CONTROLE TOLERANTE A FALHAS DE ROBÔS MANIPULADORES

Marco Henrique Terra[*], Marcel Bergerman[**], Renato Tinós[*], Adriano A. G. Siqueira[*]

[*] *Laboratório de Sistemas Inteligentes*
Departamento de Engenharia Elétrica
Escola de Engenharia de São Carlos
Universidade de São Paulo
Caixa Postal 359
São Carlos SP 13560-970
E-mail: terra@sel.eesc.sc.usp.br

[**] Laboratório de Robótica e Visão
Instituto Nacional de Tecnologia da Informação
Caixa Postal 6162
Campinas SP 13083-970
E-mail: *mbergerman@genius.org.br*

Resumo: Com o crescente uso de robôs manipuladores em locais de difícil acesso ou inóspitos a seres humanos, torna-se cada vez mais fundamental o desenvolvimento de métodos seguros de controle tolerante a falhas para estas máquinas. Este artigo apresenta o trabalho desenvolvido pelos autores nos últimos três anos na criação de métodos de controle de robôs que sejam tolerantes a falhas. A teoria apresentada é validada no ambiente de experimentação existente na EESC/USP, que inclui o robô manipulador UARM II.

Palavras Chaves: Sistemas tolerantes a falhas; Manipuladores robóticos; Detecção de falhas; Diagnóstico de falhas; Isolação de falhas; Controle de robôs, Controle robusto.

Abstract: With the crescent utilization of robot manipulators in hard-to-access or inhospitable sites, the development of reliable fault tolerant control methods for these machines becomes vital. This course presents the work developed by the authors in the last three years on fault tolerant control methods for robot manipulators. The theory presented is validated in the experimental setup available at EESC/USP, which includes the UARM II robot manipulator.

Keywords: Fault-tolerant systems; Robotic manipulators; Fault detection; Fault diagnosis; Fault isolation; Robot control; Robust control.

1 INTRODUÇÃO

É crescente a migração dos sistemas robóticos das indústrias e laboratórios para ambientes pouco estruturados ou perigosos para o ser humano. Robôs são cada vez mais comuns em exploração espacial e do fundo dos oceanos, em plantas nucleares e outros ambientes hostis, em medicina, etc. Em um futuro próximo, robôs serão utilizados como um acessório comum dentro das casas para a execução de tarefas domésticas; algumas pesquisas indicam que em 2010, o número estimado de robôs em domicílios chegará a vários milhões (Dhillon e Fashandi, 1997). Tal evolução exige uma substancial melhora da confiabilidade e segurança dos sistemas robóticos atuais.

Existem diversas fontes de falhas em sistemas robóticos (mecânicas, elétricas, hidráulicas, pneumáticas, de *software*, etc.) e, portanto, falhas em robôs são comuns (Visinsky *et al.*, 1994). De acordo com algumas publicações, o tempo médio entre falhas em robôs industriais é de apenas 500 a 2500 horas (Dhillon e Fashandi, 1997). Em um estudo do Ministério do Trabalho do Japão, 28,7% dos robôs industriais estudados tinha um tempo médio entre falhas de 100 horas ou menos; 60% tinha um tempo médio entre falhas menor que 500 horas (Dhillon, 1991 *apud* em Groom *et al.*, 1999). Desta forma, existem boas razões para se estudar métodos de controle tolerante a falhas em robôs manipuladores.

Este artigo apresenta o trabalho desenvolvido pelos autores nos últimos três anos na criação de métodos de controle de robôs que sejam tolerantes a falhas. O artigo está dividido nas seguintes seções: a seção 2 apresenta a metodologia concebida para a solução do problema em questão; a seção 3 apresenta o subsistema de detecção e isolação de falhas; a seção 4 apresenta o subsistema de controle pós-falha de robôs manipuladores; a seção 5 apresenta o ambiente de simulação e experimentação existente na EESC/USP, utilizado nesta pesquisa, incluindo o robô manipulador UARM II; a seção 6 apresenta um extenso estudo de caso incluindo simulações e

Artigo Submetido em 01/11/00
Revisão em 23/04/01
Aceito sob recomendação do Ed. Consultor Prof. Dr. Liu Hsu

resultados experimentais; e a seção 7 apresenta a conclusão do trabalho.

2 UMA ESTRUTURA DE CONTROLE TOLERANTE A FALHAS PARA ROBÔS MANIPULADORES

Falhas em robôs manipuladores podem originar movimentos descontrolados nos elos, podendo causar sérios danos ao robô e ao ambiente de trabalho. Por exemplo, uma falha no sensor de velocidade da segunda junta de um robô industrial com seis graus de liberdade pode fazer com que o sistema de controle aplique nessa junta um torque alto. Esta ação pode levar o robô a, por exemplo, bater no chão ou em outros objetos do ambiente de trabalho (Visinsky *et al.*, 1994).

Em alguns ambientes, prover os robôs apenas com capacidade de detecção e isolação de falhas (DIF) não é suficiente. Nos robôs em missões espaciais ou operando na manutenção de plataformas marítimas, usinas nucleares ou outros ambientes inóspitos, sistemas que garantam tolerância a falhas são essenciais. Nestes ambientes, enviar seres humanos para eventuais consertos no robô com falha é, na maioria dos casos, inviável. Além disso, dependendo do objetivo e das consequências da falha, o robô precisa rapidamente concluir sua tarefa ou se reconfigurar para ser recolhido.

Sistemas dinâmicos podem ser caracterizados por serem robustos ou reconfiguráveis. Um sistema é dito robusto se retém satisfatoriamente o desempenho na presença de erros de modelagem, ruídos ou falhas. O sistema é dito reconfigurável se a sua estrutura ou seus parâmetros puderem ser alterados em resposta a falhas. Neste caso, a falha deve ser detectada e isolada para posterior modificação do sistema ou de suas leis de controle para a manutenção de um desempenho aceitável (Stengel, 1991).

Muitos sistemas empregam a chamada redundância física (ou de *hardware*) para obter tolerância a falhas. Na redundância física, alguns componentes do sistema de controle (sensores, atuadores e controladores) são duplicados, ou seja, existem dois componentes para o desempenho da mesma função. Os componentes redundantes são aqueles mais sujeitos a falhas ou mais cruciais em relação ao desempenho do sistema.

Uma derivação de tal método é a redundância física em paralelo, na qual dois ou mais conjuntos de sensores, atuadores e controladores, cada qual com capacidade individual satisfatória de controle, são instalados para a execução da mesma tarefa. Um sistema gerenciador compara os sinais de controle para detectar o conjunto faltoso. Com dois sinais redundantes, um sistema votante pode detectar a existência de um conjunto faltoso mas não pode identificá-lo. Com três sinais redundantes, um sistema votante pode detectar e isolar o conjunto faltoso, selecionando, assim, aquele que não deve ser utilizado para o controle da planta.

Redundância física pode proteger o sistema contra falhas nos componentes do sistema de controle, mas não nos componentes da planta. Além disso, em robótica, o uso de redundância física é quase sempre limitado pelo fatores custo, tamanho e potência (Visinsky *et al*., 1994).

Alternativamente, redundância cinemática ou funcional podem ser utilizadas para se conseguir tolerância a falhas. No primeiro caso, o robô é provido com um número maior de graus de liberdade do que o requerido para a realização da sua tarefa. Por exemplo, um robô com sete juntas pode ser utilizado para manipular uma carga em um espaço tridimensional livre de obstáculos (neste caso, seis graus de liberdade são necessários: três para posição e três para orientação). Em caso de falha em uma das juntas, esta é bloqueada e o manipulador ainda assim tem condições de concluir sua tarefa (Lewis e Maciejewski, 1997), (Paredis e Khosla, 1996a). Alguns estudos têm examinado quais são as áreas do espaço de trabalho e as configurações do robô em que as falhas podem ter consequências mais desastrosas (English e Maciejewski, 1998), (Paredis e Khosla, 1996b). Evitando estas áreas e estas configurações, o robô cinematicamente redundante torna-se tolerante a falhas.

Já a redundância funcional utiliza dados funcionais equivalentes para substituir as medidas que foram afetadas por falhas. Assim, por exemplo, se um *encoder* falha, os dados de posição daquela junta podem ser substituídos por valores obtidos pela integração numérica das velocidades medidas pelos tacômetros (Visinsky *et al.*, 1994).

Redundância funcional pode ser aplicada somente para certos tipos de falhas (por exemplo, nos sensores). Já redundância cinemática ou física requer certas características construtivas especiais. Muitas vezes, tais características não podem ser respeitadas devido aos custos ou à limitações estruturais.

Neste artigo é apresentada uma metodologia para controle tolerante a falhas que não exige que o robô seja cinematica ou fisicamente redundante. A tolerância a falhas para manipuladores robóticos com elos seriais constituindo uma cadeia mecânica aberta é conseguida através da reconfiguração do controle após a detecção e isolação da falha. Três módulos são usados para se alcançar a tolerância a falhas: 1) um módulo de DIF utilizando redes neurais artificiais; 2) um módulo para reconfiguração das leis de controle de acordo com a operação do sistema (com ou sem falhas); e 3) um módulo de controle responsável pela aplicação dos torques e pela ativação dos freios nas juntas (Tinós *et al.*, 2000), (Terra and Tinós 2000), (Terra *et al.*, 2000) e (Bergerman *et al.*, 2000).

Neste ponto, é importante que algumas considerações sejam feitas:

- Somente falhas do tipo *junta livre* são consideradas. Este tipo de falha pode ocorrer, por exemplo, por uma perda de fluxo em um atuador hidráulico, por uma perda de potência elétrica em um atuador elétrico ou por uma falha no sistema de transmissão de potência mecânica nas juntas (English and Maciejewski, 1998). Outras falhas podem ser incluídas no futuro; para isso, o sistema DIF deve ser treinado para identificar seus efeitos e um novo sistema de reconfiguração de controle deve ser usado. Por exemplo, se uma falha no sensor de velocidade for detectada, deve-se ignorar seu sinal e estimar a velocidade através das medidas de posição (Visinsky *et al.*, 1994);

- As juntas são equipadas com freios liga-desliga que operam independentemente dos atuadores e com sensores de posição (*encoder*) e velocidade (tacômetro) nas juntas. Tais características são comuns nos manipuladores comerciais;

- Considera-se que as falhas ocorrem uma de cada vez. Uma hipótese mais abrangente pode ser considerada sem maiores problemas se o sistema DIF for treinado para

detectar mais de uma falha ocorrendo no mesmo instante, já que o método de reconfiguração aqui apresentado pode controlar manipuladores com um número qualquer de juntas falhas (Bergerman e Xu, 1997 e 1998);

- É assumido que o planejamento de trajetórias e as características do manipulador são conhecidos *a priori*.

Estas hipóteses são suficientemente gerais para que o sistema de tolerância a falhas seja aplicado em uma variedade muito grande de cenários.

Com as colocações acima, pode-se agora definir o sistema de tolerância a falhas. Considere a dinâmica do manipulador robótico sem falhas

$$\boldsymbol{\tau} = \mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{b}(\mathbf{q}, \dot{\mathbf{q}}) + \mathbf{d}(t) \tag{2.1}$$

sendo **q** o vetor *n* x 1 das posições das juntas, *n* é o número de graus de liberdade (GDLs), **M** representa a matriz *n* x *n* de inércia, **b** representa o vetor *n* x 1 dos torques de Coriolis, centrífugos, gravitacionais e friccionais, e **d** representa o vetor *n* x 1 dos distúrbios não-correlacionados. Desprezando por simplicidade o vetor de distúrbios e usando a equação anterior, tem-se a equação dinâmica do manipulador robótico sem falhas

$$\ddot{\mathbf{q}} = \mathbf{M}(\mathbf{q})^{-1}\left[\boldsymbol{\tau} - \mathbf{b}(\mathbf{q}, \dot{\mathbf{q}})\right] \tag{2.2}$$

Quando ocorre a falha ϕ no manipulador a dinâmica torna-se

$$\ddot{\mathbf{q}} = \mathbf{M}(\mathbf{q})^{-1}\left[\boldsymbol{\tau} - \mathbf{b}(\mathbf{q}, \dot{\mathbf{q}})\right] + \boldsymbol{\phi}(\mathbf{q}, \dot{\mathbf{q}}, \boldsymbol{\tau}) \tag{2.3}$$

sendo $\boldsymbol{\phi}(.)$ o vetor de falhas. O papel do módulo DIF é isolar e classificar o vetor de falhas $\boldsymbol{\phi}(.)$. Ele é descrito em detalhes na Seção 3.

Uma vez detectada e isolada a falha, o manipulador torna-se um sistema subatuado, por possuir menos atuadores operantes do que GDLs. O sistema de controle tolerante a falhas invoca então o módulo de reconfiguração da lei de controle, que determina a ordem em que as juntas serão controladas de forma que o efetuador atinja a pose desejada. Finalmente, o módulo de controle aciona ou desaciona os freios e aplica os torques apropriados em cada junta. O controle pós-falha é descrito em detalhes na Seção 4.

A Figura 1 mostra a metodologia de controle tolerância a falhas.

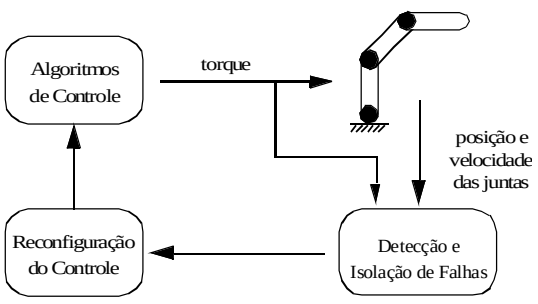


Figura 1. Controle tolerante a falhas de robôs manipuladores.

3 DETECÇÃO E ISOLAÇÃO DE FALHAS

3.1 Introdução

Em geral, os métodos utilizados para DIF em robôs manipuladores empregam redundância analítica. Portanto, utilizam o modelo matemático do robô manipulador. A diferença básica entre os métodos encontra-se na análise dos resíduos. Analisar simplesmente se o resíduo é diferente de zero é inviável para a detecção da falha em sistemas robóticos. Este procedimento leva a um grande número de alarmes falsos, já que os erros de modelagem inerentes ao sistema sem falhas aumentam devido à linearização das equações do sistema e às incertezas dos parâmetros do modelo.

O que acontece tipicamente nos sistemas de DIF para robôs manipuladores é que bandas de detecção (*thresholds*) fixas são estipuladas para mascarar tais incertezas. Geralmente, a banda de detecção é determinada empiricamente, tomando-se cuidado para que seu valor não seja grande o suficiente para esconder as falhas. No entanto, os efeitos de erros de modelagem flutuam dinamicamente conforme o manipulador se move e quando ocorrem falhas. Desta forma, alarmes falsos ainda podem aparecer, pois a banda de detecção fixa não é projetada para todas as situações dinâmicas possíveis. Como exemplo prático deste problema, pode-se citar o sistema DIF utilizado no robô manipulador do Ônibus Espacial Americano na missão STS-49 (maio de 1992). Neste vôo, uma sequência de 13 alarmes falsos foram disparados porque condições operacionais especiais, tais como entradas do controlador manual e carga no manipulador não usuais, não foram levadas em consideração na modelagem do sistema (Visinsky *et al*., 1994). Assim, as bandas de detecção devem ser variáveis para que os efeitos dos erros de modelagem possam ser superados.

Vários métodos foram criados para superar os problemas dos sistemas dinâmicos incertos. Dentre esses, pode-se citar o RMI (*reachable measurement intervals*) desenvolvido por Horak (1988) para sistemas de aeronaves. RMI provê condições para se achar os extremos possíveis das saídas medidas para cada entrada e estado do sistema sob condições livres de falhas, dadas faixas de variações paramétricas nos coeficientes das matrizes das equações dinâmicas. RMI, contudo, é concebido para sistemas com dinâmica linear e requer procedimentos especiais custosos para tratar sistemas dinâmicos não-lineares.

Para robôs manipuladores, um outro método baseado nos conceitos de RMI foi desenvolvido. É o chamado ThMB (*model-based threshold algorithm*) que utiliza a idéia de se encontrar a máxima variância possível devido às faixas dos erros paramétricos em cada iteração (Visinsky *et al*., 1995). Utilizando tais dados, bandas de detecção dependentes dos estados e das entradas são construídas para se alcançar robustez na análise dos resíduos. Este método explora as similaridades entre RMI e redundância analítica.

Um outro enfoque é utilizar técnicas de Inteligência Artificial para criar bandas de detecção adaptativas. Primeiro, os resíduos são gerados através de observadores de estado. A seguir, os resíduos são analisados. Schneider e Frank (1996) utilizaram lógica difusa e Naughton *et al*. (1996) utilizaram um *multilayer perceptron* (MLP) com treinamento por retropropagação para produzir bandas de detecção variáveis. O trabalho de Schneider e Frank preocupa-se basicamente com os erros de modelagem causados pela fricção. Como a fricção é difícil de ser modelada em robôs manipuladores, construiu-se um conjunto de regras

difusas para variar as bandas de detecção em função dos dados de velocidade e aceleração nas juntas. Seu método produz, assim, bandas de detecção dinâmicas. Neste caso, o conhecimento de um especialista é necessário para a confecção das regras. Já Naughton *et al*. utilizaram um MLP com treinamento por retropropagação para a classificação dos resíduos. Treinando o MLP com dados do vetor de resíduos como entrada e das falhas como saída desejada, a RNA pôde construir bandas de detecção variáveis. No entanto, o sistema DIF neste caso consegue apenas detectar as falhas em um conjunto de trajetórias limitado (aquele usado no treinamento) apesar de os autores acreditarem que novas trajetórias podem ser utilizadas se o conjunto de treinamento empregado for maior.

Todos os métodos citados acima utilizam o modelo matemático do manipulador. Empregam técnicas robustas para a geração de resíduos (através, por exemplo, de observadores robustos) ou para a análise dos mesmos (através de bandas de detecção adaptativas). Os métodos descritos acima têm as seguintes características: 1) o modelo nominal do sistema é considerado linear e, 2) as falhas são modeladas como entradas aditivas externas (ou seja, dependentes apenas do tempo e não dos estados e das entradas do sistema). Apesar de ser conveniente do ponto de vista analítico o estudo de problemas DIF em estruturas lineares, a dinâmica do manipulador é inerentemente não-linear e a maioria das falhas reais são funções não-lineares dos estados e das entradas (Vemuri e Polycarpou, 1997).

Um enfoque para DIF em robôs manipuladores diferente dos utilizados anteriormente é o desenvolvido por Vemuri e Polycarpou (1997), no qual uma rede neural artificial (RNA) é empregada para mapear o vetor de falhas (veja mais detalhes na próxima subseção). Seu método é interessante pois a falha não é considerada como uma entrada aditiva externa e sim como uma função das variáveis do sistema. A tarefa da RNA é mapear o vetor de falhas, tendo como entradas as velocidades e as posições das juntas. Assim, o sistema pode não só detectar, mas também reproduzir a função da falha. O método utiliza um observador para, baseado no modelo matemático e na função de falha estimada pela RNA, gerar uma estimativa das variáveis medidas. O erro entre o valor estimado e o valor medido é utilizado para o ajuste dos pesos da rede. O trabalho não propõe nenhum esquema para isolação da falha, propondo que algum método de classificação de padrões seja utilizado. O maior problema deste método é que erros de modelagem podem comprometer, durante o treinamento da RNA, o mapeamento do vetor de falhas.

Em (Terra e Tinós, 1998 e 1999), (Tinós *et al*., 1999), (Tinós, 1999) e (Terra e Tinós, 2001) o modelo matemático não é utilizado. Primeiro, um MLP é utilizado para reproduzir a dinâmica do manipulador robótico. A diferença entre as velocidades reais medidas nas juntas e os valores estimados pelo MLP é chamado vetor de resíduos. Se não ocorrem falhas no manipulador, o vetor de resíduos é próximo de zero. Quando ocorrem falhas no manipulador, o vetor de resíduos é diferente de zero. Esta primeira fase é chamada de geração dos resíduos. Na segunda fase, chamada de análise dos resíduos, outra RNA conhecida como rede de função de base radial (*radial basis function network* - rede RBF) é empregada para classificar o vetor de resíduos. De acordo com o comportamento dos resíduos (assinatura da falha) a rede RBF identifica os diferentes tipos de falhas.

Na sequência, descrevemos brevemente o conceito de redes neurais artificiais, para então descrever o módulo de DIF.

3.2 3.2 Redes Neurais Artificiais

As redes neurais artificiais (RNAs) têm sido aplicadas na solução de uma infinidade de problemas. O adjetivo "neural" é usado porque muito da inspiração de tais redes vem da neurociência (Hertz *et al*., 1991). As RNAs empregam como unidade de processamento fundamental o neurônio artificial, inspirado no funcionamento básico dos neurônios biológicos.

Neste trabalho, as RNA são utilizadas para duas tarefas distintas: aproximação de funções não-lineares e classificação de padrões. O *perceptron* multicamadas (*multilayer perceptron* - MLP) com aprendizado por retropropagação do erro (*backpropagation*) é, atualmente, a RNA mais utilizada para a aproximação de funções não-lineares contínuas. Para a classificação de padrões em DIF, no entanto, esta RNA apresenta alguns problemas. Tais problemas levam à utilização de uma rede RBF para a classificação dos resíduos, veja para maiores detalhes (Tinós e Terra, 2001).

3.2.1 MLP com treinamento por retropropagação do erro

O *perceptron* multicamadas pode ser visto como um veículo prático para realizar mapeamentos de funções não-lineares de maneira geral (Haykin, 1994). A relação entrada/saída do MLP define um mapeamento de um espaço de entrada Euclidiano *p*-dimensional para um espaço de saída Euclidiano *q*-dimensional continuamente diferenciável. Um MLP com apenas uma camada escondida pode ser visto na Figura 2.

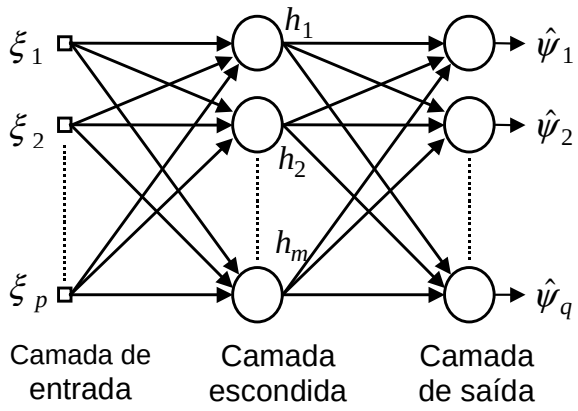


Figura 2. *Perceptron* multicamadas com uma única camada escondida.

Para um MLP com uma única camada escondida, apresentando-se o *n*-ésimo padrão de entrada $\boldsymbol{\xi}(n) = [\xi_1(n)\ \xi_2(n)\ \ldots\ \xi_p(n)]^T$, a ativação do neurônio de saída *k* e a ativação do neurônio *j* da camada escondida, respectivamente, são dadas por

$$\hat{\psi}_k(n) = \varphi_k\left(\sum_{j=0}^{m} \omega_{kj}(n) h_j(n)\right) \tag{3.1}$$

$$h_j(n) = \varphi_j\left(\sum_{i=0}^{p} \omega_{ji}(n) \xi_i(n)\right) \tag{3.2}$$

sendo φ_a a função de ativação não-linear do neurônio *a*, ω_{cb} é o peso entre a saída do neurônio *b* (camada anterior) e a entrada

do neurônio *c* (camada posterior), *i* = 1, ..., *p* é o índice dos neurônios da camada de entrada, *j* = 1, ..., *m* é o índice dos neurônios da camada escondida e *k* = 1, ..., *q* é o índice dos neurônios da camada de saída.

Uma função de ativação não-linear comumente empregada no MLP é a sigmoidal, que é dada por

$$\varphi_a(v_a) = \frac{1}{1+\exp(-v_a)} \tag{3.3}$$

sendo v_a o nível de ativação interna no neurônio *a*.

A seguir, mostra-se o algoritmo para treinamento do MLP por retropropagação do erro. Apresentando-se o *n*-ésimo padrão de treinamento (*n* = 1, ..., n_p) na entrada do MLP, o erro instantâneo na saída *k* é dado por

$$e_k(n) = \psi_k(n) - \hat{\psi}_k(n) \tag{3.4}$$

sendo $\psi_k(n)$ a variável que deve ser estimada pela saída $\hat{\psi}_k(n)$ da RNA. A soma dos erros quadráticos instantâneos nas saídas do MLP para o padrão *n* é dada por

$$E(n) = \frac{1}{2}\sum_{k=1}^{q} e_k^2(n) \tag{3.5}$$

e o erro médio quadrático sobre o conjunto de treinamento é dado por

$$E_{medio} = \frac{1}{n_p}\sum_{n=1}^{n_p} E(n) \tag{3.6}$$

Utilizando-se a lei *delta* (Hertz *et al*., 1991), as conexões entre a camada escondida e a camada de saída são ajustadas por

$$\Delta\omega_{kj}(n) = -\eta(n)\frac{\partial E(n)}{\partial \omega_{kj}(n)} \tag{3.7}$$

sendo η a taxa de aprendizagem e $\Delta\omega_{kj}$ é a correção aplicada ao peso ω_{kj}. Da equação anterior e da Equação (3.5), chega-se à lei de ajuste dos pesos

$$\Delta\omega_{kj}(n) = \eta(n)\delta_k(n)h_j(n) \tag{3.8}$$

sendo

$$\delta_k(n) = e_k(n)\varphi_k'(v_k(n)) \tag{3.9}$$

Do mesmo modo, para as conexões entre a camada de entrada e a camada escondida, tem-se que o ajuste de pesos é dado por

$$\Delta\omega_{ji}(n) = \eta(n)\delta_j(n)\xi_i(n) \tag{3.10}$$

sendo

$$\delta_j(n) = \varphi_j'(v_j(n))\sum_{k=1}^{q}\delta_k(n)\omega_{kj}(n) \tag{3.11}$$

A definição da taxa de aprendizagem tem um papel importante no treinamento por retropropagação do erro. Se a taxa é muito baixa, o algoritmo demorará para convergir. Se, por outro lado, a taxa é muito alta, o algoritmo pode se tornar instável. Um método simples de aumentar a velocidade de convergência e evitar a instabilidade é modificar a lei de ajuste adicionando um termo de *momentum* que é proporcional ao ajuste de pesos anterior (Haykin, 1994).

Neste trabalho, o MLP tem apenas uma camada escondida. O seguinte teorema estabelece que uma RNA direta (*feedforward*) com uma única camada escondida e com um número suficiente de unidades escondidas é capaz de aproximar qualquer função contínua f: $\Re^p \rightarrow \Re^q$ com qualquer precisão desejada.

Teorema 3.1: (Cybenko, 1989) Seja φ qualquer função de ativação contínua. Então, dada qualquer função contínua com valores reais $f(.)$, em um subespaço compacto $s \subset \Re^{n_p}$ e $\varepsilon > 0$, existem vetores $\boldsymbol{\omega}_1$, $\boldsymbol{\omega}_2$,..., $\boldsymbol{\omega}_m$, $\boldsymbol{\alpha}$, $\boldsymbol{\theta}$ e uma função parametrizada $G(.\,, \boldsymbol{\omega}, \boldsymbol{\alpha}, \boldsymbol{\theta}) : \rightarrow \Re$ tal que

$$\left|G(\boldsymbol{\xi}, \boldsymbol{\omega}, \boldsymbol{\alpha}, \boldsymbol{\theta}) - f(\boldsymbol{\xi})\right| < \varepsilon \tag{3.12}$$

$$G(\boldsymbol{\xi}, \boldsymbol{\omega}, \boldsymbol{\alpha}, \boldsymbol{\theta}) = \sum_{j=1}^{m}\alpha_j\varphi\left(\boldsymbol{\omega}_j^T\boldsymbol{\xi} + \theta_j\right) \tag{3.13}$$

sendo que $\boldsymbol{\omega}_j \in \Re^p$, $\boldsymbol{\xi} \in \Re^p$, $\alpha_j \in \Re$, $\theta_j \in \Re$, $\boldsymbol{\omega}_j = [\omega_{j1} ... \omega_{jp}]^T$, $\boldsymbol{\alpha} = [\alpha_1 ... \alpha_m]^T$ e $\boldsymbol{\theta} = [\theta_1 ... \theta_m]^T$.

Este teorema pode ser interpretado da seguinte maneira: uma falha na função de mapeamento do MLP é resultado de uma escolha de parâmetros inadequada ou de um insuficiente número de unidades escondidas (Efrati, 1997).

Neste trabalho, o MLP com treinamento por retropropagação do erro será utilizado para aproximar o modelo dinâmico do manipulador.

Para o problema de classificação, o MLP produz bordas de decisão que separam os padrões das diferentes classes. Bordas de decisão são superfícies (ou linhas para o caso de dois neurônios na entrada) no espaço de entradas onde a saída dos dois (ou mais) neurônios com maior ativação na última camada são iguais para um mesmo padrão. As bordas de decisão produzidas pelo MLP com treinamento por retropopagação são posicionadas muito perto da superfície que separa os padrões de treinamento pertencentes às diferentes classes. Esta característica do MLP pode gerar má classificação dos padrões não treinados em problemas (como por exemplo em DIF) em que as superfícies que separam as diferentes classes são difíceis de serem estipuladas através de um conjunto limitado de padrões. Este problema poderia ser evitado se as bordas de decisão estivessem em posições mais conservadoras. Além disso, nas áreas do espaço de entradas não ocupadas pelos padrões do conjunto de treinamento a classificação é arbitrária já que os pontos de saída desejados somente refletem as ativações da rede para os padrões empregados no treinamento (Leonard e Kramer, 1991). Portanto, sob certas condições, o MLP com treinamento por retropropagação pode produzir bordas de decisão que são não-intuitivas e não-robustas.

3.2.2 Rede com Função de Base Radial (RBF)

Funções de base radial são, simplesmente, uma classe de funções. Moody e Darken (1989) propuseram seu uso em um novo tipo de RNA chamada rede com função de base radial (RBF). Esta RNA é inspirada em neurônios que têm ativações localmente sintonizadas ou neurônios seletivos, que respondem para determinadas faixas de sinais de entrada e são encontrados em diversas partes do corpo humano e de outros animais. Em princípio, as redes RBF podem ser multicamadas e terem funções de ativação na saída não-lineares. Contudo, redes RBF têm tradicionalmente sido associadas com funções radiais em

uma única camada escondida e funções de saída lineares (Orr, 1996).

A rede RBF utilizada neste trabalho tem três camadas distintas. Na primeira, os padrões de entrada são apresentados. Não existem pesos entre a primeira e a segunda camadas, sendo os padrões de entrada repassados para os neurônios da segunda camada (camada escondida). As unidades desta camada têm função de ativação radial. Cada neurônio *j* da camada escondida (chamado unidade radial *j*) é responsável pela criação de um campo receptivo no espaço de entradas centrado em um vetor $\boldsymbol{\mu}_j$, chamado de centro da unidade radial. A unidade radial *j* tem ativação de acordo com a distância entre o vetor de entrada e o centro da unidade radial. Quanto mais próximos forem os dois vetores, maior será a ativação do neurônio (Figura 3). Entre a segunda e a terceira camadas existem pesos e a terceira camada apresenta ativação linear. A *k*-ésima ($k = 1, ..., q$) saída da rede RBF para o *n*-ésimo ($n = 1, ..., n_p$) vetor de entrada $\boldsymbol{\xi}_n$, é dada por

$$\hat{\psi}_k(n) = \sum_{j=0}^{m} \omega_{kj} h_j(n) \tag{3.14}$$

sendo h_j a ativação da unidade radial *j* da camada escondida e ω_{kj} é o peso entre a unidade radial *j* e o neurônio de saída *k*. A ativação das unidades da camada escondida é definida por uma função radial. Uma função radial de ativação comum é a Gaussiana. Aplicando tal função na unidade radial *j*, a ativação desta unidade para o *n*-ésimo vetor de entrada é dada por

$$h_j(n) = \exp\left(-\frac{\left\| \boldsymbol{\xi}(n) - \boldsymbol{\mu}_j \right\|^2}{2\rho^2} \right) \tag{3.15}$$

sendo que ρ determina o tamanho do campo receptivo da unidade radial *j* e $\| \cdot \|$ define a norma do vetor (geralmente a norma Euclidiana).

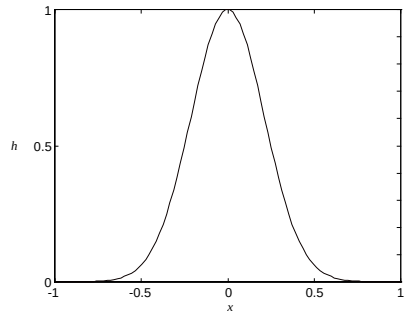


Figura 3. Resposta da função Gaussiana com o centro em 0 ($\mu = 0$) e $\rho = 0{,}3$. Note que a ativação máxima ocorre em $x = \mu$.

Aqui, a função de ativação utilizada nas unidades radiais é a função de Cauchy. Esta função foi escolhida por apresentar um decaimento mais suave conforme os padrões se distanciam do centro da unidade radial. Assim, mesmo os padrões nas regiões do espaço de entradas que não são ocupadas pelos padrões de treinamento serão classificados de acordo com a distância até o centro mais próximo, sem a necessidade de se aumentar o tamanho do campo receptivo. A função de Cauchy é dada por

$$h_j(n) = \frac{1}{1 + \left\| R^{-1}\left(\boldsymbol{\xi}(n) - \boldsymbol{\mu}_j\right) \right\|^2} \tag{3.16}$$

sendo **R** uma matriz diagonal formada pelos parâmetros individuais que definem o tamanho do campo receptivo em cada dimensão do espaço de entradas, ou seja

$$R = \begin{bmatrix} \rho_1 & 0 & 0 & 0 \\ 0 & \rho_2 & 0 & 0 \\ 0 & 0 & \ddots & 0 \\ 0 & 0 & 0 & \rho_p \end{bmatrix} \tag{3.17}$$

Desta forma, o campo receptivo não precisa necessariamente ter o mesmo tamanho em todas as dimensões do espaço de entradas. Se uma determinada variável do vetor de entrada tem um poder discriminatório menor, o tamanho do campo receptivo em sua dimensão pode ser maior, priorizando-se, assim, as outras variáveis.

Para os problemas DIF, a rede RBF pode produzir bordas de decisão que são mais robustas e intuitivas que as do MLP, já que a classificação é feita considerando-se a proximidade entre o padrão a ser classificado e os centros das unidades radiais. É claro que este resultado é dependente da escolha adequada das unidades radiais e de seus parâmetros.

As redes RBF ainda apresentam outras vantagens sobre o MLP, tais como inexistência de mínimos locais no cálculo dos pesos e um menor tempo de treinamento (Looney, 1997). Entre as desvantagens das redes RBF, pode-se citar: maior ocupação de memória devido ao alto número de unidades escondidas (unidades radiais), menor velocidade de operação devido ao maior número de unidades escondidas e possível escolha subótima das unidades radiais e de seus parâmetros. Neste ponto, uma questão que pode surgir é: por que não utilizar também a rede RBF para a aproximação da função dinâmica do sistema no problema de DIF? Realmente, a rede RBF apresenta bons resultados no mapeamento de sistemas não-lineares com poucas variáveis medidas (Nelles e Isermann, 1995). Contudo, se a dimensão do espaço de entradas é grande, um número de unidades radiais bastante alto é requerido para tratar a complexidade do problema (Warwick e Craddock, 1996). Isto pode resultar em uma baixa generalização e em um esforço computacional extremamente alto durante a fase de treinamento da rede RBF.

Consideraremos agora o problema do treinamento da rede RBF. Se a rede RBF tem apenas uma camada escondida e as unidades radiais são fixadas (isto é, o número de unidades radiais e seus parâmetros são constantes), então esta RNA pode ser vista como um modelo linear. Neste caso, o treinamento pode ser feito de maneira desacoplada: primeiro determinam-se as unidades radiais e seus parâmetros e depois calculam-se os pesos da segunda camada de acordo com as ativações das unidades radiais e com as saídas desejadas. Assim, evita-se o uso de algoritmos de otimização não-lineares, como os do tipo gradiente descendente, que apresentam um esforço computacional relativamente alto para cálculo dos pesos. Outra vantagem é que evitam-se os mínimos locais no cálculo dos pesos.

A determinação do número de unidades radiais e seus parâmetros é a primeira etapa do treinamento. O método RBF original utiliza como centros todos os padrões de treinamento. No entanto, como o número de padrões é muito grande em DIF, este método é raramente utilizado. Além disso, utilizando-se um número grande de unidades radiais pode haver problemas de sobreajuste, ou seja, a classificação será sensível à escolha do conjunto de treinamento, apresentando uma baixa

generalização (maus resultados para padrões não-apresentados).

Neste trabalho, o mapa auto-organizável de Kohonen (MAOK) é empregado para a seleção das unidades radiais da rede RBF. O uso do MAOK para o treinamento de redes RBF não é um enfoque novo. Em Ojala e Vuorimaa (1995) por exemplo, o MAOK é usado para a seleção inicial dos centros das unidades radiais e, então, um algoritmo de *Learning Vector Quantization* modificado (Kohonen, 1995) é utilizado para sintonizar todos os parâmetros da rede (como os centros e os pesos). Neste trabalho, algumas mudanças são feitas para adequar o MAOK para o treinamento da rede RBF aplicada no problema de DIF. Apesar deste algoritmo ser não-supervisionado (ou seja, não necessita do conhecimento das saídas desejadas), ele será aplicado em um problema supervisionado. Assim, aproveitando as características do problema, o conjunto de treinamento será separado de acordo com as diferentes classes. Este procedimento é adotado para evitar que padrões de treinamento pertencentes a diferentes classes sejam sintonizados em uma mesma unidade radial. Portanto, o algoritmo descrito abaixo deve ser repetido para cada classe.

Inicialmente, todos os padrões de treinamento são considerados como centros das unidades radiais. Utilizando os padrões pertencentes a uma determinada classe, calcula-se as ativações das unidades radias (Equação (3.15)) para cada padrão de treinamento (da mesma classe) e a unidade com a maior ativação é selecionada de acordo com

$$h_c(t) = \max_j \{h_j(t)\} \tag{3.18}$$

sendo c a unidade radial selecionada, j=1,...,m_k (m_k é o número de padrões da classe k), k=1,...,q (q é o número de classes) e t=1,...,t_{max} é o índice do tempo discreto (t_{max} deve ser múltiplo de m_k). Em cada amostra t, um padrão diferente do subconjunto de treinamento deve ser apresentado (para que o algoritmo apresente uma convergência conveniente, o subconjunto de treinamento deve ser apresentado diversas vezes). O passo seguinte é atualizar as posições dos centros das unidades radiais de acordo com

$$\boldsymbol{\mu}_j(t+1) = \boldsymbol{\mu}_j(t) + \alpha(t)\beta(t)[\boldsymbol{\xi}(t) - \boldsymbol{\mu}_j(t)] \tag{3.19}$$

sendo $\alpha(t)$ uma função de decaimento no tempo que define a taxa de aprendizagem e $\beta(t)$ uma função da distância vetorial entre o centro da unidade radial j ($\boldsymbol{\mu}_j$) e o centro da unidade radial selecionada ($\boldsymbol{\mu}_c$). Aqui, esta função é dada por

$$\beta(t) = \begin{cases} \dfrac{1}{1 + \left\| R^{-1}(\boldsymbol{\mu}_c - \boldsymbol{\mu}_j) \right\|^2}, & \text{se} \left\| R^{-1}(\boldsymbol{\mu}_c - \boldsymbol{\mu}_j) \right\| < \sigma(t), \\ 0, & \text{se} \left\| R^{-1}(\boldsymbol{\mu}_c - \boldsymbol{\mu}_j) \right\| \geq \sigma(t) \end{cases} \tag{3.20}$$

sendo que a função de espalhamento $\sigma(t)$ decai com o tempo e define o tamanho da vizinhança ao redor do centro da unidade radial selecionada c ($\boldsymbol{\mu}_c$). Se o número de iterações (t_{max}) é suficientemente grande e os parâmetros de treinamento são apropriadamente escolhidos, os centros das unidades radiais em um mesmo aglomerado (*cluster*) deverão se mover até as proximidades do centro do aglomerado (ou seja, no local em que a média das ativações das unidades radiais para todos os padrões de um mesmo aglomerado seja a maior). Após o treinamento, vários centros de um mesmo aglomerado estarão na mesma posição ou em posições muito próximas. Pode-se, portanto, juntar estas unidades radiais em uma única, já que as ativações de duas unidades radiais com o mesmo centro são iguais. Agindo desta forma, a complexidade da rede RBF diminui pois o número de parâmetros adaptativos (pesos) decresce. Este procedimento também é importante para evitar problemas de singularidade na matriz usada para determinar os pesos ótimos. Se existem dois centros muito próximos, a inversa da matriz utilizada para cálculo dos pesos sofre problemas de mau condicionamento. A união das unidades radiais com centros muito próximos é feita verificando se a norma da distância entre duas unidades radiais é muito pequena. Se a resposta for afirmativa, uma unidade radial é retirada.

Repetindo o procedimento descrito acima para todas as classes, as unidades radiais de cada subconjunto são agrupadas em uma única rede RBF e a matriz de pesos ótima é calculada. Minimizando a soma dos erros quadráticos, a matriz de pesos ótima é dada por

$$\hat{\mathbf{W}} = \left(\mathbf{H}^T\mathbf{H}\right)^{-1}\mathbf{H}^T\boldsymbol{\Psi} \tag{3.21}$$

sendo a matriz n_p x m $\mathbf{H}$ formada pelas ativações h_j das unidades radiais para os n_p padrões e a matriz de saída n_p x m $\boldsymbol{\Psi}$ formada pelas saídas desejadas do conjunto de treinamento.

3.3 Geração dos Resíduos

Para a utilização da equação dinâmica do manipulador livre de falhas (Equação (2.2)) como função a ser reproduzida pelo MLP, é preciso que as acelerações das juntas sejam medidas. Entretanto, os dados de aceleração geralmente não são disponíveis nos robôs manipuladores. Considerando uma taxa amostral suficientemente pequena, pode-se substituir

$$\ddot{\mathbf{q}}(t) = \frac{\dot{\mathbf{q}}(t+\Delta t) - \dot{\mathbf{q}}(t)}{\Delta t} \tag{3.22}$$

na Equação (2.2), de modo que a dinâmica do manipulador sem falhas é dada por

$$\dot{\mathbf{q}}(t+\Delta t) = \mathbf{M}(\mathbf{q}(t))^{-1}\left[\boldsymbol{\tau}(t) - \mathbf{b}(\mathbf{q}(t), \dot{\mathbf{q}}(t))\right]\Delta t + \dot{\mathbf{q}}(t). \tag{3.23}$$

Procedendo da mesma forma para a Equação (2.3), tem-se que a dinâmica do manipulador com falhas é dada por

$$\begin{aligned} \dot{\mathbf{q}}(t+\Delta t) = \mathbf{M}(\mathbf{q}(t))^{-1}\left[\boldsymbol{\tau}(t) - \mathbf{b}(\mathbf{q}(t), \dot{\mathbf{q}}(t))\right]\Delta t + \\ + \boldsymbol{\phi}(\mathbf{q}(t), \dot{\mathbf{q}}(t), \boldsymbol{\tau}(t))\Delta t + \dot{\mathbf{q}}(t) \end{aligned}. \tag{3.24}$$

O MLP deve reproduzir a equação dinâmica do manipulador sem falhas (Equação 3.23). Assim, os dados de entrada do MLP são as posições, velocidades e torques nas juntas no instante t. As saídas do MLP são as estimativas das velocidades medidas no instante (t+Δt). Deste modo, as entradas e as saídas desejadas no padrão n são

$$\boldsymbol{\xi}(n) = \left[\mathbf{q}(t)^T \quad \dot{\mathbf{q}}(t)^T \quad \boldsymbol{\tau}(t)^T\right]^T, \ \boldsymbol{\psi}(n) = \left[\dot{\mathbf{q}}(t+\Delta t)\right].$$

É importante observar que o período amostral está implícito no mapeamento realizado pelo MLP. O erro de mapeamento do MLP (manipulador sem falhas) no padrão n é dado por

$$\boldsymbol{\psi}(n) - \hat{\boldsymbol{\psi}}(n) = \dot{\mathbf{q}}(t+\Delta t) - \hat{\dot{\mathbf{q}}}(t+\Delta t) = \mathbf{e}(t+\Delta t). \tag{3.25}$$

Note que os distúrbios não-correlacionados (por exemplo, os ruídos de medida) estão implícitos nesta equação. Assim, o vetor de resíduos do manipulador livre de falhas é

$$\hat{\boldsymbol{\phi}}(t+\Delta t) = \mathbf{e}(t+\Delta t). \tag{3.26}$$

Portanto, para o manipulador sem falhas, o vetor de resíduos varia exclusivamente devido aos erros de mapeamento e aos

distúrbios não-correlacionados do MLP. Considerando agora os efeitos das falhas no manipulador, o vetor de resíduos é dado pela diferença das Equações (3.23) e (3.24) acrescida do erro de mapeamento

$$\hat{\boldsymbol{\phi}}(t+\Delta t) = \boldsymbol{\phi}(\mathbf{q}(t), \dot{\mathbf{q}}(t), \boldsymbol{\tau}(t))\Delta t + \mathbf{e}(t+\Delta t). \quad (3.27)$$

Desta forma, se o erro de mapeamento do MLP e os distúrbios não-correlacionados forem significativos, o vetor de falhas poderá ser encoberto, ocasionando erros na DIF. Como os erros de mapeamento e os ruídos de medida são inevitáveis em sistemas reais, utiliza-se a rede RBF para construir bandas de detecção variáveis. O esquema de geração dos resíduos pode ser visto na Figura 4.

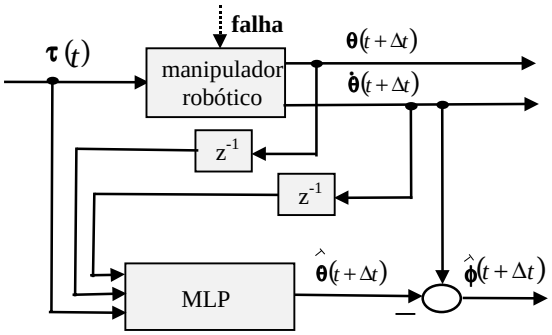
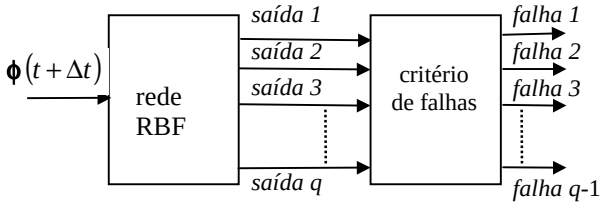


Figura 4. Geração dos resíduos.

3.4 Análise dos Resíduos

A análise dos resíduos é feita através da rede RBF. Os padrões de entrada desta rede são as componentes do vetor de resíduos. A arquitetura do sistema de análises dos resíduos pode ser vista na Figura 5. A saída da rede RBF apresenta um vetor *q*-dimensional, sendo que as *q*-1 primeiras saídas indicam a ocorrências das *q*-1 falhas e a última saída indica a operação sem falhas. A rede RBF deve separar o espaço de entradas *p*-dimensional (*p* resíduos) em *q* regiões associadas à operação normal do robô manipulador ou às diferentes falhas.

Figura 5. Análise dos resíduos.



A rede RBF é treinada para que suas saídas apresentem sinal 1 caso o padrão de entrada esteja na classe relacionada à saída e 0 caso contrário (por exemplo, se ocorre uma falha do tipo 2, o neurônio de saída 2 deve ter ativação igual a 1 e os outros neurônios devem ter ativação igual a 0). O critério de detecção considera que para uma falha ser detectada, o neurônio de saída correspondente deve ter uma sequência (com um número pré-definido de amostras) de ativações maiores que as ativações das outras saídas. Este critério é adotado para evitar que padrões isolados mal-classificados causem alarmes falsos. Assim, se é considerado que o número de amostras para a detecção deve ser 3, então a falha 1 é detectada quando o neurônio de saída 1 apresentar 3 ativações consecutivas maiores que as ativações dos outros neurônios de saída. Este critério não diminui significativamente a sensibilidade do sistema de DIF pois, em geral, o período amostral é baixo.

3.5 Procedimento para DIF

A seguir, os procedimentos para treinamento e operação do sistema de DIF são resumidos.

3.5.1 Procedimento para Treinamento

Gera-se um conjunto significativo de trajetórias livres de falhas e, em cada amostra, medem-se as posições, as velocidades angulares e os torques nas juntas;

as amostras deste conjunto de treinamento são normalizadas e, posteriormente, empregadas para treinar o MLP. Na entrada do MLP, apresentam-se as posições, as velocidades e os torques das juntas em cada instante amostral. O vetor de saídas do MLP é comparado com as velocidades das juntas do instante amostral seguinte. A diferença entre estes dois vetores é utilizada para ajustar os pesos do MLP. Este procedimento deve ser repetido até que um critério de parada (como o número máximo de épocas de treinamento) seja alcançado;

avalia-se o mapeamento do MLP apresentando-se os dados normalizados de trajetórias não-treinadas. Se a análise não é satisfatória, repetem-se os passos anteriores mudando-se o conjunto de treinamento, pois provavelmente o conjunto não é suficientemente representativo, ou os parâmetros de treinamento. Se a análise é satisfatória, passa-se para o passo seguinte;

gera-se um conjunto significativo de trajetórias. Os dados de cada trajetória são normalizados e devem ser apresentados *q* vezes (*q* é o número de classes): em cada apresentação, um tipo de falha diferente deve ocorrer e, na última, nenhuma falha ocorre. Em cada amostra, as posições, as velocidades e os torques das juntas são apresentadas ao MLP que gera, assim, o vetor de resíduos;

as amostras dos vetores de resíduos gerados são apresentadas à rede RBF. O MAOK é utilizado para seleção de centros. Os pesos são calculados pela Equação (3.21). O vetor de saídas da rede RBF, após ser analisado pelo critério de falhas, é comparado com as informações reais das falhas e,

avalia-se a classificação realizada pela rede RBF apresentando-se dados de trajetórias não-treinadas em que ocorrem cada tipo de falha e operação normal. Analisa-se a ocorrência de alarmes falsos e se as falhas são detectadas e isoladas corretamente. Se a análise é negativa, repetem-se os passos 4, 5 e 6 mudando-se o conjunto de treinamento, pois provavelmente o conjunto não é suficientemente representativo, ou os parâmetros de treinamento. Se a análise é positiva, finaliza-se o treinamento.

3.5.2 Procedimento para Operação

1. Medem-se as posições e velocidades das juntas no instante $t+\Delta t$ através de sensores;

2. os sinais medidos e os torques (fornecidos pelo sistema de controle) no instante $t+\Delta t$ sofrem pré-prossessamentos (conversões, normalizações, etc.);

3. os valores das velocidades, posições e torques das juntas no instante t (já pré-processados) são apresentados à entrada do MLP, que tem como saída a estimativa das velocidades das juntas no instante $t+\Delta t$;

4. a saída do MLP é comparada com as velocidades das juntas pré-processadas medidas no instante $t+\Delta t$, gerando o vetor de resíduos;

5. vetor de resíduos no instante $t+\Delta t$ é apresentado à entrada da rede RBF;

6. a rede RBF classifica os padrões de entrada e gera um vetor de saídas que, após ser analisado pelo critério de falhas, indica o estado de operação do sistema (sem falhas ou ocorrência de determinada falha) e,

7. incrementa-se o tempo t e retorna-se para o passo 1.

4 CONTROLE PÓS-FALHA DE ROBÔS MANIPULADORES

4.1 Manipuladores Subatuados

Após a ocorrência de uma falha em um de seus atuadores, o manipulador torna-se um sistema subatuado. O termo subatuado refere-se ao fato de a ausência de atuação em uma ou mais juntas torna estas juntas não diretamente controláveis. Controle *de todas as juntas* do manipulador após a falha é fundamental do ponto de vista de tolerância à falhas, principalmente nos casos em que a localização do manipulador impede sua manutenção.

A presença de juntas não atuadas torna os manipuladores subatuados fundamentalmente diferentes dos convencionais, onde todas as juntas são atuadas, com relação aos seguintes pontos:

- acoplamento dinâmico: controle dos graus de liberdade (GDLs) não atuados só é possível quando o acoplamento entre eles e os GDLs atuados é suficiente para permitir transmissão de forças e torques destes para aqueles;

- controlabilidade dos GDLs não atuados: todos os GDLs de um manipulador subatuado são, em geral, controláveis, embora não concorrentemente. Enquanto o controle dos GDLs atuados pode ser feito diretamente, controle dos GDLs não atuados só é possível indiretamente através da aplicação de forças nos GDLs atuados;

- restrições não-holonômicas: as equações dinâmicas de manipuladores subatuados geralmente incluem restrições não-holonômicas, que aparecem devido à falta de atuação em alguns dos GDLs;

- não-linearidades: as equações dinâmicas de manipuladores subatuados são não-lineares, e dependem não só dos parâmetros cinemáticos mas também dos parâmetros dinâmicos do sistema.

O estudo de manipuladores mecânicos subatuados iniciou-se recentemente, com o estudo e controle de um manipulador de dois braços equipado com um motor e um freio (Arai e Tachi, 1991). Outros experimentos seguiram-se, voltados ao controle de manipuladores montados em satélites espaciais (Papadopoulos e Dubowsky, 1991), (Mukherjee e Chen, 1993), e manipuladores sem restrições não-holonômicas (Oriolo e Nakamura, 1991). A modelagem e análise de manipuladores subatuados também foram estudadas (Jain e Rodriguez., 1993). Mais recentemente, os problemas de modelagem, controlabilidade, controle, e planejamento de trajetórias para manipuladores subatuados foram estudados pelos autores (Bergerman, 1996) e (Bergerman e Xu, 1996). Todos estes trabalhos têm em comum o fato de que as juntas não-atuadas são equipadas com freios, o que facilita o controle do respectivo manipulador.

Antes de passar à explicação matemática de como as posições de todas as juntas de um manipulador subatuado podem ser controladas, apresentaremos um exemplo como motivação. Considere um manipulador com duas juntas. Imagine que durante o movimento das duas juntas de suas posições iniciais em direção aos seus *set-points* a junta 2 sofre uma falha. A falha é identificada em tempo real pelo método apresentado anteriormente, e imediatamente ambas as juntas são freadas. A partir deste momento o manipulador torna-se subatuado. Como a junta 2 não mais possui atuação, para levá-la ao *set-point* usamos o movimento da junta 1 e o acoplamento entre as juntas; obviamente, neste movimento não se controla a posição da junta 1. Assim que a junta 2 atinge seu *set-point*, ela é freada permanentemente, e pode-se proceder ao controle da posição da junta 1. O leitor perceberá, então, que este manipulador subatuado pôde ter todas suas juntas controladas em duas fases: controle da junta não atuada, seguida pelo controle da junta atuada.

Quando o manipulador possui mais juntas não atuadas do que atuadas, o raciocínio pode ser estendido facilmente. Primeiramente, controla-se um subconjunto das juntas não atuadas através de seu acoplamento com as juntas atuadas, mantendo-se as outras juntas não atuadas freadas. Na sequência, controlam-se outros subconjuntos das juntas não atuadas, até que todas tenham atingido seus *set-points*. Finalmente, controla-se a posição das juntas atuadas. Este método é descrito nos próximos parágrafos do ponto de vista de teoria de controle.

4.2 Modelagem de Manipuladores Subatuados

Considere um manipulador com n graus de liberdade. Seja q o vetor de posição das juntas e τ o vetor de torque; as equações dinâmicas do manipulador são determinadas pelo método de Lagrange como:

$$\tau = M(q)\ddot{q} + b(q,\dot{q}) . \tag{4.1}$$

Aqui, assume-se que n_a graus de liberdade do manipulador são juntas ativas com atuadores funcionais e sensores de posição. Os $n_p = n - n_a$ graus de liberdade restantes são juntas passivas, ou seja, juntas cujos atuadores falharam, e possuem pelo menos os freios operantes.

Para distinguir entre as posições e os torques das juntas ativas e passivas, particiona-se a equação (4.1) como:

$$\begin{bmatrix} \tau_a \\ \tau_p \end{bmatrix} = \begin{bmatrix} M_{aa} & M_{ap} \\ M_{pa} & M_{pp} \end{bmatrix} \begin{bmatrix} \ddot{q}_a \\ \ddot{q}_p \end{bmatrix} + \begin{bmatrix} b_a \\ b_p \end{bmatrix} \tag{4.2}$$

sendo que os subscritos a e p denotam quantidades relacionadas às juntas ativas e passivas, respectivamente.

A diferença entre este manipulador e um completamente atuado com o mesmo número de graus de liberdade é que os elementos de τ_p, correspondentes aos torques das juntas passivas, não podem ser controlados diretamente. De fato, estes elementos podem assumir apenas os valores 0 ou τ_l, sendo τ_l o torque nominal do freio das juntas passivas, necessário para

manter as juntas passivas bloqueadas enquanto outras juntas são controladas. Para distinguir entre juntas passivas bloqueadas e livres, particiona-se a equação (4.2) como:

$$\begin{bmatrix}\tau_a\\ \tau_u\\ \tau_l\end{bmatrix}=\begin{bmatrix}M_{aa} & M_{au} & M_{al}\\ M_{ua} & M_{uu} & M_{ul}\\ M_{la} & M_{lu} & M_{ll}\end{bmatrix}\begin{bmatrix}\ddot{q}_a\\ \ddot{q}_u\\ \ddot{q}_l\end{bmatrix}+\begin{bmatrix}b_a\\ b_u\\ b_l\end{bmatrix} \quad (4.3)$$

sendo que os subscritos *u* e *l* denotam quantidades relacionadas às juntas passivas livres (*unlocked*) e freadas (*locked*), respectivamente. Note que $\tau_u = 0$ pois não são aplicados torques nas juntas passivas livres, e que $\ddot{q}_l = 0$ pois as juntas passivas freadas não se movimentam. A terceira linha desta equação pode ser eliminada, pois os torques das juntas passivas freadas compensam exatamente os torques inerciais e não inerciais do lado direito. Desta forma, o modelo dinâmico do manipulador subatuado pode ser escrito como:

$$\begin{bmatrix}\tau_a\\ 0\end{bmatrix}=\begin{bmatrix}M_{au} & M_{aa}\\ M_{uu} & M_{ua}\end{bmatrix}\begin{bmatrix}\ddot{q}_u\\ \ddot{q}_a\end{bmatrix}+\begin{bmatrix}b_a\\ b_u\end{bmatrix}. \quad (4.4)$$

A dimensão do vetor das juntas passivas livres é limitada ao número de atuadores. A razão é que se pode controlar no máximo n_a juntas a cada instante, sejam elas juntas ativas ou passivas (Arai e Tachi, 1991). Portanto, não mais que n_a juntas passivas devem estar livres em um dado instante. Sendo assim, as n_a juntas controladas em um dado momento podem ser as juntas ativas ou um subconjunto de n_a juntas passivas (quando $n_p \geq n_a$), ou uma combinação de juntas passivas e ativas. Para o projeto do controlador do manipulador, é conveniente rescrever a equação dinâmica em uma forma em que pode-se distinguir entre as juntas sendo controladas e as restantes:

$$\begin{bmatrix}\tau_a\\ 0\end{bmatrix}=\begin{bmatrix}M_{ac} & M_{ar}\\ M_{uc} & M_{ur}\end{bmatrix}\begin{bmatrix}\ddot{q}_c\\ \ddot{q}_r\end{bmatrix}+\begin{bmatrix}b_a\\ b_u\end{bmatrix} \quad (4.5)$$

sendo q_c o vetor contendo as n_a juntas controladas e q_r o vetor das juntas restantes. Não se incluem as juntas bloqueadas q_l no vetor q_r, pois estas não contribuem para a dinâmica do sistema.

Como mencionado acima, existem três possibilidades de se formar o vetor q_c, correspondendo às três estratégias de controle de um manipulador subatuado. Primeiramente, q_c pode conter apenas juntas ativas. Neste caso, as demais juntas, isto é, as juntas passivas, são mantidas freadas, o que permite considerar o manipulador como sendo completamente atuado. Esta estratégia é denominada estratégia de controle A, pois somente juntas ativas estão sendo controladas. Assim, $q_c = q_a$, a dimensão de q_r é zero e a equação dinâmica fica:

$$\tau_a = M_{aa}\ddot{q}_a + b_a . \quad (4.6)$$

A segunda estratégia de controle é denominada P, quando somente juntas passivas são controladas. Neste caso, $\ddot{q}_c = \ddot{q}_u$ e $\ddot{q}_r = \ddot{q}_a$ e a equação dinâmica fica:

$$\begin{bmatrix}\tau_a\\ 0\end{bmatrix}=\begin{bmatrix}M_{au} & M_{aa}\\ M_{uu} & M_{ua}\end{bmatrix}\begin{bmatrix}\ddot{q}_u\\ \ddot{q}_a\end{bmatrix}+\begin{bmatrix}b_a\\ b_u\end{bmatrix}. \quad (4.7)$$

Isolando $\ddot{q}_a$ na segunda linha da equação (4.7) e substituindo o resultado na primeira linha, obtém-se a relação de malha aberta entre $\ddot{q}_u$ e τ_a:

$$\tau_a = (M_{au} - M_{aa}M_{ua}^{-1}M_{uu})\ddot{q}_u + b_a - M_{aa}M_{ua}^{-1}b_u \quad (4.8)$$

Finalmente, a terceira estratégia é denominada AP; nela uma combinação de juntas ativas e passivas são controladas. O vetor q_c conterá todos os elementos de q_u e alguns elementos de q_a, enquanto que q_r conterá os elementos restantes de q_a não contidos em q_c. Neste caso uma relação equivalente a (4.8) pode ser obtida como:

$$\tau_a = (M_{ac} - M_{ar}M_{ur}^{-1}M_{uc})\ddot{q}_c +$$

Com esta caracterização das estratégias de controle para um manipulador subatuado, pode-se formalizar o método de controle de todas as suas juntas para um dado *set-point*. Quando o manipulador tem mais juntas ativas do que passivas, a sequência de controle consiste em uma estratégia P ou AP (para controle das posições das juntas passivas) seguida pelo travamento das juntas passivas e por uma estratégia A (para controle das juntas ativas). Quando o manipulador tem mais juntas passivas do que ativas, a sequência de controle consiste em várias estratégias P ou AP (para controle da posições das juntas passivas) seguidas pelo travamento das juntas passivas e por uma estratégia A (para controle das juntas ativas).

As três estratégias acima levam a uma relação entre $\ddot{q}_c$ e τ_a da forma:

$$\tau_a = \overline{M}_{ac}\ddot{q}_c + \overline{b}_a . \quad (4.10)$$

A semelhança desta equação com a equação de um manipulador completamente atuado leva à escolha natural por um controlador do tipo torque calculado, composto por uma linearização por realimentação de estados, para controlar a posição das juntas em q_c. Este método consiste em definir uma entrada auxiliar u com:

$$\tau_a = \overline{M}_{ac}u + \overline{b}_a \quad (4.11)$$

de forma que, quando $\overline{M}_{ac}$ for não-singular:

$$\ddot{q}_c = u . \quad (4.12)$$

Neste artigo, assume-se que o acoplamento dinâmico entre as juntas ativas e as controladas é sempre não-nulo em todo o espaço de trabalho, independentemente da estratégia de controle utilizada. Ou seja, $\overline{M}_{ac}$ é sempre não-singular e a condição de controlabilidade das juntas controladas (Bergerman *et al.*, 1995) é garantida.

O efeito da linearização por realimentação de estados é desacoplar e linearizar o sistema não linear (4.10). Na ausência de erros de modelagem e distúrbios externos, a substituição de u em (4.10) por um controlador PID da forma:

$$u = \ddot{q}_c^d + K_d(\dot{q}_c^d - \dot{q}_c) + K_p(q_c^d - q_c) + K_i\int_{t_0}^{t}(q_c^d - q_c)dt \quad (4.13)$$

garante que q_c segue uma trajetória desejada $q_c^d(t)$.

Entretanto, erros de modelagem e distúrbios externos são comuns, o que leva à necessidade da síntese de controladores robustos capazes de diminuir a influência destes erros e distúrbios no desempenho do manipulador. Na sequência apresentamos alguns controladores já estudados pelos autores e que provêem desempenho com robustez bastante superior à do controlador PID da equação (4.13).

No que segue, usaremos uma notação especial para denotar a estrutura do manipulador com relação às suas juntas ativas e passivas. Cada junta ativa será denotada por A, e cada junta passiva por P, esteja ela freada ou não. Desta forma, um manipulador AAP é um manipulador de três juntas com as duas primeiras ativas e a terceira passiva. O leitor deve tomar o cuidado para não confundir esta notação com as estratégias de controle A, P, e AP definidas acima.

4.3 Controle por Estrutura Variável

O controle por estrutura variável (CEV) (Hung *et al.,* 1993) possui reconhecida robustez a incertezas no modelo e distúrbios externos. O CEV consegue tal robustez forçando a trajetória do estado do sistema a seguir uma superfície pré-definida de deslizamento no espaço de estados. Uma vez atingida a superfície de deslizamento, é ela quem dita o comportamento dinâmico do sistema e, portanto, erros e distúrbios não mais influenciam o desempenho. A trajetória de estados resultante é conhecida como modo deslizante. A trajetória do estado desde o estado inicial até a superfície deslizante é conhecida como fase de alcance.

É responsabilidade do projetista garantir que o modo deslizante tenha início depois de um tempo finito desde o início do movimento. Seja $s(x) = 0$, a descrição da superfície deslizante como uma função do estado x. A superfície deslizante será alcançada em um tempo finito se:

$$s_i \dot{s}_i < 0 \tag{4.14}$$

para todos os elementos de s.

No CEV, a entrada auxiliar u em (4.11) é dada por:

$$u = \Gamma_c \tilde{\dot{q}}_c + \ddot{q}_c^d + P_c \operatorname{sgn}(s_c) \tag{4.15}$$

sendo $\tilde{\dot{q}}_C$ a derivada do erro entre o valor desejado e o valor atual de q_c, $\ddot{q}_c^d$ é a aceleração desejada das juntas controladas, e Γ_c e P_c são matrizes de ganho diagonais com elementos positivos.

Para mostrar que a superfície de deslizamento é atingida em tempo finito, define-se a superfície:

$$s = \Gamma_c \tilde{q}_c + \tilde{\dot{q}}_c . \tag{4.16}$$

Usando (4.12) e (4.15) pode-se verificar que:

$$s_{c,i} \dot{s}_{c,i} = -P_{ci,i} \left| s_{c,i} \right| < 0 \tag{4.17}$$

sendo $s_{c,i}$ o i-ésimo elemento de s_c e $P_{ci,i}$ o i-ésimo elemento da diagonal de P_c. Esta condição garante que a trajetória de estado alcança a superfície $s_c = 0$ em um tempo finito. Uma vez alcançada a superfície deslizante, a dinâmica de $(\tilde{q}_c, \tilde{\dot{q}}_c)$ é ditada por $s_c = \Gamma_c \tilde{q}_c + \tilde{\dot{q}}_c = 0$, independentemente de erros de modelo ou distúrbios externos. Por representar um sistema linear de primeira ordem, uma apropriada escolha de Γ_c garante convergência exponencial de $(\tilde{q}_c, \tilde{\dot{q}}_c)$ para (0,0) ou , equivalentemente, de q_c para o desejado q_c^d.

O uso da função sgn(.) em (4.15) introduz *chattering* na trajetória de estados. Para eliminar este *chattering* e evitar a excitação de componentes dinâmicos de alta freqüência não modelados, pode-se utilizar alternativamente a função saturação definida como:

$$\operatorname{sat}(x) = \begin{cases} \operatorname{sgn}(x), & x > \varepsilon \\ x/\varepsilon, & x \le \varepsilon \end{cases} \tag{4.18}$$

sendo ε a largura da chamada camada-limite, ou a tolerância para a variação do deslizamento. O valor de ε deve ser escolhido pelo projetista como um compromisso entre a precisão do estado em regime e o nível de *chattering* aceitável.

A Figura 6 apresenta o diagrama de blocos do controlador robusto.

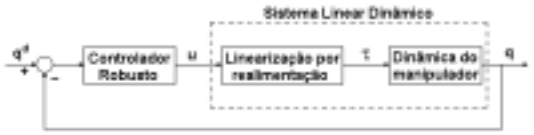


Figura 6. Diagrama de blocos do controlador robusto via linearização por realimentação de estados.

Embora apresente propriedades de robustez muito superiores às do controlador PID, o CEV ainda apresenta uma de suas desvantagens: a necessidade do cálculo do modelo dinâmico do manipulador em tempo real, uma tarefa computacionalmente cara. Natural é, então, buscar-se um controlador robusto cujo cálculo *on-line* independa do modelo do robô. Tal controlador é apresentado na sequência.

4.4 Controle por Desigualdades Matriciais Lineares

O controle por desigualdades matriciais lineares (DML) utiliza o modelo dinâmico do manipulador apenas na fase de projeto, *off-line*, prescindindo do mesmo durante o controle. Desta forma, o controlador DML possui um custo computacional baixo ao mesmo tempo em que apresenta robustez como o CEV (Boyd *et al.*, 1994) e (Terra *et al.*, 1999).

No projeto do controlador DML, a equação dinâmica do manipulador subatuado (4.10) é linearizada para se obter uma representação na forma de variáveis de estado. Sendo a matriz de inércia $\overline{M}_{ac}$ não singular, tem-se:

$$\ddot{q}_c - \overline{M}_{ac}^{-1} \tau_a + \overline{M}_{ac}^{-1} b_a = 0 . \tag{4.19}$$

Quando esta equação é linearizada nas vizinhanças de um ponto de operação, ou seja, um par (posição, velocidade) dentro do espaço de trabalho, o sistema pode ser representado pela seguinte equação de estados (Arai e Tachi, 1991):

$$\dot{x} = Ax + B\tau_a \tag{4.20}$$

sendo

$$A = \begin{bmatrix} 0 & I \\ -\overline{M}_{ac}^{-1} \dfrac{\partial b_a}{\partial q_c} & -\overline{M}_{ac}^{-1} \dfrac{\partial b_a}{\partial \dot{q}_c} \end{bmatrix}$$

$$B = \begin{bmatrix} 0 \\ \overline{M}_{ac}^{-1} \end{bmatrix}$$

$$x = \begin{bmatrix} \delta q_c \\ \delta \dot{q}_c \end{bmatrix} . \tag{4.21}$$

Para cada estratégia de controle definida anteriormente (A, P ou AP), as matrizes $\overline{M}_{ac}$ e $\overline{b}_{ac}$ possuem dimensões compatíveis com a dimensão de q_c. Portanto, as matrizes A e B terão dimensões relacionadas com o número de juntas sendo controladas.

No projeto do controlador DML (vide Apêndice A), devem ser obtidas matrizes constantes A e B, e matrizes de incertezas variantes no tempo $\Delta A(t)$ e $\Delta B(t)$. Para o manipulador subatuado, estas matrizes podem ser calculadas de acordo com os seguintes passos:

1. Para cada estratégia de controle, linearize a equação (4.10) em torno de um pré-determinado ponto de operação (posição/velocidade) pertencente à trajetória que o manipulador seguirá durante a execução de sua tarefa, gerando as matrizes A e B.

2. Para cada estratégia de controle, linearize a equação (4.10) em torno de um conjunto de pontos de operação em torno da configuração do passo 1, gerando um segundo conjunto de matrizes A e B. Calcule $\Delta A(t)$ e $\Delta B(t)$ como sendo a variação destas matrizes em relação às matrizes A e B geradas no passo 1.

3. Calcule o ganho pelo procedimento DML.

A lei de controle aplicada às juntas ativas possui uma forma bastante simples, independente do modelo, e é dada por:

$$\tau_a = K_1(\dot{q}_c^d - \dot{q}_c) + K_2(q_c^d - q_c) \qquad (4.22)$$

sendo os ganhos K_1 e K_2 partições do ganho projetado $K_{DML} = [K_1 \;\; K_2]$. Note a simplicidade da lei de controle (4.22) em comparação à equação (4.11).

4.5 Outros Métodos de Controle

Além dos controladores CEV e DML descritos acima, os autores já implementaram controladores H_2, H_∞ e síntese μ para o controle de posição de manipuladores subatuados. Por brevidade, estes métodos não são aqui descritos, mas podem ser encontrados na bibliografia citada na Seção 10.

5 AMBIENTE DE SIMULAÇÃO E EXPERIMENTAÇÃO

A síntese de métodos de detecção de falhas e controle para sistemas robóticos pode ser facilitada utilizando-se ambientes de simulação e experimentação que integrem projeto, teste e validação dos algoritmos. O ambiente será tanto mais útil quanto mais integrados forem seus diversos componentes, notadamente o simulador e o manipulador real.

Apesar de já existirem programas comerciais para simular e controlar robôs manipuladores, os autores desconhecem a existência de programas para estudar métodos de detecção de falhas e controle de manipuladores subatuados. Para preencher esta lacuna, foi criado um ambiente de simulação e experimentação baseado no software de programação MATLAB®, que permite projetar, testar e validar métodos de controle e detecção de falhas para manipuladores subatuados (Bergerman *et al.*, 1999), (Siqueira *et al.*, 1999) e (Soares *et al.*, 1999).

O ambiente é composto por um manipulador de três ligamentos UARM II, construído por H. Ben Brown e R. Casciola de Pittsburgh, PA, EUA; um simulador gráfico através do qual os parâmetros do manipulador, o algoritmo de controle e a metodologia de detecção de falhas podem ser escolhidos; uma placa de aquisição de dados para interfacear o simulador com o manipulador; um computador pessoal; e circuitos de potência (Figuras 7 e 8). Os parâmetros do robô são dados na Tabela 1.

Tabela 1: Parâmetros do Manipulador UARM II.

Junta	massa (kg)	inércia (kg m^2)	Comprimento (m)	centro de massa (m)
1	0.730	0.0075	0,203	0,096
2	0,730	0.0075	0,203	0,096
3	0,625	0.0060	0,203	0,077

Cada junta do manipulador UARM II é equipada com um motor DC, um freio e um *encoder*, e são especialmente projetadas com atrito mínimo. Isto permite que cada junta possa ser considerada ativa em um experimento e passiva em outro, o que garante uma enorme flexibilidade de experimentação aos autores.

Figura 7. Ambiente de simulação e experimentação.

Figura 8. Robô manipulador UARM II.

A característica mais importante do ambiente é que ele permite o acesso ao robô real, isto é, seu controle em tempo real, com o simples toque de um botão. Os algoritmos desenvolvidos em Matlab, após validados em simulação, comunicam-se com o robô real de forma indistinta através da placa de aquisição de dados.

O ambiente mostra o movimento do manipulador durante o processamento do controle, seja durante uma simulação, seja durante um experimento. Falhas nas juntas podem ser introduzidas em qualquer instante durante o movimento. Os algoritmos DIF detectam a junta com problema e o ambiente reconfigura automaticamente o manipulador para uma nova lei de controle capaz de levar todas as suas juntas para os *set-points* especificados.

O ambiente tem sido desenvolvido de maneira que as diferentes escolhas para os algoritmos DIF e métodos de controle possam ser acessadas com apenas o toque de um

botão. O ambiente de simulação também permite que a inclusão de novos algoritmos de detecção e controle seja feita de maneira rápida e fácil. Na versão atual o pacote do ambiente simulador apresenta várias opções para o tipo de controlador: torque calculado, DML e CEV.

5.1 Operação do Ambiente

O acesso ao ambiente é feito dentro do MATLAB; quando carregado o mesmo aparece como na Figura 9.

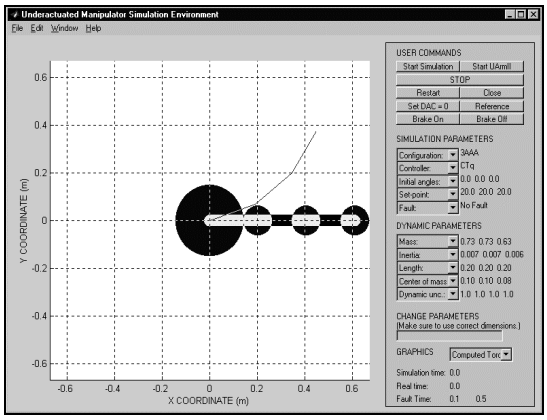


Figura 9. Interface gráfica do ambiente.

A interface gráfica é dividida em duas áreas: ***msg_frame***, onde se encontram todos os botões utilizados para entrada de comandos, mensagens e um *prompt* para entrada de dados; e ***movie_axis***, onde se encontra uma representação gráfica do manipulador. A área ***msg_frame*** é subdividida nas seguintes sub-áreas:

User Commands: Os botões desta área realizam as seguintes tarefas (Figura 10):

Figura 10. Menu de comandos.

Start Simulation: Inicia uma simulação. O botão se torna invisível durante o processamento e volta a se tornar visível com o término do movimento ou quando é pressionado o botão *Stop*.

Start UArmII: Inicia um experimento no manipulador real.

Stop: Pára a simulação em qualquer instante e retorna os botões *Start Simulation* e *Start UArmII* para seus estados visíveis.

Restart: Reinicia a interface gráfica carregando novamente todo o programa.

Close: Fecha a interface gráfica e limpa todas as variáveis do *workspace* do MATLAB.

Free Brake: Libera todos os freios do robô UARM II. Quando as juntas alcançam os seus *set-points* os freios são automaticamente acionados.

Reference: Adquire as posições atuais das juntas com a finalidade de gerar referências para o *encoder* de cada junta.

Brake On: Aciona os freios de todas as juntas. Este comando é utilizado após o manipulador ser posicionado na posição correta para se gerar a referência.

Set DAC=0: Força todas as saídas analógicas dos amplificadores em zero volts. Esta opção é utilizada como medida de segurança nos casos em que o botão de emergência é acionado, pois resíduos de tensão nas saídas podem provocar um movimento inesperado do manipulador.

Simulation Parameters: Esta área mostra os parâmetros que podem ser definidos pelo usuário. Os seguintes dados são mostrados e podem ser alterados (Figura 11):

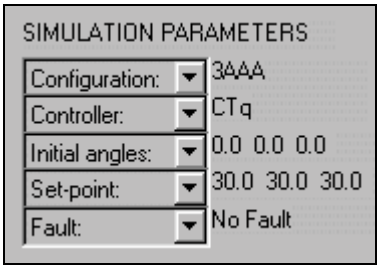


Figura 11. Menu de parâmetros.

Configuration: Define o número de juntas ativas e passivas e também sua localização no mecanismo. As configurações possíveis são definidas pelo tipo das juntas. Ex.: 3PAP representa um manipulador de 3 juntas, com a primeira e a terceira passivas e a segunda ativa.

Controller: Na configuração padrão, torque calculado e controladores PID são disponibilizados para todas as configurações. O usuário também tem a opção de utilizar um controlador C ou um controlador projetado via DML.

Initial angles: Definem os ângulos iniciais do manipulador. Para a configuração padrão de ângulos inicias o usuário deve escolher a opção *default*. Para escolher ângulos iniciais aleatórios a opção *random* deve ser selecionada. Caso o usuário queira entrar com os ângulos iniciais, o seguinte procedimento deve ser realizado: no campo de entrada de dados digita-se os ângulos iniciais como um vetor (p.e., [30 30 45]), clica-se na opção *Inital angles* e depois escolhe-se a opção *User defined*. Os ângulos devem ser entrados em graus.

Set-point: Define os ângulos desejados para as juntas. Para alterar o *set-point* dos ângulos o procedimento é o mesmo dos ângulos iniciais.

Fault: Define a junta na qual a falha ocorrerá. Pode-se escolher: *none*, *joint 1*, *joint 2* ou *joint 3*. O tempo inicial e final da falha também devem ser definidos. É necessário definir o tempo final porque se a falha não for detectada o UARM II pode ser controlado para o *set-point* após o término do tempo da falha.

Dynamic parameters: Esta área mostra os parâmetros que definem o manipulador (Figura 12). Os parâmetros padrões foram calculados para o robô experimental UARM II. Os seguintes dados são mostrados e podem ser alterados:

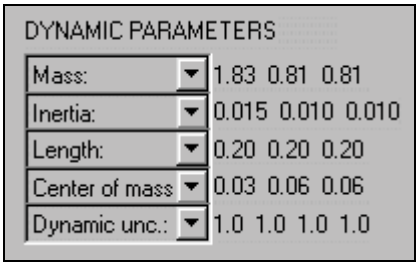


Figura 12. Parâmetros cinemáticos e dinâmicos.

- *Mass*: Define as massas dos ligamentos.
- *Inertia*: Define a inércia dos ligamentos.
- *Length*: Define o comprimento dos ligamentos.
- *Center of mass*: Define a posição dos centros de massa do ligamento.
- *Dynamic uncertainty:* Define o grau de incerteza nos parâmetros dinâmicos e cinemáticos. Quando este valor é igual a 1 o modelo é assumido com sendo perfeitamente conhecido. Quando apresenta um valor diferente, todos os parâmetros dinâmicos e cinemáticos são multiplicados por este valor e estes valores estimados são utilizados pelo controlador. Note que o modelo dinâmico simulado ainda é computado utilizando os parâmetros dinâmicos e cinemáticos nominais. Esta opção é utilizada para testar a robustez das leis de controle em relação a incertezas paramétricas.

Change Parameters: Fornece um *prompt* para a entrada numérica de dados (Figura 13). Quando um dado inválido é entrado a mensagem "*Invalid data! Default values set*" aparece.

Graphics: Este menu *pull-down* (Figura 13) mostra as opções disponíveis para a apresentação dos dados na forma gráfica. Quando uma opção é selecionada uma nova janela chamada "Graphics" é aberta. O usuário pode escolher entre várias opções de janelas de gráfico. A primeira mostra a posição, velocidade e torque das juntas para um dos controladores (torque calculado, VSC ou DML – vide Figura 14) resultantes da simulação ou adquiridos durante o experimento. A segunda opção fornece os gráficos simulados e experimentais de posição e velocidade ou posição das juntas para os três controladores. Gráficos dos resíduos de posição e velocidade do MLP, saída da RBF e o sinal de falha para cada junta são exibidos quando a terceira opção é escolhida (Figura 15).

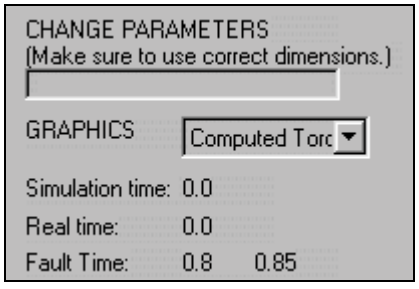


Figura 13: Menu de gráficos.

Os seguintes dados são mostrados e não podem ser alterados pelo usuário:

Simulation time: Mostra o progresso do tempo de simulação. Esta mesma informação é exibida no *workspace* do MATLAB ao término da simulação.

Real time: Exibe o progresso do tempo real de simulação ou execução de uma trajetória para o robô UARM II.

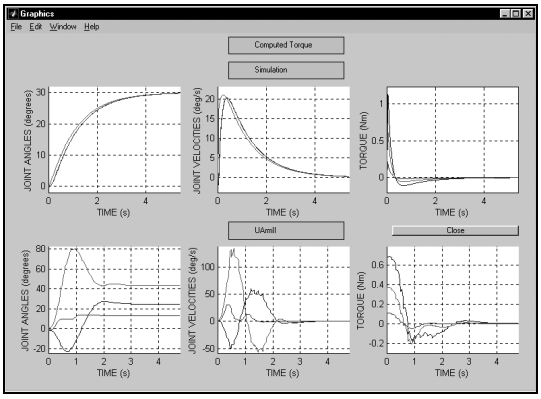


Figura 14. Janela gráfica 1.

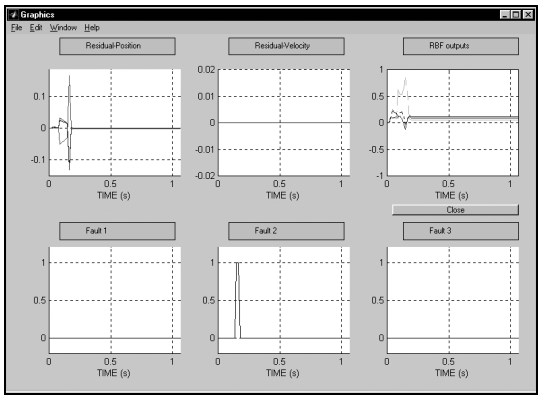


Figura 15. Janela gráfica 3.

6 ESTUDO DE CASO

Nesta seção são apresentados dois estudos de caso envolvendo todos os conceitos apresentados nas seções anteriores, para permitir ao leitor um melhor entendimento da teoria através de exemplos práticos. Nestes estudos o cenário consiste em um manipulador executando tarefas de movimentação de peças de um ponto a outro de seu espaço de trabalho. Durante a movimentação, o método de DIF permanentemente checa se falhas ocorreram ou não. Em caso positivo, todas as juntas são freadas e o método de controle pós-falha é invocado para trazer todas as juntas ao seu *set-point*.

No primeiro estudo de caso utilizamos apenas o ambiente de simulação. A tarefa do manipulador consiste em atingir o *set-point* $q^d = [30^o \;\; 30^o \;\; 30^o]$ a partir da condição inicial $q(0) = [0^o \;\; 0^o \;\; 0^o]$. Antes de atingí-lo, entretanto, uma falha na junta 2 é introduzida no instante t = 0,285s. O CEV é utilizado antes e após a falha para controle das posições das juntas.

As Figuras 16, 17, 18 e 19 mostram os resultados da simulação. As etapas da simulação são:

(a) Sem falhas (configuração AAA)

(b) Tempo de detecção da falha

(c) Bloqueio das juntas após detecção

(d) Controle da junta passiva

(e) Controle das juntas ativas

A Figura 16 mostra a saída da rede RBF durante as etapas iniciais do movimento do manipulador. O critério para DIF utilizado é o seguinte: três amostras consecutivas da saída da rede RBF têm que ser maiores que 0,5 para a falha ser detectada. Note que nesta trajetória, a falha foi rapidamente

detectada (em aproximadamente 0,1 segundo). Após a detecção da falha, utiliza-se as estratégias de controle AP e A para controlar as posições de todas as juntas. As Figuras 17, 18 e 19 mostram respectivamente as posições, as velocidades angulares e os torques nas juntas para a trajetória com falha. Após a falha ser detectada, as juntas são bloqueadas durante um curto intervalo de tempo. A seguir, a junta passiva (neste caso a junta 2) é controlada através de seu acoplamento dinâmico com a junta ativa 1. Observe que neste período a junta ativa 3 também é controlada (estratégia AP). Ao alcançar a sua posição, a junta passiva é freada e as juntas ativas 1 e 3 são controladas (estratégia A).

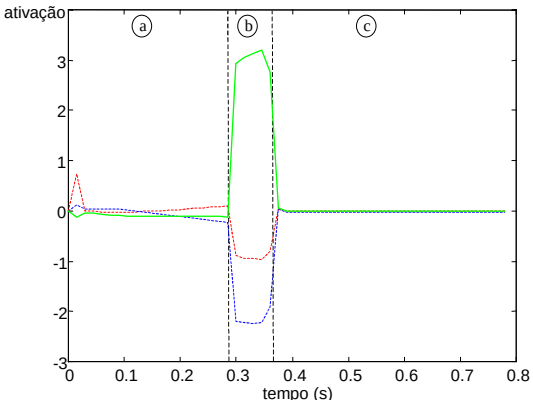


Figura 16. Saídas da rede RBF. A linha contínua representa a saída responsável pela detecção da falha na junta 2.

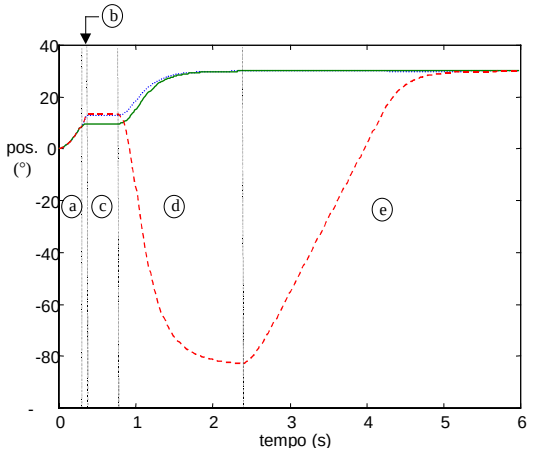


Figura 17. Posições das juntas (junta 1 - linha tracejada, junta 2 - linha contínua, junta 3 - linha pontilhada).

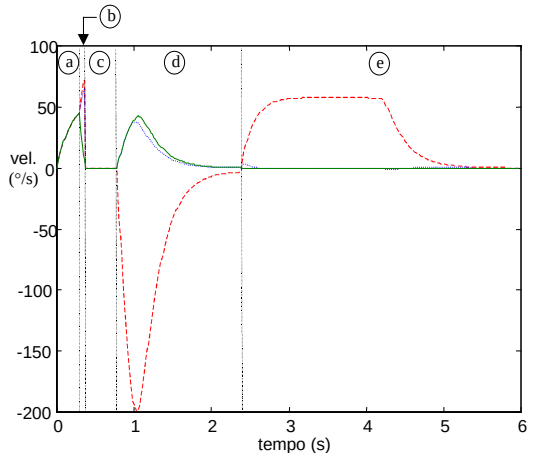


Figura 18. Velocidades das juntas (junta 1 - linha tracejada, junta 2 - linha contínua, junta 3 - linha pontilhada).

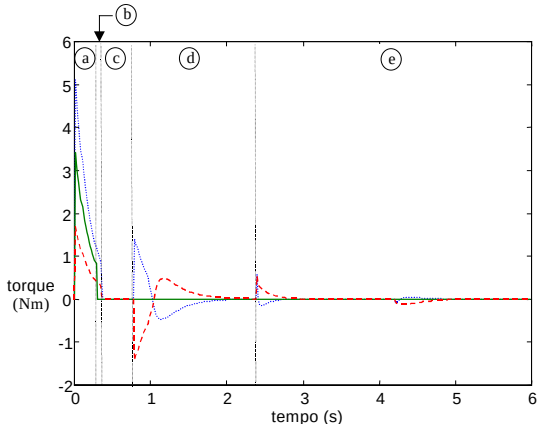


Figura 19. Torques nas juntas (junta 1 - linha tracejada, junta 2 - linha contínua, junta 3 - linha pontilhada).

Como visto acima, o módulo de DIF é capaz de detectar e isolar a falha corretamente, e o CEV atua no sentido de trazer todas as juntas para seus *set-points*.

No segundo estudo de caso, consideramos apenas o controle de posição das juntas para três configurações diferentes: AAA (todas as juntas ativas), APA (a segunda junta é passiva) e APP (somente a primeira junta é ativa). Neste estudo todos os resultados apresentados são experimentais, isto é, obtidos com o manipulador UARM II. Para os três casos as trajetórias a serem seguidas pelas juntas são polinômios de quinta ordem com condição inicial $q(0) = [0°\ \ 0°\ \ 0°]$ e posição final desejada $q^d = [20°\ \ 20°\ \ 20°]$. O controlador utilizado é o DML.

Com o objetivo de cobrir todo o espaço de trabalho do manipulador, foram projetados, para cada configuração, vários controladores, sendo cada um referente a um ponto de operação. Portanto, foi calculada uma família de plantas nominais (A, B). As matrizes de incertezas foram calculadas adotando-se uma variação em torno do ponto de operação de cada planta nominal. Após testar e analizar os controladores obtidos, foi adotado aquele que apresentou melhor desempenho para cada configuração (AAA, APA e APP). Os ganhos calculados de acordo com o projeto detalhado no Apêndice A são apresentados no Apêndice B.

As Figuras 20 a 28 apresentam as posições, velocidades e torques nas juntas para as três configurações.

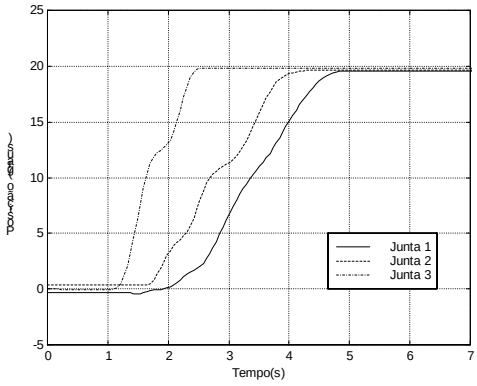


Figura 20. Controle do manipulador subatuado UArm II via DML, configuração AAA. Posição das juntas.

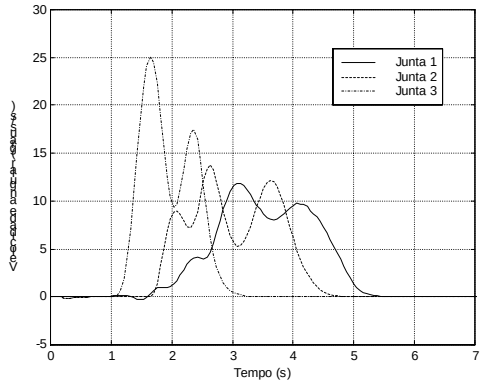


Figura 21. Controle do manipulador subatuado UArm II via DML, configuração AAA. Velocidade das juntas.

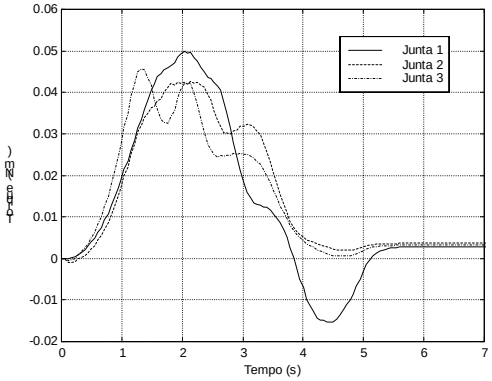


Figura 22. Controle do manipulador subatuado UArm II via DML, configuração AAA. Torques nas juntas.

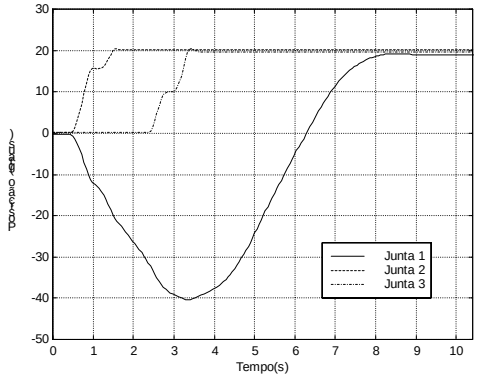


Figura 23. Controle do manipulador subatuado UArm II via DML, configuração APA. Posição das juntas.

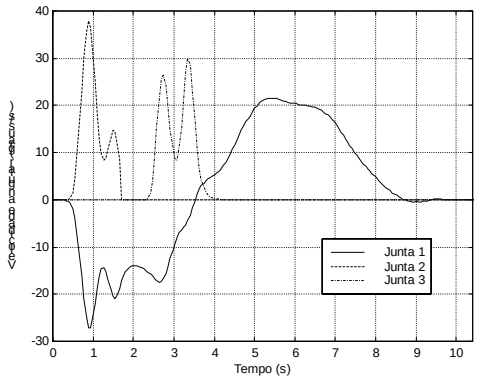


Figura 24. Controle do manipulador subatuado UArm II via DML, configuração APA. Velocidade das juntas.

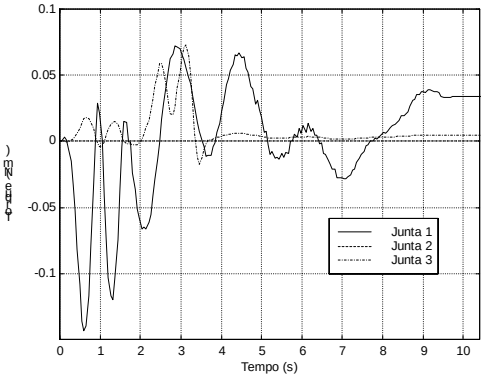


Figura 25. Controle do manipulador subatuado UArm II via DML, configuração APA. Torque nas juntas.

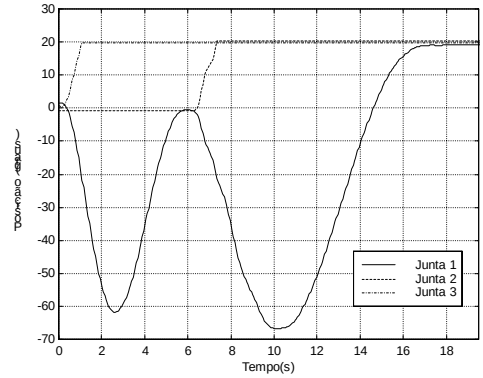


Figura 26. Controle do manipulador subatuado UArm II via DML, configuração APP. Posição das juntas.

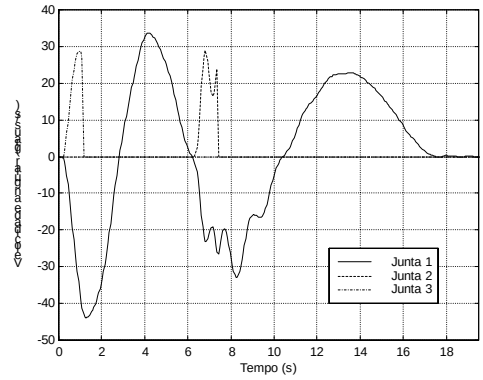


Figura 27. Controle do manipulador subatuado UArm II via DML, configuração APP. Velocidade das juntas.

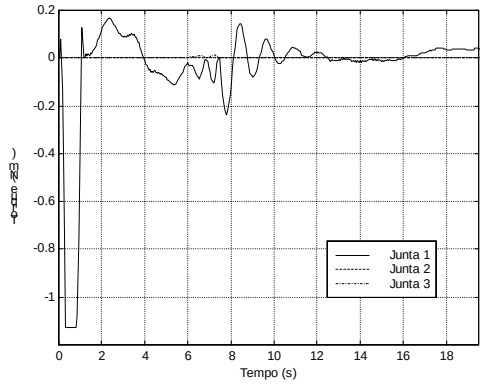


Figura 28. Controle do manipulador subatuado UArm II via DML, configuração APP. Torque nas juntas.

Como visto nas figuras acima, o controlador DML mostrou-se bastante eficiente no controle do manipulador subatuado UArm

II, pois o acompanhamento da trajetória desejada foi satisfatório para todas as configurações. Embora o controlador DML tenha sido projetado para uma variação pequena em torno da planta nominal, o controle fora desta faixa de ângulos resultou em convergência, mostrando ser o controlador DML bastante robusto.

Para testar a robustez do controlador face a distúrbios externos, foram realizados testes de perturbação tipo impulso na configuração AAA. O impulso é da ordem de dez vezes o torque no instante 15 segundos, com duração de 0,5 segundos. As Figuras 29 a 31 mostram os resultados obtidos.

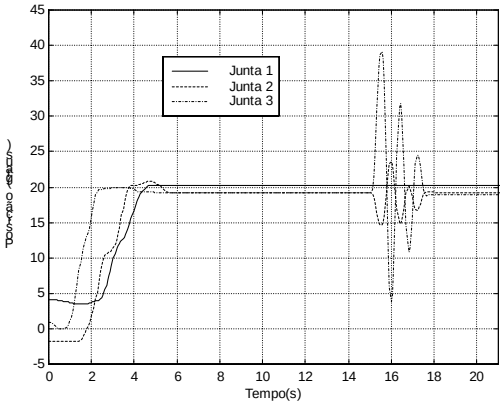


Figura 29. Teste de robustez do controlador DML. Posição das juntas.

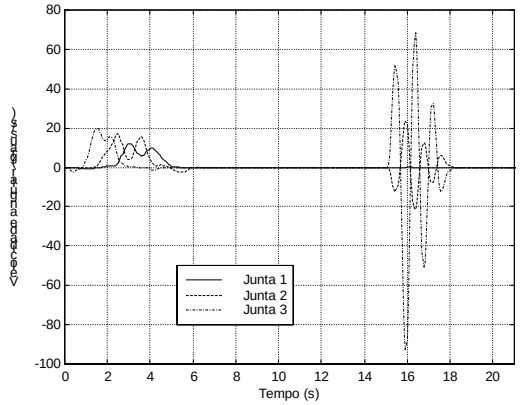


Figura 30. Teste de robustez do controlador DML. Velocidade das juntas.

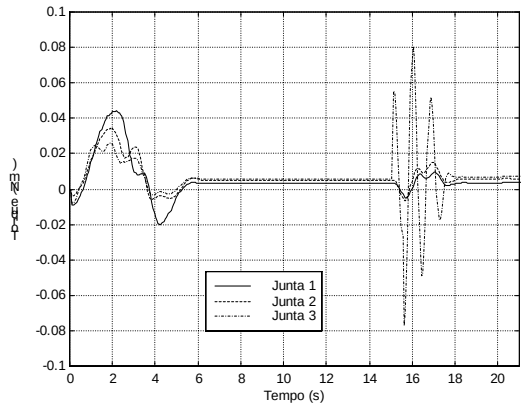


Figura 31. Teste de robustez do controlador DML. Torque nas juntas.

Nota-se mais uma vez a robustez do controlador, que é capaz de lidar com o distúrbio externo trazendo as juntas de volta ao seu *set-point*.

7 CONCLUSÃO

A crescente demanda pelo uso de robôs manipuladores em ambientes não industriais exige do usuário atenção para questões de segurança e tolerância a falhas. Este artigo apresentou uma metodologia integrada de detecção e isolação de falhas e controle pós-falha de robôs manipuladores. Mesmo após a ocorrência de uma ou mais falhas, é possível controlar as posições de todas as juntas do robô, inclusive aquelas que falharam, o que permite fazer com que o efetuador atinja uma posição Cartesiana desejada. No futuro, os autores intencionam aplicar estas técnicas em manipuladores atuando em ambientes inóspitos, como na exploração de petróleo no fundo do mar.

8 AGRADECIMENTOS

Este trabalho recebe apoio parcial da FAPESP através dos processos no. 97/13384-7, 98/15732-5, 98/00723-0 e 99/10031-1. Este artigo foi originalmente apresentado como um minicurso no XIII Congresso Brasileiro de Automática, em Florianópolis, SC, em 10 de setembro de 2000.

9 REFERÊNCIAS

Arai, H. e Tachi, S. (1991). "Position control of a manipulator with passive joints using dynamic coupling". *IEEE Transactions on Robotics and Automation*, **7**(4), p. 528-534.

Bergerman, M. (1996). *Dynamics and control of underactuated manipulators*. Tese de Doutorado, Electrical and Computer Engineering Department, Carnegie Mellon University, Pittsburgh, EUA.

Bergerman, M.; Lee, C.; Xu, Y. (1995) "A dynamic coupling index for underactuated manipulators." *Journal of Robotic Systems*, vol. 12, no. 10, pp. 693-707.

Bergerman, M.; Xu, Y. (1998) "Optimal control of manipulators with any number of passive joints.'' *Journal of Robotic Systems*, vol. 15, no. 3, pp. 115-130.

Bergerman, M. e Xu, Y. (1996). "Robust joint and Cartesian control of underactuated manipulators." *Transactions of the ASME, Journal of Dynamic Systems, Measurement, and Control*, **118**(3), p. 557-565.

Bergerman, M.; Xu, Y. (1997) "Planning collision-free motions for underactuated manipulators in constrained configuration space." *IEEE International Conference on Robotics and Automation*, Albuquerque, NM, USA, pp. 549-555.

Bergerman, M., Siqueira, A. A. G. e Terra, M. H. (1999). "Underactuated Manipulator Control System Development Environment.'' *15th International Conference on CAD/CAM, Robotics, & Factories of the Future*, Águas de Lindóia, SP, Brazil.

Bergerman, M; Terra, M.H.; Tinós, R.; Siqueira, A A G; Xu, Y. and Loi-Wah, S. (2000) "Fault Tolerant Control of Mechanical Manipulators: A Hybrid Systems

Approach.'' *IFAC Symposium on Robot Control*, Vienna, Austria, pp. 85-90.

Boyd, S.; Ghaoui, L. El; Feron, E. e Balakrishnan, V. (1994). *Linear Matrix Inequalities in System and Control Theory*. SIAM - Society for Industrial and Applied Mathematics.

Cybenko, G. (1989). "Approximation by superpositions of a sigmoidal function." *Mathematics of Control, Signals, and Systems*, **2**, p. 303-314.

Dhillon, B. S. (1991). *Robotic reliability and safety*. Springer-Verlag, New York.

Dhillon, B. S. e Fashandi, A. R. M. (1997). "Robotic systems probabilistic analysis." *Microelectronics and Reliability*, **37**(2), p. 211-224.

Efrati, H. F. T. (1997). "Tracking of mechanical systems using artificial neural networks." *Revista Brasileira de Ciências Mecânicas*, **19**(2), p. 217-227.

English, J. D. e Maciejewski, A. A. (1998). "Fault tolerance for kinematically redundant manipulators: anticipating free-swinging joint failures." *IEEE Transactions on Robotics and Automation*, **14**(4), p. 566-575.

Groom, K. N., Maciejewski, A. A. e Balakrishnan, R. (1999). "Real-time failure-tolerant control of kinematically redundant manipulators." *IEEE Transactions on Robotics and Automation*, **5**(6), p. 1109-1115.

Haykin, S. S. (1994). *Neural networks: a comprehensive foundation*. Macmillan, New York.

Hertz, J.; Krogh, A. e Palmer, R. G. (1991). *Introduction to the theory of neural computation*. Addison-Wesley Publishing Company.

Horak, D. T. (1988). "Failure detection in dynamic systems with modeling errors", *AIAA Journal of Guidance, Control and Dynamics*, **11**(6), p. 508-516.

Hung, J.Y., Gao, W. e Hung, J.C. (1993) "Varible Structure Control: A Survey." *IEEE Transactions on Industrial Eletronics*, **40**(1), p. 2-22.

Jain, A. e Rodriguez, G. (1993) "An analysis of the kinematics and dynamics of underactuated manipulators." *IEEE Trans. on Robotics and Automation*, **9**(4), pp. 411-422.

Kohonen, T. (1995). *Self-organizing maps*. Springer-Verlag, Berlin.

Leonard, J. A. e Kramer, M. A. (1991). "Radial basis function networks for classifying process faults", *IEEE Control Systems Magazine*, **11**(3), pp. 31-38.

Lewis, C. L. e Maciejewski, A. A. (1997). "Fault tolerant operation of kinematically redundant manipulators for locked joint failures." *IEEE Transactions on Robotics and Automation*, **13**(4), p. 622-629.

Looney, C. G. (1997). *Pattern recognition using neural networks*. Oxford University Press.

Moody, J. e Darken, C. (1989). "Fast learning in networks of locally-tuned processing units." *Neural Computation*, **1**, p. 289-303.

Mukherjee, R. e Chen, D. (1993). "Control of free-flying underactuated space manipulators to equilibrium manifolds." *IEEE Trans. on Robotics and Automation*, **9**(5), p. 561-570.

Naughton, J. M., Chen, Y. C. e Jiang, J. (1996). "A neural network aplication to fault diagnosis for robotic manipulator." *IEEE Int. Conference on Control Applications*, p. 988-1003.

Nelles, O. e Isermann, R. (1995). "A comparison between RBF networks and classical methods for identification of nonlinear dynamic systems." *Proceedings of the IFAC Congress on Adaptive System in Control and Sygnal Processing*, Budapest, Hungary, p. 233-238.

Ojala, T. e Vuorimaa, P. (1995). "Modified Kohonen's learning laws for RBF networks." *Proceedings of the International Conference on Artificial Neural Nets and Genetic Algorithms*, Ales, France, p. 356-359.

Oriolo, G. e Nakamura, Y. (1991). "Free-joint manipulators: motion control under second-order nonholonomic constraints." *IEEE International Workshop on Intelligent Robots and Systems*, p. 1248-1253.

Orr, M. J. L. (1996). *Introduction to radial basis function networks.* Technical Report, Center for Cognitive Science, Edinburgh University, Scotland, U. K.

Papadopoulos, E. e Dubowsky, S. (1991). "Failure recovery control for space robotic systems." *American Control Conference – ACC 91*, p. 1485-1490.

Paredis, C. J. J. e Khosla, P. K. (1996a). "Designing Fault Tolerant Manipulators: How Many Degrees-of-Freedom?'' *International Journal of Robotics Research*, **15**(6), p. 611-628.

Paredis, C. J. J. e Khosla, P. K. (1996b). "Fault Tolerant Task Execution through Global Trajectory Planning.'' *Reliability Engineering and System Safety (special issue on Safety of Robotic Systems)*, **53**(3), p. 225-235.

Schneider, H. e Frank, P. M. (1996). "Observer-based supervision and fault-detection in robots using nonlinear and fuzzy logic residual evaluation." *IEEE Transactions on Control Systems Technology*, **4**(3), p. 274-282.

Siqueira, A.A.G.; Bergerman, M.; Terra, M.H. (1999) "Underactuated manipulator control system development environment." *International Conference on CAD/CAM, Robotics, and Factories of the Future*, Águas de Lindóia, SP, Brazil, pp. RW2-13 a RW2-18.

Soares, M. R.; Terra, M.H.; Bergerman, M. e Tinós, R. (1999). "An Environment for Simulation of Underactuated Manipulator Robots with Fault Detection and Control." *15th International Conference on CAD/CAM, Robotics, and Future*, Águas de Lindóia, SP, Brazil, pp. RW5-13 a RW5-17.

Stengel, R. F. (1991). "Intelligent failure-tolerant control." *IEEE Control Systems*, **11**(4), p. 14-23.

Terra, M.H.; Maciel, B.C.O.; Nakashima, P.H.R.; Bergerman, M. (2000) "Underactuated manipulator robot control by state feedback linearization via H_∞." *IFAC Symposium on Robust Control Design 2000*, Prague, Czech Republic.

Terra, M. H. and Tinos, R. (1998) "Fault Detection and Isolation for Robotic Systems Using a Multilayer Perceptron and a Radial Basis Function." *IEEE International Conference on Systems, Man, and Cybernetics*, San Diego, USA.

Terra, M. H. and Tinós, R. (1999). "Fault detection and isolation in a Puma 560 manipulator via neural networks." *Proc. of the 14th Word Congress of IFAC*, Beijiing, China.

Terra, M. H. and Tinós, R. (2000) "Fault detection and isolation in robotic systems via artificial neural networks – b." *Symposium on Fault Detection, Supervision, and Safety for Technical Processes*, Budapest, Hungary, p. 180-185.

Terra, M. H. and Tinós, R. (2001) " Fault Detection and Isolation in Robotic Manipulators via Neural Networks - A Comparison Among Three Architectures for Residual Analysis.'' *Journal of Robotic Systems* (to appear - July 2001).

Terra, M. H.; Siqueira, A. A. G. ; Bergerman, M. (1999) "Underactuated Manipulator Robot Control via Linear Matrix Inequalities.'' *39th IEEE Conference on Decision and Control -CDC 99*.

Tinós, R. e Terra, M. H. (1998). "Fault detection and isolation in robotic manipulators and the RBF network trained by the Kohonen's SOM." *Anais do 5º Simpósio Brasileiro em Redes Neurais*, Belo Horizonte, Brasil, p. 85-90.

Tinós, R. and Terra, M. H. (2001) "Fault Detection and Isolation in Robotic Manipulators Using a Multilayer Perceptron and a RBF Network Trained by the Kohonen's Self-Organizing Map.'' *Revista Controle & Automação*, vol. 12, n. 1, p. 11-18.

Tinós, R.; Terra, M.H.; Bergerman, M. (2000) "A fault tolerance system for robotic manipulators with free-swinging joint failures." *Symposium on Fault Detection*, Supervision, *and Safety for Technical Processes*, Budapest, Hungary, p. 840-845.

Tinós, R.; Terra, M.H.; Bergerman, M.; Soares, M.R. (1999) "Post-failure control reconfiguration for manipulator robots." *4º Simpósio Brasileiro de Automação Inteligente*, São Paulo, SP, Brazil, pp. 107-112.

Tinós, R. (1999). "*Detecção e diagnóstico de falhas em robôs manipuladores via redes neurais artificiais*", Dissertação de Mestrado, Escola de Engenharia de São Carlos, USP.

Vemuri, A. T. e Polycarpou, M. M. (1997). "Neural-network-based robust fault diagnosis in robotic systems." *IEEE Transaction on Neural Networks*, **8**(6), p. 1410-1420.

Visinsky, M. L., Cavallaro, J. R. e Walker, I. D. (1994). "Robotic fault detection and fault tolerance: a survey." *Reliability Eng. and System Safety*, **46**, p. 139-158.

Visinsky, M. L., Cavallaro, J. R. e Walker, I. D. (1995). "A dynamic fault tolerance framework for remote robots." *IEEE Transactions on Robotics and Automation*, **11**(4), p. 477-490.

Warwick, K. e Craddock, R. (1996). "An introduction to radial basis function for system identification: A comparison with other neural networks." *Conference on Decision and Control – CDC 96*.

APÊNDICE A: PROJETO DO CONTROLADOR DML

O controlador projetado via DML define o comportamento do torque da seguinte maneira:

$$\tau_a = K_1(\dot{q}_c^d - \dot{q}_c) + K_2(q_c^d - q_c) \tag{A1}$$

sendo K_1 e K_2 os ganhos calculados da seguinte maneira (estes ganhos multiplicam os erros de posição e velocidade):

$$K_{DML} = [K_1 \;\; K_2] = \frac{1}{\varepsilon_c} R_c^{-1} B' P_c \tag{A2}$$

O método DML pode ser resumido como segue. Considere o sistema linear sujeito a incertezas que definem, neste caso, pontos de operação do sistema,

$$\begin{aligned} \dot{x}(t) &= (A + \Delta A(t))x(t) + (B + \Delta B(t))u(t) \\ y(t) &= (C + \Delta C(t))x(t) \end{aligned} \tag{A3}$$

sendo $x(t)$, $u(t)$ (K_{DML},, $x(t)$), $y(t)$, A, B e C os estados (posições e velocidades), entradas, saídas e as respectivas matrizes constantes com dimensões apropriadas para cada configuração do robô subatuado (AAA, AAP, APP, etc.), e

$$\begin{aligned} \Delta A(t) &= \sum_{i=1}^{p} \alpha_i(t) A_i \\ \Delta C(t) &= \sum_{i=1}^{r} \chi_i(t) C_i \,. \end{aligned} \tag{A4}$$

As funções escalares $\alpha_i(t)$, $\beta_i(t)$ e $\chi_i(t)$ são mensuráveis segundo Lebesgue:

$$|\alpha_i(t)|, |\beta_i(t)|, |\chi_i(t)| \le 1 \tag{A5}$$

A_i, B_i e C_i são matrizes conhecidas, assumidas constantes e de posto 1, e são decompostas da seguinte maneira:

$$A_i = d_i \, o_i' \,, \quad B_i = f_i \, g_i' \,, \quad C_i = h_i \, j_i' \,. \tag{A6}$$

Os escalares v_i e s_i são definidos para B_i e Ci, respectivamente. Matrizes constantes T, W, S, U, V e Y representam o modelo variante no tempo, formam o limite superior das variações dos modelos lineares que são definidos a cada instante, neste caso em função das velocidades e posições do robô manipulador.

$$T \triangleq \sum_{i=1}^{p} l_i \, d_i \, d_i' = D \hat{\mathrm{L}} \, D'$$

$$W \triangleq \sum_{i=1}^{q} v_i \;\; f_i \; f_i' = F \hat{\mathrm{V}} \, F'$$

$$S \triangleq \sum_{i=1}^{r} s_i \; h_i \, h_i' = H \hat{S} \, H'$$

$$U \triangleq \sum_{i=1}^{p} l_i^{-1} o_i \, o_i' = O' \hat{\mathrm{L}}^{-1} O$$

$$V \triangleq \sum_{i=1}^{q} v_i^{-1} g_i g_i' = G' \hat{\mathrm{V}}^{-1} G$$

$$Y \triangleq \sum_{i=1}^{r} s_i^{-1} j_i j_i' = J' \hat{S}^{-1} J \qquad \text{(A7)}$$

sendo

$$D \triangleq [\, d_1 \ \ldots \ d_p]$$

$$F \triangleq [\, f_1 \ \ldots \ f_q]$$

$$H \triangleq [\, h_1 \ \ldots \ h_r]$$

$$O \triangleq [\, o_1 \ \ldots \ o_p]$$

$$G \triangleq [\, g_1 \ \ldots \ g_q]$$

$$J \triangleq [\, j_1 \ \ldots \ j_r]$$

$$\hat{\mathrm{L}} \triangleq diag\,(\hat{l}_1 \ \ldots \ \hat{l}_p)$$

$$\hat{\mathrm{V}} \triangleq diag\,(\hat{v}_1 \ \ldots \ \hat{v}_q)$$

$$\hat{S} \triangleq diag\,(\hat{s}_1 \ \ldots \ \hat{s}_r).$$

Se existirem constantes positivas δ_c e matrizes simétricas definidas positivas W_c, R_c, e Q_c, e matrizes diagonais definidas positivas $\tilde{L}$, $\tilde{S}$, $\tilde{\mathrm{V}}$ tal que a seguinte DML é satisfeita

$$\Lambda_c \triangleq \begin{bmatrix} \Phi_c & 2\delta_c B R_c^{-1} G' & W_c O' & W_c J' & W_c \\ G R_c^{-1} B' \delta_c 2 & \tilde{V} & 0 & 0 & 0 \\ O W_c & 0 & \tilde{L} & 0 & 0 \\ J W_c & 0 & 0 & \tilde{S} & 0 \\ W_c & 0 & 0 & 0 & \delta_c Q_c^{-1} \end{bmatrix}$$

> 0

(A8)

sendo

$$\Phi_c = -W_c A' - A W_c + 2\delta_c B R_c^{-1} B' - F \tilde{V} F' - 2 D \tilde{L} D'$$

$$P_c \triangleq W_c^{-1}$$

$$\hat{L} \triangleq \tilde{L}^{-1}$$

$$\hat{V} \triangleq \tilde{V}^{-1}$$

$$\hat{S} \triangleq \tilde{S}^{-1}$$

então o sistema linear variante no tempo, realimentado com a lei de controle K_{DML} definida na Equação (A2), é assintoticamente estável.

O procedimento de projeto DML é definido da seguinte maneira:

1. Escolher as matrizes Q_c e R_c, tal que o seguinte problema de otimização P1 tenha um conjunto de soluções não vazio (M_c, W_c, $\tilde{V}$, $\tilde{L}$, $\tilde{S}$, δ_c), sendo M_c e W_c matrizes simétricas positivas definidas, $\tilde{V}$, $\tilde{L}$ e $\tilde{S}$ matrizes diagonais definidas positivas e δ_c um escalar.

$$\text{P1: min } f_c(M_c, W_c, \tilde{V}, \tilde{L}, \tilde{S}, \delta_c) = \mathrm{tr}(M_c) \qquad \text{(A9)}$$
$$M_c, W_c, \tilde{V}, \tilde{L}, \tilde{S}, \delta_c$$

sujeito a

$$\begin{bmatrix} M_c & I \\ I & W_c \end{bmatrix} > 0,\ \Lambda_c > 0,\ M_c, W_c, \tilde{V}, \tilde{L}, \tilde{S}, \delta_c > 0 \qquad \text{(A10)}$$

2. Calcular K_{DML} pela Equação A2. A Equação (A8) garante estabilidade assintótica para o sistema (A3) com um controle por realimentação de estados.

APÊNDICE B: GANHOS DO CONTROLADOR DML

Configuração AAA

$$K_{AAA} = \begin{bmatrix} 0.1826 & 0.1804 & 0.0635 & 0.1728 & 0.0131 & 0.00014 \\ 0.1443 & 0.2612 & 0.0874 & 0.0263 & 0.0277 & 0.0055 \\ 0.1533 & 0.2013 & 0.2458 & 0.0374 & 0.0245 & 0.0022 \end{bmatrix}$$

Configuração APA

$$K_P = \begin{bmatrix} -2.1314 & -0.001166 \end{bmatrix}$$

$$K_A = \begin{bmatrix} 2.0587 & 0.3090 & 0.5326 & 0.0434 \\ 0.0984 & 0.4790 & 0.0262 & 0.0171 \end{bmatrix}$$

Configuração APP

$$K_{P3} = \begin{bmatrix} -11.9679 & -0.7726 \end{bmatrix}$$

$$K_{P2} = \begin{bmatrix} -2.1314 & -0.001166 \end{bmatrix}$$

$$K_A = \begin{bmatrix} 2.3605 & 0.9593 \end{bmatrix}.$$