Anais do IX Congresso Brasileiro de Redes Neurais / Inteligência Computacional (IX CBRN)
Ouro Preto 25-28 de Outubro de 2009 

# A DIMENSÃO TEMPORAL NO APRENDIZADO DE TAREFAS DE NAVEGAÇÃO DE ROBÔS MÓVEIS USANDO REDES NEURAIS ARTIFICIAIS

Ananda L. Freire*, Guilherme A. Barreto*, Marcus V. D. Veloso*, Antonio T. Varela†

*Universidade Federal do Ceará, Depto. Engenharia de Teleinformática, Fortaleza, Ceará

†Instituto Federal de Educação, Ciência e Tecnologia do Ceará, Fortaleza, Ceará

Emails: anandalf@gmail.com, guilherme@deti.ufc.br, marcus@laboratoriocentauro.net, themoteo@cefetce.br

**Abstract**— This paper reports results of an investigation on the degree of influence that the inclusion of short-term memory mechanisms has on the performance of neural classifiers when applied to robot navigation tasks. In particular, we deal with the well-known strategy of navigating by "wall-following". For this purpose, four well-known neural architectures (Logistic Perceptron, Multilayer Perceptron, Mixture of Experts and Elman network) are used to associate different sensory input patterns with four predetermined action categories. All stages of the experiments - data acquisition, selection and training of the architectures in a simulator and their execution on a real mobile robot - are described. The obtained results suggest that the wall-following task, formulated as a pattern classification problem, is nonlinearly separable, a result that favors the MLP network if no memory of input patters are taken into account. If short-term memory mechanisms are used, then even a linear network is able to perform the same task successfully.

**Keywords**— Robot navigation, Wall-following, Neural classifiers, Short-term memory.

**Resumo**— Este artigo reporta resultados de uma investigação acerca do grau de influência que a inclusão de mecanismos de memória de curta duração (MCD) exercem sobre o desempenho de classificadores neurais quando aplicados em tarefas de navegação de robôs. Em particular, este artigo trata da navegação do tipo *Wall Following*. Para este fim, quatro conhecidas arquiteturas neurais (Perceptron Logístico, Perceptron Multicamadas, Mistura de Especialistas e rede de Elman) são usadas com o intuito de associar diferentes padrões de leituras sensoriais com quatro classes de ações pré-determinadas. Todas as etapas dos experimentos - aquisição dos dados, seleção e treinamento das arquiteturas em um simulador e execução das arquiteturas em um robô móvel real (SCITOS G5), são descritas em detalhes. Os resultados obtidos sugerem que a tarefa de seguir paredes, formulada como um problema de classificação de padrões, é não-linearmente separável, resultado este que favorece a rede MLP quando os classificadores são treinados sem MCD. Contudo, se mecanismos de MCD são usados, então até mesmo uma rede linear é capaz de executar a tarefa de interesse com sucesso.

**Palavras-chave**— Navegação de robôs, Seguir paredes, Classificadores neurais, Memória de curta duração.

## 1 Introdução

Este artigo apresenta os resultados dos testes realizados com o robô móvel SCITOS G5, que a partir de dados sensoriais coletados após navegar em uma sala seguindo paredes, é treinado com quatro diferentes arquiteturas de redes neurais para tomar quatro decisões que determinam a sua movimentação. Pode-se entender o processo de aprendizado das arquiteturas neurais como um tipo de aprendizado por imitação, uma vez que os dados são coletados por uma rotina construída à base de regras heurísticas IF-THEN, cuja função é guiar o robô ao longo das paredes de uma sala. A qualidade do aprendizado é então avaliado pela habilidade das arquiteturas neurais em "imitar" o desempenho do controlador heurístico.

As principais contribuições deste artigo estão na verificação empírica que a clássica tarefa de navegação de robôs estilo *wall-following* (Bekey 2005), formulada como um problema de classificação de padrões, é não-linearmente separável, e na verificação da hipótese que memória de curta duração (MCD) melhora o desempenho dos classificadores na tarefa supracitada. Esta memória é introduzida de duas formas diferentes neste trabalho: para redes neurais estáticas (e.g. Perceptron logístico e MLP), a MCD é introduzida nas unidades de entrada através do uso de observações sensoriais passadas, e no caso de redes neurais dinâmicas (e.g. rede de Elman), a MCD é intrínseca à própria arquitetura.

## 2 Classificadores Neurais Avaliados

As redes neurais avaliadas neste artigo são a rede perceptron logístico (PL), a rede perceptron multicamadas (MLP), a rede recorrente de Elman e a arquitetura de mistura de especialistas (ME). Por conta da popularidade das três primeiras redes, apenas a arquitetura ME é detalhada a seguir.

**Mistura de Especialistas** - Em poucas palavras, o objetivo da abordagem via mistura de especialistas é modelar a distribuição de probabilidade dos padrões do conjunto de treinamento, $\{\mathbf{x}, \mathbf{d}\}$ utilizando uma arquitetura modular com $K$ redes especialistas e uma rede seletora (Jacobs et al. 1991, Alpaydin & Jordan 1996). A rede seletora determina a saída dentre as fornecidas pelos $K$ especialistas,

de acordo com suas probabilidades $g_i$, $i = 1, ..., K$, tal que $0 \leq g_i \leq 1$ e $\sum_{i=1}^{K} g_i = 1$. O procedimento para a adaptação da arquitetura de mistura de especialistas, construídas por módulos de perceptron logísticos, com uma rede seletora também de camada única e função de ativação *softmax*, é descrito a seguir.

1. *Inicialização*: atribuir valores pequenos e aleatórios para os pesos dos especialistas $\mathbf{w}_i^{(m)}(n)$ e do seletor $\mathbf{a}_i(n)$, sendo as saídas $i = 1, 2, ..., K$ e os neurônios $m = 1, 2, ..., c$;
2. *Adaptando os Especialistas e o Seletor*: apresentar o padrão de entrada $\mathbf{x}(t)$ e a resposta desejada $\mathbf{d}(t)$ para o instante atual $t$:

$$u_i(t) = \mathbf{x}^T(t)\mathbf{a}_i(t), \quad (1)$$

$$g_i(t) = \frac{\exp\{u_i(t)\}}{\sum_{j=1}^{K} \exp\{u_j(t)\}}, \quad (2)$$

$$y_i^{(m)}(t) = \mathbf{x}^T(t)\mathbf{w}_i^{(m)}(t), \quad (3)$$

$$\mathbf{y}_i(t) = [y_i^{(1)}(t)\ y_i^{(2)}(t)\ \cdots\ y_i^{(c)}(t)]^T, \quad (4)$$

$$h_i(t) = \frac{g_i(t) \exp\left\{-\frac{1}{2}\|\mathbf{d}(t) - \mathbf{y}_i(t)\|^2\right\}}{\sum_{j=1}^{K} g_j(t) \exp\left\{-\frac{1}{2}\|\mathbf{d}(t) - \mathbf{y}_i(t)\|^2\right\}}, \quad (5)$$

$$e_i^{(m)}(t) = d^m(t) - y_i^{(m)}(t), \quad (6)$$

$$\delta_i^{(m)}(t) = e_i^{(m)}(t)\left[y_i^{(m)}(t)(1 - y_i^{(m)}(t)\right], \quad (7)$$

$$\mathbf{w}_i^{(m)}(t+1) = \mathbf{w}_i^{(m)}(t) + \eta h_i(t)\delta_i^{(m)}(t)\mathbf{x}(t), \quad (8)$$

$$\mathbf{a}_i(t+1) = \mathbf{a}_i(t) + \eta\left[h_i(t) - g_i(t)\right]\mathbf{x}(t). \quad (9)$$

3. Repetir passo 2 para todos os padrões de treinamento, i.e. $t = 1, 2, ..., N$.
4. Repetir passos 2 e 3 até o processo convergir.

## 3 Robô Móvel SCITOS G5

Figura 1: Robô SCITOS G5.

O robô móvel SCITOS G5, mostrado na Figura 1, foi adquirido da empresa MetraLabs Robotics (http://metralabs.com), situada em Ilmenau, Alemanha. Este robô foi construído para o uso em ambientes fechados, movimentando-se por meio de um sistema motor diferencial, carregando sua plataforma de $60kg$ a uma velocidade de até $1,4m/s$, capaz de rotacionar 360 graus, e empurrar cargas de até $50kg$.

O robô tem $570mm \times 735mm \times 610mm$ de altura, comprimento e largura, respectivamente. Com uma autonomia de $12h$ com um PC embarcado. Este é constituído de uma placa-mãe Mini-ITX, processador Intel Core Duo 1,6 ou 2.0GHz, memória RAM de 2GB, HD de 120GB, com entrada PS/2 para teclado e mouse, 5 entradas USB, 3 entradas Firewire, TV-out, RS232, VGA, saída SPDIF, LVDS, e entrada RJ-45 para Ethernet 10/100 e rede wireless *on-board*, padrão IEEE 802.11a/b/g. Como sistema operacional, utiliza Linux (Fedora 9). O sistema operacional vem provido de uma coleção de bibliotecas C++ Robot-API, desenvolvidas pela MetraLabs para permitir a criação de softwares para controle e navegação da plataforma SCITOS G5. Possui ainda *encoders* para cálculo de posição com 460 "ticks" por rotação da roda, sensor de colisão, um cinturão de 24 sensores de ultra-som com range de $20cm - 300cm$, sensor a laser SICK S300, cabeça robótica dotada de um câmera omnidirecional.

## 4 Coleta de Dados

A coleta de dados para treinamento das redes neurais obedeceu o mesmo procedimento que Antonelo et al. (2008). Assim, foi criada uma rotina em C++ para guiar e rotular o comportamento de seguir paredes, dadas as leituras sensoriais naquele instante. A lógica desse é apresentada no Algoritmo 4.1 e, por ser a responsável por gerar as decisões que o robô precisa tomar sobre a sua movimentação, é colocada dentro de uma *thread* que é chamada a cada vez que os sensores de ultra-som captam alguma mudança significativa. Caso não haja nenhuma atividade sensorial em $500ms$, o robô pára.

Percebe-se que a lógica do algoritmo de navegação faz uso de regras heurísticas IF-THEN atuando sobre as medidas "distância à frente" e ' 'distância à esquerda", além também calcular as distâncias à direita e atrás. Cada uma dessas distâncias não é calculada por um único sensor, mas sim por um conjunto deles que formam juntos um feixe de 60°, e a distância utilizada é a menor encontrada pelos sensores que fazem parte do conjunto. Essas quatro distâncias são referidas ao longo do artigo como "distâncias resumidas".

Além da lógica de movimentação, a *thread* também é responsável por enviar através da rede wireless as seguintes informações ao banco de dados localizado

no computador de apoio: as 24 leituras dos sensores ultra-som, as posições $x$, $y$ e angular retornadas pelo *encoder* do robô, a velocidade translacional e rotacional do robô naquele momento, o rótulo numérico da ação que foi realizada pelo robô de acordo com as configurações do ambiente, as quatro "distâncias resumidas" e os valores das velocidades de translação e rotação.

**Algorithm 4.1:** SEGUEPAREDE($frontDist, leftDist$)

**if** $leftDist > 0,9$
**then**
  **if** $frontDist <= 0,9$
    **then** Parar e girar à direita
    **else** Reduz a velocidade e gira à esquerda
**else**
  **if** $frontDist <= 0,9$
    **then** Parar e girar à direita
  **else if** $leftDist < 0,55$
    **then** Reduz a velocidade e gira à direita
  **else** Segue em Frente

Essas informações são coletadas à medida que o SCITOS G5 navega pela sala, realizando quatro voltas. A disposição dos objetos localizados na sala de teste pode ser visto na Figura 2 e a trajetória que o robô percorreu na Figura 3.

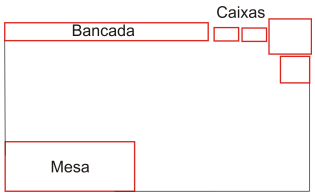


Figura 2: Diagrama do ambiente de navegação.

A coleta de dados foi realizada a uma taxa de 9 amostras por segundo e gerou um banco de dados com 5456 exemplos. Estes dados são então utilizadas para o treinamento dos classificadores neurais apresentados anteriormente, a fim de verificar qual deles melhor "imita" o Algoritmo 4.1.

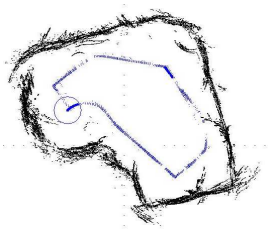

Figura 3: Percepção do ambiente e trajetória do SCITOS G5 realizada pelo Algoritmo 4.1.

## 5 Resultados dos Experimentos

**Classificação Estática**: Os primeiros experimentos são realizados sem o uso de memória, de tal forma que a tarefa de navegação pode ser formatada como um problema de classificação estática. Neste caso, a rede PL é formada por uma única camada de quatro neurônios de saída, cada um representando uma ação (classe) diferente: seguir em frente (classe 1), curva aberta à direita (classe 2), curva fechada à direita (classe 3) e curva aberta à esquerda (classe 4). A rede PL é treinada por 200 épocas a uma taxa de aprendizagem fixa de $\eta$=0,02. A arquitetura ME é constituída por uma rede seletora formada por uma rede perceptron simples de quatro neurônios, mais $K = 4$ especialistas que também consistem em redes PL com quatro neurônios. O treinamento da arquitetura ME é realizado por 35 épocas a uma taxa de aprendizagem fixa de $\eta$=0,02.

Com a rede MLP, foram feitos testes com diferentes quantidades de neurônios ocultos para verificar qual a quantidade mínima capaz de realizar com sucesso a separação das classes. Observou-se que $q = 6$ é essa quantidade mínima. A camada de saída da rede MLP contém 4 neurônios com função de ativação logística. O treinamento da rede MLP é realizado por 500 épocas com taxa de aprendizagem fixada em $\eta$=0,05.

Para o restante do artigo adota-se a seguinte notação: PL($I$,$O$) - rede perceptron logístico, em que $I$ é a dimensão da entrada e $O$ o número de neurônios de saída; ME($I$,$K$,$O$) - rede de mistura de especialistas, em que $K$ é o número de especialistas; e MLP($I$,$q$,$O$) - rede perceptron multicamadas com $q$ neurônios ocultos.

Para a tarefa de navegação via classificação estática são utilizadas apenas 2 unidades de entrada, correspondentes à distância resumida à frente e à esquerda do robô SCITOS. Cada experimento foi dividido em duas fases. A primeira consiste no treinamento e teste offline de uma rede no computador (Pentium 4, 2.4GHz com memória de 500Mb, Sistema Operacional Ubuntu 8.04), resultados mostrados nas Tabelas 1 e 2, resultando em um arquivo texto contendo os valores dos pesos sinápticos da rede. De posse dos valores dos pesos, embarca-se a rede gerada para que esta controle a movimentação do robô pela sala, comparando seu desempenho com o do algoritmo heurístico que gerou os dados de treinamento.

No experimento realizado com a rede PL(2,4) sem MCD, a trajetória realizada pelo robô pode ser vista na Figura 4. Observa-se que não houve nenhuma colisão, mas o robô não foi capaz de realizar a curva aberta à esquerda. Uma capacidade de

Tabela 1: Estatísticas do treinamento *offline* sem MCD.

| | **PL(2,4)** | **ME(2,4,4)** | **MLP(2,6,4)** |
|---|---|---|---|
| Média de acertos: | $3,8\%$ | $67,8\%$ | $96,82\%$ |
| Taxa Máxima de acertos: | $13,8\%$ | $67,85\%$ | $97,95\%$ |
| Taxa Mínima de acertos: | $0\%$ | $67,77\%$ | $82,07\%$ |
| Variância dos resultados: | $31311,3$ | $9,69 \times 10^{-8}$ | $9,32 \times 10^{-4}$ |

Tabela 2: Resultado dos testes *offline* sem MCD.

| | **PL(2,4)** | | | | **ME(2,4,4)** | | | | **MLP(2,6,4)** | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Erro Quadrático Médio: | $0,657797$ | | | | $0,74781$ | | | | $0.0294259$ | | | |
| Taxa de acerto: | $42,71\%$ | | | | $5,22\%$ | | | | $97,59\%$ | | | |
| Matriz de Confusão: | 0 | 0 | 594 | 0 | 0 | 602 | 0 | 0 | 571 | 7 | 6 | 0 |
| | 0 | 0 | 207 | 0 | 0 | 203 | 0 | 0 | 0 | 213 | 8 | 0 |
| | 0 | 0 | 534 | 14 | 0 | 558 | 6 | 0 | 1 | 0 | 559 | 0 |
| | 0 | 0 | 19 | 88 | 0 | 85 | 1 | 0 | 11 | 0 | 2 | 78 |

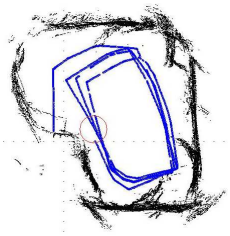

Figura 4: Percepção do ambiente e trajetória do SCITOS G5 com a rede PL(2,4) sem MCD.

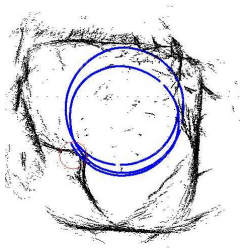

Figura 5: Percepção do ambiente e trajetória do SCITOS G5 com a rede ME(2,4,4) sem MCD.

classificação reduzida já era esperada visto a pouca quantidade de amostras corretamente classificadas na matriz de confusão correspondente (Tabela 2). Com a arquitetura ME(2,4,4) sem MCD, o robô foi capaz de realizar quase uma volta completa na sala antes de colidir (Figura 5). Mas ao posicionar o robô a partir da posição em que ele colidiu, onde ele deveria fazer uma curva à esquerda, fez uma aberta à direita e permaneceu nesse movimento. Analisando as ações armazenadas no banco de dados, percebe-se que o robô fez a maior parte da volta da sala apenas fazendo uso do movimento de curva aberta à direita ou, em raríssimas tentativas, uma curva fechada à direita. A rede MLP obteve o melhor desempenho, como mostrado na Figura 6. Os valores correspondentes das taxas de acerto são mostrados na Tabela 2.

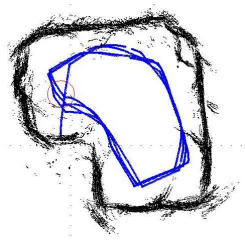

Figura 6: Percepção do ambiente e trajetória do SCITOS G5 com a rede MLP(2,6,4) sem MCD.

**Classificação Temporal**: Uma vez observado o comportamento do robô utilizando classificadores estáticos (i.e. sem MCD), são realizados experimentos adicionais a fim de verificar se (1) o uso de MCD melhora o desempenho das redes que não foram bem sucedidas, e (2) o uso de MCD permite que haja uma redução nas dimensões da arquitetura MLP, diminuindo seu custo computacional.

A primeira rede a ser testada é a PL(2,4), que passa a ser alimentada não apenas com as distâncias resumidas atuais, percebidas pelos sensores de ultrasom, mas também pelo histórico recente dessas distâncias. Com exceção da alteração na dimensionalidade do vetor de entrada, os parâmetros de treinamento continuam o mesmo do treinamento realizado sem MCD. No caso desta rede foram realizados testes com $1, 2, 3, 5, 9, 34$ e $49$ amostras (observações) passadas. Apenas os resultados para uma MCD composta de 9 amostras passadas das distâncias são apresentados neste artigo por ser a única configuração que permitiu que a rede PL fosse capaz de realizar todos os movimentos sem colisão. Assim, o vetor de entrada é composto pela amostra atual e as nove amostras passadas para cada uma das duas distâncias resumidas, implicando num vetor de entrada com dimensão 20. Analisando-se os resultados mostrados na Figura 7 e nas Tabelas 3 e 4) percebe-se claramente a melhoria no desempenho da rede PL(2,4) com a inclusão de MCD.

Tabela 3: Estatísticas do treinamento *offline* com MCD.

| | **PL(2,4)** | **Elman(2,4,4)** |
|---|---|---|
| Média de acertos: | $13,57\%$ | $85,16\%$ |
| Taxa Máx. de acertos: | $14,7\%$ | $96,42\%$ |
| Taxa Mín. de acertos: | $13,55\%$ | $75,52\%$ |
| Variância: | 0.43 | 0.00390116 |

Tabela 4: Resultado dos testes *offline* com MCD.

| | **PL(2,4)** | | | | **Elman(2,4,4)** | | | |
|---|---|---|---|---|---|---|---|---|
| Erro Quadrático Médio: | $0,232588$ | | | | $0,0555338$ | | | |
| Taxa de acerto: | $67,08\%$ | | | | $96,22\%$ | | | |
| Matriz de Confusão: | 287 | 135 | 146 | 19 | 578 | 9 | 7 | 0 |
| | 93 | 110 | 4 | 0 | 10 | 197 | 0 | 0 |
| | 10 | 0 | 522 | 13 | 6 | 3 | 539 | 0 |
| | 47 | 0 | 9 | 51 | 17 | 0 | 3 | 87 |

No caso da arquitetura ME, como o acréscimo de MCD não surtiu nenhuma diferença no desempenho da navegação do robô, os resultados não são mostrados neste artigo.

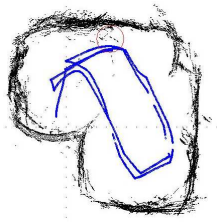

Figura 7: Percepção do ambiente e trajetória do SCITOS G5 com a rede PL(2,4) com MCD.

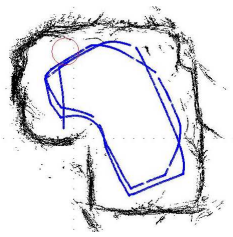

Figura 8: Percepção do ambiente e trajetória do SCITOS G5 com a rede Elman(2,4,4).

Uma forma alternativa de inclusão de MCD é através de recorrência. A fim de verificar se este tipo de MCD melhora o desempenho de arquiteturas neurais multicamadas, a rede foi utilizada na tarefa de navegação por classificação temporal. Como entrada da rede de Elman são usadas apenas as leituras atuais dos sensores de ultra-som, além das unidades de contexto. Os parâmetros de treinamento são os mesmos da rede MLP usada na navegação via classificação estática, exceto a taxa de aprendizagem ($\eta$=0,05). Após vários testes variando-se o número de neurônios da camada oculta verificou-se que a menor quantidade de neurônios ocultos necessários para que o robô realize a tarefa é $q = 4$, ou seja, com 2 neurônios a menos que no caso da rede MLP utilizada na abordagem via classificação estática. É importante ressaltar que a rede MLP(2,6,4) já havia sido capaz de resolver a tarefa sem MCD; contudo, a inclusão de MCD via recorrência como na rede de Elman permitiu que o classificador tivesse menos neurônios ocultos.

**Superfícies de Decisão**: Uma maneira de avaliar o desempenho dos classificadores em realizar a tarefa de navegação de interesse é através da visualização das superfícies de decisão implementadas por cada um. Nas Figuras 9(a) e 9(b) observa-se que a inclusão de MCD permitiu uma melhor separação das classes por parte dos classificadores, mesmo no caso da rede PL[1].

Em suma, percebe-se uma melhora na distinção das quatro classes pela rede PL com MCD em relação à rede PL sem MCD. Já as redes MLP (Figura 9(c)) e Elman (Figura 9(d)), que apresentaram os melhores desempenhos na tarefa de navegação, geraram superfícies de decisão semelhantes, embora a rede de Elman tenha chegado ao resultado mostrado com dois neurônios a menos em sua camada oculta.

## 6 Conclusões

Este trabalho comparou os desempenhos de quatro classificadores neurais usados como controladores da navegação ao estilo *wall-following* de um robô móvel real. Para isso, a tarefa de navegação foi formulado como um problema de classificação de padrões, em que os padrões consistiam nas leituras sensoriais e as classes em ações a serem tomadas

[1]Cada ponto nestas figuras corresponde a um par de medidas das distâncias resumidas em determinado instante de tempo ao longo da trajetória executada pelo robô. As cores permite diferenciar as leituras sensoriais correspondentes a cada uma das quatro classes de ações definidas para a tarefa.

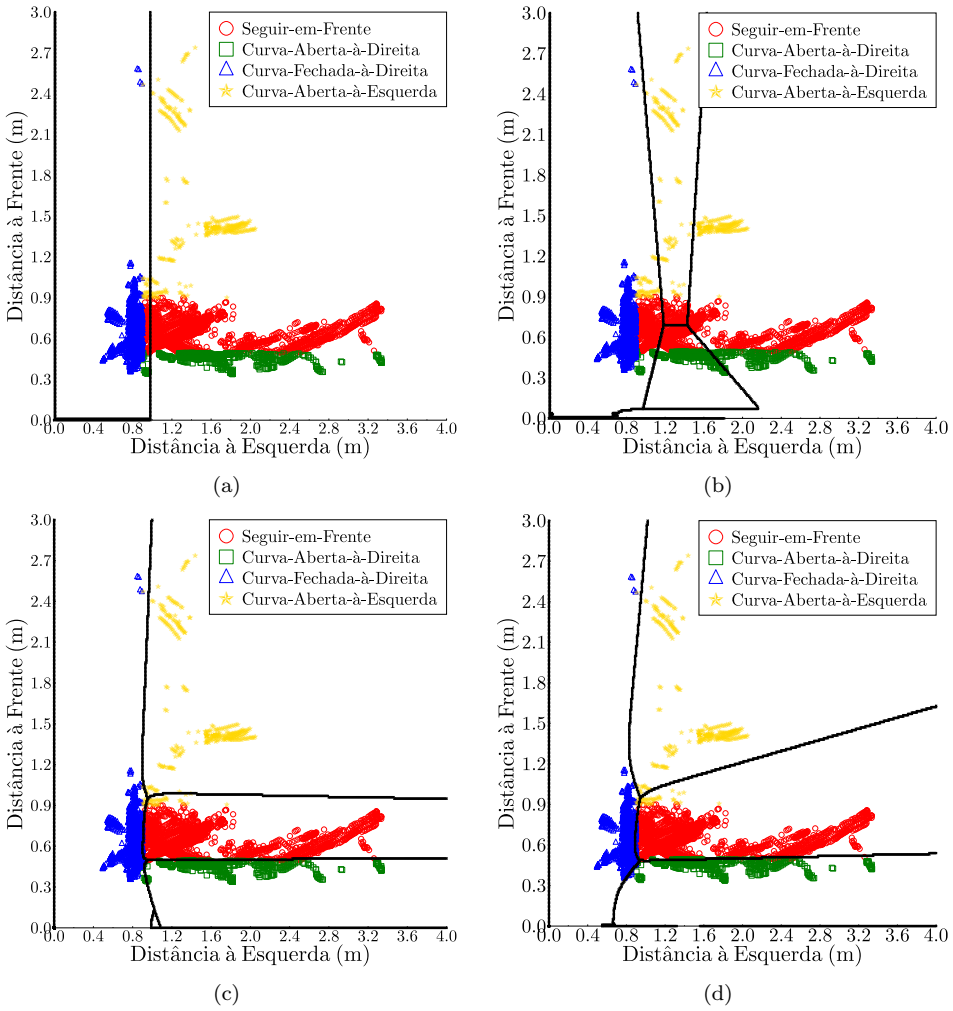


Figura 9: Superfícies de decisão da a) PL(2,4) sem MCD, b) PL com MCD, c) MLP(2,6,4) e d) Elman.

pelo robô. Foram avaliadas a rede perceptron logístico, a arquitetura mistura de especialistas, com especialistas formados por redes perceptron logístico, rede perceptron multicamadas, além da rede recorrente de Elman. De um modo geral, a rede MLP obteve o melhor desempenho dentre os quatro classificadores avaliados.

Os experimentos realizados permitiram verificar que uma tarefa de navegação supostamente simples, constitui-se na verdade em uma complexa tarefa de tomada de decisões, e que o acréscimo de MCD à entrada do classificador perceptron logístico permitiu que este passasse a ser capaz de realizar com sucesso uma tarefa de classificação não-linear. O uso de MCD via recorrência permite que uma arquitetura neural do tipo MLP reduz ainda mais o número de neurônios ocultos, sem comprometer a capacidade discriminatória do classificador.

Em trabalhos futuros pretende-se investigar o desempenho de classificadores SVM na navegação do tipo *wall-following*, bem como avaliar outros tipos de mecanismos de memória de curta duração, tais como o implementado pela arquitetura *Echo-state network* (Antonelo et al. 2008).

## Agradecimentos

Os autores agradecem à FUNCAP e à SECITECE pelo apoio financeiro.

### Referências

Alpaydin, E. & Jordan, M. I. (1996). Local linear perceptrons for classification, *IEEE Transactions on Neural Networks* **7**(3): 788–792.

Antonelo, E., Schrauwen, B. & Stroobandt, D. (2008). Mobile robot control in the road sign problem using reservoir computing networks, *IEEE International Conference on Robotics and Automation (ICRA'2008)*, pp. 911 – 916.

Bekey, G. A. (2005). *Autonomous Robots*, MIT Press.

Jacobs, R., Jordan, M., Nowlan, S. & Hinton, G. (1991). Adaptive mixtures of local experts, *Neural Computation* (3): 79–87.