Avaliação de Árvores de Decisão no Controle de Navegação Robótica

Jefferson R. Souza, Gustavo Pessin, Fernando S. Osório e Denis F. Wolf
Universidade de São Paulo (USP)
Instituto de Ciências Matemáticas e de Computação (ICMC)
Av. do Trabalhador São-Carlense, 400 - Caixa Postal 668 - 13.560-970 - São Carlos, SP
Email: {jrsouza, pessin, fosorio, denis}@icmc.usp.br

***Resumo*—Este artigo apresenta a avaliação de Árvores de Decisão (AD) aplicadas à navegação robótica. Navegação robótica é uma tarefa fundamental em robôs móveis, sendo que o robô deve desviar dos obstáculos de maneira apropriada para concluir um trajeto. Neste artigo, o robô móvel desenvolvido atua em um ambiente de simulação realístico, respeitando propriedades físicas. As entradas das AD são adquiridas pelos sensores presentes no robô. O método proposto é empregado na navegação robótica com o objetivo de realizar o trajeto pela AD desviando de obstáculos. As regras e as AD *best-first*, *J48* e *CART* foram utilizadas na avaliação.**

***Palavras-chave*-Aprendizado de Máquina, Árvore de Decisão, Robôs Móveis e Controle de Navegação.**

I. Introdução

A robótica móvel tem alcançado um progresso significativo nos últimos anos. A aplicação dos robôs móveis nas atividades cotidianas demonstra que esta área é promissora. Alguns exemplos de aplicações dos robôs móveis podem ser encontrados, como: robôs com pernas capazes de caminhar como o cachorro *Aibo* [25], o humanóide *Asimo* [10], robôs domésticos utilizados para limpar a casa com o *Roomba* e o *Scooba* [11], exploração espacial, desarme de bombas, entre outros.

O desenvolvimento de algoritmos e técnicas computacionais úteis para a coordenação dos conjuntos físicos (*hardware* do robô, sensores e atuadores) em um ambiente dinâmico é um desafio extremamente complexo [8]. A utilização de ambientes simulados de robôs móveis demonstram uma série de vantagens comparadas com os robôs reais, como: economia de tempo (pode-se realizar um número maior de experimentos por meio da simulação), evita danos nos robôs (através das simulações verifica-se as situações prévias como colisões, acionamento indevido dos motores, entre outros), e, por fim segurança (nestes ambientes, diversos testes são realizados para garantir uma maior segurança, confiabilidade e robustez para o sistema proposto).

Em [17] foi realizada a avaliação de uma Rede Neural Artificial (RNA) aplicada ao controle de navegação dos robôs móveis, onde o sistema proposto atuou em um terreno irregular e com simulação de ruído nos sensores e atuadores. Os resultados demonstram que a RNA é capaz de controlar os robôs móveis de forma satisfatória, e que o método proposto de RNA pode ser utilizado em sistemas onde seja necessário o controle reativo de navegação. O controle reativo de navegação consiste de um sistema de reação sensorial-motora. Neste tipo de controle, existe um laço sensório-motor contendo: leitura dos sensores, processamento das informações de forma imediata e geração de comandos para os atuadores. Normalmente um sistema de controle reativo precisa apenas das leituras atuais sensorias realizadas com o objetivo de tomada de decisão e geração de comandos de ação [5].

Aprendizado de Máquina é uma aréa da Inteligência Artificial em que são investigados técnicas computacionais de aprendizagem e aquisição de conhecimento [20]. Uma possível técnica de Aprendizado de Máquina empregada para o controle de navegação são Árvores de Decisão, dada sua eficiência na construção dos modelos, o tamanho das árvores geradas são normalmente medianas e inteligíveis, facilmente convertidas para regras, além de serem boas para tratar com atributos irrelevantes [7]. O uso dos algoritmos de AD aplicado a navegação robótica torna a pesquisa relevante de acordo com os trabalhos correlatos propostos em [22], [26] e [23].

O método proposto neste artigo tem como objetivo propor uma possível solução para uma deficiência conhecida na robótica móvel que é a navegação. Tratando-se de navegação é válido observar três aspectos fundamentais: a localização, a orientação e o controle do motor. Um robô móvel necessita de sensores (e.g. GPS, bússola) para saber seu estado atual no ambiente em que está inserido (localização e orientação). Para o controle de atuação, é preciso uma quantidade adequada de motores, onde um veículo autônomo usualmente necessita de um motor angular, para giro das rodas e um motor linear, para tração. Portanto, o método proposto realiza a ação adequada com o veículo móvel simulado, concluindo o trajeto adequadamente.

Este artigo é organizado da seguinte forma: A Seção I descreve os objetivos e motivações da pesquisa. As Seções II e III detalham, respectivamente, fundamentos essenciais da robótica móvel e de simulação e modelagem de sistemas robóticos. Na Seção IV são apresentados conceitos de Árvores de Decisão. Na Seção V é descrito o método proposto. A Seção VI descreve os resultados experimentais e a discussão. Concluindo, a Seção VII aborda a conclusão e os trabalhos futuros.

II. Robótica Móvel

A robótica móvel é uma aréa relativamente recente que aborda o controle de sistemas autônomos e semi-autônomos [6]. Cada vez mais tem crescido o interesse e a demanda por Robôs Móveis Autônomos (RMA) e Veículos Autônomos Inteligentes [14], tanto no meio acadêmico como empresarial.

Um robô móvel é um dispositivo mecânico com uma base não fixa agindo sob o controle de um sistema computacional, equipado com sensores e atuadores, os quais permitem interação com o ambiente [3]. Esta interação é dada por meio de ciclos de percepção-ação que consistem em três passos: obtenção da informação através dos sensores, processamento das informações para a seleção de ação e execução da ação por meio da ação dos atuadores. Os sensores representam a percepção de um robô e medições físicas como: contato, obstáculos e orientação; além de prover dados que necessitem ser interpretados pelo robô. Esta interpretação é a única maneira de um robô autônomo compreender o ambiente para realizar as ações devidas [13]. Os atuadores realizam as ações do robô (e.g. motores, pistões), controlados por circuitos eletrônicos que recebem valores de ação que devem ser calculados pelo robô e precisam estar configurados adequadamente.

III. Simulação e Modelagem

Na robótica móvel os experimentos podem ser realizados de duas maneiras: utilizando um robô real ou através de um simulador de ambiente virtual realística [18]. Como dito anteriormente, a utilização de um robô simulado apresenta vantagens como: economia de tempo, evita danos nos robôs e uma maior segurança. Neste trabalho, para realizar os testes de maneira adequada, é essencial que existam algumas restrições físicas no modelo e um terreno contendo diversos obstáculos. Para a implementação do método proposto foram utilizadas bibliotecas de programação em C/C++ descritas a seguir.

Open Dynamics Engine (ODE) [24] é uma biblioteca desenvolvida para simulação física de corpos rígidos articulados. Uma estrutura articulada é desenvolvida quando os corpos rígidos são conectados por algum tipo de articulação, por exemplo, um veículo terrestre que possua uma conexão de rodas em um chassi. A utilização desta biblioteca é essencial por fornecer restrições físicas, principalmente quanto a morfologia do robô. *DrawStuff* é biblioteca de visualização usada neste trabalho, disponibilizada conjuntamente com a ODE. A Figura 1 apresenta o ambiente de simulação desenvolvido com a biblioteca ODE em 3D com chão plano mostrando o veículo desviando dos troncos. Simuladores 2D avaliam um conjunto de questões relativas ao projeto de controle robótico, porém apresentam algumas limitações quanto a representar o retorno sensorial quando o ambiente possui elementos de diferentes formas e tamanhos [5].

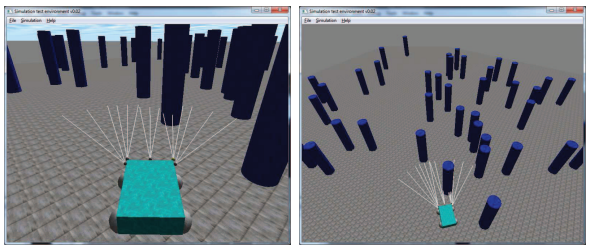


Figura 1. Imagens de simulações utilizando o veículo desenvolvido.

IV. Árvores de Decisão

Árvores de decisão são um dos métodos de aprendizado mais amplamente utilizados para inferência indutiva. O aprendizado de AD é um método usado para obter funções de saída discreta, no qual a função aprendida é representada por uma AD. As árvores aprendidas podem ser também representadas como conjunto de regras *if-then* para melhorar a legibilidade humana [15].

As AD consistem de nodos que representam os atributos, de arcos, provenientes destes nodos e que recebem os valores possíveis para estes atributos, e de nodos terminais ou folhas, que representam as diferentes classes de um conjunto de dados. AD é uma técnica de aprendizado supervisionado vastamente utilizada em Aprendizado de Máquina e Mineração de Dados [2]. Os modelos de AD são construídos por um processo de particionamento recursivo (particionar recursivamente um conjunto de dados até que cada subconjunto obtido deste particionamento contenha casos de uma única classe), além de utilizar uma estratégia de dividir-para-conquistar [1], por exemplo, um problema com uma complexidade alta é dividido em sub-problemas mais simples.

A. Modelos

Existem diversos modelos ou algoritmos de aprendizado de máquina baseados em AD, os mais populares são *ID3*, *C4.5* e *CART* (*Classification and Regression Tree*) ou (Árvore de Classificação e Regressão). Em geral, os algoritmos de AD atuais são recursivos. Neste trabalho foram utilizados três modelos são:

- ***Best-First*** O algoritmo *Best-First* (*BFirst*) adiciona o nodo na melhor divisão em cada etapa da árvore, isto é, o melhor nodo é aquele que reduz a impureza (ruído) máxima entre todos os nodos disponíveis [21]. Ambos pré-poda e pós-poda podem ser realizados neste algoritmo. O problema deste modelo é como determinar qual atributo dividir e como dividir os dados.
- ***J48*** O modelo *J48* implementa o algoritmo *C4.5* [19] para gerar uma decisão com poda ou sem poda. *C4.5* é uma extensão do algoritmo *ID3*. *J48* constrói AD a partir de um conjunto de dados de treinamento rotulado usando o conceito da entropia de informação; utiliza o

fato de que cada atributo de dados pode ser usado para fazer uma decisão dividindo os dados dentro de subconjuntos menores. O *J48* examina o ganho de informação normalizado (diferença na entropia), isto resulta em escolher um atributo para divisão dos dados. Para fazer a decisão, o atributo com o ganho de informação normalizado mais alto é utilizado. Então, o algoritmo retorna nos subconjuntos menores. O algoritmo de divisão pára se todas as instâncias em um subconjunto pertencem à mesma classe. Um nodo folha ou terminal é criado na árvore de decisão informando para a escolha desta classe. Mas pode acontecer também que nenhuma das características dê qualquer ganho de informação. Neste caso, *J48* cria um nodo decisão mais alto na árvore usando o valor esperado da classe [19].

- ***CART*** É uma árvore de decisão binária construída pela divisão de um nodo raiz em dois nodos filhos respectivamente, começando com o nodo raiz o qual contém uma amostra de aprendizado completa. Este modelo é um algoritmo de predição e exploração de dados similar ao *C4.5*, além de ser um algoritmo de construção de árvore. Em [4] é explicado em detalhes o conceito da árvore de classificação e regressão. O *CART* é utilizado em um conjunto de problemas de classificação, comparado com as AD já mencionadas, ele não é o melhor algoritmo a ser empregado, mesmo sendo fácil de utilizar e compreender. Cada nodo da decisão tem uma condição que é representada por uma função f e um parâmetro de divisão, com o objetivo de dividir o conjunto de dados de forma adequada. Cada nodo folha pertence a uma classe C. Usualmente é fácil de utilizar AD para interpretar as regras da árvore, dessa forma podemos analisar e interpretar a representação de um mapeamento de entrada e saída não-linear [12].

V. Método Proposto

Nesta seção é apresentado o método proposto, tendo como objetivo uma análise comparativa entre as AD (*BFirst*, *J48* e *CART*) aplicadas no controle de navegação robótica em um ambiente de simulação realístico.

O simulador (desenvolvido em [16]) permite extrair os dados do trajeto que o robô móvel realiza no ambiente simulado, onde existem diversos obstáculos. Após obter os dados de treinamento é realizada a construção das AD. Utilizamos sete atributos de entrada, que correspondem as seguintes informações: (i) Orientação (ângulo) do veículo, em relação ao plano (x,y), obtido através de uma bússola simulada; (ii) Azimute (ângulo para o alvo) do veículo, obtido através da bússola, do GPS e da mensagem contendo a coordenada do alvo; (iii) Cinco valores de sensores de distância (sonares) são extraídos do veículo. E duas saídas, que são: (i) Força a aplicar no motor angular (giro da barra de direção, de -1.5 a 1.5); (ii) Força a aplicar no motor linear (torque, de 1.5). Um mapeamento de entradas e saídas é fundamental para configurar adequadamente os dados obtidos do simulador, e estes serem aplicados como entrada para o WEKA [27].

Tabela I
Resultados das instâncias e percentuais classificados corretamente das AD sem poda e com poda.

Modelos	Instâncias		Percentuais	
	Sem poda	Com poda	Sem poda	Com poda
Best-First	52679	52673	**98.3441%**	98.3329%
J48	52801	52791	**98.5719%**	98.5532%
CART	52682	52687	98.3497%	**98.3590%**

Os dados são inseridos na ferramenta *Waikato Environment for Knowledge Analysis* (WEKA) [27], que representa uma coleção de algoritmos de aprendizado de máquina. WEKA possui pré-processamento de dados, classificação, regressão, agrupamento, associação de regras e visualização de dados, como suas características. Nesta ferramenta foram geradas as regras das AD (*Best-First*, *J48* e *CART*) sem poda e com poda (utilizando a opção pós-poda). Pós-poda cria uma árvore completa que pode gerar o sobre-ajustamento (*overfitting*) na primeira etapa, então as sub-árvores desnecessárias são cortadas na segunda etapa. A pré-poda pára de crescer a árvore quando um nodo é alcançado próximo a uma acurácia acima de um *threshold* predefinido (ao invés de 100% como na versão básica do algoritmo *ID3*). A pós-poda é mais utilizada e confiável, mas requer um processo mais lento, enquanto a pré-poda tem a vantagem de não gastar tempo na construção de uma estrutura que não será utilizada no final da árvore [9].

Depois de obter as AD sem poda e com poda, são adquiridas as taxas de classificação, e as árvores que apresentarem uma maior classificação serão desenvolvidas com o objetivo de serem aplicadas para o controle robótico realizando o trajeto sobre o ambiente.

VI. Resultados Experimentais

No treinamento das AD, utilizamos a validação cruzada com 10-*folds* sobre os dados de treino, com o objetivo de termos dados mais confiáveis; além de variar o parâmetro poda (opções disponíveis sem poda e com poda) para a obtenção das AD na ferramenta WEKA.

Para validar nosso método proposto, partimos do princípio da classificação do conjunto de dados teste, que foram aplicados nas três AD (*BFirst*, *J48* e *CART*) com o propósito de navegar o robô móvel sob um trajeto completo (ponto de início até ponto de chegada). A Tabela I apresenta em detalhes os resultados das instâncias e os percentuais classificados corretamente dos modelos sem poda e com poda.

Na Tabela I são apresentados os modelos das AD seguido da quantidade de instâncias e percentuais classificados corretamente, dessa maneira podemos ver que a *J48* sem poda,

apresenta o maior percentual de instâncias classificadas corretamente. Essa árvore foi escolhida para ser convertida em um programa em C/C++ e aplicada no controle de navegação. Para analisarmos os detalhes das instâncias que foram classificadas de maneira correta e incorreta as Tabelas II, III e IV apresentam as matrizes de confusão das AD desenvolvidas neste artigo. Nas Tabelas II, III e IV são apresentadas 15 saídas, os quais representam o mapeamento da força aplicada no motor angular (giro da direção de -1.5 a 1.5, mais especificamente -1.5, -1.0, -0.8, -0.5, -0.3, -0.2, -0.1, 0.0, 0.1, 0.2, 0.3, 0.5, 0.8, 1.0, 1.5) combinado diretamente com a força aplicada no motor linear (torque de 1.5). Totalizando 15 combinações possíveis (folhas da árvore).

As instâncias classificadas corretamente que apresentaram valores iguais entre os modelos foram: *BFirst* e *CART*, correspondendo mais especificamente as saídas 1, 2, 7, 8, 9, 10, 12 e 15. Como já observado anteriormente na Tabela I, o algoritmo *J48* apresentou 98.5719% de percentual de acerto. Então, a matriz de confusão da Tabela III apresenta as saídas que foram maiores, ou seja, seus valores classificados comparada com as matrizes de confusão *BFirst* e *CART*, são elas: a maioria das saídas menos a 1, 3 e 12. Com relação as saídas que apresentaram valores menores, isto é, percentual de classificação ou quantidade de instâncias distribuídas corretamente foram: a maioria das saídas do algoritmo *BFirst* menos as saídas 2, 3, 12 e 14. Desta forma, também pode ser conferida na Tabela I, pois foi o modelo que apresentou um percentual menor de classificação correta (98.3441%).

Depois de descrever os resultados das taxas de classificação e matrizes de confusão das AD, serão empregados no controle do robô móvel, inserindo no sistema de controle os códigos (C/C++) da AD *J48*, buscamos avaliar se o controle desenvolvido pela AD é eficiente para realizar a navegação entre os pontos iniciais e finais solicitados para o robô. O robô móvel atua em um ambiente simulado fisicamente, desenvolvido em C/C++ com a biblioteca ODE. As entradas da AD são adquiridas pelos sensores utilizados no robô (e.g. GPS, bússola, sonar). Experimentos com diferentes quantidades de obstáculos no ambiente foram utilizados, com 1%, 2% e 3% de ocupação.

No decorrer do trabalho [16], foi desenvolvido um conjunto de regras para realizar navegação com desvio de obstáculos em robôs móveis. Nesta seção de experimentos são comparadas essas regras com as AD aplicadas na navegação, considerando coordenadas iniciais e finais aleatórias, buscando desviar de obstáculos (sem derrubar nenhum tronco no trajeto realizado), o resultado da navegação pode ser visto na Tabela V.

A Tabela V apresenta os resultados de 40 simulações, sendo 10 simulações com cada tipo de controle e/ou diferentes ocupações[1] por obstáculos no ambiente, apresentando a comparação entre as regras e a AD *J48*. O modelo *J48* com 1% de densidade de ocupação apresenta 100% de resultados satisfatórios de navegação, ou seja, o robô móvel realizou o trajeto perfeitamente de um ponto inicial a um ponto final sem colidir com os obstáculos em 10 simulações. Usando *J48* no robô com o ambiente com 2% e 3% de densidade de ocupação resultou em 80% de simulações satisfatórias. 80% é um percentual considerado visto as dificuldades que o robô tem ao realizar o trajeto proposto. Portanto, acima de 3% na densidade de ocupação, o trajeto apresenta uma dificuldade maior, pois os obstáculos estão muito próximos uns dos outros e o modelo necessita de uma generalização para desviar perfeitamente os obstáculos.

Tabela V
RESULTADOS DAS SIMULAÇÕES REALIZANDO A COMPARAÇÃO ENTRE AS REGRAS E A ÁRVORE DE DECISÃO *J48*.

Número de Simulações	Regras	AD	AD	AD
	Ocupação com obstáculo			
	2%	1%	2%	3%
	Resultados satisfatórios da navegação			
10	100%	100%	80%	80%

A Figura 2 apresenta as imagens de simulações com aplicação das regras e da AD *J48* controlando o robô. Ambas as imagens apresentam trajetórias de navegação com densidade de ocupação de 3%. Podemos ver que a trajetória dos robôs realizada pelas regras e pela AD *J48* é satisfatória, ou seja, as duas técnicas realizaram a trajetória de maneira correta, desviando dos obstáculos.

VII. CONCLUSÃO E TRABALHOS FUTUROS

Neste artigo foi realizada uma análise comparativa de AD aplicadas no controle de navegação robótica em um ambiente virtual de simulação. O robô móvel atua neste ambiente, contendo diversos obstáculos. O *J48* apresentou resultados satisfatórios capazes de controlar o robô móvel, obtendo dados pelos sensores do veículo em um ambiente com propriedades físicas; mesmo apresentando 80% nos casos críticos, como 2% e 3% com densidade de ocupação (Tabela V). Para trabalhos futuros, serão investigados o comportamento do robô após serem inseridos ruídos nos sensores e atuadores para comparar com os resultados obtidos da RNA no trabalho descrito em [17], além de aplicar as AD descritas neste trabalho em um robô real e ser observado o comportamento que elas desempenham.

AGRADECIMENTOS

Os autores agradecem a FAPESP e CAPES pelo apoio financeiro através de bolsa de estudo; e também ao CNPq

[1]Vídeo da navegação usando a AD *J48* em um ambiente com 2% de ocupação, disponível em: http://www.youtube.com/watch?v=BeYUcqypMdE

Tabela II
RESULTADOS DA MATRIZ DE CONFUSÃO DA ÁRVORE DE DECISÃO *Best-First* SEM PODA.

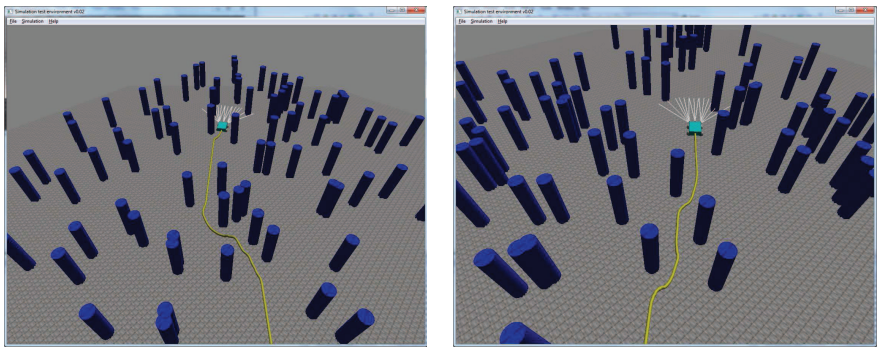
a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	classificação
3011	19	0	1	2	3	0	0	0	0	0	0	0	0	0	saida1
24	**4485**	9	1	2	5	0	1	0	0	0	0	0	0	0	saida2
0	23	**4136**	186	4	1	0	0	3	0	1	0	0	0	0	saida3
0	0	102	**26217**	110	1	0	0	0	1	0	0	0	0	0	saida4
0	0	5	195	**3882**	3	0	1	1	0	20	1	2	1	0	saida5
2	6	2	3	3	**761**	10	6	0	0	0	0	2	0	0	saida6
0	0	1	0	1	10	**133**	0	0	0	0	1	0	0	0	saida7
0	1	2	0	0	6	0	**562**	0	0	0	0	6	0	0	saida8
0	0	7	0	1	0	0	0	**471**	1	1	1	0	0	0	saida9
0	0	0	4	2	0	0	0	3	**451**	3	0	0	0	1	saida10
0	0	0	2	20	0	0	0	0	0	**4129**	3	0	7	0	saida11
0	0	1	0	0	0	0	0	1	0	2	**506**	0	0	3	saida12
0	0	0	0	6	2	1	3	0	0	0	0	**347**	0	0	saida13
0	0	0	1	2	0	0	0	0	0	10	1	0	**3473**	0	saida14
0	0	1	0	1	1	0	0	0	3	0	2	0	0	**115**	saida15

Tabela III
RESULTADOS DA MATRIZ DE CONFUSÃO DA ÁRVORE DE DECISÃO *J48* SEM PODA.

a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	classificação
3011	24	0	0	0	1	0	0	0	0	0	0	0	0	0	saida1
22	**4488**	8	3	1	5	0	0	0	0	0	0	0	0	0	saida2
0	11	**4129**	210	3	0	0	0	0	0	1	0	0	0	0	saida3
0	0	85	**26246**	99	1	0	0	0	0	0	0	0	0	0	saida4
0	0	4	179	**3905**	3	0	0	0	1	18	1	0	0	0	saida5
0	4	0	0	2	**778**	8	2	0	0	0	0	0	0	1	saida6
0	0	0	0	0	10	**135**	0	0	0	0	1	0	0	0	saida7
0	0	0	0	0	2	0	**574**	0	0	0	0	1	0	0	saida8
0	0	0	0	1	0	0	0	**480**	0	0	1	0	0	0	saida9
0	0	1	0	0	0	0	0	1	**462**	0	0	0	0	0	saida10
0	0	0	0	11	0	0	0	0	0	**4137**	3	0	10	0	saida11
0	0	1	0	3	0	0	0	1	0	1	**505**	0	1	1	saida12
0	0	0	0	1	3	0	0	0	0	0	0	**355**	0	0	saida13
0	0	0	0	0	0	0	0	0	0	8	1	0	**3478**	0	saida14
0	0	0	0	2	2	0	0	0	0	0	1	0	0	**118**	saida15

Tabela IV
RESULTADOS DA MATRIZ DE CONFUSÃO DA ÁRVORE DE DECISÃO *CART* COM PODA.

a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	classificação
3011	19	0	1	2	3	0	0	0	0	0	0	0	0	0	saida1
22	**4485**	11	1	2	5	0	1	0	0	0	0	0	0	0	saida2
0	23	**4125**	196	4	2	0	0	3	0	1	0	0	0	0	saida3
0	0	89	**26229**	111	1	0	0	0	1	0	0	0	0	0	saida4
0	0	4	193	**3884**	3	0	1	1	0	20	1	3	1	0	saida5
2	5	2	1	3	**764**	10	6	0	0	0	0	2	0	0	saida6
0	0	1	0	1	10	**133**	0	0	0	0	1	0	0	0	saida7
0	1	2	0	0	6	0	**562**	0	0	0	0	6	0	0	saida8
0	0	7	0	1	0	0	0	**471**	1	1	1	0	0	0	saida9
0	0	0	4	2	0	0	0	3	**451**	3	0	0	0	1	saida10
0	0	0	2	19	0	0	0	0	0	**4130**	3	0	7	0	saida11
0	0	1	0	0	0	0	0	1	0	2	**506**	0	0	3	saida12
0	0	0	0	7	2	0	1	0	0	0	0	**349**	0	0	saida13
0	0	0	1	2	0	0	0	0	0	11	1	0	**3472**	0	saida14
0	0	1	0	1	1	0	0	0	3	0	2	0	0	**115**	saida15



(a) (b)

Figura 2. Imagens apresentando trajetórias de navegação. (a) Robô usando Regras. (b) Robô utilizando a árvore de decisão *J48*.

e FAPESP pelo financiamento ao INCT-SEC (Instituto Nacional de Ciência e Tecnologia - Sistemas Embarcados Críticos), processos 2009/11614-4, 2008/573963-9 e 2008/57870-9.

REFERÊNCIAS

[1] Alpaydin, E. Introduction to Machine Learning. 2 ed. The MIT Press Cambridge, 2010.

[2] Barry, D. V. Decision Trees for Business Intelligence and Data Mining: Using SAS. SAS Institute Inc., 2006.

[3] Bekey, G. A. Autonomous Robots: From Biological Inspiration to Implementation and Control. MIT Press, 2005.

[4] Breiman, L., Friedman, J., Olshen, R. and Stone, C. J. Classification and Regression Tree. Wadsworth and Brooks/Cole Advanced Books and Software, Pacific California, 1984.

[5] Carvalho, A. C. P. L. F. e Kowaltowski, T. Atualizações em informática 2009. Rio de Janeiro: PUC, v. 1, 432 p., 2009.

[6] Dudek, G. and Jenkin, M. Computational Principles of Mobile Robotics. Cambridge University Press, 280 p., 2000.

[7] Gama, J. Árvores de Decisão. www.liaad.up.pt/∼jgama/Bdc/arv.pdf (Acesso 06/07/2010).

[8] Go, J., Browning, B. and Veloso, M. Accurate and flexible simulation for dynamic, vision-centric robots. In: International Joint Conference on Autonomous Agents, 2004.

[9] Halmenschlager, C., Um algoritmo para indução de árvores e regras de decisão. Dissertação (Mestrado), UFRGS, 2002.

[10] Humanoid robot ASIMO. http://world.honda.com/ASIMO/ (Acesso 30/06/2010).

[11] Cleaning Robots (Roomba, Scooba, Dirt Dog, Verro). http://www.irobot.com/ (Acesso 30/06/2010).

[12] Jang, J. S. R. Structure Determination in Fuzzy Modeling: A Fuzzy CART Approach. In Proceedings of the IEEE International Conference on Fuzzy Systems, pp. 1-6, 1994.

[13] Jet Propulsion Laboratory/NASA. http://www-robotics.jpl.nasa.gov/ (Acesso 01/07/2010).

[14] Jung, C. R., Osório, Kelber, F. S. e Heinen, F. Computação embarcada: Projeto e implementação de veículos autônomos inteligentes. In: Jornada de Atualização em Informática – Congresso da SBC 2005, pp. 1358-1406, 2005.

[15] Mitchell, T. Machine Learning. McGraw Hill, 1997.

[16] Pessin, G., Evolução de Estratégias e Controle Inteligente em Sistemas Multi-Robóticos Robustos. Dissertação (Mestrado), Universidade do Vale do Rio dos Sinos, UNISINOS, 2008.

[17] Pessin, G., Osório, F. S. e Musse, S. Utilizando RNAs no Controle Robusto de Navegação de Robôs Móveis. In: 12 Cong. de Inf. e Telecomunicações (MT Digital), 2008.

[18] Pfeifer, R. and Scheier, C. Understanding Intelligence. The MIT Press, 1999.

[19] Quinlan, R. S. C4.5: Programs for Machine Learning. Morgan Kaufmann Publishers, San Mateo, 1993.

[20] Rezende, S. O. Sistemas Inteligentes: Fundamentos e Aplicações. Barueri, Manole, 2003.

[21] Shi, H. Best-First Decision Tree Learning. Master in Computer Science, University of Waikato, 2007.

[22] Sillitoe, I. and Elomaa, T. Learning decision trees for mapping the local environment in mobile robot navigation. MLC-COLT Workshop on Robot Learning, pp. 119-125, 1994.

[23] Slusny, S., Neruda, R. and Vidnerova, P. Learning Algorithms for Small Mobile Robots: Case Study on Maze Exploration. Proc. of the Conf. on Theory and Practice of IT, 2008.

[24] Smith, R. Open Dynamics Engine v0.5 User Guide, 2006.

[25] AIBO Entertainment Robots. http://support.sony-europe.com/aibo/ (Acesso 30/06/2010).

[26] Swere, E. and Mulvaney, D. J. Robot Navigation Using Decision Trees. Elec. Sys. and Control Div. Research, 2003.

[27] Waikato Environment for Knowledge Analysis (WEKA). http://www.cs.waikato.ac.nz (Acesso 23/06/2010).