Campus de São Carlos

**UM AGENTE AUTÔNOMO BASEADO EM APRENDIZAGEM
POR REFORÇO DIRECIONADO À META**

Autor: Arthur Plínio de Souza Braga

Orientador: Prof. Dr. Aluizio Fausto Ribeiro Araújo

UNIVERSIDADE DE SÃO PAULO

**ESCOLA DE ENGENHARIA
DE SÃO CARLOS**

***UM AGENTE AUTÔNOMO BASEADO EM
APRENDIZAGEM POR REFORÇO
DIRECIONADO À META***

Arthur Plínio de Souza Braga

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

1998

Class.	TESE
Cutt.	0901
Tombo	0046/99

S/S 1031631

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

B813a Braga, Arthur Plínio de Souza
Um agente autônomo baseado em aprendizagem por reforço
direcionando à meta / Arthur Plínio de Souza Braga. --
São Carlos, 1998.

Dissertação (Mestrado) -- Escola de Engenharia de São
Carlos-Universidade de São Paulo, 1998.
Área: Engenharia Elétrica.
Orientador: Prof. Dr. Aluizio Fausto Ribeiro Araújo.

1. Aprendizagem por reforço. 2. Agente autônomo.
3. Inteligência artificial. 4. Planejamento.
5. Comportamento reativo. 6. Navegação de robô móvel.
6. Problema de aprendizagem por reforço direcionada à
meta. 7. Taxa de aprendizagem variável. I. Título.

FOLHA DE APROVAÇÃO

Candidato: Engenheiro **ARTHUR PLINIO DE SOUZA BRAGA**

Dissertação defendida e aprovada em 16.12.98
pela Comissão Julgadora:

Prof. Dr. **ALUIZIO FAUSTO RIBEIRO ARAÚJO (Orientador)**
(Escola de Engenharia de São Carlos – Universidade de São Paulo)

Prof. Assoc. **MARCELO GODOY SIMÕES**
(Escola Politécnica – Universidade de São Paulo)

Prof. Assoc. **ANDRÉ PONCE DE LEON F. DE CARVALHO**
(Instituto de Ciências Matemáticas e Computação – Universidade São Paulo)

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

JOSÉ CARLOS A. CINTRA
Presidente da Comissão de Pós-Graduação da EESC

*Dedico este trabalho aos meus pais,
José Newton e Elizete,
pelo contínuo apoio e incentivo.*

AGRADECIMENTOS

Ao Prof. Dr. Aluizio Fausto Ribeiro Araújo que, mais que um orientador acadêmico, soube ser um grande amigo.

A Beatriz Adelino Brasileiro Silva que, pelo apoio, carinho e compreensão, contribuiu muito para a conclusão deste trabalho.

A todas as pessoas que me ajudaram nesta caminhada, agradeço a paciência, o incentivo e a amizade. Em especial, agradeço aos companheiros do Departamento de Engenharia Elétrica: Guilherme Barreto, Marcelo Rosa, Renato Tinós, Marcelo Vieira, Hélio D'Arbo, Ricardo Sovat, João Bosco, Patrícia Leite e Mário Oleskowski.

A Renato Iwamoto pelo auxílio na implementação da parte prática desta pesquisa.

Ao Departamento de Engenharia Elétrica, pela utilização e fácil acesso aos seus laboratórios, bem como aos professores e funcionários que estiveram sempre dispostos a colaborar.

À Fundação Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES), pelo suporte financeiro fornecido durante o período da realização deste trabalho, sem o qual o mesmo não seria possível.

SUMÁRIO

LISTA DE FIGURAS.....	I
LISTA DE TABELAS.....	V
LISTA DE SIGLAS E ABREVIATURAS.....	VI
LISTA DE SÍMBOLOS	VII
RESUMO	VIII
<i>ABSTRACT</i>	IX
CAPÍTULO 1	1
INTRODUÇÃO	1
1.1 DESENVOLVIMENTO DA PESQUISA.....	2
1.2 ORGANIZAÇÃO DA DISSERTAÇÃO	3
CAPÍTULO 2	5
O AGENTE E SUA NAVEGAÇÃO	5
2.1 O AGENTE AUTÔNOMO.....	5
2.2 A NAVEGAÇÃO AUTÔNOMA DE UM ROBÔ MÓVEL	8
2.3 CARACTERÍSTICAS PRESENTES NO AGENTE DESEJADO	11
2.4 CONCLUSÃO	13
CAPÍTULO 3	15
ABORDAGENS PARA O AGENTE AUTÔNOMO	15
3.1 ABORDAGENS PARA A CRIAÇÃO DE UM AGENTE AUTÔNOMO ROBÓTICO DE NAVEGAÇÃO	15
3.1.1 Sistemas Planejadores (Deliberativos).....	16
3.1.1.1 Exemplo desta abordagem.....	17
3.1.1.2 Limitações.....	19
3.1.2 Sistemas Reativos	19
3.1.2.1 Exemplo desta abordagem.....	22
3.1.2.2 Limitações.....	23
3.1.3 Sistemas Deliberativos/Reativos.....	23
3.1.3.1 Aprendizagem por Reforço: A Metodologia Adotada	25

3.2 PROBLEMA DE APRENDIZAGEM POR REFORÇO DIRECIONADO À META.....	25
3.3 CONCLUSÕES.....	26
CAPÍTULO 4	28
APRENDIZAGEM POR REFORÇO.....	28
4.1 CONCEITOS GERAIS DE APRENDIZAGEM POR REFORÇO	28
4.2 APRENDIZAGEM POR DIFERENÇA TEMPORAL	34
4.3 APRENDIZAGEM POR REFORÇO EM ROBÔS MÓVEIS	37
4.4 CONCLUSÕES.....	38
CAPÍTULO 5	40
DESCRIÇÃO DO AGENTE PROPOSTO	40
5.1 NÍVEL COMPUTACIONAL	40
5.2 NÍVEL ALGORÍTMICO	44
5.2.1 Definição da Representação do Sinal de Reforço	44
5.2.1.1 Representação por Premiação da Meta (Goal-Reward Representation).....	44
5.2.1.2 Representação por Penalização das Ações (Action-Penalty Representation).....	45
5.2.1.3 Representação Adotada para os Sinais de Reforço.....	46
5.2.2 Apresentação do Processamento do Agente.....	50
5.3 NÍVEL IMPLEMENTACIONAL	55
5.3.1 O Robô Real.....	55
5.3.2 Considerações para a Interação Robô-Ambiente ser um PDM.....	56
5.3.3 Problemas Enfrentados com o Robô Real	59
5.4 CONCLUSÕES.....	61
CAPÍTULO 6	62
TESTES COM O AGENTE PROPOSTO	62
6.1 A FERRAMENTA DE SIMULAÇÃO	62
6.2 SOLUÇÃO DO PROBLEMA DE EXPLORAÇÃO DIRECIONADA À META.....	66
6.3 SOLUÇÃO DO PROBLEMA DE APRENDIZAGEM POR REFORÇO DIRECIONADA À META	71
6.3.1 A Otimalidade dos Caminhos.....	75
6.3.2 Reaproveitamento de Trajetórias Aprendidas	77
6.3.3 A Estrutura do Ambiente Influenciando o Desempenho do Agente	78
6.3.4 Influência do Tamanho do Espaço de Estados no Desempenho do Agente	80
6.4 OS COMPORTAMENTOS REATIVOS E DELIBERATIVOS DO AGENTE	83
6.4.1 Gerando Comportamentos Reativos	83
6.4.2 Planejamento Deliberativo do Agente	85
6.5 O AGENTE EM AMBIENTES DINÂMICOS	85
6.6 CONCLUSÕES.....	87

CAPÍTULO 7	89
CONCLUSÕES E TRABALHOS FUTUROS.....	89
7.1 ADEQUAÇÃO DA SOLUÇÃO PROPOSTA ÀS CARACTERÍSTICAS ESPERADAS DO AGENTE	89
7.1.1 Habilidade para Operar em Ambientes Inicialmente Desconhecidos	90
7.1.2 Capacidade de Lidar com Ambientes Dinâmicos	90
7.1.3 Habilidade para Gerenciar Múltiplos Objetivos	91
7.1.4 Capacidade para Aprendizagem Incremental	91
7.1.5 Capacidade de Melhorar Soluções Iniciais.....	91
7.1.6 Simplicidade do Mecanismo	92
7.2 CONTRIBUIÇÕES DA PESQUISA.....	92
7.3 LIMITAÇÕES DO AGENTE.....	93
7.4 PROPOSTAS PARA FUTURAS PESQUISAS	94
REFERÊNCIAS.....	96
APÊNDICE I.....	I
PROCESSOS MARKOVIANOS.....	I
I.1 PROCESSOS DE DECISÃO MARKOVIANOS (PDMs).....	IV
APÊNDICE II.....	V
OS PRIMEIROS AGENTES E SEUS PROBLEMAS	V
II.1 PRIMEIRA VERSÃO DO AGENTE.....	V
II.2 SEGUNDA VERSÃO DO AGENTE.....	VIII
II.3 TERCEIRA VERSÃO DO AGENTE	IX
II.4 QUARTA VERSÃO DO AGENTE	IX

LISTA DE FIGURAS

FIGURA 1 - DIAGRAMA ESQUEMÁTICO GERAL DO AGENTE E SUA INTERAÇÃO COM O AMBIENTE DE TRABALHO.....	1
FIGURA 2 - UM AGENTE GENÉRICO É CAPAZ DE PERCEBER O AMBIENTE EM QUE ESTÁ E GERAR AÇÕES QUE AFETEM O ESTADO DESTA AMBIENTE.....	6
FIGURA 3 - CLASSIFICAÇÃO DOS AGENTES AUTÔNOMOS SEGUNDO FRANKLIN E GRASSER (1997).....	7
FIGURA 4 - DIAGRAMA GERAL DO SISTEMA DE CONTROLE DE NAVEGAÇÃO BASEADO EM GRAY E CALDWELL (1996).....	11
FIGURA 5 - DIAGRAMA CONSIDERADO PARA O FUNCIONAMENTO DO AGENTE.....	14
FIGURA 6 - ARQUITETURA GERAL DOS SISTEMAS DELIBERATIVOS.....	16
FIGURA 7 - (A) APARÊNCIA DO ROBÔ SHAKEY. (B) AMBIENTE DE TESTE DO ROBÔ MÓVEL.....	17
FIGURA 8 - OPERADORES QUE DESCREVEM AS AÇÕES DO ROBÔ DE: (A) ATRAVESSAR UMA PORTA E (B) ARRASTAR UM OBJETO ATRAVÉS DE UMA PORTA.....	18
FIGURA 9 - (A) ESTADOS INICIAL E FINAL NO MUNDO DOS BLOCOS. (B) PLANO GERADO PELO PLANEJAMENTO SE-ADICIONE-DELETE.....	18
FIGURA 10 - SISTEMA REATIVO E SUAS CAMADAS.....	21
FIGURA 11 - (A) O ROBÔ HERBERT. (B) UM EXEMPLO DE TRAJETÓRIA GERADA PELO MANIPULADOR DO ROBÔ (IMAGENS OBTIDAS NA HOME-PAGE: HTTP://WWW.PCREALM.NET/~JOHUCO/D-COORD.HTML).....	22
FIGURA 12 - ESTRUTURA GENÉRICA DO APRENDIZADO POR REFORÇO.....	30
FIGURA 13 - (A) AGENTE COMPOSTO POR SEIS ESTADOS. (B) O REFORÇO R_t É MEDIDO NO INSTANTE t E CORRESPONDE AO TEMPO DECORRIDO DO ESTADO S_0 ATÉ O ESTADO S_t , A AVALIAÇÃO DO ESTADO, $V(S_t)$, É A ESTIMATIVA DE TEMPO PARA ALCANÇAR O ESTADO META S_5 A PARTIR DO ESTADO S_t E O VALOR QUE SE DESEJA CALCULAR. T É O TEMPO REAL DECORRIDO AO PARTIR DO ESTADO S_0 ATÉ ATINGIR O ESTADO META S_5	34
FIGURA 14 - (A) EXEMPLO DE MUDANÇAS REALIZADAS EM MONTE CARLO. (B) EXEMPLO DE MUDANÇAS REALIZADAS EM TD. NOS DOIS GRÁFICOS, AS SETAS INDICAM O INCREMENTO QUE É DADO EM $V(S_t)$ PARA CORRIGIR A ESTIMATIVA.....	36
FIGURA 15 - ARQUITETURA PARA O CRÍTICO HEURÍSTICO ADAPTATIVO (KAELBLING <i>ET AL.</i> , 1996).....	36
FIGURA 16 - GERAÇÃO DE UM COMPORTAMENTO NO QUAL O ANIMAL LEVA EM CONSIDERAÇÃO OS ESTÍMULOS EXTERNOS E O SEU ESTADO INTERNO.....	41
FIGURA 17 - PROCESSO DE INTERAÇÃO AGENTE-AMBIENTE.....	43
FIGURA 18 - (A) AMBIENTE BIDIMENSIONAL COMPOSTO POR 9x9 ESTADOS COM UM ESTADO META E DOIS ESTADOS COM OBSTÁCULO. (B) SUPERFÍCIE DE RECOMPENSA CORRESPONDENTE. (C) SUPERFÍCIE DE PENALIZAÇÃO CORRESPONDENTE.....	48
FIGURA 19 - SUPERFÍCIE DE PENALIZAÇÃO ALTERNATIVA PARA O AMBIENTE DA FIGURA 18.A.....	48

FIGURA 20 - EXEMPLO DA EVOLUÇÃO DAS SUPERFÍCIES DE RECOMPENSA E PENALIZAÇÃO: (A) FORMAS INICIAIS ($\forall s: V(s) = 0$); (B) APARÊNCIA APÓS EXPLORAÇÃO DO AMBIENTE, SEM TER SIDO LOCALIZADO O ALVO; (C) ASPECTO SEGUINTE À LOCALIZAÇÃO DO ALVO.	49
FIGURA 21 - ALGORITMO IMPLEMENTADO EM PORTUGUÊS PROCEDIMENTAL.	50
FIGURA 22 - TENDÊNCIAS DE COMPORTAMENTO QUE COMPÕEM A POLÍTICA DO AGENTE.	52
FIGURA 23 - EXEMPLO DE ESTADO (s_c) VISITADO MAIS QUE UMA VEZ EM TENTATIVAS DIFERENTES.	53
FIGURA 24- EXEMPLO DE ESTADO (s_c) VISITADO MAIS QUE UMA VEZ EM UMA MESMA TENTATIVA.	53
FIGURA 25- EXEMPLO DE COMPORTAMENTO ESPERADO PARA A TAXA DE APRENDIZAGEM (EIXO X = NÚMERO DE VISITAS A UM PONTO E EIXO Y = VALOR DA TAXA DE APRENDIZAGEM).	54
FIGURA 26 - FOTOS DE UM PROTÓTIPO DO ROBÔ MÓVEL SMARTROB.	56
FIGURA 27 - (A) FIGURA REPRODUZIDA DE ZALAMA <i>ET AL.</i> (1995): AS VELOCIDADES ANGULARES ω_L E ω_R SÃO APLICADAS RESPECTIVAMENTE ÀS RODAS ESQUERDA E DIREITA DO ROBÔ. AS RODAS DA FRENTE CONTROLAM A DIREÇÃO DO MOVIMENTO DO ROBÔ. DURANTE UM CURTO INTERVALO DE TEMPO Δt , O ROBÔ MOVE-SE AO LONGO DE UMA TRAJETÓRIA CIRCULAR COM CENTRO EM X_C ATRAVÉS DE UMA DISTÂNCIA ΔS E ÂNGULO $\Delta \theta$. (B) OITO POSSÍVEIS AÇÕES DO AGENTE A CADA ESTADO (EM PRETO).	57
FIGURA 28 - DISCRETIZAÇÃO UTILIZADA PARA O AMBIENTE DO ROBÔ MÓVEL: CADA ESTADO É O ESPAÇO SUFICIENTE PARA ESTE SER CAPAZ DE REALIZAR UMA ROTAÇÃO EM TORNO DE SEU EIXO CENTRAL. AS DISTÂNCIAS PERCORRIDAS PELO ROBÔ, EM CADA TRANSIÇÃO DE ESTADO, DEPENDEM DO MOVIMENTO SER REALIZADO NAS DIAGONAIS OU NOS SENTIDOS DOS EIXOS VERTICAL OU HORIZONTAL.	58
FIGURA 29 - (A) ÁRVORE DE DECISÃO CORRESPONDENTE À NAVEGAÇÃO DE UM ROBÔ MÓVEL. (B) MATRIZ DE TRANSIÇÃO DOS ESTADOS DA ÁRVORE DE BUSCA. (C) DESTAQUE DA TRANSIÇÃO ENTRE DOIS ESTADOS E A PROBABILIDADE DA OCORRÊNCIA DA TRANSIÇÃO.	59
FIGURA 30 - (A) EXEMPLO DE CAMINHO REALIZADO PELO AGENTE DURANTE A EXPLORAÇÃO DO AMBIENTE PARA ENCONTRAR O ESTADO META. (B) ERRO ENTRE A TRAJETÓRIA PLANEJADA (VIRTUAL) PELO AGENTE E A TRAJETÓRIA REAL EXECUTADA PELO ROBÔ.	60
FIGURA 31 - (A)-(C) E (B)-(D) CONFIGURAÇÕES REAIS DO AMBIENTE DO ROBÔ MÓVEL E SUAS RESPECTIVAS REPRESENTAÇÕES NA FERRAMENTA DE SIMULAÇÃO DESENVOLVIDA.	63
FIGURA 32 – (A)-(D) DIFERENTES CONFIGURAÇÕES DOS AMBIENTE UTILIZADOS PARA TESTAR O COMPORTAMENTO DO AGENTE PROPOSTO.	64
FIGURA 33 - INTERFACE GRÁFICA DO AMBIENTE DE SIMULAÇÃO.	65
FIGURA 34 - PLANO XY DAS SUPERFÍCIES DE AVALIAÇÃO DA RECOMPENSA E PENALIZAÇÃO.	66
FIGURA 35 - OS PONTOS MARCADOS COM ‘O’ SÃO PONTOS DE PARTIDA E O INDICADO ‘X’ É A META.	67
FIGURA 36 - (A) SUPERFÍCIE DE RECOMPENSA. (B) SUPERFÍCIE DE PENALIZAÇÃO.	67
FIGURA 37 – (A)-(C) EXEMPLOS DE ALGUMAS SITUAÇÕES EM QUE O AGENTE FICA PRESO EM CAMINHOS FECHADOS.	68
FIGURA 38 - (A) OUTRO EXEMPLO DE SUPERFÍCIE DE RECOMPENSA COM DESCONTINUIDADE NA AVALIAÇÃO QUE CAUSAM ILHAS DE RECOMPENSA. (B) TRAJETÓRIA PERCORRIDA PELO AGENTE DURANTE A FASE DE TREINAMENTO QUE GEROU A SUPERFÍCIE DE RECOMPENSA COM	

DESCONTINUIDADES. O GRANDE NÚMERO DE VISITAS AOS ESTADOS DA REGIÃO À DIREITA DO AMBIENTE TORNAM MUITO BAIXAS AS AVALIAÇÃO DESTES ESTADOS COMO PODE SER VISUALIZADO NA ILHA B.....	68
FIGURA 39 - (A) SUPERFÍCIE DE RECOMPENSA. (B) TRAJETÓRIA PERCORRIDA DURANTE A FASE DE TREINAMENTO. (C) TRAJETÓRIA REALIZADA PELO ROBÔ A PARTIR DA SUPERFÍCIE DE RECOMPENSA OBTIDA.....	69
FIGURA 40 – (A)-(B) SUPERFÍCIES GERADAS AO TREINAR O AGENTE COM RESTRIÇÃO DE ATUALIZAR AS AVALIAÇÕES APENAS NA PRIMEIRA VISTA.	69
FIGURA 41 - ALGUNS EXEMPLOS DE TRAJETÓRIAS OBTIDAS. OS PONTOS ASSINALADOS POR “O” CORRESPONDEM AOS PONTOS INICIAIS DA TRAJETÓRIA, E O PONTO “X” AO PONTO META. (A)-(C) PONTOS INICIAIS TREINADOS DURANTE A FASE DE APRENDIZAGEM. (D)-(I) PONTOS INICIAIS NÃO TREINADOS.	70
FIGURA 42 - TRAJETÓRIAS OBTIDAS APÓS: (A) 1 TREINAMENTO, (B) 100 TREINAMENTOS, (C) 1000 TREINAMENTOS, (D) 10000 TREINAMENTOS.	71
FIGURA 43 - (A)-(D) SUPERFÍCIES DE RECOMPENSA APRENDIDAS PELO AGENTE APÓS, RESPECTIVAMENTE, 1, 100, 1000 E 5574 TREINAMENTOS. (E)-(H) SUPERFÍCIES DE PENALIZAÇÃO PARA OS MESMOS ESTÁGIOS DE TREINAMENTO.	72
FIGURA 44 - (A)-(H) EXEMPLOS DE COMO AS TRAJETÓRIAS REDUZEM O NÚMERO DE ESTADOS VISITADOS COM A EVOLUÇÃO DO TREINAMENTO EM 1, 100, 1000 E 5574 CICLOS DE TREINAMENTO.....	73
FIGURA 45 - (A) EXEMPLO DE UMA TRAJETÓRIA INICIALMENTE ESTABELECIDADA ENTRE OS PONTOS INICIAL (I) E META (M). (B) POSSÍVEL NOVA TRAJETÓRIA, EM NEGRITO, CONSEGUIDA APÓS ATUALIZAÇÃO DAS SUPERFÍCIES DE AVALIAÇÃO QUE DURANTE A FASE DE APRENDIZAGEM MELHORARAM A AVALIAÇÃO DO PONTO A, RESULTANDO EM UM CAMINHO MAIS CURTO.	74
FIGURA 46 - EXEMPLO DE TRÊS TRAJETÓRIAS QUE CONECTAM OS ESTADOS INICIAL E FINAL E POSSUEM O MESMO NÚMERO DE ESTADOS QUE O MENOR CAMINHO ENTRE ESSES DOIS PONTOS.....	75
FIGURA 47 - EVOLUÇÃO DA TRAJETÓRIA REALIZADA PELO AGENTE COM: (A) 500 TREINAMENTO, (B) 1000 TREINAMENTOS E (C) 1500 TREINAMENTOS	76
FIGURA 48 - GRÁFICO DO NÚMERO DE ESTADOS POR TRAJETÓRIA PARA CADA PASSO DE TREINAMENTO.	76
FIGURA 49 - EXEMPLOS DE TRECHOS REAPROVEITADOS DE TRAJETÓRIAS APONTADOS NOS DETALHES. OS CAMINHOS DA PARTE SUPERIOR DA FIGURA PARTEM DE PONTOS TREINADOS, ENQUANTO OS DA PARTE INFERIOR PARTEM DE PONTOS NÃO TREINADOS. EM TODAS AS TRAJETÓRIAS O PONTO ALVO ESTÁ SITUADO NA PARTE INFERIOR DA SALA A ESQUERDA NO AMBIENTE.	77
FIGURA 50 – (A)-(D) SUPERFÍCIES DE RECOMPENSA GERADAS COM 1, 100, 1000 E 10000 TREINAMENTOS. (E)-(H) TRAJETÓRIAS GERADAS PELO AGENTE, PARTINDO DE UM MESMO PONTO PARA ALCANÇAR A META, A PARTIR DAS SUPERFÍCIES OBTIDAS COM 1, 100, 1000 E 10000 TREINAMENTOS, RESPECTIVAMENTE.	78
FIGURA 51 - SUPERFÍCIES DE RECOMPENSA REFERENTES ÀS TRAJETÓRIAS APRESENTADAS NA FIGURA 52 APÓS: (A) 1 TREINAMENTO, (B) 100 TREINAMENTOS, (C) 1000 TREINAMENTOS E (D) 10000 TREINAMENTOS.....	79

FIGURA 52 - (A) TRAJETÓRIA OBTIDA APÓS O PRIMEIRO TREINAMENTO. COM A OBTENÇÃO DA PRIMEIRA SOLUÇÃO QUE INTERLIGA OS PONTOS INICIAL E META NO LABIRINTO, A MELHORIA DA SOLUÇÃO FICA RESTRITA ÀS PROXIMIDADES DESSA PRIMEIRA TRAJETÓRIA CONFORME PODE SER VISTO APÓS: (B) 100 TREINAMENTOS, (C) 1000 TREINAMENTOS E (D) 10000 TREINAMENTOS.	79
FIGURA 53 - SUPERFÍCIES DE RECOMPENSA RELATIVAS ÀS TRAJETÓRIAS DA FIGURA 54: (A) 1 TREINAMENTO, (B) 100 TREINAMENTOS, (C) 1000 TREINAMENTOS.	80
FIGURA 54 - TRAJETÓRIAS OBTIDAS APÓS: (A) 1 TREINAMENTO, (B) 100 TREINAMENTOS, (C) 1000 TREINAMENTOS.	80
FIGURA 55 - (A)-(C): SUPERFÍCIES GERADAS COM LIMITE INFERIOR DA TAXA DE APRENDIZAGEM IGUAL A 0.1 APÓS 1 TREINAMENTO. (D)-(F) TRAJETÓRIAS OBTIDAS A PARTIR DAS SUPERFÍCIES. NAS TRÊS TRAJETÓRIAS O AGENTE FICOU PRESO EM TORNO DE UM REGIÃO CORRESPONDENTE A UM ÓTIMO LOCAL NA SUPERFÍCIE DE RECOMPENSA.	81
FIGURA 56 - (A)-(C): SUPERFÍCIES GERADAS COM LIMITE INFERIOR DA TAXA DE APRENDIZAGEM IGUAL A 0.01 APÓS 1 TREINAMENTO. (D)-(F) TRAJETÓRIAS OBTIDAS A PARTIR DAS SUPERFÍCIES.	82
FIGURA 57 - AMBIENTE SIMPLES COM OBSTÁCULO EM FORMA DE U. O ESTADO INICIAL DO AGENTE ESTÁ MARCADO COM “O” E O ESTADO META COM “X”.	83
FIGURA 58 - CAMPOS VETORIAIS OBTIDOS A PARTIR DO CÁLCULO DO GRADIENTE DA SUPERFÍCIE DE PENALIZAÇÃO EM INSTANTES DIFERENTES DA EXPLORAÇÃO DO AGENTE.	84
FIGURA 59 - GRÁFICO DO NÚMERO DE COLISÕES OCORRIDAS A CADA TREINAMENTO.	85
FIGURA 60 - (A) TRAJETÓRIAS INICIALMENTE APRENDIDAS. (B) TRAJETÓRIAS DESENVOLVIDAS PELO AGENTE PARA CONTORNAR OS NOVOS OBSTÁCULOS.	86
FIGURA 61 - (A) TRAJETÓRIAS INICIALMENTE APRENDIDAS. (B) TRAJETÓRIAS DESENVOLVIDAS PELO AGENTE PARA CONTORNAR OS OBSTÁCULOS TEMPORÁRIOS.	87
FIGURA I-1 - ESQUEMA GERAL DE UM PDM (SINGH, 1994).	IV
FIGURA II-1 - PRIMEIRO ALGORITMO IMPLEMENTADO.	VI
FIGURA II-2 - TRAJETÓRIA EM <i>LOOP</i> . O PONTO MARCADO COM UM ‘X’ É O ESTADO META. O PEQUENO CÍRCULO DENOTA O ESTADO CENTRAL CUJO VALOR ESTÁ MARCADO EM CINZA NA TABELA II-1. .	VI
FIGURA II-3 - CONFIGURAÇÃO DE TREINAMENTO PARA A QUAL O AGENTE FICOU PRESO. (A) REGIÃO OBSERVADA DO AMBIENTE. (B) VALORES QUANTIZADOS DA AVALIAÇÃO DOS ESTADOS DA REGIÃO EM DESTAQUE.	VII

LISTA DE TABELAS

TABELA 1 – PRINCIPAIS CARACTERÍSTICAS ENCONTRADAS NO AGENTE	7
TABELA 2 – EXEMPLO DE VALORES PARA O AGENTE.	35
TABELA 3 - VALORES DOS <i>ENCODERS</i> PARA OS DESLOCAMENTOS DE TRANSLAÇÃO E ROTAÇÃO DO ROBÔ.	60
TABELA II-1 - AVALIAÇÕES V_E+V_I PARA OS ESTADOS AO REDOR DO ESTADO ASSINALADO NA FIGURA II- 2.	VII

LISTA DE SIGLAS E ABREVIATURAS

AR	-	Aprendizagem por Reforço
CAR	-	Componente de Aprendizagem por Reforço
CHA	-	Crítico Heurístico Adaptativo
CR	-	Conditioned Response
CS	-	Conditioned Stimulus
DC	-	Direct Current
GUI	-	Graphical User Interface
IA	-	Inteligência Artificial
IfR	-	Institute of Robotics
MC	-	Monte Carlo
PARDM	-	Problema de Aprendizagem por Reforço Direcionado à Meta
PC	-	Personal Computer
PD	-	Programação Dinâmica
PDM	-	Processo de Decisão Markoviano
PEDM	-	Problema de Exploração Direcionada à Meta
SRI	-	Stanford Research Institute
STRIPS	-	STANford Research Institute Problem Solver
TD	-	Temporal Difference
UCR	-	UnConditioned Response
UCS	-	UnConditioned Stimulus

LISTA DE SÍMBOLOS

A	- conjunto finito das ações do agente
F	- conjunto finito dos estados livres
G	- conjunto finito dos estados meta
O	- conjunto finito dos estados com obstáculo
S	- conjunto finito dos estados do ambiente
$\mathcal{R}$	- conjunto dos números reais
a_t	- ação realizada no instante t
s_t	- estado do ambiente no instante t ;
$r_t, r_t(s_t, a_t, s_{t+1})$	- retorno do ambiente no instante t , este depende de s_t , a_t e s_{t+1}
$\mathbf{r}_t$	- acumulado dos retornos que se seguem ao instante t .
$P^a_{s_t s_{t+1}}$	- probabilidade de transição do estado s_t para o estado s_{t+1} ao se realizar a ação a
$R^a_{s_t s_{t+1}}$	- premiação imediata esperada na transição de s_t para s_{t+1} ao se realizar a ação a
π	- política do agente
π^*	- política ótima do agente
$V^\pi(s)$	- avaliação do estado s ao se seguir a política π
$V^*(s)$	- avaliação do estado s ao se seguir a política ótima
$Q^\pi(s, a)$	- avaliação da ação a no estado s ao se seguir a política π
$Q^*(s, a)$	- avaliação da ação a no estado s ao se seguir a política ótima
$V, V(s), V_k(s)$	- estimativa da avaliação do estado s
$Q, Q(s, a), Q_k(s, a)$	- estimativa da avaliação da ação a executada no estado s
$\Delta V(s)$	- atualização da estimativa da avaliação do estado s
$\Delta Q(s, a)$	- atualização da estimativa da avaliação da ação a executada no estado s
$k(s)$	- número de visitas ao estado s
γ	- fator de desconto
α	- taxa de aprendizagem

RESUMO

BRAGA, A. P. S. (1998). Um agente autônomo baseado em aprendizagem por reforço direcionado à meta. São Carlos, 1998. 104p. Dissertação (mestrado) – Escola de Engenharia de São Carlos, Universidade de São Paulo.

Uma meta procurada em inteligência artificial (IA) é o desenvolvimento de mecanismos inteligentes capazes de cumprir com objetivos preestabelecidos, de forma totalmente independente, em ambientes dinâmicos e complexos. Uma recente vertente das pesquisas em IA, os agentes autônomos, vem conseguindo resultados cada vez mais promissores para o cumprimento desta meta. A motivação deste trabalho é a proposição e implementação de um agente que aprenda a executar tarefas, sem a interferência de um tutor, em um ambiente não estruturado. A tarefa prática proposta para testar o agente é a navegação de um robô móvel em ambientes com diferentes configurações, e cujas estruturas são inicialmente desconhecidas pelo agente.

O paradigma de aprendizagem por reforço, através de variações dos métodos de diferença temporal, foi utilizado para implementar o agente descrito nesta pesquisa. O resultado final obtido foi um agente autônomo que utiliza um algoritmo simples para desempenhar propriedades como: aprendizagem a partir de *tabula rasa*, aprendizagem incremental, planejamento deliberativo, comportamento reativo, capacidade de melhoria do desempenho e habilidade para gerenciar múltiplos objetivos. O agente proposto também apresenta um desempenho promissor em ambientes cuja estrutura se altera com o tempo, porém diante de certas situações seus comportamentos em tais ambientes tendem a se tornar inconsistentes.

Palavras-chave: Aprendizagem por reforço, agente autônomo, inteligência artificial, planejamento, comportamento reativo, navegação de robô móvel, problema de aprendizagem por reforço direcionada à meta, taxa de aprendizagem variável.

ABSTRACT

BRAGA, A. P. S. (1998). An autonomous agent based on goal-directed reinforcement learning. São Carlos, 1998. 104p. Dissertação (mestrado) – Escola de Engenharia de São Carlos, Universidade de São Paulo.

One of the current goals of research in Artificial Intelligence is the proposition of intelligent entities that are able to reach a particular target in a dynamic and complex environment without help of a tutor. This objective has been becoming reality through the propositions of the autonomous agents. Thus, the main motivation of this work is to propose and implement an autonomous agent that can match the mentioned goals. This agent, a mobile robot, has to navigate in environments which are initially unknown and may have different structures. The agent learns through one of the main reinforcement learning strategies: temporal difference. The proposed autonomous employs a simple learning mechanisms with the following features: learns incrementally from *tabula rasa*, executes deliberative and reactive planning, improves its performance through interactions with the environment, and manages multiple objectives. The agent presented promising results when moving in a dynamic environment. However, there are situations in which the agent do not follow this last property.

Keyword: Reinforcement learning, autonomous agent, artificial intelligence, planning, reactive behavior, mobile robot navigation, goal-directed reinforcement learning problem, variable learning rate.

CAPÍTULO 1

Introdução

O estudo de agentes autônomos é uma linha de pesquisa em inteligência artificial (IA) que vem despertando o interesse de muitos pesquisadores por propor soluções alternativas mais flexíveis que as fornecidas pela IA tradicional (Russel e Norvig, 1995). Um agente autônomo é um sistema que tenta realizar um conjunto de metas, em um ambiente complexo e dinâmico, estabelecendo por si só quais decisões devem ser tomadas, a cada situação encontrada, para satisfazer estas metas. Assim, projetar um agente autônomo é propor um mecanismo capaz de gerar sua própria estrutura de conhecimento que o permita solucionar o problema considerado. Uma possibilidade para a criação de sistemas capazes de trabalhar em ambientes dinâmicos é a adoção de estruturas que codifiquem o conhecimento igualmente dinâmicas que guiem as ações destes sistemas. Por isso, a capacidade de aprendizagem está associada muitas vezes ao conceito de agente autônomo.

O foco deste projeto está na criação de um sistema capaz de aprender autonomamente por tentativa e erro as regras básicas necessárias para o cumprimento de tarefas em um ambiente parcialmente conhecido e/ou inicialmente desconhecido, realizando o planejamento e a execução de ações que não sejam apenas pré-definidas. Neste caso, as informações do agente sobre o ambiente de trabalho são obtidas unicamente através de sensores conforme a Figura 1.

Figura 1 - Diagrama esquemático geral do agente e sua interação com o ambiente de trabalho.

Através dos dados sensoriais que descrevem o mundo a sua volta, e sua situação com relação a esse mundo, o agente deve ter condições de planejar, resolvendo problemas simples e se auto-adaptando quando for necessário, mesmo diante de situações não vistas anteriormente. A solução do problema proposto contribuirá para a área de inteligência artificial na medida em que desenvolverá um algoritmo auto-adaptável, capaz de compatibilizar planejamento global com planejamento reativo sem a necessidade de um modelo prévio do mundo sob o qual esteja atuando.

Nesta pesquisa, optou-se por empregar as técnicas de aprendizagem por reforço (AR) para manipular a informação sensorial e realizar o aprendizado do agente. O *framework* adotado por este paradigma de aprendizagem para representar a interação agente-ambiente incorpora características de inteligência artificial (IA) e da teoria de controle (Sutton, 1997; Kaelbling *et al.*, 1996; Dean, 1994; Singh, 1994): os processos de decisão markovianos (PDM), *Markovian Decision Processes*. Tal estrutura admite que o ambiente seja descrito por uma relação estocástica entre suas entradas (as ações de um agente) e suas saídas (seu estado atual e um sinal de reforço).

A tarefa teste escolhida para o agente é a navegação autônoma de um robô móvel que deve ser aprendida *on-line* em um ambiente inicialmente desconhecido. O agente tem o propósito de trabalhar em ambientes fechados que apresentem tanto obstáculos fixos quanto dinâmicos utilizando apenas a informação local fornecida por seus sensores e o conhecimento parcial acumulado durante a interação com o mundo. Os cenários utilizados para a fase de testes do agente buscam ser reproduções dos problemas encontrados em situações reais envolvendo configurações dos obstáculos com diferentes níveis de complexidade.

1.1 Desenvolvimento da Pesquisa

Neste trabalho, o algoritmo do agente aprende a realizar a tarefa de navegação obedecendo as seguintes características:

- Habilidade para operar em ambientes inicialmente desconhecidos;
- Capacidade de lidar com ambientes dinâmicos;
- Habilidade para gerenciar múltiplos objetivos;

- Capacidade para aprendizagem incremental;
- Capacidade de melhorar soluções iniciais;
- Simplicidade do mecanismo.

O agente proposto é capaz de realizar planejamento deliberativo (Winston, 1992; Russel e Norvig, 1995) e reativo (Brooks, 1986) em um único mecanismo treinado através de métodos de diferença temporal (Sutton, 1988). O conhecimento do agente sobre o ambiente é codificado em duas estruturas distintas que contribuem para cada tipo de planejamento. A primeira dessas estruturas é denominada superfície de recompensa. Nesta, são armazenadas avaliações da contribuição de cada estado para o cumprimento da tarefa. A segunda estrutura é a superfície de penalização que armazena informações sobre os estados que devem ser evitados durante a execução da tarefa, e que no caso da navegação correspondem aos estados nos quais se encontram obstáculos que impeçam a movimentação do robô móvel. Da composição das informações destas duas superfícies emergem comportamentos que levam o robô a se aproximar cada vez mais do estado meta e a evitar regiões próximas a obstáculos.

O funcionamento do agente é dividida em duas fases: aprendizagem ou treinamento, e teste ou operação. A fase de treinamento é realizada considerando os obstáculos estáticos e um alvo único. Nesta fase inicial, o agente aprende a mapear seu ambiente de trabalho. Na fase de teste, diversos pontos de partida (treinados e não-treinados) são tomados e obstáculos não-treinados (dinâmicos) com formatos diversos são acrescentados ao ambientes. Esta divisão de fases de funcionamento do agente foi adotada para uma melhor apresentação dos experimentos realizados com o agente proposto presentes no Capítulo 6.

1.2 Organização da Dissertação

Os capítulos deste trabalho obedecem a organização a seguir. O Capítulo 2 apresenta o conceito de agente autônomo, especificando o tipo de agente com o qual se trabalhará. Neste mesmo capítulo será descrita a tarefa que servirá de teste para o agente sendo comentadas as características desejadas do agente. O Capítulo 3 discute alguns trabalhos anteriores que são relevantes para o tema desta dissertação, buscando explicitar as características de cada uma dessas soluções que são desejáveis para o agente e justificando a escolha do paradigma de aprendizagem por reforço (AR) para o funcionamento do agente proposto. O Capítulo 4 apresenta uma revisão de

aprendizagem por reforço na qual se enfatiza os métodos de diferença temporal. No Capítulo 5, o agente proposto é descrito sendo explicados o seu funcionamento, as estruturas de armazenamento de informação adotadas bem como algumas decisões tomadas para sua implementação. Os resultados obtidos nos testes com o agente em situações diversas são apresentados e comentados no Capítulo 6. O Capítulo 7 encerra este trabalho apresentando as conclusões obtidas e apontando caminhos promissores para trabalhos futuros que melhorem os resultados conseguidos nesta pesquisa.

CAPÍTULO 2

O Agente e sua Navegação

O interesse do presente trabalho está na criação de um algoritmo que implemente um mecanismo que possua a habilidade de escolher ações, objetivando alcançar uma meta pré-definida, sem a interferência de um tutor externo e sem nenhum conhecimento inicial, ou com conhecimento apenas parcial, do ambiente em que está atuando. Este algoritmo deve poder melhorar uma estratégia determinada e evitar obstáculos em seu caminho para a meta. Tal mecanismo encaixa-se no conceito de agente autônomo (Maes, 1990) apresentado na Seção 2.1. Este conceito é bastante abrangente, sendo o escopo desta dissertação limitada aos agentes robóticos. Em particular, aos robôs móveis que realizam a tarefa de navegação autônoma (Seção 2.2) em ambientes fechados, dinâmicos e inicialmente desconhecidos. Além das características inerentes dos agentes autônomos, a tarefa colocada impõe a necessidade de outras características que são listadas e discutidas na Seção 2.3 e que servem para orientar a escolha da abordagem para a implementação do agente.

2.1 O Agente Autônomo

O conceito de agente autônomo se presta para analisar um sistema (Russel e Norvig, 1995) e pode ser entendido, de maneira geral, como qualquer entidade que possui capacidade de perceber e agir sobre o seu ambiente (Figura 2). Entidades tão diferentes como um ser humano, uma bactéria ou um robô móvel são exemplos de agentes que atuam no mundo real. Agentes também podem ter uma existência apenas virtual como os agentes de software (Brustoloni, 1991) que vivem em sistemas operacionais, bancos de dados ou redes de computadores, e os agentes de vida artificial que habitam ambientes artificiais como telas ou memórias de computadores (Langton, 1989). Todos esses exemplos, reúnem propriedades comuns: os agentes estão imersos em um meio sobre o qual conseguem sensoriar e atuar

independentemente de qualquer outra entidade. As ações dos agentes buscam satisfazer às suas próprias necessidades, como em humanos e animais, ou alcançar metas pré-estabelecidas, como em agentes de software, e são tomadas de forma a alterar o ambiente de trabalho. Outra característica comum aos agentes é que eles agem continuamente durante o período de tempo no qual realizam uma dada tarefa. Por exemplo, um agente de software, uma vez solicitado, tipicamente realiza seus procedimentos até que o próprio decida finalizar seu funcionamento; um robô móvel move-se no terreno até alcançar sua posição meta; um animal caça seu alimento até saciar sua fome.

Figura 2 - Um agente genérico é capaz de perceber o ambiente em que está e gerar ações que afetem o estado deste ambiente.

Singh (1994) argumenta que para o agente estar apto a trabalhar autonomamente com a incerteza presente em um ambiente complexo, este deve ser capaz de aprender. Agentes capazes de aprender podem prover soluções para novas tarefas de forma mais eficiente que agentes pré-programados pois a habilidade de aprender permite adaptação a mudanças no ambiente não previstas pelo projetista de um agente pré-programado. O mundo real é complexo, imprevisível, e dinâmico, dificultando a um projetista prever todas as circunstâncias que devem ser enfrentadas por um agente em uma interação contínua e de longo prazo com tal ambiente (Beer *et al.*, 1990).

Um agente que aprenda não apresenta necessariamente melhor desempenho que outro que não tenha essa capacidade (Kaelbling, 1990), porém, uma razão para construir agentes com habilidade para aprender é contornar a dificuldade em criar programas específicos para agentes que trabalhem em ambientes complexos e incertos permitindo ao agente experimentar e aprender a partir de seus próprios sensores e atuadores. Mesmo que o programador especifique completamente o programa para um agente operar em um determinado ambiente, este agente teria que ser

completamente reprogramado para movê-lo para um novo ambiente. Com a aprendizagem, o agente torna-se capaz de adaptar suas estratégias de percepção e ação para o mundo em que se encontra, o que o torna flexível o suficiente para atuar em uma variedade de ambientes. Em resumo, a habilidade de aprender permite ao agente: lidar com mudanças não previstas do ambiente, atingir novos objetivos e melhorar seu desempenho no decorrer do tempo. A Tabela 1 apresenta as principais propriedades esperadas para um agente robótico autônomo.

Tabela 1 – Principais características encontradas no agente

Propriedade	Significado
Autonomia de funcionamento	Possuir o controle de suas próprias ações.
Continuidade de processamento	Ser um processo continuamente ativo.
Capacidade de aprendizagem	Modificar seu comportamento baseado em experiência.
Exibição de comportamento reativo	Responder a mudanças no ambiente detectadas por seu sensoramento.
Exibição de comportamento direcionado à meta	Selecionar ações considerando como estas contribuem para atingir uma dada meta.

Franklin e Grasser (1997) apresentam três grandes classes de agentes autônomos: os biológicos, nos quais estão incluídos todos os seres considerados vivos, os robóticos, os quais são entidades com componentes mecânicas projetadas para executar alguma tarefa, e os computacionais, os quais são *softwares* que possuem as características da Tabela 1 (Figura 3). Os agentes robóticos são a classe de interesse nesta pesquisa, sendo de particular interesse os robôs móveis por serem um caso teste que atua com o mundo real e que têm aplicações práticas como a atuação em áreas desconhecidas que representem risco a operadores humanos (Gray e Caldwell, 1996), no auxílio a portadores de deficiência (Bugmann *et al.*, 1998) ou no transporte de materiais na indústria (Warnecke e Schuler, 1985), para citar alguns exemplos.

Figura 3 - Classificação dos agentes autônomos segundo Franklin e Grasser (1997).

A tarefa escolhida para o agente robótico desta dissertação é a navegação autônoma em um ambiente inicialmente desconhecido¹. Esta tarefa é um excelente cenário para a utilização das propriedades listadas na Tabela 1. Por não possuir informação inicial sobre seu ambiente de trabalho, o que permite sua aplicação a diversos ambientes, o agente deve realizar associações entre as informações que recebe e as consequências de suas ações (**aprendizagem**) de forma a conseguir julgar se uma ação é mais adequada que outra para se atingir seus objetivos a curto prazo (**comportamento reativo**) e suas metas a longo prazo (**planejamento direcionado à meta**). Essa tomada de decisões ocorre ininterruptamente (**continuidade do processamento**) até que os objetivos tenham sido alcançados, sem a interferência de um tutor (**autonomia de funcionamento**). A próxima seção descreve o problema de navegação a ser tratado nesta dissertação, situando onde o agente atuará.

2.2 A Navegação Autônoma de um Robô Móvel

O problema da navegação autônoma de um robô móvel em um mundo dinâmico e inicialmente desconhecido é uma tarefa não trivial (Thrun *et al.*, 1991) e consiste na atuação do agente em um ambiente que possui obstáculos estáticos e obstáculos dinâmicos decorrentes de mudanças no ambiente ou presença de outros agentes objetivando alcançar uma posição alvo pré-estabelecida. Segundo Leonard e Durrant-Whyte (1991), o problema de navegação pode ser particionado em três subproblemas:

1. **Descrição da situação do ambiente do robô:** Esta etapa serve de base para as duas etapas seguintes, uma vez que ela é responsável por fornecer o estado do ambiente. Este estado deve prover, de forma compacta para o agente, toda a informação que seja relevante para o estado futuro do ambiente. Em um robô móvel, esta etapa é cumprida pelo processamento dos dados sensorizados.
2. **Estabelecimento da meta do robô:** Esta fase trata da determinação do objetivo que o robô deve cumprir. Em alguns trabalhos, (Zelinsky, 1992; Xiao *et al.*, 1997) o estado meta do robô é explicitamente determinado, porém, há problemas em que

¹ Em geral, a complexidade deste problema reduz-se à proporção que o agente responsável pela geração da trajetória possui uma melhor descrição do ambiente. Visto desta forma, a situação inicial de total desconhecimento da estrutura do ambiente aumenta, em muito, a complexidade do problema, e esta é a situação inicial que o agente encontra neste trabalho.

não se tem conhecimento suficiente do ambiente para que seja explicitado qual é o estado meta (Zalama *et al.*, 1995). Nestes casos, costuma-se estabelecer alguma característica do estado desejado determinando-o implicitamente. Um exemplo de determinação implícita da meta é o caso de um robô móvel buscando uma posição que esteja marcada por uma fonte luminosa: o robô alcança o estado meta quando os sensores detectam a maior emissão luminosa, porém, antes de chegar a esse ponto não há conhecimento de onde ele esteja no espaço de estados.

3. Estabelecimento de um caminho que leve o robô da posição inicial à meta :

Esta é a etapa na qual é formulado o plano que guiará o robô no espaço de estados em direção ao objetivo determinado na etapa anterior. O interesse maior deste trabalho está nesta etapa que pode ser desdobrada em duas sub-etapas: geração de trajetória e controle de trajetória. Devido às mudanças do ambiente ou ao desconhecimento do agente, um plano gerado pode ter que ser refeito de forma a considerar novas informações obtidas do ambiente. Por isso, esta divisão é feita de forma a distinguir entre a geração da sequência de ações necessárias para atingir a meta (geração de trajetória) e a verificação se o plano gerado está sendo adequado ao objetivo esperado (controle de trajetória).

O robô deve aprender a partir de informação local, dentro do alcance de seus sensores, e decidir *on-line* qual ação, em cada estado, deve ser tomada de forma a que o robô seja levado à configuração desejada. Para tal, este agente receberá alguma indicação da meta a ser alcançada, e estabelecerá uma política a ser seguida para a realização da tarefa desejada, melhorando esta política caso se faça necessário. Com base em Gray e Caldwell (1996), este sistema pode ser entendido como possuindo dois módulos principais de processamento: Módulo de Planejamento do Sensoriamento e Módulo de Planejamento das Ações de Controle.

- ♦ **Módulo de Planejamento do Sensoriamento:** A descrição do estado do mundo pode necessitar de um refinamento para que se determine, dentre um conjunto de possíveis ações, a ação de controle mais adequada. Por motivos como limitação no tempo de processamento da informação sensorial, imprecisão dos sensores ou restrições no campo de alcance dos sensores disponíveis, faz-se necessária uma estratégia que permita otimizar o uso dos sensores disponíveis no sentido de obter a melhor descrição do ambiente com o menor custo computacional. A função do

módulo de planejamento do sensoriamento é justamente buscar essa otimização determinando quais sensores devem ser utilizados a cada instante ou buscando posicionar os sensores de forma a melhor capturar as características do ambiente.

- ♦ **Módulo de Planejamento das Ações de Controle:** O agente deve possuir um conjunto ações de controle. Assim, este módulo deve decidir qual, dentre as possíveis ações de controle, a ação de controle mais adequada para levar o sistema de seu estado atual para o estado objetivo. Este módulo constitui a implementação física da etapa de planejamento abrangendo as duas divisões já mencionadas (geração de planos e controle do planos).

Como todo o planejamento deve ser *on-line*, as restrições de tempo para as tomadas de decisão são considerações importantes a serem levadas em conta. Desta forma, a atuação do ciclo de sensoriamento, onde se busca a melhor estimativa possível do estado do ambiente, deve ocorrer em um período limitado de tempo. De forma resumida, o agente deverá realizar os seguintes passos (Figura 4):

- 1 - Leitura e integração dos dados sensoriais para a criação de uma descrição do estado do ambiente;
- 2 - Avaliação da ação externa a ser tomada (planejamento);
- 3 - a) Caso o tempo para tomada de decisão termine, a ação externa escolhida pelo planejamento de ações de controle deve ser realizada;
- 3 - b) Caso contrário, o planejamento do sensoriamento deve buscar uma ação de sensoriamento que melhore a estimativa do estado do mundo;
- 3 - b.1) Caso encontre a ação de sensoriamento, esta é realizada e o ciclo de sensoriamento é repetido.

Figura 4 - Diagrama geral do sistema de controle de navegação baseado em Gray e Caldwell (1996).

Visto o funcionamento geral do sistema de controle da navegação (Gray e Caldwell, 1996), o problema principal desta dissertação é estabelecido como sendo o desenvolvimento do módulo de planejamento das ações de controle. A próxima seção apresentará as características desejadas para este módulo que serão incorporadas ao agente robótico desta pesquisa.

2.3 Características presentes no agente desejado

Há diversas soluções para a tarefa de planejamento de ações tratada nesta dissertação, sendo as principais apresentadas no Capítulo 3. As condições de operação do agente robótico para o problema aqui tratado tornam inviáveis a aplicação de muitas dessas soluções. No caso em estudo, por exemplo, as soluções que empregam a pré-definição das ações do agente são totalmente descartadas. As principais características que devem estar presentes no agente são colocadas a seguir e servem como critérios para a escolha da abordagem a ser adotada para solução do problema de planejamento da navegação:

- ♦ **Habilidade para operar em ambientes inicialmente desconhecidos:** Busca-se um agente capaz de trabalhar em ambientes diferentes. Assim, um modelo geral não pode ser criado para o mundo em que o agente atuará, tornando necessário que o robô construa do zero o seu conhecimento. Operar a partir de *tabula rasa* envolve a dificuldade de capacitar o agente a executar uma exploração do espaço de estados segundo uma sistemática que permita, no menor número possível de

interações com o ambiente, extrair o máximo de informação sobre o meio em que está atuando (Koenig e Simmons, 1996). Para permitir a exploração do espaço de estados, geralmente recorre-se à inclusão de uma componente aleatória. Esta componente, com o decorrer do funcionamento do agente, entretanto, passa a prejudicar as soluções encontradas por forçar modificações nas decisões em trechos das seqüências de ações que já não podem ser melhorados. Este problema é tratado na literatura como o compromisso entre exploração e repetição de uma estratégia (*exploration / exploitation tradeoff*) (Kaelbling *et al.*, 1996).

- ♦ **Capacidade de lidar com ambientes dinâmicos:** Ambientes reais devem ser considerados como inerentemente dinâmicos (Luo e Kay, 1995). Ruídos nos sensores e falhas nos atuadores são duas ocorrências constantes em implementações reais. Apenas estas características já justificariam a inclusão de incerteza em qualquer modelo real do mundo. Fatores como modificações do ambiente causadas pelo tempo ou a existência de outros agentes fortalecem a necessidade de se considerar que um ambiente real seja dinâmico. Para lidar com as mudanças do ambiente de trabalho, o agente robótico deve possuir um mecanismo flexível de tomada de decisões que possa se adaptar às alterações no meio em que está trabalhando.
- ♦ **Habilidade para gerenciar múltiplos objetivos:** Brooks (1986) coloca que no planejamento das ações de controle frequentemente o robô tem que trabalhar com múltiplos objetivos os quais podem ser conflitantes entre si. Um exemplo seria o caso em que o robô busca chegar a um ponto à sua frente tendo antes que desviar de obstáculos locais. Outro exemplo seria compatibilizar os objetivos de alcançar determinado ponto no ambiente, minimizar o tempo necessário para o percurso e reduzir o consumo de suas baterias. Em geral, a importância relativa dos objetivos é dependente do contexto. Assim, o agente deste problema deve responder às metas de maior prioridade (planejamento global), estando ainda disponível para servir às metas de nível de abstração inferior necessárias na evolução do sistema (comportamento reativo).
- ♦ **Capacidade para aprendizagem incremental:** Um dos pontos que orientará as modificações a serem realizadas no modelo escolhido é a garantia de que o

aprendizado ocorra incrementalmente. Esta característica é desejável pois, ao explorar o espaço de trabalho para aprender a mapear estados do mundo em ações de controle, o agente deve sofrer interferência mínima no restante do conhecimento já adquirido ao absorver uma nova relação estado-ação.

- ♦ **Capacidade de melhorar soluções iniciais:** O problema de navegação pode possuir múltiplas soluções e, ao obter um caminho que conecte um estado inicial ao estado meta, o agente não consegue necessariamente a melhor. Como o deslocamento do agente de um estado para o outro implica na utilização de recursos (disponibilidade energética, tempo de operação), deve-se buscar otimizar as trajetórias do robô para reduzir os gastos com os recursos envolvidos no funcionamento do robô.
- ♦ **Simplicidade do mecanismo:** Para poder ser aplicável em situações reais, a quantidade de processamento necessária para o funcionamento do agente é um fator decisivo. Assim, o modelo do agente deve ser matematicamente simples, dependendo de poucos cálculos. Além disso, soluções complexas tendem a ser pouco flexíveis, tornando-se específicas para cada ambiente de trabalho.

No próximo capítulo, ao serem apresentadas algumas das abordagens existentes para o planejamento de ações na navegação de um robô móvel, será possível ao leitor perceber que o cumprimento, isolado, de cada um dos critérios acima listados não representa grande dificuldade: para cada propriedade, uma das abordagens tradicionais se destaca das demais. Porém, estabelecer uma solução que satisfaça razoavelmente a todos os critérios implica em desenvolver uma nova solução que consiga unir em um único agente o melhor de cada abordagem.

2.4 Conclusão

Agente autônomo é um conceito que pode ser aplicado a entidades que atuam em um determinado ambiente buscando cumprir objetivos específicos, usando apenas seus próprios recursos, sem nenhuma intervenção externa. O estudo de tais agentes pode gerar ferramentas que permitam a automação de diversas atividades que, por envolverem processos cognitivos, são difíceis de serem realizadas por máquinas. Um

exemplo de uma atividade que para um animal é simples mas ainda é difícil de ser automatizada é a navegação em um ambiente inicialmente desconhecido. Este trabalho busca um agente autônomo que seja capaz de realizar a tarefa de navegação de um robô móvel. O algoritmo para implementar tal agente deve reunir as seguintes propriedades: habilidade para operar em ambientes inicialmente desconhecidos, capacidade de lidar com ambientes dinâmicos, habilidade para gerenciar múltiplos objetivos, capacidade para aprendizagem incremental, capacidade de melhorar soluções iniciais e simplicidade do mecanismo.

O funcionamento geral do agente é esboçado no diagrama da Figura 5, sendo os critérios listados utilizados para definir a maneira como serão realizados a geração e o controle dos planos pelo agente.

Figura 5 – Diagrama considerado para o funcionamento do agente.

Conforme a Figura 5, a tarefa que o agente deve realizar é descrita pela indicação do estado meta. De posse da indicação do estado que deve ser alcançado e de dados sobre o ambiente, fornecidos pelo sensoriamento, o planejamento das ações do agente é realizado criando estratégias para alcançar a meta (geração de planos) ou adaptando as estratégias às mudanças do ambiente (controle dos planos). Para permitir o acompanhamento da evolução dos planos gerados, o funcionamento do agente deve ocorrer em malha fechada, estando nos dados fornecidos pelo sensoriamento indicações de como as ações executadas modificaram o ambiente.

O próximo capítulo realiza uma breve revisão bibliográfica de algumas das abordagens empregadas em navegação autônoma, e com base nas técnicas existentes será proposto um modelo de agente.

CAPÍTULO 3

Abordagens para o Agente Autônomo

O problema da navegação de um robô móvel é tratado em diversos trabalhos de robótica (Kortenkamp *et al.*, 1998). Uma solução definitiva que reuna o melhor de cada abordagem ainda não foi encontrada, porém muitos avanços têm sido obtidos na área e um dos propósitos deste capítulo é fornecer ao leitor uma visão geral, de maneira sucinta, deste campo de pesquisa. A Seção 3.1 apresenta as abordagens mais comuns usadas para a solução da navegação autônoma. Destas abordagens, os sistemas que incorporam comportamento reativo e planejamento deliberativo são os que mais se adequam às características do agente desejado (Seção 2.3), sendo AR uma maneira promissora de implementar o agente ao considerar o planejamento das ações como um problema de aprendizagem por reforço direcionado à meta (PARDM), cuja definição é vista na Seção 3.2.

3.1 Abordagens para a Criação de um Agente Autônomo Robótico de Navegação

Basicamente, há duas grandes classes de soluções para a navegação autônoma de robôs móveis²: os sistemas planejadores e os sistemas reativos. Cada abordagem tem seus pontos fortes nas deficiências da outra. Assim, a tendência atual dos projetistas é a criação de uma terceira classe de soluções, os sistemas deliberativos-reativos ou híbridos (Arkin, 1990), que possuem em um mecanismo único vantagens das duas outras abordagens. As duas próximas subseções apresentam as idéias principais dos sistemas deliberativos e reativos, e a terceira comenta sobre os sistemas

² As soluções com trajetórias pré-estabelecidas não são citadas por fugirem do escopo desta pesquisa ao não proporcionarem as características desejadas do agente (Seção 2.3).

híbridos e a metodologia escolhida para implementar um agente autônomo de navegação híbrido.

3.1.1 Sistemas Planejadores (Deliberativos)

Esta abordagem, de maneira geral, decompõe o planejamento em três etapas (Kortenkamp *et al.*, 1998; Brooks, 1986): sensoriamento, planejamento e execução. A função do sensoriamento é traduzir os dados sensoriais para um modelo do mundo. A partir deste modelo do mundo e uma meta, o planejamento gera uma sequência de ações que levam o agente a atingir a meta estabelecida. Criado o plano, a etapa de execução realiza as ações nele prescritas. A Figura 6 esboça um diagrama do funcionamento dos planejadores deliberativos.

Figura 6 - Arquitetura geral dos sistemas deliberativos.

Uma das características da arquitetura dos sistemas deliberativos é o caráter unidirecional do fluxo de informações. A informação flui dos sensores para os atuadores e nunca em sentido contrário. Este funcionamento em malha aberta é uma consequência da ligação desta abordagem com as técnicas de inteligência artificial (IA). As pesquisas em sistemas deliberativos, desde de seu início, são muitas vezes vistas como uma aplicação de duas técnicas padrões de IA (Winston, 1992; Luger e Stubblefield, 1993): algoritmos de busca, como o algoritmo A* (Rich e Knight, 1994), e os provadores de teorema (Russel e Norvig, 1995). Os algoritmos de busca realizam o planejamento gerando sequências de transformações (operadores) para um ambiente a partir de um estado inicial. A cada passo, há a escolha de qual transformação utilizar e, enquanto a solução final não é obtida, o algoritmo retorna ou avança na árvore de busca conforme os estados estejam se afastando ou aproximando da solução. No planejamento por provadores de teoremas, o comportamento inteligente é conseguido utilizando lógica matemática e descrevendo a tarefa a ser realizada por predicativos.

3.1.1.1 Exemplo desta abordagem

Um exemplo de sistema deliberativo é o STRIPS (*Stanford Research Institute Problem Solver*) (Fikes e Nilsson, 1993) Este planejador foi utilizado no robô móvel Shakey (Fikes *et al.*, 1972) desenvolvido no SRI (*Stanford Research Institute*) (Figura 7.a).

Figura 7 - (a) Aparência do robô Shakey. (b) Ambiente de teste do robô móvel

Neste método, o agente faz uso de uma única lista que contém os objetivos e os operadores que foram propostos para satisfazer aqueles objetivos. O solucionador de problemas também depende de uma base de dados que descreve a situação corrente de um conjunto de operadores descritos na forma de listas SE, RETIRE e ADICIONE (Fikes e Nilsson, 1971):

- ♦ SE: Relaciona as declarações que devem ser verdadeiras para que seja possível aplicar o operador.
- ♦ ADICIONE: Inclui as declarações que se tornam verdadeiras após a aplicação do operador.
- ♦ RETIRE: Exclui as declarações que deixam de ser verdadeiras após a aplicação do operador.

Os operadores representados por estas listas são as ações que o robô pode realizar. A Figura 8 exibe a descrição de dois operadores com base em dois tipos de declaração:

- $Na_Sala(x, y)$ - indica que o objeto x (robô ou caixa) está na sala y .
- $Conecta(w, k, z)$ - indica que a porta w liga a sala k à sala z .

Figura 8 - Operadores que descrevem as ações do robô de: (a) atravessar uma porta e (b) arrastar um objeto através de uma porta.

Um exemplo simples do funcionamento do planejador STRIPS no controle do robô Shakey é apresentado na Figura 9. As configurações inicial e final desejadas do robô e do ambiente são exibidas na Figura 9.a, sendo levar a Caixa1 situada na sala R2 para a sala R3 a tarefa a ser executada pelo robô.

Figura 9 - (a) Estados inicial e final no mundo dos blocos. (b) Plano gerado pelo planejamento se-adicione-delete.

A escolha dos operadores da sequência apresentada na Figura 9.b é realizada por técnicas de IA como, por exemplo, os provadores de teoremas (Green, 1969). Neste caso, o estado meta é estabelecido por uma declaração como em (1) e busca-se provar a contradição da sua negação.

$$\text{Na_Sala}(\text{Caixa1}, R3) \wedge \text{Na_Sala}(\text{Robô}, R3) \quad (1)$$

Provada a contradição, o plano é gerado ao se adicionar os operadores na ordem inversa de sua aparição na prova do teorema.

3.1.1.2 Limitações

Os planejadores deliberativos são interessantes por selecionarem as ações a serem executadas pelo agente de forma direcionada à meta³. Entretanto, um ponto fraco do planejamento deliberativo é sua necessidade de um modelo do mundo para seu funcionamento: um modelo inadequado leva o agente a supor conseqüências erradas para suas ações, tornando inválidas suas decisões para o mundo real. Como estes sistemas costumam operar em malha aberta, baseando-se em um modelo determinístico do mundo, as falhas podem não ser detectadas para futura correção o que torna pouco recomendável a utilização de agentes que seguem esta abordagem em ambientes dinâmicos. Outro ponto a contar contra os sistemas puramente deliberativos é o tempo necessário para a escolha de suas ações: somente após ter sido gerado todo o plano é que uma ação é executada e, como a geração de um plano é essencialmente um problema de busca (Hendler *et al.*, 1990), esta busca pelo espaço de estados pode chegar a um crescimento exponencial das interações entre os estados (Chapman, 1987). Estes, e outros, problemas com os sistemas deliberativos fizeram com que as pesquisas em robótica em meados da década de 80 buscassem caminhos alternativos para a implementação de agentes autônomos, surgindo com isso as soluções que empregavam planejamento reativo.

3.1.2 Sistemas Reativos

Sistemas reativos são um progresso relativamente recente em robótica que surgiu a partir da insatisfação com os métodos puramente deliberativos e uma conscientização crescente da importância de observar sistemas biológicos como a

³ O plano é criado em função da meta a ser atingida. Conforme a complexidade do problema, a meta pode ser alcançada dividindo o problema em submetas e compondo suas soluções.

base para a construção de comportamentos inteligentes (Brooks, 1986; Beer *et al.*, 1990). Robôs reativos, também chamados de robôs baseados em comportamento, foram concebidos para reagir a estímulos do ambiente utilizando uma coleção de comportamentos primitivos de baixo nível de abstração. Comportamentos físicos complexos emergem através da interação de um conjunto de comportamentos simples e da complexidade inerente ao ambiente em que o robô se encontra (Brooks, 1990).

Algumas características que identificam um sistema de controle robótico puramente reativo são (Arkin, 1995):

1. Os comportamentos são blocos básicos de construção: Um comportamento nestes sistemas é geralmente um simples par sensor-motor, onde a atividade sensorial fornece a informação necessária para a resposta reativa motora de baixo nível (por exemplo, desvio de obstáculos, fuga de predadores, atração do agente pela meta).
2. Uma representação abstrata do conhecimento é evitada: A criação e manutenção de um modelo preciso do mundo é um processo propenso a erro e que consome tempo. Sistemas puramente reativos não mantêm modelos do mundo, reagindo diretamente aos estímulos do ambiente, no lugar disso. Esta propriedade é particularmente útil em ambientes extremamente dinâmicos e perigosos, onde o ambiente é imprevisível e potencialmente hostil.
3. Modelos de comportamento animal são geralmente usados como base para esses sistemas: Modelos oriundos de neurociência, psicologia cognitiva e etologia⁴ são utilizados para capturar a natureza dos comportamentos que são necessários para um robô interagir de maneira segura com um mundo hostil.
4. Resultados robóticos satisfatórios têm sido obtidos: Estas técnicas têm sido aplicadas a uma larga variedade de robôs (Kortenkamp *et al.*, 1998). Todos esses sistemas são extremamente modulares e podem ser construídos incrementalmente, adicionando novos comportamentos ao repertório já existente. Por uma perspectiva de engenharia, isto é bastante desejável pois facilita o crescimento e a aplicação de software e hardware já existente a novos domínios.

⁴ Etologia é o estudo do comportamento dos animais em seu *habitat* natural com a crença de que o comportamento é, em grande parte, resultado de uma resposta inata do animal a estímulos do ambiente. Os interesses da etologia são de natureza biológica; em adição à causa do comportamento, havendo interesse em determinar qual função é provida ao animal e suas conseqüências evolucionárias (Anderson e Donath, 1990).

A *subsumption architecture* (Brooks, 1986) é talvez o primeiro e mais conhecido representante da navegação reativa. Esta abordagem exprime bem a idéia central dos sistemas reativos. Nela a tarefa é decomposta em camadas horizontais onde cada camada possui um comportamento independente das demais (Figura 10). Da composição dos comportamentos gerados por cada camada emergem comportamentos complexos que se adaptam à complexidade do ambiente de trabalho com um baixo custo computacional.

Figura 10 - Sistema reativo e suas camadas.

Nestas camadas são implementados comportamentos como a exploração do ambiente, a construção de mapas, a identificação de objetos, a busca pela meta e o desvio de obstáculos. Estes comportamentos podem se tornar conflitantes, e para evitar que eles se anulem, estes possuem níveis de hierarquia (Figura 10). Cada camada da *subsumption architecture* define uma classe de comportamentos válidos que estabelecem tanto mais restrições na escolha das ações quanto maior for a camada.

A camada 0 é a base de toda a arquitetura e, geralmente, trata do desvio de obstáculos, que é uma das principais preocupações de um agente móvel que irá interagir com o mundo real. Há, por exemplo, de forma geral, três metodologias principais que realizam o desvio dos obstáculos (Borenstein e Koren, 1991): a detecção de bordas (*edge-detection*) (Elfes, 1987; Flynn, 1988), que busca a descoberta das bordas do ambiente para a determinação de caminhos livres por onde o

robô possa navegar; as grades de certeza (*certainty grids*) (Moravec, 1988), em que a presença de um obstáculo é representada pela probabilidade de sua existência e com base nesse valor é selecionada a ação que leva à célula com maior probabilidade de ser um trecho livre de obstáculos; e os métodos de campo potencial (*potencial field methods*) (Khatib, 1986; Rimon e Koditschek, 1992; Connolly e Grupen, 1993; Connolly, 1997), em que o robô comporta-se como uma partícula de carga elétrica positiva que está se movendo sob a influência de dois campos potenciais virtuais criados, respectivamente, pela configuração do alvo (campo atrator negativo) e dos obstáculos (campo repulsor positivo). Dada a modularidade dos sistemas reativos, qualquer metodologia de implementação de desvio de obstáculos pode ser utilizada na camada 0 independente da implementação das demais camadas.

3.1.2.1 Exemplo desta abordagem

Um exemplo de sistema reativo é o robô Herbert (Brooks, 1990). Este robô é equipado com 24 processadores, 30 sensores infravermelho, um manipulador com sensores, um sistema laser para detectar profundidade e uma câmera que coleta a cada segundo uma imagem de 256x32 pixels. O planejamento das ações do robô (Figura 11.a) é realizado seguindo a *subsumption architecture*. Foram utilizadas 15 camadas que não trocam informações entre si e emulam diferentes comportamentos em resposta aos dados fornecidos pelos sensores. A tarefa a ser executada é navegar em um ambiente de trabalho e coletar latas vazias que estejam sobre as mesas.

Figura 11 - (a) O robô Herbert. (b) Um exemplo de trajetória gerada pelo manipulador do robô (imagens obtidas na Home-Page: <http://www.pcrealm.net/~johuco/d-coord.html>).

A Figura 11.b exibe uma trajetória típica do manipulador utilizando alguns dos comportamentos programados em Herbert. Estes comportamentos permitem ao robô pegar uma lata em uma mesa. Os comportamentos ENGARDE⁵ e DESCEND localizam o topo da mesa. Em seguida, o comportamento BOUNCE assume o comando e ergue o manipulador que deslizará pela mesa sob influência de SURFACE. Antes que um contato com a lata ocorra, os sensores infravermelhos frontais do manipulador são ativados levando OVER a erguer o manipulador acima da lata. EXTEND fará com que a lata fique entre os dedos da garra e o comportamento GRAB, então, fechará a garra capturando a lata. Finalmente, HOIST e PARK trarão a lata para o robô.

A sequência apresentada não necessariamente ocorrerá sempre por ser uma composição de comportamentos que são ativados pelas leituras enviadas pelos sensores. Com isso, situações imprevistas são superadas pelos comportamentos que elas ativarem e a trajetória é programada como reação à situação do próprio ambiente.

3.1.2.2 Limitações

A adoção de um controle puramente reativo é uma solução interessante por permitir uma resposta rápida, não fazer uso de um modelo elaborado do mundo e ser modular. Porém, Millán (1996) lembra que sistemas puramente reativos apresentam duas limitações principais para a navegação autônoma em ambientes inicialmente desconhecidos. A primeira, e mais importante, é que controladores puramente reativos podem gerar trajetórias ineficientes por selecionarem a próxima ação simplesmente como uma função das leituras instantâneas dos sensores, o que não garante que o robô siga em direção à posição meta ou que ele evite uma ação que tenha consequências danosas, e a segunda é a dificuldade em programar tais sistemas pois a especificação de quais comportamentos devem ser implementados para a solução de uma tarefa é uma dificuldade no projeto de sistemas reativos que se assemelha à da criação de um modelo do mundo nos sistemas deliberativos.

3.1.3 Sistemas Deliberativos/Reativos.

Recentemente, arquiteturas robóticas híbridas (deliberativas/reativas) (Arkin, 1990, Payton *et al.*, 1990) que permitem a reconfiguração de sistemas reativos com base no conhecimento que o agente dispõe do ambiente têm adicionado considerável

⁵ As palavras apresentadas em letras maiúsculas e em inglês denominam comportamentos programados no robô.

flexibilidade aos sistemas puramente reativos. Os sistemas híbridos combinam o processamento em tempo real, que permite capacidade significativa de adaptação do robô ao ambiente, dos sistemas reativos com a capacidade de avaliar as consequências das ações executadas, que permite antever situações indesejáveis, dos sistemas deliberativos.

Conforme foi visto, as técnicas de IA, tradicionalmente, dão maior atenção a problemas bem definidos, nos quais se tem à disposição um modelo do ambiente, que tem levado ao desenvolvimento do planejamento seguindo técnicas heurísticas de busca. Quando não se dispõe de um modelo do mundo, ou há incerteza a respeito desse modelo, essas soluções são ineficientes. As imprecisões, inevitáveis, de um modelo do mundo real podem fazer com que o agente atue em um ambiente virtual diferente do real, levando-o a tomar ações errôneas. Por outro lado, a teoria de controle (Chen, 1984; Davis e Vinter, 1985) fornece uma base teórica para o tratamento de incertezas nos sistemas: muitas soluções têm sido desenvolvidas para manipular com a incerteza calculando soluções em malha fechada que descrevem as ações como uma função do estado do ambiente e possivelmente do tempo. O comportamento do agente deve ser tal que comporte a capacidade de aprendizado das técnicas de IA e a capacidade de tratar as incertezas do ambiente da teoria de controle.

Na teoria clássica de controle, o procedimento geral é a utilização de um sinal de referência que indique a saída que se deseja do sistema (neste caso, a posição do robô no ambiente). Considerando o erro entre a saída real e a desejada toma-se uma ação para reduzir o erro. Esta abordagem não se adequa ao problema aqui proposto por não haver uma trajetória pré-estabelecida que sirva de sinal de referência, a trajetória deve ser descoberta pelo agente. Uma outra abordagem seria buscar a codificação da trajetória na forma de parâmetros e utilizar controle adaptativo para regular esses parâmetros conforme o robô explorasse seu espaço de trabalho. As técnicas de controle adaptativo (Burghes e Graham, 1980; Stengel, 1986) estão relacionadas com algoritmos que melhoram o desempenho de suas decisões com a experiência (Kaelbling *et al.*, 1996). Esta é uma área madura e que trata com sistemas dinâmicos nos quais estados e ações são vetores e os sistemas dinâmicos são lineares ou localmente linearizáveis em torno de uma trajetória desejada. O ajuste dos parâmetros é realizado de forma a otimizar uma função de custo que, em geral, corresponde às penalidades quadráticas no desvio dos vetores desejados de estado e ação.

Sutton *et al.* (1992) argumenta que os métodos AR formam uma abordagem adequada para o controle adaptativo ótimo. Para o agente autônomo desta pesquisa, que deve ser capaz de atuar em um ambiente inicialmente desconhecido, o paradigma de AR fornece uma forma adequada de estabelecer uma função de custo para as suas ações.

3.1.3.1 Aprendizagem por Reforço: A Metodologia Adotada

Segundo Whitehead e Ballard (1991), o imediatismo na tomada de decisões coloca os sistemas que utilizam métodos de aprendizagem por reforço estreitamente próximos dos sistemas reativos (Brooks, 1986; Agre e Chapman, 1987; Georgeff e Lansky, 1987). Entretanto, diferentemente da vasta maioria dos sistemas reativos os métodos AR possuem a capacidade de se adaptar aprendendo as tarefas a serem executadas. Em sistemas reativos, o conhecimento para a tomada de decisões é codificado pelos projetistas de forma explícita (Brooks, 1986; Agre, 1988) ou através da especificação de modelos de mundo que eventualmente são compilados em um conjunto de regras reativas (Fikes *et al.*, 1972; Laird *et al.*, 1986). Os sistemas AR não precisam ter conhecimento pré-definido para a tomada de decisões. Eles aprendem estratégias de controle por tentativa-e-erro ao interagir com o mundo, recebendo realimentações na forma de sinais de reforço. Esta adaptabilidade atenua o trabalho de ter que fornecer um completo conhecimento do domínio de trabalho ao agente, uma vez que este é adquirido com a experiência. Isto também permite ao sistema se adaptar a mudanças circunstanciais e aprender novas tarefas. A abordagem de aprendizado por reforço se enquadra no raciocínio apresentado por Brooks (1986): “Um comportamento complexo não precisa necessariamente ser produto de um sistema de controle extremamente complexo. No lugar disso, um comportamento complexo deve ser reflexo da complexidade do ambiente”.

3.2 Problema de Aprendizagem por Reforço Direcionado à Meta

O problema apresentado na Seção 2.2 pode ser considerado um caso da classe de problemas que Koenig e Simmons (1996) definiram como problema de aprendizagem por reforço direcionado à meta (PARDM): o agente deve ser capaz de aprender um caminho, entre um estado inicial e o estado meta, no espaço de estados através de AR e melhorar esta solução inicial, procurando chegar a uma política

ótima, conforme seu treinamento transcorra. O PARDM pode ser considerado como tendo duas etapas:

- ♦ **Solução do problema de exploração direcionada à meta (PEDM):** Este estágio envolve a exploração do espaço de estados em busca da determinação de ao menos um caminho viável, não necessariamente o melhor, que ligue o estado inicial ao estado meta.
- ♦ **Melhoria da solução encontrada:** Esta etapa aproveita o conhecimento adquirido do ambiente durante a exploração do espaço de estados para melhorar as escolhas de trajetórias, buscando alcançar uma política ótima para o agente.

A solução do PEDM em um ambiente dinâmico e inicialmente, ou parcialmente, desconhecido é a etapa mais difícil do PARDM por levar o agente a operar com pouca informação sobre o ambiente. Nesta etapa, os comportamentos reativos do agente híbrido serão mais exigidos para permitir sua adaptação ao ambiente. Superada esta fase, o conhecimento acumulado durante a exploração permitirá ao agente antever as consequências de suas ações e, com isso, planejá-las de forma a evitar aquelas que levem a situações perigosas, buscando as que aproximem o agente do estado meta.

O Capítulo 4 apresenta os principais conceitos de AR e algumas das dificuldades em implementar agentes autônomos AR que atuem em ambientes reais. O agente a ser proposto busca reduzir essas limitações através de adaptações em algoritmos anteriormente propostos, adaptações essas que são apresentadas no Capítulo 5.

3.3 Conclusões

Uma das principais diferenças entre o planejamento deliberativo e o reativo é a abrangência de cada abordagem. O planejamento deliberativo trabalha com uma visão global da tarefa a ser executada: a elaboração do plano necessita de um modelo do mundo para que uma sequência de ações que leve o agente a alcançar a meta seja gerada. Diferentemente, o planejamento reativo possui visão local do problema: os comportamentos reativos são selecionados em função dos estados vizinhos ao estado atual do agente. O planejamento global (deliberativo) permite um maior direcionamento do agente à meta a ser alcançada, porém consome muito tempo de

processamento. O planejamento local (reativo) é mais robusto para tratar com as variações de um ambiente dinâmico, é rápido na seleção das ações, entretanto, por não considerar um modelo do ambiente, escolhe ações que não necessariamente aproximarão o agente de seu estado meta. A combinação dessas duas abordagens gera uma nova classe de soluções para o planejamento de ações, a classe dos sistemas híbridos, que incorpora as vantagens de ambas e reduz suas limitações.

A metodologia escolhida para a implementação do agente autônomo desta pesquisa é a aprendizagem por reforço por ser um paradigma que pode apresentar características deliberativas e reativas. O potencial das técnicas AR em robótica móvel já foi explorado em trabalhos anteriores, e problemas práticos comuns são encontrados nestas implementações. Estas limitações estão relacionadas diretamente com o funcionamento dos algoritmos padrões para AR (Capítulo 4). A redução de tais limitações é mais um critério a ser cumprido pelo agente a ser proposto, e a forma como este objetivo será alcançado é explicada na apresentação do agente feita no Capítulo 5.

CAPÍTULO 4

Aprendizagem por Reforço

Aprendizagem por reforço é uma metodologia de inteligência artificial em que o agente emprega tentativa e erro para aprender como realizar uma tarefa em seu ambiente de atuação. Este aprendizado ocorre quando o agente estabelece um mapeamento entre suas entradas (os estados do ambiente) e suas saídas (as ações realizadas). Tal mapeamento é realizado por influência de realimentações do ambiente, os sinais de reforço, que correspondem a medidas de adequação das ações tomadas conforme a contribuição das mesmas para a realização da meta desejada. Este mapeamento representa o comportamento do agente autônomo, e os conceitos gerais de como ele é construído neste paradigma são apresentados na Seção 4.1. Há diferentes implementações de algoritmos de AR (programação dinâmica, métodos de Monte Carlo, métodos de diferença temporal) que diferem por características como a existência ou não de um modelo do mundo ou a quantidade de informação necessária para gerar os mapeamentos. Entre estas implementações há uma classe que se destaca por ser independente de um modelamento do mundo, ter aprendizado incremental e trabalhar com pouca informação: são os métodos de diferença temporal. Esta abordagem é discutida em detalhes na Seção 4.2 pois será a metodologia escolhida neste trabalho para o agente. A Seção 4.3 apresenta algumas limitações comuns à utilização de AR em robôs móveis, limitações essas que serão parcialmente reduzidas pelo agente proposto, descrito no Capítulo 5.

4.1 Conceitos Gerais de Aprendizagem por Reforço

Em AR, para alcançar um objetivo estabelecido, o problema a ser tratado deve ser solucionado pelo conhecimento que o agente obtém ao interagir com o ambiente. Formalmente, a estrutura de AR trabalha com três conjuntos (Kaelbling *et al.*, 1996):

1. Um conjunto discreto de estados do ambiente, S ;
2. Um conjunto discreto das ações do agente, A ;
3. Um conjunto de valores escalares para os sinais de retorno do ambiente, $\mathcal{R}$.

Estes três conjuntos são utilizados por quatro componentes principais (Sutton e Barto, 1998):

1. **Uma política (π):** Função que mapeia estados do ambiente em ações ($\pi: S \rightarrow A$).
2. **Uma função de retorno (r):** Função que mapeia estados do ambiente em um valor real, o reforço ($r: S \rightarrow \mathcal{R}$).
3. **Uma função de avaliação (V ou Q):** Valor que estima um acumulado para o conjunto dos retornos entre um dado estado s e o estado final desejado. Esta função mapeia estados em valores reais ($V: S \rightarrow \mathcal{R}$) ou pares estado-ação em valores reais ($Q: S, A \rightarrow \mathcal{R}$).
4. **Um modelo do mundo⁶:** Este componente imita o comportamento do ambiente. Ele mapeia estados do ambiente em sinais de reforço.

Nos métodos AR, a dinâmica do ambiente é tratada como uma relação estocástica entre o estado do ambiente e a ação tomada pelo agente a cada instante de tempo, e este estado e o sinal de retorno do ambiente no instante posterior (Sutton e Barto, 1998). Tendo o ambiente propriedades markovianas⁷, o que raramente ocorre totalmente mas que pode ser aproximado por uma escolha adequada da representação do estado (Wyatt, 1997), o conhecimento do estado atual será suficiente para prever o estado futuro do ambiente, e este é tido como um Processo de Decisão Markoviano⁸ (PDM), detalhes no APÊNDICE I. A estrutura de PDM (Bellman, 1957) é bastante flexível, o que permite que seja aplicada a diversos problemas, e vem sendo

⁶ O modelo de mundo não está presente em todo agente de AR, alguns métodos o aprendem ou não fazem uso de um modelo. O caso mais geral é o de um agente que não tenha *a priori* um modelo perfeito do ambiente mas que utilize métodos de aprendizagem para construir um modelo mais elaborado.

⁷ Um processo é dito markoviano se toda a informação necessária para determinar seu comportamento futuro está contida na descrição do estado atual (Clarke e Disney, 1979).

⁸ Um problema de tomada de decisão em um ambiente com propriedades markovianas define um processo de decisão markoviano.

extensivamente estudada em IA como uma nova e adequada forma de solucionar problemas de planejamento (Sutton, 1997). Muito da atual teoria de aprendizagem por reforço é restrita aos PDMs finitos (um PDM finito é um PDM com um conjunto limitado de ações e estados). A maioria dos algoritmos de aprendizagem por reforço são baseados na estimativa da função de avaliação⁹ e buscam maximizar os reforços recebidos do ambiente pelo agente.

Dado que a interação entre agente e ambiente é definida como um PDM, esta se inicia quando o agente, conhecendo o estado (s_t) do ambiente e seu sinal de reforço (r_t) no instante de tempo t , produz uma ação (a_t) que atua no ambiente. Esta ação leva o ambiente a um novo estado (s_{t+1}) e a produzir um novo reforço (r_{t+1}), cujo acumulado (função de avaliação, V ou Q) o agente deverá maximizar no decorrer do tempo em que se proceder a interação (Figura 12). A cada instante t , o agente realiza um mapeamento, a política π ¹⁰, do estado apresentado (s_t) para as probabilidades de selecionar cada uma das possíveis ações.

Figura 12 - Estrutura genérica do aprendizado por reforço.

O sinal de retorno é a forma de se comunicar o **que** se deseja do agente, sem entrar no mérito de **como** se deseja que ele aja. O objetivo do agente é a maximização dos retornos a longo prazo: o acumulado dos valores de r_t é que deve ser maximizado, não seu valor instantâneo. A idéia por trás da busca do maior acumulado no lugar da busca do maior retorno imediato é que uma sequência de ações ótimas não é necessariamente formada por ações que fornecem a cada instante o maior retorno¹¹. No caso mais simples, o que se deseja é a maximização da expressão:

$$r_t = r_{t+1} + r_{t+2} + r_{t+3} + \dots + r_T \quad (2)$$

sendo T o instante final da interação.

⁹ Alguns métodos de busca como algoritmos genéticos, *simulated annealing* e outros métodos de otimização podem ser utilizados para solucionar problemas de aprendizagem por reforço inferindo diretamente a política do agente sem realizar uma avaliação da contribuição de cada estado s para uma dada tarefa.

¹⁰ Um exemplo simples de política é a política *greedy*. Nesta, o agente busca sempre executar a ação que leve ao maior valor de V ou Q .

Muitos problemas não possuem um instante final bem determinado. Há casos em que não há um instante final, nestes casos: $T \rightarrow \infty$. Tais problemas são denominados **problemas de horizonte infinito**. Como a expressão (2), que deveria ser maximizada, poderia vir a ter valor infinito nos problemas de horizonte infinito, adota-se a premiação descontada que é uma modificação da expressão (2) e tem o seguinte formato:

$$r_t = r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + \dots = \sum_{k=0}^{\infty} \gamma^k r_{t+k+1}, \quad (3)$$

sendo $0 \leq \gamma < 1$ denominado taxa de desconto. A taxa de desconto faz com que a expressão (3) valorize os sinais de premiação mais próximos, reduza a importância dos sinais futuros e faça com que o somatório convirja para um valor finito mesmo quando $T \rightarrow \infty$. Todo problema de AR pode ser tratado como um problema de horizonte infinito: nos casos em que há um instante T final, os reforços a partir deste instante são considerados nulos.

Para PDMs, o valor de um estado s sob uma política π , denotado por $V^\pi(s)$, estima o acumulado esperado dos retornos do ambiente ao iniciar o processo no estado s e seguir a política π . Este valor é definido da seguinte forma:

$$V^\pi(s) = E_\pi \{ r_t \mid s_t = s \} = E_\pi \left\{ \sum_{k=0}^{\infty} \gamma^k r_{t+k+1} \mid s_t = s \right\} \quad (4)$$

sendo $E_\pi \{ \cdot \}$ o valor esperado dado que o agente siga a política π para $t \geq 0$. A função $V^\pi(s)$ é denominada **função de avaliação do estado para a política π** (*value function for policy π*).

De maneira semelhante, define-se como o valor de uma ação a tomada em um estado s sob uma política π , denotado por $Q^\pi(s, a)$, como o acumulado esperado dos retornos ao se começar o processo em um estado s , tomar uma ação a e a partir desta seguir a política π :

$$Q^\pi(s, a) = E_\pi \{ r_t \mid s_t = s, a_t = a \} = E_\pi \left\{ \sum_{k=0}^{\infty} \gamma^k r_{t+k+1} \mid s_t = s, a_t = a \right\} \quad (5)$$

A função $Q^\pi(s, a)$ é denominada **função de avaliação da ação para a política π** (*action-value function for policy π*).

¹¹ Uma ação que forneça maior retorno imediato pode levar a uma situação futura de baixo retorno do ambiente.

As funções de avaliação V^π e Q^π podem ser estimadas experimentalmente. Caso um agente, ao seguir uma política π , mantenha uma média para cada estado do retorno real obtido então esta média convergirá para V^π conforme o número de ocorrências do estado tenda para infinito. Caso sejam mantidas médias separadas para cada possível ação tomada em cada estado, então estas médias convergirão para Q^π .

Uma propriedade fundamental das funções de avaliação usadas em aprendizagem por reforço e programação dinâmica é que elas devem satisfazer determinadas relações recursivas (Sutton e Barto, 1998). Para qualquer política π e qualquer estado s , a seguinte condição de consistência se mantém entre os valores de s e os valores de seus possíveis estados sucessores:

$$\begin{aligned}
 V^\pi(s) &= E_\pi \{r_t \mid s_t = s\} \\
 &= E_\pi \left\{ \sum_{k=0}^{\infty} \gamma^k r_{t+k+1} \mid s_t = s \right\} \\
 &= E_\pi \left\{ r_{t+1} + \gamma \sum_{k=0}^{\infty} \gamma^k r_{t+k+2} \mid s_t = s \right\} \\
 &= E_\pi \{r_t + \gamma V^\pi(s_{t+1}) \mid s_t = s\}
 \end{aligned} \tag{6}$$

A equação (6) expressa a relação recursiva entre o valor de um estado e os valores de seus estados sucessores. Esta equação é a base para o cálculo de V^π .

A partir de V^π , pode-se fazer modificações na política buscando alcançar uma política ótima (π^*). Uma política π é melhor que ou igual a outra política π' se a expectativa de retorno acumulado seguindo π é maior que ou igual à expectativa de retorno seguindo π' para todos os estados. Formalmente:

$$\pi \geq \pi' \text{ se e somente se } V^\pi(s) \geq V^{\pi'}(s), \text{ para qualquer } s \in S. \tag{7}$$

Uma política melhor que ou igual a todas as outras políticas é denominada política ótima (π^*). A função de avaliação de uma política ótima é a função de avaliação ótima (V^*) definida como sendo:

$$V^*(s) = V^{\pi^*}(s) = \max_{\pi} V^\pi(s), \text{ para qualquer } s \in S. \tag{8}$$

Políticas ótimas também possuem funções de avaliação de ações ótimas (Q^*) definidas como:

$$Q^*(s, a) = Q^{\pi^*}(s, a) = \max_{\pi} Q^\pi(s, a), \text{ para qualquer } s \in S \text{ e } a \in A. \tag{9}$$

Para os pares estado-ação (s, a) , a função $Q^*(s, a)$ fornece o retorno máximo esperado ao se realizar uma ação a em um estado s e se continuar seguindo uma política ótima. A função Q^* pode ser escrita em termos de V^* :

$$Q^*(s, a) = E\{r_{t+1} + \gamma V^*(s_{t+1}) | s_t = s, a_t = a\}, \text{ para qualquer } t \geq 0. \quad (10)$$

Dos conceitos apresentados, tem-se, em resumo, que a solução de um problema de aprendizagem por reforço está na obtenção da função de avaliação ótima e para tal há três classes fundamentais para obter soluções: programação dinâmica, métodos de Monte Carlo e aprendizagem por diferença temporal. O termo programação dinâmica é utilizado para uma classe de algoritmos que podem gerar políticas ótimas em PDMs desde que um modelo completo e preciso do ambiente seja conhecido¹². Métodos de programação dinâmica (Bellman, 1957; Bertsekas, 1987) são bem fundamentados matematicamente, porém na grande maioria dos problemas não se dispõe *a priori* do modelo do ambiente de trabalho. Os métodos de Monte Carlo (Kalos e Whitlock, 1986; Rubinstein, 1981) não requerem um modelo e são conceitualmente simples: as estimativas das avaliações dos estados são calculadas a partir da amostragem de seqüências de estados, ações e reforços tomada da interação direta, ou simulada entre agente e ambiente. Esta metodologia, entretanto, necessita que primeiramente uma grande quantidade de informação seja gerada para que, depois de todo esse levantamento de dados, as avaliações sejam atualizadas.

Os métodos de diferença temporal (*Temporal Difference* - TD) (Sutton, 1988) são uma combinação da programação dinâmica (PD) e dos métodos de Monte Carlo (MC): como em Monte Carlo, as estimativas das avaliações dos estados são aprendidas diretamente da interação com o ambiente sem a necessidade de um modelo da dinâmica do ambiente. Como em programação dinâmica, a atualização das avaliações é feita sem que seja necessário o acúmulo de uma grande quantidade de informação. Os métodos TD são totalmente incrementais, e também possuem vantagens com relação aos outros em aspectos como a eficiência e velocidade de convergência, sendo, atualmente, a metodologia adotada pela grande maioria dos trabalhos em AR (Millán, 1996), exemplos disso são os trabalhos de Barto *et al.*

¹² Seguindo o formalismo dos PDMs, um modelo completo do ambiente está definido se forem conhecidas a matriz de transição dos estados e as estimativas de retorno imediato do ambiente para todos os estados do ambiente.

(1983), Tesauro (1995) e Millán (1996) e, por suas características, a metodologia empregada no agente proposto.

4.2 Aprendizagem por Diferença Temporal

Enquanto os métodos convencionais de aprendizagem de predição baseiam-se nos erros entre os resultados finais previstos e o real, os métodos TD são guiados pelo erro ou diferença entre predições temporárias sucessivas. O aprendizado ocorre sempre que há, no decorrer do tempo, uma mudança na predição. Para exemplificar o funcionamento de TD, considere o problema simples de predição apresentado na Figura 13: um agente deve estimar, a cada estado, o tempo que levará para alcançar seu estado meta, no caso s_5 .

Figura 13 – (a) Agente composto por seis estados. (b) O reforço r_t é medido no instante t e corresponde ao tempo decorrido do estado s_0 até o estado s_t , a avaliação do estado, $V(s_t)$, é a estimativa de tempo para alcançar o estado meta s_5 a partir do estado s_t e o valor que se deseja calcular. T é o tempo real decorrido ao partir do estado s_0 até atingir o estado meta s_5 .

O reforço r_t é um valor medido que indica o tempo gasto a partir do estado inicial s_0 para atingir o estado atual s_t . A cada estado s_t , o valor de $V(s_t)$ é uma estimativa do tempo restante para que o agente alcance seu estado meta, s_5 . O tempo total gasto pelo agente para ir do estado s_0 para o estado s_5 é denotado por T . Um conjunto de avaliações iniciais é apresentado na coluna $V(s_t)$ ¹³ da Tabela 2, e os sinais de reforço medidos a cada estado são dados na coluna r_t .

¹³ Os valores iniciais são quaisquer e servem apenas como um ponto de partida para o cálculo das estimativas.

Tabela 2 – Exemplo de valores para o agente.

Estado [s_t]	Avaliação Inicial [$V(s_t)$]	Sinal de Reforço [r_t]	Estimativa Inicial de tempo total: $r_t + V(s_t)$
s_0	30	0	30
s_1	35	5	40
s_2	15	20	35
s_3	10	30	40
s_4	3	40	43
s_5	0	43	43

O tempo total real para a transição de todos os estados¹⁴ é $T = 43$ (valor medido de r_5). Pela quarta coluna da Tabela 2, apenas as estimativas dos estados s_4 e s_5 estão corretas. Assim, faz-se necessário corrigir as demais estimativas atualizando as avaliações $V(s_t)$. Nos métodos de Monte Carlo, uma regra simples de atualização das avaliações teria a seguinte forma:

$$\Delta V(s_t) = \alpha [T - (r_t + V(s_t))] \quad (11)$$

sendo $\Delta V(s_t)$ o incremento a ser dado em $V(s_t)$ para corrigir a estimativa e α uma taxa de aprendizagem. Por (11), a correção das estimativas é realizada apenas quando o estado meta é alcançado e o valor de T é conhecido.

Em métodos TD, entretanto, a atualização ocorre, como em programação dinâmica, baseando a correção das estimativas em outras estimativas aprendidas, sem esperar o resultado final T . Enquanto os métodos de Monte Carlo aguardam o final de uma tentativa¹⁵, os métodos TD aguardam apenas até o passo seguinte para realizar a correção das estimativas. No instante $t+1$, os métodos TD atualizam o valor da função de avaliação para s_t utilizando a premiação r_{t+1} e a estimativa $V(s_{t+1})$. Uma regra de atualização em métodos TD seria:

$$\Delta V(s_t) = \alpha [(r_{t+1} + V(s_{t+1})) - (r_t + V(s_t))] \quad (12)$$

Um fato a ser notado em (12) é que o alvo para a atualização é $r_{t+1} + V(s_{t+1})$. Por basear parte de sua atualização em uma estimativa já existente, os métodos TD são ditos *bootstrapping*. Os gráficos da Figura 14 mostram como ocorrem as mudanças nas estimativas segundo Monte Carlo e por TD.

¹⁴ O valor real de T é conhecido apenas quando o estado s_5 é atingido, sendo $T = r_5$.

¹⁵ Uma tentativa é uma sequência alternada de estados e pares estado-ação que termina no estado meta.

Figura 14 - (a) Exemplo de mudanças realizadas em Monte Carlo. (b) Exemplo de mudanças realizadas em TD. Nos dois gráficos, as setas indicam o incremento que é dado em $V(s_t)$ para corrigir a estimativa.

Os métodos TD possuem capacidade incremental de aprendizagem pois a atualização da função de avaliação em TD pode ser calculada por incrementos que dependem unicamente do par de estimativas sucessivas (12), isto permite que o agente durante a exploração do espaço de estados já esteja melhorando as estimativas das avaliações dos estados. Os métodos TD combinam a amostragem de MC com o *bootstrapping* de PD: como não dispõem de um modelo do mundo, os valores do retorno são obtidos por amostragem e o cálculo das estimativas utilizam o valor atual da estimativa de $V(s_{t+1})$ no lugar do valor real.

As duas principais estratégias encontradas na literatura para AR utilizando TD são (Kaelbling *et al.*, 1996):

- ♦ **Crítico Heurístico Adaptativo (CHA)** (Barto *et al.*, 1983): Este algoritmo consiste de duas componentes (Figura 15): um crítico adaptativo (CHA), e uma componente de aprendizado por reforço (CAR).

Figura 15 - Arquitetura para o crítico heurístico adaptativo (Kaelbling *et al.*, 1996).

A componente de aprendizado por reforço busca agir de forma a maximizar o valor heurístico, v , que é calculado pelo crítico. O crítico usa o sinal real externo de reforço, r , para aprender a avaliar os estados s . Na maioria das implementações, as componentes, CHA e CAR, operam simultaneamente e apenas para essas implementações há garantia, sob condições adequadas, de convergência para uma política ótima (Kaelbling *et al.*, 1996).

- ♦ ***Q-Learning*** (Watkins e Dayan, 1992): O trabalho dos dois componentes do crítico heurístico é realizado de maneira unificada pelo algoritmo *Q-learning*. Este é o algoritmo de AR mais estudado para os propósitos de controle, possuindo provas de convergência (Jaakkola *et al.*, 1994) para o mesmo. *Q-learning* estima a avaliação do par estado-ação diretamente a partir dos sinais externos de reforço utilizando uma política *greedy*. Devido ao fato da função $Q(s,a)$ ser dependente da ação, os valores de Q podem ser estimados *on-line* usando metodologia TD, e esses valores definem a política, pois a ação é escolhida apenas considerando aquela que maximiza o valor de Q para o estado atual conforme a regra de atualização (13).

$$\Delta Q(s_t, a_t) = \alpha(r + \gamma \max_{a_{t+1}} Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t)) \quad (13)$$

A presente pesquisa buscará mudanças nas estratégias tradicionais de TD visando reduzir certas limitações enfrentadas na aplicação de algoritmos AR em ambientes reais. A próxima seção cita algumas dessas limitações na navegação autônoma de robôs móveis.

4.3 Aprendizagem por Reforço em Robôs Móveis

Nesta década, diversos trabalhos têm utilizado algoritmos de AR em robôs móveis reais (Kaelbling, 1990; Milán e Torras, 1992; Gullapalli *et al.*, 1994; Donnart e Meyer, 1996; Milán, 1996; Mataric, 1997). Há, entretanto, uma série de dificuldades associadas à aplicação de métodos AR a robôs reais. De forma geral, Wyatt (1997) cita os seguintes problemas:

- i) **A interação do ambiente ser tratada como um PDM:** Caso o agente não possua informação suficiente para prever o comportamento futuro do processo, então ocorre o que Whitehead denomina *perceptual aliasing* (Whitehead e Ballard,

1991). Isto ocorre quando o agente não distingue entre dois estados que são significativamente diferentes com respeito aos seus comportamentos sob uma mesma política.

- ii) **Lenta convergência dos algoritmos:** Os métodos de aprendizagem por reforço são bastante lentos e empobrecem seu desempenho com o aumento do tamanho dos espaços de estado e ação tornando com isso difíceis de serem utilizados por robôs reais. Devido ao número de possíveis estados crescer exponencialmente com o aumento do número de características do ambiente (Dean *et. al.*, 1995), o tempo decorrido para a solução do PDM cresce rapidamente conforme a complexidade do ambiente aumenta. Conseqüentemente, em ambientes estocásticos com um espaço de muitos estados os métodos AR levam um tempo proibitivo para convergir.
- iii) **Tabula rasa:** Por não possuir nenhuma informação inicial sobre o ambiente, o agente realiza uma exploração do espaço de estados recorrendo, geralmente, à inclusão de uma componente aleatória em sua política. Esta componente, com o decorrer do funcionamento do agente, entretanto, passa a prejudicar as soluções encontradas por forçar modificações nas decisões em trechos das seqüências de ações tomadas que já não podem ser melhorados. Este problema é tratado na literatura como o compromisso entre exploração e repetição (*exploration / exploitation tradeoff*) (Kaelbling *et al.*, 1996).

As características do agente determinadas na Seção 2.3 são um resumo de como se tentará reduzir essas limitações. O Capítulo 5 descreve o agente e no decorrer de sua apresentação comenta-se como seu funcionamento aborda os problemas acima citados.

4.4 Conclusões

Os métodos de Aprendizagem por Reforço (AR) permitem a criação de agentes que aprendem a mapear autonomamente estados do ambiente em ações. Este aprendizado é realizado por interações do tipo tentativa-e-erro entre o agente e o ambiente. A base matemática de AR vem do estudo dos processos de decisão markovianos (PDM) originados em controle ótimo estocástico. A teoria de programação dinâmica fornece ferramentas para a solução de PDMs, porém a necessidade de um modelo completo do mundo inviabiliza a aplicação direta de

programação dinâmica em problemas reais. Assim, outros métodos inspirados em PD foram desenvolvidos de forma a tratar problemas reais que possam ser modelados como PDMs. Desses novos métodos, os principais são: os métodos de Monte Carlo (MC) e os métodos de aprendizagem por diferença temporal, *temporal difference* (TD). Os métodos MC, apesar de não necessitarem de um modelo do mundo em que o agente está atuando, têm o inconveniente de necessitarem de uma grande quantidade de amostras do comportamento do ambiente para que o aprendizado seja realizado.

Os métodos TD são uma extensão, por excelência, dos conceitos de aprendizado autônomo para o controle ótimo na maneira que são técnicas mais simples de calcular diretamente a política ótima sem um passo intermediário de avaliação de custo e sem o uso de um modelo. Na área de AR, os métodos TD são o foco atual dos estudos por suas características permitirem um aprendizado rápido comparado com outras metodologias e não estar atrelado a qualquer conhecimento prévio do ambiente de trabalho do agente. Esta pesquisa utiliza variações de métodos TD para a resolução da tarefa descrita na Seção 2.2.

CAPÍTULO 5

Descrição do Agente Proposto

Marr (1982) argumenta que um sistema de processamento de informação complexo, como é o caso do agente aqui tratado, pode ser entendido em três diferentes níveis de descrição: o nível computacional, o nível algorítmico e o nível implementacional. O **nível computacional** visa caracterizar o objetivo do processamento do modelo e justificar sua adequação ao problema. O **nível algorítmico** determina um procedimento para a realização da meta e uma representação conveniente para as variáveis do problema. Finalmente, o **nível implementacional** especifica os detalhes relacionados ao modo como o algoritmo é executado.

Tomando por base as definições dadas acima, o agente proposto é apresentado nas três seções seguintes de acordo com os níveis definidos por Marr.

5.1 Nível computacional

Alguns pesquisadores buscam estabelecer comportamentos básicos dos quais emergem outros mais complexos em um agente durante sua interação com o ambiente (Anderson e Donath, 1990; Balkenius, 1995; Beer *et al.*, 1990). Esta abordagem no projeto de agentes visa torná-los mais rápidos no aprendizado de suas tarefas, pois são apontadas direções para a melhor forma de alcançar os objetivos desejados. Entretanto, a definição de quais comportamentos básicos podem ser importantes para as tarefas do agente exige que o projetista se aprofunde no estudo de tais tarefas e que codifique o conhecimento por ele adquirido nos comportamentos básicos. Tal procedimento limita os comportamentos emergentes a composições dos comportamentos básicos, o que pode impedir o agente de buscar soluções melhores.

O funcionamento procurado no processamento do agente a ser proposto é que seu aprendizado seja baseado em um mecanismo simples de geração de

comportamentos que não tenha dependência significativa com comportamentos pré-estabelecidos, tornando-o mais flexível. A flexibilidade entretanto não deve implicar em uma longa exploração por soluções. A inspiração para o cumprimento de tais objetivos foi buscada nos estudos de comportamento animal por ser uma fonte para os princípios que geram as tomadas de decisão nos agentes biológicos. Manning (1979) argumenta que comportamento inclui todos os processos pelos quais um animal, ao interpretar o mundo externo e o estado interno de seu organismo, responde às mudanças que ele percebe. Assim, o comportamento não é função apenas dos estímulos externos recebidos pelo animal mas também é associado com processos internos (Anderson e Donath, 1990). A Figura 16 mostra um diagrama das componentes que devem estar presentes no agente: uma fonte externa de estímulo que sirva de alimentação do ambiente para o agente, uma representação de seu estado interno que servirá para o agente contextualizar o estímulo externo e um processamento que avalie as ações a serem tomadas a partir das entradas citadas.

Figura 16 - Geração de um comportamento no qual o animal leva em consideração os estímulos externos e o seu estado interno.

Este processamento deve ocorrer de forma a replicar a lei do efeito enunciada por Thorndike (*Thorndike's law of effect*) (1911):

“Dentre as diversas respostas obtidas para uma mesma situação, aquelas que são acompanhadas ou seguidas de perto pela satisfação do animal serão, de outra maneira permanecem como estavam, relacionadas mais firmemente com a situação, de tal forma que quando da repetição da situação tais respostas ocorrerão com maior frequência; aquelas que são acompanhadas ou seguidas de perto por desconforto do animal serão, de outra maneira permanecem como estavam, relacionadas mais fracamente com a situação, de tal forma que quando da repetição da situação tais respostas dificilmente ocorrerão. Quanto maior a satisfação ou o desconforto tanto maior será o fortalecimento ou o enfraquecimento do laço entre a situação e a resposta.”

Modelos de condicionamento clássico (Grossberg e Levine, 1987; Sutton e Barton, 1990) têm surgido como forma de tentar explicar como um organismo pode adquirir um comportamento autônomo em um ambiente em constante mudança. Em condicionamento clássico, o aprendizado ocorre pela associação repetitiva de um estímulo condicionado (CS), o qual normalmente não possui nenhum significado particular para um animal, com um estímulo não condicionado (UCS), o qual possui significado para o animal e sempre fornece uma resposta não-condicionada (UCR). Um exemplo típico é um cachorro que ao ouvir repetidamente uma campainha antes de ser alimentado acaba por passar a salivar ao escutar a campainha. A resposta que passa a ser provocada pelo CS após o condicionamento clássico é conhecido como uma resposta condicionada (CR).

Uma forma relacionada de aprendizagem é conhecida como condicionamento operante (Chang e Gaudiano, 1997). Neste caso, um animal aprende as consequências de suas ações. Mais especificamente, o animal aprende a exibir com maior frequência um comportamento que seja premiado, e a seguir com menor frequência um comportamento que o leve a uma punição¹⁶. Nesta situação, o animal não pode simplesmente esperar o decorrer dos acontecimento, mas deve gerar diferentes comportamentos e aprender aqueles que são mais apropriados. Um algoritmo de AR que utilize uma regra de atualização TD pode reproduzir o condicionamento operante: comportamentos aleatórios podem ser inicialmente gerados e, com o funcionamento do robô, aqueles que levem às metas passarão a ser preferidos. Na tarefa a ser executada pelo agente proposto (Seção 2.2), o mesmo deve aprender a valorizar as ações que o leve ao estado meta e que evitem eventuais obstáculos do ambiente.

A Figura 17 mostra como deve se processar a interação agente-ambiente. Uma descrição do estado em que o ambiente se encontra e um sinal de reforço - que no caso da navegação acontecerá devido à presença de obstáculos ou ao encontro da meta no estado para o qual a ação tomada tenha levado o ambiente - são as entradas do agente e suas saídas são suas ações. O aprendizado tem a função de realizar a associação entre entradas e saídas de forma a atribuir crédito ou responsabilidade para as consequências de cada decisão interna realizada pelo sistema. Este procedimento é denominado designação de crédito (Minsky, 1961)

¹⁶ Por exemplo, um gato faminto preso em uma gaiola de onde pode ver comida aprende a pressionar uma alavanca que o permite fugir da gaiola e alcançar a comida.

Figura 17 - Processo de interação agente-ambiente.

Estabelecido que o agente que realizará navegação autônoma seguirá o paradigma de AR (Seção 3.1.3.1), surge a questão de como este agente solucionará o **problema de exploração direcionada à meta**¹⁷ (PEDM) (Koenig e Simmons, 1996). Em um PEDM, o agente tem que encontrar algum caminho para o estado meta em seu espaço de estado inicialmente desconhecido, ou apenas parcialmente conhecido, mas não necessita encontrar o melhor caminho. Whitehead (1991a) mostra que a exploração direcionada à meta com AR pode ser intratável. Em particular, ele argumenta que o comportamento de um agente que é controlado por um algoritmo de AR inicialmente desinformado pode degenerar para uma caminhada aleatória até que o estado meta seja alcançado. Neste caso, tem-se que executar uma quantidade de ações que têm, em média, uma relação exponencial com o tamanho do espaço de estado, e mesmo técnicas que aumentam a velocidade da busca, como a técnica de Lin (1992), não melhoram significativamente o desempenho do agente. Koenig e Simmons (1996), porém, argumentam que a representação dada ao sinal de reforço e a determinação das estimativas de retorno iniciais de cada estado exercem um papel importante no funcionamento de um agente AR, podendo contribuir significativamente para a melhoria na performance do mesmo. As observações experimentais de Kaelbling (1990) e do próprio Whitehead (1991b) que registram bons resultados na performance de seus agentes AR, e que utilizam representações diferentes das empregadas por Whitehead (1991a), são exemplos da validade das considerações de Koenig e Simmons.

¹⁷ O PEDM é uma etapa para a solução do problema de aprendizagem por reforço direcionada à meta (PARDM) mencionada na Seção 3.2. Solucionada esta primeira etapa, a solução do PARDM é conseguida melhorando a solução inicial obtida, buscando alcançar a solução ótima.

A representação dos sinais de reforço, juntamente com uma representação adequada do estado do ambiente, são dois fatores explorados nesta pesquisa para levar à redução dos problemas em aplicar AR a robôs reais, apresentados na Seção 4.3. A próxima seção comenta sobre o formato dos sinais de reforço adotados e o algoritmo utilizado para processar esses sinais e gerar o aprendizado do agente. A Seção 5.3, que trata do nível implementacional, explica como foram escolhidas representações simples dos estados para evitar o problema denominado *perceptual aliasing*.

5.2 Nível algorítmico

Retomando o diagrama da Figura 16, a geração de um comportamento é dependente de uma fonte externa de estímulo, a qual informa sobre a interação agente-ambiente e que em um agente AR corresponde ao sinal de reforço. O comportamento é também dependente de um estado interno que contém a informação necessária para diferenciar situações distintas que podem ser geradas a partir de um mesmo estímulo e um processamento que seleciona as ações a partir destas duas fontes de informação. Sinais de reforço e processamento são os temas principais desta seção e estão apresentados em suas duas subseções. O estado interno do agente, por ser uma componente sob a qual considerações de implementação devem ser tomadas, é comentada apenas indiretamente nesta seção, ao se tratar do algoritmo do agente, ficando para o nível implementacional, Seção 5.3, uma discussão maior.

5.2.1 Definição da Representação do Sinal de Reforço

Há basicamente duas representações na literatura para o reforço dado ao agente para aprender caminhos no espaço de estados: recompensa da meta e penalização das ações. As duas são apresentadas nos itens a seguir e servem como argumentos para a representação escolhida dos sinais de reforço do agente que é comentada no item 5.2.1.3.

5.2.1.1 Representação por Premiação da Meta (Goal-Reward Representation)

Nesta representação o agente é recompensado quando atinge o estado meta, mas não é premiado ou penalizado de outra forma (Sutton, 1990; Whitehead, 1991a). Formalmente:

$$r_{t+1}(s_t, a_t, s_{t+1}) = \begin{cases} 1, & \text{caso } s \text{ seja o estado meta.} \\ 0, & \text{para qualquer outro estado.} \end{cases} \quad (14)$$

sendo $r_{t+1}(s_t, a_t, s_{t+1})$ ¹⁸ o sinal de reforço do ambiente no instante $t+1$ dado que o agente executou a ação a_t que realiza a transição do estado s_t para o estado s_{t+1} .

Na representação por premiação da meta faz-se necessário o uso do fator de desconto, γ , para o cálculo da premiação acumulada, como em (3), para o aprendizado dos menores caminhos no espaço de estados. Caso o fator de desconto não seja usado, como em (2), todos os comportamentos que eventualmente levam o agente à meta terão um retorno total igual a 1 e os algoritmos de aprendizagem por reforço não poderão distinguir entre os caminhos com tamanhos diferentes. Ao utilizar o desconto, quanto maior o número de ações de um caminho que leve à meta tanto maior será o desconto sobre o retorno acumulado desse caminho e o agente tentará alcançar o estado meta com a menor quantidade possível de ações de forma a obter a máxima premiação acumulada.

Inicializar o agente de AR com zeros em uma representação por premiação da meta resulta em premiações esparsas, fazendo com que o agente realize uma caminhada aleatória no espaço de estados. Nesta caminhada aleatória, o agente deverá alcançar a meta com probabilidade 1 porém a média do número de ações executadas terá uma relação exponencial com o tamanho do espaço de estados (Whitehead, 1991a). A utilização da representação por premiação da meta com inicialização nula das estimativas dos estados pode tornar o problema de exploração intratável: como o agente recebe um valor não nulo apenas quando alcança a meta, até que esteja no alvo o agente não tem nenhuma informação que possa melhorar sua busca. Caso ainda o agente realize uma escolha sistemática das ações, como sempre escolher a ação que leve ao estado com menor avaliação, então o agente pode mesmo entrar em um *loop* infinito no espaço de estados.

5.2.1.2 Representação por Penalização das Ações (Action-Penalty Representation)

Neste caso, o agente é penalizado por cada ação que execute. Esta estrutura de retorno é mais densa que a representação por premiação da meta (pois o agente recebe

¹⁸ As notações r_{t+1} e $r_{t+1}(s_t, a_t, s_{t+1})$ têm o mesmo significado sendo a segunda notação empregada apenas para destacar a ação e transição de estados em que o sinal de reforço ocorre.

retornos não-nulos com maior frequência) se as metas forem relativamente esparsas (Barto *et al.*, 1989; Koenig, 1991; Barto *et al.*, 1995). Formalmente:

$$r_{t+1}(s_t, a_t, s_{t+1}) = -1, \text{ para qualquer } s \in S \text{ e } a \in A \quad (15)$$

Para o aprendizado de caminhos mais curtos, o desconto pode ser usado, mas não é necessário. Utilizando-se ou não o fator de desconto no cálculo da premiação acumulada, o agente tenta alcançar o estado meta realizando o menor número possível de ações, buscando minimizar a quantidade de penalizações que o agente recebe.

A representação do sinal de reforço por penalização das ações, pela maior ocorrência de retornos não nulos, é mais informativa para o agente tornando-o menos propenso a entrar em uma caminhada aleatória pelo espaço de estados. Entretanto, o direcionamento do agente para o estado meta ainda é muito dependente de uma componente aleatória da política e não são utilizadas informações sobre a estrutura do ambiente que auxiliem na busca pelo estado meta.

5.2.1.3 Representação Adotada para os Sinais de Reforço

Koenig e Simmons (1996) indicam que os algoritmos de AR podem tratar do problema de exploração direcionado à meta desde que certos pontos sejam considerados, sendo os principais: o agente deve selecionar suas ações segundo uma distribuição de probabilidades uniforme e a representação do sinal de reforço deve ser densa de forma a reduzir a busca no espaço de estados como ocorre, por exemplo, na representação por penalização das ações. Neste trabalho, é proposta uma representação densa como a penalização das ações que faz uso, além da indicação do estado meta e do número de ações que levam a este estado, de informações sobre a estrutura do ambiente, no caso a presença de obstáculos, para diminuir a exploração do espaço de estados e direcionar mais rapidamente o agente para seu estado meta.

Um esquema de penalizações e premiações é adotado para os sinais de reforço do agente que está relacionado com os três tipos de estados do ambiente adotados (Seção 5.3): estado meta, estado com obstáculo e estado livre. Assim, o agente é premiado por entrar em um estado alvo e penalizado por colidir com obstáculos. Em estados livres o sinal de reforço é nulo. Formalmente:

$$r_{t+1}(s_t, a_t, s_{t+1}) = \begin{cases} +1, \forall s_t \in F, a \in A, e s_{t+1} \in G; \\ -1, \forall s_t \in F, a \in A, e s_{t+1} \in O; \\ 0, \forall s_t \in F, a \in A, e s_{t+1} \in F; \end{cases} \quad (16)$$

sendo:

- G - o conjunto finito dos estados alvo;
- O - o conjunto finito dos estados com obstáculos;
- F - o conjunto finito dos estados livres;
- $s_t \in S, \emptyset \neq G \subseteq S, \emptyset \neq O \subset S, G \neq O, \text{ e } F = S - (G \cup O).$

Conforme a Seção 2.2, considera-se que o agente inicia sua interação com o ambiente sem nenhuma informação sobre o mesmo. Assim, as avaliações iniciais de seus estados são consideradas nulas. Formalmente:

$$V(s_t) = 0; \forall s_t \in S, t = 0 \quad (17)$$

As avaliações dos estados são modificadas pela regra de atualização da avaliação do estado s dada pela expressão:

$$\Delta V(s_t) = \alpha [r_{t+1}(s_t, a_t, s_{t+1}) + \gamma V(s_{t+1}) - V(s_t)] \quad (18)$$

sendo:

α - taxa de aprendizagem, $0 < \alpha < 1$;

γ - fator de desconto, $0 \leq \gamma < 1$.

Deseja-se que a informação da posição do estado meta seja distribuída o máximo possível. Porém, se a distribuição da informação da posição dos obstáculos for tão intensa quanto a da posição do alvo, barreiras de penalização podem fazer com que o robô evite estados promissores (ver exemplo no primeiro agente do APÊNDICE II). Desta forma, estruturas distintas foram adotadas para codificar as avaliações da recompensa e da penalização de cada estado e em cada estrutura as avaliações são distribuídas diferentemente no espaço de estados.

Neste trabalho, as estruturas definidas são semelhantes aos campos potenciais sugeridos por Khatib (1986). Uma superfície atratora indicará os estados mais promissores e será formada pelas avaliações das premiações. Para o desvio dos obstáculos, uma segunda superfície, com a avaliação das penalizações, indicará o quanto cada estado deverá ser evitado. A intenção dessa representação é que a superfície atratora forneça ao agente uma visão global do ambiente, sendo responsável pelo planejamento deliberativo, e a superfície repulsora tenha uma atuação mais local contribuindo que o agente apresente comportamentos reativos (Araújo e Braga, 1998a; Braga e Araújo, 1998a). A Figura 18 mostra um exemplo de como deve ser a

distribuição dos valores de avaliação nas duas superfícies: a penalização deve ocorrer apenas próxima ao obstáculo, indicando ao agente apenas o contorno da região a ser evitada que no caso da figura são dois obstáculos pontuais; a recompensa tem um alcance maior para que em qualquer estado em que esteja, o agente saiba como se movimentar para a meta.

Figura 18 - (a) Ambiente bidimensional composto por 9x9 estados com um estado meta e dois estados com obstáculo. (b) Superfície de recompensa correspondente. (c) Superfície de penalização correspondente.

No exemplo da Figura 18, a região entre os obstáculos faria parte da trajetória de um agente que partisse do ponto (5,2) por ser baixa a penalização desta região na superfície de penalização (Figura 18.c). Porém, caso a distribuição da penalização no espaço de estados tivesse uma intensidade semelhante à da recompensa, o formato da superfície de penalização seria como a Figura 19 e a região entre os obstáculos teria valores de penalização que tenderiam a impedir o agente de atravessá-la.

Figura 19 - Superfície de penalização alternativa para o ambiente da Figura 18.a.

Cada superfície tem uma relevância diferente na solução do problema de aprendizagem por reforço direcionada à meta: o agente utiliza a superfície de penalização principalmente para a solução do PEDM enquanto na fase de otimização ocorre um maior emprego da superfície de recompensa. Isto acontece pela ausência de conhecimento inicial do agente sobre seu ambiente de trabalho. Assim, o agente assume um comportamento totalmente aleatório que vai se tornando cada vez mais reativo conforme as colisões vão ocorrendo. Este comportamento evolui até alcançar um caminho que leve ao alvo e a superfície de recompensa passe a ser modificada. O processo descrito pode ser visualizado na Figura 20 onde a modificação das superfícies é apresentada conforme o aprendizado do agente vai ocorrendo. O processo de aprendizagem modifica a forma inicialmente plana das superfícies de forma a criar colinas na superfície de recompensa e cavar vales na superfície de penalização. A política do agente o orientará a alcançar as posições mais altas das colinas (alvos) e evitar os vales (obstáculos).

Figura 20 - Exemplo da evolução das superfícies de recompensa e penalização: (a) formas iniciais ($\forall s: V(s) = 0$); (b) aparência após exploração do ambiente, sem ter sido localizado o alvo; (c) aspecto seguinte à localização do alvo.

Definidos os sinais de reforço e as estruturas que codificam as avaliações dos estados, resta definir como essas avaliações são aprendidas pelo agente e como o agente as utilizará para selecionar suas ações. Estes são os temas do próximo item.

5.2.2 Apresentação do Processamento do Agente

O aprendizado do agente materializa-se na adaptação das superfícies de penalização e recompensa conforme o algoritmo da Figura 21. Este algoritmo considera a existência de um único estado meta e considera que vários estados iniciais sejam tomados para uma melhor exploração do ambiente.

```

Inicialize com zeros as avaliações de todos os estados em ambas as superfícies;
Repita (loop principal)
  Limpe o vetor que armazena a trajetória;
  Tome um estado  $s$  inicial para o agente;
  Repita (loop central)
    Selecione uma ação  $a$  conforme a política do agente;
    Execute a ação  $a$ ;
    Atualize a avaliação da penalização do estado  $s$ ,  $V_p(s_t)$ , de acordo com (19);
    Armazene o estado  $s$  no vetor trajetória;
  Até que o agente atinja um estado alvo ou obstáculo;
  Se o novo estado é um alvo, atualize a avaliação da recompensa,  $V_r(s_t)$ , de
  todos os estados armazenados no vetor de trajetória conforme (20);
Até que o agente encontre o alvo por um número pré-determinado de vezes.

```

Figura 21 - Algoritmo implementado em Português procedimental.

No *loop* central da Figura 21, as ações do agente são selecionadas e os comportamentos reativos são aprendidos através da atualização da superfície de penalização. O agente constrói esta superfície atualizando os valores $V_p(s_t)$ sempre que o agente realiza uma mudança no seu posicionamento espacial, levando em consideração apenas o estado atual (s_t) e o seguinte (s_{t+1}):

$$\Delta V_p(s_t) = \alpha_p [r_{t+1} + \gamma_p V_p(s_{t+1}) - V_p(s_t)] \beta_p \quad (19)$$

sendo:

$$\beta_p = \begin{cases} 1, & \text{se um obstáculo está no campo de ação de qualquer sensor do agente;} \\ 0, & \text{caso contrário.} \end{cases}$$

$$r_{t+1} = \begin{cases} -1, & s_{t+1} \in 0; \\ 0, & \text{caso contrário.} \end{cases}$$

$$-1 \leq V_p(s_t) \leq 0$$

A confirmação sensorial da presença dos obstáculos é levada em consideração para garantir que a superfície de penalização seja atualizada apenas na vizinhança do obstáculo definida pelo alcance dos sensores. Este caráter local do *loop* central do algoritmo possibilita rapidez ao agente em suas tomadas de decisão por realizar um processamento que considera pouca quantidade de informação.

O *loop* principal do algoritmo da Figura 21 trata do aprendizado do comportamento deliberativo do agente através da atualização da superfície de recompensa nos estados contidos no vetor trajetória. A atualização da superfície de recompensa só pode ocorrer após o agente localizar o alvo, até então o agente armazena todos os estados ao longo da trajetória aguardando a confirmação sensorial de que o alvo tenha sido alcançado. Confirmação obtida, a recompensa é distribuída ao longo de toda a trajetória seguindo a expressão:

$$\Delta V_r(s_t) = \alpha_r [r_{t+1} + \gamma_r V_r(s_{t+1}) - V_r(s_t)] \beta_r \quad (20)$$

sendo:

$$\beta_r = \begin{cases} 1, & \forall s_t \in T_g; \\ 0, & \text{caso contrário.} \end{cases}$$

$$r_{t+1} = \begin{cases} 1, & s_{t+1} \in G; \\ 0, & \text{caso contrário.} \end{cases}$$

$$0 \leq V_r(s_t) \leq 1$$

T_g é a trajetória armazenada atual que levou ao estado meta.

Durante a aprendizagem do agente¹⁹, a política que seleciona as ações é baseada em uma composição²⁰ das avaliações de recompensa e penalização:

¹⁹ Durante a fase de testes do agente, a componente totalmente aleatória foi excluída da política para verificar a eficiência das trajetórias geradas sob influência apenas do conhecimento armazenado sobre o ambiente. Maiores detalhes serão apresentados no Capítulo 6.

²⁰ A distribuição apresentada para a política do agente foi estabelecida heurísticamente.

- Em 60% dos casos, o algoritmo seleciona a ação que leva a um maior valor combinado (recompensa + penalização);
- Em 10% dos casos, o algoritmo seleciona a ação que leva o agente ao estado de direção oposta ao de maior penalização;
- Em 5% dos casos, o algoritmo escolhe a ação que leva à maior recompensa;
- Em 25% dos casos, o algoritmo realiza uma escolha aleatória da ação;
- Em caso de empate²¹, a ação é escolhida aleatoriamente.

As primeiras três componentes da política favorecem o agente a manter uma tendência de repetir um caminho já percorrido, seguindo o crescimento das premiações e evitando as penalizações. A quarta componente, entretanto, adiciona uma aleatoriedade ao agente que o permite testar variações do caminho inicialmente aprendido. Esta aleatoriedade possibilita a obtenção de melhorias nas trajetórias conforme ocorrem os ciclos de treinamento²². O funcionamento da política do agente segue a idéia da estrutura da *subsumption architecture* de Brooks (1986) (Figura 22). Contudo, as camadas na abordagem aqui apresentada não são compostos por comportamentos pré-programas mas sim por tendências de comportamento que vão sendo aprendidas através de um mecanismo único conforme as superfícies vão se formando e cujos conflitos são solucionados pela distribuição percentual acima descrita. Esta abordagem torna o agente mais adaptável ao ambiente em que está operando por não ter que se basear em comportamentos fixos.

Figura 22 - Tendências de comportamento que compõem a política do agente.

Uma das características desejadas para o agente é a capacidade de melhorar seu desempenho. Em testes iniciais, onde os parâmetros da regra de atualização das estimativas foram considerados fixos (ver Seção 6.2), verificou-se que em estados

²¹ É considerado um empate quando as avaliações de todos os estados vizinhos são iguais.

²² Um ciclo de treinamento corresponde à seleção aleatória de alguns estados iniciais para que o agente atinja o estado meta a partir de todos eles.

visitados diversas vezes havia o surgimento de mínimos locais. Na Figura 23 há um exemplo de como podem surgir os mínimos locais: duas trajetórias que ligam o ponto alvo e o ponto de partida e possuem um estado de cruzamento (s_c). Nesta situação, para cada trajetória a estimativa de s_c possui um valor diferente, sendo:

$$V_1(s_c) < V_2(s_c) \quad (21)$$

onde V_1 e V_2 designam respectivamente as estimativas devido as trajetórias 1 e 2.

Figura 23 - Exemplo de estado (s_c) visitado mais que uma vez em tentativas diferentes.

A expressão (21) ocorre por ser a trajetória 1 maior que a trajetória 2, fazendo com que na distribuição da recompensa, seguindo a equação (20), a estimativa de s_c seja menor na trajetória 1. Assim, as duas visitas ao estado s_c ocorridas seguindo as trajetórias 1 e 2 levam a estimativa deste estado a ser descontínua em relação às estimativas dos estados vizinhos.

A Figura 24 apresenta outro exemplo de ocorrência de mínimos locais: uma trajetória em que um mesmo estado foi visitado duas vezes. Nesta situação, ocorre uma descontinuidade abrupta dos valores ao longo da trajetória. Na primeira passagem, um valor $V(s_c) = X$ é atribuído ao estado s_c , na segunda passagem um novo valor é calculado $V(s_c) = Y$, com $X \neq Y$.

Figura 24- Exemplo de estado (s_c) visitado mais que uma vez em uma mesma tentativa.

Para obter a melhoria do aprendizado e evitar o surgimento de mínimos locais, o agente proposto utiliza uma taxa de aprendizagem variável em sua atualização de avaliações. Uma taxa de aprendizagem que diminua, para cada estado, proporcionalmente ao aumento do número visitas a este estado evita descontinuidades abruptas, os mínimos locais, ao impedir grandes modificações na estimativa inicial de $V(s)$. O gráfico da Figura 25 exibe um exemplo de como a taxa de aprendizagem deve se comportar durante o treinamento do agente.

Figura 25- Exemplo de comportamento esperado para a taxa de aprendizagem (eixo x = número de visitas a um ponto e eixo y = valor da taxa de aprendizagem).

Inicialmente, grandes variações nas avaliações são desejáveis, pois estas partem de um valor nulo. Porém, conforme o estado vá sendo visitado, tanto mais correta deve ser a estimativa de sua avaliação, e para garantir sua convergência para um valor fixo a taxa de aprendizagem deve decair, tendendo para 0 (Figura 25). A taxa de aprendizagem adotada é calculada por (Braga e Araújo, 1998b):

$$\alpha(s_t) = \alpha_2 \left(\frac{\alpha_1}{\alpha_2} \right)^{\frac{k(s_t)-1}{k(s_t)}} \quad (22)$$

sendo²³:

$\alpha(s_t)$ - a taxa de aprendizagem do estado s no instante t ;

α_1 - o valor mínimo da taxa de aprendizagem;

α_2 - o valor máximo da taxa de aprendizagem;

$k(s_t)$ - o número de visitas ao estado s em tentativas que tenham conduzido ao alvo²⁴.

²³ Os valores dos parâmetros são dependentes do tamanho do espaço de estados do agente (ver Seção 6.3).

²⁴ São consideradas apenas as visitas realizadas quando a tentativa levou o agente a um alvo.

Os resultados obtidos em simulação com o algoritmo descrito nesta seção são apresentados e discutidos no Capítulo 6. O agente autônomo foi, até o momento, tratado sob seus aspectos funcionais, sendo desconsiderados seus aspectos físicos como, por exemplo, a representação de seus estados. Tais considerações são importantes por tratarem de como viabilizar a aplicação do agente a ambientes reais. A próxima seção trata de preencher esta lacuna, encerrando a descrição do agente proposto.

5.3 Nível Implementacional

A implementação do agente foi realizada para um robô real e através de simulações deste robô. Aspectos físicos do robô, como suas dimensões, foram considerados para a definição dos estados do ambiente utilizados no algoritmo da Figura 21. Os itens que se seguem apresentam o sistema robótico utilizado nesta pesquisa, as considerações que foram tomadas para permitir que a interação entre este robô e o ambiente possa ser tratada como um processo de decisão markoviano e os principais problemas encontrados na implementação real do agente. Tais problemas, conforme será exposto, envolvem limitações encontradas no equipamento disponível que fogem ao escopo deste trabalho e não se relacionam com a eficiência do agente proposto. Assim, para uma melhor avaliação do agente, os testes apresentados no Capítulo 6 são realizados em simulações que seguem as observações discutidas nesta seção.

5.3.1 O Robô Real

O sistema robótico utilizado para implementar o agente foi um SmartROB semelhante ao apresentado na Figura 26. Este robô móvel foi desenvolvido pelo IfR (Instituto de Robótica) do Instituto Federal de Tecnologia da Suíça. Seu hardware é baseado em um microprocessador 68332 da Motorola, sendo equipado com canais de entrada/saída digitais e analógicos que permitem controlar motores de passo, motores DC e servo-mecanismos. Sua configuração também permite a conexão de diversos tipos de sensores como sensores infravermelho, de ultra-som e *encoders*. Para o presente trabalho, o SmartROB está equipado com dois *encoders* acoplados aos motores DC que fornecem movimento ao veículo e um sensor de distância infravermelho acoplado a uma base móvel na frente do robô.

Figura 26 - Fotos de um protótipo do robô móvel smartROB
(imagens obtidas na Home-Page da IfR- <http://www.ifr.mavt.ethz.ch>).

O software básico para o desenvolvimento de aplicações no robô móvel é o XOberon. Este software é um ambiente integrado que inclui as capacidades de edição, compilação e execução de programas, inclusive programas que trabalham em tempo real. O XOberon é programado em plataforma PC, na qual é realizada a programação e a geração do código objeto. Através da porta serial do computador é realizada a comunicação para transferência do código objeto do PC para o robô, que uma vez carregado com o programa, pode ser desconectado do cabo serial e se movimentar independentemente no espaço de trabalho.

5.3.2 Considerações para a Interação Robô-Ambiente ser um PDM

Robôs móveis são caracterizados como sistemas não-holonômicos, ou são ditos terem restrições não-holonômicos. Tais restrições podem ter diversos efeitos em veículos autônomos (Schmitt, 1996):

- Um veículo é limitado em como pode se movimentar. Este tem menos graus locais de liberdade que ele tem globalmente. Em outras palavras, enquanto uma posição e orientação puderem ser alcançadas, a configuração presente do veículo freqüentemente limita sua habilidade para alcançar o posicionamento sem o devido cuidado no uso de seus graus locais de liberdade
- As variáveis que descrevem a posição e a orientação do veículo são interdependentes. Por exemplo, um automóvel não pode alterar sua orientação sem modificar sua posição.
- As variáveis que descrevem a posição e orientação do veículo dependem do histórico dos valores das variáveis.

O robô móvel utilizado (Figura 27.a) está sujeito a duas restrições não-holonômicas: a primeira é que a velocidade da roda da frente tem a direção para a qual a roda está apontando, e a segunda é a de que as rodas traseiras não podem mover-se lateralmente. Estas restrições dificultam a representação da navegação como um PDM pois informações anteriores ao estado anterior, como a velocidade, são necessárias, descaracterizando o problema como sendo markoviano.

Figura 27 - (a) Figura reproduzida de Zalama *et al.* (1995): as velocidades angulares ω_L e ω_R são aplicadas respectivamente às rodas esquerda e direita do robô. As rodas da frente controlam a direção do movimento do robô. Durante um curto intervalo de tempo Δt , o robô move-se ao longo de uma trajetória circular com centro em X_C através de uma distância ΔS e ângulo $\Delta \theta$. (b) Oito possíveis ações do agente a cada estado (em preto).

Para trazer o problema para o formato de um PDM, e poder tratá-lo com o algoritmo proposto, adotou-se uma discretização do ambiente que permite uma representação dos estados em que o robô móvel pode ser tratado como sendo pontual, contendo, em cada estado, todas as informações necessárias para o planejamento de sua trajetória. Cada estado do agente corresponde a uma região discreta do ambiente de tamanho suficiente para permitir ao robô realizar um giro em torno de seu eixo central. Assim, considerando que ocorra a cada estado uma parada momentânea do robô para direcioná-lo para o próximo estado e a orientação com que o robô chega a este estado seja desconsiderada, o problema deixa de possuir restrições não-holonômicas. As regiões consideradas como estados são apresentadas na Figura 28 por círculos rajados de raio r proporcional às dimensões do robô. Os movimentos do

robô são limitados às oito ações mostradas na Figura 27.b e compostas pelos procedimentos de rotação e translação implementadas em Oberon. O espaçamento considerado entre os estados (Figura 28) foi tomado de forma a evitar sobreposição entre eles (*aliasing*), o que poderia causar avaliações errôneas dos estados causadas pela incapacidade do agente de distinguir entre dois estados próximos.

Figura 28 - Discretização utilizada para o ambiente do robô móvel: cada estado é o espaço suficiente para este ser capaz de realizar uma rotação em torno de seu eixo central. As distâncias percorridas pelo robô, em cada transição de estado, dependem do movimento ser realizado nas diagonais ou nos sentidos dos eixos vertical ou horizontal.

Seguindo a descrição de estado e ações dadas, o problema de navegação pode ser visualizado como uma árvore de decisão. A Figura 29.a mostra a árvore de decisão que representa a dinâmica da interação agente-ambiente: cada círculo representa um possível estado que fornece compactamente para o agente toda a informação que faça diferença no comportamento futuro do ambiente. As oito linhas que partem de cada estado correspondem às oito ações que o agente pode realizar e levam a outros estados (Figura 27.b). Os sensores do robô móvel indicam a posição (através dos *encoders*) e o conteúdo do estado que faz com que se considere três tipos de estado que correspondem aos três tipos de sinal de reforço indicados em (16): estados livres; estados com obstáculos e o estado meta. A heurística que guiará a busca do agente na

árvore de decisão será baseada em uma matriz de transição de estados (Figura 29.b) onde cada elemento representará a probabilidade de ocorrer a transição entre dois estados (Figura 29.c).

Figura 29 - (a) árvore de decisão correspondente à navegação de um robô móvel. (b) matriz de transição dos estados da árvore de busca. (c) destaque da transição entre dois estados e a probabilidade da ocorrência da transição.

A representação acima torna o problema um processo de decisão markoviano: como cada estado contém sozinho informação suficiente para indicar qual pode vir a ser o próximo estado, a interação agente-ambiente tem um comportamento markoviano. A informação fornecida pelos sensores sobre o estado ser livre, um obstáculo ou o alvo representa um sinal qualitativo do ambiente (sinal de reforço para o agente) sobre a ação realizada pelo agente e só pode ser obtida após a transição dos estados ter sido, caso possa ser, realizada. A matriz de transição de estados é aprendida pelo agente durante sua interação com o ambiente e está codificada nas superfícies de recompensa e penalização do agente.

5.3.3 Problemas Enfrentados com o Robô Real

Ao tomar o robô como sendo pontual, considera-se o centro de cada círculo, correspondente a sua posição no ambiente (Figura 28), como informação suficiente

para descrever o estado do robô. Na implementação real, esta posição é estimada a partir da leitura dos *encoders* segundo os valores indicados na Tabela 3.

Tabela 3 - Valores dos *encoders* para os deslocamentos de translação e rotação do robô.

Deslocamentos		Leitura do <i>encoder</i> direito	Leitura do <i>encoder</i> esquerdo
Translação	30 cm	-40800	+40800
	$30\sqrt{2}$ cm	-57700	+57700
Rotação	45°	-12485	+12485
	-45°	+12485	-12485

Porém, pequenos erros de leitura dos *encoders* tais como o escorregamento entre as rodas do robô e o piso e pequenos deslocamentos causados pela inércia do veículo são fontes geradoras de erro na estimativa do posicionamento do robô durante seu funcionamento que afetam sua navegação. Estes pequenos erros podem se tornar maiores conforme a posição em que ocorram e o número de estados da trajetória realizada. A Figura 30 descreve um teste realizado com o robô real que exemplifica como um pequeno erro em um único deslocamento implica em um erro maior na posição final do agente.

Figura 30 - (a) Exemplo de caminho realizado pelo agente durante a exploração do ambiente para encontrar o estado meta. (b) Erro entre a trajetória planejada (virtual) pelo agente e a trajetória real executada pelo robô.

Na Figura 30.b são exibidas a trajetória que o algoritmo planejou, e que leva o agente de um estado inicial para o estado meta, e a trajetória real percorrida com erros no posicionamento do robô. Devido ao limitado número de sensores disponível, estes erros não são devidamente corrigidos e o desempenho do agente pode ser encoberto

por estas falhas, principalmente em ambientes com um espaço de estados grande. Para contornar este problema, que foge ao escopo deste trabalho, optou-se pela realização dos testes do agente proposto através de uma ferramenta de simulação que segue a discretização do ambiente descrita na subseção 5.3.2. O Capítulo 6 comenta a ferramenta utilizada e apresenta os testes realizados a partir dela.

5.4 Conclusões

O agente proposto para navegar autonomamente em ambientes dinâmicos, limitados e, de início, totalmente desconhecidos se utiliza de uma representação diferenciada do conhecimento aprendido para selecionar suas ações: duas superfícies, uma de recompensa e outra de penalização, codificam avaliações de longo (planejamento) e curto (comportamento reativo) prazos dos estados do ambiente. Estas duas superfícies são aprendidas, empregando métodos de diferença temporal (TD), pela interação do agente com o ambiente e com base nas avaliações nelas codificadas são selecionadas as ações que guiam o robô. O mecanismo para a seleção das ações (Araújo e Braga, 1998b) utiliza essas duas superfícies empregando conceitos trazidos dos métodos de campo potencial e da *subsumption architecture*: uma política simples é composta por três tendências de comportamento que vão se desenvolvendo com o treinamento do robô.

O agente tem capacidade de melhorar suas soluções graças a adoção de uma taxa de aprendizagem variável no cálculo da atualização das avaliações dos estados. Tais estados, são considerados discretos para que o problema de navegação possa ser tratado como um processo de decisão markoviano e se possa empregar as técnicas de aprendizagem por reforço. Tal discretização necessita na implementação real de um sistema de posicionamento robusto que corrija erros acumulados ao longo do tempo. Como tal sistema não é do interesse deste trabalho e se torna de difícil implementação em ambientes com dimensões muito maiores que o robô, os testes com o agente são realizados em simulação para evitar a influência de erros de posicionamento e são apresentados no Capítulo 6.

CAPÍTULO 6

Testes com o Agente Proposto

Realização de planejamento deliberativo e reativo, geração de comportamentos emergentes, solução do problema de exploração direcionada à meta, capacidade de melhoria da performance na execução de tarefas em função da quantidade de treinamento e adaptação à ambientes dinâmicos são as principais propriedades esperadas do agente proposto. Neste capítulo serão apresentados testes que buscam exibir como estas características, e outras que serão comentadas ao longo do capítulo, são cumpridas pelo agente. O recurso adotado para coletar os resultados que serão mostrados foi o desenvolvimento de um ambiente de simulação que segue as considerações sobre o sistema real descritas na Seção 5.3, e que é comentado na Seção 6.1.

Ao adotar o paradigma AR para o agente, o problema de navegação autônoma foi descrito como um PARDM (Seção 3.2) e, seguindo este formalismo, as Seções 6.2 e 6.3 apresentam as soluções das duas etapas deste problema: o PEDM e a melhoria da solução inicial. Para a solução do PARDM, planejamento reativo e deliberativo são empregados pelo agente e a Seção 6.4 discute a contribuição de cada tipo de planejamento nesta solução. A combinação destes planejamentos permite ao agente se adaptar a mudanças no ambiente mantendo seu objetivo principal de alcançar o estado meta. Esta é uma propriedade importante para um agente autônomo que atue em um ambiente real. Para verificar o comportamento do agente proposto em um ambiente dinâmico, a Seção 6.5 descreve testes do agente com obstáculos não treinados que bloqueiam trajetórias aprendidas.

6.1 A Ferramenta de Simulação

Conforme apresentado na Seção 5.3, para possibilitar o teste do agente em diversos ambientes e eliminar a influência de erros de posicionamento na descrição do

estado, foi desenvolvida pelo autor desta pesquisa uma ferramenta de simulação que segue as considerações da Seção 5.3 de um robô pontual atuando em um ambiente discretizado e que foi empregada para obtenção dos resultados apresentados neste capítulo. A Figura 31 exibe dois ambientes reais montados e suas respectivas representações na simulação desenvolvida²⁵. Devido à pequena área disponível para testes do robô móvel, os ambientes reais possuíam apenas 6x6 estados. Nas simulações, foram considerados ambientes com um espaço de estados maior. Nos ambientes das Figura 31.c e d, por exemplo, o espaço de estados é formado por 20x20 estados.

Figura 31 - (a)-(c) e (b)-(d) Configurações reais do ambiente do robô móvel e suas respectivas representações na ferramenta de simulação desenvolvida.

A flexibilidade do agente pode ser testada através da ferramenta de simulação ao utilizar outras configurações de ambientes, além das mostradas na Figura 31, como as apresentadas na Figura 32.

²⁵ Poucos testes foram realizados com o robô real, porém os resultados obtidos nestes testes foram semelhantes aos conseguidos em simulação.

Figura 32 – (a)-(d) Diferentes configurações dos ambiente utilizados para testar o comportamento do agente proposto.

As diferentes configurações de ambiente seguem um nível crescente de complexidade para a realização da tarefa do agente. Quanto maior a quantidade de obstáculos no ambiente, maior é o número de restrições impostas ao agente para alcançar o estado meta. Desta forma, o ambiente da Figura 31.a, sala aberta, corresponde ao ambiente mais simples onde o agente possui total liberdade para selecionar suas ações. Os ambientes com um único obstáculo (Figura 32.a), com obstáculo em U (Figura 31.d) e com duas salas (Figura 32.b) reduzem, gradativamente, a quantidade de trajetórias possíveis de conectar o estado meta com algum estado do ambiente. Tornando, com isso, mais difícil a obtenção da solução do PARDM. Os casos de maior dificuldade compreendem os labirintos das Figura 32.c e d. Nos ambientes mais complexos, ambiente com duas salas e os labirintos, a discretização adotada foi de 50x50 estados enquanto nos demais ambientes a discretização foi de 20x20 estados.

A ferramenta de simulação foi desenvolvida no aplicativo MATLAB®²⁶ e possui dois procedimentos principais:

- **treina:** Realiza a fase de aprendizagem do agente sobre o ambiente estático;
- **roda:** Realiza a fase de teste do agente

O procedimento **treina** atualiza as superfícies de penalização e recompensa conforme as equações (19) e (20) e seleciona as ações do agente através da política descrita em 5.2.2. O procedimento **roda** não modifica as superfícies de avaliação dos estados, contudo utiliza os valores calculados por **treina** para selecionar as ações do agente. Entretanto, a política adotada neste procedimento não possui componente totalmente aleatória, tendo a seguinte distribuição heurística:

²⁶ MATLAB é um produto da The MathWorks, Inc.

- Em 85% dos casos, o algoritmo seleciona a ação que leva a um maior valor combinado (recompensa + penalização);
- Em 10% dos casos, o algoritmo seleciona a ação que leva o agente ao estado oposto ao de maior penalização;
- Em 5% dos casos, o algoritmo escolhe a ação que leva à maior recompensa;
- Em caso de empate, a ação é escolhida aleatoriamente.

Esta política mais determinística é uma forma heurística de comprovar a eficiência do aprendizado do agente. Como a caminhada aleatória em um ambiente com número limitado de estados acaba por percorrer todos os estados deste ambiente, quando o tempo tende a infinito, poderia ser a componente aleatória a responsável pelo sucesso do agente. Assim, nos testes apresentados, a política determinística acima descrita foi a adotada para selecionar as ações do agente.

Para facilitar a operação e visualização das simulações, a ferramenta de simulação emprega os recursos do MATLAB® para criação de interfaces GUI e possui a aparência exibida na Figura 33.

Figura 33 - Interface gráfica do ambiente de simulação.

A janela maior da Figura 33 exibe o movimento do robô móvel, em uma das configurações do ambiente, as janelas menores exibem o conhecimento adquirido pelo agente. Os cinco botões presentes na parte inferior da janela maior permitem ao usuário selecionar uma das seguintes funções:

- **Treinar:** Chama o procedimento *treina*. Durante a execução do procedimento, as duas janelas menores à direita da Figura 33 mostram a evolução das superfícies de recompensa (janela superior) e penalização (janela inferior). Os planos XY dos gráficos das superfícies de avaliação correspondem ao ambiente (Figura 34) e o eixo Z indica o valor estimado da avaliação de cada estado do ambiente.

Figura 34 - Plano XY das superfícies de avaliação da recompensa e penalização.

- **Rodar:** Chama o procedimento *roda*. Durante a execução deste procedimento, as ações selecionadas pelo agente podem ser acompanhadas pela trajetória que é plotada na janela maior da interface.
- **Meta:** Permite ao usuário definir com o *mouse* a posição no ambiente que será tomada como o estado meta.
- **Início:** Permite ao usuário definir com o *mouse* as posições no ambiente em que o agente iniciará o treinamento, o estado inicial.
- **Obstáculos:** Permite ao usuário definir posições no ambiente que representem obstáculos não treinados pelo agente.

As próximas seções seguem exibindo e comentando os resultados dos testes do agente obtidos através da ferramenta de simulação descrita. Estes resultados estão distribuídos em quatro tópicos: a solução do PEDM, a solução do PARDM, o agente autônomo reativo-deliberativo e o agente em ambientes dinâmicos.

6.2 Solução do Problema de Exploração Direcionada à Meta

Na solução desta etapa do PARDM, o objetivo desejado é o de que o agente, desconhecendo inicialmente a estrutura do ambiente, explore o espaço de estados, localize o estado meta e codifique nas superfícies de recompensa e penalização informações aprendidas durante a interação com o ambiente. Tal codificação deve

permitir que o agente estabeleça trajetórias, não necessariamente ótimas, que conectem pontos iniciais treinados e não treinados ao estado meta.

Como experimento inicial do agente proposto na resolução do PEDM, será utilizado o ambiente da Figura 35, sendo considerados oito estados iniciais na fase de aprendizagem do agente e um único estado meta. Tal escolha visa permitir uma maior distribuição das avaliações neste ambiente.

Figura 35 - Os pontos marcados com 'O' são pontos de partida e o indicado 'X' é a meta.

Foram tomados os seguintes valores para os parâmetros das equações de atualização das avaliações (19) e (20):

$$\alpha_p = 0.001, \gamma_p = 0.67, \alpha_r = 0.9998, \gamma_r = 0.99 \quad (23)$$

Estes parâmetros foram escolhidos de forma a realizar as distribuições da recompensa e da penalização comentadas em 5.2.1.3: a recompensa deve ser distribuída o máximo possível pelo espaço de estados (por isso, são adotados valores mais altos de α_r e γ_r) e a penalização deve ocorrer apenas próxima aos obstáculos (valores menores de α_p e γ_p). O resultado do treinamento é apresentado na Figura 36.

Figura 36 - (a) Superfície de recompensa. (b) Superfície de penalização.

No resultado do treinamento, a presença de ilhas de recompensa, como as apresentadas na Figura 36.a, levam o agente a entrar em um caminho fechado (Figura 37). Isto se deve à barreira que a descontinuidade entre as ilhas A e B cria para o agente: na Figura 36.a, um agente que tivesse seu estado inicial na região B não conseguiria alcançar o estado meta, que se encontra na região A, pois os baixos valores de avaliação entre as regiões tornam a região B um atrator indesejável, por seus valores de avaliação que prendem o agente nesta região de seu ambiente.

Figura 37 – (a)-(c) Exemplos de algumas situações em que o agente fica preso em caminhos fechados.

O gráfico da trajetória percorrida pelo agente durante seu treinamento, mostrado na Figura 38.b, ilustra como se processa o surgimento das ilhas na superfície de recompensa (Figura 38.a). Ao permitir que um mesmo estado possa ser visitado diversas vezes numa mesma trajetória, a cada visita a avaliação do estado é modificada, e conforme aumenta a quantidade de visitas, menor será a avaliação do estado (ver item 5.2.2), causando as descontinuidades.

Figura 38 - (a) Outro exemplo de superfície de recompensa com descontinuidade na avaliação que causam ilhas de recompensa. (b) Trajetória percorrida pelo agente durante a fase de treinamento que gerou a superfície de recompensa com descontinuidades. O grande número de visitas aos estados da região à direita do ambiente tornam muito baixas as avaliações destes estados como pode ser visualizado na ilha B.

Assim, pelos testes realizados com este treinamento, um dos principais problemas encontrados está em como modificar diversas vezes a avaliação de um mesmo estado sem gerar descontinuidades na superfície de recompensa (Figura 36.a). Uma opção para evitar o surgimento de ilhas na superfície de treinamento é limitar o algoritmo no sentido de serem realizadas modificações nas avaliações apenas na ocorrência da primeira visita. A Figura 39 mostra os resultados obtidos com esta restrição ao aprendizado da superfície de recompensa.

Figura 39 - (a) Superfície de recompensa. (b) Trajetória percorrida durante a fase de treinamento. (c) Trajetória realizada pelo robô a partir da superfície de recompensa obtida.

Atualizar a avaliação de cada estado apenas na primeira visita impede a ocorrência de descontinuidades na superfície de premiação, permitindo ao agente reproduzir o comportamento que o levou aleatoriamente ao alvo (Figura 39.c). De fato, o algoritmo com esta restrição no aprendizado das superfícies consegue solucionar o PEDM. Retomando os oito pontos apresentados na Figura 35 para realizar o treinamento do agente, obtêm-se superfícies de avaliação como as da Figura 40 com esta solução do PEDM. O leitor é convidado a comparar estas superfícies com os resultados anteriores (Figura 36).

Figura 40 - (a)-(b) Superfícies geradas ao treinar o agente com restrição de atualizar as avaliações apenas na primeira visita.

As trajetórias geradas a partir do treinamento acima geram trajetórias como as da Figura 41.

Figura 41 - Alguns exemplos de trajetórias obtidas. Os pontos assinalados por “O” correspondem aos pontos iniciais da trajetória, e o ponto “X” ao ponto meta. (a)-(c) Pontos iniciais treinados durante a fase de aprendizagem. (d)-(i) Pontos iniciais não treinados.

As características atuais do agente, entretanto, não asseguram melhoria significativa de seu desempenho conforme aumenta o número de treinamentos. Em um novo treinamento o agente pode aleatoriamente escolher uma ação melhor²⁷ para um estado, porém como não pode alterar os valores dos estados já visitados fica impedido de melhorar sua política. Um exemplo disso está nas trajetórias da Figura 42, nas quais treinamentos adicionais não alteram significativamente a primeira trajetória aprendida.

²⁷ Por ação melhor entenda-se como sendo uma ação que leve o agente a reduzir o número de estados visitados na trajetória que liga um ponto inicial ao estado meta.

Figura 42 - Trajetórias obtidas após: (a) 1 treinamento, (b) 100 treinamentos, (c) 1000 treinamentos, (d) 10000 treinamentos.

Para solucionar o PARDM, a restrição de atualizar apenas uma vez as avaliações de cada estado é inviável por impedir que novo conhecimento seja adequadamente acrescentado ao agente, impossibilitando a melhoria significativa de seu desempenho. A solução adotada para melhorar o desempenho do agente e evitar o surgimento de ilhas de recompensa é o emprego de uma taxa variável de aprendizagem (item 5.2.2) na atualização das avaliações dos estados. A próxima seção mostra como a utilização de tal taxa variável de aprendizagem torna o agente capaz de solucionar o PARDM e comenta algumas particularidades observadas nesta solução.

6.3 Solução do Problema de Aprendizagem por Reforço Direcionada à Meta

Pela forma como é atualizada, através de uma trajetória percorrida durante a fase de treinamento, a superfície de recompensa é significativamente afetada pela utilização de uma taxa fixa de aprendizagem. Na atualização das avaliações de penalização, o valor atualizado não está diretamente relacionado ao caminho seguido pelo agente até atingir um determinado estado e, com isso, fenômenos geradores de descontinuidades como os discutidos no item 5.2.2 e na Seção 6.2 não afetam a superfície de penalização. Desta forma, adotou-se a taxa de aprendizagem variável apenas para as avaliações de recompensa. Utilizando novamente o ambiente da Figura 35, agora com o objetivo de mostrar a influência da taxa de aprendizagem variável sobre o desempenho do agente, as superfícies da Figura 43 exibem como o aprendizado do agente evolui ao se aumentar o número de treinamentos realizados, utilizando $\alpha_1 = 0.1$ e $\alpha_2 = 0.9998$ para o cálculo da taxa de aprendizagem variável (ver equação 22).

Figura 43 - (a)-(d) Superfícies de recompensa aprendidas pelo agente após, respectivamente, 1, 100, 1000 e 5574 treinamentos. (e)-(h) Superfícies de penalização para os mesmos estágios de treinamento.

As superfícies de penalização da Figura 43 demonstram como o aumento do treinamento do agente pouco acrescenta à informação codificada nestas superfícies²⁸. Em contrapartida, observa-se nas superfícies de recompensa mostradas acima um aumento da suavidade da variação da função de avaliação da recompensa com a realização de mais treinamentos e o espaço de estados também passa a ser mais conhecido. Assim, a taxa de aprendizagem variável não causa descontinuidades no conhecimento mesmo com a continuação do aprendizado do agente sobre seu ambiente. A suavidade da superfície, sem a presença de ótimos locais, contribui para que o agente selecione suas ações na direção do crescimento da estimativa de recompensa, reduzindo o risco de ficar preso em um vale ou em um platô. As mudanças na recompensa se refletem na mudança na política adotada pelo agente para selecionar suas ações e, neste sentido, a manutenção de uma superfície suave com o aumento da região explorada corresponde a uma política que, com o aumento do número de treinamentos, consegue trajetórias cada vez mais curtas. As trajetórias da Figura 44 exibem a evolução do desempenho do agente, no decorrer de seus treinamentos, para os oito pontos iniciais treinados.

²⁸ Conforme discussão realizada no item 5.2.1.3, e mostrada na Figura 20, as mudanças na superfície de penalização são mais visíveis apenas até que o estado meta seja encontrado pela primeira vez.

Figura 44 - (a)-(h) Exemplos de como as trajetórias reduzem o número de estados visitados com a evolução do treinamento em 1, 100, 1000 e 5574 ciclos de treinamento.

Um melhor desempenho do agente ocorre graças às três tendências básicas²⁹ de comportamento do agente (Figura 22): seguir no sentido da maior recompensa, ir no sentido contrário da maior penalização e explorar o ambiente. As duas primeiras componentes guiam o robô para os estados com melhor avaliação dentre os estados já visitados. Porém a terceira busca expandir esse número de estados visitados. Como a avaliação de cada estado depende da avaliação de seus vizinhos, conforme (19) e (20), ao serem considerados novos estados visitados na exploração do ambiente, as avaliações dos estados visitados são modificadas fazendo com que estados anteriormente considerados na escolha de uma trajetória possam ser substituídos por vizinhos recém-visitados que se mostrem mais promissores.

Figura 45 - (a) Exemplo de uma trajetória inicialmente estabelecida entre os pontos inicial (I) e meta (M). (b) Possível nova trajetória, em negrito, conseguida após atualização das superfícies de avaliação que durante a fase de aprendizagem melhoraram a avaliação do ponto A, resultando em um caminho mais curto.

A Figura 45 mostra um exemplo simples de como se processa a melhoria da trajetória: ao explorar o ambiente, o agente estabelece inicialmente um caminho como o apresentado na Figura 45.a, as componentes determinísticas tendem a manter o robô na mesma trajetória já descoberta, porém a componente de exploração atua gerando um novo ponto na trajetória, como o ponto A da Figura 45.b. Logo, um novo caminho gerado pode possuir menos estados que o anterior, passando a ser repetido nas passagens seguintes do robô por influência das componentes determinísticas. Ao longo do funcionamento do agente, a componente de exploração na política é

²⁹ Seguir na direção da maior soma premiação + penalização é uma composição de duas tendências básicas: seguir na direção da meta e desviar dos obstáculos.

acionada diversas vezes levando a uma melhoria cada vez mais acentuada das trajetórias.

O mecanismo utilizado pelo agente para solucionar o PARDM possui características de operação que merecem ser discutidas: o critério de otimalidade do agente, a capacidade do agente em reaproveitar trajetórias aprendidas para gerar novas trajetórias, a influência da estrutura do ambiente no desempenho do agente e a relação entre os parâmetros escolhidos para a atualização das avaliações e o tamanho do espaço de estados do ambiente. Tais características permitem compreender algumas das vantagens e das limitações do agente que serão discutidas no Capítulo 7.

6.3.1 A Otimalidade dos Caminhos

Um ponto a se esclarecer sobre a otimização das soluções produzidas pelo agente é o de que múltiplos caminhos ótimos podem ser obtidos. O compromisso do agente AR está em seguir as ações que maximizem a avaliação de seu comportamento, o que se traduz, no caso do robô móvel, em obter trajetórias com o mínimo de estados. A Figura 46 exibe três possíveis soluções ótimas (todas com 7 estados) para um ambiente fechado sem nenhum obstáculo.

Figura 46 - Exemplo de três trajetórias que conectam os estados inicial e final e possuem o mesmo número de estados que o menor caminho entre esses dois pontos.

Nas simulações realizadas, o agente após ter encontrado uma solução ótima pode, com a continuação do treinamento, gerar outras soluções ótimas. Para fins de exemplo, verifique as três trajetórias apresentadas na Figura 47. No ambiente tratado nesta figura, o menor caminho passa por 17 estados e as três trajetórias apresentam

este número de estados. Porém, cada uma foi gerada durante etapas diferentes do treinamento.

Figura 47 - Evolução da trajetória realizada pelo agente com: (a) 500 treinamento, (b) 1000 treinamentos e (c) 1500 treinamentos

O tamanho das trajetórias não decresce monotonicamente. Como a atuação da componente de exploração da política é aleatória, os novos estados visitados durante as explorações não contribuem necessariamente, ou pelo menos de imediato, para a diminuição das trajetórias. Assim, para alcançar a solução ótima, o agente pode aprender em seus treinamentos caminhos maiores que os de treinamentos anteriores. O gráfico da Figura 48 exibe as oscilações no número de estados até que o agente encontre pela primeira vez um caminho ótimo (caminho com 17 estados).

Figura 48 - Gráfico do número de estados por trajetória para cada passo de treinamento.

Desta forma, as trajetórias geradas pelo agente, mesmo após obtido um caminho com número mínimo de estados, podem variar em tamanho.

6.3.2 Reaproveitamento de Trajetórias Aprendidas

O mecanismo do agente em discussão permite que soluções aprendidas possam ser parcialmente aproveitadas para a obtenção de novas soluções. Os gráficos da Figura 49 exemplificam isto: trajetórias aprendidas durante a fase de treinamento servem para a obtenção de trajetórias que partem de estados iniciais não treinados. Nos resultados da Figura 49, percebe-se que certos trechos de trajetórias se repetem para diferentes pontos de partida. Este reaproveitamento de trechos de caminhos é um comportamento emergente do agente, não estando preso à pré-definição de objetivos especificados *a priori*, e mostra que o algoritmo aproveita esses trechos como subplanos para alcançar o objetivo global de situar o robô na posição meta.

Figura 49 - Exemplos de trechos reaproveitados de trajetórias apontados nos detalhes. Os caminhos da parte superior da figura partem de pontos treinados, enquanto os da parte inferior partem de pontos não treinados. Em todas as trajetórias o ponto alvo está situado na parte inferior da sala a esquerda no ambiente.

A ocorrência desses comportamentos emergentes se deve ao fato das avaliações dos estados do ambiente não estarem uniformemente distribuídas no espaço de estados. As regiões já visitadas pelo agente durante a etapa de exploração, por possuírem avaliações não nulas, atraem o agente ao partir de um ponto não treinado. O agente caminha pelo espaço de estados até atingir a região já explorada durante a fase de treinamento e passa a seguir a trajetória ótima, dentro daquela região, que leve ao estado meta. Este reaproveitamento de trajetórias permite ao

agente, mesmo sem que todo o ambiente tenha sido mapeado, partir de pontos não treinados e alcançar o estado meta sem entrar em uma caminhada totalmente aleatória.

6.3.3 A Estrutura do Ambiente Influenciando o Desempenho do Agente

Por basear-se em informação local, o desempenho do agente sofre grande influência da estrutura do ambiente em que está atuando. Ambientes com poucos obstáculos permitem uma maior exploração do espaço de estados devido à menor restrição às ações que o agente pode selecionar. Uma consequência desta maior liberdade de atuação do agente é uma maior possibilidade de serem geradas trajetórias que sejam soluções ótimas globais³⁰. Os gráficos da Figura 50 ajudam a visualizar esta afirmação. Nas superfícies de recompensa das Figura 50.a-d, verifica-se que, com o aumento dos treinamentos, uma maior região do ambiente possui avaliação não nula, tendendo a cobrir todo o espaço de estados.

Figura 50 – (a)-(d) Superfícies de recompensa geradas com 1, 100, 1000 e 10000 treinamentos. (e)-(h) Trajetórias geradas pelo agente, partindo de um mesmo ponto para alcançar a meta, a partir das superfícies obtidas com 1, 100, 1000 e 10000 treinamentos, respectivamente.

Em ambientes complexos, entretanto, a maior presença de obstáculos reduz a movimentação do agente. Devido a esta menor movimentação, o espaço de estados é menos explorado e o agente se torna menos propenso a obter uma solução ótima global. Nas superfícies de recompensa da Figura 51 visualiza-se um exemplo de como

³⁰ Entende-se por solução ótima global aquela que teria o menor número de estados ao se analisar a estrutura completa do ambiente. Uma solução ótima local seria uma trajetória que para uma parte conhecida do ambiente tivesse o menor número de estados.

evolui o conhecimento do agente sobre um ambiente complexo (Figura 52) com o aumento do treinamento.

Figura 51 - Superfícies de recompensa referentes às trajetórias apresentadas na Figura 52 após: (a) 1 treinamento, (b) 100 treinamentos, (c) 1000 treinamentos e (d) 10000 treinamentos.

Nas superfícies de recompensa acima, verifica-se que a exploração realizada pelo agente fica confinada, mesmo com o aumento do treinamento, a uma região específica do ambiente. A exploração reduz-se à região próxima ao primeiro caminho viável obtido e apresentado na Figura 52.a.

Figura 52 - (a) Trajetória obtida após o primeiro treinamento. Com a obtenção da primeira solução que interliga os pontos inicial e meta no labirinto, a melhoria da solução fica restrita às proximidades dessa primeira trajetória conforme pode ser visto após: (b) 100 treinamentos, (c) 1000 treinamentos e (d) 10000 treinamentos.

Esta influência da estrutura do ambiente no aprendizado deve-se ao caráter local das informações com que o agente trabalha. No caso do robô real, por exemplo, as informações provêm das leituras dos sensores que possuem um alcance limitado a apenas alguns centímetros. É por causa desta concepção local que o agente, mesmo após vários treinamentos, pode insistir em repetir trajetórias que não sejam globalmente ótimas.

Na Figura 53 outro exemplo de como o agente tem dificuldade em conseguir melhorar seu desempenho em ambientes complexos: três instantes do treinamento do robô são apresentados e observa-se que o agente persiste em se restringir à região explorada no primeiro treinamento.

Figura 53 - Superfícies de recompensa relativas às trajetórias da Figura 54: (a) 1 treinamento, (b) 100 treinamentos, (c) 1000 treinamentos.

Nas superfícies de recompensa geradas na Figura 53, verifica-se que não há recompensa dos estados à direita do ponto inicial da trajetória (Figura 54), direção que poderia levar a um caminho com um número menor de estados para a meta, e mesmo o espalhamento da recompensa causado pela exploração não é suficiente para vencer a região atratora formada pelas premiações dos estados da primeira trajetória obtida.

Figura 54 - Trajetórias obtidas após: (a) 1 treinamento, (b) 100 treinamentos, (c) 1000 treinamentos.

Assim, soluções ótimas globais não são garantidas pelo agente proposto. A tendência observada no comportamento do agente é a busca por soluções ótimas apenas na região explorada do espaço de estados.

6.3.4 Influência do Tamanho do Espaço de Estados no Desempenho do Agente

Conforme apresentado na Seção 6.2, o aparecimento de descontinuidades na superfície de recompensa prejudica a tomada de decisões do agente. A solução empregada para evitar o surgimento destas mudanças abruptas na superfície de recompensa foi a adoção de uma taxa variável de aprendizagem (Equação 22) no cálculo das atualizações das estimativas de recompensa (Equação 20). Entretanto,

treinamentos que empreguem taxa variável de aprendizagem também podem gerar superfícies com ótimos locais, como podem ser observados nas Figura 55.a-c, que levem o agente a gerar trajetórias que não alcançam o estado meta como as das Figura 55.d-f.

Figura 55 - (a)-(c): Superfícies geradas com limite inferior da taxa de aprendizagem igual a 0.1 após 1 treinamento. (d)-(f) Trajetórias obtidas a partir das superfícies. Nas três trajetórias o agente ficou preso em torno de um região correspondente a um ótimo local na superfície de recompensa.

O surgimento dos ótimos locais, as rugosidades presentes nas superfícies de recompensa, está relacionado com a forma com que a taxa variável de aprendizagem decai. Nos resultados apresentados na Figura 55, o limite inferior da taxa de aprendizagem, o valor de α_1 na equação 22, é tomado como sendo 0.1. Porém, ao reproduzir os mesmos experimentos considerando $\alpha_1 = 0.01$ obteve-se melhorias nas superfícies de recompensa, e nas respectivas trajetórias geradas, que são exibidas na Figura 56: as rugosidades nas superfícies foram reduzidas significativamente, e o agente conseguiu trajetórias que ligam o estado inicial ao estado meta.

Figura 56 - (a)-(c): Superfícies geradas com limite inferior da taxa de aprendizagem igual a 0.01 após 1 treinamento. (d)-(f) Trajetórias obtidas a partir das superfícies.

Nas trajetórias da Figura 44, observam-se trajetórias satisfatórias³¹ que foram conseguidas utilizando $\alpha_1 = 0.1$ no cálculo da taxa de aprendizagem. Surge assim a questão de qual o motivo do mesmo valor de α_1 ter gerado trajetórias como as da Figura 55 sendo preciso modificar o valor de α_1 para que o agente voltasse a gerar trajetórias satisfatórias. A resposta está no tamanho do espaço de estados de cada ambiente: o ambiente da Figura 44 é constituído por 50x50 estados, enquanto os ambientes tratados nesta subseção são todos formados por 20x20 estados. Desta forma, verifica-se que os parâmetros para o cálculo da taxa de aprendizagem são dependentes do tamanho do espaço de estados do ambiente. Quanto menor o espaço de estados do ambiente, maior a expectativa do agente visitar um mesmo estado muitas vezes e um decaimento mais brusco da taxa de aprendizagem se faz necessário para que as mudanças nas avaliações da recompensa sejam menores, reduzindo as rugosidades na superfície de recompensa.

³¹ Entende-se por trajetória satisfatória aquela que conecta o estado inicial do agente com o estado meta.

6.4 Os Comportamentos Reativos e Deliberativos do Agente

Na tomada de decisões do agente são consideradas as consequências das ações a curto (comportamento reativo) e a longo prazo (planejamento). Os comportamentos reativos estão mais relacionados com a superfície de penalização por seu caráter local de avaliação do estado. A superfície de recompensa com sua característica de ser construída a partir da determinação de estados que levam à meta está predominantemente associada à capacidade de planejamento. As duas próximas subseções discutem essas relações.

6.4.1 Gerando Comportamentos Reativos

Um dos problemas enfrentados por um agente que parte do total desconhecimento de seu ambiente de trabalho é realizar uma exploração desse ambiente que o permita acumular conhecimento para a realização de sua tarefa. O agente proposto busca se adaptar à estrutura de seu mundo conforme aumente sua interação com o mesmo. Na ausência de qualquer avaliação que indique qual ação escolher, o agente começa a gerar ações aleatoriamente buscando encontrar sinais de reforço do ambiente. Pela representação escolhida (ver item 5.2.1.3) há duas situações que fornecem sinais de reforço: alcançar o estado meta ou atingir um obstáculo. A ocorrência da segunda situação é inicialmente mais freqüente, e a partir dela o agente gerará seus primeiros comportamentos reativos que servirão para restringir a exploração no espaço de estados.

Figura 57 - Ambiente simples com obstáculo em forma de U. O estado inicial do agente está marcado com “O” e o estado meta com “X”.

Tomando por exemplo o ambiente da Figura 57, a exploração do agente vai ocorrendo como nos gráficos da Figura 58. Nestes gráficos são apresentados campos

vetoriais, obtidos a partir do gradiente da superfície de penalização, em que os vetores têm módulo proporcional à avaliação da penalização e indicam o sentido da ação a ser selecionada para desviar de obstáculos próximos. A cada obstáculo localizado pelo agente, uma ação é tomada para afastar o agente deste obstáculo e a posição do obstáculo é guardada na superfície de penalização. Essas informações criam regiões de repulsão próximas aos obstáculos, direcionando a exploração do agente.

Figura 58 - Campos vetoriais obtidos a partir do cálculo do gradiente da superfície de penalização em instantes diferentes da exploração do agente.

Nas legenda de cada gráfico da Figura 58 estão indicados o número de obstáculos encontrados pelo agente até a geração do respectivo campo vetorial, sendo o último campo correspondente à última atualização da superfície de penalização antes do agente localizar o estado meta. Pela evolução dos campos vetoriais, percebe-se que a informação codificada na superfície de penalização contribui para que o agente reduza a região de exploração, passando a buscar o estado meta em uma região cada vez menor.

6.4.2 Planejamento Deliberativo do Agente

Uma medida quantitativa do desempenho do aprendizado do agente pode ser obtida pelo número de colisões com obstáculos que o agente realiza. A Figura 59 exibe a evolução do número de colisões do agente para os 25 primeiros treinamentos do experimento que gerou as superfícies de avaliação da Figura 43.

Figura 59 - Gráfico do número de colisões ocorridas a cada treinamento.

Por desconhecer inicialmente o ambiente, o agente apresenta no primeiro treinamento a ocorrência de muitas colisões com as paredes deste ambiente. Porém, já no segundo treinamento verifica-se (Figura 59) uma queda abrupta na quantidade de colisões por já ser o agente capaz de realizar planejamento graças à superfície de recompensa. Através do planejamento deliberativo, o agente pode prever as consequências de suas ações (codificadas nas avaliações dos estados), evitando aquelas que levem a estados com obstáculos e repetindo as que o aproxima do estado meta. Devido à componente aleatória da política durante a fase de treinamento (ver subseção 5.2.2), a partir do segundo treinamento o número de colisões fica oscilando permanecendo, entretanto, baixo.

6.5 O Agente em Ambientes Dinâmicos

Todos os testes realizados até o momento consideraram a atuação do agente em ambientes estáticos cuja estrutura não se modifica com o decorrer do tempo. Ambientes reais, entretanto, são mutáveis e a aplicação do agente proposto a situações reais é viável apenas caso o mesmo consiga se adaptar à eventuais mudanças no ambiente. Para testar o comportamento do agente diante de um obstáculo dinâmico

realizou-se o seguinte procedimento: visualiza-se a trajetória seguida normalmente pelo robô e adiciona-se obstáculos que interrompam essa trajetória. A Figura 60 exibe alguns resultados obtidos.

Figura 60 - (a) Trajetórias inicialmente aprendidas. (b) Trajetórias desenvolvidas pelo agente para contornar os novos obstáculos.

As trajetórias da Figura 60.a foram obtidas após o treinamento que gerou as superfícies de avaliação das Figura 43.d e h. A inclusão de obstáculos que impeçam o agente de repetir as trajetórias aprendidas são uma forma de simular imprevistos como a mudança de móveis em uma sala em que o robô móvel esteja navegando. Supõem-se que a permanência destes obstáculos no ambiente tenha duração que justifique a modificação das avaliações dos estados que passaram a conter estes obstáculos. A flexibilidade do agente proposto a tais condições é mostrada nas trajetórias da Figura 60.b.

Obstáculos móveis ou de curta duração no ambiente, como outros robôs móveis, não são tratados sempre adequadamente pelo agente proposto. O agente necessita de aumento do seu treinamento para incorporar as mudanças no ambiente. Assim, o pequeno período de permanência de tais obstáculos poderia fazer com que o

agente não encontrasse trajetórias que os contornassem como ocorre nas trajetórias da Figura 61.

Figura 61 - (a) Trajetórias inicialmente aprendidas. (b) Trajetórias desenvolvidas pelo agente para contornar os obstáculos temporários.

Os novos obstáculos da Figura 61 ao serem detectados pelos sensores do agente modificam a superfície de penalização. Entretanto, ao serem enfrentados pelo agente sem uma fase de treinamento podem levar o robô a ficar preso em uma trajetória fechada como na primeira coluna da Figura 61.b. O tratamento dado pelo agente aos obstáculos dinâmicos temporários é dependente da aleatoriedade da tomada de decisão, não sendo, portanto, robusto. Conforme a seleção aleatória das componentes da política, pode ser possível ou não desviar o robô da trajetória aprendida de forma a livrá-lo do obstáculo.

6.6 Conclusões

A exploração de um ambiente desconhecido é solucionada pelo agente proposto através de uma sistemática redução da região em que o estado meta é procurado. Esta redução é consequência do comportamento reativo gerado quando o agente se depara com um obstáculo cuja posição é incorporada à superfície de penalização. Ao localizar pela primeira vez o alvo, a superfície de recompensa passa a contribuir para que o agente seja capaz de prever as consequências de suas ações e

planejar de forma a selecionar aquelas que tenham maiores estimativas de retorno. Para que o desempenho do agente melhore durante sua interação com o ambiente, faz-se necessário realizar melhorias nas estimativas das avaliações. Entretanto, a forma como a atualização destas avaliações é realizada pode comprometer o aprendizado já armazenado nas superfícies devido a ocorrência de descontinuidades.

A solução adotada para permitir a constante atualização das estimativas de avaliação dos estados, sem o prejuízo das estimativas anteriormente aprendidas, foi utilizar uma taxa variável de aprendizagem na superfície de avaliação da recompensa que decai com o número de visitas a cada estado. Através desta taxa de aprendizagem, o agente consegue solucionar o PARDM apresentando características interessantes como o reaproveitamento de caminhos aprendidos para estabelecer trajetórias que liguem pontos iniciais não treinados ao estado meta e a capacidade de operar em ambientes que tenham sua estrutura modificada ao longo do tempo. Estas propriedades do agente, entretanto, estão limitadas por aspectos do ambiente como o tamanho de seu espaço de estados, a quantidade de obstáculos presente no ambiente e a velocidade com que se processa as mudanças dos obstáculos em um ambiente dinâmico.

O próximo capítulo comenta com mais detalhes os resultados obtidos neste capítulo, verificando a adequação deles aos critérios estabelecidos na Seção 2.3 para comprovar o quanto o agente proposto cumpre com as características dele esperadas e propondo caminhos que permitam melhorar o desempenho atual do agente.

CAPÍTULO 7

Conclusões e Trabalhos Futuros

Esta dissertação discutiu o desenvolvimento de um agente autônomo que utiliza métodos de aprendizagem por reforço para garantir o aprendizado da tarefa de navegação de um robô móvel em um ambiente dinâmico e inicialmente desconhecido. Diferente das representações comumente encontradas na literatura técnica para armazenar o conhecimento adquirido, o agente proposto emprega estruturas distintas para codificar avaliações positivas e negativas dos estados do ambiente de trabalho. Estas estruturas de conhecimento habilitam o agente a apresentar comportamentos reativos, que possibilitam grande flexibilidade a mudanças no ambiente, e a realizar planejamento deliberativo que direciona as ações do agente para o estado meta a ser alcançado. O mecanismo com que o agente manipula a informação de suas estruturas de conhecimento e os comportamentos que emergem a partir da combinação de seus planejamentos reativo e deliberativo possuem características que cumprem com critérios que foram estabelecidos para guiar esta pesquisa e que são discutidos na Seção 7.1.

Em consonância com o paradigma AR, o problema tratado nesta pesquisa é denominado problema de aprendizagem por reforço direcionado à meta e a solução alcançada com o agente proposto apresenta contribuições a essa classe de problemas e limitações que são comentadas nas Seções 7.2 e 7.3, respectivamente. Com base nos resultados obtidos (Capítulo 6), o agente apresenta possibilidades para futuras pesquisas que são o tema da Seção 7.4 que encerra este capítulo.

7.1 Adequação da Solução Proposta às Características Esperadas do Agente

O desenvolvimento do agente deste trabalho foi guiado por seis critérios listados na Seção 2.3 que foram escolhidos por representarem isoladamente aspectos

desejáveis para um sistema capaz de desempenhar autonomamente tarefas em ambientes reais. Os itens que se seguem analisam como a solução proposta satisfaz a cada um destes critérios.

7.1.1 Habilidade para Operar em Ambientes Inicialmente Desconhecidos

Em ambientes inicialmente desconhecidos, a grande dificuldade encontrada pelo agente é a definição de estratégias de exploração que, com o menor número de interações possível, permitam extrair informações sobre o ambiente que contribuam para que o estado meta, e por consequência o caminho para chegar até ele, seja descoberto no espaço de estados. No agente proposto, a estratégia adotada constitui em reduzir gradativamente a exploração delimitando regiões do espaço de estados que sejam consideradas mais promissoras, e priorizando a busca da trajetória para a meta em tais regiões.

Os campos vetoriais relativos à evolução do aprendizado da superfície de penalização a partir de *tabula rasa* apresentados na Subseção 6.4.1 demonstram como o agente direciona sua exploração do espaço de estados graças a esta superfície de avaliação dos estados. Ao se deparar com um estado obstáculo, a informação armazenada na superfície de penalização indica ações que levam o agente a se afastar do obstáculo e, por consequência, buscar regiões livres do espaço de estados. Assim, o agente proposto ao mesmo tempo que evita comportamentos desastrosos, como a colisão com obstáculos, demarca o perímetro de sua busca pelo estado meta.

7.1.2 Capacidade de Lidar com Ambientes Dinâmicos

A adaptabilidade que o agente proposto possui, graças a sua capacidade de gerar comportamentos reativos, possibilita mudanças em sua política com base apenas na informação local coletada por seus sensores. Desta forma, o agente reage a alterações na estrutura do ambiente conforme estas alterações ocorram dentro do *range* de operação dos sensores. As trajetórias da Figura 60 mostram exemplos de mudanças no ambiente que foram contornadas satisfatoriamente pelo agente para continuar atingindo o estado meta. Entretanto há limites para a capacidade do agente de lidar com as mudanças de um ambiente dinâmico: comportamentos reativos que surjam ao serem detectadas mudanças no ambiente levam o agente a se afastar dos novos obstáculos, porém não necessariamente contribuem para aproximar o agente do estado meta (ver Figura 61). Para descobrir qual ação, ou sequência de ações,

cumprem tanto o objetivo de desviar do novo obstáculo encontrado quanto com o de alcançar o estado meta o agente precisa realizar treinamento explorando suas possibilidades de movimentação. Caso as mudanças do ambiente sejam bruscas e de curta duração, o agente pode não descobrir como escapar dos novos obstáculos ocorrendo situações como as da Figura 61 nas quais o agente fica preso em trajetórias fechadas.

7.1.3 Habilidade para Gerenciar Múltiplos Objetivos

A política adotada para o agente realiza a composição de tendências de comportamento que são selecionadas aleatoriamente (Subseção 5.2.2). Desta forma, a política do agente possui duas maneiras principais para solucionar eventuais conflitos entre objetivos: a exploração aleatória ou a escolha da tendência de comportamento que considera as avaliações de penalização e recompensa em conjunto. A primeira opção trata melhor situações inesperadas em que o conhecimento do agente torna-se bruscamente conflitante com a situação real. Os testes realizados com obstáculos dinâmicos (Seção 6.5), em que trajetórias anteriormente aprendidas são bloqueadas por novos obstáculos adicionados ao ambiente, ilustram este caso. A segunda opção é a preferencial quando o agente possui conhecimento sobre seu ambiente de trabalho, que é confirmada pela informação sensorial, e pode ter na soma das avaliações de seus estados vizinhos uma grandeza que quantifica o quanto ir para cada estado o aproxima de sua meta e o mantém afastado dos obstáculos próximos.

7.1.4 Capacidade para Aprendizagem Incremental

A utilização de métodos de diferença temporal para a atualização das superfícies de avaliação, juntamente com a adoção de uma taxa variável de aprendizagem, garantem a capacidade de aprendizagem incremental ao agente por serem estes métodos inerentemente incrementais. A aprendizagem incremental permite ao agente se readaptar a mudanças no ambiente, seguindo a ocorrência destas, sem destruir o conhecimento anterior sobre o ambiente como ocorre nas trajetórias da Figura 60 que se adaptam às mudanças na estrutura inicial do ambiente.

7.1.5 Capacidade de Melhorar Soluções Iniciais

Uma consequência da maneira como o agente realiza aprendizagem incremental é a possibilidade da anexação de novas ações à política aprendida. Graças

a esta possibilidade, novos estados, descobertos através da componente exploratória da política na fase de treinamento do agente, que reduzam o tamanho da trajetória que conecta um dado estado inicial ao estado meta passam a ser considerados em seqüências futuras de ações tomadas a partir deste estado inicial para atingir a meta. Assim, o aumento do treinamento do agente permite melhorias nas soluções conforme pode ser visto nas trajetórias da Figura 44.

Pela característica de funcionamento do agente de reaproveitar soluções aprendidas (Subseção 6.3.2), as melhorias ocorridas em trajetórias que partem de pontos treinados também contribuem para o melhor desempenho do agente na geração de trajetórias que partem de pontos não treinados. Entretanto, por receber apenas informação local do ambiente, as modificações nas trajetórias inicialmente aprendidas pelo agente sofrem grande influência da complexidade do espaço de estados em se está atuando. Testes realizados (ver Subseção 6.3.3) indicam que em ambientes com poucos obstáculos a melhoria nas trajetórias é mais acentuada, enquanto em ambientes com grande número de obstáculos as variações nas trajetórias iniciais são, comparativamente, mais modestas.

7.1.6 Simplicidade do Mecanismo

O algoritmo adotado para o agente selecionar suas ações evita as soluções que empregam complexos níveis hierárquicos de abstração da tarefa a ser realizada. Em lugar disso, a tarefa é solucionada através de comportamentos emergentes que são aprendidos a partir de camadas que indicam tendências locais de comportamento do agente (ver Figura 22) e que são acionadas segundo uma probabilidade pré-definida. As equações utilizadas para o aprendizado do agente são apenas duas, equações (19) e (20), e demandam baixo esforço computacional.

O cumprimento em conjunto dos critérios acima relacionados e discutidos é, por si só, uma contribuição desta pesquisa para o estudo de agentes autônomos. Entretanto, outras particularidades observadas no agente proposto merecem relevância e são comentadas na seção que se segue.

7.2 Contribuições da Pesquisa

A representação do sinal de reforço empregada para treinar o agente permite uma distribuição mais densa, e informativa, das avaliações dos estados que as duas

representações básicas encontradas na literatura técnica. As estruturas de conhecimento propostas para o agente desenvolvido, que são atualizadas a partir dos sinais de reforço, conseguem através de um único e simples mecanismo capacitar o agente a prever as conseqüências a curto e longo prazos das ações tomadas, administrar conflitos entre objetivos, reaproveitar soluções anteriormente aprendidas em situações não previamente conhecidas e adaptar-se a diversos tipos de ambientes. Tal mecanismo de tomada de decisão do agente se assemelha aos métodos de campo potencial empregados na navegação de robôs móveis, entretanto, diferente da maioria destes métodos que necessitam de conhecimento global do ambiente, as estruturas de conhecimento do agente utilizam informação apenas local para serem geradas, podendo ser construídas mesmo a partir do total desconhecimento do ambiente de trabalho.

A forma com que o agente parte de *tabula rasa* para aprender como executar suas tarefas, e que é comentada no item 7.1.1 deste capítulo, também é uma contribuição deste trabalho para o desenvolvimento de algoritmos AR que venham a ter convergências mais rápidas no aprendizado, que é uma das atuais limitações para a aplicação deste paradigma de aprendizagem na solução de problemas reais.

7.3 Limitações do Agente

O agente, entretanto, também possui limitações de operação em ambientes reais: diante de modificações bruscas no ambiente o agente fica susceptível a caminhar em trajetórias fechadas, apresentando comportamentos não consistentes. A estratégia de exploração do ambiente empregada pelo agente proposto aponta caminhos para tornar mais rápida a convergência do aprendizado de agentes AR, porém os resultados atuais são apenas promissores e não uma prova da solução definitiva. A quantidade de operações realizada pelo agente estudado ainda o torna desapropriado para a aprendizagem de tarefas em tempo real em ambientes com espaço de estados grande.

A forma aleatória com que a política do agente realiza a seleção das ações é um ponto que contribui para a melhoria de seu desempenho no cumprimento da tarefa de navegação quando diminui o número de estados presentes na trajetória ao descobrir novos estados, mais promissores para o cumprimento da tarefa, não visitados anteriormente. Contudo, a aleatoriedade pode gerar prejuízos em outras situações,

como quando a componente da exploração acaba por aumentar a trajetória em uma região do espaço de estados já repetidamente visitada. O gráfico da Figura 48 é bastante elucidativo para a visualização da ocorrência destes fenômenos. Este problema é tratado na literatura como o compromisso entre exploração e repetição (*exploration / exploitation tradeoff*).

Apesar de ser capaz de aprender tarefas em ambientes com diferentes configurações, a adaptabilidade do agente também tem restrições. Os valores dos parâmetros das funções de atualização das avaliações dos estados mostraram ter dependência com o tamanho do espaço de estados do ambiente (Subseção 6.3.4). Tal dependência pode se tornar um entrave para o funcionamento do agente em ambientes reais desconhecidos, pois o tamanho do espaço de estados também não seria conhecido *a priori*.

7.4 Propostas para Futuras Pesquisas

Os resultados obtidos com o agente desta dissertação propiciam desenvolvimentos futuros do algoritmo através de, pelo menos, três linhas de pesquisa:

- ♦ **Aumento da velocidade de convergência do aprendizado do agente:** A exploração do espaço de estados é a etapa do PARDM que demanda mais tempo para ser solucionada. Assim, o estabelecimento de estratégias de exploração mais eficientes, a exemplo da que foi utilizada no agente proposto, contribuirá para aumentar a velocidade de convergência do aprendizado de agentes AR.
- ♦ **Redução gradativa da aleatoriedade da política:** A presença de aleatoriedade na política de agentes AR é uma prática comum que busca permitir ao agente procurar novas soluções no espaço de estados. Entretanto, com o decorrer do funcionamento do agente, esta aleatoriedade passa a prejudicar as soluções encontradas por forçar modificações nas decisões em trechos das seqüências de ações tomadas que já não podem ser melhorados. É intuitivo que devam ser geradas estratégias que removam gradativamente a aleatoriedade da política conforme o aprendizado for se processando. Uma abordagem possível para a redução da aleatoriedade é o estabelecimento de medidas que indiquem o grau de

confiança das avaliações obtidas para relacionar a intensidade da aleatoriedade na tomada das decisões a essa medida.

- ♦ **Determinação de parâmetros adaptáveis a qualquer espaço de estados:** O desempenho do aprendizado do agente proposto mantém dependência com o tamanho do espaço de estados do ambiente em que esteja atuando, e os valores dos parâmetros utilizados no cálculo da atualização das avaliações de recompensa dos estados são um dos fatores causadores desta dependência. Assim, estabelecer parâmetros que se adaptem a qualquer espaço de estados é uma maneira de conseguir agentes autônomos capazes de realizar tarefas em uma gama maior de ambientes mantendo um padrão semelhante de desempenho independente do ambiente.

Os avanços conseguidos em qualquer destas linhas de pesquisa proporcionariam agentes autônomos mais próximos de operar com robustez em ambientes reais.

REFERÊNCIAS

- Agre, P. E. (1988). The dynamic structure of the everyday life. *PhD thesis*, MIT Artificial Intelligence Laboratory, Cambridge, MA.
- Agre, P. E. e Chapman, D. (1987). Pengi: an implementation of a theory of activity. *In Proceedings of Sixth National Conference on Artificial Intelligence*. Morgan Kaufmann. Los Altos, CA. pp: 268-272.
- Anderson, T. L. e Donath, M. (1990). Animal behavior as a paradigm for developing robot autonomy. *Robotics and Autonomous Systems*. V. 6, N. 1-2, pp: 145-168.
- Araújo, A. F. R. e Braga, A. P. S. (1998a). Reinforcement learning based on potential field methods to navigate in initially unknown environments. *In Proceedings of the IASTED/AAAI Conference on Artificial Intelligence and Soft Computing*, Cancún, México.
- Araújo, A. F. R. e Braga, A. P. S. (1998b). Reward-penalty reinforcement learning scheme for planning and reactive behavior. *In Proceedings of 1998 IEEE International Conference on Systems, Man, and Cybernetics*. San Diego, Califórnia, USA.
- Arkin, R. C. (1990). Integrating behavioral, perceptual, and world knowledge in reactive navigation. *Robotics and Autonomous Systems*. V. 6. pp: 105-122.
- Arkin, R. C. (1995). Reactive robotic systems. *Gatech On-line Publications*: <http://www.cc.gatech.edu/aimosaic/robot-lab/mrl-online-publications.html>.
- Balkenius, C. (1995). Natural intelligence in artificial creatures. *Lund University*: <http://lucs.fil.lu.se/Staff/Christian.Balkenius/Thesis/.index.html>.

- Barto, A. G.; Sutton, R. S. e Anderson, C. W. (1983). Neuronlike adaptative elements that can solve difficult learning control problems. *IEEE Transactions on Systems, Man, and Cybernetics*, V. 3, N. 5, pp: 834-846.
- Barto, A. G.; Sutton, R.S e Watkins, C. J. (1989). Learning and sequential decision making. *Technical Report 89-95*, Department of Computer Science, University of Massachusetts at Amherst.
- Barto, A. G.; Bradtke, S. J. e Singh, S. P. (1995). Learning to act using real-time dynamic programming. *Artificial Intelligence*, V. 73, N. 1, pp: 81-138.
- Beer, R. D.; Chiel, H. J. e Sterling, L. S. (1990). A biological perspective on autonomous agent design. *Robotics and Autonomous Systems*. V.6. N. 1-2. pp: 169-186.
- Bellman, R. (1957). *Dynamic programming*. Princeton University Press, Princeton, New Jersey.
- Bertsekas, D. P. (1987). *Dynamic programming: deterministic and stochastic models*. Prentice-Hall Inc., New Jersey.
- Borenstein, J.; Koren, Y. (1991). The vector field histogram – fast obstacle avoidance for mobile robots. *IEEE Transactions on Robotics and Automation*, V. 7, N. 3, pp: 278 –288.
- Bugmann, G.; Koay, K. L.; Barlow, N.; Philips, M. e Rodney, D. (1998). Stable encoding of robot trajectories using normalised radial basis functions: application to na autonomous wheelchair. In *Proceedings of the 29th International Symposium of Robotics*. Birmingham, UK. pp: 27-30.
- Burghes, D. e Graham, A. (1980). *Introduction to control theory including optimal control*. Ellis Horwood.
- Braga, A. P. S e Araújo, A. F. R. (1998a). Navegação em ambiente fechado e inicialmente desconhecido utilizando aprendizagem por reforço. *Anais do XII Congresso Brasileiro Automática*. Uberlândia, MG, Brasil.

- Braga, A. P. S. e Araújo, A. F. R. (1998b). Goal-directed reinforcement learning using variable learning rate. *In Proceedings of SBIA'98 - XIVth Brazilian Symposium on Artificial Intelligence*. Porto Alegre, RS, Brazil.
- Brooks, R. A. (1986). A robust layered control system for a mobile robot, *IEEE Journal of Robotics and Automation*, V. RA-2, N.1, pp: 14-23.
- Brooks, R. A. (1990). Elephants don't play chess. *Robotics and Autonomous Systems*, V. 6, pp: 3-15.
- Brustoloni, J. C. (1991). Autonomous agents: characterization and requirements, *Carnegie Mellon Technical Report CMU-CS-91-204*, Pittsburgh: Carnegie Mellon University.
- Chang, C. e Gaudiano, P. (1997). Neural competitive maps for reactive and adaptive navigation. *In Proceedings of the 2nd International Conference on Computational Intelligence and Neuroscience*, pp: 19-23, Research Triangle Park, North Carolina.
- Chapman, D. (1987). Planning for conjunctive goals. *Artificial Intelligence*. V. 32. pp: 333-377.
- Chen, C. -T. (1984). *Linear system theory and design*. Holt, Rinehart and Winston, Inc. New York.
- Clarke, A. B. e Disney, R. L. (1979). *Probabilidade e processos estocásticos*. Livros Técnicos e Científicos. Rio de Janeiro.
- Connolly, C. I. (1997). Harmonic functions and collision probabilities. *International Journal of Robotics Research*, V. 16, pp. 497-507.
- Connolly, C. I. e Grupen, R. A. (1993). The applications of harmonic functions to robotics. *Journal of Robotic Systems*, V. 10, pp. 931-946.
- Davis, M. H. A. e Vinter, R. B. (1985). *Stochastic modelling and control*. Chapman and Hall Ltd. New York.

- Dean, T.; Kaelbling, L. P.; Kirman, J. e Nicholson, A. (1995). Planning under time constraints in stochastic domains. *Artificial Intelligence*, V. 76, pp: 35-74.
- Dean, T. (1994). Decision-theoric planning and markov decision process. *Tutorial of Summer Institute on Probability and Artificial Intelligence*, Corvalis, Oregon (home-page do autor: <http://www.cs.brown.edu/people/tld/>).
- Donnart, J. -Y.; Meyer, J. -A. (1996). Learning reactive and planning rules in a motivationally autonomous animat. *IEEE Transactions on Systems, Man, and Cybernetics - part B: Cybernetics*, V. 26, N. 3, pp: 381-395.
- Elfes, A. (1987). Sonar-based real-world mapping and navigation. *IEEE Journal of Robotics and Automation*, V. RA-3, N. 3, pp: 249-265.
- Fikes, R. E. e Nilsson, N. J. (1971). STRIPS: a new approach to application of theorem proving to problem solving. *Artificial Intelligence*, V. 2, N. 3-4, pp: 189-208.
- Fikes, R. E.; Hart, P. E. e Nilsson, N. J. (1972). Learning and executing generalized robot plans. *Artificial Intelligence*, V. 3, pp:251-288.
- Fikes, R. E. e Nilsson, N. J. (1993). STRIPS: a retrospective. *Artificial Intelligence*. V. 59. N. 1-2. pp: 227-232.
- Flynn, A. M. (1988). Combining sonar and infrared sensors for mobile robot navigation. *International Journal of Robotics Research*, V. 7, N. 6, pp: 5-14.
- Franklin, S. e Graesser, A. (1997). Is it an agent, or just a program?: a taxonomy for autonomous agents, *In Proceedings of the Agent Theories, Architectures, and Languages Workshop*, Berlin: Springer Verlag, pp: 193-206.
- Georgeff, M. P. e Lansky, A. L. (1987). Reactive reasoning and planning. *In Proceedings of Sixth National Conference on Artificial Intelligence*. Morgan Kaufmann. Los Altos, CA. pp: 677-682.
- Gray, J. O. e Caldwell, D. G. (1996). *Advanced robotics e intelligent machines*. IEE Control Engineering Series 51. London, United Kingdom.

- Green, C. (1969). Application of theorem proving to problem solving. *In Proceedings of the First International Joint Conference on Artificial Intelligence (IJCAI-69)*. Washington, DC. pp: 219-239.
- Grossberg, S. e Levine, D. S. (1987). Neural dynamics of attentionally modulated Pavlovian conditioning: blocking, interstimulus interval, and secondary reinforcement. *Applied Optics*, V. 26, pp: 5015-5030.
- Gullapalli, V.; Franklin, J. A. e Benbrahim, H. (1994). Acquiring robot skills via reinforcement learning. *IEEE Control Systems Magazine*, fevereiro. pp: 13-24.
- Hendler, J.; Tate, A. e Drummond, M. (1990). AI Planning: Systems and Techniques. *AI Magazine*. V. 11, N.2, pp: 61-77.
- Jaakkola, T.; Jordan, M. I. e Singh, S. P. (1994). On the convergence of stochastic iterative dynamic programming algorithms. *Neural Computation*, V. 6, N. 6, pp: 1185-1201.
- Kaelbling, L. P. (1990). *Learning in embedded systems*. MIT Press. Cambridge, Massachusetts.
- Kaelbling, L. P.; Littman, M. L. e Moore, A. W. (1996). Reinforcement learning: a survey, *Journal of Artificial Intelligence Research*, V. 4, pp: 237-285.
- Kalos, M. H. e Whitlock, P. A. (1986). *Monte carlo methods*. Wiley, New York.
- Khatib, O. (1986). Real-time obstacle avoidance for manipulators and mobile robots. *The International Journal of Robotics Research*, V. 5, N. 1, pp: 90-98.
- Koenig, S. (1991). Optimal probabilistic and decision-theoretic planning using Markovian decision theory. *Master thesis*. Computer Science Department, University of California at Berkeley.
- Koenig, S. e Simmons, R. G. (1996). The effect of representation and knowledge on goal-directed exploration with reinforcement learning algorithms. *Machine Learning*, V. 22, pp: 227-250.

- Kortenkamp, D.; Bonasso, R. P. e Murphy, R. (1998). *Artificial intelligence and mobile robots*. AAAI Press / The MIT Press.
- Laird, J. E.; Rosenbloom, P. S. e Newell, A. (1986). Chunking in soar: The anatomy of a general learning mechanism. *Machine Learning*, V. 1, pp: 11-46.
- Langton, C. (1989). *Artificial life*, Redwood City, CA: Addison-Wesley.
- Leonard, J. J. e Durrant-White, H. F. (1991). Mobile robot localization by tracking geometric beacons, *IEEE Transactions on Robotics and Automation*, V.7, N.3, pp: 376-382.
- Lin, L. -J. (1992). Self-improving reactive agents based on reinforcement learning , planning, and teaching. *Machine Learning*, V. 8, pp: 293-321.
- Luger, F. L. e Stubblefield, W. A. (1993). *Artificial intelligence*. The Benjamin / Cummings Publishing Company, Inc. Redwood City, California.
- Luo, R. C. e Kay, M. G. (1995). *Multisensor integration and fusion for intelligent machines and systems*. Ablex Publishing Corporation. Norwood, NJ.
- Manning, A. (1979). *An introduction to animal behavior*, 3rd ed, Addison-Wesley Publ. Co., Reading, Massachusetts.
- Maes, P., (1990). *Designing autonomous agents*, Cambridge, MA: MIT Press.
- Marr, D. (1982). *Vision*. San Francisco: Freeman.
- Mataric, M. J. (1997). Learning social behavior. *Robotics and Autonomous Systems*. V. 20. pp: 191-204.
- Millán, J. del R. e Torras, C. (1992). A reinforcement connectionist approach to robot path finding in non-maze-like environments. *Machine Learning*. N. 8. pp: 363-395.
- Milán, J. del R. (1996). Rapid, safe, and incremental learning of navigation strategies. *IEEE Transactions on Systems, Man, and Cybernetics*, V. 26, pp: 408-420.

- Minsky, M. L. (1961). Steps towards artificial intelligence. *In Proceedings of the Institute of Radio Engineers*. V. 49. pp: 8-30.
- Moravec, H. P. (1988). Sensor fusion in certainty grids for mobile robots. *AI Magazine*, Summer 1988, pp: 61-74.
- Payton, D. W.; Rosenblatt, J. K. e Keirsey, D. M. (1990). Plan guided reaction. *IEEE Transactions on Systems, Man and Cybernetics*. V. 20, N. 6, pp: 1370-1382.
- Rich, E. e Knight, K. (1994). *Inteligência artificial*. Makron Books do Brasil Editora Ltda.
- Rimon, E. e Koditschek, D. E. (1992). Exact robot navigation using artificial potencial functions. *IEEE Transactions on Robotics and Automation*, V.8, N. 5, pp: 501-518.
- Rubinstein, R. Y. (1981). *Simulation and the monte carlo method*. Wiley, New York.
- Russel, S. J. e Norvig, P. (1995). *Artificial intelligence: a modern approach*, Englewood Cliffs, NJ: Prentice Hall.
- Schmitt, P. R. (1996). Reactive path shaping: local path planning for autonomous mobile robots in aisles. *Master thesis*. Mechanical Engineering. Georgia Institute of Tecnology.
- Singh, S. P. (1994). Learning to solve markovian decision processes. *PhD thesis*. Department of Computer Science, University of Massachusetts.
- Stengel, R. F. (1986). *Stochastic optimal control*. John Wiley and Sons.
- Sutton, R. S. (1988). Learning to predict by methods of temporal differences. *Machine Learning*, V. 8, pp: 9-44.
- Sutton, R. S. (1990). Integrated architectures for learning, planning and reacting based on approximating dynamic programming. *In Proceedings of International Conference on Machine Learning*. pp: 216-224.

- Sutton, R. S. (1997). On the significance of markov decision processes. In W. Gerstner, A. Germond, M. Hasler, and J.-D. Nicoud (Eds.) *Artificial Neural Networks - ICANN'97*, pp. 273-282. Springer.
- Sutton, R. S. e Barto, A. G. (1990). *Learning and computational neuroscience: foundations of adaptative networks*. Ed: M. Gabriel and J. Moore. MIT Press.
- Sutton, R. S.; Barto, A. G. e Williams, R. J. (1992). Reinforcement learning is direct adaptive optimal control. *IEEE Control Systems*, V. 12, N. 2, pp: 19-22.
- Sutton, R.S. e Barto, A. (1998). *Introduction to reinforcement learning*. MIT Press / Bradford Books, Cambridge, MA.
- Tesauro, G. (1995). Temporal difference learning and TD-Gammon, *Communications of the ACM*, V. 38, No. 3., pp: 58-68.
- Thorndike, E. L. (1911). *Animal intelligence*. Darien, CT: Hafner.
- Thrun, S.; Moeller, K. e Linden, A. (1991). Planning with an adaptive world model. *In Advances in Neural Information Processing Systems (NIPS) 3*, D. Touretzky, R. Lippmann (eds.), Morgan Kaufmann, San Mateo, CA.
- Xiao, J.; Michalewicz, Z.; Zhang, L. e Trojanowski, K. (1997). Adaptive evolutionary planner/navigator for mobile robots, *IEEE Transactions on Evolutionary Computation*, V.1, N.1, pp: 18-28.
- Warnecke, H. J. e Schuler, J. (1985). Mobile robot applications. *Handbook of Industrial Robotics (Ed.: Shimon Y. Nof)*. John Wiley & Sons, Inc. New York.
- Watkins, C. J. C. H. e Dayan, P. (1992). Q-learning. *Machine Learning*, V. 3, No. 8, pp: 279-292.
- Whitehead, S. D. (1991a). A complexity analysis of cooperative mechanisms in reinforcement learning. *In Proceedings of AAAI*. pp: 607-613.
- Whitehead, S. D. (1991b). A study of cooperative mechanisms for faster reinforcement learning. *Technical Report 365*, Computer Science Department, University of Rochester.

-
- Whitehead, S. D. e Ballard, D. H. (1991). Learning to perceive and act by trial and error. *Machine Learning*, V. 7, pp: 45-83.
- Winston, P. H. (1992). *Artificial intelligence*. Reading, UK: Addison Wesley.
- Wyatt, J. (1997). Exploration and inference in learning from reinforcement, *Ph.D. thesis*, Department of Artificial Intelligence, University of Edinburgh.
- Zalama, E.; Gaudiano, P. e Coronado, J.L. (1995). A real-time, unsupervised neural network for the low-level control of a mobile robot in a nonstationary environment, *Neural Networks*, V. 8, N. 1, pp: 103-123.
- Zelinsky, A. (1992). A mobile robot exploration algorithm, *IEEE Transactions on Robotics and Automation*, V.8, N.6, pp: 707-717.

APÊNDICE I

Processos Markovianos

Seja uma seqüência aleatória ou um processo aleatório (Clarke e Disney, 1979):

$$\{X_n\} = \{X_0, X_1, X_2, \dots\} \quad (I.1)$$

e volte-se a atenção ao caso em que o espaço de estados (o conjunto de valores possíveis das variáveis aleatórias X_n) é discreto. Já que qualquer conjunto discreto pode ser indexado por um conjunto de inteiros, só se precisa considerar o caso em que o espaço dos estados consiste de um conjunto finito ou infinito de inteiros na forma: 0, 1, 2, Este trabalho utiliza a opção finita..

A estrutura de probabilidade de uma seqüência aleatória (ou processo aleatório de parâmetro discreto) é determinada pelas probabilidades conjuntas:

$$p(s_0, s_1, \dots, s_k) = \Pr[S_0 = s_0, S_1 = s_1, \dots, S_T = s_T] \quad (I.2)$$

para todo k finito e para toda seqüência $s_0, s_1, \dots, s_k$ de estados.

Tal processo é denominado um processo de Markov ou cadeia de Markov se, para cada k , a probabilidade condicionada de que o sistema esteja em um dado estado após k passos, conhecendo-se os estados do sistema em todos os passos anteriores é a mesma que a probabilidade condicionada conhecendo-se apenas o estado do passo imediatamente anterior. Intuitivamente falando, o conhecimento de $X_0, X_1, \dots, X_{k-1}$ é equivalente, com fins de prever o valor de X_k , ao conhecimento de X_{k-1} sozinho. Esta definição é re-escrita como abaixo:

$$p(s_k | s_0, s_1, \dots, s_{k-1}) = \Pr[X_k = s_k | X_0 = s_0, X_1 = s_1, \dots, X_{k-1} = s_{k-1}] \quad (I.3)$$

Então, um processo markoviano discreto será uma sequência aleatória que satisfaz a identidade:

$$\begin{aligned} p(s_k | s_0, s_1, \dots, s_{k-1}) &= \Pr[X_k = s_k | X_{k-1} = s_{k-1}] \\ &= p(s_k | s_{k-1}) \end{aligned} \quad (I.4)$$

para cada k e para cada sequência $s_0, s_1, \dots, s_k$ de estados.

Se (I.4) é verdadeiro para todo k , então se pode fazer uso repetido da propriedade $\Pr[A \cap B] = \Pr[A]\Pr[B|A]$ e obter:

$$\begin{aligned} p(s_0, s_1, \dots, s_k) &= p(s_0, s_1, \dots, s_{k-1})p(s_k | s_0, s_1, \dots, s_{k-1}) \\ &= p(s_0, s_1, \dots, s_{k-1})p(s_k | s_{k-1}) \\ &\quad \vdots \\ &= p(s_0)p(s_1 | s_0) \cdots p(s_k | s_{k-1}) \end{aligned} \quad (I.5)$$

Assim, todas as probabilidades conjuntas necessárias à descrição do processo são determinadas em termos das probabilidades condicionadas simples $p(s_k | s_{k-1})$, chamadas de probabilidades de transição de um passo, e da probabilidade inicial $p(s_0)$.

Em todos os exemplos aqui considerados, as probabilidades de transição de um passo não dependem de que passo está sendo considerado:

$$\Pr[X_k = s_j | X_{k-1} = s_i] = \Pr[X_r = s_j | X_{r-1} = s_i] \quad (I.6)$$

para qualquer k e r (desde que X_{k-1} e X_{r-1} têm probabilidades não nulas de serem iguais a s_i). Esse processo é dito como tendo um mecanismo de transição estacionário, e só se considerará processos markovianos discretos com essa propriedade. Neste caso, a notação usada pode ser simplificada, indicando-se as probabilidades de transição de um passo por:

$$P_{s_i s_j} = \Pr[X_k = s_j | X_{k-1} = s_i] \quad (I.7)$$

que não depende de k , e as probabilidades iniciais por:

$$p_{s_i} = p_{s_i}^{(0)} = \Pr[X_0 = s_i] \quad (I.8)$$

para $i, j = 0, 1, 2, \dots$.

Assim, (I.5) pode ser escrita como:

$$p(s_0, s_1, \dots, s_k) = p_{s_0} p_{s_0 s_1} p_{s_1 s_2} \dots p_{s_{k-1} s_k} \quad (\text{I.9})$$

Sendo:

$$\sum_j p_{ij} = 1 \quad (\text{I.10})$$

para cada i , já que começando no estado i após $k-1$ passos, o processo terá que estar em algum estado j após o próximo passo.

Deste modo, define-se a matriz P cujas entradas são as probabilidades de transição de um passo p_{ij} ,

$$P = (p_{ij}) = \begin{pmatrix} p_{s_0 s_0} & p_{s_0 s_1} & p_{s_0 s_2} & \dots \\ p_{s_1 s_0} & p_{s_1 s_1} & p_{s_1 s_2} & \dots \\ \vdots & \vdots & \vdots & \ddots \end{pmatrix} \quad (\text{I.11})$$

é denominada a matriz de transição do processo, e a condição (I.10) assegura, simplesmente, que a soma das entradas em cada linha P seja 1. Qualquer matriz que tenha entradas não negativas com todas as somas de cada linhas 1 é chamada uma matriz estocástica.

Seja ainda o vetor de probabilidade inicial:

$$p^0 = (p_0, p_1, p_2, \dots) \quad (\text{I.12})$$

Define-se ainda a probabilidade de transição de n -passos $p_{ij}^{(n)}$ por

$$p_{ij}^{(n)} = \Pr[X_{k+n} = j | X_k = i] \quad (\text{I.13})$$

para $i, j = 0, 1, 2, \dots$, e para cada $n \geq 0$; $p_{ij}^{(n)}$ é a probabilidade de que o processo vai passar do estado i ao estado j em n passos. É claro, $p_{ij}^{(1)} = p_{ij}$.

De forma semelhante,

$$p_i^{(n)} = \Pr[X_n = i] \quad (\text{I.14})$$

é a probabilidade incondicional de que o processo esteja no estado i após n passos.

I.1 Processos de Decisão Markovianos (PDMs)

A Figura I-1 apresenta um diagrama de blocos que descreve de maneira geral um Processo de Decisão Markoviano (Singh, 1994): um agente interage com um ambiente externo em um ciclo, discreto no tempo, de percepção-e-ação: a cada passo de tempo, o agente percebe seu ambiente, executa uma ação e recebe um reforço como retorno. O ambiente é tido como sendo um processo markoviano, podendo ser descrito por uma matriz de transição P , descrita em (I.11), em que a probabilidade de passar de um estado s_t para um estado s_{t+1} em função de uma ação a é denotado por $P_{s_t s_{t+1}}^a$. O reforço é um critério do desempenho do agente e permite avaliar o custo de mudar de um estado s_t para um estado s_{t+1} através de uma ação a , $R_{s_t s_{t+1}}^a$.

Figura I-1- Esquema geral de um PDM (Singh, 1994).

Concluindo, conhecidos $P_{s_t s_{t+1}}^a$ e $R_{s_t s_{t+1}}^a$ para todos os estados s_t e s_{t+1} pertencentes ao conjunto S de estados e todas as ações a pertencentes ao conjunto A de ações, conhece-se completamente a dinâmica do ambiente. A tarefa do agente é determinar uma política que otimize alguma medida acumulativa dos reforços no decorrer do tempo.

APÊNDICE II

OS PRIMEIROS AGENTES E SEUS PROBLEMAS

Neste apêndice são apresentados, por ordem cronológica de implementação, versões anteriores do algoritmo final descrito no Capítulo 5 que foram desenvolvidas durante esta pesquisa. As principais falhas de cada versão serão comentadas, servindo de justificativa para as alterações que originaram as versões seguintes.

II.1 Primeira versão do agente

A Figura II-1 mostra o primeiro algoritmo implementado. Nele, como não há sinais de reforço em outros estados que não sejam obstáculos ou o alvo, não foi colocado r na atualização das avaliações. Quando o agente encontra um obstáculo, assinala-se diretamente o valor -1 à avaliação daquele estado e, para o alvo, assinala-se o valor 1. A política para a escolha dos estados é ϵ -greedy: uma variável aleatória decide se a escolha do próximo estado será aleatória ou determinística. Com probabilidade ϵ a escolha é aleatória (escolhe-se ao acaso um dentre os estados vizinhos), e com probabilidade $1-\epsilon$ determinística (o estado seguinte escolhido é aquele que tenha o maior valor de avaliação entre os estados vizinhos). A avaliação é feita através da soma algébrica das superfícies de recompensa (V_e) e penalização (V_i) semelhantes as descritas no item 5.2.1.3. A componente aleatória da política tem a função de proporcionar uma exploração do espaço de estados, permitindo que se encontre novas soluções.


```

Inicialize  $V_i(s)$ ,  $V_e(s)$  com zeros;
Repita
  Inicialize  $s$ ;
  Repita (para cada tentativa):
    Escolha  $s'$  usando uma política derivada de  $V_i$  e  $V_e$  (ex.  $\epsilon$ -greedy);
     $V_i(s) = V_i(s) + \alpha_1[\gamma_1 V_i(s') - V_i(s)]$ ;
     $V_e(s) = V_e(s) + \alpha_2[\gamma_2 V_e(s') - V_e(s)]$ ;
     $s = s'$ ;
  até que  $s$  seja o estado meta ou um obstáculo.
até que para todo  $s$  tenha-se que:  $V_i(s+1)+V_e(s+1) \geq V_i(s)+V_e(s)$  e o  $s$  seja a meta.

```

Figura II-1 - Primeiro algoritmo implementado.

Ao se obter uma seqüência de passos que liguem o estado inicial e o estado alvo então uma solução foi encontrada. Porém se esta seqüência não puder ser reproduzida em situações semelhantes pelo agente não terá havido aprendizagem - apenas o que se obteve foi uma seqüência aleatória que eventualmente alcançou os objetivos desejados. O critério de parada deve de alguma forma assegurar que as avaliações aprendidas garantam a repetição da solução do problema. No caso implementado, utiliza-se como critério de parada que a seqüência de estados resultante seja tal que o estado presente tenha uma avaliação maior que a do seu estado anterior e menor que a do estado posterior. Esse critério garante que o agente não retorne para o estado anterior e acabe por ficar preso num loop. A Figura II-2 mostra uma trajetória que foi aprendida sem utilizar o critério de se garantir um crescimento das avaliações no decorrer da seqüência - o agente entrou em um *loop* entre os dois últimos estados da seqüência.

Figura II-2 - Trajetória em *loop*. O ponto marcado com um 'X' é o estado meta. O pequeno círculo denota o estado central cujo valor está marcado em cinza na Tabela II-1.

A Tabela II-1 mostra o motivo: o ponto central da tabela corresponde à avaliação do estado circundado na Figura II-2, ao se escolher entre os oito estados vizinhos aquele que fornece maior avaliação opta-se pelo estado imediatamente

superior à esquerda ($V_i + V_e = -4.6001e-076$). Ao mudar para esse estado e se buscar o estado seguinte com maior avaliação volta-se ao estado anterior ($V_i + V_e = -4.6137e-076$) e assim acaba por se entrar em um loop.

Tabela II-1 - Avaliações $V_e + V_i$ para os estados ao redor do estado assinalado na Figura II-2.

-5.0604e-067	-5.0452e-067	-4.7518e-073	-1.2224e-048	-2.5828e-042
-4.8939e-070	-4.6001e-076	-4.8988e-070	-1.2539e-045	-2.5078e-045
-4.7518e-073	-4.7423e-073	-4.6137e-076	-1.2151e-048	-1.1798e-051
-2.1542e-081	-9.2001e-076	-1.1810e-051	-2.2876e-054	-1.1774e-051
-4.4620e-079	-4.4665e-079	-1.1112e-057	-1.1090e-057	-1.1421e-054

Um problema observado com este algoritmo é a possibilidade do surgimento de “barreiras de penalizações”. No laço mais interno do algoritmo da Figura II-1 verifica-se que as avaliações de recompensa e penalizações são atualizadas ao mesmo tempo para cada estado. Como há uma probabilidade maior de ocorrerem mais tentativas que esbarrem em um obstáculo que tentativas que encontrem o alvo, as penalizações serão mais distribuídas entre os estados que as recompensas. Com isso, trechos do ambiente que teriam que ser percorridos para alcançar a meta podem passar a ser evitados.

Figura II-3 - Configuração de treinamento para a qual o agente ficou preso. (a) Região observada do ambiente. (b) Valores quantizados da avaliação dos estados da região em destaque.

Um exemplo de um trecho importante no ambiente de trabalho aqui tratado para testes é a pequena passagem que liga os dois compartimentos da Figura II-3.a. No caso do alvo estar em um compartimento e o agente em outro, o robô tem necessariamente que atravessar a pequena passagem. Como ela é estreita, há uma grande chance de que antes de encontrar o alvo o agente visite muitas vezes a região entre os compartimentos e ao esbarrar nos obstáculos dessa região passe a avaliá-la como um trecho a ser evitado, impedindo com isso que uma trajetória possa ser aprendida.

Realizado um treinamento com alvo e ponto inicial em compartimentos distintos, o agente não conseguiu transpor a pequena passagem, ficando preso no compartimento que continha o ponto de partida. A Figura II-3 ilustra a configuração do treinamento, onde o destaque da figura apresenta o sinal da composição das avaliações de premiação e penalização para a região entre os compartimentos. Pode-se verificar que uma “barreira de penalizações” (marcadas com -1) é o motivo do agente não conseguir completar a trajetória, a não ser que ele siga um comportamento totalmente aleatório.

Para evitar o surgimento dessas “barreiras”, as penalizações devem se fazer sentir apenas na vizinhança imediata dos obstáculos, não podendo se distribuir muito pelo espaço de estados. Para conseguir esse efeito, os parâmetros α e γ presentes no cálculo da atualização (algoritmo da Figura II-1) devem ter valores diferentes na premiação e na penalização de forma a que os valores caiam suavemente na superfície de premiação e bruscamente na de penalização.

II.2 Segunda versão do agente

Pelo algoritmo anterior, após o agente ter alcançado pela primeira vez o estado alvo apenas este estado terá uma avaliação não nula, na segunda vez haverá dois estados com avaliações não nulas, e desta maneira seguirá até que toda uma trajetória seja aprendida. Desta forma, a distribuição da recompensa fica muito esparsa no espaço de estados e o aprendizado extremamente lento. Para acelerar o aprendizado, o agente armazena toda sua trajetória percorrida e, quando da ocorrência da meta, atualiza as avaliações das recompensas de todos os estados da trajetória. Como a penalização é restrita à vizinhança de um obstáculo, a estratégia de atualização da superfície de penalização permanece a mesma. Seguindo essa estratégia, o

aprendizado do ambiente pelo agente foi realizado a partir de vários pontos de forma a distribuir avaliações no espaço de estados: tomou-se pontos de partida nos extremos do ambiente e um único alvo. Os resultados desta versão do agente são apresentados no início da Seção 6.2.

II.3 Terceira versão do agente

Para solucionar o surgimento de ilhas atratoras na superfície de recompensa observadas após o treinamento da versão anterior do agente, o algoritmo foi alterado no sentido de que sejam realizadas modificações nas avaliações da recompensa apenas nos estados ainda não visitados. As melhorias conseguidas com esta versão do agente são apresentadas ao final da Seção 6.2.

II.4 Quarta versão do agente

A restrição colocada no agente da versão anterior da avaliação de cada estado ser corrigida apenas na primeira visita ao estado implica em impedir que o agente aprimore as estimativas das avaliações dos estados. Para permitir que o agente continue a evitar o surgimento das ilhas atratoras na superfície de recompensa, porém possibilitando que o desempenho do mesmo evoluísse com o aumento dos treinamentos, uma taxa variável de aprendizagem foi empregada no cálculo das atualizações das estimativas de avaliação de recompensa realizadas pelo agente. Esta quarta versão do agente corresponde à versão final descrita nesta pesquisa.