RICARDO MATONE
Eng. Mecânico, Universidade de São Paulo, 1989

MÉTODOS NUMÉRICOS PARA CINEMÁTICA INVERSA DE ROBÔS MANIPULADORES

Dissertação apresentada ao Depto. de Eng. Mecânica da Escola Politécnica da USP para obtenção do título de Mestre em Engenharia.

Orientador: Lucas Antônio Moscato
Professor Titular do Depto.
de Engenharia Mecânica

São Paulo, 1991

FOLHA GUIA PARA ELABORAÇÃO DE RESUMO

AUTOR: MATONE, Ricardo

TÍTULO: Métodos numéricos para cinemática inversa de robôs ...

NESTE TRABALHO FAZ-SE UM ESTUDO SOBRE TECNICAS DE MODELAGEM E SOBRE ALGORITMOS NÚMERICOS PARA CINEMATICA INVERSA DE ROBÔS MANIPULADORES. ASSIM, ANALISA-SE TRÊS CLASSES DE MÉTODOS: BASEADOS NO NEWTON-RAPSON, MÉTODOS QUE NÃO NECESSITAM INVERTER MATRIZES (BASEADOS NA TRANSPOSTA DO JACOBIANO) E MÉTODOS QUE SOLUCIONAM A CINEMATICA INVERSA PARA MANIPULADORES REDUNDANTES. QUANDO ESTUDA-SE OS MÉTODOS BASEADOS NO DE NEWTON-RAPHSON, PROPÕE-SE UMA ADAPTAÇÃO QUE AUMENTA A SUA VELOCIDADE PARA CASOS DE TRAJETÓRIA (MÉTODO DE NEWTON-RAPHSON ADAPTADO); QUANDO ESTUDA-SE ALGORITMOS PARA MANIPULADORES REDUNDANTES, PROPÕE-SE UTILIZAR A IDÉIA DE SCIAVICCO E SICILIANO, QUE CONSISTE EM INTRODUZIR EQUAÇÕES ADICIONAIS EM UM SISTEMA REDUNDANTE, DE FORMA A TORNA-LO DETERMINADO E A "TRAZER" A SOLUÇÃO DO SISTEMA ORIGINAL PARA ALGUMA FAIXA DE INTERESSE (SEGUNDO O CRITÉRIO QUE SE DESEJE OTIMIZAR), EM CONJUNTO COM O MÉTODO DE NEWTON-RAPHSON ADAPTADO, DE FORMA A GERAR UM ALGORITMO MAIS EFICIENTE QUE O ORIGINAL (MÉTODO HIBRIDO). AO FINAL, APLICA-SE O MÉTODO HÍBRIDO NO DESENVOLVIMENTO DE UM SIMULADOR CINEMATICO PARA O SISTEMA "ROBÔ ASEA IRB L6 DE 6 GRAUS DE LIBERDADE + TRILHO".

RESUMO

Neste trabalho faz-se um estudo sobre técnicas de modelagem e sobre algoritmos numéricos para cinemática inversa de robôs manipuladores de cadeia aberta. Assim, analiza-se três classes de métodos para a solução deste problema: métodos baseados no de *Newton-Raphson*, métodos que não necessitam inverter matrizes (algoritmos baseados na transposta do Jacobiano) e métodos que solucionam a cinemática inversa para manipuladores redundantes. Quando estuda-se os métodos baseados no de *Newton-Raphson*, propõe-se uma adaptação que aumenta a sua velocidade para casos de trajetória (método de *Newton-Raphson* adaptado); quando estuda-se algoritmos para manipuladores redundantes, propõe-se utilizar a idéia de *Sciavicco e Siciliano* [SCI88], que consiste em introduzir equações adicionais em um sistema redundante, de forma a torná-lo determinado e a "trazer" a solução do sistema original para alguma faixa de interesse (segundo o critério que se deseje otimizar), em conjunto com o método de *Newton-Raphson* adaptado, de forma a gerar um algoritmo mais eficiente que o original (método híbrido). Ao final, aplica-se o método híbrido no desenvolvimento de um simulador cinemático para o sistema "robô ASEA IRBL6 (existente no Departamento de Engenharia Mecânica da EPUSP) de 6 graus de liberdade + trilho". Com o simulador, pretende-se mostrar uma forma de solucionar um problema freqüentemente encontrado em células e sistemas flexíveis de manufatura, ou seja, *quando se adapta um trilho a um robô de 6 graus de liberdade visando aumentar o seu volume de trabalho, como aproveitar a introdução do trilho para gerar trajetórias otimizadas no espaço de juntas, dada uma trajetória em termos de coordenadas de posição e orientação do efetuador*?

Palavras-Chave: Robôs, Cinemática Inversa, Métodos Numéricos, Manipuladores Redundantes.

ABSTRACT

This work presents a study about modelling technics and numerical algorithms for solving the inverse kinematics of robot manipulators. Three classes of methods are analized for the solution of this problem: (1) *Newton-Raphson* based methods; (2) methods that don't require matrix inversion (transpose Jacobian based algorithms); and (3) methods for redundant manipulators. An adaptation which increases the computational speed of *Newton-Raphson* based methods in the case of trajectories is proposed (adapted *Newton-Raphson* method). For redundant manipulators it is proposed to use the *Sciavicco and Siciliano* idea with the adapted *Newton-Raphson* method in order to generate a more efficient algorithm (hibrid method). The *Sciavicco and Siciliano* idea consists of putting additional equations in a redundant system so that the resulting system would be determinated. The additional equations imposes a new condition to the redundant manipulator according to some pre-established criteria. The criteria used in this work are: obstacle avoiding and limits for the joint coordinates. At the end it is applied the hibrid method in the development of a kinematic simulator for the "ASEA IRBL6 robot (with 6 degrees of freedom) + track" system. The purpose of the simulator is to solve a problem frequently found in flexible manufacturing cells and systems, which is: supposing that a 6 degree of freedom robot is adapted in a track, *how to make use of the new degree of freedom to generate optimized trajectories in joint coordinates space, given a trajectory in terms of end effector position and orientation?*

Key-Words: Robots, Inverse Kinematics, Numerical Methods, Redundant Manipulators.

AGRADECIMENTOS

O autor deseja expressar seus mais sinceros agradecimentos às seguintes pessoas:

Ao Prof. Dr. Lucas A. Moscato, orientador deste trabalho, pelo constante apoio, pela valiosa orientação e pelo determinante incentivo e compreensão;

ao Prof. Dr. Eduardo L.L. Cabral, pelas decisivas sugestões, pelas inúmeras discussões esclarecedoras sobre robótica, pelo incentivo para seguir com este tema de mestrado e por ter me introduzido nesta área de pesquisa;

ao Prof. Dr. Paulo Cordaro, do Instituto de Matemática e Estatística da USP, pelo indispensável auxílio no desenvolvimento do capítulo 3 deste texto;

ao Prof. Dr. Jaime Cruz, do Depto. de Eng. Elétrica da EPUSP, pela pacienciosa ajuda no entendimento de alguns algoritmos de cinemática inversa de manipuladores redundantes;

ao colega Prof. Flávio Cipparrone, pelas várias e importantes conversas sobre robótica que, certamente, contribuiram para a realização desta dissertação;

ao colega Prof. Marcos Tsuzuki, pela sempre disposta colaboração no tocante a utilização de seu sistema gráfico de modelagem de sólidos;

ao aluno do Depto. de Eng. Mecânica Rogério Ferraz Penalva, pelo valioso auxílio no desenvolvimento da interface gráfica do simulador e pela constante cooperação durante a execução deste trabalho;

aos colegas e amigos do Depto. de Eng. Mecânica por terem criado um ambiente agradável e favorável à pesquisa durante o período deste mestrado.

Conteúdo

Lista de Figuras

Capítulo 1

INTRODUÇÃO

1.1 JUSTIFICATIVA E OBJETIVOS

Nos últimos anos, tem sido desenvolvida muita pesquisa visando obter algoritmos mais eficientes para computação da cinemática inversa de manipuladores de cadeia aberta.

Como se sabe, este problema consiste em determinar o vetor $\boldsymbol{q}$ de coordenadas generalizadas de junta do robô, dado o vetor que contém a posição e orientação $\boldsymbol{y}$ de seu efetuador. A solução analítica da cinemática inversa somente é possível para manipuladores com estrutura cinemática bastante particular [ROT76], de forma que quando esta não é possível, o único meio de se obter as coordenadas de junta é através de métodos numéricos.

Observando-se o problema diferencial direto ($d\boldsymbol{y}/dt = \boldsymbol{J}(\boldsymbol{q}).d\boldsymbol{q}/dt$), verifica-se que o problema inverso pode ser resolvido a partir do cálculo da inversa da matriz $\boldsymbol{J}(\boldsymbol{q})$ (Jacobiano), ou inversa generalizada [GRE59,BOU71,BEN74] de $\boldsymbol{J}$ quando esta não é quadrada ou possui determinante nulo. De fato, os primeiros algoritmos que surgiram para cinemática inversa necessitam da computação da inversa do Jacobiano. Isto, entretanto, além de demandar um número elevado de operações matemáticas por iteração, gera problemas numéricos (divisão por números próximos de zero) em posições de singularidade do robô [1] (posições em que não existe a inversa de $\boldsymbol{J}$).

Na metade da década de 80 começou a surgir uma nova classe de algoritmos iterativos que solucionam a cinemática inversa sem necessitar inverter o Jacobiano. Ao invés disto estes algoritmos necessitam apenas transpor a matriz $\boldsymbol{J}$. Grande parte destes métodos surgiram a partir de equações de teorias de controle e a estabilidade dos mesmos é demonstrada utilizando-se a teoria de *Lyapunov*. É o caso dos métodos que utilizam *"impedance control"* [ASA85,CAB91] e *"sliding mode control"* [NOV90] para solucionar a cinemática inversa.

[1] Quando utiliza-se a pseudo-inversa ao invés da inversa, diminuem os problemas numéricos em posições de singularidade, mas a complexidade de cálculo aumenta bastante.

É importante notar que grande parte dos manipuladores com 6 graus de liberdade (aqueles que possuem $posto(\boldsymbol{J}) = 6$) possuem um número finito de vetores $\boldsymbol{q}$ correspondentes a um vetor $\boldsymbol{y}$ dado [2]. Observando-se o problema de pontos de singularidade e de presença de obstáculos, situações em que o robô não consegue movimentar-se segundo certas direções, nota-se que seria muito desejável que o robô possuisse mais do que seis graus de liberdade. Neste caso, um dado vetor de posição e orientação $\boldsymbol{y}$, corresponderia a infinitos vetores de coordenadas generalizadas de junta $\boldsymbol{q}$, permitindo maior flexibilidade para o planejamento de tarefas. Sistemas com esta característica recebem o nome de manipuladores redundantes. Uma nova classe de algoritmos para solução da cinemática inversa tem sido desenvolvida visando aplicação em manipuladores redundantes [DAS88,CHE88,SCI88,DUB88].

Muitas vezes, a redundância aparece como conseqüência da solução de um problema prático. É freqüente, por exemplo, a instalação de trilhos em manipuladores de células e sistemas flexíveis de manufatura visando aumentar o volume de trabalho do robô. A introdução do trilho aumenta o número de graus de liberdade do sistema, de forma que se o mesmo possuisse, por exemplo, 6 graus, passaria a ter 7 e a ser um sistema redundante.

No sistema "robô + trilho" colocado acima, costuma-se utilizar dois controladores, um para o trilho e outro para o robô; isto, porque o controle da grande maioria dos robôs industriais possui arquitetura "fechada", ou seja, é muito difícil o acesso a parâmetros do controlador. Desta forma, torna-se mais fácil trabalhar com dois controladores do que modificar o sistema de controle do robô para aceitar mais um grau de liberdade (movimento do robô no trilho). Neste caso, a otimização dos movimentos do sistema fica bastante dificultada. Isto ocorre porque dada uma localização do robô especificada em termos de posição e orientação do efetuador, o controlador do robô calcula as coordenadas de junta correspondentes, sem levar em conta a existência do trilho. Estas coordenadas, calculadas desta forma, não são necessariamente as melhores (segundo algum critério que se queira otimizar) para o sistema "robô + trilho".

Uma proposta aqui sugerida para solucionar este problema é implementar um simulador cinemático para o sistema "robô + trilho", de forma que a geração da trajetória no espaço de juntas seja executada "off-line", externamente ao sistema, num simulador. Assim, para uma trajetória fornecida em termos de coordenadas de posição e orientação do efetuador, o simulador nos forneceria a trajetória do robô escrita no espaço de juntas e a trajetória do mesmo no trilho, de maneira que cada um destes dados poderia ser tratado pelo controlador correspondente.

Neste trabalho faz-se um estudo sobre técnicas de modelagem e sobre algoritmos numéricos para cinemática inversa de robôs manipuladores de cadeia aberta. Assim, analiza-se três classes de métodos para a solução deste problema: métodos baseados no de *Newton-Raphson*, métodos que não necessitam inverter matrizes (algoritmos baseados na transposta do Jacobiano) e métodos que solucionam a cinemática inversa para manipuladores redundantes. Quando estuda-se os métodos baseados no de *Newton-Raphson*, propõe-se uma adaptação que aumenta a sua velocidade para casos de trajetória (método

[2] Segundo *Roth* [ROT76], manipuladores com 6 graus de liberdade de rotação com $posto(\boldsymbol{J}) = 6$ possuem no máximo 32 soluções para a cinemática inversa.

de *Newton-Raphson* adaptado); quando estuda-se algoritmos para manipuladores redundantes, propõe-se utilizar a idéia de *Sciavicco e Siciliano* em conjunto com o método de *Newton-Raphson* adaptado, de forma a gerar um algoritmo mais eficiente que o original (método híbrido). A idéia de *Sciavicco e Siciliano* consiste em introduzir equações adicionais em um sistema redundante, de forma a torná-lo determinado e a "trazer" a solução do sistema original para alguma faixa de interesse, segundo o critério que se deseje otimizar. Os critérios utilizados nesta dissertação são: limitar coordenadas de junta e evitar obstáculos. Ao final, aplica-se o método híbrido no desenvolvimento de um simulador cinemático, nos moldes colocados no parágrafo anterior, para o sistema "robô ASEA IRBL6 (existente no Departamento de Engenharia Mecânica da EPUSP) de 6 graus de liberdade + trilho".

1.2 TÓPICOS DE CADA CAPÍTULO

A seguir, coloca-se os temas que são tratados em cada capítulo da dissertação de mestrado.

CAPÍTULO 2. ASPECTOS CINEMÁTICOS DE ROBÔS

Neste capítulo, são abordados os seguintes tópicos: posição e orientação de corpos rígidos, transformação de coordenadas, notação de *Denavit -Hartenberg*, equacionamento cinemático de manipuladores de cadeia aberta (obtenção dos modelos direto e diferencial direto) e colocação do problema de cinemática inversa iterativa (cinemática inversa analítica versus cinemática inversa por métodos iterativos).

Ao final do capítulo, aplica-se estes conceitos no desenvolvimento do modelo cinemático direto e diferencial direto do robô ASEA IRBL6 e do sistema "robô ASEA IRBL6 + trilho".

CAPÍTULO 3. MÉTODOS DE NEWTON-RAPHSON

Faz-se uma revisão bibliográfica sobre métodos baseados no algoritmo de *Newton-Raphson* para solução de sistemas de equações não lineares e sobre métodos que necessitam inverter matrizes, apresenta-se o método de *Newton-Raphson*, deduz-se expressões que mostram o seu significado geométrico, propõe-se uma alteração no método de *Newton-Raphson* que aumenta a sua eficiência em casos de trajetória (chama-se este novo algoritmo de método de *Newton-Raphson* adaptado) e, finalmente, mostra-se resultados de simulações dos métodos vistos, para o robô ASEA IRBL6.

CAPÍTULO 4. MÉTODOS BASEADOS NA TRANSPOSTA DO JACOBIANO

Faz-se uma revisão bibliográfica sobre métodos baseados na transposta do Jacobiano, apresenta-se os métodos que surgiram a partir das teorias de *"impedance control"* [ASA85, 2HO85] (para ponto-fixo) e *"sliding mode control"* [NOV90] (para trajetória), mostra-se as equações de um terceiro método [BAL84,SCI88], para trajetória, e, finalmente, apresenta-se resultados de simulações dos métodos vistos, para o robô ASEA IRBL6.

CAPÍTULO 5. CINEMÁTICA INVERSA DE MANIPULADORES REDUNDANTES

Como colocado acima, manipuladores redundantes possuem mais de um vetor $\boldsymbol{q}$ correspondente a um vetor $\boldsymbol{y}$ dado. Assim, quando se faz a cinemática inversa destes manipuladores, procura-se a solução que otimize o desempenho do robô segundo algum critério. Na literatura, os critérios mais utilizados para otimização são: procurar soluções nas quais as coordenadas de junta estejam dentro de determinada faixa; procurar soluções que evitem obstáculos; procurar soluções que evitem posições de singularidade.

Neste capítulo, apresenta-se o método de *Sciavicco e Siciliano* [SCI88], que resolve a cinemática inversa de manipuladores redundantes selecionando soluções que satisfaçam um dos dois primeiros critérios colocados acima. Em seguida propõe-se utilizar este algoritmo em conjunto com o de *Newton-Raphson*, de forma a gerar um novo método, mais eficiente que o original (método híbrido). Por fim, apresenta-se resultados de simulações do método híbrido para o sistema "robô ASEA IRBL6 + trilho".

CAPÍTULO 6. SIMULADOR CINEMÁTICO

Neste capítulo aplica-se o método híbrido, do capítulo anterior, no desenvolvimento de um simulador cinemático, nos moldes colocados em 1.1, para o sistema "robô ASEA IRBL6 + trilho". Com isto propõe-se uma forma de solucionar um problema freqüentemente encontrado em células e sistemas flexíveis de manufatura, ou seja, *quando se adapta um trilho a um robô de 6 graus de liberdade visando aumentar o seu volume de trabalho, como aproveitar a introdução do trilho para gerar trajetórias otimizadas no espaço de juntas (capítulo 5), dada uma trajetória em termos de coordenadas de posição e orientação do efetuador?*

Capítulo 2

ASPECTOS CINEMÁTICOS DE ROBÔS

2.1 INTRODUÇÃO

A cinemática de robôs estuda a geometria dos mesmos durante o seu movimento sem levar em conta os esforços que geram este movimento e que são gerados por ele. Para este estudo é necessário que se faça a modelagem cinemática do manipulador. Realizar esta modelagem significa equacionar a geometria do robô no decorrer do tempo em relação a um sistema de coordenadas fixo. Esta tarefa, entretanto, resulta em equações cuja complexidade cresce bastante com o número de graus de liberdade do robô. Visando simplificar este trabalho, surgiram metodologias de representação e equacionamento de estruturas robóticas.

Neste capítulo apresentam-se conceitos, notações, formulações matemáticas e procedimentos utilizados na modelagem cinemática de manipuladores e necessários para a compreensão dos capítulos posteriores. Feito isto, coloca-se o problema da cinemática inversa de manipuladores de cadeia aberta e introduz-se a necessidade de métodos iterativos para solucionar este problema. Ao final, modela-se o sistema "robô ASEA IRBL6 + trilho".

2.2 NOMENCLATURA

A estrutura mecânica de um robô é chamada de manipulador. Um manipulador é formado por uma seqüência de corpos rígidos aqui tratados por ligamentos. Estes são conectados entre si por vínculos, aqui denominados juntas, que impedem a movimentação (linear e/ou angular) relativa entre os ligamentos segundo certas direções (o vínculo que impede qualquer movimento entre dois ligamentos chama-se engastamento e aqui este não é considerado uma junta).

Estruturas robóticas podem ser modeladas como um conjunto de juntas prismáticas e/ou de revolução. Juntas prismáticas permitem a translação ao longo de um único eixo e impedem a rotação em torno de qualquer eixo. Juntas de revolução permitem a rotação em torno de um único eixo e impedem a translação ao longo de qualquer eixo. A cada par junta/ligamento (prismática ou de revolução) está associado um grau de liberdade do robô. Desta forma, em um robô de n graus de liberdade existem n conjuntos ligamento/junta, numerados de 1 a n.

Aqui, considera-se um ligamento 0 que é tratado por base do robô. Esta não é considerada parte do robô e sim um elo entre este e o ambiente. À base costuma-se associar um sistema de coordenadas (sistema de coordenadas fixo) em relação ao qual são derivadas as equações de movimento do robô.

O último ligamento será tratado por efetuador. É nele que estará situada a garra do robô. Ao conjunto de todas as posições que podem ser atingidas por determinado ponto do efetuador dá-se o nome de volume de trabalho do robô relativo a este ponto.

Os manipuladores que são estudados neste trabalho, possuem as três características citadas abaixo:

- Todas as juntas deste manipulador conectam somente dois ligamentos;
- Possuem dois e somente dois ligamentos (base e efetuador) que se unem a somente uma junta cada;
- Um destes dois ligamentos (efetuador) não se une a nenhum engastamento.

A esta classe de manipuladores dá-se o nome de manipuladores de cadeia aberta[1].

2.3 POSIÇÃO E ORIENTAÇÃO

Neste ítem são apresentadas formas de descrição da posição e orientação de um corpo rígido no espaço em relação a um sistema de coordenadas fixo. No ítem 2.6 conceitos vistos aqui são utilizados para descrever a posição e orientação do efetuador de um manipulador em relação ao sistema de coordenadas da base.

De modo geral, para se descrever a posição e orientação de um corpo rígido livre no espaço, o número mínimo de parâmetros independentes necessários são seis, três para posição e três para orientação. Em casos particulares, pode ser necessário um número menor de parâmetros. Desta forma, se o corpo pode ser representado no plano XY, por exemplo, um parâmetro de orientação (ângulo em torno do eixo Z) e dois de posição

[1] A literatura é falha na definição de manipuladores de cadeia aberta. *Asada* [ASA85] define estes manipuladores como sendo aqueles que possuem "estrutura aberta", mas não define este termo; *Renaud* [GOR84], utiliza ilustrações para descrever o tipo de estrutura, mas não as define; *Fu* e *Paul* [FU87,PAU81] não definem manipuladores de cadeia aberta.

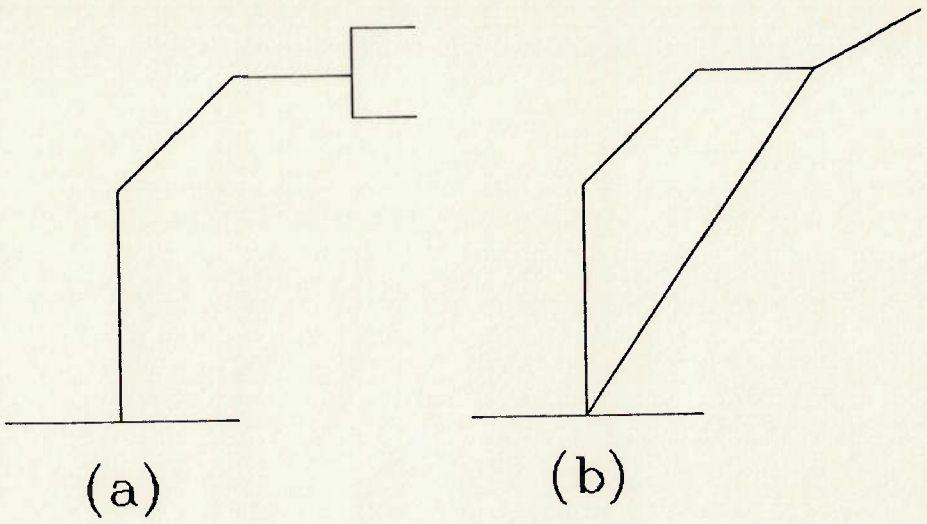


Figura 2.1: Manipuladores de cadeia aberta (a) e de cadeia fechada (b)

(coordenada x e y) definem completamente a localização do corpo em relação a um sistema fixo $OXYZ$; se o corpo possui vínculos que impedem o movimento em determinadas direções, pode-se descrever a localização do corpo com um número menor do que seis parâmetros. *Denavit* e *Hartenberg* (apresentado detalhadamente no ítem 2.5) propuseram quatro parâmetros para descrever a posição e orientação de um ligamento vinculado por juntas prismáticas ou de revolução em relação a um sistema de coordenadas fixo a uma das juntas do ligamento. O conjunto dos parâmetros de posição e de orientação escolhidos para descrever a localização do efetuador em relação ao sistema de coordenadas da base de um robô chama-se Coordenadas Operacionais.

2.3.1 POSIÇÃO

Os conjuntos mais usuais de parâmetros necessários para descrição da posição de um ponto em relação a um sistema de coordenadas fixo são (Fig. 2.2):

- Coordenadas cartesianas do ponto

$$[x,y,z]^t$$

- Coordenadas cilíndricas do ponto

$$[\rho,\theta,z]^t$$

- Coordenadas esféricas do ponto

$$[r,\theta,\phi]^t$$

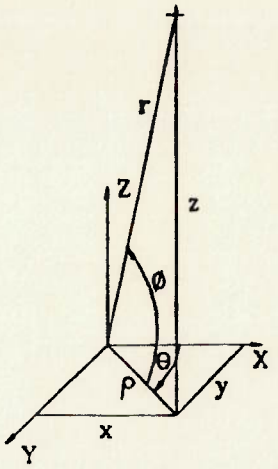


Figura 2.2: Representação da posição de um ponto em rel. a um sist. fixo

O robô ASEA IRBL6, assim como a grande maioria dos robôs industriais, utiliza coordenadas cartesianas para descrever a posição de seu efetuador em relação ao sistema da base. Desta forma, os modelos cinemáticos deste trabalho foram desenvolvidos em coordenadas cartesianas.

2.3.2 ORIENTAÇÃO

Apesar de três parâmetros definirem completamente a orientação de um sistema de coordenadas em relação a um sistema fixo, em muitos casos é mais vantajoso utilizar mais do que três parâmetros. Os conjuntos mais usuais de parâmetros de orientação são:

1 Roll, pitch, yaw: Para se obter o sistema de coordenadas $i+1$ a partir do sistema i, gira-se o primeiro de um ângulo θ_z ao redor do eixo $\boldsymbol{Z}$; o sistema resultante gira-se de um ângulo θ_y ao redor do novo eixo $\boldsymbol{Y}$; finalmente, gira-se o novo sistema ao redor do novo eixo $\boldsymbol{X}$ de um ângulo θ_x (Fig. 2.3).

$$[\theta_z,\theta_y,\theta_x]^t$$

Note que alterando a ordem de rotação, muda o sistema $i+1$ resultante.

2 Ângulos de Euler: Para se obter o sistema de coordenadas $i+1$ a partir do sistema i, gira-se o primeiro de um ângulo θ_{z1} ao redor do eixo $\boldsymbol{Z}$; o sistema resultante gira-se de um ângulo θ_{x1} ao redor do novo eixo $\boldsymbol{X}$; finalmente, gira-se o novo sistema ao redor do novo eixo $\boldsymbol{Z}$ de um ângulo θ_{z2} (Fig. 2.4).

$$[\theta_{z1},\theta_{x1},\theta_{z2}]^t$$

Novamente, a rotação em outra ordem nos leva a outro sistema de coordenadas $i+1$.

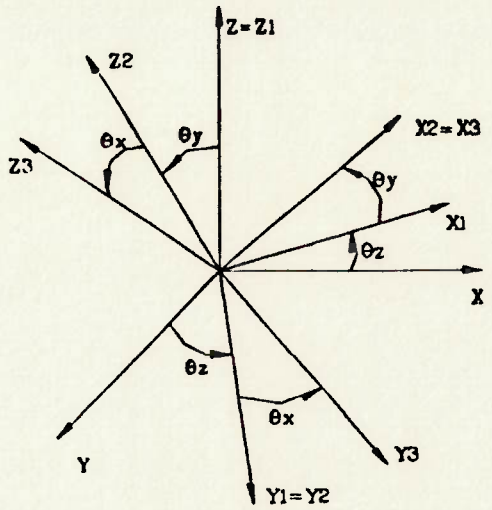


Figura 2.3: Ângulos roll, pitch, yaw

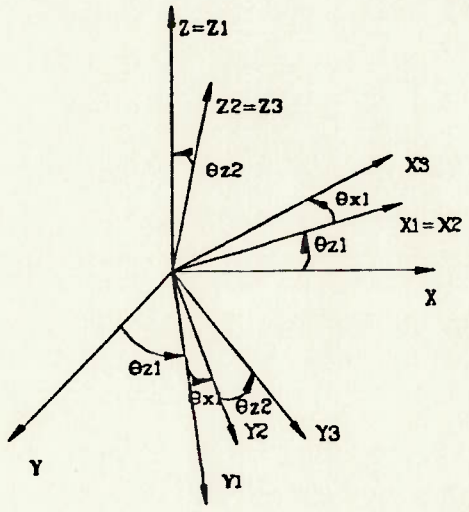


Figura 2.4: Ângulos de Euler

3 Quaternion: É uma entidade matemática construída a partir de um número real θ e de um vetor $\boldsymbol{n}$ de três componentes. Pode-se escrever um quaternion Q da seguinte forma [2]:

$Q = (\theta + \boldsymbol{n})$

Q pode ser utilizado para descrever uma rotação em torno de $\boldsymbol{n}$ de um ângulo θ. A orientação de um sistema $i+1$ em relação a um sistema i pode, então, ser dada pelo quaternion $\theta+\boldsymbol{n}$. O vetor nos indica em torno de qual eixo o sistema i deve girar para chegar ao sistema $i+1$, θ nos indica o ângulo de rotação em torno do vetor $\boldsymbol{n}$.

Seja $[Q_1, Q_2, Q_3, Q_4]^t$, um vetor que contém as quatro componentes de um quaternion. Se estas componentes descrevem a rotação de um ângulo θ em torno de um vetor $\boldsymbol{n}$ de módulo 1 e componentes $[n_x, n_y, n_z]^t$ utilizando os parâmetros de *Euler-Rodrigues*, este quaternion possui uma propriedade bastante útil, (demonstrada pelo matemático francês *Olinde Rodrigues* em 1840), que justifica a utilização destes parâmetros:

Parâmetros de *Euler-Rodrigues*

$Q_1 = n_x.sin(\theta/2)$

$Q_2 = n_y.sin(\theta/2)$

$Q_3 = n_z.sin(\theta/2)$

$Q_4 = cos(\theta/2)$

Propriedade: Quando um corpo rígido sofre uma rotação composta por duas rotações finitas, a primeira definida pelo quaternion Q e a segunda pelo quaternion Q_{int}, a rotação resultante é dada pelo quaternion Q_f abaixo:

$$Q_f = Q \star Q_{int} \tag{2.1}$$

onde

$$\begin{aligned} Q &= [Q_1, Q_2, Q_3, Q_4]^t \\ Q_{int} &= [Q_{1int}, Q_{2int}, Q_{3int}, Q_{4int}]^t \\ Q_f &= [Q_{1f}, Q_{2f}, Q_{3f}, Q_{4f}]^t \end{aligned}$$

e

$$\begin{aligned} Q_1 &= n_x.sin(\theta/2) \\ Q_2 &= n_y.sin(\theta/2) \\ Q_3 &= n_z.sin(\theta/2) \\ Q_4 &= cos(\theta/2) \end{aligned}$$

[2] Outra forma usual de se lidar com quaternions é pensar nos mesmos como um número complexo com três partes imaginárias e uma parte real [FU87].

$$
\begin{aligned}
Q_{1int} &= n_{xint}.sin(\theta_{int}/2) \\
Q_{2int} &= n_{yint}.sin(\theta_{int}/2) \\
Q_{3int} &= n_{zint}.sin(\theta_{int}/2) \\
Q_{4int} &= cos(\theta_{int}/2)
\end{aligned}
$$

$$
\begin{aligned}
Q_{1f} &= n_{xf}.sin(\theta_f/2) \\
Q_{2f} &= n_{yf}.sin(\theta_f/2) \\
Q_{3f} &= n_{zf}.sin(\theta_f/2) \\
Q_{4f} &= cos(\theta_f/2)
\end{aligned}
$$

$$Q_{1f} = Q_4.Q_{1int} + Q_1.Q_{4int} - Q_2.Q_{3int} + Q_3.Q_{2int} \quad (2.2)$$

$$Q_{2f} = Q_4.Q_{2int} + Q_1.Q_{3int} + Q_2.Q_{4int} - Q_3.Q_{1int} \quad (2.3)$$

$$Q_{3f} = Q_4.Q_{3int} - Q_1.Q_{2int} + Q_2.Q_{1int} + Q_3.Q_{4int} \quad (2.4)$$

$$Q_{4f} = Q_4.Q_{4int} - Q_1.Q_{1int} - Q_2.Q_{2int} - Q_3.Q_{3int} \quad (2.5)$$

ou, chamando de $\boldsymbol{k}$ o vetor formado pelas três primeiras componentes de Q e de x o número real correspondente à quarta componente de Q, pode-se escrever as expressões acima na forma vetorial, conforme segue:

$$Q_f = Q \star Q_{int} = x.x_{int} - \boldsymbol{k}.\boldsymbol{k}_{int} + x.\boldsymbol{k}_{int} + x_{int}.\boldsymbol{k} + \boldsymbol{k} \times \boldsymbol{k}_{int} \quad (2.6)$$

À regra "$\star$", dá-se o nome de produto de quaternions. A partir desta regra pode-se encontrar Q^{-1} (quaternion inverso) que satisfaz a equação abaixo (esta equação é válida mesmo para quaternions que não são formados por parâmetros de *Euler-Rodrigues*, uma vez que para estes se define a mesma regra "$\star$"):

$$Q \star Q^{-1} = Q^{-1} \star Q = 1$$

$$Q^{-1} = (x - \boldsymbol{k})/(x^2 + \boldsymbol{k}.\boldsymbol{k}) \quad (2.7)$$

Quando um quaternion satisfaz a seguinte relação, este recebe o nome de quaternion normalizado (é o caso dos quaternions cujos elementos são parâmetros de *Euler-Rodrigues*):

$$Q_1^2 + Q_2^2 + Q_3^2 + Q_4^2 = 1 \quad (2.8)$$

Note que o denominador da expressão (2.7) vale 1 para quaternions normalizados. Assim, para estes, $Q^{-1} = x - \boldsymbol{k}$.

Quaternions são muito utilizados em robôs industriais (o robô ASEA IRBL6 é um exemplo). Isto se deve à facilidade que os mesmos fornecem para interpolar orientações intermediárias durante uma rotação do robô. Quando se deseja, por exemplo, promover a rotação ao redor de um vetor $\boldsymbol{n}$ de um ângulo θ, as orientações intermediárias podem ser dadas pelos quaternions definidos pelo vetor $\boldsymbol{n}$ e pelos ângulos que vão de 0 a θ.

Já que a teoria de quaternions ganha força quando utilizada com os parâmetros de *Euler-Rodrigues*, e como quaternions são muito utilizados em robôs industriais, inclusive no robô ASEA IRBL6, optamos, neste trabalho, por utilizar estes parâmetros para a construção dos modelos cinemáticos do robô e do sistema "robô + trilho".

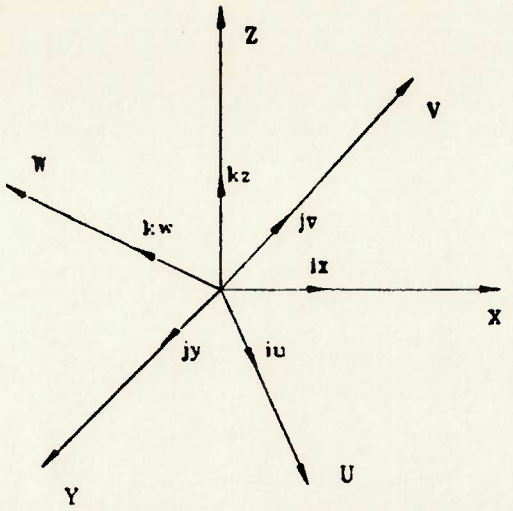


Figura 2.5: Rotação entre sistemas de coordenadas

4 Parâmetros de rotação finita: Quaternions normalizados, como colocado acima, são relacionados pela equação (2.8). Aqui, apresenta-se um conjunto de três parâmetros que utilizam o vetor $\boldsymbol{n}$ e o ângulo θ, como acontece com quaternions, mas são independentes entre si.

$$[P_1,P_2,P_3]^t$$

P_1,P_2 e P_3 são dados por:

$P_1 = n_x.tan(\theta/2)$

$P_2 = n_y.tan(\theta/2)$

$P_3 = n_z.tan(\theta/2)$

Note que:

$P_1 = Q_1/Q_4$

$P_2 = Q_2/Q_4$

$P_3 = Q_3/Q_4$

5 Matriz de Rotação: Trata-se de uma forma matricial de representar a orientação entre dois sistemas de coordenadas $OXYZ$ e $OUVW$. Permite, através da multiplicação de uma matriz por um vetor posição $\boldsymbol{p}$, determinar as componentes deste vetor escritas em relação ao sistema $OXYZ$ dadas as suas componentes escritas em relação ao sistema $OUVW$.

Sejam os sistemas $OXYZ$ e $OUVW$ (Fig. 2.5) com versores, respectivamente, $\boldsymbol{i_x}$,$\boldsymbol{j_y}$,$\boldsymbol{k_z}$e $\boldsymbol{i_u}$,$\boldsymbol{j_v}$,$\boldsymbol{k_w}$, escritos em relação a $OXYZ$. Seja, ainda, o vetor posição $\boldsymbol{p}$ com componentes $\boldsymbol{p}_{xyz}$= $[p_x,p_y,p_z]^t$ em relação a $OXYZ$ e $\boldsymbol{p}_{uvw}$= $[p_u,p_v,p_w]^t$ em relação a $OUVW$. Considere-se os seguintes problemas:

É possível encontrar uma matriz $\boldsymbol{R}$ que satisfaça a relação abaixo?

$$\boldsymbol{p}_{xyz} = \boldsymbol{R}.\boldsymbol{p}_{uvw} \tag{2.9}$$

Se possível quais as componentes desta matriz?

Para resolver estas questões, parte-se da equação

$$\boldsymbol{p}_{uvw} = p_u.\boldsymbol{i}_u + p_v.\boldsymbol{j}_v + p_w.\boldsymbol{k}_w \tag{2.10}$$

e lembra-se que

$$\begin{aligned} p_x &= \boldsymbol{i}_x.\boldsymbol{p}_{uvw} \\ p_y &= \boldsymbol{j}_y.\boldsymbol{p}_{uvw} \\ p_z &= \boldsymbol{k}_z.\boldsymbol{p}_{uvw} \end{aligned} \tag{2.11}$$

ou seja:

$$\begin{aligned} p_x &= \boldsymbol{i}_x.\boldsymbol{i}_u.p_u + \boldsymbol{i}_x.\boldsymbol{j}_v.p_v + \boldsymbol{i}_x.\boldsymbol{k}_w.p_w \\ p_y &= \boldsymbol{j}_y.\boldsymbol{i}_u.p_u + \boldsymbol{j}_y.\boldsymbol{j}_v.p_v + \boldsymbol{j}_y.\boldsymbol{k}_w.p_w \\ p_z &= \boldsymbol{k}_z.\boldsymbol{i}_u.p_u + \boldsymbol{k}_z.\boldsymbol{j}_v.p_v + \boldsymbol{k}_z.\boldsymbol{k}_w.p_w \end{aligned} \tag{2.12}$$

Escrevendo na forma matricial, tem-se:

$$\begin{bmatrix} p_x \\ p_y \\ p_z \end{bmatrix} = \begin{bmatrix} \boldsymbol{i}_x.\boldsymbol{i}_u & \boldsymbol{i}_x.\boldsymbol{j}_v & \boldsymbol{i}_x.\boldsymbol{k}_w \\ \boldsymbol{j}_y.\boldsymbol{i}_u & \boldsymbol{j}_y.\boldsymbol{j}_v & \boldsymbol{j}_y.\boldsymbol{k}_w \\ \boldsymbol{k}_z.\boldsymbol{i}_u & \boldsymbol{k}_z.\boldsymbol{j}_v & \boldsymbol{k}_z.\boldsymbol{k}_w \end{bmatrix} \begin{bmatrix} p_u \\ p_v \\ p_w \end{bmatrix} \tag{2.13}$$

Observando a expressão (2.9), percebe-se que

$$\boldsymbol{R} = \begin{bmatrix} \boldsymbol{i}_x.\boldsymbol{i}_u & \boldsymbol{i}_x.\boldsymbol{j}_v & \boldsymbol{i}_x.\boldsymbol{k}_w \\ \boldsymbol{j}_y.\boldsymbol{i}_u & \boldsymbol{j}_y.\boldsymbol{j}_v & \boldsymbol{j}_y.\boldsymbol{k}_w \\ \boldsymbol{k}_z.\boldsymbol{i}_u & \boldsymbol{k}_z.\boldsymbol{j}_v & \boldsymbol{k}_z.\boldsymbol{k}_w \end{bmatrix} \tag{2.14}$$

Nas linhas abaixo coloca-se algumas propriedades da matriz de rotação $\boldsymbol{R}$.

1. As três colunas de $\boldsymbol{R}$ possuem, respectivamente, as componentes dos vetores $\boldsymbol{i}_u$, $\boldsymbol{j}_v$ e $\boldsymbol{k}_w$ escritos em relação ao sistema $OXYZ$. Esta propriedade é facilmente verificada lembrando que o produto escalar de dois vetores $\boldsymbol{u}.\boldsymbol{v}$ nos dá a projeção de $\boldsymbol{v}$ em $\boldsymbol{u}$. Assim, as três linhas da primeira coluna de $\boldsymbol{R}$, por exemplo, nos informa as projeções (coordenadas) de $\boldsymbol{i}_u$ nos eixos x, y e z, respectivamente.
2. $\boldsymbol{R}^{-1} = \boldsymbol{R}^t$

   Para chegar a este resultado, basta notar que:

$$\begin{aligned} p_u &= \boldsymbol{i}_u.\boldsymbol{p}_{xyz} \\ p_v &= \boldsymbol{j}_v.\boldsymbol{p}_{xyz} \\ p_w &= \boldsymbol{k}_w.\boldsymbol{p}_{xyz} \end{aligned} \tag{2.15}$$

   ou seja:

$$\begin{aligned} p_u &= \boldsymbol{i}_u.\boldsymbol{i}_x.p_x + \boldsymbol{i}_u.\boldsymbol{j}_y.p_y + \boldsymbol{i}_u.\boldsymbol{k}_z.p_z \\ p_v &= \boldsymbol{j}_v.\boldsymbol{i}_x.p_x + \boldsymbol{j}_v.\boldsymbol{j}_y.p_y + \boldsymbol{j}_v.\boldsymbol{k}_z.p_z \\ p_w &= \boldsymbol{k}_w.\boldsymbol{i}_x.p_x + \boldsymbol{k}_w.\boldsymbol{j}_y.p_y + \boldsymbol{k}_w.\boldsymbol{k}_z.p_z \end{aligned} \tag{2.16}$$

A matriz de rotação resultante é a inversa de $\boldsymbol{R}$:

$$\boldsymbol{R}^{-1} = \begin{bmatrix} \boldsymbol{i}_u.\boldsymbol{i}_x & \boldsymbol{i}_u.\boldsymbol{j}_y & \boldsymbol{i}_u.\boldsymbol{k}_z \\ \boldsymbol{j}_v.\boldsymbol{i}_x & \boldsymbol{j}_v.\boldsymbol{j}_y & \boldsymbol{j}_v.\boldsymbol{k}_z \\ \boldsymbol{k}_w.\boldsymbol{i}_x & \boldsymbol{k}_w.\boldsymbol{j}_y & \boldsymbol{k}_w.\boldsymbol{k}_z \end{bmatrix} \tag{2.17}$$

Assim,

$$\boldsymbol{R}^{-1} = \boldsymbol{R}^t \tag{2.18}$$

Matrizes que satisfazem a relação acima são ditas ortonormais. Para estas, os vetores colunas são perpendiculares entre si (produto escalar vale zero) e o módulo destes vetores vale 1.

3. Para rotações finitas, o produto de matrizes de rotação não é comutativo. Abaixo ilustra-se esta afirmação multiplicando-se as matrizes de rotação de θ_x em torno do eixo $\boldsymbol{X}$ ($\boldsymbol{R}_{\theta_x,x}$) e θ_z em torno de $\boldsymbol{Z}$ ($\boldsymbol{R}_{\theta_z,z}$) nas duas ordens possíveis. Para se encontrar $\boldsymbol{R}_{\theta_x,x}$ e $\boldsymbol{R}_{\theta_z,z}$ basta fazer, respectivamente, $\boldsymbol{i}_x$=$\boldsymbol{i}_u$ e $\boldsymbol{k}_z$=$\boldsymbol{k}_w$ em 2.14.
Assim, tem-se:

$$\boldsymbol{R}_{\theta_x,x} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & cos(\theta_x) & -sin(\theta_x) \\ 0 & sin(\theta_x) & cos(\theta_x) \end{bmatrix} \tag{2.19}$$

e

$$\boldsymbol{R}_{\theta_z,z} = \begin{bmatrix} cos(\theta_z) & -sin(\theta_z) & 0 \\ sin(\theta_z) & cos(\theta_z) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2.20}$$

$\boldsymbol{R}_{\theta_x,x}.\boldsymbol{R}_{\theta_z,z}$ vale:

$$\boldsymbol{R}_{\theta_x,x}.\boldsymbol{R}_{\theta_z,z} = \begin{bmatrix} cos(\theta_z) & -sin(\theta_z) & 0 \\ sin(\theta_z).cos(\theta_x) & cos(\theta_z).cos(\theta_x) & -sin(\theta_x) \\ sin(\theta_z).sin(\theta_x) & cos(\theta_z).sin(\theta_x) & cos(\theta_x) \end{bmatrix}$$

e $\boldsymbol{R}_{\theta_z,z}.\boldsymbol{R}_{\theta_x,x}$ vale:

$$\boldsymbol{R}_{\theta_z,z}.\boldsymbol{R}_{\theta_x,x} = \begin{bmatrix} cos(\theta_z) & -sin(\theta_z).cos(\theta_x) & sin(\theta_z).sin(\theta_x) \\ sin(\theta_z) & cos(\theta_z).cos(\theta_x) & -cos(\theta_z).sin(\theta_x) \\ 0 & sin(\theta_x) & cos(\theta_x) \end{bmatrix}$$

Perceba que $\boldsymbol{R}_{\theta_x,x}.\boldsymbol{R}_{\theta_z,z} \neq \boldsymbol{R}_{\theta_z,z}.\boldsymbol{R}_{\theta_x,x}$. Para rotações infinitesimais isto não ocorre. Desta forma, $\boldsymbol{R}_{d\theta_x,x}.\boldsymbol{R}_{d\theta_z,z}$ vale:

$$\boldsymbol{R}_{d\theta_x,x}.\boldsymbol{R}_{d\theta_z,z} = \begin{bmatrix} cos(d\theta_z) & -sin(d\theta_z) & 0 \\ sin(d\theta_z).cos(d\theta_x) & cos(d\theta_z).cos(d\theta_x) & -sin(d\theta_x) \\ sin(d\theta_z).sin(d\theta_x) & cos(d\theta_z).sin(d\theta_x) & cos(d\theta_x) \end{bmatrix}$$

e $\boldsymbol{R}_{d\theta_z,z}.\boldsymbol{R}_{d\theta_x,x}$ vale:

$$\boldsymbol{R}_{d\theta_z,z}.\boldsymbol{R}_{d\theta_x,x} = \begin{bmatrix} cos(d\theta_z) & -sin(d\theta_z).cos(d\theta_x) & sin(d\theta_z).sin(d\theta_x) \\ sin(d\theta_z) & cos(d\theta_z).cos(d\theta_x) & -cos(d\theta_z).sin(d\theta_x) \\ 0 & sin(d\theta_x) & cos(d\theta_x) \end{bmatrix}$$

Fazendo $sin(d\theta_z) \simeq d\theta_z$, $sin(d\theta_x) \simeq d\theta_x$, $cos(d\theta_z) \simeq cos(d\theta_x) \simeq 1$ e $d\theta_x.d\theta_z \simeq 0$, resulta

$$\boldsymbol{R}_{d\theta_x,x}.\boldsymbol{R}_{d\theta_z,z} = \boldsymbol{R}_{d\theta_z,z}.\boldsymbol{R}_{d\theta_x,x} = \begin{bmatrix} 1 & -d\theta_z & 0 \\ d\theta_z & 1 & -d\theta_x \\ 0 & d\theta_x & 1 \end{bmatrix}$$

A expressão para uma rotação diferencial genérica é obtida multiplicando-se $\boldsymbol{R}_{d\theta_x,x}.\boldsymbol{R}_{d\theta_y,y}.\boldsymbol{R}_{d\theta_z,z}$:

$$\boldsymbol{R}_{d\theta_x,x}.\boldsymbol{R}_{d\theta_y,y}.\boldsymbol{R}_{d\theta_z,z} = \begin{bmatrix} 1 & -d\theta_z & d\theta_y \\ d\theta_z & 1 & -d\theta_x \\ -d\theta_y & d\theta_x & 1 \end{bmatrix}$$

2.4 TRANSFORMAÇÕES HOMOGÊNEAS

Trata-se de um método que permite descrever rotações e translações entre dois sistemas de coordenadas, utilizando uma única matriz ($\boldsymbol{T}$).

Sejam os sistemas $OXYZ$ e $OUVW$ com matriz de rotação $\boldsymbol{R}$ (conforme ítem anterior). A origem do sistema $OUVW$ possui componentes $[O_{ux}, O_{vy}, O_{wz}]^t$ em relação a $OXYZ$. Seja, ainda, o vetor posição $\boldsymbol{p}$ com componentes $\boldsymbol{p}_{xyz} = [p_x, p_y, p_z]^t$ em relação a $OXYZ$ e $\boldsymbol{p}_{uvw} = [p_u, p_v, p_w]^t$ em relação a $OUVW$. A matriz de transformação homogênea associada a transformação que nos leva de $\boldsymbol{p}_{uvw} = [p_u, p_v, p_w]^t$ a $\boldsymbol{p}_{xyz} = [p_x, p_y, p_z]^t$ é:

$$\boldsymbol{T}_{uvw}^{xyz} = \begin{bmatrix} \boldsymbol{R}_{3\times3} & [O_{ux}, O_{vy}, O_{wz}]^t_{3\times1} \\ 0_{1\times3} & 1_{1\times1} \end{bmatrix}$$

Ou seja:

$$\begin{bmatrix} p_x \\ p_y \\ p_z \\ 1 \end{bmatrix} = \begin{bmatrix} \boldsymbol{R}_{3\times3} & [O_{ux}, O_{vy}, O_{wz}]^t_{3\times1} \\ 0_{1\times3} & 1_{1\times1} \end{bmatrix} \cdot \begin{bmatrix} p_u \\ p_v \\ p_w \\ 1 \end{bmatrix}$$

2.5 NOTAÇÃO DE DENAVIT-HARTENBERG

Denavit e *Hartenberg* [HAR64] propuseram uma metodologia para representar uma estrutura robótica que permite, de uma forma sistemática, estabelecer sistemas de coordenadas para cada ligamento da estrutura. Esta metodologia torna-se uma forma eficiente de descrição de translações e rotações entre dois ligamentos vizinhos, quando utilizada com transformações homogêneas de coordenadas.

Denavit e *Hartenberg* utilizam, para o estabelecimento dos sistemas de coordenadas, 4 regras básicas e, para a descrição das translações e rotações entre sistemas vizinhos, 4 parâmetros.

Regras:

- A origem do sistema de coordenadas i localiza-se na intersecção entre o eixo da junta $i+1$ e a normal comum entre os eixos das juntas [3] i e $i+1$ (Se a junta é prismática, o eixo da junta está na direção do movimento, se é de revolução, o eixo da junta é perpendicular ao plano de rotação);
  - Se os eixos das juntas i e $i+1$ são paralelos, a origem do sistema i pode ser escolhida em qualquer ponto do eixo da junta $i+1$;
  - A origem do sistema de coordenadas 0, pode ser escolhida em qualquer ponto do eixo da junta 1;
  - A origem do último sistema de coordenadas, pode ser escolhida em qualquer ponto do efetuador;
- O eixo $\boldsymbol{Z}_{i-1}$ está na direção do eixo da junta i. A sua orientação é escolhida arbitrariamente;
  - A direção do último eixo $\boldsymbol{Z}_i$ pode ser escolhida arbitrariamente, devendo, somente, ser perpendicular ao eixo $\boldsymbol{X}_i$;
- O eixo $\boldsymbol{X}_i$ está na direção da normal comum entre os eixos $\boldsymbol{Z}_{i-1}$ e $\boldsymbol{Z}_i$. A sua orientação está no sentido que vai da intersecção entre esta normal comum e o eixo $\boldsymbol{Z}_{i-1}$ para a intersecção entre a normal comum e o eixo $\boldsymbol{Z}_i$;
  - Se os eixos das juntas i e $i+1$ são perpendiculares, a orientação de $\boldsymbol{X}_i$ é arbitrária;
  - O último eixo $\boldsymbol{X}_n$ deve ser perpendicular ao eixo $\boldsymbol{Z}_{n-1}$, passando pela origem O_n. A orientação deve ser no sentido da perpendicular comum aos eixos $\boldsymbol{X}_n$ e $\boldsymbol{Z}_{n-1}$ à origem O_n. Se $\boldsymbol{Z}_{n-1}$ passa por O_n, a direção de $\boldsymbol{X}_n$ é arbitrária;
  - A direção de $\boldsymbol{X}_0$ pode ser escolhida arbitrariamente, devendo, somente, ser perpendicular a $\boldsymbol{Z}_0$;
- O eixo $\boldsymbol{Y}_{i-1}$ é encontrado multiplicando-se $\boldsymbol{Z}_{i-1}$ por $\boldsymbol{X}_{i-1}$ ($\boldsymbol{Y}_{i-1} = \boldsymbol{Z}_{i-1} \times \boldsymbol{X}_{i-1}$).

Parâmetros:

θ_i É o ângulo formado entre os eixos $\boldsymbol{X}_{i-1}$ e $\boldsymbol{X}_i$, sendo positivo quando a rotação de $\boldsymbol{X}_{i-1}$ para $\boldsymbol{X}_i$ se dá no sentido positivo do eixo $\boldsymbol{Z}_{i-1}$;

[3] Existe, na literatura, variações da proposta de *Denavit* e *Hartenberg* [HAR64]. *Renaud* [GOR84], por exemplo, coloca o sistema de coordenadas i na intersecção entre o eixo da junta i (não $i+1$) e a normal comum entre os eixos das juntas i e $i+1$.

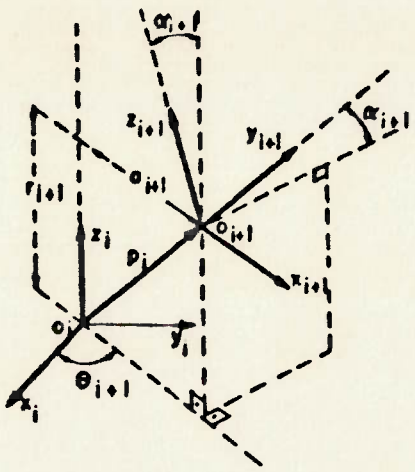


Figura 2.6: Sistema $i+1$ a partir do sistema i e parâmetros α_{i+1} , θ_{i+1} , r_{i+1} , a_{i+1} .

r_i O seu módulo [4] é a distância entre a origem do sistema de coordenadas $i-1$ e a intersecção entre os eixos Z_{i-1} e X_i . Se a perpendicular comum entre os eixos Z_{i-1} e X_i estiver do lado positivo de Z_{i-1} , r_i é positivo, caso contrário é negativo;

a_i É a distância [5] entre a intersecção dos eixos Z_{i-1} e X_i e a origem do sistema de coordenadas i (é a menor distância entre os eixos Z_{i-1} e Z_i);

α_i É o ângulo formado entre os eixos Z_{i-1} e Z_i , sendo positivo quando a rotação de Z_{i-1} para Z_i se dá no sentido positivo do eixo X_i .

Observações:

1. Se a junta i é de revolução, o parâmetro θ_i é variável e r_i é constante. Neste caso, θ_i é chamado de coordenada de junta;
2. Se a junta i é prismática, o parâmetro r_i é variável e θ_i é constante.Neste caso, r_i é chamado de coordenada de junta;

Ao conjunto de todas as coordenadas de junta do manipulador, chama-se coordenadas generalizadas de junta.

Com estas regras e parâmetros, dado um sistema de coordenadas i, pode-se construir o sistema $i+1$, como mostrado na Fig. 2.6.

Abaixo, coloca-se um algoritmo apresentado por *Fu* [FU87] que permite estabelecer os sistemas de coordenadas de cada junta de um manipulador genérico e os parâmetros de *Denavit-Hartenberg* associados a estes sistemas.

1. Estabelecer um sistema de coordenadas ortonormal fixo à base do robô (X_o,Y_o,Z_o) com o eixo Z_o na direção do eixo da junta 1. Os eixos X_o e Y_o podem ser estabelecidos convenientemente;

[4] Na literatura, muitas vezes, não se menciona que r_i pode ser negativo [ASA85,FU87].
[5] Se a orientação de X_i for desobedecida, a_i pode ser negativo [FU87].

junta	a_i	r_i	α_i	θ_i	valor de a_i e r_i
1	0	r_1	$\pi/2$	q_1	r_1= 0.700 m
2	a_2	0	0	q_2	a_2= 0.690 m
3	a_3	0	0	q_3	a_3= 0.670 m
4	0	0	$\pi/2$	q_4	
5	0	r_5	-$\pi/2$	q_5	r_5= 0.230 m
6	0	r_6	-$\pi/2$	q_6	r_6= 0.150 m

Tabela 2.1: Parâmetros de *Denavit-Hartenberg* do robô ASEA IRBL6

2. Para $i = 1$ até *número de juntas - 1* repetir de 3 a 6;
3. Estabelecer eixo Z_i na direção da junta $i + 1$;
4. Estabelecer a origem do sistema de coordenasdas i na intersecção entre os eixos Z_i e Z_{i-1} ou na intersecção da normal comum entre Z_i e Z_{i-1} . Se Z_i e Z_{i-1} são paralelos a normal comum pode ser escolhida arbitrariamente;
5. Estabelecer eixo X_i utilizando a expressão $X_i = Z_{i-1} \times Z_i \; / \; \| \; Z_{i-1} \times Z_i \; \|$;
6. Estabelecer eixo Y_i utilizando a expressão $Y_i = Z_i \times X_i \; / \; \| \; Z_i \times X_i \; \|$;
7. Estabelecer origem do último sistema de coordenadas em qualquer ponto do efetuador;
8. Estabelecer último sistema de coordenadas. O eixo X_n é perpendicular a Z_{n-1} e passa pela origem do sistema n. O eixo Z_n pode ser escolhido arbitrariamente, devendo ser perpendicular a X_n. Y_n é encontrado multiplicando-se Z_n por X_n;
9. Para $i = 1$ até *número de juntas - 1* repetir de 10 a 13;
10. Encontrar r_i pela definição deste parâmetro;
11. Encontrar a_i pela definição deste parâmetro;
12. Encontrar θ_i pela definição deste parâmetro;
13. Encontrar α_i pela definição deste parâmetro.

Na Fig. 2.7 podem ser vistos os sistemas de coordenadas do robô ASEA IRBL6. Os parâmetros de *Denavit-Hartenberg* deste mesmo robô estão colocados na Tab. 2.1.

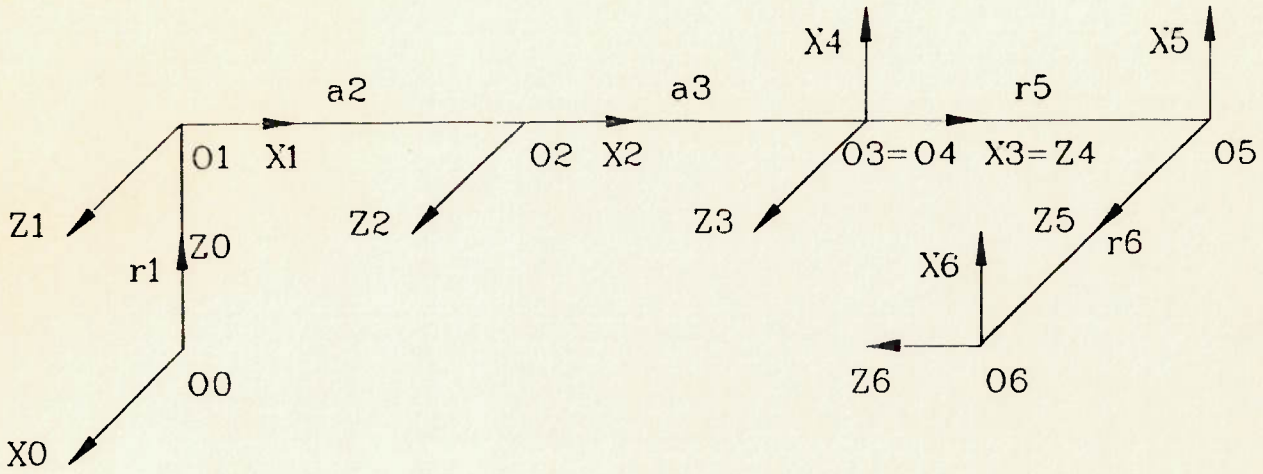


Figura 2.7: Sistemas de coordenadas para cada junta do robô ASEA IRBL6

2.6 CINEMÁTICA DIRETA

Neste ítem trata-se dos seguintes problemas:

1. *Como relacionar o sistema de coordenadas do efetuador com o sistema de coordenadas da base?* (Cinemática direta).
2. *Como relacionar variações diferenciais de coordenadas de junta com variações diferenciais de posição e orientação do efetuador para uma dada posição e orientação?* (Cinemática diferencial direta).

2.6.1 CINEMÁTICA DIRETA

O problema de cinemática direta é facilmente resolvido utilizando-se a notação de *Denavit* e *Hartenberg* em conjunto com transformações homogêneas. Assim, a relação entre dois sistemas de coordenadas vizinhos $i-1$ e i é dada pela matriz $\boldsymbol{T}_i^{i-1}$, ou seja:

$$\overline{\boldsymbol{x}_{i-1}} = \boldsymbol{T}_i^{i-1}.\overline{\boldsymbol{x}_i} \tag{2.22}$$

onde

$$\boldsymbol{T}_i^{i-1} = \begin{bmatrix} cos(\theta_i) & -sin(\theta_i).cos(\alpha_i) & sin(\theta_i).sin(\alpha_i) & a_i.cos(\theta_i) \\ sin(\theta_i) & cos(\theta_i).cos(\alpha_i) & -cos(\theta_i).sin(\alpha_i) & a_i.sin(\theta_i) \\ 0 & sin(\alpha_i) & cos(\alpha_i) & r_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.23}$$

$$\begin{aligned} \overline{x_{i-1}} &= [x_{i-1}, y_{i-1}, z_{i-1}, 1]^t \\ \overline{x_i} &= [x_i, y_i, z_i, 1]^t \end{aligned}$$

A relação entre o sistema de coordenadas da base (0) e o sistema do efetuador (n) é dada pela matriz T_n^0,

$$T_n^0 = T_1^0.T_2^1...T_n^{n-1} \tag{2.24}$$

Note que a matriz T_n^0 é função das coordenadas de junta do manipulador. Desta matriz pode-se extrair vários conjuntos de coordenadas operacionais, inclusive coordenadas cartesianas (posição) e parâmetros de *Euler-Rodrigues* (orientação) que, como descrito nos ítens 2.3.1 e 2.3.2, são utilizados nos modelos aqui desenvolvidos. A seguir apresenta-se as relações entre estas coordenadas operacionais e os elementos de T_n^0.

- Coordenadas Cartesianas:

  A partir de

  $$\overline{x_o} = T_n^0.\overline{x_n} \tag{2.25}$$

  com $\overline{x_n} = [0,0,0,1]^t$ (origem do sistema de coordenadas do efetuador em relação ao sistema do efetuador), conclui-se que $\overline{x_o} = [x,y,z,1]^t$ vale:

  $$\begin{aligned} x &= t_{14} \\ y &= t_{24} \\ z &= t_{34} \end{aligned} \tag{2.26}$$

- Parâmetros de *Euler-Rodrigues*:

  As relações entre os elementos da matriz de rotação (3 primeiras linhas e 3 primeiras colunas de T_n^0) e os parâmetros de *Euler-Rodrigues*, foram derivadas pelo matemático *Euler* (1775). Estas relações podem ser obtidas a partir da matriz de rotação equivalente a uma rotação de um ângulo θ em torno de um vetor $\boldsymbol{n}$ (*Euler*, 1775), encontrada em [GOR84,FU87,CHE89]. A expressão de *Euler* é dada por:

  $$\boldsymbol{R} = \begin{bmatrix} 2(Q_4^2+Q_1^2)-1 & 2(Q_1.Q_2-Q_3.Q_4) & 2(Q_1.Q_3+Q_2.Q_4) \\ 2(Q_1.Q_2+Q_3.Q_4) & 2(Q_4^2+Q_2^2)-1 & 2(Q_2.Q_3-Q_1.Q_4) \\ 2(Q_1.Q_3-Q_2.Q_4) & 2(Q_2.Q_3+Q_1.Q_4) & 2(Q_4^2+Q_3^2)-1 \end{bmatrix} \tag{2.27}$$

  Por inspeção na equação acima, conclui-se que [6]:

  $$\begin{aligned} Q_1 &= (1/2).sgn(t_{32}-t_{23}).\sqrt{t_{11}-t_{22}-t_{33}+1} \\ Q_2 &= (1/2).sgn(t_{13}-t_{31}).\sqrt{-t_{11}+t_{22}-t_{33}+1} \\ Q_3 &= (1/2).sgn(t_{21}-t_{12}).\sqrt{-t_{11}-t_{22}+t_{33}+1} \\ Q_4 &= (1/2).\sqrt{t_{11}+t_{22}+t_{33}+1} \end{aligned} \tag{2.28}$$

[6] Para isto utilizou-se o fato de que, com parâmetros de *Euler-Rodrigues*, pode-se descrever a orientação de um sistema de coordenadas utilizando-se ângulos de 0 a 180 graus, de forma que Q_4 é positivo (varia de 0 a 1) e, portanto, sgn(Q_i.Q_4)=sgn(Q_i), com i=1,2,3.

Deve-se notar que os parâmetros [-Q_1,-Q_2,-Q_3,-Q_4]^t, representam a mesma orientação que [Q_1,Q_2,Q_3,Q_4]^t, de forma que também poderiam ser utilizados.

Perceba que uma vez conhecida a relação entre o sistema de coordenadas do efetuador e o sistema de coordenadas da base (matriz $\boldsymbol{T}_n^0$), dadas as coordenadas generalizadas de junta, determina-se facilmente as coordenadas operacionais correspondentes. Assim, o problema de cinemática direta poderia ser enunciado da seguinte forma: *Dadas as coordenadas generalizadas de junta, quais os valores das coordenadas operacionais (posição e orientação) correspondentes?*

Desta forma, as expressões (2.26) e (2.28) podem ser expressas por uma função f conforme segue:

$$\boldsymbol{y} = f(\boldsymbol{q}) \tag{2.29}$$

onde $\boldsymbol{y}$ é o vetor de coordenadas operacionais e $\boldsymbol{q}$ é o vetor de coordenadas generalizadas de junta do manipulador.

2.6.2 CINEMÁTICA DIFERENCIAL DIRETA

Na literatura, encontra-se três métodos distintos para solução do problema de cinemática diferencial direta: Método de translações e rotações diferenciais [PAU81], Método de coordenadas operacionais [GOR84] e Método de velocidade do efetuador [WHI72]. Os três utilizam o conceito de matriz Jacobiano ($\boldsymbol{J}$) para relacionar variações diferenciais de coordenadas de junta com variações diferenciais de posição e orientação do efetuador, porém cada um define uma matriz $\boldsymbol{J}$ diferente (aqui chamaremos, respectivamente, $\boldsymbol{J}_a$, $\boldsymbol{J}_b$ e $\boldsymbol{J}_c$ as matrizes jacobiano dos métodos a, b e c).

a) Método de Translações e Rotações Diferenciais:

Paul introduziu o conceito dos operadores matriciais $\boldsymbol{\Delta}$ e ${}^T\boldsymbol{\Delta}$. O primeiro é uma matriz que quando pré-multiplicada pela matriz de transformação homogênea $\boldsymbol{T}$, nos dá a diferencial da matriz de transformação homogênea[7] $d\boldsymbol{T}$. $\boldsymbol{\Delta}$ é função das rotações e translações diferenciais ao redor e ao longo dos eixos do sistema de coordenadas da base. ${}^T\boldsymbol{\Delta}$ é uma matriz que quando pós-multiplicada pela matriz de transformação homogênea $\boldsymbol{T}$, nos dá a diferencial da matriz de transformação homogênea $d\boldsymbol{T}$. ${}^T\boldsymbol{\Delta}$ é função das rotações e tanslações diferenciais ao redor e ao longo dos eixos definidos pela matriz $\boldsymbol{T}$.

Paul mostra que a matriz ${}^T\boldsymbol{\Delta}$ pode ser escrita conforme segue:

$${}^T\boldsymbol{\Delta} = \begin{bmatrix} 0 & -{}^T\delta_z & {}^T\delta_y & {}^Td_x \\ {}^T\delta_z & 0 & -{}^T\delta_x & {}^Td_y \\ -{}^T\delta_y & {}^T\delta_x & 0 & {}^Td_z \\ 0 & 0 & 0 & 0 \end{bmatrix} \tag{2.30}$$

Com estes conceitos, para se descrever variações diferenciais na posição e orientação

[7] A diferencial da matriz de transformação homogênea $d\boldsymbol{T}$, segundo *Paul*, é a matriz que nos indica o acréscimo que sofrem os elementos de $\boldsymbol{T}$ quando o sistema de coordenadas definido pela mesma é submetido a rotações e translações diferenciais. Note que, no caso de manipuladores, este acréscimo pode ser ocasionado por rotações e translações diferenciais ao redor e ao longo do sistema de coordenadas de outro ligamento.

do sistema de coordenadas do efetuador, basta conhecer os elementos das matrizes Δ ou ${}^{T}\Delta$ associadas ao sistema de coordenadas do efetuador.

Paul relaciona os elementos de ${}^{T}\Delta$ com variações diferenciais nas coordenadas de junta para um manipulador de 6 graus de liberdade. Para isto, define a matriz U_i, correspondente ao ligamento i, da seguinte forma:

$$U_i = T_i^{i-1}.T_{i+1}^{i}...T_6^5$$

Os elementos desta matriz são dados por:

$$U_i = \begin{bmatrix} u_{11} & u_{12} & u_{13} & u_{41} \\ u_{21} & u_{22} & u_{23} & u_{42} \\ u_{31} & u_{32} & u_{33} & u_{43} \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Quando a coordenada de junta q_i sofre variação diferencial dq_i, os elementos da equação (2.30) relacionados ao sistema de coordenadas do efetuador são dados por:

$$\begin{bmatrix} {}^{T}d_x \\ {}^{T}d_y \\ {}^{T}d_z \\ {}^{T}\delta_x \\ {}^{T}\delta_y \\ {}^{T}\delta_z \end{bmatrix} = \begin{bmatrix} u_{14}.u_{21} - u_{24}.u_{11} \\ u_{14}.u_{22} - u_{24}.u_{12} \\ u_{14}.u_{23} - u_{24}.u_{13} \\ u_{31} \\ u_{32} \\ u_{33} \end{bmatrix} . \; d\theta_i \qquad (2.31)$$

se a junta i é de revolução, e

$$\begin{bmatrix} {}^{T}d_x \\ {}^{T}d_y \\ {}^{T}d_z \\ {}^{T}\delta_x \\ {}^{T}\delta_y \\ {}^{T}\delta_z \end{bmatrix} = \begin{bmatrix} u_{31} \\ u_{32} \\ u_{33} \\ 0 \\ 0 \\ 0 \end{bmatrix} . \; dr_i \qquad (2.32)$$

se a junta i é prismática.

Os vetores coluna colocados acima para a i ésima junta, nos dão a i ésima coluna da matriz Jacobiano J_a. Pode-se, portanto, escrever as equações acima conforme segue:

$$\begin{bmatrix} {}^{T}d_x \\ {}^{T}d_y \\ {}^{T}d_z \\ {}^{T}\delta_x \\ {}^{T}\delta_y \\ {}^{T}\delta_z \end{bmatrix} = J_a(q). \begin{bmatrix} dq_1 \\ dq_2 \\ dq_3 \\ dq_4 \\ dq_5 \\ dq_6 \end{bmatrix} \qquad (2.33)$$

b) Método de Coordenadas Operacionais:

Sejam as coordenadas operacionais (posição e orientação) $\boldsymbol{y}$ definidas por $[y_1, ..., y_m]^t$. Conforme visto no ítem 2.6.1, pode-se relacionar y_i com o vetor de coordenadas generalizadas de junta $\boldsymbol{q}$. Seja, desta forma, $y_i = f_i(\boldsymbol{q})$. Segundo o método de coordenadas operacionais [GOR84], o valor do elemento ij da matriz Jacobiano $\boldsymbol{J}_b$ é dado por:

$$\boldsymbol{J}_{ij} = \partial f_i / \partial q_j$$

onde $i = 1, 2, .., m$ $j = 1, 2, ..., n$ e n é o número de graus de liberdade do manipulador.

Para as coordenadas operacionais adotadas neste trabalho ($[x, y, z, Q_1, Q_2, Q_3, Q_4]^t$), tem-se a seguinte matriz $\boldsymbol{J}_b$:

$$\boldsymbol{J}_b = \begin{bmatrix} \partial x/\partial q_1 & \partial x/\partial q_2 & & \partial x/\partial q_n \\ \partial y/\partial q_1 & \partial y/\partial q_2 & & \partial y/\partial q_n \\ \partial z/\partial q_1 & \partial z/\partial q_2 & & \partial z/\partial q_n \\ \partial Q_1/\partial q_1 & \partial Q_1/\partial q_2 & \cdots & \partial Q_1/\partial q_n \\ \partial Q_2/\partial q_1 & \partial Q_2/\partial q_2 & & \partial Q_2/\partial q_n \\ \partial Q_3/\partial q_1 & \partial Q_3/\partial q_2 & & \partial Q_3/\partial q_n \\ \partial Q_4/\partial q_1 & \partial Q_4/\partial q_2 & & \partial Q_4/\partial q_n \end{bmatrix} \tag{2.34}$$

desta forma, tem-se:

$$\begin{bmatrix} dx \\ dy \\ dz \\ dQ_1 \\ dQ_2 \\ dQ_3 \\ dQ_4 \end{bmatrix} = \boldsymbol{J}_b(\boldsymbol{q}). \begin{bmatrix} dq_1 \\ dq_2 \\ dq_3 \\ . \\ . \\ . \\ dq_n \end{bmatrix} \tag{2.35}$$

ou, dividindo ambos os lados por dt:

$$\dot{\boldsymbol{y}} = \boldsymbol{J}_b(\boldsymbol{q}).\dot{\boldsymbol{q}} \tag{2.36}$$

Derivando-se as expressões (2.26) e (2.28) encontra-se as seguintes relações para as derivadas parciais enunciadas acima:

$$\begin{aligned} \partial x/\partial q_i &= \partial t_{14}/\partial q_i \\ \partial y/\partial q_i &= \partial t_{24}/\partial q_i \\ \partial z/\partial q_i &= \partial t_{34}/\partial q_i \\ \partial Q_1/\partial q_i &= (\partial t_{11}/\partial q_i - \partial t_{22}/\partial q_i - \partial t_{33}/\partial q_i)/8.Q_1 \\ \partial Q_2/\partial q_i &= (-\partial t_{11}/\partial q_i + \partial t_{22}/\partial q_i - \partial t_{33}/\partial q_i)/8.Q_2 \\ \partial Q_3/\partial q_i &= (-\partial t_{11}/\partial q_i - \partial t_{22}/\partial q_i + \partial t_{33}/\partial q_i)/8.Q_3 \\ \partial Q_4/\partial q_i &= (\partial t_{11}/\partial q_i + \partial t_{22}/\partial q_i + \partial t_{33}/\partial q_i)/8.Q_4 \end{aligned} \tag{2.37}$$

c) Método de Velocidade do Efetuador:

Define-se o vetor velocidade do efetuador ($\boldsymbol{v}$) como sendo $\boldsymbol{v} = [v_x, v_y, v_z, w_x, w_y, w_z]^t$, onde v_x, v_y, v_z e w_x, w_y, w_z são, respectivamente, as três componentes do vetor velocidade e velocidade angular escritas em relação ao sistema da base.

O método em questão relaciona o vetor de coordenadas generalizadas de junta $\boldsymbol{q}$ com o vetor $\boldsymbol{v}$ colocado acima e, a seguir, relaciona este vetor com diferenciais de coordenadas operacionais. *Whitney* [WHI72] demonstrou a seguinte relação:

$$\boldsymbol{v} = \boldsymbol{J}_c(\boldsymbol{q}).\dot{\boldsymbol{q}} = [\boldsymbol{J}_{c1}(\boldsymbol{q}), \boldsymbol{J}_{c2}(\boldsymbol{q}), ..., \boldsymbol{J}_{cn}(\boldsymbol{q})].\dot{\boldsymbol{q}} \tag{2.38}$$

onde n é o número de graus de liberdade do robô, e

$$\boldsymbol{J}_{ci}(\boldsymbol{q}) = \begin{bmatrix} ver(\boldsymbol{Z}_{i-1}) \times \boldsymbol{p}_{i-1,a} \\ ver(\boldsymbol{Z}_{i-1}) \end{bmatrix} \tag{2.39}$$

se a junta i é de revolução

$$\boldsymbol{J}_{ci}(\boldsymbol{q}) = \begin{bmatrix} ver(\boldsymbol{Z}_{i-1}) \\ 0 \end{bmatrix} \tag{2.40}$$

se a junta i é prismática.

$ver(\boldsymbol{Z}_{i-1})$ é o versor do eixo $\boldsymbol{Z}_{i-1}$ (terceira coluna de $\boldsymbol{T}^0_{i-1}$). $\boldsymbol{p}_{i-1,a}$ é o vetor posição que vai da origem do sistema $i-1$ para a origem do sistema do efetuador.

$\boldsymbol{J}_b$ calculado a partir de $\boldsymbol{J}_c$:

Como relatado no ítem 2.3, os modelos cinemáticos do robô ASEA IRBL6 e do sistema "robô ASEA IRBL6 + trilho" desenvolvidos neste capítulo, utilizam as coordenadas operacionais cartesianas (posição) e parâmetros de *Euler-Rodrigues* (orientação). Desta forma, o método de coordenadas operacionais com estes parâmetros, seria o de aplicação mais intuitiva. Este método, entretanto, possui o inconveniente de gerar expressões para a parte angular da matriz jacobiano em que o denominador freqüentemente se anula (2.37). A seguir apresenta-se uma forma de gerar $\boldsymbol{J}_b$ que não possui este defeito [GOR84].

Lembrando-se que as três primeiras linhas de $\boldsymbol{v}$ (velocidade linear do efetuador dada em relação ao sistema da base) são as derivadas do vetor posição do efetuador $\boldsymbol{v} = \dot{\boldsymbol{y}}$, percebe-se que as três primeiras linhas da matriz $\boldsymbol{J}_c$ são iguais às três primeiras linhas da matriz $\boldsymbol{J}_b$ com a parte de posição fornecida por coordenadas cartesianas.

Para a parte de orientação, esta regra não vale, uma vez que a derivada dos parâmetros de *Euler-Rodrigues* não correspondem à velocidade angular do efetuador. Entretanto, a orientação do efetuador é especificada em termos de parâmetros de *Euler-Rodrigues*, de forma que a relação entre diferenciais de coordenadas de junta e diferenciais destes parâmetros, que nos fornece o modelo diferencial direto do manipulador, pode ser bastante útil. Como $\boldsymbol{w} = \boldsymbol{J}_{co}.\dot{\boldsymbol{q}}$ (onde $\boldsymbol{w}$ é o vetor formado pelas três últimas linhas de $\boldsymbol{v}$ e $\boldsymbol{J}_{co}$ é a matriz 3×3 formada pelas três últimas linhas ou parte de orientação da matriz $\boldsymbol{J}_c$ definida por (2.39) e (2.40)), para relacionar-se $\dot{\boldsymbol{q}}$ e $\dot{\boldsymbol{y}}_o$ (quatro últimas linhas ou parte de orientação do vetor $\boldsymbol{y}$) basta encontrar-se uma matriz $\boldsymbol{M}(4\times3)$, tal que $\dot{\boldsymbol{y}}_o = \boldsymbol{M}.\boldsymbol{w}$. Neste caso, poder-se-ia escrever $\dot{\boldsymbol{y}}_o = \boldsymbol{M}.\boldsymbol{J}_{co}.\dot{\boldsymbol{q}}$ e assim a matriz $\boldsymbol{J}_{bo}$ (formada pelas quatro últimas linhas ou parte de orientação da matriz $\boldsymbol{J}_b$) poderia ser obtida por:

$$\boldsymbol{J}_{bo} = \boldsymbol{M}.\boldsymbol{J}_{co} \tag{2.41}$$

Para obter-se $\boldsymbol{M}$, seja o vetor $\boldsymbol{r_0}$ escrito em relação a um sistema de coordenadas fixo qualquer. Quando este vetor sofre a rotação definida pela matriz de rotação $\boldsymbol{R}$, pode-se escrever:

$$\boldsymbol{r} = \boldsymbol{R}.\boldsymbol{r_0}$$

onde $\boldsymbol{r}$ é o vetor resultante da rotação. Derivando esta expressão em relação ao tempo, tem-se

$$\dot{\boldsymbol{r}} = \dot{\boldsymbol{R}}.\boldsymbol{r_0} + \boldsymbol{R}.\dot{\boldsymbol{r_0}} = \dot{\boldsymbol{R}}.\boldsymbol{r_0} \tag{2.42}$$

onde $\dot{\boldsymbol{r_0}}$ foi tomado igual a zero.

Para um ponto em um corpo rígido girando com velocidade angular $\boldsymbol{w}$ definido pelo vetor posição $\boldsymbol{r}$, também pode-se escrever

$$\dot{\boldsymbol{r}} = \boldsymbol{w} \times \boldsymbol{r}$$

ou

$$\dot{\boldsymbol{r}} = \boldsymbol{\Omega}.\boldsymbol{r} = \boldsymbol{\Omega}.\boldsymbol{R}.\boldsymbol{r_0} \tag{2.43}$$

onde $\boldsymbol{\Omega}$ é a matriz dada por:

$$\boldsymbol{\Omega} = \begin{bmatrix} 0 & -w_z & w_y \\ w_z & 0 & -w_x \\ -w_y & w_x & 0 \end{bmatrix} \tag{2.44}$$

Comparando-se (2.42) e (2.43), obtém-se:

$$\dot{\boldsymbol{R}} = \boldsymbol{\Omega}.\boldsymbol{R} \tag{2.45}$$

A partir da derivada de (2.8) e da equação (2.45), encontra-se as derivadas dos parâmetros de *Euler-Rodrigues* em função de $\boldsymbol{w}$, o que resulta na seguinte matriz $\boldsymbol{M}$:

$$\boldsymbol{M} = 1/2. \begin{bmatrix} Q_4 & Q_3 & -Q_2 \\ -Q_3 & Q_4 & Q_1 \\ Q_2 & -Q_1 & Q_4 \\ -Q_1 & -Q_2 & -Q_3 \end{bmatrix} \tag{2.46}$$

Perceba, observando as expressões (2.41) e (2.46), que utilizando-se este procedimento, encontrou-se a matriz $\boldsymbol{J_b}$ sem o problema de singularidade no denominador que ocorre quando $\boldsymbol{J_b}$ é encontrada pelo método das coordenadas operacionais (2.37). Assim, optou-se por este método para encontrar os modelos diferenciais deste trabalho.

2.7 CINEMÁTICA INVERSA

Conforme visto no ítem 2.6.1, pode-se relacionar coordenadas de posição e orientação do efetuador (coordenadas operacionais) com coordenadas generalizadas de junta.

Neste ítem coloca-se os seguintes problemas:

- *É possível encontrar uma expressão que forneça as coordenadas generalizadas de junta do manipulador dadas as coordenadas operacionais (posição e orientação) do efetuador?* (Cinemática Inversa Analítica)
- *É possível encontrar uma expressão que forneça as derivadas em relação ao tempo das coordenadas generalizadas de junta do manipulador dadas as coordenadas operacionais e as derivadas em relação ao tempo das coordenadas operacionais (posição e orientação) do efetuador?* (Cinemática Diferencial Inversa Analítica)

A resposta a estas questões ainda é um problema aberto na robótica [ROT76]. Atualmente, sabe-se apenas que existe a solução analítica da cinemática inversa para manipuladores que satisfaçam alguma das seguintes características [8]:

- Tenham 6 graus de liberdade com duas juntas prismáticas;
- Tenham 6 graus de liberdade com uma junta prismática e uma junta de revolução coaxiais;
- Tenham 6 graus de liberdade com dois pares de juntas de revolução concorrentes;
- Tenham 6 graus de liberdade com três juntas de revolução concorrentes;
- Tenham menos de 6 graus de liberdade.

Quando a solução analítica não é possível, as coordenadas generalizadas de junta só podem ser encontradas por meio de métodos numéricos, iterativos. Estes, são estudados detalhadamente nos próximos capítulos.

Para efeito de simplificação de linguagem trataremos, neste trabalho, ambos os termos cinemática inversa e cinemática diferencial inversa por cinemática inversa, procedimento comum na literatura.

2.8 ROBÔ E SISTEMA "ROBÔ ASEA IRBL6 + TRILHO"

Neste ítem apresenta-se a modelagem cinemática direta do robô ASEA IRBL6 e do sistema "robô ASEA IRBL6 + trilho" (Fig. 2.8). Para esta tarefa encontrou-se a matriz J_c (método de velocidade do efetuador), e a partir desta e de M(2.46), a matriz J_b (relaciona diferenciais de coordenadas generalizadas de junta e diferenciais de coordenadas operacionais com coordenadas cartesianas e parâmetros de *Euler-Rodriges*). As expressões resultantes são utilizadas nos capítulos seguintes.

[8] *Roth* não estudou a solução analítica para a cinemática inversa de manipuladores com mais do que 6 graus de liberdade.

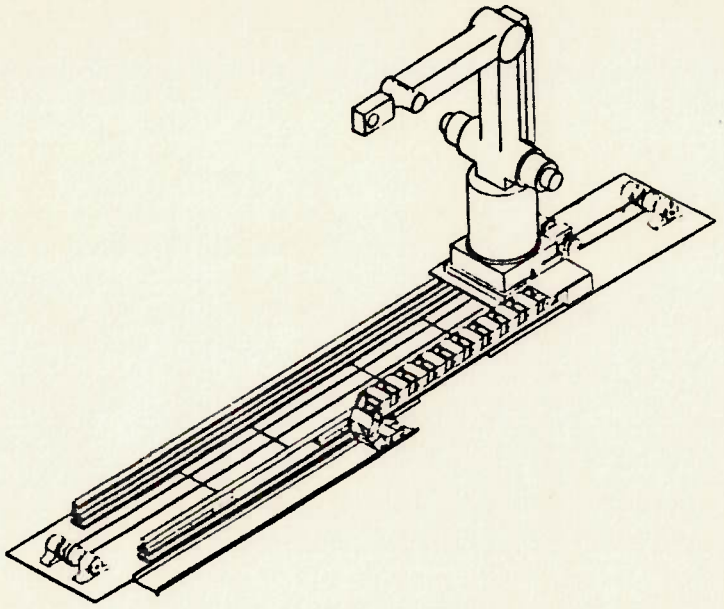


Figura 2.8: "Robô ASEA IRBL6 + trilho"

Os sistemas de coordenadas de cada junta do robô podem ser vistos na Fig. 2.7 e os parâmetros de *Denavit-Hartenberg* na Tab. 2.1.

A Fig. 2.9 mostra os sistemas de coordenadas de cada junta do sistema "robô + trilho". Note que o trilho é modelado como uma junta prismática. Os parâmetros de *Denavit-Hartenberg* são mostrados na Tab. 2.2.

junta	a_i	r_i	α_i	θ_i
1	0	q_1	$\pi/2$	0
2	0	r_2	$\pi/2$	q_2
3	a_3	0	0	q_3
4	a_4	0	0	q_4
5	0	0	$\pi/2$	q_5
6	0	r_6	$-\pi/2$	q_6
7	0	r_7	$-\pi/2$	q_7

Tabela 2.2: Parâmetros de *Denavit-Hartenberg* do sistema "robô ASEA IRBL6 + trilho"

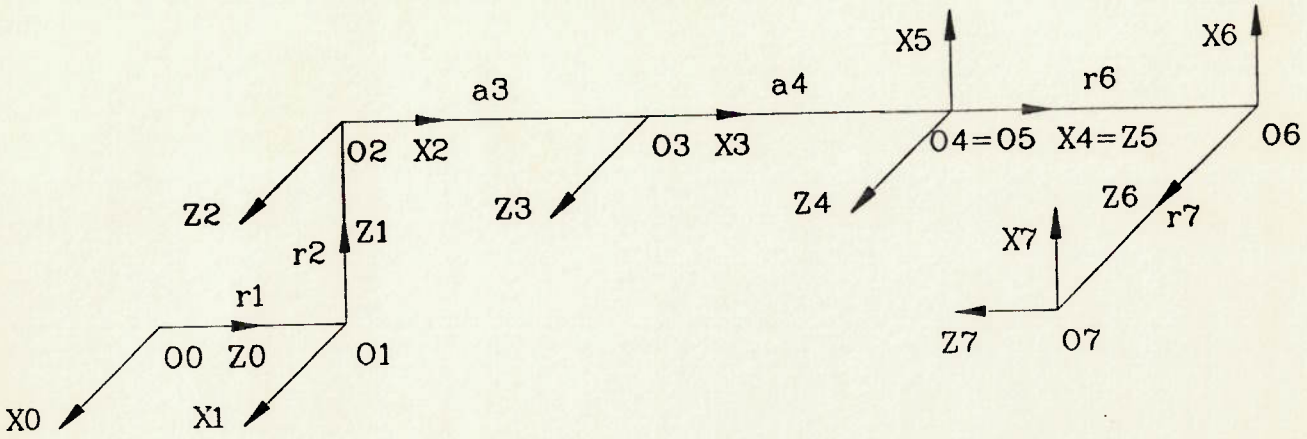


Figura 2.9: Sistemas de coordenadas para cada junta do "robô ASEA IRBL6 + trilho"

2.8.1 MODELAGEM DO ROBÔ ASEA IRBL6

Observando os parâmetros mostrados na Tab. 2.1, pode-se montar as seguintes matrizes de transformação homogênea para o robô ASEA IRBL6 (2.23):

$$T_1^0 = \begin{bmatrix} c1 & 0 & s1 & 0 \\ s1 & 0 & -c1 & 0 \\ 0 & 1 & 0 & r_1 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad T_4^3 = \begin{bmatrix} c4 & 0 & s4 & 0 \\ s4 & 0 & -c4 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$T_2^1 = \begin{bmatrix} c2 & -s2 & 0 & a_2.c2 \\ s2 & c2 & 0 & a_2.s2 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad T_5^4 = \begin{bmatrix} c5 & 0 & -s5 & 0 \\ s5 & 0 & c5 & 0 \\ 0 & -1 & 0 & r_5 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$T_3^2 = \begin{bmatrix} c3 & -s3 & 0 & a_3.c3 \\ s3 & c3 & 0 & a_3.s3 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad T_6^5 = \begin{bmatrix} c6 & 0 & -s6 & 0 \\ s6 & 0 & c6 & 0 \\ 0 & -1 & 0 & r_6 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Através da expressão (2.22), encontramos:

$$T_6^0 = \begin{bmatrix} t_{11} & t_{12} & t_{13} & t_{14} \\ t_{21} & t_{22} & t_{23} & t_{24} \\ t_{31} & t_{32} & t_{33} & t_{34} \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad (2.47)$$

onde

$$
\begin{aligned}
t_{11} &= c6.(c1.c234.c5 + s1.s5) - c1.s234.s6 \\
t_{12} &= c1.c234.s5 - s1.c5 \\
t_{13} &= -s6.(c1.c234.c5 + s1.s5) - c1.s234.c6 \\
t_{14} &= r_6.(-c1.c234.s5 + s1.c5) + c1.s234.r_5 + a_3.c1.c23 + c1.c2.a_2 \\
t_{21} &= c6.(s1.c234.c5 - c1.s5) - s1.s234.s6 \\
t_{22} &= s1.c234.s5 + c1.c5 \\
t_{23} &= -s6.(s1.c234.c5 - c1.s5) - s1.s234.c6 \\
t_{24} &= r_6.(-s1.c234.s5 - c1.c5) + s1.s234.r_5 + a_3.s1.c23 + a_2.s1.c2 \\
t_{31} &= s234.c5.c6 + c234.s6 \\
t_{32} &= s234.s5 \\
t_{33} &= -s234.c5.s6 + c234.c6 \\
t_{34} &= -s234.s5.r_6 - c234.r_5 + a_3.s23 + a_2.s2 + r_1
\end{aligned}
$$

Utilizando-se as expressões (2.39), encontramos os termos da matriz J_c (a parte de posição foi encontrada a partir de (2.37), uma vez que, conforme discutido em 2.6.2, as três primeiras linhas de J_b coincidem com as três primeiras linhas de J_c).

$$
\begin{aligned}
J_{c11} &= \partial x/\partial q_1 = r_6(s5c234s1 + c1c5) - s1s234r_5 - a_3s1c23 - s1c2a_2 \\
J_{c21} &= \partial y/\partial q_1 = r_6(-s5c234c1 + s1c5) + c1s234r_5 + a_3c1c23 + c1c2a_2 \\
J_{c31} &= \partial z/\partial q_1 = 0 \\
J_{c41} &= 0 \\
J_{c51} &= 0 \\
J_{c61} &= 1
\end{aligned}
$$

$$
\begin{aligned}
J_{c12} &= \partial x/\partial q_2 = r_6c1s5s234 + c1r_5c234 - a_3c1s23 - a_2c1s2 \\
J_{c22} &= \partial y/\partial q_2 = r_6s1s234s5 + s1c234r_5 - a_3s1s23 - a_2s1s2 \\
J_{c32} &= \partial z/\partial q_2 = -r_6s5c234 + r_5s234 + a_3c23 + a_2c2 \\
J_{c42} &= s1 \\
J_{c52} &= -c1 \\
J_{c62} &= 0
\end{aligned}
$$

$$
\begin{aligned}
J_{c13} &= \partial x/\partial q_3 = r_6c1s5s234 + c1r_5c234 - a_3c1s23 \\
J_{c23} &= \partial y/\partial q_3 = r_6s1s234s5 + s1c234r_5 - a_3s1s23 \\
J_{c33} &= \partial z/\partial q_3 = -r_6s5c234 + r_5s234 + a_3c23 \\
J_{c43} &= s1 \\
J_{c53} &= -c1 \\
J_{c63} &= 0
\end{aligned}
\qquad (2.48)
$$

$$
\begin{aligned}
J_{c14} &= \partial x/\partial q_4 = r_6c1s5s234 + c1r_5c234 \\
J_{c24} &= \partial y/\partial q_4 = r_6s1s234s5 + s1c234r_5 \\
J_{c34} &= \partial z/\partial q_4 = -r_6s5c234 + r_5s234 \\
J_{c44} &= s1 \\
J_{c54} &= -c1 \\
J_{c64} &= 0
\end{aligned}
$$

$$J_{c15} = \partial x/\partial q_5 = -r_6 c1 c5 c234 - r_6 s1 s5$$
$$J_{c25} = \partial y/\partial q_5 = r_6(-s1 c234 c5 + c1 s5)$$
$$J_{c35} = \partial z/\partial q_5 = -r_6 s234 c5$$
$$J_{c45} = c1 s234$$
$$J_{c55} = s1 s234$$
$$J_{c65} = -c234$$

$$J_{c16} = \partial x/\partial q_6 = 0$$
$$J_{c26} = \partial y/\partial q_6 = 0$$
$$J_{c36} = \partial z/\partial q_6 = 0$$
$$J_{c46} = -c1 c234 s5 + s1 c5$$
$$J_{c56} = -s1 c234 s5 - c1 c5$$
$$J_{c66} = -s234 s5$$

Segundo as equações (2.41) e (2.46), tem-se:

$$\begin{bmatrix} J_{b4} \\ J_{b5} \\ J_{b6} \\ J_{b7} \end{bmatrix} = 1/2. \begin{bmatrix} Q_4 & Q_3 & -Q_2 \\ -Q_3 & Q_4 & Q_1 \\ Q_2 & -Q_1 & Q_4 \\ -Q_1 & -Q_2 & -Q_3 \end{bmatrix} . \begin{bmatrix} J_{c4} \\ J_{c5} \\ J_{c6} \end{bmatrix} \tag{2.49}$$

onde (2.28)

$$Q_1 = \frac{0.5 sgn(s5 s234 + s6(s1 c234 c5 - c1 s5) + s1 s234 c6).}{\sqrt{c6(c1 c234 c5 + s1 s5) - c1 s234 s6 - s1 c234 s5 - c1 c5 + s234 c5 s6 - c234 c6 + 1}}$$

$$Q_2 = \frac{0.5 sgn(-s6(c1 c234 c5 + s1 s5) - c1 s234 c6 - s234 c5 c6 - c234 s6).}{\sqrt{-c6(c1 c234 c5 + s1 s5) + c1 s234 s6 + s1 c234 s5 + c1 c5 + s234 c5 s6 - c234 c6 + 1}}$$

$$Q_3 = \frac{0.5 sgn(c6(s1 c234 c5 - c1 s5) - s1 s234 s6 - c1 c234 s5 + s1 c5).}{\sqrt{-c6(c1 c234 c5 + s1 s5) + c1 s234 s6 - s1 c234 s5 - c1 c5 - s234 c5 s6 + c234 c6 + 1}}$$

$$Q_4 = \frac{\sqrt{c6(c1 c234 c5 + s1 s5) - c1 s234 s6 + s1 c234 s5 + c1 c5 - s234 c5 s6 + c234 c6 + 1}}{2}$$

2.8.2 MODELAGEM DO SISTEMA "ROBÔ ASEA IRBL6 + TRILHO"

Observando os parâmetros mostrados na Tab 2.2, pode-se montar as seguintes matrizes de transformação homogênea para o sistema (2.23):

$$T_1^0 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 1 & 0 & r_1 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad T_4^3 = \begin{bmatrix} c4 & -s4 & 0 & a_4.c4 \\ s4 & c4 & 0 & a_4.s4 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$T_2^1 = \begin{bmatrix} c2 & 0 & s2 & 0 \\ s2 & 0 & -c2 & 0 \\ 0 & 1 & 0 & r_2 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad T_5^4 = \begin{bmatrix} c5 & 0 & s5 & 0 \\ s5 & 0 & -c5 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$T_3^2 = \begin{bmatrix} c3 & -s3 & 0 & a_3.c3 \\ s3 & c3 & 0 & a_3.s3 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad T_6^5 = \begin{bmatrix} c6 & 0 & -s6 & 0 \\ s6 & 0 & c6 & 0 \\ 0 & -1 & 0 & r_6 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$T_7^6 = \begin{bmatrix} c7 & 0 & -s7 & 0 \\ s7 & 0 & c7 & 0 \\ 0 & -1 & 0 & r_7 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Através da expressão (2.22), encontramos:

$$T_7^0 = \begin{bmatrix} t_{11} & t_{12} & t_{13} & t_{14} \\ t_{21} & t_{22} & t_{23} & t_{24} \\ t_{31} & t_{32} & t_{33} & t_{34} \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.50}$$

onde

$$\begin{aligned}
t_{11} &= c7.(c2.c345.c6 + s2.s6) - c2.s345.s7 \\
t_{12} &= c2.c345.s6 - s2.c6 \\
t_{13} &= -s7.(c2.c345.c6 + s2.s6) - c2.s345.c7 \\
t_{14} &= r_7.(-c2.c345.s6 + s2.c6) + c2.s345.r_6 + a_4.c2.c34 + c2.c3.a_3 \\
t_{21} &= -s345.c6.c7 - c345.s7 \\
t_{22} &= -s345.s6 \\
t_{23} &= s345.c6.s7 - c345.c7 \\
t_{24} &= s345.s6.r_7 + c345.r_6 - a_4.s34 - a_3.s3 - r_2 \\
t_{31} &= c7.(s2.c345.c6 - c2.s6) - s2.s345.s7 \\
t_{32} &= s2.c345.s6 + c2.c6 \\
t_{33} &= -s7.(s2.c345.c6 - c2.s6) - s2.s345.c7 \\
t_{34} &= r_7.(-s2.c345.s6 - c2.c6) + s2.s345.r_6 + a_4.s2.c34 + a_3.s2.c3 + r_1
\end{aligned}$$

Utilizando-se as expressões (2.39), encontramos os termos da matriz J_c (a parte de posição foi encontrada a partir de (2.37), uma vez que, conforme discutido em 2.6.2, as três primeiras linhas de J_b coincidem com as três primeiras linhas de J_c).

$$
\begin{aligned}
J_{c11} &= \partial x/\partial q_1 = 0 \\
J_{c21} &= \partial y/\partial q_1 = 0 \\
J_{c31} &= \partial z/\partial q_1 = 1 \\
J_{c41} &= 0 \\
J_{c51} &= 0 \\
J_{c61} &= 0
\end{aligned}
$$

$$
\begin{aligned}
J_{c12} &= \partial x/\partial q_2 = r_7(s6c345s2 + c2c6) - s2s345r_6 - a_4s2c34 - s2c3a_3 \\
J_{c22} &= \partial y/\partial q_2 = 0 \\
J_{c32} &= \partial z/\partial q_2 = r_7(-s6c345c2 + s2c6) + c2s345r_6 + a_4c2c34 + c2c3a_3 \\
J_{c42} &= 0 \\
J_{c52} &= -1 \\
J_{c62} &= 0
\end{aligned}
$$

$$
\begin{aligned}
J_{c13} &= \partial x/\partial q_3 = r_7c2s6s345 + c2r_6c345 - a_4c2s34 - a_3c2s3 \\
J_{c23} &= \partial y/\partial q_3 = r_7s6c345 - r_6s345 - a_4c34 - a_3c3 \\
J_{c33} &= \partial z/\partial q_3 = r_7s2s345s6 + s2c345r_6 - a_4s2s34 - a_3s2s3 \\
J_{c43} &= s2 \\
J_{c53} &= 0 \\
J_{c63} &= -c2
\end{aligned}
$$

$$
\begin{aligned}
J_{c14} &= \partial x/\partial q_4 = r_7c2s6s345 + c2r_6c345 - a_4c2s34 \\
J_{c24} &= \partial y/\partial q_4 = r_7s6c345 - r_6s345 - a_4c34 \\
J_{c34} &= \partial z/\partial q_4 = r_7s2s345s6 + s2c345r_6 - a_4s2s34 \\
J_{c44} &= s2 \\
J_{c54} &= 0 \\
J_{c64} &= -c2
\end{aligned}
\qquad (2.51)
$$

$$
\begin{aligned}
J_{c15} &= \partial x/\partial q_5 = r_7c2s6s345 + c2r_6c345 \\
J_{c25} &= \partial y/\partial q_5 = r_7s6c345 - r_6s345 \\
J_{c35} &= \partial z/\partial q_5 = r_7s2s345s6 + s2c345r_6 \\
J_{c45} &= s2 \\
J_{c55} &= 0 \\
J_{c65} &= -c2
\end{aligned}
$$

$$
\begin{aligned}
J_{c16} &= \partial x/\partial q_6 = -r_7c2c6c345 - r_7s2s6 \\
J_{c26} &= \partial y/\partial q_6 = r_7s345c6 \\
J_{c36} &= \partial z/\partial q_6 = r_7(-s2c345c6 + c2s6) \\
J_{c46} &= c2s345 \\
J_{c56} &= c345 \\
J_{c66} &= s2s345
\end{aligned}
$$

$$
\begin{aligned}
J_{c17} &= \partial x/\partial q_7 = 0 \\
J_{c27} &= \partial y/\partial q_7 = 0
\end{aligned}
$$

$J_{c37} = \partial z/\partial q_7 = 0$
$J_{c47} = -c2c345s6 + s2c6$
$J_{c57} = s345s6$
$J_{c67} = -s2c345s6 - c2c6$

Segundo as equações (2.41) e (2.46), tem-se:

$$\begin{bmatrix} J_{b4} \\ J_{b5} \\ J_{b6} \\ J_{b7} \end{bmatrix} = 1/2. \begin{bmatrix} Q_4 & Q_3 & -Q_2 \\ -Q_3 & Q_4 & Q_1 \\ Q_2 & -Q_1 & Q_4 \\ -Q_1 & -Q_2 & -Q_3 \end{bmatrix} \cdot \begin{bmatrix} J_{c4} \\ J_{c5} \\ J_{c6} \end{bmatrix} \qquad (2.52)$$

onde (2.28)

$$Q_1 = 0.5sgn(c6c2 + s6c345s2 + c7c345 - s7c6s345).$$
$$\sqrt{c2(c7c6c345 - s7s345 - s7s6) + s2(s7c6c345 + c7s345 + c7s6) + s6s345 + 1}$$

$$Q_2 = 0.5sgn(-s7(c2c345c6 + s2s6) - c2s345c7 - c7(s2c345c6 - c2s6) + s2s345s7).$$
$$\sqrt{c2(s7(s345 - s6) - c7c6c345) + s2(s7c6c345 + c7s345 - c7s6) - s6s345 + 1}$$

$$Q_3 = 0.5sgn(-s6c345c2 + c6s2 - s7c345 - c7c6s345).$$
$$\sqrt{c2(s7(s345 + s6) - c7c6c345) - s2(s7c6c345 + c7(s345 + s6)) + s6s345 + 1}$$

$$Q_4 = \frac{\sqrt{c2(c7c6c345 - s7s345 + s7s6) - s2(s7c6c345 + c7(s345 - s6)) - s6s345 + 1}}{2}$$

Nos próximos capítulos, a matriz J_b será tratada, simplesmente, por J.

É importante notar que, tanto para o robô ASEA quanto para o sistema "robô + trilho", as quatro últimas linhas da matriz J são linearmente dependentes, de forma que se uma destas linhas não for considerada não ocorre perda de informação. Para esta verificação, basta diferenciar a expressão 2.8. Desta forma, obtem-se:

$$Q_1 dQ_1 + Q_2 dQ_2 + Q_3 dQ_3 + Q_4 dQ_4 = 0$$

Por esta razão as matrizes J do robô e do sistema "robô + trilho", utilizadas nos próximos capítulos, são formadas, apenas, pelas seis primeiras linhas das matrizes deduzidas neste ítem, e a parte de orientação será dada apenas pelos três primeiros parâmetros de *Euler-Rodrigues*.

Capítulo 3

MÉTODOS DE NEWTON-RAPHSON

3.1 INTRODUÇÃO

Em [UIC64], encontra-se a primeira utilização do método de *Newton-Raphson* para solucionar a cinemática inversa de robôs manipuladores. Neste artigo, *Uicker* toma a localização do efetuador e a coordenada de uma das juntas de um manipulador de sete graus de liberdade como dados de entrada, e, com o método de *Newton-Raphson*, calcula as coordenadas das demais juntas. As equações do método eram encontradas através da linearização da matriz de transformação homogênea (não utilizou-se as matrizes Jacobiano vistas no capítulo 2). Em 1968, *Pieper* [PIE68] propôs, em sua tese de doutorado, um novo método que utiliza o conceito de matriz Jacobiano para relacionar diretamente diferenciais de coordenadas de junta com diferenciais de coordenadas de espaço e comparou, através de simulações, o seu método com o de *Uicker* para um manipulador de seis graus de liberdade de rotação, mostrando que o novo algoritmo era mais eficiente que o anterior.

Até então, os sistemas de equações a serem resolvidos sempre possuiam o mesmo número de incógnitas e equações, e não era possível calcular as coordenadas de junta quando a matriz a ser invertida estava em posição singular. Visando solucionar estes problemas, *Whitney* [WHI72] propôs a aplicação da pseudo-inversa [GRE59,BOU71,BEN74] ao invés da inversa para manipuladores, o que, por possuir um critério de otimização implícito (conforme pode ser visto no apêndice A) permite resolver a cinemática inversa, mesmo quando o Jacobiano não é quadrado ou possui determinante nulo. A partir daí, diversos trabalhos aparecem com a utilização de pseudo-inversas [LIE77,KLE83,ANG85, WAM86,CHA88].

[NOV90] classifica os métodos para solução da cinemática inversa em duas categorias: métodos para ponto fixo e métodos para trajetória. Os primeiros lidam com o problema de determinar o vetor $\boldsymbol{q}$ dado um vetor $\boldsymbol{y}$ constante e os outros visam determinar $\boldsymbol{q}$ para um vetor $\boldsymbol{y}$ variável segundo uma dada trajetória, parametrizada ou não no tempo. Problemas da segunda categoria, podem ser resolvidos utilizando-se algum

método para ponto fixo em diversos pontos da trajetória, ou então através de algoritmos que "trilhem" a trajetória fazendo com que o vetor $\boldsymbol{y}_{desejado}$ - $\boldsymbol{y}$ (onde $\boldsymbol{y}$ é o vetor calculado a partir de $\boldsymbol{q}$ encontrado com o algoritmo) tenda a zero.

Neste capítulo, apresenta-se e aplica-se o método de *Newton-Raphson* para sistemas de equações não lineares na solução da cinemática inversa do robô ASEA IRBL6. Em seguida mostra-se o método de *Newton-Raphson* modificado [STR86] (ponto-fixo) e propõe-se um novo algoritmo, baseado neste último, para casos de trajetória em que são utilizados métodos para ponto fixo em diversos pontos da mesma, que chamaremos de método de *Newton-Raphson* adaptado. Em todos os algoritmos deste capítulo, a inversa da matriz Jacobiano, que aparece na formulação original dos mesmos, é substituida pela pseudo-inversa (apêndice A).

3.2 CONSIDERAÇÕES SOBRE ERROS DOS ALGORITMOS

Ao invés de determinar-se os erros dos algoritmos no espaço de juntas, faz-se, neste trabalho, esta determinação no espaço de configuração do manipulador, ou seja, controla-se os erros de posição e orientação do efetuador (procedimento bastante usual quando se lida com manipuladores). Assim, defini-se erros de posição (δ_p) e de orientação (δ_o).

Seja $\boldsymbol{y}_d$ o vetor 7×1 que contém a posição (coordenadas cartesianas) e orientação [1] (parâmetros de *Euler-Rodrigues*) desejada (de referência) do efetuador e $\boldsymbol{y}$ o vetor encontrado aplicando-se a cinemática direta (2.29) com as coordenadas de junta encontradas pelo algoritmo. As três primeiras componentes destes vetores fornecem a posição e as quatro últimas, a orientação do efetuador.

O erro de posição foi definido por:

$$\delta_p = \sqrt{(\boldsymbol{y}(1) - \boldsymbol{y}_d(1))^2 + (\boldsymbol{y}(2) - \boldsymbol{y}_d(2))^2 + (\boldsymbol{y}(3) - \boldsymbol{y}_d(3))^2} \tag{3.1}$$

O erro de orientação foi definido por:

$$\delta_o = 2.cos^{-1}(\boldsymbol{y}(4).\boldsymbol{y}_d(4) + \boldsymbol{y}(5).\boldsymbol{y}_d(5) + \boldsymbol{y}(6).\boldsymbol{y}_d(6) + \boldsymbol{y}(7).\boldsymbol{y}_d(7)) \tag{3.2}$$

Este erro foi tomado [CHE89] igual ao ângulo contido nos parâmetros de *Euler-Rodrigues* correspondentes à rotação que leva o sistema de coordenadas do efetuador calculado ($\boldsymbol{y}$) no sistema desejado ($\boldsymbol{y}_d$), e foi encontrado pela expressão (2.1).

[1] Note que os vetores $\boldsymbol{y}_d$ utilizados nos algoritmos possuem 6 componentes. A componente 7 pode ser encontrada a partir das componentes 4, 5 e 6 através da expressão (2.8).

3.3 MÉTODO DE NEWTON-RAPHSON

O método de *Newton-Raphson*, permite solucionar um sistema de equações não lineares do seguinte tipo:

$$\begin{aligned} g_1(x_1,...,x_n) &= 0 \\ &... \\ g_n(x_1,...,x_n) &= 0 \end{aligned}$$

onde $g_1, g_2, ..., g_n$ são n funções não lineares nas variáveis $x_1, x_2, ..., x_n$.

Considerando-se o vetor $\boldsymbol{x}$, composto pelas n incógnitas $x_1, x_2, ..., x_n$, pode-se, nas vizinhanças de $\boldsymbol{x}$, expandir as funções g_i em séries de *Taylor* desprezando-se termos de ordem maior que um, conforme segue:

$$g_i(\boldsymbol{x} + \delta\boldsymbol{x}) = g_i(\boldsymbol{x}) + \sum_{j=1}^{n} (\partial g_i/\partial x_j)\delta x_j \tag{3.3}$$

A idéia do método consiste em utilizar esta equação para dar origem a um processo iterativo que faça, no instante final, o termo do lado esquerdo da expressão acima se anular. O processo de iteração é dado por:

$$\boldsymbol{x}^{k+1} = \boldsymbol{x}^k - \boldsymbol{J}^{-1}.\boldsymbol{g}(\boldsymbol{x}^k) \tag{3.4}$$

onde $\boldsymbol{J}$ é uma matriz $n \times n$ cujo elemento ij (da i-ésima linha e j-ésima coluna) é dado por $\partial g_i/\partial x_j$ e $\boldsymbol{g}(\boldsymbol{x}^k)$ é um vetor de n componentes, cuja componente i é dada por $g_i(\boldsymbol{x}^k)$.

Esta expressão surgiu da linearização das n funções g_i, tomando-se os termos $g_i(x^{k+1})$ iguais a zero e escrevendo-se o sistema linearizado na forma matricial.

É importante notar que o método fornece apenas uma das raízes do sistema. Esta raiz depende do vetor $\boldsymbol{x}^0$ de partida.

Não existe uma demonstração para convergência do algoritmo para qualquer sistema de equações, qualquer que seja $\boldsymbol{x}$ de partida. Demonstra-se, apenas, que para qualquer sistema existe uma vizinhança da raiz, tal que, se $\boldsymbol{J}$ é inversível nesta vizinhança, o sistema converge para esta raiz [CAR69,PRE86]. Nesta demonstração, percebe-se que a convergência, quando ocorre, é quadrática.

Neste trabalho, as funções g_i a serem resolvidas são dadas pelo lado direito da equação da cinemática direta (2.29) menos o vetor $\boldsymbol{y}$ desejado, e $\boldsymbol{x}$ é igual ao vetor de coordenadas generalizadas de junta $\boldsymbol{q}$, conforme fluxograma adiante.

3.3.1 INTERPRETAÇÃO GEOMÉTRICA

Para o caso unidimensional, a interpretação geométrica do método de *Newton-Raphson* é bastante simples [CAR69,PRE86,STR86]. Neste caso, x^{k+1} é dado pela intersecção entre

o eixo horizontal e a reta tangente à função no ponto x^k. Para o caso multidimensional, entretanto, a interpretação geométrica torna-se bem mais complexa. Neste ítem enuncia-se esta interpretação e mostra-se, através de geometria analítica, que o enunciado está correto.

Enunciado: Para encontrar-se $\boldsymbol{x}^{k+1}$ a partir de $\boldsymbol{x}^k$, procede-se da seguinte forma: Traça-se os n hiperplanos tangentes as n funções g_i em $(\boldsymbol{x}^k, g_i(\boldsymbol{x}^k))$. Encontra-se a intersecção entre os mesmos e o hiperplano que passa pela origem e tem como gradiente $[0,0,...,0,1]^t$. Esta intersecção fornece $\boldsymbol{x}^{k+1}$.

Para se verificar este enunciado, considere a equação do hiperplano tangente à curva g_i em $(\boldsymbol{x}^k, g_i(\boldsymbol{x}^k))$

$$z = \partial g_i(\boldsymbol{x}^k)/\partial x_1^k.(x - x_1^k) + ... + \partial g_i(\boldsymbol{x}^k)/\partial x_n^k.(x - x_n^k) + g_i(\boldsymbol{x}^k) \tag{3.5}$$

A montagem desta equação foi feita utilizando-se o gradiente da curva g_i em $(\boldsymbol{x}^k, g_i(\boldsymbol{x}^k))$

$$(-\partial g_i(\boldsymbol{x}^k)/\partial x_1^k, ..., -\partial g_i(\boldsymbol{x}^k)/\partial x_n^k, 1) \tag{3.6}$$

Fazendo-se z=0 para todos os n hiperplanos (3.5) (intersecção entre os mesmos e o que passa pela origem e tem como gradiente $[0,0,...,0,1]^t$), resulta o seguinte sistema de equações:

$$\begin{bmatrix} \partial g_1(\boldsymbol{x}^k)/\partial x_1 & \cdots & \partial g_1(\boldsymbol{x}^k)/\partial x_n \\ \vdots & & \vdots \\ \partial g_n(\boldsymbol{x}^k)/\partial x_1 & & \partial g_n(\boldsymbol{x}^k)/\partial x_n \end{bmatrix} \begin{bmatrix} x_1^{k+1} - x_1^k \\ \vdots \\ x_n^{k+1} - x_n^k \end{bmatrix} + \begin{bmatrix} g_1(\boldsymbol{x}^k) \\ \vdots \\ g_n(\boldsymbol{x}^k) \end{bmatrix} \tag{3.7}$$

Este sistema coincide com (3.4), ou seja, obteve-se as mesmas equações do método de *Newton-Raphson*, como queríamos demonstrar.

3.3.2 MÉTODO DE NEWTON-RAPHSON MODIFICADO

Apesar da convergência do método de *Newton-Raphson* ser bastante rápida (quadrática), cada iteração é lenta. Isto ocorre porque os processos de computação dos termos da matriz Jacobiano e de inversão da mesma, necessitam de um tempo computacional elevado.

Em [STR86] encontra-se um novo algoritmo, derivado do método apresentado acima, que abranda este problema. Segundo este método, computa-se a matriz Jacobiano e sua inversa somente para o ponto inicial do processo iterativo ($\boldsymbol{J}(\boldsymbol{x}^0)$ e $\boldsymbol{J}^{-1}(\boldsymbol{x}^0)$). Todos os demais pontos são encontrados utilizando $\boldsymbol{J}(\boldsymbol{x}^0)$ e $\boldsymbol{J}^{-1}(\boldsymbol{x}^0)$. Para o caso unidimensional, isto seria equivalente a tomar x^1 como sendo a intersecção entre o eixo horizontal e a reta tangente à função no ponto x^0, e os demais x^{k+1} como sendo a intersecção entre o eixo horizontal e a reta paralela à tangente à função no ponto x^0, que passa por x^k.

A convergência deste método não é quadrática [STR86] e, assim como ocorria com o método de *Newton-Raphson*, para que esta ocorra é necessário que a posição inicial escolhida seja suficientemente próxima da raiz.

3.3.3 FLUXOGRAMA

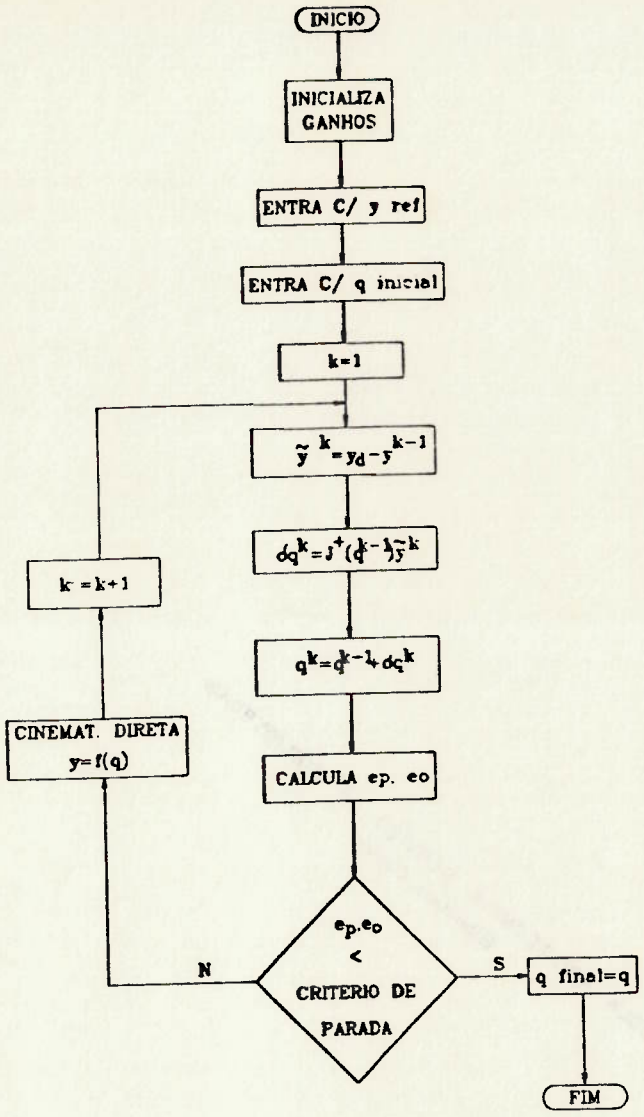


Figura 3.1: Fluxograma do método de *Newton-Raphson*

3.3.4 RESULTADOS DE SIMULAÇÕES

Nesta seção, mostra-se o comportamento do método, em simulações, para as seguintes situações: vetor q de partida próximo do vetor q esperado [2] (cada elemento do vetor de coordenadas de junta de partida difere de 5 graus ou 10 graus do vetor de coordenadas de junta correspondente à posição de referência), vetor q de partida distante do vetor q esperado (cada elemento do vetor de coordenadas de junta de partida difere de 30 graus, 50 graus ou 90 graus do vetor de coordenadas de junta correspondente à posição de referência). Para estes casos, o critério de parada foi $\delta_p < 1.10^{-4} m$ e $\delta_o < 0.1$ *grau*.

Além disto, mostra-se o comportamento do método com critérios de parada mais

[2] Para efeito de simulações, conhecia-se, de antemão, as coordenadas de junta correspondentes a posição de referência. A estas deu-se o nome de vetor de coordenadas de junta esperado.

rigorosos (δ_p<$1.10^{-5}m$, δ_o<0.01 *grau* e δ_p<$1.10^{-6}m$, δ_o<0.001 *grau*) e em posições de singularidade.

Os resultados apresentados abaixo podem ser considerados representativos do comportamento do algoritmo se comparados com as diversas simulações realizadas. O tempo de uma iteração num microcomputador PC 486 25 Mh para o robô ASEA IRBL6 é de 0.0041 s.

Os dados de cada simulação são apresentados de acordo com a tabela a seguir:

$\boldsymbol{y}$ de referência (m)
$\boldsymbol{q}$ esperado (graus)
$\boldsymbol{q}$ inicial (graus)
$\boldsymbol{y}$ final (m)
$\boldsymbol{q}$ final (graus)
número de iterações
critério de parada

Os casos a), b) e c) referem-se a simulações para posições não singulares.

a) Vetor $\boldsymbol{q}$ Inicial Próximo de $\boldsymbol{q}$ Esperado

Caso a.1: ($\boldsymbol{q}_i = \boldsymbol{q}$ *esperado* $+ 5graus$)

$\boldsymbol{y}_d = [0.10607, 0.90000, 1.28393, 0.65328, -0.27060, 0.27060]^t(m)$
$\boldsymbol{q}_{esp} = [90.0000, 90.0000, -90.0000, 90.0000, 45.0000, 0.00000]^t(graus)$
$\boldsymbol{q}_i = [95.0000, 95.0000, -85.0000, 95.0000, 50.0000, 5.00000]^t(graus)$
$\boldsymbol{y}_f = [0.10607, 0.90000, 1.28393, 0.65328, -0.27059, 0.27059]^t$ (m)
$\boldsymbol{q}_f = [90.0000, 90.0000, -90.0000, 90.0000, 45.0000, 0.00000]^t$ (graus)
número de iterações=4
critério de parada:δ_p<$1.10^{-4}m$ e δ_o<0.1 *grau*

Caso a.2: ($\boldsymbol{q}_i = \boldsymbol{q}$ *esperado* $+ 10graus$)

$\boldsymbol{y}_d = [0.10607, 0.90000, 1.28393, 0.65328, -0.27060, 0.27060]^t(m)$
$\boldsymbol{q}_{esp} = [90.0000, 90.0000, -90.0000, 90.0000, 45.0000, 0.0000]^t(graus)$
$\boldsymbol{q}_i = [100.000, 100.000, -80.0000, 100.000, 55.0000, 10.0000]^t(graus)$
$\boldsymbol{y}_f = [0.10607, 0.90000, 1.28393, 0.65328, -0.27060, 0.27060]^t$ (m)
$\boldsymbol{q}_f = [90.0000, 90.0000, -90.0000, 90.0000, 45.0000, 0.00000]^t$ (graus)
número de iterações=5
critério de parada: δ_p<$1.10^{-4}m$ e δ_o<0.1 *grau*

b) Vetor $\boldsymbol{q}$ Inicial Distante de $\boldsymbol{q}$ Esperado

Caso b.1: ($\boldsymbol{q}_i = \boldsymbol{q}$ *esperado* $+ 30graus$)

$\boldsymbol{y}_d = [0.10607, 0.90000, 1.28393, 0.65328, -0.27060, 0.27060]^t(m)$
$\boldsymbol{q}_{esp} = [90.0000, 90.0000, -90.0000, 90.0000, 45.0000, 0.00000]^t(graus)$
$\boldsymbol{q}_i = [120.000, 120.000, -60.0000, 120.000, 75.0000, 30.0000]^t(graus)$
$\boldsymbol{y}_f = [0.10607, 0.90000, 1.28393, 0.65328, -0.27060, 0.27060]^t$ (m)
$\boldsymbol{q}_f = [-270.000, 90.0000, -90.0000, 90.0000, 45.0000, -0.00000]^t$ (graus)
número de iterações=10
critério de parada: δ_p<$1.10^{-4}m$ e δ_o<0.1 *grau*

Caso b.2: ($\boldsymbol{q}_i = \boldsymbol{q}$ *esperado* + $50graus$)

$\boldsymbol{y}_d = [0.10607, 0.90000, 1.28393, 0.65328, -0.27060, 0.27060]^t(m)$
$\boldsymbol{q}_{esp} = [90.0000, 90.0000, -90.0000, 90.0000, 45.0000, 0.00000]^t(graus)$
$\boldsymbol{q}_i = [140.000, 140.000, -40.0000, 140.000, 95.0000, 50.0000]^t(graus)$
$\boldsymbol{y}_f = [0.10607, 0.90000, 1.28393, 0.65328, -0.27060, 0.27060]^t$ (m)
$\boldsymbol{q}_f = [-270.000, 18.395, -333.576, 225.181, -45.0000, -180.000]^t$ (graus)
número de iterações=25
critério de parada: δ_p<$1.10^{-4}m$ e δ_o<0.1 *grau*

Caso b.3: ($\boldsymbol{q}_i = \boldsymbol{q}$ *esperado* + $90graus$)

$\boldsymbol{y}_d = [0.10607, 0.90000, 1.28393, 0.65328, -0.27060, 0.27060]^t(m)$
$\boldsymbol{q}_{esp} = [90.0000, 90.0000, -90.0000, 90.0000, 45.0000, 0.00000]^t(graus)$
$\boldsymbol{q}_i = [180.000, 180.000, 0.00000, 180.000, 135.000, 90.0000]^t(graus)$
$\boldsymbol{y}_f = [0.10607, 0.90000, 1.28393, 0.65328, -0.27060, 0.27060]^t$ (m)
$\boldsymbol{q}_f = [-90.0000, 135.577, 26.4243, -72.0010, 135.000, 180.000]^t$ (graus)
número de iterações=10
critério de parada: δ_p<$1.10^{-4}m$ e δ_o<0.1 *grau*

Caso b.4: ($\boldsymbol{q}_i = \boldsymbol{q}$ *esperado* + $180graus$)

$\boldsymbol{y}_d = [0.10607, 0.90000, 1.28393, 0.65328, -0.27060, 0.27060]^t(m)$
$\boldsymbol{q}_{esp} = [90.0000, 90.0000, -90.0000, 90.0000, 45.0000, 0.00000]^t(graus)$
$\boldsymbol{q}_i = [270.000, 270.000, 90.0000, 270.000, 225.000, 180.0000]^t(graus)$
$\boldsymbol{y}_f = [0.10607, 0.90000, 1.28393, 0.65328, -0.27060, 0.27060]^t$ (m)
$\boldsymbol{q}_f = [89.9998, 1.68464, 90.0004, -1.68497, -315.000, -0.00024]^t$ (graus)
número de iterações=78
critério de parada: δ_p<$1.10^{-4}m$ e δ_o<0.1 *grau*

c) Critérios de Parada mais Rigorosos

Caso c.1: ($\boldsymbol{q}_i = \boldsymbol{q}$ *esperado* + $10graus$ - δ_p<$1.10^{-5}m$ e δ_o<0.01 *grau*)

$\boldsymbol{y}_d = [0.1060660, 0.9000000, 1.2839340, 0.6532815, -0.2705981, 0.2705981]^t(m)$
$\boldsymbol{q}_{esp} = [90.000000, 90.000000, -90.000000, 90.000000, 45.000000, 0.0000000]^t(graus)$
$\boldsymbol{q}_i = [100.00000, 100.00000, -80.000000, 100.00000, 55.000000, 10.000000]^t(graus)$
$\boldsymbol{y}_f = [0.1060660, 0.9000000, 1.28393340, 0.6532815, -0.2705981, 0.2705981]^t$ (m)
$\boldsymbol{q}_f = [90.000000, 90.000000, -90.000000, 90.000000, 45.000000, 0.0000000]^t$ (graus)
número de iterações=5
critério de parada: δ_p<$1.10^{-5}m$ e δ_o<0.01 *grau*

Caso c.2: ($\boldsymbol{q}_i = \boldsymbol{q}$ *esperado* + 30*graus* - δ_p<$1.10^{-5}m$ e δ_o<0.01 *grau*)

$\boldsymbol{y}_d = [0.1060660, 0.9000000, 1.2839340, 0.6532815, -0.2705981, 0.2705981]^t(m)$
$\boldsymbol{q}_{esp} = [90.000000, 90.000000, -90.000000, 90.000000, 45.000000, 0.0000000]^t(graus)$
$\boldsymbol{q}_i = [120.00000, 120.00000, -60.000000, 120.00000, 75.000000, 30.000000]^t(graus)$
$\boldsymbol{y}_f = [0.1060660, 0.9000000, 1.28393340, 0.6532815, -0.2705981, 0.2705981]^t$ (m)
$\boldsymbol{q}_f = [-270.00000, 90.000000, -90.000000, 90.000000, 45.000000, -0.0000000]^t$ (graus)
número de iterações=10
critério de parada: δ_p<$1.10^{-5}m$ e δ_o<0.01 *grau*

Caso c.3: ($\boldsymbol{q}_i = \boldsymbol{q}$ *esperado* + 30*graus* - δ_p<$1.10^{-6}m$ e δ_o<0.001 *grau*)

$\boldsymbol{y}_d = [0.1060660, 0.9000000, 1.2839340, 0.6532815, -0.2705981, 0.2705981]^t(m)$
$\boldsymbol{q}_{esp} = [90.000000, 90.000000, -90.000000, 90.000000, 45.000000, 0.0000000]^t(graus)$
$\boldsymbol{q}_i = [120.00000, 120.00000, -60.000000, 120.00000, 75.000000, 30.000000]^t(graus)$
$\boldsymbol{y}_f = [0.1060660, 0.9000000, 1.28393340, 0.6532815, -0.2705981, 0.2705981]^t$ (m)
$\boldsymbol{q}_f = [-270.00000, 90.000000, -90.000000, 90.000000, 45.000000, -0.0000000]^t$ (graus)
número de iterações=10
critério de parada: δ_p<$1.10^{-6}m$ e δ_o<0.001 *grau*

d) Posição de Singularidade

Caso d.1: ($\boldsymbol{y}_d$ no contorno do volume de trabalho do manipulador)

$\boldsymbol{y}_d = [0.106066, 1.590000, 0.593934, 0.653281, -0.270598, 0.270598]^t(m)$
$\boldsymbol{q}_{esp} = [90.00000, 0.000000, 0.000000, 90.00000, 45.00000, 0.000000]^t(graus)$
$\boldsymbol{q}_i = [100.0000, 10.00000, 10.00000, 100.0000, 55.00000, 10.00000]^t(graus)$
$\boldsymbol{y}_f = [0.106066, 1.589999, 0.593934, 0.653281, -0.270598, 0.270598]^t$ (m)
$\boldsymbol{q}_f = [90.00000, -0.054835, -0.111347, 89.94338, 45.00000, 0.000044]^t$ (graus)
número de iterações=10
critério de parada: δ_p<$1.10^{-5}m$ e δ_o<0.01 *grau*

Caso d.2: ($\boldsymbol{y}_d$ em posição de redundância local do manipulador)

Para o manipulador ASEA IRBL6, sempre que o eixo da junta 5 fica paralelo aos eixos das juntas 2, 3 e 4, existem infinitas combinações de q[2], q[3], q[4] e q[6] que fornecem a mesma posição e orientação do efetuador.

$\boldsymbol{y}_d = [0.150000, 0.900000, 1.390000, 0.500000, -0.500000, 0.500000]^t (m)$
$\boldsymbol{q}_i = [100.0000, 100.0000, -80.00000, 100.0000, 10.00000, 10.00000]^t (graus)$
$\boldsymbol{y}_f = [0.150000, 0.900000, 1.390000, 0.500000, -0.500000, 0.500000]^t$ (m)
$\boldsymbol{q}_f = [90.00000, 72.93243, -51.22456, -2.676225, 0.000000, 70.96834]^t$ (graus)
número de iterações=6
critério de parada: δ_p<$1.10^{-5}m$ e δ_o<0.01 *grau*

3.3.5 COMPORTAMENTO NAS SIMULAÇÕES

Chegou-se às seguintes conclusões sobre o comportamento do método nas simulações:

- O número de iterações necessárias para convergência foi menor para os casos em que o vetor de coordenadas de junta de partida estava próximo do vetor de coordenadas de junta esperado. Quando utilizou-se critério de parada δ_p<$1.10^{-4}m$ e δ_o<0.1 *grau*, em todas as simulações o algoritmo convergiu em menos de 8 iterações para o vetor $\boldsymbol{q}$ de partida próximo de $\boldsymbol{q}_{esp}$. Para $\boldsymbol{q}$ de partida distante de $\boldsymbol{q}_{esp}$, em algumas simulações o algoritmo não convergiu.

- Em simulações nas quais o vetor $\boldsymbol{q}$ de partida estava distante de $\boldsymbol{q}_{esp}$, não era possível prever para qual das soluções da cinemática inversa o algoritmo iria convergir.

- Critérios de parada mais rigorosos (δ_p<$1.10^{-5}m$ e δ_o<0.01 *grau* e δ_p<$1.10^{-6}m$ e δ_o<0.001 *grau*) influem muito pouco no número de iterações necessárias para convergência.

- Nas simulações em que a raiz era posição de singularidade (manipulador localmente redundante ou posição no volume de trabalho) o algoritmo comportou-se como nos casos em que não ocorria singularidade.

- Em alguns casos, principalmente quando o vetor $\boldsymbol{q}$ de partida estava distante de $\boldsymbol{q}_{esp}$, o algoritmo demandava um número muito grande de iterações para convergir (em algumas simulações, mesmo após 200 iterações não havia ocorrido convergência). Isto pode ser explicado pelo afastamento em relação a raiz (característico do método de *Newton-Raphson*) provocado quando o algoritmo passa, em alguma iteração intermediária, por posições próximas de posições singulares.

3.4 MÉTODO DE NEWTON-RAPHSON ADAPTADO

Grande parte do tempo de uma iteração do método de *Newton-Raphson* é dispendido na computação dos elementos da matriz Jacobiano e no cálculo de sua pseudo-inversa. O método de *Newton-Raphson* modificado, apresentado acima, necessita desta computação somente na primeira iteração, de forma que, em média, cada iteração dispende um tempo

bem menor que o algoritmo original. Esta característica sugere que, quando se conhece uma boa aproximação inicial para o vetor de coordenadas de junta (requisito necessário para a convergência deste método), situação que normalmente ocorre em casos de trajetória (o vetor $\boldsymbol{q}$ inicial para o ponto seguinte é calculado no ponto anterior, próximo ao seguinte), este algoritmo seja utilizado.

Com base nas considerações acima, neste ítem propõe-se aplicar o método de *Newton-Raphson* modificado, para casos de trajetória ($\boldsymbol{y}$ é variável) e sugere-se uma adaptação que aumenta a sua velocidade para estes casos. Assim, calcula-se $\boldsymbol{q}$ para diversos pontos da trajetória e computa-se os elementos da matriz $\boldsymbol{J}$ e sua pseudo-inversa de n em n pontos, conforme fluxograma a seguir. A este novo algoritmo, dá-se o nome de método de *Newton-Raphson* adaptado.

3.4.1 FLUXOGRAMA

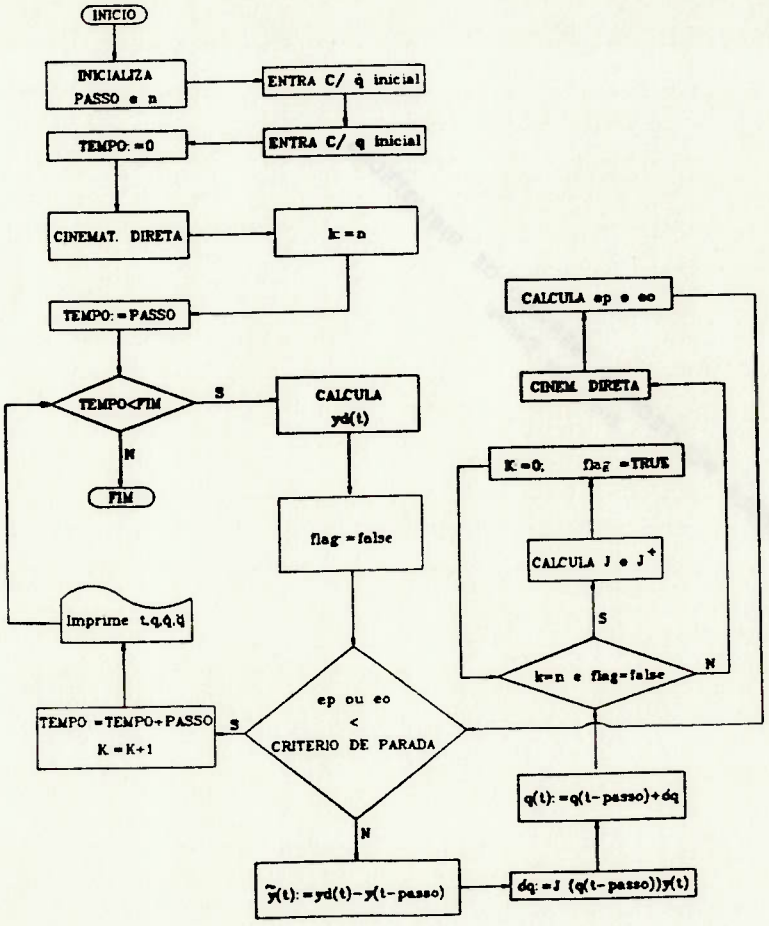


Figura 3.2: Método de *Newton-Raphson* adaptado

3.4.2 RESULTADOS DE SIMULAÇÕES

Esta seção ilustra o comportamento do método nas seguintes condições:

Diferentes valores de n (n é um número inteiro que indica de quantas em quantas iterações são computados os elementos da matriz $\boldsymbol{J}$); diferentes valores de passo; diferentes velocidades do efetuador; diferentes critérios de convergência e trajetória passando por posições singulares.

Para isto realizou-se estudos paramétricos com a seguinte trajetória (trajetória 1), [3] que também pode ser vista na Fig. 3.3:

$$\boldsymbol{y}_d = \begin{bmatrix} 0.07500 + 0.1sin(\psi) \\ 0.90000 \\ 1.16010 + 0.1cos(\psi) \\ 0.68301 \\ -0.18301 \\ 0.18301 \end{bmatrix} \tag{3.8}$$

onde

$$\psi = \begin{cases} \pi.t/6 - sin(\pi.t/3)/2 & \text{se } t < 3 \\ \pi/2 + \pi.(t-3)/3 & \text{se } 3 \leq t < 6 \\ 3.\pi/2 + \pi.(t-6)/6 + sin(\pi.(9-t)/3)/2 & \text{se } 6 \leq t < 9 \end{cases} \tag{3.9}$$

Trata-se de uma trajetória circular, no plano $\boldsymbol{X}_0\boldsymbol{Z}_0$, de raio 0.1 m, centro em $[0.07500, 0.90000, 1.16010]^t$ m e possui orientação (três primeiros parâmetros de *Euler Rodrigues*) $[0.68301, -0.18301, 0.18301]^t$.

Para casos em que o estudo requer maior variação absoluta das localizações de início e de fim da trajetória, adotou-se a seguinte (trajetória 2), que também pode ser vista na Fig. 3.4:

$$\boldsymbol{y}_d = \begin{bmatrix} 0.106066 + 0.278982.sin(\psi/4) \\ 0.900000 - 1.716922.sin(\psi/4) \\ 1.283934 - 0.023838.sin(\psi/4) \\ -0.184708.sin(2.873658.\psi/(4\pi)) + 0.653281.cos(2.873658.\psi/(4\pi)) \\ -0.724054.sin(2.873658.\psi/(4\pi)) - 0.270598.cos(2.873658.\psi/(4\pi)) \\ -0.646771.sin(2.873658.\psi/(4\pi)) + 0.270598.cos(2.873658.\psi/(4\pi)) \end{bmatrix} \tag{3.10}$$

onde ψ é dado pelas mesmas expressões que em (3.9).

Esta é uma trajetória retilínea (executada freqüentemente por robôs industriais) entre os pontos definidos por $\boldsymbol{y}$=$[0.106066, 0.9, 1.283933, 0.653281, -0.270598, 0.270598]^t$ e por $\boldsymbol{y}$=$[0.385048, -0.816923, 1.260096, -0.095796, -0.753709, -0.604833]^t$, e com velocidades inicial e final nulas. Os vetores de coordenadas generalizadas de junta correspondentes às posições inicial e final são, respectivamente, $\boldsymbol{q}$=$[\pi/2, \pi/2, -\pi/2, \pi/2, \pi/4, 0]^t$ e $\boldsymbol{q}$=$[-\pi/3, \pi/2, -\pi/2, \pi/2, \pi/3, \pi/10]^t$.

[3] Em [CHE89,GUP85] utilizou-se trajetórias semelhantes para verificação do comportamento de métodos numéricos de cinemática inversa.

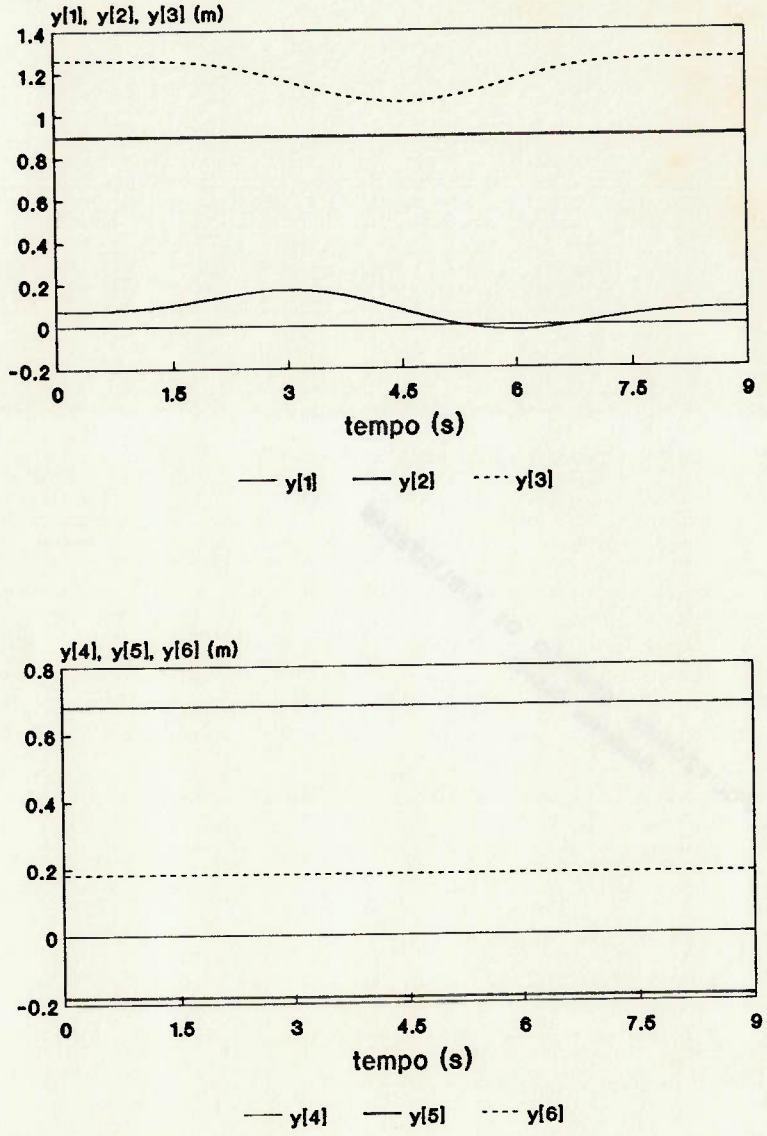
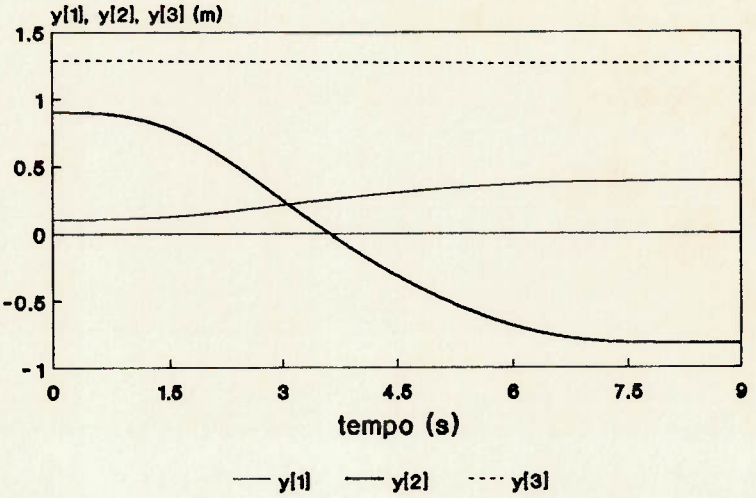


Figura 3.3: Trajetória 1



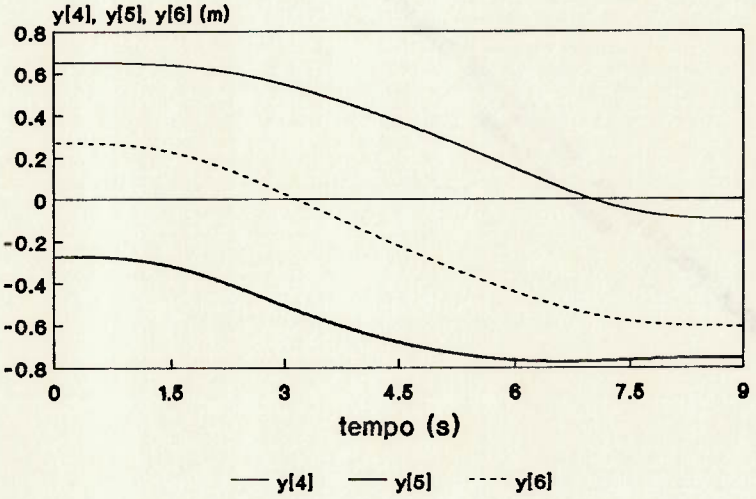


Figura 3.4: Trajetória 2

O comportamento característico do método durante a execução das trajetórias 1 e 2 pode ser observado nas Figs. 3.5, 3.6, 3.7 e 3.8 (correspondentes aos casos c.1 com $c = 1$ e c.2 com $c = 1$).

a)Diferentes Valores de n e *passo*

Nas tabelas abaixo, mostra-se o tempo necessário para a execução da trajetória 2 para diferentes valores de n e *passo*. Nestas simulações, os erros de posição e orientação máximos permitidos foram de, respectivamente, 1.10^{-4} e 0.1 *grau* e o numero máximo de iterações por ponto permitido foi 10 (nc significa que a trajetória não foi completada com sucesso).

No caso a.2, tabela-se o número médio de iterações por ponto em função dos mesmos parâmetros do caso a.1.

Caso a.1:

passo	n							
s	1	5	8	10	15	20	50	100
0.0005	76.7	40.3	36.8	35.8	34.2	33.5	32.1	31.7
0.003	13.5	7.42	6.81	6.64	6.42	6.31	6.42	6.87
0.005	8.12	4.51	4.23	4.17	4.07	4.07	4.28	nc
0.007	5.87	3.35	3.19	3.13	3.07	3.13	nc	nc
0.01	4.12	2.41	2.31	2.31	2.30	2.36	nc	nc
0.03	1.48	0.99	nc	nc	nc	nc	nc	nc
0.05	0.93	nc	nc	nc	nc	nc	nc	nc
0.1	0.49	nc	nc	nc	nc	nc	nc	nc
0.2	0.28	nc	nc	nc	nc	nc	nc	nc

Caso a.2:

passo	n							
s	1	5	8	10	15	20	50	100
0.0005	1.49	1.49	1.49	1.49	1.49	1.49	1.49	1.49
0.003	1.74	1.74	1.74	1.74	1.75	1.77	1.87	2.07
0.005	1.78	1.78	1.81	1.84	1.88	1.92	2.15	nc
0.007	1.80	1.86	1.91	1.94	2.00	2.07	nc	nc
0.01	1.82	1.96	2.03	2.08	2.18	2.28	nc	nc
0.03	2.12	2.51	nc	nc	nc	nc	nc	nc
0.05	2.34	nc	nc	nc	nc	nc	nc	nc
0.1	2.82	nc	nc	nc	nc	nc	nc	nc
0.2	3.60	nc	nc	nc	nc	nc	nc	nc

b)Diferentes Velocidades do Efetuador

Para esta análise, manteve-se os valores de *passo*, n, δ_p e δ_o fixos em, respectivamente, 0.01 s, 15, 1.10^{-4} e 0.1 *grau*, e modificou-se o raio da trajetória (1) para um valor

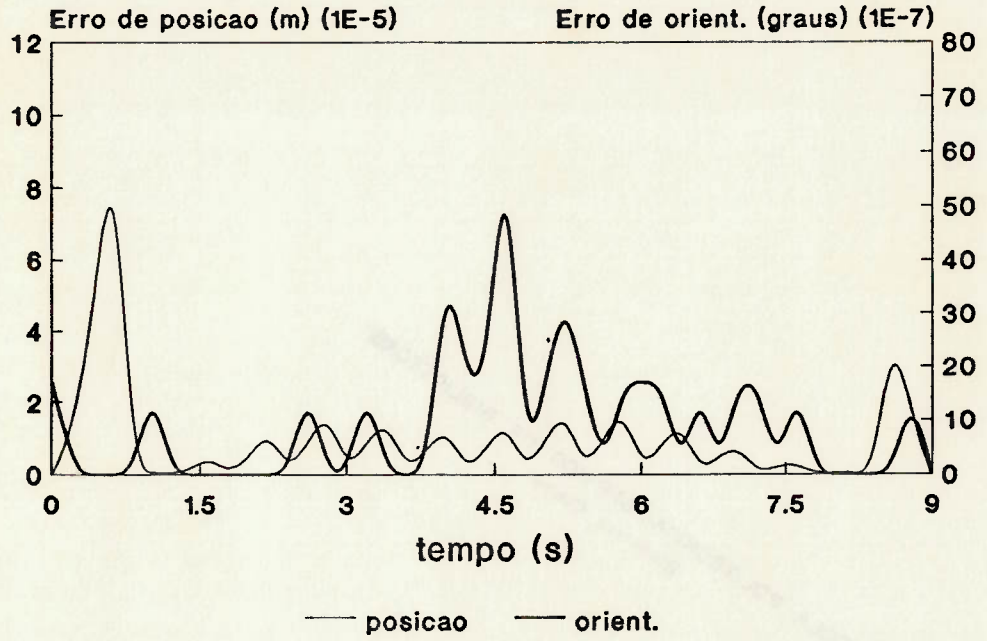
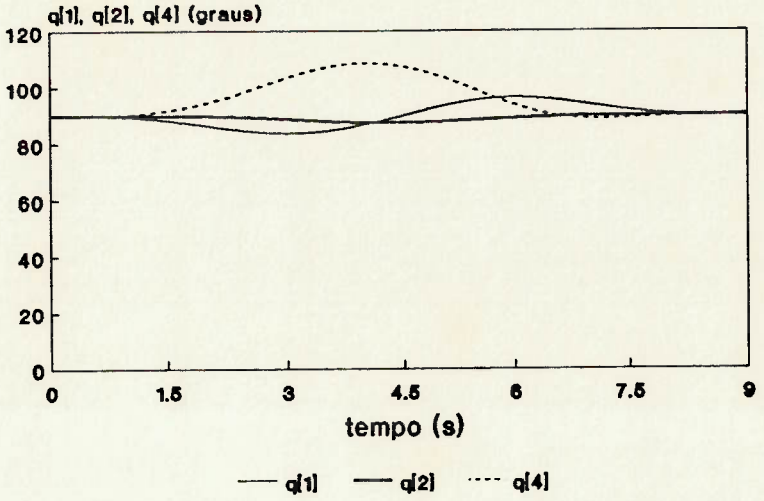


Figura 3.5: δ_p e δ_o x tempo: trajetória 1



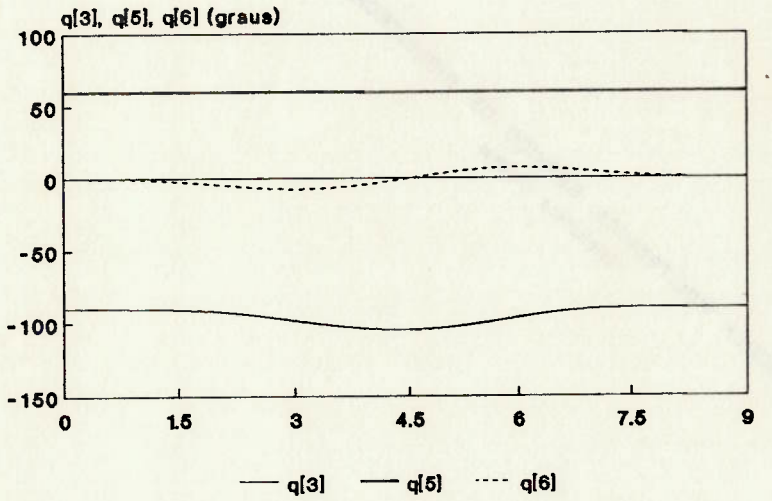


Figura 3.6: Coordenadas generalizadas de junta x tempo: trajetória 1

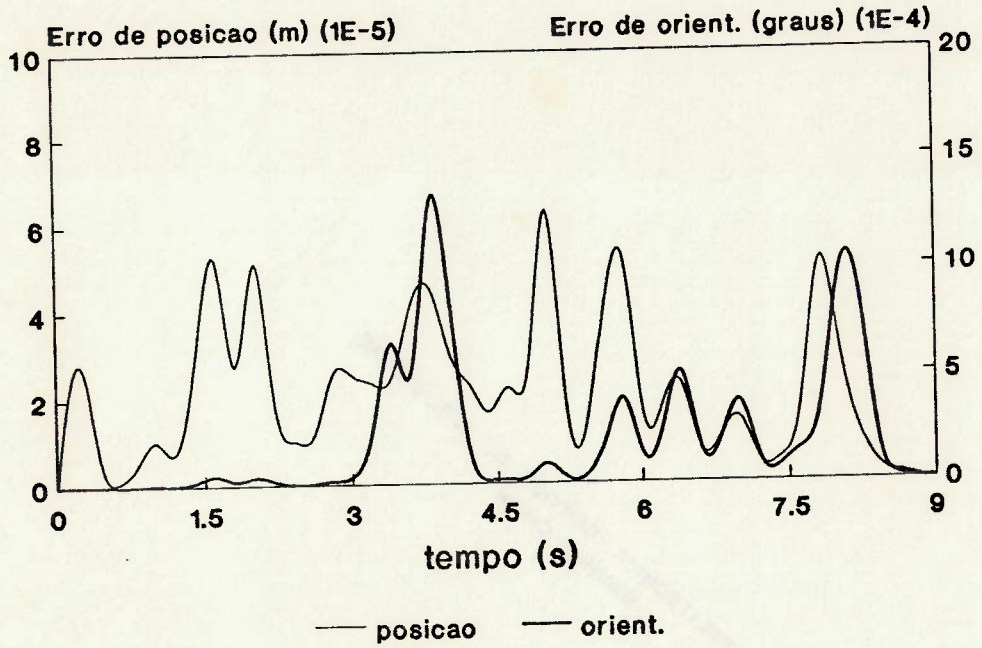


Figura 3.7: δ_p e δ_o x tempo: trajetória 2

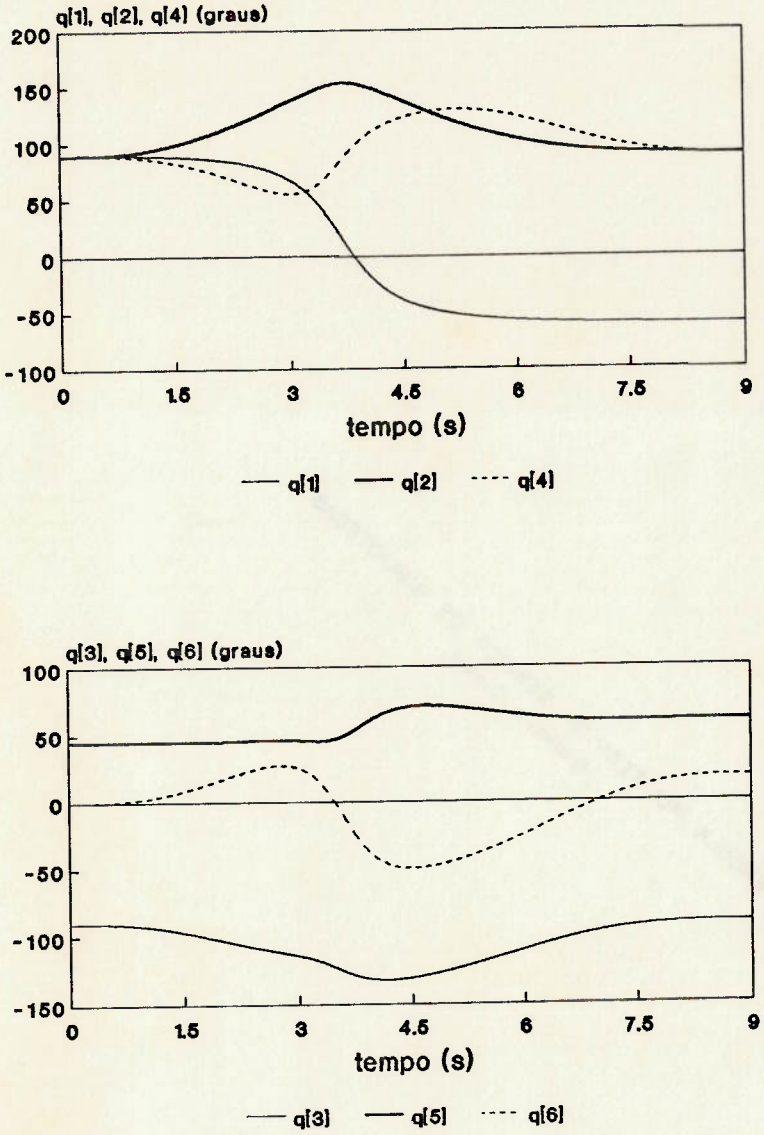


Figura 3.8: Coordenadas generalizadas de junta x tempo: trajetória 2

r (este valor r é tal que a trajetória não ultrapassa o volume de trabalho do manipulador). Com esta alteração do raio, provocou-se modificação na velocidade máxima do efetuador. Tabelou-se, então, o raio r, a máxima velocidade correspondente ao raio r e o número de iterações médio por ponto durante a trajetória.

Caso b:

r	$\| v_{max} \| \; m/s$	num. iter. médio
0.05	0.05236	1.81
0.06	0.06283	1.83
0.07	0.07329	1.84
0.08	0.08378	1.85
0.09	0.09425	1.86
0.10	0.10472	1.87
0.11	0.11519	1.87
0.12	0.12566	1.88
0.15	0.15708	1.89
0.20	0.17802	1.91
0.90	0.94248	2.74

c)Diferentes Critérios de Convergência

Com $passo = 0.01$ e $n = 15$, verificou-se o comportamento do algoritmo nas trajetórias 1 (caso c.1) e 2 (caso c.2) para diversos critérios de convergência. Tabelou-se, então, o número de iterações médio por ponto e o tempo para a execução da trajetória para diversos critérios de convergência (c é o fator que multiplica 1.10^{-4} e 0.1 *grau*, para os erros de posição e orientação, respectivamente).

Caso c.1:

c	num. iter. médio	tempo (s)
0.01	2.67	2.42
0.05	2.43	2.20
0.10	2.25	2.09
0.50	1.91	1.81
1.00	1.87	1.76
5.00	1.68	1.59
10.0	1.42	1.43
50.0	1.00	1.04

Caso c.2:

c	num. iter. médio	tempo (s)
0.01	3.47	3.36
0.05	3.02	2.96
0.10	2.84	2.80
0.50	2.37	2.41
1.00	2.18	2.31
5.00	1.79	1.97
10.0	1.63	1.87
50.0	1.00	1.32

d)Trajetória Passando por Posições Singulares

Para este estudo adotou-se trajetórias diferentes de (3.8) e (3.10).

No caso d.1, apresenta-se a simulação com uma trajetória em que o efetuador move-se no eixo $\boldsymbol{Y}$com orientação constante, até o ponto em que o mesmo fica totalmente esticado ($\boldsymbol{y}$= $[0.106066, 1.590000, 0.593934, 0.653281, -0.270598, 0.270598]^t(m)$), que é uma posição singular pertencente ao contorno do volume de trabalho do manipulador [4]. A trajetória de referência vai além deste ponto (até $\boldsymbol{y}[2]$=1.600000 (m)) de forma que o manipulador não pode acompanhá-la a partir do momento em que $\boldsymbol{y}[2]$=1.590000. Trata-se de um movimento harmônico simples, com período 9 s, entre as posições inicial e final especificadas na tabela do caso d.1. Esta tabela contém a posição inicial da trajetória ($\boldsymbol{y}_i$), a posição final desejada ($\boldsymbol{y}_d$) e a posição final alcançada ($\boldsymbol{y}_f$) no instante 4.07 s. A partir deste instante $\boldsymbol{y}[2]$ ultrapassa 1.590000 m e o algoritmo não mais conseguiu acompanhar a trajetória. Nestas simulações, *passo*, n, e os erros de posição e orientação máximos foram de, respectivamente, 0.001, 1, 1.10^{-4} e 0.1 *grau*.

No caso d.2, o manipulador executa a trajetória abaixo.

$$\boldsymbol{y}_d = \begin{bmatrix} 0.15000 \\ 0.90000 + 0.1sin(\psi) \\ 1.39000 + 0.1cos(\psi) \\ 0.50000 \\ -0.50000 \\ 0.50000 \end{bmatrix} \tag{3.11}$$

onde

$$\psi = \begin{cases} \pi.t/6 - sin(\pi.t/3)/2 & \text{se } t < 3 \\ \pi/2 + \pi.(t-3)/3 & \text{se } 3 \leq t < 6 \\ 3.\pi/2 + \pi.(t-6)/6 + sin(\pi.(9-t)/3)/2 & \text{se } 6 \leq t < 9 \end{cases}$$

Trata-se de uma trajetória circular, no plano $\boldsymbol{Y}_0\boldsymbol{Z}_0$, de raio 0.1 m, centro em $[0.15000, 0.90000, 1.39000]^t$ m e possui orientação (três primeiros parâmetros de *Euler Rodrigues*) $[0.50000, -0.50000, 0.50000]^t$.

[4] A mesma trajetória foi utilizada em [CHE89] para verificação do comportamento de método numérico em posição singular.

Durante toda a extensão da trajetória o manipulador é localmente redundante (existem infinitas combinações de q[2], q[3], q[4] e q[6] que fornecem a mesma posição e orientação do efetuador).

Nas Figs. 3.9 e 3.10 mostra-se os gráficos de δ_p e δ_o em função do tempo e das coordenadas generalizadas de junta em função do tempo. Nestas simulações, *passo*, n, e os erros de posição e orientação máximos foram de, respectivamente, 0.01, 15, 1.10^{-4} e 0.1 *grau*.

Caso d.1:

	$y_i(m)$	$y_d(m)$	$y_f(m)$
y[1]	0.106066	0.106066	0.106066
y[2]	1.400000	1.600000	1.589976
y[3]	0.593934	0.593934	0.593939
y[4]	0.653281	0.653281	0.653282
y[5]	-0.270598	-0.270598	-0.270598
y[6]	0.270598	0.270598	0.270598
q[1]	90.00000	impossi.	90.00000
q[2]	30.15469	impossi.	0.334599
q[3]	-61.30807	impossi.	-0.679185
q[4]	121.1533	impossi.	90.34459
q[5]	45.00000	impossi.	45.00000
q[6]	0.000046	impossi.	0.000000
t (s)	0	4.5	4.07

Caso d.2:

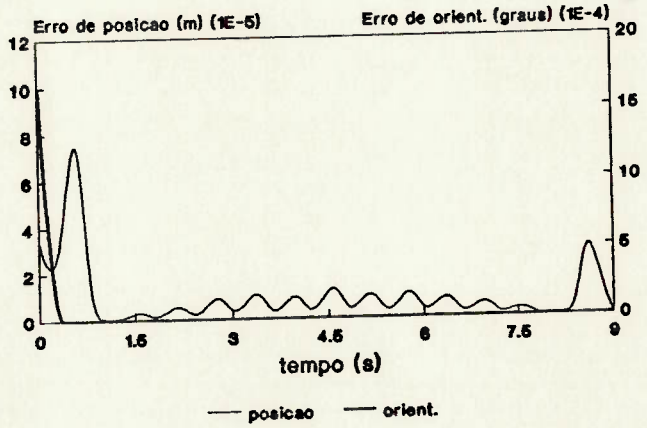


Figura 3.9: Redundância local: δ_p e δ_o x tempo

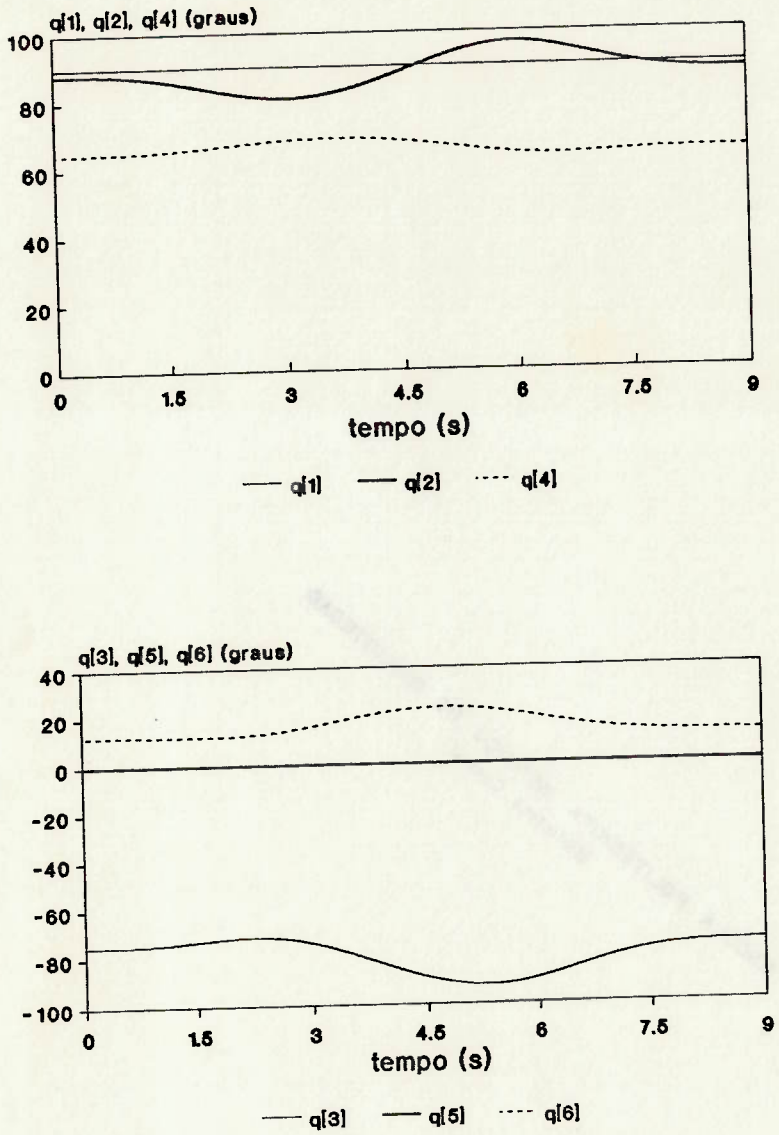


Figura 3.10: Redundância local: Coordenadas generalizadas de junta x tempo

3.4.3 COMPORTAMENTO NAS SIMULAÇÕES

Chegou-se às seguintes conclusões sobre o comportamento do método nas simulações:

- O método de *Newton-Raphson* modificado ($n = 1$) conseguiu trilhar a trajetória 2 (casos a) mesmo com valores de *passo* elevados ($0.2\ s$). Com valores de *passo* desta ordem, a adaptação proposta não pode ser aplicada com sucesso. Quando utilizou-se, entretanto, valores de *passo* menores que $0.03\ s$, o que ainda permite a execução da tajetória 2 em tempo real (até $passo = 0.003s$), a utilização da adaptação proposta foi aplicada com sucesso e aumenta bastante a velocidade do algoritmo (em alguns casos mais de 70 %).

- Para valores elevados de n, a trajetória não pode ser completada com sucesso (o número máximo de iterações por ponto é atingido).

  Valores elevados de n não são, necessariamente, os que fornecem menor tempo para execução da trajetória. Isto ocorre porque, apesar de quanto maior o valor de n menor o número de pontos em que os elementos de $\boldsymbol{J}$ e sua pseudo-inversa são calculados, quanto maior o valor de n maior o número médio de iterações por ponto.

- O número de iterações médio por ponto é tanto maior quanto maior a velocidade de execução da trajetória. Esta característica decorre da influência da distância entre o ponto de partida e a raiz desejada verificada no método de *Newton-Raphson* para ponto-fixo (acima), e que também ocorre no método de *Newton-Raphson* modificado para ponto-fixo.

  Tomando-se valores de n e *passo* adequados, é possível a utilização do método para velocidades elevadas (no caso b fez-se simulações com $1\ m/s$).

- A escolha do critério de convergência influencia bastante a velocidade do algoritmo.

- Em nenhuma simulação em que a trajetória passava por posições singulares em que o manipulador era localmente redundante ocorreu problema de convergência. Nestes casos o algoritmo comportou-se como nos casos em que não havia singularidade.

- Quando a trajetória ultrapassava o volume de trabalho do manipulador, o algoritmo não conseguia acompanhá-la. Neste tipo de simulação, até o momento em que o volume de trabalho era ultrapassado, o algoritmo acompanhava muito bem a trajetória.

Capítulo 4

MÉTODOS BASEADOS NA TRANSPOSTA DO JACOBIANO

4.1 INTRODUÇÃO

Os métodos apresentados no capítulo anterior, necessitam inverter matrizes. Visando evitar este processo, que é relativamente lento se comparado a outras operações matriciais e que, caso não se utilize inversas generalizadas (pseudo-inversa, por exemplo), o que torna o processo ainda mais lento, gera problemas numéricos em pontos de singularidade das matrizes, na metade da década de 80 começou a surgir uma nova classe de algoritmos iterativos. Estes solucionam a cinemática inversa sem necessitar inverter o Jacobiano, necessitam apenas transpor a matriz $\boldsymbol{J}$ [BAL84,ASA85,DAS88,SCI88,NOV90,CAB91,MAT91]. A origem destes métodos está em equações de teorias de controle e a convergência dos mesmos é demonstrada utilizando-se a teoria de *Lyapunov*.

Dentro destes algoritmos, conforme colocado na introdução do capítulo anterior, existem aqueles que lidam com o problema de determinar o vetor $\boldsymbol{q}$ dado um vetor $\boldsymbol{y}$ constante e existem aqueles que visam determinar $\boldsymbol{q}$ "trilhando" uma dada trajetória. Para o caso de $\boldsymbol{y}$ constante, foi apresentado apenas um método, cujas equações foram derivadas a partir da teoria de *"impedance control"* [ASA85]. Para o caso de $\boldsymbol{y}$ variável segundo uma dada trajetória, foram apresentados três métodos. As equações do primeiro [BAL84,SCI88], surgiram a partir da teoria de *Lyapunov*; o segundo [NOV90], originou-se na teoria de *"sliding mode control"* e o terceiro [CAB91], na teoria de *"impedance control"*.

Os algoritmos para trajetória, podem, ainda, ser classificados de acordo com a sua ordem. Assim, os de primeira ordem [BAL84,SCI88], geram velocidades generalizadas de junta e os de segunda ordem [NOV90,CAB91] geram acelerações generalizadas de junta, o que pode ser útil para fins de controle.

Neste capítulo, apresenta-se dois métodos baseados na transposta do Jacobiano. O primeiro [ASA85,DAS88], para o caso de $\boldsymbol{y}$ constante, e o segundo [NOV90], de segunda ordem, para os casos de trajetória. Quando estuda-se o segundo método, mostra-

se também, as equações de um terceiro [BAL84], também para os casos de trajetória, mas de primeira ordem. Ao final da apresentação de cada método, faz-se uma análise de resultados encontrados para o robô ASEA IRBL6, que ilustram seu comportamento em simulações.

4.2 MÉTODO BASEADO EM "IMPEDANCE CONTROL"

Considere a força $\boldsymbol{f}$ necessária para deslocar o manipulador da posição $\boldsymbol{y}$ com velocidade $\dot{\boldsymbol{y}}$ para a posição $\boldsymbol{y_d}$ com velocidade $\dot{\boldsymbol{y}}_d$. Esta força pode ser modelada conforme segue:

$$\boldsymbol{f} = (\boldsymbol{K_p}.\tilde{\boldsymbol{y}} + \boldsymbol{K_d}.\dot{\tilde{\boldsymbol{y}}}), \tag{4.1}$$

onde

$$\tilde{\boldsymbol{y}} = \boldsymbol{y_d} - \boldsymbol{y}, \quad \dot{\tilde{\boldsymbol{y}}} = \dot{\boldsymbol{y}}_d - \dot{\boldsymbol{y}},$$

$\boldsymbol{K_p}$ e $\boldsymbol{K_d}$ são matrizes definidas e positivas.

Pode-se relacionar $\boldsymbol{f}$ e $\boldsymbol{\tau}$ (vetor que contém as forças e torques nos atuadores do manipulador) pela expressão:

$$\boldsymbol{\tau} = \boldsymbol{J}^t(\boldsymbol{q}).\boldsymbol{f}. \tag{4.2}$$

Esta relação é facilmente deduzida igualando-se a potência necessária para a execução do movimento escrita no espaço de junta com a potência escrita no espaço de configuração do manipulador.

Como se sabe, a expressão geral para a dinâmica de um manipulador é dada pela equação abaixo:

$$\boldsymbol{\tau} = \boldsymbol{H}(\boldsymbol{q}).\ddot{\boldsymbol{q}} + \boldsymbol{D}(\boldsymbol{q}, \dot{\boldsymbol{q}}).\dot{\boldsymbol{q}} + \boldsymbol{g}(\boldsymbol{q}), \tag{4.3}$$

onde $\boldsymbol{H}$ é a matriz de inércia do manipulador, $\boldsymbol{D}$ é a matriz que contém os termos de aceleração centrífuga e de coriolis e $\boldsymbol{g}(\boldsymbol{q})$ é um vetor que contém os termos de gravidade. A partir de (4.1), (4.2) e (4.3) obtém-se a seguinte relação:

$$\boldsymbol{H}(\boldsymbol{q}).\ddot{\boldsymbol{q}} + \boldsymbol{D}(\boldsymbol{q}, \dot{\boldsymbol{q}}).\dot{\boldsymbol{q}} + \boldsymbol{g}(\boldsymbol{q}) = \boldsymbol{J}^t(\boldsymbol{q}).(\boldsymbol{K_p}.\tilde{\boldsymbol{y}} + \boldsymbol{K_d}.\dot{\tilde{\boldsymbol{y}}}). \tag{4.4}$$

Esta expressão permite controlar a iteração de um manipulador com o meio ambiente, ou seja permite controlar a força que o manipulador aplica no meio ambiente. A este controle dá-se o nome de *"impedance control"*. Observando-se (4.4), percebe-se que esta equação relaciona coordenadas de junta com coordenadas de espaço do manipulador. Isto sugere a sua utilização para realização da cinemática inversa.

A utilização de *"impedance control"* para realizar a cinemática inversa pode ser entendida fisicamente como a aplicação de uma força externa, imaginária no efetuador de um manipulador cinematicamente equivalente (mesma geometria), mas dinamicamente mais simples (parâmetros dinâmicos tais que simplifiquem as equações dinâmicas) na direção e sentido da posição e velocidade desejadas. O módulo desta força é proporcional à distância

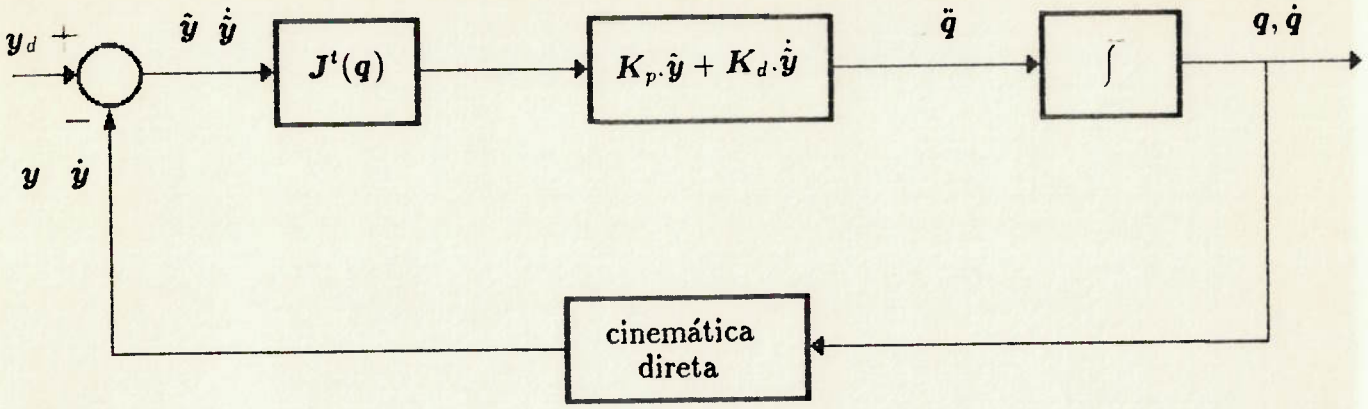


Figura 4.1: Diagrama de blocos do algoritmo

entre a posição atual y e a posição desejada y_d e à diferença entre a velocidade atual $\dot{y}$ e a velocidade desejada $\dot{y}_d$. As constantes de proporcionalidade são, respectivamente, duas matrizes definidas e positivas K_p e K_d. A partir destas considerações, e observando-se a expressão (4.4), pode-se derivar o seguinte algoritmo, cujo diagrama de blocos pode ser visto na Fig. 4.1 (neste diagrama, o termo cinemática direta está sendo utilizado para indicar as transformações $y = y(q)$ e $\dot{y} = J(q).\dot{q}$):

$$\ddot{q} = J^t(q).(K_p.\tilde{y} + K_d.\dot{\tilde{y}}), \tag{4.5}$$

onde

$$\tilde{y} = y_d - y(q) \quad e \quad \dot{\tilde{y}} = \dot{y}_d - \dot{y}(q).$$

Note que esta expressão foi deduzida a partir de (4.4) tomando-se H=matriz identidade, D=0 (forças centrífugas e de coriolis nulas) e g=0 (gravidade nula).

A seguir demonstra-se a estabilidade deste método, para o caso de $\dot{y} = 0$ [1].

4.2.1 DEMONSTRAÇÃO DA ESTABILIDADE

Seja a seguinte função de *Lyapunov*,

$$V = 1/2.(\tilde{y}^t.K_p.\tilde{y} + \dot{q}^t.\dot{q}), \tag{4.6}$$

Diferenciando-se esta função e fazendo-se $\dot{y}_d = 0$, tem-se:

$$\dot{V} = \dot{\tilde{y}}^t.K_p.\tilde{y} + \dot{q}^t.\ddot{q}. \tag{4.7}$$

Substituindo-se (4.5) na equação acima, resulta:

$$\dot{V} = \dot{\tilde{y}}^t.K_p.\tilde{y} + \dot{q}^t.J^t.(K_p.\tilde{y} + K_d.\dot{\tilde{y}}), \tag{4.8}$$

[1] Em [CAB91] propõe-se a aplicação deste método para o caso de $\dot{y}_d$ diferente de 0.

e lembrando-se que $\dot{\boldsymbol{y}} = \boldsymbol{J}.\dot{\boldsymbol{q}}$ ($\dot{\boldsymbol{y}}^t = \dot{\boldsymbol{q}}^t.\boldsymbol{J}^t$), tem-se:

$$\dot{V} = -\dot{\boldsymbol{y}}^t.\boldsymbol{K}_d.\dot{\boldsymbol{y}} \leq 0, \tag{4.9}$$

ou seja, o algoritmo (4.5) converge. Resta saber se o ponto de convergência é, ou não, o ponto desejado ($\boldsymbol{y}$=$\boldsymbol{y}_d$ e $\dot{\boldsymbol{q}} = 0$). Segundo *Lyapunov*, este ponto é aquele para o qual $\dot{V}$ se anula.

Como $\dot{V}$ é menor ou igual a zero, a partir do momento em que $\dot{V}$ atinge o valor zero, continua sempre com este valor.

Seja t_{eq} o instante em que $\dot{V}$ se anula. Observando-se (4.9), nota-se que para todo t maior ou igual a t_{eq} $\dot{\boldsymbol{y}}$ também se anula. Se $\boldsymbol{J}$ é de posto cheio e se o manipulador não é redundante [2], $\dot{\boldsymbol{y}} = 0 \rightarrow \dot{\boldsymbol{q}} = 0$. Ou seja, neste caso, para todo $t \geq t_{eq}$ $\dot{\boldsymbol{q}}$ é igual a zero e, portanto, $\ddot{\boldsymbol{q}}$ também é igual a zero. Desta forma, como $\ddot{\boldsymbol{q}}$ vale zero e $\boldsymbol{J}$ é de posto cheio, pode-se concluir, a partir de (2.36), que $\tilde{\boldsymbol{y}} = 0$.

Nos casos em que $\boldsymbol{J}$ não possui posto cheio ou em que o manipulador é redundante, a hipótese de que $\dot{\boldsymbol{y}} = 0 \rightarrow \dot{\boldsymbol{q}} = 0$ não é válida. Nestes casos, pode-se concluir que $\boldsymbol{y}$ tende para uma constante ($\dot{\boldsymbol{y}} = 0$) e, como V também tende para uma constante ($\dot{V}$ tende a zero), $\dot{\boldsymbol{q}}$ tende, novamente, para uma constante (4.6). Assim, o manipulador pode estabilizar ($\dot{\boldsymbol{y}} = 0$) com $\dot{\boldsymbol{q}} \neq 0$ e com $\tilde{\boldsymbol{y}} \neq 0$. Nas simulações realizadas (casos d), a primeira situação ocorreu quando a posição de referência era localmente redundante [3]. Neste caso o manipulador convergiu para $\boldsymbol{y} = \boldsymbol{y}_d$, mas as coordenadas de junta não convergiram para uma valor fixo (ficaram oscilando entre as infinitas soluções existentes para a cinemática inversa). O algoritmo parou quando encontrou um vetor de coordenadas generalizadas de junta que satisfez o critério de parada. A segunda situação ocorreu quando a posição de referência estava fora do volume de trabalho do robô. Neste caso o algoritmo convergia para um valor diferente de $\boldsymbol{y}$=$\boldsymbol{y}_d$.

A seguir apresenta-se o fluxograma do algoritmo.

---

[2]Como pode ser visto no capítulo 5, quando um manipulador é redundante, o posto de $\boldsymbol{J}$ é menor que o número de colunas de $\boldsymbol{J}$.

[3]No manipulador ASEA IRBL6, sempre que o eixo da junta 6 fica paralelo aos eixos das juntas 2, 3 e 4, ocorre redundância local (quatro eixos de junta paralelos).

4.2.2 FLUXOGRAMA

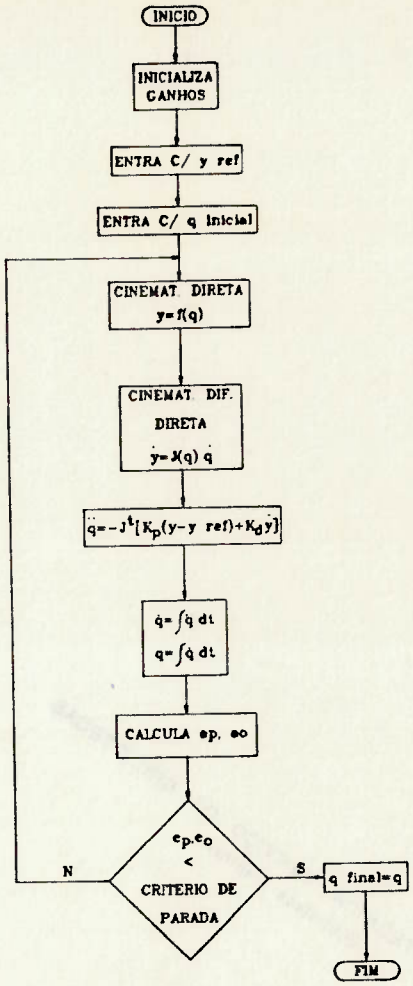


Figura 4.2: Fluxograma do algoritmo baseado em *"impedance control"*

4.2.3 RESULTADOS DE SIMULAÇÕES

Nesta seção, mostra-se o comportamento do método, em simulações, para as seguintes situações: vetor $\boldsymbol{q}$ de partida próximo do vetor $\boldsymbol{q}$ esperado [4] (cada elemento do vetor de coordenadas de junta de partida difere de 5 graus ou 10 graus do vetor de coordenadas de junta correspondente à posição de referência), vetor $\boldsymbol{q}$ de partida distante do vetor $\boldsymbol{q}$ esperado (cada elemento do vetor de coordenadas de junta de partida difere de 30 graus, 50 graus ou 90 graus do vetor de coordenadas de junta correspondente à posição de referência). Para estes casos, o critério de parada foi δ_p<$1.10^{-4}m$ e δ_o<$0.1\ grau$.

Além disto, mostra-se o comportamento do método com critérios de parada mais rigorosos (δ_p<$1.10^{-5}m$, δ_o<$0.01\ grau$ e δ_p<$1.10^{-6}m$, δ_o<$0.001\ grau$), em posições de singularidade, e apresenta-se resultados que ilustram a sua sensibilidade elevada aos ganhos $\boldsymbol{K}_p$ e $\boldsymbol{K}_d$.

[4] Assim, como ocorria no capítulo anterior, para efeito de simulações, conhecia-se, de antemão, as coordenadas de junta correspondentes à posição de referência. A estas deu-se o nome de vetor de coordenadas de junta esperado.

Os resultados apresentados abaixo podem ser considerados representativos do comportamento do algoritmo se comparados com as diversas simulações realizadas. O tempo de uma iteração num microcomputador PC 486 25 Mh para o robô ASEA IRBL6 é de 0.0020 s.

Os dados de cada simulação são apresentados de acordo com a tabela a seguir:

$\boldsymbol{y}$ de referência (m)
$\boldsymbol{q}$ esperado (graus)
$\boldsymbol{q}$ inicial (graus)
$\boldsymbol{y}$ final (m)
$\boldsymbol{q}$ final (graus)
número de iterações, passo
$\boldsymbol{K}_p$
$\boldsymbol{K}_d$
critério de parada

Os casos a), b) e c) referem-se a simulações para posições não singulares.

No caso d.1, $\boldsymbol{y}$ de referência está fora do volume de trabalho do manipulador mas a uma distância do mesmo menor que o critério de parada, de forma que o algoritmo possa convergir. Neste caso, a convergência se dá para uma posição singular pertencente ao volume de trabalho.

O comportamento característico do método, até a convergência, pode ser observado nas Figs. 4.3 e 4.4 (correspondentes ao caso b.1).

a) Vetor $\boldsymbol{q}$ Inicial Próximo de $\boldsymbol{q}$ Esperado

Caso a.1: $(\boldsymbol{q}_i = \boldsymbol{q}\ esperado + 5graus)$

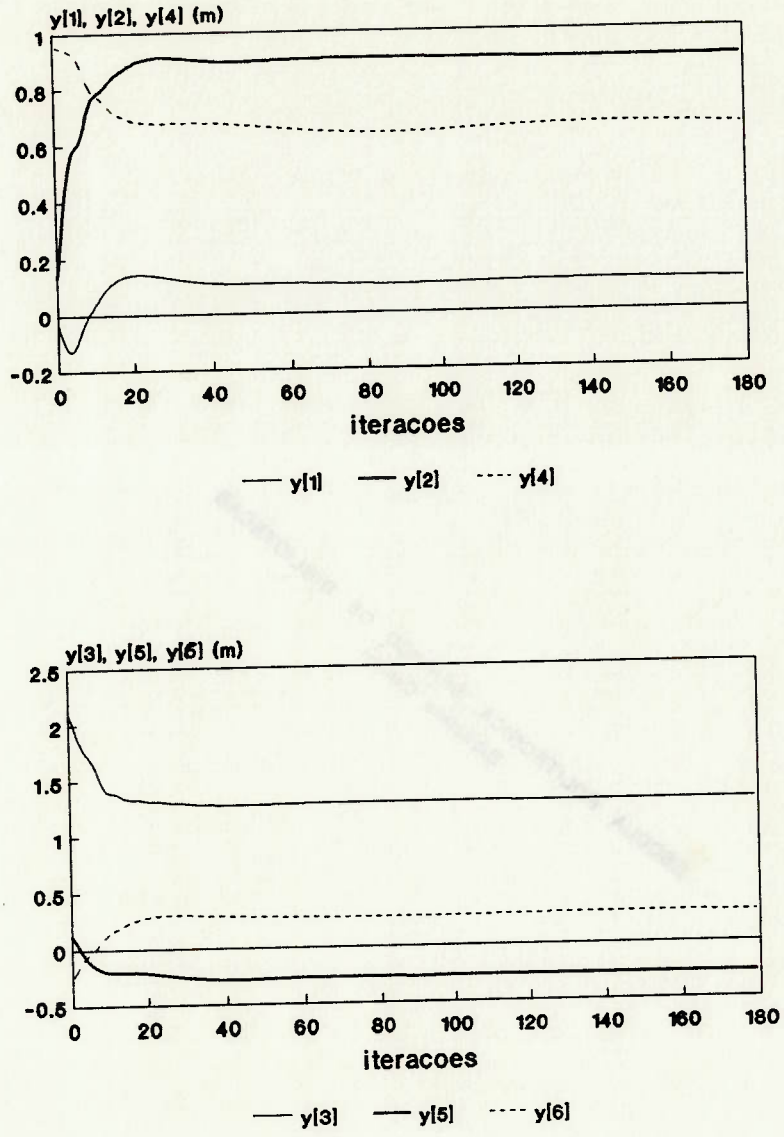


Figura 4.3: Vetor de posição e orientação x tempo

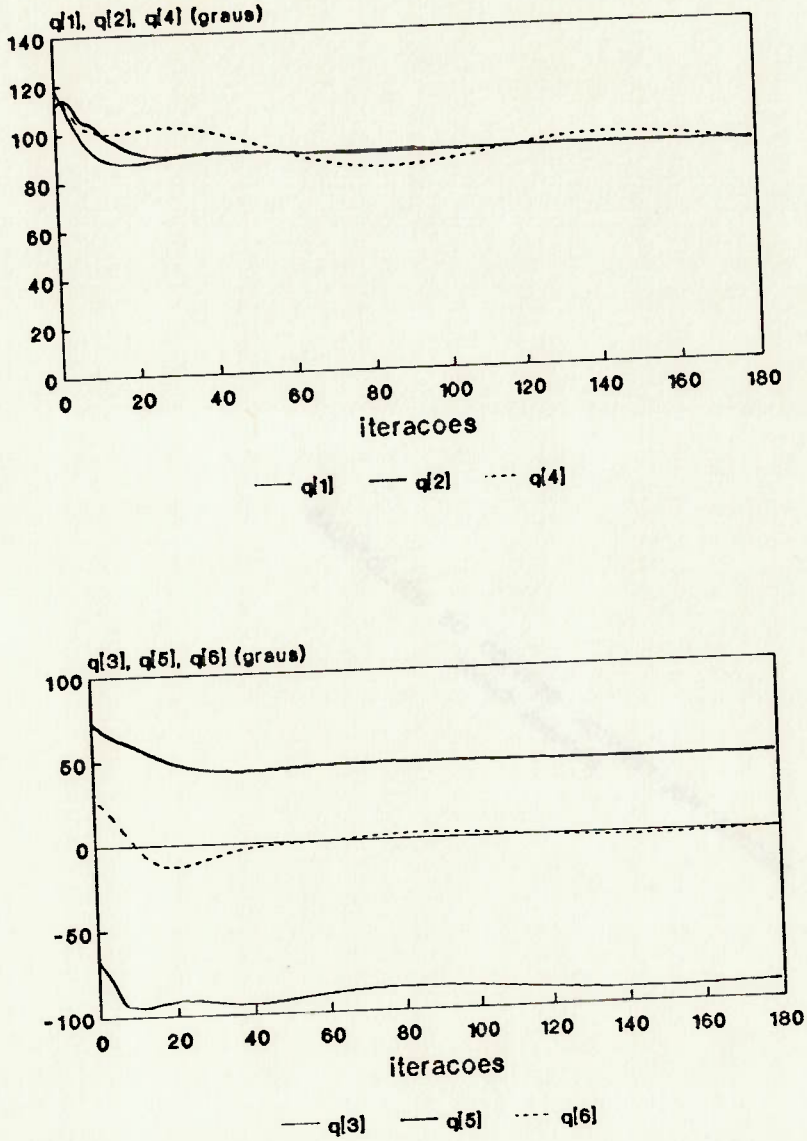


Figura 4.4: Coordenadas generalizadas de junta x tempo

$y_d = [0.10607, 0.90000, 1.28393, 0.65328, -0.27060, 0.27060]^t (m)$
$q_{esp} = [90.0000, 90.0000, -90.0000, 90.0000, 45.0000, 0.00000]^t (graus)$
$q_i = [95.0000, 95.0000, -85.0000, 95.0000, 50.0000, 5.00000]^t (graus)$
$y_f = [0.10598, 0.89997, 1.28393, 0.65319, -0.27071, 0.27055]^t$ (m)
$q_f = [90.0057, 89.9996, -89.9930, 89.9699, 44.9942, 0.02850]^t$ (graus)
número de iteraçôes=81, passo=0.0001

$$K_p = \begin{bmatrix} 18.10^6 & 0 & 0 & 0 & 0 & 0 \\ 0 & 18.10^6 & 0 & 0 & 0 & 0 \\ 0 & 0 & 18.10^6 & 0 & 0 & 0 \\ 0 & 0 & 0 & 18.10^6 & 0 & 0 \\ 0 & 0 & 0 & 0 & 18.10^6 & 0 \\ 0 & 0 & 0 & 0 & 0 & 18.10^6 \end{bmatrix}$$

$$K_d = \begin{bmatrix} 7800 & 0 & 0 & 0 & 0 & 0 \\ 0 & 7800 & 0 & 0 & 0 & 0 \\ 0 & 0 & 7800 & 0 & 0 & 0 \\ 0 & 0 & 0 & 7800 & 0 & 0 \\ 0 & 0 & 0 & 0 & 7800 & 0 \\ 0 & 0 & 0 & 0 & 0 & 7800 \end{bmatrix}$$

critério de parada: $\delta_p < 1.10^{-4} m$ e $\delta_o < 0.1\ grau$

Caso a.2: $(q_i = q\ esperado + 10 graus)$

$y_d = [0.10607, 0.90000, 1.28393, 0.65328, -0.27060, 0.27060]^t (m)$
$q_{esp} = [90.0000, 90.0000, -90.0000, 90.0000, 45.0000, 0.0000]^t (graus)$
$q_i = [100.000, 100.000, -80.0000, 100.000, 55.0000, 10.0000]^t (graus)$
$y_f = [0.10598, 0.89998, 1.28394, 0.65319, -0.27070, 0.27055]^t$ (m)
$q_f = [90.0043, 90.0004, -89.9974, 89.9819, 44.9949, 0.02000]^t$ (graus)
número de iterações=94, passo=0.0001

$$K_p = \begin{bmatrix} 15.10^6 & 0 & 0 & 0 & 0 & 0 \\ 0 & 15.10^6 & 0 & 0 & 0 & 0 \\ 0 & 0 & 15.10^6 & 0 & 0 & 0 \\ 0 & 0 & 0 & 15.10^6 & 0 & 0 \\ 0 & 0 & 0 & 0 & 15.10^6 & 0 \\ 0 & 0 & 0 & 0 & 0 & 15.10^6 \end{bmatrix}$$

$$K_d = \begin{bmatrix} 8000 & 0 & 0 & 0 & 0 & 0 \\ 0 & 8000 & 0 & 0 & 0 & 0 \\ 0 & 0 & 8000 & 0 & 0 & 0 \\ 0 & 0 & 0 & 8000 & 0 & 0 \\ 0 & 0 & 0 & 0 & 8000 & 0 \\ 0 & 0 & 0 & 0 & 0 & 8000 \end{bmatrix}$$

critério de parada: $\delta_p < 1.10^{-4} m$ e $\delta_o < 0.1\ grau$

b) Vetor q Inicial Distante de q Esperado

Caso b.1: $(q_i = q\ esperado + 30 graus)$

$y_d = [0.10607, 0.90000, 1.28393, 0.65328, -0.27060, 0.27060]^t (m)$
$q_{esp} = [90.0000, 90.0000, -90.0000, 90.0000, 45.0000, 0.00000]^t (graus)$
$q_i = [120.000, 120.000, -60.0000, 120.000, 75.0000, 30.0000]^t (graus)$
$y_f = [0.10602, 0.89996, 1.28395, 0.65320, -0.27063, 0.27057]^t$ (m)
$q_f = [90.0105, 89.9806, -89.9201, 89.8058, 45.0155, 0.09381]^t$ (graus)
número de iterações=179, passo=0.0001

$$K_p = \begin{bmatrix} 13.10^6 & 0 & 0 & 0 & 0 & 0 \\ 0 & 13.10^6 & 0 & 0 & 0 & 0 \\ 0 & 0 & 13.10^6 & 0 & 0 & 0 \\ 0 & 0 & 0 & 13.10^6 & 0 & 0 \\ 0 & 0 & 0 & 0 & 13.10^6 & 0 \\ 0 & 0 & 0 & 0 & 0 & 13.10^6 \end{bmatrix}$$

$$K_d = \begin{bmatrix} 8100 & 0 & 0 & 0 & 0 & 0 \\ 0 & 8100 & 0 & 0 & 0 & 0 \\ 0 & 0 & 8100 & 0 & 0 & 0 \\ 0 & 0 & 0 & 8100 & 0 & 0 \\ 0 & 0 & 0 & 0 & 8100 & 0 \\ 0 & 0 & 0 & 0 & 0 & 8100 \end{bmatrix}$$

critério de parada: $\delta_p < 1.10^{-4} m$ e $\delta_o < 0.1\ grau$

Caso b.2: ($q_i = q$ *esperado* $+ 50 graus$)

$y_d = [0.10607, 0.90000, 1.28393, 0.65328, -0.27060, 0.27060]^t (m)$
$q_{esp} = [90.0000, 90.0000, -90.0000, 90.0000, 45.0000, 0.00000]^t (graus)$
$q_i = [140.000, 140.000, -40.0000, 140.000, 95.0000, 50.0000]^t (graus)$
$y_f = [0.10602, 0.89995, 1.28394, 0.65317, -0.27063, 0.27057]^t$ (m)
$q_f = [89.9919, 1.64381, 90.0577, 358.390, 44.9856, -0.07121]^t$ (graus)
número de iterações=214, passo=0.0001

$$K_p = \begin{bmatrix} 14.10^6 & 0 & 0 & 0 & 0 & 0 \\ 0 & 14.10^6 & 0 & 0 & 0 & 0 \\ 0 & 0 & 14.10^6 & 0 & 0 & 0 \\ 0 & 0 & 0 & 14.10^6 & 0 & 0 \\ 0 & 0 & 0 & 0 & 14.10^6 & 0 \\ 0 & 0 & 0 & 0 & 0 & 14.10^6 \end{bmatrix}$$

$$K_d = \begin{bmatrix} 8100 & 0 & 0 & 0 & 0 & 0 \\ 0 & 8100 & 0 & 0 & 0 & 0 \\ 0 & 0 & 8100 & 0 & 0 & 0 \\ 0 & 0 & 0 & 8100 & 0 & 0 \\ 0 & 0 & 0 & 0 & 8100 & 0 \\ 0 & 0 & 0 & 0 & 0 & 8100 \end{bmatrix}$$

critério de parada: $\delta_p < 1.10^{-4} m$ e $\delta_o < 0.1\ grau$

Caso b.3: ($q_i = q$ *esperado* $+ 90 graus$)

$y_d = [0.10607, 0.90000, 1.28393, 0.65328, -0.27060, 0.27060]^t (m)$
$q_{esp} = [90.0000, 90.0000, -90.0000, 90.0000, 45.0000, 0.00000]^t (graus)$
$q_i = [180.000, 180.000, 0.00000, 180.000, 135.000, 90.0000]^t (graus)$
$y_f = [0.10607, 0.89994, 1.28389, 0.65330, -0.27060, 0.27062]^t$ (m)
$q_f = [270.000, 135.573, 26.4304, 287.992, 135.001, 179.997]^t$ (graus)
número de iterações=417, passo=0.0001

$$K_p = \begin{bmatrix} 1.10^7 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1.10^7 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1.10^7 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1.10^7 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1.10^7 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1.10^7 \end{bmatrix}$$

$$K_d = \begin{bmatrix} 8100 & 0 & 0 & 0 & 0 & 0 \\ 0 & 8100 & 0 & 0 & 0 & 0 \\ 0 & 0 & 8100 & 0 & 0 & 0 \\ 0 & 0 & 0 & 8100 & 0 & 0 \\ 0 & 0 & 0 & 0 & 8100 & 0 \\ 0 & 0 & 0 & 0 & 0 & 8100 \end{bmatrix}$$

critério de parada: $\delta_p < 1.10^{-4} m$ e $\delta_o < 0.1\ grau$

c) Critérios de Parada mais Rigorosos

Caso c.1: ($q_i = q\ esperado + 10 graus$ - $\delta_p < 1.10^{-5} m$ e $\delta_o < 0.01\ grau$)

$y_d = [0.106066, 0.900000, 1.283933, 0.653281, -0.270598, 0.270598]^t (m)$
$q_{esp} = [90.00000, 90.00000, -90.00000, 90.00000, 45.00000, 0.000000]^t (graus)$
$q_i = [100.0000, 100.0000, -80.00000, 100.0000, 55.00000, 10.00000]^t (graus)$
$y_f = [0.106065, 0.900001, 1.283936, 0.653276, -0.270597, 0.270595]^t$ (m)
$q_f = [90.00062, 89.99813, -89.99343, 89.98524, 45.00158, 0.006627]^t$ (graus)
número de iterações=152, passo=0.0001

$$K_p = \begin{bmatrix} 15.10^6 & 0 & 0 & 0 & 0 & 0 \\ 0 & 15.10^6 & 0 & 0 & 0 & 0 \\ 0 & 0 & 15.10^6 & 0 & 0 & 0 \\ 0 & 0 & 0 & 15.10^6 & 0 & 0 \\ 0 & 0 & 0 & 0 & 15.10^6 & 0 \\ 0 & 0 & 0 & 0 & 0 & 15.10^6 \end{bmatrix}$$

$$K_d = \begin{bmatrix} 8000 & 0 & 0 & 0 & 0 & 0 \\ 0 & 8000 & 0 & 0 & 0 & 0 \\ 0 & 0 & 8000 & 0 & 0 & 0 \\ 0 & 0 & 0 & 8000 & 0 & 0 \\ 0 & 0 & 0 & 0 & 8000 & 0 \\ 0 & 0 & 0 & 0 & 0 & 8000 \end{bmatrix}$$

critério de parada: $\delta_p < 1.10^{-5} m$ e $\delta_o < 0.01\ grau$

Caso c.2: ($q_i = q\ esperado + 30 graus$ - $\delta_p < 1.10^{-5} m$ e $\delta_o < 0.01\ grau$)

$y_d = [0.106066, 0.900000, 1.283933, 0.653281, -0.270698, 0.270698]^t (m)$
$q_{esp} = [90.00000, 90.00000, -90.00000, 90.00000, 45.00000, 0.000000]^t (graus)$
$q_i = [120.0000, 120.0000, -60.00000, 120.0000, 75.00000, 30.00000]^t (graus)$
$y_f = [0.106063, 0.900000, 1.283935, 0.653274, -0.270599, 0.270594]^t$ (m)
$q_f = [90.00110, 89.99723, -89.99000, 89.97700, 45.00215, 0.010774]^t$ (graus)
número de iterações=419, passo=0.0001

$$K_p = \begin{bmatrix} 13.10^6 & 0 & 0 & 0 & 0 & 0 \\ 0 & 13.10^6 & 0 & 0 & 0 & 0 \\ 0 & 0 & 13.10^6 & 0 & 0 & 0 \\ 0 & 0 & 0 & 13.10^6 & 0 & 0 \\ 0 & 0 & 0 & 0 & 13.10^6 & 0 \\ 0 & 0 & 0 & 0 & 0 & 13.10^6 \end{bmatrix}$$

$$K_d = \begin{bmatrix} 8100 & 0 & 0 & 0 & 0 & 0 \\ 0 & 8100 & 0 & 0 & 0 & 0 \\ 0 & 0 & 8100 & 0 & 0 & 0 \\ 0 & 0 & 0 & 8100 & 0 & 0 \\ 0 & 0 & 0 & 0 & 8100 & 0 \\ 0 & 0 & 0 & 0 & 0 & 8100 \end{bmatrix}$$

critério de parada: $\delta_p < 1.10^{-5} m$ e $\delta_o < 0.01\ grau$

Caso c.3: ($q_i = q$ *esperado* $+ 30 graus$ - $\delta_p < 1.10^{-6} m$ e $\delta_o < 0.001\ grau$)

$y_d = [0.106066, 0.900000, 1.283934, 0.653282, -0.270698, 0.270698]^t (m)$
$q_{esp} = [90.00000, 90.00000, -90.00000, 90.00000, 45.00000, 0.000000]^t (graus)$
$q_i = [120.0000, 120.0000, -60.00000, 120.0000, 75.00000, 30.00000]^t (graus)$
$y_f = [0.106066, 0.900000, 1.283933, 0.653281, -0.270598, 0.270598]^t$ (m)
$q_f = [90.00034, 89.99911, -89.99688, 89.99277, 45.00069, 0.003385]^t$ (graus)
número de iterações=539, passo=0.0001

$$K_p = \begin{bmatrix} 13.10^6 & 0 & 0 & 0 & 0 & 0 \\ 0 & 13.10^6 & 0 & 0 & 0 & 0 \\ 0 & 0 & 13.10^6 & 0 & 0 & 0 \\ 0 & 0 & 0 & 13.10^6 & 0 & 0 \\ 0 & 0 & 0 & 0 & 13.10^6 & 0 \\ 0 & 0 & 0 & 0 & 0 & 13.10^6 \end{bmatrix}$$

$$K_d = \begin{bmatrix} 8100 & 0 & 0 & 0 & 0 & 0 \\ 0 & 8100 & 0 & 0 & 0 & 0 \\ 0 & 0 & 8100 & 0 & 0 & 0 \\ 0 & 0 & 0 & 8100 & 0 & 0 \\ 0 & 0 & 0 & 0 & 8100 & 0 \\ 0 & 0 & 0 & 0 & 0 & 8100 \end{bmatrix}$$

critério de parada: $\delta_p < 1.10^{-6} m$ e $\delta_o < 0.001\ grau$

d) Posição de Singularidade

Caso d.1: (y_d fora do volume de trabalho do manipulador)

$y_d = [0.106066, 1.590001, 0.593934, 0.653281, -0.270598, 0.270598]^t (m)$
$q_{esp} = [90.00000, 0.000000, 0.000000, 90.00000, 45.00000, 0.000000]^t (graus)$
$q_i = [100.0000, 10.00000, 10.00000, 100.0000, 55.00000, 10.00000]^t (graus)$
$y_f = [0.106067, 1.589993, 0.593934, 0.653287, -0.270597, 0.270601]^t$ (m)
$q_f = [89.99991, 0.233686, -0.476181, 90.24692, 44.99936, -0.003060]^t$ (graus)
número de iterações=485, passo=0.0001

$$K_p = \begin{bmatrix} 1.10^7 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1.10^7 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1.10^7 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1.10^7 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1.10^7 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1.10^7 \end{bmatrix}$$

$$K_d = \begin{bmatrix} 2800 & 0 & 0 & 0 & 0 & 0 \\ 0 & 2800 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2800 & 0 & 0 & 0 \\ 0 & 0 & 0 & 2800 & 0 & 0 \\ 0 & 0 & 0 & 0 & 2800 & 0 \\ 0 & 0 & 0 & 0 & 0 & 2800 \end{bmatrix}$$

critério de parada: $\delta_p < 1.10^{-5} m$ e $\delta_o < 0.01\ grau$

Caso d.2: (y_d em posição de redundância local do manipulador)

Existem infinitas combinações de q[2], q[3], q[4] e q[6] que fornecem a mesma posição e orientação do efetuador.

$y_d = [0.150000, 0.900000, 1.390000, 0.500000, -0.500000, 0.500000]^t (m)$
$q_i = [100.0000, 100.0000, -80.00000, 100.0000, 10.00000, 10.00000]^t (graus)$
$y_f = [0.149997, 0.899997, 1.389996, 0.500000, -0.500008, 0.500001]^t$ (m)
$q_f = [89.99990, 89.99836, -90.21667, 90.85519, 0.000791, -0.637478]^t$ (graus)
número de iterações=115, passo=0.0001

$$K_p = \begin{bmatrix} 14.10^6 & 0 & 0 & 0 & 0 & 0 \\ 0 & 14.10^6 & 0 & 0 & 0 & 0 \\ 0 & 0 & 14.10^6 & 0 & 0 & 0 \\ 0 & 0 & 0 & 14.10^6 & 0 & 0 \\ 0 & 0 & 0 & 0 & 14.10^6 & 0 \\ 0 & 0 & 0 & 0 & 0 & 14.10^6 \end{bmatrix}$$

$$K_d = \begin{bmatrix} 7000 & 0 & 0 & 0 & 0 & 0 \\ 0 & 7000 & 0 & 0 & 0 & 0 \\ 0 & 0 & 7000 & 0 & 0 & 0 \\ 0 & 0 & 0 & 7000 & 0 & 0 \\ 0 & 0 & 0 & 0 & 7000 & 0 \\ 0 & 0 & 0 & 0 & 0 & 7000 \end{bmatrix}$$

critério de parada: $\delta_p < 1.10^{-5} m$ e $\delta_o < 0.01\ grau$

e) Sensibilidade aos Ganhos

Para esta análise, utilizou-se fatores de multiplicação dos ganhos (p para K_p e d para K_d), ou seja, realizou-se várias simulações multiplicando-se, em cada uma, os ganhos ótimos (ganhos que fornecem o menor número de iterações) por fatores diferentes. Procedendo-se desta forma, resultou, em cada caso, um número diferente de iterações para convergência. Neste estudo, quando variava-se p, mantinha-se $d = 1$ e quando variava-se d, mantinha-se $p = 1$. Para ilustrar a sensibilidade do método elaborou-se a tabela abaixo, obtida a partir do caso a.1 com critério de parada δ_p<$1.10^{-6}m$ e δ_o<0.001 $grau$. Nesta tabela, nc significa não convergiu em 3000 iterações e in significa que o algoritmo tornou-se instável.

Caso e.1:

Fator p	iterações	Fator d	iterações
0.01	nc	0.01	nc
0.05	1085	0.05	nc
0.1	573	0.1	1829
0.5	251	0.5	449
1.0	187	1.0	187
1.5	in	1.5	in

4.2.4 COMPORTAMENTO NAS SIMULAÇÕES

Chegou-se às seguintes conclusões sobre o comportamento do método nas simulações:

- Como ocorria com os métodos do capítulo anterior, nas simulações o número de iterações necessárias para convergência foi menor para os casos em que o vetor de coordenadas de junta de partida estava próximo do vetor de coordenadas de junta esperado. Quando utilizou-se critério de parada δ_p<$1.10^{-4}m$ e δ_o<0.1 $grau$, sempre foi possível regular ganhos que fizessem o algoritmo convergir em menos de 200 iterações para o vetor q de partida próximo de q_{esp}, e 500 iterações para o vetor q de partida distante de q_{esp} (quando a posição de chegada era não singular).

  Quando utilizou-se critério de parada δ_p<$1.10^{-6}m$ e δ_o<0.001 $grau$, sempre foi possível regular ganhos que fizessem o algoritmo convergir em menos de 400 iterações para o vetor q de partida próximo de q_{esp}, e 700 iterações para o vetor q de partida distante de q_{esp} (quando a posição de chegada era não singular).

- Para q_i distante de q_{esp}, não foi possível prever para qual das soluções o algoritmo converge (casos b.2 e b.3).

- Nas simulações, percebeu-se que, em posições de singularidade próximas ao contorno do volume de trabalho, o algoritmo demandava um número de iterações maior para convergir. Quando utilizou-se critério de parada δ_p<$1.10^{-5}m$ e δ_o<0.01 $grau$, sempre foi possível regular ganhos que fizessem o algoritmo convergir em menos de 1000 iterações. Quando utilizou-se critério de parada mais rigoroso (δ_p<1.10^{-6} e δ_o<0.001 $grau$), em nenhuma simulação o algoritmo convergiu em menos de 3000 iterações.

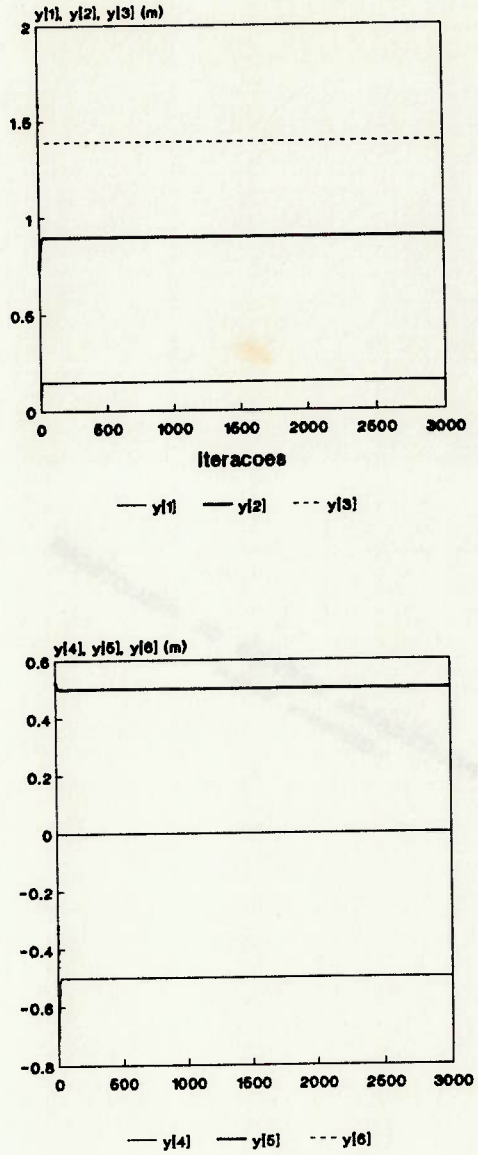


Figura 4.5: Redundância local - Vetor de posição e orientação x tempo

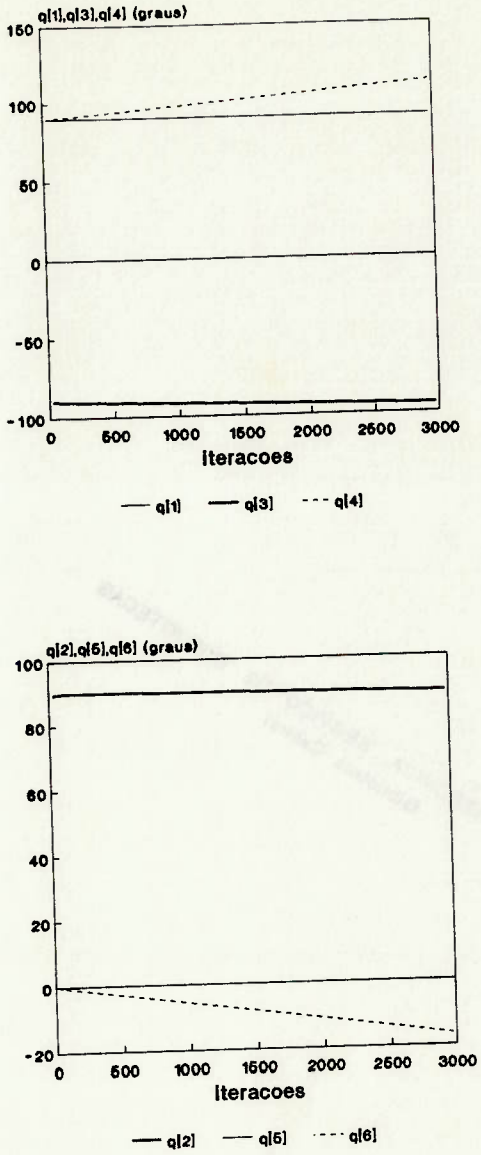


Figura 4.6: Redundância local - Coordenadas generalizadas de junta x tempo

Em posições de singularidade em que o manipulador é localmente redundante (infinitas soluções), as simulações mostraram que $\boldsymbol{y}$ converge para $\boldsymbol{y}$ de referência, mas $\boldsymbol{q}$ se mantém oscilando entre as infinitas soluções existentes. O algoritmo pára quando atinge o critério de parada desejado, em uma das infinitas soluções. As Figs. 4.5 e 4.6, correspondentes ao caso d.2 permitindo o algoritmo evoluir até 3000 iterações, ilustram o comportamento nas simulações. Neste caso, existem infinitos valores de $\boldsymbol{q}[2]$, $\boldsymbol{q}[3]$, $\boldsymbol{q}[4]$ e $\boldsymbol{q}[6]$ que correspondem a mesma posição e orientação do efetuador.

- Observou-se que o método é muito sensível aos ganhos $\boldsymbol{K_p}$ e $\boldsymbol{K_d}$. Esta característica dificulta bastante a sua utilização.

4.3 MÉTODO BASEADO EM "SLIDING MODE CONTROL"

O método em questão recebe este nome, porque as equações que o descrevem surgiram a partir da mesma função de *Lyapunov* que utiliza-se em *"sliding mode control"*, ou seja:

$$V = 1/2.\boldsymbol{s}^t.\boldsymbol{s}, \tag{4.10}$$

onde

$$\boldsymbol{s} = \dot{\tilde{\boldsymbol{y}}} + \beta.\tilde{\boldsymbol{y}} \tag{4.11}$$

e $\dot{\tilde{\boldsymbol{y}}}$ e $\tilde{\boldsymbol{y}}$ são dados por (4.1). A idéia consiste em encontrar um algoritmo que, ao ser substituído em (4.10) juntamente com a equação da cinemática diferencial direta (2.36), faça $\dot{V}$ ser sempre menor ou igual a zero e se anular somente quando $V = 0$. Neste caso, como $V \geq 0$ e $\dot{V} \leq 0$, V, e por conseqüência $\boldsymbol{s}$ (4.10), tendem a um valor constante. Uma vez que V somente pára de decrescer quando $\dot{V}$ se anula e já que isto ocorre quando $V = 0$, este valor constante é zero. Desta forma, o algoritmo encontrado estaria fazendo com que a trajetória convergisse para $\boldsymbol{s}$=0.

$\boldsymbol{s}$=0 é uma equação diferencial que tem a seguinte solução:

$$\tilde{\boldsymbol{y}} = \tilde{\boldsymbol{y}}_0.e^{-\beta.(t-t_0)}, \tag{4.12}$$

onde $\tilde{\boldsymbol{y}}_0$ é o erro de posição no instante $t = t_0$.

Observando-se esta equação, é fácil notar que, a medida que o tempo cresce, $\tilde{\boldsymbol{y}}$, e conseqüentemente $\dot{\tilde{\boldsymbol{y}}}$, decrescem, de forma que estes erros tendem a se anular. Desta forma, procurar um algoritmo que faça $\tilde{\boldsymbol{y}}$ e $\dot{\tilde{\boldsymbol{y}}}$ tenderem a zero, é equivalente a procurar um algoritmo que faça V, e consequentemente $\boldsymbol{s}$, tender a zero.

4.3.1 DEMONSTRAÇÃO DA ESTABILIDADE

Diferenciando-se (4.10),

$$\dot{V} = \boldsymbol{s}^t.\dot{\boldsymbol{s}} = \boldsymbol{s}^t.(\beta.\dot{\tilde{\boldsymbol{y}}} + \ddot{\boldsymbol{y}}_d - \dot{\boldsymbol{J}}.\dot{\boldsymbol{q}}) - \boldsymbol{s}^t.\boldsymbol{J}.\ddot{\boldsymbol{q}} \tag{4.13}$$

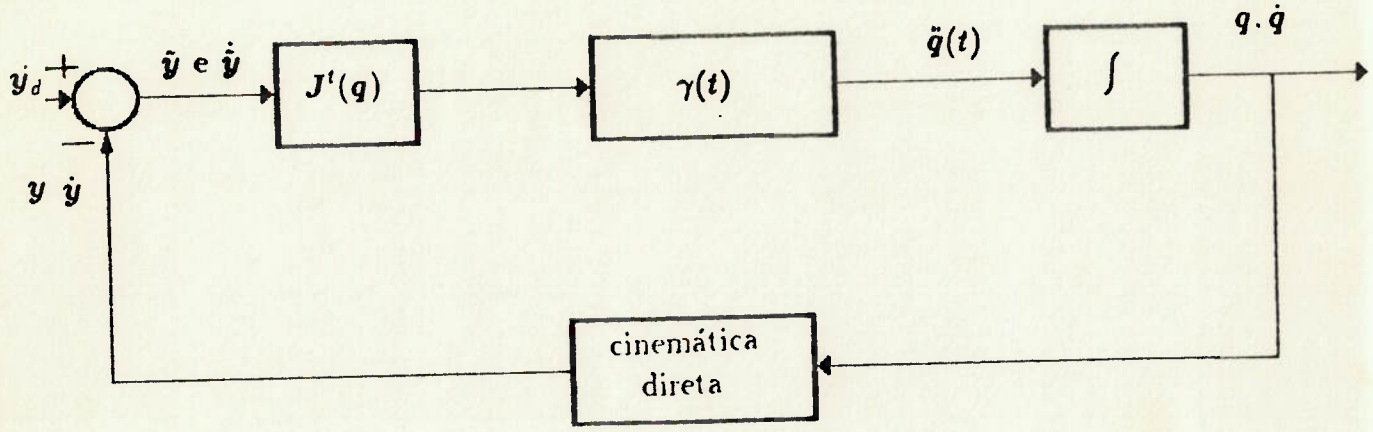


Figura 4.7: Diagrama de blocos do algoritmo

e escolhendo-se o seguinte algoritmo:

$$\ddot{q}(t) = \gamma . J^t . s \tag{4.14}$$

onde

$$\gamma(t) = (w + \alpha . V)/(s^t . J . J^t . s) \tag{4.15}$$
$$w(t) = s^t . (\beta . \dot{\tilde{y}} + \ddot{y}_d - \dot{J} . \dot{q}) \tag{4.16}$$
$$V = 1/2 . s^t . s \tag{4.17}$$
$$s = \dot{\tilde{y}} + \beta . \tilde{y} \tag{4.18}$$

e α e β são números arbitrários (não precisam ser necessariamente constantes) e positivos, conclui-se, substituindo-se (4.14) em (4.13) e admitindo-se que o denominador de γ é diferente de zero, que

$$\dot{V} = -\alpha . V \tag{4.19}$$

o que, conforme discutido no ítem anterior, garante a convergência do algoritmo, uma vez que, como $V \geq 0$ e α é positivo, $\dot{V} \leq 0$.

Para propósitos práticos, se em algum instante durante a execução do algoritmo, o termo $s^t . J . J^t . s$ (denominador de γ) tornar-se muito pequeno (menor que o número real positivo η), de forma a interferir na estabilidade do algoritmo, uma dentre as quatro medidas abaixo podem ser aplicadas [NOV90]:

- Trocar o denominador de γ por η;

- Trocar o denominador de γ por $s^t.J.J^t.s + \eta$;
- Não avaliar um novo γ, ou seja, continuar utilizando o termo γ que possuia denominador maior que η;
- Utilizar $y_d + n$ (onde n é um número real adequado), ao invés de y_d, para o cálculo de s. n deve ser tal que faça o denominador de γ ficar maior do que η.

Estas providências não garantem que V tenda a zero. Entretanto, utilizando-se valores de η adequados, o algoritmo consegue trilhar a trajetória com erro máximo de posição aceitável, conforme pode-se verificar nas simulações. Por simplicidade, optou-se pela primeira medida nas simulações efetuadas.

Observando-se a solução da equação diferencial (4.19) e a expressão (4.12), verifica-se que, teoricamente, quanto maior os valores de α e β, maior a taxa de convergência do algoritmo. De fato, as simulações comprovam esta característica. Porém deve-se ressaltar que, no caso discreto, não é possível a escolha de valores arbitrariamente elevados de α e β. Quando estes são escolhidos muito altos, ocorre instabilidade numérica no algoritmo.

4.3.2 MÉTODO DE PRIMEIRA ORDEM

Em ([BAL84]), encontra-se um algoritmo muito parecido com o método em questão, porém trata-se de um algoritmo de primeira ordem. As equações que o regem são:

$$\dot{q}(t) = \gamma.J^t.s \tag{4.20}$$

onde

$$\gamma(t) = \alpha + (s^t.\dot{y}_d)/(s^t.J.J^t.s) \tag{4.21}$$

$$s = \tilde{y} = y_d - y \tag{4.22}$$

e α é um número arbitrário (não precisa ser necessariamente constante) e positivo.

A demonstração da estabilidade deste algoritmo é feita através da função de *Lyapunov* $V = 0.5.s^t.s$ [BAL84].

As mesmas considerações feitas sobre o denominador de $\tilde{q}$ em (4.14), podem ser feitas sobre o denominador de $\dot{q}$ em (4.20).

4.3.3 FLUXOGRAMA

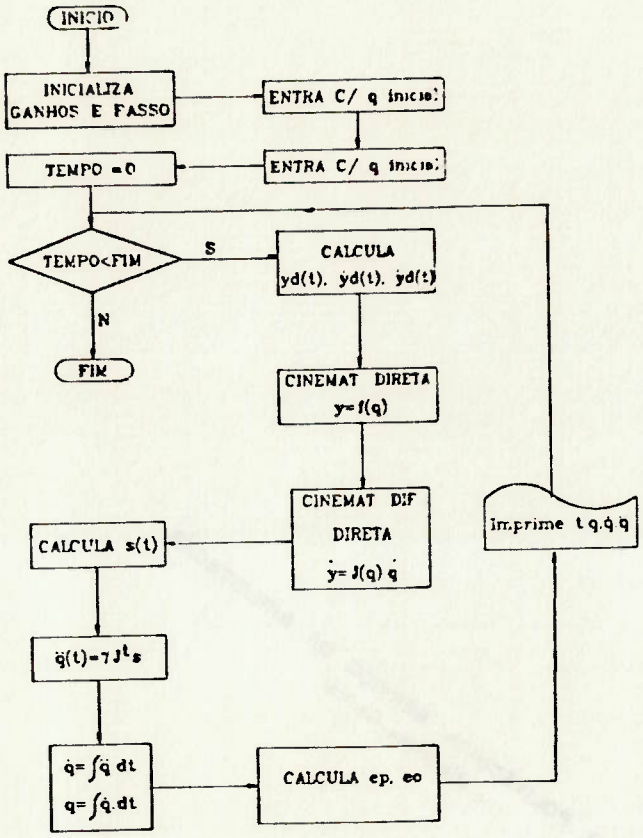


Figura 4.8: Fluxograma do algoritmo baseado em "Sliding Mode Control"

4.3.4 RESULTADOS DE SIMULAÇÕES

Esta seção ilustra o comportamento do método nas seguintes condições:

Posição de partida diferente da posição da trajetória no instante zero; diferentes velocidades do efetuador; diferentes valores de passo; trajetória passando por posições singulares; diferentes ganhos e γ constante.

Para isto realizou-se estudos paramétricos com a trajetória especificada em (3.8), utilizada no capítulo anterior.

Em um microcomputador PC 486 de 25 MHz, o tempo de uma iteração do algoritmo é de 0.0031 s, o que corresponde a uma freqüência de 327 Hz.

O comportamento característico do método durante a trajetória pode ser observado nas Figs. 4.9 e 4.10 (correspondente ao caso b com r = 0.10).

a)Posição de Partida Diferente de y_d Inicial

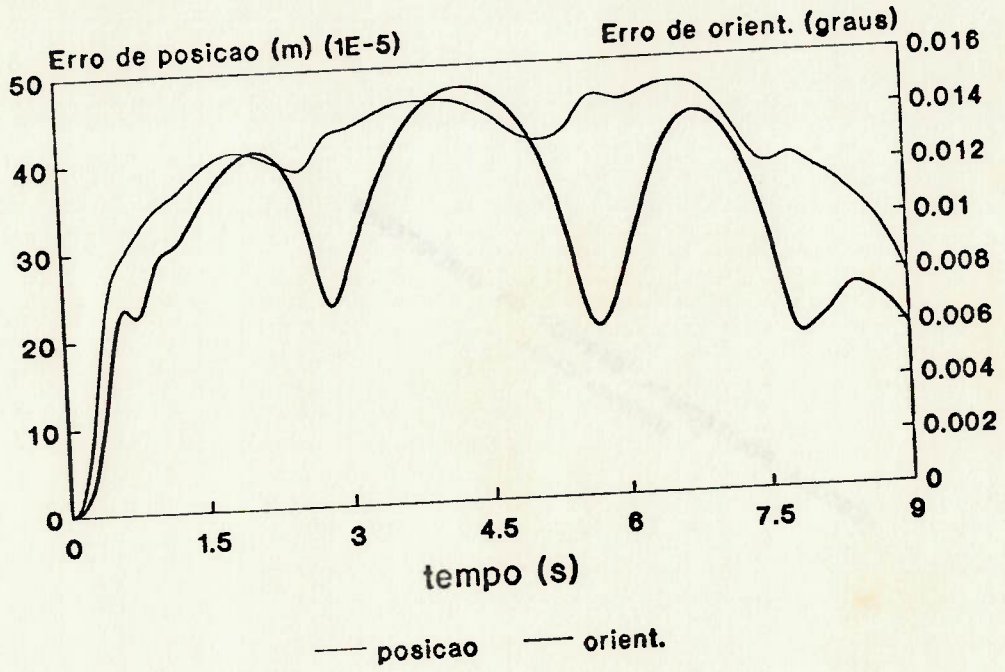


Figura 4.9: δ_p e δ_o x tempo

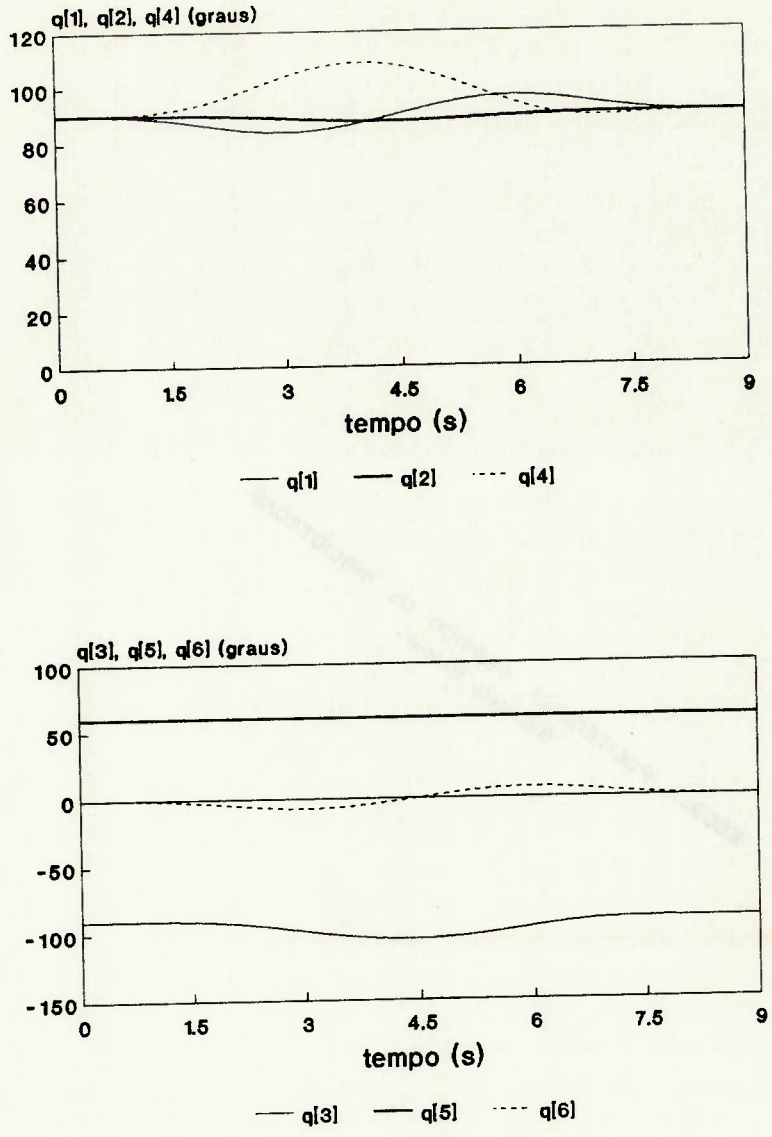


Figura 4.10: Coordenadas generalizadas de junta x tempo

Na tabela abaixo, mostra-se o tempo necessário para que δ_p fique menor do que 1.10^{-4} e δ_o fique menor do que 0.1 *grau*, em função da norma de δ_p e δ_o iniciais. Para este estudo, inicialmente (caso a.1) a coordenada da junta 1 de partida foi tomada como sendo a coordenada da junta 1 correspondente à posição e orientação do efetuador no instante zero da trajetória mais um valor x. As demais coordenadas de junta de partida foram tomadas iguais às coordenadas de junta correspondentes à posição e orientação do efetuador no instante zero. utilizou-se $passo = 0.005$, $\beta = \alpha = 100$ e $eta = 5.10^{-5}$. Em seguida (caso a.2), fez-se o mesmo estudo acrescendo-se x no valor da coordenada da junta 2, ao invés da junta 1.

Caso a.1:

x	δ_p	$\delta_o(graus)$	tempo (s)	iterações
π	1.80624	180.000	1.140	228
$\pi/2$	1.27720	90.0000	0.635	127
$\pi/4$	0.69122	45.0000	0.310	62
$\pi/10$	0.28256	18.0000	0.300	60
$\pi/18$	0.15742	10.0000	0.225	45
$\pi/180$	0.01576	1.00000	0.075	15
$\pi/1800$	0.00158	0.10000	0.050	10

Caso a.2:

x	δ_p	$\delta_o(graus)$	tempo (s)	iterações
π	2.12010	180.000	0.825	165
$\pi/2$	1.49914	90.0000	0.730	146
$\pi/4$	0.81133	45.0000	0.430	86
$\pi/10$	0.33166	18.0000	0.400	80
$\pi/18$	0.18478	10.0000	0.405	81
$\pi/180$	0.01850	1.00000	0.060	12
$\pi/1800$	0.00185	0.10000	0.055	11

b)Diferentes Velocidades do Efetuador

Para esta análise, manteve-se os valores do *passo*, α, β e η fixos em, respectivamente, 0.005, 100, 100 e 5.10^{-5}, e modificou-se o raio da trajetória para um valor r (este valor r é tal que a trajetória não ultrapassa o volume de trabalho do manipulador). Com esta alteração do raio, provocou-se modificação na velocidade máxima do efetuador. Tabelou-se, então, o raio r, a máxima velocidade correspondente ao raio r, o erro de posição máximo e o erro de orientação máximo.

Velocidades maiores, com erros menores, podem ser conseguidas diminuindo-se o *passo*. Por exemplo, com $passo = 0.001$ e mantendo-se os valores de α, β e η do caso b, chegou-se a velocidade de 0.76 m/s com $\delta_{pmax} = 0.073879.(10^{-3}m)$ e $\delta_{omax} = 0.02424grau$.

Caso b:

r	$\| v_{max} \| \ m/s$	δ_p max. $(10^{-3}m)$	δ_o max. $(graus)$
0.05	0.05236	0.03919	0.01218
0.06	0.06283	0.04109	0.01291
0.07	0.07329	0.04288	0.01356
0.08	0.08378	0.04448	0.01415
0.09	0.09425	0.04590	0.01470
0.10	0.10472	0.04716	0.01521
0.11	0.11519	0.04829	0.01569
0.12	0.12566	0.04933	0.01615
0.15	0.15708	0.05192	0.01740
0.20	0.17802	0.08818	0.02431

c)Diferentes Valores de Passo

Com $\alpha = \beta = 100$, variou-se o *passo*, e η e verificou-se se a trajetória foi completada com sucesso (s), se δ_p ultrapassou 1.10^{-4} ou δ_o ultrapassou 0.1 (u) sem ocorrer instabilidade, se ocorreu problema de instabilidade na trajetória mas a mesma foi completada com erros menores que os especificados acima (i), ou se a trajetória não pode ser completada por problema de instabilidade com erros menores do que os especificados (n).

Caso c:

passo	$\eta.10^{-5}$						
s	100	10.0	5.00	1.50	1.00	0.10	0.01
0.0001	u	s	s	s	s	s	s
0.0005	u	s	s	s	s	s	s
0.0008	u	s	s	s	s	s	s
0.001	u	s	s	s	s	s	i
0.003	u	s	s	s	s	s	i
0.005	u	s	s	s	s	i	i
0.007	u	u	i	n	n	n	n
0.01	n	n	n	n	n	n	n
0.02	n	n	n	n	n	n	n

d)Diferentes Ganhos

Com $passo = 0.005$ e $\eta = 5.10^{-5}$, variou-se α e β e verificou-se as mesmas questões analizadas no ítem anterior.

Caso d:

α	β							
	0.10	1.00	10.0	25.0	50.0	75.0	100	250
0.10	n	n	n	n	n	n	n	i
1.00	n	n	n	n	n	n	n	s
10.0	u	u	u	u	u	u	s	s
25.0	u	u	u	u	u	s	s	s
50.0	u	u	u	u	u	s	s	s
75.0	u	u	u	u	u	s	s	s
100	u	u	u	u	s	s	s	s
250	u	u	u	u	s	s	s	n
250	u	u	u	u	i	i	n	n

e)γ Constante

Em [NOV90], propõe-se utilizar o método com γ constante. Neste caso o algoritmo rodou com freqüência de 504 Hz (54% mais veloz que o algoritmo original).

Abaixo apresenta-se uma tabela que ilustra os erros de posição e de orientação em função de valores de γ (constante). As simulações foram efetuadas com *passo*, α, β e η fixos em, respectivamente, 0.005, 100, 100 e 5.10^{-5}, e os erros podem ser comparados com os obtidos no caso b com raio 0.1 m (γ calculado pela expressão (4.14)). Nesta tabela, (n) significa que ocorreu problema de instabilidade na trajetória.

Caso e:

γ	δ_p max. $(10^{-3}m)$	δ_o max. $(graus)$
0.10	75.4761	24.7049
1.00	16.4734	5.07862
25.0	0.24720	0.07791
50.0	0.11818	0.03826
75.0	0.07811	0.02521
100	0.05741	0.01881
150	0.03737	0.01252
200	n	n
300	n	n

f)Trajetória Passando por Posições Singulares

No caso f.1, apresenta-se a simulação da mesma trajetória do caso d.1, ítem 3.5.2. A tabela abaixo mostra a posição inicial da trajetória ($\boldsymbol{y}_i$), a posição final desejada ($\boldsymbol{y}_d$) e a posição final alcançada ($\boldsymbol{y}_f$). Nesta simulação foram utilizados *passo*, α, β e η fixos em, respectivamente, 0.001, 1000, 1000 e 5.10^{-3}

No caso f.2, o manipulador executa a trajetória (3.11). Nas Figs. 4.11 e 4.12 mostra-se os gráficos de δ_p e δ_o em função do tempo e das coordenadas generalizadas de junta em função do tempo. Foram utilizados os mesmos valores de *passo*, α, β e η do caso f.1.

Caso f.1:

	$y_i(m)$	$y_d(m)$	$y_f(m)$
$y[1]$	0.106066	0.106066	0.105687
$y[2]$	1.400000	1.600000	1.590854
$y[3]$	0.593934	0.593934	0.588809
$y[4]$	0.653281	0.653281	0.655099
$y[5]$	-0.270598	-0.270598	-0.269867
$y[6]$	0.270598	0.270598	0.272573
$q[1]$	90.00000	impossi.	90.02082
$q[2]$	30.15469	impossi.	-1.155692
$q[3]$	-61.30807	impossi.	1.704903
$q[4]$	121.1533	impossi.	89.99023
$q[5]$	45.00000	impossi.	44.89209
$q[6]$	0.000046	impossi.	-0.440648

Caso f.2:

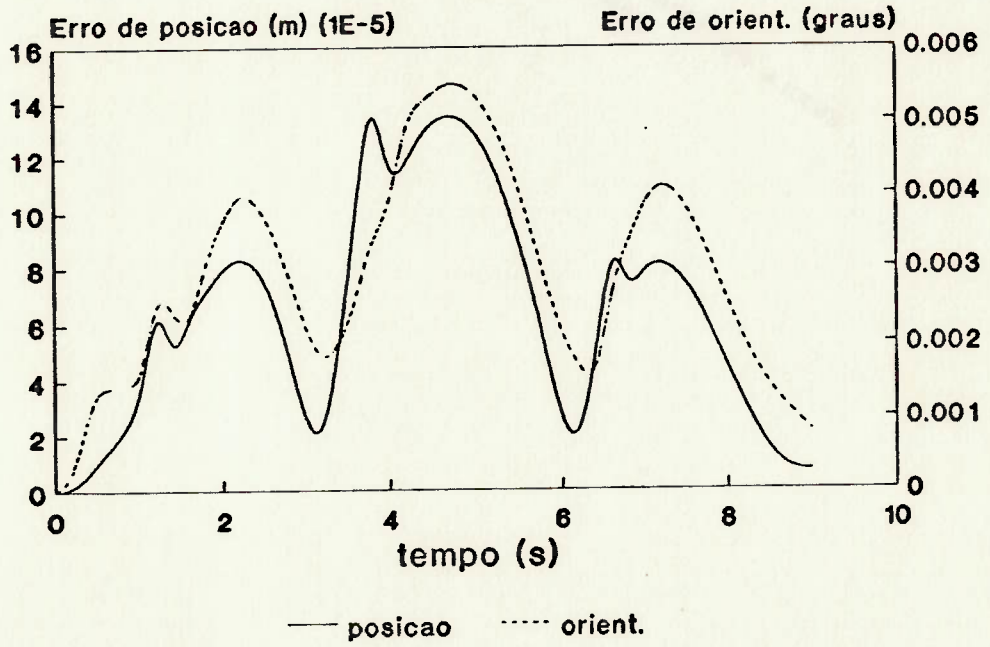


Figura 4.11: Redundância local: δ_p e δ_o x tempo

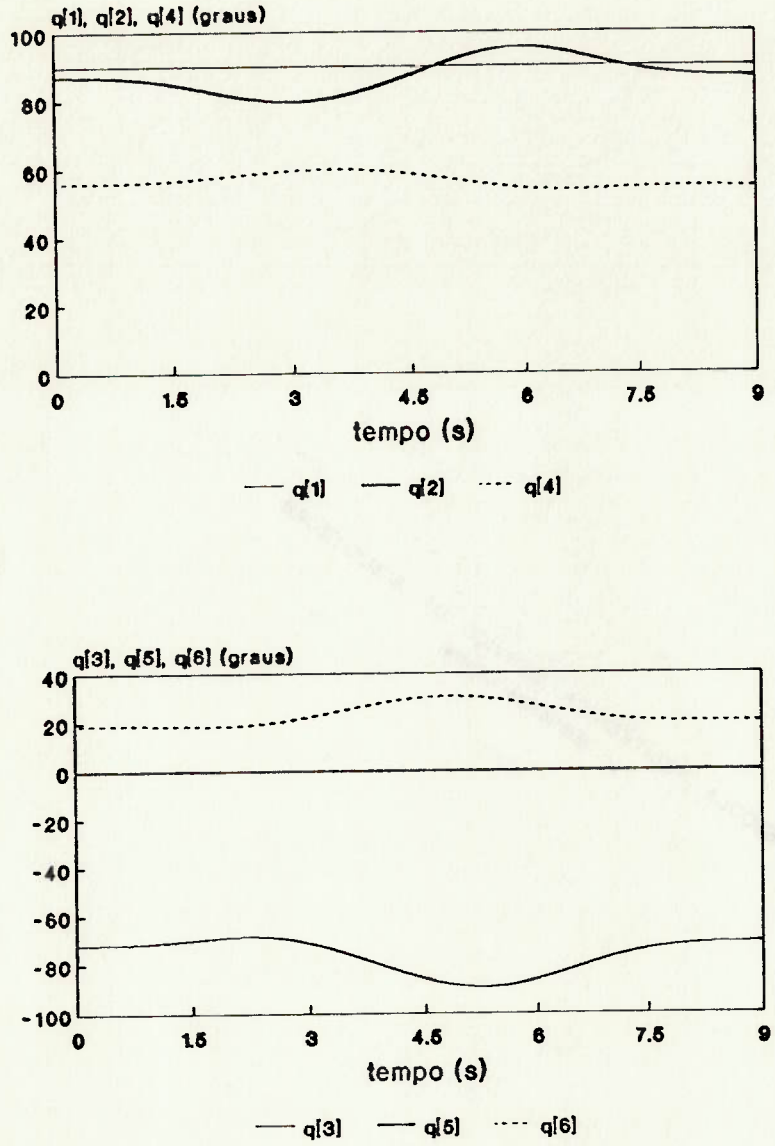


Figura 4.12: Redundância local: Coordenadas generalizadas de junta x tempo

4.3.5 COMPORTAMENTO NAS SIMULAÇÕES

Chegou-se às seguintes conclusões sobre o comportamento do método nas simulações:

- O algoritmo converge para a trajetória desejada, mesmo que a posição de partida não pertença à trajetória (casos a). Neste caso, o tempo necessário para convergência é tanto maior quanto maiores os erros de posição e de orientação iniciais.

- Os erros de posição e de orientação máximos que ocorrem durante a trajetória, dependem da velocidade com que o manipulador está se movimentando (casos b). Para velocidades elevadas (para o manipulador ASEA IRBL6, velocidades da ordem de 1 m/s, são difíceis de se atingir com este algoritmo sem ocorrer instabilidade numérica) os erros são grandes, e, para não ocorrer instabilidade, é necessário utilizar-se passos pequenos.

- Valores de η elevados, acarretam erros de posição e orientação elevados (o que pode ocasionar instabilidade numérica); valores de η muito baixos, acarretam instabilidade no algoritmo (caso c).

- Utilizando-se *passos* maiores do que 0.003 s, a trajetória estudada pode ser executada em tempo real em um microcomputador PC 486 com 25 MHz. Com *passos* maiores do que 0.01, entretanto, não foi possível completar a trajetória sem instabilidade para nenhum conjunto de ganhos.

- Assim como ocorria com os métodos para ponto fixo, a regulagem dos ganhos influencia muito o comportamento do algoritmo. No caso d, encontra-se valores dos ganhos α e β com os quais a trajetória pode ser completada com erros de posição e orientação pequenos (δ_p<$1.10^{-4}m$ e δ_o<0.1 $grau$).

- A utilização do algoritmo com γ constante (proposta em [NOV90]), aumenta a freqüência do algoritmo de 327 Hz para 504 Hz (1.54 vezes). A regulagem desta constante, apesar de bastante trabalhosa, pode gerar erros de posição e orientação menores que os gerados para γ variável (caso e).

- Em posições singulares em que o manipulador é redundante, o algoritmo comportou-se sem problemas (encontrava uma dentre as infinitas soluçoes para a cinemática inversa).

  Em posições singulares próximas ao contorno do volume de trabalho, o algoritmo não se mostrou robusto. Em todas as simulações semelhantes ao caso f.1, ocorreram problemas de instabilidade numérica quando se aproximava da singularidade. Mesmo assim, em geral, as trajetórias puderam ser completadas sem ocorrência de "overflow", porém com erros de posição e orientação elevados (δ_p<$1.10^{-3}m$ e δ_o<1 $grau$).

4.4 MÉTODOS DE NEWTON-RAPHSON X MÉTODOS BASEADOS NA TRANSPOSTA DO JACOBIANO

Para ponto-fixo, as simulações efetuadas mostraram que:

- O método de *Newton-Raphson* [5] converge com velocidade bem maior que o método baseado em "Impedance Control". Isto ocorre porque, apesar de cada iteração do primeiro ser mais lenta que as do segundo (0.0041 s e 0.0020 s), o método de *Newton-Raphson* demanda muito menos iterações para convergir.
- Em ambos os métodos notou-se a influência da distância entre q de partida e q esperado. Quanto maior a distância, maior o número de iterações necessárias para convergência.
- Quando a raiz é posição de redundância local, ambos os métodos convergem como nos casos em que não ocorre singularidade.
- Quando a raiz está no contorno do volume de trabalho, o método de *Newton-Raphson* converge mais facilmente que o baseado em "Impedance Control".
- O método de *Newton-Raphson* apresenta problema de afastamento em relação à raiz quando passa por posição de singularidade em iterações intermediárias.
- De forma geral, principalmente quando a posição de partida está próxima da raiz, o método de *Newton-Raphson* é mais rápido, de regulagem mais fácil e mais robusto quando a solução está em posição singular que o algoritmo baseado em "Impedance Control".

Para trajetória, chegou-se às seguintes conclusões:

- O método de *Newton-Raphson* adaptado permite o controle dos erros de posição e orientação durante a execução da trajetória, o que não ocorre com o método baseado em "Sliding Mode Control".
- O método de *Newton-Raphson* adaptado, apesar de necessitar da computação da pseudo-inversa de J , é mais veloz que o método baseado em "Sliding Mode Control". Isto ocorre porque, tanto a pseudo-inversa de J como os seus elementos são computados apenas de n em n pontos e porque é possível a utilização do algoritmo com valores de *passo* mais altos.
- O algoritmo baseado em "Sliding Mode Control", para não tornar-se instável em altas velocidades, necessita de passos bem menores que o método de *Newton-Raphson* adaptado.

[5] Ao invés da inversa de J computada no algoritmo original calculava-se a pseudo-inversa.

- A regulagem dos parâmetros do algoritmo são bem mais simples no caso do método de *Newton-Raphson* adaptado.
- O algoritmo baseado em "Sliding Mode Control" possui a vantagem de gerar acelerações das coordenadas generalizadas de junta, enquanto que o método de *Newton-Raphson* adaptado gera, apenas, as coordenadas generalizadas de junta.
- Ambos os métodos comportaram-se bem em trajetórias passando por posições de redundância local do manipulador.
- Para trajetórias passando por posição pertencente ao volume de trabalho, o método de *Newton-Raphson* adaptado mostrou-se menos problemático.
- De forma geral, o método de *Newton-Raphson* adaptado é mais rápido, de utilização mais simples, de regulagem mais fácil, mais eficaz no controle dos erros e mais robusto para trajetórias que passam por posições singulares que o algoritmo baseado em "Sliding Mode Control".

Capítulo 5

CINEMÁTICA INVERSA DE MANIPULADORES REDUNDANTES

5.1 INTRODUÇÃO

É comum fabricantes de robôs industriais optarem pelo desenvolvimento de manipuladores com 6 graus de liberdade ($n = 6$). Isto ocorre porque grande parte destes manipuladores [ROT76] podem localizar (posicionar segundo uma orientação dada) o seu efetuador em qualquer posição de seu volume de trabalho. Há, entretanto, vários fatores que restringem a flexibilidade, aqui entendida como a capacidade de movimentação segundo qualquer direção do espaço, do robô. Assim, volume de trabalho pequeno, presença de obstáculos, posições de singularidade e limitações de coordenadas de junta, impedem que o robô possa localizar o seu efetuador em qualquer ponto de seu volume de trabalho.

Estas limitações podem ser minimizadas introduzindo-se mais graus de liberdade no manipulador. Manipuladores que possuem $posto(\boldsymbol{J}(\boldsymbol{q})) < n$ para qualquer configuração de juntas, são ditos manipuladores redundantes. Para estes, dada determinada localização do efetuador, existem infinitos vetores $\boldsymbol{q}$ de coordenadas generalizadas de junta correspondentes. Manipuladores redundantes com $posto(\boldsymbol{J}(\boldsymbol{q})) = 6$, como é o caso do sistema "robô ASEA IRBL6 + trilho", permitem a localização de seu efetuador em qualquer posição de seu volume de trabalho e, além disto, possuem infinitos vetores $\boldsymbol{q}$ correspondentes a uma localização dada do efetuador. Esta última propriedade incentivou o surgimento de métodos que solucionam a cinemática inversa de manipuladores redundantes selecionando soluções que satisfaçam certas restrições [SCI88], como presença de obstáculos e restrições nas coordenadas de junta, por exemplo, ou então maximizando o desempenho do robô segundo algum critério [LIE77,DAS88,CHE88,DUB88].

Neste capítulo, apresenta-se o algoritmo de *Sciavicco e Siciliano* [SCI88] para solução da cinemática inversa de manipuladores redundantes, e propõe-se modificá-lo para ser utilizado em conjunto com o método de *Newton-Raphson* adaptado (capítulo 3), o que aumenta a sua eficiência e simplifica a sua implementação. A este novo algoritmo dá-se o nome de método híbrido. Assim como o método de *Sciavicco e Siciliano*, o método híbrido

seleciona das infinitas soluções da cinemática inversa aquelas que evitam obstáculos ou que tenham as coordenadas generalizadas de junta dentro de determinada faixa. No capítulo 6, aplica-se este algoritmo, implementado com o jacobiano relacionando diferenciais de coordenadas de junta com diferenciais de coordenadas cartesianas (posição) e parâmetros de *Euler-Rodrigues* (orientação), no desenvolvimento do simulador cinemático para o sistema "robô ASEA IRBL6 + trilho".

5.2 MÉTODO DE SCIAVICCO E SICILIANO

Sciavicco e Siciliano [SCI88] sugeriram aproveitar o fato de que manipuladores redundantes possuem um número maior de coordenadas generalizadas de junta independentes (g) do que coordenadas de espaço independentes (m), para introduzir $g - m$ expressões no sistema de equações da cinemática direta (2.29), de forma que o novo sistema tenha o mesmo número de equações e incógnitas. Com isto, a solução deste conjunto de equações satisfaz, além das expressões da cinemática direta, as $g - m$ equações adicionais.

Estas equações adicionais são escolhidas de forma a selecionar soluções que evitem obstáculos e/ou que tenham determinada coordenada de junta dentro de dados limites. Assim, deixa-se o algoritmo original (sem as espressões adicionais) calcular uma dentre as infinitas soluções da cinemática inversa e, caso esta solução não seja interessante, ou seja, não satisfaça alguma das restrições desejadas, acrescenta-se a espressão correspondente àquela restrição.

O equacionamento proposto em [SCI88] foi desenvolvido para ser aplicado com o algoritmo (4.20), apresentado no capítulo anterior. A seguir mostra-se, separadamente, este desenvolvimento para o caso de obstáculos e coordenadas de junta e, em seguida, para o caso geral.

5.2.1 OBSTÁCULOS

Para o equacionamento deste ítem, utiliza-se a seguinte convenção (conforme ilustrado na Fig. 5.2.1):

- $\boldsymbol{c}$ = vetor posição do ponto de interesse do obstáculo (3×1);
- $\boldsymbol{p}$ = vetor posição do ponto de menor distância do manipulador ao obstáculo (3×1);
- $\boldsymbol{d} = \boldsymbol{p}\text{-}\boldsymbol{c}$ (3×1);
- d_d = distância de segurança (se $\| \boldsymbol{d} \|$ ficar menor do que d_d, existe perigo de colisão e a equação (4.20) deve ser modificada, conforme colocado adiante).

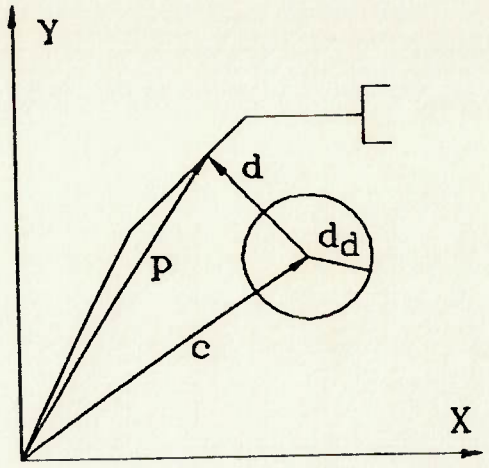


Figura 5.1: Convenção para o caso de obstáculos

Neste trabalho, os obstáculos são modelados como esferas. Desta forma, d_d é o raio da esfera.

Fazendo-se analogia com o erro $\tilde{y}$ definido e utilizado em (4.20), defini-se o erro

$$e_o = 0.5.(d_d^2 - d^t.d) \tag{5.1}$$

que, após diferenciado fornece

$$\dot{e_o} = d_d.\dot{d_d} - d^t.\dot{c} - j_d^t.\dot{q} \tag{5.2}$$

com

$$j_d^t = d^t.J_p^t \tag{5.3}$$

onde J_p é a matriz Jacobiano do ponto definido pelo vetor p. Na expressão (5.1), ambos os casos de obstáculo movendo-se ($\dot{c} \neq 0$) e distância de segurança variável ($\dot{d_d} \neq 0$) foram considerados. No que segue, assume-se que $\dot{c} = 0$ e $\dot{d_d} = 0$.

A idéia de *Sciavicco e Siciliano* consiste em, assim que o erro $\| d \|$ ficar menor do que d_d, extender o vetor $\tilde{y}$ $(m \times 1)$ com o termo e_o, e a matriz J com a linha j_d. Assim, a partir deste instante, tanto o vetor $\tilde{y}$ extendido $(\tilde{y}_e)$ como a matriz Jacobiano extendida (J_e) passam a ter $m + 1$ linhas.

Realizando-se este procedimento, o algoritmo "traz" a solução da cinemática inversa para valores que tornem a distância $\| d \|$ igual a d_d e o ligamento candidato à colisão é forçado a mover-se tangencialmente à esfera de segurança. É possível utilizar este método para $g - m$ obstáculos. Para cada um acrescenta-se uma equação adicional em (4.20).

5.2.2 COORDENADAS DE JUNTA

A idéia para limites nas coordenadas generalizadas de junta é semelhante ao caso de obstáculos. Assim, deseja-se encontrar soluções para a cinemática inversa nas quais a

coordenada de junta $q[i]$ seja menor que $q[i]_{max}$ e maior que $q[i]_{min}$. Para isto, utiliza-se uma distância de segurança d_{dq} que, se em determinado instante tornar-se menor que a distância entre $q[i]$ e $q[i]_{min}$ ou $q[i]_{max}$, faz com que o algoritmo (4.20) seja modificado. Com esta finalidade, defini-se o erro

$$\epsilon_q = d_{dq} - d_q \tag{5.4}$$

onde $d_q = q[i] - q[i]_{min}$ ou $d_q = q[i]_{max} - q[i]$, dependendo do limite envolvido. Esta expressão, após diferenciada fornece

$$\dot{e_q} = \dot{d_{dq}} - \boldsymbol{j}^t_{q[i]} \cdot \dot{\boldsymbol{q}} \tag{5.5}$$

onde

$$\boldsymbol{j}^t_{q[i]} = [000... \pm 1...0] \tag{5.6}$$

Como feito acima, adiante considera-se somente os casos em que $\dot{d_{dq}} = 0$.

Na expressão (5.6), o sinal + aplica-se para $q[i]_{min}$, o sinal - para $q[i]_{max}$, e o valor 1 aparece na posição i de $\boldsymbol{j}^t_{q[i]}$. A modificação no algoritmo consiste em extender o vetor $\tilde{\boldsymbol{y}}$ com o termo ϵ_q e a matriz $\boldsymbol{J}$ com a linha $\boldsymbol{j}^t_{q[i]}$. Quando isto ocorre, a coordenada de junta em questão é forçada a tender ao limite estipulado. Assim, como ocorria com o caso de obstáculos, é possível estabelecer limites para até $g - m$ coordenadas de junta.

5.2.3 CASO GERAL

Com base nos resultados apresentados nos dois ítens anteriores, neste ítem descreve-se o algoritmo geral, que engloba obstáculos e limites em coordenadas de junta, para um manipulador genérico com r graus de liberdade de redundância ($g - m = r$). Estes r graus redundantes são distribuídos de forma que p graus são utilizados para evitar p obstáculos e t graus para t intervalos de coordenadas de junta. No que segue mostra-se como ficam o vetor $\tilde{\boldsymbol{y}}_e$ extendido e a matriz $\boldsymbol{J}_e$ extendida, que aparecem na equação (4.20), reproduzida abaixo, após terem sido atingidos os limites de segurança.

$$\dot{\boldsymbol{q}}(t) = \gamma . \boldsymbol{J}^t . \tilde{\boldsymbol{y}} \tag{5.7}$$

Vetor $\boldsymbol{y}$ extendido ($(m + r) \times 1$):

$$\boldsymbol{y}_e = \begin{bmatrix} \boldsymbol{y} \\ \boldsymbol{y}_o \\ \boldsymbol{y}_q \end{bmatrix} \tag{5.8}$$

onde $\boldsymbol{y}_o$ é o vetor de p componentes que contém as quantidades $\boldsymbol{d}^t.\boldsymbol{d}/2$ dadas em (5.1), e $\boldsymbol{y}_q$ é o vetor de t componentes que contém as quantidades $\boldsymbol{d}_q$ dadas em (5.4).

Vetor $\boldsymbol{y}_d$ extendido ($(m + r) \times 1$):

$$\boldsymbol{y}_{de} = \begin{bmatrix} \boldsymbol{y}_d \\ \boldsymbol{y}_{do} \\ \boldsymbol{y}_{dq} \end{bmatrix} \tag{5.9}$$

onde $\boldsymbol{y}_{do}$ é o vetor de p componentes que contém as quantidades $d_d^2/2$ dadas em (5.1), e $\boldsymbol{y}_{dq}$ é o vetor de t componentes que contém as quantidades d_{dq} dadas em (5.4).

Matriz $\boldsymbol{J}$ extendida $(n \times n)$:

$$\boldsymbol{J}_e = \begin{bmatrix} \boldsymbol{J} \\ \boldsymbol{J}_o \\ \boldsymbol{J}_q \end{bmatrix} \tag{5.10}$$

onde $\boldsymbol{J}_o$ é a matriz $p \times n$ que contém as linhas $\boldsymbol{j}_d$ dadas em (5.3), e $\boldsymbol{J}_q$ é a matriz $t \times n$ que contém as linhas $\boldsymbol{j}_q$ dadas em (5.6).

5.3 MÉTODO HÍBRIDO

As vantagens dos métodos baseados no de *Newton-Raphson* sobre os baseados na transposta do Jacobiano, analisadas no capítulo 4, sugerem que se utilize a idéia de *Sciavicco e Siciliano* em conjunto com o método de *Newton-Raphson* adaptado, ao invés de utilizá-la com o algoritmo (4.20). Com isto seria possível selecionar soluções que evitem obstáculos e/ou soluções nas quais as coordenadas de junta estejam dentro de determinada faixa, utilizando-se um algoritmo bem mais eficiente que aquele aplicado por *Sciavicco e Siciliano* em [SCI88]. Para isto, neste ítem propõe-se alterações nas formulações desenvolvidas no ítem anterior, de forma a dar origem a um novo algoritmo, que chamaremos de método híbrido.

5.3.1 OBSTÁCULOS

É intuitivo que a distância do manipulador a um obstáculo fixo, que neste trabalho será modelado como uma esfera, é função da geometria do obstáculo e das coordenadas de junta do manipulador. Assim, dadas as coordenadas generalizadas de junta em determinado instante e o raio do obstáculo, pode-se calcular a distância de cada ligamento ao obstáculo e, tomando-se a menor destas distâncias, encontrar a distância desejada naquele instante.

No caso da intersecção entre a perpendicular à reta que define o ligamento e esta reta ser interna ao mesmo, a distância do obstáculo ao ligamento é dada pela distância do centro da esfera a esta intersecção. No caso da intersecção ser externa ao ligamento, a distância é dada pela distância entre o centro da esfera e o centro da junta do ligamento em questão mais próxima à esfera.

Como estratégia para introdução das $g-m$ equações adicionais que tornam o sistema $\boldsymbol{y} = f(\boldsymbol{q})$ (cinemática direta) determinado, utilizaremos a mesma sugerida por *Sciavicco e Siciliano*, ou seja, deixa-se o algoritmo de *Newton-Raphson* adaptado gerar uma das infinitas soluções da cinemática inversa e quando uma destas soluções resultar em distância menor que d_d (distância de segurança desejada), introduz-se a equação adicional correspondente aquele obstáculo.

Assim, seja o manipulador redundante dado pelo sistema de equações da cinemática direta

$$\boldsymbol{y} = f(\boldsymbol{q}) \tag{5.11}$$

onde $\boldsymbol{y}$ e $\boldsymbol{q}$ são vetores de, respectivamente, m e g componentes independentes. Suponha, ainda, por simplicidade mas sem perda de generalidade, que este manipulador possui somente um grau de liberdade de redundância ($g - m = 1$). O novo sistema (agora com mesmo número de incógnitas e equações) a ser resolvido é dado por:

$$\boldsymbol{y}_e = \begin{bmatrix} \boldsymbol{y}_d \\ d_d \end{bmatrix} = \begin{bmatrix} f(\boldsymbol{q}) \\ d(\boldsymbol{q}) \end{bmatrix} \tag{5.12}$$

e a nova matriz Jacobiano extendida ($\boldsymbol{J}_e$), é dada por:

$$\boldsymbol{J}_e = \begin{bmatrix} \boldsymbol{J} \\ \boldsymbol{j}_d^t \end{bmatrix} \tag{5.13}$$

onde o elemento i da linha $\boldsymbol{j}_d^t$ é obtido calculando-se numericamente $(\delta d/\delta q_i)$.

Desta forma, a partir do instante em que a distância do manipulador ao obstáculo torna-se menor que d_d, a matriz $\boldsymbol{J}$ é substituída por $\boldsymbol{J}_e$ e o vetor $\boldsymbol{y}$ por $\boldsymbol{y}_e$, no algoritmo de *Newton-Raphson* adaptado. Com isto, além de resolver as equações da cinemática direta, impõe-se uma distância mínima desejada, como queríamos.

Esta implementação da idéia de *Sciavicco e Siciliano* é de compreensão bem mais simples que a original (ítem 5.2.1), uma vez que, neste caso, não é necessário calcular o Jacobiano da intersecção entre a reta do ligamento e a perpendicular a mesma que passa pelo centro do obstáculo.

5.3.2 COORDENADAS DE JUNTA

Neste caso, a utilização da idéia de *Sciavicco e Siciliano* com o método de *Newton-Raphson*, é de implementação imediata. Basta introduzir no sistema a ser resolvido, a equação da igualdade entre a coordenada de junta cujo limite está sendo violado e este limite. Assim, se $10° \leq q[1] \leq 80°$ e se, em dado instante, $q[1]$ ultrapassa $80°$, a partir deste instante introduz-se a expressão $q[1] = 80°$ no sistema de equações da cinemática direta.

Esta expressão adicional, corresponde a uma linha adicional na matriz Jacobiano. Uma vez que nos casos de limites em coordenadas de junta a equação adicional introduzida na cinemática direta sempre será do tipo $q[i] = cte$, a linha adicional ($\boldsymbol{j}_q$) em $\boldsymbol{J}$ sempre será do tipo

$$\boldsymbol{j}_{q[i]}^t = [000...1...0] \tag{5.14}$$

onde 1 aparece na posição i de $\boldsymbol{j}_q^t$.

5.3.3 CASO GERAL

Da mesma forma que foi feito no ítem 5.2, apresenta-se abaixo o equacionamento do método híbrido para o caso geral, em que ambos os casos presença de obstáculos e limites nas coordenadas de junta estão presentes, e em que o manipulador possui r graus de liberdade de redundância ($g - m = r$) distribuídos de forma que p graus são utilizados para evitar p obstáculos e t graus para t intervalos de coordenadas de junta.

Assim, no que segue mostra-se como ficam o novo sistema a ser resolvido (cinemática direta mais equações adicionais) e a nova matriz $\boldsymbol{J}_e$ (extendida) que aparece no método de *Newton-Raphson* adaptado.

Sistema extendido a ser resolvido ($(m + r) \times 1$):

$$\boldsymbol{y}_e = \begin{bmatrix} \boldsymbol{y} \\ \boldsymbol{y}_o \\ \boldsymbol{y}_q \end{bmatrix} = \begin{bmatrix} \boldsymbol{y}_d \\ \boldsymbol{y}_{od} \\ \boldsymbol{y}_{qd} \end{bmatrix} \tag{5.15}$$

onde $\boldsymbol{y}$é calculado pela cinemática direta ($\boldsymbol{y}(\boldsymbol{q})$), $\boldsymbol{y}_o$ é o vetor de p componentes que contém as distâncias calculadas do manipulador aos p obstáculos (função de $\boldsymbol{q}$), $\boldsymbol{y}_q$ é o vetor de t componentes que contém os termos $q[i]$, $\boldsymbol{y}_{od}$ é o vetor de p componentes que contém as distâncias mínimas desejadas do manipulador aos p obstáculos e $\boldsymbol{y}_{qd}$ é o vetor de t componentes que contém as constantes *ctes* desejadas para as coordenadas de junta.

Matriz $\boldsymbol{J}_e$ extendida ($n \times n$):

$$\boldsymbol{J}_e = \begin{bmatrix} \boldsymbol{J} \\ \boldsymbol{J}_o \\ \boldsymbol{J}_q \end{bmatrix} \tag{5.16}$$

onde $\boldsymbol{J}_o$ é a matriz $p \times n$ que contém as linhas $\boldsymbol{j}_o^t$ dadas em (5.13), e $\boldsymbol{J}_q$ é a matriz $t \times n$ que contém as linhas $\boldsymbol{j}_q^t$ dadas em (5.14).

5.3.4 "TRAVAMENTO"

Se uma vez inseridas, as equações adicionais continuarem sempre presentes no algoritmo, é possível que, a partir de determinado instante no decorrer da trajetória, o sistema extendido passe a não ter solução apesar do sistema original (cinemática direta), possuir. Assim, pode acontecer de existir solução da cinemática inversa que evite certo obstáculo mas de não existir solução que tenha distância do obstáculo igual a distância de segurança, ou então, pode ocorrer de existir solução na qual, por exemplo, $10^o \leq q[1] \leq 80^o$, mas de não existir solução que tenha $q[1]$ igual a 80^o.

Tendo em vista estas possibilidades, percebe-se a necessidade de se retirar as equações adicionais a partir de certo instante. Na implementação deste trabalho, isto é feito quando ocorre "travamento" do sistema, ou seja, quando é atingido o número máximo de iterações permitido por ponto da trajetória.

5.3.5 ESTUDO DE CASO

Neste ítem, submeteu-se o método híbrido à seguinte trajetória, que também pode ser vista na Fig. 5.2:

$$\boldsymbol{y}_d = \begin{bmatrix} 0.106066 + 0.278982.sin(\psi/4) \\ -1.283934 + 0.023838.sin(\psi/4) \\ 2.400000 - 1.716922.sin(\psi/4) \\ 0.094363.sin(3.127113.\psi/(4\pi)) + 0.912505.cos(3.127113.\psi/(4\pi)) \\ -0.102538.sin(3.127113.\psi/(4\pi)) - 0.377972.cos(3.127113.\psi/(4\pi)) \\ -0.961706.sin(3.127113.\psi/(4\pi)) + 0.059865.cos(3.127113.\psi/(4\pi)) \end{bmatrix} \quad (5.17)$$

onde ψ é dado pelas mesmas expressões que em (3.9).

Trata-se de uma trajetória retilínea entre os pontos definidos pelos vetores $\boldsymbol{y}$=[0.106066, 0.9, 1.283933, 0.912505, −0.377972, 0.144527]^t (posição e orientação inicial) e $\boldsymbol{y}$=[0.385048, −0.816923, 1.260096, 0.100967, −0.105271, −0.960634]^t (posição e orientação final), e com velocidades inicial e final nulas.

Apresenta-se, abaixo, os resultados de três simulações: Na primeira (caso 1), utilizou-se o método de *Newton-Raphson* adaptado; na segunda (caso 2), o método híbrido com obstáculo de raio=0.1 m (d_d=0.1 m) na posição [0,-0.2,0.6]^t (m); na terceira (caso 3), o método híbrido foi utilizado com q[2]$_{min}$=55°.

Note que, no primeiro caso, ambas as restrições colocadas nos casos 2 e 3 são infringidas.

Caso 1:

O método de *Newton-Raphson* adaptado foi utilizado com erros de posição e orientação máximos permitidos de, respectivamente, 1.10^{-4} e 0.1 *grau*, com numero máximo de iterações por ponto permitido de 10 e computando-se os elementos da matriz $\boldsymbol{J}$ e sua pseudo-inversa de 15 em 15 pontos ($n = 15$).

Na fig. 5.3, mostra-se as coordenadas generalizadas de junta × tempo (s). Na fig. 5.4 mostra-se a distância ao obstáculo do caso 2 × tempo (s).

O tempo para a execução desta trajetória num microcomputador PC 486 25 Mh foi de 2.91 s.

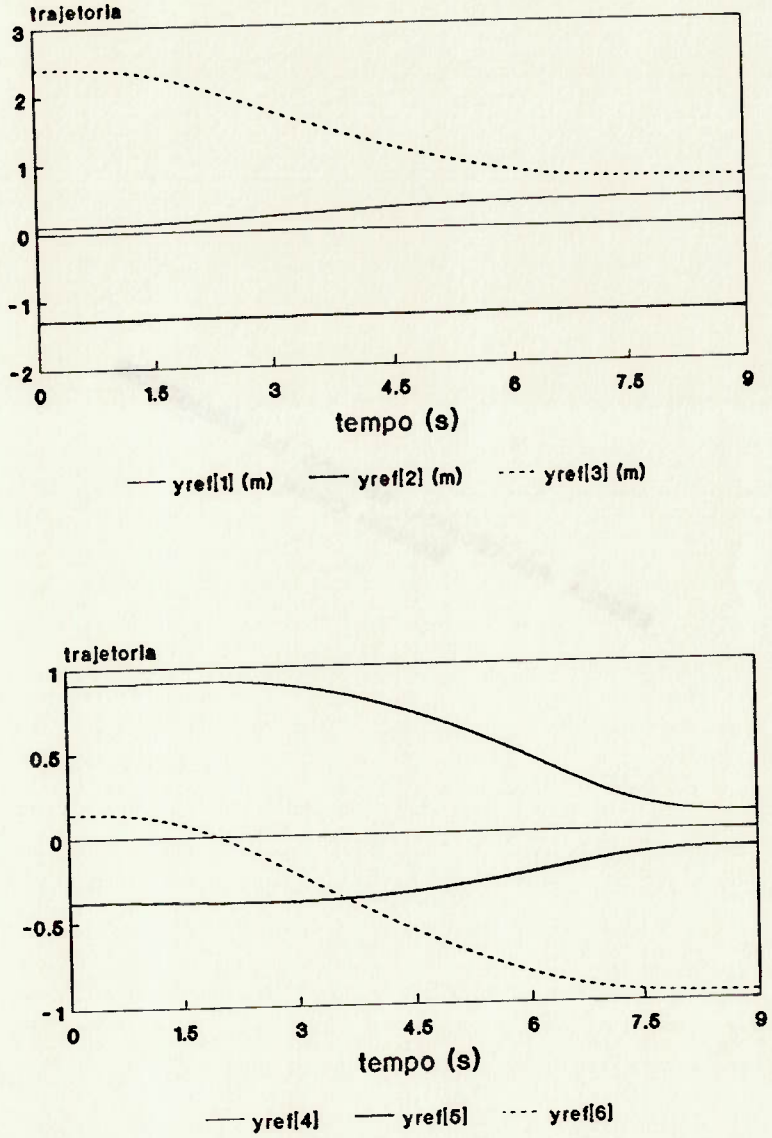


Figura 5.2: Trajetória

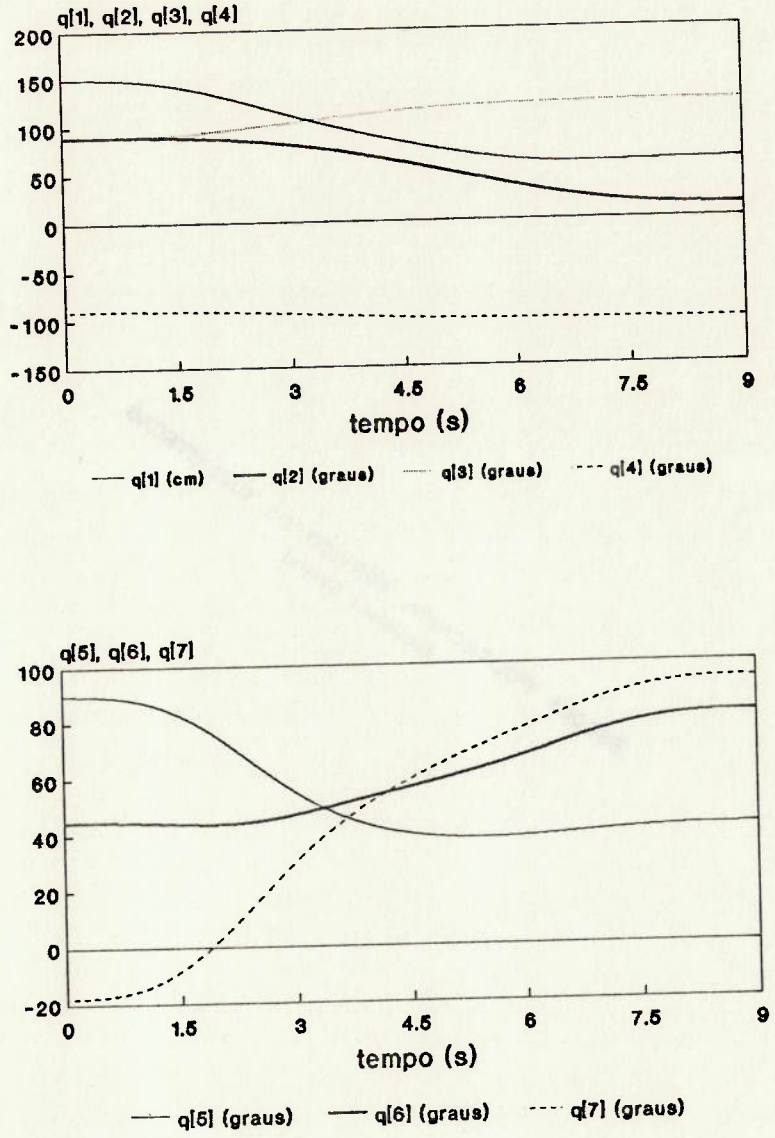


Figura 5.3: Coordenadas de junta × tempo

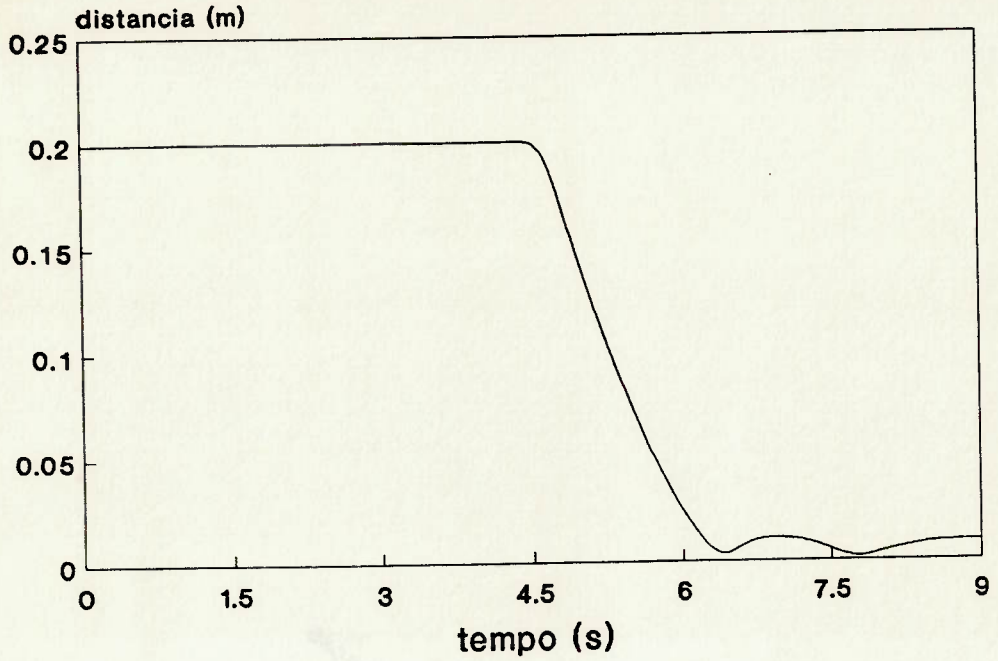


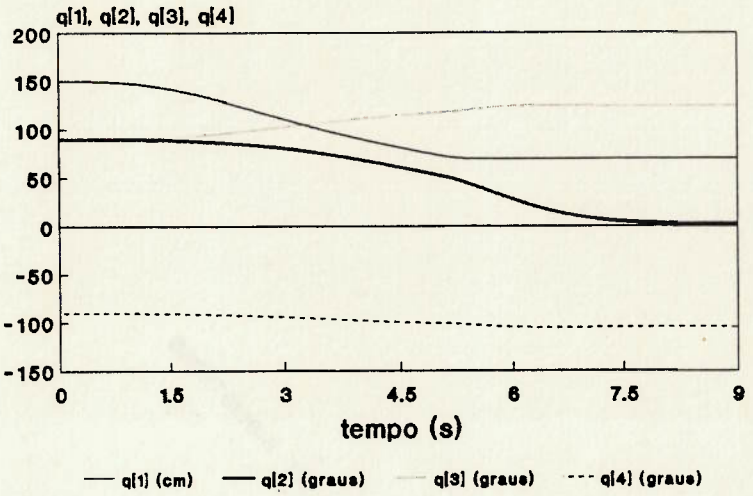
Figura 5.4: Distância ao obstáculo × tempo

Caso 2:

O método híbrido foi utilizado com erros de posição e orientação máximos permitidos de, respectivamente, 1.10^{-4} e 0.1 *grau*, com numero máximo de iterações por ponto permitido de 10, computando-se os elementos da matriz $\boldsymbol{J}$ e sua pseudo-inversa de 15 em 15 pontos ($n = 15$) e com obstáculo de raio=0.1 m ($d_d = 0.1m$) na posição $[0,-0.2,0.6]^t(m)$.

Na fig. 5.5, mostra-se as coordenadas generalizadas de junta × tempo (s). Na fig. 5.6 mostra-se a distância ao obstáculo × tempo (s).

O tempo para a execução desta trajetória num microcomputador PC 486 25 Mh foi de 4.5 s.



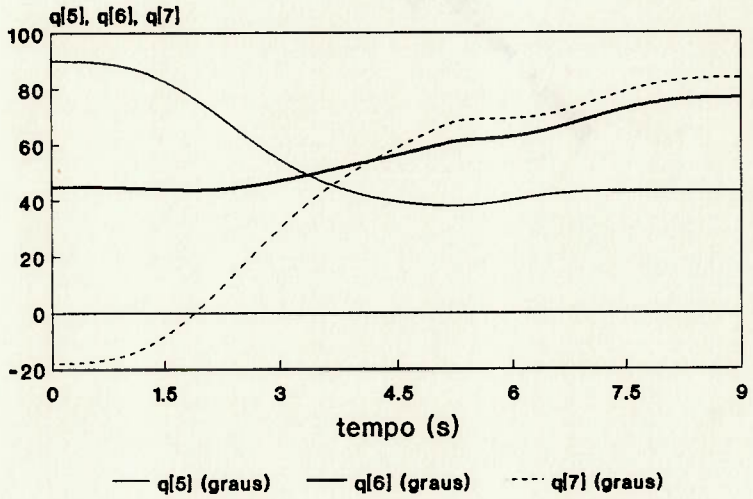


Figura 5.5: Coordenadas de junta × tempo

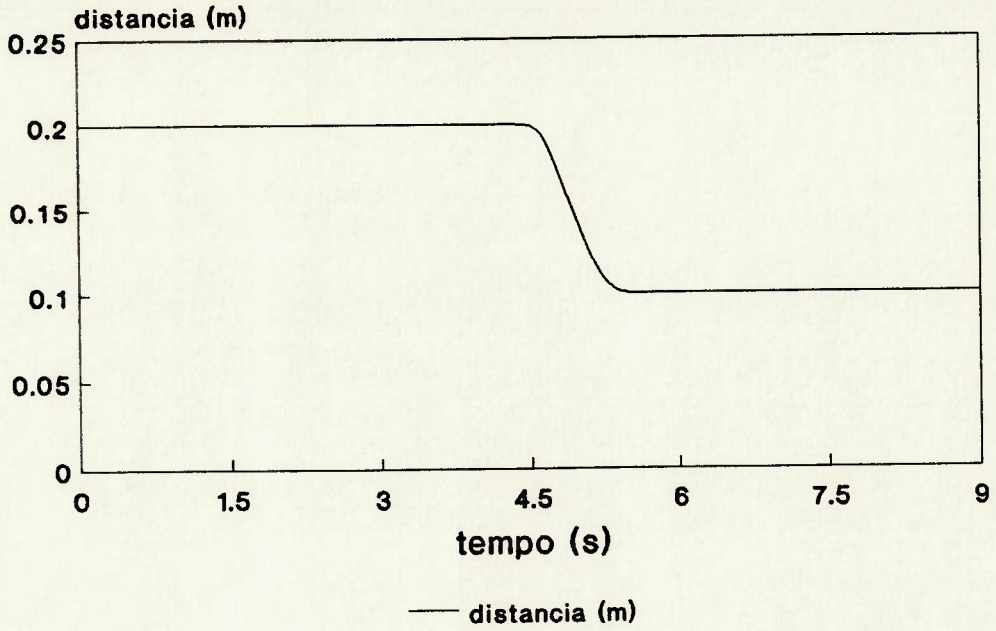


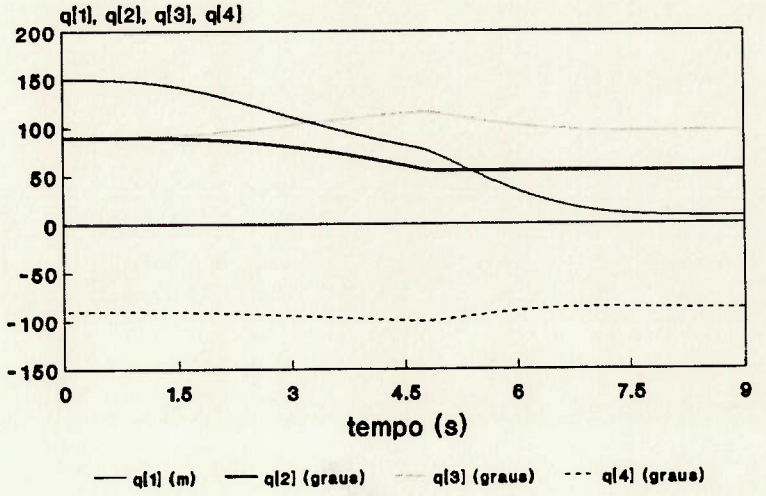
Figura 5.6: Distância ao obstáculo × tempo

Caso 3:

O método híbrido foi utilizado com erros de posição e orientação máximos permitidos de, respectivamente, 1.10^{-4} e 0.1 *grau*, com numero máximo de iterações por ponto permitido de 10, computando-se os elementos da matriz $\boldsymbol{J}$ e sua pseudo-inversa de 15 em 15 pontos ($n = 15$) e com q[2]$min = 55^o$.

Na fig. 5.7. mostra-se as coordenadas generalizadas de junta × tempo (s). Na fig. 5.8 mostra-se a distância ao obstáculo do caso 2 × tempo (s).

O tempo para a execução desta trajetória num microcomputador PC 486 25 Mh foi de 2.97 s.



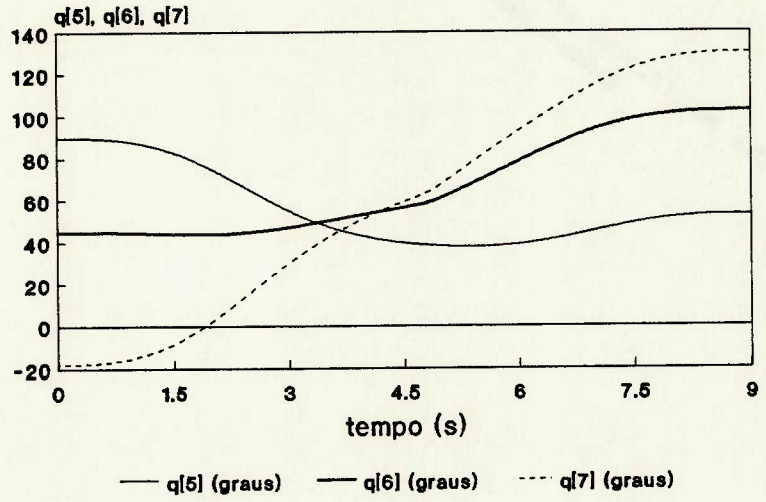


Figura 5.7: Coordenadas de junta × tempo

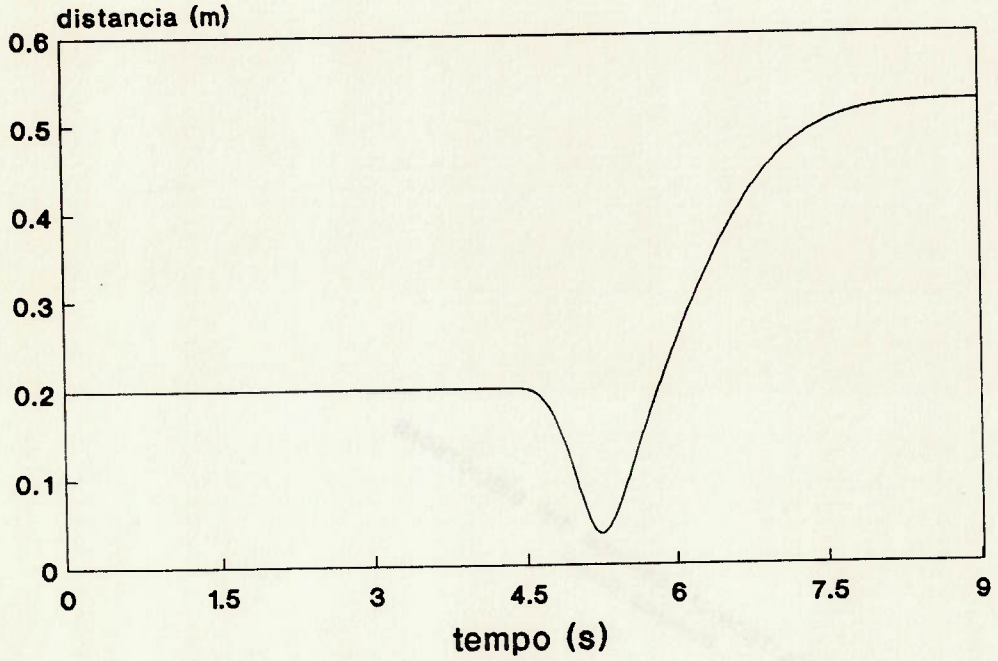


Figura 5.8: Distância ao obstáculo × tempo

Capítulo 6

SIMULADOR CINEMÁTICO

Neste capítulo, comenta-se aspectos do desenvolvimento do simulador e apresenta-se recursos e detalhes de seu funcionamento. Uma vez que as duas opções principais do simulador são **PONTO FIXO** e **TRAJETÓRIA** (Fig. 6.1), que resolvem a cinemática inversa para, respectivamente, estes dois casos, neste capítulo utiliza-se estas opções para descrição do sistema.

Assim como todos os algoritmos vistos nesta dissertação, o simulador foi elaborado utilizando-se o compilador Turbo Pascal 6.0.

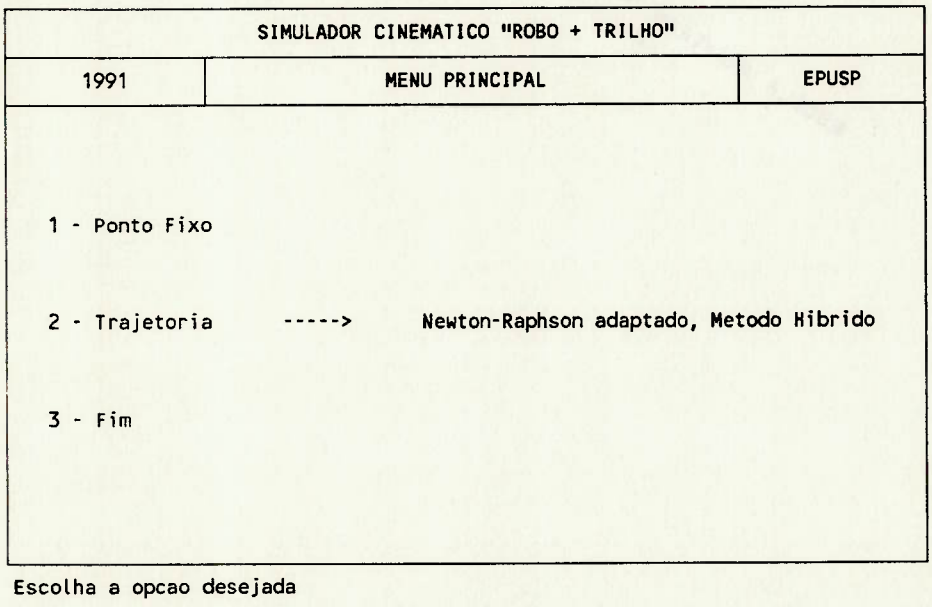


Figura 6.1: Menu principal do simulador

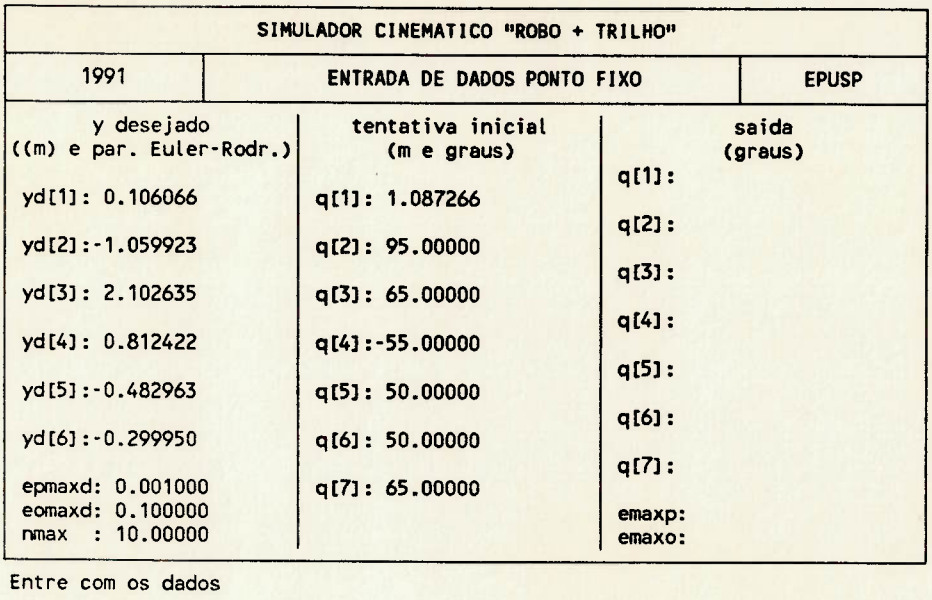


Figura 6.2: Entrada de dados para PONTO FIXO

6.1 PONTO-FIXO

Nesta opção, o simulador resolve a cinemática inversa para o sistema "robô ASEA IRBL6 + trilho" utilizando o método de *Newton-Raphson* para ponto-fixo (capítulo 3).

O sistema tem como entradas (Fig. 6.2) a posição e orientação desejadas (coordenadas cartesianas e parâmetros de *Euler-Rodrigues*), os erros máximos de posição e orientação desejados, o número máximo de iterações permitidas e a tentativa inicial (vetor $\boldsymbol{q}$inicial), e como saídas as coordenadas generalizadas de junta e os erros máximos (δ_p e δ_o) de posição e orientação correspondentes (Fig. 6.3).

6.2 TRAJETÓRIA

Neste caso, utilizou-se o método de *Newton-Raphson* adaptado e o método híbrido para resolver a cinemática inversa do sistema "robô ASEA IRBL6 + trilho" para casos de trajetória. O segundo método é utilizado quando deseja-se otimizar o vetor de coordenadas de junta segundo um dos dois critérios a seguir: limitar as coordenadas de alguma das juntas ou evitar obstáculos (Fig. 6.4).

Desta forma. escolhido o método a ser utilizado, o sistema desenvolvido tem como dados de entrada os parâmetros do algoritmo (Figs. 6.5, 6.6, 6.7) e a trajetória do efetuador em coordenadas cartesianas e parâmetros de *Euler-Rodrigues*, e como saídas, um arquivo contendo as coordenadas generalizadas de junta × tempo, os gráficos destas coordenadas × tempo, os erros máximos de posição e orientação (δ_p e δ_o) que ocorreram durante a

SIMULADOR CINEMATICO "ROBO + TRILHO"

1991	SAIDA DE DADOS PONTO FIXO	EPUSP

y desejado ((m) e par. Euler-Rodr.)	tentativa inicial (m e graus)	saida (graus)
yd[1]: 0.106066	q[1]: 1.087266	q[1]: 1.215000
yd[2]:-1.059923	q[2]: 95.00000	q[2]: 90.00000
yd[3]: 2.102635	q[3]: 65.00000	q[3]: 78.71331
yd[4]: 0.812422	q[4]:-55.00000	q[4]:-85.49327
yd[5]:-0.482963	q[5]: 50.00000	q[5]: 51.77996
yd[6]:-0.299950	q[6]: 50.00000	q[6]: 45.00000
epmaxd: 0.001000	q[7]: 65.00000	q[7]: 60.00000
eomaxd: 0.100000		emaxp: 0.000000
nmax : 10.00000		emaxo: 0.000000

Entre com os dados

Figura 6.3: Saída do simulador

execução da trajetória e o código de um programa escrito na linguagem de programação gráfica do "Sistema de Modelagem de Sólidos" (SMS) [TSU91], discutido abaixo, que permite a simulação gráfica dos movimentos do sistema.

6.2.1 PARTE GRÁFICA

Haja visto a dificuldade para interpretação dos resultados dos algoritmos para cinemática inversa, elaborou-se uma interface com o sistema gráfico SMS [TSU91]. Trata-se de um sistema CAD que possui recursos para modelagem de sólidos, animação gráfica e uma linguagem de programação gráfica.

A interface desenvolvida consiste basicamente de um modelo do sistema "robô ASEA IRBL6 + trilho" (apêndice B.1) e de um procedimento para geração do programa de animação. Assim, uma vez determinadas as coordenadas generalizadas de junta em certo instante da trajetória, o procedimento recebe estas coordenadas e gera um novo trecho de programa de animação. Este novo trecho representa o incremento que o vetor de coordenadas generalizadas de junta do instante anterior recebeu para colocar-se na nova posição. Com o programa de animação gerado pelo simulador (exemplo no apêndice B.2), basta entrar no sistema SMS e executar a animação.

Na Fig. 6.8, pode-se ver seis "frames" de animação da trajetória correspondentes ao caso 3 do estudo de caso do capítulo 5.

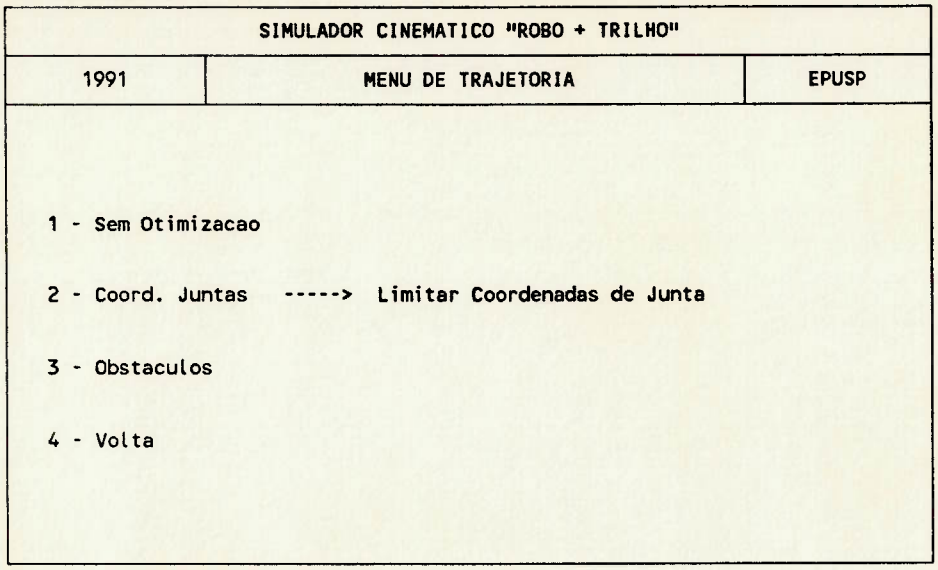


Figura 6.4: Menu de TRAJETÓRIA

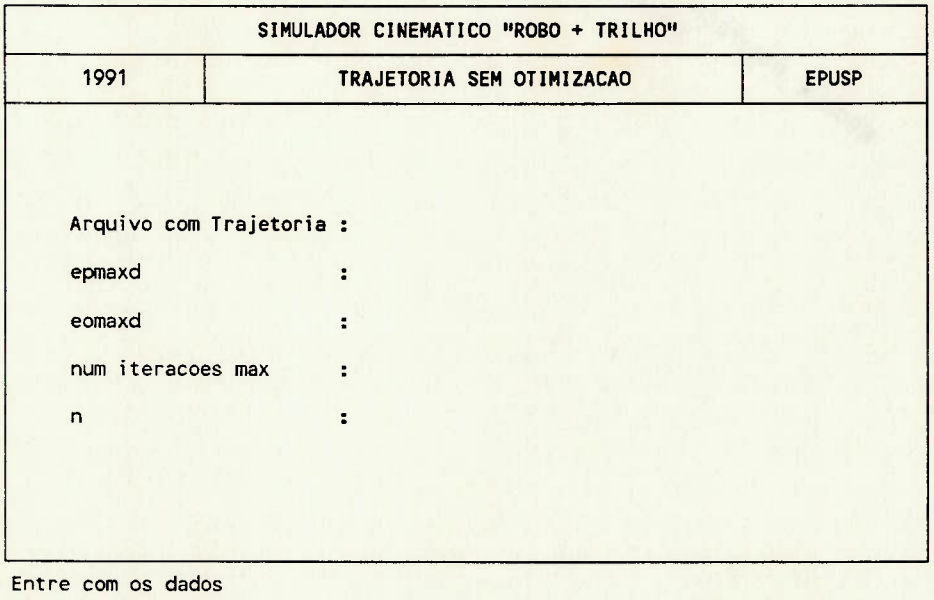


Figura 6.5: Entrada de dados para NEWTON-RAPHSON adaptado

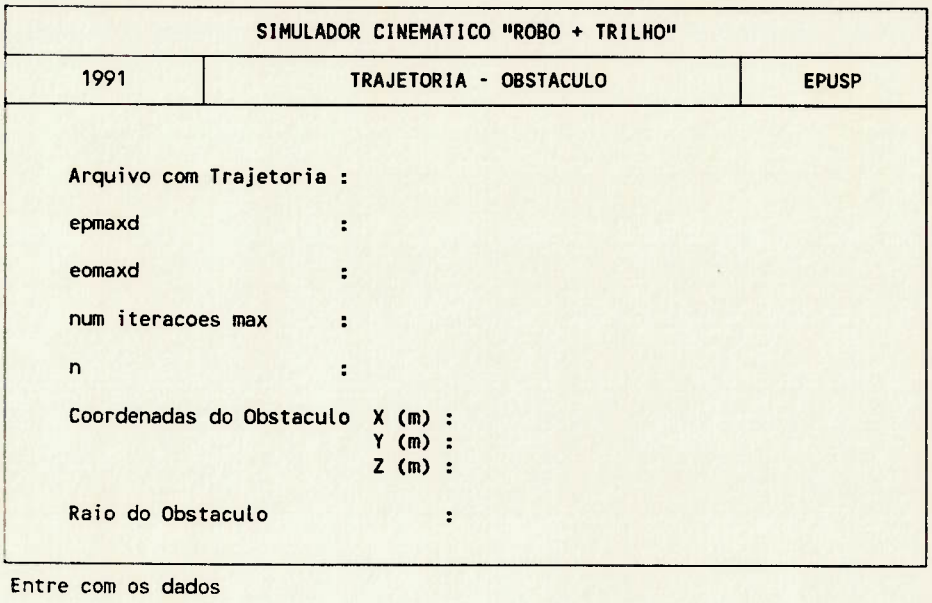


Figura 6.6: Entrada de dados para método híbrido: obstáculos

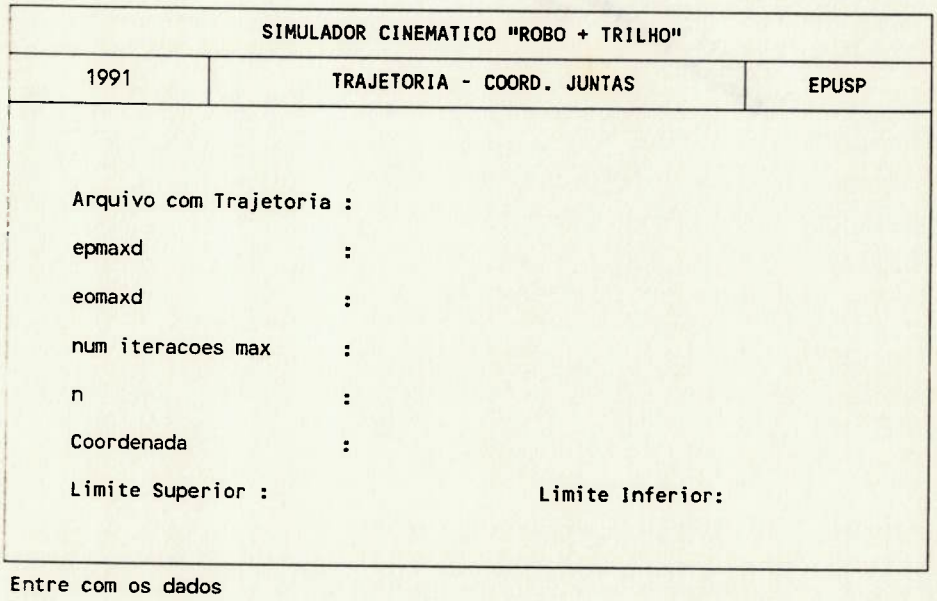


Figura 6.7: Entrada de dados para método híbrido: coordenadas de junta

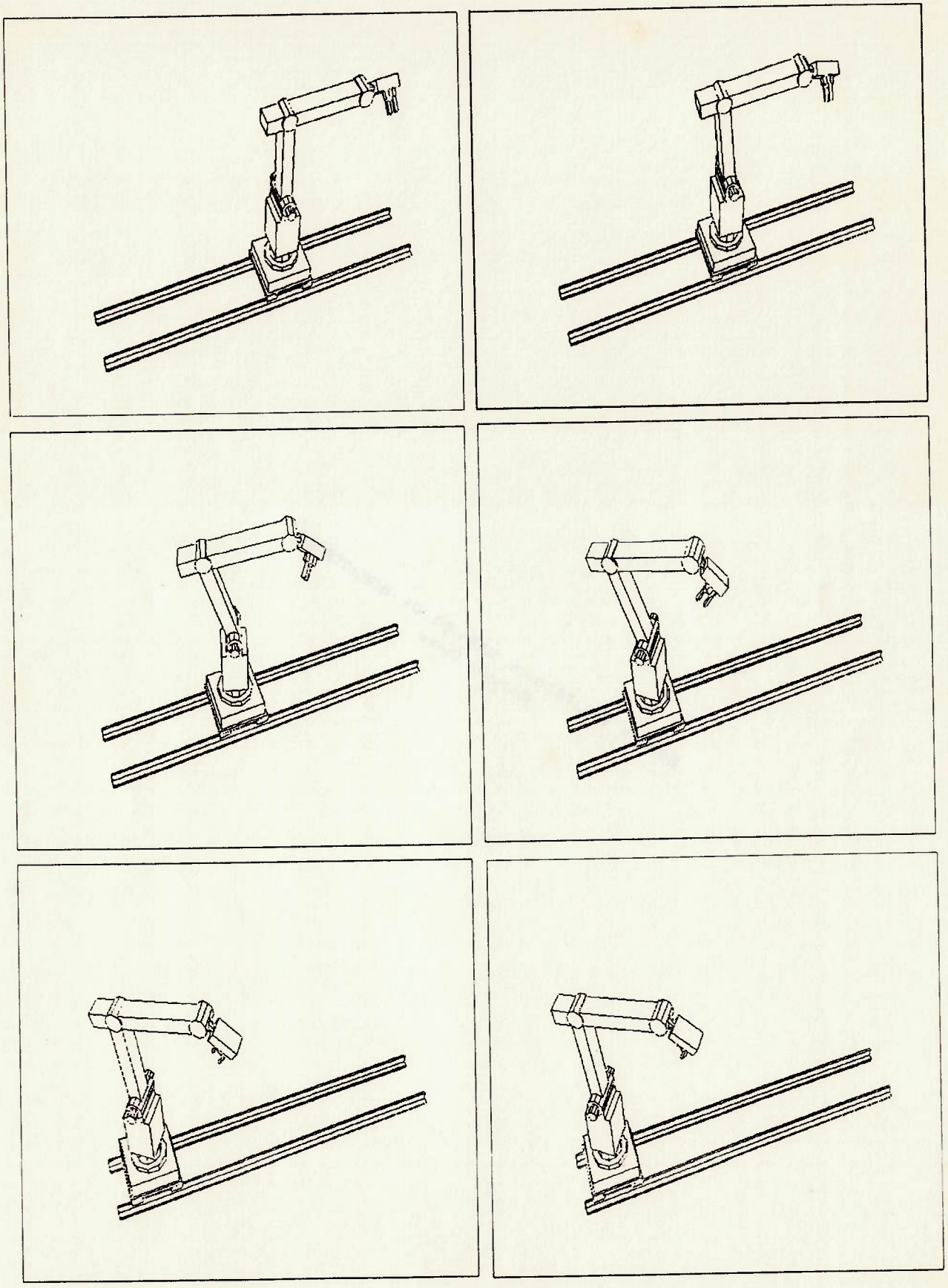


Figura 6.8: Animação do SMS

Capítulo 7

CONCLUSÕES

7.1 CONCLUSÕES

Nesta dissertação analizou-se técnicas para modelagem da posição e orientação do efetuador de robôs manipuladores em função das coordenadas de junta, e estudou-se algoritmos numéricos para a solução do problema inverso. Deste estudo resultaram adaptações em dois métodos numéricos (*Newton-Raphson* e *Sciavicco e Siciliano*), que geraram dois algoritmos novos.

As simulações efetuadas mostraram que a adaptação sugerida no capítulo 3 (método de *Newton-Raphson* adaptado), para casos de trajetória, torna o algoritmo mais veloz que o original e mais eficiente que o método baseado em *Sliding Mode Control*, também para trajetória. Este último, pertence a uma nova classe de métodos numéricos para cinemática inversa (começaram a surgir a partir da metade da década de 80), estudados no capítulo 4, que não necessitam inverter matrizes.

A proposta de utilizar a idéia de *Sciavicco e Siciliano* (capítulo 5), que consiste em introduzir equações adicionais em um sistema redundante, de forma a torná-lo determinado e a "trazer" a solução do sistema original para alguma faixa de interesse (segundo o critério que se deseje otimizar), em conjunto com o método de *Newton-Raphson* adaptado, mostrou-se de entendimento e implementação mais fáceis e de eficiência maior que o método original. Chamou-se este novo algoritmo de método híbrido.

Realizou-se simulações com o método híbrido utilizando-se o modelo do sistema "robô ASEA IRBL6 + trilho" (7 graus de liberdade), deduzido no capítulo 2, que ilustram o seu bom funcionamento para os dois critérios de otimização abordados: limitar coordenadas de junta e evitar obstáculos.

Finalmente (capítulo 6), utilizou-se o método de *Newton-Raphson* adaptado e o método híbrido para o desenvolvimento de um simulador cinemático para o sistema "robô ASEA IRBL6 + trilho". Assim, mostrou-se que a introdução do trilho em um robô de 6 graus de liberdade, muito comum em células e sistemas flexíveis de manufatura, pode ser

explorada de forma a aproveitar a redundância introduzida, para gerar coordenadas de junta otimizadas segundo algum critério.

7.2 SUGESTÕES PARA TRABALHOS FUTUROS

O assunto cinemática de manipuladores tem sido muito estudado nos últimos anos e encontra-se diversos temas de pesquisa nesta área. Neste ítem coloca-se algumas idéias e propostas para trabalhos futuros que surgiram durante a execução desta dissertação.

7.2.1 MODELAGEM DIFERENCIAL DIRETA

No capítulo 2, foi visto que a obtenção do modelo diferencial direto pelo método de coordenadas operacionais (extração direta dos parâmetros de *Euler-Rodrigues* da matriz de transformação homogênea), fornece expressões com denominadores que podem anular-se com freqüência. Foi visto, ainda, que ao encontra-se o modelo diferencial direto a partir do modelo de *Whitney*, este problema é contornado. Este fato sugere que, apesar de ainda não descobertas, devem existir expressões que permitem a extração do modelo diferencial diretamente da matriz de transformação homogênea. Uma pesquisa desta natureza poderia ser extendida a diversos parâmetros para descrição da orientação de efetuadores (ângulos de *Euler*, ângulos roll, pitch e yaw, etc...) e, certamente, as expressões resultantes seriam muito úteis para a modelagem de manipuladores.

7.2.2 OUTROS CRITÉRIOS PARA MANIPULADORES REDUNDANTES

Além dos dois critérios de otimização estudados no capítulo 5 (limitar coordenadas de junta e evitar obstáculos), na literatura encontra-se diversos outros, como evitar posições de singularidade e limitar torques nos atuadores a determinados valores máximos, por exemplo. Uma pesquisa aqui sugerida, é extender a utilização do método híbrido a outros critérios de otimização. A extensão do método para limitar alguma função das coordenadas generalizadas de junta a certos limites, pode ser feita mesmo que não se conheça a expressão da função, como acontecia com a função distância a obstáculos (capítulo 5).

7.2.3 SISTEMA PARA GERENCIAMENTO DAS EQUAÇÕES ADICIONAIS

O método híbrido não permite a introdução de mais funções adicionais que o número de graus de liberdade redundantes do manipulador. Este problema poderia ser amenizado se fossem estudados critérios para introdução e retirada das equações adicionais. Assim, pode ser que em determinado instante da execução da trajetória, o critério limitar coordenada

de junta q_n esteja ativo e impeça que, por exemplo, um limite em q_x seja introduzido, mesmo que o limite em q_n não seja mais necessário. Um estudo deste tipo poderia resultar em estratégias que tornassem o método híbrido bem mais flexível.

7.2.4 ALGORITMOS PARA CÁLCULO DE DISTÂNCIA A OBJETOS DE FORMAS VARIADAS

No caso de evitar obstáculos, o método híbrido ganharia, novamente, mais flexibilidade se fossem estudados e implementados algoritmos que permitissem o cálculo de distância a obstáculos de formas variadas. Uma vez implementados os algoritmos, o equacionamento desenvolvido no capítulo 5, para obstáculos esféricos, continua válido, independentemente do formato dos obstáculos.

7.2.5 SISTEMA PARA ENSINO DE ROBÓTICA

Os métodos numéricos para cinemática inversa vistos nesta dissertação, poderiam ser incluídos no sistema para ensino de robótica, proposto por *Molinaro* [MOL91], em sua tese de doutorado, realizada nesta escola. Atualmente, o sistema soluciona a cinemática inversa somente para alguns manipuladores que possuem solução analítica. Com esta adaptação, o sistema de *Molinaro* seria capaz de solucionar a cinemática inversa de um manipulador genérico, incluindo manipuladores redundantes, o que seria bastante útil para fins de ensino de robótica (proposta do sistema de *Molinaro*).

Apêndice A

PSEUDO-INVERSA DE UMA MATRIZ

Seja o sistema linear de equações $\boldsymbol{A}.\boldsymbol{x} = \boldsymbol{y}$, onde $\boldsymbol{A}$ é uma matriz $m \times n$, $\boldsymbol{x}$ é um vetor de n componentes e $\boldsymbol{y}$ é um vetor de m componentes. Uma condição necessária e suficiente para que este sistema tenha solução é que $\boldsymbol{A}.\boldsymbol{A}_1^*.\boldsymbol{y} = \boldsymbol{y}$ [GRE59,BOU71], e neste caso a solução geral é dada por:

$$\boldsymbol{x} = \boldsymbol{A}_1^*.\boldsymbol{y} + (\boldsymbol{A}_2^*.\boldsymbol{A} - \boldsymbol{I}_n).\boldsymbol{Z} \qquad \text{(A.1)}$$

onde

- $\boldsymbol{A}_1^*$ e $\boldsymbol{A}_2^*$ são quaisquer matrizes $n \times m$, tais que $\boldsymbol{A}.\boldsymbol{A}_1^*.\boldsymbol{A} = \boldsymbol{A}.\boldsymbol{A}_2^*.\boldsymbol{A} = \boldsymbol{A}$ e são denominadas inversas generalizadas de $\boldsymbol{A}$;
- $\boldsymbol{I}_n$ é a matriz identidade $n \times n$;
- $\boldsymbol{Z}$ é um vetor qualquer pertencente a $\boldsymbol{R}^n$.

O matemático Inglês *Penrose* mostrou, em 1955, que quando toma-se $\boldsymbol{A}_1^*=\boldsymbol{A}_2^*=\boldsymbol{A}^+$, onde $\boldsymbol{A}^+$ é uma inversa generalizada de $\boldsymbol{A}$ que satisfaz, além da primeira condição acima, as seguintes,

$$\begin{aligned} \boldsymbol{A}^+.\boldsymbol{A}.\boldsymbol{A}^+ &= \boldsymbol{A}^+ \\ (\boldsymbol{A}^+.\boldsymbol{A})^t &= \boldsymbol{A}^+.\boldsymbol{A} \\ (\boldsymbol{A}.\boldsymbol{A}^+)^t &= \boldsymbol{A}.\boldsymbol{A}^+ \end{aligned}$$

o primeiro termo de (A.1), fornece o resultado de menor norma euclidiana do vetor $\boldsymbol{x}$, quando existe solução para o sistema, ou então o resultado mais próximo (menor norma do vetor $\boldsymbol{A}.\boldsymbol{x} - \boldsymbol{y}$), quando a solução não existe. O segundo termo de (A.1), fornece a projeção do vetor $\boldsymbol{Z}$, no espaço nulo de $\boldsymbol{A}$.

À matriz $\boldsymbol{A}^+$, dá-se o nome de pseudo-inversa de $\boldsymbol{A}$. *Penrose* demostrou que a pseudo-inversa sempre existe e é única. Em [GRE59], encontra-se as seguintes expressões

[1] para $\boldsymbol{A}^+$, válidas quando $\boldsymbol{A}$ é de posto máximo:

$$\boldsymbol{A}^+ = \boldsymbol{A}^t.(\boldsymbol{A}.\boldsymbol{A}^t)^{-1} \qquad \text{se } m < n, \tag{A.2}$$

$$\boldsymbol{A}^+ = (\boldsymbol{A}^t.\boldsymbol{A})^{-1}.\boldsymbol{A}^t \qquad \text{se } n < m. \tag{A.3}$$

Note que quando $\boldsymbol{A}^{-1}$ existe, $\boldsymbol{A}^+ = \boldsymbol{A}^{-1}$.

Uma vez que o sistema (2.36), assim como $\boldsymbol{A}.\boldsymbol{x}=\boldsymbol{y}$, é linear, e como a pseudo-inversa possui vantagens em relação à inversa, neste trabalho utilizou-se a matriz $\boldsymbol{J}^+$ ao invés de $\boldsymbol{J}^{-1}$. Para o cálculo da pseudo-inversa de $\boldsymbol{J}$ utilizou-se o algoritmo de *Greville* [GRE59].

[1] Estas expressões podem ser encontradas por álgebra linear a partir da definição de pseudo-inversa [BOU71], ou então utilizando-se multiplicadores de Lagrange com a igualdade $\boldsymbol{A}.\boldsymbol{x}=\boldsymbol{y}$ tomada como restrição [ASA85].

Apêndice B

B.1 MODELO DO SISTEMA "Robô ASEA IRBL6 + trilho"

Aqui, apresenta-se o modelo do sistema "robô ASEA IRBL6 + trilho" no CAD SMS [TSU91].

```
camera -g cam1 300 400 500 150 -40 0 3 1 3 0 0
camera -g cam2 -300 400 500 450 -40 0 3 1 3 0 0
group -G hand1
prism hand 8 3
0 0
9 0
9 10
7.5 10
7.5 2
1.5 2
1.5 10
0 10
move hand 107.9 34 151
axis -r ehand1 112.4 29 152.5 0 1 0
axis -a ehand1 hand1
group -c \
group -G gar
group -p gar hand1
cube mao 20 8 12.5
move mao 99.5 21 146.25
cylinder garra 10 5 5
rotate garra 0 0 0 0 1 0 90
rotate garra 0 0 0 0 0 1 90
move garra 112.5 29 152.5
axis -r egar 89.5 25 152.5 1 0 0
axis -a egar gar
```

```
group -c \
group -G gar2
group -p gar2 gar hand1
cone con 10 5.5 12
rotate con 0 0 0 0 1 0 90
move con 92 25 152.5
axis -r egar2 89.5 25 152.5 0 -1 0
axis -a egar2 gar2
group -c \
group -G brac1
group -p brac1 gar2 gar hand1
cube braco2 14 10 86
rotate braco2 0 0 0 1 0 0 90
rotate braco2 0 0 0 0 0 1 90
move braco2 3.5 18 147.5
cylinder corpcil4 10 6.5 16
rotate corpcil4 0 0 0 0 1 0 90
rotate corpcil4 0 0 0 0 0 1 90
move corpcil4 89.5 17 152.5
axis -r ebrac1 22.5 25 152.5 0 -1 0
axis -a ebrac1 brac1
group -c \
group -G brac2
group -p brac2 brac1 gar2 gar hand1
cube braco 10 14 69
move braco 17.5 18 83.5
cylinder corpcil3 10 6.5 16
rotate corpcil3 0 0 0 0 1 0 90
rotate corpcil3 0 0 0 0 0 1 90
move corpcil3 22.5 17 152.5
axis -r ebrac2 22.5 25 83.5 0 -1 0
axis -a ebrac2 brac2
group -c \
group -G corp
group -p corp brac2 brac1 gar2 gar hand1
cylinder robcil 10 15.5 4
move robcil 22.5 25 24.5
cylinder robcil2 10 11 10
move robcil2 22.5 25 28.5
cube corpo 22 25 35
move corpo 11.5 12.5 38.5
cube corpo2 13 25 6.5
move corpo2 16 12.5 73.5
cylinder corpcil 10 3.7 40
rotate corpcil 0 0 0 0 1 0 90
rotate corpcil 0 0 0 0 0 1 90
```

```
move corpcil 22.5 5 83.5
cylinder corpcil2 10 6.5 27
rotate corpcil2 0 0 0 0 1 0 90
rotate corpcil2 0 0 0 0 0 1 90
move corpcil2 22.5 11.5 83.5
axis -r ecorp 22.5 25 24.5 0 0 1
axis -a ecorp corp
group -c \
group -G bas
group -p bas corp brac2 brac1 gar2 gar hand1
cube rolam 10 5 4
move rolam  0 2.5 7
copy rolam rolam2
move rolam2 35 0 0
copy rolam rolam3
move rolam3 0 40 0
copy rolam2 rolam4
move rolam4 0 40 0
cube base 45 45 2.5
move base 0 2.5 11
cube robase 41 41 11
move robase 2 4.5 13.5
axis -s ebas -1 0 0
axis -a ebas bas
group -c \
group -G trilho
group -p trilho bas corp brac2 brac1 gar2 gar hand1
prism tril 8 300
0 0
10 0
10 1.5
5.75 1.5
5.75 7.5
4.25 7.5
4.25 1.5
0 1.5
rotate tril 0 0 0 0 1 0 90
rotate tril 0 0 0 1 0 0 90
copy tril tril2
move tril2 0 40 0
axis -m ebas -255
axis -m ecorp 90
axis -m ebrac2 -90
axis -m ebrac1 90
axis -m egar 180
axis -m egar2 -90
```

```
axis -m ehand1 90
cl -g
end
```

B.2 EXEMPLO DE PROGRAMA GERADO

O programa a seguir foi gerado para o caso 3 do capítulo 5.

```
camera -g cam1 300 400 500 150 -40 0 3 1 3 0 0
camera -g cam2 -300 400 500 450 -40 0 3 1 3 0 0
group -G hand1
prism hand 8 3
0 0
9 0
9 10
7.5 10
7.5 2
1.5 2
1.5 10
0 10
move hand 107.9 34 151
axis -r ehand1 112.4 29 152.5 0 1 0
axis -a ehand1 hand1
group -c \
group -G gar
group -p gar hand1
cube mao 20 8 12.5
move mao 99.5 21 146.25
cylinder garra 10 5 5
rotate garra 0 0 0 0 1 0 90
rotate garra 0 0 0 0 0 1 90
move garra 112.5 29 152.5
axis -r egar 89.5 25 152.5 1 0 0
axis -a egar gar
group -c \
group -G gar2
group -p gar2 gar hand1
cone con 10 5.5 12
rotate con 0 0 0 0 1 0 90
move con 92 25 152.5
axis -r egar2 89.5 25 152.5 0 -1 0
axis -a egar2 gar2
group -c \
group -G brac1
```

```
group -p brac1 gar2 gar hand1
cube braco2 14 10 86
rotate braco2 0 0 0 1 0 0 90
rotate braco2 0 0 0 0 0 1 90
move braco2 3.5 18 147.5
cylinder corpcil4 10 6.5 16
rotate corpcil4 0 0 0 0 1 0 90
rotate corpcil4 0 0 0 0 0 1 90
move corpcil4 89.5 17 152.5
axis -r ebrac1 22.5 25 152.5 0 -1 0
axis -a ebrac1 brac1
group -c \
group -G brac2
group -p brac2 brac1 gar2 gar hand1
cube braco 10 14 69
move braco 17.5 18 83.5
cylinder corpcil3 10 6.5 16
rotate corpcil3 0 0 0 0 1 0 90
rotate corpcil3 0 0 0 0 0 1 90
move corpcil3 22.5 17 152.5
axis -r ebrac2 22.5 25 83.5 0 -1 0
axis -a ebrac2 brac2
group -c \
group -G corp
group -p corp brac2 brac1 gar2 gar hand1
cylinder robcil 10 15.5 4
move robcil 22.5 25 24.5
cylinder robcil2 10 11 10
move robcil2 22.5 25 28.5
cube corpo 22 25 35
move corpo 11.5 12.5 38.5
cube corpo2 13 25 6.5
move corpo2 16 12.5 73.5
cylinder corpcil 10 3.7 40
rotate corpcil 0 0 0 0 1 0 90
rotate corpcil 0 0 0 0 0 1 90
move corpcil 22.5 5 83.5
cylinder corpcil2 10 6.5 27
rotate corpcil2 0 0 0 0 1 0 90
rotate corpcil2 0 0 0 0 0 1 90
move corpcil2 22.5 11.5 83.5
axis -r ecorp 22.5 25 24.5 0 0 1
axis -a ecorp corp
group -c \
group -G bas
group -p bas corp brac2 brac1 gar2 gar hand1
```

```
cube rolam 10 5 4
move rolam  0 2.5 7
copy rolam rolam2
move rolam2 35 0 0
copy rolam rola.5 7
copy rolam rolam2
move rolam2 35 0 0
copy rolam rolam3
move rolam3 0 40 0
copy rolam2 rolam4
move rolam4 0 40 0
cube base 45 45 2.5
move base 0 2.5 11
cube robase 41 41 11
move robase 2 4.5 13.5
axis -s ebas -1 0 0
axis -a ebas bas
group -c \
group -G trilho
group -p trilho bas corp brac2 brac1 gar2 gar hand1
prism tril 8 300
0 0
10 0
10 1.5
5.75 1.5
5.75 7.5
4.25 7.5
4.25 1.5
0 1.5
rotate tril 0 0 0 0 1 0 90
rotate tril 0 0 0 1 0 0 90
copy tril tril2
move tril2 0 40 0
axis -m ebas -255
axis -m ecorp 90
axis -m ebrac2 -90
axis -m ebrac1 90
axis -m egar 180
axis -m egar2 -90
axis -m ehand1 90
cl -g
axis -m ebas 150.000
axis -m ecorp 90.000
axis -m ebrac2 90.000
axis -m ebrac1 -90.000
axis -m egar2 90.000
axis -m egar 45.000
axis -m ehand1 -18.000
ani -i
```

```
ani -a ebas -0.022
ani -a ecorp -0.003
ani -a ebrac2  0.006
ani -a ebrac1 -0.002
ani -a egar2 -0.019
ani -a egar -0.003
ani -a ehand1  0.026
ani -m   1 1 avm cam1
ani -i
ani -a ebas -0.152
ani -a ecorp -0.021
ani -a ebrac2  0.044
ani -a ebrac1 -0.016
ani -a egar2 -0.130
ani -a egar -0.022
ani -a ehand1  0.179
ani -m   2 1 avm cam1
ani -i
ani -a ebas -0.405
ani -a ecorp -0.056
ani -a ebrac2  0.118
ani -a ebrac1 -0.042
ani -a egar2 -0.347
ani -a egar -0.059
ani -a ehand1  0.479
ani -m   3 1 avm cam1
ani -i
ani -a ebas -0.770
ani -a ecorp -0.109
ani -a ebrac2  0.226
ani -a ebrac1 -0.079
ani -a egar2 -0.664
ani -a egar -0.108
ani -a ehand1  0.913
ani -m   4 1 avm cam1
ani -i
ani -a ebas -1.225
ani -a ecorp -0.177
ani -a ebrac2  0.362
ani -a ebrac1 -0.124
ani -a egar2 -1.068
ani -a egar -0.160
ani -a ehand1  1.462
ani -m   5 1 avm cam1
ani -i
ani -a ebas -1.745
```

```
ani -a ecorp -0.262
ani -a ebrac2  0.522
ani -a ebrac1 -0.173
ani -a egar2 -1.547
ani -a egar -0.203
ani -a ehand1  2.103
ani -m   6 1 avm cam1
ani -i
ani -a ebas -2.298
ani -a ecorp -0.365
ani -a ebrac2  0.699
ani -a ebrac1 -0.223
ani -a egar2 -2.077
ani -a egar -0.221
ani -a ehand1  2.804
ani -m   7 1 avm cam1
ani -i
ani -a ebas -2.851
ani -a ecorp -0.485
ani -a ebrac2  0.884
ani -a ebrac1 -0.272
ani -a egar2 -2.630
ani -a egar -0.199
ani -a ehand1  3.526
ani -m   8 1 avm cam1
ani -i
ani -a ebas -3.369
ani -a ecorp -0.624
ani -a ebrac2  1.070
ani -a ebrac1 -0.317
ani -a egar2 -3.173
ani -a egar -0.126
ani -a ehand1  4.226
ani -m   9 1 avm cam1
ani -i
ani -a ebas -3.819
ani -a ecorp -0.784
ani -a ebrac2  1.250
ani -a ebrac1 -0.365
ani -a egar2 -3.642
ani -a egar  0.007
ani -a ehand1  4.846
ani -m  10 1 avm cam1
ani -i
ani -a ebas -4.185
ani -a ecorp -0.964
```

```
ani -a ebrac2  1.409
ani -a ebrac1 -0.416
ani -a egar2 -3.991
ani -a egar  0.194
ani -a ehand1  5.336
ani -m  11 1 avm cam1
ani -i
ani -a ebas -4.444
ani -a ecorp -1.161
ani -a ebrac2  1.546
ani -a ebrac1 -0.474
ani -a egar2 -4.188
ani -a egar  0.415
ani -a ehand1  5.655
ani -m  12 1 avm cam1
ani -i
ani -a ebas -4.589
ani -a ecorp -1.374
ani -a ebrac2  1.657
ani -a ebrac1 -0.549
ani -a egar2 -4.180
ani -a egar  0.651
ani -a ehand1  5.765
ani -m  13 1 avm cam1
ani -i
ani -a ebas -4.634
ani -a ecorp -1.591
ani -a ebrac2  1.724
ani -a ebrac1 -0.620
ani -a egar2 -3.993
ani -a egar  0.869
ani -a ehand1  5.676
ani -m  14 1 avm cam1
ani -i
ani -a ebas -4.568
ani -a ecorp -1.799
ani -a ebrac2  1.754
ani -a ebrac1 -0.680
ani -a egar2 -3.673
ani -a egar  1.040
ani -a ehand1  5.414
ani -m  15 1 avm cam1
ani -i
ani -a ebas -4.413
ani -a ecorp -1.994
ani -a ebrac2  1.753
```

```
ani -a ebrac1 -0.733
ani -a egar2 -3.246
ani -a egar  1.157
ani -a ehand1  5.021
ani -m  16 1 avm cam1
ani -i
ani -a ebas -4.257
ani -a ecorp -2.189
ani -a ebrac2  1.724
ani -a ebrac1 -0.759
ani -a egar2 -2.822
ani -a egar  1.229
ani -a ehand1  4.617
ani -m  17 1 avm cam1
ani -i
ani -a ebas -4.098
ani -a ecorp -2.387
ani -a ebrac2  1.684
ani -a ebrac1 -0.769
ani -a egar2 -2.427
ani -a egar  1.268
ani -a ehand1  4.231
ani -m  18 1 avm cam1
ani -i
ani -a ebas -3.937
ani -a ecorp -2.589
ani -a ebrac2  1.630
ani -a ebrac1 -0.766
ani -a egar2 -2.045
ani -a egar  1.280
ani -a ehand1  3.860
ani -m  19 1 avm cam1
ani -i
ani -a ebas -3.790
ani -a ecorp -2.779
ani -a ebrac2  1.544
ani -a ebrac1 -0.734
ani -a egar2 -1.693
ani -a egar  1.277
ani -a ehand1  3.524
ani -m  20 1 avm cam1
ani -i
ani -a ebas -3.643
ani -a ecorp -2.953
ani -a ebrac2  1.437
ani -a ebrac1 -0.685
```

```
ani -a egar2 -1.375
ani -a egar  1.268
ani -a ehand1  3.224
ani -m  21 1 avm cam1
ani -i
ani -a ebas -3.494
ani -a ecorp -3.105
ani -a ebrac2  1.311
ani -a ebrac1 -0.624
ani -a egar2 -1.079
ani -a egar  1.259
ani -a ehand1  2.960
ani -m  22 1 avm cam1
ani -i
ani -a ebas -3.348
ani -a ecorp -3.223
ani -a ebrac2  1.165
ani -a ebrac1 -0.551
ani -a egar2 -0.815
ani -a egar  1.263
ani -a ehand1  2.743
ani -m  23 1 avm cam1
ani -i
ani -a ebas -3.187
ani -a ecorp -3.305
ani -a ebrac2  1.015
ani -a ebrac1 -0.475
ani -a egar2 -0.585
ani -a egar  1.283
ani -a ehand1  2.570
ani -m  24 1 avm cam1
ani -i
ani -a ebas -7.140
ani -a ecorp -0.703
ani -a ebrac2 -2.153
ani -a ebrac1  1.559
ani -a egar2 -0.538
ani -a egar  2.870
ani -a ehand1  4.905
ani -m  25 1 avm cam1
ani -i
ani -a ebas -7.883
ani -a ecorp  0.000
ani -a ebrac2 -2.792
ani -a ebrac1  2.023
ani -a egar2 -0.318
```

```
ani -a egar  3.381
ani -a ehand1  5.323
ani -m  26 1 avm cam1
ani -i
ani -a ebas -7.475
ani -a ecorp  0.000
ani -a ebrac2 -2.582
ani -a ebrac1  1.930
ani -a egar2 -0.066
ani -a egar  3.454
ani -a ehand1  5.117
ani -m  27 1 avm cam1
ani -i
ani -a ebas -7.037
ani -a ecorp  0.000
ani -a ebrac2 -2.363
ani -a ebrac1  1.812
ani -a egar2  0.177
ani -a egar  3.499
ani -a ehand1  4.957
ani -m  28 1 avm cam1
ani -i
ani -a ebas -6.565
ani -a ecorp  0.000
ani -a ebrac2 -2.136
ani -a ebrac1  1.668
ani -a egar2  0.416
ani -a egar  3.517
ani -a ehand1  4.843
ani -m  29 1 avm cam1
ani -i
ani -a ebas -6.063
ani -a ecorp  0.000
ani -a ebrac2 -1.901
ani -a ebrac1  1.495
ani -a egar2  0.656
ani -a egar  3.507
ani -a ehand1  4.772
ani -m  30 1 avm cam1
ani -i
ani -a ebas -5.513
ani -a ecorp  0.000
ani -a ebrac2 -1.650
ani -a ebrac1  1.287
ani -a egar2  0.902
ani -a egar  3.464
```

```
ani -a ehand1  4.719
ani -m  31 1 avm cam1
ani -i
ani -a ebas -4.858
ani -a ecorp  0.000
ani -a ebrac2 -1.370
ani -a ebrac1  1.035
ani -a egar2  1.135
ani -a egar  3.335
ani -a ehand1  4.618
ani -m  32 1 avm cam1
ani -i
ani -a ebas -4.126
ani -a ecorp  0.000
ani -a ebrac2 -1.078
ani -a ebrac1  0.756
ani -a egar2  1.332
ani -a egar  3.119
ani -a ehand1  4.447
ani -m  33 1 avm cam1
ani -i
ani -a ebas -3.379
ani -a ecorp  0.000
ani -a ebrac2 -0.796
ani -a ebrac1  0.482
ani -a egar2  1.474
ani -a egar  2.835
ani -a ehand1  4.203
ani -m  34 1 avm cam1
ani -i
ani -a ebas -2.664
ani -a ecorp  0.000
ani -a ebrac2 -0.545
ani -a ebrac1  0.236
ani -a egar2  1.546
ani -a egar  2.502
ani -a ehand1  3.886
ani -m  35 1 avm cam1
ani -i
ani -a ebas -2.023
ani -a ecorp  0.000
ani -a ebrac2 -0.338
ani -a ebrac1  0.040
ani -a egar2  1.538
ani -a egar  2.141
ani -a ehand1  3.500
```

```
ani -m  36 1 avm cam1
ani -i
ani -a ebas -1.478
ani -a ecorp  0.000
ani -a ebrac2 -0.182
ani -a ebrac1 -0.096
ani -a egar2  1.451
ani -a egar  1.776
ani -a ehand1  3.055
ani -m  37 1 avm cam1
ani -i
ani -a ebas -1.041
ani -a ecorp  0.000
ani -a ebrac2 -0.075
ani -a ebrac1 -0.172
ani -a egar2  1.294
ani -a egar  1.422
ani -a ehand1  2.568
ani -m  38 1 avm cam1
ani -i
ani -a ebas -0.704
ani -a ecorp  0.000
ani -a ebrac2 -0.011
ani -a ebrac1 -0.197
ani -a egar2  1.086
ani -a egar  1.093
ani -a ehand1  2.061
ani -m  39 1 avm cam1
ani -i
ani -a ebas -0.459
ani -a ecorp  0.000
ani -a ebrac2  0.019
ani -a ebrac1 -0.182
ani -a egar2  0.848
ani -a egar  0.799
ani -a ehand1  1.560
ani -m  40 1 avm cam1
ani -i
ani -a ebas -0.283
ani -a ecorp  0.000
ani -a ebrac2  0.027
ani -a ebrac1 -0.144
ani -a egar2  0.609
ani -a egar  0.544
ani -a ehand1  1.094
ani -m  41 1 avm cam1
```

```
ani -i
ani -a ebas -0.161
ani -a ecorp  0.000
ani -a ebrac2  0.023
ani -a ebrac1 -0.098
ani -a egar2  0.389
ani -a egar  0.334
ani -a ehand1  0.687
ani -m  42 1 avm cam1
ani -i
ani -a ebas -0.080
ani -a ecorp  0.000
ani -a ebrac2  0.014
ani -a ebrac1 -0.053
ani -a egar2  0.206
ani -a egar  0.175
ani -a ehand1  0.362
ani -m  43 1 avm cam1
ani -i
ani -a ebas -0.030
ani -a ecorp  0.000
ani -a ebrac2  0.005
ani -a ebrac1 -0.020
ani -a egar2  0.077
ani -a egar  0.065
ani -a ehand1  0.135
ani -m  44 1 avm cam1
ani -i
ani -a ebas -0.004
ani -a ecorp  0.000
ani -a ebrac2  0.001
ani -a ebrac1 -0.003
ani -a egar2  0.011
ani -a egar  0.009
ani -a ehand1  0.020
ani -m  45 1 avm cam1
end
```

Bibliografia

[ANG85] ANGELES, J. "On the Numerical Solution for the Inverse Kinematic Problem", *The International Journal of Robotics Research*, Vol.4, n.2, 1985.

[ASA85] ASADA, H. e SLOTINE, J.J.E. - *"Robot Analysis and Control"*, Cambridge, novembro de 1985.

[ASE87] *ASEA ROBOTICS* - "Quaternions or How to Twist a Robot", outubro de 1987.

[BAL84] BALESTRINO, A.; MARIA, G. e SCIAVICCO, L. - "Robust Control of Robotic Manipulators", *Proc. 9th IFAC world Congress*, Hungria, Julho de 1984.

[BEN74] BEN-ISRAEL, A. e GREVILLE, T.N.E. - *"Generalized Inverses: Theory and Applications"*, New York:Wiley, 1974.

[BOT79] BOTTEMA, O. e ROTH, B. - *Theoretical Kinematics*, North-Holland Publishing Comp., Amsterdam, 1979.

[BOU71] BOULLION, T.L. e ODELL, P.L. - *" Generalized Inverse Matrices"*, Wiley-Interscience, 1971.

[BUR88] BURDICK, J.W.IV - *"Kinematic Analysis and Design of Redundant Robot Manipulators"*, tese de doutorado, Stanford University, 1988.

[CAB91] CABRAL, E.L.L.; MATONE, R. e CIPPARRONE, F.A.M. - "Um Método Eficiente para Computação da Cinemática Inversa de Manipuladores de Cadeia Aberta Utilizando 'Impedance Control' ", *XI COBEM*, Brasil, 1991 (a ser publicado).

[CAR69] CARNAHAN, B.; LUTHER, H.A. e WILKES, J.O. - *"Applied Numerical Methods"*, John Wiley Sons, 1969.

[CHA88] CHAN, S.K. e LAWRENCE, P.D. - "General Inverse Kinematics with the Error Damped PseudoInverse", *Proc. IEEE Conf. on Robotics and Automation*, 1988.

[CHE88] CHEVALLEREAU, C. e KHALIL, W. - "A New Method for the Solution of the Inverse Kinematics of Redundant Robots", *Proc. IEEE Conf. on Robotics and Automation*, 1988.

[CHG89] CHENG, H. e GUPTA, K.C. - "An Historical Note on Finite Rotations", *ASME Transactions Journal of Applied Mechanics*, 56 (1), 1989.

[CHE89] CHENG, H. - *"Computational Aspects of Robot Manipulation"*, tese de doutorado, University of Illinois at Chicago, 1989.

[DAS88] DAS, H.; SLOTINE, E. e SHERIDAN, T.B. - "Inverse Kinematic Algorithms for Redundant Systems", *Proc. IEEE Conf. on Robotics and Automation*, 1988.

[DUB88] DUBEY, R. e LUH, J.Y.S. - "Redundant Robot Control Using Task Based Performance Measures", *Journal of Robotic Systems*, 5(5), 409-432, 1988.

[FEA83] FEATHERSTONE, R. - "Position and Velocity Transformations Between Robot End-Effector Coordinates ans Joint Angles", *The International Journal of Robotics Research*, Vol.2, n.2, agosto de 1983.

[FU87] FU, K.S.; GONZALEZ, R.C. e LEE, C.S.G. - *"Robotics Control, Sensing, Vision and Intelligence"*, McGraw-Hill, 1987.

[GOR84] GORLA, B. e RENAUD, M. - *"Modeles des Robots Manipulateurs, Application à Leur Commande"*, Automatisation Production, 1984.

[GRE59] GREVILLE, T.N.E. - "The Pseudoinverse of a Rectangular or Singular Matrix and its Application to the Solution of Systems of Linear Equations", S.I.A.M. Review, Vol.I, 1959.

[GUP85] GUPTA, K.C. e KAZEROUNIAN, S.M.K. - "Improved Numerical Solutions of Inverse Kinematics of Robots", *International Conf. on Robotics and Automation*, St. Louis, Março de 1985.

[HAN83] HANSEN, J.A.; GUPTA K.C. e KAZEROUNIAN, S.M.K. - "Generation and Evaluation of the Workspace of Manipulators", *The International Journal of Robotics Research*, Vol.2, n.3, 1983.

[HAR64] HARTENBERG, R.S. e DENAVIT, J. - *"Kinematic Synthesis of linkages"*, McGraw-Hill, 1964.

[HIL88] HILLER, M. e WOERNIE, C. - "The characteristic pair of joints - An effective approach for the inverse kinematic problem of robots", *Proc. IEEE Conf. on Robotics and Automation*, 1988.

[1HO85] HOGAN, N. - "Impedance Control: An Approach to Manipulation : Part I - Theory", *Trans. ASME, Journal of Dynamics Systems, Measurement, and Control*, Vol.107, março de 1985.

[2HO85] HOGAN, N. - "Impedance Control: An Approach to Manipulation : Part II - Implementation", *Trans. ASME, Journal of Dynamics Systems, Measurement, and Control*, Vol.107, março de 1985.

[3HO85] HOGAN, N. - "Impedance Control: An Approach to Manipulation : Part III - Applications", *Trans. ASME, Journal of Dynamics Systems, Measurement, and Control*, Vol.107, março de 1985.

[KAR89] KARMANOV, G.V. - *"Mathematical Programming"*, MIR Publishers, 1989.

[KLE83] KLEIN, C.A. e HUANG, C.H. - "Review of Pseudo-Inverse Control for Use with Kinematically Redundant Manipulators", *IEEE Transactions on Systems, Man, and Cybernetics*, Vol.SMC-13, N.3, março/abril de 1983.

[LAS70] LASDON, L.S. - *"Optimization Theory for Large Systems"*, Macmillan Series in Operations Research, 1970.

[LIE77] LIÉGEOIS, A. - "Automatic Supervisory Control of the Configuration and Behavior of Multibody Mechanisms", *IEEE Transactions on Systems, Man, and Cybernetics*, Vol.SMC-7, N.12, dezembro de 1977.

[MAR85] MARIA, G. e MARINO, R. - "A discrete inverse kinematic problem for robotic manipulators", *Proc. 2nd Int. Conf. on Advanced Robotics*, setembro de 1985.

[MAT91] MATONE, R. e MOSCATO, L.A. - "Métodos para Cinemática Inversa de Robôs Manipuladores Baseados na Transposta do Jacobiano", Monografia n.65/91, Depto. de Eng. Mecânica, EPUSP, setembro de 1991.

[MAT91] MATONE, R.; CABRAL, E.L. e MOSCATO, L.A. - "Um Estudo sobre Parâmetros para Descrição da Orientação do Efetuador de Robôs Manipuladores", 9. CBA, Brasil (em preparação).

[MAT91] MATONE, R.; MOSCATO, L.A. e CABRAL, E.L. - "Uma Adaptação Eficiente no Método de Newton-Raphson para Cinemática Inversa de Robôs Manipuladores", 9. CBA, Brasil (em preparação).

[MAY88] MAYORGA, R.V. e WONG, A.K.C. - "A singularities avoidance approach for the optimal local path generation of redundant manipulators", *Proc. IEEE Conf. on Robotics and Automation*, 1988.

[MOL91] MOLINARO, L.F. - *"Sistema de Auxílio ao Ensino de Técnicas de Modelamento e Planejamento Fora-de-Linha de Robôs"*, tese de doutorado, EPUSP, 1991.

[NOV90] NOVAKOVIC, Z.R. e NEMEC, B. - "A Solution of the Inverse Kinematics Problem Using Sliding Mode", *IEEE Transactions on Robotics and Automation*, Abril 1990.

[PA81] PAUL, R.P.; SHIMANO, B. e MAYER G.E. - "Kinematic Control Equations for Simple Manipulators", *IEEE Transactions on Systems, Man, and Cybernetics*, Vol.SMC-11, N.6, junho de 1981.

[PAU81] PAUL, R.P. - *"Robot Manipulator: Mathematics, Programming and Control"*, MIT Press, Cambridge, 1981.

[PIE68] PIEPER, D.L. - *"The Kinematics of Manipulators under Computer Control"*, tese de doutorado, Stanford University, 1968.

[PRE86] PRESS, W.H.; FLANNERY, B.P.; TEULKOLSKY, S.A. e VETTERLING, W.T. - *"Numerical Recipes : The Art of Scientific Computing"*, Cambridge, 1986.

[RAI81] RAIRBERT, M.H. e CRAIG J.J. - "Hybrid Position/Force Control of Manipulators", *Transactions of the ASME*, junho de 1981.

[ROT76] ROTH, B. - "Performance Evaluation of Manipulators from a Kinematic Viewpoint", *NBS* Special Publication, 1976.

[SCI86] SCIAVICCO, L. e SICILIANO, B. - "Coordinate Transformation: A Solution Algorithm for One Class of Robot", *IEEE Transactions on Systems, Man, and Cybernetics*, Vol.SMC-16, N.4, julho/agosto de 1986.

[SCI87] SCIAVICCO, L. e SICILIANO, B. - "A Dynamic Solution to the Inverse Kinematic Problem for Redundant Manipulators", *Proc. 1987 IEEE Int. Conf. on Robotics and Automation*, março/april 1987.

[SCI88] SCIAVICCO, L. e SICILIANO, B. - "A Solution Algorithm to the Inverse Kinematic Problem for Redundant Manipulators", *IEEE Journal of Robotics and Automation*, Vol.4, n.4, agosto de 1988.

[SIM89] SIMEON, T. - *"Generation Automatique de Trajectoires Sans Collision et Planification de Taches de Manipulation en Robotique"*, tese de doutorado, Universite Paul Sabatier de Toulose, 1989.

[STR86] STRANG, G. - *"Introdution to Applied Mathematics"*, Wellesley-Cambridge Press, Massachusetts, 1986.

[TSU91] TSUZUKI, M. e MIYAGI, P.E. - "Sistema para Modelagem de Sólidos", *SOBRACOM*, São Paulo, Brasil, 1991.

[UIC64] UICKER, J.J.; DENAVIT, J. e HARTEMBERG, R.S. - "An iterative Method for the Displacement Analysis of Spacial Mechanisms", *Journal of Applied Mechanics*, Vol. 32, Trans. ASME, Vol. 86, 1964.

[WAM86] WAMPLER, C.W. - "Manipulator Inverse Kinematic Solutions Based on Vector Formulations and Damped Least-Squares Methods", *IEEE Transactions on Systems, Man, and Cybernetics*, Vol.SMC-16, N.1, janeiro/fevereiro de 1986.

[WHI72] WHITNEY, D.E. - "The Mathematics of Coordinated Control of Prosthetic Arms and Manipulators", *Trans. ASME, J. Dynamic Systems, Measurement and Control*, Vol.122, 1972.