Campus de São Carlos

MODELOS DE MEMÓRIA ASSOCIATIVA EM REDES NEURAIS
PARA PLANEJAMENTO E CONTROLE PONTO A PONTO
DE TRAJETÓRIA DE UM BRAÇO MECÂNICO

Autor: Marcelo Vieira

Orientador: Prof.Dr. Aluizio F.R. Araújo

UNIVERSIDADE DE SÃO PAULO

**ESCOLA DE ENGENHARIA
DE SÃO CARLOS**

✓

**MODELOS DE MEMÓRIA ASSOCIATIVA EM REDES
NEURAIS PARA PLANEJAMENTO E CONTROLE PONTO A
PONTO DE TRAJETÓRIA DE UM BRAÇO MECÂNICO**

Marcelo Vieira

Dissertação apresentada à escola de
Engenharia de São Carlos da
Universidade de São Paulo, como parte
dos requisitos para a obtenção do título
de Mestre em Engenharia Elétrica

ORIENTADOR: Prof. Dr. Aluizio
Fausto Ribeiro Araújo

São Carlos

1997

Class.	TESE - EESC
Curr.	6250
Arquivo	T 0029/98

S/S 942974

Ficha catalográfica preparada pela Seção de Tratamento
da Informação do Serviço de Biblioteca - EESC-USP

V658m Vieira, Marcelo
Modelos de memória associativa em redes neurais para planejamento e controle ponto a ponto de trajetória de um braço mecânico / Marcelo Vieira. -- São Carlos, 1997.

Tese (Mestrado). -- Escola de Engenharia de São Carlos-Universidade de São Paulo, 1997.
Área: Engenharia Elétrica.
Orientador: Prof. Dr. Aluizio Fausto Ribeiro Araújo.

1. Memória associativa multidirecional temporal. 2. Manipulador mecânico. 3. Redes neurais. 4. Geração de trajetória. 5. Cinemática inversa. 6. Meios não estruturados. I. Título

FOLHA DE APROVAÇÃO

Candidato: Engenheiro **MARCELO VIEIRA**

Dissertação defendida e aprovada em 12-12-1997
pela Comissão Julgadora:

Prof. Doutor **ALUIZIO FAUSTO RIBEIRO ARAUJO (Orientador)**
(Escola de Engenharia de São Carlos - Universidade de São Paulo)

Prof. Titular **ÁLVARO GERALDO BADAN PALHARES**
(Universidade Estadual de Campinas - UNICAMP)

Prof. Associado **FERNANDO ANTONIO CAMPOS GOMIDE**
(Universidade Estadual de Campinas - UNICAMP)

---

p/ Prof. Doutor **DENIS VINICIUS COURY**
Coordenador da Área de Engenharia Elétrica

JOSÉ CARLOS A CINTRA
Presidente da Comissão de Pós-Graduação

AGRADECIMENTOS

Ao Professor e amigo Dr. Aluizio Fausto Ribeiro Araújo, pela excelente orientação fornecida durante a elaboração deste trabalho.

À Fundação de Amparo a Pesquisa do Estado de São Paulo - FAPESP, pelo auxílio financeiro fornecido no período da realização deste trabalho.

Aos amigos, Hélio Dárbo Junior, Guilherme A. Barreto, Renato Tinos e Arthur P. S. Braga que tornaram o laboratório um local de trabalho sério, ao mesmo tempo sabendo dosar os momentos de descontração.

A todos os professores e funcionários de Departamento de Engenharia Elétrica da EESC/USP que estiveram sempre dispostos a colaborar.

A todos, muito obrigado.

Dedico este trabalho aos meus pais, irmãs, avós, Patricia Policarpo e amigos, por apoiarem-me constantemente durante minha caminhada.

SUMÁRIO

LISTA DE FIGURAS.....	i
LISTA DE TABELAS.....	vii
LISTA DE ABREVIATURAS E SIGLAS.....	viii
LISTA DE SÍMBOLOS.....	ix
RESUMO.....	xi
<i>ABSTRACT</i>	xii
1. INTRODUÇÃO.....	1
1.1. OBJETIVO.....	3
2. DESCRIÇÃO DO PROBLEMA.....	5
2.1. DESCRIÇÃO DA TAREFA	5
2.2. FORMALIZAÇÃO DO PROBLEMA.....	6
3. REDES NEURAIS EM CONTROLE SENSO-MOTOR.....	10
3.1. FUNDAMENTOS DE REDES NEURAIS.....	10
3.2. TRABALHOS RELEVANTES SELECIONADOS.....	15
3.3. CLASSIFICAÇÃO DAS DIFERENTES ABORDAGENS.....	22
4. MODELOS DE REDES NEURAIS PARA MEMÓRIA ASSOCIATIVA.....	25
4.1. DEFINIÇÃO DE MEMÓRIA ASSOCIATIVA.....	25
4.2. PRINCIPAIS MODELOS DE MEMÓRIA ASSOCIATIVA.....	27
4.2.1. PRIMEIROS MODELOS.....	27
4.2.2. MODELOS COM AUTO-RELAXAMENTO.....	33
4.3. FUNÇÃO DE BASE RADIAL (RBF).....	45
4.3.1. FUNÇÃO BASE RADIAL.....	45

4.3.2. REDES DE FUNÇÃO BASE RADIAL (RBF).....	46
5. UM SISTEMA NEURAL EM 2 PLANOS.....	50
5.1. CENÁRIO DO PROBLEMA.....	50
5.1.1. MANIPULADOR PUMA560.....	52
5.2. PROPONDO UM MODELO.....	56
5.3. MODELO PARA O PLANO SEQUÊNCIA TEMPORAL.....	59
5.3.1. ALGORITMOS DE APRENDIZAGEM.....	61
5.4. MODELO PARA O PLANO ÂNGULO.....	62
6. SIMULAÇÕES.....	66
6.1. OBTENÇÃO DOS DADOS PARA SIMULAÇÃO.....	66
6.2. TESTANDO DO PLANO SEQUÊNCIA TEMPORAL.....	69
6.2.1. TESTES COM TRAJETÓRIA ÚNICA.....	69
6.2.2. TESTES COM TRAJETÓRIAS MÚLTIPLAS.....	77
6.3. TESTES DO PLANO ÂNGULO.....	90
6.3.1. TESTES COM TRAJETÓRIA ÚNICA.....	91
6.3.2. TESTES COM MÚLTIPLAS TRAJETÓRIAS.....	93
7. CONCLUSÕES.....	98
8. REFERÊNCIAS BIBLIOGRÁFICAS.....	102

LISTAS DE FIGURAS

FIGURA 01 - Representação esquemática dos problemas de cinemática direta e inversa do braço mecânico (Fu et al., 1987).....	7
FIGURA 02 - Diagrama do planejador de trajetórias do braço (Fu et al., 1987).....	8
FIGURA 03 - Esquema simplificado de um neurônio.....	11
FIGURA 04 - Modelo chamado Perceptron.....	12
FIGURA 05 - Robô manipulador bidimensional com dois graus de liberdade.....	15
FIGURA 06 - Diagrama de blocos das etapas do modelo computacional para planejamento e controle da trajetória de um braço de robô, baseado em Kawato (1991).....	18
FIGURA 07 - Modelo da arquitetura de Amari.....	27
FIGURA 08 - (a) Configuração de uma entidade formada por dois padrões e uma área neutra, (b) conceito representado por cada padrão.....	30
FIGURA 09 - Recuperação das: (a) Entidade 1 apresentando um padrão ruidoso; (b) entidade 1 e entidade 2 apresentando um padrão comum a ambas.....	31
FIGURA 10 - Padrões com partes em comum que formam uma sequência no tempo.....	32
FIGURA 11 - Modelo de rede de Hopfield.....	33
FIGURA 12 - Modelo de BAM proposto por Kosko.....	36

FIGURA 13 - Processo de convergência de BAM, onde W^T é a transposta de W	37
FIGURA 14 - Arquitetura de MAM composta por cinco camadas.....	41
FIGURA 15 - Arquitetura padrão de uma rede de função radial de base.....	47
FIGURA 16 - Curvas de nível circulares.....	48
FIGURA 17 - Diagrama de blocos das etapas do modelo computacional para planejamento e controle da trajetória de um braço de robô, baseado em Kawato (1991).....	51
FIGURA 18 - Fotografia do braço do robô PUMA560.....	52
FIGURA 19 - Esquema dos eixos de rotação.....	53
FIGURA 20 - Esboço do modelo proposto para controle e planejamento de trajetória.....	58
FIGURA 21 - Conexões da camada 1 do modelo de TMAM para as outras quatro camadas e a ela mesma.....	59
FIGURA 22 - Arquitetura de conexões entre o plano 1 e plano 2.....	63
FIGURA 23 - Esboço de trajetória do tipo 2-1.....	67
FIGURA 24 - Esboço de trajetória do tipo 1-2.....	67
FIGURA 25 - Esboço de trajetória do tipo 1-1 sem pontos de contato entre elas.....	68

FIGURA 26 - Esboço de trajetória do tipo 1-1 com pontos intermediários em comum.....	68
FIGURA 27 - Recuperação dos pontos treinados da sequência temporal para: (a) sequência 1; (b) sequência 2 e (c) sequência 3.....	72
FIGURA 28 - Recuperação dos pontos interpolados da sequência temporal para: (a) sequência 1; (b) sequência 2 e (c) sequência 3.....	72
FIGURA 29 - Recuperação dos pontos interpolados e extrapolados treinados da sequência temporal para: (a) sequência 1; (b) sequência 2 e (c) sequência 3.....	73
FIGURA 30 - Recuperação dos pontos treinados da sequência temporal para: (a) sequência 1; (b) sequência 2 e (c) sequência 3.....	74
FIGURA 31 - Recuperação dos pontos interpolados da sequência temporal para: (a) sequência 1; (b) sequência 2 e (c) sequência 3.....	74
FIGURA 32 - Recuperação dos pontos interpolados e extrapolados treinados da sequência temporal para: (a) sequência 2 e (b) sequência 3.....	75
FIGURA 33 - Geração das sequências: (a) sequência 1; (b) sequência 2 e (c) sequência 3.....	76
FIGURA 34 - Sequência 4. (a) recuperação; (b) interpolação e (c) extrapolação.....	76
FIGURA 35 - Sequência 5. (a) recuperação; (b) interpolação e (c) extrapolação.....	77
FIGURA 36 - Testes de produção das trajetórias com ponto inicial treinado e ponto final treinado: (a) trajetória 1 e (b) trajetória 2.....	79

FIGURA 37 - Testes de produção das trajetórias com ponto inicial treinado e ponto final ruidoso (a) trajetória 1 e (b) trajetória 2.....	80
FIGURA 38 - Testes de recuperação das trajetórias com ponto inicial ruidoso e ponto final treinado: (a) trajetória recuperada.....	80
FIGURA 39 - Testes de recuperação das trajetórias com ponto inicial ruidoso e final ruidoso: (a) trajetória recuperada.....	80
FIGURA 40 - Testes de produção das trajetórias com ponto inicial treinado e ponto final treinado: (a) trajetória 1 e (b) trajetória 2.....	81
FIGURA 41 - Testes de produção das trajetórias com ponto inicial ruidoso e ponto final treinado: (a) trajetória 1 e (b) trajetória 2.....	82
FIGURA 42 - Testes de produção das trajetórias com ponto inicial treinado e ponto final ruidoso: (a) trajetória 1 e (b) trajetória 2.....	82
FIGURA 43 - Testes de produção das trajetórias com ponto inicial ruidoso e ponto final ruidoso: (a) trajetória recuperada.....	83
FIGURA 44 - Testes de produção das trajetórias com ponto inicial e ponto final treinados: (a) trajetória 1 e (b) trajetória 2.....	84
FIGURA 45 - Testes de produção das trajetórias com ponto inicial ruidoso e ponto final treinado: (a) trajetória 1 e (b) trajetória 2.....	84
FIGURA 46 - Testes de produção das trajetórias com ponto inicial treinado e ponto final ruidoso: (a) trajetória 1 e (b) trajetória 2.....	85

FIGURA 47 - Testes de produção das trajetórias com ponto inicial ruidoso e ponto final ruidoso: (a) trajetória.....	85
FIGURA 48 - Testes de produção das trajetórias com ponto inicial treinado e ponto final treinado: (a) trajetória 1 e (b) trajetória 2.....	86
FIGURA 49 - Testes de produção das trajetórias com ponto inicial ruidoso e ponto final treinado: (a) trajetória 1 e (b) trajetória 2.....	87
FIGURA 50 - Testes de produção das trajetórias com ponto inicial treinado e ponto final ruidoso: (a) trajetória 1 e (b) trajetória 2.....	87
FIGURA 51 - Testes de produção das trajetórias com ponto inicial ruidoso e ponto final ruidoso: (a) trajetória.....	87
FIGURA 52 - Testes de produção das trajetórias com ponto inicial treinado e ponto final treinado: (a) trajetória 1 e (b) trajetória 2.....	88
FIGURA 53 - Testes de produção das trajetórias com ponto inicial ruidoso e ponto final treinado: (a) trajetória 1 e (b) trajetória 2.....	89
FIGURA 54 - Testes de produção das trajetórias com ponto inicial treinado e ponto final ruidoso: (a) trajetória 1 e (b) trajetória 2.....	89
FIGURA 55 - Testes de produção das trajetórias com ponto inicial ruidoso e ponto final ruidoso: (a) trajetória 1 e (b) trajetória 2.....	90
FIGURA 56 - Recuperação das trajetórias únicas no segundo plano. (a) pontos interpolados na trajetórias; (b), (c), (d) e (e) pontos interpolados e extrapolados na trajetória.....	92

- FIGURA 57 - Recuperação das trajetórias do tipo n-1; (a) trajetória 1, (b) trajetória 2 e (c) trajetória recuperada.....94
- FIGURA 58 - Recuperação das trajetórias do tipo 1-m; (a) trajetória 1, (b) trajetória 2 e (c) trajetória recuperada.....95
- FIGURA 59 - Recuperação das trajetórias do tipo 1-1 sem cruzamento; (a) trajetória 1, (b) trajetória 2 e (c) trajetória recuperada.....96
- FIGURA 60 - Recuperação das trajetórias do tipo 1-1 com cruzamento; (a) trajetória 1, (b) trajetória 2, (c) trajetória 1 e (d) trajetória 2.....97

LISTAS DE TABELAS

TABELA 1 - Parâmetros do manipulador PUMA560.....55

TABELA 2 - Tabela de distribuição dos pontos das trajetórias únicas para
treinamento de TMAM.....70

TABELA 3 - Tabela de distribuição dos pontos das trajetórias múltiplas para
treinamento de TMAM.....78

LISTA DE ABREVIATURAS E SIGLAS

DMM - Determinação da meta do movimento.

ETB - Estabelecimento da trajetória do braço.

TCB - Transformação das coordenadas do braço.

GCM - Geração de comando motor.

ABM - Ação do braço mecânico.

SVS - Sensoriamento das variáveis do sistema.

ACAM - Memória associativa de conteúdo endereçável.

BPTT - Backpropagation ao longo do tempo.

RBF - Função radial de base.

SC - Capacidade de armazenamento do modelo de Hopfield.

BAM - Memória associativa bidirecional.

PRLAB - Algoritmo de aprendizagem pseudo relaxamento.

MAM - Memória associativa multidirecional.

MLP - Perceptrons de múltiplas camadas.

TMAM - Memória associativa multidirecional temporal.

LISTAS DE SÍMBOLOS

$\mathbf{q} = [q_1 \ q_2 \dots q_n]^T$ - Vetor coluna representando ângulo das juntas, $Q \in \mathfrak{R}^n$.

$\mathbf{p} = [p_x \ p_y \ p_z]^T$ - Vetor coluna representando posição do atuador.

Φ - Vetor coluna de orientação.

$\mathbf{v}$ - Vetor coluna de velocidade.

Ω - Vetor coluna de aceleração.

$\theta(t)$ - Vetor posição angular no tempo.

$\dot{\theta}(t)$ - Vetor de velocidade angular no tempo.

$\ddot{\theta}(t)$ - Vetor de aceleração angular no tempo.

w_{ij} - Conexão partindo da unidade j e chegando a unidade i .

$\text{net}_i(\cdot)$ - Soma ponderada da entrada à unidade i .

$f(a_i(t))$ - Função de saída genérica da ativação da unidade i .

$\tau_i(t)$ - Torque da junta i .

l_i - Comprimento do link i do manipulador.

sgm - Função sigmoideal.

$\tanh$ - Função tangente hiperbólica.

r - Raio utilizado no cálculo do tamanho da bacia de atração do RBF.

np - Número de padrões considerados.

$E(\cdot)$ - Função de Lyapunov.

λ - Taxa de aprendizagem.

ξ - Constante que atua no tamanho da bacia de atração da rede.

$x_i(0)$ - Limiar da unidade i .

$\mathbf{x}^{(\text{pad})}$ - Vetor linha, $\mathbf{x} \in \mathfrak{R}^n$.

$\mathbf{y}^{(\text{pad})}$ - Vetor linha, $\mathbf{y} \in \mathfrak{R}^m$.

$\mathbf{s}$ - Vetor linha representando o atrator espúrio, $\mathbf{s} \in \mathfrak{R}^n$.

$\mathbf{r}$ - Vetor linha representando o resultado desejado (vetor linha), $\mathbf{r} \in \mathfrak{R}^m$.

W - Matriz de correlação.

W^T - Matriz de correlação transposta.

$\|\cdot\|$ - Norma Euclidiana.

$\mathbf{u}^{(\text{pad})}$ - Vetor coluna de entrada da rede RBF representando padrão pad.

$\mathbf{d}$ - Vetor coluna de saída obtida da rede RBF.

$\mathbf{D}$ - Matriz com as saídas desejadas da rede RBF.

nr - Número de unidades radiais.

$\mathbf{sd}(\mathbf{u}^{(\text{pad})})$ - Vetor coluna representando a saída desejada para o vetor $\mathbf{u}^{(\text{pad})}$.

p - Probabilidade da existência de ruído.

pad - Índice dos padrões, onde $\text{pad} = 1, \dots, \text{np}$.

$\mathbf{A}$ - Matriz Jacobiana utilizada na rede do tipo Kohonen.

$\text{sgn1}(\cdot)$ - Função sinal definida por Amari.

$\text{sgn2}(\cdot)$ - Função sinal definida por Hopfield.

$\text{sgn3}(\cdot)$ - Função sinal definida por Hagiawara.

nent - Número de entidades.

ϕ - Função de quantização.

x_{i0} - Limiar da unidade i da camada X .

y_{i0} - Limiar da unidade i da camada Y .

nc - Número de camadas de um modelo de rede.

nl - Número de unidades de processamento na camada ℓ .

nk - Número de unidades de processamento na camada k .

$\mathbf{net}(t)$ - Vetor composto por $\mathbf{net}(\cdot)$.

v_i - Componente i do vetor $\mathbf{v}$.

$\exp(\cdot)$ - Função exponencial.

$h_i(\cdot)$ - Função base radial.

x, y, z - Coordenadas espaciais.

$\hat{\mathbf{R}}^{-1}(\cdot)$ - Transformação inversa para obtenção dos torques.

RESUMO

VIEIRA, M. *Modelos de memória associativa em redes neurais para planejamento e controle ponto a ponto de trajetória de um braço mecânico*. São Carlos, 1997. 108p. Dissertação (Mestrado) - Escola de Engenharia de São Carlos, Universidade de São Paulo.

A contribuição e objetivo desta tese é desenvolver um modelo de redes neurais artificiais, baseado em princípios de memória associativa, capaz de resolver o problema de planejamento e controle ponto a ponto de trajetória de um braço mecânico imerso em um ambiente parcialmente conhecido e/ou sujeito a ruídos. O modelo proposto é formado por dois planos: plano seqüência temporal e plano ângulo. Para o plano seqüência temporal, o novo modelo proposto chamado de Memória Associativa Multidirecional Temporal (TMAM) é capaz de armazenar e recuperar n-tuplas de informações, lidar com informações ruidosas e/ou incompletas e aprender seqüências temporais. TMAM utiliza representação contínua e realimentação autoassociativa. O plano ângulo é formado pelo modelo RBF que é responsável por produzir as informações de ângulos das juntas do braço mecânico. A composição dos dois planos forma o sistema completo que é responsável pelo planejamento e controle ponto a ponto de trajetória. Em resumo, o sistema recebe informações do ponto origem e do ponto alvo, estabelece uma trajetória para atingir o ponto alvo a partir do ponto de origem e transforma os pontos espaciais da trajetória em valores de ângulos das juntas. Os resultados obtidos mostram que o modelo TMAM é capaz de recuperar, interpolar e extrapolar pontos nas seqüências, é capaz de gerar trajetórias, de memorizar seqüências de diferentes tamanhos e de lidar com duas trajetórias ao mesmo tempo. O modelo apresenta também rápido treinamento. O modelo RBF é capaz de recuperar as saídas desejadas apresentando um erro pequeno e é capaz de receber um padrão que apresenta um ponto final inatingível e gerar um conjunto de ângulos que representa um ponto final atingível.

Palavras chave: Memória associativa multidirecional temporal, manipulador mecânico, redes neurais, geração de trajetória, cinemática inversa, meios não estruturados.

ABSTRACT

VIEIRA, M. *Associative memory models in neural networks for point to point control and planning robot arm trajectory*. São Carlos, 1997. 108p. Dissertação (Mestrado) - Escola de Engenharia de São Carlos, Universidade de São Paulo.

The aim of this project is to develop an artificial neural networks model based on principles of associative memory. This neural network model must be able to solve the problem of trajectory planning and point to point control of a robot arm, which is located in a partially known and/or noisy environment. The proposed model is composed by two surfaces: the temporal sequence surface and the angle surface. For the temporal sequence surface the new propose model Temporal Multidirectional Associative Memory (TMAM) is able to store and recall n-tuplas of information, to deal with noisy and/or incomplete information and to learn temporal sequences. TMAM uses a continuos representation and autoassociative feedback. A RBF model is used to implement the angle surface, which is liable for producing the angle information for the joint of the robot arm. The two surfaces compose the whole system which is liable for the trajectory planning and system control. Hence, the system receives information about the initial point and the target point, constructs the trajectory to reach the target point from the initial point and converts the spatial points which compose the trajectory, in values of joint angles. The obtained results show that TMAM model can recall, interpolate and extrapolate points in the sequences. The model has the ability of generating new trajectories and memorizing different size of sequences at the same time. This model also shows fast learning. The RBF model can recall the desired outputs with a small error and can receive a pattern which is formed by an unreachable final point and generate a set of angles which, in turn, represent a reachable final point.

Key words: Temporal multidirectional associative memory, robot arm, neural networks, trajectory generation, inverse kinematics, unstructured environments.

1. Introdução

Os braços de robôs encontram numerosas aplicações, nos cenários industriais, em tarefas simples e repetitivas. Criar braços de robôs que possam ser programados para executar uma ampla variedade de tarefas é uma antiga meta da robótica e uma das motivações deste trabalho. Deseja-se que os braços projetados sejam capazes de executar tarefas inerentes aos seres humanos, embora não necessariamente da mesma maneira. Entre as capacidades humanas que os robôs devem reproduzir destacam-se:

- (i) Capacidade de resolver problemas orientados pelo espaço e pelo objeto. Os seres humanos têm a capacidade de planejar e seguir trajetórias espaciais com a finalidade de atingir um objeto alvo.
- (ii) Utilização de sensoriamento para manipulação de objeto. A visão é a nossa principal capacidade sensorial. Percepção de força e tato também são importantes. Essas capacidades permitem que os humanos achem objetos, identifique-os e colete-os.
- (iii) Habilidade senso-motora. Os seres humanos têm braços capazes de movimentos rápidos e precisos, unidos a mãos flexíveis.

Estas habilidades podem fundamentar o deslocamento espacial de um manipulador mecânico baseado em informações sensoriais. Neste caso, o braço do robô recebe informações visuais e de outros sensores que serão processadas por programas de computadores, produzindo, como saídas, a posição e a orientação corretas do efetuador, além dos ângulos e torques associados a junta.

Freqüentemente, técnicas tradicionais de controle não apresentam desempenho satisfatório em tarefas que lidam com controle de sistemas complexos e não-lineares, atuando em condições de incerteza e/ou de ruído.

Sistemas de controle com capacidade de auto-adaptação podem resolver algumas destas limitações. Portanto, tais estratégias de controle precisam ter a capacidade de aprender de maneira autônoma.

Grande parte dos paradigmas envolvendo aprendizado levam em conta exemplos de sistemas de controle eficazes no reino animal. A eficiência com que sistemas naturais executam processamento de informação sensorial e interação com ambientes incertos tem motivado cientistas e engenheiros a propor sistemas artificiais que empregam algumas destas habilidades. Deste modo, técnicas de aprendizagem de máquinas podem melhorar algumas das limitações de um sistema de controle complexo e não-linear de um braço mecânico para movimentação espacial.

Para este tipo de problema, a área de Inteligência Artificial clássica parece ser promissora. No entanto, a aplicação de tais técnicas em robótica também apresenta limitações significativas. Por exemplo, constata-se dificuldades em gerar e armazenar todas as possibilidades de modelos suficientemente precisos de robô e do ambiente. As estratégias de controle devem se adaptar a novas situações a partir de conhecimento parcial de seu universo de atuação. O paradigma de redes neurais artificiais, chamada de redes neurais a partir deste ponto, é usado para treinar os controladores do robô através de demonstração e não por programação de tipos de comportamento.

Este trabalho utiliza redes neurais, para simular deslocamento espacial de um braço de robô baseado em informações sensoriais compostas de coordenadas espaciais e coordenadas angulares, que servem de entrada para a rede neural. Resumindo, este projeto concentra-se no uso de aprendizagem por redes neurais, utilizando memória associativa, para controle motor em aplicações de robótica. Mais objetivamente, trata-se de problemas de planejamento e controle de movimentos em braços mecânicos.

Foi desenvolvido um sistema neural capaz de lidar com o problema de planejamento e controle de movimentos de um manipulador. Este sistema foi testado e suas características foram analisadas em relação à capacidade de recuperação, geração, interpolação e extrapolação de pontos em uma trajetória. O sistema neural proposto tem como objetivo ser utilizado em diferentes tipos de manipuladores. Este

sistema deve ser capaz de gerar trajetórias espaciais e resolver o problema da cinemática inversa. Para gerar as trajetórias e os ângulos para treinamento e teste do novo sistema foi escolhido o manipulador PUMA560.

Neste documento, o Capítulo 2º. descreve o problema. O Capítulo 3º. discute o emprego de redes neurais em controle senso-motor. O Capítulo 4º. apresenta os principais modelos de redes neurais para memória associativa. O sistema neural em dois planos para resolver o problema e o manipulador escolhido para simulação são apresentados e discutidos no Capítulo 5º. As simulações e os resultados são descritos no Capítulo 6º. As conclusões são relatadas no Capítulos 7º.

1.1. Objetivo

Este trabalho tem como objetivo principal desenvolver novas maneiras para controlar robôs em ambientes não estruturados e que estejam sujeitos a ruídos. Para isto, empregam-se redes neurais para planejamento e controle de deslocamento espacial de um braço de robô baseado em informações sensoriais. A pesquisa visa também propor e implementar um novo modelo de memória associativa em redes neurais para realizar o planejamento de trajetórias do braço do robô. Por último, este trabalho busca realizar a cinemática inversa da sequência temporal proposta.

Em termos de tarefa do sistema, o objetivo é posicionar o braço do robô em um local pré-estabelecido, seguindo uma trajetória determinada, empregando estratégias de controle inteligente que tornem o sistema estável, robusto e confiável. Espera-se substituir os modelos clássicos de cálculo da cinemática do robô por modelos de redes neurais que sejam capazes de executar estas tarefas com maior rapidez e de responder adequadamente a mudanças do meio e a informações ruidosas.

O modelo desenvolvido deve ser capaz de lidar com trajetórias espaciais, que apresentam graus de complexidade diferentes. Estas trajetórias representam os tipos de funções que um manipulador pode exercer em diferentes tarefas. As trajetórias podem ser dos seguintes tipos: as que partem de um mesmo ponto para diferentes pontos meta, trajetórias que partem de diferentes pontos, mas apresentam o mesmo

ponto meta, trajetórias que partem de um ponto para um ponto final sem apresentar cruzamento entre trajetórias e apresentando cruzamento.

O modelo proposto deve ser capaz de aprender estas trajetórias a partir de poucos exemplos e ser capaz de reproduzir, interpolar e extrapolar pontos na trajetória. Espera-se também que o modelo consiga gerar uma trajetória, dados o ponto inicial e o ponto meta.

2. Descrição do Problema

Fu et al. (1987) definiram um robô industrial como um manipulador de proposição geral, controlado por computador, formado por um conjunto de partes rígidas ligadas em série através de juntas rotativas ou juntas de translação. Uma extremidade do braço mecânico é fixada em uma base, enquanto que a outra extremidade, equipada com um efetuador, fica livre para manipular objetos ou realizar tarefas que requeiram “habilidade manual”. O deslocamento do braço se dá em função de movimento coordenado das juntas.

Nesta pesquisa, um braço com articulações rotativas move-se com três graus de liberdade e seu punho possui mais três graus de liberdade. A combinação do movimento nestes dois subsistemas leva o braço a se postar, em termos de posição e orientação, em um local desejado.

Neste Capítulo, a tarefa principal na qual se utilizará o controle senso-motor é descrita, seguindo-se a formalização deste problema.

2.1. Descrição da Tarefa

Como mencionado anteriormente, o foco principal desta pesquisa é o planejamento e controle de trajetória do robô que utilize informações sensoriais, fornecendo à rede neural informações de posição espacial e posição angular, para levar o braço mecânico e seu efetuador a uma dada situação final.

Para realizar esta tarefa o robô utiliza informações sensoriais que descrevem sua posição no espaço de trabalho e, informações angulares da posição e orientação do braço do robô e seu efetuador.

A partir deste ponto, o objetivo é gerar uma trajetória que faça com que o braço do robô se desloque da sua posição inicial até a posição desejada, acompanhando uma trajetória desejada da melhor maneira possível.

O posicionamento do braço do robô, em um dado objetivo, compreende:

- (a) fornecimento dos pontos iniciais e finais da trajetória;
- (b) estabelecimento de um conjunto de pontos espaciais entre a posição inicial do braço e a posição final desejada;
- (c) estabelecimento da seqüência de ângulos das juntas entre a posição de origem do robô e o ponto final, seguindo a trajetória preestabelecida.

2.2. Formalização do Problema

A tarefa acima se resume a determinar uma trajetória entre duas posições requeridas para posicionar o manipulador adequadamente. Para análise deste problema, utiliza-se o método de redução de problemas, proposto por Winston (1992) que reconhece os objetivos globais e os converte em submetas apropriadas. Isto é, o método reduz problemas complexos a um conjunto de subproblemas mais simples. Neste caso, os subproblemas são: cinemática do braço do robô, dinâmica do braço, planejamento e controle da trajetória do braço. Para que estes problemas sejam resolvidos, existe a necessidade de sensoriar o movimento do braço e adicionar “inteligência” a seu controle.

O problema da cinemática do braço do robô envolve a descrição do deslocamento espacial do robô, ao longo do tempo, relacionando-se os ângulos das juntas do robô com a posição dada em coordenadas cartesianas e a orientação de seu efetuador. O problema de cinemática é dividido em dois subproblemas: cinemática direta e cinemática inversa. O subproblema da cinemática direta transforma os ângulos das juntas do robô em informações de posição e orientação do efetuador. O subproblema da cinemática inversa transforma a posição e orientação do efetuador em coordenadas angulares das juntas do braço mecânico. Este problema tem grau de complexidade bem maior que o anterior, causado sobretudo pela não-linearidade das equações cinemáticas. Existe ainda a possibilidade de não haver solução ou de haver

soluções múltiplas para este problema. Em geral, a trajetória é definida em termos de pontos no espaço, o que torna este último subproblema mais freqüente.

Seja, então, um braço com n graus de liberdade no qual:

$$\mathbf{q} = [q_1 \ q_2 \ \dots q_n]^T \text{ é o vetor ângulo das juntas do braço;} \quad (1)$$

$$\mathbf{p} = [p_x \ p_y \ p_z]^T \text{ é o vetor posição do efetuador.} \quad (2)$$

No problema de cinemática deseja-se obter as relações que transformem o vetor $\mathbf{q}$ no vetor $\mathbf{p}$ e vice-versa, como esquematizado na Figura 1.

Figura 1. Representação esquemática dos problemas de cinemática direta e inversa do braço mecânico (Fu et al., 1987).

O problema de dinâmica do braço do robô tem como objetivo descrever o comportamento do manipulador, através da determinação das forças e dos momentos que regem seus movimentos. O problema é dividido em: subproblema de dinâmica direta e subproblema de dinâmica inversa. O subproblema da dinâmica direta transforma os torques e forças do sistema do braço do robô em aceleração, velocidade e coordenadas das juntas. O subproblema da dinâmica inversa objetiva transformar ângulos de juntas ou comprimentos dos “músculos” em ações de controle. Deseja-se obter um conjunto de torques que coordenem as atividades dos braços para seguir a trajetória desejada.

O planejamento e controle da trajetória do braço mecânico objetiva selecionar e seguir uma dada trajetória para realizar uma tarefa específica. Esta trajetória deve levar em conta possíveis obstáculos e restrições de especificações de caminho. Este problema pode ser dividido em duas partes: o subproblema do planejamento de

trajetória e o subproblema do controle do movimento ou rastreamento. O planejamento de trajetória estabelece um caminho, formado por um conjunto de pontos que o sistema deve seguir para sair da posição de origem e chegar à posição alvo. O movimento, ao longo da trajetória especificada, é monitorado e corrigido através de estratégias de controle. O caminho pode ser especificado em coordenadas das juntas ou em coordenadas cartesianas. Esta última é a maneira mais usual de se especificar a posição, orientação, velocidade e aceleração do braço.

A Figura 2 mostra o problema de planejamento de trajetória esquematicamente. O planejador de trajetórias recebe como entrada: especificações de percurso, restrições do percurso e restrições da dinâmica e da cinemática do manipulador. O planejador produz como saída uma seqüência temporal de configurações intermediárias do manipulador entre o ponto de partida e o ponto alvo. O controle de trajetória tem a função de avaliar se o planejamento está sendo seguido e corrigir a trajetória quando necessário.

Figura 2. Diagrama do planejador de trajetórias do braço (Fu et al., 1987).

O sensoriamento permite que o robô se adapte a mudanças ocorridas no meio ambiente. Fu et al. (1987) dividem o sensoriamento em interno e externo. O primeiro acompanha as variáveis utilizadas diretamente no controle do robô. O segundo tipo é

empregado para reconhecimento do espaço de trabalho, guiar o robô e ajudar na manipulação de objetos. Van der Smagt e Krose (1993) utilizam uma câmera no efetuador como sensor externo e uma câmera fixa como interno. A primeira é empregada para guiar o robô e ajudar na manipulação de objetos; e a segunda câmera, para fornecer informações de posições das juntas.

A adição de “inteligência” ao robô compreende habilitá-lo para realizar tarefas que não sejam apenas repetitivas. Isto é, o robô deve ser capaz de planejar, resolver problemas simples e se auto-adaptar quando for necessário.

O problema aqui tratado pode ser resumido como se segue. Define-se um ponto origem e um ponto alvo. Estabelece-se uma trajetória para atingir o ponto alvo a partir do ponto de origem. Transformam-se os pontos espaciais que compõem a trajetória em valores de ângulos das juntas. O movimento do braço é sensoriado para realimentação. O controlador executa as correções para seguir a trajetória desejada.

As soluções de controle clássico para este tipo de problema apresentam limitações, entre as quais se destaca a incapacidade de adaptação do sistema de controle para mudanças no ambiente, ou para presença de ruídos. Este trabalho se concentra na utilização de redes neurais para sobrepor tais dificuldades. Além disto este trabalho visa avaliar para que classe de tarefas o modelo a ser proposto é conveniente.

3. Redes Neurais em Controle Senso-Motor

Este capítulo introduz brevemente o conceito de redes neurais, descreve trabalhos relevantes de redes neurais para planejamento e controle de trajetória de manipuladores mecânicos e classifica os trabalhos discutidos de acordo com a natureza de sua abordagem.

3.1. Fundamentos de Redes Neurais

Na busca de construir máquinas inteligentes, um modelo que ocorre naturalmente é o cérebro humano. O cérebro humano possui características desejáveis para um sistema inteligente artificial: robustez e tolerância a faltas; flexibilidade, adaptabilidade e capacidade de aprendizado; capacidade de processar informação nebulosa, probabilística, ruidosa ou inconsistente; processamento paralelo; tamanho reduzido, grande capacidade de processamento e baixo consumo de energia. Logo, procura-se criar um sistema inteligente com neurônios artificiais para simular o funcionamento do cérebro humano em um computador.

Diferentes modelos de redes neurais visam replicar performance semelhante a de organismos inteligentes na resolução de problemas através da construção de sistemas computacionais inspirados no funcionamento destes organismos. Desta maneira, uma rede neural pode ser vista como um conjunto de elementos processadores simples, baseados em neurônios, que são ligados uns aos outros através de conexões análogas às sinapses. Estas conexões guardam o “conhecimento” da rede e os diversos padrões de atividade expressam os vários objetos codificados pela rede.

Em outras palavras, as conexões fazem o papel de memória de longo prazo, enquanto que o estado de ativação das unidades de uma rede realizam o papel de memória de curto prazo. O conhecimento da rede é adquirido por meio de um processo de treinamento no qual, em suas versões mais básicas, apenas as conexões entre as unidades são variadas através da mudanças de pesos. O entendimento do neurônio é, portanto, o ponto de partida para criação de uma rede neural.

O cérebro humano é composto por aproximadamente 10^{11} células nervosas, com cerca de 10^{15} interconexões em caminhos de transmissão. Os neurônios se ligam entre si formando uma imensa e complexa rede, na qual recebem impulsos eletroquímicos, processam estes impulsos e os retransmitem às outras células nervosas. São os neurônios e suas conexões sinápticas que servem de inspiração às redes neurais artificiais. Roseblatt (1962) aponta cinco componentes importantes em um neurônio (Figura 3).

Figura 3. Esquema simplificado de um neurônio.

- **DENDRITOS:** são captadores e condutores de estímulos nervosos, excitatórios ou inibitórios, de outros neurônios, transportando-os ao corpo celular.
- **CORPO CELULAR:** neste encontra-se o **Núcleo**, que é o responsável pelo processamento dos estímulos nervosos aferentes. Envia um novo impulso ao axônio, dependendo da comparação entre o resultado do processamento e a diferença de potencial entre as paredes interna e externa do neurônio.
- **AXÔNIO:** é uma fibra nervosa única cujas ramificações transmitem os impulsos provenientes do corpo celular.

- **SINAPSES:** são as ligações entre a árvore dendrital aos axônios de outras células nervosas. Estas funcionam como válvulas capazes de controlar a transmissão de impulsos nervosos ou potenciais de ação e regular a sua intensidade. Durante a vida de um sistema nervoso, as sinapses estão em constante formação e modificação.

O modelo mais explorado e que serve como base para a maioria dos estudos, é o PERCEPTRON que foi proposto originalmente por Roseblatt (1962). O Perceptron (Figura 4) equivale a um único neurônio, tem sua forma e características inspiradas no neurônio biológico. É através desta abstração que se constróem diferentes modelos de redes neurais com características próprias, mas que possuem o Perceptron como unidade básica. As entradas equivalem a sinais que chegam aos dendritos, os pesos representam as sinapses, as regras de agregação e ativação são a função do núcleo e a saída representa o sinal que o axônio propaga.

As entradas x_j , $j = 1, 2, 3, \dots, n$ são binárias; os pesos w_{ij} podem ser positivos ou negativos; a regra de agregação é dada por: $Net_i = \sum_{j=1}^n w_{ij} x_j$, a regra de ativação determina se o somatório é maior ou não que o limiar; a saída será 1 se a somatória for maior ou igual ao limiar, e será 0 se a somatória for menor que o limiar. O conhecimento do sistema é armazenado pelas conexões. O treinamento do sistema consiste em alterar os valores destas conexões, empregando algoritmos de treinamento, até obtermos uma saída desejada.

Figura 4. Modelo chamado Perceptron.

Algoritmos de treinamento de redes neurais podem ser supervisionados ou não-supervisionados. No primeiro caso, utiliza-se um “professor”. Durante o treinamento, são fornecidos pares entrada-saída e apenas as conexões entre as unidades são variadas através das mudanças dos pesos ajustados como função do erro entre a saída desejada e a saída obtida pela rede em treinamento. No caso dos algoritmos não-supervisionados, a própria rede se encarrega de levar um espaço de entrada para um espaço de saída, auto-arranjando seus pesos para isto. Os algoritmos supervisionados são normalmente mais usados em processos de engenharia, por encontrarem uma resposta desejada. Contudo, os modelos não-supervisionados são mais plausíveis sob os pontos de vista biológico e psicológico.

Oito são as características comuns à maior parte dos modelos de redes neurais propostas por Rumelhart e McClelland (1988), as quais formam um “framework” básico discutido a seguir.

1) **Um conjunto de unidades de processamento:** É o primeiro estágio da elaboração de um modelo de redes neurais. Possui dois aspectos importantes: o primeiro é quanto cada unidade de processamento vai se parecer com uma célula neural; o segundo é o significado de cada unidade, ou seja, o que ela representa individualmente e em conjunto.

2) **Definição de diferentes estados de ativação:** Cada estado de ativação de uma unidade em uma rede neural tem um significado associado. Existem inúmeras possibilidades de estados de ativação das unidades e suas escolhas levam em conta a representação escolhida. O estado de ativação pode ser: contínuo ou discreto; limitado ou ilimitado; ter valores reais, binários, bipolares, ou série de valores.

3) **Uma função de saída para cada unidade:** Os sinais que fazem com que as unidades interajam são transmitidos por uma função de saída $f(a_i(t))$. A função de saída transforma o estado de ativação de uma unidade $a_i(t)$ para um sinal de saída $O_i(t)$.

4) **Um padrão de conectividade entre as diferentes unidades:** O padrão de conectividade define o conhecimento do sistema, determinando como uma rede

responde a uma dada entrada. Tal padrão é codificado por pesos que representam conexões sinápticas excitatórias ou inibitórias.

5) **Uma Regra de agregação**: A função desta regra é combinar os valores de entrada de uma unidade com os valores da matriz de conexões para produzir o efeito total “sentido” pela unidade. A regra de agregação equivale à modificação da situação eletroquímica do neurônio.

6) **Uma regra de ativação**: Esta regra atualiza o estado de ativação de cada unidade, considerando-se a entrada ponderada desta unidade e seu estado de ativação presente. A regra determina se a unidade está ativa ou não.

7) **Uma regra de aprendizagem**: As mudanças dos pesos (conexões) envolvem o processo de aprendizagem que pode ser de três maneiras: através do aparecimento de novas conexões; da perda de conexões existentes e pela modificação de conexões já existentes.

8) **Um meio no qual o sistema opera**: A definição do ambiente deve considerar: sinais de entrada, que são fornecidos à rede como características do meio; um contexto, representando na rede informações importantes para o processamento das entradas, e uma saída que leva informação processada ao meio.

Definidas as características de um modelo de rede neural tais como: o número de unidades de processamento nas camadas de entrada, intermediária e de saída, assim como as regras de ativação e agregação, a regra de aprendizagem, os tipos de conexões entre unidades da mesma camada e entre diferentes camadas e também o meio em que vai atuar, este modelo está apto a ser utilizado em diferentes áreas.

Uma área de aplicação de redes neurais é o controle senso-motor, no qual elas substituem exaustivos cálculos da cinemática direta e inversa e da dinâmica direta e inversa. A rede neural pode determinar a trajetória, executar a transformação das coordenadas, gerar o torque para mudanças de trajetória e controlar as múltiplas juntas do braço do robô (Kawato, 1991).

3.2. Trabalhos relevantes selecionados

O problema de planejamento e controle de trajetória do braço do robô tem diferentes abordagens em termos de redes neurais. Em todas, o aprendizado do comportamento do robô ocorre por meio de exemplos. O mecanismo de aprendizagem substitui um conjunto de comportamentos pré-selecionados. A escolha do comportamento é alcançada ao final do processo de treinamento da rede, que mapeia o espaço de entradas do sistema em seu espaço de saída.

Os primeiros modelos, utilizando redes neurais em robótica, mantiveram o problema e o modelo o mais simples possível. Inicialmente, tratou-se do problema de cinemática para um robô bidimensional com dois graus de liberdade (Figura 5).

Uma rede neural de alimentação direta com método de aprendizagem supervisionado foi proposta por Josin (1988). Ele simulou um manipulador bidimensional com dois “links” de comprimento L_1 e L_2 e ângulos de juntas θ_1 e θ_2 . As equações de cinemática direta e inversa do manipulador geraram as relações entre a posição do efetuador e os ângulos das juntas do robô (θ_1, θ_2).

Figura 5. Ângulos e links de um manipulador bidimensional com dois graus de liberdade.

Os resultados das equações de cinemática inversa foram “ensinados” a uma rede com 2 unidades de entrada, 32 unidades escondidas e 2 unidades de saída. A rede foi treinada para um “grid” bidimensional 5x5. O treinamento, usando retropropagação, proposto por Rumelhart et al. (1986), foi rápido e as respostas obtidas foram as desejadas. Esta rede é bastante limitada, pois, entre outras coisas, o

espaço de trabalho é muito pequeno, e os resultados de cinemática inversa não são obtidos por tentativa e erro.

Para resolver o mesmo problema, Psaltis et al. (1988) propuseram uma rede neural de alimentação direta auto-supervisionada que aprende através de experimentação em duas dimensões. Neste caso, as saídas desejadas da rede são geradas por três estratégias diferentes e não são conhecidas a “priori”, como no caso passado.

A aprendizagem indireta, o primeiro método, utiliza uma rede neural para realizar o mapeamento dos pontos cartesianos do efetuador em ângulos das juntas do robô. Dado um par de coordenadas espaciais a rede produz os ângulos correspondentes. O manipulador move-se para esta posição de ângulo de juntas e gera, por sensoriamento, um novo par de coordenadas espaciais. Este novo par é apresentado à rede que produz novos ângulos. A rede adapta seus parâmetros para minimizar o erro entre os ângulos gerados. Tal minimização não garante a minimização do erro total. Na aprendizagem geral, o segundo método, Psaltis et al. empregaram o mesmo método, contudo os ângulos de entrada da planta do robô, são dados pelo usuário. A rede interpola pontos intermediários.

O terceiro método, chamado de aprendizagem especializada, utiliza a matriz Jacobiana da função da planta para calcular sua variação quando seus parâmetros mudam. O valor do erro calculado usando-se o Jacobiano é retropropagado.

Trabalhos mais recentes aumentam a complexidade do problema considerando a atuação em três dimensões e um braço de robô com três graus de liberdade.

Uma rede de alimentação direta em conjunto com uma estratégia de ajuste de entrada foi o método proposto por Van der Smagt et al. (1993). Tal método não demanda conhecimento prévio da planta. Os autores introduziram um sistema para coordenação estática mão-olho que é auto-adaptável em função de mudanças no meio-ambiente. O sistema de controle deve posicionar o efetuador sobre um dado alvo para coletá-lo. O controlador só lida com informações de posição, mapeando informações visuais de posição correntes em informações pertinentes para a construção da trajetória desejada.

O modelo acima usou dois tipos de premissas sobre o robô. Primeiro, os autores assumiram que o robô é controlado, especificando-se a mudança de ângulos de suas juntas e que os valores instantâneos destes ângulos são monitorados. Em segundo lugar, o sistema visual provê apenas informações sobre o objeto observado.

A entrada da rede é composta pelos dados visuais de observação do objeto alvo com respeito à posição alvo e pelos dados posicionais do braço mecânico. O controlador adaptável, a rede neural, aprende a gerar os diferenciais de posição que movimentam o efetuador na direção do objeto alvo. A transformação estabelecida neste caso é do tipo múltipla entrada para saída única, e é obtida através de treinamento supervisionado cujas posições ambíguas são deliberadamente evitadas.

Todas as abordagens, até então, só trataram do problema da cinemática. Alguns outros modelos também lidam com o problema da dinâmica.

Um modelo não-supervisionado foi proposto nos trabalhos de Martinez e co-autores (1990) e Ritter et al. (1992). Eles utilizaram uma rede do tipo Kohonen para controle de um braço com três graus de liberdade, o qual deve coletar um objeto no espaço tridimensional. Informações sensoriais foram fornecidas por duas câmeras fixas, que geravam as coordenadas do objeto alvo e do efetuador.

O movimento neste sistema foi dividido em duas etapas: um movimento grosseiro ou de aproximação da posição alvo e um movimento fino ou de ajuste. Na primeira etapa, a posição espacial do objeto é fornecida à rede do tipo Kohonen (1989) que ativa uma unidade em um “lattice” tridimensional. A unidade ativa corresponde à posição do objeto no espaço de trabalho do robô. Cada unidade ativa tem um vetor θ dos ângulos das juntas e uma matriz Jacobiana da função da planta (A) associada. Estes valores permitem o segundo movimento que corrige a discretização anterior por expansão em Taylor de primeira ordem. A aprendizagem da rede consiste em treinar os pesos entre cada entrada e cada unidade do “lattice” como na rede de Kohonen, e adaptar os valores de θ e A por minimização do erro de posição do efetuador.

Baseado no nível computacional de Marr (1982), Kawato (1991) propôs um modelo computacional para movimento voluntário de um braço mecânico multi-articulado para realização de uma dada tarefa.

Kawato (1989) apresentou evidências de que a trajetória desejada em coordenadas visuais e em coordenadas do corpo e o torque ativo são internamente representados no cérebro. Kawato substitui a metodologia de robótica clássica por duas outras estratégias: (i) resolução em etapa única dos problemas de determinação de trajetórias, transformação de coordenadas e geração de comando motor; (ii) resolução simultânea dos problemas de transformação de coordenadas e geração de comando motor.

Inicialmente, realiza-se a determinação da meta do movimento (DMM) que será dada juntamente com o ponto inicial do efetuador. Em seguida, processa-se o estabelecimento da trajetória do braço (ETB), isto é, transformam-se as metas em coordenadas espaciais do efetuador. O próximo passo consiste na transformação das coordenadas do braço (TCB), passa-se de coordenadas espaciais para ângulos das juntas. Estas últimas demandam geração de comando motor (GCM). Estes torques necessários são fornecidos à planta dirigindo a ação do braço mecânico (ABM) no meio ambiente. O desempenho do braço mecânico é acompanhado através do sensoriamento das variáveis do sistema (SVS) que serão realimentadas ao sistema para fins de controle. Neste contexto, as redes neurais serão responsáveis pelas funções dos blocos 2, 3, e 4, mostrados na Figura 6. Na primeira etapa, uma rede neural estabelecerá trajetórias espaciais. Na segunda etapa, uma rede neural executará as duas últimas etapas, determinando os ângulos das juntas e seus torques para se seguir a trajetória.

Figura 6. Diagrama de blocos das etapas do modelo computacional para planejamento e controle da trajetória de um braço de robô, baseado em Kawato (1991).

Kawato (1991) compara a solução por estratégia clássica de robótica com dois modelos de redes neurais para formação de trajetória e controle do movimento. Os modelos em redes neurais usaram resultados de experimentos comportamentais e conhecimento de fisiologia. Kawato propôs a resolução de problemas de trajetória, cinemática e dinâmica inversa por controle de movimentos voluntários. Este problema não tem solução única¹; logo, o autor combinou um índice de performance compatível a dados esquemas computacionais com algoritmos e modelos de redes neurais para realização do controle motor.

A primeira estratégia emprega uma rede neural com módulos em cascata para determinação da trajetória do robô. O braço é treinado por retropropagação proposto por Rumelhart et al. (1986), através do tempo, para produzir caminhos ótimos para o braço de acordo com o critério de mudança mínima de torque. Kawato et al. construíram um modelo de rede neural invariante no tempo composto por “n” módulos de quatro camadas cada, dispostos em cascata. A entrada de cada módulo recebe o torque ótimo, a posição e a velocidade espacial do objeto em um dado instante de tempo. A saída gera o valor das duas últimas variáveis de entrada no próximo instante de tempo. Os valores de saída de um bloco são alimentados no bloco posterior juntamente com o torque ótimo do instante de tempo seguinte. As correções dos pesos entre as camadas dos módulos são feitas por retropropagação, comparando-se os valores de saída de cada bloco com os valores sensorizados no sistema físico. Wada e Kawato (1993) discutiram as limitações deste modelo e propuseram uma nova rede baseada no mesmo critério. O novo modelo diminui o número de interações para se obter uma solução ótima aproximada pois, não requer representação espacial do tempo ou retropropagação.

O segundo modelo baseia-se em evidências de que áreas de associação sensorial do córtex (áreas 5 e 7) realizam a transformação de coordenadas e geração de comando motor através de aprendizado iterativo, isto é, por tentativa e erro. Neste caso, é necessário se conhecer a história recente da trajetória e do torque do braço para se resolver o problema da dinâmica inversa. Este é resolvido por uma rede neural de três camadas, treinada por retropropagação, ligada ao sistema em paralelo

---

¹Kawato chama de problema bem proposto todo aquele que tem solução única e depende continuamente dos dados iniciais. O problema é mal colocado caso não atenda ao menos a um destes critérios.

com o controlador do manipulador. Os ângulos desejados de trajetória são fornecidos simultaneamente ao controlador e à rede. Esta última produz um torque que somado ao torque do controlador é entregue ao robô. O torque do controlador é resultado da diferença entre o vetor ângulo desejado e aquele obtido de fato. Esta diferença cai a zero quando a rede completa o estágio de treinamento.

Atkeson e Reinkensmeyer (1991) discutiram o uso de memória associativa em controle de robô. Eles propuseram um algoritmo do tipo vizinho mais próximo paralelo, implementado numa "Connection Machine" (Hillis, 1985). Neste modelo, as experiências são armazenadas na memória e previsões são realizadas com base em buscas na memória. Foi empregado o modelo de memória associativa de conteúdo endereçável, ACAM (Kohonen, 1980), para controlar simulação de um braço planar com duas juntas. O emprego do vizinho mais próximo produziu resultados iniciais promissores.

Este modelo parte de uma trajetória preestabelecida e o robô usa os sinais de comando em conjunto com os sinais sensoriais realimentados para controlar o braço. O modelo armazena na memória, para cada instante de tempo t , os valores de posição, velocidade, aceleração e torques de cada junta "i" ($\theta_i(t), \dot{\theta}_i(t), \ddot{\theta}_i(t), \tau_i(t)$).

Em robótica, o modelo da dinâmica inversa produz o torque necessário ($\tau_i(t)$) para se achar uma dada trajetória: $\tau_i(t) = \hat{R}^{-1}(\theta_i(t), \dot{\theta}_i(t), \ddot{\theta}_i(t))$. A transformação $\hat{R}^{-1}$ é obtida no modelo de Atkeson e Reinkensmeyer por busca nos valores armazenados na memória. Logo, dado um conjunto de valores ($\theta_i(t_d), \dot{\theta}_i(t_d), \ddot{\theta}_i(t_d)$) procura-se na memória o conjunto destas variáveis que seja mais próximo dos valores dados. O torque armazenado com o conjunto achado é assumido como o torque para as grandezas fornecidas. A busca exaustiva entre os vários componentes da memória é um fator limitante do método. Schaal e Atkeson (1994) implementaram este esquema, usando aprendizado fundamentado numa técnica de regressão não-paramétrica.

Lin (1993) empregou aprendizagem por reforço, implementado em redes neurais, para criar estratégias de controle reativas sem qualquer conhecimento prévio dos efeitos causados pelas ações do robô. O agente recebe uma avaliação escalar do meio-ambiente que recompensa tarefas completadas com sucesso e penaliza tarefas

que fracassam. O agente tem que maximizar os sinais que o premiam. O agente pode então generalizar para eventos imprevistos. Este método se mostrou mais eficiente que compilação de ações para melhoria de performance no domínio do mundo real, onde a resposta a mudanças em seu meio-ambiente deve ser rápida. Em contrapartida, a convergência da aprendizagem foi tipicamente mais demorada e os planos produzidos nos primeiros estágios de aprendizagem eram perigosos para o robô.

Hirayama, Kawato e Jordan (1993) propuseram um sistema composto por duas fases para movimentação de um braço mecânico. O sistema é composto por uma fase de balística formada por um mecanismo de alimentação direta, que é responsável pelas informações sobre a distância entre pontos no espaço. A fase de postura utiliza as informações visuais sobre a posição do alvo para especificar níveis de comandos de motor para um grupo de músculos.

Neste trabalho, eles desenvolveram um modelo em rede neural que incorpora o controlador de alimentação direta e o controlador de postura. O primeiro é um modelo de rede neural em cascata que gera trajetórias baseadas nas seguintes informações: a duração do movimento de balística, a posição do alvo, a posição inicial e os níveis de ativação dos músculos no início e no fim da duração do movimento de balística. O controlador de postura é referido ao modelo estático inverso que fornece ao motor comandos sobre o final da fase de balística.

No modelo proposto, os movimentos do braço compreendem vários passos. Primeiro, valores dos parâmetros que especificam o tempo do movimento, a posição do alvo e o tamanho do alvo são selecionados. Estes valores são passados pelo controlador de alimentação direta (rede em cascata), o qual prepara o movimento de balística para uma duração específica. No final da fase de balística, o controle é passado para o controlador de postura. Se a posição da mão, no final da fase de balística, coincidir com o comando de postura estacionário do mecanismo de realimentação (controlador de postura), nenhuma oscilação é observada. Se as posições diferem ou mesmo se elas forem as mesmas, mas a velocidade não for zero, observam-se oscilações. Estas oscilações convergem para valores especificados pelo controlador de postura.

Kim et al. (1997) utilizam em seu trabalho uma estrutura de rede neural caótica com aprendizagem por retropropagação para atualizar as matrizes de pesos. A rede deve trabalhar em paralelo com um controlador PD para simular o controle de rastreamento de trajetória do manipulador de um robô PUMA de 3 eixos. Os resultados são comparados com os resultados de um modelo de rede recorrente.

A arquitetura direta e adaptativa de rastreamento de trajetória para um manipulador PUMA de 3 eixos consiste de um controlador PD e um controlador de rede neural caótica em paralelo. O controlador PD é usado para controlar o manipulador durante a fase de aprendizagem da rede neural caótica. Durante a fase de aprendizagem a rede neural caótica é utilizada para compensar o torque do controlador PD para redução do erro. Depois da fase de aprendizagem, o papel do controlador PD é reduzido.

O controlador de rede neural caótica possui seis entradas compostas pelos ângulos e as velocidades angulares do planejamento de trajetória para o robô; possui uma camada intermediária com dez unidades, e a camada de saída é composta por três unidades que são responsáveis pelos torques. Na rede neural caótica, cada unidade de processamento é composta por três tipos de entradas: entrada externa das unidades das outras camadas; entradas de realimentação das unidades da mesma camada e entradas de auto realimentação. As matrizes de pesos são atualizadas pelo algoritmo retropropagação.

Os resultados obtidos sugerem que o desempenho do controlador de rede neural caótica é melhor que o controlador de rede recorrente, a rede neural caótica apresenta uma rápida aprendizagem devido à estrutura interconectada dos neurônios, e o processamento de controle é rápido devido à pequena estrutura.

3.3. Classificação das diferentes abordagens.

Podemos classificar os vários trabalhos que empregam redes neurais para controle senso-motor de acordo com a maneira como eles aplicam o processo inteligente ao controle do braço mecânico. Werbos (1991a) aponta cinco abordagens de utilização de redes neurais em controle motor: (i) emprego de redes neurais

supervisionadas; (ii) uso de controle direto-inverso; (iii) emprego de controle adaptativo neural; (iv) utilização de retropropagação temporal; (v) uso de métodos com crítico adaptativo.

Uma rede neural supervisionada, quando aplicada em controle, deve repetir de forma aproximada o comportamento de um ser vivo ou de um programa de computador que sabe realizar uma dada tarefa. Em robótica, este tipo de controle aprende como os movimentos variam, ou como são tomadas as ações em resposta a informação sensorial (Widrow, 1991).

Em controle direto-inverso, uma rede neural mapeia o espaço de posições de um braço de robô (suas coordenadas espaciais) em coordenadas no espaço dos sinais atuantes que controlam a posição do braço (os ângulos de suas juntas). A rede faz com que o braço siga uma trajetória pré-estabelecida ou atinja um alvo determinado. Em geral, empregam-se métodos supervisionados cujas saídas são os sinais atuadores e as entradas são os sinais sensoriais. O mapeamento é realizado, deixando o sistema se mover livremente no espaço de trabalho e gravando as relações que interessam (Kawato, 1991).

Em controle adaptativo neural, uma rede neural artificial busca uma meta dada. A rede substitui os mapeamentos lineares empregados em controle adaptativo convencional, por exemplo: regulador auto-sintonizante, controle adaptativo modelo-referência (Kraft e David, 1991).

Em retropropagação, temporal (BPTT), estabelece-se uma função objetivo a ser otimizada e um modelo de meio ambiente externo (Williams, 1991). A rede calcula a variação da função objetivo no futuro com base em ações presentes. Este método requer modelos exatos do meio ambiente e cálculos de retropropagação que são incompatíveis com aprendizagem em tempo real. É fundamental a escolha de uma função objetivo que represente realmente a finalidade do braço.

Em métodos usando um crítico adaptador ou aprendizado por reforço, busca-se otimizar uma função objetivo que mostra conseqüências futuras de ações presentes. Métodos diferentes para propor críticos são discutidos por Barto (1991) e Werbos (1991b).

Com base em outros modelos propostos de redes neurais, a divisão sugerida por Werbos pode ser aumentada. Portanto, acrescentam-se duas outras abordagens: (vi) utilização de redes não-supervisionadas; (vii) emprego de modelos de memória associativa. Além disto, generaliza-se a abordagem (iv) para: utilização de redes recorrentes.

As redes não-supervisionadas realizam aprendizagem semelhante à das redes supervisionadas, mas sem a presença de um “professor”. Os modelos de memória associativa visam à associação direta entre conjuntos completos de variáveis. Neste caso, qualquer combinação de variáveis pode ser usada como entrada para produzir uma saída. Por último, abordagens distintas de redes recorrentes são viáveis para realizar a tarefa de gerar uma seqüência de posições espaciais ao longo de um determinado período de tempo.

Dentre os paradigmas mencionados acima, foi escolhido o de memória associativa para resolver o problema de planejamento e controle de trajetória. Modelos de memória associativa foram escolhidos devido a sua capacidade de lidar com armazenamento e associação de informações de naturezas diferentes, como: informações de posições espaciais, posições angulares e informações de torque, por apresentarem rápido treinamento e rápido tempo de resposta; por possuírem tolerância a falhas e plausibilidade maior que modelos treinados por retropropagação. Além disto, este é um grupo de modelos pouco explorado na literatura para este tipo de problema. Logo, é importante ampliar a gama de aplicação destes modelos, capacitando-os a lidar com trajetórias.

4. Modelos de Redes Neurais

Este capítulo tem como objetivo descrever os vários trabalhos estudados, relacionados à memória associativa que fundamentam o desenvolvimento de um novo modelo de memória associativa para lidar com o problema de planejamento e controle de trajetória. Traz também algoritmos de treinamento que auxiliam no treinamento do modelo proposto. Apresenta também o modelo RBF que é responsável pela parte de cinemática inversa. Este modelo juntamente com o novo modelo de memória associativa formam um sistema de dois planos para resolver o problema de geração de trajetória e cinemática inversa.

A Seção 4.1 faz uma breve introdução sobre memória associativa, os trabalhos descritos na Seção 4.2 fundamentam o desenvolvimento do modelo a ser proposto para resolver o problema de geração de trajetórias. São trabalhos relacionados a modelos de memória associativa e algoritmos de treinamento para aumentar a capacidade de armazenamento destas redes. A Seção 4.3 apresenta o modelo RBF que será usado juntamente com o modelo de memória associativa para resolver o problema proposto.

4.1. Definição de memória associativa

Associações são comuns dentro das nossas experiências do dia a dia. É fácil para um ser humano associar nomes a rostos, recordar alguém que lhe pareça familiar. Os seres humanos formam ligações entre pessoas, eventos e lugares, entre formas de objetos e conceitos e esta habilidade permite que se construam representações de como vemos o mundo. Os estímulos sensoriais que os seres

humanos processam normalmente disparam uma cascata de associações e lembranças. Pode-se descrever memória associativa como um sistema de memória em que uma entrada específica evoca uma resposta associada àquela entrada. Computacionalmente, modelos de memória associativa têm sido estudados por muitos anos, e importantes trabalhos em redes neurais têm sido desenvolvidos neste campo.

Araújo (1994) define memória associativa como uma transformação de vetor de espaço ($T: \mathfrak{R}^n \rightarrow \mathfrak{R}^m$), geralmente não linear, que mapeia vetores $\mathbf{x}^{(pad)} \in \mathfrak{R}^n$ em vetores $\mathbf{y}^{(pad)} \in \mathfrak{R}^m$. Uma vez fornecida como entrada da rede o vetor $\mathbf{x}^{(pad)}(0)$, vetor suficientemente similar ao vetor $\mathbf{x}^{(pad)}$ armazenado, é possível recuperar o vetor $\mathbf{y}^{(pad)}$.

Se os conjuntos de vetores $\mathbf{x}^{(pad)}$ e $\mathbf{y}^{(pad)}$ são os mesmos e ($T: \mathfrak{R}^n \rightarrow \mathfrak{R}^n$) então chama-se a operação de auto-associação. Uma memória associativa vem a ser heteroassociativa quando $\mathbf{x}^{(pad)}$ e $\mathbf{y}^{(pad)}$ possuem conjuntos distintos. Pode-se modelar memória heteroassociativa por modelos de redes neurais de alimentação direta ou com realimentação. Os vetores desta definição bem como todos os demais a partir deste ponto são vetores linha como é em geral considerado nos modelos de memória associativa.

Os modelos de Memória Associativa podem ser classificados de várias maneiras, dependendo de sua arquitetura (estática versus recorrente), da natureza das associações entre padrões (auto-associativas versus heteroassociativa), da complexidade e da capacidade dos algoritmos de armazenamento e recuperação, entre outras características.

A escolha de um desses modelos para uma aplicação deve levar em conta o tempo de treinamento, a facilidade para construir mapeamentos, a capacidade para classificar um sistema e a quantidade de informações existentes a serem armazenadas. Além disso, deve-se observar, no modelo, características importantes tais como: generalização como uma propriedade resultante, rápido tempo de resposta, tolerância a falhas, capacidade de abstração e de aprender com ou sem a presença de um supervisor.

4.2. Principais modelos de memória associativa

Vários modelos de Memórias Associativas foram propostos nas últimas duas décadas, entre os quais se destacam os trabalhos de Amari (1972), Anderson (1972), Nakano (1972), Kohonen (1972), Hopfield (1982), Kosko (1988), Hagiwara (1990), Oh e Kothari (1994), Hattori e Hagiwara (1995), Araújo e Haga (1997). Neste capítulo, são discutidos os trabalhos mais importantes para o desenvolvimento do modelo a ser proposto.

4.2.1. Primeiros Modelos

Amari (1972) propôs a aprendizagem e recuperação de padrões e seqüências de padrões por redes auto-organizáveis. Amari divide seu trabalho em duas partes. Na primeira, ele estuda a estabilidade de estados de transição da rede. Na segunda parte, três características das redes auto-organizáveis são analisadas. A arquitetura proposta por Amari é mostrada na Figura 7.

Figura 7. Modelo da arquitetura de Amari.

A primeira parte mostra como ocorre a atualização das matrizes de conexão em relação à estabilidade de estado de transição em redes de elementos com limiares. A função de ativação de uma unidade j na rede é definida pela Equação 3.

$$x_i(t) = \text{sgn} l \left[\sum_{j=1}^n w_{ij} x_j(k-1) - x_i(0) \right] \quad (3)$$

$$\text{na qual} \begin{cases} \text{sgn} l(u) = \begin{cases} 1, \text{ se } u > 0 \\ -1, \text{ se } u < 0 \end{cases} \\ x_i(0) = \text{limiar da unidade } i \end{cases}$$

Os elementos de conexão (w_{ij}) são atualizados de acordo com a seguinte estratégia: os pesos são incrementados de uma unidade se o j -ésimo padrão de entrada e o i -ésimo padrão de saída coincidem na ativação, caso contrário os pesos são decrementados de uma unidade.

Três características são estudadas na segunda parte. A primeira analisa a aprendizagem de muitos padrões ruidosos, concluindo que a rede é capaz de recuperar um padrão x , corretamente, quando atinge um estado de equilíbrio. A intensidade do ruído que é apresentado à rede é determinada pela probabilidade p da existência deste ruído em um padrão de entrada. A rede é capaz de recuperar o padrão x a partir do padrão ruidoso se o número de entradas n for suficientemente grande. A probabilidade p da existência de ruído em um padrão pode ser:

$$p = \frac{1}{4}, \quad p = \frac{1}{3} \quad \text{e} \quad p = \frac{2}{3} \quad (4)$$

dadas as probabilidades de ruídos temos respectivamente:

$$n \geq 2k+3, \quad n \geq 2k+8 \quad \text{e} \quad n \geq 2k+24 \quad (5)$$

para k unidades de processamento vizinhas.

Se n e k forem grandes, o efeito do ruído vem a ser relativamente pequeno. Tendo, como exemplo, o número de vizinhos k igual a 50, a rede deve possuir n

unidades de processamento de entrada igual à: $n \geq 103$, $n \geq 108$ e $n \geq 124$ respectivamente, para poder recuperar o vetor x

A segunda característica mostra qual a condição para ocorrer o aprendizado de muitas seqüências de padrões simultaneamente. Neste caso, os padrões que constituem uma dada seqüência têm que ser ortogonais aos padrões da outra seqüência.

A terceira, mostra a formação de um padrão representativo a partir da apresentação de vários padrões ou seqüências de padrões. Este padrão representativo pode ser recuperado a partir de qualquer membro do conjunto de padrões que o gerou. Este tipo de rede auto-organizável pode ser considerado como modelo de memória associativa para recuperação de seqüência e formação de conceito.

Na mesma época, um mesmo modelo de associação linear foi simultânea e independentemente proposto por Kohonen (1972) e Anderson (1972). O primeiro concentra-se na modelagem matemática, enquanto o segundo mantém o foco na plausibilidade fisiológica.

O modelo é caracterizado por um conjunto de elementos de entrada, receptores, os quais possuem conexões com o grupo de componentes de saída, associadores. O valor de saída contínuo é proporcional à somatória de cada unidade multiplicada pelo peso associado. Este modelo de memória associa linearmente np pares de padrões $(x^{(pad)}, y^{(pad)})$ através da regra de ativação:

$$y^{(pad)} = x^{(pad)}W \quad (6)$$

$$\text{onde: } \begin{cases} \text{pad} = 1, \dots, np. \\ W = \text{matriz } n \times m \\ x \in \mathfrak{R}^n, y \in \mathfrak{R}^m \end{cases}$$

A influência dos pesos associados às sinapses é proporcional ao produto da ativação antes e depois das sinapses. A matriz W é construída através da regra de aprendizagem Hebbiana.

$$W = \sum_{pad=1}^{np} \mathbf{x}^{(pad)T} \mathbf{y}^{(pad)} \quad (7)$$

Para ocorrer a recuperação correta de um padrão, é necessário que o vetor de entrada $\mathbf{x}^{(pad)}$ seja linearmente independente do vetor $\mathbf{y}^{(pad)}$. Esta condição elimina o “cross-talk” entre cada par de padrão de entrada; entretanto, este é um limite severo para o uso deste modelo.

Nakano (1972) propôs um modelo de memória associativa chamada “Associatron”. Este modelo lida com padrões e entidades simultaneamente. Os padrões são informações representadas através de vetores de estados com valores +1, 0 ou -1, que formam um conceito e compõem uma entidade. Uma entidade é uma matriz de componentes bidimensionais contendo um ou mais padrões, como mostra a Figura 8.

Figura 8. (a) Configuração de uma entidade formada por dois padrões e uma área neutra, (b) conceito representado por cada padrão.

A Figura 8(a) mostra um exemplo de entidade. Esta entidade possui dois padrões e uma área neutra. Cada padrão representa um conceito como mostrado na Figura 8(b). A área neutra é formada apenas por zeros e não contém nenhum conceito.

O “Associatron” possui três propriedades importantes: (i) capacidade de recuperação de padrões a partir de entradas parcialmente completas; (ii) capacidade

de associação entre entidades; (iii) capacidade de gerar uma seqüência através dos padrões em comum entre as entidades.

As associações entre padrões podem ocorrer de duas formas: na primeira, dentro da própria entidade na qual há uma correlação entre os padrões que compõem a mesma entidade; e na segunda maneira, ocorre a associação entre entidades que possuam um padrão em comum, formando-se um elo de associação entre elas.

A recuperação de uma entidade é feita fornecendo-se à rede parte desta entidade para que ela recupere seu todo. Como ilustra a Figura 9(a), apresentando à rede parte do padrão 4 da entidade 1, esta recupera todos os padrões que compõem a entidade 1. Um dado padrão pode causar a recuperação de diferentes entidades, neste caso, ambas as entidades devem possuir este padrão. Por exemplo, a Figura 9b ilustra que, ao fornecermos o padrão 1 da Figura 8(b), o modelo recupera os padrões da entidade 1 (Figura 9(a)) e os padrões da entidade 2 (Figura 8(b)).

Figura 9. Recuperação das: (a) Entidade 1 apresentando um padrão ruidoso; (b) entidade 1 e entidade 2 apresentando um padrão comum a ambas.

O modelo “Associatron” é uma rede na qual todo par de unidades de processamento x_i e x_j é totalmente conectado através do peso da conexão w_{ij} . As informações das entidades são representadas por vetores linhas $x^{(1)}, x^{(2)} \dots x^{(nent)}$ armazenados pela rede, em forma de matriz construída pela equação:

$$W = \mathbf{x}^{(1)T} \mathbf{x}^{(1)} + \mathbf{x}^{(2)T} \mathbf{x}^{(2)} + \dots + \mathbf{x}^{(nent)T} \mathbf{x}^{(nent)} \quad (8)$$

na qual: nent é o número de entidades

Para recuperação das entidades a partir de uma entrada $\mathbf{y}^{(1)}, \mathbf{y}^{(2)}, \dots, \mathbf{y}^{(nent)}$, a memória pode produzir um vetor linha X cuja ativação é dada por:

$$X = \phi(Y\phi(W)) \quad (9)$$

$$\text{onde: } \begin{cases} X = [(\mathbf{x}^{(1)})^T (\mathbf{x}^{(2)})^T \dots (\mathbf{x}^{(n)})^T] \\ Y = [(\mathbf{y}^{(1)})^T (\mathbf{y}^{(2)})^T \dots (\mathbf{y}^{(n)})^T] \end{cases}$$

Esta equação é usada para extrair qualquer X a partir de W .

A função de quantização é definida por:

$$\phi(a) = \begin{cases} -1, & \text{se } a < 0 \\ 0, & \text{se } a = 0 \\ 1, & \text{se } a > 0 \end{cases} \quad (10)$$

Se o processo de recuperação de informação é repetido de tal forma que o padrão obtido é usado como próxima entrada, uma cadeia de associações pode ser traçada. Este mecanismo é considerado um processo de raciocínio sequencial ao invés de um processo de recuperação de entidades armazenadas. Nele, a estrutura de associações não foi diretamente memorizada, mas foi sendo formada na memória através da experiência a partir de partes de padrões compartilhados.

Figura 10. Padrões com partes em comum que formam uma sequência no tempo.

Na figura 10, entidades são memorizadas em momentos diferentes. Observar-se também que alguns pares possuem padrões em comum, pares J-F, F-G, G-H e H-I. Neste caso, quando aplicada uma entrada J, a rede retorna às entidades J, F, G, H e I, desde que cada saída seja realimentada à rede.

“Associatron” memoriza entidades distribuidamente no tempo e as chama associativamente no espaço. Conseqüentemente, os modelos têm propriedades que os diferem dos dispositivos de memória convencionais. Essas propriedades mostram-se úteis na realização de máquinas inteligentes. A habilidade do “Associatron” cresce conforme aumenta seu número de unidades de processamento. O modelo apresenta limitação em relação à recuperação perfeita das entidades armazenadas. Isto ocorre quando as entidades possuem padrões em comum umas com as outras. Para que não ocorra sobreposição de entidades, elas não podem possuir padrões em comum.

4.2.2. Modelos com Auto-relaxamento.

Hopfield (1982) propôs um outro modelo de memória auto-associativa que consiste em um conjunto de N unidades de processamento interconectadas, como mostra a Figura 11.

Figura 11. Modelo da rede de Hopfield.

A atualização dos valores de ativação das unidades de processamento é assíncrona e independente das outras unidades. Todas as unidades de processamento

são unidades de entrada e saída ao mesmo tempo. Os valores de ativação podem ser binários ou bipolares.

Este modelo é capaz de armazenar e recuperar padrões, generalizar, categorizar e corrigir erros na entrada.

A função de ativação para cada uma das n unidades é dada pela equação abaixo:

$$x_i(k) = \text{sgn} 2 \left[\sum_{j=1}^n w_{ij} x_j(k-1) - x_i(0) \right] \quad (11)$$

na qual: $\begin{cases} x_i(0) \text{ representa o limiar da unidade } i \\ n = \text{numero de unidades} \end{cases}$

Onde a função sinal é definida por Hopfield como:

$$\text{sgn} 2(v_i) = \begin{cases} 1 & \text{se } v_i \geq 0 \\ -1 & \text{se } v_i < 0 \end{cases} \quad (12)$$

Cada elemento w_{ij} da matriz W é construída como mostra a equação abaixo:

$$w_{ij} = \sum_{\text{pad}=1}^{np} [(2x_i^{(\text{pad})} - 1)(2x_j^{(\text{pad})} - 1)] \quad (13)$$

na qual $\begin{cases} \text{pad} = \text{indica o índice do padrão;} \\ np = \text{quantidade de padrões apresentados à rede.} \end{cases}$

Aplicada uma dada entrada, esta passa por sucessivos estados até convergir para uma solução estável. Esta acontece quando os valores de ativação das unidades de processamento permanecerem estáveis. A saída da rede é dada pelo valor de todas as unidades de processamento quando a rede alcança seu equilíbrio.

Neste modelo existem três comportamentos possíveis: comportamento estável, cíclico ou caótico. No comportamento estável, a rede converge para um

mínimo local ou global. No comportamento cíclico, a rede permanece em uma seqüência de estados; e no comportamento caótico, a rede fica oscilando sem conseguir alcançar um estado estável.

Uma das mais importantes contribuições de Hopfield dentro da teoria de redes neurais é a introdução do conceito da função de Lyapunov para estabelecer a estabilidade da rede. A idéia da função de Lyapunov é encontrar o estado mínimo de energia do sistema no qual a rede fica presa.

A função energia (E) foi determinada por Hopfield, para $w_{ij} = w_{ji}$ e $w_{ii} = 0$, como é mostrada abaixo:

$$E(k) = -\frac{1}{2} \mathbf{x}^{(pad)} \mathbf{W} (\mathbf{x}^{(pad)})^T \quad (14)$$

O algoritmo para alterar $\mathbf{x}$ faz com que E seja uma função monotonicamente decrescente. As mudanças de estados ocorrem até que esta função energia não se altere mais.

A capacidade de armazenamento do modelo de Hopfield é definida como o máximo número de padrões armazenados que podem ser perfeitamente recuperados. A capacidade de armazenamento é calculada pela equação (16) (McEliece et al., 1987):

$$SC = \frac{n}{4 \log n} \quad (15)$$

$$\text{na qual} \left\{ \begin{array}{l} SC \text{ é a capacidade de armazenamento do modelo;} \\ n \text{ é quantidade de padrões gerados aleatoriamente.} \end{array} \right.$$

Kosko (1988) propôs um modelo de rede neural heteroassociativa de duas camadas de unidades de processamento com representação binária ou bipolar, realimentação e pesos de conexões simétricos entre as duas camadas (Figura 12). Kosko argumenta que BAM é uma generalização de Hopfield.

Figura 12. Modelo de BAM proposto por Kosko.

A finalidade de BAM é armazenar e recuperar pares de padrões. Os pares de padrões armazenados possuem estados bidirecionalmente estáveis. BAM também tem a capacidade de recuperar padrões armazenados a partir de entradas incompletas e ruidosas. A informação é armazenada em BAM na forma de matrizes de pesos de conexões dada pela equação abaixo.

$$W = \sum_{pad=1}^{np} (x^{(pad)})^T y^{(pad)} \quad (16)$$

na qual: $x^{(pad)}$, $y^{(pad)}$ são pares de vetores que serão armazenados pela rede;

Em sua dinâmica, dada uma entrada ($x^{(i)}$), a rede opera fazendo fluir a informação da camada X para a camada Y e vice-versa, até que seu estado convirja para um mesmo valor de energia. Considerando os limiares nulos, sem perda de generalidade, a sequência dos estados da rede é:

Figura 13. Processo de convergência de BAM, onde W^T é a transposta de W .

É necessário mostrar que a rede é bidirecionalmente estável para assegurar sua convergência. Para provar a convergência Kosko emprega a função de Lyapunov que mapeia os estados da rede numa função energia E escalar.

$$E: X^n Y^m \rightarrow \mathfrak{R} \quad (17)$$

Baseado no trabalho de Hopfield, Kosko propôs uma medida da função de energia E :

$$E(\mathbf{x}, \mathbf{y}) = -\frac{1}{2}(\mathbf{x}W\mathbf{y}^T) - \frac{1}{2}(\mathbf{y}W^T\mathbf{x}^T) \quad (18)$$

Conseqüentemente, a estabilidade assintótica de BAM discreto é provada pela demonstração de que a função energia é limitada e decrementada sempre que o estado muda até atingir o mínimo. O estado de ativação das unidades pode mudar sincronamente ou assincronamente, mas a mudança do estado só ocorre em uma camada de cada vez.

Pode se dizer que BAM converge para um par $(\mathbf{x}^f, \mathbf{y}^f)$, o qual pode ou não ser a saída desejada, que representa um mínimo local da função energia E .

No modelo BAM, a equação (15) é usada para calcular o número máximo de pares que podem ser memorizados, na qual SC é a capacidade de pares que podem ser armazenados e n é a quantidade de unidades da menor camada.

Em resumo, BAM possui as seguintes características: (a) convergência rápida; (b) treinamento rápido; (c) cada estado de equilíbrio corresponde a um mínimo local de energia da rede; (d) a recuperação do par correto não é garantida; (e) baixa capacidade de memória.

Para melhorar a capacidade de armazenamento de BAM, Oh e Kothari (1994) apresentaram o algoritmo de treinamento Pseudo Relaxamento (PRLAB). Este modelo é uma adaptação do método de relaxamento para resolver um sistema de equações lineares. PRLAB ajusta adequadamente os pesos de conexões e os limiares das unidades de processamento. Em uma primeira etapa, PRLAB constrói as matrizes de correlação de acordo com a equação (16). Na segunda etapa, PRLAB examina cada par de treinamento $(\mathbf{x}^{(pad)}, \mathbf{y}^{(pad)})$ um por um, e sistematicamente muda os valores de seus pesos e limiares como descrito abaixo.

Para as unidades da primeira camada, se $s_{xi}^{(pad)} x_i^{(pad)} \leq 0$,

$$\Delta w_{ij} = -\frac{\lambda}{1+n} (s_{xi}^{(pad)} - \xi x_i^{(pad)}) y_j^{(pad)} \quad (19)$$

$$\Delta x_{i0} = +\frac{\lambda}{1+n} (s_{xi}^{(pad)} - \xi x_i^{(pad)}) \quad (20)$$

$$\text{Nais quais : } \left\{ \begin{array}{l} s_{xi}^{(pad)} = \sum_{j=1}^m w_{ij} y_j^{(pad)} - x_{i0} , \\ \lambda = \text{fator de relaxamento} \\ n = \text{número de neurônios da camada X} \\ m = \text{número de neurônios da camada Y} \\ \xi = \text{constante que atua no tamanho da bacia de atração da rede} \end{array} \right.$$

Para as unidades da segunda camada, se $s_{yi}^{(pad)} y_i^{(pad)} \leq 0$

$$\Delta w_{ij} = -\frac{\lambda}{1+m} (s_{yi}^{(pad)} - \xi y_i^{(pad)}) x_j^{(pad)} \quad (21)$$

$$\Delta y_{i0} = +\frac{\lambda}{1+m} (s_{yi}^{(pad)} - \xi y_i^{(pad)}) \quad (22)$$

$$\text{Nais quais: } \begin{cases} s_{yi}^{(pad)} = \sum_{j=1}^n w_{ij} x_j^{(pad)} - y_{i0} \\ \lambda = \text{fator de relaxamento} \\ n = \text{número de neurônios da camada X} \\ m = \text{número de neurônios da camada Y} \\ \xi = \text{constante que atua no tamanho da bacia de atração da rede} \end{cases}$$

O algoritmo de aprendizagem proposto apresenta as seguintes características: (i) rápido treinamento; (ii) o número de ciclos de treinamento permanece razoavelmente constante quando o número de pares e o número de unidades de processamento são incrementados ao mesmo tempo; (iii) o algoritmo proposto é pouco sensível às escolhas de pesos iniciais e de parâmetros dentro de um certo limite; (iv) a capacidade de armazenamento de BAM aumenta significativamente.

Outros dois algoritmos de aprendizagem são propostos por Araújo e Haga (1997) para melhorar a capacidade de armazenamento de BAM. O primeiro algoritmo de aprendizagem é chamado Desaprendizagem de Atratores Espúrios (USA) e é baseado na interferência retroativa na memória humana. Esta é definida como a interferência da aprendizagem atual no conhecimento anterior (Baddeley, 1990). O novo conhecimento de que a associação de um par em particular é incorreta, permite mudanças no traço de memória sem destruir o traço antigo.

Hopfield et al. (1983) sugeriram tal mecanismo de desaprendizagem para a rede de Hopfield. Araújo e Haga adaptaram aquele algoritmo, introduzindo um estágio de treinamento supervisionado adicional, no qual o produto externo ponderado de cada atrator espúrio vem a ser subtraído da matriz de correlação. Desta

forma, enquanto o atrator espúrio não desaparecer, W é variado para cada padrão pad como mostra a equação abaixo:

$$\Delta W^{(pad)} = -\left(\frac{\lambda}{m+1}\right) \left(\left(s^{(pad)} \right)^T r^{(pad)} \right) \quad (23)$$

$$\text{Na qual: } \begin{cases} s \in \mathfrak{R}^n, r \in \mathfrak{R}^m \\ s^{(pad)} \Rightarrow \text{atrator espúrio (vetor linha)} \\ r^{(pad)} \Rightarrow \text{resultado desejado (vetor linha)} \\ \lambda / m + 1 \Rightarrow \text{taxa de desaprendizagem} \end{cases}$$

O ótimo local, que produz o atrator espúrio, desaparece suavemente a cada novo passo de desaprendizagem. Desta forma, os atratores desejados podem reaparecer.

O segundo método é o algoritmo Regra Delta Bidirecional no qual os pesos são atualizados bidirecionalmente. A Regra Delta Bidirecional atualiza as matrizes de correlação, enquanto existirem atratores espúrios, com base na Regra de Widrow-Hoff (1960) como segue:

Quando a informação flui no sentido saída-entrada.

$$\Delta w_{ji} = \frac{\lambda}{n+1} (x_j^{(pad)} - s_j^{(pad)}) y_i^{(pad)} \quad (24)$$

Quando a informação flui no sentido entrada-saída.

$$\Delta w_{ij} = \frac{\lambda}{m+1} (y_i^{(pad)} - r_i^{(pad)}) x_j^{(pad)} \quad (25)$$

$$\text{nas quais: } \begin{cases} \frac{\lambda}{q+1} \text{ representa a taxa de aprendizagem;} \\ q = n \text{ ou } m; \\ (s^{(pad)}, r^{(pad)}) \rightarrow \text{é um par de atratores espúrios} \end{cases}$$

Estas regras de aprendizagem são baseadas em suposições cognitivas, são apropriadas para aprendizagem “on-line”, não requerem pré-processamento das entradas, possuem comportamento estável, são tolerantes a entradas ruidosas e apresentam habilidade de abstração.

O modelo de Memória Associativa Multidirecional (MAM) foi introduzido por Hagiwara (1990) como uma generalização de BAM. Este modelo tem como objetivo associar e armazenar n-tuplas de padrões entre si e recuperá-las através de informações parciais destas. MAM apresenta características tais como: (i) associação múltipla; (ii) capacidade de recuperação de dados incompletos; (iii) robustez e capacidade de filtragem de informações ruidosas.

A arquitetura de MAM descrita na Figura 14 é composta por n_l camadas totalmente conectadas, com padrões de representação bipolar. Cada camada é formada por p_k unidades de processamento, cujas p_k é o número de unidades p para a camada k .

Figura 14. Arquitetura de MAM composta por cinco camadas.

As ligações entre duas camadas l e k são compostas por matrizes de correlação W^{lk} construídas por regra Hebbiana de acordo com a equação:

$$w_{ij}^{lk} = \sum_{pad=1}^{np} (x_i^{(pad)}) y_j^{(pad)} \quad (26)$$

na qual: $\begin{cases} k \text{ e } l \Rightarrow \text{representam os números das camadas} \\ x_i^{(pad)} \text{ e } y_j^{(pad)} \Rightarrow \text{são componentes de vetores que serão armazenados} \\ \text{pela rede} \end{cases}$

Uma vez construídas as matrizes, a informação é propagada entre as camadas da rede através da equação abaixo:

$$Net_i(t) = \sum_{\substack{j=1 \\ i \neq k}}^{nl} w_{ij} y_j(t-1) \quad (27)$$

A função de ativação das unidades de cada camada é descrita pela equação (28):

$$x_i(t) = \text{sgn3}(Net_i(t)) \quad (28)$$

Na qual a função $\text{sgn3}(\cdot)$ é definida por Hagiwara como:

$$\text{sgn3}(v) = \begin{cases} +1 & \text{se } v > 0 \\ -1 & \text{se } v \leq 0 \end{cases} \quad (29)$$

A dinâmica de MAM pode ser descrita da seguinte maneira. Dado um padrão de entrada representado por uma tupla $(v^{(1)}, \dots, v^{(nc)})_{ini}$ o qual pode ser uma versão incompleta ou com ruído do vetor desejado $(v^{(1)}, \dots, v^{(nc)})$, a rede muda de estado como descrito abaixo:

$$(v^{(1)}, \dots, v^{(nc)})_{ini} \rightarrow W_{lk} \rightarrow (v^{(1)}, \dots, v^{(nc)})_1 \rightarrow \dots \rightarrow (v^{(1)}, \dots, v^{(nc)})_{fin} \rightarrow W_{lk} \rightarrow (v^{(1)}, \dots, v^{(nc)})_{fin}$$

Na qual: nc representa o número de camadas de MAM.

Cada estado gera um novo estado. Esta dinâmica é encerrada quando o estado da rede pára de mudar, gerando um vetor final $(v^{(1)}, \dots, v^{(nc)})_{\text{fin}}$ que pode ser o vetor desejado $(v^{(1)}, \dots, v^{(nc)})$. Hagiwara afirma, que a rede é multidirecionalmente estável, assegurando sua convergência.

MAM possui as seguintes características: (i) é um algoritmo de treinamento rápido; (ii) necessita de uma única passada das n-tuplas de entrada; (iii) a recuperação da tupla correta não é garantida; (iv) tem capacidade de armazenamento muito pequena.

Hattori e Hagiwara (1995) propuseram mudar o algoritmo de treinamento de MAM, visando a aumentar sua capacidade de armazenamento. O algoritmo escolhido pelos autores foi o PRLAB, por apresentar bons resultados no aumento da capacidade de armazenamento apresentada em BAM. Hattori e Hagiwara propuseram dois estágios de treinamento para MAM. No primeiro estágio, as matrizes são construídas por regra Hebbiana. No segundo estágio, MAM é treinado pelo algoritmo de aprendizagem PRLAB.

Considere o conjunto $\{x_1^{(\text{pad})}, x_2^{(\text{pad})}, \dots, x_{nc}^{(\text{pad})}\}$ para $\text{pad} = 1, 2, \dots, np$, no qual np é o número de padrões de treinamento a ser armazenado e nc o número de camadas de MAM. Considere ainda $w_{ij}^{(lk)}$ como a conexão chegando à i -ésima unidade de processamento na ℓ -ésima camada vinda da j -ésima unidade na k -ésima camada; x_{li} é o estado da i -ésima unidade de processamento na ℓ -ésima camada e x_{li0} é o limiar desta unidade de processamento respectivamente e nl o número de unidades de processamento na ℓ -ésima camada e nk o número de unidades de processamento na k -ésima camada. O primeiro estágio constrói as várias matrizes de correlação. No segundo estágio, MAM é treinado por PRLAB, para $\ell = 1, 2, \dots, nc-1$ e $k = \ell + 1, \ell + 2, \dots, nc$, e k -ésimo treinamento dos padrões são aprendidos como mostramos abaixo:

Para as unidades da ℓ -ésima camada, se $s_{li}^{(\text{pad})} x_{li}^{(\text{pad})} \leq 0$,

$$\Delta w_{ij} = -\frac{\lambda}{1 + nk} (s_{li}^{(\text{pad})} - \xi x_i^{(\text{pad})}) x_j^{(\text{pad})} \quad (30)$$

$$\Delta x_{i0} = + \frac{\lambda}{1 + nk} (s_{li}^{(pad)} - \xi x_{li}^{(pad)}) \quad (31)$$

Para as unidades da k-ésima camada, se $s_{ki}^{(pad)} x_{ki}^{(pad)} \leq 0$,

$$\Delta w_{ij} = - \frac{\lambda}{1 + nl} (s_{ki}^{(pad)} - \xi x_{ki}^{(pad)}) x_{lj}^{(pad)} \quad (32)$$

$$\Delta x_{i0} = + \frac{\lambda}{1 + nl} (s_{ki}^{(pad)} - \xi x_{ki}^{(pad)}) \quad (33)$$

$$\text{nasquais: } \begin{cases} s_{li}^{(pad)} = \sum_{j=1}^{N_k} w_{ij} x_{kj}^{(pad)} - x_{i0}, \\ s_{ki}^{(pad)} = \sum_{j=1}^{N_l} w_{ij} x_{lj}^{(pad)} - x_{i0} \end{cases}$$

Este procedimento é feito para todo $pad = 1, 2, \dots, np$.

Hagiwara afirma que este algoritmo se caracteriza por garantir a recuperação de todos os pares treinados; ser robusto em relação a entradas ruidosas; apresentar melhoras em relação à capacidade de armazenamento de MAM.

Concluindo, tem-se que alguns trabalhos estudados apresentam importância histórica e serviram de base para a compreensão de modelo de memória associativa. Outros trabalhos apresentaram diferentes arquiteturas e algoritmos de treinamento, os quais foram analisados quanto à capacidade de armazenamento, tempo de resposta, capacidade de lidar com seqüências temporais, robustez para lidar com informações ruidosas ou incompletas, capacidade de generalização, capacidade de associar n-tuplas e algoritmos que possibilitem aumentar a capacidade de armazenamento de uma rede. Tais características ajudaram propor um modelo de memória associativa para resolver o problema de planejamento e controle de trajetória. O Capítulo 5º. deixa claro o que está sendo usado entre estes modelos.

4.3. Função Base Radial (RBF)

4.3. Função Base Radial (RBF)

São atribuídas às redes neurais habilidades de aprender e generalizar a partir de exemplos. Os dois esquemas mais populares são os Perceptrons de Múltiplas Camadas (MLP) e Rede Base Radial (RBF). Ambos são aproximadores universais; são capazes de aproximar qualquer função contínua, dados suficientes graus de liberdade.

Ambos esquemas MLP e RBF são formados por unidades de processamento que realizam transformações não lineares de dados de entrada. Ambos os esquemas consistem basicamente três camadas de neurônios. A primeira camada é normalmente referida como camada de entrada e é simplesmente usada para passar a informação para a camada não linear escondida. A última camada ou camada de saída utiliza o somatório dos resultados a partir do processamento não linear dos elementos da camada intermediária para prover uma ou mais saídas.

Redes neurais são freqüentemente vistas no contexto de funções de aproximação, nas quais uma função desconhecida f será estimada a partir de dados incompletos. MLP e RBF podem ser considerados como representantes de duas maneiras para realizar esta aproximação; elas são baseadas respectivamente em produto escalar no caso de MLP e distância vetorial para RBF.

Este capítulo descreve brevemente o que é uma função base radial, o modelo RBF e seu funcionamento.

4.3.1. Função Base Radial

Função base radial (Leonard et al., 1991) faz parte de uma classe especial de funções. É caracterizada por saídas monotonicamente decrescentes ou crescentes de acordo com a distância a partir do ponto de centro. O centro, a escala de distância e a forma precisa da função base radial são parâmetros do modelo.

Uma função base radial típica é a função Gaussiana:

$$h_j(\mathbf{u}^{(pad)}) = \exp(-z_j^2) \quad (34)$$

Outras duas funções de base radial são também exploradas: função multiquadrática RBF e função Cauchy.

$$\text{Função multiquadrática RBF: } h_j(\mathbf{u}^{(\text{pad})}) = \frac{\sqrt{1 + z_j^2}}{r_j} \quad (35)$$

$$\text{Função Cauchy: } h_j(\mathbf{u}^{(\text{pad})}) = \frac{1}{1 + z_j^2} \quad (36)$$

$$\text{nas quais : } \begin{cases} \text{pad} = 1, \dots, \text{np}, \\ j = 1, \dots, \text{nr}, \\ \text{nr} = \text{Número de unidades da camada intermediária} \\ z_j = \frac{\|\mathbf{u}^{(\text{pad})} - \mathbf{c}^{(j)}\|}{r_j}, \\ r_j = \text{raio (escalar)} \\ \|\cdot\| = \text{Distância Euclidiana} \\ \mathbf{u}^{(\text{pad})} = \text{Vetor de padrão de entrada} \\ \mathbf{c}^{(j)} = \text{Vetor de centro da unidade } j \end{cases}$$

4.3.2. Redes de Função Base Radial (RBF)

Uma função base radial pode ser empregada em qualquer tipo de modelo (linear ou não linear) e em qualquer rede (uma camada ou múltiplas camadas). Redes de função base radial (RBF) possuem topologia como mostrada na Figura 15.

A arquitetura padrão de uma rede de função base radial é formada por 3 camadas do tipo de alimentação direta. A primeira camada é responsável pela distribuição da informação, a segunda camada é responsável pelo processamento da informação e a terceira camada é responsável pelo somatório das informações processadas na segunda camada.

Figura 15. Arquitetura padrão de uma rede de função base radial.

Na figura acima os n elementos de vetores de entrada $\mathbf{u}^{(\text{pad})}$ são propagados para frente para m funções base radial $h_{nr}(\mathbf{u}^{(\text{pad})})$ cujas saídas são combinadas linearmente com os pesos $\{w_j\}_{j=1}^{nr}$ dentro da função de saída $d_i(\mathbf{u}^{(\text{pad})})$ definida como mostra a equação 37.

O componente i da camada de saída tem ativação:

$$d_i(\mathbf{u}^{(\text{pad})}) = \sum_{j=1}^{np} w_{ij} h_j(\mathbf{u}^{(\text{pad})}) \quad (37)$$

A matriz de pesos W é construída como mostra a Equação 38.

$$W = (H^T H)^{-1} H^T D \quad (38)$$

$$\text{Na qual } H = \begin{bmatrix} h_1(\mathbf{u}^{(1)}) & h_2(\mathbf{u}^{(1)}) & \dots & h_m(\mathbf{u}^{(1)}) \\ h_1(\mathbf{u}^{(2)}) & h_2(\mathbf{u}^{(2)}) & \dots & h_m(\mathbf{u}^{(2)}) \\ \vdots & \vdots & & \vdots \\ h_1(\mathbf{u}^{(np)}) & h_2(\mathbf{u}^{(np)}) & \dots & h_m(\mathbf{u}^{(np)}) \end{bmatrix},$$

$$D = \begin{bmatrix} \mathbf{s} d^T(\mathbf{u}^{(1)}) \\ \mathbf{s} d^T(\mathbf{u}^{(2)}) \\ \vdots \\ \mathbf{s} d^T(\mathbf{u}^{(np)}) \end{bmatrix}$$

O modelamento com redes RBF encontra fundamentos nas redes neurais biológicas as quais são identificados em algumas estruturas corticais e em retinas, células altamente sintonizadas em torno de campos receptivos claramente definidos. Uma rede RBF, definida na Figura 15, apresenta, como função de ativação dos neurônios mais usada na camada escondida, a função Gaussiana. Os vetores c_j são os centros dos campos receptivos e, portanto, os pontos de máxima resposta do neurônio N_j . A função $h_j(u^{(pad)})$ reflete a forma de abertura dos campos receptivos de cada um dos neurônios. A resposta máxima do neurônio ocorrerá quando o vetor u igual ao vetor c , e quanto mais distante o vetor u estiver de c tanto menor será a sua resposta. O cone gaussiano será tanto mais estreito quanto menor for a variância r_i^2 . As curvas de nível são círculos concêntricos em torno do vetor c , representados na Figura 16 e, por esta razão, estas funções são identificadas como radiais.

Figura 16. Curvas de nível em forma de círculos.

O processamento de redes RBF consiste basicamente nas seguintes etapas: (i) definição dos tamanhos das camadas; (ii) determinação dos valores para os parâmetros da camada RBF, isto é, qual o tipo de função base radial a ser usada, o tamanho do raio e a quantidade de centros; (iii) construção da matriz de pesos e (iv) estabelecimento do tipo de treinamento, o qual envolve técnicas de análise de agrupamentos regularmente utilizadas em estatísticas e LMS.

A quantidade de centros a serem utilizados para representar uma categoria pode ser a mesma quantidade de pontos da amostra do padrão de treinamento quando este não for muito grande. Quando a quantidade de amostras para treinamento for muito grande, procedimentos mais elaborados podem ser aplicados para identificar agrupamentos, determinando, assim, os centros que os representariam. Estes agrupamentos raramente são evidentes.

5. Um Sistema Neural em Dois Planos

Neste capítulo encontra-se a contribuição principal deste trabalho que é o desenvolvimento do modelo chamado de Memória Associativa Multidirecional Temporal (TMAM). A arquitetura, as características, sua dinâmica e seus algoritmos de aprendizagem encontram-se descritos neste capítulo. Este capítulo descreve também o cenário para o sistema em questão, a rede RBF que será empregada para resolver o problema de cinemática inversa e o manipulador que foi escolhido para gerar os dados para treinamento e testes do novo modelo.

5.1. Cenário do Problema

Os vários modelos tratados anteriormente permitem propor um cenário para o sistema em questão. A Figura 17 mostra o modelo computacional para a operação do braço no qual o planejamento e controle de trajetória do manipulador são divididos em etapas. Inicialmente, realiza-se a determinação da meta do movimento (DMM). Esta etapa compreende informar especificações do percurso tais como ponto inicial e ponto final. Em seguida, processa-se o estabelecimento da trajetória do braço (ETB), isto é, estabelece-se uma seqüência temporal de pontos da trajetória do manipulador entre o ponto inicial e o ponto final desejados. O próximo passo consiste na transformação das coordenadas do braço (TCB), isto é, uma vez de posse dos pontos da trajetória realiza-se cinemática inversa.

O desempenho do braço mecânico é acompanhado através do sensoramento das variáveis do sistema (SVS) que serão realimentadas ao sistema para fins de controle.

Neste contexto as redes neurais serão responsáveis pelas funções dos blocos 2 e 3 mostradas na Figura 17. Na primeira etapa, uma rede neural estabelecerá trajetórias espaciais. Na segunda etapa, uma outra rede neural será responsável pelo cálculo dos ângulos das juntas para se seguir a trajetória.

Figura 17. Diagrama de blocos das etapas do modelo computacional para planejamento e controle da trajetória de um braço de robô.

Em resumo, é preciso escolher um braço mecânico para gerar as trajetórias e ângulos para treinamento dos modelos de redes neurais responsáveis pelos blocos 2 e 3. Escolheu-se um modelo que possui seis graus de liberdade: três no braço e três no punho, como mostram as Figuras 18 e 19. O sistema terá como entrada informações do ponto inicial e meta ou uma sequência de pontos de uma dada trajetória. O espaço de trabalho é tridimensional. A saída desejada do sistema será representada pelas informações de posição do efetuador e dos ângulos das seis juntas. Para obtenção de um conjunto de treinamento, os padrões de treinamento serão estabelecidos através de equações clássicas de robótica. A tarefa de posicionar o braço estará completa quando ele estiver situado em um dado alvo. O projeto se concentrará basicamente em simulações do sistema, realizadas no cenário estabelecido acima.

O objetivo acima será alcançado pela construção de um sistema neural de dois planos. No primeiro plano, uma rede estabelece a trajetória espacial do robô. No

segundo plano, uma outra rede transforma esta trajetória em uma sequência de ângulos das juntas do robô.

5.1.1. Manipulador PUMA 560

Para geração das trajetórias espaciais e de ângulos para treinamento e teste do sistema neural foi usado o software MATLAB para Windows. Neste software utilizou-se o toolbox Robotics para elaboração de programas que realizem os cálculos de robótica clássica de cinemática para a obtenção das informações de posições espaciais e angulares. Para isto, foi preciso escolher um modelo de braço de robô que já esteja equacionado, possua os parâmetros definidos e tenha seis graus de liberdade. O toolbox Robotics (Corke, 1996) apresenta um braço de robô com tais características, o manipulador PUMA560.

Figura 18: Fotografia do braço do robô PUMA560.

O manipulador PUMA560 possui seis graus de liberdade decompostos em: rotação da base, rotação do antebraço, rotação do braço e três graus de liberdade no punho, como mostra a Figura 19.

Figura 19. Esquema dos eixos de rotação

Os parâmetros de elos e juntas são definidos de acordo com as convenções de Denavit-Hartenberg como a seguir (Figura 20):

- ♦ o comprimento do elo a_i é a distância entre os eixos z_{i-1} e z_i ao longo do eixo x_i .
- ♦ a torção α_i do elo consiste no ângulo a partir do eixo z_{i-1} para o eixo z_i sobre o eixo x_i .
- ♦ o ajuste d_i do elo consiste na distância a partir do frame de origem $i-1$ para o eixo x_i ao longo do eixo z_{i-1} .
- ♦ o ângulo θ_i da junta representa o ângulo entre os eixos x_{i-1} e x_i sobre o eixo z_{i-1} .

Os eixos do manipulador PUMA560 são de revolução onde θ_i é a variável da junta e d_i é constante. Cada grau de liberdade do manipulador possui um intervalo de atuação. A rotação da base varia entre -160 e 160 graus, a rotação do antebraço está entre -225 e 45 graus, a rotação do braço está entre -45 e 225, a rotação do pulso está entre -110 e 170 graus, a capacidade de dobrar o pulso está entre -100 e 100 graus e a “flange” está entre -266 e 266 graus.

O manipulador PUMA560 (Cheng et al., 1997) apresenta os seguintes parâmetros: altura = 0.6604 m; antebraço = 0.43307 m; braço = 0.4318 m. Os demais parâmetros estão descritos na Tabela 1.

Tabela 1. Parâmetros do manipulador PUMA560.

i	θ_i	d_i	a_i	α_i
1	q_1	0	0	$\pi / 2$
2	q_2	0	0.4318	0
3	q_3	0.15005	0.0203	$-\pi / 2$
4	q_4	0.43307	0	$\pi / 2$
5	q_5	0	0	$-\pi / 2$
6	q_6	0	0	0

O software escolhido pode gerar trajetórias e executar funções como cinemática inversa e direta. Para utilizar estas funções foram necessários definir o tipo de manipulador, o lado do braço do robô, o tipo de aproximação ao ponto desejado e se o pulso é capaz de dobrar ou não. Para o nosso problema foi escolhido o braço de robô PUMA560, representado por (p560) com a seguinte configuração: braço posicionado do lado direito, a aproximação do manipulador é realizado de baixo para cima e o pulso não pode dobrar. Estes parâmetros são fornecidos às funções representadas pelo vetor $p=[1 \ -1 \ 1]$.

$$\text{No qual: } \begin{cases} 1 \Rightarrow \text{braço do lado direito} \\ -1 \Rightarrow \text{aproximação de baixo para cima} \\ 1 \Rightarrow \text{pulso sem dobrar} \end{cases}$$

Para gerar as trajetórias são fornecidos ao programa o ponto inicial, o ponto final e a quantidade de pontos que se deseja entre os pontos inicial e final. Uma vez feito isto, o programa produz uma trajetória discreta composta de 10 pontos, começando pelo ponto inicial e terminando pelo ponto final dados. Trajetórias assim geradas foram usadas para treinamento do sistema neural.

Uma vez estipulados os parâmetros e os pontos espaciais de cada trajetória, são calculados os ângulos dos últimos pontos de cada trajetória que se encontram em

um intervalo entre $-\pi$ e π . Estes ângulos são normalizados entre -1 e 1, dividem-se todos os ângulos gerados por π para serem usados no treinamento. Cada ponto espacial é associado a um conjunto de seis ângulos relativos aos graus de liberdade do robô $(\theta_1, \theta_2, \theta_3, \theta_4, \theta_5, \theta_6)$.

5.2. Propondo um modelo

Os modelos de memória associativa, anteriormente discutidos, permitem o levantamento de um conjunto de características desejadas para se propor um sistema neural que gere trajetória espacial. Alguns dos trabalhos estudados, de Anderson, Kohonen e Kohonen e Ruohonen, têm importância histórica e servem como base para o conhecimento de modelos de memória associativa.

As características mais importantes dos vários modelos de memória associativa discutidos são apontadas a seguir com vistas à proposição do novo modelo. Iniciando pelo modelo de Amari, este busca a aprendizagem de muitas seqüências ao mesmo tempo utilizando padrões com ruídos. O “Associatron” é capaz de recuperar uma entidade através da informação parcial de um ou mais padrões que compõem esta entidade, associar entidades através de padrões em comum e de gerar uma seqüência através dos padrões comuns a entidades armazenadas em instantes diferentes. Hopfield introduziu o conceito da função de Lyapunov para provar a estabilidade da rede. BAM armazena pares de padrões e os recupera a partir de entradas completas, incompletas ou ruidosas. Este é um algoritmo de treinamento rápido e que lida com padrões sem pré-processamento. MAM é uma extensão de BAM, composta por mais de duas camadas e tem como características: capacidade de associar a coexistência de n-tuplas, capacidade de recuperar n-tuplas a partir de informações parciais, robustez para informações ruidosas e treinamento rápido.

Deste modo, o modelo de memória associativa para produção de seqüência temporal deve armazenar padrões que não sejam ortogonais ou linearmente independentes, armazenar n-tuplas de informações, ser robusto a informações ruidosas e/ou incompletas, aprender seqüências temporais e ser treinável para melhorar sua capacidade de armazenamento. O modelo que mais se aproxima desta

descrição é MAM, no entanto ele precisa ser modificado para atender todos os requisitos listados. Além disto, a seqüência temporal de pontos espaciais da trajetória deve ser transformada em dados de ângulos de juntas.

Em síntese, o objetivo deste modelo é ter a capacidade de produzir seqüências temporais dadas versões de padrões de treinamento perfeitas, ruidosas ou incompletas destas seqüências. Estes padrões são constituídos por pontos de seqüências temporais e de posição espacial. Além disto, o novo modelo deve ser capaz de interpolar informações, gerar novas trajetórias e gerar os ângulos das juntas associados ao deslocamento temporal dos estados do sistema.

Para realizar a transformação das coordenadas espaciais em ângulos das juntas foi escolhido o modelo RBF para trabalhar em conjunto com o modelo de memória associativa. O modelo RBF foi escolhido por ser robusto a informações ruidosas ou incompletas, por apresentar rápido treinamento, por ser capaz de aproximar padrões de testes a padrões treinados, por apresentar capacidade de interpolação e por apresentar boa capacidade de armazenamento.

O modelo idealizado é composto por dois planos, como mostra a Figura 20. O primeiro plano é constituído por um modelo de memória associativa, chamado Memória Associativa Multidirecional Temporal (TMAM) (Araújo e Vieira, 1997) que é responsável pela associação entre as posições espaciais do efetuador em diferentes instantes de tempo. Este plano é formado por cinco camadas com quatro unidades de processamento cada (Seção 5.3). O segundo plano lida com informações angulares, é formado por 20 unidades de entrada e 6 unidades de saída que representam os ângulos das juntas. As unidades de processamento do plano 1 enviam sinais para aquelas do plano 2 que são totalmente conectadas às primeiras. Estas conexões associam cada seqüência temporal de posições espaciais a valores de ângulo do último ponto espacial desta seqüência (Seção 5.4).

Este modelo separa o processamento da informação em dois blocos. O primeiro bloco, representado pelo plano seqüência temporal, recebe informação de posição espacial do manipulador e gera uma seqüência destas posições. O segundo bloco, representado pelo plano ângulo, correlaciona os valores de ângulo com a seqüência dos últimos 5 pontos no espaço. As conexões entre planos associam a

seqüência temporal de posições espaciais com os ângulos do último ponto de tal seqüência.

Figura 20. Esboço do modelo proposto para controle e planejamento de trajetória.

No plano seqüência temporal, TMAM tem como entrada informações de posições espaciais do efetuador em cinco instantes diferentes de tempo e com seus respectivos valores de tempos. TMAM objetiva recuperar, gerar, interpolar e extrapolar estas posições espaciais. A saída é composta por cinco posições espaciais diferentes e seus respectivos tempos referentes ao resultado da recuperação, geração, interpolação e extrapolação do modelo.

O plano ângulo é responsável por produzir as informações de ângulo das juntas do braço. Este plano recebe informações de entrada, provenientes do plano seqüência temporal. As cinco últimas posições espaciais geradas e seus respectivos tempos constituem tais informações. O plano ângulo gera o valor de ângulo associado à última posição espacial produzida pelo plano seqüência temporal.

5.3. Modelo para o Plano Seqüência Temporal

TMAM apresenta duas mudanças importantes com relação a MAM: (i) representação contínua entre -1 e 1; (ii) realimentação autoassociativa em cada camada. A primeira modificação foi proposta para resolver o problema de representação da trajetória, de modo que as informações de posições espaciais são representados por valores reais e contínuos entre -1 e 1. Na segunda modificação é acrescentada uma realimentação autoassociativa para fornecer a cada unidade informações históricas de todas as outras unidades e dela mesma.

Neste plano, o número de camadas é determinado pelo tamanho da seqüência a ser armazenada e o número de unidades de cada camada depende da quantidade de informação armazenada. Foi escolhido para o plano seqüência temporal uma arquitetura com cinco camadas; e cada camada codifica um ponto espacial de uma dada trajetória, em um instante de tempo diferente (Figura 14). Logo são utilizadas quatro unidades de processamento para cada camada; as três primeiras unidades representam o ponto em coordenadas espaciais e a quarta unidade identifica o instante de tempo em que este ponto ocorre. As camadas são totalmente conectadas entre si, e cada camada possui conexões autoassociativas como mostra a Figura 21.

Figura 21. Conexões da camada 1 do modelo de TMAM para as outras quatro camadas e para ela mesma.

TMAM utiliza os dados de coordenadas espaciais (x, y, z) e seus respectivos tempos como entradas da rede. Portanto, cada camada recebe como entrada um ponto no espaço e seu tempo correspondente. As matrizes de correlação são construídas através de regra Hebbiana. A saída do modelo é formada pelo conjunto dos cinco pontos e seus respectivos instantes de tempo.

As matrizes de correlação entre as camadas l e k são calculadas por:

$$W_{lk} = \sum_{pad=1}^{np} (\mathbf{x}_l^{(pad)})^T \mathbf{x}_k^{(pad)} \quad (39)$$

As matrizes de correlação de uma mesma camada são construídas por:

$$W_{ll} = \sum_{pad=1}^{np} (\mathbf{x}_l^{(pad)})^T \mathbf{x}_l^{(pad)} \quad (40)$$

A regra de propagação entre a camada l e k é dada por:

$$\text{net}(k) = \sum_{\substack{k=1 \\ l \neq k}}^{nl} W_{lk} \mathbf{x}_k^T(k-1) + W_{ll} \mathbf{x}_l^T(k-1) \quad (41)$$

A regra de ativação da camada l é dada pela equação:

$$\mathbf{x}_i^{(l)}(t) = \tanh(\text{net}(k)) \quad (42)$$

A dinâmica de TMAM é descrita da seguinte maneira. Dado um padrão de entrada representado pelo vetor $(v^{(1)}, \dots, v^{(nc)})_{ini}$ que pode ser informação incompleta ou ruidosa do padrão desejado $(v^{(1)}, \dots, v^{(nc)})$, a rede troca de estados como mostramos a seguir:

Na qual: nc representa o número de camadas de TMAM.

$$(v^{(1)}, \dots, v^{(nc)})_{ini} \rightarrow W_{lk} \rightarrow (v^{(1)}, \dots, v^{(nc)})_1 \rightarrow \dots \rightarrow (v^{(1)}, \dots, v^{(nc)})_{fin} \rightarrow W_{lk} \rightarrow (v^{(1)}, \dots, v^{(nc)})_{fin}$$

O padrão de entrada $(v^{(1)}, \dots, v^{(nc)})_{fin}$ é apresentado à rede, gerando um novo estado através das matrizes de correlação. Cada novo estado gera outro estado. Este processo se encerra quando os estados da rede não mudam mais, gerando um padrão final $(v^{(1)}, \dots, v^{(nc)})_{fin}$ que deve ser o padrão desejado $(v^{(1)}, \dots, v^{(nc)})$. Entretanto, muitas vezes isto não ocorre.

5.3.1. Algoritmos de Aprendizagem

TMAM utiliza dois estágios de aprendizagem para aumentar a capacidade de armazenamento. No primeiro estágio as matrizes são construídas de acordo com as equações (39) e (40). No segundo estágio ocorre o treinamento supervisionado, com o objetivo de melhorar a capacidade de armazenamento do modelo.

Baseado nos bons resultados obtidos aplicando a Regra de Widrow-Hoff em BAM para melhorar sua capacidade de armazenamento (Araújo e Haga, 1997), foi aplicado, no segundo estágio, esta regra, para treinar as matrizes de correlação. Para diminuir o erro, foi usado, juntamente com a regra de Widrow-Hoff, o método de desaprendizagem proposto por Araújo e Haga (1997). Este método desaprende os atratores espúrios através da subtração da influência dos mesmos nas matrizes de correlação. Nesta fase de treinamento em uma arquitetura de 5 camadas, TMAM trabalha da seguinte maneira:

1. Um padrão inicial $(v^{(1)}, \dots, v^{(nc)})_{ini}$ é apresentado à rede;
2. Se a rede produzir um atrator espúrio $(s^{(1)}, \dots, s^{(nc)})_{ini}$ como saída;

2.1. Cada matriz de correlação é atualizada de acordo com a regra Widrow-Hoff:

$$\Delta W_{lk}(k) = (\lambda / (n + 1)) \left((x_k^{(pad)}(k) - s_k^{(pad)}(k))^T * x_l^{(pad)}(k - 1) \right) \quad (43)$$

$$\text{na qual} \left\{ \begin{array}{l} \lambda/(n+1) \text{ é a taxa de aprendizagem;} \\ \mathbf{x}_k^{(\text{pad})}, \mathbf{x}_l^{(\text{pad})} \text{ são os resultados desejados para as camadas } k \text{ e } l; \\ \mathbf{s}_k^{(\text{pad})} \text{ é um atrator espúrio na camada } k. \end{array} \right.$$

2.2. Depois a matriz de correlação entre as camadas l e k é atualizada pela regra de desaprendizagem:

$$\Delta W_{lk}(k) = -\left(\frac{\lambda}{n+1}\right) (\mathbf{s}_k^{(\text{pad})}(k))^T \mathbf{x}_l^{(\text{pad})}(k-1) \quad (44)$$

$$\text{na qual} \left\{ \begin{array}{l} \lambda/(n+1) \text{ é a taxa de aprendizagem;} \\ \mathbf{x}_l^{(\text{pad})} \text{ são os resultados desejados para a camada } l; \\ \mathbf{s}_k^{(\text{pad})} \text{ é o atrator espúrio na camada } k. \end{array} \right.$$

3. Em caso contrário, pare.

5.4. Modelo para o plano ângulo

Este plano é formado pelo modelo de redes de função base radial. O plano é composto por uma camada de entrada com 20 unidades, uma camada escondida de 20 unidades e uma camada de saída com 6 unidades. A camada de entrada é responsável pela distribuição da informação para a camada escondida, a camada escondida é responsável pelo processamento não linear da informação e a camada de saída é o somatório dos resultados a partir do processamento da camada escondida.

A entrada da rede RBF (Hassoun, 1995) é formada pelas saídas de cada camada do plano sequência temporal, que é composta por 5 camadas, e cada uma possui 4 unidades. A composição destas informações forma o vetor de entrada com 20 unidades. A saída desejada é o conjunto de seis ângulos correspondentes ao quinto ponto espacial pertencente à sequência gerada pelo primeiro plano. O objetivo é associar, para cada padrão de entrada de treinamento, o conjunto de ângulos correspondente ao último ponto. A rede RBF foi escolhida por ser uma classe de

modelos apropriados para aproximar valores reais contínuos. Esta aproximação depende da função base radial e do raio escolhidos.

Para os testes, o RBF usa a distância Euclidiana para aproximar os vetores de entrada de teste a um dos vetores treinados e assim produzir como resposta a saída correspondente ao vetor treinado que mais se aproxime do vetor de teste.

Figura 22. Arquitetura de conexões entre o plano 1 e plano 2.

Para o treinamento da RBF é fornecida à rede uma vetor coluna de entrada formada por pad padrões de dimensão 20. Em seguida, são determinados os centros, formados pelos pad padrões de entrada. Em seguida, o programa determina o número de núcleos que é associado um vetor de saída de dimensão 6.

Uma vez determinado o tamanho da matriz, fornece-se à rede o tamanho do raio que formará a bacia de atração para cada núcleo; em seguida, é fornecido o tipo de função radial a ser usada. No modelo proposto, foi escolhida a função Cauchy;

esta apresentou, em média, melhores resultados quando comparada com as demais funções.

A dinâmica do modelo RBF para a construção da matriz de pesos é descrita a seguir:

- 1) definição dos tamanhos das matrizes de entrada para treinamento, da matriz de teste e da matriz de saída desejada;
- 2) definição do tamanho do raio r ;
- 3) cálculo da função Cauchy dada pela equação abaixo,

$$h_j(\mathbf{u}^{(pad)}) = \frac{1}{1 + z_j^2} \quad (45)$$

$$\text{na qual: } \left\{ z_j = \frac{\|\mathbf{u}^{(pad)} - \mathbf{c}^{(j)}\|}{r_j^2}, \right.$$

- 4) cálculo da matriz de pesos dada pela expressão (38)

O funcionamento do modelo RBF para teste de padrões não treinados é descrito a seguir:

- 1) é fornecida à rede a matriz de padrões para teste;
- 2) calcula-se a nova matriz H usando-se a função Cauchy da etapa 3 do processo de treinamento;
- 3) calcula-se a nova saída de acordo com a equação (37)

Para a escolha dos parâmetros raio e função radial tanto na etapa da construção da matriz de pesos como na etapa de testes, foi necessário estudar o comportamento deles para a escolha do valor do raio que melhor se adequasse ao modelo e qual função apresenta melhor aproximação.

O primeiro teste realizado foi em relação ao tamanho do raio a ser definido pelo usuário. De acordo com os testes observados, podem-se tirar as seguintes conclusões:

- ♦ se o tamanho do raio for muito pequeno, o modelo não consegue abranger todos os pontos que deveria.

- ♦ se o tamanho do raio for muito grande, pode haver sobreposição de raios, podendo ocorrer a atração de pontos que não pertencem ao seu conjunto.

Uma vez estudadas estas características, o valor de raio que apresentou melhor resultado foi $\text{raio} = 1$, que será usado nos testes do modelo.

O segundo teste foi escolher a função radial a ser usada. Três funções foram analisadas e testadas para recuperação das trajetórias. Foi usado o software MATLAB para testar as funções: Gaussiana, a função multiquadrática e a função Cauchy. A função que apresentou melhor resultado, função Cauchy, é utilizada neste trabalho.

Para o treinamento do modelo, foram usadas as mesmas trajetórias utilizadas para treinar o primeiro plano. O treinamento foi realizado de duas maneiras. Na primeira, o modelo foi treinado e analisado separadamente para cada trajetória única, seguindo o mesmo padrão de treinamento do primeiro plano. Na segunda maneira, a rede está sendo testada para lidar com cada tipo de trajetória múltipla separadamente, cujas trajetórias múltiplas são divididas em quatro tipos, n-1, 1-m, 1-1 sem cruzamento e 1-1 com cruzamento.

Em ambas as maneiras, o modelo é treinado para fornecer como saída o conjunto de seis ângulos correspondentes a cada último ponto da trajetória treinada. O objetivo é analisar o comportamento do modelo em relação à recuperação, interpolação e generalização.

6. Simulações

Este capítulo descreve como foram gerados os diferentes tipos de trajetórias que serão utilizadas para treinamento do sistema completo. Descreve também os resultados obtidos por TMAM e RBF, quando utilizados para lidar com trajetórias únicas e múltiplas.

Para tal, é reportado na Seção 6.1 como foram organizados os vários testes para o sistema neural. Na Seção 6.2, a metodologia para a execução dos testes e os resultados do plano seqüência temporal são descritos. Os resultados do plano ângulo são apresentados na Seção 6.3.

6.1. Obtenção dos Dados para Simulação

Para gerar padrões de treinamento foi utilizado o software MATLAB para Windows. Neste software utilizou-se o toolbox Robotics para elaboração de programas que realizem os cálculos de robótica clássica, cinemática e dinâmica. Estes resultados levantaram informações de posições espaciais e angulares. Para obter tais informações, é preciso escolher um modelo de braço e definir seus parâmetros. Para isto foi escolhido o modelo Puma 560. O braço escolhido é composto de seis graus de liberdade, três graus para posicionamento e três graus para orientação do efetuador.

A obtenção dos dados foi dividida em duas etapas: (i) geração de trajetória cartesiana; (ii) cálculo da cinemática inversa.

Na etapa da geração da trajetória, foram gerados nove grupos de trajetórias. As cinco primeiras são trajetórias únicas com nível de complexidade variada. Os quatro últimos conjuntos são trajetórias múltiplas. Os padrões de trajetórias são:

- (1) Trajetória composta por seis pontos e de baixa complexidade.
- (2) Trajetória composta por oito pontos e de complexidade maior em relação à 1ª. trajetória.
- (3) Trajetória composta por oito pontos e de complexidade maior em relação à 2ª. trajetória.
- (4) Trajetória composta por oito pontos e de complexidade maior em relação à 3ª. trajetória.
- (5) Trajetória composta por oito pontos e de complexidade maior em relação à 4ª. trajetória.

A complexidade das trajetórias únicas é estabelecida pelo número de mudanças de direção que ela realiza ao longo do tempo.

Para trajetórias múltiplas, cada um dos quatro próximos grupos define duas trajetórias partindo de n pontos para chegar a m pontos, onde $n = 1, 2$ e $m = 1, 2$. Logo, as trajetórias múltiplas são definidas como:

- (6) Trajetórias $m-1$: trajetórias que possuem o mesmo ponto final e pontos iniciais diferentes.

Figura 23. Esboço de trajetória do tipo 2-1.

- (7) Trajetórias $1-n$: trajetórias que possuem o mesmo ponto inicial e pontos finais diferentes.

Figura 24. Esboço de trajetória do tipo 1-2.

(8) Trajetórias 1-1 sem cruzamento: trajetórias com pontos iniciais e finais diferentes sem possuir um ponto em comum entre elas.

Figura 25. Esboço de trajetória do tipo 1-1 sem pontos de contato entre elas.

(9) Trajetórias 1-1 com cruzamento: trajetórias com pontos iniciais e finais diferentes, mas cruzando-se em um ponto comum.

Figura 26. Esboço de trajetória do tipo 1-1 com pontos intermediários em comum.

Para gerar as trajetórias, são fornecidos ao programa do MATLAB o ponto inicial e o ponto final, ambos em coordenadas espaciais, e a quantidade de pontos que se deseja interpolar entre eles. O programa produz como saída uma trajetória entre esses pontos; e cada ponto espacial está associado a um instante de tempo diferente. A partir do item 6, as trajetórias são compostas de cinco pontos em instantes de tempo diferentes, com o objetivo de treinar a rede com mais do que uma trajetória ao mesmo tempo. A segunda etapa da geração de padrões de teste realiza o cálculo da cinemática inversa, transformando os pontos gerados pela etapa anterior em coordenadas angulares.

6.2. Testando Plano Sequência Temporal

Os primeiros testes foram realizados em relação à arquitetura TMAM que compõe o primeiro plano, com o objetivo de estudar seu comportamento em relação ao armazenamento de informações de posições espaciais, a habilidade da rede em recuperar as seqüências treinadas, a capacidade de interpolação e extrapolação de pontos na trajetória e sua habilidade em lidar com diferentes trajetórias.

Os dados para o treinamento e teste do modelo foram gerados como descrito na Seção 6.1. Em resumo, é dado o ponto inicial, o ponto final e a quantidade de pontos que se deseja entre os pontos dados. Cada ponto é associado a um instante de tempo diferente. A representação de cada posição espacial é real e limitada, variando entre -1 e 1. Os testes foram divididos em dois tipos. O primeiro tipo é realizado com trajetórias únicas formadas pelas trajetórias 1, 2, 3, 4 e 5 da Seção 6.1, para testar a capacidade do modelo em realizar “tracking”, isto é, se o modelo é capaz de seguir uma trajetória. O segundo tipo é realizado com múltiplas trajetórias, do tipo 6, 7, 8 e 9, mencionadas na Seção 6.1, para estudar a capacidade do modelo em lidar com mais de uma trajetória. Em todos os casos o espaço de trabalho varia de -1 até 1. São considerados pontos próximos, aqueles que se encontram em um vizinhança do ponto desejado. Esta vizinhança foi arbitrariamente estipulada em 5% do total de cada dimensão.

6.2.1. Testes com Trajetória Única

Neste caso, as trajetórias possuem graus diferentes de complexidade e foram treinadas uma de cada vez. O objetivo dos testes é analisar se o modelo é capaz de gerar, recuperar, interpolar e extrapolar pontos em uma trajetória. Para cada trajetória gerada pelo MATLAB, foram escolhidos 8 ou 5 pontos para representá-la de acordo com seu grau de complexidade. Para as trajetórias que apresentam mudanças de direção significativa e brusca o autor escolheu 8 pontos para treinamento, e para as trajetórias mais suaves foram escolhidos 5 pontos para treinamento.

Para escolher 5 e 8 pontos o problema foi analisado da seguinte forma: uma trajetória que mais se aproxima de uma reta não precisa de muitos pontos para representá-la; já aquela que possui muitas quebras precisaria de mais pontos para representá-la. Na arquitetura proposta, cada camada representa um ponto em um instante de tempo particular. Nos casos de trajetórias compostas por cinco pontos, podemos representar cada trajetória completa sem precisar realizar defasagem de pontos. Nas trajetórias com oito pontos, a representação da trajetória no modelo de cinco camadas é realizada através de quatro padrões de treinamento como mostrados na Tabela 2.

Tabela 2. Tabela de distribuição dos pontos das trajetórias únicas para treinamento de TMAM.

camadas	Padrão 1		Padrão 2		Padrão 3		Padrão 4	
	t(s)	ponto	t(s)	ponto	t(s)	ponto	t(s)	ponto
1	0.01	1	0.02	2	0.03	3	0.04	4
2	0.02	2	0.03	3	0.04	4	0.05	5
3	0.03	3	0.04	4	0.05	5	0.06	6
4	0.04	4	0.05	5	0.06	6	0.07	7
5	0.05	5	0.06	6	0.07	7	0.08	8

Foi observado, que com trajetórias compostas de oito pontos, o modelo consegue recuperar, interpolar e extrapolar pontos na trajetória com uma certa porcentagem de erro. Para diminuir este erro, pode-se empregar duas maneiras: (i) aumentar o número de padrões a serem treinados ou (ii) aumentar o número de camadas no primeiro plano. A primeira opção pressupõe que o aumento do número de pontos das trajetórias mais complexas possibilite codificar pontos mais próximos das quebras para treinamento. Deste modo, o modelo apresentaria um erro muito menor, gerando uma trajetória mais próxima da treinada. A segunda opção aumentaria o número de camadas de 5 para 6, 7 ou até mesmo 8, para não ter que subdividir a trajetória na hora do treinamento.

É importante observar que, em se tratando de treinamento de trajetórias únicas, elas foram treinadas separadamente, para que duas trajetórias não fossem

interpretadas com uma só, pois, quando ocorre a subdivisão de trajetórias com mais do que cinco pontos, a rede pode interpretar o começo de uma trajetória como continuação da outra, sendo isto não real.

Os testes realizados nesta etapa utilizam o modelo formado por cinco camadas, cada uma com quatro unidades de processamento; três unidades representam a posição espacial de um ponto e a quarta unidade representa o tempo associado a esta posição. Utilizam-se as três primeiras trajetórias descritas na Seção 6.1. Nesta bateria de testes, o modelo estava utilizando somente a regra Widrow-Hoff para treinamento dos pesos da rede.

Na primeira etapa, as três primeiras trajetórias únicas foram testadas. As matrizes de correlação são construídas de acordo com as equações (39) e (40) e são treinadas de acordo com a equação (43) para testar a capacidade de recuperação, geração, interpolação e extrapolação de seqüências. As trajetórias levaram 2421, 736 e 212 épocas de treinamento para a rede alcançar o erro final de 0.0606, 0.0528 e 0.1446 respectivamente. Nas Figuras 27, 28 e 29 as trajetórias obtidas são representadas por linhas pontilhadas e círculos, e as seqüências desejadas são representadas por linhas sólidas e asteriscos. Os pontos mostrados na parte superior representam os pontos finais e os pontos mostrados na parte inferior representam os pontos iniciais.

No primeiro conjunto de testes, a capacidade da rede em recuperar a seqüência treinada é testada. Neste caso, são fornecidos à rede os pontos espaciais ruidosos de cada seqüência e os tempos treinados para recuperar a seqüência desejada. Como se pode observar os resultados mostrados na Figura 27, a capacidade do modelo diminui conforme a complexidade da tarefa cresce. Com isto observa-se que os pontos que mudam bruscamente de direção têm uma recuperação mais pobre se comparados com os pontos que mudam suavemente.

Figura 27. Recuperação dos pontos treinados da seqüência temporal para: (a) seqüência 1; (b) seqüência 2 e (c) seqüência 3.

O segundo conjunto de experimentos tem como objetivo avaliar a capacidade da rede em interpolar pontos de uma seqüência. Isto é feito apresentando à rede um conjunto de quatro pontos treinados. O ponto que deverá ser interpolado, em qualquer posição da seqüência, tem o valor da posição espacial do ponto anterior ou posterior com o valor do tempo desejado. A Figura 28 mostra a capacidade da rede em interpolar pontos.

Figura 28. Recuperação dos pontos interpolados da seqüência temporal para: (a) seqüência 1; (b) seqüência 2 e (c) seqüência 3.

O terceiro grupo de testes avalia a capacidade da rede em extrapolar pontos em uma seqüência temporal. São fornecidos, como entrada, para a rede, 4 pontos treinados ou interpolados e os tempos respectivos; e no 5º. ponto repete-se o último ponto já conhecido juntamente com o tempo desejado. Observa-se que a rede é capaz de extrapolar em ambas as direções da trajetória.

Figura 29. Recuperação dos pontos interpolados e extrapolados treinados da sequência temporal para: (a) sequência 1; (b) sequência 2 e (c) sequência 3.

No conjunto de testes apresentado acima, observa-se que o modelo é capaz de recuperar, interpolar e extrapolar as seqüências treinadas, onde a complexidade da trajetória influencia no desempenho da rede. Observa-se também que o modelo apresenta discrepâncias entre a trajetória desejada e a obtida na segunda casa decimal e muitas vezes em apenas um dos eixos. Para melhorar esta capacidade e diminuir as diferenças, as trajetórias acima foram treinadas e testadas novamente, utilizando-se a regra de Desaprendizagem juntamente com a de Widrow-Hoff para treinamento das matrizes de pesos (Araújo e Vieira, 1997a).

Nesta segunda bateria, as três trajetórias anteriores são testadas novamente, mudando o método de aprendizagem, visando diminuir o erro final da rede. As matrizes de correlação são construídas de acordo com as equações (39) e (40), e são treinadas de acordo com as equações (43) e (44), para testar a capacidade de recuperação, geração, interpolação e extrapolação de seqüências. As trajetórias são treinadas e testadas uma de cada vez. As trajetórias levaram 3000, 849 e 484 épocas de treinamento para alcançar o erro final da rede de 0.06, 0.033 e 0.039 respectivamente. Nas Figuras 30, 31 e 32 as trajetórias obtidas são representadas por linhas pontilhadas e círculos e as seqüências desejadas são representadas por linhas sólidas e asteriscos. Os pontos mostrados na parte superior representam os pontos finais e os pontos mostrados na parte inferior representam os pontos iniciais.

No primeiro conjunto de testes a capacidade da rede em recuperar a seqüência treinada é testada, como mostra a Figura 30. Neste caso, são fornecidos à rede os

pontos espaciais ruidosos de cada seqüência e os tempos treinados para recuperar a seqüência desejada.

Figura 30. Recuperação dos pontos treinados da seqüência temporal para: (a) seqüência 1; (b) seqüência 2 e (c) seqüência 3.

O segundo conjunto de experimentos tem como objetivo avaliar a capacidade da rede em interpolar pontos de uma seqüência. Isto é feito, apresentando à rede um conjunto de quatro pontos treinados. O ponto que deverá ser interpolado, em qualquer posição da seqüência, tem o valor da posição espacial do ponto anterior ou posterior com o valor do tempo desejado. A Figura 31 mostra a capacidade da rede em interpolar pontos.

Figura 31. Recuperação dos pontos interpolados da seqüência temporal para: (a) seqüência 1; (b) seqüência 2 e (c) seqüência 3.

O terceiro grupo de testes avalia a capacidade da rede em extrapolar pontos em uma seqüência temporal. São fornecidos, como entrada, à rede, 4 pontos treinados ou interpolados e os tempos respectivos, e no primeiro ou quinto ponto repete-se o ponto conhecido mais próximo com o tempo desejado. Observa-se que a rede é capaz de extrapolar em ambas as direções da trajetória.

Figura 32. Recuperação dos pontos interpolados e extrapolados treinados da sequência temporal para: (a) sequência 2 e (b) sequência 3.

Os resultados do modelo treinado pela composição dos dois métodos de treinamento sugerem que ele possui melhor desempenho que o modelo que utilizou apenas o método de Widrow-Hoff para treinamento da rede. Observa-se também que, na recuperação dos pontos interpolados e extrapolados, o modelo não conseguiu extrapolar pontos para a primeira sequência.

Em uma terceira etapa de testes, a rede é avaliada em sua capacidade de gerar as sequências treinadas. As matrizes são construídas de acordo com as equações (39) e (40), e são treinadas de acordo com as equações (43) e (44), com o objetivo de testar a capacidade da rede em gerar as sequências treinadas. Na Figura 33, as trajetórias obtidas são representadas por linhas pontilhadas e círculos; e as sequências desejadas são representadas por linhas sólidas e asteriscos. Os pontos mostrados na parte superior representam os pontos finais e os pontos mostrados na parte inferior representam os pontos iniciais.

As sequências levaram 3000, 849 e 484 épocas de treinamento para alcançar o erro final da rede de 0.0600, 0.0333 e 0.0390. Na Figura 33, as trajetórias obtidas são representadas por linhas pontilhadas e as sequências desejadas são representadas por linhas sólidas. Este conjunto de testes analisa a capacidade da rede em gerar sequências. Neste caso, o primeiro ponto espacial treinado de cada sequência é fornecido para as quatro primeiras camadas juntamente com os respectivos tempos; e na quinta camada é fornecido o ponto meta, com o objetivo de gerar toda sequência, como mostra a Figura 33.

Figura 33. Geração das seqüências: (a) seqüência 1; (b) seqüência 2 e (c) seqüência 3.

Como se pode observar nos testes realizados, a arquitetura proposta é capaz de recuperar, gerar, interpolar e extrapolar seqüências. Uma vez de posse desses resultados, as outras duas últimas trajetórias de graus de complexidade diferentes foram treinadas e testadas. Nesta quarta etapa, as matrizes são construídas de acordo com as equações (39) e (40), e são treinadas de acordo com as equações (43) e (44).

As seqüências 4 e 5 levaram 369 e 360 épocas de treinamento para alcançar o erro final da rede de 0.2298 e 0.148. Nas Figuras 34 e 35, as trajetórias obtidas são representadas por linhas pontilhadas e as seqüências desejadas são representadas por linhas sólidas. Os pontos mostrados na parte superior representam os pontos finais e os pontos mostrados na parte inferior representam os pontos iniciais.

Figura 34. Testes para seqüência 4: (a) recuperação; (b) interpolação e (c) extrapolação.

Figura 35. Testes para sequência 5: (a) recuperação; (b) interpolação e (c) extrapolação.

Como se pode observar, a rede consegue recuperar, interpolar e extrapolar pontos nas diferentes trajetórias treinadas e testadas. Observa-se também nas partes sem mudanças bruscas de direção o modelo apresenta pequenas diferenças entre as trajetórias treinadas e obtidas. Tais discrepâncias aumentam com a complexidade da trajetória. A partir dos resultados destes testes, partiu-se para analisar o modelo quando treinado com mais de uma trajetória simultaneamente.

6.2.2. Testes com Trajetórias Múltiplas

Para os testes com os tipos de trajetórias múltiplas, citadas no item 6.1, foram escolhidos 5 pontos para representarem cada trajetória, por serem estas trajetórias suaves (Vieira e Araújo, 1997). Cada trajetória é representada por cinco pontos com seus respectivos tempos. Por possuir cinco camadas o modelo TMAM associa a cada camada um instante de tempo e, a esta camada, o ponto correspondente de cada trajetória. A representação de cada trajetória ocorre da seguinte maneira:

Tabela 3. Tabela de distribuição dos pontos das trajetórias múltiplas para treinamento de TMAM.

camadas	trajetória 1	
	tempo(s)	ponto
1	0.01	1
2	0.02	2
3	0.03	3
4	0.04	4
5	0.05	5

O objetivo destes testes é treinar a rede com mais do que uma trajetória simultaneamente, observando se ela consegue recuperar, gerar, interpolar e extrapolar pontos na trajetória. Observa-se também se, ao testar uma trajetória não treinada, a rede aproxima esta trajetória a uma trajetória treinada ou se ela gera uma trajetória interpolando pontos entre as duas trajetórias treinadas.

Os testes realizados nesta etapa utilizam o modelo formado por cinco camadas, cada uma com quatro unidades de processamento; três unidades representam a posição espacial de um ponto e a quarta unidade representa o tempo associado a este ponto. As trajetórias são aquelas definidas na Seção 6.1.

O modelo foi treinado com duas trajetórias, para cada caso citado acima. Nesta etapa de testes, as matrizes de correlação são construídas de acordo com as equações (39) e (40), e são treinadas de acordo com a equação (43) e (44). Os testes realizados para avaliação do desempenho do modelo utilizaram:

- A) pontos inicial e final treinados;
- B) ponto inicial ruidoso e ponto final treinado;
- C) ponto inicial treinado e ponto final ruidoso;
- D) pontos inicial e final ruidosos.

Os testes acima citados foram realizados da seguinte maneira: foi fornecido, na primeira camada, o ponto inicial treinado ou ruidoso com o instante de tempo correto, nas camadas 2, 3 e 4 foram repetidos o ponto inicial treinado ou ruidoso, mas com o instante de tempo correto em cada camada; e na quinta camada é fornecido o

ponto final treinado ou ruidoso. Os quatro conjuntos de trajetórias citados levaram respectivamente 426, 302, 682 e 193 épocas para convergirem para um erro de 0.106, 0.11, 0.034 e 0.164 respectivamente. O objetivo destes testes é analisar se o modelo é capaz de produzir tais seqüências.

Nos gráficos a seguir, o traço contínuo representa a trajetória desejada, e a linha tracejada representa a trajetória obtida.

Testes das trajetórias tipo n-1.

Para este conjunto de testes, a rede foi treinada com duas trajetórias simultaneamente e testada sua capacidade de produzir cada trajetória separadamente. Os pontos mostrados na parte superior das Figuras 36 a 39 representam os pontos iniciais, e os pontos mostrados na parte inferior representam os pontos finais.

Figura 36. Testes de produção das trajetórias com ponto inicial treinado e ponto final treinado: (a) trajetória 1 e (b) trajetória 2.

Na Figura 36, o modelo tende a atingir o ponto final treinado. A trajetória obtida em (a) apresenta uma melhor aproximação da trajetória desejada em relação a trajetória obtida em (b), mas em ambos os casos o ponto final obtido está dentro da vizinhança preestabelecida.

(a)

(b)

Figura 37. Testes de produção das trajetórias com ponto inicial treinado e ponto final ruidoso (a) trajetória 1 e (b) trajetória 2.

A Figura 37 mostra que o modelo não rastreia as trajetórias treinadas; na trajetória obtida em (a), o modelo conseguiu recuperar o ponto final desejado enquanto que a trajetória obtida em (b), o ponto final encontrado está na vizinhança estabelecida.

(a)

(a)

Figura 38. Testes de recuperação das trajetórias com pto. inicial ruidoso e pto. final treinado: (a) trajetória recuperada.

Figura 39. Testes de recuperação das trajetórias com ponto inicial ruidoso e final ruidoso: (a) trajetória recuperada.

As Figuras 38 e 39 mostram que o modelo atinge o ponto final das trajetórias treinadas e apresenta a capacidade de interpolar novas trajetórias àquelas treinadas. Isto é verdade quando o ponto inicial do teste está entre aqueles treinados.

Como pode ser observado nos testes realizados, para as trajetórias do tipo n-1 o modelo é capaz de recuperar as seqüências treinadas com diferenças nos pontos intermediários. Nas Figuras 38 e 39, o modelo interpolou uma trajetória entre as duas treinadas; as trajetórias recuperadas 1 e 2 são idênticas e representam a trajetória interpolada e são representadas pela linha tracejada.

Testes das trajetórias tipo 1-m .

Para este conjunto de testes, a rede foi treinada com duas trajetórias simultaneamente e testada em sua capacidade de produzir cada trajetória separadamente. Os pontos mostrados na parte superior das Figuras 40 a 43 representam os pontos finais, e os pontos mostrados na parte inferior representam os pontos iniciais.

Figura 40. Testes de produção das trajetórias com ponto inicial treinado e ponto final treinado: (a) trajetória 1 e (b) trajetória 2.

A Figura 40 mostra que o modelo seguiu as trajetórias treinadas. A trajetória obtida em (a) apresenta uma melhor aproximação da trajetória desejada em relação à trajetória obtida em (b). A trajetória obtida em (a) se aproximou mais do ponto final desejado em relação à trajetória obtida em (b). Contudo, ambas possuem diferenças entre o desejado e o obtido.

Figura 41. Testes de produção das trajetórias com ponto inicial ruidoso e ponto final treinado: (a) trajetória 1 e (b) trajetória 2.

A Figura 41 mostra que o modelo conseguiu seguir as trajetórias treinadas para um ponto inicial ruidoso. As trajetórias obtidas ilustram que o modelo se comporta analogamente ao caso no qual os pontos iniciais e finais não tem ruídos (Figura 4c).

Figura 42. Testes de produção das trajetórias com ponto inicial treinado e ponto final ruidoso: (a) trajetória 1 e (b) trajetória 2.

No teste representado pela Figura 42, o modelo acompanha mas não rastreia as trajetórias treinadas. A trajetória obtida representada em (a) apresentou uma melhor recuperação em relação à trajetória obtida representada pela letra (b).

Figura 43. Testes de produção das trajetórias com ponto inicial ruidoso e ponto final ruidoso.

A Figura 43 mostra que o modelo foi capaz de interpolar uma trajetória entre as duas treinadas, onde a trajetória obtida aproxima-se mais da trajetória treinada, representada pela letra (a) nos outros gráficos.

Como se pode observar nos testes mostrados, o modelo é capaz de recuperar as seqüências treinadas aproximadamente, mas não as reproduz precisamente. Na Figura 43, o modelo interpolou uma trajetória entre as duas treinadas; as trajetórias recuperadas 1 e 2 são idênticas e representam a trajetória interpolada e são representadas pela linha tracejada. A trajetória interpolada se aproxima muito de uma das trajetórias treinadas.

Testes das trajetórias tipo 1-1 sem cruzamento

Para este conjunto de testes a rede foi treinada com duas trajetórias e testada sua capacidade de reproduzir cada trajetória. Os pontos mostrados na parte superior esquerda das Figuras 44 a 47 são os pontos iniciais, e os pontos mostrados na parte inferior direita são os pontos finais.

Figura 44. Testes de produção das trajetórias com ponto inicial e ponto final treinados: (a) trajetória 1 e (b) trajetória 2.

A Figura 44 mostra que o modelo conseguiu seguir as trajetórias treinadas sem precisão para ser considerado rastreamento. A trajetória obtida em (a) apresenta, em média, uma melhor aproximação da trajetória treinada em relação à trajetória obtida em (b).

Figura 45. Testes de produção das trajetórias com ponto inicial ruidoso e ponto final treinado: (a) trajetória 1 e (b) trajetória 2.

A Figura 45 mostra que o modelo conseguiu seguir as trajetórias treinadas, mesmo fornecendo um ponto inicial ruidoso como entrada. A trajetória obtida em (a) apresenta uma aproximação melhor do ponto inicial que aquela da trajetória plotada

em (b). Enquanto que esta última trajetória consegue alcançar e até ultrapassar o ponto final.

Figura 46. Testes de produção das trajetórias com ponto inicial treinado e ponto final ruidoso: (a) trajetória 1 e (b) trajetória 2.

A Figura 46 mostra que o modelo conseguiu seguir proximamente as trajetórias treinadas. A trajetória obtida em (a) apresenta uma aproximação melhor do ponto inicial, enquanto a trajetória obtida em (b) consegue alcançar e até ultrapassar o ponto final.

Figura 47. Testes de produção das trajetórias com ponto inicial ruidoso e ponto final ruidoso.

Na Figura 47, o modelo foi capaz de interpolar uma trajetória entre as duas treinadas. É importante destacar que as trajetórias treinadas não possuem nem um ponto em comum.

Como se pode observar nos testes realizados acima, o modelo é capaz de recuperar as seqüências treinadas com pequenas diferenças, contudo não faz rastreamento. Na Figura 47, o modelo interpolou uma trajetória nova entre as duas treinadas. As trajetórias recuperadas são idênticas e são representadas na figura pela linha tracejada.

Testes das trajetórias tipo 1-1 com cruzamento.

Para este conjunto de testes, a rede foi treinada com duas trajetórias, e testada sua capacidade de produzir cada trajetória separadamente. Os pontos mostrados à esquerda, nas Figuras 48 à 51 (a), representam os pontos iniciais, e os pontos mostrados à direita representam os pontos finais. Os pontos mostrados na parte superior nas Figuras 48 à 51 (b) representam os pontos iniciais, e os pontos mostrados na parte inferior representam os pontos finais.

Figura 48. Testes de produção das trajetórias com ponto inicial treinado e ponto final treinado: (a) trajetória 1 e (b) trajetória 2.

(a)

(b)

Figura 49. Testes de produção das trajetórias com ponto inicial ruidoso e ponto final treinado: (a) trajetória 1 e (b) trajetória 2.

(a)

(b)

Figura 50. Testes de produção das trajetórias com ponto inicial treinado e ponto final ruidoso: (a) trajetória 1 e (b) trajetória 2.

Figura 51. Testes de produção das trajetórias com ponto inicial ruidoso e ponto final ruidoso.

Nos testes representados pelas Figuras 48 a 51, o modelo não apresenta boa recuperação das seqüências treinadas quando elas possuem pontos de cruzamento e são quase perpendiculares entre si. Os pontos estão fora da vizinhança desejada. Na Figura 51, as duas trajetórias recuperadas são idênticas e são representadas pela linha tracejada.

Para uma melhor análise deste tipo de trajetória, foram geradas e testadas duas outras seqüências que se cruzam. Os pontos mostrados na parte superior das Figuras 52, 53, 54 e 55, representam os pontos finais, e os pontos mostrados na parte inferior representam os pontos iniciais.

Figura 52. Testes de produção das trajetórias com ponto inicial treinado e ponto final treinado: (a) trajetória 1 e (b) trajetória 2.

A Figura 52 mostra que o modelo conseguiu seguir aproximadamente as trajetórias treinadas. No item (a), a trajetória obtida está mais próxima da trajetória desejada que a trajetória obtida no item (b). Em compensação, a trajetória obtida no item (b) conseguiu extrapolar o ponto final desejado. Em ambos os casos, o ponto final se localiza dentro da vizinhança desejada.

Figura 53. Testes de produção das trajetórias com ponto inicial ruidoso e ponto final treinado: (a) trajetória 1 e (b) trajetória 2.

A Figura 53 mostra que o modelo conseguiu seguir aproximadamente as trajetórias treinadas. No item (a), a trajetória obtida está mais próxima da trajetória desejada que a trajetória obtida no item (b). Em compensação, a trajetória obtida no item (b) conseguiu atingir e extrapolar o ponto final desejado.

Figura 54. Testes de produção das trajetórias com ponto inicial treinado e ponto final ruidoso: (a) trajetória 1 e (b) trajetória 2.

A Figura 54 mostra que o modelo conseguiu seguir aproximadamente as trajetórias treinadas. No item (a), a trajetória obtida está mais próxima da trajetória desejada que a trajetória obtida no item (b). Em compensação, a trajetória obtida no item (b) conseguiu atingir e extrapolar o ponto final desejado, e a trajetória obtida no item (a) conseguiu alcançar o ponto final.

Figura 55. Testes de produção das trajetórias com ponto inicial ruidoso e ponto final ruidoso: (a) trajetória 1 e (b) trajetória 2.

A Figura 55 mostra que o modelo conseguiu seguir aproximadamente as trajetórias treinadas. No item (a), a trajetória obtida está mais próxima da trajetória desejada que a trajetória obtida no item (b). Em compensação, a trajetória obtida no item (b) conseguiu atingir e extrapolar o ponto final desejado, e a trajetória obtida no item (a) conseguiu alcançar o ponto final.

Como se pode observar nos testes realizados acima, o modelo é capaz de produzir aproximações das seqüências treinadas mas não é capaz de rastrear-las. Observa-se também que o modelo apresentou uma reprodução melhor das seqüências treinadas quando elas não possuem trajetórias muito diferentes.

6.3. Testes do Plano ângulo

Esta seção descreve os testes realizados para o modelo RBF, descrito na Seção 5.4. O objetivo é realizar a cinemática inversa, transformando os pontos espaciais das trajetórias geradas no plano de seqüência temporal em posições angulares do manipulador e observar se a rede é capaz de recuperar, interpolar e generalizar pontos da trajetória.

O treinamento do modelo é realizado em duas etapas. Na primeira etapa, o modelo RBF é treinado somente com as trajetórias únicas; e na segunda etapa, é treinado somente com as trajetórias múltiplas. Os dados para teste são os dados gerados no plano seqüência temporal.

O modelo é composto por 3 camadas. A camada de entrada é formada por 20 unidades responsáveis pela distribuição da informação. A camada intermediária é composta por 20 unidades que realizam o processamento não linear da informação, e a camada de saída é composta por 6 unidades de processamento que realizam o somatório dos resultados a partir do processamento da camada intermediária.

O teste da modelo RBF é realizado da seguinte maneira. São fornecidas, como entrada, à rede, as saídas das 5 camadas do plano seqüência temporal; esta entrada é distribuída pela primeira camada, processada na camada intermediária e fornece como saída o conjunto de ângulos correspondentes ao último ponto de cada padrão de entrada. Resumindo, o objetivo do modelo é gerar os 6 ângulos que corresponderiam ao último ponto de cada padrão apresentado à rede.

Para analisar a saída da rede, transformam-se os ângulos gerados em coordenadas espaciais e comparam-se os pontos gerados no segundo plano com os pontos da trajetória desejada.

6.3.1 Trajetórias Únicas

Para os testes com trajetórias únicas, o modelo RBF foi treinado seguindo a mesma metodologia do plano seqüência temporal. Cada trajetória única foi treinada e testada separadamente. As trajetórias obtidas são representadas por linhas pontilhadas e círculos, e as trajetórias desejadas são representadas por linhas contínuas e asteriscos. As trajetórias espaciais usadas para testar o segundo plano são aquelas obtidas no primeiro plano e plotadas na Figura 31a, Figura 32a e 32b, Figura 34c e Figura 35c. Os testes são realizados da seguinte maneira: cada trajetória única é testada separadamente; as trajetórias interpoladas e extrapoladas, obtidas no primeiro plano, são padrões de entrada para o segundo plano; o segundo plano fornece como saída o conjunto de ângulos correspondentes ao último ponto de cada padrão. Estes conjuntos de ângulos são convertidos em posições espaciais e comparados com as trajetórias desejadas.

Na Figura 56, todas as trajetórias possuem pontos interpolados e extrapolados gerados pelo plano seqüência temporal. A trajetória única representada em (a) é

composta pelos dois últimos pontos da trajetória desejada e pelos dois últimos pontos da trajetória interpolada, obtida pelo segundo plano. Esta trajetória difere das demais por não apresentar pontos extrapolados no primeiro plano.

Figura 56. Recuperação das trajetórias únicas no segundo plano. (a) pontos interpolados na trajetórias; (b), (c), (d) e (e) pontos interpolados e extrapolados na trajetória.

A Figura 56a mostrou que o modelo gerou dois pontos entre os dois últimos pontos desejados da trajetória, aproximando-os mais do penúltimo ponto. As trajetórias obtidas em (b), (c) e (d) conseguiram produzir trajetórias próximas às trajetórias desejadas. Contudo, as trajetórias obtidas não são exatamente aquelas treinadas. No item (e) o modelo produziu pontos em uma região atratora, formando uma trajetória composta por pontos entre o segundo e o quarto ponto da trajetória desejada.

As trajetórias obtidas no primeiro plano acompanharam melhor as trajetórias desejadas que as trajetórias geradas no segundo plano. É importante ressaltar que

alguns pontos gerados no primeiro plano são pontos infactíveis. Contudo, o segundo plano foi capaz de transformar os pontos infactíveis gerados pelo primeiro plano em pontos factíveis que não correspondem exatamente à posição espacial desejada.

Para melhorar a capacidade do modelo em gerar ângulos que tornem as trajetórias obtidas mais próximas das trajetórias desejadas, sugere-se, para uma próxima etapa, realizar o treinamento da matriz de pesos do modelo RBF.

6.3.2. Trajetórias Múltiplas

Para os testes com trajetórias múltiplas, o modelo RBF foi treinado seguindo a mesma metodologia do plano sequência temporal, isto é, cada tipo de trajetória múltipla foi treinada e testada separadamente. As trajetórias obtidas são representadas por linhas pontilhadas e círculos, enquanto que as trajetórias desejadas são representadas por linhas contínuas e asteriscos. As trajetórias usadas para testar o segundo plano são as trajetórias obtidas no primeiro plano, plotadas na Figura 36a e 36b, Figura 39a, Figura 40a e 40b, Figura 43a, Figura 44a e 44b, Figura 47a, Figura 52a e 52b e Figura 55a e 55b. Logo, para cada trajetória múltipla testada no segundo plano, foram usadas como entradas as trajetórias geradas no primeiro plano para pontos iniciais e finais perfeitos e/ou ruidosos. O segundo plano fornece como saída o conjunto de ângulos correspondente ao último ponto de cada padrão. Os conjuntos de ângulos gerados são convertidos em posições espaciais e são comparados com as trajetórias desejadas.

Nas Figuras 57, 58, 59 e 60 as letras (a) e (b) correspondem às trajetórias recuperadas pelo RBF, quando fornecidas como entradas as trajetórias obtidas para ponto inicial e final treinados no primeiro plano. As letras (c) e (d) correspondem às trajetórias recuperadas pelo RBF quando fornecidas como entradas as trajetórias obtidas para ponto inicial e final ruidosos no primeiro plano. Note que as Figuras 57, 58 e 59 (c) apresentam somente uma trajetória recuperada, pois o primeiro plano gerou como saída uma trajetória interpolada entre as duas treinadas em cada caso.

(a)

(b)

(c)

Figura 57. Recuperação das trajetórias do tipo n-1; (a) trajetória 1, (b) trajetória 2 e (c) trajetória recuperada.

Na Figura 57, o modelo seguiu as trajetórias treinadas. A trajetória obtida em (b) apresenta uma melhor aproximação da trajetória desejada em relação à trajetória obtida em (a). Na trajetória obtida em (c), o modelo gerou uma trajetória interpolada próxima da trajetória em (a). Em todos os casos o ponto final obtido está próximo do desejado, apesar da pequena discrepância.

Figura 58. Recuperação das trajetórias do tipo 1-m; (a) trajetória 1, (b) trajetória 2 e (c) trajetória recuperada.

Na Figura 58, o modelo acompanhou as trajetórias treinadas. Ambas as trajetórias obtidas em (a) e (b) apresentaram uma ótima aproximação da trajetória desejada. Na trajetória obtida em (c), o modelo produziu uma trajetória interpolada muito próxima da trajetória em (a). Nos dois primeiros casos, o ponto final é muito próximo daquele desejado. Para a letra (c), o ponto final é próximo daquele gerado em (a).

Figura 59. Recuperação das trajetórias do tipo 1-1 sem cruzamento; (a) trajetória 1, (b) trajetória 2 e (c) trajetória recuperada.

Na Figura 59, o modelo seguiu as trajetórias treinadas. A trajetória obtida em (a) apresenta, em média, uma melhor aproximação da trajetória treinada em relação à trajetória obtida em (b). Para a trajetória obtida em (c), o modelo foi capaz de interpolar uma trajetória entre as duas treinadas. Estes resultados foram os mais imprecisos até o momento.

Figura 60. Recuperação das trajetórias do tipo 1-1 com cruzamento; (a) e (c) trajetória 1, (b) e (d) trajetória 2.

A Figura 60 mostra que o modelo seguiu as trajetórias treinadas. As trajetórias obtidas em (a), (b), (c) e (d) aproximaram-se muito das trajetórias desejadas; as trajetórias (b) e (d) apresentaram uma melhor aproximação em relação as trajetórias mostradas em (a) e (c).

As trajetórias obtidas no segundo plano, que possuem diferentes pontos iniciais para um mesmo ponto final, um mesmo ponto inicial para diferentes pontos finais, e um ponto inicial para um ponto final sem cruzamento e com cruzamento, apresentaram resultados semelhantes ou melhores que os resultados obtidos pelo primeiro plano.

7. Conclusões

Este projeto visa a desenvolver um novo modelo baseado em princípios de memória associativa em redes neurais o qual seja capaz de resolver o problema de planejamento e controle de trajetória de um braço mecânico.

Para propor o modelo, foram primeiramente estudadas as características, limitações e comportamento de modelos de memórias associativas. O modelo resultante é composto por dois planos: plano seqüência temporal e plano ângulo. Para o plano seqüência temporal, o novo modelo introduzido é responsável por produzir seqüências temporais, deve armazenar padrões que não sejam ortogonais ou linearmente independentes, armazenar n-tuplas de informações que podem ser de naturezas diferentes, ser robusto a informações ruidosas e/ou incompletas, aprender seqüências temporais e ser treinável para melhorar sua capacidade de armazenamento. O modelo estudado que mais se aproxima desta descrição é MAM que, no entanto, precisa ser modificado para atender a todos os requisitos listados.

As modificações propostas são: representação contínua e realimentação auto-associativa para todas as unidades de processamento. A primeira modificação foi proposta para resolver o problema de representação da trajetória, cujas informações de posições espaciais e angulares trabalham com valores reais; logo, será usada representação contínua no intervalo $[-1, 1]$. A segunda modificação consiste na introdução de realimentação auto-associativa para fornecer, ao modelo, informações de cada unidade para ela mesma, deslocadas no tempo.

O plano ângulo é responsável pela cinemática inversa. As cinco últimas posições espaciais geradas pelo primeiro plano e seus respectivos tempos constituem os padrões de entrada do segundo plano. O plano é responsável por gerar os valores de ângulos associados à última posição espacial produzida pelo plano sequência temporal. A composição dos dois planos forma o sistema completo, que é responsável pelo planejamento e controle de trajetória.

Em resumo, o sistema recebe informações do ponto origem e do ponto alvo, estabelece uma trajetória para atingir o ponto alvo a partir do ponto origem e transforma os pontos espaciais que compõem a trajetória em valores de ângulos das juntas.

Foram realizadas a implementação e testes do modelo TMAM relativos ao plano sequência temporal. Os resultados obtidos mostram que o modelo TMAM é capaz de recuperar aproximações das seqüências temporais treinadas, interpolar e extrapolar pontos em uma seqüência e gerar trajetórias. O modelo é capaz ainda de acomodar seqüências de vários tamanhos e de lidar com duas trajetórias ao mesmo tempo. O modelo apresenta também rápido treinamento, capacidade de associar e de recuperar n-tuplas a partir de informações parciais ou ruidosas.

Nos testes realizados com trajetórias múltiplas, observou-se que, no caso das trajetórias tipo n-1, o modelo conseguiu recuperar o ponto final desejado ou se situa em sua vizinhança, conseguiu interpolar trajetória com recuperação do ponto final e apresentou aproximações das trajetórias treinadas. Nas trajetórias do tipo 1-m, o modelo reproduziu pontos iniciais treinados e pontos finais próximos aos desejados; tendeu a ser atraído por uma das trajetórias treinadas na interpolação de trajetórias e produziu trajetórias semelhantes as treinadas. Nos testes com trajetórias tipo 1-1 sem cruzamento, o modelo recuperou trajetórias próximas às trajetórias desejadas, recuperou pontos extremos de trajetórias próximas àqueles treinados e interpolou trajetórias entre aquelas treinadas. Na trajetória do tipo 1-1 com cruzamento, o modelo não apresentou boa recuperação quando as trajetória desejadas são perpendiculares entre si. Já na segunda bateria de testes, cujas trajetórias são mais parecidas, o modelo atingiu e até extrapolou pontos após o ponto final desejado,

tendeu acompanhar melhor uma das trajetórias desejadas e apresentou pequenos erros.

Nos testes realizados no modelo RBF, observou-se que, em geral, a rede é capaz de recuperar um ponto final dentro da vizinhança do desejado. Na generalização, observou-se que, quando a rede é treinada com poucos padrões, o modelo não apresenta uma boa generalização; mas, quando se aumenta o número de padrões para treinamento, consegue-se uma melhora na generalização, pois o aumento do número de padrões para treinamento divide melhor o espaço de trabalho. O modelo apresentou duas características importantes: (i) o modelo é capaz de receber um padrão que apresenta um ponto final inatingível, isto é, um ponto em que o manipulador não consegue alcançar e gerar um conjunto de ângulos que representa um ponto final atingível e (ii) uma vez treinado com trajetórias que possuam seus pontos dentro da região do espaço de trabalho do manipulador, o RBF só gera conjuntos de ângulos dentro deste espaço.

Nos testes realizados no modelo TMAM, descritos na Seção 6.2, podem-se observar algumas limitações tais como: baixa capacidade de armazenamento e sensibilidade em relação a mudanças bruscas de trajetórias, apresentadas nos testes da Subseção 6.2.1, nos quais baixa recuperação de trajetórias mais complexas pode ser melhorada com a interpolação e extrapolação das trajetórias treinadas. Na Subseção 6.2.2 observa-se má recuperação das seqüências geradas no teste 1-1 com cruzamento por serem estas quase perpendiculares entre si. Observou-se também que nas trajetórias múltiplas, freqüentemente aparecem pequenas diferenças entre as trajetórias treinadas e aquelas obtidas pelo modelo.

Nos testes realizados no modelo RBF, descritos no Capítulo 6 Seção 6.3., observou-se que o modelo foi capaz de lidar com trajetórias únicas e múltiplas. Nos testes com trajetórias únicas, o modelo mostrou-se capaz de reproduzir as trajetórias formadas por pontos interpolados e extrapolados próximas às trajetórias desejadas, apresentando pequenas discrepâncias. Uma limitação encontrada foi a formação de uma região atratora em uma das trajetórias treinadas.

Nos testes com trajetórias múltiplas, o modelo apresentou uma boa recuperação dos pontos das trajetórias de todos os tipos; as trajetórias obtidas são semelhantes ou algumas vezes melhores que aquelas recuperadas no primeiro plano.

Para os próximos passos referentes ao estudo iniciado, propõe-se o estudo de outros algoritmos de aprendizagem, para aumentar a capacidade de armazenamento do modelo TMAM e tornar o número de camadas variáveis, podendo assim comportar mais pontos de uma trajetória, sem precisar decompô-la. Em relação ao plano ângulo, será implementado o estudo de outras funções radiais e algoritmos para escolha dos centros para melhorar a capacidade de recuperação do modelo RBF. É proposto também implementar o cálculo da dinâmica direta para o cálculo dos torques.

As sugestões acima podem vir a permitir que o modelo realize rastreamento o que ele não consegue nos moldes em que foi treinado. Como conclusão final, é importante salientar que este modelo, na grande maioria dos casos testados, mostrou capacidade de realizar planejamento e controle ponto a ponto de trajetória. Além disto, ele pode se adaptar através de um treinamento rápido a novas situações do espaço de trabalho do robô.

8. Referências Bibliográficas

- Amari, S. (1972). Learning patterns and pattern sequences by self-organizing nets of threshold elements. *IEEE Transactions on Computers*, vol. C-21, No.11, pp. 1197-1206.
- Anderson, A . J. (1972). A simple neural network generating na interactive Memory. *Mathematical Biosciences* , vol. 14, pp. 197-220.
- Araújo, A. F. R (1994). Memory, Emotions, and Neural Networks: Associative Learning and Memory Recall Influenced by Affective Evaluation and Task Difficulty. *Ph.D. Thesis*. School of Cognitive and Computing Sciences, University of Sussex, Brighton, UK.
- Araújo, A. F. R. and Haga, G. M. (1997). Two Simple Strategies to Improve Bidirectional Associative Memory's Performance: Unlearning and Delta Rule. *International Conference on Neural Network (IEEE-ICNN)*, Houston, USA, vol 2, pp. 1178-1182.
- Araújo, A. F. R. and Vieira, M. (1997). Temporal Multidirectional Associative Memory: Adaptable, Continuous, and Self-connected MAM. *International Conference on Neural Network (IEEE-ICNN)*, Houston, USA, vol2, pp 1194-1200.

- Araújo, A. F. R. and Vieira, M. (1997a). Employ a Multiple Associative Model for Temporal Sequence Reproduction. *Journal of the Brazilian Computer Society(JBCS)*, Brazil.
- Atkeson, C. G. and Reinkensmeyer, D. J. (1991). Using Associative Content-Addressable Memories to Control Robots. In W. T. Miller III, R. S. Sutton and P. J. Werbos (eds). *Neural Networks for Control*, pp. 255-285. Cambridge, MA: MIT Press.
- Baddeley, A. (1990). *Human Memory Theory and Practice*. Hove, UK: Lawrence Erlbaum.
- Barto, A. G. (1991). Connectionist Learning for Control: An Overview. In W. T. Miller III, R. S. Sutton and P. J. Werbos (eds). *Neural Networks for Control*, pp. 5-58. Cambridge, MA: MIT Press.
- Cheng, F., and Hour, T., and Sun, Y., and Chen, T. (1997). Study and Resolution of Singularities for a 6-DOF PUMA Manipulator, *IEEE Transactions on Systems Man and Cybernetics*, vol. 27. pp. 332-339.
- Corke, P. I. (1996). "Robotics Toolbox for use with MATLAB (Release 4)". *IEEE Robotics and Automation Magazine*, vol 3, nº 1, 24-32.
- Fu, K. S., Gonzalez, R. C., and Lee, C.S.G. (1987). *Robotics: Control, Sensing, Vision and Intelligence*. New York: McGraw-Hill Book Company.
- Gordon, L. K.III, and David P. Campagna. (1991). A Summary Comparison of CMAC Neural Network and Traditional Adaptive Control System. In W. T. Miller III, R. S. Sutton and P. J. Werbos (eds). *Neural Networks for Control*, pp. 143-170. Cambridge, MA: MIT Press.

- Hagiwara, M. (1990). Multidirectional associative memory. *International Joint Conference on Neural Networks*, vol. 1, pp. 3-6.
- Hassoun, M. H. (1995). *"Fundamentals of Artificial Neural Networks"*, The MIT Press, Cambridg.
- Hattori, M. and Hagiwara, M. (1995). Quick learning for multidirectional associative memories. *IEEE International Conference on Neural Networks*, pp.1949-1954.
- Hillis, D. (1985). *The Connection Machine*. Cambridge, MA: MIT Press.
- Hirayama, M., and Kawato, M. and Jordan, M. I. (1993). The Cascade neural network model and a speed-accuracy trade-off of arm movement. *Journal of Motor Behavior*, vol. 25, pp. 162-174.
- Hopfield, J.J. (1982). Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the National Academy of Sciences*, vol. 79, pp. 2554-2558.
- Hopfield, J. J., Feinstein, D. I. and Palmer R. G. (1983). Unlearning has a Stabilizing Effect in Collective Memories. *Nature*, vol. 304.
- Josin, G. (1988). Neural-Space Generalization of a Topological Transformation. *Biological Cybernetics*, vol. 59, pp. 283-290.
- Kim, S., and Jang, C., and Chai, C., and Choi, H. (1997). Trajectory control of Robotic Manipulators using chaotic Neural Networks. *International Conference on Neural Network (IEEE-ICNN)*, Houston, USA, vol. 3, pp. 1685 -1688.

- Kosko, B. (1988). Bidirectional associative memories. *IEEE Transactions on Systems Man and Cybernetics*, vol. 18, pp. 49-60.
- Kohonen, T. (1972). Correlation Matrix memories. *IEEE Transaction on Computers*, vol. C-21, pp. 353-359.
- Kohonen, T. and Ruohonen, M. (1973). Representation of Associative Data by Matrix Operators. *IEEE Transactions on Computers*, pp. 701-702.
- Kohonen, T. (1989). Self-Organization and Associative Memory. Springer-Verlog, 3rd. edition, Berlin.
- Kawato, M. (1991). Computational Schemes and Neural Network Models for Formation and Control of Multijoint Arm Trajectory. In W. T. Miller III, R. S. Sutton and P. J. Werbos (eds). *Neural Networks for Control*, pp. 197-228. Cambridge, MA: MIT Press.
- Kawato, M. (1989) Adaptation and Learning in Control of Voluntary Movement by the Central Nervous System. *Advanced Robotics*, vol. 3(3).
- Kohonen, T. (1980). *Content-Addressable Memories*. New York: Springer-Verlag.
- Leonard, A. J. and Kramer, A. M. (1991). Radial Basis Function Networks for Classifying Process Faults. *IEEE Control Systems*, vol 11, no. 3, pp 31-38.
- Lin, L. J. (1993). Reinforcement Learning for Robots Using Neural Networks. *PhD Thesis*, Carnegie Mellon University
- Marr, D. (1982). *Vision*. New York: W. H. Freeman.

- Martinez, T. M., Ritter, H. J., and Schulten, K. J. (1990). Three-Dimensional Neural Net for Learning Visuomotor Coordination of a Robot Arm. *IEEE Transactions on Neural Networks*, vol. 1, pp.131-136.
- McEliece, R. J., Posner, E. C., Rodemich, E. R. and Venkatesh, S. S. (1987). The Capacity of Hopfield Associative Memory. *IEEE Transactions on Information Theory*, IT-33, pp. 461-482.
- Nakano, K. (1972). Associatron - a model of associative memory. *IEEE Transactions on Systems, Man and Cybernetics*, vol. 2, no. 3, pp. 380 - 388.
- Oh, H. and Kothari, S. C. (1994). Adaptation of the relaxation method for learning in bidirectional associative memory. *IEEE Transactions on Neural Networks*, vol. 5, no. 4, pp.576-583.
- Psaltis, D., Sideris, A., & Yamamura, A. A. (1988). A Multilayer Neural Network Controller. *IEEE Control Systems Magazine*, vol. 8, pp. 17-21.
- Ritter, H. J., Martinez, T. M., and Schulten, K. J. (1992). *Neural Computation and Self-Organizing Maps: An Introduction*. Reading: Addison-Wesley.
- Rosembblatt, F. (1962). *Principles of Neurodynamics: Perceptron and theory of Brain Mechanisms*. Washington, DC: Spartan Books.
- Rumelhart, D. E., and Hinton, G. E., and Williams, R. J. (1986). Learning internal representations by error propagation. In D. E. Rumelhart and J. L. McClelland (Eds.), *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*, vol. 1. Cambridge, MA: Bradford Books.
- Rumelhart, D. E. and McClelland, J. L. (1988). *Parallel Distributed Processing*, 1. Cambridge, MA: MIT Press.

- Schaal, S. & Atkeson, C. G. (1994). Robot Juggling: Implementation of Memory-Based Learning. *IEEE Control Systems Magazine*, vol. 14(1), pp. 57-71.
- Van der Smagt, P., Groen, F. e Kröse, B. (1993). Robot Hand-Eye Coordination Using Neural Networks. *Technical Report (CS-93-10) of the Department of Computer Systems*, University of Amsterdam.
- Vieira, M. and Araújo, A. F. R. (1997). Memória Associativa Temporal para Planejar Trajetória de um Manipulador. *Simpósio Brasileiro de Automação Inteligente (3º.SBAI)*, Vitória, Espírito Santo, Brasil, pp. 459-464.
- Wada, Y. and Kawato, M. (1993). Neural Network Model for Arm Trajectory Formation Using Forward and Inverse Dynamics Models. *Neural Networks*, vol. 6 (7), pp. 919-932.
- Werbos, P. J. (1991a). Overview of Designs and Capabilities. In W. T. Miller III, R. S. Sutton and P. J. Werbos (eds). *Neural Networks for Control*, pp. 59-65. Cambridge, MA: MIT Press.
- Werbos, P. J. (1991b). A Menu of Designs for Reinforcement Learning Over Time. In W. T. Miller III, R. S. Sutton and P. J. Werbos (eds). *Neural Networks for Control*, pp. 67-95. Cambridge, MA: MIT Press.
- Widrow, B. and Hoff, M. E. (1960). Adaptive switching circuits. *IRE WESCON Convention Record*, pp. 96-104.
- Widrow, B. (1991). The Truck Backer-Upper: An example of Self-Learning in Neural Networks. In W. T. Miller III, R. S. Sutton and P. J. Werbos (eds). *Neural Networks for Control*, pp. 67-95. Cambridge, MA: MIT Press.

Williams, R. J. (1991). Adaptive State Representation and Estimation Using Recurrent Connectionist Networks. In W. T. Miller III, R. S. Sutton and P. J. Werbos (eds). *Neural Networks for Control*, pp. 97-114. Cambridge, MA: MIT Press.

Winston, P. H. (1992). *Artificial Intelligence*. Reading, UK: Addison-Wesley, 3rd. edition.