,001
4	0,792	-0,80	50	0,001	0,7919	-0,8007	49	0,001
5	1,904	-0,78	100	157,2	1,5708	0,0000	100	100,00

Tabela 6.17 - Resultados do Teste 6 para o robô com 3 graus de liberdade.

Ponto	Método Geométrico					Método da Posição-Zero				
	$\theta_1$ (rad)	$\theta_2$ (rad)	$\theta_3$ (rad)	Iterações	Erro	$\theta_1$ (rad)	$\theta_2$ (rad)	$\theta_3$ (rad)	Iterações	Erro
1	2,315	-0,91	0,602	100	0,004	2,0210	-0,0008	0,0000	100	46,285
2	2,146	-1,25	-1,01	5	0,000	2,0419	-1,4186	1,1512	19	0,001
3	1,583	-1,68	0,124	10	0,000	1,3984	-1,4186	1,1512	19	0,001
4	0,785	-0,88	0,265	100	0,495	0,4523	-0,0009	0,0000	100	45,821
5	1,826	-0,79	0,530	100	140,1	1,5708	0,0000	0,0000	100	200,00

O teste comprova a eficácia do método da Posição-Zero para manipuladores com menos de seis graus de liberdade e redundantes.

Com relação aos resultados para o robô com **dois graus de liberdade**, tem-se algumas observações. Não ocorreu convergência para o ponto 1 no método geométrico devido à limitação das juntas impostas naquele caso. Para o método da

Posição-Zero ocorreu a convergência porque não se implementou nenhum procedimento que levasse em conta os limites das juntas. A semelhança dos métodos neste caso específico está comprovada pelo número de iterações necessários para os pontos 2, 3 e 4. Desconsiderando erros de aproximação, o número de iterações é o mesmo para ambos os métodos. O ponto 5 situa-se fora do espaço de trabalho, razão pela qual nenhum dos métodos obteve solução. Note-se, no entanto que o erro mostra que o robô está com a extensão máxima.

Para o robô com **três graus de liberdade** os resultados diferem por se tratar de um robô redundante. Há infinitas soluções para cada ponto. Nota-se no entanto uma certa proximidade entre as soluções por se tratar de métodos similares. Pode-se atribuir a diferença dos resultados também às posições iniciais adotadas em cada método. Como a referência [72] não apresenta a posição inicial adotada, não se pode comparar com exatidão os resultados. O teste é válido, no entanto para mostrar a viabilidade dos dois métodos no caso de robôs redundantes. Não se obteve a solução para os pontos 1, 4 e 5 por estarem fora do espaço de trabalho do robô.

### **g) COMENTÁRIOS**

Os resultados mostram que os dois métodos testados apresentam vantagens e desvantagens. O método de Newton-Raphson converge rapidamente para uma tolerância apertada quando não se encontra nas proximidades de regimes singulares e quando a posição inicial se encontra afastada da solução. Em se tratando de estabilidade, o método da Posição-Zero apresenta-se de forma satisfatória, mesmo quando se trata de um ponto fora do espaço de trabalho. Por outro lado, este método é mais lento que o primeiro.

O teste 5, apesar de não avaliar diretamente a cinemática inversa, mostra a importância de se utilizar programas simuladores, não só na fase de planejamento de tarefas, mas também em outras áreas da robótica que envolvam projeto e utilização dos robôs.



## **CAPÍTULO 7**

### **CONCLUSÕES E RECOMENDAÇÕES**

#### **7.1 - CONCLUSÕES**

Sobre o estudo dos métodos para a obtenção da cinemática inversa e sobre a proposta de um programa para a simulação cinemática de robôs, tem-se as conclusões apresentadas a seguir.

A respeito do estudo da cinemática inversa, não se pode compreender o seu comportamento sem o conhecimento da cadeia cinemática, do espaço de trabalho correspondente e das regiões singulares nele existentes. Há necessidade, portanto, de se explorar mais estas três áreas, para se ter um melhor controle das soluções.

Os métodos analíticos são os mais adequados em termos de velocidade e das soluções obtidas, mas são limitados a cada tipo de robô. Assim sendo, estes métodos não são adequados para aplicações genéricas, característica desejável nas fases de projeto de novas estruturas cinemáticas.

Com relação às exigências apresentadas para a escolha de um método eficiente para o cálculo da cinemática inversa, o método da análise da Posição-Zero possui a vantagem de ser matematicamente estável, o que não ocorre com o método de Newton-Raphson.

Os testes mostram que, se por um lado o método da Posição-Zero apresenta maior estabilidade, por outro lado, ele é mais lento que o método de Newton-Raphson quando o ponto se encontra numa região distante de singularidades.

Deve-se tomar um certo cuidado na definição da posição inicial para o cálculo da cinemática inversa, para que não ocorra a convergência para uma configuração fisicamente impossível. Um "travamento" ocorre quando o pulso assume posições com singularidades internas, semelhantes à apresentada na Figura 3.5-(b), inviabilizando a convergência.

Um problema encontrado no método da Posição Zero relaciona-se com a definição do erro  $\delta$ , dado pela equação (A4.11). Esta quantidade, representada por um número, não quantifica separadamente os erros de posição e de orientação, tornando-se inadequada quando se deseja controlar o erro de orientação em uma determinada tarefa.

Em função das vantagens e desvantagens apresentadas pelos dois métodos testados, deve-se pensar na implementação de outros métodos. Um que merece atenção é o proposto por Nakamura e Hanafusa [50], apresentado no item 4.4.4, baseado no método de Newton-Raphson, mas que apresenta um comportamento melhor em regiões singulares, apesar de sua complexidade computacional.

Com o que foi implementado no programa, já se pode comprovar a eficácia da simulação como ferramenta de análise e aprendizado. A representação gráfica da cadeia cinemática é essencial para a visualização e validação dos resultados.

A estrutura modular do programa permite alterações e inclusões de forma rápida. Com pequenas modificações pode-se incluir outros métodos de solução da cinemática inversa.

A proposta de geração dos parâmetros cinemáticos, apresentada no Apêndice 1, traz problemas em casos de juntas paralelas. Daí a necessidade, nestes casos, da correção manual. Apesar disso, o procedimento mostra-se satisfatório na maior parte dos casos, garantindo a definição correta dos parâmetros cinemáticos.

A representação gráfica, proposta no Apêndice 2, mostra-se adequada para fins de comparação do comportamento cinemático. Deve-se melhorar, no entanto, a representação da estrutura de modo a se poder identificar os tipos de juntas.

## 7.2 - RECOMENDAÇÕES

Com relação ao estudo da cinemática de robôs, como recomendações para trabalhos futuros, sugere-se o seguinte:

- 1 Aprofundar o estudo do espaço de trabalho, buscando propriedades que permitam a sua representação e o mapeamento das regiões singulares. Tal estudo pode auxiliar no planejamento de trajetórias e no estudo de geometrias otimizadas que aproveitem o efeito das singularidades.
- 2 Aplicar ferramentas matemáticas alternativas, tais como os vetores *screw* e *quaternions*, que demonstraram ser bastante eficientes no estudo da cinemática de robôs.
- 3 Buscar, para o método da Análise da Posição-Zero, uma definição mais clara para a medida do erro  $\delta$ , da equação (A4.11), de modo a permitir a sua utilização em tarefas que exijam controle mais preciso dos erros de orientação e posição.

Com relação ao programa, sugere-se os seguintes melhoramentos:

- 1 Aumentar a velocidade das funções que calculam a cinemática inversa mediante otimização das operações matemáticas nelas existentes. Uma idéia é utilizar *quaternions* ao invés das matrizes de rotação.
- 2 Utilização de ferramentas de CAD que melhorem a modelagem geométrica do robô e que levem em consideração inclusive as propriedades físicas tais como massas, inércias e centro de gravidade. Trajetórias com obstáculos e o comportamento dinâmico podem ser estudados com maior facilidade, além de se ter uma visualização mais realista da estrutura.
- 3 Implementação de outros métodos para o cálculo da cinemática inversa e das opções do programa ainda não implementadas.

- 4 Melhoria da "comunicação" do programa pela utilização de menus mais adequados e de acesso mais rápido.
- 5 Utilizar recursos de programação mais modernos tais como a programação por objeto e ferramentas para facilitar a comunicação com o usuário.

Por fim espera-se que este trabalho tenha contribuído para o crescimento do conhecimento na área da cinemática de robôs tanto na pesquisa quanto no ensino.

## REFERÊNCIA BIBLIOGRÁFICA

- [01] RAO,S.S. & BHATTI,P.K. Optimization In The Design and Control of Robotic Manipulators: A Survey. **Applied Mechanics Review**, v. 42, n. 4, abril 1989.
- [02] FU,K.S.; GONZALEZ,R.C.; LEE,C.S. **Robotics: Control, Sensing, Vision and Intelligence**. New York: McGraw-Hill, 1987.
- [03] CRAIG,J. **Introduction to Robotics: Mechanics and Control**. Massachusetts: Addison-Wesley, 1985.
- [04] RIVIN,E.I. **Mechanical Design of Robots**. New York: McGraw-Hill, 1987.
- [05] MORRIS,H.M. Robot Programming Goes Off-line. **Control Engineering**, setembro 1985.
- [06] DWIVEDI,S.N. Use of Simulation in Planning of Robot Work Cell. **Robotics and Factories of the Future**.
- [07] NOVAK,B. Robotic Simulation Facilities Assembly Design. **Simulation**, dezembro 1984.
- [08] SCHROER,B.J. & TEOH,W. A Graphical Simulation Tool with Off-Line Robot Programming. **Simulation**, agosto 1986.

- [09] ARONSON,R.B. Try Before You Buy Through Simulation. **Machine Design**, setembro 1987.
- [10] FERREIRA,E.P.; ANDRADE FILHO,J.P.; SILVA,J.V.L.; NEVES JR.,O.R. e FERREIRA,A.C. GMSIR - Uma Ferramenta de Simulação para Síntese de Controle e de Arquitetura de Robôs Manipuladores. Centro Tecnológico para Informática - Campinas -S.P.
- [11] DAWSON,G.A. Simulating Manufacturing. **Tooling & Production**, dezembro 1984.
- [12] GOLDEMBERG,A.A. & KELLY,A. A Kinematic and Dynamic Simulator of the IBM-7565 Robot Using AML. **Journal of Robotic Systems**, v. 2, n. 4, p. 353-371, 1985.
- [13] BUCHAL,R.O.; CHERCHAS,D.B.; SASSANI,F.; DUNCAN,J.P. Simulated Off-line Programming of Welding Robots. **International Journal of Robotics Research**, v. 8, n. 3, p. 31-43, junho 1989.
- [14] CALLAN,J.F. The Simulation and Programming of Multiple-Arm Robot Systems. **Robotics Engineering**, p. 26-29, abril 1986.
- [15] MAHAJAN,R. & WALTER,S.E. Computer Aided Automation Planning: Workcell Design and Simulation. **Robotics Engineering**, p.12-15, agosto 1986.
- [16] Four Steps to Simulating Robotic Workcells. **Robotics Engineering**, p. 23-25, agosto 1986.
- [17] CRAIG,J.J. Anatomy of an Off-line Programming System. **Robotics Today**, p 45-47, fevereiro 1985.

- [18] HAFFENDEN,A. Robotics Simulation in Practice. **The Industrial Robot**, p. 250-251, dezembro 1985.
- [19] ROGERS,M.A. Choosing the Right Robot for the Job. **The Industrial Robot**, p. 33-34, março 1986.
- [20] ISENHOUR,T.L.; ECKERT,S.E.; MARSHALL,J.C. Intelligent Robots - The Next Step in Laboratory Automation. **Analytical Chemistry**, v. 61, n. 13, julho 1989.
- [21] DAWSON, G.A. Advanced Manufacturing Technology. **Tooling & Production**, p. 65-69, dezembro 1984.
- [22] PARKER, J.R. A Graphics Based Robot Simulation. **Transactions of the Society for Computer Simulation**, v. 3, n. 2, p. 125-134, abril 1986.
- [23] WORN,H. & STARK,G. Robot Applications Supported by CAD Simulation. **Robotics & Computer Integrated Manufacturing**, v. 3, n.1, p. 55-62, 1987.
- [24] ROSA,E **Sistemas CAE/CAD/CAM**. Florianópolis: UFSC, 1989. Publicação Interna.
- [25] CUGY,A. **Les Robots - Specifications Techniques**. Paris: Hermes, 1983.
- [26] COIFFET,P. **Robot Technology Vol.1: modelling and control**. London: Prentice-Hall, 1983.
- [27] VUKOBRATOVIC,M. & KIRCSNSKI,M. **Scientific Fundamentals of Robotics 3: Kinematics and Trajectory Synthesis of Manipulation Robots**. New York: Springer-Verlag, 1986.

- [28] ASADA,H. & SLOTINE,J.J.E. **Robot Analysis and Control**. New York: John Wiley & Sons, 1986.
- [29] PAUL,R.P. **Robot manipulators: Mathematics, Programming and Control**. Cambridge: MIT, 1981.
- [30] GOLDEMBERG,A.A.; BENHABIB,B.; FENTON,R.G.A Complete Generalized Solution to the Inverse Kinematics of Robots. **IEEE Journal of Robotics and Automation**, v. 1, março 1985.
- [31] SHABANA,A.A. **Dynamics of Multibody Systems**. New York: John Wiley & Sons, 1989.
- [32] KOREN,Y. **Robotics for Engineers**. New York: McGraw-Hill, 1985.
- [33] GROOVER,M.P.; WEISS,M.; NAGEL,R.N.; ODREY,N.G. **Industrial Robotics: Technology, Programming and Applications**. New York: McGraw-Hill, 1987.
- [34] VUKOBRATOVIC,M. & POTKONJAK, V. **Scientific Fundamentals of Robotics 6: Applied Dynamics and CAD of Manipulation Robots**. New York: Springer-Verlag, 1985.
- [35] ANDEEN,G.B. **Robot Design Handbook**. New York: McGraw-Hill, 1988.
- [36] RÁNKY,P.G. & HO,C.Y. **Robot modelling: control and applications with software**. New York: Springer-Verlag, 1985.
- [37] CHEN,F.Y. Gripping Mechanisms for Industrial Robots: an Overview. **Mechanism and Machine Theory**, v. 17, n. 5, p. 299-311, 1982.



- [38] PHAM, D.T. & YEO, S.H. A Knowledge-based System for Robot Gripper Selection: Criteria for Choosing Grippers and Surfaces for Gripping. **International Journal of Machine Tools & Manufacture**, v. 18, n. 4, p. 301-313, 1988.
- [39] HIROSE, S. & UMETANI, Y. The Development of Soft Gripper for the Versatile Robot Hand. **Mechanism and Machine Theory**, v. 13, p. 351-359, 1978.
- [40] LIÉGEOIS, A. **Robot Technology Vol.7: performance and computer-aided design**. London: Prentice-Hall, 1984.
- [41] KUMAR, A. & WALDRON, K.J. The Workspace of a Mechanical Manipulator. **Journal of Mechanical Design**, v. 103, julho 1981.
- [42] GUPTA, K.C. & ROTH, B. Design Considerations for Manipulator Workspace. **Journal of Mechanical Design**, v. 104, outubro 1982.
- [43] RASTEGAR, J. & DERAVIDI, P. Methods to Determine Workspace, Its Subspaces With Different Numbers of Configurations and All the Possible Configurations of a Manipulator. **Mechanism and Machine Theory**, v. 22, n. 4, 1987.
- [44] TSAI, Y.C. & SONI, A.H. The Effect of Link Parameter on the Working Space of General 3R Robot Arms. **Mechanism and Machine Theory**, v. 19, n. 1, p 9-16, 1984.
- [45] HSU, M. & KOHLI, D. Boundary Surfaces and Accessibility Regions for Regional Structures of Manipulators **Mechanism and Machine Theory**, v. 22, n. 3, p. 277-289, 1987.
- [46] TSAI, Y.C. & SONI, A.H. Accessible Region and Synthesis of Robot Arms. **Journal of Mechanical Design**, v. 103, outubro 1981.

- [47] WU,C. A Kinematic CAD Tool for the Design and Control of a Robot Manipulator. **International Journal of Robotics Research**, v. 3, n. 1, 1984.
- [48] LUH,J.Y.S.; WALKER,M.W.; PAUL,R.P.C. On-line Computational Scheme for Mechanical Manipulators. **Journal of Dynamic Systems, Measurement and Control**, v. 102, p. 69-76, junho 1980.
- [49] ANG,M.H. & TOURASSIS,V.D. General-purpose Inverse Kinematics Transformations for Robotic Manipulators. **Journal of Robotic Systems**, v. 4, n. 4, p. 527-549, 1987.
- [50] NAKAMURA,Y. & HANAFUSA,H. Inverse Kinematic Solutions with Singularity Robustness for Robot Manipulator Control. **Journal of Dynamic Systems, Measurement and Control**, v. 108, p. 163-171, setembro 1986.
- [51] KAZEROUNIAN,K. On the Numerical Inverse Kinematics of Robotic Manipulators. **Journal of Mechanisms, Transmissions and Automation in Design**, v. 109, p. 8-13, março 1987.
- [52] GUPTA,K.C. A Note on Position Analysis of Manipulators. **Mechanism and Machine Theory**, v.19 , n.1, p. 5-8, 1984.
- [53] PAUL,R.P.; SHIMANO,B. & MAYER,G.E. Kinematic Control Equations for Simple Manipulators. **IEEE Transactions on Systems, Man and Cybernetics**, v. 11, n. 6, junho 1981.
- [54] HUNT,K.H. The Particular or the General ? (Some Examples from Robot Kinematics. **Mechanism and Machine Theory**, v. 21, n. 6, p. 481-487, 1986.

- [55] RAMOS, J.J.G. **Geração de Trajetórias Contínuas para Robôs: Aspectos Cinemáticos e Computacionais**. Dissertação (Mestrado em Engenharia Mecânica) - UNICAMP, 1986
- [56] CARVALHO, J.C.M.; STEFFAN Jr., V.; LOPRE NETO, E.F. Modelo Geométrico Inverso de um Robô Manipulador. **IX Congresso Brasileiro de Engenharia Mecânica**, Florianópolis, 1987.
- [57] PAUL, R.P.; SHIMANO, B.; MAYER, G.E. Differential Kinematic Control Equations for Simple Manipulators. **IEEE Transactions on Systems, Man and Cybernetics**, v. 11, n. 6, junho 1981.
- [58] HUNT, K.H. Robot Kinematics - A Compact Analytic Inverse Solution for Velocities. **Journal of Mechanisms, Transmissions and Automation in Design**, v. 109, p. 42-49, março 1987.
- [59] CHENG, P.Y.; CHEN, C.K.; WENG, C.I. Application of Symbolic Representation Method to the Inverse Kinematics Problem of Robot Manipulators. **Mechanism and Machine Theory**, v. 24, n. 1, p. 61-72, 1989.
- [60] WHITNEY, D.E. Optimum Step Size Control for Newton-Raphson Solution of Nonlinear Vector Equations. **IEEE Transactions on Automatic Control**, v. 14, n. 5, p. 572-574, 1969.
- [61] TUCKER, M. & PEREIRA, N.D. Generalized Inverses for Robotics Manipulators. **Mechanism and Machine Theory**, v. 22, n. 6, p. 507-514, 1987.
- [62] LIU, T.S. & TSAY, S.Y. Singularity of Robotic Kinematics: A Differential Motion Approach. **Mechanism and Machine Theory**, v. 25, n. 4, p. 439-444, 1990.

- [63] GUPTA, K.C. & KAZEROUNIAN, S.M.K. An Investigation of the Time Efficiency and Near Singular Behavior of Numerical Robot Kinematics. **Mechanism and Machine Theory**, v. 22, n. 4, p. 371-381, 1987.
- [64] TSAI, Y.T. & ORIN, D.E. A Strictly Convergent Real-Time Solution for Inverse Kinematics of Robot Manipulators. **Journal of Robotics Systems**, v. 4, n. 4, p. 477-501, 1987.
- [65] LUMELSKY, V.J. Iterative Coordinate Transformation Procedure for one class of robots. **IEEE Transactions on Systems, Man and Cybernetics**, v. 14, n. 3, p. 500-505, 1984.
- [66] BENATI, M.; MORASSO, P.; TAGLIASCO, V. The Inverse Kinematic Problem for Anthropomorphic Manipulator Arms. **Journal of Dynamic Systems, Measurement and Control**, v. 104, p. 110-113, março 1982.
- [67] ROSA, E.; BACK, N.; ERTHAL, J.L. Planejamento de Trajetória de Manipuladores Usando Polinômios Cúbicos de Hermite. **X Congresso Brasileiro de Engenharia Mecânica**, Rio de Janeiro, 1989.
- [68] PAUL, R. Manipulator Cartesian Path Control. **IEEE Transactions on Systems, Man and Cybernetics**, v. 9, n. 11, novembro 1979.
- [69] BRONSTEIN, I. & SEMENDISEV, K. **Manual de Matemática para Engenheiros e Estudantes**. Moscou: MIR, 1979.
- [70] GIOVANONNI, C.F.; ERTHAL, J.L.; FORCELLINI, F.A.; BACK, N. Análise Cinemática de Manipuladores Industriais Utilizando o Método de Newton-Raphson. **II Congresso de Engenharia Mecânica Norte-Nordeste**, João Pessoa, 1992.

- [71] MAHALINGAN, S & SHAVAN, A.N. The Nonlinear Displacement Analysis of Robotic Manipulators Using the Complex Optimization Method. **Mechanism and Machine Theory**, v. 22, n. 1, p. 89-95, 1987.
- [72] PINTO, F.A.N.C. & BEVILACQUA, L.. Método Geométrico Iterativo para Solução da Cinemática Inversa de Manipuladores Mecânicos. **X Congresso Brasileiro de Engenharia Mecânica**, Rio de Janeiro, 1989.

## APÊNDICE 1

### CÁLCULO DOS PARÂMETROS CINEMÁTICOS

#### A1.1 - INTRODUÇÃO

Na análise de cadeias cinemáticas espaciais com elevado número de juntas, a modelagem, utilizando parâmetros de Denavit-Hartenberg ou vetores da Posição-Zero, pode se tornar complexa e passível de erros, invalidando assim o modelo. Além disso, quando se trata do projeto de um novo manipulador, onde se deseja estudar várias possibilidades de cadeias cinemáticas, tal tarefa pode tornar-se enfadonha.

Neste Apêndice é apresentado um método para a obtenção automática dos parâmetros de Denavit-Hartenberg e dos vetores da Posição-Zero, para um robô qualquer. Três casos são apresentados:

- - Caso 1 - Conhece-se apenas a posição e a orientação das juntas e do efetuador;
- - Caso 2 - Conhece-se apenas os vetores da Posição-Zero;
- - Caso 3 - Conhece-se apenas os parâmetros de Denavit-Hartenberg.

Nos três casos são obtidos os dois conjuntos de parâmetros cinemáticos: Denavit-Hartenberg e Posição-Zero. Consegue-se assim, uma grande agilização na modelagem cinemática de robôs, principalmente na fase de projeto.

## A1.2 - CÁLCULO DOS PARÂMETROS A PARTIR DO POSICIONAMENTO DAS JUNTAS E DO EFETUADOR.

Neste primeiro caso, são conhecidos apenas os vetores que representam a posição e a orientação das juntas e do efetuador. São calculados, portanto, os dois conjuntos de parâmetros.

O método compõe-se de quatro etapas:

- cálculo das origens dos sistemas de coordenadas de cada ligação,
- cálculo dos vetores de corpo da Posição-Zero,
- cálculo dos eixos dos sistemas de coordenadas das ligações e
- cálculo dos parâmetros de Denavit-Hartenberg.

Com os dados de posição e orientação das juntas e do efetuador, calcula-se as origens dos sistemas de coordenadas. A partir das origens, calcula-se os vetores de corpo, ficando assim definidos todos os vetores da Posição-Zero. Tendo a posição das origens e a orientação das juntas, calcula-se os eixos dos sistemas das ligações e, finalmente, com base nestes sistemas, calcula-se os parâmetros de Denavit-Hartenberg.

### a) Cálculo das origens dos sistemas de coordenadas.

Como a determinação das origens é feita com o auxílio da normal comum aos eixos de duas juntas sucessivas, o problema pode ser tratado como o de interseção de retas. Neste caso será utilizada a representação de retas na forma vetorial [69]:

$$\mathbf{P} = \mathbf{P}_0 + pu \quad (\text{A1.1})$$

onde  $\mathbf{P}_0$  é um ponto conhecido da reta,  $\mathbf{P}$  é um ponto qualquer sobre a reta, distante  $p$  unidades de  $\mathbf{P}_0$ , e  $\mathbf{u}$  é um vetor unitário na direção da reta.

Dois casos de interseção, apresentados na Figura A1.1, são utilizados:

- Caso 1 - Ponto de interseção entre a reta normal comum a duas retas e uma das retas:

$$\mathbf{Y} = \mathbf{P}_2 + \mathbf{u}_2 \left\{ U \left[ \mathbf{u}_1 \cdot (\Delta \mathbf{P} - \{ \mathbf{u}_2 \cdot \Delta \mathbf{P} \} \mathbf{u}_2) \right] \frac{1}{1 - U^2} - \mathbf{u}_2 \cdot \Delta \mathbf{P} \right\} \quad (\text{A1.2})$$

onde:

$$U = \mathbf{u}_1 \cdot \mathbf{u}_2$$

e

$$\Delta \mathbf{P} = \mathbf{P}_2 - \mathbf{P}_1$$

Nas expressões acima,  $\mathbf{Y}$  é o ponto de interseção,  $\mathbf{P}_1$  e  $\mathbf{u}_1$  representam a reta 1,  $\mathbf{P}_2$  e  $\mathbf{u}_2$  a reta 2 e “ $\cdot$ ” representa o produto escalar. Todos os vetores são representados num único sistema de referência.

- Caso 2 - Ponto de interseção de uma reta com a normal a ela que passa por um ponto  $\mathbf{Q}$ :

$$\mathbf{Y} = \mathbf{P}_1 + \{ (\mathbf{Q} - \mathbf{P}_1) \cdot \mathbf{u}_1 \} \mathbf{u}_1 \quad (\text{A1.3})$$

onde  $\mathbf{Y}$  é o ponto de interseção,  $\mathbf{P}_1$  e  $\mathbf{u}_1$  representam a reta 1 e  $\mathbf{Q}$  representa um ponto por onde passa a normal.

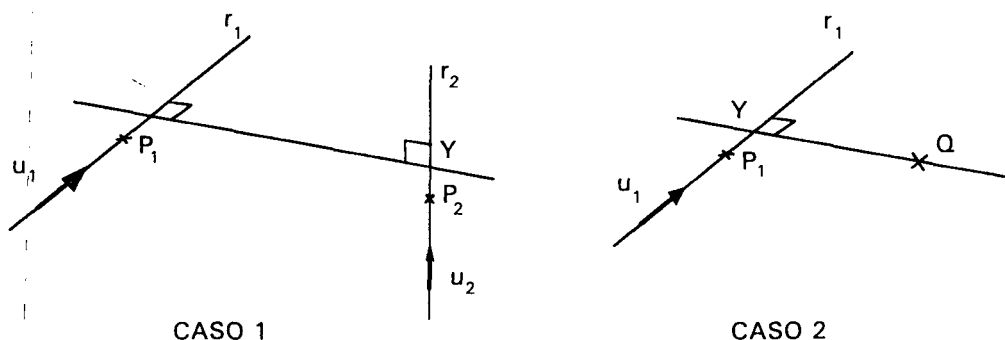


Figura A1.1: Casos de interseção de retas.

Para o cálculo da origem dos sistemas de coordenadas (ponto  $\mathbf{Y}$ ) a escolha da equação depende do tipo e da posição relativa das juntas.



Considerando os dois tipos de juntas (rotativa e prismática) e as posições relativas (reversa e paralela) entre juntas vizinhas, tem-se trinta e duas combinações possíveis para cada três juntas sucessivas. Destas, oito são descartadas por serem redundantes (juntas prismáticas paralelas). As restantes são agrupadas em sete casos distintos. A Tabela A1.1 apresenta as trinta e duas combinações possíveis.

Tabela A1.1 - Combinações possíveis de três juntas sucessivas.

POSICÃO	CASO	$u_{i-1}$	$P_{i-1}$	$u_i$	$P_i$	$u_{i+1}$	$P_{i+1}$	$O_{i-2}$	$O_i$
1	Rc R Rc	A	x	x	x	x			
2	Rc R Rp	A	x	x	x	x			
3	Rc R Pc	A	x	x	x	x			
4	Rc R Pp	A	x	x	x	x			
5	Rp R Rc	B			x	x	x	x	
6	Rp R Rp	C			x	x			x
7	Rp R Pc	C			x	x			x
8	Rp R Pp	C			x	x			x
9	Pc R Rc	A	x	x	x	x			
10	Pc R Rp	A	x	x	x	x			
11	Pc R Pc	A	x	x	x	x			
12	Pc R Pp	A	x	x	x	x			
13	Pp R Rc	B			x	x	x	x	
14	Pp R Rp	C			x	x			x
15	Pp R Pc	C			x	x			x
16	Pp R Pp	*							
17	Rc P Rc	D	x	x	x				x
18	Rc P Rp	E	x	x	x		x		
19	Rc P Pc	D	x	x	x				x
20	Rc P Pp	*							
21	Rp P Rc	F			x			x	x
22	Rp P Rp	G			x		x	x	
23	Rp P Pc	F			x			x	x
24	Rp P Pp	*							
25	Pc P Rc	D	x	x	x				x
26	Pc P Rp	E	x	x	x		x		
27	Pc P Pc	D	x	x	x				x
28	Pc P Pp	*							
29	Pp P Rc	*							
30	Pp P Rp	*							
31	Pp P Pc	*							
32	Pp P Pp	*							

R - JUNTA ROTATIVA  
P - JUNTA PRISMÁTICA  
\* - REDUNDÂNCIA

c - JUNTA CONCORRENTE OU REVERSA  
p - JUNTA PARALELA

Para o cálculo da origem  $O_{i-1}$  através das equações (A1.2) e (A1.3), são necessários os vetores de orientação e de posição das juntas. Tais vetores também estão indicados na Tabela A1.1. As situações semelhantes foram classificadas em sete casos diferentes (A a G), resumidos na Tabela A1.2. Nela também está indicada a equação necessária para cada caso. O ponto Y corresponde sempre à origem  $O_{i-1}$ .

Tabela A1.2 - Equivalência das grandezas envolvidas no cálculo das origens com as grandezas das equações (A1.2) e (A1.3).

CASO	EQUAÇÃO	$P_1$	$u_1$	$P_2$	$u_2$	Q
A	A1.2	$P_{i-1}$	$u_{i-1}$	$P_i$	$u_i$	
B	A1.2	$P_{i+1}$	$u_{i+1}$	$P_i$	$u_i$	
C	A1.3	$P_i$	$u_i$			$O_i$
D	A1.2	$P_{i-1}$	$u_{i-1}$	$O_i$	$u_i$	
E	A1.2	$P_{i-1}$	$u_{i-1}$	$P_{i+1}$	$u_{i+1}$	
F	A1.3	$O_i$	$u_i$			$O_{i-2}$
G	A1.3	$P_{i+1}$	$u_{i+1}$			$O_{i+2}$

As origens dos sistemas da base do robô ( $O_0$ ) e do efetuador ( $O_N$ ), são definidas no início.

A Figura A1.2 apresenta o algoritmo para o cálculo das origens dos sistemas de coordenadas para um robô com  $N$  graus de liberdade.

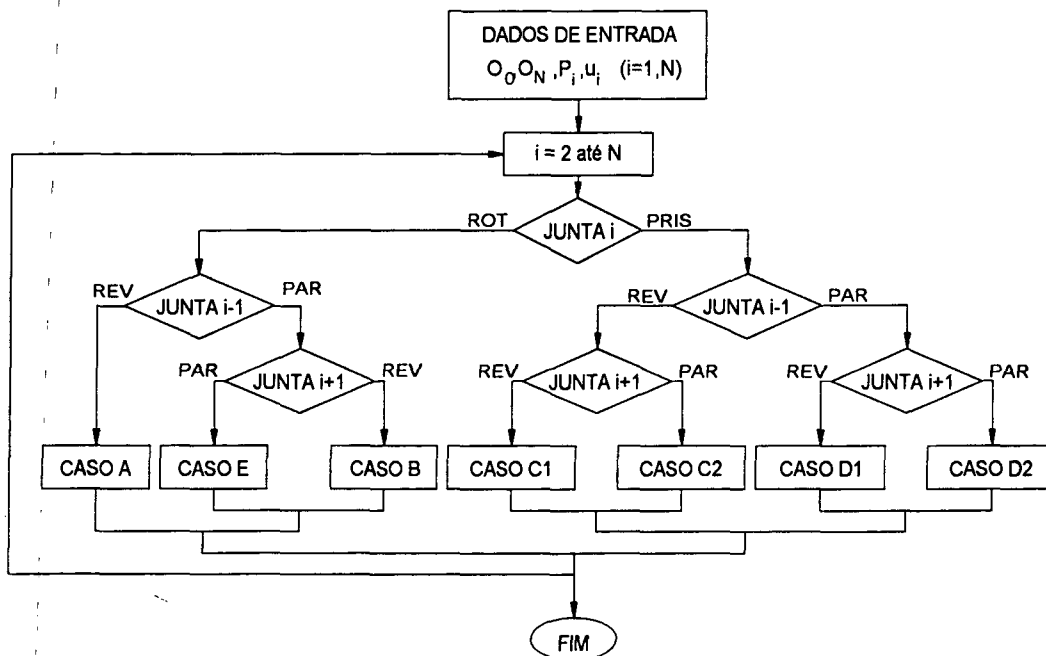


Figura A1.2 - Cálculo das origens dos sistemas de coordenadas das ligações.

**b) Cálculo dos vetores de corpo da Posição-Zero.**

Os vetores de corpo  $b_i$ , que representam o comprimento da ligação  $i$ , são obtidos pela diferença entre as posições das origens, conforme a Figura A1.3.

$$\mathbf{b}_{i+1} = \mathbf{O}_i - \mathbf{O}_{i-1} \quad (A1.4)$$

com  $i$  variando de 1 a  $N$ .

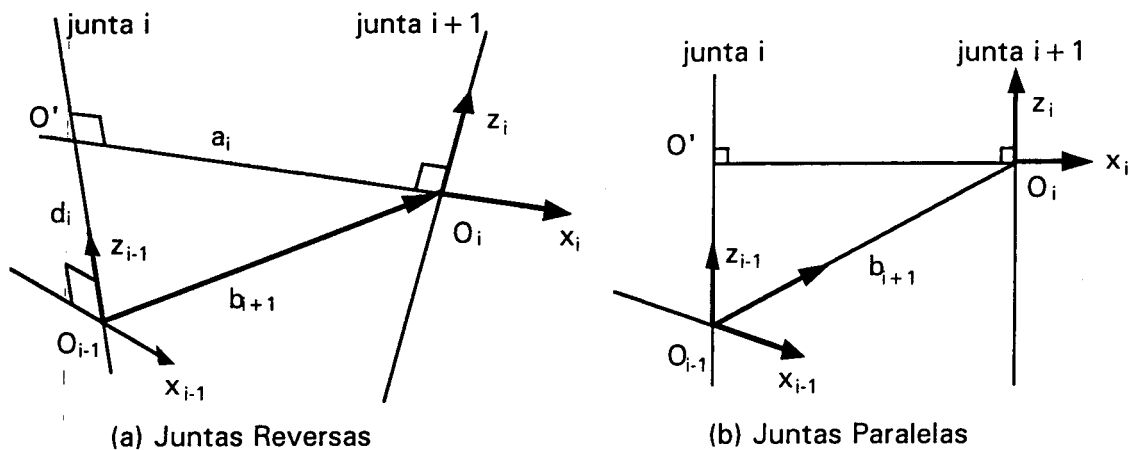


Figura A1.3 - Vetores de corpo e eixos dos sistemas de coordenadas para juntas reversas (a) e paralelas (b).

### c) Cálculo dos eixos dos sistemas de coordenadas das ligações

Os sistemas de coordenadas das ligações obedecem a convenção de Denavit-Hartenberg [02]. O algoritmo para o cálculo dos eixos é mostrado na Figura A1.4.

O eixo  $z_{i-1}$  coincide com a direção da junta  $i$ , ou seja

$$\mathbf{z}_{i-1} = \mathbf{u}_i \quad (A1.5)$$

e o eixo  $z_N$  coincide com a direção  $\mathbf{u}_a$  do efetuator.

O eixo  $x_i$  é obtido de dois modos diferentes, dependendo das juntas consecutivas serem concorrentes ou paralelas.

Se as juntas  $i$  e  $i+1$  forem **concorrentes** ou **reversas**,  $x_i$  é o produto vetorial de  $z_{i-1}$  com  $z_i$ . (Figura A1.3-a).

$$x_i = \pm \frac{z_{i-1} \times z_i}{|z_{i-1} \times z_i|} \tag{A1.6}$$

onde o sinal (+1 ou -1) deve ser escolhido. O operador "x" representa o produto vetorial.

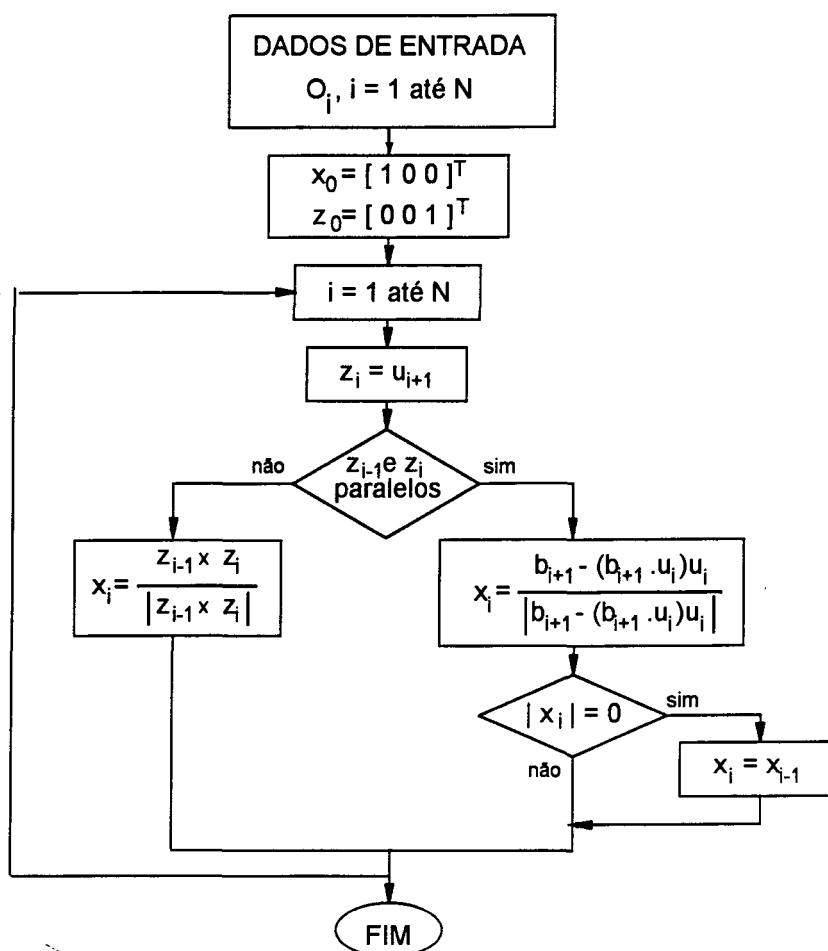


Figura A1.4 - Cálculo dos eixos x e z dos sistemas das juntas

No caso das juntas i e i+1 serem **paralelas**, x<sub>i</sub> é obtido pelo vetor diferença entre b<sub>i+1</sub> e o vetor O<sub>i-1</sub>O' normalizado (Figura A1.3-b):

$$O_{i-1}O' = (b_{i-1} \cdot u_i)u_i \tag{A1.7}$$

$$O'O_i = b_{i+1} - O_{i-1}O' \tag{A1.8}$$

$$\mathbf{x}_i = \frac{\mathbf{O}'\mathbf{O}_i}{|\mathbf{O}'\mathbf{O}_i|} \quad (\text{A1.9})$$

Se  $\mathbf{z}_{i-1}$  for **coincidente** com  $\mathbf{z}_i$ ,  $\mathbf{x}_i$  fica indeterminado. Adota-se, então  $\mathbf{x}_i$  paralelo a  $\mathbf{x}_{i-1}$ .

O eixo  $\mathbf{y}_i$  é obtido formando com  $\mathbf{x}_i$  e  $\mathbf{z}_i$  um sistema destrógiro.

#### d) Cálculo dos parâmetros de Denavit-Hartenberg.

O comprimento  $a_i$  da ligação  $i$  é a distância entre  $\mathbf{z}_i$  e  $\mathbf{z}_{i-1}$  medida sobre  $\mathbf{x}_i$  (Figura A1.3-a). Ele pode ser calculado projetando-se o vetor  $\mathbf{b}_{i+1}$  sobre  $\mathbf{x}_i$ .

$$a_i = \mathbf{b}_{i+1} \cdot \mathbf{x}_i \quad (\text{A1.10})$$

O deslocamento  $d_i$  é obtido pela projeção de  $\mathbf{b}_{i+1}$  sobre o eixo da junta  $i$ .

$$d_i = \mathbf{b}_{i+1} \cdot \mathbf{z}_i \quad (\text{A1.11})$$

O cálculo dos ângulos  $\alpha_i$  e  $\theta_i$  é feito pelo produto escalar entre dois vetores unitários e obedece a convenção representada na Figura A1.5.

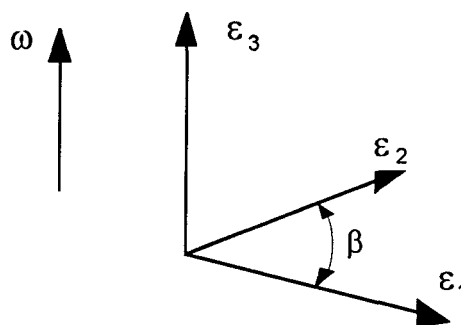


Figura A1.5 - Ângulo entre dois vetores.

O ângulo  $\beta$  entre dois vetores  $\varepsilon_1$  e  $\varepsilon_2$  unitários vale:

$$\beta = \arccos(\varepsilon_1 \cdot \varepsilon_2) \quad (\text{A1.12})$$

Um vetor unitário  $\omega$ , perpendicular a  $\varepsilon_1$  e a  $\varepsilon_2$ , indica o sentido positivo de  $\beta$  segundo a regra da mão direita. Como  $\beta$  é medido de  $\varepsilon_1$  para  $\varepsilon_2$ , o seu sinal pode ser obtido pela comparação do vetor  $\varepsilon_3$ , dado por

$$\varepsilon_3 = \frac{\varepsilon_1 \times \varepsilon_2}{|\varepsilon_1 \times \varepsilon_2|} \quad (\text{A1.13})$$

com o vetor unitário  $\omega$ . Se  $\varepsilon_3 \cdot \omega > 0$ , o sinal de  $\beta$  permanece. Caso contrário, inverte-se o sinal.

Utilizando-se este procedimento, o ângulo  $\alpha_i$  pode ser calculado da seguinte forma:

$$\alpha = \arccos(\mathbf{z}_{i-1} \cdot \mathbf{z}_i) \quad (\text{A1.14})$$

$$\varepsilon_3 = \frac{\mathbf{z}_{i-1} \times \mathbf{z}_i}{|\mathbf{z}_{i-1} \times \mathbf{z}_i|} \quad (\text{A1.15})$$

Se  $\varepsilon_3 \cdot \mathbf{x}_i < 0$  então  $\alpha_i = -\alpha$ .

Similarmente, para o ângulo  $\theta_i$ :

$$\theta_i = \arccos(\mathbf{x}_{i-1} \cdot \mathbf{x}_i) \quad (\text{A1.16})$$

$$\varepsilon_3 = \frac{\mathbf{x}_{i-1} \times \mathbf{x}_i}{|\mathbf{x}_{i-1} \times \mathbf{x}_i|} \quad (\text{A1.17})$$

Se  $\varepsilon_3 \cdot \mathbf{z}_{i-1} < 0$  então  $\theta_i = -\theta$ .

A Figura A1.6 apresenta o algoritmo para o cálculo dos parâmetros de Denavit-Hartenberg.

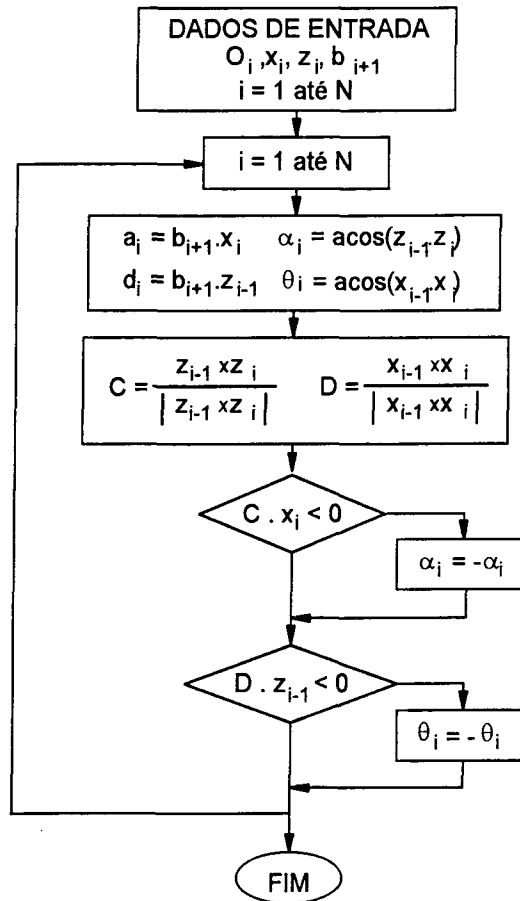


Figura A1.6 - Cálculo dos parâmetros de Denavit-Hartenberg.

### A1.3 - CÁLCULO DOS PARÂMETROS DE DENAVIT-HARTENBERG A PARTIR DOS VETORES DA POSIÇÃO-ZERO.

Quando são fornecidos os vetores da Posição-Zero, transforma-se os vetores de corpo,  $b_i$ , em vetores posição das juntas, fazendo:

$$p_1 = b \tag{A1.18}$$

$$p_{i+1} = p_i + b_{i+1} \tag{A1.19}$$

para  $i$  variando de 1 até o número de graus de liberdade.

Calculadas as posições das juntas, recai-se no mesmo procedimento do item anterior, onde são conhecidas as posições e orientações das juntas e do efetuador.

O procedimento, implementado na função `calcula_DH_dePZ()`, obedece a sequência esquematizada na Figura A1.7.

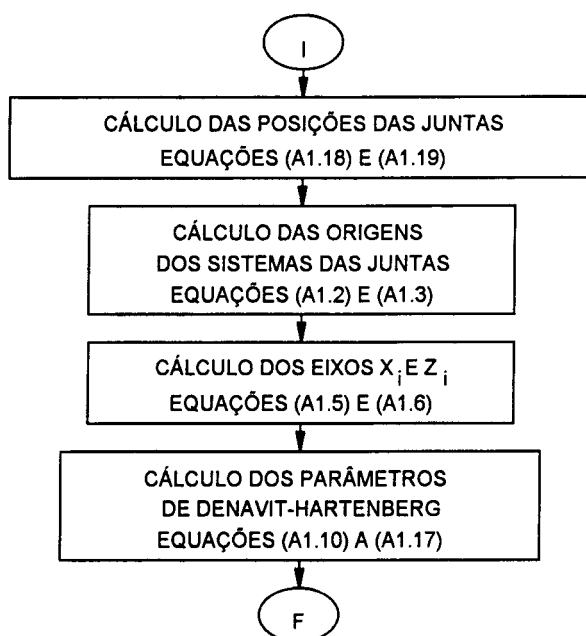


Figura A1.7 - Cálculo dos parâmetros de Denavit-Hartenberg a partir dos vetores da Posição-Zero.

#### A1.4 - CÁLCULO DOS VETORES DA POSIÇÃO-ZERO A PARTIR DOS PARÂMETROS DE DENAVIT-HARTENBERG.

Se são conhecidos os parâmetros de Denavit-Hartenberg, pode-se obter um conjunto de vetores da Posição-Zero, utilizando-se do método recursivo proposto por Ang e Tourassis [49]. Este método, apresentado no Capítulo 3, utiliza as equações (3.18) a (3.21).



Os vetores de corpo  $b_i$  podem ser obtidos pela diferença entre as posições das origens, conforme mostra a Figura A1.3.

Levando-se em conta que não há informação nenhuma sobre as posições das juntas e que os vetores de corpo da Posição-Zero não são únicos, pode-se defini-los como sendo a diferença entre as origens dos sistemas de coordenadas das ligações vizinhas. É importante ressaltar que os vetores de corpo calculados deste modo não coincidem com os apresentados por Kazerounian [51].

A função `calcula_PZ_de_DH()` executa os cálculos, resumidos no fluxograma da Figura A1.8.

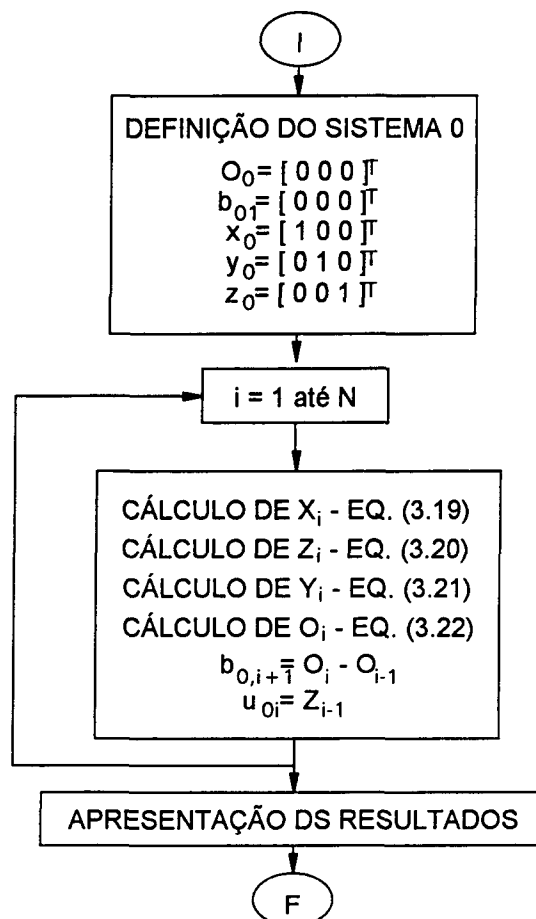


Figura A1.8 - Cálculo dos vetores da Posição-Zero a partir dos parâmetros de Denavit-Hartenberg.

### A1.5 - COMENTÁRIOS SOBRE OS PROCEDIMENTOS.

A finalidade principal destes procedimentos é a de se ter os dois conjuntos de parâmetros cinemáticos, independentemente do que se conhece sobre a cadeia.

Um problema encontrado foi a definição correta das orientações dos eixos  $x_i$ , dos sistemas intermediários. Calcula-se apenas a direção dos eixos, não existindo nenhuma regra para a definição do sentido. Optou-se pelo sentido positivo, apesar de que, em alguns casos, ele não seja adequado por fornecer uma configuração fisicamente impraticável, quando as juntas são zeradas. O problema foi contornado procedendo-se a correção manual da orientação, após o seu cálculo.

A posição das origens nos casos de eixos paralelos também é problemática, já que qualquer posição sobre o eixo pode ser escolhida. Um caso típico é o do robô SCARA, que possui todas as juntas paralelas. Para este robô, todas as origens intermediárias devem ser reposicionadas manualmente devido a este problema.

Outro cuidado que se deve ter é com relação à definição dos vetores de corpo  $b_i$ . Mesmo que eles sejam conhecidos, é preciso transformá-los em vetores de corpo relacionados com a posição das origens, tornando-os equivalentes aos parâmetros de Denavit-Hartenberg. Tal equivalência é necessária para evitar incompatibilidade no cálculo da cinemática inversa e na representação gráfica, onde são utilizados.

## APÊNDICE 2

### REPRESENTAÇÃO GRÁFICA DA CADEIA CINEMÁTICA

#### A2.1 - INTRODUÇÃO

Neste Apêndice é apresentado um modo simplificado de representação gráfica de robôs industriais de cadeia aberta.

O procedimento baseia-se na representação da cadeia cinemática do robô por meio de paralelepípedos, utilizando a representação *wireframe*. Os paralelepípedos estão relacionados com os parâmetros de Denavit-Hartenberg. Sistemas de coordenadas localizados nas ligações e na garra podem também ser representados auxiliando no posicionamento relativo das juntas e do efetuador.

#### A2.2 - DEFINIÇÃO DOS PONTOS NOS SISTEMAS LOCAIS

Cada ligação é representada por dois paralelepípedos, cada um dos quais definidos em um sistema local, com base nos parâmetros de Denavit-Hartenberg da ligação. A Figura A2.1 apresenta os pontos que definem cada paralelepípedo. De acordo com a figura, as coordenadas dos vértices do paralelepípedo são definidas em função dos valores  $X_p$ ,  $X_n$ ,  $Y_p$ ,  $Y_n$ ,  $Z_p$  e  $Z_n$ , que representam a interseção de cada eixo do sistema de coordenadas com as faces do volume.

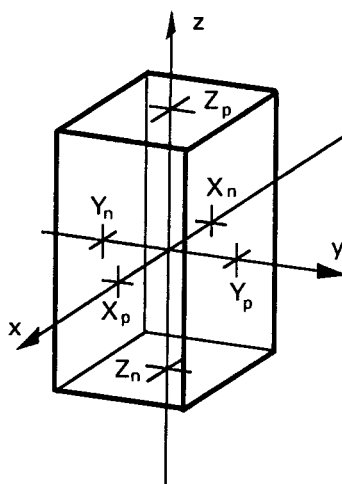


Figura A2.1 - Representação de um paralelepípedo no seu sistema local.

Conforme apresentado no item 3.2.1, a cadeia cinemática de um robô, representada pelos parâmetros de Denavit-Hartenberg, utiliza sistemas de coordenadas em cada ligação. A transformação de um sistema para o seguinte é feita por meio de duas translações e duas rotações [28], conforme mostra a Figura A2.2.

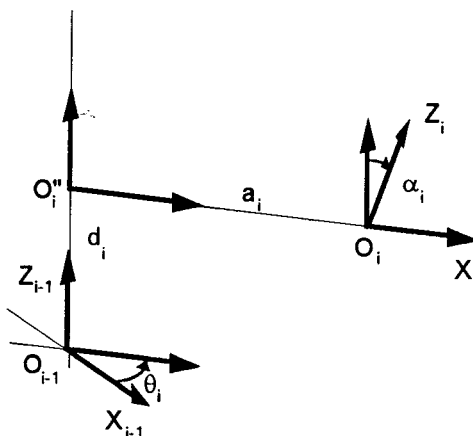


Figura A2.2 - Transformações entre sistemas vizinhos segundo Denavit-Hartenberg.

Com base em tais transformações, pode-se representar uma ligação por meio de paralelepípedos dispostos ao longo das translações  $d$  e  $a$ .

Utilizando-se os sistemas  $i-1$  e o sistema intermediário cuja origem situa-se em  $O''_i$  ( Figura A2.2 ), pode-se representar uma ligação conforme a Figura A2.3, utilizando-se as coordenadas locais definidas na Figura A2.1.

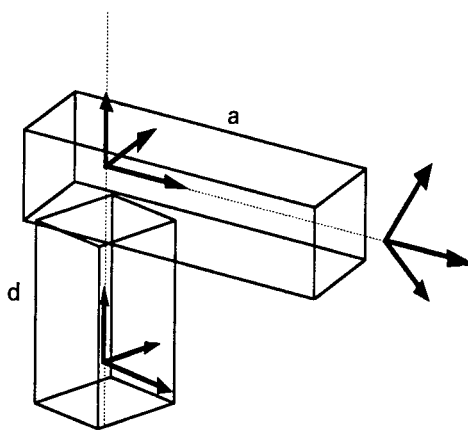


Figura A2.3 - Representação gráfica de uma ligação.

O primeiro volume, construído com base no sistema i-1, possui largura constante e altura correspondente ao parâmetro d. Sobre este volume situa-se o segundo, representado localmente no sistema com origem em O", com altura correspondente ao parâmetro a. A Figura A2.4 apresenta os pontos que formam cada volume e a Tabela A2.1, suas coordenadas nos sistemas locais.

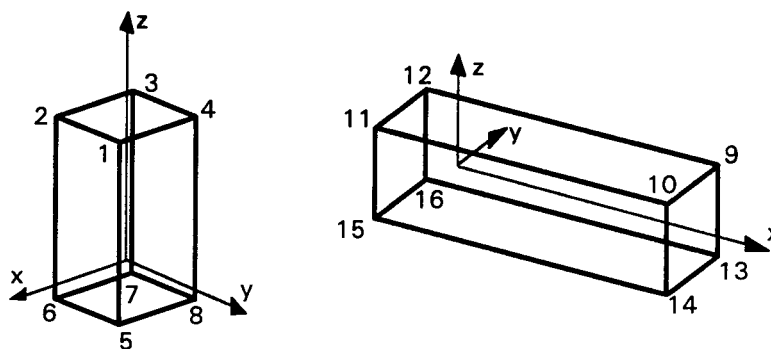


Figura A2.4 - Representação gráfica dos volumes que formam cada ligação.

A espessura de cada paralelepípedo depende de uma constante C, definida para cada ligação. Assim, as coordenadas dos pontos são obtidas do seguinte modo:

$X_p = C$	$X_n = -C$	$X'_p = a_i - C$	$X'_n = -C$
$Y_p = C$	$Y_n = -C$	$Y'_p = C$	$Y'_n = -C$
$Z_p = d_i - C$	$Z_n = -C$	$Z'_p = C$	$Z'_n = -C$

Tabela A2.1: Coordenadas dos vértices e identificação das retas que formam os volumes da Figura A2.4 no sistema local.

PONTO	X	Y	Z
0	0	0	0
1	$X_p$	$Y_p$	$Z_p$
2	$X_p$	$Y_n$	$Z_p$
3	$X_n$	$Y_n$	$Z_p$
4	$X_n$	$Y_p$	$Z_p$
5	$X_p$	$Y_p$	$Z_n$
6	$X_p$	$Y_n$	$Z_n$
7	$X_n$	$Y_n$	$Z_n$
8	$X_n$	$Y_p$	$Z_n$
9	$X'_p$	$Y'_p$	$Z'_p$
10	$X'_p$	$Y'_n$	$Z'_p$
11	$X'_n$	$Y'_n$	$Z'_p$
12	$X'_n$	$Y'_p$	$Z'_p$
13	$X'_p$	$Y'_p$	$Z'_n$
14	$X'_p$	$Y'_n$	$Z'_n$
15	$X'_n$	$Y'_n$	$Z'_n$
16	$X'_n$	$Y'_p$	$Z'_n$
17	0	0	0

RETA	PTO. INICIAL	PTO. FINAL
0	1	2
1	2	3
2	3	4
3	4	1
4	5	6
5	6	7
6	7	8
7	8	5
8	1	5
9	2	6
10	3	7
11	4	8
12	9	10
13	10	11
14	11	12
15	12	9
16	13	14
17	14	15
18	15	16
19	16	13
20	9	13
21	10	14
22	11	15
23	12	16

Para a ligação correspondente à base,  $Z_n = 0$ , significando que o sistema de coordenadas se encontra na face inferior do paralelepípedo.

A representação da garra obedece um procedimento semelhante à das ligações. A Figura A2.5 apresenta os pontos que definem a garra e a Tabela A2.2, as suas coordenadas. O posicionamento do sistema de coordenadas da garra obedece a convenção utilizada por Asada [28] e por Paul [29].

Do mesmo modo que para as ligações, as coordenadas dos pontos da garra são dadas em função de uma constante, conforme mostrado a seguir.

$$X_p = C$$

$$X_n = -C$$

$$Y_p = 2C$$

$$Y_n = -2C$$

$$Z_p = C$$

$$Z_n = 0$$

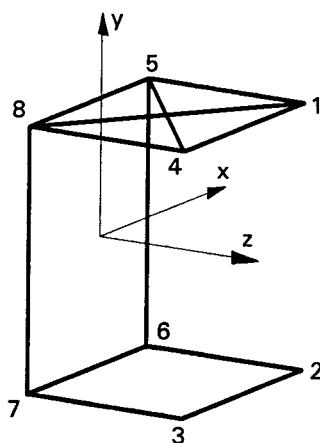


Figura A2.5 - Representação gráfica da garra.

Tabela A2.2: Coordenadas dos vértices e identificação das retas que formam a garra, no sistema local.

PONTO	X	Y	Z
0	0	0	0
1	$X_p$	$Y_p$	$Z_p$
2	$X_p$	$Y_n$	$Z_p$
3	$X_n$	$Y_n$	$Z_p$
4	$X_n$	$Y_p$	$Z_p$
5	$X_p$	$Y_p$	$Z_n$
6	$X_p$	$Y_n$	$Z_n$
7	$X_n$	$Y_n$	$Z_n$
8	$X_n$	$Y_p$	$Z_n$

RETA	PTO. INICIAL	PTO. FINAL
0	1	4
1	4	8
2	8	5
3	5	1
4	1	8
5	4	5
6	5	6
7	6	7
8	7	8
9	2	6
10	2	7
11	3	3

A representação dos sistemas de coordenadas locais de cada ligação, da base e da garra seguem o esquema mostrado na Figura A2.6 cujas coordenadas são apresentadas na Tabela A2.3.

As coordenadas dos pontos são obtidas com base nas seguintes constantes:

$$C_1 = 2C$$

$$C_2 = 1.8C$$

$$C_3 = 0.08C$$

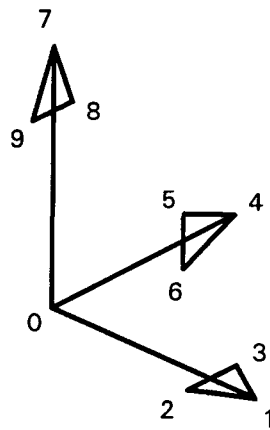


Figura A2.6 - Representação gráfica dos eixos dos sistemas de coordenadas.

Tabela A2.3: Coordenadas dos vértices e identificação das retas que formam os sistemas de coordenadas.

PONTO	X	Y	Z
0	0	0	0
1	C <sub>1</sub>	0	0
2	C <sub>2</sub>	-C <sub>3</sub>	0
3	C <sub>2</sub>	C <sub>3</sub>	0
4	0	C <sub>1</sub>	0
5	0	C <sub>2</sub>	C <sub>3</sub>
6	0	C <sub>2</sub>	-C <sub>3</sub>
7	0	0	C <sub>1</sub>
8	0	C <sub>3</sub>	C <sub>2</sub>
9	0	-C <sub>3</sub>	C <sub>2</sub>

RETA	PTO. INICIAL	PTO. FINAL
0	0	1
1	1	2
2	2	3
3	3	1
4	0	4
5	4	5
6	5	6
7	6	4
8	0	7
9	7	8
10	8	9
11	9	7

### A2.3 - DEFINIÇÃO DOS PONTOS NO SISTEMA GLOBAL

Os pontos no sistema global (base do robô) são obtidos mediante as matrizes de transformação baseadas na notação de Denavit-Hartenberg.

As matrizes  $A_i$ , entre sistemas vizinhos, são formadas pelo produto de quatro transformações [28].

$$A_i = \text{Rot}(\theta_i) \cdot \text{Tr}(d_i) \cdot \text{Tr}(a_i) \cdot \text{Rot}(\alpha_i) \tag{A2.1}$$



Esta transformação pode ser dividida em duas matrizes,  $M1_i$  e  $M2_i$ , de modo que

$$A_i = M1_i . M2_i \tag{A2.2}$$

onde as matrizes intermediárias são dadas por:

$$M1_i = \begin{bmatrix} \cos\theta_i & -\text{sen}\theta_i & 0 & 0 \\ \text{sen}\theta_i & \cos\theta_i & 0 & 0 \\ 0 & 0 & 1 & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{A2.3}$$

$$M2_i = \begin{bmatrix} 1 & 0 & 0 & a_i \\ 0 & \cos\alpha_i & -\text{sen}\alpha_i & 0 \\ 0 & \text{sen}\alpha_i & \cos\alpha_i & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{A2.4}$$

Define-se a matriz  $W_i$ , que transforma coordenadas do sistema de cada junta para o global, e a matriz  $V_i$ , que transforma coordenadas do sistema intermediário entre duas juntas para o global, conforme mostra a Figura A2.7.

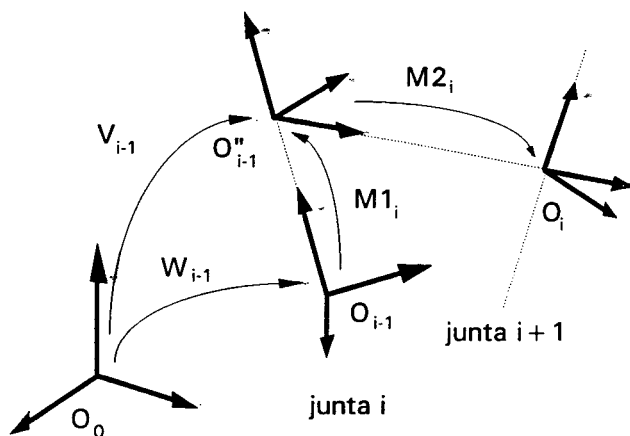


Figura A2.7 - Transformações entre os sistemas de coordenadas.

Pode-se então obter os pontos no sistema global por intermédio das matrizes:

$$\mathbf{V}_{i-1} = \mathbf{W}_{i-1} \cdot \mathbf{M1}_i \quad (\text{A2.5})$$

$$\mathbf{W}_i = \mathbf{V}_{i-1} \cdot \mathbf{M2}_i \quad (\text{A2.6})$$

com  $\mathbf{W}_0 = \mathbf{I}$  e  $i$  variando de 1 a  $N$ .

Após determinado o conjunto de matrizes  $\mathbf{V}$  e  $\mathbf{W}$ , pode-se calcular as coordenadas dos pontos no sistema global. Os pontos do primeiro volume (0 a 8) são transformados pelas matrizes  $\mathbf{W}$  e os do segundo volume (9 a 17), pelas matrizes  $\mathbf{V}$ .

Os pontos que formam os sistemas de coordenadas são obtidos pelas matrizes  $\mathbf{W}$  e os da garra, pela matriz  $\mathbf{W}_N$ , sendo  $N$  o número de graus de liberdade do robô.

#### A2.4 - COMENTÁRIOS SOBRE O PROCEDIMENTO

O procedimento de representação, apesar de não apresentar detalhes característicos de cada robô, mostra-se satisfatório para estudos sobre comportamento cinemático. Melhoramentos podem ser feitos para uma melhor representação do tipo de junta, visto que nesta representação não se pode diferenciar juntas prismáticas e rotativas, a não ser através do movimento da cadeia ou pelos seus parâmetros cinemáticos.

## APÊNDICE 3

### ESTRUTURA DOS ARQUIVOS DE DADOS

#### A3.1 - INTRODUÇÃO

Do modo como o programa foi idealizado, utiliza-se uma série de arquivos de dados, relacionados com os parâmetros cinemáticos do robô e com a sua representação gráfica.

Este Apêndice visa apresentar a estrutura e a finalidade dos arquivos de dados utilizados no programa.

#### A3.2 - ARQUIVOS REFERENTES AO ROBÔ

Cada robô é catalogado por meio de dois arquivos com os seguintes nomes:

- "nome do robô".PAR
- "nome do robô".TXT

O arquivo com extensão **PAR** apresenta todos os parâmetros necessários para a modelagem cinemática e para a representação gráfica. O arquivo possui a estrutura apresentada a seguir:

				<i>dados gerais do robô</i>
xxxxxx				nome do robô
xx				
x				número de graus de liberdade
				<i>dados da junta 1</i>
x				tipo de junta (rotativa ou prismática)
xxxxxx	xxxxxx	xxxxxx		posição da junta
xxxxxx	xxxxxx			limites de posição
xxxxxx				limite de velocidade
xxxxxx				limite de aceleração
xxxxxx	xxxxxx	xxxxxx		vetor de corpo
xxxxxx	xxxxxx	xxxxxx		vetor orientação
xxxxxx	xxxxxx	xxxxxx	xxxxxx	parâmetros de Denavit-Hartenberg
...	...	...	...	
				<i>dados da junta N</i>
x				tipo de junta (rotativa ou prismática)
xxxxxx	xxxxxx	xxxxxx		posição da junta
xxxxxx	xxxxxx			limites de posição
xxxxxx				limite de velocidade
xxxxxx				limite de aceleração
xxxxxx	xxxxxx	xxxxxx		vetor de corpo
xxxxxx	xxxxxx	xxxxxx		vetor orientação
xxxxxx	xxxxxx	xxxxxx	xxxxxx	parâmetros de Denavit-Hartenberg
				<i>dados para a representação gráfica</i>
x				representação (isométrica = 0; cônica = 1)
xxxxxx	xxxxxx			ângulos de visualização (horizontal e vertical)
xxxxxx				zoom
xxxxxx	xxxxxx	xxxxxx		posição do ponto objeto
xxxxxx				distância do observador
xxxxxx	xxxxxx	xxxxxx	xxxxxx	limites da janela do desenho
xxxxxx	xxxxxx			espessuras da base e da garra

O arquivo com extensão **TXT** apresenta os mesmos parâmetros que o arquivo com extensão **PAR**, só que de modo ordenado, para que se possa identificar os valores nele contido. A única finalidade deste arquivo é fornecer os parâmetros do robô em forma de relatório.

Em ambos os arquivos, as unidades de comprimento estão em milímetro e os ângulos em grau.

Todos os robôs cadastrados pelo programa, são incluídos no arquivo **coleção.dat**. O conteúdo compreende o número de robôs cadastrados seguido dos nomes dos robôs. Um exemplo é apresentado a seguir:

A seguir são apresentados, com exemplo, os arquivos **SCARA.PAR**, **SCARA.TXT** e **coleção.dat**.

• Arquivo SCARA.PAR

scara				
4				
r	0.000000	0.000000	980.000000	
	180.000000	-180.000000		
	50.000000			
	100.000000			
	0.000000	0.000000	0.000000	
	0.000000	0.000000	1.000000	
	425.000000	0.000000	980.000000	0.000000
r	425.000000	0.000000	1110.000000	
	180.000000	-180.000000		
	50.000000			
	100.000000			
	425.000000	0.000000	980.000000	
	0.000000	0.000000	1.000000	
	375.000000	0.000000	130.000000	0.000000
r	800.000000	0.000000	1110.000000	
	180.000000	-180.000000		
	50.000000			
	100.000000			
	375.000000	0.000000	130.000000	
	0.000000	0.000000	1.000000	
	0.000000	180.000000	0.000000	0.000000
p	800.000000	0.000000	910.000000	
	300.000000	100.000000		
	50.000000			
	100.000000			
	0.000000	0.000000	0.000000	
	0.000000	0.000000	-1.000000	
	-0.000000	0.000000	250.000000	0.000000
	800.000000	0.000000	860.000000	
	0.000000	0.000000	-250.000000	
	0.000000	0.000000	-1.000000	
	0.000000	-1.000000	0.000000	
0				
60		-30		
1				
0.0		0.0	0.0	
3000				
-1000		1000	-500	1800
100		50		

## • Arquivo SCARA.TXT

NOME: SCARA

N.GRAUS DE LIBERDADE: 4

JUNTA	TIPO	POSIÇÃO	ORIENTAÇÃO	LIM.POSIÇÃO	LIM.VELOC	LIM.ACEL
1	r	0.00	0.0000	s: 180.000	50.00	100.00
		0.00	0.0000	i: -180.000		
		980.00	1.0000			
2	r	425.00	0.0000	s: 180.000	50.00	100.00
		0.00	0.0000	i: -180.000		
		1110.00	1.0000			
3	r	800.00	0.0000	s: 180.000	50.00	100.00
		0.00	0.0000	i: -180.000		
		1110.00	1.0000			
4	p	800.00	0.0000	s:300.000	50.00	100.00
		0.00	0.0000	i: 100.000		
		910.00	-1.0000			

EFETUADOR:	POSIÇÃO	DIREÇÃO	ORIENTAÇÃO
	800.00	0.0000	0.0000
	0.00	0.0000	-1.0000
	860.00	-1.0000	0.0000

## PARÂMETROS DE DENAVIT-HARTENBERG

i	COMPR.	TORÇÃO	DESLOC.	ÂNGULO
1	425.00	0.00	980.00	0.00
2	375.00	0.00	130.00	0.00
3	0.00	180.00	0.00	0.00
4	0.00	0.00	250.00	0.00

## VETORES DA POSIÇÃO-ZERO

i	b[i]			u[i]		
1	0.00	0.00	980.00	0.0000	0.0000	1.0000
2	425.00	0.00	130.00	0.0000	0.0000	1.0000
3	375.00	0.00	0.00	0.0000	0.0000	1.0000
4	0.00	0.00	-200.00	0.0000	0.0000	1.0000
5	0.00	0.00	-50.00			

## APROXIMAÇÃO / ORIENTAÇÃO DO EFETUADOR

ua	0.0000	0.0000	-1.0000
ut	0.0000	-1.0000	0.0000

## DADOS SOBRE A REPRESENTAÇÃO GRÁFICA

REPRESENTAÇÃO	0		
ÂNGULOS HORIZONTAL E VERTICAL ( graus )	0.000	-90.000	
ZOOM	1.000		
PONTO OBJETO ( x y z ) (mm)	0.000	0.000	0.000
POSIÇÃO DO OBSERVADOR (mm)	500000.000		
JANELA DO DESENHO (mm)	-1000.00	1000.00	-1500.00
ESPESSURAS DA BASE E DA GARRA ( mm )		100.000	50.000

- Arquivo **coleção.dat**

6            *número de robôs cadastrados*  
 artrop  
 carte  
 elbow        *lista dos nomes dos robôs*  
 puma  
 scara  
 stanford

### A3.3 - ARQUIVO DE PARÂMETROS GRÁFICOS

Os pontos iniciais e finais que definem cada segmento de reta bem como a identificação dos segmentos, são armazenados no arquivo **par\_cte.dat**, cuja estrutura é apresentada a seguir.

- Arquivo **par\_cte.dat**

													<i>retas que formam as ligações</i>			
1	2	2	3	3	4	4	1	5	6	6	7	7	8			
8	5	1	5	2	6	3	7	4	8	9	10	10	11			
11	12	12	9	13	14	14	15	15	16	16	13	9	13			
10	14	11	15	12	16											
													<i>retas que formam a garra</i>			
1	4	4	8	8	5	5	1	1	8	4	5	5	6			
6	7	7	8	2	6	2	3	3	7							
													<i>retas que formam os eixos dos sistemas</i>			
0	1	1	2	2	3	3	1	0	4	4	5	5	6			
6	4	0	7	7	8	8	9	9	7							

### A3.4 - ARQUIVOS DE TAREFA

As tarefas são armazenadas nos arquivos com extensão **AMB**. Fazem parte deste arquivo:

- matrizes de posicionamento do robô e do sistema auxiliar em relação ao sistema de referência;
- matriz de posicionamento do efetuador em relação à extremidade do robô;
- número de segmentos que formam o percurso;
- número de pontos por segmento;

- matrizes de posicionamento do início de cada segmento, em relação ao sistema auxiliar;
- matriz de posicionamento do final do último segmento, em relação ao sistema auxiliar;

A seguir é apresentado um exemplo de tarefa para o robô **SCARA**.

• - Arquivo **SCARA1.AMB**

1	0	0	-100	<i>posicionamento do robô</i>
0	1	0	0	
0	0	1	0	
0	0	0	1	
1	0	0	0	<i>posicionamento do sistema auxiliar</i>
0	1	0	0	
0	0	1	600	
0	0	0	1	
1	0	0	0	<i>posicionamento do efetuador</i>
0	1	0	0	
0	0	1	0	
0	0	0	1	
4				<i>número de segmentos</i>
10				<i>número de pontos por segmento</i>
0	-1	0	200	<i>início do segmento 0</i>
-1	0	0	-200	
0	0	-1	0	
0	0	0	1	
1	0	0	500	<i>início do segmento 1</i>
0	-1	0	-200	
0	0	-1	200	
0	0	0	1	
-1	0	0	500	<i>início do segmento 2</i>
0	1	0	200	
0	0	-1	200	
0	0	0	1	
-1	0	0	200	<i>início do segmento 3</i>
0	1	0	200	
0	0	-1	200	
0	0	0	1	
0	-1	0	200	<i>final do segmento 3</i>
-1	0	0	-200	
0	0	-1	0	
0	0	0	1	



Os resultados das tarefas, ou seja, os valores das juntas calculadas pela cinemática inversa, podem ser armazenados em arquivos com extensão **RES**. O nome do arquivo permanece igual ao correspondente à tarefa. Fazem parte do arquivo resultado:

- nome da tarefa;
- número de segmentos;
- número de pontos por segmento;
- identificação do ponto;
- segmento a que pertence;
- posição dentro do segmento;
- coordenadas das juntas;

A seguir é apresentado, com exemplo, os resultados da tarefa contida no arquivo **SCARA1.AMB**:

•- Arquivo **SCARA1.RES**

SCARA							
4	10						
1	0	0	-1.5708	2.2142	-2.2145	510.0000	
2	0	1	-1.5000	2.1150	-2.0115	487.7778	
3	0	2	-1.4293	2.0158	-1.8085	465.5556	
4	0	3	-1.3585	1.9167	-1.6055	443.3333	
5	0	4	-1.2877	1.8175	-1.4025	421.1111	
6	0	5	-1.2170	1.7183	-1.1995	398.8889	
7	0	6	-1.1462	1.6191	-0.9965	376.6667	
8	0	7	-1.0755	1.5199	-0.7935	354.4444	
9	0	8	-1.0047	1.4207	-0.5904	332.2222	
10	0	9	-0.9340	1.3215	-0.3874	310.0000	
11	1	0	-0.9340	1.3215	-0.3874	310.0000	
12	1	1	-0.8625	1.3215	-0.1099	310.0000	
13	1	2	-0.7910	1.3215	0.1676	310.0000	
14	1	3	-0.7195	1.3215	0.4452	310.0000	
15	1	4	-0.6480	1.3215	0.7227	310.0000	
16	1	5	-0.5765	1.3215	1.0003	310.0000	
17	1	6	-0.5050	1.3215	1.2778	310.0000	
18	1	7	-0.4335	1.3214	1.5554	310.0000	
19	1	8	-0.3620	1.3214	1.8329	310.0000	
20	1	9	-0.2905	1.3214	2.1105	310.0000	

## • Arquivo SCARA1.RES (continuação)

21	2	0	-0.2905	1.3214	2.1105	310.0000
22	2	1	-0.3021	1.4206	2.0228	310.0000
23	2	2	-0.3137	1.5198	1.9352	310.0000
24	2	3	-0.3253	1.6190	1.8476	310.0000
25	2	4	-0.3368	1.7182	1.7600	310.0000
26	2	5	-0.3484	1.8174	1.6723	310.0000
27	2	6	-0.3600	1.9166	1.5847	310.0000
28	2	7	-0.3716	2.0158	1.4971	310.0000
29	2	8	-0.3832	2.1150	1.4095	310.0000
30	2	9	-0.3948	2.2142	1.3218	310.0000
31	3	0	-0.3948	2.2142	1.3218	310.0000
32	3	1	-0.5254	2.2142	1.6271	332.2222
33	3	2	-0.6561	2.2142	1.9323	354.4444
34	3	3	-0.7868	2.2142	2.2375	376.6667
35	3	4	-0.9174	2.2142	2.5427	398.8889
36	3	5	-1.0481	2.2142	2.8480	421.1111
37	3	6	-1.1788	2.2143	3.1532	443.3333
38	3	7	-1.3094	2.2143	3.4584	465.5556
39	3	8	-1.4401	2.2143	3.7636	487.7778
40	3	9	-1.5708	2.2143	4.0689	510.0000

## APÊNDICE 4

### CINEMÁTICA INVERSA SEGUNDO O MÉTODO DA POSIÇÃO-ZERO

#### A4.1 - INTRODUÇÃO

Este Apêndice apresenta a formulação proposta por Kazerounian [51] para o cálculo da cinemática inversa. O método baseia-se na rotação dos chamados **vetores de corpo** em torno dos **vetores orientação** que indicam a direção de movimento das juntas. As rotações são obtidas por meio de matrizes de rotação.

#### A4.2 - DESCRIÇÃO DO MÉTODO

Para a descrição da formulação segundo a análise da Posição-Zero, utiliza-se as seguintes grandezas:

- $q$  - variáveis das juntas
- $P_H^C$  - vetor posição atual do efetuador
- $R_H^C$  - matriz orientação atual do efetuador
- $P_i^C$  - vetor posição atual da junta  $i$
- $R_H^G$  - matriz orientação desejada do efetuador
- $P_H^G$  - posição desejada do efetuador

Todas as grandezas acima estão representadas no **sistema de base** que é um sistema fixo na base do robô.

Os posicionamentos atual e desejado do efetuador são definidos respectivamente por:

$$[R_H^c; P_H^c] \tag{A4.1}$$

e

$$[R_H^G; P_H^G] \tag{A4.2}$$

Para uma **junta rotativa**, o posicionamento do efetuador, em relação à base, após o movimento da junta  $i$ , é dado pelas expressões:

$$R_H' = R(\Delta q_i, u_i) \cdot R_H^c \tag{A4.3}$$

$$P_H' = P_i^c + R(\Delta q_i, u_i) \cdot (P_H^c - P_i^c) \tag{A4.4}$$

A equação (A4.3) representa a **rotação**  $\Delta q_i$  do sistema de coordenadas do **efetuador** em torno do vetor  $u_i$  da junta  $i$ . A equação (A4.4) representa a nova posição do efetuador após a mesma movimentação. A Figura A4.1 apresenta o significado das equações acima, para um manipulador planar com três graus de liberdade.

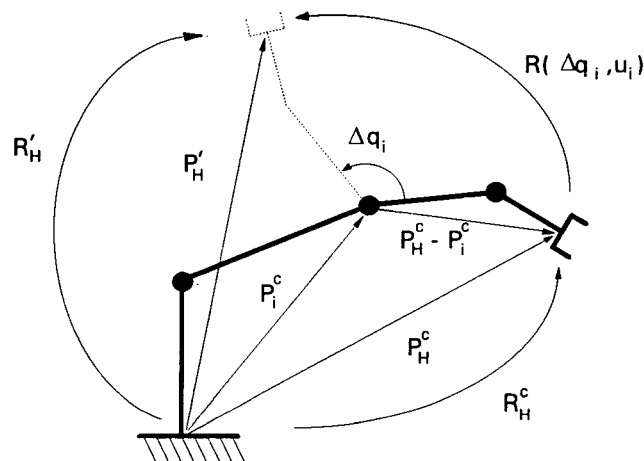


Figura A4.1 - Representação das equações (A4.3) e (A4.4).

Para juntas prismáticas, o fato de não ocorrer rotação faz com que as equações (A4.3) e (A4.4) tomem o seguinte aspecto:

$$\mathbf{R}'_H = \mathbf{R}_H^c \quad (\text{A4.5})$$

$$\mathbf{P}'_H = \mathbf{P}_H^c + \Delta q_i \mathbf{u}_i \quad (\text{A4.6})$$

A Figura A4.2 representa graficamente os deslocamentos representados nas equações (A4.5) e (A4.6).

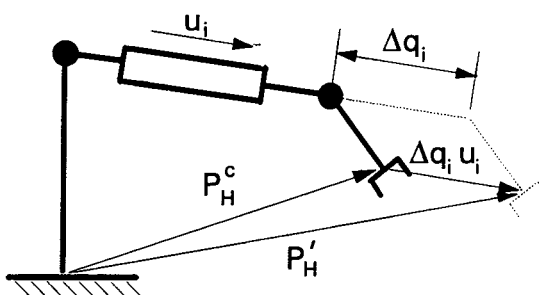


Figura A4.2 - Representação das equações (A4.5) e (A4.6).

Tendo-se o posicionamento atual  $[\mathbf{R}_H^c; \mathbf{P}_H^c]$ , o posicionamento após a movimentação da junta  $i$   $[\mathbf{R}_H; \mathbf{P}_H]$ , e o desejado  $[\mathbf{R}_H^G; \mathbf{P}_H^G]$ , todos representados na Figura A4.3, pode-se obter, a partir da posição atual, a mudança no efetuador devido ao movimento  $\Delta q_i$ , ou seja  $[\bar{\mathbf{R}}; \bar{\mathbf{P}}]$  e a mudança para o posicionamento desejado  $[\mathbf{M}; \mathbf{d}]$ .

A mudança de posicionamento calculada  $[\bar{\mathbf{R}}; \bar{\mathbf{P}}]$  é dada por:

$$\bar{\mathbf{R}} = \mathbf{R}'_H \cdot (\mathbf{R}_H^c)^{-1} \quad (\text{A4.7})$$

$$\bar{\mathbf{P}} = \mathbf{P}'_H - \mathbf{P}_H^c \quad (\text{A4.8})$$

Analogamente, para o posicionamento desejado  $[\mathbf{M}; \mathbf{d}]$ , tem-se:

$$M = R_H^G \cdot (R_H^c)^{-1} \tag{A4.9}$$

$$\bar{P} = P_H^G - P_H^c \tag{A4.10}$$

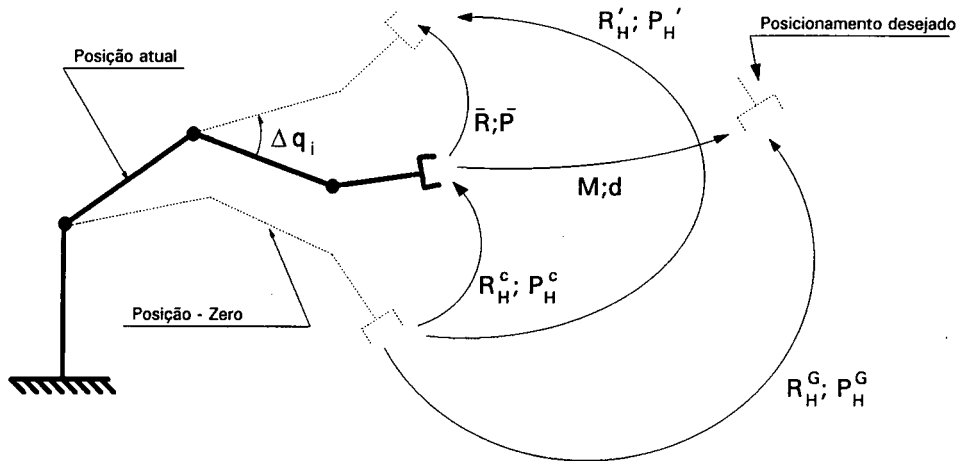


Figura A4.3 - Representação dos vários posicionamentos.

Para quantificar a diferença entre o posicionamento calculado e o desejado, define-se a seguinte norma:

$$\delta = (1 - c) \cdot \delta_R + c \cdot \delta_P$$

sendo:

$$\delta_R = \sum_{j=1}^3 \sum_{k=1}^3 [M_{jk} - \bar{R}_{jk}]^2 \tag{A4.12}$$

$$\delta_P = \sum_{j=1}^3 [P_{Hj}^G - P_{Hj}']^2 \tag{A4.13}$$

Nas expressões acima,  $\delta_R$  representa o erro de orientação,  $\delta_P$  o erro de posição e  $c$  é uma constante no intervalo de 0 a 1, que serve como fator peso.

Como a expressão (A4.11) é uma função cuja única variável é  $\Delta q_i$ , pode-se chegar a um valor mínimo para  $\delta$  fazendo:

$$\frac{d\delta}{d(\Delta q_i)} = 0 \tag{A4.14}$$

e

$$\frac{d^2\delta}{d(\Delta q_i)^2} > 0 \quad (\text{A4.15})$$

Para **juntas rotativas**, desenvolve-se o procedimento separando-se as parcelas constantes das variáveis nas expressões (A4.11) a (A4.13).

De (3.12), tem-se:

$$\mathbf{R}(\Delta q_i, \mathbf{u}_i) = (1 - \cos \Delta q_i) \cdot \mathbf{A} + \sin \Delta q_i \cdot \mathbf{B} + \mathbf{I} \quad (\text{A4.16})$$

onde:

$$\mathbf{A} = \begin{bmatrix} u_x^2 - 1 & u_x u_y & u_x u_z \\ u_x u_y & u_y^2 - 1 & u_y u_z \\ u_x u_z & u_y u_z & u_z^2 - 1 \end{bmatrix} \quad (\text{A4.17})$$

$$\mathbf{B} = \begin{bmatrix} 0 & -u_z & u_y \\ u_z & 0 & -u_x \\ -u_y & u_x & 0 \end{bmatrix} \quad (\text{A4.18})$$

Considerando-se ainda que as grandezas  $\mathbf{q}$ ,  $\mathbf{R}_H^G$ ,  $\mathbf{P}_H^G$ ,  $\mathbf{R}_H^c$ ,  $\mathbf{P}_H^c$  e  $\mathbf{P}_i^c$  são valores conhecidos, as grandezas  $\mathbf{M}$  e  $\mathbf{d}$ , representadas nas equações (A4.9) e (A4.10) podem ser calculadas, juntamente com o vetor

$$\mathbf{h} = \mathbf{P}_H^c - \mathbf{P}_i^c \quad (\text{A4.19})$$

Substituindo-se (A4.3), (A4.4), (A4.9), (A4.10), (A4.16) e (A4.19) em (A4.11), chega-se a:

$$\delta = (1 - c) \cdot \sum_{i=1}^3 \sum_{j=1}^3 \left[ \mathbf{M}_{jk} - (1 - \cos \Delta q_i) \cdot \mathbf{A}_{jk} - \sin \Delta q_i \cdot \mathbf{B}_{jk} - \mathbf{I}_{jk} \right]^2 + \dots + c \cdot \sum_{j=1}^3 \left[ \mathbf{d}_j - (1 - \cos \Delta q_i) \cdot (\mathbf{A} \cdot \mathbf{h})_j - \sin \Delta q_i \cdot (\mathbf{B} \cdot \mathbf{h})_j \right]^2 \quad (\text{A4.20})$$

Nesta expressão,  $M_{jk}$  representa os elementos da matriz  $M$ , dada pela equação (A4.9). Da mesma forma,  $A_{jk}$  e  $B_{jk}$  representam os elementos das matrizes  $A$  e  $B$ , das expressões (A4.17) e (A4.18) e  $I_{jk}$  representa os elementos da matriz Identidade.

Procedendo-se a minimização dada por (A4.14) e (A4.15), após algumas simplificações, chega-se a:

$$\frac{d\delta}{d\Delta q} = Z_1 \cdot \text{sen} \Delta q + Z_2 \cdot \text{cos} \Delta q = 0 \quad (\text{A4.21})$$

onde:

$$Z_1 = (1-c) \cdot \sum_{j=1}^3 \sum_{k=1}^3 [A_{jk}^2 - M_{jk} \cdot A_{jk} + I_{jk} \cdot A_{jk}] + c \cdot \sum_{j=1}^3 [(\mathbf{A} \cdot \mathbf{h})_j^2 - d_j \cdot (\mathbf{A} \cdot \mathbf{h})_j] \quad (\text{A4.22})$$

$$Z_2 = (1-c) \cdot \sum_{j=1}^3 \sum_{k=1}^3 [-M_{jk} \cdot B_{jk} + I_{jk} \cdot B_{jk}] + c \cdot \sum_{j=1}^3 [-d_j \cdot (\mathbf{B} \cdot \mathbf{h})_j] \quad (\text{A4.23})$$

As primeiras parcelas de (A4.22) e (A4.23) estão relacionadas com o erro de orientação, enquanto que as segundas estão relacionadas com o erro de posição.

Da expressão (A4.21) pode-se obter o valor de  $\Delta q_i$ :

$$\Delta q_i = \text{atg} \left( -\frac{Z_2}{Z_1} \right) \quad (\text{A4.24})$$

Da condição de mínimo imposta em (A4.15), resulta:

$$\frac{d^2\delta}{d(\Delta q_i)^2} = Z_1 \cdot \text{cos} \Delta q_i - Z_2 \cdot \text{sen} \Delta q_i \quad (\text{A4.25})$$

Se (A4.25) resultar num valor positivo,  $\Delta q_i$  calculado em (A4.24) é a resposta. Caso contrário, a solução é dada por:



$$\Delta q_i = \pi + \operatorname{atg}\left(-\frac{Z_2}{Z_1}\right) \quad (\text{A4.26})$$

Para o caso de **juntas prismáticas**,  $\delta$  é calculado substituindo-se (A4.5) e (A4.6) em (A4.11). Tem-se então:

$$\delta = (1-c) \cdot \sum_{i=1}^3 \sum_{j=1}^3 [M_{jk} - l_{jk}]^2 + c \cdot \sum_{j=1}^3 [d_j - (\Delta q_i \mathbf{u}_i)_j]^2 \quad (\text{A4.27})$$

Para um  $\Delta q_i$  mínimo, de acordo com a equação (A4.14), tem-se:

$$\frac{d\delta}{d\Delta q_i} = 2 \cdot c \cdot \sum_{j=1}^3 [d_j - (\Delta q_i \mathbf{u}_i)_j] \cdot [-(\mathbf{u}_i)_j] = 0 \quad (\text{A4.28})$$

donde tem-se:

$$\Delta q_i = \sum_{j=1}^3 [d_j \cdot (\mathbf{u}_i)_j] \quad (\text{A4.29})$$

As equações (A4.24) e (A4.29) calculam o deslocamento necessário na junta  $i$ , para que o efetuador se posicione de modo que o erro produzido entre a sua posição atual e a desejada seja o menor possível. Pode-se notar que a expressão (A4.29), ao contrário da (A4.24), corrige apenas posição. Isto já era esperado por se tratar de uma junta prismática. A expressão (A4.24) pode chegar a vários valores diferentes, dependendo do peso  $c$ . Com  $c = 1$ , a correção será toda feita com relação à **posição**. Para  $c = 0$ , somente a **orientação** é corrigida. Assim, dependendo da junta de que se está tratando, pode-se atribuir valores adequados para  $c$  de modo a se chegar mais rapidamente ao posicionamento desejado. Kazerounian [51] sugere que se calcule o peso  $c$  por meio de uma expressão que balanceia o erro total em função dos valores atuais de  $\delta_R$  e  $\delta_P$ .

$$c = \frac{\delta_p}{\delta_R + \delta_p} \quad (\text{A4.30})$$

Deste modo, o maior erro sempre terá maior influência sobre  $c$ . Alguns testes mostraram que se pode melhorar a convergência adotando-se critérios diferentes para as juntas do braço e do pulso. Tais resultados estão apresentados do Capítulo 6.

O cálculo da cinemática inversa, baseado nas equações (A4.24) e (A4.29), está apresentado, na sua forma básica, na Figura A4.4. O algoritmo inicia com uma posição  $q$  conhecida. O posicionamento correspondente do efetuador  $[R_H^c; P_H^c]$  pode ser calculado pelas expressões (3.5) a (3.8) do Capítulo 3. As mudanças  $\Delta q_i$  são calculadas uma a cada vez de modo a minimizar a norma definida pela expressão (A4.11). O procedimento se repete até que o efetuador atinja o posicionamento desejado dentro de um critério de convergência e especificado.

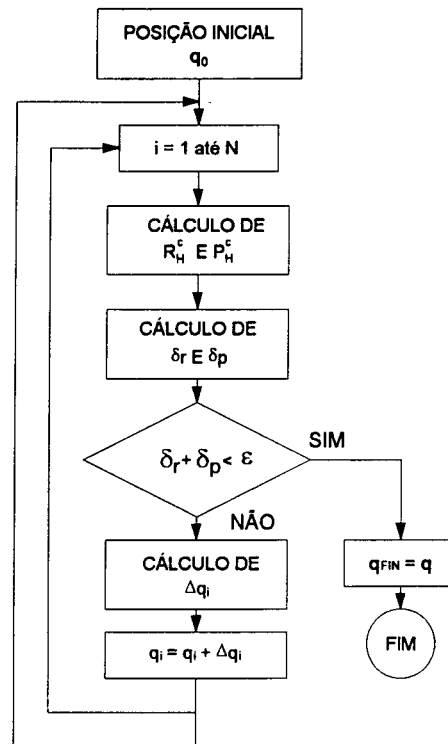


Figura A4.4 - Algoritmo básico para o cálculo da cinemática inversa segundo Kazerounian [51].

O autor ainda sugere algumas melhorias no método, tornando-o um pouco mais eficiente. No cálculo da cinemática direta, após a mudança na junta  $i$ , somente se recalcula os vetores que estão entre tal junta e o efetuador, pois somente aqueles vetores se movimentam. Quando a variação  $\Delta q_i$  for muito pequena, não se calcula a cinemática direta. Quando entre duas iterações, a redução do erro  $\delta$  for muito pequena e apesar disso o valor  $\delta$  do erro ainda for elevado ( $\delta > \epsilon$ ), a configuração é interpretada como uma singularidade.

## APÊNDICE 5

### FUNÇÕES QUE CALCULAM A CINEMÁTICA INVERSA

#### A5.1 - INTRODUÇÃO

Neste Apêndice são descritas as funções do programa que calculam a cinemática inversa. As funções implementadas são as seguintes:

- **cin\_inv\_pz()** - método da Posição Zero;
- **cin\_inv\_PUMA()** - método algébrico - robô PUMA.

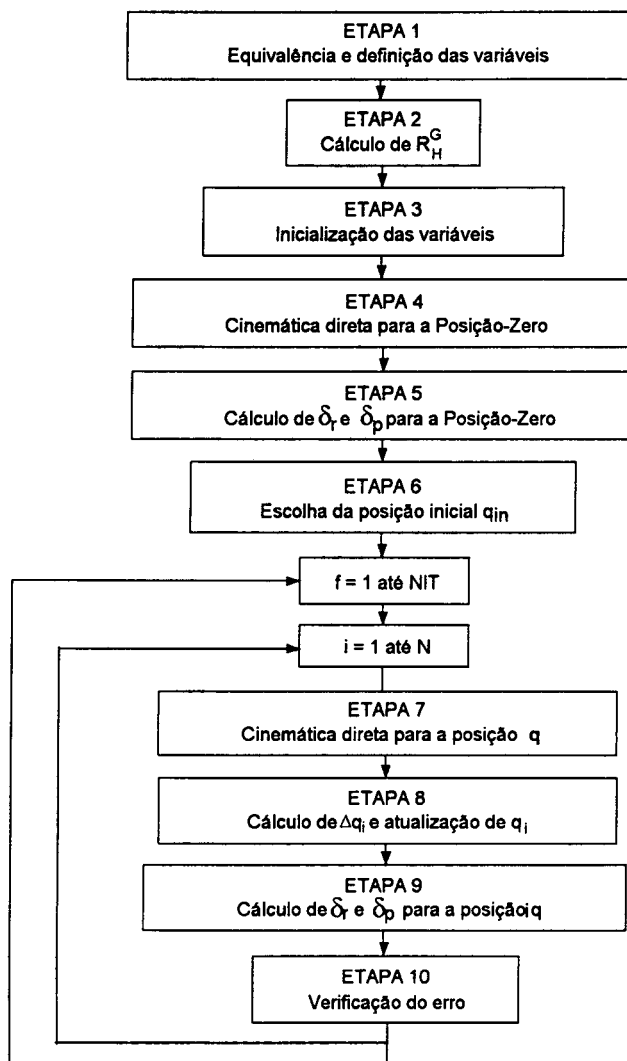
A fim de agilizar a escolha do método, é necessário que os dados de entrada sejam iguais para todas as funções. Além disso, isto facilita a implementação de novos métodos. Assim, optou-se pela passagem dos seguintes parâmetros para as funções:

- **$q_{in}$**  coordenadas generalizadas da configuração atual;
- **$p_{fin}$**  posição desejada do efetuador;
- **$b_{fin}$**  vetor direção desejada do efetuador
- **$t_{fin}$**  vetor orientação desejada do efetuador
- **$\epsilon$**  precisão desejada;
- **$q_{fin}$**  coordenadas generalizadas calculadas.

Os vetores  $\mathbf{b}_{fin}$  e  $\mathbf{t}_{fin}$  correspondem, respectivamente, aos vetores  $\mathbf{u}_a$  e  $\mathbf{u}_t$ , para a Posição-Zero, e aos vetores  $\mathbf{z}$  e  $\mathbf{y}$ , para a modelagem segundo Denavit-Hartenberg. Segue-se a descrição das funções citadas.

### A5.2 - FUNÇÃO cin\_inv\_pz()

Esta função faz o cálculo da cinemática inversa pelo método da análise da Posição-Zero [51], descrita no Apêndice 4. Ela calcula as coordenadas das juntas, correspondentes a um dado posicionamento ( $\mathbf{P}_H^G$ ,  $\mathbf{u}_{aG}$  e  $\mathbf{u}_{tG}$ ) do efetuador. A Figura A5.1 apresenta o fluxograma da função, com as etapas que a compõem as quais estão apresentadas a seguir.



### a) Etapa 1 - Equivalência e definição das variáveis.

Na etapa 1 faz-se a equivalência das variáveis das juntas. A Posição-Zero, representada por  $q_0$ , corresponde aos valores de deslocamento da junta ( $d$ ) ou ângulo da junta ( $\theta$ ), armazenados no arquivo de parâmetros do robô. Faz-se também a definição das variáveis locais.

### b) Etapa 2 - Cálculo de $R_H^G$

O cálculo da matriz rotação referente à posição desejada se baseia nos vetores  $u_{aG}$ ,  $u_{tG}$ ,  $u_{a0}$  e  $u_{t0}$ . Estes últimos correspondem à orientação do efetuador na **Posição-Zero**. Um terceiro vetor,  $u_b$ , é definido, conforme a Figura A5.2, pelo produto vetorial:

$$u_b = u_t \times u_a \quad (A5.1)$$

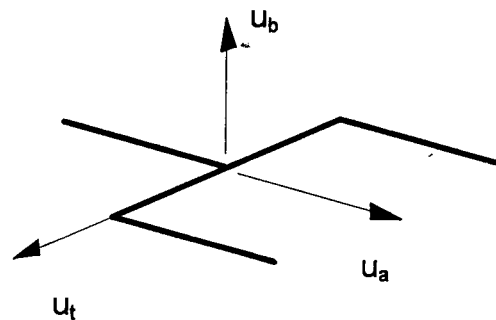


Figura A5.2 - Vetores orientação do efetuador [51].

Para a orientação desejada, as orientações de cada vetor são dadas por:

$$u_{aG} = R_H^G \cdot u_{a0} \quad (A5.2)$$

$$u_{tG} = R_H^G \cdot u_{t0} \quad (A5.3)$$

$$u_{bG} = R_H^G \cdot u_{b0} \quad (A5.4)$$

Tais expressões podem ser agrupadas em um produto de matrizes na forma:

$$\begin{bmatrix} \mathbf{u}_{aG} & \mathbf{u}_{tG} & \mathbf{u}_{bG} \end{bmatrix} = \mathbf{R}_H^G \cdot \begin{bmatrix} \mathbf{u}_{a0} & \mathbf{u}_{t0} & \mathbf{u}_{b0} \end{bmatrix} \quad (\text{A5.5})$$

Devido à ortonormalidade da matriz formada pelos vetores orientação, chega-se a:

$$\mathbf{R}_H^G = \begin{bmatrix} \mathbf{u}_{aG} & \mathbf{u}_{tG} & \mathbf{u}_{bG} \end{bmatrix} \cdot \begin{bmatrix} \mathbf{u}_{a0} & \mathbf{u}_{t0} & \mathbf{u}_{b0} \end{bmatrix}^T \quad (\text{A5.6})$$

O cálculo da expressão (A5.6) se encontra na função `calcula_rhg()`.

### c) Etapa 3 - Inicialização das variáveis.

Os valores iniciais são atribuídos às variáveis nesta etapa. São zeradas as variáveis correspondentes às posições e rotações das juntas bem como as variáveis auxiliares. A matriz de rotação  $\mathbf{R}_0^c$  é igualada à matriz identidade. Apesar de não ter representação, ela é necessária para o cálculo recursivo das rotações de cada junta.

### d) Etapa 4 - Cinemática direta para a Posição-Zero.

Conhecidos os vetores  $\mathbf{u}_0$  e  $\mathbf{b}_0$ , calcula-se a posição,  $\mathbf{P}_H^0$ , e a orientação,  $\mathbf{R}_H^0$ , do efetuador para a Posição-Zero. O cálculo da posição é feito pela soma dos vetores de corpo  $\mathbf{b}_0$ . A matriz de rotação é a identidade por se tratar da Posição-Zero (não ocorreu nenhuma rotação). O fluxograma desta etapa está apresentado na Figura A5.2.

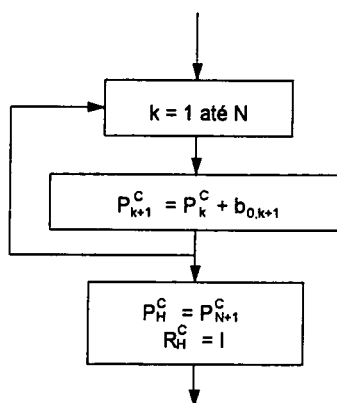


Figura A5.2 - Fluxograma correspondente à Etapa 4.

**e) Etapa 5 - Cálculo dos erros para a Posição-Zero.**

Os erros iniciais de posição e orientação, dados pelas expressões (A4.12) e (A4.13), do Apêndice 4, são calculados através da função `calcula_deltar_e_deltap()`. Para isto é necessário o cálculo de  $R_H'$ ,  $P_H$  e a transposta de  $R_H^c$ .

Como ainda não houve nenhuma mudança no posicionamento, faz-se

$$R_H' = R_H^c \tag{A5.7}$$

$$P_H' = P_H^c \tag{A5.8}$$

Estes cálculos são executados, respectivamente, nas funções `iguala_mat_33()` e `iguala_vet_3()`. A transposta de  $R_H^c$  é calculada através da função `mat_transp_33()`. A Figura A5.3 apresenta o fluxograma correspondente.

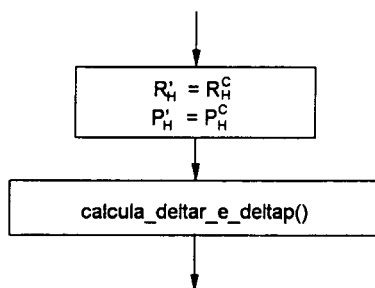


Figura A5.3 - Fluxograma correspondente à Etapa 5.



**f) Etapa 6 - Escolha da posição inicial.**

Como já foi observado no item 3.6, a cinemática inversa pode possuir mais de uma solução. Pode-se chegar à solução desejada induzindo-se a sua configuração na posição inicial do robô, o que é feito nesta etapa. Os valores da posição inicial aqui inseridos são armazenados num vetor  $\mathbf{q}$ , o qual será atualizado em cada iteração. A Figura A5.4 apresenta o fluxograma correspondente.

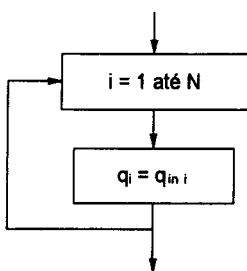


Figura A5.4 - Fluxograma correspondente à Etapa 6.

**g) Etapa 7 - Cinemática direta para a posição  $\mathbf{q}$ .**

A partir desta etapa inicia-se o processo iterativo. A posição e a orientação do efetuador são calculadas com base nos valores atualizados do vetor  $\mathbf{q}$ , de coordenadas das juntas. Cada vetor de corpo  $\mathbf{b}_0$  e cada vetor orientação  $\mathbf{u}_0$ , são girados do valor correspondente às rotações das juntas anteriores a eles. Este valor acumulado está representado pela matriz rotação  $\mathbf{R}_k^c$ , de cada junta  $k$ . O cálculo é baseado nas expressões (3.5) a (3.17).

A Figura A5.5 apresenta os passos da Etapa 7.

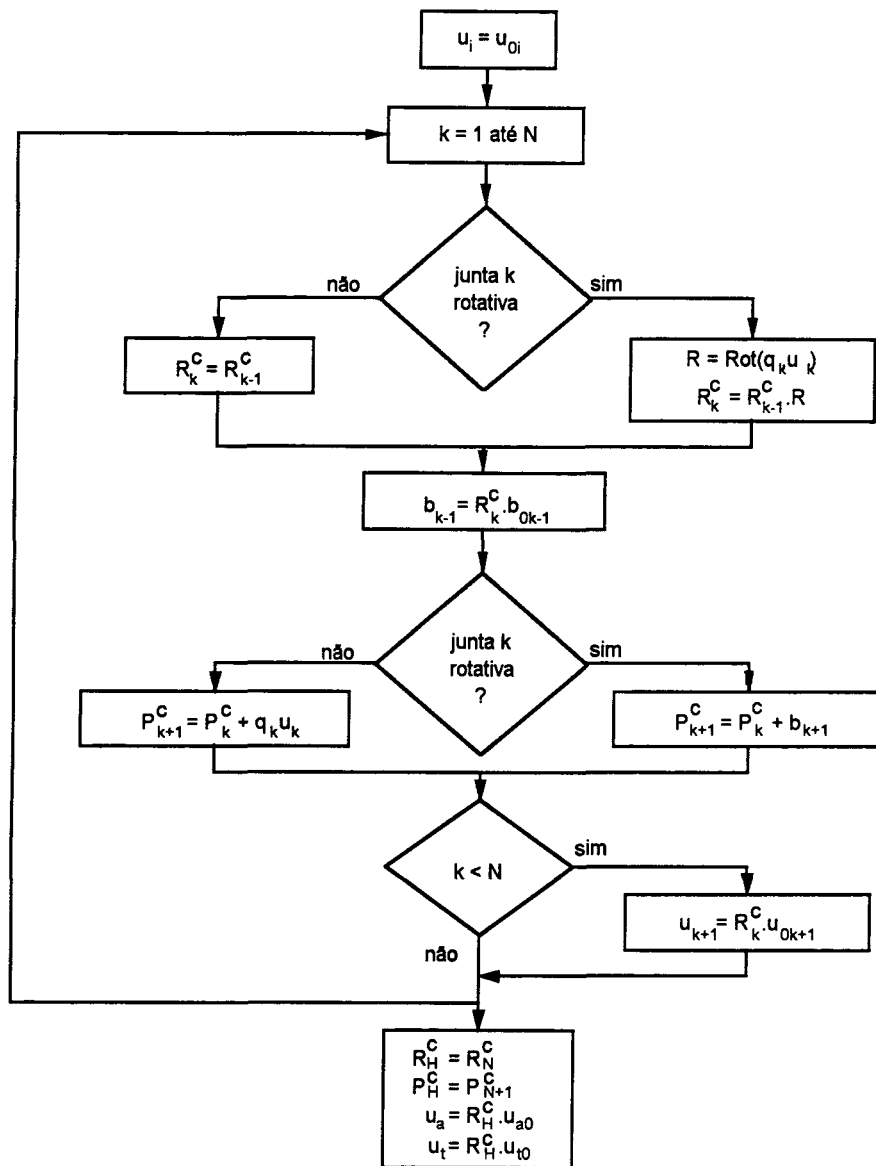


Figura A5.5 - Fluxograma correspondente à Etapa 7.

**h) Etapa 8 - Cálculo de  $\Delta q_i$  e atualização de  $q$  .**

Na etapa 8, representada no fluxograma da Figura A5.6, o valor de  $\Delta q_i$ , para juntas rotativas e prismáticas, é calculado a partir do posicionamento atual do efetuador. O seu valor vai representar deslocamento necessário na junta  $i$  que minimize o erro entre o posicionamento atingido e o desejado.

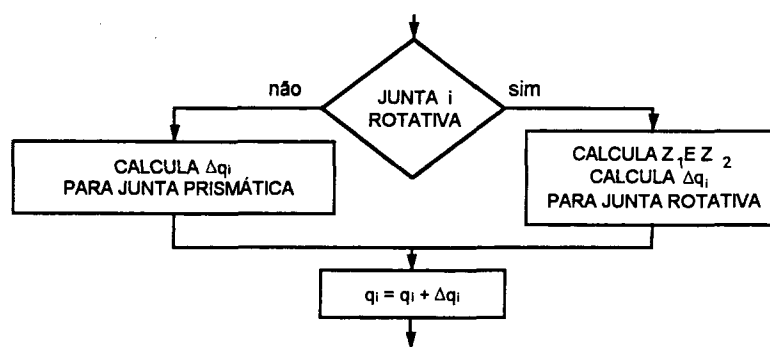


Figura A5.6 - Fluxograma correspondente à Etapa 8.

Para juntas rotativas utiliza-se as funções **calcula\_z1\_e\_z2()** e **calcula\_deltaqi\_rot()**, construídas com base nas expressões (A4.22) a (A4.26). Na função **calcula\_z1\_e\_z2()** se encontra o cálculo do peso  $c$ , descrito no Apêndice 4, que é calculado pela expressão (A4.30). Dependendo do valor fixado para  $c$ , corrige-se mais a orientação ou a posição. Considerando que a última junta corrige apenas orientação, fixou-se  $c = 0$  no cálculo do valor de  $\Delta q_i$  correspondente. O cálculo para as juntas prismáticas é feito na função **calcula\_deltaqi\_pris()**, com base na expressão (A4.29). Após o cálculo de  $\Delta q_i$ , atualiza-se a coordenada  $q_i$ .

#### i) Etapa 9 - Cálculo do erro para a posição atualizada.

Nesta etapa, representada na Figura A5.7, os erros de posição e orientação após a atualização, dados pelas expressões (A4.12) e (A4.13), são calculados na função **calcula\_deltar\_e\_deltap()**. A matriz  $R_H'$  e o vetor  $P_H'$ , representando o posicionamento do efetuador devido à mudança  $\Delta q_i$ , na junta  $i$ , são calculados com base nas expressões (A4.3) e (A4.4), para juntas rotativas e (A4.5) e (A4.6), para juntas prismáticas. A Figura A5.7 apresenta o fluxograma correspondente.

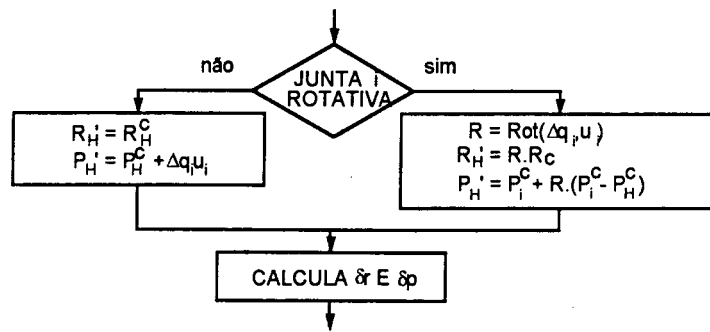


Figura A5.7 - Fluxograma correspondente à Etapa 9.

**j) Etapa 10 - Verificação do erro.**

Caso a soma dos erros  $\delta_r$  e  $\delta_p$  seja menor que um valor  $\epsilon$  pré-definido, chegou-se ao vetor  $q_{fin}$  desejado. Caso contrário, o processo iterativo continua para as demais juntas. Se ainda assim não se chegar ao resultado, reinicia-se a iteração pela junta 1. Esta sequência está apresentada na Figura A5.8.

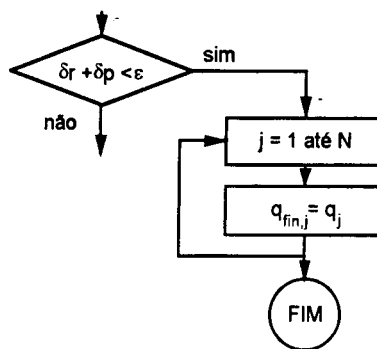


Figura A5.8 - Fluxograma correspondente à Etapa 10.

**A5.3 - FUNÇÃO cin\_inv\_PUMA()**

Esta função foi implementada com base no procedimento analítico apresentado por Fu *et alli* [02], para a solução da cinemática inversa do robô PUMA-560. A Figura A5.9 apresenta o fluxograma correspondente.

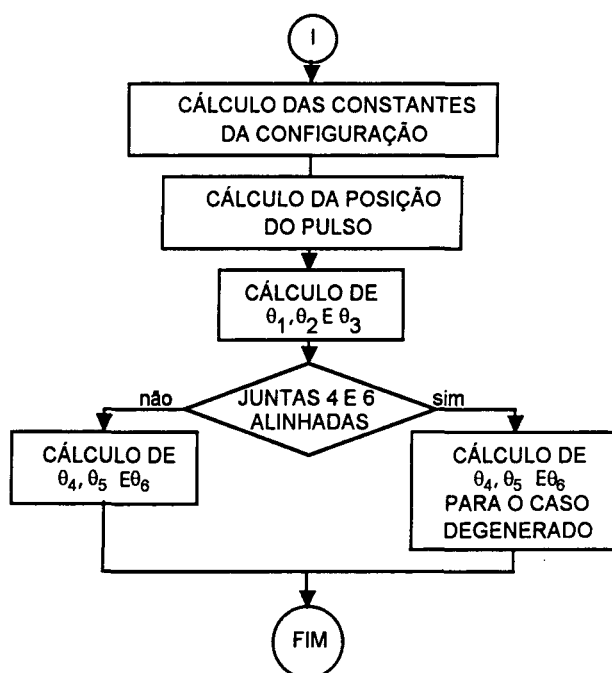


Figura A5.9 - Função `cin_inv_PUMA()`.

Uma das características da cinemática inversa analítica é a possibilidade da obtenção de todas as soluções. A solução de interesse pode ser definida em função de uma configuração inicial, como no caso dos métodos numéricos. Neste procedimento, a identificação da configuração se faz por meio de constantes (+1 e -1), indicando a posição do "ombro" (esquerdo ou direito), "cotovelo" (alto ou baixo) e "pulso" (alto ou baixo). As constantes são definidas de acordo com a posição relativa entre certos sistemas de coordenadas intermediários [02].

Definidas as constantes de configuração, calcula-se a posição do pulso, que é o ponto base para o cálculo das coordenadas.

O cálculo dos três primeiros ângulos é feito por primeiro, não havendo problemas de degeneração. Já nas três juntas do pulso, tal problema ocorre quando as juntas 4 e 6 se alinham. Neste caso um tratamento particular deve ser utilizado, tendo em vista que existem infinitas soluções para aquelas juntas. Optou-se por zerar a junta 4 de modo que a junta 6 assumo o valor necessário para o posicionamento correto do efetuador.