4as JORNADAS LUSO-ESPANHOLAS DE ENGENHARIA ELECTROTÉCNICA

ACTAS
Volume 3

PORTO, 6 - 8 JULHO 1995

FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO
DEPARTAMENTO DE ENGENHARIA ELECTROTÉCNICA E DE COMPUTADORES

4 Jornadas Luso-Espanholas de Engenharia Electrotécnica
6-8 de Julho de 1995, Porto, Portugal

PLANEAMENTO DE TRAJECTÓRIAS PARA ROBOT INDUSTRIAL

A. Mendes Lopes[1] e J.A. Tenreiro Machado[2]
[1]Faculdade de Engenharia da Universidade do Porto, Dep. Eng. Mecânica e Gestão Industrial, 4099 Porto Codex, Portugal
[2]Faculdade de Engenharia da Universidade do Porto, Dep. Eng. Electrotécnica e de Computadores, 4099 Porto Codex, Portugal

Resumo

Num sistema robótico, o planeamento de trajectórias é responsável pela geração (em cada intervalo de amostragem) de referências de posição, de velocidade e de aceleração para cada junta. Consideram-se duas abordagens distintas ao planeamento de trajectórias. A primeira requer a especificação de um conjunto de pontos e de um conjunto de condições a serem obedecidas. Neste caso, para executar um movimento entre os pontos definidos, é suficiente interpolar uma função que satisfaça as condições pré-estabelecidas. A segunda abordagem requer a especificação da "curva" que o órgão terminal deve descrever, sendo pois necessário gerar uma sequência de pontos que aproxime a "curva" desejada. Neste artigo são estudados alguns algoritmos para a geração de trajectórias sob o ponto de vista cinemático. Assim, são apresentadas diversas formas de especificar uma trajectória, são estudados algoritmos de interpolação de trajectórias no espaço operacional ou no espaço das juntas e, por último, compara-se o desempenho desses algoritmos para um robot industrial.

Palavras-chave: Trajectórias, interpolação linear, "splines" cúbicas, "splines" trigonométricas.

1. Introdução

Para que um manipulador possa executar uma tarefa tem que se movimentar no espaço, assumindo uma sequência de configurações (trajectória) ao longo do tempo. O planeamento de trajectórias pode ser visto como um bloco que aceita um conjunto de variáveis indicando as condições a que deve obedecer a trajectória, devolvendo, em cada intervalo de amostragem, os valores de posição, velocidade e aceleração para cada junta. Esses valores são posteriormente usados como referências no bloco de controlo, responsável pelo seguimento da trajectória ("tracking"). São comuns duas abordagens distintas ao planeamento de trajectórias. A primeira requer apenas a especificação de um conjunto de pontos (onde o órgão terminal do robot deve passar em determinados instantes de tempo) e de um conjunto de condições a serem obedecidas (por exemplo, continuidade e "suavidade" na posição e nas suas derivadas). Deste modo, o movimento entre os pontos definidos pode ser um qualquer, sendo suficiente interpolar uma função que satisfaça as condições pré-estabelecidas. A segunda abordagem requer a especificação da "curva" que o órgão terminal deve descrever (por exemplo deslocar-se em linha recta). Neste caso, é necessário gerar uma sequência de pontos que aproxime a "curva" desejada.

Neste trabalho são revistos alguns algoritmos para a geração de trajectórias, com base no modelo cinemático do manipulador. Nesta ordem de ideias, na secção 2, são apresentadas diversas formas alternativas de especificar uma trajectória. De seguida, são estudados alguns algoritmos usados na

geração de trajectórias no espaço operacional (secção 3) e algoritmos de geração de trajectórias no espaço das juntas (secção 4). Assim, compara-se o desempenho desses algoritmos para o robot industrial TI ER 6000 e por último, na secção 5, apontam-se as principais conclusões que decorrem deste estudo.

2. Definição de uma Trajectória

Uma trajectória pode ser definida a partir de uma sequência de pontos $\mathbf{x}_i$, pelos quais o órgão terminal do robot deve passar em determinados instantes de tempo t_i, $i = 1,...,k$. Esses pontos representam a posição e orientação do órgão terminal e podem ser expressos no espaço operacional em vários tipos de coordenadas. Os instantes t_i são, normalmente, escolhidos a partir de um compromisso entre os valores máximos estimados para as velocidades e para as acelerações das juntas e os requisitos impostos pela tarefa a executar. Depois de definidos os pontos $\mathbf{x}_i$ e os instantes de tempo t_i, são usados algoritmos de interpolação que geram pontos intermédios e que determinam o tipo de movimento (i.e. o caminho descrito pelo órgão terminal, a sua velocidade e a sua aceleração). Normalmente a capacidade de processamento determina o espaço onde é feita a interpolação: no espaço das juntas, no espaço operacional ou simultaneamente nos dois espaços, e ainda se a trajectória é pré-calculada ou se é gerada em tempo real.

3. Interpolação no espaço operacional

Uma trajectória ligando vários pontos $\mathbf{x}_i$ ($i = 1,...,k$, $k \geq 2$) por segmentos de recta, pode ser conseguida usando uma interpolação linear entre cada dois pontos consecutivos. No entanto, tal procedimento implica que o órgão terminal do robot tenha que parar em cada ponto de transição entre dois segmentos (porque a transição entre dois segmentos implica descontinuidades na velocidade e, por conseguinte, exige acelerações infinitas). Caso não seja necessário passar exactamente nos pontos, esse problema pode ser resolvido com um algoritmo de interpolação capaz de "suavizar" os cantos da curva, na vizinhança dos pontos de transição. O algoritmo apresentado é capaz de gerar uma trajectória contínua na posição e nas suas duas primeiras derivadas para qualquer instante $t \in [0, T]$, em que T é a duração da trajectória [2-3].

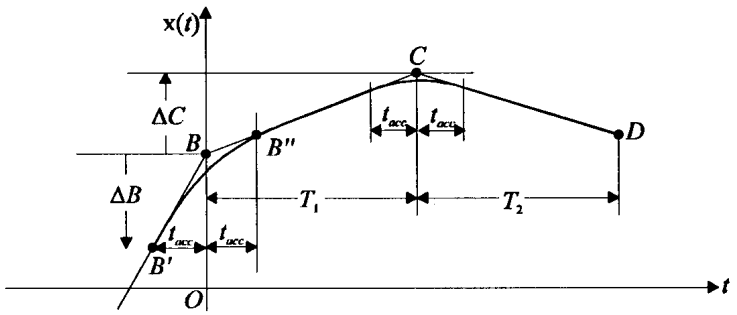

Figura 1 - Transição entre dois segmentos.

Para tal é necessário começar a preparar a transição um tempo t_{acc} (ponto B') antes do fim do segmento corrente e completá-la um tempo t_{acc} (ponto B'') depois de começar o novo segmento. Na figura 1 representa-se parte de uma trajectória em que o manipulador acabou de passar no ponto B' a caminho do ponto B. No instante $-t_{acc}$ começa a dar-se a transição para o segmento BC. O tempo de duração da transição entre os dois segmentos é $2t_{acc}$ que deve permitir, caso necessário, acelerar desde a máxima velocidade negativa até à máxima velocidade positiva ou *vice-versa*. Usando uma função polinomial para interpolar na vizinhança do ponto de transição verifica-se que existem seis condições fronteira. No entanto, devido à simetria da região de transição, é suficiente interpolar um polinómio de quarta ordem. Assim, para o intervalo $-t_{acc} \leq t \leq t_{acc}$ a posição, a velocidade e a aceleração são dadas por:

$$x(t) = \left[\left(\Delta C \frac{t_{acc}}{T_1} + \Delta B\right)(2-h)h^2 - 2\Delta B\right]h + B + \Delta B \qquad (1)$$

$$\dot{x}(t) = \left[\left(\Delta C \frac{t_{acc}}{T_1} + \Delta B\right)(1.5 - h)2h^2 - \Delta B\right]\frac{1}{t_{acc}} \tag{2}$$

$$\ddot{x}(t) = \left(\Delta C \frac{t_{acc}}{T_1} + \Delta B\right)(1 - h)\frac{3h}{t_{acc}^2} \tag{3}$$

$$h = \frac{t + t_{acc}}{2t_{acc}},\ -t_{acc} \le t \le t_{acc},\ \Delta C = C - B,\ \Delta B = B' - B \tag{4}$$

Fora da região de transição a velocidade é constante:

$$x(t) = \Delta C\, h + B,\ \dot{x}(t) = \frac{\Delta C}{T_1},\ \ddot{x}(t) = 0,\ h = \frac{t}{T_1} \tag{5}$$

O algoritmo proposto consegue "suavizar" a trajectória na proximidade dos pontos de transição entre segmentos de modo a serem exigidas acelerações possíveis de cumprir pelos actuadores. No entanto, ainda existem dois pontos onde a velocidade é descontínua e a aceleração infinita: os pontos inicial e final da trajectória. A solução para este problema consiste em repetir o primeiro ponto um tempo t_{acc} antes e repetir o último ponto um tempo t_{acc} depois dos, respectivamente, primeiro e último pontos especificados. A trajectória passa assim a ter mais dois pontos de transição (que são tratados como os restantes) sendo também um tempo $2t_{acc}$ mais longa.

De seguida apresentam-se alguns resultados obtidos para o robot TI ER 6000 (tipo 6R). A trajectória é definida a partir de quatro pontos (que definem um quadrado no plano *YZ* no espaço operacional), de tal forma que o manipulador deve demorar 2 seg a percorrer cada lado. Na figura 2 mostram-se as trajectórias no espaço operacional geradas para diferentes tempos de aceleração/desaceleração t_{acc}. Repare-se que quanto menor o tempo t_{acc} menores são os erros cometidos ao "suavizar"a curva. Em contrapartida, a velocidade e a aceleração exigida aos actuadores vai sendo maior. No caso limite, em que $t_{acc} = 0$ seg, a aceleração requerida para transitar entre segmentos é infinita (figura 3) [1].

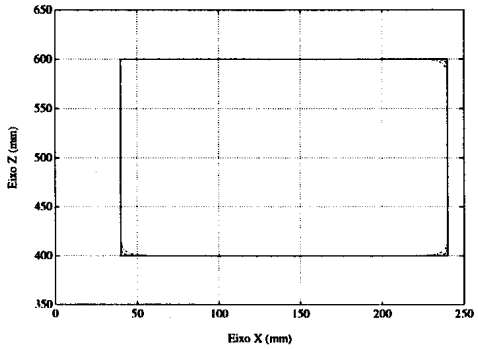

Figura 2 - Trajectórias descritas no espaço operacional para vários valores de t_{acc}.
— t_{acc} = 0 seg; --- t_{acc} = 0.2 seg; -·-· t_{acc} = 0.1 seg

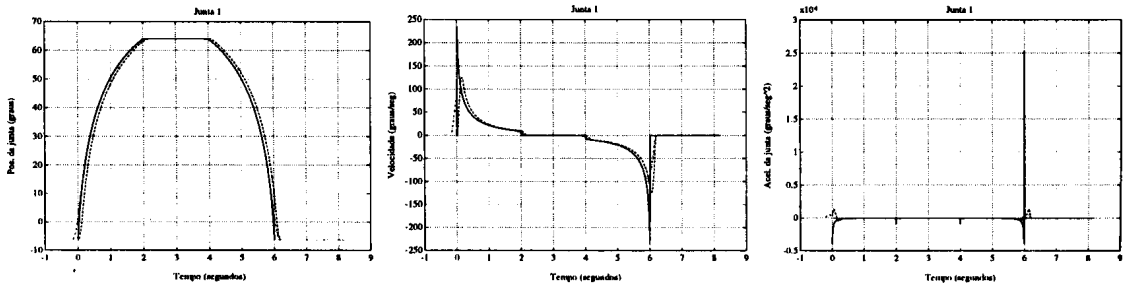

Figura 3 - Evolução temporal da posição, velocidade e aceleração da junta 1 para vários valores de t_{acc}.
— t_{acc} = 0 seg; --- t_{acc} = 0.2 seg; -·-· t_{acc} = 0.1 seg

4. Interpolação no Espaço das Juntas

A interpolação no espaço operacional requer, em cada intervalo de amostragem, a resolução do problema da cinemática inversa. Esta operação, devido ao seu peso computacional, pode tornar impossível a geração de trajectórias em tempo real. A interpolação simultânea no espaço operacional e no espaço das juntas tem a vantagem de poder ser implementada como um algoritmo descentralizado (figura 4) [1].

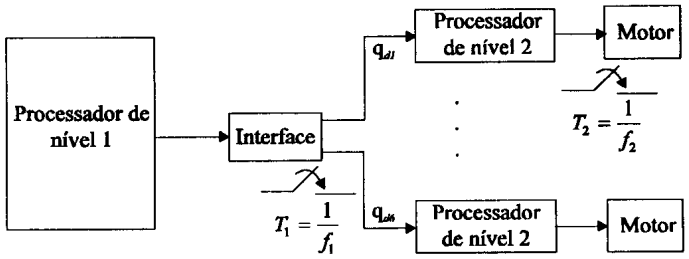

Figura 4 - Algoritmo descentralizado de geração de trajectórias.

Neste caso o processador de nível 1 faz uma interpolação no espaço operacional, gerando um novo vector $\mathbf{q}_d$ (no espaço das juntas, por resolução do problema da cinemática inversa) em cada instante T_1. Os processadores de nível 2 (um por cada junta) fazem uma interpolação no espaço das juntas entre cada dois pontos $\mathbf{q}_d$ consecutivos, gerando referências em cada intervalo de amostragem T_2. A vantagem deste procedimento é a de diminuir o peso computacional devido à redução do número de vezes em que é efectuado o cálculo da cinemática inversa. Em contrapartida as trajectórias geradas desviam-se das trajectórias requeridas. Como exemplo veja-se a trajectória definida na secção 3, interpolada em dois estágios: no primeiro (espaço operacional) a uma frequência de f_1 = 20 Hz (40 pontos por cada lado), f_1 = 1 Hz (2 pontos por cada lado) e f_1 = 0.5 Hz (uma interpolação apenas no espaço das juntas). A frequência no segundo estágio (espaço das juntas) é de f_2 = 100 Hz em todos os casos (Figs. 5 e 6).

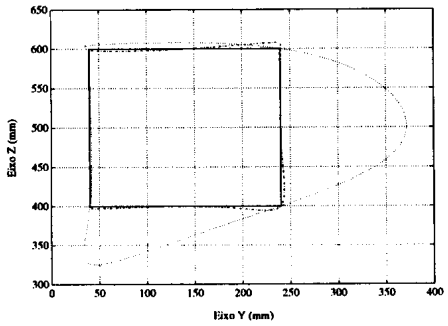

Figura 5 - Trajectórias no espaço operacional para um algoritmo de interpolação misto.
— Trajectória ideal; --- Primeira interpolação a f_1 = 20 Hz; -·- Primeira interpolação a f_1 = 1 Hz; ···· Só com interpolação no espaço das juntas.

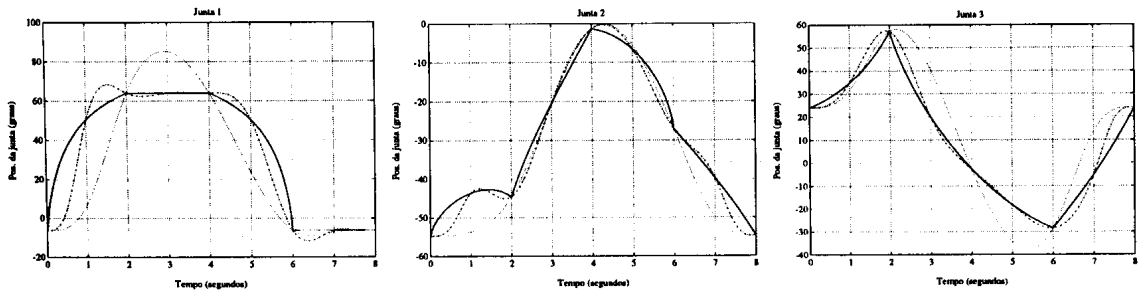

Figura 6 - Trajectórias para as juntas 1, 2 e 3 geradas por um algoritmo de interpolação misto.
— Trajectória ideal; --- Primeira interpolação a f_1 = 20 Hz; -·- Primeira interpolação a f_1 = 1 Hz; ···· Só com interpolação no espaço das juntas.

Considere-se agora que os pontos $\mathbf{x}_i$ pertencentes ao espaço operacional são convertidos para $\mathbf{q}_i$ pertencentes ao espaço das juntas. Então, os pontos intermédios entre $\mathbf{x}_i$ e $\mathbf{x}_{i+1}$ podem ser calculados usando uma interpolação no espaço das juntas.

4.1. Método I: Interpolação Linear

O algoritmo de interpolação no espaço operacional [equações (1) a (5)] pode também ser usado para fazer uma interpolação no espaço das juntas.

4.2. Método II: "Splines" Cúbicas

O número de condições fronteira que podem ser impostas com uma função polinomial depende do seu grau. Os polinómios de terceiro grau possuem quatro coeficientes, pelo que, por cada ponto interior é possível satisfazer quatro restrições: duas para cumprir a posição e mais duas para garantir a continuidade na velocidade e na aceleração. Para os primeiro e último segmentos é necessário cumprir a posição, a velocidade e a aceleração nos pontos extremos e ainda garantir continuidade na velocidade e na aceleração nos segundo e penúltimo pontos. Para não aumentar a ordem dos polinómios, são acrescentados dois pontos: um entre o primeiro e o segundo pontos e outro entre o penúltimo e o último.

Em geral, um conjunto de pontos para interpolar é dado sob a forma $(q_{11}, q_{21},...,q_{N1})$, $(q_{12}, q_{22},...,q_{N2})$, ... ,$(q_{1n}, q_{2n},...,q_{Nn})$, onde q_{ji} é a posição angular da junta j no ponto i. A interpolação é feita para cada junta, para todos os pontos, $q_{j1}, q_{j2},...,q_{jn}$. Seja $t_1 < t_2 < t_3 < ... < t_{n-2} < t_{n-1} < t_n$ uma sequência de instantes no tempo. Nos instantes inicial $t = t_1$ e final $t = t_n$, são especificadas a posição, a velocidade e a aceleração, respectivamente, q_1, v_1, a_1, e q_n, v_n, a_n. São ainda especificadas as posições q_k nos instantes t_k, para $k = 3, 4,...,n-2$. Os valores de q_2 e q_{n-1} não são especificados constituindo os dois pontos auxiliares necessários para a resolução do problema. Seja $q_i(t)$ uma função polinomial cúbica definida no intervalo $t \in [t_i, t_{i+1}]$. É necessário encontrar as funções $q_i(t)$, $i = 1, 2,...,n-1$, de modo a satisfazer as restrições impostas. Dado que $q_i(t)$ é cúbica, a sua segunda derivada em ordem ao tempo, $\ddot{q}_i(t)$ é uma função linear. Assim, $\ddot{q}_i(t)$ pode ser expressa como [5-7]:

$$\ddot{q}_i(t) = \frac{t_{i+1} - t}{h_i}\ddot{q}_i(t_i) + \frac{(t - t_i)}{h_i}\ddot{q}_i(t_{i+1}),\ h_i = t_{i+1} - t_i,\ i = 1, 2,...,n-1 \tag{6}$$

Integrando $\ddot{q}_i(t)$ duas vezes e impondo as condições $q_i(t_i) = q_i$ e $q_i(t_{i+1}) = q_{i+1}$ obtêm-se as funções de interpolação:

$$q_i = \frac{\ddot{q}_i(t_i)}{6h_i}(t_{i+1} - t)^3 + \frac{\ddot{q}_i(t_{i+1})}{6h_i}(t - t_i)^3 + \left[\frac{q_{i+1}}{h_i} - \frac{h_i\,\ddot{q}_i(t_{i+1})}{6}\right](t - t_i) + \left[\frac{q_i}{h_i} - \frac{h_i\,\ddot{q}_i(t_i)}{6}\right](t_{i+1} - t) \tag{7}$$

Para $i = 1, 2,...,n-1$, $q_i(t)$ fica determinada se forem conhecidos $\ddot{q}_i(t_i)$ e $\ddot{q}_i(t_{i+1})$, o que é conseguido resolvendo um sistema de equações lineares.

4.3. "Splines" Trigonométricas

Uma "spline" trigonométrica de ordem m é por definição uma função $q(t)$ com as propriedades: $q(t)$ é periódica com período 2π; $q(t)$ satisfaz as condições de interpolação $q(t_j) = q_j$, $j = 0,...,n$ e $t_j \in [0, 2\pi[$; a função $q(t)$ tem $2m$ restrições em cada um dos n arcos fechados $[t_{i-1}, t_i]$, $i = 1,...,n$. Assim, $q_i(t)$ vem [4]:

$$q_i(t) = a_{i,0} + \sum_{k=1}^{m-1}(a_{i,k}\cos kt + b_{i,k}\,\mathrm{sen}\,kt) + a_{i,m}\,\mathrm{sen}\,m(t - \gamma_i),\ \gamma_i = \frac{(\tau_{i,1} + \tau_{i,2} + \cdots + \tau_{i,2m})}{2m} \tag{8}$$

onde $\tau_{i,j}$ são os $2m$ valores de t onde o segmento i tem aplicada uma condição fronteira. O valor resultante para γ_i garante a existência e unicidade de $q(t)$. A equação (8) mostra que existem exactamente $2m$ coeficientes para cada segmento de "spline" trigonométrica, isto é, mostra que podem ser impostas $2m$ condições fronteira em cada segmento. Essas condições podem ser escolhidas como sendo $q^{(r)}(t_i) = q_i^{(r)}$, $r = 0,...,m-1$ e $i = 0,...,n$, ou seja, podem ser fixados os valores da função e suas derivadas até à ordem $m-1$. Para determinar os coeficientes de cada segmento de "spline", é necessário resolver um sistema de $2m$ equações lineares. Se for pretendida a continuidade da "spline" trigonométrica e das suas $m-1$ derivadas, terá que fazer-se $q^{(r)}(t_i^-) = q^{(r)}(t_i^+)$. No entanto, se em vez de requerer apenas a continuidade em $q^{(r)}(t)$, se impuserem os valores de $q^{(r)}(t_i)$, então a determinação dos coeficientes vem desacoplada para cada segmento. Este procedimento reduz bastante o esforço computacional, podendo continuar a exigir-se $q^{(r)}(t_i^-) = q^{(r)}(t_i^+)$, desde que se façam ambas as derivadas iguais à mesma constante. Essas constantes podem ser escolhidas de modo a satisfazer um critério de optimização ou podem ser escolhidas heuristicamente.

4.3.1. Método III: "Splines" Trigonométricas de Quarta Ordem

Para interpolar uma "spline" trigonométrica pelos $n+1$ pontos, têm que ser ligados entre si n segmentos $q_i(t)$, $i = 1,...,n$, de quarta ordem, definidos no intervalo $[t_{i-1}, t_i]$. Sendo $q_i(t)$ de quarta ordem, existem oito coeficientes para determinar. As oito condições usadas para a sua determinação são os valores da função e suas primeiras três derivadas temporais nos pontos t_{i-1} e t_i.. A determinação dos oito coeficientes para cada segmento $q_i(t)$ implica a inversão de uma matriz $\mathbf{A}_i$ de dimensão 8×8. Como essa matriz apenas depende de dois parâmetros (os extremos t_{i-1} e t_i do intervalo onde $q_i(t)$ está definida), esse intervalo pode ser normalizado de modo a torná-lo fixo. Deste modo, a inversão de $\mathbf{A}_i$ não necessita de ser feita em tempo real, podendo ser feita *a priori*. Mostra-se que $t_{i-1} = 0$ e $t_i = \pi/4$ conduzem, em geral, a trajectórias mais "suaves". Assim, da equação (8) vem $\gamma_i = \pi/8$ ($i = 1,...,n$) resultando [4]:

$$q_i(t) = a_{i,0} + \sum_{k=1}^{3} (a_{i,k} \cos kt + b_{i,k} \operatorname{sen} kt) + a_{i,4} \cos 4t \,,\; 0 \le t \le \pi/4 \tag{9}$$

A determinação dos $8n$ coeficientes que descrevem completamente a "spline" trigonométrica (e as suas derivadas) que passa pelos pontos $\mathbf{q}_i$, $i = 0,...,n$ e que apresenta velocidade, aceleração e "jerk" contínuos em toda a trajectória é feita recorrendo à resolução de um sistema de equações lineares (equação 10).

$$\left(a_{i,0}\; a_{i,1}\; b_{i,1}\; a_{i,2}\; b_{i,2}\; a_{i,3}\; b_{i,3}\right)^T = \mathbf{A}_i^{-1}\left(q_{i-1}\; q_i\; \dot{q}_{i-1}\; \dot{q}_i\; \ddot{q}_{i-1}\; \ddot{q}_i\; \dddot{q}_{i-1}\; \dddot{q}_i\right)^T,\; i = 1, 2,...,n \tag{10}$$

4.3.2. Método IV: "Splines" Trigonométricas de Terceira Ordem

Para reduzir o esforço computacional do algoritmo de planeamento de trajectórias pode baixar-se a ordem das "splines". As "splines" trigonométricas de terceira ordem podem garantir continuidade da posição e das suas primeiras duas derivadas. À semelhança do que acontece com as "splines" cúbicas passa a não ser requerida a continuidade na terceira derivada ("jerk").

Da equação (8) a "spline" trigonométrica de terceira ordem é dada por:

$$q_i(t) = a_{i,0} + \sum_{k=1}^{2} (a_{i,k} \cos kt + b_{i,k} \operatorname{sen} kt) + a_{i,3} \operatorname{sen} 3(t - \pi/8) \,,\; 0 \le t \le \pi/4 \tag{11}$$

Os coeficientes de uma "spline" de terceira ordem podem ser determinados de modo semelhante ao referido na sub-secção anterior.

4.4. Comparação dos Algoritmos de Interpolação no Espaço das Juntas

Para avaliar o desempenho dos algoritmos apresentados anteriormente, são mostrados os resultados obtidos para a trajectória da tabela I, definida por oito pontos (posições angulares) igualmente espaçados no tempo. O tempo total de execução é de 32 segundos.

Ponto	1	2	3	4	5	6	7	8
Valor (graus)	10	60	75	130	110	100	-10	-50

Tabela I - Pontos de definição de uma trajectória.

Para os métodos III e IV foram impostos valores nulos para a velocidade, para a aceleração e para o "jerk" nos pontos extremos (só é necessário especificar o "jerk" no método III), enquanto que, as três primeiras derivadas para os pontos interiores da trajectória normalizada foram calculadas usando um algoritmo heurístico baseado na expansão em série de Taylor das derivadas de q(t) (equação 12):

$$\dot{q}_i = \frac{q_{i+1} - q_{i-1}}{\pi/2}, \ddot{q}_i = \frac{2(q_{i+1} - 2q_i + q_{i-1})}{(\pi/4)^2}, \dddot{q}_i = \frac{4(\ddot{q}_{i+1} - \ddot{q}_{i-1})}{\pi/2} \tag{12}$$

Para o método II foram impostas velocidade e aceleração nulas nos pontos extremos. Nos pontos interiores, dado que apenas se garante a continuidade da velocidade e da aceleração, não é necessário explicitar os seus valores, sendo suficiente especificar a posição. Os pontos adicionais foram colocados, respectivamente, nos pontos médios dos intervalos $[t_1, t_2]$ e $[t_{n-1}, t_n]$. Para o método I, a velocidade e aceleração nulas no início e no fim da trajectória são implícitas a este método.

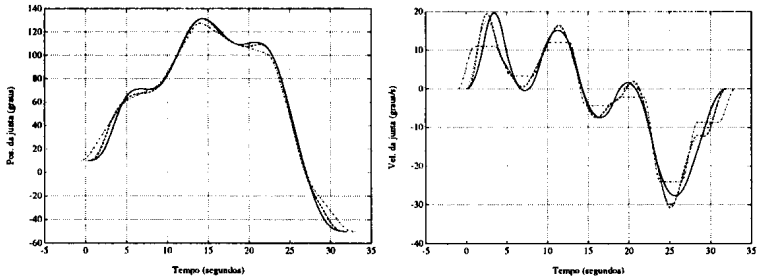

Figura 7a - Posição e velocidade para a trajectória definida na tabela I
-·-· Algoritmo I; — Algoritmo II; --- Algoritmo III; ···· Algoritmo IV.

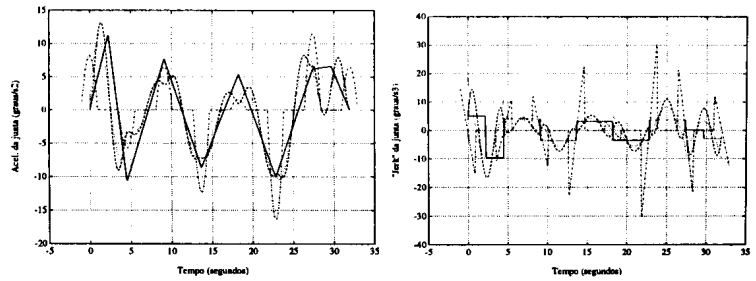

Figura 7b - Aceleração e "Jerk" para a trajectória definida na tabela I.
-·-· Algoritmo I; — Algoritmo II; --- Algoritmo III; ···· Algoritmo IV.

Qualquer dos algoritmos gera trajectórias contínuas e com continuidade nas derivadas até, pelo menos, à segunda ordem. O método I tem a vantagem de não produzir uma sobre-elongação nos pontos de definição da trajectória mas, em contrapartida, exige acelerações elevadas. Ao contrário do que acontece com o método II, os métodos III e IV apresentam uma evolução temporal da aceleração "oscilante" (figura 7).

Muitas vezes requer-se continuidade na posição, na velocidade e na aceleração. A continuidade na quarta derivada ("jerk") minimiza as "vibrações" mas, exige algoritmos de maior peso computacional. Dos algoritmos estudados, apenas o método III garante continuidade no "jerk".

O comando de robots em tempo real exige algoritmos de geração de trajectórias "on-line", isto é, que possam ser usados conhecendo apenas alguns pontos em avanço. Nos métodos I, III e IV basta conhecer um ponto em avanço.

Se as trajectórias forem geradas "on-line" basta guardar os coeficientes necessários para interpolar no próximo segmento. Deste modo, o problema da quantidade de memória necessária só se põe para trajectórias geradas "off-line", uma vez que é necessário gerar toda a trajectória (ou pelo menos os coeficientes das funções) antes de se iniciar o movimento. No caso do método II, para uma trajectória definida à custa de $n-2$ pontos (mais os dois pontos adicionais) são necessários $n-1$ segmentos. Como cada segmento tem quatro coeficientes é necessário guardar $4(n-1)$ coeficientes por cada junta. Se o robot tiver seis juntas são necessários $24(n-1)$ coeficientes.

Os algoritmos de interpolação no espaço das juntas estudados neste capítulo foram programados em Turbo Pascal 7.0 para DOS. Os tempos de computação obtidos num computador com processador 80486 SX a 25 MHz foram os seguintes (para a trajectória definida anteriormente): método I: 0.474 mseg por iteração; método II: 1.16 mseg por iteração; método III: 8.725 mseg por iteração; método IV: 6.17 mseg por iteração.

5. Conclusões

Neste trabalho foram revistos alguns algoritmos de interpolação passíveis de serem usados num sistema robótico na geração de trajectórias. A geração de trajectórias no espaço das juntas tem as vantagens de não requerer o cálculo da cinemática inversa em cada intervalo de amostragem (libertando assim tempo de CPU que pode ser usado noutras tarefas) e não apresentar problemas de cálculo nos pontos singulares. A sua desvantagem advém do facto das trajectórias geradas se afastarem das trajectórias requeridas no espaço operacional. A geração de trajectórias no espaço operacional tem como desvantagens a necessidade de calcular a cinemática inversa em cada intervalo de amostragem e a dependência de métodos capazes de resolver as singularidades. A sua maior vantagem consiste na facilidade de visualização das trajectórias que são geradas. Assim, a geração de trajectórias no espaço operacional tenderá a ser mais atractiva quer devido ao aumento das capacidades de cálculo quer recorrendo à utilização de algoritmos capazes de resolver o problema das singularidades.

Referências

[1] A. Mendes Lopes, "Análise Cinemática e Planeamento de Trajectórias para um Robot Industrial", Dissertação de Mestrado, DEEC-FEUP, Novembro de 1994.

[2] M. Vukobratovic, M. Kircanski, "Scientific Fundamentals of Robotics 3: Kinematics and Trajectory Synthesis of Manipulation Robots", Springer-Verlag, 1986.

[3] Richard P. Paul, "Robot Manipulators: Mathematics, Programming, and Control", The MIT Press series in artificial intelligence", 1982.

[4] Dan Simon, Can Isik, "A trigonometric trajectory generator for robotic arms", International Journal of Control, 1993, vol. 57, N. 3, 505-517.

[5] Chun-Shin Lin, Po-Rong Chang, J. Y. S. Luh, "Formulation and Optimization of Cubic Polynomial Joint Trajectories for Industrial Robots", IEEE Transactions on Automatic Control, Vol AC-28, N. 12, December 1983.

[6] A. A. Goldenberg, D. L. Lawrence, "End Effector Path Generation", Transactions of the ASME, Journal of Dynamic Systems, Measurement, and Control, Vol 108, June 1986.

[7] Chun-Shin Lin, Po-Rong Chang, "Joint Trajectories of Mechanical Manipulators for Cartesian Path Approtimation", IEEE Transactions on Systems, Man, and Cybernetics, Vol. SMC-13, N. 6, November/December 1983.